COPPE
Instituto Alberto Luiz Coimbra de
Pós-Graduação e Pesquisa de Engenharia
UFRJ

ON (IN)TRACTABILITY OF CONNECTION AND CUT PROBLEMS

Alexsander Andrade de Melo

Tese de Doutorado apresentada ao Programa de Pós-graduação em Engenharia de Sistemas e Computação, COPPE, da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Doutor em Engenharia de Sistemas e Computação.

Orientadores: Celina Miraglia Herrera de Figueiredo
Uéverton dos Santos Souza
Ana Shirley Ferreira da Silva

Rio de Janeiro
Julho de 2022

ON (IN)TRACTABILITY OF CONNECTION AND CUT PROBLEMS

Alexsander Andrade de Melo

TESE SUBMETIDA AO CORPO DOCENTE DO INSTITUTO ALBERTO LUIZ COIMBRA DE PÓS-GRADUAÇÃO E PESQUISA DE ENGENHARIA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE DOUTOR EM CIÊNCIAS EM ENGENHARIA DE SISTEMAS E COMPUTAÇÃO.

Orientadores: Celina Miraglia Herrera de Figueiredo
Uéverton dos Santos Souza
Ana Shirley Ferreira da Silva

Aprovada por: Profa. Celina Miraglia Herrera de Figueiredo
Prof. Uéverton dos Santos Souza
Profa. Ana Shirley Ferreira da Silva
Prof. Jayme Luiz Szwarcfiter
Prof. Mitre Costa Dourado
Prof. Mateus de Oliveira Oliveira
Profa. Paloma Thomé de Lima

RIO DE JANEIRO, RJ – BRASIL
JULHO DE 2022

Melo, Alexsander Andrade de

On (in)tractability of connection and cut problems/Alexsander Andrade de Melo. – Rio de Janeiro: UFRJ/COPPE, 2022.

XI, 244 p.: il.; $29,7$cm.

Orientadores: Celina Miraglia Herrera de Figueiredo
Uéverton dos Santos Souza
Ana Shirley Ferreira da Silva

Tese (doutorado) – UFRJ/COPPE/Programa de Engenharia de Sistemas e Computação, 2022.

Referências Bibliográficas: p. 90 – 100.

1. Computational complexity. 2. Connection problems. 3. Steiner tree. 4. Cut problems. 5. Maximum cut. 6. Graph classes. 7. Zig-zag number. I. Figueiredo, Celina Miraglia Herrera de *et al.* II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia de Sistemas e Computação. III. Título.

Agradecimentos

Primeiramente, agradeço a Deus por me capacitar, abençoar e sustentar. Agradeço aos meus pais, Samuel e Wânia, e aos meus irmãos, Priscila e Lucas, por todo apoio e ensinamentos concedidos ao longo da minha vida. Agradeço também aos meus demais familiares por sempre desejarem o meu sucesso; em especial, agradeço aos meus avós por todo incentivo e suporte.

Agradeço imensamente aos meu orientadores: profa. Celina Figueiredo, prof. Uéverton Souza e profa. Ana Silva, por terem me aceitado como seu orientando, e por, muito além disso, todos os ensinamentos transmitidos e toda paciência concedida a mim durante a execução deste trabalho. I also would like to thank professor Michael Fellows for accepting to be my supervisor during my internship at University of Bergen, and professor Frances Rosamond for receiving me very well in Bergen.

Agradeço aos demais professores que compuseram a banca examinadora deste trabalho, prof. Jayme Szwarcfiter, prof. Mitre Dourado, prof. Mateus Oliveira e profa. Paloma Lima. Muito obrigado por se disponibilizarem para avaliar este trabalho e pelos comentários valiosos.

Agradeço aos meus co-autores, com quem tive o prazer de aprender imensamente: Ana Silva, Celina Figueiredo, Diana Sasaki, Fabiano Oliveira, Mitre Dourado, Raul Lopes e Uéverton Souza. Em particular, ao Mateus Oliveira por toda frutífera colaboração durante, e após, a minha visita à Universidade de Bergen.

Agradeço aos meus amigos da UFRJ. Em especial, Alesom Zorzi, Alexandre Santiago, Ana Luisa, Caroline Patrão, Edinelço Dalcumune, Dani Castello Branco, Diana Sasaki, Fernanda Couto, Leando Santiago, Moysés Sampaio e Wanderson Lomenha. Aos meus amigos da UFRRJ. Em particular, ao Ygor Canalli pela amizade ao longo dos anos. I also would like to thank Lars Jaffke and Palma Lima for all support provided while I was visiting Bergen. You made my stay much more enjoyable. I am also very grateful to the friends that I made in Bergen: Ícaro Malta, Julian Melo, Jonathan Prieto, Katherine Beltran, and Sandra Macià.

Agradeço aos professores da UFRJ, com os quais tive a honra de ter aula, e aos demais funcionários pela atenção e prontidão. Aos meus amigos do Cultura Inglesa, em especial à Marcelle Duarte. Aos meus colegas e amigos da Mendelics.

Por fim, agradeço ao CNPq e à CAPES pelo suporte financeiro.

Resumo da Tese apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Doutor em Ciências (D.Sc.)

SOBRE A (IN)TRATABILIDADE DE PROBLEMAS DE CONEXÃO E CORTE

Alexsander Andrade de Melo

Julho/2022

Orientadores: Celina Miraglia Herrera de Figueiredo
Uéverton dos Santos Souza
Ana Shirley Ferreira da Silva

Programa: Engenharia de Sistemas e Computação

Problemas de conexão e corte são problemas em grafos que foram amplamente estudados aos longos dos anos. Informalmente, problemas de conexão visam obter o menor/maior número de elementos necessários cuja inclusão resulte em um grafo conexo que satisfaça certas condições, enquanto que problemas de corte visam obter o menor/maior número de elementos necessários cuja remoção origine um grafo (desconexo) com mais componentes conexos. Nesta tese, abordamos problemas de conexão e corte sob a ótica de classes de grafos e complexidade computacional.

Especificamente, analisamos a complexidade do problema Conexão de terminais (TCP), que pode ser visto como uma generalização do problema clássico Árvore de Steiner. Propomos diversos resultados de complexidade para o TCP e para sua variante estrita (S-TCP), quando alguns dos parâmetros de entrada são fixos, e quando restritos a classes de grafos específicas, tais como grafos *split*, grafos de caminhos direcionados enraizados e grafos de *clique-width* limitado. Concentramo-nos especialmente em resultados que diferenciam a complexidade do TCP da complexidade do problema da Árvore de Steiner.

Ademais, analisamos a complexidade do problema clássico Corte máximo. Provamos que o problema é NP-completo em grafos de intervalo de contagem de intervalos igual a 4. Provamos também que Corte máximo é NP-completo em grafos de permutação. Este resultado resolve uma pergunta proposta por David S. Johnson, em *Ongoing Guide to NP-completeness*, que permanecia em aberto por diversos anos. Por fim, investigamos a complexidade do problema de computar o número de zig-zag de grafos direcionados. Provamos que k-zig-zag number está em NP para todo valor fixo de k, e que 2-zig-zag number é NP-completo.

Abstract of Thesis presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Doctor of Science (D.Sc.)

ON (IN)TRACTABILITY OF CONNECTION AND CUT PROBLEMS

Alexsander Andrade de Melo

July/2022

Advisors: Celina Miraglia Herrera de Figueiredo
Uéverton dos Santos Souza
Ana Shirley Ferreira da Silva

Department: Systems Engineering and Computer Science

Connection and cut problems are general graph problems widely studied over the years. Roughly, connection problems aim to obtain a minimum/maximum number of required elements whose inclusion yields a connected graph satisfying certain conditions, while cut problems aims to obtain a minimum/maximum number of required elements whose removal yields a (disconnected) graph with more connected components. This thesis addresses connection and cut problems from the perspective of graph classes and computational complexity.

Specifically, we analyse the computational complexity of TERMINAL CONNECTION (TCP), which can be seen as a generalisation of the classical STEINER TREE problem. We propose several complexity results for TCP and for its strict variant (S-TCP), when some of the input parameters are fixed, and they are restricted to specific graph classes, such as split graphs, rooted directed path graphs, and graphs of bounded clique-width. We mainly concentrate on results that differentiate the complexity of TCP from the complexity of STEINER TREE.

Additionally, we analyse the computational complexity of the classical MAXCUT problem. We propose the first complexity classification for the problem with respect to interval graphs of bounded interval count, by proving that it remains NP-complete on interval graphs of interval count 4. We also prove that MAXCUT is NP-complete on permutation graphs, settling a long-standing open question from *Ongoing Guide to NP-completeness* by David S. Johnson. Finally, we investigate the complexity of computing the zig-zag number of a directed graph, which is a directed width measure defined over cuts of a graph. We prove that k-ZIG-ZAG NUMBER is in NP for every fixed k, and that 2-ZIG-ZAG NUMBER is already an NP-complete problem.

Contents

List of Figures

Chapter 1

Introduction

The theory of computational complexity is the branch of computer science that studies the classification of problems according to the level of hardness of being solved by a generic model of computation *cf.* [104]. A major change in the field occurred in the seventies with the introduction of NP-*completeness theory* by Cook [25], and with the subsequent results due to Karp, classifying 21 notable problems as NP-complete [86].

The 1979 book *Computers and Intractability, A Guide to the Theory of* NP-*completeness* by Michael R. Garey and David S. Johnson [66] is regarded by the computational complexity community as the single most important book. One of its distinguishing features is an appendix with a thorough list of 300 NP-complete problems, organised into 13 categories according to subject matter. In particular, we highlight the category *Network design*, which contains several classical problems on graphs, such as Disjoint paths, Network flow, Steiner tree, and Maximum cut. From 1981 until 2007, David S. Johnson continuously updated the book in 26 columns, entitled *The* NP-*completeness Column: An Ongoing Guide*. The sixteenth column [83], *Graph restrictions and their effect*, presents a summary table containing 30 graph classes disposed as rows and 11 problems disposed as columns. This edition of ongoing guide focus on NP-complete problems on graphs, emphasising how the restrictions on the input graphs affect the complexity of the selected problems.

Another breakthrough in computational complexity was the introduction of the theory of *parameterized complexity* by Downey and Fellows [54] in the late eighties. Informally, in this theory, instead of analysing the complexity of a problem only in terms of the size of the input instances, dependence on one or more parameters is additionally taken into account *cf.* [31]. This is motivated by the fact that, for many natural problems, only a small range of possible values for such parameters is of practical significance *cf.* [53]. Thus, the fundamental idea of parameterized complexity is restricting the (apparently inevitable) combinatorial explosion to solve these problems to certain fixed parameters, which in practice do not depend on the input size and can actually assume only small values.

In [34] (see Appendix A for more details), we proposed an updated version of the summary table presented in [83], with the same graph classes and problems, where several entries that were open in [83] have then been classified into polynomial-time solvable or NP-complete. Two examples are the entries corresponding to MaxCut on interval graphs [1] and to Steiner tree on undirected path graphs [34], both proved to be NP-complete. Moreover, we revised the original table according to the granularity provided by the theory of parameterized complexity.

In this thesis, two groups of problems on graphs are considered from the perspective of classical and parameterized computational complexity, when restrictions on the input graphs are imposed. These groups concern *connection* and *cut* problems. More specifically, we investigate the computational complexity of the so-called Terminal connection problem (TCP), and of its strict variant (S-TCP), when restricted to specific graph classes and when some of the input parameters are fixed. We mainly focus on graph classes that distinguish the complexity of TCP from the complexity of Steiner tree, which is a closely and very famous related problem appearing in [83]. Regarding cut problems, we investigate the computational complexity of the MaxCut problem on interval graphs of bounded interval count, and on permutation graphs, filling an open entry from [34, 83]. We also study the problem of computing the *zig-zag number* of a directed graph, which is a parameter defined over cuts of a graph.

This thesis consists of five chapters, followed by eight appendixes. The remainder of Chapter 1 is organised as follows: in Section 1.1, we provide a short overview of the problems studied; in Section 1.2, we establish the notation and terminology used throughout this text. We refer to the first appendix for more details on David S. Johnson's ongoing guide.

- Appendix A. Manuscript: Celina M. H. de Figueiredo, Alexsander A. de Melo, Diana Sasaki, Ana Silva. Revising Johnson's Table for the 21st Century. Accepted for publication in *Discrete Applied Mathematics* [34].

Chapter 2 comprises the results regarding connection problems. This chapter corresponds to the following appendixes.

- Appendix B. Manuscript: Alexsander A. de Melo, Celina M. H. de Figueiredo, Uéverton S. Souza. A Multivariate Analysis of the Strict Terminal Connection Problem. Published in *Journal of Computer and System Sciences* (2020) [96].

- Appendix C. Manuscript: Alexsander A. de Melo, Celina M. H. de Figueiredo, Uéverton S. Souza. The Strict Terminal Connection Problem on Chordal Bipartite Graphs. Published in *Matemática Contemporânea* (2021) [38].

- Appendix D. Manuscript: Alexsander A. de Melo, Celina M. H. de Figueiredo, Uéverton S. Souza. On the Computational Difficulty of the Terminal Connection Problem. Presented in the *47th International Conference on Current Trends in Theory and Practice of Computer Science* (SOFSEM 2021) [41], and submitted in March 2022 to *RAIRO - Theoretical Informatics and Applications.*

Chapter 3 consists of the results for the MAXCUT problem, which are contained with the complete proofs in the following appendixes.

- Appendix E. Manuscript: Celina M. H. de Figueiredo, Alexsander A. de Melo, Fabiano de Oliveira, Ana Silva. Maximum Cut on Interval Graphs of Interval Count Four is NP-complete. Presented in the *46th International Symposium on Mathematical Foundations of Computer Science (MFCS 2021)* [35], and submitted in December 2021 to *Discrete & Computational Geometry.*

- Appendix F. Manuscript: Celina M. H. de Figueiredo, Alexsander A. de Melo, Fabiano de Oliveira, Ana Silva. MaxCut on Permutation Graphs is NP-complete. Submitted in March 2022 to *Journal of Graph Theory* [36].

Chapter 4 is devoted to the problem of computing the zig-zag number of a directed graph. The results presented in this chapter are contained in the following appendix.

- Appendix G. Manuscript: Mitre C. Dourado, Celina M. H. de Figueiredo, Alexsander A. de Melo, Mateus de Oliveira Oliveira, Uéverton S. Souza. *Computing the Zig-Zag Number of Directed Graphs*. Published in Discrete Applied Mathematics 312 (2022) [50].

Chapter 5 consists of the conclusions of this work, and a selection of the main open questions motivated by the research. Last but not least, Appendix H consists of one-page abstracts of five additional works published during the doctoral studies. The first abstract corresponds to the work developed during the Master's thesis [39], but presented and fully published during the doctorate. The second abstract corresponds to current work recently presented [37]. The remaining three abstracts register and acknowledge the very fruitful visit to the University of Bergen [42–44].

For readability, we present only a sketch of the proof for some of the proposed results. In those cases, we explicitly refer the reader to the respective appendix containing the complete proof.

1.1 An Overview of the Selected Problems

In this section, we provide an overview of the problems addressed in this thesis.

1.1.1 Connection Problems

One of the most important connection problems is the classical Steiner tree problem, which not surprisingly is used to model several natural applications [21, 77, 88, 112]. Given a graph G and a terminal set $W \subseteq V(G)$, the objective of Steiner tree is to find a tree subgraph containing W that has the minimum possible number of non-terminal vertices. Steiner tree is one of the 21 problems proved to be NP-hard by Karp [86], and it has been investigated from distinct classes of algorithmic paradigms, such as structural graph classes [24, 33, 34, 65, 99, 115], approximation algorithms [14, 17, 22, 87], and parameterized complexity [8, 31, 32, 37, 56, 100].

In particular, Steiner tree was proved to be NP-complete on planar graphs [65] and on undirected path graphs [34]; on the other hand, it is known to be polynomial-time solvable on strongly chordal graphs [115]. Regarding parameterized complexity, the problem is in FPT when parameterized by the number of terminal vertices [56], while it is W[2]-hard when parameterized by the maximum number of non-terminal vertices allowed to be in the sought tree [97]. We refer to [34] for a detailed overview of the complexity of Steiner tree restricted to particular graph classes.

In this thesis, we study the variant of Steiner tree called Terminal connection problem (or TCP, for short). Given a graph G and a terminal set $W \subseteq V(G)$, the objective of TCP is to find a tree subgraph containing W, whose leaves belong to W, and that has a bounded number of non-terminal vertices of degree exactly 2, called *linkers*, and a bounded number of non-terminal vertices of degree at least 3, called *routers*. We remark that, while in Steiner tree the non-terminal vertices are not distinguishable among themselves, in TCP such vertices are partitioned into linkers and routers, according to their degree in the tree. Thus, by bounding the number of linkers by a non-negative integer ℓ and the number of routers by a non-negative r, the complexity analysis of the problem may vary considerably according to the values of ℓ and r, if compared to Steiner tree. Indeed, as we show in Chapter 2, there are graph classes on which TCP is polynomial-time solvable if r is fixed (and ℓ is arbitrary), whereas Steiner tree is NP-complete, and vice-versa.

TCP was introduced by Dourado et al. [52], and it was proved to be polynomial-time solvable when the parameters ℓ and r are both fixed [52], and to be NP-complete even if either $\ell \geq 0$ or $r \geq 0$ is fixed [52]. In addition to TCP, Dourado et al. [51] proposed its strict variant, called Strict terminal connection problem (or, S-TCP for short), which has the same input and objective of TCP apart from requiring that the leaf set of the sought tree coincides with the input terminal set. S-TCP was also proved to be polynomial-time solvable if ℓ and r are both fixed [51], and to be NP-complete if ℓ is fixed. Nevertheless, the complexity of the problem with r fixed remains an open question. This was addressed in the master's thesis [39] and in [40].

In addition to TCP and S-TCP, several other variants of Steiner tree have been studied over the years, such as Full Steiner tree [79, 80, 92], Group Steiner tree [49, 76], and Directed Steiner tree [76, 85]. Another variant that has been investigated is the one in which the number of *branching nodes* in the sought tree T, *i.e.* vertices (which not necessarily are non-terminal) of degree at least 3 in T, is bounded. In [68, 113, 114], the authors considered the undirected and directed cases of this variant, for which they devised approximation and parameterized tractable algorithms, besides providing some intractability results.

1.1.2 Maximum Cut

A *cut* is a partition $[A, B]$ of the vertex set of a graph G into two disjoint parts $A, B \subseteq V(G)$. Given a graph, the objective of the MaxCut problem is to obtain a cut $[A, B]$ that has the maximum number of *crossing edges*, *i.e.* edges with an endpoint in A and the other endpoint in B. The decision version of MaxCut is a classical NP-complete problem [67], and it is known to remain NP-complete on split graphs [10], undirected path graphs [10], and on comparability graphs [106]. On the other hand, the problem is polynomial-time solvable on planar graphs [75] and on split unit interval graphs [9]. Additionally, the problem is known to be in FPT on general graphs [93]. Despite being widely studied, only recently the restriction of MaxCut to interval graphs has been settled to be hard [1].

An *interval model* is a family of closed intervals of the real line. A graph is an *interval graph* if there exists an interval model, for which each interval corresponds to a vertex of the graph, such that distinct vertices are adjacent in the graph if and only if the corresponding intervals intersect. The notion of *interval count* of an interval graph was introduced in the eighties, and it is defined as the smallest number of interval lengths used by an interval model of the graph *cf.* [90]. Interval graphs having interval count 1 are called *unit interval.* Understanding the interval count, besides being an interesting and challenging problem by itself, can be also of value for the investigation of problems that are hard for general interval graphs, and easy for unit interval graphs (e.g. geodetic number [20, 57], optimal linear arrangement [23, 82], sum coloring [94, 101]). Surprisingly enough, the recognition of interval graphs with interval count k is open, even for $k = 2$ [19, 48].

In the same way as MaxCut on interval graphs has evaded being solved for so long, the community has been puzzled by the restriction to unit interval graphs. As a matter of fact, two attempts at solving it in polynomial time were proposed in [11, 15], but they were both disproved closely later [9, 89]. In this thesis, we provide the first classification that bounds the interval count, namely, we prove that

MaxCut is NP-complete when restricted to interval graphs of interval count 4.

In addition, we also prove that MaxCut is NP-complete on permutation graphs, which is the class of graphs that are both comparability and co-comparability [105]. This result answers a long-standing open question from [83].

1.1.3 Zig-Zag Number

A significant factor for the thriving of the theory of parameterized complexity was the introduction of width measures defined upon the structure of graphs, such as tree-width, cut-width and clique-width. This owns to the fact that many problems that are hard on general graphs can be solved efficiently when parameterized by such measures. However, most of these measures disregard edge direction. For instance, directed acyclic graphs (DAGs) in general have unbounded width with respect to any of the measures mentioned above, whereas several problems on directed graphs become tractable when restricted to DAGs by using straightforward algorithms. For instance, Directed Hamiltonian path can be solved in linear time on DAGs with a depth-first search algorithm.

Building on this observation, Johnson, Robertson, Seymour and Thomas [84] initiated a quest for the development of width measures that explicitly take the direction of edges into account. In particular, they defined in [84] the notion of directed tree-width and showed that some linkage problems that are NP-hard on general directed graphs can be solved in polynomial-time on directed graphs of constant directed tree-width. Additionally, a directed analogous notion of path-width was defined around the same time *cf.* [2]. This motivated the development of many other measures for directed graphs that focus on distinct algorithmic or structural properties [6, 7, 64, 78, 107, 109].

A general algorithmic framework for directed width measures was developed in [45] with the definition of *zig-zag* number of a directed graph, and subsequently generalized in [46] with the definition of *tree-zig-zag*. More specifically, it was shown in [45] that if $\mathcal{G}$ is a class of directed graphs expressible by a monadic-second order logic formula φ and there is a positive integer p such that each directed graph in $\mathcal{G}$ can be cast as a union of p directed paths, then, given a decomposition of a directed graph G of zig-zag number at most k, one can count in time $f(\varphi, p, k) \cdot |G|^{\mathcal{O}(p \cdot k)}$ the number of subgraphs of G isomorphic to some member of $\mathcal{G}$. Since directed path decompositions of width d can be efficiently converted into decompositions of zig-zag number $\mathcal{O}(d)$ *cf.* [45], the mentioned counting problem can also be solved in time $f(\varphi, p, d) \cdot |G|^{\mathcal{O}(p \cdot d)}$ on directed graphs of directed path-width at most d. This result was generalized in [46] to its respective counterpart for directed graphs of tree-zig-zag number at most k and of directed tree-width at most d.

An interesting aspect of zig-zag number and tree-zig-zag number is the fact that, besides being algorithmically relevant measures, they can be regarded as graph invariants, defined over cuts of a graph, with challenging theoretical open problems by themselves, from the perspectives of computational complexity and graph theory. In fact, many complexity questions with respect to computing zig-zag number and tree-zig-zag number of a directed graph remain open. In this thesis, we prove that k-ZIG-ZAG NUMBER, the problem of deciding whether a directed graph G has zig-zag number at most k, can be solved *non-deterministically* in time $|G|^{\mathcal{O}(k)}$, implying that this problem lies in NP for each fixed k. While the respective statement is almost trivial with respect to most of natural decision problems, our proof settling k-ZIG-ZAG NUMBER in NP turned out to be an interesting quest. This is due to the fact that the definition of zig-zag number, formally given in Chapter 4, involves the alternation of an existential and a universal quantifiers. Thus, a naive application of the definition would only lead to a $\mathbf{\Sigma}_2^{\mathsf{P}}$-upper bound for the problem. To circumvent this, and settle the problem in NP, our proof may be regarded as a way of redefining the property of a directed graph having zig-zag number at most k in a purely existential fashion. Additionally, we prove that 2-ZIG-ZAG NUMBER is an NP-hard problem.

1.2 Preliminaries

In this section, we present the basic definitions and terminology of graph theory that are used throughout this thesis. For any missing definition, we refer to the book [12]. The definitions and terminology of computational complexity used are the usual and are not explicitly presented. We refer to the books [31, 55, 66] for background on this subject.

Throughout this text, consider $[n] = \{1, \ldots, n\}$ for every positive integer n. Moreover, for every set V on n elements and every bijection $\pi\colon V \to [n]$, we let $<_\pi \subseteq V \times V$ be the linear order associated with π such that, for each two elements $u, v \in V$, $u <_\pi v$ if and only if $\pi(u) < \pi(v)$. Analogously, we let $>_\pi \subseteq V \times V$ be the linear order such that, for each two elements $u, v \in V$, $u >_\pi v$ if and only if $\pi(u) > \pi(v)$. If $V = \{u_1, \ldots, u_n\}$ and $\pi(u_i) = i$ for each $i \in [n]$, then we may write $\pi = (u_1, \ldots, u_n)$ to explicitly denote π; in this case, we also refer to π as an *ordering* of V. Let $X, Y \subseteq V$ be two non-empty sets. We write $X <_\pi Y$ to denote that $u <_\pi v$ for each $u \in X$ and each $v \in Y$. We define $X >_\pi Y$ similarly. In addition, for any non-empty set $X \subseteq V$, we write $\min_\pi X$ to denote the unique element $u \in X$ such that $\{u\} <_\pi X \setminus \{u\}$. We define $\max_\pi X$ similarly. When π is clear in the context, we may simply write $<$ and $>$ to refer to $<_\pi$ and $>_\pi$, respectively.

Graphs. A *directed graph* is an ordered pair $G = (V(G), E(G))$ such that $V(G)$

is a non-empty finite set of elements, called *vertices*, and $E(G)$ is a set of *ordered* pairs, called *edges*, of distinct vertices, *i.e.* $E(G) \subseteq \{(u,v) \in V(G) \times V(G) \colon u \neq v\}$. The notion of *undirected graphs* is defined analogously: An *undirected graph* (or, simply *graph*) is an ordered pair $G = (V(G), E(G))$ consisting of a non-empty finite set $V(G)$ of elements, called *vertices*, and a set $E(G)$ of *unordered* pairs, called *edges*, of distinct vertices, *i.e.* $E(G) \subseteq \{\{u,v\} \colon u, v \in V(G),\ u \neq v\}$.

We remark that, without loss of generality, *undirected* graphs can be regarded as a particular case of *directed* graphs, where the edge set must be a symmetric relation. In other words, for any two distinct vertices $u, v \in V(G)$, (u,v) is an edge of an undirected graph G if and only if (v,u) also is an edge of G. In this case, we actually consider the pairs (u,v) and (v,u) as being the single (undirected) edge $\{u,v\}$ of G; and, for simplicity of notation, we write uv to denote $\{u,v\}$. Based on that, all the definitions and notation described next for directed graphs can be immediately extended to undirected graphs as well, whenever it makes sense.

Let G be a directed graph. We let $|G|$ denote the number of vertices of G. Additionally, when G is clear in the context, we may simply write n and m to denote the number of vertices and the number of edges of G, respectively. We say that G is *trivial* if $|G| = 1$; otherwise, we say that G is a *non-trivial* graph.

Adjacency. If $e = (u,v) \in E(G)$, then we say that u and v are the *endpoints* of e, and that they are *incident* to e in G. In this case, we also say that e is an *in-edge* of v and an *out-edge* of u in G, and that u is an *in-neighbour* of v, and v is an *out-neighbour* of u in G. More generally, if $\{(u,v),(v,u)\} \cap E(G) \neq \emptyset$, then we simply say that u and v are *neighbours* and are *adjacent* in G.

For every vertex $v \in V(G)$, we write

$$N_G^-(v) = \{u \in V(G) \colon (u,v) \in E(G)\}, \quad N_G^+(v) = \{u \in V(G) \colon (v,u) \in E(G)\},$$

$N_G(v) = N_G^-(v) \cup N_G^+(v)$, and $N_G[v] = N_G(v) \cup \{v\}$ to denote the *in-neighbourhood*, the *out-neighbourhood*, the *neighbourhood*, and the *closed neighbourhood* of v in G, respectively. For every subset $V' \subseteq V(G)$, we let $N_G(V') = \bigcup_{v \in V'} N_G(v)$. The sets $N_G^-(V')$, $N_G^+(V')$, and $N_G[V']$ are defined analogously.

We write $d_G^-(v) = |N_G^-(v)|$, $d_G^+(v) = |N_G^+(v)|$, and $d_G(v) = |N_G(v)|$ to denote the *in-degree*, the *out-degree*, and the *degree* of v in G, respectively. The *maximum degree* of G, denoted by $\Delta(G)$ is defined as the integer $\max_{v \in V'} d_G(v)$.

A vertex v is an *isolated* vertex of G if $N_G(v) = \emptyset$; on the other hand, v is a *universal* vertex of G if $N_G^-(v) \cup \{v\} = N_G^+(v) \cup \{v\} = V(G)$. A *pendant* vertex of G is a vertex of degree exactly 1 in G.

Two vertices u and v are said to be *true twins* in G if $N_G[u] = N_G[v]$; and they are said to be *false twins* in G if $N_G(u) = N_G(v)$.

Clique and Stable Set. A subset $K \subseteq V(G)$ is called a *clique* of a graph G if every two distinct vertices belonging to K are adjacent in G, *i.e.* $K \subseteq N_G[v]$ for every $v \in K$. On the other hand, a subset $S \subseteq V(G)$ is called a *stable set* of G if no two vertices belonging to S are adjacent in G, *i.e.* $S \cap N_G(v) = \emptyset$ for every $v \in S$.

A graph is called *complete* if its vertex set is a maximal clique. A graph is called *bipartite* if its vertex set can be partitioned into two disjoint stable sets. A graph is called a *split graph* if its vertex set can be partitioned into a clique and a stable set.

Let X and Y be two disjoint subsets of $V(G)$. We say that X is *complete* to Y if every vertex in X is adjacent to every vertex in Y, and that X is *anti-complete* to Y if there are no edges between the vertices in X and the vertices in Y.

Matching. Two distinct edges of a directed graph are said to be *adjacent* if they share an endpoint. A *matching* of G is a set of pairwise non-adjacent edges of G. Given a matching M of G, we say that M *saturates* a vertex of G if this vertex is an endpoint of an edge in M.

Cut and Cut-Set. We let $E_G(X, Y)$ denote the set of edges of G *between* X and Y, *i.e.* the edges with an endpoint in X and the other endpoint in Y. In particular, for every proper subset $X \subset V(G)$, we may simply write $E_G(X)$ to denote $E_G(X, V(G) \setminus X)$.

A *cut* of G is a partition of $V(G)$ into two parts $X, Y \subseteq V(G)$, denoted by $[X, Y]$. For every cut $[X, Y]$ of G, the edge set $E_G(X, Y)$ is called the *cut-set* of G associated with $[X, Y]$. For every two vertices $u, v \in V(G)$, we say that u and v *are in a same part of a cut* $[X, Y]$ if either $\{u, v\} \subseteq X$ or $\{u, v\} \subseteq Y$; otherwise, we say that u and v *are in opposite parts of* $[X, Y]$.

Subgraphs. A directed graph H is a *subgraph* of G if $V(H) \subseteq V(G)$ and $E(H) \subseteq E(G)$. In this case, we also say that G *contains* H.

A subgraph H of G is said to be *spanning* if $V(H) = V(G)$.

Given a non-empty subset $V' \subseteq V(G)$, the *subgraph of* G *induced by* V', denoted by $G[V']$, is the subgraph H of G such that $V(H) = V'$ and, for every two vertices $u, v \in V(H)$, it holds that $(u, v) \in E(H)$ if and only if $(u, v) \in E(G)$. An *induced subgraph* of G is a directed graph H such that, for some $V' \subseteq V(G)$, H is the induced subgraph of G induced by V'. For every proper subset $V' \subset V(G)$, we denote by $G - V'$ the directed graph obtained from G by removing the vertices belonging to V' and their incident edges, *i.e.* $G - V'$ is the subgraph of G induced by $V(G) \setminus V'$.

The notion of subgraphs induced by edges can be defined analogously. Given a subset $E' \subseteq E(G)$, the *subgraph of* G *induced by* E', denoted by $G[E']$, is the subgraph H of G such that $E(H) = E'$ and, for every $v \in V(G)$, it holds that $v \in V(H)$ if and only if v is an endpoint of an edge in E'. For every $E' \subseteq E(G)$, we let $G - E'$ denote the directed graph obtained from G by removing the edges in E', *i.e.* $G - E'$ is the directed graph with vertex set $V(G)$ and edge set $E(G) \setminus E'$.

Paths, Cycles and Trees. An *undirected path* (or, simply *path*) of G is a sequence $P = (v_1, \dots, v_q)$ of distinct vertices of G, such that $\{(v_i, v_{i+1}), (v_{i+1}, v_i)\} \cap E(G) \neq \emptyset$ for every $i \in [q-1]$, where $q \geq 1$. In this case, we say that P is a path *between* v_1 and v_q. Such a sequence is called a *directed path from* v_1 *to* v_q if, for every $i \in [q-1]$, $(v_i, v_{i+1}) \in E(G)$. An *undirected cycle* (or, simple *cycle*) of G is a sequence $(v_1, \dots, v_q)$ of vertices of G, such that $(v_1, \dots, v_q)$ is a path of G and $\{(v_q, v_1), (v_1, v_q)\} \cap E(G) \neq \emptyset$. Analogously, a *directed cycle* of G is an undirected cycle $(v_1, \dots, v_q)$ such that $(v_1, \dots, v_q)$ is a directed path of G and $(v_q, v_1) \in E(G)$.

The *length* of a path (of a cycle) is the number of distinct vertices belonging to it. The *distance* from a vertex u to a vertex v in a graph (resp. directed graph) G, denoted by $\operatorname{dist}_G(u, v)$, is defined as the minimum length among all paths (resp. directed paths) in G from u to v. If there is no such a path, then we define $\operatorname{dist}_G(u, v) = \infty$. For every vertex $u \in V(G)$ and every subset $V' \subseteq V(G)$, we let $\operatorname{dist}_G(u, V) = \min_{v \in V'} \operatorname{dist}_G(u, v)$.

We say that a directed graph G is a *path/directed path/cycle/directed cycle* if there exists a linear order of its vertices $(v_1, \dots, v_n)$ which is a *path/directed path/cycle/directed cycle* of G, respectively. A graph is said to be *acyclic* (resp. *directed acyclic*) if it does not contain any undirected cycle (resp. directed cycle).

A *weakly connected component* of G is a maximal subgraph H of G such that, for every two vertices $u, v \in V(H)$, there is an undirected path in H between u and v. A *connected component* of G is a maximal subgraph H of G such that, for every two vertices $u, v \in V(H)$, there is a directed path in H from u to v, or from v to u. Then, we say that a directed graph is *connected* (resp. *weakly connected*) if it is the only connected (resp. weakly connected) component of itself.

A *forest* is a directed graph that is acyclic. A *tree* is a weakly connected forest. A vertex is called a *leaf* of a tree T if its degree in T is equal to 1. The *leaf set* of a tree T, denoted by $\operatorname{leaves}(T)$, is defined as the set of all leaves of T.

A *rooted directed tree* is a tree T that contains a vertex r *reaching* every other vertex, *i.e.* for every $v \in V(T) \setminus \{r\}$, there is a directed path in T from r to v; in this case, we say that r is the *root* of $T$and that T is *rooted at* r.

Disjoint Union and Join. Let $G_1, \dots, G_k$ be $k \geq 2$ graphs, with mutually disjoint vertex sets. The *disjoint union* of $G_1, \dots, G_k$, denoted by $G_1 \oplus \cdots \oplus G_k$, is the graph H with vertex set $V(H) = V(G_1) \cup \cdots \cup V(G_k)$ and edge set $E(H) = E(G_1) \cup \cdots \cup E(G_k)$. We note that, for each $i \in [k]$, G_i is a connected component of $G_1 \oplus \cdots \oplus G_k$. The *join* of $G_1, \dots, G_k$, denoted by $G_1 \wedge \cdots \wedge G_k$, is the graph H with vertex set $V(H) = V(G_1 \oplus \cdots \oplus G_k)$ and edge set

$$E(H) = E(G_1 \oplus \cdots \oplus G_k) \cup \{uv \colon u \in V(G_i),\ v \in V(G_j),\ i, j \in [k],\ i \neq j\}.$$

We note that $G_1 \wedge \cdots \wedge G_k$ is a connected graph, such that, for each pair $i, j \in [k]$ with $i \neq j$, $V(G_i)$ is complete to $V(G_j)$.

Chapter 2

Connection Problems

In this chapter, we analyse the computational complexity of the network design problem called Terminal connection, and of its strict variant, called Strict terminal connection.

Network design problems are generally combinatorial questions of great practical and theoretical interest, related to several real-world applications. One of the most fundamental and well-known problems in this field is Steiner tree, proved to be NP-hard by Karp in his seminal paper [86]. Since then, Steiner tree has been extensively studied from the perspective of graph classes [24, 33, 34, 65, 99, 115] and computational complexity [8, 32, 56, 100], being one of the eleven problems selected by David S. Johnson to appear in the *Ongoing Guide to NP-completeness* [83].

Motivated by applications in information security, network routing and telecommunications, Dourado et al. [52] introduced recently the Terminal connection problem (TCP), which can be seen as a generalisation of Steiner tree, and proved that on general graphs TCP is NP-complete. We investigate the complexity of TCP and of its strict variant, S-TCP, when restricted to specific graph classes and some of the input parameters are fixed. In particular, we mainly give emphasis on results that separate the complexity of TCP from the complexity of Steiner tree.

This chapter is organised as follows. In Section 2.1, we define the notion of *connection tree* of a graph, which is the main object of study in this chapter. Based on that, we formally define the Steiner tree, Terminal connection and Strict terminal connection problems, and then we present how these problems are related to each other. In Section 2.2, we prove that the classes of split graphs and rooted directed path graphs separates the complexity of TCP from the complexity of Steiner tree. Additionally, in Section 2.3, we investigate the complexity of TCP on graphs of bounded clique-width. More specifically, we prove that, when parameterized by clique-width, TCP is W[1]-hard while Steiner tree is known to be in FPT [4]. On the other hand, agreeing with the complexity of Steiner tree [24, 99], we prove in Section 2.3.2 that TCP is polynomial-time solvable on cographs (*i.e.* graphs of clique-width at most 2). Finally, in Section 2.4, we present the concluding remarks of this chapter, focusing on the open questions.

2.1 Connection Tree

Let G be a graph and $W \subseteq V(G)$ be a non-empty set. A *connection tree* T of G for W is a tree subgraph of G such that $\text{leaves}(T) \subseteq W \subseteq V(T)$, where $\text{leaves}(T)$ denotes the leaf set of T.

The Steiner tree problem has as input a connected graph G, a non-empty terminal set $W \subseteq V(G)$, and a non-negative integer k, and it asks whether there exists a connected subgraph T of G such that $W \subseteq V(T)$ and $|V(T) \setminus W| \leq k$. Note that, if such a connected subgraph exists, then it admits a spanning tree with at most k vertices not in W. Thus, Steiner tree can be alternatively defined in terms of connection tree, as described next.

Steiner Tree

Input: A connected graph G, a non-empty terminal set $W \subseteq V(G)$ and a non-negative integer k.

Question: Is there a connection tree T of G for W such that $|V(T) \setminus W| \leq k$?

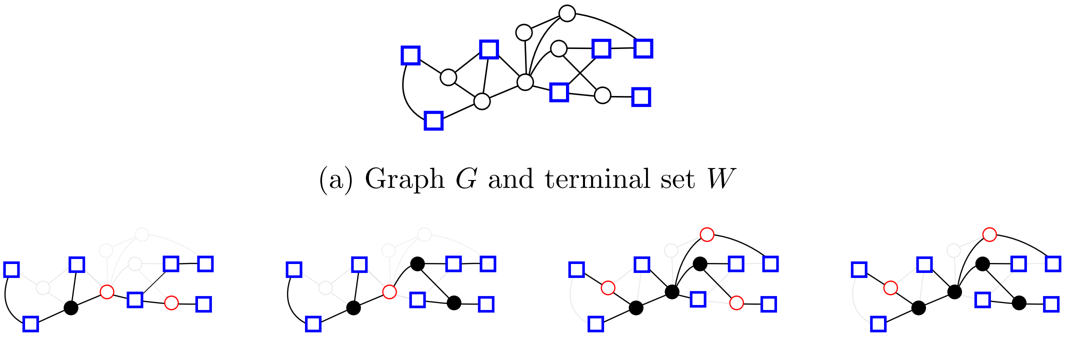
In a connection tree T for W, the vertices belonging to W are called *terminal*, and the vertices belonging to $V(T) \setminus W$ are called *non-terminal* and are classified into two types according to their respective degrees in T, namely (see Figure 2.1): the non-terminal vertices of degree exactly 2 in T are called *linkers* and the non-terminal vertices of degree at least 3 in T are called *routers*. Thus, the vertex set of every connection tree can be partitioned into terminal vertices, linkers and routers. For each connection tree T, we let $\mathsf{L}(T)$ denote the linker set of T and $\mathsf{R}(T)$ denote the router set of T. Next, we formally define the Terminal connection problem.

Terminal Connection (TCP)

Input: A connected graph G, a non-empty terminal set $W \subseteq V(G)$ and two non-negative integers ℓ and r.

Question: Is there a connection tree T of G for W such that $|\mathsf{L}(T)| \leq \ell$ and $|\mathsf{R}(T)| \leq r$?

Dourado et al. [52] introduced TCP and proved that the problem is polynomial-time solvable (more specifically, that it is in XP) when the parameters ℓ and r are both fixed, whereas it is NP-complete even if exactly one of the parameters $\ell \geq 0$ or $r \geq 0$ is fixed. We extended this result by proving that TCP remains NP-complete



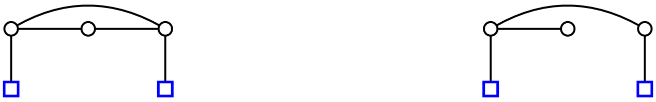
(a) Graph G and terminal set W

(b) T_1 (c) T_2 (d) T_3 (e) T_4

Figure 2.1: A graph G, a terminal set W (blue squares), and connection trees of G for W, each with a distinct number of linkers (red circles) and routers (solid black circles). Indeed, T_1 has 2 linkers and 1 router; T_2 has 1 linker and 3 routers; T_4 has 3 linkers and 3 routers; and T_4 has 2 linkers and 4 routers. In particular, note that T_3 and T_4 are strict connection trees, *i.e.* their leaf sets coincide with W.

on graphs of maximum degree 3 even if either ℓ or r is fixed [39, 41] (for more details, see Section 4 in Appendix D).

It is worth mentioning that, regarding Steiner tree, the constraints on T being a tree and its leaf set being a subset of W can be omitted without loss of generality from the definition of the problem, since every minimal solution T for a given instance of Steiner tree is necessarily a connection tree for W. However, this does not apply to TCP: neither constraint can be ignored. In fact, as a result of the number of non-terminal vertices with degree 2 being bounded, there exist instances (G, W, ℓ, r) that would be considered yes-instances of TCP even though all connected subgraphs of G containing the vertices in W, and with at most ℓ non-terminal vertices of degree 2 and at most r non-terminal vertices of degree at least 3, have cycles or non-terminal vertices that are leaves (see Figure 2.2 for an example).



(a) Graph G and terminal set W (b) Tree subgraph T of G

Figure 2.2: (a) A graph G, a terminal set W (blue squares) such that, for every $r \geq 0$ and for $\ell = 1$, (G, W, ℓ, r) is a no-instance of TCP, since every connection tree of G for W has at least 2 linkers. (b) A tree subgraph T of G such that $W \subseteq V(T)$ but leaves$(T) \nsubseteq W$, which contains exactly one non-terminal vertex of degree 2 and exactly one non-terminal vertex of degree at least 3.

A connection tree T for a terminal set W is called *strict* if its leaf set coincides with W, *i.e.* leaves$(T) = W$ (see Figure 2.1). The Strict terminal connection problem has the same input and the same question as TCP except for requiring a *strict* connection tree, instead of a regular connection tree, with a bounded number of linkers and a bounded number of routers.

Strict Terminal Connection (S-TCP)

Input:	A connected graph G, a non-empty terminal set $W \subseteq V(G)$ and two non-negative integers ℓ and r.
Question:	Is there exist a strict connection tree T of G for W, *i.e.* a connection tree T with leaves$(T) = W$, such that $\vert\mathsf{L}(T)\vert \leq \ell$ and $\vert\mathsf{R}(T)\vert \leq r$?

Similarly to TCP, Dourado et al. [51] introduced S-TCP, and proved that S-TCP is polynomial-time solvable when the parameters $\ell \geq 0$ and $r \geq 0$ are both fixed, while it is still NP-complete if $\ell \geq 0$ is fixed. As for fixing r, except for the case $r \in \{0, 1\}$, which was shown to be polynomial-time solvable [39, 95], the complexity of S-TCP for fixed $r \geq 2$ has remained open. It is also worth mentioning that, just as TCP can be seen as a generalization of Steiner tree, as we show next, S-TCP can be seen as a generalization of Full Steiner tree (also called Terminal Steiner tree), which is a widely studied variant of Steiner tree [80, 91, 92].

2.2 Separating Graph Classes

An important remark about TCP and Steiner tree is the fact that TCP can be seen as a generalisation of Steiner tree, in the sense that every instance of Steiner tree can be described as an equivalent disjunction of instances of TCP.

Indeed, there is a Turing reduction from Steiner tree to TCP, namely: (G, W, k) is a yes-instance of Steiner tree if and only if (G, W, ℓ, r) is a yes-instance of TCP for some pair $\ell, r \in \{0, \ldots, k\}$ such that $\ell + r = k$. An interesting aspect of this Turing reduction is the fact that it preserves the structure of the input graph. Consequently, if TCP is polynomial-time solvable on some graph class $\mathcal{G}$, then so is Steiner tree. Analogously, if Steiner tree is NP-complete on some graph class $\mathcal{G}$, then TCP cannot be solved in polynomial-time on $\mathcal{G}$, unless P=NP. Nevertheless, if either $\ell \geq 0$ or $r \geq 0$ is fixed, then possibly TCP is polynomial-time solvable on $\mathcal{G}$, while Steiner tree remains NP-complete on $\mathcal{G}$. In addition, there might exist a graph class $\mathcal{G}$ on which Steiner tree is polynomial-time solvable whereas TCP remains NP-complete.

In this section, we confirm the existence of such complexity separating classes, by proving that, on *split* graphs, TCP is polynomial-time solvable if $r \geq 1$ is fixed, whereas Steiner tree is known to be NP-complete [115]. Besides, we prove that TCP remains NP-complete on *rooted directed path* graphs even if $r \geq 0$ is fixed, whereas Steiner tree is known to be polynomial-time solvable on *strongly chordal* graphs [115], which in turn is a superclass of rooted directed path graphs [16, 59].

2.2.1 Split Graphs

We recall that a split graph is a graph whose vertex set can be partitioned into a clique and a stable set. For simplicity, throughout this section, we write $G\langle K, S\rangle$ to refer to a split graph G such that $K \cup S$ is a partition of the vertex set of G into a clique K and a stable set S. We show that TCP on split graphs can be solved in time $n^{\mathcal{O}(r)}$ for every $r \geq 1$, and that, under complexity assumptions broadly believed to hold, this time complexity cannot be considerably improved. More specifically, we prove the theorem below.

Theorem 2.1. *The following statements hold for TCP restricted to split graphs:*

- *For every $r \geq 1$, the problem can be solved in time $n^{\mathcal{O}(r)}$;*
- *For any computable functions g and h, the problem cannot be solved in time $g(r) \cdot n^{h(\ell)}$, unless $\mathsf{FPT} = \mathsf{W}[2]$;*
- *For any computable function g, the problem cannot be solved in time $g(r) \cdot n^{o(r)}$, unless ETH fails;*

To prove Theorem 2.1, we first show that S-TCP can be solved in time $n^{\mathcal{O}(r)}$ for every $r \geq 0$. In a nutshell, we present an algorithm for S-TCP that enumerates each possible candidate router set $R \subseteq V(G) \setminus W$, with $|R| \leq r$, and then decides through matching techniques whether the input split sgraph G admits a connection tree T for the terminal set W, such that $|\mathsf{L}(T)| \leq \ell$, $\mathsf{R}(T) = R$ and leaves$(T) = W$.

Fact 2.1. *Let $G\langle K, S\rangle$ be a split graph and $W \subseteq V(G)$, with $|W| \geq 3$. The following statements hold:*

- *If K is a subset of W, then G does not admit a strict connection tree for W.*
- *If W is a proper subset of K, then G admits a strict connection tree T for W such that $|\mathsf{L}(T)| = 0$ and $|\mathsf{R}(T)| = 1$.*

Lemma 2.1. *Let $G\langle K, S\rangle$ be a split graph and $W \subseteq V(G)$, with $|W| \geq 3$. Suppose that $K \setminus W \neq \varnothing$ and $W \cap S \neq \varnothing$. If G admits a strict connection tree T' for W, then there exists a strict connection tree T of G for W such that $|\mathsf{L}(T)| \leq |\mathsf{L}(T')|$, $|\mathsf{R}(T)| \leq |\mathsf{R}(T')|$, and $\mathsf{R}(T) \subseteq K$.*

Proof. Since S is a stable set of G, $N_{T'}(S) \subseteq K$. Moreover, it follows from the assumptions $W \cap S \neq \varnothing$ and $|W| \geq 3$ that T' contains a non-terminal vertex in K, otherwise T' would no be strict. Let v be such a vertex. Since K is a clique of G, $v \in N_G(v')$ for all $v' \in N_{T'}(S) \setminus \{v\}$. Then, let T be the graph obtained from T' by removing the router vertices in S and adding the edge vv' for each $v' \in N_{T'}(\mathsf{R}(T') \cap S)$. One can verify that T is a strict connection tree of G for W such that $|\mathsf{L}(T)| \leq |\mathsf{L}(T')|$, $|\mathsf{R}(T)| \leq |\mathsf{R}(T')|$, and $\mathsf{R}(T) \subseteq K$. □

Lemma 2.2. *Let $G\langle K, S\rangle$ be a split graph and $W \subseteq V(G)$, with $|W| \geq 3$. Let T' be a strict connection tree of G for W such that $\mathsf{R}(T') \subseteq K$. There exists a strict connection tree T of G for W, with $|\mathsf{L}(T)| \leq |\mathsf{L}(T')|$ and $\mathsf{R}(T) \subseteq \mathsf{R}(T')$, satisfying the following conditions:*

(i) $\mathsf{L}(T) \subseteq K$*;*

(ii) Each vertex in $\mathsf{L}(T)$ is adjacent to exactly one vertex in $\mathsf{R}(T)$ and exactly one vertex $w \in W$, where $w \in S$ and $w \notin N_G(\mathsf{R}(T))$;

(iii) $T[R(T)]$ is a path.

Proof. (i). For every vertex $u \in \mathsf{L}(T') \cap S$, if x_u and y_u are the two distinct neighbours of u in T', then $x_u, y_u \in K$. Thus, the graph obtained from T' by removing all the vertices in $\mathsf{L}(T') \cap S$ and adding all the edges in $\{x_u y_u \colon x_u, y_u \in N_{T'}(\mathsf{L}(T') \cap S)\}$ is a strict connection tree of G for W with linker set $\mathsf{L}(T') \setminus S \subseteq K$ and router set $\mathsf{R}(T')$. Thus, for simplicity, we assume hereinafter that $\mathsf{L}(T') \subseteq K$.

(ii). Since $|W| \geq 3$, for every vertex $u \in \mathsf{L}(T')$, if x_u and y_u are the two distinct neighbours of u in T', then $x_u \notin W$ or $y_u \notin W$, otherwise T' would not be a strict connection tree for W. Since $(\mathsf{L}(T') \cup \mathsf{R}(T')) \cap S = \emptyset$, it holds that if $x_u, y_u \notin W$, then $x_u, y_u \in K$. Hence, we can remove u from T' and add the edge $x_u y_u$. Thus, suppose that $x_u \in W$ and $y_u \notin W$. If $y_u \in \mathsf{L}(T')$, then y_u has exactly one neighbour in T' in addition to u. Let z be this second neighbour of y_u in T'. Since $|W| \geq 3$ and we are supposing that $x_u \in W$, we have $z \notin W$, which again implies $z \in K$. As a result, the graph obtained from T' by removing y_u and adding the edge uz is a strict connection tree of G for W with linker set $\mathsf{L}(T') \setminus \{y_u\}$ and router set $\mathsf{R}(T')$. Suppose now that $y_u \in \mathsf{R}(T')$ but there exists a vertex $\rho \in \mathsf{R}(T')$, possibly $\rho = y_u$, such that $\rho x_u \in E(G)$. Consequently, the graph H obtained from T' by removing u and adding the edge ρx_u is a strict connection tree of G for W such that $\mathsf{L}(H) = \mathsf{L}(T') \setminus \{u\}$ and $\mathsf{R}(H) = \mathsf{R}(T')$ if $d_{T'}(y_u) > 3$ or $\rho = y_u$, and $\mathsf{L}(H) = (\mathsf{L}(T') \setminus \{u\}) \cup \{y_u\}$ and $\mathsf{R}(H) = \mathsf{R}(T') \setminus \{y_u\}$ otherwise. Therefore, by applying successively the steps described above, it is always possible to obtain a strict connection tree of G for W satisfying condition (ii).

(iii). If $|\mathsf{R}(T')| \leq 1$, then trivially T' satisfies condition (iii). Thus, assume that $|\mathsf{R}(T')| \geq 2$. Additionally, assume that T' satisfies condition (ii). Consequently, $H_R = T[\mathsf{R}(T')]$ is a tree. Note that, H_R contains at least two leaves. Let R^* be the set defined in the following way: $\rho^* \in R^*$ if and only if $\rho^* \in \mathsf{R}(T')$ and there is at least one terminal vertex $w \in W$ such that $\mathrm{dist}_{T'}(w, \rho^*) = \mathrm{dist}_{T'}(w, \mathsf{R}(T'))$, *i.e.* the path between w and ρ^* in T' does not contain any other router. Note that every leaf of H_R necessarily belongs to R^*; more specifically, for every leaf ρ^* of H_R, there exist at least two distinct terminal vertices $w^1_{\rho^*}, w^2_{\rho^*} \in W$ such that $\mathrm{dist}_{T'}(w^i_{\rho^*}, \rho^*) = \mathrm{dist}_{T'}(w^i_{\rho^*}, \mathsf{R}(T'))$ for $i \in \{1, 2\}$, otherwise the degree of ρ^* in T' would be less than 3. Let $(\rho_1, \dots, \rho_k)$ be an ordering of the vertices in R^*, where $k = |R^*|$, and suppose that ρ_1 and ρ_k are leaves of H_R. Then, consider the graph T with vertex set $V(T) = V(T') \setminus (\mathsf{R}(T') \setminus R^*)$ and edge set

$$E(T) = (E(T') \setminus E(H_R)) \cup \{\rho_i\rho_{i+1} \colon i \in [k-1]\}.$$

One can verify that T is a strict connection tree of G for W such that $\mathsf{L}(T) = \mathsf{L}(T')$, $\mathsf{R}(T) = R^* \subseteq \mathsf{R}(T')$, and $T[\mathsf{R}(T)]$ is a path. □

Proposition 2.1. *Let $G\langle K, S\rangle$ be a split graph and $W \subseteq V(G)$, with $|W| \geq 3$. Suppose that $K \setminus W \neq \varnothing$ and $W \cap S \neq \varnothing$. Given two non-negative integers ℓ and r, we can in time $n^{\mathcal{O}(r)}$ obtain a strict connection tree T of G for W such that $|\mathsf{L}(T)| \leq \ell$ and $|\mathsf{R}(T)| \leq r$, or conclude that such a tree does not exist.*

Proof. It is known that, for $r \in \{0, 1\}$, S-TCP can be solved in polynomial-time on general graphs *cf.* [95]. Thus, for simplicity, assume that $r \geq 2$ and that G does not admit a strict connection tree T for W such that $|\mathsf{L}(T)| \leq \ell$ and $|\mathsf{R}(T)| \leq 1$. Based on Lemmas 2.1 and 2.2, our strategy consists in enumerating all possible router subsets $R \subseteq K \setminus W$, with $2 \leq |R| = k \leq r$, and all possible unordered pairs $\{\rho_1, \rho_k\} \subseteq R$ of distinct vertices in order to try to obtain a strict connection T of G for W such that $|\mathsf{L}(T)| \leq \ell$, $\mathsf{R}(T) = R$ and $T[\mathsf{R}(T)]$ is a path with endpoints ρ_1 and ρ_k. Hence, let $R \subseteq K \setminus W$, with $2 \leq |R| = k \leq r$, and ρ_1 and ρ_k be two distinct vertices belonging to R.

Let $W_R = W \cap N_G(R)$ and $\overline{W_R} = W \setminus W_R$. Note that, if $|\overline{W_R}| > \ell$, then there is no connection tree of G for W such that $|\mathsf{L}(T)| \leq \ell$ and $\mathsf{R}(T) = R$. Thus, assume $|\overline{W_R}| \leq \ell$. Let H_1 be the bipartite graph defined as follows:

$$V(H_1) = X_1 \cup Y_1 \quad \text{and} \quad E(H_1) = \{xy \in E(G) \colon x \in X_1, y \in Y_1\},$$

where $X_1 = \overline{W_R}$ and $Y_1 = N_G(X_1) \setminus W$. We remark that, since R is a subset of K, which is a clique, we have that $X_1 \subseteq S$. Thus, since S is a stable set, we additionally have that $Y_1 \subseteq K$.

Claim 2.1. *If $X_1 \neq \varnothing$ and G admits a strict connection tree T for W such that $|\mathsf{L}(T)| \leq \ell$ and $\mathsf{R}(T) = R$, then there exists a matching M_1 in H_1 that saturates all vertices belonging to X_1.*

Proof of claim. Assume that T satisfies conditions (i)–(iii) described in Lemma 2.2. Thus, each linker $u \in \mathsf{L}(T)$ is adjacent to exactly one vertex in $\mathsf{R}(T)$ and exactly one vertex $w \in W$ such that $w \notin N_G(\mathsf{R}(T))$. As a consequence, the set of terminal vertices which are adjacent to a linker of T coincides with X_1. In addition, since $|W| \geq 3$, each vertex belonging to $\mathsf{L}(T)$ is adjacent in T to at most one vertex belonging to X_1. Therefore, the set $M_1 = \{uw \in E(T) \colon u \in \mathsf{L}(T),\, w \in W\}$ is a matching in H_1 that saturates all vertices belonging to X_1. –

Based on Claim 2.1, we assume that $X_1 = \varnothing$, or that there exists a matching M_1 in H_1 that saturates all vertices belonging to X_1. Thus, given such a matching M_1 (if any), we let $L = \varnothing$ if $X_1 = \varnothing$, and $L = \{u \in Y_1 \colon uw \in M_1\}$ otherwise. Also, let $\rho_1^1, \rho_1^2, \rho_k^1, \rho_k^2$ be four new auxiliary vertices, not belonging to G. Then, consider $X_2 = (R \setminus \{\rho_1, \rho_k\}) \cup \{\rho_1^1, \rho_1^2, \rho_k^1, \rho_k^2\}$ and $Y_2 = W_R \cup L$. We define H_2 as the bipartite graph with vertex set $V(H_2) = X_2 \cup Y_2$ and edge set

$$\begin{aligned} E(H_2) = {} & \{xy \in E(G) \colon x \in R \setminus \{\rho_1, \rho_k\},\, y \in Y_2\} \\ & \cup \{\rho_i^j y \colon \rho_i y \in E(G),\, y \in Y_2,\, i \in \{1,k\},\, j \in \{1,2\}\}. \end{aligned}$$

Claim 2.2. *G admits a strict connection tree T for W such that $|\mathsf{L}(T)| \leq \ell$, $\mathsf{R}(T) = R$ and $T[\mathsf{R}(T)]$ is a path with endpoints ρ_1 and ρ_k if and only if there exists a matching M_2 in H_2 that saturates all vertices belonging to X_2.*

Proof of claim. First, suppose that such a tree T exists. Additionally, assume that T satisfies conditions (i)–(iii) described in Lemma 2.2. Then, $|L| = |X_1| = |\mathsf{L}(T)|$. Let $\phi \colon \mathsf{L}(T) \to L$ be an arbitrary bijection. Since all routers of T have degree at least 3, each endpoint of the path $T[\mathsf{R}(T)]$ — *i.e.* the vertices ρ_1 and ρ_k — must be adjacent to at least two distinct vertices in $W_R \cup \mathsf{L}(T)$; thus, for $i \in \{1,k\}$, let $v_i^1, v_i^2 \in W_R \cup \mathsf{L}(T)$ be two arbitrary distinct neighbours of ρ_i in T. Furthermore, we have that each internal vertex of $T[\mathsf{R}(T)]$ must be adjacent to at least one vertex in $W_R \cup \mathsf{L}(T)$; thus, for $i \in \{2, \dots, k-1\}$, let $v_i \in W_R \cup \mathsf{L}(T)$ be an arbitrary neighbour of ρ_i in T. Let $Y_{M_2} = \{v_i^j \colon i \in \{1,k\},\, j \in \{1,2\}\} \cup \{v_i \colon i \in \{2, \dots, k-1\}\}$. We remark that $Y_{M_2} \setminus W_R \subseteq \mathsf{L}(T)$. Additionally, note that R and L are both contained in K, which is a clique. Therefore, one can verify that

$$\begin{aligned} M_2 = {} & \{\rho_i^j v_i^j \colon \rho_i v_i^j \in E(T),\, v_i^j \in Y_{M_2} \cap W_R,\, i \in \{1,k\},\, j \in \{1,2\}\} \\ & \cup \{\rho_i v_i \in E(T) \colon v_i \in Y_{M_2} \cap W_R,\, i \in \{2, \dots, k-1\}\} \\ & \cup \{\rho_i^j \phi(v_i^j) \colon \rho_i v_i^j \in E(T),\, v_i^j \in Y_{M_2} \setminus W_R,\, i \in \{1,k\},\, j \in \{1,2\}\} \\ & \cup \{\rho_i \phi(v_i) \colon \rho_i v_i \in E(T),\, v_i \in Y_{M_2} \setminus W_R,\, i \in \{2, \dots, k-1\}\} \end{aligned}$$

is a matching in H_2 that saturates all vertices belonging to X_2.

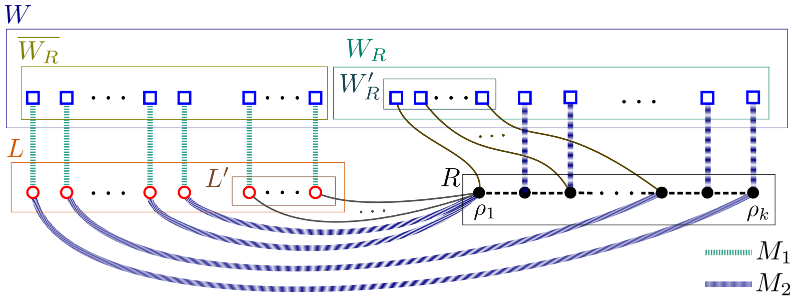


Figure 2.3: Strict connection tree of G for W obtained from a matching M_2 in H_2 that saturates all vertices belonging to X_2.

Conversely, suppose that there exists a matching M_2 in H_2 that saturates all vertices belonging to X_2. Let W'_R and L' be the subsets of W_R and L, respectively, composed by the vertices which are not saturated by M_2. Also, let $\varphi\colon W'_R \to R$ be a mapping such that, for each $w \in W'_R$, if $\varphi(w) = \rho$, then $w \in N_G(\rho)$. Consider the graph T defined as follows (see Figure 2.3): $V(T) = W \cup L \cup R$ and

$$\begin{aligned} E(T) = {} & M_1 \cup (M_2 \setminus \{\rho_i^j v_i \colon v_i \in Y_2,\, i \in \{1,k\},\, j \in \{1,2\}\}) \\ & \cup \{\rho_i v_i \colon \rho_i^j v_i \in M_2,\, v_i \in Y_2, i \in \{1,k\},\, j \in \{1,2\}\} \\ & \cup \{\rho_1 v \colon v \in L'\} \cup \{\varphi(w)w \colon w \in W'_R\} \cup \{\rho_i\rho_{i+1} \colon i \in [k-1]\}. \end{aligned}$$

One can verify that T is a strict connection tree of G for W such that $\mathsf{L}(T) = L$, $\mathsf{R}(T) = R$ and $T[\mathsf{R}(T)]$ is a path with endpoints ρ_1 and ρ_k. –

To complete the proof of this proposition, we remark that, based on Lemmas 2.1 and 2.2, there exists a strict connection tree of G for W with at most ℓ linkers and at most r routers if and only if, for some set $R \subseteq K \setminus W$, with $2 \leq |R| = k \leq r$, and some unordered pair $\{\rho_1, \rho_k\} \subseteq R$ of distinct vertices, there exists a strict connection tree T of G for W such that $|\mathsf{L}(T)| \leq \ell$, $\mathsf{R}(T) = R$ and $T[\mathsf{R}(T)]$ is a path with endpoints ρ_1 and ρ_k. Furthermore, based on the previous claims, we have that, for a given set $R \subseteq K \setminus W$, with $2 \leq |R| = k \leq r$, and a given unordered pair $\{\rho_1, \rho_k\} \subseteq R$ of distinct vertices, we can obtain a strict connection tree T of G for W such that $|\mathsf{L}(T)| \leq \ell$, $\mathsf{R}(T) = R$ and $T[\mathsf{R}(T)]$ is a path with endpoints ρ_1 and ρ_k, or conclude that such a tree T does not exist, in time polynomial in n. Therefore, since all unordered pair $\{\rho_1, \rho_k\} \subseteq R$ of distinct vertices can be enumerated in time $\mathcal{O}(n^2)$ and all subsets $R \subseteq K \setminus W$, with $2 \leq |R| \leq r$, can be enumerated in time $n^{\mathcal{O}(r)}$, the total running time of the algorithm is $n^{\mathcal{O}(r)}$. □

Next lemma immediately follows from Fact 2.1 and Proposition 2.1.

Theorem 2.2. *For every $r \geq 0$, S-TCP can be solved in time $n^{\mathcal{O}(r)}$ on split graphs.*

Now, we are finally able to prove Theorem 2.1. In what follows, we show that, for $r \geq 1$, TCP is polynomial-time reducible to S-TCP when both are restricted to split graphs. As an immediate consequence, we obtain that TCP on split graphs can be solved in time $n^{\mathcal{O}(r)}$ for every $r \geq 1$. Consider the following auxiliary lemmas.

Lemma 2.3. *Let $G\langle K, S\rangle$ be a split graph and $W \subseteq V(G)$, with $|W| \geq 3$. If $W \cap K = \emptyset$ and there is a connection tree T of G for W such that $\mathsf{R}(T) = \emptyset$, then there is a connection tree T' of G for W such that $\mathsf{L}(T') \subseteq |\mathsf{L}(T)|$ and $|\mathsf{R}(T')| = 1$.*

Proof. Since $|W| \geq 3$ and $\mathsf{R}(T) = \emptyset$, there exists a terminal vertex $w \in W$ whose degree in T is at least 2. Then, let u and u' be two distinct neighbours of w in T. Since $W \cap K = \emptyset$, $u, u' \in \mathsf{L}(T) \cap K$. Let T' be the graph obtained from T by removing the edge wu' and adding the edge uu'. Clearly, T' is a connection tree of G for W such that $\mathsf{L}(T') = \mathsf{L}(T) \setminus \{u\}$ and $\mathsf{R}(T') = \{u\}$. □

Lemma 2.4. *Let $G\langle K, S\rangle$ be a split graph and $W \subseteq V(G)$ be a non-empty set. Suppose that G admits a connection tree T for W. There exists a connection tree T' of G for W, with $\mathsf{L}(T') \subseteq \mathsf{L}(T)$ and $|\mathsf{R}(T')| \leq |\mathsf{R}(T)|$, that simultaneously satisfies the following conditions:*

(i) $\mathsf{L}(T') \cup \mathsf{R}(T') \subseteq K$;

(ii) If $(\mathsf{R}(T) \cup W) \cap K \neq \emptyset$, *then every vertex in* $W \cap S$ *is a leaf of* T'.

Proof. (i) Suppose that $(\mathsf{L}(T) \cup \mathsf{R}(T)) \cap S \neq \emptyset$. Then, there exists a vertex $u \in V(T) \cap K$. Let T' be the graph obtained from T as follows:

- Remove all vertices belonging to $(\mathsf{L}(T) \cup \mathsf{R}(T)) \cap S$ and their incident edges;
- For each $u' \in \mathsf{L}(T) \cap S$, add the edge vv', where $N_T(u') = \{v, v'\}$;
- For each $u' \in N_T(\mathsf{R}(T) \cap S)$, add the edge uu'.

Clearly, T' is a connection tree of G for W such that $\mathsf{L}(T') \subseteq K$ and $\mathsf{R}(T') \subseteq K$. Moreover, note that $\mathsf{L}(T') \subseteq \mathsf{L}(T) \setminus S$, $\mathsf{R}(T') = \mathsf{R}(T)$ if $\mathsf{R}(T) \cap S = \emptyset$, and $\mathsf{R}(T') \subseteq (\mathsf{R}(T) \cup \{u\}) \setminus S$ otherwise.

(ii) Suppose that $W \cap S \neq \emptyset$ and that there exists a vertex $u \in (\mathsf{R}(T) \cup W) \cap K$. In this case, every vertex $w \in W \cap S$ has at least one neighbour in T, say $\alpha(w)$. Then, let T' be the graph obtained from T as follows:

- For each $w \in W \cap S$, remove all edges of T that are incident to w except for $w\alpha(w)$; additionally, for each $v \in N_T(w)$, add the edge uv if there is no path between u and v in $T[K]$.

Clearly, T' is a connection tree of G for W such that every vertex in $W \cap S$ is a leaf of T'. Furthermore, one can verify that $\mathsf{L}(T') = \mathsf{L}(T)$ and $\mathsf{R}(T') = \mathsf{R}(T)$. □

Next, we present our polynomial-time reduction to S-TCP.

Construction 2.1 (Reduction from TCP to S-TCP on split graphs)**.** Let $G\langle K, S\rangle$ be a split graph and $I = (G, W, \ell, r)$ be an instance of TCP. If $W \cap K = \emptyset$, then we define our reduction instance of S-TCP as simply $g(I) = I$. Otherwise, let $\rho \in W \cap K$ and consider the graph G' obtained from G as follows (see Figure 2.4):

- Add all vertices and all edges of G;
- For each terminal vertex $w \in W \cap S \setminus N_G(\rho)$ that is neighbour in G of a vertex in $W \cap K$, add the edge ρw;
- Add three new vertices w'_1, w'_2 and w'_3, and make them adjacent to ρ.

Note that G' is a split graph, and that $K \cup S'$ is a partition of $V(G')$ into a clique and a stable set, where $S' = S \cup \{w'_1, w'_2, w'_3\}$. We then define our reduction instance of S-TCP as $g(I) = (G', W', \ell, r+1)$, where $W' = (W \setminus \{\rho\}) \cup \{w'_1, w'_2, w'_3\}$.

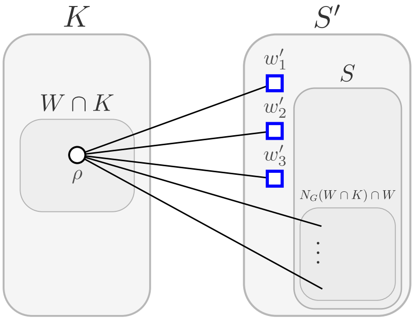


Figure 2.4: Split graph $G'\langle K, S'\rangle$ of the instance $g(I)$ of S-TCP described in Construction 2.1, obtained from a split graph $G\langle K, S\rangle$ of an instance I of TCP, with $W \cap K \neq \emptyset$.

The following lemma, along with Theorem 2.2, finishes the proof that TCP on split graphs can be solved in time $n^{\mathcal{O}(r)}$ for each $r \geq 1$.

Lemma 2.5. *Let $G\langle K, S\rangle$ be a split graph and $I = (G, W, \ell, r)$ be an instance of TCP such that $|W| \geq 3$. Also, let $g(I)$ be the instance of S-TCP obtained from I, as described in Construction 2.1. If $r \geq 1$ or $W \cap K \neq \emptyset$, then I is a* yes*-instance of TCP if and only if $g(I)$ is a* yes*-instance of S-TCP.*

Proof. First, suppose that I is a yes-instance of TCP. Then, there exists a connection tree T of G for W such that $|\mathsf{L}(T)| \leq \ell$ and $|\mathsf{R}(T)| \leq r$. Since $r \geq 1$, by Lemma 2.3, we can assume that $\mathsf{R}(T) \neq \emptyset$ or $W \cap K \neq \emptyset$. Furthermore, by Lemma 2.4, we can assume that every vertex in $W \cap S$ is a leaf of T. This implies $W \setminus \text{leaves}(T) \subseteq K$. If $W \cap K = \emptyset$, then $\text{leaves}(T) = W$ and, therefore, we immediately obtain that $g(I) = I$ is a yes-instance of S-TCP. Thus, suppose that $W \cap K \neq \emptyset$. Additionally, by Lemma 2.4, assume that $\mathsf{L}(T) \subseteq K$ and $\mathsf{R}(T) \subseteq K$. Note that every vertex in $V(T) \cap S$ is a leaf of T. Since T is a tree and $\rho \in V(T)$, for each vertex $w \in W \cap K \setminus \{\rho\}$, there exists a single path between ρ and w in T and a single vertex in this path, say $\alpha(w)$, that belongs to $N_T(w) \cap K$. Thus, let T' be the graph obtained from T as follows:

- For each $w \in W \cap K \setminus \{\rho\}$ and each $w' \in N_T(w) \setminus \{\alpha(w)\}$, remove the edge ww' and add the edge $\rho w'$;
- For each $i \in \{1, 2, 3\}$, add the vertex w'_i and the edge $\rho w'_i$.

One can verify that T' is a connection tree of G' for W', such that leaves$(T') = W'$, $\mathsf{L}(T') = \mathsf{L}(T)$ and $\mathsf{R}(T) = \mathsf{R}(T') \cup \{\rho\}$.

Conversely, suppose that $g(I)$ is a yes-instance of S-TCP. If $W \cap K = \emptyset$, then $g(I) = I$ and, therefore, I is a yes-instance of TCP. Thus, suppose that $W \cap K \neq \emptyset$, and let T' be a connection tree of G' for W', such that leaves$(T') = W'$, $|\mathsf{L}(T')| \leq \ell$ and $|\mathsf{R}(T')| \leq r + 1$. Since the only neighbour of the terminal vertices w'_1, w'_2 and w'_3 in G' is the vertex ρ, we have that ρ necessarily belongs to T' and, besides that, is a router of T'. Moreover, by construction of G', if a vertex w is a neighbour of ρ in T' but is not a neighbour of ρ in G, then $w \in W \cap S$ and there exists a vertex in $W \cap K$, say $\beta(w)$, which is a neighbour of w in G. Then, let T be the graph obtained from T' as follows:

- Remove the vertices w'_1, w'_2 and w'_3 and their incident edges;
- For each $w \in N_{T'}(\rho) \setminus N_G(\rho)$, remove the edge ρw and add the edge $\beta(w)w$.

One can verify that T is a connection tree of G for W, such that $\mathsf{L}(T) = \mathsf{L}(T')$ and $\mathsf{R}(T) = \mathsf{R}(T') \setminus \{\rho\}$. □

Now, through a parameterized reduction from Set cover, we prove the lower bound for the time complexity of TCP on split graphs, completing therefore the proof of Theorem 2.1. Below, we formally define Set cover, which is a classical problem, known to be $\mathsf{W}[2]$-hard when parameterized by the solution size *cf.* [31].

Set cover

Input: A universe set U, a collection $\mathcal{F} = \{F_1, \ldots, F_q\}$ of non-empty sets over U, and a positive integer k.

Question: Is there a subset $\mathcal{F}' \subseteq \mathcal{F}$, with $|\mathcal{F}'| \leq k$, such that $\mathcal{F}'$ covers all elements of U, *i.e.* $\bigcup_{F \in \mathcal{F}'} \{x \colon x \in F\} = U$?

Lemma 2.6. *For any computable functions g and h, TCP cannot be solved in time $g(r) \cdot n^{h(\ell)}$, unless $\mathsf{FPT} = \mathsf{W}[2]$, and cannot be solved in time $g(r) \cdot n^{o(r)}$, unless ETH fails.*

Proof. Let $I = (U, \mathcal{F}, k)$ be an instance of Set cover. We construct an instance $f(I, \ell) = (G, W, \ell, r = k)$ of S-TCP as follows:

- For each $x_i \in U$, create the vertex w_i;
- For each $F_j \in \mathcal{F}$, create the vertex v_{F_i}; let $K_{\mathcal{F}} = \{v_{F_i} : F_i \in \mathcal{F}\}$;
- For each $x_i \in U$ and each $F_j \in \mathcal{F}$ with $x_i \in F_j$, add the edge $v_{F_j} w_i$;
- For each $i \in [\ell]$, create the vertices u_i and w'_i, and add the edge $u_i w'_i$; let $L = \{u_1, \ldots, u_\ell\}$;
- Create the vertices w_a and w_b and, for each $F_i \in \mathcal{F}$, add the edges $w_a v_{F_i}$ and $w_b v_{F_i}$;
- Define $W = \{w_a, w_b\} \cup \{w'_i : i \in [\ell]\} \cup \{w_i : x_i \in U\}$;
- Finally, make all the vertices in $L \cup K_{\mathcal{F}}$ adjacent to all vertices in W.

One can readily verify that the construction described above yields a split graph $G\langle K, S\rangle$, where $K = L \cup K_{\mathcal{F}}$ and $S = W$. Now, we prove that I is a yes-instance of Set cover if and only if $f(I, \ell)$ is a yes-instance of TCP.

First, suppose that I is a yes-instance of Set cover, and let $\mathcal{F}' = \{F'_1, \ldots, F'_z\}$ be a subset of $\mathcal{F}$ such that $\bigcup_{F' \in \mathcal{F}'} F' = U$ and $z \leq k$. Assume without loss of generality that $\mathcal{F}'$ is minimal with respect to the property of covering all elements of U, *i.e.* for any set $F' \in \mathcal{F}'$, $\mathcal{F}' \setminus \{F'\}$ does not cover all elements of U. Then, let T be the connection tree of G for W defined as follows:

- For each $i \in [\ell]$, add the vertices u_i and w'_i to T along with the edges $u_i w'_i$;
- For each $F'_j \in \mathcal{F}'$, add the vertex $v_{F'_j}$ to T;
- For each $i \in [z-1]$, add the edge $v_{F'_i} v_{F'_{i+1}}$ to T;
- Additionally, for each $i \in [\ell]$, add the edge $u_i v_{F'_1}$ to T; also add the edges $w_a v_{F'_1}$ and $w_b v_{F'_z}$ to T;
- Finally, for each $x_i \in U$, add the vertex w_i to T along with the edge $v_{F'_j} w_i$, where j is the minimum index in $[z]$ such that $x_i \in F'_j$ and $F'_j \in \mathcal{F}'$.

It is easy to verify that T is a connection tree of G for W. Now, we prove that $|\mathsf{L}(T)| \leq \ell$ and $|\mathsf{R}(T)| \leq r$. First, note that the vertices $u_1, \ldots, u_\ell$ have degree exactly 2 in T. Thus, $\mathsf{L}(T) \supseteq L$. On the other hand, it follows from the minimality of $\mathcal{F}'$ that, for every set $F'_j \in \mathcal{F}$, the vertex $v_{F'_j}$ is adjacent to at least one terminal $w_i \in W$, since F'_j covers at least one element $x_i \in U$ which is not covered by any other set in $\mathcal{F}'$. Consequently, every vertex $v_{F_j'}$ has degree at least 3 in T. Thus, $\mathsf{L}(T) = L$ and $\mathsf{R}(T) = \{v_{F_j'} \colon F_j' \in \mathcal{F}'\}$, which implies $|\mathsf{L}(T)| \leq \ell$ and $|\mathsf{R}(T)| = z \leq k = r$.

Conversely, suppose that G admits a connection tree T for W such that $|\mathsf{L}(T)| \leq \ell$ and $|\mathsf{R}(T)| \leq r = k$. Note that, for every $i \in [\ell]$, the path in T between the terminal w_i' and any other terminal belonging to W necessarily contains the vertex $u_i \in L$. Hence, $V(T) \supseteq L$. Furthermore, we recall that $W \supset W_U$ is a stable set of G and that $N_G(W_U) \subseteq K_{\mathcal{F}}$, where $W_U = \{w_i \colon x_i \in U\}$. Thus, for every $w_i \in W_U$, there must exist a vertex $v_{F_j} \in K_{\mathcal{F}}$ such that $v_{F_j} w_i \in E(T)$, otherwise w_i would not belong to T. However, T contains at most $\ell + r$ non-terminal vertices. Thus, since $V(T) \supseteq L$ and $|L| = \ell$, there are at most r vertices belonging to $K_{\mathcal{F}}$ in T, *i.e.* $|V(T) \cap K_{\mathcal{F}}| \leq r = k$. Then, $\mathcal{F}' = \{F_j \colon v_{F_j} \in V(T) \cap K_{\mathcal{F}}\}$ is a subset of $\mathcal{F}$ such that $|\mathcal{F}'| \leq k$. Finally, it follows from the fact that $W \subseteq V(T)$ that $\bigcup_{F' \in \mathcal{F}'} F' = U$.

Based on the argumentation above, we have that, for every $\ell \geq 0$, TCP parameterized by r is $\mathsf{W}[2]$-hard. Thus, assuming $\mathsf{FPT} \neq \mathsf{W}[2]$, TCP cannot be solved in time $g(r) \cdot n^{h(\ell)}$. To prove that that TCP cannot be solved in time $g(r) \cdot n^{o(r)}$, unless ETH fails, it suffices to observe that in our reduction $r = k$ and that the instance $f(I, \ell)$ can actually be constructed in time polynomial in I. This, along with the fact that, under ETH, SET COVER cannot be solved in time $g(k) \cdot n^{o(k)}$ *cf.* [31] we obtain the desired result (for more details, we refer to Theorem 7 of Appendix B). □

The following corollary immediately follows from Theorem 2.2 and Lemma 2.6, since the proof of Lemma 2.6 remains exactly the same if the sought connection tree is required to be strict.

Corollary 2.1. *For every $r \geq 0$, S-TCP on split graphs can be solved in time $n^{\mathcal{O}(r)}$ but, assuming $\mathsf{FPT} \neq \mathsf{W}[2]$, cannot be solved in time $g(r) \cdot n^{h(\ell)}$ and, assuming ETH, cannot be solved in time $g(r) \cdot n^{o(r)}$, for any computable functions g and h.*

2.2.2 Rooted Directed Path Graphs

Given a collection $\mathcal{F}$ over a universe set U, a graph G is called an *intersection graph* of the *intersection model* $(U, \mathcal{F})$ if there exists a bijection $\alpha \colon V(G) \to \mathcal{F}$ such that, for every two distinct vertices $u, v \in V(G)$, $uv \in E(G)$ if and only if $\alpha(u) \cap \alpha(v) \neq \emptyset$.

Chordal graphs is one of the most import graph classes, and it is defined as the class of graphs without induced cycles of length greater than 3. Alternatively, Gavril [69] characterized chordal graphs as the class of intersection graphs of subtrees of a tree. In other words, a graph G is chordal if and only if there is a tree T, called *characteristic tree*, and a collection $\{T_u\}_{u \in V(G)}$ of subtrees of T, such that $uv \in E(G)$ if and only if $V(T_u) \cap V(T_v) \neq \emptyset$ for every two distinct vertices $u, v \in V(G)$.

Nested subclasses of chordal graphs can be defined by imposing further constraints on either the characteristic tree or on the collection of subtrees of the intersection model. Indeed, *undirected path graphs* are the intersection graphs of undirected paths of a tree [71]; *directed path graphs* are the intersection graphs of directed paths of a directed tree [98]; *rooted directed path graphs* are the intersection graphs of directed paths paths of a rooted directed tree [70]; and *interval graphs* are the intersection graphs of subpaths of a path [13].

We prove that TCP remains NP-complete on rooted directed path graphs:

Theorem 2.3. *For each $r \geq 0$, TCP remains* NP*-complete when restricted to rooted directed path graphs.*

This results contrasts with the complexity of Steiner tree, which is known to be polynomial-time solvable on *strongly chordal graphs*, a superclass of rooted directed path graphs [16, 59]. In order to prove Theorem 2.3, we provide a polynomial-time reduction from the Hamiltonian path problem, which was shown to be NP-complete on rooted directed path graphs by Panda and Pradhan [103]. The Hamiltonian path problem is formally defined below.

Hamiltonian path

Input: A connected graph G.

Question: Is there a *Hamiltonian path* in G, *i.e.* a path that contains all vertices of G?

Given a graph G, we let $\mathcal{C}(G)$ denote *clique set* of G, *i.e.* the set of all maximal cliques of G. Additionally, for each vertex $u \in V(G)$, we denote by $\mathcal{C}_G(u)$ the set of all maximal cliques of G that contain u. The following theorem, due to Gavril [70], establishes an intersection model characterization for rooted directed graphs in terms of a characteristic tree whose vertex set is the clique set of G.

Theorem 2.4 ([70, 98])**.** *A graph G is a rooted directed path graph if and only if there exists a rooted directed tree T, with vertex set $\mathcal{C}(G)$, such that, for every vertex $u \in V(G)$, the subgraph of T induced by $\mathcal{C}_G(u)$ is a directed path of T.*

Next lemma presents some important properties of the class of rooted directed path graphs that are used in our reduction.

Lemma 2.7. *The class of rooted directed path graphs is closed under the following operations:*

(i) For any pair of true twin vertices u and v, adding a new vertex v' and adding the edges uv' and vv'.

(ii) Adding a pendant vertex;

(iii) Adding true twin vertices;

Proof. Let G be a rooted directed path graph and $\mathcal{T} = (T, \{T_u\}_{u \in V(G)})$ be an intersection model of G, such that T is the *clique tree* of G, *i.e.* T is a rooted directed tree satisfying the conditions of Theorem 2.4.

(i) and (ii) Let $X \subseteq V(G)$ be a set of vertices such that, for every pair $u, v \in X$, $N_G[u] = N_G[v]$. Also, let G' be a graph obtained from G by adding a new vertex v' and making it adjacent to every vertex in X, *i.e.* $N_{G'}(v') = X$. In what follows, we prove that G' is a rooted directed path graph. Note that, if X consists of a single vertex, then G' is the graph obtained from G by adding a pendant vertex. Moreover, note that, for every two vertices $u, v \in X$, we have that $T_u = T_v$, since u and v are true twins in G, and therefore $\mathcal{C}_G(u) = \mathcal{C}_G(v)$. Thus, let v be an arbitrary vertex in X. Since $T_v = T[\mathcal{C}_G(v)]$ is a directed path, there exists exactly one vertex in T_v, say $\mathbf{v}$, of out-degree 0. Then, consider the intersection model $\mathcal{T}' = (T', \{T'_u\}_{u \in V(G')})$ obtained from $\mathcal{T}$ as follows:

- Define T' exactly as T, except for adding a new vertex $\mathbf{v}'$, corresponding to the maximal clique $X \cup \{v'\}$ of G', and by adding the edge from $\mathbf{v}$ to $\mathbf{v}'$;
- For each $u \in X$, define T'_u exactly as T_u, except for adding the vertex $\mathbf{v}'$ and the edge from $\mathbf{v}$ to $\mathbf{v}'$;
- Define $T'_{v'}$ as the tree with $V(T'_{v'}) = \{\mathbf{v}'\}$ and $E(T'_{v'}) = \{(\mathbf{v}, \mathbf{v}')\}$;
- Finally, for each $u \in V(G) \setminus X$, define T'_u exactly as T_u.

One can readily verify that $\mathcal{T}'$ is an intersection model of G' satisfying the conditions of Theorem 2.4. Therefore, G' is a rooted directed path graph.

(iii) Let G' be the graph obtained from G by adding, for some vertex $v \in V(G)$, a true twin v' of v. Then, let $\mathcal{T}' = (T', \{T'_u\}_{u \in V(G')})$ be the intersection model obtained from $\mathcal{T}$ as follows:

- Define T' exactly as T and, for each $u \in V(G)$, define T'_u exactly as T_u;
- Finally, define $T_{v'}$ exactly as T_v.

Since v' is a true twin of G', we have that $K \subseteq V(G)$ is a maximal clique of G if and only if: either K is maximal clique of G' and $v \notin K$, or $K \cup \{v'\}$ is maximal clique of G' and $v \in K$. Thus, one can verify that $\mathcal{T}'$ is an intersection model of G' satisfying the conditions of Theorem 2.4. □

Construction 2.2 (Gadget H_r and Terminal Set W_r)**.** Let r be a positive integer. We define the gadget H_r as the graph (see Figure 2.5) with vertex set

$$V(H_r) = \{\rho_1, \ldots, \rho_r\} \cup \{x_1^1, x_1^2\} \cup \{x_i \colon i \in \{2, \ldots, r\}\}, \text{ and edge set}$$

$$E(H_r) = \{\rho_i \rho_{i+1} \colon i \in \{1, \ldots, r-1\}\} \cup \{x_1^1 \rho_1, x_1^2 \rho_1\} \cup \{x_i \rho_i \colon i \in \{2, \ldots, r\}\}.$$

Moreover, we let $W_r = \{x_1^1, x_1^2\} \cup \{x_2, \ldots, x_r\}$ be the terminal set of H_r.

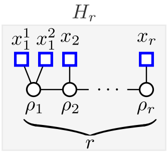


Figure 2.5: Gadget H_r for $r \geq 1$, described in Construction 2.2.

Construction 2.3 (Reduction from HAMILTONIAN PATH to TCP)**.** Let G be a graph, with vertex set $V(G) = \{u_1, \ldots, u_n\}$, and r be a non-negative integer. We let G' be the graph obtained from G and r as follows (see Figure 2.6):

- Add all vertices and all edges of G to G';
- If $r \geq 1$, create the gadget H_r and define W_r as described in Construction 2.2, besides adding the edge $\rho_r w_1$; otherwise, if $r = 0$, define $W_r = \emptyset$;
- For each vertex $u_i \in V(G)$, add a true twin u'_i of u_i, in such a way that in the resulting graph G', after performing all operations, u'_i of u_i are still true twins, *i.e.* $N_{G'}[u'_i] = N_{G'}[u_i]$;
- For each vertex $u_i \in V(G)$, add a new vertex w_i and add the edges $u_i w_i$ and $u'_i w_i$, where u'_i denotes the true twin of u_i added in the last step (note that this operation does not affect the fact of u_i being a true twin of u'_i).

We then define our reduction instance of TCP as $g(G, r) = (G', W, \ell, r)$, where $W = \{w_1, \ldots, w_n\} \cup W_r$ and $\ell = 2n - 2$.

We remark that, the graph G' described in Construction 2.3 is similar to the one constructed in [52] to prove the NP-completeness of TCP on general graphs for fixed $r \geq 0$. The main difference is the fact that, in construction of [52], it is added a

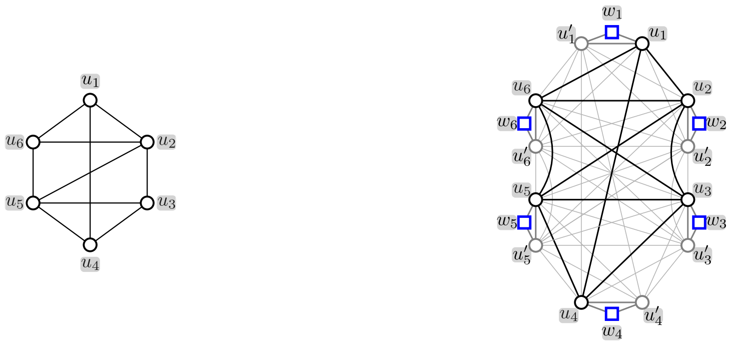


Figure 2.6: A graph G and the graph G' obtained from G (and $r = 0$) as described in Construction 2.3.

false twin of u_i instead of a true twin of u_i for each $u_i \in V(G)$. However, this makes the original graph not be chordal, and therefore not a rooted directed path graph, even if the input graph is a rooted directed path graph. For instance, a cycle C_3 of length 3 is trivially a rooted directed path graph, but the graph resulting from adding a false twin (preserving previously existent false twins) for each vertex of C_3 is not chordal, since it contains an induced cycle of length 4. The following lemma, which states that, whenever the input graph is rooted directed path, our constructed graph is a rooted directed path as well, immediately follows from Lemma 2.7.

Lemma 2.8. *Let G be a graph and r be a non-negative integer. Also, let G' be the graph of the instance $g(G, r)$ of TCP obtained from G and r, as described in Construction 2.3. If G is a rooted directed path graph, then so is G'.*

The following Lemma concludes the proof of Theorem 2.3.

Lemma 2.9. *Let G be a graph and r be a non-negative integer. Also, let $g(G, r)$ be the instance of TCP obtained from G and r, as described in Construction 2.3. Then, G admits a Hamiltonian path if and only if $g(G, r)$ is a yes-instance of TCP.*

Proof. Assume that $V(G) = \{u_1, \ldots, u_n\}$ and that $g(G, r) = (G', W, \ell, r)$. Additionally, for simplicity, consider $W_r = V(H_r) = E(H_r) = \emptyset$ if $r = 0$.

First, suppose that there exists in G a Hamiltonian path $(u_{j_1}, \ldots, u_{j_n})$. Then, let T be the graph with vertex set

$$V(T) = V(H_r) \cup V(P) \cup \{w_{j_1}, u'_{j_1}, u_{j_n}, w_{j_n}\} \cup \{u_{j_i}, w_{j_i}, u'_{j_i} : i \in \{2, \ldots, n-1\}\}$$

and edge set

$$\begin{aligned} E(T) = {} & E(H_r) \cup \{\rho_r w_1, w_1 u'_1\} \\ & \cup \{u'_{j_{i-1}} u_{j_i}, u_{j_i} w_{j_i}, w_{j_i} u'_{j_i} : i \in \{2, \ldots, n-1\}\} \cup \{u'_{j_{n-1}} u_{j_n}, u_{j_n} w_{j_n}\}, \end{aligned}$$

where u'_{j_i} denotes the true twin of u_{j_i} added in the construction of G'. Note that T is a connection tree of G' for W with $\mathsf{L}(T) = \{u'_{j_1}, u_{j_n}\} \cup \{u_{j_2}, u'_{j_2}, \ldots, u_{j_{n-1}}, u'_{j_{n-1}}\}$ and $\mathsf{R}(T) = \{\rho_1, \ldots, \rho_r\}$. Therefore, $g(G, r)$ is a yes-instance of TCP.

Conversely, suppose that $g(G, r)$ is a yes-instance of TCP. Let T be a connection tree of G' for W such that $|\mathsf{L}(T)| \leq 2n-2$ and $|\mathsf{R}(T)| \leq r$. We remark that ρ_1 is the only neighbour of the terminal vertices $x_1^1, x_1^2 \in W_r$ and, for each $i \in \{2, \ldots, r\}$, ρ_i is the only neighbour of the terminal vertex $x_i \in W_r$. As a result, T must contain all the vertices $\rho_1, \ldots, \rho_r$. More specifically, such vertices must be routers of T. This implies that $T' = T - H_r$ cannot contain any router, and all non-terminal vertices of T' must be linkers. Hence, T' is a path, since, by construction of G', w_i has degree at most 2 in T' for every $i \in [n]$. Then, let $P' = (w_{j_1}, \ldots, w_{j_n})$ be a sequence of distinct vertices such that, for each $i \in [n-1]$, the path in T' between w_{j_i} and $w_{j_{i+1}}$ does not contain any other terminal vertex. Note that, since $|\mathsf{L}(T)| \leq \ell = 2n-2$, every path in T' between any two consecutive vertices w_{j_i} and $w_{j_{i+1}}$ in P' must be of one of the following forms: $(w_{j_i}, u'_{j_i}, u'_{j_{i+1}}, w_{j_{i+1}})$, $(w_{j_i}, u'_{j_i}, u_{j_{i+1}}, w_{j_{i+1}})$, $(w_{j_i}, u_{j_i}, u_{j_{i+1}}, w_{j_{i+1}})$, or $(w_{j_i}, u_{j_i}, u'_{j_{i+1}}, w_{j_{i+1}})$. As a result, it follows from the construction of G' that, for each $i \in [n-1]$, u_{j_i} and $u_{j_{i+1}}$ are adjacent in G. Therefore, $(u_{j_1}, \ldots, u_{j_n})$ is a Hamiltonian path of G. □

Corollary 2.2. *S-TCP remains* NP*-complete when restricted to rooted directed path graphs.*

Proof. Let G be a graph and $f(G, r) = (G', W, \ell, r)$ be the instance of TCP obtained from G and $r = 0$ as described in Construction 2.3. Then, let G'' be the graph obtained from G' by adding two new vertices w_i^1 and w_i^2 and the edges $w_i w_i^1$ and $w_i w_i^2$, for each $w_i \in W$. We remark that, if G is a rooted directed path graph, then so is G'', since by Lemma 2.7 the class of rooted directed path graphs is closed under the operation of adding pendant vertices. Consider $W'' = \{w_i^1, w_i^2 \colon w_i \in W\}$. Note that every terminal vertex belonging to W'' has degree exactly 1 in G''. Thus, one can verify that, for $|W| > 1$, $f(G, r)$ is a yes-instance of TCP if and only if $(G'', W'', \ell, r + |W|)$ is a yes-instance of S-TCP *cf.* [39]. Therefore, it follows from Lemma 2.9 that S-TCP is NP-complete on rooted directed path graphs. □

2.3 Graphs of Bounded Clique-Width

In this section, we prove that TCP parameterized by the clique-width of the input graph is W[1]-hard. Similarly to the results presented in Section 2.2, this contrasts with the complexity of Steiner tree, which is known to be in FPT when parameterized by clique-width [4]. On the other hand, agreeing with the complexity of Steiner tree, we prove that TCP is linear-time solvable on cographs, which are precisely the graphs of clique-width at most 2.

The notion of clique-width was introduced by Courcelle, Engelfriet and Rozenberg [29], and it is one of the most studied graph parameters. Next, we present the definition of this notion *cf.* [60, 62].

Let k be a positive integer. A graph is called a *k-graph* if its vertices are labelled with integers in $[k]$. An *initial k-graph* is a k-labelled graph on a single vertex. The *clique-width* of a graph G, denoted by $\mathsf{cwd}(G)$, is the smallest positive integer k such that G can be constructed by repeated application of the following four operations:

1. *introducing* (denoted by $\mathsf{int}(u, i)$): construction of an initial k-graph, whose single vertex u is labelled by an integer $i \in [k]$ and has not been introduced yet;

2. *disjoint union* (here, denoted by $\oplus$);

3. *relabelling* (denoted by $\mathsf{rel}_{i,j}$): changing all labels i to j, for $i, j \in [k]$;

4. *join* (denoted by $\eta_{i,j}$): connecting all vertices labelled by i with all vertices labelled by j, for $i, j \in [k]$, $i \neq j$.

A construction of a graph G using the operations (1)-(4) described above can be represented by an algebraic term, called *cwd-expression* defining G, composed of int, $\oplus$, $\mathsf{rel}_{i,j}$, and $\eta_{i,j}$ *cf.* [60], where i and j are distinct positive integers. Note that cwd-expressions define a tree language, where each expression can be represented by a rooted tree T *cf.* [62], where each $\mathsf{int}(u, i)$ of the expression is associated with a leaf of T, and each vertex of G is introduced exactly once. A *k-expression* is a cwd-expression that contains at most k distinct labels *cf.* [60]. Thus, a graph G has clique-width at most k if and only if there exists a k-expression defining G.

2.3.1 Parameterization by clique-width

Now, we prove the following theorem.

Theorem 2.5. *For each $r \geq 0$, TCP parameterized by clique-width is* $\mathsf{W}[1]$*-hard.*

More specifically, we show that, if a graph G has clique-width at most k, then the graph G' obtained from G as described in Construction 2.3 has clique-width at most $k + 1$. This, along with Lemma 2.9 and the fact that HAMILTONIAN PATH is $\mathsf{W}[1]$-hard parameterized by clique-width [62], implies the $\mathsf{W}[1]$-hardness of TCP.

The following lemma is a well-known fact, and it can be immediately verified by an inductive argument on the number of vertices of the tree.

Lemma 2.10. *Every tree has clique-width at most* 3*. Moreover, if T is a tree and u is a leaf of T, then there exists a* 3*-expression defining a construction of T in which at the root all vertices but u have the same label.*

Lemma 2.11. *Let G be a graph. For each $r \geq 0$, if $\mathsf{cwd}(G) = k$ for some $k \geq 2$, then $\mathsf{cwd}(G') \leq k+1$, where G' denotes the graph obtained from G and r as described in Construction 2.3.*

Proof. Assume that $V(G) = \{u_1, \ldots, u_n\}$ and $\mathsf{cwd}(G) = k$. Then, let γ_G be a k-expression defining G. Also, let H' be the subgraph of G' induced by the vertex set $V(H_r) \cup \{w_1\}$. Note that H' is a tree and w_1 is a leaf in H'. Thus, by Lemma 2.10, there exists a construction (3-expression) of a vertex-labelled copy of H' (for short γ'_H) in which all vertices but w_1 have the same label. Assume without loss of generality that w_1 is labelled by 1 at the root of $\gamma_{H'}$, and that all the other vertices of $\gamma_{H'}$ are labelled by 2. Moreover, since $k \geq 2$, assume without loss of generality that u_1 is introduced in γ_G with a label different from 1. In what follows, we show that we can obtain from γ_G and $\gamma_{H'}$ a $(k+1)$-expression $\gamma_{G'}$ defining our constructed graph G'. Consider $a = k+1$. We recall that each vertex $u_i \in V(G)$ has a true twin u'_i in G', and that $N_{G'}(w_i) = \{u_i, u'_i\}$ for every $i \in \{2, \ldots, k\}$, while $N_{G'}(w_1) = \{u_1, u'_1, \rho_r\}$. Then, let $\gamma_{G'}$ be the cwd-expression obtained from γ_G and γ'_H as described next.

- Let $\mathsf{int}(u_1, i)$ be the leaf term of u_1 in γ_G for some $i \in [k] \setminus \{1\}$. Replace the occurrence of $\mathsf{int}(u_1, i)$ in γ_G with (see Figure 2.7)

$$\mathsf{rel}_{1,a}(\eta_{i,1}(\mathsf{rel}_{a,i}\big(\eta_{i,a}(\mathsf{int}(u_1,i), \mathsf{int}(u'_1,a))\big), \mathsf{rel}_{2,a}(\gamma_{H'}))). \tag{2.1}$$

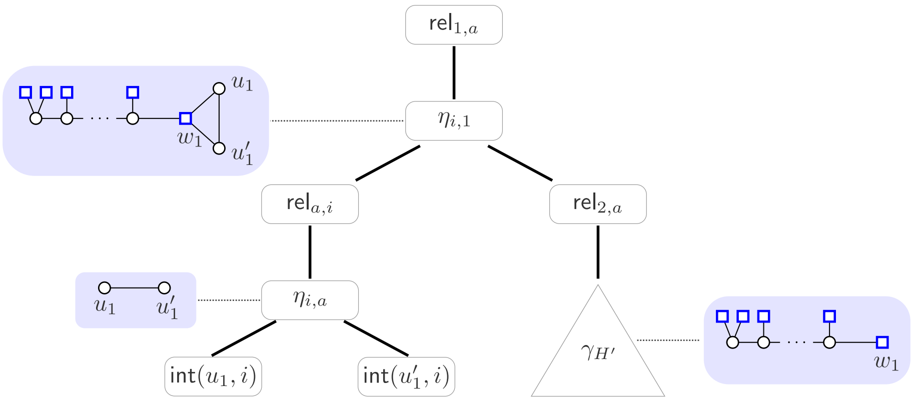


Figure 2.7: Cwd-expression described in (2.1). Beside some nodes, it is illustrated the graph resulting from the corresponding operation.

- For each $u_j \in V(G) \setminus \{u_1\}$, if $\mathsf{int}(u_j, i)$ is the leaf term of u_j in γ_G for some $i \in [k]$, then replace the occurrence of $\mathsf{int}(u_j, i)$ with (see Figure 2.8)

$$\eta_{i,a}(\mathsf{rel}_{a,i}\big(\eta_{i,a}(\mathsf{int}(u_j,i), \mathsf{int}(u'_j,a))\big), \mathsf{int}(w_j,a)). \tag{2.2}$$

We recall that, besides being represented by leaves, each vertex is introduced exactly once in a expression tree. Moreover, note that the operations described above consists in local replacements in the corresponding leaves of the expression tree associated to γ_G. Thus, one can verify that $\gamma_{G'}$ defines G'. Additionally, by construction, $\gamma_{G'}$ uses at most $k+1$ distinct labels, whenever $k \geq 2$. Therefore, we have that $\mathsf{cwd}(G') \leq k+1$. □

Corollary 2.3. *S-TCP parameterized by clique-width is* $\mathsf{W}[1]$*-hard.*

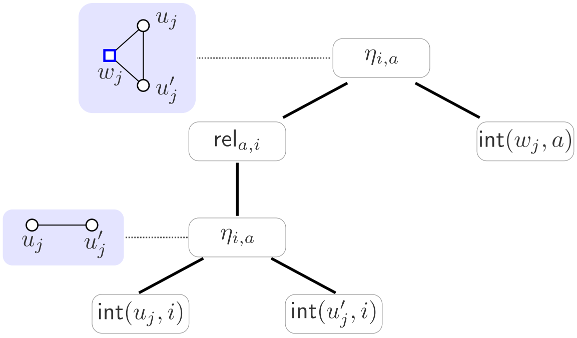


Figure 2.8: Cwd-expression described in (2.2). Beside some nodes, it is illustrated the graph resulting from the corresponding operation.

Proof. Let G be a graph and $f(G, r) = (G', W, \ell, r)$ be the instance of TCP obtained from G and $r = 0$ as described in Construction 2.3. Also, let $(G'', W'', \ell, r + |W|)$ be the instance of S-TCP obtained from $f(G, r)$ as described in Corollary 2.2. We show that, if $\mathsf{cwd}(G) = k$ for some $k \geq 2$, then $\mathsf{cwd}(G'') = \mathsf{cwd}(G') = k + 1$. Note that, since $r = 0$, the expression $\gamma_{H'}$ described in Lemma 2.11 is exactly equal to $\mathsf{int}(w_1, 1)$. Then, let $\gamma_{G''}$ be the cwd-expression obtained from $\gamma_{G'}$ as described next, where $a = k + 1$.

- Replace the occurrence of $\gamma_{H'}$ in (2.1) with

$$\mathsf{rel}_{a,2}(\eta_{1,a}(\mathsf{int}(w_1, 1), \oplus(\mathsf{int}(w_1^1, a), \mathsf{int}(w_1^2, a)))).$$

- For each $w_j \in W \setminus \{w_1\}$, replace expression (2.2) with

$$\mathsf{rel}_{i',a}(\eta_{i,a}(\mathsf{rel}_{a,i}\big(\eta_{i,a}(\mathsf{int}(u_j, i), \mathsf{int}(u'_j, a))\big), \gamma'')),$$

  where $\gamma'' = \eta_{i',a}(\mathsf{int}(w_j, i'), \oplus(\mathsf{int}(w_j^1, a), \mathsf{int}(w_j^2, a))$ and $i' \in [k] \setminus \{i\}$.

One can verify that $\gamma_{G''}$ defines G'' and that $\mathsf{cwd}(G'') = \mathsf{cwd}(G')$. □

2.3.2 Cographs

A *cograph*, or *complement-reducible graph*, is a graph that does not contain a path of length 4 as an induced subgraph. Alternatively, cographs are characterized by the following recursive definition, given by Corneil et al. [26]:

- A graph on a single vertex is a cograph;
- If $G_1, \ldots, G_k$ are cographs, then so is their *disjoint union* $G_1 \oplus \cdots \oplus G_k$;
- If $G_1, \ldots, G_k$ are cographs, then so is their *join* $G_1 \wedge \cdots \wedge G_k$.

We note that a graph is a cograph if and only if its clique-width is at most 2.

A key algorithmic property of cographs is the fact that, up to isomorphism, each cograph G can be uniquely represented by a rooted tree $\mathcal{T}_G$, called *cotree* [26], which can be seen as a specialization of a 2-expression defining G. The leaves of $\mathcal{T}_G$ correspond to the vertices of G, and each internal node u of $\mathcal{T}_G$ represents either the disjoint union or the join operation of the respective cographs induced by the leaves of the subtrees of $\mathcal{T}_G$ rooted at each child of u. Another important property is that, given a graph G, recognising G as a cograph, as well as obtaining its respective cotree (if any), can be performed in time linear in the number of vertices and the number of edges of G [27].

Let $I = (G, W, \ell, r)$ be an instance of TCP, where G is a cograph. Since TCP can be easily solved in linear-time if $|W| < 3$ or $G[W]$ is connected, we assume throughout this section that $|W| \geq 3$ and $G[W]$ is not connected. Moreover, we assume that G is connected and therefore is the join of $k \geq 2$ cographs $G_1, \ldots, G_k$.

We omit the proof of the following lemma and refer to Lemma 9 in Appendix D for more details.

Lemma 2.12. *Let G be a cograph that is the join of $k \geq 2$ cographs $G_1, \ldots, G_k$, and let $W \subseteq V(G)$ be a terminal set such that $|W| \geq 3$ and $G[W]$ is not connected. There exists a unique $i \in [k]$ such that $V(G_i) \cap W \neq \emptyset$. Moreover, G admits a connection tree for W that contains exactly one router and no linker.*

Considering the input graph G as the join of $k \geq 2$ cographs $G_1, \ldots, G_k$, it follows from Lemma 2.12 that TCP can be trivially solved if $r \geq 1$, or $V(G_i) \cap W \neq \emptyset$ and $V(G_j) \cap W \neq \emptyset$ for some $i, j \in [k]$, with $i \neq j$. Thus, we dedicate the remainder of this section to resolve the case in which $r = 0$ and there is a unique $i \in [k]$ such that $V(G_i) \cap W \neq \emptyset$.

Lemma 2.13. *Let G be a cograph and $W \subseteq V(G)$ be a non-empty terminal set. If T is a connection tree of G for W such that $\mathsf{R}(T) = \emptyset$ and $|\mathsf{L}(T)|$ is minimum, then $N_T(u) \subseteq W$ for each $u \in \mathsf{L}(T)$.*

Proof. For the sake of contradiction, suppose that $N_T(u) \not\subseteq W$ for some linker $u \in \mathsf{L}(T)$. Since $\mathsf{R}(T) = \emptyset$ and leaves$(T) \subseteq W$, u belongs to a path P of T between two terminal vertices $w, w' \in W$, such that $(V(P) \backslash \{w, w'\}) \cap W = \emptyset$. Thus, it follows from the assumption $N_T(u) \not\subseteq W$ that $|V(P)| \geq 4$. Since cographs do not contain paths of length 4 as induced subgraphs, there exists a path P' of G between w and w' such that $|V(P')| \leq 3$ and $V(P') \subseteq V(P)$. Then, let T' be the graph with vertex set $V(T') = (V(T) \backslash V(P)) \cup V(P')$ and edge set $E(T') = (E(T) \backslash E(P)) \cup E(P')$. Observe that T' is a connection tree of G for W such that $\mathsf{R}(T) = \emptyset$ and $\mathsf{L}(T') \subsetneq \mathsf{L}(T)$, which contradicts the minimality of $|\mathsf{L}(T)|$. □

For each graph G, we let $\mathsf{cc}(G)$ denote the set of connected components of G, and we let $o(G) = |\mathsf{cc}(G)|$ denote the number of connected components of G.

Corollary 2.4. *Let G be a cograph, $W \subseteq V(G)$ be a non-empty terminal set, and let T be a connection tree of G for W such that $\mathsf{R}(T) = \emptyset$. If $|\mathsf{L}(T)|$ is minimum, then $|\mathsf{L}(T)| = o(G[W]) - 1$.*

Proof. Since $\mathsf{R}(T) = \emptyset$ and each $u \in \mathsf{L}(T)$ connects at most two connected components, it follow that $|\mathsf{L}(T)| \geq o(G[W]) - 1$. On the other hand, from Lemma 2.13, $N_T(u) \subseteq W$ for each $u \in \mathsf{L}(T)$. In addition, we note that, if $u \in \mathsf{L}(T)$ and $N_T(u) = \{w, w'\}$, then w and w' belong to distinct connected components of $G[W]$, otherwise the path (w, u, w') of T could be replaced by a shortest path of $G[W]$ between w and w', yielding a connection tree T' of G for W such that $\mathsf{L}(T') \subsetneq \mathsf{L}(T)$. Therefore, $|\mathsf{L}(T)| \leq o(G[W]) - 1$. □

Corollary 2.4 establishes that, whenever a cograph G admits a connection tree for a non-empty terminal set $W \subseteq V(G)$ that does not contain routers, G admits a connection tree T for W such that $\mathsf{R}(T) = \emptyset$ and $\mathsf{L}(T) = o(G[W]) - 1$. More importantly, it establishes that $o(G[W]) - 1$ is the minimum possible number of linkers that such a tree T can have. Therefore, if $I = (G, W, \ell, r)$ is an instance of TCP such that G is a cograph and $r = 0$, then ℓ must be at least $o(G[W]) - 1$, otherwise I is certainly a no-instance of the problem.

A *connection forest* of a graph G for a non-empty terminal set W is a subgraph F of G such that F is a forest and $\bigcup_{T \in \mathsf{cc}(F)} \text{leaves}(T) \subseteq W \subseteq V(F)$. A connection forest F is said to be *routerless* if $\bigcup_{T \in \mathsf{cc}(F)} \mathsf{R}(T) = \emptyset$. For each graph G and each non-empty terminal $W \subseteq V(G)$, we let

$$\lambda[G, W] = \min\{o(F) \colon F \text{ is a routerless connection forest of } G \text{ for } W\}.$$

As a degenerate case, we define $\lambda[G, \emptyset] = 0$.

We note that $\lambda[G, W] = 1$ if and only if G admits a connection tree of G for W such that $\mathsf{R}(T) = \emptyset$.

Lemma 2.14. *Let G be a cograph and $W \subseteq V(G)$ be a terminal set. If G is the disjoint union of $k \geq 2$ cographs $G_1, \ldots, G_k$, then*

$$\lambda[G, W] = \sum_{i \in [k]} \lambda[G_i, V(G_i) \cap W].$$

Proof. Since G is the disjoint union of $G_1, \ldots, G_k$, there is no edge between the vertices of G_i and the vertices of G_j for any $i, j \in [k]$, with $i \neq j$. Thus, $\lambda[G, W] \geq \sum_{i \in [k]} \lambda[G_i, V(G_i) \cap W]$. On the other hand, let $\{j_1, \ldots, j_a\}$ be the set of indexes in $[k]$ such that $V(G_{j_i}) \cap W \neq \emptyset$ for each $i \in [a]$. Also, for each $i \in [a]$, let F_{j_i} be a routerless connection forests of G_i for $V(G_{j_i}) \cap W$ with the minimum number of connected components. One can readily verify that $F = F_{j_1} \cup \cdots \cup F_{ja}$ is a routerless connection forests of G for W. Therefore, $\lambda[G, W] \leq \sum_{i \in [k]} \lambda[G_i, V(G_i) \cap W]$. □

Lemma 2.15. *Let G be a cograph and $W \subseteq V(G)$ be a terminal set. If G is the join of $k \geq 2$ cographs $G_1, \ldots, G_k$ and there exists a unique $i \in [k]$ such that $V(G_i) \cap W \neq \emptyset$, then*

$$\lambda[G, W] = \max\{1, \lambda[G_i, W] - n + n_i\},$$

where $n = |V(G)|$ and $n_i = |V(G_i)|$.

Proof. Let F be a routerless connection forest of G for W. Since $W \subseteq V(G_i)$, $d_F(u) = 2$ for each $u \in V(F) \setminus V(G_i)$. This implies that, if T is the connected component of F that contains a vertex $u \in V(G) \setminus V(G_i)$, then $o(T - u) \leq 2$. In other words, for each $u \in V(G) \setminus V(G_i)$, there at most two distinct connected components of G_i that are connected in F by u. More generally, $F - (V(G) \setminus V(G_i))$ has at most $|V(G) \setminus V(G_i)| = n - n_i$ more connected components than F. Thus,

$$\lambda[G, W] \geq \max\{1, \lambda[G_i, W] - n + n_i\}.$$

On the other hand, let F_i be a routerless connection forest of G_i for W with the minimum number of connected components, *i.e.* $o(F_i) = \lambda[G_i, W]$, and let $S \subseteq V(G) \setminus V(G_i)$ such that $|S| = \min\{|V(G) \setminus V(G_i)|, o(F_i) - 1\}$. Then, choose an arbitrary connected component T of F_i, and some vertex $w_T \in V(T) \cap W$. Additionally, let $\alpha \colon S \to \mathsf{cc}(F_i) \setminus \{T\}$ be an injective map, and $w_H \in V(H) \cap W$ for each $H \in \mathsf{cc}(F_i) \setminus \{T\}$. We define F as the graph (see Figure 2.9) with vertex set $V(F) = V(F_i) \cup S$ and edge set

$$E(F) = E(F_i) \cup \{w_T u, u w_H \colon u \in S, H \in \mathsf{cc}(F_i) \setminus \{T\}, \alpha(u) = H\}.$$

One can verify that F is as routerless connection forest of G for W such that $o(F) = \lambda[G_i, W] - |S| = \max\{1, \lambda[G_i, W] - |V(G) \setminus V(G_i)|\}$. This implies that

$$\lambda[G, W] \leq \max\{1, \lambda[G_i, W] - n + n_i\},$$

concluding the proof. □

Theorem 2.6. *TCP is linear-time solvable on cographs.*

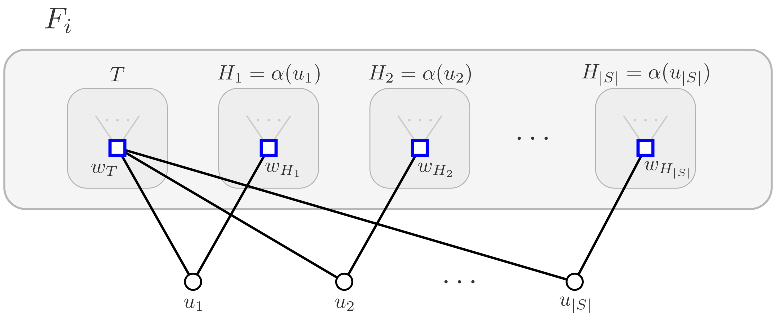


Figure 2.9: Graph F, with $S = \{u_1, \dots, u_{|S|}\}$ and $\alpha(u_l) = H_l$ for each $l \in [|S|]$.

Proof. Let $I = (G, W, \ell, r)$ be an instance of TCP, where G is a cograph on n vertices and m edges. Assume without loss of generality that $|W| \geq 3$, G is connected but $G[W]$ is not connected. Moreover, based on Lemma 2.12 and on Corollary 2.4, assume that $r = 0$ and $\ell \geq o(G[W]) - 1$, respectively. Then, compute $\lambda[G, W]$ by following the rules described below:

$$\lambda[G,W] = \begin{cases} \left[\begin{array}{l} \textbf{case 1.}\ |V(G)| = 1 : \\ \quad 0 \quad \text{if } V(G) \cap W = \emptyset, \\ \quad 1 \quad \text{otherwise;} \end{array}\right. \\ \\ \left[\begin{array}{l} \textbf{case 2.}\ G = G_1 \oplus \cdots \oplus G_k,\ \text{for some } k \geq 2 : \\ \quad \sum_{i \in [k]} \lambda[G_i, V(G_i) \cap W]; \end{array}\right. \\ \\ \left[\begin{array}{l} \textbf{case 3.}\ G = G_1 \wedge \cdots \wedge G_k,\ \text{for some } k \geq 2 : \\ \quad 0 \quad \text{if } \forall\, i \in [k],\ V(G_i) \cap W = \emptyset, \\ \quad 1 \quad \text{if } \exists\, i, j \in [k],\ i \neq j,\ V(G_i) \cap W \neq \emptyset \text{ and } V(G_j) \cap W \neq \emptyset, \\ \quad \max\{1, \lambda[G_i, W] - n + n_i\} \quad \text{if } \exists!\, i \in [k],\ V(G_i) \cap W \neq \emptyset, \\ \text{where } n = |V(G)| \text{ and } n_i = |V(G_i)|. \end{array}\right. \end{cases}$$

The correctness of the rules follows from Lemmas 2.12, 2.14 and 2.15. Since G admits a routerless connection tree if and only if $\lambda[G, W] = 1$, we have that I is a yes-instance of TCP if and only if $\lambda[G, W] = 1$.

Now, we analyse the time complexity of this algorithm. First, we note that $\lambda[G, W]$ can be computed in a bottom-up manner, according to the post-order traversal of the cotree $\mathcal{T}_G$ associated with G, using a dynamic programming matrix indexed by the nodes of $\mathcal{T}_G$. Moreover, we recall that $\mathcal{T}_G$ can be obtained in time $\mathcal{O}(n + m)$ *cf.* [27], and that, by definition, the number of nodes of $\mathcal{T}_G$ is $\mathcal{O}(n)$. Additionally, we note that, before computing $\lambda[G, W]$, $\mathcal{T}_G$ can be preprocessed in time $\mathcal{O}(n)$ so that each node u of $\mathcal{T}_G$ is associated with a flag which informs whether or not $V(G_u) \cap W \neq \emptyset$, where G_u denotes the subgraph of G corresponding to the subtree $\mathcal{T}_G^u$ of $\mathcal{T}_G$ rooted at u, i.e. G_u is the subgraph of G induced by the leaves of $\mathcal{T}_G^u$. Thus, one can verify that, for each node u of $\mathcal{T}_G$, the cell related to u of our dynamic programming matrix, which corresponds to $\lambda[G_u, V(G_u) \cap W]$, can be computed in time $\mathcal{O}\left(d_{\mathcal{T}_G}(u)\right)$. Since $\mathcal{T}_G$ is a tree on $\mathcal{O}(n)$ nodes, we have that $\sum_{u \in V(\mathcal{T}_G)} d_{\mathcal{T}_G}(u) = \mathcal{O}(n)$. Therefore, $\lambda[G, W]$ can be computed in time $\mathcal{O}(n + m)$. □

In addition to Theorem 2.6, we note that S-TCP can also be solved in polynomial-time on cographs. To prove this result, we use a similar approach, by defining a dynamic programming matrix based on the cotree of the input graph. We omit here the details of this proof and refer the reader to Section 5 in Appendix B.

2.4 Concluding Remarks

In this chapter, we have analysed the computational complexity of TCP and of S-TCP when they are restricted to specific graph classes and some of their input parameters are fixed. In particular, we have provided results that separate the complexity of TCP from the complexity of Steiner tree (see Table 2.1).

	Problem			
Graph class/Parameter	Steiner tree	TCP	TCP param. ℓ	TCP param. r
Split	NP-c [115]	NP-c Lemma 2.6	paraNP-c Lemma 2.6	XP, for $r \geq 1$ but W[1]-h Theorem 2.1
Rooted directed path graphs	Poly [115]	NP-c Theorem 2.3	Open	paraNP-c Theorem 2.3
Directed path graphs	Open	NP-c Theorem 2.3	Open	paraNP-c Theorem 2.3
Undirected path graphs	NP-c [34]	NP-c Theorem 2.3	Open	paraNP-c Theorem 2.3
Interval graphs	Poly [115]	Open	Open	Open
Cographs	Poly [24]	Poly Theorem 2.6	Poly Theorem 2.6	Poly Theorem 2.6
Clique-width	FPT [4]	W[1]-h Theorem 2.5	Open	W[1]-h Theorem 2.5

Table 2.1: Comparison between the computational complexity of TCP with the computational complexity of Steiner tree.

It is worth mentioning that, besides the proofs and statements explicitly presented, we have obtained other results for TCP and for S-TCP in Appendixes B–D. For instance, in Appendix C, we have shown that S-TCP is NP-complete on chordal bipartite graphs even when $\ell \geq 0$ is fixed. The exact same proof can be used to show that, for fixed $\ell \geq 0$, TCP is also NP-complete on chordal bipartite graphs, agreeing with the complexity of Steiner tree [99]. In Section 3 of Appendix B, we have shown that S-TCP parameterized by the maximum degree of the input graph

Δ, ℓ and r does not admit a polynomial kernel, unless $\mathsf{NP} \subseteq \text{coNP/poly}$, while it is known to be in FPT [51]. Despite that, as for TCP parameterized by Δ, ℓ and r, it is unknown whether the problem is in FPT. In addition, many other interesting questions remain open. In what follows, we highlight some of those questions.

We have proved that, on split graphs, TCP is polynomial-time solvable for every fixed $r \geq 1$, whereas STEINER TREE is known to be NP-complete [115]; and that, on rooted directed path graphs, TCP is NP-complete for every $r \geq 0$, whereas STEINER TREE is known to be polynomial-time solvable [16, 59, 115]. However, up to our knowledge there is no known example of a graph class $\mathcal{G}$ on which, for fixed ℓ, TCP is polynomial-time solvable while STEINER TREE is NP-complete, or vice-versa.

In addition, it is worth mentioning that, in our tractability proof for TCP on split graphs, only the cases in which $r \geq 1$ or $W \cap K \neq \emptyset$ are considered. Such hypotheses are imperative in our argumentation so as to ensure the connectivity of the sought tree. Thus, we leave as an open question whether TCP can be solved in polynomial-time on split graphs when $r = 0$ and $W \cap K = \emptyset$. We also ask whether TCP and S-TCP parameterized by clique-width are in XP. Through a parameterized-reduction from HAMILTONIAN PATH, we have shown that these problems parameterized by clique-width are W[1]-hard. Another intriguing question is related to the case in which the number of terminals is fixed. Even though it is well-known that STEINER TREE parameterized by the number of terminal vertices is in FPT [56], the complexity of the corresponding parameterizations of TCP and of S-TCP is widely open.

Concerning S-TCP in particular, the main open question is whether the problem parameterized by r is in XP. It follows from Corollary 2.1 that the problem is not in FPT, unless FPT = W[2]. Additionally, it is known that S-TCP can be solved in polynomial-time when $r \in \{0, 1\}$ *cf.* [39, 95]. However, deciding whether S-TCP is in XP, for fixed $r \geq 2$, is still unknown. For more details on this, we refer to [39, 40], where we established close connections between S-TCP with fixed r and disjoint path problems, besides investigating the complexity of some variants of S-TCP in order to better understanding what makes the problem difficult.

Concerning STEINER TREE, we proved in [34] (see Section 4 of Appendix A) that STEINER TREE is NP-complete on undirected path graphs (for short, UV). On the other hand, it is known that the problem is polynomial-time solvable on strongly chordal graphs [115], a superclass of rooted directed path graphs (for short, RDV). However, it remains unsettled whether STEINER TREE is polynomial-time solvable on directed path graphs (for short, DV), a superclass of RDV and a subclass of UV. We note that DV graphs are not strongly chordal, since the graph 4-*sun* is DV but is not strongly chordal [58, 102]. Additionally, we note that STEINER TREE is akin to domination problems, in particular to CONNECTED DOMINATING SET [115], and that the complexity of such problems on DV graphs also remains unsettled [18, 37].

Chapter 3

Maximum Cut

In this chapter, we analyse the computational complexity of the MaxCut problem when restricted to interval graphs of bounded interval count, and when restricted to permutation graphs.

Many important graph classes are defined or can be characterized by a geometric intersection model. Two well-studied examples are the classes of interval graphs and of permutation graphs [13, 61, 74, 111]. In their respective models, the intersecting objects are line segments in the plane, with different constraints on their positions. In interval graphs, each line segment must have its endpoints on a single line, while in permutation graphs, the endpoints must lie on two distinct parallel lines.

MaxCut is one of the most classical problems on graphs, and it is known to be NP-complete since the seventies [67]. Nonetheless, only recently its restriction to interval graphs has been announced to be hard. This result was proved by Adhikary, Bose, Mukherjee, and Roy [1], settling a long-standing open problem from the *Ongoing Guide to NP-completeness* by David S. Johnson [83]. Yet, the complexity on the even more restrict class of unit/proper interval graphs, *i.e.* interval graphs of interval count 1, still remains unknown. In fact, many flawed proofs of polynomial-time solvability for the problem on the class of unit interval graphs have been presented along the years [11, 15], just to be disproved closely after [9, 89].

We present the first complexity result for MaxCut on interval graphs of bounded interval count. More specifically, we prove that the problem remains NP-complete on interval graphs of interval count 4. Our contribution is an improvement towards filling the complexity gap between interval and unit interval graphs. Additionally, we prove that the problem is also NP-complete on permutation graphs, closing another long-standing question from [83]. Our results are built on the general idea behind the NP-completeness proof of MaxCut on interval graphs.

This chapter is organised as follows. In Section 3.1, we formally define the MaxCut problem. In addition, we present the definitions of the classes of interval graphs and permutation graphs, and of the notion of interval count of an interval graph. In Section 3.2, we define the concept of grained gadgets. Such gadgets play a central role in the proof given by Adhikary et al. [1], and they are extensively used in both of our hardness proofs. In Section 3.3, we revisit Adhikary et al.'s reduction, and we show that their constructed interval graph has unbounded interval count and is not a permutation graph. In Sections 3.4 and 3.5, we present our NP-completeness proofs for MaxCut on interval graphs of interval count 4 and on permutation graphs, respectively. Finally, in Section 3.6 we present the concluding remarks of this chapter, focusing on the open questions.

3.1 Basic Definitions

Let G be a graph. We recall that a cut of G is a partition $[A, B]$ of $V(G)$ into two parts $A, B \subseteq V(G)$, and that the cut-set of G associated with $[A, B]$ is the set $E_G(A, B)$ of edges of G with an endpoint in A and the other endpoint in B. We denote by $\mathsf{mc}(G)$ the maximum size of a cut-set of G, *i.e.*

$$\mathsf{mc}(G) = \max\{|E_G(A, B)| \colon [A, B] \text{ is a cut of } G\}.$$

Next, we formally define the MaxCut problem.

MaxCut

Input: A graph G and a positive integer k.

Question: Is $\mathsf{mc}(G) \geq k$?

If $I \subseteq \mathbb{R}$ is a closed interval of the real line, then we let $\ell(I) = \min\{p \colon p \in I\}$ and $r(I) = \max\{p \colon p \in I\}$ be the *left* and the *right endpoints* of I, respectively. We denote a closed interval I by $[\ell(I), r(I)]$. The *length* of an interval I is defined as $|I| = r(I) - \ell(I)$. An *interval model* is a finite multiset $\mathcal{M}$ of intervals. The *interval count* of an interval model $\mathcal{M}$, denoted by $\mathsf{ic}(\mathcal{M})$, is defined as the number of distinct lengths of the intervals in $\mathcal{M}$, *i.e.* $\mathsf{ic}(\mathcal{M}) = |\{|I| \colon I \in \mathcal{M}\}|$. Let $\mathcal{M}$ be an interval model and $I, I' \in \mathcal{M}$ be two intervals such that $I \cap I' \neq \emptyset$. We say that I *covers* I' if $I \supseteq I'$, I intersects I' *to the left* if $\ell(I) < \ell(I')$ and $r(I) < r(I')$, and that I intersects I' *to the right* if $\ell(I) > \ell(I')$ and $r(I) > r(I')$.

Let G be a graph and $\mathcal{M}$ be an interval model. An $\mathcal{M}$*-representation* of G is a bijection $\phi \colon V(G) \to \mathcal{M}$ such that, for every two distinct vertices $u, v \in V(G)$, we have that $uv \in E(G)$ if and only if $\phi(u) \cap \phi(v) \neq \emptyset$. If such an $\mathcal{M}$-representation exists, we say that $\mathcal{M}$ is an *interval model of* G. We note that a graph may have either no interval model or arbitrarily many distinct interval models. A graph is called an *interval graph* if it has an interval model. The *interval count of an interval graph* G, denoted by $\mathsf{ic}(G)$, is defined as $\mathsf{ic}(G) = \min\{\mathsf{ic}(\mathcal{M})\colon \mathcal{M}$ is an interval model of $G\}$. An interval graph is called a *unit interval graph* if its interval count is equal to 1.

Note that, for every interval model $\mathcal{M}$, there exists a unique (up to isomorphism) graph that admits an $\mathcal{M}$-representation. Thus, for every interval model $\mathcal{M} = \{I_1, \ldots, I_n\}$, we let $\mathbb{G}_{\mathcal{M}}$ be the graph with vertex set $V(\mathbb{G}_{\mathcal{M}}) = \{1, \ldots, n\}$ and edge set $E(\mathbb{G}_{\mathcal{M}}) = \{ij\colon I_i, I_j \in \mathcal{M},\ I_i \cap I_j \neq \emptyset,\ i \neq j\}$. Since $\mathbb{G}_{\mathcal{M}}$ is uniquely determined (up to isomorphism) from $\mathcal{M}$, in what follows we may make an abuse of language and use graph terminologies to describe properties related to the intervals in $\mathcal{M}$. For instance, two intervals $I_i, I_j \in \mathcal{M}$ are said to be *true twins* in $\mathbb{G}_{\mathcal{M}}$ if their respective vertices have the same closed neighbourhood in $\mathbb{G}_{\mathcal{M}}$.

Let π and π' be two permutations of a same set, say V. A graph G is called the *intersection graph related to* $\{\pi, \pi'\}$ if $V(G) = V$ and, for each two vertices $u, v \in V(G)$, $uv \in E(G)$ if and only if $u <_\pi v$ and $v <_{\pi'} u$. In this case, we also say that $\{\pi, \pi'\}$ is a *permutation model* of G. A graph is a *permutation graph* if it is the intersection graph related to a permutation model.

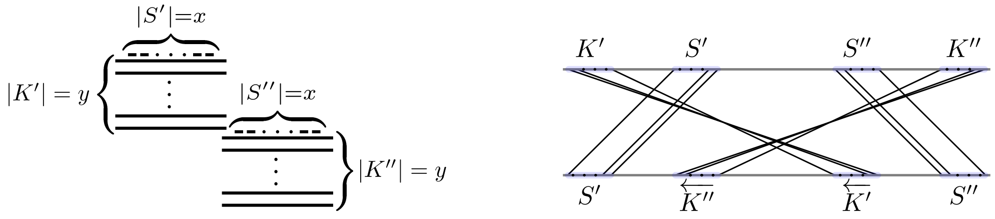
Given two permutations π and γ of disjoint subsets X and Y, respectively, we write $\pi\gamma$ to denote the permutation of $X \cup Y$ given by the *concatenation* of π with γ. Also, we write $\overleftarrow{\pi}$ to denote the reverse of the permutation π, *i.e.* if $\pi = (v_1, \ldots, v_i)$, then $\overleftarrow{\pi} = (v_i, \ldots, v_1)$. In order to simplify the notation, given a set Z, we sometimes use the same symbol, Z, to denote also a chosen permutation of the elements of Z; in such cases, $\overleftarrow{Z}$ represents the reverse of the chosen permutation for Z.

3.2 Grained Gadgets

Let x and y be positive integers. An *(x, y)-grained gadget* is a split graph H formed by a clique $K' \cup K''$ of size $2y$ and a stable set $S' \cup S''$ of size $2x$ with K' being complete to S', K'' being complete to S'', and satisfying $|K'| = |K''| = y$ and $|S'| = |S''| = x$. Figure 3.1a depicts an interval representation of an (x, y)-grained gadget. Furthermore, one can readily verify that the intersection graph related to the pair of permutations $\{K'S'S''K'', S'\overleftarrow{K''}\overleftarrow{K'}S''\}$ (see Figure 3.1b) is an (x, y)-grained gadget. Thus, grained gadgets are both interval graphs and permutation graphs.

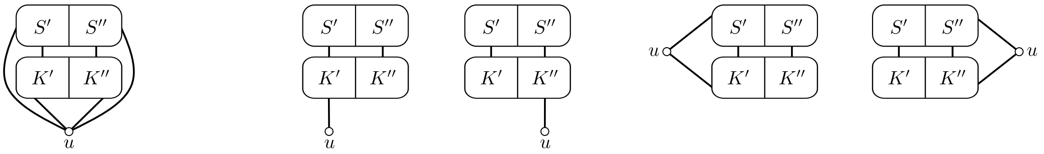
Let H be an (x, y)-grained gadget and G be a supergraph of H. For each vertex $u \in V(G) \setminus V(H)$, we say that (see Figure 3.2): *u covers H* if $V(H) \subseteq N_G(u)$, *i.e.* u is universal with respect to H; *u weakly intersects H* if either $N_G(u) \cap V(H) = K'$ or $N_G(u) \cap V(H) = K''$; and that *u strongly intersects H* if either $N_G(u) \cap V(H) = K' \cup S'$ or $N_G(u) \cap V(H) = K'' \cup S''$. Moreover, we say that *G respects the structure* of H if, for each vertex $u \in V(G) \setminus V(H)$, either $N_G(u) \cap V(H) = \emptyset$ or u satisfies one of the previous conditions.

The next lemma establishes the key property of grained gadgets with respect to



(a) Interval representation (b) Permutation model

Figure 3.1: Interval representation and a permutation model of an (x, y)-grained gadget, respectively.



(a) Covering intersection (b) Weak intersection (c) Strong intersection

Figure 3.2: Vertex $u \in V(G) \setminus V(H)$ (a) covering H, (b) weakly intersecting H, and (c) strongly intersecting H. The set $K' \cup K''$ is a clique and the set $S' \cup S''$ is a stable set. A line between sets, or between u and some set, means that all the edges occur.

the MaxCut problem. Intuitively, it states that, for suitable values of x and y, if G is a supergraph that respects the structure of an (x, y)-grained gadget, then, in any maximum cut $[A, B]$ of G, the vertices belonging to $K' \cup S''$ are placed in a same part of $[A, B]$, opposite to the part containing the vertices belonging to $K'' \cup S'$.

Lemma 3.1. *Let x and y be positive integers, H be an (x, y)-grained gadget and G be a supergraph that respects the structure of H. Also, let $[A, B]$ be a maximum cut of G, t be the number of vertices in $V(G) \setminus V(H)$ adjacent to some vertex of H, ℓ be the number of vertices of G adjacent to some vertex in S', and r be the number of vertices of G adjacent to some vertex in S''. If ℓ and r are odd, $y > 2t$ and $x > t + 2y$, then each of the following holds:*

- *$S' \subseteq A$ and $K' \subseteq B$, or vice versa;*
- *$S'' \subseteq A$ and $K'' \subseteq B$, or vice versa;*
- *$K' \subseteq A$ and $K'' \subseteq B$, or vice versa.*

Proof. First, we prove that the vertices in S' belong to the same part of $[A, B]$. Since ℓ is odd, either A or B contains more than half of the vertices in $N_G(S')$. Suppose without loss of generality that this is satisfied by B. This implies that $|B \cap N_G(S')| > |A \cap N_G(S')|$. Thus, since S' is a stable set, every vertex in S' must belong to A, otherwise $[A, B]$ would not be maximum. Note that a similar argument applies to the vertices in S'', since r is odd and S'' is a stable set. Therefore, in the remainder of this proof, assume that $S' \subseteq X$ and $S'' \subseteq Y$, for some $X, Y \in \{A, B\}$.

Now, we consider the vertices in K'. Note that, the number of vertices in $V(G)\backslash S'$ adjacent to some vertex in K' is less than $t + 2y < x$. Therefore, since $|S'| = x$, $K' \subseteq V(G) \setminus X$, *i.e.* K' must be contained in the part of $[A, B]$ opposite to the part containing S', otherwise $[A, B]$ would not be maximum. We note that an analogous argument applies to the vertices in K'', and thus $K'' \subseteq V(G) \setminus Y$.

Finally, we prove that $X \neq Y$. Let α denote the number of edges in the cut-set $E_G(A, B)$ that are incident to some vertex in $K' \cup K''$. By definition, every edge of G incident to a vertex in K' (resp. K'') has as the other endpoint a vertex in $V(G) \setminus V(H)$, a vertex in S' (resp. S''), or a vertex in K'' (resp. K'). Moreover, we have shown that $K' \subseteq V(G) \setminus X$ and $K'' \subseteq V(G) \setminus Y$. Hence, if $X = Y$, then $\alpha \leq 2ty + 2xy$. On the other hand, if $X \neq Y$, then $\alpha \geq y^2 + 2xy$. Therefore, since $y > 2t$, $X \neq Y$, otherwise $[A, B]$ would not be maximum. □

Let G be a graph containing a grained gadget H as a subgraph. We say that the pair (H, G) is *well-valued* if G respects the structure of H and, in addition, the conditions of Lemma 3.1 are satisfied by H and G. More generally, we say that G is *well-valued* if, for every grained gadget H' in G, (H', G) is well-valued. Let $[A, B]$ be a maximum cut of G. We say that H is *A-partitioned* by $[A, B]$ if $S' \cup K'' \subseteq A$ and $S'' \cup K' \subseteq B$. The notion of *B-partitioned* is defined analogously.

Corollary 3.1. *Let G be a graph and H be a (x, y)-grained gadget in G, such that (H, G) is well-valued. Also, let Z and Z' be the sets of vertices that weakly intersect and strongly intersect H, respectively. Assume that Z' is non-empty and $|Z| \leq 2|Z'|$. If $[A, B]$ is a maximum cut of G and $Z' \subseteq B$, then H is A-partitioned.*

Proof. Since (H, G) is well-valued, $x > 2y$. Moreover, by Lemma 3.1, H is either A-partitioned or B-partitioned. Let h denote the number of edges in the cut-set that are between the vertices of H and the vertices in $V(G) \setminus V(H)$, and let c denote the number of vertices of G that cover H. Since $Z' \subseteq B$, one can verify that, if H is B-partitioned, then $h \leq (x + y)c + y|Z|$; while, if H is A-partitioned, then $h \geq (x + y)c + x|Z'|$. Therefore, since $x > 2y$ and $|Z| \leq 2|Z'|$, H must be A-partitioned, otherwise $[A, B]$ would not be maximum. □

In the remainder of the text, when a grained gadget H is not clear in the context, we write $S'(H)$, $S''(H)$, $K'(H)$ and $K''(H)$ to denote the stable sets S' and S'' and the cliques K' and K'' of H, respectively. Additionally, for simplicity of notation, when considering interval graphs, we may make an abuse of language and regard a grained gadget H as any of its interval models. In such cases, motivated by the interval representation (see Figure 3.1a), we may call the vertices in $S' \cup S''$ *short intervals* and the vertices in $K' \cup K''$ *long intervals* of H.

3.3 Adhikary et al.'s Reduction

In this section, we briefly present the polynomial-time reduction given by Adhikary et al. [1] so as to prove the NP-completeness of MaxCut on interval graphs. Their reduction is from MaxCut on cubic graphs, which is known to be NP-complete [5]. We show that their constructed interval graph has unbounded interval count and is not a permutation graph.

Given a cubic graph G, let $\pi_V = (v_1, \dots, v_n)$ and $\pi_E = (e_1, \dots, e_m)$ be arbitrary orderings of $V(G)$ and $E(G)$, respectively. Define the values: $q = 200n^3 + 1$, $p = 2q + 7n$, $q' = 10n^2 + 1$, and $p' = 2q' + 7n$. An interval graph G' is defined through the construction of one of its interval models $\mathcal{M}$, defined as described below (observe Figure 3.3 to follow the construction).

1. For each vertex $v_i \in V(G)$, add a (p, q)-grained gadget $\mathcal{H}_i$. These gadgets should be pairwise disjoint, with $\mathcal{H}_i$ appearing completely to the left of $\mathcal{H}_{i+1}$ for every $i \in [n-1]$.

2. For each edge $e_j \in E(G)$, add a (p', q')-grained gadget $\mathcal{E}_j$. Likewise, these gadgets should be pairwise disjoint, with $\mathcal{E}_j$ appearing completely to the left of $\mathcal{E}_{j+1}$ for every $j \in [m-1]$. Additionally, $\mathcal{E}_1$ should appear completely to the right of $\mathcal{H}_n$, without intersecting it.

3. Finally, for each edge $e_j = v_i v_{i'} \in E(G)$, with $i < i'$, add four intervals $C^1_{i,j}, C^2_{i,j}, C^1_{i',j}, C^2_{i',j}$, called *incidence intervals*, such that:
   - $C^1_{i,j}$ and $C^2_{i,j}$ (resp. $C^1_{i',j}$ and $C^2_{i',j}$) weakly intersect $\mathcal{H}_i$ (resp. $\mathcal{H}_{i'}$) to the right of $\mathcal{H}_i$ (resp. $\mathcal{H}_{i'}$);
   - $C^1_{i,j}$ and $C^2_{i,j}$ (resp. $C^1_{i',j}$ and $C^2_{i',j}$) weakly intersect (resp. strongly intersect) $\mathcal{E}_j$ to the left of $\mathcal{E}_j$.

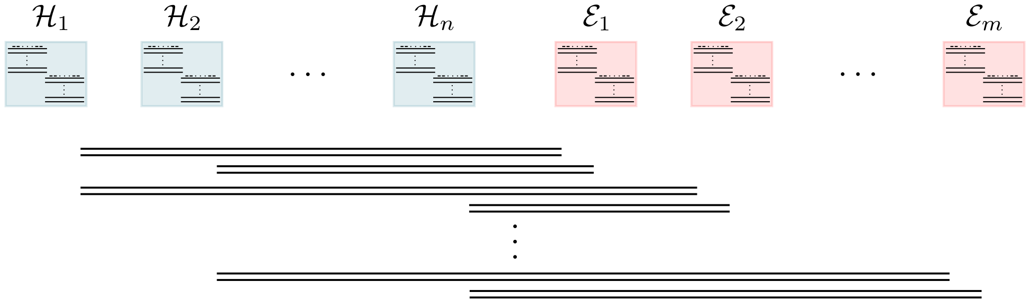
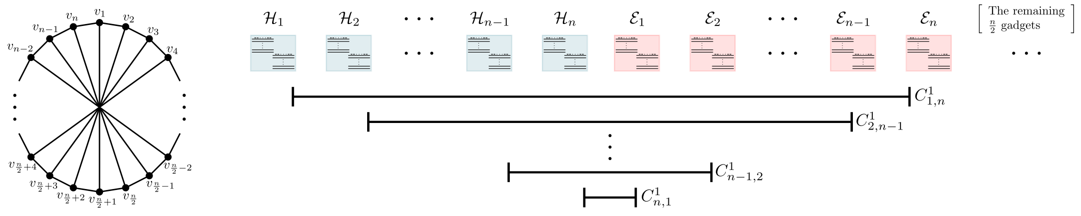


Figure 3.3: Example of Adhikary et al.'s interval model $\mathcal{M}$, obtained from a graph with edges $e_1 = v_1v_2$, $e_2 = v_1v_n$, and $e_m = v_2v_n$.

Now, we show that the interval count of the constructed interval graph G' is unbounded. More specifically, we prove that, there exist a cubic graph G and orderings π_V and π_E of $V(G)$ and of $E(G)$, respectively, from which the resulting interval graph G' has interval count $\Omega(n)$, where n denotes the number of vertices of G. It is worth noticing that the number of intervals in $\mathcal{M}$, and therefore the number of vertices in G', is invariant under the particular choices of π_V and π_E. Nonetheless, one can verify that distinct orderings yield distinct interval graphs, possibly having distinct interval counts, since the set of intervals covered by any of the intervals $C^1_{i,j}, C^2_{i,j}, C^1_{i',j}, C^2_{i',j}$ depends heavily on π_V and π_E.

Consider the cubic graph G depicted in Figure 3.4a, which consists in an even cycle $C = (v_1, v_2, \ldots, v_n)$ with the addition of the edges $v_i v_{i+\frac{n}{2}}$ for every $i \in [n/2]$. Additionally, consider the ordering $\pi_V = (u_1, \ldots, u_n)$ of $V(G)$, where $u_i = v_{n-i+1}$ for every $i \in [n]$. Also, let $\pi_E = (e_1, \ldots, e_m)$ be any ordering of $E(G)$ such that $e_i = u_{n-i}u_{n-i+1} = v_{i+1}v_i$ for every $i \in [n-1]$, and $e_n = u_1u_n = v_nv_1$. One can verify that, for the graph G and the orderings π_V and π_E, the reduction yields a model $\mathcal{M}$ that contains a chain $C^1_{n,1} \subset C^1_{n-1,2} \subset \ldots \subset C^1_{1,n}$ of nested intervals (see Figure 3.4b). This immediately implies that $\mathsf{ic}(\mathcal{M}) \geq n$.



(a) (b)

Figure 3.4: (a) A cubic graph G, and (b) a chain of nested intervals in the model $\mathcal{M}$.

Now, we prove that the constructed graph G' is not a permutation graph, and that this holds regardless of the input cubic graph and the input orderings π_V and π_E. For that, it suffices to note that G' contains the graph $\overline{X}_{34}$ depicted in Figure 3.5a as an induced subgraph, which is a known forbidden subgraph for comparability graphs [47, 63], in turn a superclass of permutation graphs. Indeed, observe Figure 3.5b. Given an edge $e_j = v_iv_{i'} \in E(G)$, with $i < i'$, it is depicted the intervals in the grained gadgets of v_i, $v_{i'}$ and e_j, as well as some incidence intervals related to e_j. The adjacencies can be easily checked to be as in the graph of Figure 3.5a.

3.4 Interval Graphs of Bounded Interval Count

In this section, we extended the result given by Adhikary et al [1], by showing that MaxCut remains NP-complete even on interval graphs of bounded interval count. More specifically, through a polynomial-time reduction from MaxCut on cubic graphs, we prove the following theorem:

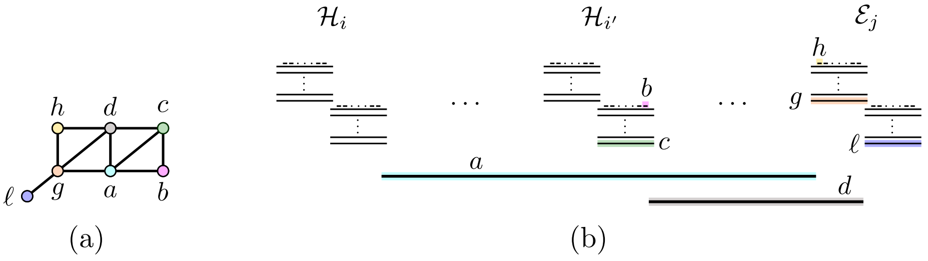


Figure 3.5: (a) Forbidden induced subgraph $\overline{X}_{34}$ for comparability graphs *cf.* [47]. (b) $\overline{X}_{34}$ as an induced subgraph in Adhikary et al.'s construction.

Theorem 3.1. MaxCut *is* NP*-complete on interval graphs of interval count* 4.

3.4.1 Reduction Graph

Let G be a cubic graph on n vertices and $m = 3n/2$ edges, $\pi_V = (v_1, \ldots, v_n)$ be an ordering of the vertices of G, and $\pi_E = (e_1, \ldots, e_m)$ be an ordering of the edges of G. Also, let $\mathfrak{G}$ denote the triple (G, π_V, π_E).

Intuitively, we select m mutually disjoint regions in the real line, such that, for each $j \in [m]$, the j-th region is related to the edge e_j and holds the information whether e_j is in the cut-set. To accomplish this, the edge e_j is represented by a grained gadget $\mathcal{E}_j$, which must be contained within the j-th region, and each vertex v_i is represented by a grained gadget $\mathcal{H}_i^j$ and special intervals, called *link intervals*, connecting $\mathcal{H}_i^j$ to $\mathcal{H}_i^{j+1}$. The aim of such link intervals is to propagate, from a region to the next, the information about to which part of the cut the respective vertex v_i belongs. In addition, in order to represent that a vertex v_i is an endpoint of an edge e_j, a couple of intervals, called *incidence intervals*, connecting $\mathcal{H}_i^j$ to $\mathcal{E}_j$ is added. Figure 3.6 depicts this general idea of our construction.

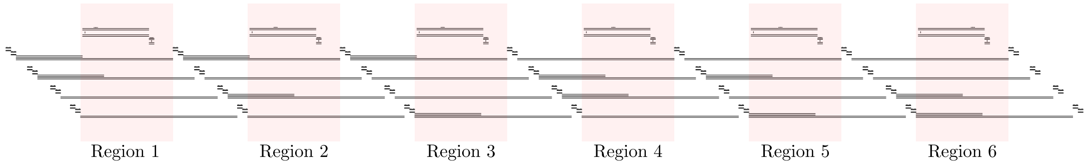


Figure 3.6: General structure of our interval model. In this example, the complete graph on 4 vertices, K_4, is given as the input cubic graph.

Now, we formally define our interval model. We first describe the gadgets related to the vertices. Please, refer to Figure 3.7 to follow the construction. The values of p, q used next are properly defined later. An (n, m)-*escalator* is an interval model comprising, for each vertex $v_i \in V(G)$, $m+1$ mutually disjoint (p, q)-grained gadgets, denoted by $\mathcal{H}_i^1, \ldots, \mathcal{H}_i^{m+1}$, along with $2m$ *link intervals*, denoted by $L_i^1, \ldots, L_i^{2m}$, such that L_i^{2j-1} and L_i^{2j} weakly intersect $\mathcal{H}_i^j$ to the right and weakly intersect $\mathcal{H}_i^{j+1}$ to the left. Additionally, for each $j \in [m+1]$ and each pair $i, i' \in [n]$ with $i < i'$, the grained gadget $\mathcal{H}_i^j$ must be placed to the left of $\mathcal{H}_{i'}^j$, and the grained gadget $\mathcal{H}_n^j$ must be placed to the left of $\mathcal{H}_1^{j+1}$ for $j \in [m]$.

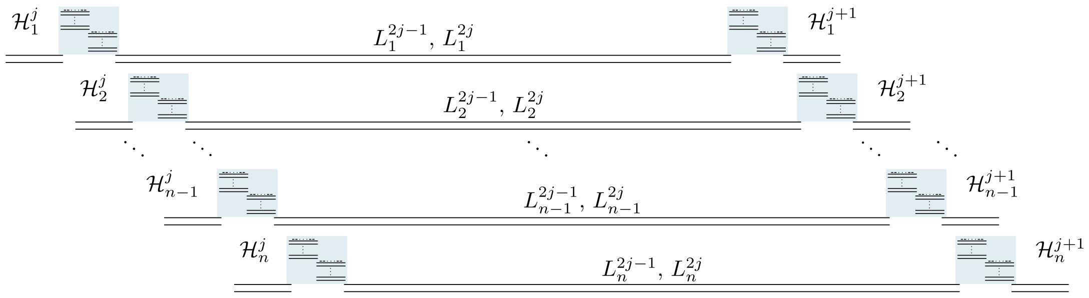


Figure 3.7: An (n, m)-escalator. The shaded rectangles represent the vertex (p, q)-grained gadgets $\mathcal{H}_1^j, \ldots, \mathcal{H}_n^j$ and $\mathcal{H}_1^{j+1}, \ldots, \mathcal{H}_n^{j+1}$.

Finally, we describe the gadgets related to the edges. Please, refer to Figure 3.8 to follow the construction. The values of p', q' used next are properly defined later. For each edge $e_j = v_i v_{i'} \in E(G)$, with $i < i'$, create a (p', q')-grained gadget $\mathcal{E}_j$ and intervals $C_j^1, C_j^2, C_j^3, C_j^4$, called *incidence intervals*[1], in such a way that $\mathcal{E}_j$ is entirely contained in the j-th region (i.e., in the open interval between the right endpoint of $\mathcal{H}_n^j$ and the left endpoint of $\mathcal{H}_1^{j+1}$), C_j^1 and C_j^2 weakly intersect $\mathcal{H}_i^j$ to the right and weakly intersect $\mathcal{E}_j$ to the left, and C_j^3 and C_j^4 weakly intersect $H_{i'}^j$ to the right and strongly intersect $\mathcal{E}_j$ to the left.

We write $\mathcal{M}(\mathfrak{G})$ to denote any selected interval model, obtained from $\mathfrak{G}$, that satisfies the conditions described above. This model defines our reduction interval graph $\mathbb{G}_{\mathcal{M}(\mathfrak{G})}$. We note that, if $\mathcal{M}$ and $\mathcal{M}'$ are any two interval models satisfying the conditions described above, then their corresponding interval graphs $\mathbb{G}_{\mathcal{M}}$ and $\mathbb{G}_{\mathcal{M}'}$ are isomorphic. In what follows, when $\mathfrak{G}$ is clear in the context, we may omit it, and simply write $\mathcal{M}$ and $\mathbb{G}_{\mathcal{M}}$ to denote $\mathcal{M}(\mathfrak{G})$ and $\mathbb{G}_{\mathcal{M}(\mathfrak{G})}$, respectively.

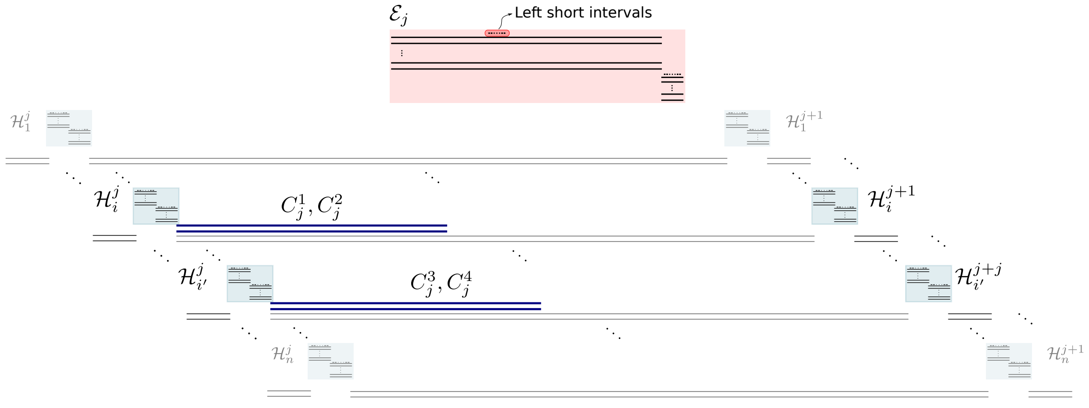


Figure 3.8: Interval model $\mathcal{M}(\mathfrak{G})$. The shaded rectangle on the top represents the edge (p', q')-grained gadget $\mathcal{E}_j$.

[1] In Appendix E, such intervals are called *intervals of type C*.

The following lemma is straightforward, and it is used in the next sections in order to ensure that, for suitable values of p, q, p' and q', Lemma 3.1 can always be applied to every grained gadget of $\mathcal{M}(\mathfrak{G})$.

Lemma 3.2. *Let G be a graph, $\pi_V = (v_1, \ldots, v_n)$ and $\pi_E = (e_1, \ldots, e_m)$ be orderings of $V(G)$ and $E(G)$, respectively, and $\mathfrak{G} = (G, \pi_V, \pi_E)$. The following properties hold for every vertex/edge grained gadget $\mathcal{H}$ of $\mathcal{M}(\mathfrak{G})$:*

- *$\mathcal{M}(\mathfrak{G})$ respects the structure of $\mathcal{H}$;*
- *The number of intervals in $\mathcal{M}(\mathfrak{G})$ covering $\mathcal{H}$ is even; and*
- *The number of intervals in $\mathcal{M}(\mathfrak{G})$ strongly intersecting $\mathcal{H}$ to the left and the number of intervals in $\mathcal{M}(\mathfrak{G})$ strongly intersecting $\mathcal{H}$ to the right are both even.*

Observe that Lemma 3.2 implies that, in order for the values ℓ and r in Lemma 3.1 to be odd, it suffices to choose odd values for q and q'.

3.4.2 Maximum Cut of the Reduction Graph

We now prove that $\mathsf{mc}(G) \geq k$ if and only if $\mathsf{mc}(\mathbb{G}_{\mathcal{M}(\mathfrak{G})}) \geq \phi(n, k)$, where ϕ is a suitable function, properly defined at the end of this subsection.

As usual in these types of reductions, constructing a maximum cut of the reduction graph $\mathbb{G}_{\mathcal{M}(\mathfrak{G})}$, from a given maximum cut $[X, Y]$ of the input graph G, is generally an easy task, which does not depend on any additional properties of $[X, Y]$. On the other hand, constructing a maximum cut $[X, Y]$ of G, from a given a maximum cut $[A, B]$ of $\mathbb{G}_{\mathcal{M}(\mathfrak{G})}$, requires that $[A, B]$ exhibits some particular properties. Thus, in order to ensure that any maximum cut of $\mathbb{G}_{\mathcal{M}(\mathfrak{G})}$ always owns such properties, we must manipulate appropriately the values of p, q, p', q', as done in Lemma 3.1.

Next lemma imposes some further conditions on the values of p, q, p', q' so that, in any maximum cut of $\mathbb{G}_{\mathcal{M}(\mathfrak{G})}$, the partitioning of the edge grained gadget related to an edge $e_j = v_i v_{i'}$, with $i < i'$, depends solely on the partitioning of $\mathcal{H}^j_{i'}$.

A cut $[A, B]$ of $\mathbb{G}_{\mathcal{M}(\mathfrak{G})}$ is said to be *locally well-behaved* if, for every $i \in [n]$ and every $j \in [m+1]$, $\mathcal{H}^j_i$ is either A-partitioned or B-partitioned by $[A, B]$, and for every edge $e_j = v_i v_{i'}$ of G, with $i < i'$, the following conditions are satisfied:

- If $\mathcal{H}^j_i$ is A-partitioned by $[A, B]$, then $\{C^1_j, C^2_j\} \subseteq B$; otherwise, $\{C^1_j, C^2_j\} \subseteq A$;
- If $\mathcal{H}^j_{i'}$ is A-partitioned by $[A, B]$, then $\{C^3_j, C^4_j\} \subseteq B$ and $\mathcal{E}_j$ is A-partitioned by $[A, B]$; otherwise, $\{C^3_j, C^4_j\} \subseteq A$ and $\mathcal{E}_j$ is B-partitioned by $[A, B]$.

Figure 3.9 illustrates this notion of locally well-behaved cut.

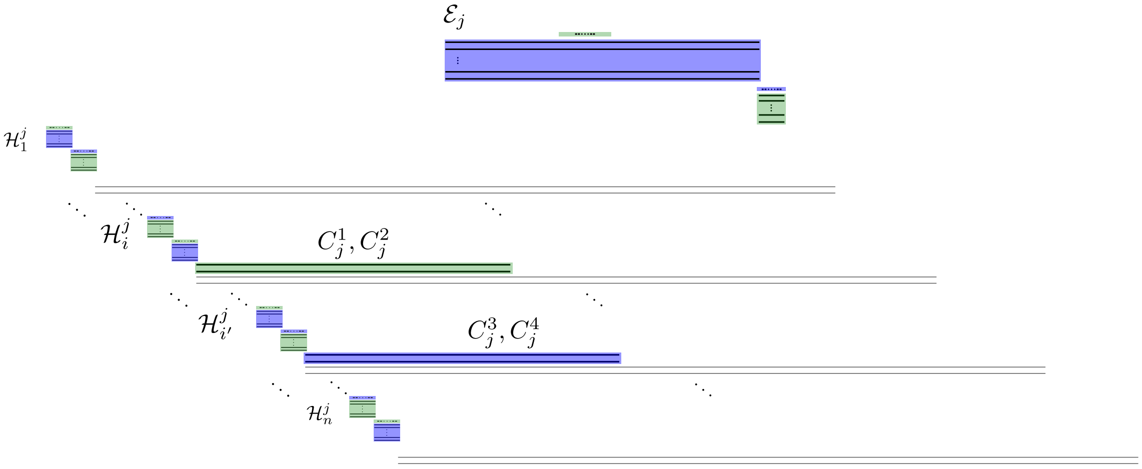


Figure 3.9: General idea of locally well-behaved cuts of $\mathbb{G}_{\mathcal{M}(\mathfrak{G})}$. Each part of the cut is represented by a distinct colour. For instance, considering a cut $[A, B]$ and the part A represented by the colour green, we have in this case that the grained gadgets $\mathcal{H}_1^j$, $\mathcal{H}_{i'}^j$ and $\mathcal{E}_j$ are A-partitioned, while $\mathcal{H}_i^j$ and $\mathcal{H}_n^j$ are B-partitioned. Note that, since $\mathcal{H}_{i'}^j$ is A-partitioned, $\{C_j^1, C_j^2\} \subseteq B$ and $\mathcal{E}_j$ is A-partitioned as well.

Lemma 3.3. *Let G be a cubic graph, $\pi_V = (v_1, \ldots, v_n)$ and $\pi_E = (e_1, \ldots, e_m)$ be orderings of $V(G)$ and $E(G)$, respectively, $\mathfrak{G} = (G, \pi_V, \pi_E)$, and $e_j = v_i v_{i'}$ be an edge of G, with $i < i'$. If $[A, B]$ is a maximum cut of $\mathbb{G}_{\mathcal{M}(\mathfrak{G})}$, $\mathbb{G}_{\mathcal{M}(\mathfrak{G})}$ is well-valued and, in addition, $q > 4n + p' + q'$, then $[A, B]$ is locally well-behaved.*

Proof. Since $\mathcal{M}(\mathfrak{G})$ is well-valued, it follows from Lemma 3.1 that every grained gadget in $\mathcal{M}(\mathfrak{G})$ is either A-partitioned or B-partitioned. Suppose without loss of generality that $\mathcal{H}_i^j$ is A-partitioned, *i.e.* $S' \cup K'' \subseteq A$ and $S'' \cup K' \subseteq B$. We prove that $C_j^l \in B$, where $l \in \{1, 2\}$. First, note that, based on Lemma 3.1, all grained gadgets in $\mathcal{M}(\mathfrak{G})$ have the same number of intervals belonging to A and of intervals belonging to B. In particular, $\mathcal{H}_{i+1}^j, \ldots, \mathcal{H}_n^j$ comprise all the grained gadgets covered by C_j^l, and in total there are in such gadgets exactly $(n-i)(p+q)$ intervals belonging to A, and exactly $(n-i)(p+q)$ intervals belonging to B. In addition, there are at most $2(n-i)$ link intervals intersecting C_j^l to the left (these are the link intervals related to $v_{i''}$ for $i'' > i$ in the $(j-1)$-th region, if $j > 1$), exactly $2(n-i)$ link intervals intersecting C_j^l to the right (these are the link intervals related to $v_{i''}$ for $i'' > i$ in the j-th region), and exactly $2i$ link intervals covering C_j^l (these are the link intervals related to $v_{i''}$ for $i'' \leq i$ in the j-th region). This amounts to at most $2(n-i) + 2(n-i) + 2i = 4n - 2i < 4n$ intervals. Thus, if $C_j^l \in A$, then there are less than $(n-i)(p+q) + 4n + q'$ edges in the cut-set incident to C_j^l; while if $C_j^l \in B$, then there are at least $(n-i)(p+q) + q$ edges in the cut-set incident to C_j^l. Therefore, since $q > 4n + p' + q' \geq 4n + q'$, C_j^l must belong to B.

We note that an analogous argument can be applied to the intervals C_j^3, C_j^4, with respect to the grained gadget $\mathcal{H}_{i'}^j$. Indeed, suppose without loss of generality that

$\mathcal{H}_{i'}^{j}$ is A-partitioned, and consider $l \in \{3,4\}$. If $C_j^l \in A$, then there are less than $(n-i')(p+q)+4n+p'+q'$ edges in the cut-set incident to C_j^l; while if $C_j^l \in B$, then there are at least $(n-i')(p+q)+q$ edges in the cut-set incident to C_j^l. Therefore, since $q > 4n+p'+q'$, C_j^l must belong to B. By Corollary 3.1, this additionally implies that $\mathcal{E}_j$ must be A-partitioned. □

Informally, it follows from Lemma 3.3 that, in any maximum cut of $\mathbb{G}_{\mathcal{M}(\mathfrak{G})}$, the partitioning of every grained gadget, with its incidence intervals, behaves well within each region individually. Nonetheless, note that, by definition, a cut $[A,B]$ being locally well-behaved does not necessarily imply that, in this cut, the partitioning of the intervals related to a region influences the partitioning of the intervals related to neighbouring regions. In order to make it possible to associate maximum cuts of $\mathbb{G}_{\mathcal{M}(\mathfrak{G})}$ with maximum cuts of G', it is necessary that all grained gadgets corresponding to a vertex v_i of G agree with one another. In other words, for each $j \in [m]$, the partitioning of $\mathcal{H}_i^j$ must influence in a well-defined way the partitioning of $\mathcal{H}_i^{j+1}$.

Let $[A,B]$ be a cut of $\mathbb{G}_{\mathcal{M}(\mathfrak{G})}$. Given a vertex $v_i \in V(G)$, we say that *the gadgets of v_i alternate in* $[A,B]$ if, for every $j \in [m]$, it holds that $\mathcal{H}_i^j$ is A-partitioned if and only if $\{L_i^{2j-1}, L_i^{2j}\} \subseteq B$ and $\mathcal{H}_i^{j+1}$ is B-partitioned. Figure 3.10 illustrates this notion. In addition, we say that $[A,B]$ is *alternating partitioned* if, for every vertex $v_i \in V(G)$, the gadgets of v_i alternate in $[A,B]$. Finally, we say that $[A,B]$ is *globally well-behaved* if it is simultaneously alternating partitioned and locally well-behaved. In the following lemma, a further condition is imposed on the values of p, q, p' and q' in order to ensure that every maximum cut of $\mathbb{G}_{\mathcal{M}(\mathfrak{G})}$ is globally well-defined.

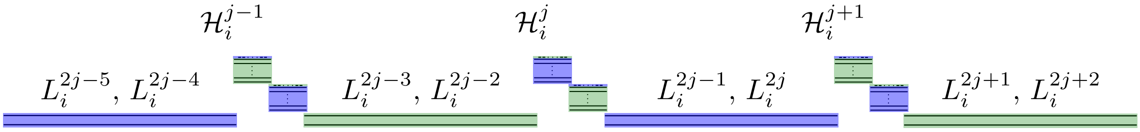


Figure 3.10: A cut $[A,B]$ of $\mathbb{G}_{\mathcal{M}(\mathfrak{G})}$ in which the gadgets of a vertex $v_i \in V(G)$ alternate. Each colour represents a part of the cut.

Lemma 3.4. *Let G be a cubic graph, $\pi_V = (v_1,\ldots,v_n)$ and $\pi_E = (e_1,\ldots,e_m)$ be orderings of $V(G)$ and of $E(G)$, respectively, and $\mathfrak{G} = (G,\pi_V,\pi_E)$. If $[A,B]$ is a maximum cut of $\mathbb{G}_{\mathcal{M}(\mathfrak{G})}$, $\mathcal{M}(\mathfrak{G})$ is well-valued, $q > 4n+p'+q'$, and, in addition, $q > 3(2n^2+n+q'+2)$, then $[A,B]$ is globally well-behaved.*

Proof. By hypothesis, the conditions of Lemmas 3.1 and 3.3 are satisfied. Thus, we can assume that every maximum cut of $\mathbb{G}_{\mathcal{M}(\mathfrak{G})}$ is locally well-behaved. Based on that, we note that, for some pairs of subsets $\mathcal{S}$ and $\mathcal{S}'$ of $\mathcal{M}(\mathfrak{G})$, the number of edges between $\mathcal{S}$ and $\mathcal{S}'$ in the cut-set of any maximum cut $[A',B']$ of $\mathbb{G}_{\mathcal{M}(\mathfrak{G})}$ is always the same, regardless of whether $[A',B']$ is alternating partitioned or not. Thus, to the context of alternating partitioned cuts, the edges between $\mathcal{S}$ and $\mathcal{S}'$ are *irrelevant*. For instance, every (x,y)-grained gadget H of $\mathbb{G}_{\mathcal{M}(\mathfrak{G})}$ has exactly $x+y$ intervals in either part of a maximum cut of $\mathcal{M}(\mathfrak{G})$. Hence, the edges between H and any subset $\mathcal{C}$ of intervals covering H are irrelevant, since there always are exactly $(x+y)\cdot|\mathcal{C}|$ such edges in the cut-set of any maximum cut of $\mathcal{M}(\mathfrak{G})$.

For each $i \in [n]$, let $\mathcal{M}_i$ denote the set of intervals in $\mathcal{M}(\mathfrak{G})$ related to the vertex v_i; that is to say, $\mathcal{M}_i$ consists of every interval in $\mathcal{H}_i^j$ for $j \in [m+1]$, every link interval L_i^j for $j \in [2m]$, every interval in $\mathcal{E}_j$ for e_j incident to v_i in G, and every incidence interval C_j^l weakly intersecting $\mathcal{H}_i^j$. Additionally, for each $i \in [n]$, let f_i denote the number of *relevant* edges in the cut-set of $[A,B]$ that are incident to some interval in $\mathcal{M}_i$. In what follows, we determine an upper bound for f_i, and based on that we argue that, if the gadgets of v_i do not alternate in $[A,B]$, then, by rearranging the partitioning of $\mathcal{M}_i$, it is possible to obtain a cut of larger size, contradicting therefore the maximality of $[A,B]$.

First, for each $i \in [n]$ and each $j \in [m]$, we count the maximum possible number of edges in the cut-set that are incident to the link interval L_i^l, where $l \in \{2j-1, 2j\}$. Let $\lambda_{i,l}^A$ denote the number of link intervals in $\mathcal{M}(\mathfrak{G}) \cap A \setminus \{L_i^l\}$ that intersect L_i^l; define $\lambda_{i,l}^B$ similarly. Note that, in addition to $L_i^{l'}$ for $l' \in \{2j-1,2j\}\setminus\{l\}$, the link intervals that intersect L_i^l in $\mathcal{M}(\mathfrak{G})$ are precisely: $L_\iota^{2j-1}, L_\iota^{2j}$ for $\iota \in [n] \setminus \{i\}$, $L_\iota^{2j-3}, L_\iota^{2j-2}$ for $\iota \in \{i+1,\dots,n\}$, and $L_\iota^{2j+1}, L_l^{2j+2}$ for $\iota \in [i-1]$. Thus, $\lambda_{i,l}^A + \lambda_{i,l}^B < 4n-2$. Let $z_{i,l} \in \{0,1\}$, such that $z_{i,l} = 1$ if and only if L_i^l and the long intervals in $K''(\mathcal{H}_i^j)$ are in opposite parts of the cut; and, let $z'_{i,l} \in \{0,1\}$, such that $z'_{i,l} = 1$ if and only if L_i^l and the long intervals in $K'(\mathcal{H}_i^{j+1})$ are in opposite parts of the cut. Finally, note that L_i^l and the incidence intervals $C_j^1, \dots, C_j^4$ might also be in opposite parts of the cut. Hence, in total, the number of relevant edges in the cut-set that are incident to L_i^l is at most $q(z_{i,l} + z'_{i,l}) + \lambda_{i,l}^A + \lambda_{i,l}^B + 4$.

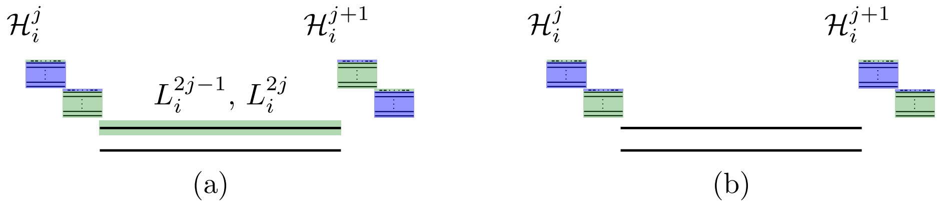
Now, for each $i \in [n]$ and each $j \in [m]$, with e_j incident to v_i in G, we count the number of relevant edges in the cut-set that are incident to the intervals in $\mathcal{M}_i \cap (\mathcal{E}_j \cup \{C_j^1, \dots, C_j^4\})$. We note that the edges between $\{C_j^3, C_j^4\}$ and the short intervals in $S'(\mathcal{E}_j)$ are irrelevant, since $\mathcal{E}_j$ is always partitioned according to C_j^3, C_j^4. Thus, suppose without loss of generality that $\{C_j^1, C_j^2\} \subseteq A$. If $\{C_j^3, C_j^4\} \subseteq A$, then one can verify that there are no relevant edges in the corresponding cut-set that are incident to the intervals in $\mathcal{M}_i \cap (\mathcal{E}_j \cup \{C_j^1, \dots, C_j^4\})$. On the other hand, if $\{C_j^3, C_j^4\} \subseteq B$, then there are exactly $2q'+4$ relevant edges, namely: the edges between $\{C_j^1, C_j^2\}$ and $\{C_j^3, C_j^4\}$, and the edges between $\{C_j^1, C_j^2\}$ and the long intervals in $K'(\mathcal{E}_j)$. Finally, we note that the edges between $\{C_j^1, \dots, C_j^4\}$ and $\mathcal{H}_i^j$ are irrelevant, and that the edges between $\{C_j^1, \dots, C_j^4\}$ and the link intervals have already been counted.

Thus, putting everything together, let e_{j_1}, e_{j_2}, e_{j_3} be the three edges incident to v_i in G, and for each $h \in \{1,2,3\}$, let $c_{i,h} \in \{0,1\}$, such that $c_{i,h} = 1$ if and only if $\mathcal{H}_i^j$ and $\mathcal{H}_{i_h}^j$ are partitioned differently (*e.g.*, $\mathcal{H}_i^j$ is A-partitioned while $\mathcal{H}_{i_h}^j$ is B-partitioned), where v_{i_h} denotes the other endpoint of e_{j_h}, apart from v_i. Then, we obtain that

$$f_i \leq \sum_{j=1}^{m} \sum_{l=2j-1}^{2j} (q(z_{i,l} + z'_{i,l}) + \lambda_{i,l}^A + \lambda_{i,l}^B + 4) + \sum_{h=1}^{3} c_{i,h}(2q' + 4). \tag{3.1}$$

Now, based on equation (3.1), we finally show that $[A, B]$ must be alternating partitioned, otherwise it would not be maximum.

Suppose that, for some $i \in [n]$ and some $j \in [m]$, $\mathcal{H}_i^j$ and $\mathcal{H}_i^{j+1}$ are partitioned differently by $[A, B]$, but, for some $l \in \{2j-1, 2j\}$, the link interval L_i^l belongs to the same part of the cut as the long intervals in $K''(\mathcal{H}_i^j)$ and in $K'(\mathcal{H}_{i+1}^j)$; say $\mathcal{H}_i^j$ is A-partitioned, $\mathcal{H}_i^{j+1}$ is B-partitioned, but $L_i^l \in A$ (see Figure 3.11a). In this case, f_i can be increased by simply switching the part of L_i^l. Indeed, compared to $[A, B]$, the resulting cut loses at most $\max\{\lambda_{i,l}^A, \lambda_{i,l}^B\} + 4 < 4n + 2$ edges, whereas it gains $2q$ new edges, namely: the edges between L_i^l and the long intervals in $K''(\mathcal{H}_i^j) \cup K'(\mathcal{H}_{i+1}^j)$. This is a positive exchange, since by hypothesis $q > 4n$. Therefore, in what follows, assume that the link intervals L_i^{2j-1}, L_i^{2j} and the long intervals in $K''(\mathcal{H}_i^j) \cup K'(\mathcal{H}_{i+1}^j)$ belong to opposite parts of the cut $[A, B]$, whenever $\mathcal{H}_i^j$ and $\mathcal{H}_i^{j+1}$ are partitioned differently.



(a)

(b)

Figure 3.11: (a) $\mathcal{H}_i^j$ is A-partitioned, $\mathcal{H}_i^{j+1}$ is B-partitioned, but $L_i^l \in A$ for some $l \in \{2j-1, 2j\}$. (b) $\mathcal{H}_i^j$ and $\mathcal{H}_i^{j+1}$ are both A-partitioned. The green colour represents the part A, while the blue colour represents the part B.

Finally, suppose that, for some $i \in [n]$ and some $j \in [m]$, $\mathcal{H}_i^j$ and $\mathcal{H}_i^{j+1}$ are partitioned in the same way by $[A, B]$; say they are both A-partitioned (see Figure 3.11b). Assume that j is the minimum integer in $[m]$ satisfying this condition. Let j' be the minimum integer in $\{j+1, \ldots, m\}$ such that $\mathcal{H}_i^{j'}$ and $\mathcal{H}_i^{j'+1}$ are also partitioned in the same way by $[A, B]$, *i.e.* they are either both A-partitioned or both B-partitioned; if it does not exist, let $j' = m + 1$. Note that, if $j' < m$, then we can assume based on the previous paragraph that the link intervals $L_i^{2j'-1}, L_i^{2j'}$ and the long intervals in $K'(\mathcal{H}_i^{j'+1})$ are in opposite parts of the cut, since, by the definition of j', $\mathcal{H}_i^{j'+1}$ and $\mathcal{H}_i^{j'+2}$ are partitioned differently by $[A, B]$. Then, let $[A', B']$ be the cut obtained from $[A, B]$ by switching of parts the intervals in $\mathcal{M}_i$ related to the h-region for each $h \in \{j+1, \ldots, j'\}$, and placing the link intervals L_i^{2j-1}, L_i^{2j}

in the opposite part of the long intervals in $K''(\mathcal{H}_i^j)$. More formally, $[A', B']$ is the cut obtained from $[A, B]$ such that, except for satisfying the following conditions, A' and B' are defined exactly as A and B, respectively.

- $\{L_i^{2j-1}, L_i^{2j}\} \subseteq B'$;
- For each $h \in \{j+1, \ldots, j'\}$, if $\mathcal{H}_i^h$ is A-partitioned by $[A, B]$, then $\mathcal{H}_i^h$ is B'-partitioned by $[A', B']$; otherwise, $\mathcal{H}_i^h$ is A'-partitioned by $[A', B']$;
- For each $h \in \{j+1, \ldots, \min\{j', m\}\}$, if $\mathcal{H}_i^h$ is A-partitioned by $[A, B]$, then $\{L_i^{2h-1}, L_i^{2h}\} \subseteq A'$; otherwise, $\{L_i^{2h-1}, L_i^{2h}\} \subseteq B'$;
- For each $h \in \{j+1, \ldots, \min\{j', m\}\}$ with $e_h = v_i v_{i'}$ and $i < i'$, if $\{C_h^1, C_h^2\} \subseteq B$, then $\{C_h^1, C_h^2\} \subseteq A'$; otherwise $\{C_h^1, C_h^2\} \subseteq B'$;
- For each $h \in \{j+1, \ldots, \min\{j', m\}\}$ with $e_h = v_{i'} v_i$ and $i > i'$, if $\{C_h^3, C_h^4\} \subseteq B$, then $\{C_h^3, C_h^4\} \subseteq A'$ and $\mathcal{E}_j$ is B'-partitioned by $[A', B']$; otherwise $\{C_h^3, C_h^4\} \subseteq B'$ and $\mathcal{E}_j$ is A'-partitioned by $[A', B']$.

Note that $[A', B']$ is locally well-behaved. Moreover, one can verify that, compared to $[A, B]$, the resulting cut $[A', B']$ gains at least $2q$ new edges, since now the long intervals in $K''(\mathcal{H}_i^j)$ and in $K'(\mathcal{H}_i^{j+1})$ belong to the same part of the cut, which is opposite to the part containing the link intervals L_i^{2j-1} and L_i^{2j}. Next, we prove that $[A', B']$ loses at most $6(2n^2+n) + 6(q'+2) = 6(2n^2+n+q'+2)$ edges, contradicting therefore the maximality of $[A, B]$ since $q > 3(2n^2+n+q'+2)$.

We recall that we are supposing that $\mathcal{H}_i^j$ is A-partitioned. Thus, concerning the link intervals L_i^{2j-1} and L_i^{2j}, in the worst case, we have that $\{L_i^{2j-1}, L_i^{2j}\} \subseteq A$ and $\{C_j^1 \ldots, C_j^4\} \subseteq A$, and then $[A', B']$ might lose at most

$$\textstyle\sum_{l=2j-1}^{2j}(\lambda_{i,l}^B + 4) \leq \sum_{l=2j-1}^{2j}(\max\{\lambda_{i,l}^A, \lambda_{i,l}^B\} + 4) < 8n + 4$$

edges. As for the intervals L_i^{2h-1}, L_i^{2h} for $h \in \{j+1, \ldots, j'\}$, the cut $[A', B']$ might also lose at most $\sum_{l=2h-1}^{2h}(\max\{\lambda_{i,l}^A, \lambda_{i,l}^B\} + 4) < 8n+4$, while, by the definition of j', the number of edges in the cut-set between such link intervals and the vertex grained gadgets can only increase. Therefore, concerning the link intervals in $\mathcal{M}_i$, $[A', B']$ loses in total at most $m(8n+4) = 6(2n^2+n)$ edges. Additionally, based on the upper bound (3.1) for f_i, we have that in the worst case $\{j_1, j_2, j_3\} \subseteq \{j+1, \ldots, j'\}$ and the values of $c_{i,1}, c_{i,2}, c_{i,3}$ are all equal to 1 with respect to $[A, B]$ (*i.e.* all the three edges e_{j_1}, e_{j_2}, e_{j_3} incident to v_i belong to the cut-set of $[A, B]$), but with respect to $[A', B']$ they are all equal to 0 (i.e. none of the three edges e_{j_1}, e_{j_2}, e_{j_3} incident to v_i belong to the cut-set of $[A', B']$). This leads to a loss of at most $6(q'+2)$ edges. □

Now, if $[X, Y]$ is a cut of G, then we let $\Phi(X, Y) = [A, B]$ be the globally well-behaved cut of $\mathbb{G}_{\mathcal{M}(\mathfrak{G})}$ such that, for each vertex $v_i \in V(G)$,

$$v_i \in X \text{ if and only if } \mathcal{H}_i^1 \text{ is } A\text{-partitioned by } [A, B].$$

Note that $[A, B]$ is well-defined and uniquely determined by $[X, Y]$. On the other hand, given a globally well-behaved cut $[A, B]$ of $\mathbb{G}_{\mathcal{M}(\mathfrak{G})}$, there is a unique cut $[X, Y] = \Phi^{-1}(A, B)$ of G such that $[A, B] = \Phi(X, Y)$. Therefore, Φ is a bijection.

Next lemma establishes that, for q' large enough, the size of a globally well-behaved cut $[A, B]$ grows as a function of the size of $\Phi^{-1}(A, B)$. We present the general idea behind the proof of this result, and we refer the reader to Lemma 5 of Appendix E for a thorough proof.

Lemma 3.5. *Let G be a cubic graph on n vertices, π_V and π_E be orderings of $V(G)$ and of $E(G)$, respectively, and $\mathfrak{G} = (G, \pi_V, \pi_E)$. For every positive integer k, if $[A, B]$ is a globally well-behaved cut of $\mathbb{G}_{\mathcal{M}(\mathfrak{G})}$ and $q' \geq 13n^2$, then*

$$|E_G(X, Y)| \geq k \text{ if and only if } |E_{\mathbb{G}}(A, B)| \geq \phi(n, k),$$

where $\mathbb{G}$ denotes $\mathbb{G}_{\mathcal{M}(\mathfrak{G})}$, $[X, Y] = \Phi(A, B)$ and ϕ is a well-defined function.

Proof sketch. Since $[A, B]$ is globally well-behaved, one can verify that the following statements hold, with respect to $E_{\mathbb{G}}(A, B)$, where m denotes the number of edges of G, *i.e.* $m = 3n/2$.

- Among intervals of vertex/edge grained-gadgets, there are exactly $\beta_1 = n(m + 1)(q^2 + 2pq) + m((q')^2 + 2p'q')$ edges in the cut-set.

- Between intervals of a vertex grained-gadget and link intervals, there are exactly $\beta_2 = 2mn[n(p + q) + q - p]$ edges in the cut-set.

- Between intervals of an edge grained-gadget and any other intervals, and among incidence intervals, there are exactly $\beta_3 + (2q' + 4) \cdot |E_G(X, Y)|$ edges in the-cut set, where $\beta_3 = 2nm(p' + q') + 2p'm$.

- Among link intervals, there are exactly $\beta_4 + 4|X||Y|$ edges in the cut-set, where $\beta_4 = n(n - 1)(3n - 2)$. Note that, $4(n - 1) \leq 4|X||Y| \leq n(n + 1)$.

- Between intervals of a vertex grained-gadget and incidence intervals, there are exactly
$$\begin{aligned} \beta_5 = {} & 4m[n(p + q) + q] - 2(p + q) \textstyle\sum_{j=1}^{m} (j_i + j_{i'}) \\ \leq {} & 4m[n(p + q) + q] \end{aligned}$$
edges in the cut-set, where $e_j = v_{j_i} v_{j_{i'}}$ for each $j \in [m]$.

- Finally, between link intervals and incidence intervals, there at most $16nm = 24n^2$ edges in the cut-set.

By the construction of $\mathbb{G}$, one can additionally that the edge set of $\mathbb{G}$ can be partitioned into the groups described above.

Let $\beta = \sum_{i=1}^{5} \beta_i$ and $\gamma = \beta + 4(n-1)$. We have just shown that

$$\begin{aligned} \overbrace{\beta + 4(n-1)}^{\gamma} + (2q' + 4)|E_G(X,Y)| \leq & \ |E_{\mathbb{G}}(A,B)| \\ \leq & \ \beta + n(n+1) + 24n^2 + (2q'+4)|E_G(X,Y)|. \end{aligned}$$

If $|E_G(X,Y)| \geq k$, then it follows from the first inequality that

$$|E_{\mathbb{G}}(A,B)| \geq \beta + 4(n-1) + (2q'+4)k.$$

Conversely, if $|E_{\mathbb{G}}(A,B)| \geq \beta + 4(n-1) + (2q'+4)k$, then it follows from the second inequality that

$$|E_G(X,Y)| \geq k - \frac{25n^2 - 3n + 4}{2q' + 4} \geq k - \frac{26n^2}{2q'+4}.$$

Since $q' \geq 13n^2$, $|E_G(X,Y)| > k - 1$. Finally, we note that p, q, p' and q' are defined depending exclusively on n. Therefore, by setting $\phi(n,k) = \gamma + (2q'+4)k$, we obtain that $|E_G(X,Y)| \geq k$ if and only if $|E_{\mathbb{G}}(A,B)| \geq \phi(n,k)$. □

To finish the proof that the reduction works, we need to choose appropriate values for p, q, p' and q'. Recall all necessary conditions:

- For each (x,y)-grained gadget $\mathcal{H}$ in $\mathcal{M}$, if ℓ is the number of intervals in $\mathcal{M}$ intersecting the short intervals in $S'(\mathcal{H})$, and r is the number of intervals in $\mathcal{M}$ intersecting the short intervals $S'(\mathcal{H})$, then ℓ and r must both be odd (from Lemma 3.1); moreover, $y > 2t$ and $x > t + 2y$ (from Lemma 3.1), where t denotes number of intervals in $\mathcal{M} \setminus \mathcal{H}$ intersecting $\mathcal{H}$;

- $q > 4n + p' + q' = \alpha_1$ (from Lemma 3.3);

- $q > 3(2n^2 + n + q' + 2) = \alpha_2$ (from Lemma 3.4); and

- $q' \geq 13n^2$ (from Lemma 3.5).

It follows from Lemma 3.2 that, in order for r and ℓ to be odd, it suffices to define q and q' odd. Moreover, we note that $n \geq 4$, since G is a cubic graph. For a edge grained gadget $\mathcal{E}_j$, there are exactly $2n+4$ intervals in $\mathcal{M} \setminus \mathcal{E}_j$ intersecting it. Hence, if we define $q' = 13n^2 + 1$, all the conditions on q' are satisfied, since $13n^2 + 1 > 4n + 8$. As for p', it suffices to define $p' = 2n + 4 + 2q' + 1 = 26n^2 + 2n + 7$.

Now, for a vertex grained gadget $\mathcal{H}_i^j$, there are at most $2(n-1)+8 = 2n+6$ intervals in $\mathcal{M} \setminus \mathcal{H}_i^j$ intersecting it. Thus, in order to satisfy the conditions of Lemma 3.1, it suffices to ensure $q > 4n+12$ and $p > 2q+2n+6$. Considering the chosen values for p' and q' above, we obtain that

$$\alpha_1 = 4n + p' + q' = 39n^2 + 6n + 8 \text{ and } \alpha_2 = 3(2n^2 + n + q' + 2) = 45n^2 + 3n + 9.$$

Moreover, one can verify that $47n^2 + 1 > \max\{4n+12, \alpha_1, \alpha_2\}$ for $n \geq 4$. Hence, it suffices to define $q = 47n^2 + 1$ and $p = 2(47n^2+1) + 2n + 6 + 1 = 94n^2 + 2n + 9$.

In the next section, we prove that the interval count of our reduction graph is exactly 4, concluding therefore the proof of Theorem 3.1.

3.4.3 Bounding the Interval Count

Now, we finally prove that, for every cubic graph G and every pair of orderings π_V, π_E of $V(G)$ and of $E(G)$, respectively, there exists an interval model for (G, π_V, π_E), satisfying the conditions described in Section 3.4.1, whose interval count is bounded.

Upper bound

Let G be a cubic graph on n vertices and $m = 3n/2$ edges, $\pi_V = (v_1, \ldots, v_n)$ and $\pi_E = (e_1, \ldots, e_m)$ be orderings of $V(G)$ and of $E(G)$, respectively, and let $\mathfrak{G}$ denote the triple (G, π_V, π_E). We note that there exist infinitely many interval models (possibly of distinct interval counts) satisfying the conditions described in Section 3.4.1 for $\mathfrak{G}$. In other words, $\mathcal{M}(\mathfrak{G})$ is not uniquely determined and, actually, can be seem as a meta interval model. In what follows, we provide a construction of an instance $\mathcal{M}'$ of $\mathcal{M}(\mathfrak{G})$ that has interval count 4.

For each $j \in [m]$, let $\mathcal{S}_j$ denote the set of intervals related to the j-th region, i.e.,

$$\mathcal{S}_j = \mathcal{E}_j \cup \bigcup_{\ell=1}^{4} C_j^\ell \cup \bigcup_{i=1}^{n} (\mathcal{H}_i^j \cup \{L_i^{2j} \cup L_i^{2j-1}\}).$$

Next, we show how to accommodate $\mathcal{S}_j$ within the closed interval $[t_j, 6n-2+t_j]$, where $t_j = 4n \cdot (j-1)$. Note that t_j acts as a shifting function. Figure 3.12 exemplifies the closed intervals in $\mathcal{S}_1 \cup \bigcup_{i=1}^{4} \mathcal{H}_i^2$ of a graph on 4 vertices.

For each each $l \in [n]$ and each $j \in [m+1]$, the intervals composing the vertex grained gadget $\mathcal{H}_l^j$ are defined as follows:

- The long intervals in $K'(\mathcal{H}_l^j)$ are equal to $[2l-2+t_j, 2l-\frac{3}{2}+t_j]$ and the short intervals in $S'(\mathcal{H}_l^j)$ are any choice of q distinct points within the corresponding open interval $(2l-2+t_j, 2l-\frac{3}{2}+t_j)$;

- The long intervals in $K''(\mathcal{H}_l^j)$ are equal to $[2l-\frac{3}{2}+t_j, 2l-1+t_j]$ and the short intervals in $S''(\mathcal{H}_l^j)$ are any choice of q distinct points within the corresponding open interval $(2l-\frac{3}{2}+t_j, 2l-1+t_j)$.

For each $j \in [m]$, with $e_j = v_i v_{i'}$ and $i < i'$, the intervals $C_j^1, C_j^2, C_j^3, C_j^4$, the intervals composing the edge grained gadget $\mathcal{E}_j$, and the link intervals are defined as follows, respectively:

- The intervals C_j^1 and C_j^2 are equal to $[2i-1+t_j, 2i+2n-2+t_j]$, and the intervals C_j^3 and C_j^4 are equal to $[2i'-1+t_j, 2i'+2n-2+t_j]$;
- The long intervals in $K'(\mathcal{E}_j)$ are equal to $[2n+t_j, 4n-1+t_j]$ and the short intervals in $S'(\mathcal{E}_j)$ are any choice of p' distinct points in the open interval $(2i+2n-2+t_j, 2i'+2n-2+t_j)$;
- The long intervals in $K''(\mathcal{E}_j)$ are equal to $[4n-1+t_j, 4n-\frac{1}{2}+t_j]$ and the short intervals in $S''(\mathcal{E}_j)$ are any choice of p' distinct points within the corresponding open interval $(4n-1+t_j, 4n-\frac{1}{2}+t_j)$;
- For each $l \in [n]$, the intervals L_l^{2j-1}, L_l^{2j} are equal to $[2l-1+t_j, 4n+2(l-1)+t_j]$.

We remark that the open intervals described above are only used to locate the closed intervals of length zero, but that the short intervals themselves are not open.

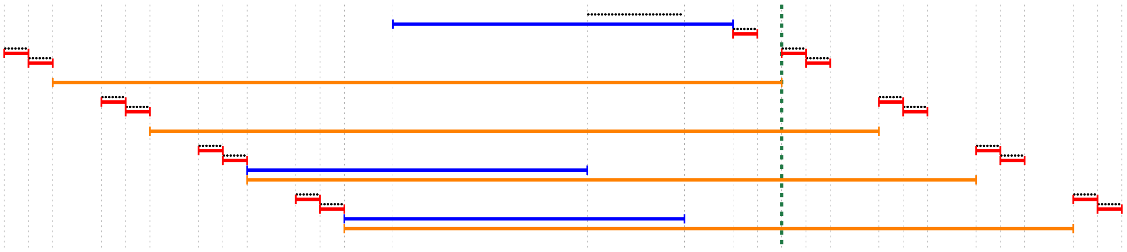


Figure 3.12: The closed intervals in $\mathcal{S}_1 \cup \bigcup_{i=1}^4 \mathcal{H}_i^2$ of a graph on 4 vertices. In this example, we consider e_1 to be equal to v_3v_4. Each colour represents a different interval size. The short intervals are represented by the dots located inside the open interval. Vertical lines mark the endpoints of the intervals in $\mathcal{S}_1$, while the green vertical line marks the beginning of the intervals in $\mathcal{S}_2$.

One can verify that the closed intervals defined previously have the the following lengths (see Figure 3.12).

1. Length 0: short intervals of all grained gadgets (dots in Figure 3.12);
2. Length 1/2: long intervals of each vertex grained gadget $\mathcal{H}_i^j$, and long intervals in $K''(\mathcal{E}_j)$ for each $j \in [m]$ (red intervals in Figure 3.12);

3. Length $2n-1$: intervals $C_j^1, C_j^2, C_j^3, C_j^4$ and long intervals in $K'(\mathcal{E}_j)$ for each $j \in [m]$ (blue intervals in Figure 3.12);

4. Length $4n-1$: intervals L_i^{2j-1}, L_i^{2j} for each $i \in [n]$ and each $j \in [m]$ (orange intervals in Figure 3.12).

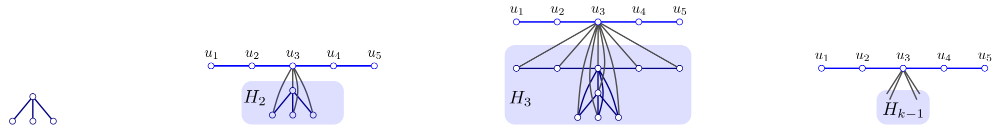
As a result, the resulting interval model $\mathcal{M}'$ clearly has interval count 4. It remains to prove that $\mathcal{M}'$ is an instance of $\mathcal{M}(\mathfrak{G})$, *i.e.* that $\mathcal{M}'$ satisfies the conditions imposed in Section 3.4.1. This is achieved through Lemma 3.6, whose proof is omitted and deferred to Lemma 6 in Appendix E.

Lemma 3.6. *Let G be a cubic graph. Then, there exists an interval model $\mathcal{M}(\mathfrak{G})$ of interval count 4 for $\mathfrak{G} = (G, \pi_V, \pi_E)$, for every ordering π_V and every ordering π_E of the vertex set and edge set of G, respectively.*

Lower bound

We have just shown that, for every cubic graph G and every pair of orderings π_V and π_E, there exists an interval model $\mathcal{M}(\mathfrak{G})$ of interval count 4, where $\mathfrak{G}$ denotes the triple (G, π_V, π_E). On the other hand, we prove in the remainder of this section that any graph isomorphic to $\mathbb{G}_{\mathcal{M}(\mathfrak{G})}$ has interval count at least 4. For this, we show that all such graphs contain as an induced subgraph a certain graph, which we denote by H_4, of interval count exactly 4. Next, we define the family $\{H_k\}_{k\geq 2}$.

Let $P_5 = (u_1, \ldots, u_5)$ be a path on 5 vertices. For every graph H', we let $P_5 \circ H'$ be the graph obtained from the disjoint union of P_5 with H' by making u_3, the central vertex of P_5, adjacent to every vertex of H'. In other words, $P_5 \circ H'$ is the graph with vertex set $V(P_5) \cup V(H')$ and edge set $E(P_5) \cup E(H') \cup \{u_3 v \mid v \in V(H')\}$. Then, for every $k \geq 2$, we let H_k be the graph defined recursively as follows (see Figure 3.13): $H_2 = K_{1,3}$; and $H_k = P_5 \circ H_{k-1}$ for $k > 2$.



(a) $H_2 = K_{1,3}$ (b) $H_3 = P_5 \circ H_2$ (c) $H_4 = P_5 \circ H_3$ (d) $H_k = P_5 \circ H_{k-1}$

Figure 3.13: Graph H_k for $k \geq 2$.

By induction on k, and based on the fact that H_2 has interval count exactly 2 *cf.* [108], one can prove that $\mathsf{ic}(H_k) = k$ for every $k \geq 2$. We refer the reader to Lemma 8 in Appendix E for the complete proof of this result.

Now, we finally show that, if $\mathbb{G}'$ is a graph isomorphic to our reduction graph $\mathbb{G}_{\mathcal{M}(\mathfrak{G})}$, then $\mathbb{G}'$ has an H_4 as an induced subgraph. Then, let $\mathcal{M}'$ be an interval model of $\mathbb{G}'$. Note that $\mathcal{M}'$ exhibits the same intersection properties of $\mathcal{M}(\mathfrak{G})$

described in Section 3.4.1, otherwise $\mathbb{G}'$ would not be isomorphic to $\mathbb{G}_{\mathcal{M}(\mathfrak{G})}$. Consequently, there exist orderings π'_V and π'_E of the vertex set and edge set of G, respectively, for which $\mathcal{M}'$ can be defined as a composition of vertex grained gadgets $\mathcal{H}_i^j$ for $i \in [n]$ and $j \in [m+1]$, edge grained gadgets $\mathcal{E}_j$ for $j \in [m]$, link intervals L_i^{2j-1}, L_i^{2j} for $i \in [n]$ and $j \in [m]$, and intervals $C_j^1, C_j^2, C_j^3, C_j^4$ for $j \in [m]$. Additionally, since the input graph G is cubic, there exists an edge $e_j = (v_i, v_{i'}) \in E(G)$ such that, with respect to π'_V and π'_E, $1 < i < i'$. Thus, let (see Figure 3.8):

- I_1 and I_2 be intervals in $S''(\mathcal{H}_1^j)$ and in $K''(\mathcal{H}_1^j)$, respectively;
- I_3 be the link interval L_1^{2j-1};
- I_4 and I_5 be intervals in $K'(\mathcal{H}_1^{j+1})$ and in $S'(\mathcal{H}_1^{j+1})$, respectively;
- I'_1 and I'_2 be intervals in $S''(\mathcal{H}_i^j)$ and in $K''(\mathcal{H}_i^j)$, respectively;
- I'_3 be the interval C_j^1;
- I'_4 and I'_5 be intervals in $K'(\mathcal{E}_j)$ and in $S'(\mathcal{E}_j)$, respectively;
- J_1, J_2 and J_3 be three distinct intervals in $S'(\mathcal{H}_{i+1}^j)$; and
- J be an interval in $K'(\mathcal{H}_{i+1}^j)$.

We note that the intervals described above exist and are well-defined in $\mathcal{M}'$. Moreover, the interval graph related to the model comprised by such intervals is isomorphic to H_4. More specifically, observe first that $\mathcal{J} = \{J, J_1, J_2, J_3\}$ models $K_{1,3}$. Then, note that $\mathcal{P} = \{I_1, \ldots, I_5\}$ and $\mathcal{P}' = \{I'_1, \ldots, I'_5\}$ model paths on 5 vertices, in this order. Finally observe that I'_3 is adjacent to every $I \in \mathcal{J}$, while there are no edges between $\mathcal{J}$ and $\mathcal{P}' \setminus \{I'_3\}$; hence, $\mathcal{J} \cup \mathcal{P}'$ is a model for H_3. Similarly, I_3 is adjacent to every $I \in \mathcal{J} \cup \mathcal{P}'$, while there are no edges between $\mathcal{J} \cup \mathcal{P}'$ and $\mathcal{P} \setminus \{I_3\}$; hence $\mathcal{J} \cup \mathcal{P}' \cup \mathcal{P}$ is a model for H_4. Therefore, $\mathbb{G}'$ has an H_4 as an induced subgraph, as we wanted to prove.

3.5 Permutation Graphs

In this section, we prove that MaxCut also remains NP-complete on permutation graphs, settling a long-standing open problem from the *Ongoing Guide to NP-completeness* by David S. Johnson [83]. Based on Adhikary et al.'s construction [1], through a polynomial-time reduction from MaxCut on cubic graphs, we prove the following theorem:

Theorem 3.2. MaxCut *is* NP-*complete on permutation graphs.*

3.5.1 Reduction Graph

Let G be a cubic graph on n vertices and $m = 3n/2$ edges, and let $\pi_V = (v_1, \dots, v_n)$ and $\pi_E = (e_1, \dots, e_m)$ be orderings of $V(G)$ and $E(G)$, respectively.

The values of p, q, p', q' used next are not the same as in Sections 3.3 and 3.4, and they are properly defined later. For each vertex $v_i \in V(G)$, create a (p, q)-grained gadget, $\mathcal{H}_i$, and for each edge $e_j \in E(G)$, create a (p', q')-grained gadget $\mathcal{E}_j$.

For simplicity, in the remainder of this section, we write S'_i, S''_i, K'_i and K''_i to denote the sets $S'(\mathcal{H}_i)$, $S''(\mathcal{H}_i)$, $K'(\mathcal{H}_i)$ and $K''(\mathcal{H}_i)$, respectively. Similarly, we write S'_{e_j}, S''_{e_j}, K'_{e_j} and K''_{e_j} to denote the sets $S'(\mathcal{E}_j)$, $S''(\mathcal{E}_j)$, $K'(\mathcal{E}_j)$ and $K''(\mathcal{E}_j)$, respectively. We recall that, for each $i \in [n]$, the permutation model of $\mathcal{H}_i$ consists in the pair of permutations $\{\pi_i^1, \pi_i^2\}$, where $\pi_i^1 = K'_i S'_i S''_i K''_i$ and $\pi_i^2 = S'_i \overleftarrow{K''_i} \overleftarrow{K'_i} S''_i$. Analogously, for each $j \in [m]$, the permutation model of $\mathcal{E}_j$ consists in the pair of permutations $\{\gamma_j^1, \gamma_j^2\}$, where $\gamma_j^1 = K'_{e_j} S'_{e_j} S''_{e_j} K''_{e_j}$ and $\gamma_j^2 = S'_{e_j} \overleftarrow{K''_{e_j}} \overleftarrow{K'_{e_j}} S''_{e_j}$.

Now, for each edge $e_j = v_i v_{i'}$, with $i < i'$, add four new vertices $C^1_{i,j}$, $C^2_{i,j}$, $C^1_{i',j}$ and $C^2_{i',j}$, called *incidence* vertices[2]. In what follows, we modify some of the permutations representing grained gadgets in order to make $C^1_{i,j}, C^2_{i,j}$ (resp. $C^1_{i',j}, C^2_{i',j}$) weakly intersect $\mathcal{H}_i$ (resp. $\mathcal{H}_{i'}$) and strongly intersect (resp. weakly intersect) $\mathcal{E}_j$.

If v_i is incident to the edges $e_{j_1}, e_{j_2}, e_{j_3}$, with $j_1 < j_2 < j_3$, then modify the permutation π_i^1, defining $\mathcal{H}_i$, to include the incidence vertices related to v_i, as follows:

$$\pi_i^1 = K'_i S'_i S''_i \boxed{C^1_{i,j_1} C^2_{i,j_1} C^1_{i,j_2} C^2_{i,j_2} C^1_{i,j_3} C^2_{i,j_3}} K''_i.$$

Similarly, for each edge $e_j = v_i v_{i'}$, $i < i'$, modify the permutation γ_j^1, defining $\mathcal{E}_j$, to include the incidence vertices related to e_j, as follows:

$$\gamma_j^1 = K'_{e_j} \boxed{C^2_{i',j} C^1_{i',j}} S'_{e_j} \boxed{C^2_{i,j} C^1_{i,j}} S''_{e_j} K''_{e_j}.$$

We do not modify π_i^2 and γ_j^2, and keep denoting by π_i^2 the permutation $S'_i \overleftarrow{K''}_i \overleftarrow{K'}_i S''_i$, and by γ_j^2 the permutation $S'_{e_j} \overleftarrow{K''^e}_j \overleftarrow{K'^e}_j S''_{e_j}$. Finally, let G' be the permutation graph related to $\{\Pi, \Pi'\}$, where:

$$\Pi = \pi_1^1 \dots \pi_n^1 \gamma_1^2, \dots, \gamma_m^2, \text{ and}$$

$$\Pi' = \pi_1^2 \dots \pi_n^2 \gamma_1^1, \dots, \gamma_m^1.$$

Figure 3.14 illustrates our permutation model $\{\Pi, \Pi'\}$, focusing on the vertex grained gadgets $\mathcal{H}_i$ and $\mathcal{H}_{i'}$, the edge grained gadget $\mathcal{E}_j$, and the incidence vertices $C^1_{i,j}, C^2_{i,j}$ and $C^1_{i',j}, C^2_{i',j}$ related to an edge $e_j = v_i v_{i'}$, with $i < i'$.

[2] In Appendix F, such vertices are called *link vertices* and are denoted by L. However, we note that they are not related to the notion of *link intervals* defined in Section 3.4.

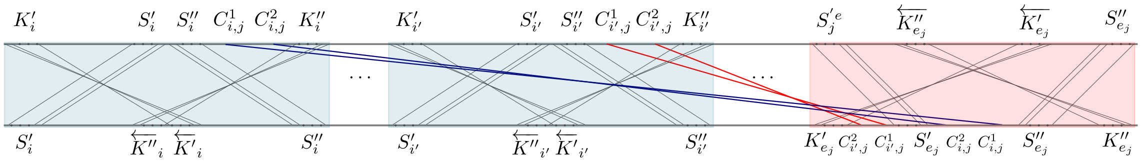


Figure 3.14: Vertex and edge grained gadgets, and incidence vertices related to an edge $e_j = v_i v_{i'}$, with $i < i'$, in our permutation model $\{\Pi, \Pi'\}$.

Note that, if C is an incidence vertex of G' and H is a vertex/edge grained gadget of G', then C and H are related to one another in exactly one of the following ways:

- The relative order between C and $V(H)$ in Π is the reverse of their relative order in Π', and therefore case C is complete to $V(H)$;
- The relative order between C and $V(H)$ is the same in both Π and Π', and therefore C is anti-complete to $V(H)$;
- $C \in \{C^1_{i,j}, C^2_{i,j}\}$ and H is a vertex grained gadget $\mathcal{H}_i$, and therefore only the relative orders between C and K''_i are reversed in Π and Π', i.e., C is complete to K''_i and anti-complete to $V(\mathcal{H}_i) \setminus K''_i$;
- $C \in \{C^1_{i,j}, C^2_{i,j}\}$ and H is an edge grained gadget $\mathcal{E}_j$, with $e_j = v_i v_{i'}$, $i < i'$, and therefore the relative orders between C and $K'_{e_j} \cup S'_{e_j}$ are reversed in Π and Π', i.e., C is complete to $K'_{e_j} \cup S'_{e_j}$ and anti-complete to $V(\mathcal{E}_j) \setminus (K'_{e_j} \cup S'_{e_j})$; or
- $C \in \{C^1_{i',j}, C^2_{i',j}\}$ and H is an edge grained gadget $\mathcal{E}_j$, with $e_j = v_i v_{i'}$, $i < i'$, and therefore only the relative orders between C and K'_{e_j} are reversed in Π and Π', i.e., C is complete to K'_{e_j} and anti-complete to $V(\mathcal{E}_j) \setminus K'_{e_j}$.

As an additional remark, we note that the main difference of our permutation graph from the Adhikary et al.'s interval graph is the fact that, in Adhikary et al.'s interval graph, the incidence vertices form a clique, whereas some incidence vertices are not adjacent in our permutation graph. Furthermore, for an edge $e_j = v_i v_{i'} \in E(G)$, with $i < i'$, the incidence vertices $C^1_{i,j}, C^2_{i,j}$ (resp. $C^1_{i',j}, C^2_{i',j}$) weakly intersect (resp. strongly intersect) the edge grained gadget $\mathcal{E}_j$ in Adhikary et al.'s interval graph, whereas in our permutation graph the incidence vertices $C^1_{i,j}, C^2_{i,j}$ (resp. $C^1_{i',j}, C^2_{i',j}$) strongly intersect (resp. weakly intersect) $\mathcal{E}_j$.

3.5.2 Maximum Cut of the Reduction Graph

We now prove that $\mathsf{mc}(G) \geq k$ if and only if $\mathsf{mc}(G') \geq \phi(n, k)$, where ϕ is a suitable function, properly defined at the end of this subsection.

First, we need to introduce the notion of *well-behaved*[3] cuts for G'. We say that a maximum cut $[A,B]$ of G' is *well-behaved* if, for every $i \in [n]$, $\mathcal{H}_i$ is either A-partitioned or B-partitioned by $[A,B]$, and for every edge $e_j \in E(G)$, with $e_j = v_i v_{i'}$ and $i < i'$, the following conditions are satisfied:

1. If $\mathcal{H}_i$ is A-partitioned, then $\{C^1_{i,j}, C^2_{i,j}\} \subseteq B$; otherwise, $\{C^1_{i,j}, C^2_{i,j}\} \subseteq A$.

2. If $\{C^1_{i,j}, C^2_{i,j}\} \subseteq B$, then $\mathcal{E}_j$ is A-partitioned; otherwise, $\mathcal{E}_j$ is B-partitioned.

By counting the possible number of edges in the cut-set, one can show that, if G' is well-valued and, in addition, $q > 6n + p'$ and $p' > 2q'$, then every maximum cut $[A,B]$ of G' is well-behaved. For a detailed proof of this result, we refer the reader to Section 4 in Appendix F. Assume in the remainder of this section that G' is well-valued and $q > 6n + p'$ and $p' > 2q'$.

Then, for each well-behaved cut $[A,B]$ of G', we let $\Phi(A,B)$ be the cut $[X,Y]$ of G defined as follows: for each vertex $v_i \in V(G)$,

$$v_i \in X \text{ if and only if } \mathcal{H}_i \text{ is } A\text{-partitioned by } [A,B].$$

One can readily verify that Φ is well-defined and, actually, consists in a bijective relation between the well-behaved cuts of G' and the cuts of G.

Lemma 3.7 establishes that, for q' large enough, the size of a well-behaved cut $[A,B]$ of G' grows as a function of the size of $\Phi^{-1}(A,B)$. The following notation is useful in the proof of this lemma. For each edge $e_j \in E(G)$, with $e_j = v_i v_{i'}$, we let

$$C(e_j) = \{C^1_{i,j}, C^2_{i,j}, C^1_{i',j}, C^2_{i',j}\};$$

and for each vertex $v_i \in V(G)$, we let

$$C(v_i) = \{C^1_{i,j}, C^2_{i,j} \mid e_j \text{ is incident to } v_i\}.$$

Also, we denote by $\mathcal{C} = \bigcup_{j=1}^{m} C(e_j)$ the set of all incidence vertices.

Lemma 3.7. *Let G be a cubic graph, and π_V and π_E be orderings of $V(G)$ and π_E, respectively. Also, let $\{\Pi, \Pi'\}$ be the reduction permutation model obtained from G, π_V and π_E, as previously defined, and G' be the permutation graph of $\{\Pi, \Pi'\}$. For every positive integer k, if $[A,B]$ is a well-behaved cut of G' and $2q' > 9n^2$, then*

$$|E_G(X,Y)| \geq k \ \textit{ if and only if } \ |E_{G'}(A,B)| \geq \phi(n,k),$$

where $[X,Y] = \Phi(A,B)$ and ϕ is a well-defined function.

[3] This is analogous to the notion of *locally well-behaved* cut, defined in Section 3.4.2, for the interval graph $\mathbb{G}_{\mathcal{M}(\mathfrak{S})}$.

Proof. We count the number of edges in $E_{G'}(A, B)$ as a function of n, m, p, q, p', q' and the size of the cut-set $E_G(X, Y)$.

First, consider a vertex v_i of G. By construction, there are $2pq + q^2$ edges in the cut-set among the vertices of $\mathcal{H}_i$. Additionally, since G is cubic, there are exactly 6 incidence vertices weakly intersecting $\mathcal{H}_i$, while all other incidence vertices are either complete or anti-complete to $V(\mathcal{H}_i)$. Note also that the number of incidence vertices complete to $V(\mathcal{H}_i)$ is exactly equal to $6(i-1)$; these are the incidence vertices related to $\{v_1, \ldots, v_{i-1}\}$. This gives us a total of $6[q + (i-1)(p+q)]$ edges between the vertices of $\mathcal{H}_i$ and the vertices belonging to $\mathcal{C}$ in the cut-set. Summing up these values for every $v_i \in V(G)$, we obtain a total of

$$\alpha_1 = n[2pq + q^2 + 6q] + 6\sum_{i=1}^{n}((i-1)(p+q)) = n[2pq + q^2 + 6q + 3(p+q)(n-1)]$$

edges in the cut-set $E_{G'}(A, B)$ incident to vertex grained gadgets.

Now, let $e_j \in E(G)$, with $e_j = v_i v_{i'}$ and $i < i'$. By construction, there are $2p'q' + (q')^2$ edges of the cut-set among the vertices of $\mathcal{E}_j$, and $2p'$ edges of the cut-set between $C^1_{i,j}, C^2_{i,j}$ and the vertices of $\mathcal{E}_j$. Moreover, note that there are exactly $4(m-j)$ incidence vertices that cover and, therefore, are complete to $\mathcal{E}_j$; these are the incidence vertices related to $\{e_{j+1}, \ldots, e_m\}$. This gives us a total of $4(m-j)(p'+q')$ edges between the vertices of $\mathcal{E}_j$ and the vertices belonging to $\mathcal{C} \setminus C(e_j)$.

Suppose without loss of generality that $\mathcal{H}_i$ is A-partitioned (the count is analogous if it is B-partitioned). Then, $v_i \in X$. Moreover, since $[A, B]$ is well-behaved, we have that $\{C^1_{i,j}, C^2_{i,j}\} \subseteq B$ and that $\mathcal{E}_j$ is A-partitioned. Hence, if $\mathcal{H}_{i'}$ is A-partitioned, then $\{C^1_{i',j}, C^2_{i',j}\} \subseteq B$. Consequently, in this case, there are no edges between the vertices $C^1_{i',j}, C^2_{i',j}$ and the vertices of $\mathcal{E}_j$ in the cut-set $E_{G'}(A, B)$. Also, note that $v_{i'} \in X$, and thus $e_j \notin E_G(X, Y)$. On the other hand, if $\mathcal{H}_{i'}$ is B-partitioned, then $\{C^1_{i',j}, C^2_{i',j}\} \subseteq A$. This implies that, in this case, there are $2q'$ additional edges in the cut-set $E_{G'}(A, B)$, namely: the edges between the vertices $C^1_{i',j}, C^2_{i',j}$ and the vertices in K'_{e_j}. Furthermore, $v_{i'} \in Y$, and thus $e_j \in E_G(X, Y)$. Summing up these values for every $e_j \in E(G)$, we obtain a total of $\alpha_2 + 2q'|E_G(X, Y)|$ edges in the cut-set $E_{G'}(A, B)$ incident to edge grained gadgets, where

$$\begin{aligned} \alpha_2 &= m[2p'q' + (q')^2 + 2p'] + \textstyle\sum_{j=1}^{m}(4(m-j)(p'+q')) \\ &= m[2p'q' + (q')^2 + 2p' + 2(p'+q')(m-1)]. \end{aligned}$$

Finally, note that there are at most $|A \cap \mathcal{C}| \cdot |B \cap \mathcal{C}|$ edges among incidence vertices in the cut-set. Note also that $|A \cap \mathcal{C}| = 6|X|$ since each vertex in X is related to 6 incidence vertices, which are all placed in A. Similarly, we have $|B \cap \mathcal{C}| = 6|Y|$. This gives us at most $36|X| \cdot |Y| \leq 9n^2$ edges in the cut-set between incidence vertices.

Putting everything together, we obtain:

$$\alpha_1 + \alpha_2 + 2q'|E_G(X,Y)| \leq |E_{G'}(A,B)| \leq \alpha_1 + \alpha_2 + 2q'|E_G(X,Y)| + 9n^2.$$

Thus, base on the facts that p, q, p', q' are defined depending exclusively on n, we let $\phi(n,k) = \alpha_1 + \alpha_2 + 2q'k$. Now, we finally prove that $|E_G(X,Y)| \geq k$ if and only if $|E_{G'}(A,B)| \geq \phi(n,k)$. First, suppose that $|E_G(X,Y)| \geq k$. Then, it follows from the first inequality that $|E_{G'}(X,Y)| \geq \alpha_1 + \alpha_2 + 2q'k = \phi(n,k)$. Conversely, suppose that $|E_{G'}(A,B)| \geq \phi(n,k) = \alpha_1 + \alpha_2 + 2q'k$. Then, it follows from the second inequality that $|E_G(X,Y)| \geq k - 9n^2/2q'$. Since we are assuming that $2q' > 9n^2$, we obtain that $k - 9n^2/2q' > k - 1$. Therefore, $|E_G(X,Y)| \geq k$. □

Now, to conclude the proof of Theorem 3.2, it only remains to properly define the values of p, q, p' and q'. Note that:

- For every grained gadget H, the total number of vertices in $V(G') \setminus H$ adjacent to a vertex of H is at most $6n$ (these are exactly the incidence vertices).

- For every vertex grained gadget H, the total number of vertices adjacent to some vertex $u \in S'(H)$ is exactly $q + 2h$, for some positive integer h (this is because the number of incidence vertices adjacent to the vertices in $S'(H)$ is always even). The same holds for the number of vertices adjacent to the vertices in $S''(H)$. We then get that the parity of ℓ and r in the conditions of Lemma 3.1 applied to H depends only on the parity of q.

- Similarly, if H is an edge grained gadget, then the parity of the total number of vertices adjacent to some $u \in S'(H) \cup S''(H)$ is equal to the parity of q'.

Therefore, for the graph G', the necessary conditions of Lemma 3.1 translate to: $q > 12n$ and $q' > 12n$; $p > 2q + 6n$ and $p' > 2q' + 6n$; and q and q' are odd. Additionally, we need to ensure: $q > 6n + p'$ and $p' > 2q' > 9n^2$. Hence, consider: $q' = 5n^2 + 1$; $p' = 11n^2 + 6n$; $q = 12n^2 + 12n + 1$; and $p = 25n^2 + 30n$. Since $n \geq 4$, one can readily verify that these values satisfy all the required conditions.

3.6 Concluding Remarks

In this chapter, we have presented the first complexity result for MaxCut when restricted to interval graphs of bounded interval count. More specifically, we have settled the problem as NP-complete on interval graphs of interval count 4. This is an important improvement towards filling the complexity gap between interval graphs and unit interval graphs (*i.e.*, interval graphs of interval count 1). In this sense, it is also worth mentioning the very recent proof due to Barsukov, Bose, and Roy, available in the preprint [3], which claims to extend our result, by showing that the problem is still NP-complete on interval graphs of interval count at most 2. Despite all these advancements, it remains a challenging question to determine the complexity of MaxCut on interval graphs of interval count 1.

Additionally, we have proved that MaxCut remains NP-complete on permutation graphs, answering a long-standing open question from [83]. Our proof is built on the notion of grained gadgets, described in Section 3.2, which also plays an important role in Adhikary et al.'s construction for interval graphs [1]. Even though grained gadgets are both interval and permutation graphs, we have shown in Section 3.3 that being permutation is not a property that holds for the full construction of Adhikary et al. [1]. In addition, through the same arguments, one can verify that this property does not hold for our interval graph of bounded interval count either. On the other hand, one could wonder whether our reduction permutation graph is interval. However, the answer to that is no, as we argue next.

Let G be a cubic graph, and $\pi_V = (v_1, \ldots, v_n)$ and $\pi_E = (e_1, \ldots, e_m)$ be arbitrary orderings of $V(G)$ and $E(G)$, respectively. Let j_1, j_2, j_3 be the indices of the edges incident to v_1, with $j_1 < j_2 < j_3$. Also, let v_i be the other endpoint of e_{j_2}. We show that the permutation graph G', with permutation model $\{\Pi, \Pi'\}$, obtained from (G, π_V, π_E) as described in Section 3.5.1, has a C_4 as an induced subgraph. This implies that G' is not chordal, and hence not interval either [74]. Observe Figure 3.15 to follow our argument. Let a be equal to C^1_{1,j_1}, b be any vertex in K''_i, c be equal to C^1_{i,j_2}, and d be any vertex in $K'^e_{j_1}$. Since $j_2 > j_1$ and $i > 1$, the relative order between a and c in Π is the same as in Π'; hence $ac \notin E(G')$. Also, the relative order in Π between a and any vertex of $\mathcal{H}_i$ is reversed in Π', the same holds between c and any vertex belonging to K''_i; hence $\{ab, bc\} \subseteq E(G')$. Similarly, the relative order between a and any vertex belonging to $K'^e_{j_1}$ in Π is reversed in Π', and the same holds between c and any vertex of $\mathcal{E}_{j_1}$; hence $\{ad, cd\} \subseteq E(G')$. Finally, since $j_2 > j_1$, the relative order between b and d in Π is the same as in Π', and therefore $bd \notin E(G')$, thus finishing our argument.

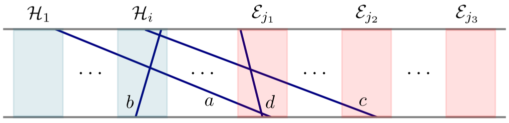


Figure 3.15: Existence of a $C_4 = (a, b, c, d)$ as an induced subgraph in our permutation graph.

The previous paragraph establishes that, for any chosen orderings of $V(G)$ and $E(G)$, the graph constructed in Section 3.5.1 contains a C_4 as an induced subgraph. Since it is known that the class of C_4-free co-comparability graphs is precisely the class of interval graphs [72], and that the class of permutation graphs is equal to the class of comparability co-comparability graphs [105], we obtain that interval permutation graphs are exactly the class of C_4-free permutation graphs.

A good question is whether there is a construction that produces a permutation graph that is also C_4-free (and hence interval). Up to our knowledge, the largest class in the intersection of permutation and interval graphs for which the complexity is known is the class of the trivially perfect graphs, on which MaxCut is polynomial-time solvable thanks to the algorithm given for cographs [10], a subclass of permutation graphs that is a superclass of trivially perfect graphs [73].

Chapter 4

Zig-Zag Number

In this chapter, we analyse the computational complexity of the problem of computing the zig-zag number of a directed graph.

Structural graph parameters have been crucial in the development of parameterized complexity theory. Indeed, many problems that are hard on general graphs become tractable when parameterized by such parameters [28, 30]. However, one of their limitations is the fact that, when dealing with directed graphs, they do not take the direction of edges into account. Then, in [84], Johnson, Robertson, Seymour and Thomas initiated a quest for the development of width measures that explicitly take the direction of edges into consideration, which motivated the development of several width measures for directed graphs. In particular, the notion of zig-zag number was introduced in [45] as an attempt to provide a unified algorithmic framework for directed graphs. Nevertheless, little was known about the complexity of computing the zig-zag number of a directed graph.

We provide the first results in this direction. We prove that k-ZIG-ZAG NUMBER, the problem of deciding whether a directed graph has zig-zag number at most k, is in NP for each fixed $k \geq 0$. Although for most of the natural decision problems this is an almost trivial result, settling k-ZIG-ZAG NUMBER in NP is surprisingly difficult. Finally, we prove that 2-ZIG-ZAG NUMBER is an already NP-hard problem.

This chapter is organised as follows. In Section 4.1, we define the notion of zig-zag number of a directed graph. In Section 4.2, we show that k-ZIG-ZAG NUMBER can be solved *non-deterministically* in time $|G|^{\mathcal{O}(k)}$, implying therefore the NP-membership of the problem for each fixed k. On the other hand, in Section 4.3, we prove that 2-ZIG-ZAG NUMBER is an NP-hard problem. This hardness proof is built on a polynomial-time reduction from POSITIVE NOT ALL EQUAL 3SAT. Finally, in Section 4.4, we present the concluding remarks of this chapter, focusing on the main open questions.

4.1 The Zig-Zag Number of a Directed Graph

Let n be a positive integer, G be a directed graph on n vertices and $\pi\colon V(G) \to [n]$ be a bijection. For simplicity, assume that $V(G) = \{u_1, \dots, u_n\}$ and $\pi(u_i) = i$ for each vertex $u_i \in V(G)$.

For each $i \in [n-1]$, we let $S_G(\pi, i) = E_G(\{u_1, \dots, u_i\})$ be the *i-th cut-set of G with respect to π*, *i.e.* the set of all edges of G between the vertices in $\{u_1, \dots, u_i\}$ and the vertices in $\{u_{i+1}, \dots, u_n\}$. The *cutwidth of G with respect to π* is defined as

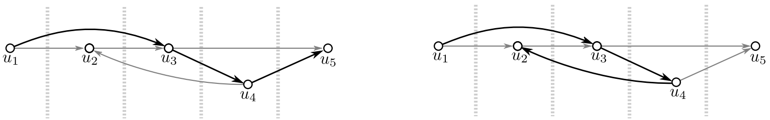
$$\mathrm{cw}(G, \pi) = \max_{i \in [n-1]} |S_G(\pi, i)|,$$

and the *cutwidth of G* is defined as the minimum $\mathrm{cw}(G, \pi)$ over all bijections $\pi\colon V(G) \to [n]$. Let P be a directed path of G. We let $\mathsf{zn}(G, \pi, P)$ be the maximum number of edges of P that are part of the cut-set $S_G(\pi, i)$, where the maximum is taken over all $i \in [n-1]$. More formally,

$$\mathsf{zn}(G, \pi, P) = \max_{i \in [n-1]} |E(P) \cap S_G(\pi, i)|.$$

Then, we let $\mathsf{zn}(G, \pi)$ be the maximum $\mathsf{zn}(G, \pi, P)$ over all directed paths P of G. Finally, we define the *zig-zag number of G*, denoted by $\mathsf{zn}(G)$, as the minimum $\mathsf{zn}(G, \pi)$ over all bijections $\pi\colon V(G) \to [n]$.

Figure 4.1 exemplifies a directed graph G and a bijection $\pi\colon V(G) \to [n]$ such that $\mathsf{zn}(G, \pi) = 2$. In fact, one can verify that $\mathsf{zn}(G) = \mathsf{zn}(G, \pi) = 2$.



(a) $\mathsf{zn}(G, \pi, P_1) = 1$

(b) $\mathsf{zn}(G, \pi, P_2) = 2$

Figure 4.1: Directed graph G, bijection $\pi\colon V(G) \to \{1 \dots, 5\}$, where $i < j$ iff $u_i <_\pi u_j$, and directed paths P_1 and P_2 (in bold), such that $\mathsf{zn}(G, \pi, P_1) = 1$ and $\mathsf{zn}(G, \pi, P_2) = 2$, respectively.

It is straightforward from the definition of zig-zag number that a directed graph has zig-zag number 0 if and only if it does not contain any edge. Moreover, one can verify that every directed acyclic graph with at least one edge has zig-zag number 1. Indeed, it is known that a directed graph G is directed acyclic if and only if it admits a *topological ordering*, *i.e.* a linear order $<_\pi$ such that $u <_\pi v$ for each $(u, v) \in E(G)$. Thus, one can verify that, if G is a directed acylic graph and π corresponds to a topological ordering of G, then $\mathsf{zn}(G, \pi) = 1$. In other words, graphs of zig-zag number at least 2 must contain directed cycles. On the other hand, every directed graph G with a directed cycle of length at least 3 necessarily has zig-zag number greater than or equal to 2. In this case, for each bijection $\pi\colon V(G) \to [|G|]$, there always exist three distinct vertices $a, b, c \in V(G)$ such that (a, b, c) is a directed path of G, where $a <_\pi b$ and $c <_\pi b$. Intuitively, the zig-zag number of a directed graph measures how much its directed cycles are nested.

Next, we formally define the Zig-zag number problem.

Zig-zag number

Input: A directed graph G and a non-negative integer k.

Question: Is $\mathsf{zn}(G) \leq k$? In other words, does there exist a bijection $\pi\colon V(G) \to [|G|]$ such that, for every directed path P of G,

$$\mathsf{zn}(G, \pi, P) = \max_{i \in [|G|-1]} |E(P) \cap S_G(\pi, i)| \leq k?$$

In particular, for each fixed non-negative integer k, we define k-zig-zag number as the decision problem that, given a directed graph G, asks whether $\mathsf{zn}(G) \leq k$. More formally:

k-zig-zag number

Input: A directed graph G.

Question: Is $\mathsf{zn}(G) \leq k$? In other words, does there exist a bijection $\pi\colon V(G) \to [|G|]$ such that, for every directed path P of G,

$$\mathsf{zn}(G, \pi, P) = \max_{i \in [|G|-1]} |E(P) \cap S_G(\pi, i)| \leq k?$$

4.2 NP-Membership for Fixed k

In this section, we prove that k-zig-zag number is in NP for each fixed k. We remark that a naive application of the definition of zig-zag number of a directed graph naturally leads to a $\mathbf{\Sigma}_2^{\mathsf{P}}$-upper bound. To circumvent this and settle k-zig-zag number in NP, we show how to replace the inner universal quantifier, which iterates over all directed paths, with an XP-time deterministic computation corresponding to a guessed linear order of the vertices of the input graph and the integer k. More specifically, we prove the following theorem.

Theorem 4.1. *Let G be a directed graph and k be a non-negative integer. One can non-deterministically decide in time $|G|^{\mathcal{O}(k)}$ whether $\mathsf{zn}(G) \leq k$.*

In order to prove Theorem 4.1, we reduce the problem of deciding whether $\mathsf{zn}(G, \pi) \geq k + 1$, for a guessed bijection $\pi\colon V(G) \to [|G|]$, to the Reachability problem in a suitably defined directed acyclic graph, denoted by $D_G(\pi, k)$, which we call *compatibility graph* of the triple (G, π, k). The formal definition of such a graph is properly given later on. Next, we describe how this section is organised.

In Section 4.2.1, we define the concept of *compatible subcut sequence* of a directed graph G with respect to a bijection $\pi\colon V(G) \to [|G|]$. Based on that, we provide a necessary and sufficient condition for $\mathsf{zn}(G, \pi) \geq k+1$. Then, we formally define in Section 4.2.2 the notion of *compatibility graph*, and we introduce a characterization relating the existence of the compatible subcut sequences of interest to the existence of directed paths with $|G|-1$ vertices in the compatibility graph of (G, π, k).

4.2.1 Compatible Subcut Sequence

Let G be a directed graph on n vertices and $\pi\colon V(G) \to [n]$ be a bijection. For simplicity, assume throughout this section that $V(G) = \{u_1, \ldots, u_n\}$ and $\pi(u_i) = i$ for each vertex $u_i \in V(G)$.

The *cut sequence of* G *with respect to* π is defined as the sequence

$$\gamma_{G,\pi} = (S_G(\pi, 1), \ldots, S_G(\pi, n-1)).$$

For each $i, j \in [n-1]$, with $i < j$, and each two *subcuts* $S'_i \subseteq S_G(\pi, i)$ and $S'_j \subseteq S_G(\pi, j)$, we say that S'_i is *compatible* with S'_j, and we denote this fact by $S'_i \prec_{G,\pi} S'_j$, if for each $e = (u, v) \in E(G)$ the two conditions below are observed.

1. If $e \in S'_i$, and either $\pi(u) > j$ or $\pi(v) > j$, then $e \in S'_j$.

2. If $e \in S_G(\pi, i) \setminus S'_i$, then $e \notin S'_j$.

Intuitively, Conditions 1 and 2 say that, if S'_i and S'_j are two compatible subcuts, then it holds that e belongs to S'_i if and only if it belongs to S'_j, for any edge e belonging simultaneously to the cut-sets $S_G(\pi, i)$ and $S_G(\pi, j)$.

A *compatible subcut sequence* of $\gamma_{G,\pi}$ is a sequence of subcuts

$$\gamma' = (S'_1, \ldots, S'_{n-1})$$

such that $S'_i \subseteq S_G(\pi, i)$ for each $i \in [n-1]$, and $S'_j \prec_{G,\pi} S'_{j+1}$ for each $j \in [n-2]$.

We note that, based on Conditions 1 and 2 described above, if S'_i and S'_j are two subcuts in a same compatible subcut sequence, then there do not exist any inconsistency with respect to the edges that belong to S'_i and to S'_j.

We let $\mathbf{\Gamma}(\gamma_{G,\pi})$ denote the set of all compatible subcut sequences of $\gamma_{G,\pi}$.

For every $\gamma' = (S'_1, \ldots, S'_{n-1}) \in \mathbf{\Gamma}(\gamma_{G,\pi})$, we define the *width* of γ' as

$$\omega(\gamma') = \max_{i \in [n-1]} |S'_i|.$$

If $E' = \bigcup_{i \in [n-1]} S'_i \neq \emptyset$, then we define $G[\gamma'] = G[E']$ as the directed graph induced by E'. In particular, we remark that $\gamma_{G,\pi}$ is a compatible subcut sequence itself of width $\mathrm{cw}(G, \pi)$ and that $G[\gamma_{G,\pi}]$ consists in the directed graph obtained from G by removing all of its isolated vertices.

The next lemma, which plays a central role in the proof of Theorem 4.1, states that deciding whether $\mathsf{zn}(G, \pi) \geq k + 1$ is equivalent to deciding whether there is a compatible subcut sequence of $\gamma_{G,\pi}$ of width at least $k+1$, whose associated directed graph is a directed path.

Lemma 4.1. *Let G be a directed graph, $\pi\colon V(G) \to [|G|]$ be a bijection and k be a non-negative integer. Then, $\mathsf{zn}(G, \pi) \geq k + 1$ if and only if there is a compatible subcut sequence $\gamma' = (S'_1, \ldots, S'_{|G|-1}) \in \mathbf{\Gamma}(\gamma_{G,\pi})$ such that $\omega(\gamma') \geq k + 1$ and $G[\gamma']$ is a directed path.*

Proof. First, suppose that $\mathsf{zn}(G, \pi) \geq k + 1$, and let P be a directed path of G such that $\mathsf{zn}(G, \pi, P) \geq k + 1$. Consider the sequence $\gamma' = (S'_1, \ldots, S'_{|G|-1})$ of subcuts such that $S'_i = E(P) \cap S_G(\pi, i)$ for each $i \in [|G| - 1]$. We prove that γ' is a compatible subcut sequence of $\gamma_{G,\pi}$. In other words, we prove that $S'_i \prec_{G,\pi} S'_{i+1}$ for each $i \in [|G| - 2]$. Note that, if for some $i \in [|G| - 2]$ there exists an edge $e \in S_G(\pi, i) \setminus S'_i$, then $e \in E(G) \setminus E(P)$ and, consequently, $e \notin S'_{i+1}$. Now, suppose that for some $i \in [|G| - 2]$ there exists an edge $e = (u, v) \in S'_i$ such that either $\pi(u) > i + 1$ or $\pi(v) > i + 1$. Clearly, $e \in E(P)$. Moreover, note that $e \in S_G(\pi, i + 1)$, and thus $e \in S'_{i+1}$. Therefore, γ' is indeed a compatible subcut sequence of $\gamma_{G,\pi}$. Additionally, by the choice of P and the construction of γ', it follows that $\omega(\gamma') = \mathsf{zn}(G, \pi, P) \geq k + 1$ and that $G[\gamma']$ is equal to P, which is a directed path.

Conversely, suppose that there exists a compatible subcut sequence $\gamma' = (S'_1, \ldots, S'_{|G|-1})$ of $\gamma_{G,\pi}$ such that $\omega(\gamma') \geq k + 1$ and $G[\gamma']$ is a directed path. Thus, there exists $i \in [|G| - 1]$ such that $|S'_i| \geq k + 1$. As a result, if $P = G[\gamma']$, then $\mathsf{zn}(G, \pi, P) \geq |S'_i| \geq k + 1$. Therefore, $\mathsf{zn}(G, \pi) \geq \mathsf{zn}(G, \pi, P) \geq k + 1$. □

4.2.2 Compatibility Graph

In this section, we define the notion of *compatibility graph.* Intuitively, each directed path with $|G| - 1$ vertices of the *compatibility graph* $D_G(\pi, k)$ corresponds to a compatible subcut sequence γ' of $\gamma_{G,\pi}$ satisfying the conditions described in Lemma 4.1. More specifically, the vertices of $D_G(\pi, k)$ consist in special tuples which, along with the directed edges between them, define a dynamic programming table. This table stores all the information needed to ensure that, if there is a directed path in $D_G(\pi, k)$ with $|G| - 1$ vertices, then there exists a compatible subcut sequence γ' of $\gamma_{G,\pi}$ such that $\omega(\gamma') \geq k + 1$.

In order to capture the mentioned property, we partition the vertex set of $D_G(\pi, k)$ into $|G| - 1$ distinct levels, such that each level $i \in [|G| - 1]$ is associated with the cut-set $S_G(\pi, i)$ and there is a *directed* edge in $D_G(\pi, k)$ from a vertex $\mathfrak{u}$ to a vertex $\mathfrak{v}$ only if $\mathfrak{u}$ belongs to a level i and $\mathfrak{v}$ belongs to the level $i + 1$, and some additional constraints (described in Section 4.2.2) are satisfied. The vertices in the level $i = 1$ are called *initial*, the vertices in a level $i \in [|G| - 1] \setminus \{1, |G| - 1\}$ are called *intermediary*, and the vertices in the level $i = |G| - 1$ are called *final*. We note that, by definition, the initial vertices of $D_G(\pi, k)$ have in-degree 0 and the final vertices of $D_G(\pi, k)$ have out-degree 0. Figure 4.2 illustrates the partitioning of the vertex set of the compatibility graph $D_G(\pi, k)$ into these $|G| - 1$ distinct levels.

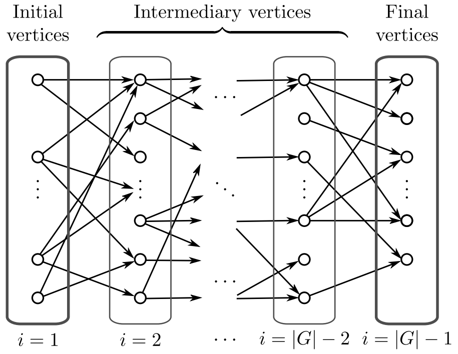


Figure 4.2: A compatibility graph.

One can alternatively regard $D_G(\pi, k)$ as an acyclic finite automaton — with transition set defined by the adjacency relation described in Section 4.2.2. From this perspective, the initial vertices represent the initial states of the automaton and the final vertices represent the final states of the automaton.

The following immediate observation provides the basis for the definition of compatibility graph.

Observation 4.1. *A non-trivial directed graph P is a directed path if and only if it satisfies the following four conditions:*

I. P has exactly one vertex, called source vertex*, of in-degree* 0 *and out-degree* 1*;*

II. P has exactly one vertex, called target vertex*, of in-degree* 1 *and out-degree* 0*;*

III. All the other vertices of P have in-degree 1 *and out-degree* 1*;*

IV. P is weakly connected.

In particular, for a compatible subcut sequence γ' of $\gamma_{G,\pi}$, we have that $G[\gamma']$ is a directed path if and only if it satisfies Conditions (I)–(IV). Based on that, let $\boldsymbol{B}_G(\pi, k)$ be the set consisting of all tuples of the form

$$\mathfrak{u}_i = (i, S'_i, \phi_i, \varphi_i, \boldsymbol{\mathcal{S}}_i, \tau_i, \psi_i),$$

where $i \in [n-1]$, S'_i is a subcut of $S_G(\pi, i)$ with cardinality at most $k+1$, $\mathcal{S}_i$ is either an empty set or a partition of S'_i, $\phi_i, \varphi_i, \tau_i \in \{0,1,2\}$ are ternary flags and $\psi_i \in \{0,1\}$ is a boolean flag. We remark that, for each $i \in [n-1]$, there are at most $n^{\mathcal{O}(k)}$ distinct tuples $\mathfrak{u}_i \in \boldsymbol{B}_G(\pi, k)$. Indeed, for each $i \in [n-1]$, the cut-set $S_G(\pi, i)$ has at most $2i(n-i) \leq n^2/2$ directed edges, which is the maximum possible number of directed edges between the vertices belonging to $\{u_1, \ldots, u_i\}$ and the vertices belonging to $\{u_{i+1}, \ldots, u_n\}$. Thus, $S_G(\pi, i)$ has at most $\binom{n^2/2}{k+1} = \mathcal{O}\left(n^{2k+2}\right)$ distinct subcuts S'_i of cardinality at most $k+1$, and each such a subcut S'_i admits at most $(k+1)^{\mathcal{O}(k)}$ distinct partitions.

Let $i \in [n-1]$ and $\mathfrak{p}_i = (1, S'_1, \phi_1, \varphi_1, \mathcal{S}_1, \tau_1, \psi_1), \ldots, (i, S'_i, \phi_i, \varphi_i, \mathcal{S}_i, \tau_i, \psi_i)$ be a sequence of tuples, such that S'_j is compatible with S'_{j+1} for each $j \in [i-1]$. Then, let H_i be the subgraph of G with vertex set $V(H_i) = \{u_1, \ldots, u_i\} \cup X_i$ and edge set $E(H_i) = S'_1 \cup \cdots \cup S'_i$, where X_i denotes the set of endpoints of the edges in $S'_1 \cup \cdots \cup S'_i$. Note that H_i may contain isolated vertices. For instance, consider the compatible subcut sequence γ' illustrated in Figure 4.3. In this example, H_1 is the trivial directed graph containing only the vertex u_1, and H_4 is the directed graph with vertex set $V(H_4) = \{u_1, \ldots, u_4\} \cup \{u_5, u_6, u_8\}$ and edge set $E(H_4) = \{e_1, \ldots, e_5\}$.

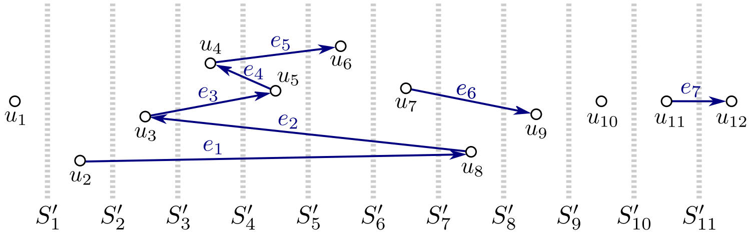


Figure 4.3: Example of a compatible subcut sequence: $\gamma' = (S'_1, \ldots, S'_{11})$, where $S'_1 = \emptyset$, $S'_2 = \{e_1\}$, $S'_3 = \{e_1, e_2, e_3\}$, $S'_4 = \{e_1, e_2, e_3, e_4, e_5\}$, $S'_5 = \{e_1, e_2, e_5\}$, $S'_6 = \{e_1, e_2\}$, $S'_7 = \{e_1, e_2, e_6\}$, $S'_8 = \{e_6\}$, $S'_9 = \emptyset$, $S'_{10} = \emptyset$, and $S'_{11} = \{e_7\}$.

Intuitively, the ternary flag ϕ_i (resp. the ternary flag φ_i) informs whether there exist zero, one, or more than one vertices from $\{u_1, \ldots, u_i\}$ that are source vertices (resp. target vertices) of H_i.

The partition $\mathcal{S}_i$ represents the set of all non-trivial weakly connected components of H_i, *restricted to the subcut* S'_i, that are defined by only taking into account the vertices from $\{u_1, \ldots, u_i\}$. In other words, two edges $e, e' \in S'_i$ belong to a same part of $\mathcal{S}_i$ if and only if there exists an undirected path of H_i between an endpoint of e and an endpoint of e' that only uses vertices from $\{u_1, \ldots, u_i\}$. For instance, considering the compatible subcut sequence γ' illustrated in Figure 4.3, we have that $\mathcal{S}_1 = \emptyset$, $\mathcal{S}_2 = \{\{e_1\}\}$, $\mathcal{S}_3 = \{\{e_1\}, \{e_2, e_3\}\}$, $\mathcal{S}_4 = \{\{e_1\}, \{e_2, e_3\}, \{e_4, e_5\}\}$, $\mathcal{S}_5 = \{\{e_1\}, \{e_2, e_5\}\}$, $\mathcal{S}_6 = \{\{e_1\}, \{e_2\}\}$, $\mathcal{S}_7 = \{\{e_1\}, \{e_2\}, \{e_6\}\}$, $\mathcal{S}_8 = \{\{e_6\}\}$, $\mathcal{S}_9 = \emptyset$, $\mathcal{S}_{10} = \emptyset$, and $\mathcal{S}_{11} = \{\{e_7\}\}$. We remark that, if two distinct edges $e, e' \in S'_i$ have the vertex u_i as an endpoint, then e and e' belong to the same part of $\mathcal{S}_i$.

However, the converse does not hold: possibly, e and e' belong to the same part of $\mathcal{S}_i$, while e and e' do not share any endpoint. Indeed, consider for instance the edges e_2 and e_5 depicted in Figure 4.3. These edges do not share any endpoint, but they do belong to the same part of $\mathcal{S}_5$, since there is an undirected path in H_5 between u_3 and u_4, which are endpoints of e_2 and e_5, respectively.

The ternary flag τ_i informs whether there exist zero, one, or more than one non-trivial weakly connected components of H_i whose vertices are contained in $\{u_{i+1}, \dots, u_n\}$. For instance, considering again the compatible subcut sequence γ' illustrated in Figure 4.3, we have that $\tau_i = 0$ for each $i \in \{1, \dots, 7\}$, $\tau_8 = 1$, and $\tau_i = 2$ for each $i \in \{9, 10, 11\}$.

Finally, the boolean flag ψ_i informs whether or not there exists a subcut of width $k+1$ among the subcuts $S'_1, \dots, S'_i$.

Initial, Final and Intermediary tuples

Now, we formally present the notions of *initial*, *final* and *intermediary* tuples. Such tuples define a proper subset of $\boldsymbol{B}_G(\pi, k)$, which is precisely equal to the vertex set of the compatibility graph $D_G(\pi, k)$. More specifically, the initial tuples, final tuples and intermediary tuples correspond to the initial vertices, final vertices and intermediary vertices of $D_G(\pi, k)$, respectively.

Let $\mathfrak{u} = (i, S'_i, \phi_i, \varphi_i, \mathcal{S}_i, \tau_i, \psi_i) \in \boldsymbol{B}_G(\pi, k)$. Intuitively, $\mathfrak{u}$ is called initial (resp. final) if $i = 1$ (resp. $i = n-1$) and the values assigned to the parameters S'_i, ϕ_i, φ_i, $\mathcal{S}_i$, τ_i and ψ_i establish a valid configuration with respect to the semantic of each parameter itself and with respect to Conditions (I)–(IV).

Thus, we say that $\mathfrak{u}$ is *initial* if $i = 1$ and the following conditions are satisfied:

1. Vertex u_1 has at most one in-edge and at most one out-edge in S'_1;

2. If u_1 has no in-edge and one out-edge in S'_1, then $\phi_1 = 1$ and $\varphi_1 = 0$;

3. If u_1 has one in-edge and no out-edge in S'_1, then $\phi_1 = 0$ and $\varphi_1 = 1$;

4. If u_1 has one in-edge and one out-edge in S'_1 or does not have any incident edge in S'_1, then $\phi_1 = 0$ and $\varphi_1 = 0$;

5. If $S'_1 = \emptyset$, then $\mathcal{S}_1 = \emptyset$; otherwise, $\mathcal{S}_1 = \{S'_1\}$;

6. $\tau_1 = 0$;

7. If $|S'_1| = k+1$, then $\psi_1 = 1$; otherwise, $\psi_1 = 0$.

On the other hand, we say that $\mathfrak{u}$ is *final* if $i = n-1$ and the following conditions are satisfied:

1. Vertex u_n has at most one in-edge and at most one out-edge in S'_{n-1};
2. If u_n has no in-edge and one out-edge in S'_{n-1}, then $\phi_{n-1} = 0$ and $\varphi_{n-1} = 1$;
3. If u_n has one in-edge and no out-edge in S'_{n-1}, then $\phi_{n-1} = 1$ and $\varphi_{n-1} = 0$;
4. If u_n has one in-edge and one out-edge in S'_{n-1} or does not have any incident edge in S'_{n-1}, then $\phi_{n-1} = 1$ and $\varphi_{n-1} = 1$;
5. $|\mathcal{S}_{n-1}| \leq 2$, and if $|\mathcal{S}_{n-1}| = 1$, then $|S'_{n-1}| = 1$;
6. $\tau_{n-1} \leq 1$, and if $\tau_{n-1} = 1$, then $S'_{n-1} = \emptyset$;
7. $\psi_{n-1} = 1$.

Finally, we say $\mathfrak{u}$ is *intermediary* if $i \in [n-1] \setminus \{1, n-1\}$.

Compatibility relation

Let $\mathfrak{u} = (i, S'_i, \phi_i, \varphi_i, \mathcal{S}_i, \tau_i, \psi_i)$ and $\mathfrak{v} = (j, S'_j, \phi_j, \varphi_j, \mathcal{S}_j, \tau_j, \psi_j)$ be a pair of tuples from $\boldsymbol{B}_G(\pi, k)$. We say that $\mathfrak{u}$ is *compatible* with $\mathfrak{v}$, and we denote such a fact by $\mathfrak{u} \rightsquigarrow \mathfrak{v}$, if the following conditions are satisfied: $j = i + 1$, $S'_i \prec_{G,\pi} S'_j$, and $\mathfrak{u}$ and $\mathfrak{v}$ satisfy the Vertex degree, Connectedness and Minimum subcut width rules, described next. These rules are defined in a way that, for any sequence $(\mathfrak{u}_1, \ldots, \mathfrak{u}_{n-1})$ of tuples from $\boldsymbol{B}_G(\pi, k)$, such that $\mathfrak{u}_i$ is compatible with $\mathfrak{u}_{i+1}$ for each $i \in [n-2]$, there exists a unique associated compatible subcut sequence $\gamma' \in \boldsymbol{\Gamma}(\gamma_{G,\pi})$.

The intuition behind the Vertex degree and Connectedness rules is ensuring that, if γ' is the subcut sequence associated with a directed path $(\mathfrak{u}_1, \ldots, \mathfrak{u}_{n-1})$ in $D_G(\pi, k)$, then $G[\gamma']$ satisfies Conditions (I)–(IV); and, the intuition behind the Minimum subcut width rule is ensuring that the width of any such compatible subcut sequences γ' is at least $k + 1$. In a nutshell, each group of rules corresponds to the computation of the related parameters ϕ_j, φ_j, τ_j, $\mathcal{S}_j$, ψ_i, which are determined based on the values of ϕ_i, φ_i, τ_i, $\mathcal{S}_i$, ψ_i, and on the edges in $S'_i \cup S'_j$.

Vertex degree rules These rules mainly correspond to the computation of the flags ϕ_j and φ_j. In particular, ϕ_j (resp. φ_j) is updates according to whether the vertex u_j is a new source (resp. target vertex).

1. The vertex u_j has at most one in-edge and at most one out-edge in $S'_i \cup S'_j$.
2. If u_j has no in-edge and one out-edge in $S'_i \cup S'_j$, then $\phi_j = \min\{2, \phi_i + 1\}$ and $\varphi_j = \varphi_i$.
3. If u_j has one in-edge and no out-edge in $S'_i \cup S'_j$, then $\phi_j = \phi_i$ and $\varphi_j = \min\{2, \varphi_i + 1\}$.

4. If u_j has one in-edge and one out-edge in $S'_i \cup S'_j$ or does not have any incident edge in $S'_i \cup S'_j$, then $\phi_j = \phi_i$ and $\varphi_j = \varphi_i$.

Connectedness rules These rules mainly correspond to the computation of the flag τ_j and of the partition $\mathcal{S}_j$.

1. If $S'_i \setminus S'_j = \emptyset$ and u_j has no incident edge in S'_j (see Figure 4.4a), then $\tau_j = \tau_i$ and $\mathcal{S}_j = \mathcal{S}_i$.

2. If $S'_i \setminus S'_j = \emptyset$ but u_j has some incident edge in S'_j (see Figure 4.4b), then $\tau_j = \tau_i$ and $\mathcal{S}_j = \mathcal{S}_i \cup \{S'_j \setminus S'_i\}$.

3. If $S'_i \setminus S'_j \neq \emptyset$ and $S'_j \neq \emptyset$ (see Figures 4.4c and 4.4d), then $\tau_j = \tau_i$ and $\mathcal{S}_j = (\mathcal{S}_i \setminus \mathcal{Q}'_j) \cup \mathcal{Q}_j$, where $\mathcal{Q}'_j$ denotes the collection of all sets in $\mathcal{S}_i$ that have at least one edge in S'_i with u_j as an endpoint, *i.e.*

   $$\mathcal{Q}'_j = \{Q \in \mathcal{S}_i \colon Q \cap (S'_i \setminus S'_j) \neq \emptyset\},$$

   and $\mathcal{Q}_j$ denotes the singleton collection whose set comprises all edges in S'_j with u_j as an endpoint, along with all edges in S'_j that belong to a set of $\mathcal{Q}'_j$, *i.e.*

   $$\mathcal{Q}_j = \Big\{(S'_j \setminus S'_i) \cup \big(\textstyle\bigcup_{Q \in \mathcal{Q}'_j} Q \cap S'_j\big)\Big\}.$$

   In this case, we further require $\mathcal{Q}_j \neq \emptyset$.

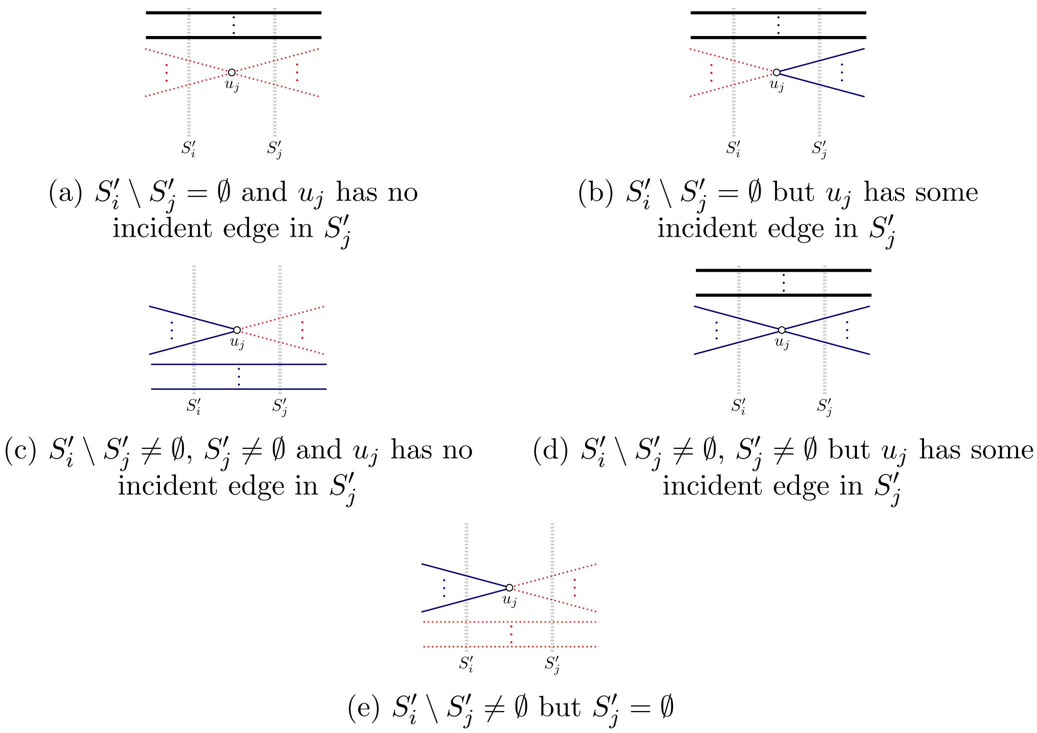
   Informally, $\mathcal{Q}'_j$ represents the set of non-trivial weakly connected components *restricted to* S'_i that have at least one edge in S'_i with u_j as an endpoint. Since, when considering the subcut S'_j, all such components contain u_j as a common vertex, they actually form a single non-trivial weakly connected component *restricted to* S'_j. This single component is represented by $\mathcal{Q}_j$, which, besides the edges that are already present in S'_i, contains all the edges in S'_j with u_j as an endpoint.

4. If $S'_i \setminus S'_j \neq \emptyset$ but $S'_j = \emptyset$ (see Figure 4.4e), then $\tau_j = \min\{2, \tau_i + 1\}$ and $\mathcal{S}_j = \emptyset$.

Minimum subcut width rule This rule corresponds to the computation of the flag ψ_i.

1. If $\psi_i = 1$ or $|S'_j| = k + 1$, then $\psi_j = 1$.

Now, we are finally able to formally define the notion of *compatibility graph* and then prove Theorem 4.1.



(a) $S'_i \setminus S'_j = \emptyset$ and u_j has no incident edge in S'_j

(b) $S'_i \setminus S'_j = \emptyset$ but u_j has some incident edge in S'_j

(c) $S'_i \setminus S'_j \neq \emptyset$, $S'_j \neq \emptyset$ and u_j has no incident edge in S'_j

(d) $S'_i \setminus S'_j \neq \emptyset$, $S'_j \neq \emptyset$ but u_j has some incident edge in S'_j

(e) $S'_i \setminus S'_j \neq \emptyset$ but $S'_j = \emptyset$

Figure 4.4: Connectedness rules. (Red) dotted lines represent non-edges, (black) thicker lines represent non-mandatory edges, and (blue) normal style lines represent mandatory edges.

For each directed graph G, each bijection $\pi\colon V(G) \to [|G|]$ and each non-negative integer k, we define the *compatibility graph* of the triple (G, π, k) as the directed acyclic graph $D_G(\pi, k)$ with vertex set

$$V = \{\mathfrak{u} \in \boldsymbol{B}_G(\pi, k) \colon \mathfrak{u} \text{ is initial, intermediary or final}\}$$

and edge set

$$E = \{(\mathfrak{u}, \mathfrak{v}) \in V \times V \colon \mathfrak{u} \rightsquigarrow \mathfrak{v}\}.$$

We recall that, based on Lemma 4.1, our goal is to decide whether there exists a compatible subcut sequence of $\gamma_{G,\pi}$ of width at least $k+1$ whose associated directed graph is a directed path. Next lemma establishes that this is equivalent to deciding whether there exists a directed path of $D_G(\pi, k)$ with $n-1$ vertices.

Due to the fact that the proof of Lemma 4.2 requires a highly technical analysis, we omit it and refer the reader to Section 3.3 of Appendix G.

Lemma 4.2. *Let G be a directed graph, $\pi\colon V(G) \to [|G|]$ be a bijection and k be a non-negative integer. There exists a compatible subcut sequence $\gamma' \in \boldsymbol{\Gamma}(\gamma_{G,\pi})$ such that $\omega(\gamma') \geq k+1$ and $G[\gamma']$ is a directed path if and only if there exists a directed path of $D_G(\pi, k)$ with $|G|-1$ vertices.*

Lemma 4.3. *Given a directed graph G, a bijection $\pi\colon V(G) \to [|G|]$ and a* *non-negative integer k, one can deterministically decide in time $|G|^{\mathcal{O}(k)}$ whether* $\mathsf{zn}(G,\pi) \leq k$.

Proof. First, we construct the directed graph $D_G(\pi,k)$. Note that, for each tuple

$$\mathfrak{u}_i = (i, S'_i, \phi_i, \varphi_i, \mathcal{S}_i, \tau_i, \psi_i) \in \boldsymbol{B}_G(\pi,k),$$

the subcut S'_i has at most $k+1$ distinct elements. As a result, one can easily check in time polynomial in k if $\mathfrak{u}_i$ is an *initial*, an *intermediary* or a *final* tuple. Moreover, since there are $|G|^{\mathcal{O}(k)}$ distinct tuples in $\boldsymbol{B}_G(\pi,k)$, the vertex set of $D_G(\pi,k)$ can be determined in time $|G|^{\mathcal{O}(k)} \cdot k^{\mathcal{O}(1)} = |G|^{\mathcal{O}(k)}$. Regarding the edge set of $D_G(\pi,k)$, we have by definition that there exists a directed edge from a vertex $\mathfrak{u}_i = (i, S'_i, \phi_i, \varphi_i, \mathcal{S}_i, \tau_i, \psi_i)$ to a vertex $\mathfrak{u}_j = \big(j, S'_j, \phi_j, \varphi_j, \mathcal{S}_j, \tau_j, \psi_j\big)$ of $D_G(\pi,k)$ if and only if $\mathfrak{u}_i$ is *compatible with* $\mathfrak{u}_j$, *i.e.*, $j = i+1$, $S'_i \prec_{G,\pi} S'_j$, and $\mathfrak{u}_i$ and $\mathfrak{u}_j$ satisfy the Vertex degree, Connectedness and Minimum subcut width rules. Since $|S'_i| \leq k+1$ and $|S'_j| \leq k+1$, the satisfaction of the Vertex degree, Connectedness and Minimum subcut width rules by $\mathfrak{u}_i$ and $\mathfrak{u}_j$ can be clearly checked in time polynomial in k. In addition, one can verify whether $S'_i \prec_{G,\pi} S'_j$ in time polynomial in $|G|$. Thus, it can be checked in time $k^{\mathcal{O}(1)} + |G|^{\mathcal{O}(1)}$ whether there should exist in $D_G(\pi,k)$ a directed edge from $\mathfrak{u}_i$ to $\mathfrak{u}_j$. This implies that the edge set of $D_G(\pi,k)$ can be determined in time $|G|^{\mathcal{O}(k)} \cdot |G|^{\mathcal{O}(k)} \cdot (k^{\mathcal{O}(1)} + |G|^{\mathcal{O}(1)}) = |G|^{\mathcal{O}(k)}$. Therefore, $D_G(\pi,k)$ can be wholly constructed in time $|G|^{\mathcal{O}(k)}$.

Then, by using an algorithm for the Reachability problem, we decide in time linear in the number of vertices and edges of $D_G(\pi,k)$, *i.e.* in time $|G|^{\mathcal{O}(k)}$, whether there is a directed path of $D_G(\pi,k)$ with $|G|-1$ vertices. By Lemmas 4.1 and 4.2, such a path exists if and only if $\mathsf{zn}(G,\pi) \geq k+1$. Therefore, we can decide in time $|G|^{\mathcal{O}(k)}$ whether $\mathsf{zn}(G,\pi) \leq k$. □

As a result, by using a permutation π of the vertices of the input graph G as a certificate, we obtain that deciding whether $\mathsf{zn}(G) \leq k$ is in NP for each fixed $k \geq 0$, concluding thereby the proof of Theorem 4.1.

4.3 NP-Hardness

In this section, we prove that 2-zig-zag number is an NP-hard problem. For that, we present a polynomial-time reduction from Positive not all equal 3SAT, which is a well-known NP-complete problem [110], defined next.

Positive not all equal 3SAT (PNAE 3SAT)

Input: Set X of variables and a collection $\mathcal{C}$ of clauses over X such that each clause has no negative literal and exactly three positive literals.

Question: Is there a truth assignment $\alpha\colon X \to \{\mathit{false}, \mathit{true}\}$ such that each clause in $\mathcal{C}$ has at least one true literal and at least one false literal under α?

Construction 4.1 (Reduction from 2-zig-zag number to PNAE 3SAT)**.** Let $I = (X, \mathcal{C})$ be an instance of PNAE 3SAT with variable set X and clause set $\mathcal{C}$. We let G_I be the directed graph obtained from I as follows.

- For each variable $x_i \in X$, add the vertices u_i^1, u_i^2 and u_i^3, and add the edges (u_i^1, u_i^2), (u_i^2, u_i^3) and (u_i^3, u_i^1).
- For each clause $C_j \in \mathcal{C}$, add the vertices v_j^1, v_j^2 and v_j^3, and add the edges (v_j^1, v_j^2), (v_j^2, v_j^3) and (v_j^3, v_j^1). Moreover, assuming $C_j = \{x_{l_1}, x_{l_2}, x_{l_3}\}$ with $l_1 < l_2 < l_3$, add the edges $(u_{l_1}^1, v_j^1)$, $(u_{l_1}^3, v_j^1)$, $(u_{l_2}^1, v_j^2)$, $(u_{l_2}^3, v_j^2)$, $(u_{l_3}^1, v_j^3)$ and $(u_{l_3}^3, v_j^3)$.

For each variable $x_i \in X$, we let H_i denote the subgraph of G_I induced by the vertices in $\{u_i^1, u_i^2, u_i^3\}$. And, for each clause $C_j \in \mathcal{C}$, we let $\widetilde{H}_j$ denote the subgraph of G_I induced by the vertices in $\{v_j^1, v_j^2, v_j^3\}$. We remark that H_i and $\widetilde{H}_j$ are directed cycles of length 3.

Figure 4.5 exemplifies the directed graph G_I, described in Construction 4.1.

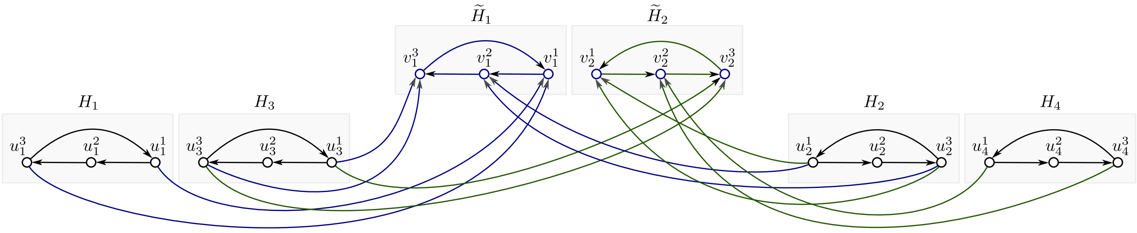


Figure 4.5: Directed graph G_I obtained from the instance $I = (X, \mathcal{C})$ of PNAE 3SAT where $X = \{x_1, x_2, x_3, x_4\}$ and $\mathcal{C} = \{C_1 = \{x_1, x_2, x_3\}, C_2 = \{x_2, x_3, x_4\}\}$.

We establish in Lemmas 4.4 and 4.6 that there exists a satisfying truth assignment for an instance I of PNAE 3SAT if and only if there exists a linear order of zig-zag number at most 2 for the vertices of G_I. The central idea of our proof is to explore the possible internal relative orderings of the vertices of each directed cycle of G_I and, for each clause $C_j = \{x_{l_1}, x_{l_2}, x_{l_3}\} \in \mathcal{C}$, the possible ordered relative placements among the subgraphs H_{l_1}, H_{l_2}, H_{l_3}, and $\widetilde{H}_j$.

Lemma 4.4. *Let $I = (X, \mathcal{C})$ be an instance of* PNAE 3SAT*. If I is a* yes *instance of* PNAE 3SAT*, then* $\mathsf{zn}(G_I) \leq 2$.

Proof. Let $\alpha\colon X \to \{\mathit{false}, \mathit{true}\}$ be a truth assignment such that each clause in $\mathcal{C}$ has at least one true literal and at least one false literal under α. In what follows, we define from α a linear order $<_\pi$ of the vertices of G_I such that $\mathsf{zn}(G_I, \pi) \leq 2$.

Throughout this proof, consider $X = \{x_1, \ldots, x_{|X|}\}$ and $\mathcal{C} = \{C_1, \ldots, C_{|\mathcal{C}|}\}$.

For each variable $x_i \in X$, set

$$\begin{cases} u_i^1 <_\pi u_i^2 <_\pi u_i^3 & \text{if } \alpha(x_i) = \mathit{true} \\ u_i^1 >_\pi u_i^2 >_\pi u_i^3 & \qquad \text{otherwise.} \end{cases}$$

Let $V'_{\mathit{false}} = \{u_i^1, u_i^2, u_i^3 \colon \alpha(x_i) = \mathit{false}\}$ and $V'_{\mathit{true}} = \{u_i^1, u_i^2, u_i^3 \colon \alpha(x_i) = \mathit{true}\}$. Then, place all the vertices belonging to V'_{true} at the end of the order π, and all the vertices belonging to V'_{false} at the beginning of π. More formally, for each $y \in V'_{\mathit{false}}$ and each $z \in V(G_I) \setminus V'_{\mathit{false}}$, set $y <_\pi z$. And, for each $y \in V'_{\mathit{true}}$ and each $z \in V(G_I) \setminus V'_{\mathit{true}}$, set $y >_\pi z$.

Let C_j be a clause in $\mathcal{C}$. Assume that $C_j = \{x_{l_1}, x_{l_2}, x_{l_3}\}$ with $l_1 < l_2 < l_3$. There are two cases to be considered. First, suppose that C_j has exactly one true literal under α, say l_q for some $q \in \{1, 2, 3\}$. Then, order the clause cycle (v_j^1, v_j^2, v_j^3) as $v_j^p <_\pi v_j^q <_\pi v_j^r$, in such a way that the vertex v_j^q, corresponding to the literal l_q, appears in the middle and $(v_j^r, v_j^q), (v_j^q, v_j^p)$ are edges of the cycle. More formally, set

$$v_j^p <_\pi v_j^q <_\pi v_j^r,$$

where $p = q \bmod 3 + 1$ and $r = (q + 1) \bmod 3 + 1$. Now, suppose that C_j has exactly two true literals under α. Thus, C_j has exactly one false literal under α, say l_q for some $q \in \{1, 2, 3\}$. Then, set

$$v_j^p >_\pi v_j^q >_\pi v_j^r,$$

where $p = q \bmod 3 + 1$ and $r = (q + 1) \bmod 3 + 1$.

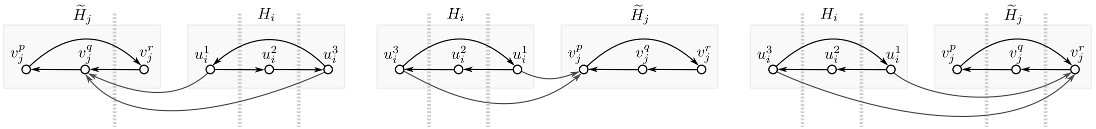
Finally, for each pair of distinct variables $x_i, x_{i'} \in X$ with $i < i'$, such that $\alpha(x_i) = \alpha(x_{i'})$, set $u_i^p <_\pi u_{i'}^q$ for each $p, q \in \{1, 2, 3\}$. And, for each pair of distinct clauses $C_j, C_{j'} \in \mathcal{C}$ with $j < j'$, set $v_j^p <_\pi v_{j'}^q$ for each $p, q \in \{1, 2, 3\}$.

One can readily verify that $<_\pi$ is indeed a linear order of the vertices of G_I.

Now, we prove that $\mathsf{zn}(G_I, \pi) \leq 2$. For the sake of contradiction, suppose that there exists a directed path P in G_I such that $\mathsf{zn}(G_I, \pi, P) \geq 3$. Assume without loss of generality that P is a minimal path with respect to the property that $\mathsf{zn}(G_I, \pi, P) \geq 3$. Recall that, for each variable $x_i \in X$, H_i is a directed cycle of length 3. Similarly, for each clause $C_j \in \mathcal{C}$, $\widetilde{H}_j$ is a directed cycle of length 3. Consequently, P is neither a subgraph of H_i nor a subgraph of $\widetilde{H}_j$, for any $x_i \in X$ and any

$C_j \in \mathcal{C}$, otherwise $\mathsf{zn}(G_I, \pi, P) < 3$. Moreover, every edge of G_I is either an edge of one of these subgraphs H_i and $\widetilde{H}_j$ or is an edge from a vertex of H_i to a vertex of $\widetilde{H}_j$, for some $x_i \in X$ and some $C_j \in \mathcal{C}$. As a result, there exists precisely one variable $x_i \in X$ and there exists precisely one clause $C_j \in \mathcal{C}$ such that $V(P) \cap V(H_i) \neq \emptyset$ and $V(P) \cap V(\widetilde{H}_j) \neq \emptyset$. More specifically, P consists in a directed path on at most 4 vertices (by its minimality) from a vertex of H_i to a vertex of $\widetilde{H}_j$ that only contains vertices belonging to $V(H_i) \cup V(\widetilde{H}_j)$. Assume that $C_j = \{x_{l_1}, x_{l_2}, x_{l_3}\}$ with $l_1 < l_2 < l_3$.

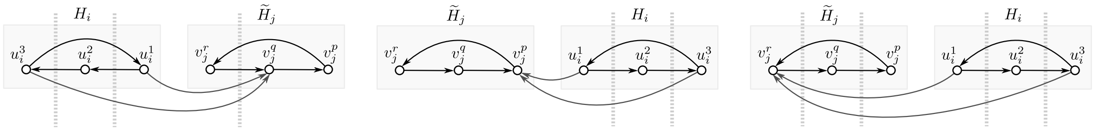
First, consider the case in which C_j has exactly one true literal under α, and let x_{l_q} be such a literal for some $q \in \{1, 2, 3\}$. Thus, $v_j^p <_\pi v_j^q <_\pi v_j^r$, where $p = q \bmod 3 + 1$ and $r = (q+1) \bmod 3 + 1$. Consequently, if $i = l_q$, then $u_i^1 <_\pi u_i^2 <_\pi u_i^3$ and $V(H_i) >_\pi V(\widetilde{H}_j)$, which implies $\mathsf{zn}(G_I, \pi, P) < 3$ (see Figure 4.6a). On the other hand, if $i = l_p$ or $i = l_r$, then $u_i^1 >_\pi u_i^2 >_\pi u_i^3$ and $V(H_i) <_\pi V(\widetilde{H}_j)$, which also implies $\mathsf{zn}(G_I, \pi, P) < 3$ (see Figures 4.6b and 4.6c).



(a) $i = l_q$ (b) $i = l_p$ (c) $i = l_r$

Figure 4.6: Case in which the clause $C_j = \{x_{l_1}, x_{l_2}, x_{l_3}\}$ has exactly one true literal under the truth assignment α, say x_{l_q} for some $q \in \{1, 2, 3\}$.

Now, consider the case in which C_j has exactly two true literals under α, and let l_q be the only false literal of C_j under α for some $q \in \{1, 2, 3\}$. Thus, $v_j^p >_\pi v_j^q >_\pi v_j^r$, where $p = q \bmod 3 + 1$ and $r = (q+1) \bmod 3 + 1$. If $i = l_q$, then $u_i^1 >_\pi u_i^2 >_\pi u_i^3$ and $V(H_i) <_\pi V(\widetilde{H}_j)$, which implies $\mathsf{zn}(G_I, \pi, P) < 3$ (see Figure 4.7a). On the other hand, if $i = l_p$ or $i = l_r$, then $u_i^1 <_\pi u_i^2 <_\pi u_i^3$ and $V(H_i) >_\pi V(\widetilde{H}_j)$, which also implies $\mathsf{zn}(G_I, \pi, P) < 3$ (see Figures 4.7b and 4.7c).



(a) $i = l_q$ (b) $i = l_p$ (c) $i = l_r$

Figure 4.7: Case in which the clause $C_j = \{x_{l_1}, x_{l_2}, x_{l_3}\}$ has exactly one false literal under the truth assignment α, say x_{l_q} for some $q \in \{1, 2, 3\}$.

Therefore, such a path P does not exist in G_I, and consequently we obtain that $\mathsf{zn}(G_I) \leq \mathsf{zn}(G_I, \pi) \leq 2$. □

Next lemma provides the basis for the proof of Lemma 4.6. Intuitively, it establishes that, if the zig-zag number of G_I is at most 2 with respect to some bijection

$\pi\colon V(G_I) \to [|G_I|]$, and C_j and $C_{j'}$ are any two clauses containing a common literal x_i, then, with respect to π, the gadgets $\widetilde{H}_j$ and $\widetilde{H}_{j'}$ are both comparable with H_i and have the same order relative to H_i. The proof of this lemma is by case analysis, and it heavily relies on the fact that, if neither $V(H_i) <_\pi V(\widetilde{H}_j)$ nor $V(H_i) >_\pi V(\widetilde{H}_j)$, then there exist two (not necessarily distinct) vertices $v_j^p, v_j^{p'} \in V(\widetilde{H}_j)$ such that

$$v_j^p <_\pi \max{}_\pi V(H_i) \ \text{ and } \ v_j^{p'} >_\pi \min{}_\pi V(H_i),$$

contradicting therefore the hypothesis that $\mathsf{zn}(G_I, \pi) \leq 2$. For the complete proof, we refer the reader to Lemma 5 of Appendix G.

Lemma 4.5. *Let $I = (X, \mathcal{C})$ be an instance of* PNAE 3SAT, $\pi\colon V(G_I) \to [|G_I|]$ *be a bijection such that $\mathsf{zn}(G_I, \pi) \leq 2$, and let $x_i \in X$. The following statements hold:*

- *If $C_j \in \mathcal{C}$ is a clause containing x_i as a literal, then either $V(H_i) <_\pi V(\widetilde{H}_j)$ or $V(H_i) >_\pi V(\widetilde{H}_j)$;*
- *Furthermore, if there exists a clause $C_j \in \mathcal{C}$ containing x_i as a literal such that $V(H_i) <_\pi V(\widetilde{H}_j)$, then $V(H_i) <_\pi V(\widetilde{H}_{j'})$ for every other clause $C_{j'} \in \mathcal{C}$ containing x_i as a literal.*

Figure 4.8 illustrates an example of a bijection π for which the second statement of Lemma 4.5 is not satisfied, implying therefore $\mathsf{zn}(G_I, \pi) \geq 3$.

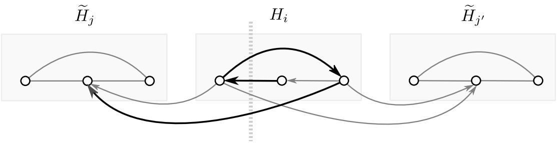


Figure 4.8: An example of a bijection π, such that C_j and $C_{j'}$ are clauses containing the literal x_i, and $V(\widetilde{H}_j) <_\pi V(H_i) <_\pi V(\widetilde{H}_j)$. In this case, $\mathsf{zn}(G_I, \pi) \geq 3$.

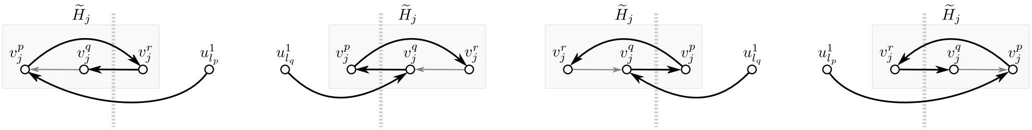
Lemma 4.6. *Let $I = (X, \mathcal{C})$ be an instance of* PNAE 3SAT. *If $\mathsf{zn}(G_I) \leq 2$, then I is a* yes *instance of* PNAE 3SAT.

Proof. Let $\pi\colon V(G_I) \to [|G_I|]$ be a bijection such that $\mathsf{zn}(G_I, \pi) \leq 2$. By Lemma 4.5, for each variable $x_i \in X$ and each clause $C_j \in \mathcal{C}$, if $V(H_i) >_\pi V(\widetilde{H}_j)$, then $V(H_i) >_\pi V(\widetilde{H}_{j'})$ for each clause $C_{j'} \in \mathcal{C}$ containing x_i as a literal. Thus, we let $\alpha\colon x \to \{\mathit{false}, \mathit{true}\}$ be the truth assignment defined as follows: for each variable $x_i \in X$, $\alpha(x_i) = \mathit{true}$ if and only if $V(H_i) >_\pi V(\widetilde{H}_j)$ for each clause $C_j \in \mathcal{C}$.

Now, we prove that each clause in $\mathcal{C}$ has at least one true literal and at least one false literal under α. For the sake of contradiction, suppose that there exists a clause $C_j = \{x_{l_1}, x_{l_2}, x_{l_3}\}$ in $\mathcal{C}$ such that $\alpha(x_{l_1}) = \alpha(x_{l_2}) = \alpha(x_{l_3})$. Let $q \in \{1, 2, 3\}$, $p = q \bmod 3 + 1$ and $r = (q + 1) \bmod 3 + 1$.

Suppose that $\alpha(x_{l_1}) = \alpha(x_{l_2}) = \alpha(x_{l_3}) = \mathit{true}$. Thus, $\{u^1_{l_1}, u^1_{l_2}, u^1_{l_3}\} >_\pi V(\widetilde{H}_j)$. Consequently, if $v^p_j <_\pi v^q_j <_\pi v^r_j$, then $P = (u^1_{l_p}, v^p_j, v^r_j, v^q_j)$ is a directed path of G_I such that $\mathsf{zn}(G_I, \pi, P) = 3$ (see Figures 4.9a); on the other hand, if $v^p_j >_\pi v^q_j >_\pi v^r_j$, then $P = (u^1_{l_q}, v^q_j, v^p_j, v^r_j)$ is a directed path of G_I such that $\mathsf{zn}(G_I, \pi, P) = 3$ (see Figures 4.9c).

Suppose that $\alpha(x_{l_1}) = \alpha(x_{l_2}) = \alpha(x_{l_3}) = \mathit{false}$. Thus, $\{u^1_{l_1}, u^1_{l_2}, u^1_{l_3}\} <_\pi V(\widetilde{H}_j)$. Consequently, if $v^p_j <_\pi v^q_j <_\pi v^r_j$, then $P = (u^1_{l_q}, v^q_j, v^p_j, v^r_j)$ is a directed path of G_I such that $\mathsf{zn}(G_I, \pi, P) = 3$ (see Figures 4.9b); on the other hand, if $v^p_j >_\pi v^q_j >_\pi v^r_j$, then $P = (u^1_{l_p}, v^p_j, v^r_j, v^q_j)$ is a directed path of G_I such that $\mathsf{zn}(G_I, \pi, P) = 3$ (see Figures 4.9d).



(a) (b) (c) (d)

Figure 4.9: (a) and (c) $\alpha(x_{l_1}) = \alpha(x_{l_2}) = \alpha(x_{l_3}) = \mathit{true}$. (b) and (d) $\alpha(x_{l_1}) = \alpha(x_{l_2}) = \alpha(x_{l_3}) = \mathit{false}$. (a) and (b) $v^p_j <_\pi v^q_j <_\pi v^r_j$. (c) and (d) $v^p_j >_\pi v^q_j >_\pi v^r_j$.

Therefore, each clause in $\mathcal{C}$ has at least one true literal and at least one false literal under α, and consequently I is a yes instance of PNAE 3SAT. □

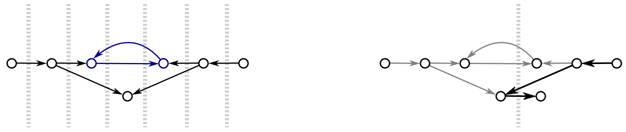
Theorem 4.2. 2-zig-zag Number *is* NP*-complete.*

Proof. By Theorem 4.1, 2-zig-zag Number is in NP. It follows from Lemmas 4.4 and 4.6 that I is a yes instance of PNAE 3SAT if and only if $\mathsf{zn}(G_I) \leq 2$. Therefore, since G_I can be constructed in time polynomial in $|I|$, 2-zig-zag Number is NP-complete. □

4.4 Concluding Remarks

We have proved that one can non-deterministically decide in time $|G|^{\mathcal{O}(k)}$ whether a directed graph G admits zig-zag number at most k, settling therefore k-zig-zag number in NP for each fixed $k \geq 0$. Nevertheless, it remains unknown whether k-zig-zag number admits a non-deterministic FPT-time algorithm. Another question corresponds to determining whether Zig-zag number is also in NP for non-fixed k. It is worth mentioning that, to settle k-zig-zag number in NP, we have actually proved that, given a directed graph G and a bijection $\pi\colon V(G) \to [|G|]$, deciding whether $\mathsf{zn}(G, \pi) \leq k$ is polynomial-time solvable for fixed k. However, for non-fixed k, deciding whether $\mathsf{zn}(G, \pi) \leq k$ is coNP-complete. As a matter of fact, given a bipartite directed graph G with bipartition $V(G) = X \cup Y$, if $\pi\colon V(G) \to [|G|]$ is defined in such a way that $x <_\pi y$ for each $x \in X$ and each $y \in Y$, then deciding whether $\mathsf{zn}(G, \pi) \geq |G|-1$ is equivalent to deciding whether G admits a Hamiltonian path, which is a well-known NP-complete problem [81].

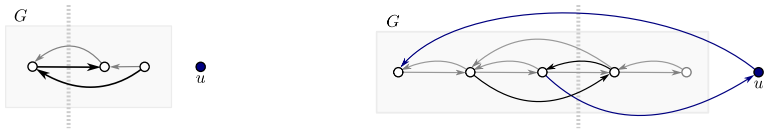
Another intriguing question corresponds to determining whether 1-ZIG-ZAG NUMBER is polynomial-time solvable. As already mentioned, every directed acyclic graph has zig-zag number at most 1, and every directed graph containing directed cycles of length at least 3 must have zig-zag number at least 2. However, there exist directed graphs that are not directed acyclic but still have zig-zag number at most 1 (see Figure 4.10a). Note that, such graphs can only contain directed cycles that are *digons*, *i.e.* directed cycles of length 2. Nevertheless, this is not a sufficient condition for a directed graph to have zig-zag number at most 1, since there exist directed graphs that only contain directed cycles that are digons and yet have zig-zag number at least 2 (see Figure 4.10b). A simple property that seems to be useful to resolve this problem is the fact that, for every directed graph G, $\mathsf{zn}(G) \leq 1$ if and only if there exists a bijection $\pi\colon V(G) \to [|G|]$ such that, for each three distinct vertices $a, b, c \in V(G)$, with $(a, b), (b, c) \in E(G)$, either $a <_\pi b <_\pi c$ or $c <_\pi b <_\pi a$.



(a) $\mathsf{zn}(G) = 1$. (b) $\mathsf{zn}(G) \geq 2$.

Figure 4.10: (a) Example of directed graph G that is not directed acyclic and has zig-zag number 1. (b) Example of directed graph G that does not contain directed cycles of length at least 3 and yet has zig-zag number 2.

Motivated by the NP-hardness of 2-ZIG-ZAG NUMBER, we additionally ask whether k-ZIG-ZAG NUMBER is NP-hard for $k \geq 3$. In particular, determining whether k-ZIG-ZAG NUMBER is polynomially reducible to $(k+1)$-ZIG-ZAG NUMBER is an elusive open problem. Generally, such a reduction must consist in constructing a directed graph H from a given directed graph G, such that $\mathsf{zn}(H) = \mathsf{zn}(G) + 1$. However, since for distinct bijections $\pi\colon V(G) \to [|G|]$ there might exist distinct directed paths P of G such that $\mathsf{zn}(G, \pi, P) = \mathsf{zn}(G, \pi)$, it is not clear how G could be modified to produce a directed graph with zig-zag number exactly one unit greater than $\mathsf{zn}(G)$. For instance, consider the operation of adding a *universal vertex.* There exist directed graphs G such that the addition of a universal vertex results in a directed graph with zig-zag number equal to $\mathsf{zn}(G)$ (see Figure 4.11a); while there also exist directed graphs G such that the addition of a universal vertex results in a directed graph with zig-zag number strictly greater than $\mathsf{zn}(G) + 1$ (see Figure 4.11b).



(a) $\mathsf{zn}(G + u) = \mathsf{zn}(G)$.

(b) $\mathsf{zn}(G + u) = \mathsf{zn}(G) + 2$.

Figure 4.11: (a) Example of directed graph G, with $\mathsf{zn}(G) = 2$, whose addition of a universal vertex u does not increase the zig-zag number. (b) Example of directed graph G, with $\mathsf{zn}(G) = 2$, whose addition of a universal vertex u increases the zig-zag number in more than one unit. (For readability, some edges incident to the universal vertex u are omitted.)

It is worth mentioning that, even if k-ZIG-ZAG NUMBER is proved to be NP-hard for every $k \geq 3$, zig-zag number is still a directed width measure of important theoretical and algorithmic interest. Indeed, in addition to the fact that zig-zag number is asymptotically upper bounded by directed pathwidth [45], there possibly exist efficient approximation algorithms with constant approximation factors for the k-ZIG-ZAG NUMBER problem. Motivated by that, we leave as an open question the existence of such approximation algorithms.

Finally, further investigations to be pursued concern the relationship between zig-zag number and other width measures. It was proved in [45] that directed graphs of constant directed pathwidth have constant zig-zag number, whereas there exist directed graphs of constant zig-zag number but of unbounded directed pathwidth. Nevertheless, it was unknown whether a similar result would hold with respect to zig-zag number and directed tree-width. Then, we proved in [50] that there exist directed graphs of constant directed tree-width but of unbounded zig-zag number. More specifically, we showed that there exists a bidirected graph (*i.e.*, a directed graph whose edge set is a symmetric relation) $\mathbf{B}_n$ on $3n$ vertices that has the complete binary tree B_n on n vertices as an undirected minor, and which can be described as the union of a constant number of directed paths. As a result, we were able to show that, apart from a constant factor, the zig-zag number of $\mathbf{B}_n$ is at least the undirected pathwidth of B_n. In addition, we proved that the undirected tree-width of $\mathbf{B}_n$ is at most 8. Therefore, relying on the facts that the directed tree-width (resp. pathwidth) of a bidirected graph is equal to its undirected tree-width (resp. pathwidth) [2, 84], and that the complete binary tree on n vertices has undirected pathwidth $\Omega(\log n)$ [6, 46], we obtained that the zig-zag number of $\mathbf{B}_n$ is $\Omega(\log n)$, whereas its directed tree-width is at most 8. For more details on this proof, we refer the reader to Section 5 of Appendix G.

Despite the result mentioned above, it remains unknown whether the family of directed graphs of constant directed tree-width contains the family of directed graphs of constant zig-zag number. We remark that a counter-example for such containment would also imply the existence of directed graphs of constant tree-zig-zag number but of unbounded directed tree-width, closing an open question from [46]. Related to this, we additionally ask whether there exists a characterization of zig-zag number in terms of pursuit games.

Chapter 5

Conclusion

In this work, we have analysed the computational complexity of connection and cut problems on graphs.

Concerning connection problems, we have investigated the complexity of the so-called TERMINAL CONNECTION problem (TCP), which is a variant of STEINER TREE that imposes further restrictions on the non-terminal vertices of a solution tree, demanding connection trees for the input terminal set to contain at most ℓ linker vertices (*i.e.*, non-terminal vertices of degree exactly 2 in the tree) and at most r router vertices (*i.e.*, non-terminal vertices of degree at least 3 in the tree).

We have proved that, for fixed r (and arbitrary ℓ), the classes of split graphs and rooted directed graphs separate the complexity of TCP from the complexity of STEINER TREE. In particular, in order to prove that TCP is polynomial-time solvable on split graphs for fixed r, we have employed a polynomial-time algorithm for the STRICT TERMINAL CONNECTION problem (S-TCP) on split graphs as a building block. Moreover, we have shown that, when parameterized by clique-width, TCP is $\mathsf{W}[1]$-hard, whereas STEINER TREE is known to be in FPT.

In Section 2.4, we have posed several questions with respect to TCP and S-TCP that remain unanswered. Next, we highlight the most important of such questions:

- Are TCP and S-TCP parameterized by the number of terminal vertices in FPT, or in XP? It is well-known that STEINER TREE parameterized by the number of terminal vertices is in FPT [56]. Nonetheless, the corresponding parameterization of TCP and of S-TCP has not been settled yet.

- Is S-TCP parameterized by $r \geq 2$ in XP? This problem was addressed in [39, 40], where it was shown to be closely related to the so-called MIN-SUM DISJOINT PATH and SHORTEST K-CYCLE problems.

In addition, as for STEINER TREE, we proved in [34] (see Appendix A for more details) that the problem is NP-complete on undirected path graphs, while it is polynomial-time solvable on undirected path graphs of diameter at most 2. Based on the notions of *vertex leafage* and *leafage* of chordal graphs, we proved in [37] that the problem on undirected path graphs is in FPT when parameterized by the solution size. In spite of STEINER TREE being deeply studied, it is still unknown whether the problem can be solved in polynomial-time when restricted to directed path graphs, which is a subclass of undirected path graphs, and a superclass of rooted directed path graphs (on which the problem is polynomial-time solvable [115]).

Concerning cut problems, we have investigated the complexity of the classical MAXCUT problem. We have shown that the problem remains NP-complete on interval graphs of interval count 4, providing therefore the first complexity classification for the problem on interval graphs of bounded interval count. This result represents an important step in the direction of better understanding the complexity of MAXCUT on unit interval graphs, which are precisely the interval graphs of interval count 1. Besides, we have proved that MAXCUT is also NP-complete on permutation graphs. Next, we describe the main open questions with respect to MAXCUT:

- Is MAXCUT polynomial-time solvable on unit interval graphs?

- Is MAXCUT polynomial-time solvable on interval graphs that are also permutation graphs? This question is equivalent to deciding whether the problem is polynomial-time solvable on permutation graphs that do not have a C_4 as an induced subgraph, since such graphs are precisely interval permutation graphs.

Finally, we have investigated the complexity of k-ZIG-ZAG NUMBER, the problem of deciding whether a given directed graph has zig-zag number at most k. We have proved that, for every fixed k, the problem is in NP. Also, we have shown that, for $k = 2$, the problem is NP-complete. Next, we highlight some of the main open questions described in Section 4.4, with respect to k-ZIG-ZAG NUMBER:

- Is ZIG-ZAG NUMBER in NP for arbitrary k? The running time of the non-deterministic algorithm presented in Section 4.2 depends exponentially on k. Thus, it remains undetermined whether the problem is in NP for non-fixed values of k.

- Is k-ZIG-ZAG NUMBER NP-complete for $k \geq 3$? We have proved that 2-ZIG-ZAG NUMBER is NP-complete. However, it is not clear how to construct a directed graph of zig-zag number $k + 1$ from a directed of zig-zag number k.

In addition to zig-zag number, during the doctoral studies we have developed other works regarding width measures for directed graphs [42–44], from formal language and logic standpoints. These works are related to the notion of *decisional width*, which is built on ordered decision digrams (*i.e.*, read-once oblivious branching programs). In Appendix H, we present one-page abstracts of the obtained results.

References

[1] Adhikary, R., Bose, K., Mukherjee, S., and Roy, B. Complexity of maximum cut on interval graphs. In *37th International Symposium on Computational Geometry, SoCG 2021, June 7-11, 2021, Buffalo, NY, USA (Virtual Conference)* (Dagstuhl, Germany, 2021), K. Buchin and É. C. de Verdière, Eds., vol. 189 of *LIPIcs*, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, pp. 7:1–7:11.

[2] Barát, J. Directed path-width and monotonicity in digraph searching. *Graphs and Combinatorics 22*, 2 (2006), 161–172.

[3] Barsukov, A., Bose, K., and Roy, B. Maximum cut on interval graphs of interval count two is NP-complete. *CoRR abs/2203.06630* (2022).

[4] Bergougnoux, B., and Kanté, M. Fast exact algorithms for some connectivity problems parameterized by clique-width. *Theoretical Computer Science 782* (2019), 30 – 53.

[5] Berman, P., and Karpinski, M. On some tighter inapproximability results (extended abstract). In *Automata, Languages and Programming, 26th International Colloquium, ICALP'99, Prague, Czech Republic, July 11-15, 1999, Proceedings* (Berlin, Heidelberg, 1999), J. Wiedermann, P. van Emde Boas, and M. Nielsen, Eds., vol. 1644 of *Lecture Notes in Computer Science*, Springer, pp. 200–209.

[6] Berwanger, D., Dawar, A., Hunter, P., Kreutzer, S., and Obdrzálek, J. The DAG-width of directed graphs. *Journal of Combinatorial Theory, Series B 102*, 4 (2012), 900–923.

[7] Berwanger, D., Grädel, E., Kaiser, L., and Rabinovich, R. Entanglement and the complexity of directed graphs. *Theoretical Computer Science 463* (2012), 2–25.

[8] Björklund, A., Husfeldt, T., Kaski, P., and Koivisto, M. Fourier meets möbius: Fast subset convolution. In *Proceedings of the Thirty-Ninth*

Annual ACM Symposium on Theory of Computing (New York, NY, USA, 2007), STOC '07, Association for Computing Machinery, p. 67–74.

[9] Bodlaender, H. L., de Figueiredo, C. M. H., Gutierrez, M., Kloks, T., and Niedermeier, R. SIMPLE MAX-CUT for split-indifference graphs and graphs with few P_4s. In *Experimental and Efficient Algorithms, Third International Workshop, WEA 2004, Angra dos Reis, Brazil, May 25-28, 2004, Proceedings* (Berlin, Heidelberg, 2004), C. C. Ribeiro and S. L. Martins, Eds., vol. 3059 of *Lecture Notes in Computer Science*, Springer, pp. 87–99.

[10] Bodlaender, H. L., and Jansen, K. On the complexity of the maximum cut problem. In *Annual Symposium on Theoretical Aspects of Computer Science, STACS 94* (1994), P. Enjalbert, E. W. Mayr, and K. W. Wagner, Eds., vol. 775, Springer, pp. 769–780.

[11] Bodlaender, H. L., Kloks, T., and Niedermeier, R. SIMPLE MAX-CUT for unit interval graphs and graphs with few P_4s. *Electronic Notes in Discrete Mathematics 3* (1999), 19–26.

[12] Bondy, J. A., and Murty, U. S. R. *Graph theory.* Graduate Texts in Mathematics. Springer London, 2008.

[13] Booth, K. S., and Lueker, G. S. Testing for the consecutive ones property, interval graphs, and graph planarity using PQ-tree algorithms. *Journal of Computer and System Sciences 13*, 3 (1976), 335–379.

[14] Borradaile, G., Klein, P. N., and Mathieu, C. Steiner tree in planar graphs: An o(nlogn) approximation scheme with singly-exponential dependence on epsilon. In *Algorithms and Data Structures* (Berlin, Heidelberg, 2007), F. Dehne, J.-R. Sack, and N. Zeh, Eds., Springer Berlin Heidelberg, pp. 275–286.

[15] Boyaci, A., Ekim, T., and Shalom, M. A polynomial-time algorithm for the maximum cardinality cut problem in proper interval graphs. *Information Processing Letters 121* (2017), 29–33.

[16] Brandstädt, A., Hundt, C., Mancini, F., and Wagner, P. Rooted directed path graphs are leaf powers. *Discrete Mathematics 310*, 4 (feb 2010), 897–910.

[17] Byrka, J., Grandoni, F., Rothvoss, T., and Sanità, L. Steiner tree approximation via iterative randomized rounding. *Journal of the ACM 60*, 1 (feb 2013), 1–33.

[18] Cerioli, M. R. Problemas separadores para grafos de caminho. Master's thesis, Universidade Federal do Rio de Janeiro, 1992.

[19] Cerioli, M. R., de S. Oliveira, F., and Szwarcfiter, J. L. The interval count of interval graphs and orders: a short survey. *Journal of the Brazilian Computer Society volume 18*, 2 (2012), 103–112.

[20] Chakraborty, D., Das, S., Foucaud, F., Gahlawat, H., Lajou, D., and Roy, B. Algorithms and complexity for geodetic sets on planar and chordal graphs. In *31st International Symposium on Algorithms and Computation, ISAAC 2020, December 14-18, 2020, Hong Kong, China (Virtual Conference)* (Dagstuhl, Germany, 2020), Y. Cao, S. Cheng, and M. Li, Eds., vol. 181 of *LIPIcs*, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, pp. 7:1–7:15.

[21] Cheng, X., Li, Y., Du, D.-Z., and Ngo, H. Q. *Steiner Trees in Industry.* Springer US, Boston, MA, 2005, pp. 193–216.

[22] Chlebík, M., and Chlebíková, J. The Steiner tree problem on graphs: Inapproximability results. *Theoretical Computer Science 406*, 3 (oct 2008), 207–214.

[23] Cohen, J., Fomin, F. V., Heggernes, P., Kratsch, D., and Kucherov, G. Optimal linear arrangement of interval graphs. In *Mathematical Foundations of Computer Science 2006, 31st International Symposium, MFCS 2006, Stará Lesná, Slovakia, August 28-September 1, 2006, Proceedings* (Berlin, Heidelberg, 2006), R. Kralovic and P. Urzyczyn, Eds., vol. 4162 of *Lecture Notes in Computer Science*, Springer, pp. 267–279.

[24] Colbourn, C. J., and Stewart, L. K. Permutation graphs: connected domination and Steiner trees. *Discrete Mathematics 86*, 1-3 (1990), 179–189.

[25] Cook, S. A. The complexity of theorem-proving procedures. In *Proceedings of the Third Annual ACM Symposium on Theory of Computing* (New York, NY, USA, 1971), STOC '71, Association for Computing Machinery, p. 151–158.

[26] Corneil, D. G., Lerchs, H., and Burlingham, S. L. Complement reducible graphs. *Discrete Applied Mathematics 3*, 3 (1981), 163–174.

[27] Corneil, D. G., Perl, Y., and Stewart, L. K. A linear recognition algorithm for cographs. *SIAM Journal on Computing 14*, 4 (1985), 926–934.

[28] Courcelle, B. The monadic second-order logic of graphs. I. Recognizable sets of finite graphs. *Information and Computation 85*, 1 (1990), 12–75.

[29] Courcelle, B., Engelfriet, J., and Rozenberg, G. Handle-rewriting hypergraph grammars. *Journal of Computer and System Sciences 46*, 2 (1993), 218–270.

[30] Courcelle, B., Makowsky, J. A., and Rotics, U. Linear time solvable optimization problems on graphs of bounded clique-width. *Theory of Computing Systems 33*, 2 (2000), 125–150.

[31] Cygan, M., Fomin, F. V., Kowalik, Ł., Lokshtanov, D., Marx, D., Pilipczuk, M., Pilipczuk, M., and Saurabh, S. *Parameterized Algorithms*. Springer, 2015.

[32] Cygan, M., Pilipczuk, M., Pilipczuk, M., and Wojtaszczyk, J. O. Kernelization hardness of connectivity problems in d-degenerate graphs. *Discrete Applied Mathematics 160*, 15 (2012), 2131 – 2141.

[33] D'Atri, A., and Moscarini, M. Distance-hereditary graphs, Steiner trees, and connected domination. *SIAM Journal on Computing 17*, 3 (1988), 521–538.

[34] de Figueiredo, C. M., de Melo, A. A., Sasaki, D., and Silva, A. Revising Johnson's table for the 21st century. *Discrete Applied Mathematics* (2021).

[35] de Figueiredo, C. M. H., de Melo, A. A., de S. Oliveira, F., and Silva, A. Maximum cut on interval graphs of interval count four is NP-complete. In *46th International Symposium on Mathematical Foundations of Computer Science, MFCS 2021, August 23-27, 2021, Tallinn, Estonia* (Dagstuhl, Germany, 2021), F. Bonchi and S. J. Puglisi, Eds., vol. 202 of *LIPIcs*, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, pp. 38:1–38:15.

[36] de Figueiredo, C. M. H., de Melo, A. A., de S. Oliveira, F., and Silva, A. Maxcut on permutation graphs is NP-complete. *CoRR abs/2202.13955* (2022).

[37] de Figueiredo, C. M. H., Lopes, R., de Melo, A. A., and Silva, A. Parameterized algorithms for Steiner tree and dominating set: Bounding the leafage by the vertex leafage. In *WALCOM: Algorithms and Computation*, P. Mutzel, M. S. Rahman, and Slamin, Eds., Lecture Notes in Computer Science. Springer International Publishing, 2022, pp. 251–262.

[38] de Melo, A., de Figueiredo, C., and de Souza, U. The strict terminal connection problem on chordal bipartite graphs. *Matemática Contemporânea 48*, 14 (2022), 137–145.

[39] de Melo, A. A. Conexão de terminais com limitação de roteadores: Complexidade e relação com fluxos e caminhos disjuntos. Master's thesis, Universidade Federal do Rio de Janeiro, 2017.

[40] de Melo, A. A., de Figueiredo, C. M. H., and dos Santos Souza, U. On undirected two-commodity integral flow, disjoint paths and strict terminal connection problems. *Networks 77*, 4 (2021), 559–571.

[41] de Melo, A. A., de Figueiredo, C. M. H., and Souza, U. S. On the terminal connection problem. In *SOFSEM 2021: Theory and Practice of Computer Science - 47th International Conference on Current Trends in Theory and Practice of Computer Science, SOFSEM 2021, Bolzano-Bozen, Italy, January 25-29, 2021, Proceedings* (2021), T. Bures, R. Dondi, J. Gamper, G. Guerrini, T. Jurdzinski, C. Pahl, F. Sikora, and P. W. H. Wong, Eds., vol. 12607 of *Lecture Notes in Computer Science*, Springer, pp. 278–292.

[42] de Melo, A. A., and de Oliveira Oliveira, M. On the width of regular classes of finite structures. In *Automated Deduction - CADE 27 - 27th International Conference on Automated Deduction, Natal, Brazil, August 27-30, 2019, Proceedings* (2019), P. Fontaine, Ed., vol. 11716 of *Lecture Notes in Computer Science*, Springer, pp. 18–34.

[43] de Melo, A. A., and de Oliveira Oliveira, M. Second-order finite automata. In *Computer Science - Theory and Applications - 15th International Computer Science Symposium in Russia, CSR 2020, Yekaterinburg, Russia, June 29 - July 3, 2020, Proceedings* (2020), H. Fernau, Ed., vol. 12159 of *Lecture Notes in Computer Science*, Springer, pp. 46–63.

[44] de Melo, A. A., and de Oliveira Oliveira, M. Symbolic Solutions for Symbolic Constraint Satisfaction Problems. In *Proceedings of the 17th International Conference on Principles of Knowledge Representation and Reasoning* (9 2020), pp. 49–58.

[45] de Oliveira Oliveira, M. Subgraphs satisfying MSO properties on z-topologically orderable digraphs. In *Parameterized and Exact Computation* (2013), G. Gutin and S. Szeider, Eds., vol. 8246 of *Lecture Notes in Computer Science*, Springer, pp. 123–136.

[46] de Oliveira Oliveira, M. An algorithmic metatheorem for directed treewidth. *Discrete Applied Mathematics 204* (2016), 49–76.

[47] de Ridder et al., H. N. Graphclass: comparability graphs. information system on graph classes and their inclusions (isgci). `https://www.graphclasses.org/classes/gc_72.html`. Accessed: 2022-02-17.

[48] de Souza Oliveira, F. *Problemas Separadores para Grafos de Caminho.* PhD thesis, Universidade Federal do Rio de Janeiro, 2011.

[49] Demaine, E. D., Hajiaghayi, M., and Klein, P. N. Node-weighted Steiner tree and group Steiner tree in planar graphs. *ACM Transactions on Algorithms 10*, 3 (jun 2014), 1–20.

[50] Dourado, M. C., de Figueiredo, C. M., de Melo, A. A., de Oliveira Oliveira, M., and Souza, U. S. Computing the zig-zag number of directed graphs. *Discrete Applied Mathematics 312* (2022), 86–105.

[51] Dourado, M. C., Oliveira, R. A., Protti, F., and Souza, U. S. Conexão de terminais com número restrito de roteadores e elos. In *Proceedings of XLVI Simpósio Brasileiro de Pesquisa Operacional* (2014), pp. 2965–2976.

[52] Dourado, M. C., Oliveira, R. A., Protti, F., and Souza, U. S. Design of connection networks with bounded number of non-terminal vertices. *Matemática Contemporânea 42*, 14 (2014), 39–47.

[53] Downey, R. G., and Fellows, M. R. Parameterized computational feasibility. In *Feasible Mathematics II*, P. Clote and J. B. Remmel, Eds., Progress in Computer Science and Applied Logic. Birkhäuser Boston, 1995, pp. 219–244.

[54] Downey, R. G., and Fellows, M. R. *Parameterized Complexity.* Springer-Verlag, 1999. Monographs in Computer Science.

[55] Downey, R. G., and Fellows, M. R. *Fundamentals of Parameterized Complexity.* Texts in Computer Science. Springer, 2013.

[56] Dreyfus, S. E., and Wagner, R. A. The Steiner problem in graphs. *Networks 1*, 3 (1971), 195–207.

[57] Ekim, T., Erey, A., Heggernes, P., van 't Hof, P., and Meister, D. Computing minimum geodetic sets of proper interval graphs. In *LATIN 2012: Theoretical Informatics - 10th Latin American Symposium, Arequipa, Peru, April 16-20, 2012. Proceedings* (Berlin, Heidelberg, 2012), D. Fernández-Baca, Ed., vol. 7256 of *Lecture Notes in Computer Science*, Springer, pp. 279–290.

[58] Farber, M. Characterizations of strongly chordal graphs. *Discrete Mathematics 43*, 2 (1983), 173 – 189.

[59] Farber, M. R. *Applications of Linear Programming Duality to Problems Involving Independence and Domination.* Technical report (Simon Fraser University. Department of Computing Science). 1981.

[60] Fellows, M. R., Rosamond, F. A., Rotics, U., and Szeider, S. Clique-width is NP-complete. *SIAM Journal on Discrete Mathematics 23*, 2 (2009), 909–939.

[61] Fishburn, P. C. Interval graphs and interval orders. *Discrete Mathematics 55*, 2 (1985), 135–149.

[62] Fomin, F. V., Golovach, P. A., Lokshtanov, D., and Saurabh, S. Clique-width: on the price of generality. In *Proceedings of the Twentieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2009, New York, NY, USA, January 4-6, 2009* (2009), C. Mathieu, Ed., SIAM, pp. 825–834.

[63] Gallai, T. Transitiv orientierbare graphen. *Acta Mathematica Hungarica 18*, 1-2 (1967), 25–66.

[64] Ganian, R., Hlinený, P., Kneis, J., Langer, A., Obdrzálek, J., and Rossmanith, P. On digraph width measures in parameterized algorithmics. In *Parameterized and Exact Computation, 4th International Workshop, IWPEC 2009, Copenhagen, Denmark, September 10-11, 2009, Revised Selected Papers* (2009), J. Chen and F. V. Fomin, Eds., vol. 5917 of *LNCS*, pp. 185–197.

[65] Garey, M. R., and Johnson, D. S. The rectilinear Steiner tree problem is NP-complete. *SIAM Journal on Applied Mathematics 32*, 4 (1977), 826–834.

[66] Garey, M. R., and Johnson, D. S. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W. H. Freeman & Co., New York, 1979.

[67] Garey, M. R., Johnson, D. S., and Stockmeyer, L. J. Some simplified NP-complete graph problems. *Theoretical Computer Science 1*, 3 (1976), 237–267.

[68] Gargano, L., Hammar, M., Hell, P., Stacho, L., and Vaccaro, U. Spanning spiders and light-splitting switches. *Discrete Mathematics 285*, 1 (2004), 83 – 95.

[69] Gavril, F. The intersection graphs of subtrees in trees are exactly the chordal graphs. *Journal of Combinatorial Theory, Series B 16*, 1 (1974), 47–56.

[70] Gavril, F. A recognition algorithm for the intersection graphs of directed paths in directed trees. *Discrete Mathematics 13*, 3 (1975), 237–249.

[71] Gavril, F. A recognition algorithm for the intersection graphs of paths in trees. *Discrete Mathematics 23*, 3 (1978), 211–227.

[72] Gilmore, P. C., and Hoffman, A. J. A characterization of comparability graphs and of interval graphs. *Canadian Journal of Mathematics 16* (1964), 539–548.

[73] Golumbic, M. C. Trivially perfect graphs. *Discrete Mathematics 24* (1978), 105–107.

[74] Golumbic, M. C. *Algorithmic Graph Theory and Perfect Graphs (Annals of Discrete Mathematics, Vol 57).* North-Holland Publishing Co., NLD, 2004.

[75] Hadlock, F. Finding a maximum cut of a planar graph in polynomial time. *SIAM Journal on Computing 4*, 3 (sep 1975), 221–225.

[76] Halperin, E., Kortsarz, G., Krauthgamer, R., Srinivasan, A., and Wang, N. Integrality ratio for group Steiner trees and directed Steiner trees. *SIAM Journal on Computing 36*, 5 (jan 2007), 1494–1511.

[77] Hans Jürgen Prömel, A. S. *The Steiner Tree Problem.* Vieweg+Teubner Verlag, 2012.

[78] Hunter, P., and Kreutzer, S. Digraph measures: Kelly decompositions, games, and orderings. *Theoretical Computer Science 399*, 3 (2008), 206–219.

[79] Hwang, F. A linear time algorithm for full Steiner trees. *Operations Research Letters 4*, 5 (1986), 235 – 237.

[80] Hwang, F. K., Richards, D. S., and Winter, P. *The Steiner tree problem*, vol. 53 of *Annals of Discrete Mathematics.* Elsevier, 1992.

[81] Itai, A., Papadimitriou, C. H., and Szwarcfiter, J. L. Hamilton paths in grid graphs. *SIAM Journal on Computing 11*, 4 (nov 1982), 676–686.

[82] Jinjiang, Y., and Sanming, Z. Optimal labelling of unit interval graphs. *Applied Mathematics 10*, 3 (sep 1995), 337–344.

[83] Johnson, D. S. The NP-completeness column: An ongoing guide. *Journal of Algorithms 6*, 3 (1985), 434–451.

[84] Johnson, T., Robertson, N., Seymour, P. D., and Thomas, R. Directed tree-width. *Journal of Combinatorial Theory, Series B 82*, 1 (2001), 138–154.

[85] Jones, M., Lokshtanov, D., Ramanujan, M. S., Saurabh, S., and Suchý, O. Parameterized complexity of directed Steiner tree on sparse graphs. *SIAM Journal on Discrete Mathematics 31*, 2 (jan 2017), 1294–1327.

[86] Karp, R. M. *Reducibility among Combinatorial Problems.* Springer US, Boston, MA, 1972, pp. 85–103.

[87] Klein, P., and Ravi, R. A nearly best-possible approximation algorithm for node-weighted Steiner trees. *Journal of Algorithms 19*, 1 (jul 1995), 104–115.

[88] Korte, B., Lovász, L., Prömel, H. J., and Schrijver, A. *Paths, flows, and VLSI-layout.* Springer, 1990.

[89] Kratochvíl, J., Masarík, T., and Novotná, J. U-bubble model for mixed unit interval graphs and its applications: The maxcut problem revisited. In *45th International Symposium on Mathematical Foundations of Computer Science, MFCS 2020, August 24-28, 2020, Prague, Czech Republic* (Dagstuhl, Germany, 2020), J. Esparza and D. Král', Eds., vol. 170 of *LIPIcs*, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, pp. 57:1–57:14.

[90] Leibowitz, R., Assmann, S. F., and Peck, G. W. The interval count of a graph. *SIAM Journal on Algebraic Discrete Methods 3*, 4 (dec 1982), 485–494.

[91] Lin, G., and Xue, G. On the terminal Steiner tree problem. *Information Processing Letters 84*, 2 (2002), 103–107.

[92] Lu, C. L., Tang, C. Y., and Lee, R. C.-T. The full Steiner tree problem. *Theoretical Computer Science 306*, 1-3 (2003), 55–67.

[93] Mahajan, M., and Raman, V. Parameterizing above guaranteed values: Maxsat and maxcut. *Journal of Algorithms 31*, 2 (1999), 335–354.

[94] Marx, D. A short proof of the NP-completeness of minimum sum interval coloring. *Operations Research Letters 33*, 4 (2005), 382–384.

[95] Melo, A. A., Figueiredo, C. M. H., and Souza, U. S. Connecting terminals using at most one router. *Matemática Contemporânea 45* (2017), 49–57.

[96] Melo, A. A., Figueiredo, C. M. H., and Souza, U. S. A multivariate analysis of the strict terminal connection problem. *Journal of Computer and System Sciences 111* (2020), 22–41.

[97] Mölle, D., Richter, S., and Rossmanith, P. Enumerate and expand: Improved algorithms for connected vertex cover and tree cover. *Theory Comput. Syst. 43*, 2 (2008), 234–253.

[98] Monma, C. L., and Wei, V. K. Intersection graphs of paths in a tree. *Journal of Combinatorial Theory, Series B 41*, 2 (1986), 141–181.

[99] Müller, H., and Brandstädt, A. The NP-completeness of Steiner tree and dominating set for chordal bipartite graphs. *Theoretical Computer Science 53*, 2-3 (1987), 257–265.

[100] Nederlof, J. Fast polynomial-space algorithms using inclusion-exclusion. *Algorithmica 65*, 4 (2013), 868–884.

[101] Nicoloso, S., Sarrafzadeh, M., and Song, X. On the sum coloring problem on interval graphs. *Algorithmica 23*, 2 (1999), 109–126.

[102] Panda, B. The forbidden subgraph characterization of directed vertex graphs. *Discrete Mathematics 196*, 1-3 (feb 1999), 239–256.

[103] Panda, B. S., and Pradhan, D. NP-completeness of Hamiltonian cycle problem on rooted directed path graphs. *CoRR abs/0809.2443* (2008).

[104] Papadimitriou, C. *Computational Complexity.* Theoretical computer science. Addison-Wesley, 1994.

[105] Pnueli, A., Lempel, A., and Even, S. Transitive orientation of graphs and identification of permutation graphs. *Canadian Journal of Mathematics 23* (1971), 160–175.

[106] Pocai, R. V. The complexity of SIMPLE MAX-CUT on comparability graphs. *Electronic Notes in Discrete Mathematics 55* (nov 2016), 161–164.

[107] Reed, B. A. Introducing directed tree width. *Electronic Notes in Discrete Mathematics 3* (1999), 222–229.

[108] Roberts, F. Indifference graphs, F. Harary (ed.), proof techniques in graph theory, 1969.

[109] Safari, M. A. D-width: A more natural measure for directed tree width. In *Mathematical Foundations of Computer Science 2005, 30th International Symposium, MFCS 2005, Gdansk, Poland, August 29 - September 2, 2005, Proceedings* (2005), J. Jedrzejowicz and A. Szepietowski, Eds., vol. 3618 of *Lecture Notes in Computer Science*, Springer, pp. 745–756.

[110] Schaefer, T. J. The complexity of satisfiability problems. In *Proceedings of the 10th Annual ACM Symposium on Theory of Computing, May 1-3, 1978, San Diego, California, USA* (1978), R. J. Lipton, W. A. Burkhard, W. J. Savitch, E. P. Friedman, and A. V. Aho, Eds., ACM, pp. 216–226.

[111] Spinrad, J. P. *Efficient Graph Representations.* Fields Institute monographs. American Mathematical Society, Providence, RI, 2003.

[112] Voss, S. *Steiner Tree Problems in Telecommunications.* Springer US, Boston, MA, 2006, pp. 459–492.

[113] Watel, D., Weisser, M., Bentz, C., and Barth, D. Steiner problems with limited number of branching nodes. In *Structural Information and Communication Complexity - 20th International Colloquium, SIROCCO 2013, Ischia, Italy, July 1-3, 2013, Revised Selected Papers* (2013), T. Moscibroda and A. A. Rescigno, Eds., vol. 8179 of *Lecture Notes in Computer Science*, Springer, pp. 310–321.

[114] Watel, D., Weisser, M.-A., Bentz, C., and Barth, D. Directed Steiner trees with diffusion costs. *Journal of Combinatorial Optimization 32*, 4 (2016), 1089–1106.

[115] White, K., Farber, M., and Pulleyblank, W. Steiner trees, connected domination and strongly chordal graphs. *Networks 15*, 1 (1985), 109–124.

Appendix A

Manuscript: Revising Johnson's Table for the 21st Century

This appendix contains the manuscript:
Celina M. H. de Figueiredo, Alexsander A. de Melo, Diana Sasaki, Ana Silva. Revising Johnson's Table for the 21st Century. Accepted for publication in *Discrete Applied Mathematics* [34].

ARTICLE IN PRESS

Discrete Applied Mathematics xxx (xxxx) xxx

Contents lists available at ScienceDirect

Discrete Applied Mathematics

journal homepage: www.elsevier.com/locate/dam

Revising Johnson's table for the 21st century

Celina M.H. de Figueiredo [a], Alexsander A. de Melo [a,*], Diana Sasaki [b], Ana Silva [c]

[a] Federal University of Rio de Janeiro, Rio de Janeiro, Brazil
[b] Rio de Janeiro State University, Rio de Janeiro, Brazil
[c] Federal University of Ceará, Ceará, Brazil

ARTICLE INFO

Article history:
Received 29 May 2020
Received in revised form 21 December 2020
Accepted 17 May 2021
Available online xxxx

Keywords:
Computational complexity
Parameterized complexity
NP-complete
Steiner tree
Dominating set

ABSTRACT

What does it mean today to study a problem from a computational point of view? We focus on parameterized complexity and on Column 16 "Graph Restrictions and Their Effect" of D.S. Johnson's Ongoing guide, where several puzzles were proposed in a summary table with 30 graph classes as rows and 11 problems as columns. Several of the 330 entries remain unclassified into Polynomial or NP-complete after 35 years. We provide a full dichotomy for the Steiner Tree column by proving that the problem is NP-complete when restricted to Undirected Path graphs. We revise Johnson's summary table according to the granularity provided by the parameterized complexity for NP-complete problems.

1. Graph restrictions and their effect 35 years later

The 1979 book *Computers and Intractability, A Guide to the Theory of* NP*-completeness* by Michael R. Garey and David S. Johnson [53] is considered the single most important book by the computational complexity community and it is known as The Guide, which we cite by [GJ]. The book was followed by *The* NP*-completeness Column: An Ongoing Guide* where, from 1981 until 2007, D.S. Johnson continuously updated The Guide in 26 columns published first in the *Journal of Algorithms* and then in the *ACM Transactions on Algorithms*. The Guide has an appendix where 300 NP-complete problems are organized into 13 categories according to subject matter. The first, "A1 Graph Theory", contains 65 problems and the second, "A2 Network Design", contains 51 problems. Category "A13 Open Problems" lists 12 problems in NP, at the time not classified into polynomial or NP-complete, and it is surprising that since then 5 have been classified into polynomial and 5 into NP-complete. Garey and Johnson were amazingly able to foresee a list of challenging problems which would evenly be split into tractable and intractable problems. The goal of the present paper is to propose an answer to the question: *What does it mean today to study a problem from a computational complexity point of view?* In search of an answer, we focus on parameterized complexity and on Column 16 "Graph Restrictions and Their Effect" [66], which we cite by [OG], where several puzzles were proposed by D.S. Johnson and many remain unsolved after 35 years. Consider in Table 1 the summary table from [OG] with 30 graph classes as rows and 11 columns, the first of which is Membership, followed by 10 well-known NP-complete problems, listed in The Guide's appendix as GT20, GT19, GT15, GT13, Open5, GT37, GT2, ND16, ND12, and Open1. The entries follow the notation of [OG], where the complexity of the problem restricted to the graph class is N = NP-complete, P = polynomial, or O = open. Following our convention, reference [GJ] stands for "The Guide" and [OG] stands for "Column 16", and we highlight in bold the reference updates with the corresponding new

[*] Corresponding author.
E-mail addresses: celina@cos.ufrj.br (C.M.H. de Figueiredo), aamelo@cos.ufrj.br (A.A. de Melo), diana.sasaki@ime.uerj.br (D. Sasaki), anasilva@mat.ufc.br (A. Silva).

https://doi.org/10.1016/j.dam.2021.05.021
0166-218X/© 2021 Elsevier B.V. All rights reserved.

Table 1
The updated NP-Completeness Column: An Ongoing Guide table 35 years later. Depicted in bold are the references that correspond to unresolved entries in [OG] and [GJ]. The references not in bold confirm resolved entries from [OG] or [GJ], that we updated either because they cited private communications, because the cited reference is not easily accessible, or could not be confirmed. There is one entry highlighted in italic that corrects the entry for HamCirc restricted to Circle graphs. We keep the abbreviations used by [OG], namely for entries: P = Polynomial-time solvable; N = NP-complete; I = Open, but equivalent in complexity to general Graph Isomorphism; O? = Apparently open, but possibly easy to resolve; and O = Open, and may well be hard; and for references [T] = Restriction trivializes the problem; [GJ] = the Guide [53]; and [OG] = the Ongoing guide [66], please refer to this reference for the entry.

Graph Class	Member	IndSet	Clique	CliPar	ChrNum	ChrInd	HamCir	DomSet	MaxCut	StTree	GraphIso
Trees/Forests	P [T]	P [GJ]	P [T]	P [GJ]	P [T]	P [GJ]	P [T]	P [GJ]	P [GJ]	P [T]	P [GJ]
Almost Trees (k)	P [OG]	P [OG]	P [T]	P [16]	P [5]	P [17]	P [5]	P [5]	P [20]	P [76]	P [17]
Partial k-trees	P [OG]	P [5]	P [T]	P [16]	P [5]	P [17]	P [5]	P [5]	P [20]	P [76]	P [17]
Bandwidth-k	P [OG]	P [OG]	P [T]	P [16]	P [5]	P [17]	P [5]	P [5]	P [OG]	P [76]	P [OG]
Degree-k	P [T]	N [GJ]	P [T]	N [29]	N [GJ]	N [OG]	N [GJ]	N [GJ]	N [GJ]	N [GJ]	P [OG]
Planar	P [GJ]	N [GJ]	P [T]	N [78]	N [GJ]	O	N [GJ]	N [GJ]	P [GJ]	N [OG]	P [GJ]
Series Parallel	P [OG]	P [OG]	P [T]	P [16]	P [5]	P [17]	P [5]	P [OG]	P [GJ]	P [OG]	P [GJ]
Outerplanar	P [OG]	P [OG]	P [T]	P [OG]	P [OG]	P [OG]	P [T]	P [OG]	P [GJ]	P [OG]	P [GJ]
Halin	P [OG]	P [OG]	P [T]	P [OG]	P [5]	P [17]	P [T]	P [OG]	P [GJ]	P [118]	P [GJ]
k-Outerplanar	P [OG]	P [OG]	P [T]	P [OG]	P [5]	P [17]	P [OG]	P [OG]	P [GJ]	P [76]	P [GJ]
Grid	P [OG]	P [GJ]	P [T]	P [GJ]	P [T]	P [GJ]	N [OG]	N [32]	P [T]	N [OG]	P [GJ]
$K_{3,3}$-Free*	P [OG]	N [GJ]	P [T]	N [78]	N [GJ]	O?	N [GJ]	N [GJ]	P [OG]	N [GJ]	P [40]
Thickness-k	N [OG]	N [GJ]	P [T]	N [78]	N [GJ]	N [OG]	N [GJ]	N [GJ]	N [119]	N [GJ]	**I** Proposition 3
Genus-k	P [OG]	N [GJ]	P [T]	N [78]	N [GJ]	O?	N [GJ]	N [GJ]	O?	N [GJ]	P [OG]
Perfect	P [34]	P [OG]	P [OG]	P [OG]	P [OG]	**N** [28]	N [OG]	N [OG]	**N** [20]	N [GJ]	I [84]
Chordal	P [OG]	P [OG]	P [OG]	P [OG]	P [OG]	O?	N [93]	N [OG]	**N** [20]	N [OG]	I [84]
Split	P [OG]	P [OG]	P [OG]	P [OG]	P [OG]	O?	N [93]	N [OG]	**N** [20]	N [OG]	I [108]
Strongly Chordal	P [OG]	P [OG]	P [OG]	P [OG]	P [OG]	O?	**N** [93]	P [OG]	**N** [109]	P [OG]	**I** [111]
Comparability	P [OG]	P [OG]	P [OG]	P [OG]	P [OG]	**N** [28]	N [OG]	N [94]	**N** [102]	N [GJ]	I [22]
Bipartite	P [T]	P [GJ]	P [T]	P [GJ]	P [T]	P [GJ]	N [OG]	N [94]	P [T]	N [GJ]	I [22]
Permutation	P [OG]	P [OG]	P [OG]	P [OG]	P [OG]	O?	P [44]	P [OG]	O?	P [OG]	P [OG]
Cographs	P [T]	P [OG]	P [OG]	P [OG]	P [OG]	O?	P [OG]	P [OG]	P [20]	P [OG]	P [OG]
Undirected Path	P [OG]	P [OG]	P [OG]	P [OG]	P [OG]	O?	**N** [13]	N [OG]	**N** [20]	**N** Theorem 4	I [22]
Directed Path	P [OG]	P [OG]	P [OG]	P [OG]	P [OG]	O?	**N** [99]	P [OG]	**N** [1]	P [OG]	P [7]
Interval	P [OG]	P [OG]	P [OG]	P [OG]	P [OG]	O?	P [OG]	P [OG]	**N** [1]	P [OG]	P [OG]
Circular Arc	P [OG]	P [OG]	P [OG]	P [OG]	N [OG]	O?	P [106]	P [OG]	**N** [1]	P [11]	P [80]
Circle	P [OG]	P [GJ]	P [OG]	**N** [73]	N [OG]	O?	*N* *[39]*	**N** [71]	**N** [26]	P [OG]	P [68]
Proper Circ. Arc	P [OG]	P [OG]	P [OG]	P [OG]	P [OG]	O?	P [OG]	P [OG]	O?	P [11]	P [82]
Edge (or Line)	P [OG]	P [GJ]	P [T]	N [95]	N [OG]	**N** [28]	N [OG]	N [GJ]	P [59]	N [19]	I [OG]
Claw-Free	P [T]	P [OG]	**N** [103]	N [85]	N [OG]	**N** [28]	N [OG]	N [GJ]	**N** [20]	N [19]	I [OG]

recent references. Every reference associated with each entry of Table 1 has been checked, and the updated entries are precisely those that needed to be updated. It is surprising that several O? entries remain stubbornly open. At the time, D.S. Johnson proposed only one "O! = famous open problem", Membership for Perfect graphs, which we know today to be in P, and two entries "O = may well be hard", Hamiltonian Circuit restricted to Permutation graphs, known today to be in P, and Chromatic Index restricted to Planar graphs, which is still open. We remark that in the original summary table [OG], there was only one entry co-authored by a Brazilian researcher among 330 entries, namely Hamiltonian Circuit restricted to Grids [64], and today we have two additional such entries: Maximum Cut restricted to Strongly Chordal [109] and Graph Isomorphism restricted to Proper Circular Arc [82].

We depict in Figs. 1 and 2 the relations between the graph classes, and use the convention from [OG] that an arrow from Class A to Class B means that Class A contains Class B. Since the only O! entry was Membership for Perfect graphs, the chosen 30 classes were classified into the following four categories: Trees and Near-Trees, Planarity and its Relations, A Catalog of Perfect Graphs, and Intersection Graphs. Although very similar to the figures presented in [OG], our Figs. 1 and 2 present some additional relations, that were either unknown or unobserved, and about which we comment next. Unless explicitly mentioned, we follow the definitions from [OG]. According to it, Undirected Path graphs can be modeled by a set of paths in a tree, and Directed Path graphs are undirected path graphs whose representation is such that for some root vertex in the tree, all paths are subpaths of paths from the root to a leaf. We refer to [91] for the variations called UV, DV and RDV of intersection graphs of a family of undirected or directed vertex paths in an undirected or in a directed tree.

Fig. 1 highlights the key property that 7 graph classes are subclasses of Partial k-Trees [18], also known as Bounded Treewidth graphs. Also, although Table 1 follows the same organization as the one used in [OG], our proposed new Table 2 is organized in a way as to highlight this relationship, with the first 8 rows being exactly Partial k-Trees and its subclasses. In the summary table, D.S. Johnson used entry "P? = appears to be polynomial-time solvable by standard techniques, but I haven't checked the details". D.S. Johnson is correct, since all P? entries are known today to be P entries. All former P? entries used to appear in these 8 rows, Partial$k$-Trees and the 7 subclasses [18].

Another difference is that we use $K_{3,3}$-Free* to denote the graph class referred to in [OG] by $K_{3,3}$-Free. This is to avoid confusion, since nowadays it is standard to use H-Free to refer to the class of graphs that do not contain H as

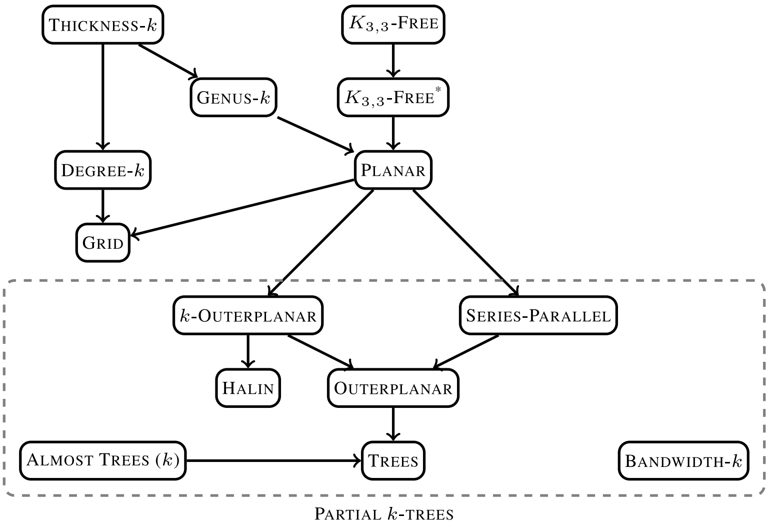


Fig. 1. Containment relations for classes from [OG], where, in particular, the subclasses of PARTIAL k-TREES are highlighted. A graph class CLASS A has an arrow to a graph class CLASS B if CLASS A contains CLASS B.
Source: Adapted from [OG].

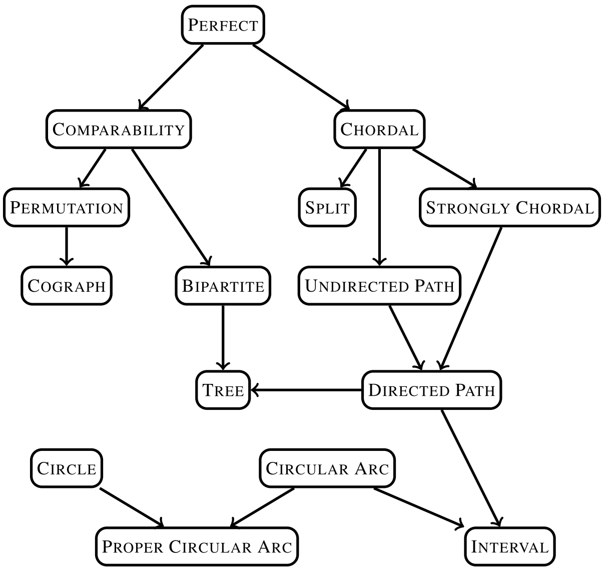


Fig. 2. Containment relations for classes from [OG], our target class is UNDIRECTED PATH.
Source: Adapted from [OG].

an *induced subgraph*. However, the class investigated in [OG] is instead the class of graphs that contain no subgraphs *homeomorphic* to $K_{3,3}$; in other words, the class of graphs that do not contain $K_{3,3}$ as a topological minor. Observe that, using our notation, we have that $K_{3,3}$-FREE* is a proper subclass of $K_{3,3}$-FREE. Also, we mention that this confusion does not occur for CLAW-FREE graphs, since we, as well as [OG], use it to denote the class of graphs that do not contain $K_{1,3}$ (also known as the claw) as an induced subgraph.

Finally, we mention two relations that do not appear in [OG], both involving the class THICKNESS-k. A graph G is said to have *thickness at most* k if $E(G)$ can be partitioned into at most k subsets, each of which forms a planar subgraph of G. In the same way as all the other graph classes that have a parameter in their names such as the class of PARTIAL k-TREES that is also known as BOUNDED TREEWIDTH graphs, the class THICKNESS-k means BOUNDED THICKNESS graphs. First, note that if G has degree at most $\Delta(G)$, then by Vizing's Theorem, we get that $E(G)$ can be colored with at most $\Delta(G) + 1$ colors. In other words, this means that the edge set of G can be partitioned into $\Delta(G) + 1$ matchings, which are planar graphs, and hence G has thickness at most $\Delta(G) + 1$. Therefore, we get that DEGREE-k is a subclass of THICKNESS-k. Another non-trivial

Table 2

The parameterized NP-Completeness Column: An Ongoing Guide table revised for the 21st century. The parameterized puzzle is to classify every O entry, every O? entry and every N entry into FPT = Fixed parameter tractable, W_1= W [1]-hard, W_2= W [2]-hard, and pN = paraNP-complete, where the considered parameterization is with respect to the natural parameter of each corresponding problem. We highlight as O* the N entry of Table 1 that constitutes the parameterized puzzle, for which so far we were not able to provide a parameterized complexity classification.

Graph Class	Member	IndSet	Clique	CliPar	ChrNum	ChrInd	HamCir	DomSet	MaxCut	κ-StTree	GraphIso
Partial k-trees	P [OG]	P [5]	P [T]	P [16]	P [5]	P [17]	P [5]	P [5]	P [20]	P [76]	P [17]
Trees/Forests	P [T]	P [GJ]	P [T]	P [GJ]	P [T]	P [GJ]	P [T]	P [GJ]	P [GJ]	P [T]	P [GJ]
Almost Trees (k)	P [OG]	P [OG]	P [T]	P [16]	P [5]	P [17]	P [5]	P [5]	P [20]	P [76]	P [17]
Bandwidth-k	P [OG]	P [OG]	P [T]	P [16]	P [5]	P [17]	P [5]	P [5]	P [OG]	P [76]	P [OG]
Series Parallel	P [OG]	P [OG]	P [T]	P [16]	P [5]	P [17]	P [5]	P [OG]	P [GJ]	P [OG]	P [GJ]
Outerplanar	P [OG]	P [OG]	P [T]	P [OG]	P [OG]	P [OG]	P [T]	P [OG]	P [GJ]	P [OG]	P [GJ]
Halin	P [OG]	P [OG]	P [T]	P [OG]	P [5]	P [17]	P [T]	P [OG]	P [GJ]	P [118]	P [GJ]
k-Outerplanar	P [OG]	P [OG]	P [T]	P [OG]	P [5]	P [17]	P [OG]	P [OG]	P [GJ]	P [76]	P [GJ]
Planar	P [GJ]	FPT [96]	P [T]	FPT [T]	pN [55]	O	FPT [89]	FPT [47]	P [GJ]	FPT [101]	P [GJ]
Grid	P [OG]	P [GJ]	P [T]	P [GJ]	P [T]	P [GJ]	FPT [89]	FPT [47]	P [T]	FPT [101]	P [GJ]
$K_{3,3}$-Free*	P [OG]	FPT [42]	P [T]	FPT [T]	pN [55]	O?	FPT [89]	FPT [100]	P [OG]	XP [T]	P [40]
Thickness-k	pN [OG]	FPT [74]	P [T]	FPT [T]	pN [55]	pN [63]	FPT [89]	XP [T]	FPT [86]	XP [T]	FPT [6]
Genus-k	P [OG]	FPT [30]	P [T]	FPT [T]	pN [55]	O?	FPT [89]	FPT [51]	FPT [86]	FPT [101]	P [OG]
Degree-k	P [T]	FPT [48]	P [T]	FPT [T]	pN [55]	pN [63]	FPT [89]	FPT [3]	FPT [86]	FPT [67]	P [OG]
Perfect	P [34]	P [OG]	P [OG]	P [OG]	P [OG]	pN [28]	FPT [89]	W_2 [104]	FPT [86]	W_2 [104]	FPT [6]
Chordal	P [OG]	P [OG]	P [OG]	P [OG]	P [OG]	O?	FPT [89]	W_2 [104]	FPT [86]	W_2 [104]	FPT [6]
Split	P [OG]	P [OG]	P [OG]	P [OG]	P [OG]	O?	FPT [89]	W_2 [104]	FPT [86]	W_2 [104]	FPT [6]
Strongly Chordal	P [OG]	P [OG]	P [OG]	P [OG]	P [OG]	O?	FPT [89]	P [OG]	FPT [86]	P [OG]	FPT [6]
Comparability	P [OG]	P [OG]	P [OG]	P [OG]	P [OG]	pN [28]	FPT [89]	W_2 [104]	FPT [86]	W_2 Proposition 1	FPT [6]
Bipartite	P [T]	P [GJ]	P [T]	P [GJ]	P [T]	P [GJ]	FPT [89]	W_2 [104]	P [T]	W_2 Proposition 1	FPT [6]
Permutation	P [OG]	P [OG]	P [OG]	P [OG]	P [OG]	O?	P [44]	P [OG]	FPT [86]	P [OG]	P [OG]
Cographs	P [T]	P [OG]	P [OG]	P [OG]	P [OG]	O?	P [OG]	P [OG]	P [20]	P [OG]	P [OG]
Undirected Path	P [OG]	P [OG]	P [OG]	P [OG]	P [OG]	O?	FPT [89]	XP [T]	FPT [86]	XP [T]	FPT [6]
Directed Path	P [OG]	P [OG]	P [OG]	P [OG]	P [OG]	O?	FPT [89]	P [OG]	FPT [86]	P [OG]	P [7]
Interval	P [OG]	P [OG]	P [OG]	P [OG]	P [OG]	O?	P [OG]	P [OG]	FPT [86]	P [OG]	P [OG]
Circular Arc	P [OG]	P [OG]	P [OG]	P [OG]	FPT [54]	O?	P [106]	P [OG]	FPT [86]	P [11]	P [80]
Circle	P [OG]	P [GJ]	P [OG]	XP [73]	pN [112]	O?	FPT [89]	W_1 [24]	FPT [86]	P [OG]	P [68]
Proper Circ. Arc	P [OG]	P [OG]	P [OG]	P [OG]	P [OG]	O?	P [OG]	P [OG]	FPT [86]	P [11]	P [82]
Edge (or Line)	P [OG]	P [GJ]	P [T]	O* [95]	pN [63]	pN [28]	FPT [89]	FPT [38]	P [59]	XP [T]	FPT [6]
Claw-Free	P [T]	P [OG]	FPT [38]	pN [85]	pN [63]	pN [28]	FPT [89]	FPT [38]	FPT [86]	XP [T]	FPT [6]

relation involving this class is with Genus-k graphs. A *k-book embedding* of a graph G is a linear ordering of its vertices along the spine of a book and an assignment of its edges to k pages so that edges assigned to the same page can be drawn on that page without crossings. The *pagenumber* of a graph G is the minimum k for which G admits a k-book embedding. Clearly, the pagenumber of G is an upper bound for the thickness of G. Additionally, in [87] the authors prove that the pagenumber of G is bounded above by a function of the genus of G. This means that if G has bounded genus, then G also has bounded thickness. Therefore, Genus-k is a subclass of Thickness-k. On the other hand, to the best of our knowledge, it is not known whether graphs with bounded thickness have bounded genus.

Our contribution. In the summary table, the Steiner Tree column had 6 unresolved entries: 5 P? entries, all of which are now known to be subclasses of Partial k-Trees and henceforth are in P, and one O? entry for Undirected Path graphs. Upon close investigation of the references given in [GJ] and [OG], we found that many consist of "private communication" or could not be confirmed. In the particular case of the Steiner Tree column, we found that this happens for the lines Circular Arc, Circle, Proper Circular Arc, Edge (or Line), and Claw-Free. We were able to find a recent reference for Edge (or Line) and Claw-Free. Additionally, based on the facts that Circular Arc and, consequently, Proper Circular Arc graphs have bounded mim-width and that Steiner tree is polynomial-time solvable for graphs of bounded mim-width, we were able to resolve such entries as well. However, we could not find any reference for Circle graphs, and therefore we underline the corresponding [OG] reference in Table 1. Moreover, the entry Undirected Path is said to be NP-complete in [107], but again with a "private communication" reference (we comment more on this in Section 4). Believing in the need to have explicit proofs for these important problems, we here give a proof of NP-completeness for Undirected Path graphs, which would provide a full dichotomy Polynomial versus NP-complete for the Steiner Tree column. Actually, we provide a second dichotomy for the Steiner Tree problem restricted to Undirected Path graphs, according to the diameter of the input graph. For the Graph Isomorphism column we also provide a full dichotomy Polynomial versus NP-complete by giving an explicit proof of GI-completeness for Thickness-k graphs (please refer to Section 3).

Besides providing a full dichotomy Polynomial versus NP-complete for the Steiner Tree column, in Table 1 we have thoroughly revised the summary table that 35 years later has 54 new resolved entries depicted in bold. Additionally, there are 36 citations for references not in bold that confirm resolved entries from [OG] or [GJ], that we updated because they cited private communications, or because the cited reference is not easily accessible, or could not be confirmed. There is one entry highlighted in italic that corrects the entry for Hamiltonian Circuit restricted to Circle graphs originally P but that actually is N [39].

In addition, we consider the parameterized complexity of hard problems to revise Table 1 into a new Table 2, a proposed summary table of what it means today to study a problem from a computational complexity point of view. This is of course just a sample of what it means, since we could even consider other classifications (e.g., the approximability complexity theory and the space complexity theory). We have kept the same 30 classes but have drawn the horizontal lines so that the Partial k-Trees subclasses appear together, and we may focus on the remaining rows, where the NP-complete entries appear. In Section 2, we discuss in detail Table 2, also presenting the basic definitions of parameterized complexity, in order to draw the reader's attention to the granularity provided by the parameterized complexity for the NP-complete problems into XP, FPT, W_1, W_2, and pN. We depict in Table 2 as O* the only N entry of Table 1 that constitutes the parameterized puzzle for which so far we were not able to provide a parameterized complexity classification. This is to show how rich the original problems posed by Garey and Johnson are, and how their initial classification continues to develop into ever evolving complexity classes, with the NP-complete class being now just the beginning of a very interesting story.

2. The parameterized complexity of hard problems

In this section, we discuss in detail new Table 2. We also further discuss some of the differences that arise between our updated Table 1 and the table presented in [OG] thirty-five years ago. We start by giving some basic definitions of parameterized complexity and its complexity hierarchy classes. After that, we discuss each of the 11 columns separately.

Parameterized complexity. We refer the reader to [37,48,49,52,96] for all basic formal definitions, as well as many techniques employed in the parameterized complexity theory. Formally, given a fixed finite alphabet Σ, a language $L \subseteq \Sigma^* \times \mathbb{N}$ is called a *parameterized problem*; given an instance $(x, \kappa) \in L$, we call κ the *parameter*. Also, we denote the *size* of an instance (x, κ) by $|(x, \kappa)|$. Observe that each possible parameter defines a different parameterized problem; for example, when considering the Clique problem on graphs, it can be parameterized by the size of the desired clique, or by the maximum degree of the input graph. In both cases, the input consists of a graph G, an integer c, and the corresponding parameter, and the problem consists of deciding whether G has a clique of size at least c, except that in the former the parameter is also c, while in the latter the parameter is the maximum degree $\Delta(G)$. When the parameterized problem is a decision problem having as parameter the size of the solution, we say that the problem is parameterized by the *natural parameter*. Table 2 is filled taking into account the natural parameter, whenever possible. We give more details about this when analyzing each of the 11 columns.

We say that a parameterized problem L is *fixed parameter tractable* (from now on denoted by FPT) if there exists an algorithm $\mathcal{A}$ that solves L on input (x, κ) in time $f(\kappa) \cdot |(x, \kappa)|^{\mathcal{O}(1)}$, where f is a computable function. In this case, the algorithm $\mathcal{A}$ is said to be an FPT algorithm for L, and we also use FPT to denote the set of FPT problems. Observe that $\mathsf{P} \subseteq \mathsf{FPT}$.

Intuitively, one could describe the FPT class as the parallel, in the parameterized complexity theory, of the P class in the traditional complexity theory. Another "approachable" class is that of the slice-wise polynomial. We say that a parameterized problem L is *slice-wise polynomial* (denoted by XP) if there exists an algorithm that solves L in time $f(\kappa) \cdot |(x, \kappa)|^{g(\kappa)}$, where f and g are two computable functions. Observe that, for each fixed value of κ, this is a polynomial algorithm.

Concerning a parallel of the NP-complete class, there are two main hard classes in the parameterized complexity, the paraNP-complete and the W-hard. A parameterized problem L is *paraNP-complete* if it is NP-complete for some fixed value of the parameter κ. For instance, the Vertex Coloring problem is paraNP-complete when parameterized by the number of colors. Note that, unless $\mathsf{P} = \mathsf{NP}$, a paraNP-complete problem cannot be XP, hence it cannot be FPT either.

Now, before defining our last parameterized complexity class, we need another definition. Given an instance (x, κ) of a parameterized problem L, a *parameterized reduction* from L to another parameterized problem L' is an algorithm that computes, in time $f(\kappa) \cdot |x|^{\mathcal{O}(1)}$ for some computable function f, an equivalent instance (x', κ') of L' such that $\kappa' \leq g(\kappa)$ for some computable function g. The class of W-hard problems can be formally defined based on a hierarchy of nested classes called $\mathsf{W}[i]$, for each $i \in \mathbb{N} \setminus \{0\}$. However, for our purposes it suffices to define the W[1]-hard and W[2]-hard classes in terms of their "base" problems; think of it as defining the NP-hard class in terms of SAT. A parameterized problem L is *W[1]-hard* if there is a parameterized reduction from Clique, parameterized by the size of the clique, to L; and it is *W[2]-hard* if there is a parameterized reduction from Dominating Set, parameterized by the size of the dominating set, to L. Observe that a parameterized version of an NP-hard problem can be classified in any of these classes, unless of course $\mathsf{P} = \mathsf{NP}$, in which case all the classes collapse to P.

Tree decompositions are an important tool in the parameterized complexity theory, as well as in the traditional computational complexity, since good algorithms can often be obtained for graphs with bounded treewidth, and also because even graphs with unbounded treewidth can sometimes be approached by applying the bidimensionality technique (see e.g. [37]). Since the publication of [OG], in addition to treewidth, many other width parameters have been introduced (see [113] for a nice Hasse diagram containing 32 graph parameters; we also refer to the survey [58]). In particular, the clique-width [69] and the mim-width [113] parameters are of special interest to us, since they are bounded for some of our proposed classes, and because some of the proposed problems can be solved in polynomial time when these parameters are bounded. More specifically, Partial kP-trees and Cographs have bounded clique-width [36], while the following have bounded mim-width: graphs with bounded clique-width [113], Permutation [10], Circular Arc [10], Directed Path [25] (this is because they are a subclass of leaf powers [65]). Regarding the proposed problems, the following can be solved in polynomial time on graphs with bounded mim-width, provided a branch decomposition of bounded mim-width is given: Independent Set, Dominating Set [27], and Steiner Tree [11]; while the following can always be solved in polynomial time on graphs with bounded clique-width, since a construction sequence of bounded clique-width can be found in polynomial (and even FPT) time [62,97,98]: Clique [35], Chromatic Number [75], Hamiltonian Circuit, Maximum Cut [116], and Clique Partition [105]. Although the problem of finding in polynomial-time a branching decomposition of bounded mim-width is still open for general graphs with bounded mim-width, it has been proven to be polynomial-time solvable for some graph classes, including Permutation and Circular Arc graphs [10]. These observations help to solve an issue about the complexity of Steiner Tree restricted to Circular Arc and Proper Circular Arc, which we discuss later on. Also, because Maximum Cut is NP-complete on Interval graphs [1], and this is a subclass of Circular Arc, it follows that the problem is NP-complete on graphs with bounded mim-width, which is in contrast with the complexity of the problem restricted to bounded clique-width [116].

We now discuss Tables 1 and 2, dividing the discussion by the problems.

Membership. The entries P are inherited from Table 1; hence the only class that can be further refined in the parameterized complexity is the class Thickness-k. It is known that deciding whether a given graph G has thickness at most k is NP-complete even if $k = 2$ (for $k = 1$ it coincides with deciding planarity, which is polynomial) [OG]. We therefore get that, considering k as the parameter, the related parameterized problem is paraNP-complete.

We mention that the reference cited by [OG] for Partial k-trees was a technical report that now has a published version [4].

Independent set. The natural parameter considered is the size of the desired independent set. The problem is trivially in XP in general: by enumerating all vertex subsets of size κ, one can readily check in time $\mathcal{O}(n^{\kappa})$ whether a graph on n vertices admits an independent set of size at least κ. Nevertheless, Independent Set is unlikely to be FPT, since it is known to be W[1]-hard [48]. In fact, by considering the complement graph, we obtain that Independent Set and Clique, for general graphs, are equivalent to each other from the parameterized complexity perspective. On the positive side, as can be seen in Table 2, the problem is FPT for Planar [96], Genus-k [30] and Degree-k [48] graphs. More generally, the problem is also FPT for Thickness-k graphs [74]: This follows from the fact that Independent Set is FPT when restricted to graphs with bounded clique number, as first observed in [74] as a special case of a more general framework *cf.* [52,104]. Additionally, Independent Set is also FPT for $K_{3,3}$-Free*. Indeed, it is well known that, if a graph H of maximum degree at most 3 is a minor of a graph G, then H is a topological minor of G as well *cf.* [46,77]. Thus, $K_{3,3}$-Free* coincides with the class $K_{3,3}$-Minor-Free* of graphs that do not contain $K_{3,3}$ as a minor. Therefore, since Independent Set was proven to be FPT for $K_{3,3}$-Minor-Free* graphs [42,43], we obtain that the problem is also FPT for $K_{3,3}$-Free*. We remark that, although Independent Set is FPT for $K_{3,3}$-Free* graphs, it remains W[1]-hard for the larger class $K_{3,3}$-Free. This latter result follows from the fact that Independent Set was proven to be W[1]-hard for another subclass of $K_{3,3}$-Free, the C_4-free graphs [21], which consists of graphs that do not contain C_4 as an induced subgraph.

We observe that, for the entry Partial k -Trees, we cite Ref. [5], which is different from Ref. [2] cited in [OG]. This is because, upon checking [2], we were not able to find any mention to the Independent Set problem restricted to Partial k -Trees.

An interesting fact is that D.S. Johnson mentions, in the caption of his summary table, that Vertex Cover was not included as a column because its complexity will always be the same as the complexity of Independent Set. While this is true for traditional complexity theory, one could say that Vertex Cover is a canonical problem in FPT since it is FPT in general [31] and many of the known techniques can be successfully applied to it, whereas Independent Set can be regarded as a canonical W[1]-hard problem.

Clique. Analogously, the natural parameter considered is the size of the desired clique. All entries here are in P, except for Claw-Free graphs, which is FPT [38].

Partition into cliques. The problem has as input a graph G and a positive integer κ, and it consists of deciding whether the vertex set of G can be partitioned into κ disjoint cliques. The natural parameter considered is the integer κ. One can straightforwardly verify that Partition into Cliques is polynomially equivalent to Chromatic Number by considering the complement graph $\overline{G}$ of the input graph G, i.e. the problem of deciding whether $\overline{G}$ can be proper colored with at most κ colors. Nevertheless, it is worth mentioning that the respective particular cases of Partition into Cliques and Chromatic Number restricted to specific graph classes are not necessarily polynomially equivalent to each other, as usually these graph classes are not closed under taking the complement. As an example, observe that in Table 1, Partition into Cliques is in P for Circular Arc graphs, while Chromatic Number remains NP-complete for Circular Arc graphs.

The problem is trivially FPT for graphs that only contain cliques whose size can be upper-bounded by a computable function f of κ. Indeed, if the size of the maximum clique of G is at most $f(\kappa)$, then either G contains at most $f(\kappa)\cdot\kappa$ vertices, or the vertex set of the input graph cannot be partitioned into at most κ disjoint cliques, and thus we are dealing with a no-instance. Based on that, we immediately obtain that Partition into Cliques is FPT for Planar and $K_{3,3}$-Free* graphs (observe that the latter graphs cannot have cliques of size bigger than 5). Additionally, one can verify that Thickness-k, Genus-k and Degree-k graphs also have cliques whose size depends only on k, and since k is constant by definition, we get that these graphs have maximum clique bounded by a constant value. As a result, we obtain that Partition into Cliques is also FPT for Thickness-k, Genus-k and Degree-k graphs.

On the other hand, Circle graphs may have cliques of unbounded size. J. Keil and L. Stewart proved that Partition into Cliques is XP for Circle graphs [73], however it remains unknown whether the problem is FPT in this class. The paraNP-completeness of Partition into Cliques for Claw-Free graphs follows from the fact that Chromatic Number is NP-complete, even when $\kappa = 3$, for Triangle-Free graphs (*i.e.*, graphs that do not contain K_3 as an induced subgraph) [85] — since the complement graphs of Triangle-Free graphs do not have independent sets of size larger than 2, such complement graphs are Claw-free.

In what follows, we discuss some discrepancies between our Table 1 and the table presented in [OG], with respect to the Partition into Cliques entries. First, [OG] cites [GJ] for the Degree-k entry. However it is not immediate how the results presented in [GJ] (or in the references cited by [GJ]) lead to the NP-completeness of Partition into Cliques for Degree-k. In fact, [GJ] proves that Partition into Cliques remains NP-complete for graphs that contain no cliques of size larger than 4; nevertheless the graph constructed in their reduction does not have bounded maximum degree. For this reason, we cite [29] instead, which gives an explicit NP-completeness proof for Partition into Cliques restricted to Cubic graphs. As for Planar graphs, [OG] cites [12], which actually proves that the following problem is NP-hard: given a planar graph G, and a fixed connected outerplanar graph H with at least three vertices, maximizes the number of vertices of G that can be covered with copies of H. This does not immediately imply that Partition into Cliques is NP-complete for Planar graphs since they limit the number of vertices in each clique to 3 (by considering $H = K_3$) when a planar graph could have a partition into cliques with cliques also of size 4. We then cite the more explicit construction given in [78]. Note that this also impacts the entries $K_{3,3}$-Free*, Thickness-k, and Genus-k, as these contain Planar graphs. Finally, we remark that the references cited by [OG] for Line graphs and Claw-Free graphs are actually private communications. Therefore, we cite [95] for Line graphs, and [85] for Claw-Free graphs, instead of [OG]. We remark that although the NP-completeness of Partition into Cliques for such graph classes directly follows from [95], we have decided to cite the oldest known reference for each result.

A problem closely related to Partition into Cliques is the so-called Clique Edge-Partition, which is defined as follows: given a graph G and a positive integer k, decide whether the edge set of G can be partitioned into at most k subsets such that each subset induces a complete subgraph of G. However, it is worth mentioning that, since the line graph of a complete graph is not necessarily a complete graph, this problem is not the same as deciding whether the vertex set of the line graph $L(G)$ of a graph G can be partitioned into at most k disjoint cliques. As a matter of fact, while Partition into Cliques is paraNP-complete [85], Clique Edge-Partition is FPT in general [92].

Chromatic number. The problem has as input a graph G and a positive integer κ, with κ being the considered parameter, and it consists of deciding whether the chromatic number of G is at most κ. Also one of Karp's 21 NP-complete problems [70], Chromatic Number is NP-complete for fixed values of κ for very restricted graph classes. Since it is NP-complete to decide whether a planar graph with maximum degree 4 is 3-colorable [55], we get that Chromatic Number is paraNP-complete for Planar, $K_{3,3}$-Free*, Genus-k, Thickness-k and Degree-k graphs. Additionally, since it is NP-complete to decide whether a circle graph is 4-colorable [112], we obtain that Chromatic Number is paraNP-complete for Circle graphs. Finally, it is also NP-complete to decide whether the line graph of a 3-regular graph is 3-colorable [63], implying the paraNP-completeness for Line and Claw-Free graphs. On the positive side, we mention that, even though the parameterized complexity theory had not yet been defined by the time of publication of [54], the algorithm presented in [54] for Circular Arc is actually an FPT algorithm. Therefore, Chromatic Number parameterized by κ, the number of colors, is FPT for Circular Arc graphs.

As for the differences between our tables and [OG]'s Table, they cite [2] for the entry Partial k-Trees. However this reference does not mention the Chromatic Number problem, and this is why we cite [5]. As for Bandwidth-k graphs, they cite [90], which treats only the case $k = 3$. The same happens for the entry k-Outerplanar [8]. Also, we could not find the reference cited by [OG] for Series–Parallel graphs [110], which is the same as the one cited for Halin graphs. Nevertheless, the polynomial results for all these classes follow from the fact they all have bounded treewidth; therefore we cite [5].

Chromatic index. This is by far the hardest problem of the table, with the largest number of open entries, remaining open even for classes considered "easy" as for instance Cographs, a graph class for which all problems besides Chromatic Index have been classified as P [75]. The problem has as input a graph G, and a positive integer κ, with κ being the considered parameter, and one wants to decide whether the chromatic index of G, denoted by $\chi'(G)$, is at most κ. Observe that, since Vizing's Theorem tells us that $\chi'(G) \in \{\Delta(G), \Delta(G) + 1\}$, we can then consider κ as being equal to the maximum degree of G as otherwise the answer is trivial. As already mentioned, deciding whether $\chi'(G) = 3$ is NP-complete even for Cubic graphs [63], which implies that Chromatic Index is paraNP-complete for Degree-k graphs, and that it is also paraNP-complete for Thickness-k graphs. In addition, L. Cai and J. A. Ellis proved that deciding whether $\chi'(G) = 3$ is NP-complete even for Comparability graphs and for Line graphs [28]. Therefore, Chromatic Index remains paraNP-complete when restricted to Comparability, Perfect, Line and Claw-Free graphs.

The reference cited by [OG] for Series–Parallel graphs and Halin graphs is the same as the one cited for these graphs on column Chromatic Number [110]. Again, even though we could not find the reference, the results follow from the fact that these graphs have bounded treewidth [17].

HAMILTONIAN CIRCUIT. Here, the size of any solution is the size of the input graph; therefore, we consider the problem of deciding whether a given graph G has an Hamiltonian cycle parameterized by $n = |V(G)|$. It is known that LONGEST CYCLE parameterized by the size κ of the cycle is FPT [89]. If follows that HAMILTONIAN CIRCUIT is FPT when parameterized by n.

The reference cited by [OG] for SERIES–PARALLEL graphs [110] is the same as the one cited for these graphs on CHROMATIC NUMBER (see comments above); thus, we cite [5] instead. Also, we were not able to access the reference for SPLIT and CHORDAL graphs [33], therefore we cite Ref. [93]. For CIRCLE graphs, we find a more serious discrepancy between our Table 1 and the table presented in [OG]. They cite [13] as providing a polynomial algorithm for CIRCLE graphs. However, the paper only provides a polynomial algorithm for INTERVAL graphs, a subclass of CIRCLE graphs. And actually, the problem has been shown to be NP-complete for CIRCLE graphs in [39].

DOMINATING SET. Given a graph G and a positive integer κ, the problem consists of deciding whether G has a dominating set of size at most κ (a set $D \subseteq V(G)$ is *dominating* if, for every vertex $v \in V(G)$, $v \in D$ or v has a neighbor that belongs to D). The natural parameter is of course κ, and as previously mentioned, this is the canonical problem in the class W[2]-hard. Because of this, DOMINATING SET is among the most investigated problems in the parameterized complexity theory, deserving a survey of its own. Here, we make a short compilation of the results that concern the classes of interest. The problem is FPT for: PLANAR graphs [47], and therefore also for GRID graphs; for $K_{3,3}$-FREE* graphs [100]; for GENUS-k graphs [51]; for k-DEGENERATE graphs [3], and therefore also for DEGREE-k graphs; and for CLAW-FREE graphs [38], and therefore also for LINE graphs. In addition, it is W[2]-hard for SPLIT and BIPARTITE graphs [104], and therefore also for CHORDAL, PERFECT, and COMPARABILITY graphs. Finally, it is W[1]-hard for CIRCLE graphs [24]. Observe that this problem is trivially in XP, since it suffices to test all the $\mathcal{O}(n^\kappa)$ subsets of size κ. However, this is not completely refined, and the entries for THICKNESS-k and UNDIRECTED PATH graphs can be regarded as the only open cases in this column.

Regarding the differences between our Table 1 and the table in [OG], they cite [57] for ALMOST TREES (k), but the paper does not seem to attack this class. Something similar happens with entry BANDWIDTH-k, where they cite [90] but the paper attack domination-related problems, but not DOMINATING SET itself. Nevertheless, we now know that the problem is indeed polynomial for these classes since they have bounded treewidth [5]. Also, the entry for GRIDS is cited as a private communication in [OG]; this is why we provide Ref. [32]. As for the BIPARTITE and COMPARABILITY entries, we were not able to find Ref. [45], cited by [OG], this is why we also provide [94].

MAXIMUM CUT. This is another problem that is FPT in general. Given a graph G, and an integer κ, we consider the problem of deciding whether there exists $S \subseteq E(G)$ that separates G and has size at least κ, parameterized by κ. The best known algorithm so far runs in time $\mathcal{O}(m + n + \kappa \cdot 4^\kappa)$ [86], where m denotes the number of edges and n denotes the number of vertices of the input graph G.

We were not able to find the reference cited by [OG] for THICKNESS-k graphs [9], and therefore we provide the same reference given by [GJ] for DEGREE-k graphs [119].

STEINER TREE. Given a graph G, a subset $X \subseteq V(G)$, called *terminal* set, and a positive integer t, the problem consists of deciding whether there exists a subset $S \subseteq V(G) \setminus X$ such that $|S \cup X| \leq t$ and $G[S \cup X]$ is connected — and hence $G[S \cup X]$ contains a tree subgraph $\mathcal{T}$ with $X \subseteq V(\mathcal{T})$, called a *Steiner tree* of G for X. The vertices in S are commonly called *Steiner vertices*. STEINER TREE parameterized by the number of terminal vertices $|X|$ is well-known to be FPT [50], with the current best algorithm running in time $\mathcal{O}(2^{|X|} \cdot n^{\mathcal{O}(1)})$ [15], where n denotes the number of vertices of the input graph G. This clearly implies that STEINER TREE parameterized by the natural parameter, *i.e.* by the size of the sought solution $|S \cup X| \leq t$, is also FPT. Interestingly enough, the problem is W[2]-hard when parameterized by the number of Steiner vertices $|S|$ as shown in [88]. We denote by κ the maximum number of Steiner vertices allowed in a given instance of the problem, and then we write κ-STEINER TREE to denote STEINER TREE parameterized by κ. Since the other parameterized versions of the problem are already known to be FPT for general graphs, this latter is the version considered in Table 2.

The κ-STEINER TREE problem is FPT for GENUS-k graphs [101], and therefore for PLANAR and GRID graphs; and for k-DEGENERATE graphs [67], and therefore for DEGREE-k graphs. Also, the proof given in [104] for W[2]-hardness of DOMINATING SET for SPLIT graphs actually holds for CONNECTED DOMINATING SET. Moreover, in [117], the authors give a parameterized reduction from CONNECTED DOMINATING SET to κ-STEINER TREE that works for any subclass of CHORDAL graphs, without changing the input graph. Therefore, based on the results presented in [104], we obtain that κ-STEINER TREE is W[2]-hard for SPLIT, CHORDAL and PERFECT graphs. In the next section, we give in Proposition 1 a simple reduction to prove that the problem is W[2]-hardness for BIPARTITE graphs (and therefore also for COMPARABILITY graphs). Finally, observe that a simple XP algorithm can be obtained by simply testing all $\mathcal{O}(n^\kappa)$ possible vertex subsets $S \subseteq V(G) \setminus X$ of size at most κ.

Regarding the differences between our tables and [OG]'s Table, Ref. [114] cited in [OG] for OUTERPLANAR graphs could not be found, but we mention that the reference cited in [OG] for SERIES–PARALLEL [115] is indeed correct, and that it can be used for OUTERPLANAR as well. Also, [OG] cites a private communication with Schäffer for the entries CIRCLE, LINE, and CLAW-FREE graphs, and cites [117] for CIRCULAR ARC graphs and PROPER CIRCULAR ARC. We were not able to find any mention to CIRCULAR ARC graphs in [117]. Also, in his book [107], Spinrad writes:

> “The status of STEINER TREE is slightly unclear; Schäffer sketched a proof that this is polynomially solvable (for both CIRCLE and CIRCULAR ARC graphs), and thus it appeared as polynomial in the table of ‘[OG]’, though no algorithm solving the problem appears in the general literature”.

Nevertheless, because Circular Arc graphs have bounded mim-width and a branch decomposition with bounded mim-width of these graphs can be computed in polynomial time [10], it follows from [11] that Steiner Tree can be solved in polynomial time for Circular Arc, and consequently also for Proper Circular Arc. As for entries Line and Claw-Free, they have been recently filled [19], while the situation remains the same for Circle graphs. We add that in [71], J. Keil proves that Connected Dominating Set is NP-complete for Circle graphs. Thus if a proof of polynomiality of Steiner tree indeed exists for Circle graphs, this will be a nice example of class that separates Connected Dominating Set from Steiner tree.

Graph isomorphism. This is problem Open1 from [GJ], and perhaps the most controversial problem in Graph Theory, being regarded as the only naturally defined problem with a high chance to be an NP-intermediate problem, thus having deserved a classification of its own. Given two graphs G and H, it consists of deciding whether G and H are isomorphic, i.e., whether there exists a bijection of the vertex sets that preserves adjacencies. A problem is GI-complete if it is equivalent in complexity to general Graph Isomorphism. As it happened with Hamiltonian Circuit, the natural parameter here is the size of the input graph. This problem is FPT in general, with the best known algorithm running in time $O^*(2^{\sqrt{n}\log n})$ [6].

There are again some discrepancies between our Table 1 and the table presented in [OG]. In [OG], they cite [GJ] as a reference for the entries Perfect and Chordal graphs to be GI-complete; however, Chordal graphs are cited as an open case in [GJ]. Nevertheless, these classes are indeed GI-complete as proven in [84] for Chordal graphs; this construction was noticed to work also for Split graphs [108]. Something similar happens in entries Bipartite and Undirected Path graphs, with them being cited as open cases in [GJ], instead of GI-complete, as cited in [OG]. Nevertheless, these are indeed GI-complete as proven in [22]. This also impacts the Comparability graphs entry. Finally, Graph Isomorphism is also GI-complete for Thickness-k *cf.* [41]. Since [41] cites an unpublished paper due to De Biasi, we provide a proof in Proposition 3, for the sake of completeness.

3. Some simple reductions

For the sake of completeness, here we present two simple proofs. First, we prove that κ-Steiner Tree is W[2]-hard when restricted to Bipartite graphs. Indeed, this result follows from a standard parameterized reduction from Dominating Set, described in Proposition 1. We remark that Raman and Saurabh present a similar reduction to prove that Dominating Set is W[2]-hard for Bipartite graphs [104].

Proposition 1. *κ-Steiner Tree remains W[2]-hard for Bipartite graphs.*

Proof. Let $I = (G, \kappa)$ be an instance of Dominating Set. We let $I' = (G', X, \kappa)$ be the instance of κ-Steiner Tree such that G' is defined as follows:

- $V(G') = \{r\} \cup \{v' : v \in V(G)\} \cup V(G)$, where r denotes a new vertex, and we add a new vertex v' for each $v \in V(G)$;
- $E(G') = \{rv : v \in V(G)\} \cup \{v'u : u \in N_G[v],\ u, v \in V(G)\}$; and

the terminal set is defined as $X = \{r\} \cup \{v' : v \in V(G)\}$. Note that X and $V(G)$ are independent sets of G'. Thus, G' is a bipartite graph.

Suppose that G admits a dominating set $D \subseteq V(G)$ of size at most κ. It is not hard to check that $D \cup X$ induces a connected subgraph of G'. Therefore, I' is a yes-instance of κ-Steiner Tree.

Conversely, suppose that G' admits a Steiner tree T for X such that $|V(T) \setminus X| \leq \kappa$. Note that neighbors of X in G' belong to $V(G)$. Thus, $V(T) \cap V(G)$ is a dominating set of G, otherwise either T would not be connected, or there would exist some terminal vertex belonging to $X \setminus \{r\}$ that is not in T. Moreover, since $|V(T) \setminus X| \leq \kappa$, we obtain that $|V(T) \cap V(G)| \leq \kappa$. Therefore, $V(T) \cap V(G)$ is a dominating set of G of size at most κ, and I is a yes-instance of Dominating Set. □

Now, we present a proof that Graph Isomorphism is GI-complete when restricted to Thickness-k graphs. This result actually follows from a simple adaptation of an argument described in [14] by De Biasi, which we present in Proposition 3. The *subdivision* of a graph G is defined as the graph $\mathfrak{s}(G)$ obtained from G by replacing each edge $e = uv \in E(G)$ with the path $\langle u, w_e, v\rangle$, where w_e denotes a new vertex. More formally, $\mathfrak{s}(G)$ is the graph with vertex set $V(\mathfrak{s}(G)) = V(G) \cup \{w_e : e \in E(G)\}$ and edge set $E(\mathfrak{s}(G)) = \{uw_e, w_ev : e = uv \in E(G)\}$.

Lemma 2. *For each graph G, the subdivision $\mathfrak{s}(G)$ of G has thickness at most 2.*

Proof. Assume without loss of generality that $V(G) = \{v_1, \ldots, v_n\}$, for some positive integer n. Let H_1 and H_2 be the spanning subgraphs of $\mathfrak{s}(G)$ defined as follows: for each edge $e = v_iv_j \in E(G)$ with $i < j$, add the edge v_iw_e to H_1 and add the edge w_ev_j to H_2. Note that, for each $e \in E(G)$, the degree of w_e in H_1, and in H_2, is exactly 1. Moreover, $V(G)$ is an independent set in H_1, and in H_2. Thus, H_1 and H_2 are forests whose components are stars, which implies that H_1 and H_2 are planar graphs. Therefore, since $\mathfrak{s}(G) = H_1 \cup H_2$, we obtain that $\mathfrak{s}(G)$ has thickness at most 2. □

Proposition 3. *Graph Isomorphism is GI-complete for Thickness-k.*

Proof. Let G_1 and G_2 be two arbitrary graphs. It follows from Lemma 2 that $\mathfrak{s}(G_1)$ and $\mathfrak{s}(G_2)$ have thickness at most 2. Moreover, one can easily verify that G_1 and G_2 are isomorphic if and only if $\mathfrak{s}(G_1)$ and $\mathfrak{s}(G_2)$ are isomorphic. □

4. Steiner tree for undirected path graphs

In this section, we prove that the Steiner Tree problem is NP-complete for Undirected Path graphs, which provides a full dichotomy Polynomial versus NP-complete for the Steiner Tree column. Our proof holds even if the input graph has diameter 3, and we show that Steiner Tree is in P when restricted to Undirected Path graphs of diameter 2, thus getting another dichotomy for the problem in terms of the diameter.

We mention that Spinrad writes in his book [107] that he was unable to find any work on the Steiner Tree problem restricted to Undirected Path graphs, but that Dieter Kratsch told him this should be NP-complete as a simple extension of a proof that Connected Dominating Set is NP-complete for Undirected Path graphs. Haynes et al. cite a paper [72], submitted in 1997, in their book [60], where the NP-completeness proof of Connected Dominating Set supposedly appears. However, we were not able to find any version of [72]. Thus, in order to fill this gap, we provide a non-trivial adaptation of the NP-completeness proof presented in [23] for the Dominating Set problem restricted to Undirected Path graphs, which finally explicitly shows that the Connected Dominating Set problem restricted to Undirected Path graphs is indeed NP-complete. Then, we use a transformation by White et al. [117] between the Steiner Tree and the Connected Dominating Set problems to obtain the desired result as a corollary.

A closely related variant of Connected Dominating Set that should be mentioned is the so-called Total Dominating Set problem, which, rather than a dominating set inducing a connected subgraph, simply requires a dominating set having no isolated vertices. Through a different non-trivial adaptation of the proof presented in [23], Total Dominating Set restricted to Undirected Path graphs was proven to be NP-complete [81]. However, it is worth noticing that the construction described in [81] cannot be used so as to further obtain the NP-completeness of Connected Dominating Set for Undirected Path graphs. Therefore, we emphasize the merit of our contribution.

We start by giving some formal definitions. A *chordal* graph can also be described as the intersection graph of subtrees of a tree: given a tree T, each vertex u of G corresponds to a subtree T_u of T, and $uv \in E(G)$ if and only if $V(T_u) \cap V(T_v) \neq \emptyset$. We call $(T, \{T_u\}_{u \in V(G)})$ a *tree model of* G. One can verify that a tree decomposition of G of width $\omega(G)$ can be obtained from this tree model. The subclasses of Undirected Path, Directed Path and Interval graphs can be derived from this definition as follows. An *undirected path graph* is a chordal graph that has a tree model $(T, \{T_u\}_{u \in V(G)})$ where each $u \in V(G)$ corresponds to a subpath of T. A *directed path graph* is an undirected path graph that has a tree model $(T, \{T_u\}_{u \in V(G)})$ such that T is rooted at a vertex r, and every subpath T_u is from a node $t \in V(T)$ to a node $t' \in V(T)$ where t belongs to the (r, t')-path of T. Finally, an *interval graph* is a chordal graph that has a tree model $(T, \{T_u\}_{u \in V(G)})$ where T is a path. From Fig. 2, we know that these classes are nested.

We recall that, given a graph G, a subset $D \subseteq V(G)$ is a *dominating set* of G if, for every vertex $v \in V(G) \backslash D$, $N_G(v) \cap D \neq \emptyset$. Additionally, D is said to be *connected* if $G[D]$ is a connected subgraph of G. Given also a subset $X \subseteq G$ of *terminals*, a *Steiner tree* of G for X is a tree subgraph $\mathcal{T}$ of G such that $X \subseteq V(\mathcal{T})$. Next, we formally state the Steiner Tree and Connected Dominating Set problems. Although the usual question for Steiner Tree asks for the minimum tree, it is more convenient for our reduction to ask for the minimum set of non terminal vertices. Notice that this gives a polynomially equivalent problem.

Steiner Tree
Input: A connected graph G, a non-empty subset $X \subseteq V(G)$, and a positive integer k.
Question: Does there exist a subset $S \subseteq V(G) \setminus X$ with $|S| \leq k$, such that $G[S \cup X]$ is connected?

Connected Dominating Set
Input: A graph G and a positive integer k.
Question: Does there exist a subset $D \subseteq V(G)$ with $|D| \leq k$, such that $N_G[D] = V(G)$ and $G[D]$ is connected?

As we said before, we first prove that Connected Dominating Set is NP-complete for Undirected Path graphs. We do this with a reduction from the following problem, which is one of Karp's 21 NP-complete problems [70].

3D-Matching
Input: Disjoint sets P, Q and R each of cardinality n, for some positive integer n, and a subset $\mathcal{S} \subseteq P \times Q \times R$.
Question: Does there exist a subset $D \subseteq \mathcal{S}$ such that $|D| = n$ and $\mathbf{s} \cap \mathbf{s}' = \emptyset$ for every two triples $\mathbf{s}, \mathbf{s}' \in D$?

Theorem 4. Connected Dominating Set *remains* NP*-complete when restricted to undirected path graphs of diameter at most 3.*

Proof. Let $P = \{p_1, \ldots, p_n\}$, $Q = \{q_1, \ldots, q_n\}$ and $R = \{r_1, \ldots, r_n\}$ be disjoint sets each of cardinality n, for some positive integer n, and let $\mathcal{S} = \{\mathbf{s}_1, \ldots, \mathbf{s}_m\}$ be a subset of $P \times Q \times R$ of cardinality m, for some positive integer m. We let $I = (P, Q, R, \mathcal{S})$ be the instance of 3D-Matching constituted by P, Q, R and $\mathcal{S}$. Then, we let G be the graph obtained from I as follows (Fig. 3 shows a tree model of the constructed graph):

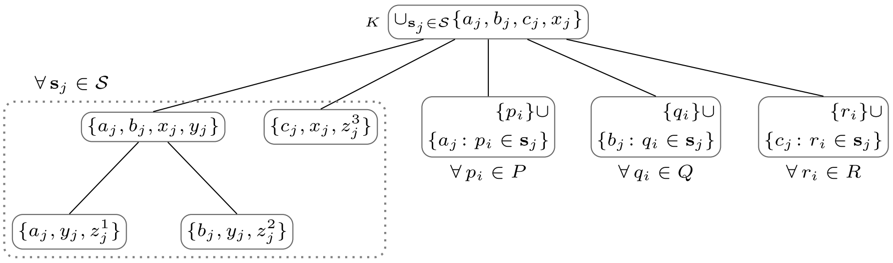

K
$\cup_{\mathbf{s}_j \in \mathcal{S}}\{a_j, b_j, c_j, x_j\}$
$\forall \mathbf{s}_j \in \mathcal{S}$
$\{a_j, b_j, x_j, y_j\}$
$\{c_j, x_j, z_j^3\}$
$\{a_j, y_j, z_j^1\}$
$\{b_j, y_j, z_j^2\}$
$\{p_i\}\cup$ $\{a_j : p_i \in \mathbf{s}_j\}$
$\forall p_i \in P$
$\{q_i\}\cup$ $\{b_j : q_i \in \mathbf{s}_j\}$
$\forall q_i \in Q$
$\{r_i\}\cup$ $\{c_j : r_i \in \mathbf{s}_j\}$
$\forall r_i \in R$

Fig. 3. A tree model $(T, \{T_u\}_{u \in V(G)})$ associated with the graph G, constructed from a given instance $I = (P, Q, R, \mathcal{S})$ of the 3D-Matching problem.

- For each $\mathbf{s}_j \in \mathcal{S}$, we let $V_j = \{a_j, b_j, c_j, x_j, y_j, z_j^1, z_j^2, z_j^3\}$. We remark that, for each two sets $\mathbf{s}_j, \mathbf{s}_\ell \in \mathcal{S}$, $V_j \cap V_\ell = \emptyset$ if and only if $j \neq \ell$;
- $V(G) = \cup_{j=1}^m V_j \cup P \cup Q \cup R$;
- $K = \bigcup_{\mathbf{s}_j \in \mathcal{S}} \{a_j, b_j, c_j, x_j\}$ is a clique of G;
- for each $\mathbf{s}_j \in \mathcal{S}$, $\{a_j, b_j, x_j, y_j\}$, $\{a_j, y_j, z_j^1\}$, $\{b_j, y_j, z_j^2\}$ and $\{c_j, x_j, z_j^3\}$ are cliques of G;
- for each $p_i \in P$, $\{p_i\} \cup \{a_j : p_i \in \mathbf{s}_j, \mathbf{s}_j \in \mathcal{S}\}$ is a clique of G;
- for each $q_i \in Q$, $\{q_i\} \cup \{b_j : q_i \in \mathbf{s}_j, \mathbf{s}_j \in \mathcal{S}\}$ is a clique of G;
- for each $r_i \in R$, $\{r_i\} \cup \{c_j : r_i \in \mathbf{s}_j, \mathbf{s}_j \in \mathcal{S}\}$ is a clique of G.

Fig. 3 illustrates a tree model $(T, \{T_u\}_{u \in V(G)})$ associated with the graph G. We depict inside a node $t \in V(T)$, the set of vertices of G that contains node t in its corresponding subtree; more formally, denoting by X_t the subset of $V(G)$ drawn inside node t, and given $v \in V(G)$, we define T_v as the subtree of T induced by $\{t \in V(T) : v \in X_t\}$. Based on $(T, \{T_u\}_{u \in V(G)})$, one can verify that G is an undirected path graph. Indeed, for each vertex $u \in V(G)$, we get that T_u is a path of T. Moreover, one can readily verify that K is a dominating clique of G. Therefore, G has diameter at most 3.

We now prove that I is a yes-instance of 3D-Matching if and only if G admits a connected dominating set D of size at most $2m + n$.

First, suppose that I is a yes-instance of 3D-Matching, and let M be a 3d-matching of I. Then, we define $D = \{a_j, b_j, c_j : \mathbf{s}_j \in M\} \cup \{x_j, y_j : \mathbf{s}_j \notin M\}$. Note that $|D| = 3n + 2(m - n) = 2m + n$. We claim that D is a connected dominating set of G. Indeed, since M is a 3d-matching of I, we have that the following holds:

- for each $p_i \in P$, there exists (exactly) one triple $\mathbf{s}_j \in M$ such that $p_i \in \mathbf{s}_j$, which implies that $a_j \in D$ and that p_i is dominated in G by a_j;
- for each $q_i \in Q$, there exists (exactly) one triple $\mathbf{s}_j \in M$ such that $q_i \in \mathbf{s}_j$, which implies that $b_j \in D$ and that q_i is dominated in G by b_j;
- for each $r_i \in R$, there exists (exactly) one triple $\mathbf{s}_j \in M$ such that $r_i \in \mathbf{s}_j$, which implies that $c_j \in D$ and that r_i is dominated in G by c_j.

Additionally, it directly follows from the construction of G that $\{a_j, b_j, c_j\}$ dominates all vertices belonging to V_j for each $\mathbf{s}_j \in M$, and that $\{x_\ell, y_\ell\}$ dominates all vertices belonging to V_ℓ for each $\mathbf{s}_\ell \in \mathcal{S} \setminus M$. Consequently, D is a dominating set of G. To verify that D induces a connected subgraph of G, first note that $K' = \bigcup_{\mathbf{s}_j \in M} \{a_j, b_j, c_j\}$ induces a connected subgraph of G since K is a clique of G and $K' \subseteq K$. In addition, it follows from the facts that $\bigcup_{\mathbf{s}_j \in \mathcal{S}} \{x_j\} \subseteq K$, and $x_j y_j \in E(G)$ for each $\mathbf{s}_j \in \mathcal{S}$, that the set $K'' = \bigcup_{\mathbf{s}_j \in \mathcal{S} \setminus M} \{x_j, y_j\}$ induces a connected subgraph of G. Therefore, the result follows since $D = K' \cup K''$ is a connected subgraph of G.

Conversely, suppose now that G admits a connected dominating set D of size at most $2m + n$. By the construction of G, for each $\mathbf{s}_j \in \mathcal{S}$, the following holds (see Fig. 4):

- $D \cap \{a_j, y_j, z_j^1\} \neq \emptyset$, otherwise z_j^1 would not be dominated in G by D;
- $D \cap \{b_j, y_j, z_j^2\} \neq \emptyset$, otherwise z_j^2 would not be dominated in G by D;
- $D \cap \{c_j, x_j, z_j^3\} \neq \emptyset$, otherwise z_j^3 would not be dominated in G by D.

We recall that $V_j = \{a_j, b_j, c_j, x_j, y_j, z_j^1, z_j^2, z_j^3\}$ for each $\mathbf{s}_j \in \mathcal{S}$.

Then, based on the above, one can verify that $|D \cap V_j| \geq 2$. Moreover, we prove that if $|D \cap V_j| = 2$ for some $\mathbf{s}_j \in \mathcal{S}$, then $D \cap V_j = \{x_j, y_j\}$. This follows from the fact that the only other possibilities for $D \cap V_j$ with $|D \cap V_j| = 2$ would be $D \cap V_j = \{c_j, y_j\}$ and $D \cap V_j = \{z_j^3, y_j\}$. However, note that, $\{a_j, b_j, x_j\}$ is a separator of c_j and y_j in G for each $\mathbf{s}_j \in \mathcal{S}$. Thus, if $D \cap V_j = \{c_j, y_j\}$, then D would not induce a connected subgraph of G. Analogously, $\{c_j, x_j\}$ is a separator of z_j^3 and y_j in G for each $\mathbf{s}_j \in \mathcal{S}$. Thus, if $D \cap V_j = \{z_j^3, y_j\}$, then D would not induce a connected subgraph of G. As a result, we have that $D \cap V_j = \{x_j, y_j\}$ whenever $|D \cap V_j| = 2$.

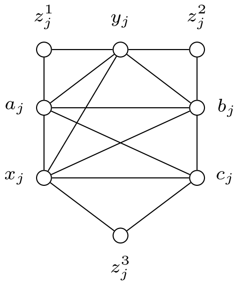


Fig. 4. Subgraph of G induced by V_j, for $\mathbf{s}_j \in \mathcal{S}$.

We prove now that we can assume that $D \cap V_j = \{a_j, b_j, c_j\}$ for each $\mathbf{s}_j \in \mathcal{S}$ with $|D \cap V_j| \geq 3$. Indeed, for each $\mathbf{s}_j \in \mathcal{S}$, $\{a_j, b_j, c_j\}$ dominates at least the same vertices of G as any other subset of V_j, which implies $N_G[D \cap V_j] \subseteq N_G[\{a_j, b_j, c_j\}]$. Moreover, since by hypothesis D induces a connected subgraph of G, we obtain that $(D \setminus V_j) \cup \{a_j, b_j, c_j\}$ also induces a connected subgraph of G for each $\mathbf{s}_j \in \mathcal{S}$. Thus, assume without loss of generality that $D \cap V_j = \{a_j, b_j, c_j\}$ whenever $|D \cap V_j| \geq 3$.

Now, let $M = \{\mathbf{s}_j \in \mathcal{S} \colon |D \cap V_j| = 3\}$. Based on the definition of M, we obtain that $|D| \geq 3|M| + 2(m - |M|) = 2m + |M|$. On the other hand, we have by hypothesis that $|D| \leq 2m + n$. Consequently, $|M| \leq n$. Towards a contradiction, suppose that $|M| < n$. Let $D' = \bigcup_{\mathbf{s}_j \in \mathcal{S}} (D \cap V_j)$. Note that, for each $U \in \{P, Q, R\}$, there are at most $|M|$ vertices from U that are dominated in G by some vertex in D'. As a result, there exist at least $3(n - |M|)$ distinct vertices from $P \cup Q \cup R$ that are not dominated in G by any vertex belonging to D', *i.e.* $|(P \cup Q \cup R) \setminus N_G[D']| \geq 3(n - |M|)$. This implies that D must further contain each of such vertices from $P \cup Q \cup R$ that are not dominated in G by D'. Then, we obtain that actually

$$|D| \geq 3|M| + 2(m - |M|) + 3(n - |M|) = 2m + n + 2(n - |M|) > 2m + n,$$

which contradicts the hypothesis of D being a connected dominating set of G of size at most $2m + n$. Consequently, $|M| = n$ and $|D| = 2m + n$. This implies that, if u is a vertex in D, then $u \in V_j$ for some $\mathbf{s}_j \in \mathcal{S}$. Hence, we obtain that M is a 3d-matching of I, otherwise there would exist a vertex $v \in P \cup Q \cup R$ such that, for every triple $\mathbf{s}_j \in \mathcal{S}$ with $v \in \mathbf{s}_j$, $|D \cap V_j| = 2$, which would imply that v is not dominated in G by any vertex belonging to D. Therefore, I is a `yes`-instance of 3D-Matching. □

As previously mentioned, in [117], the authors give a reduction from Connected Dominating Set to Steiner Tree that works in any subclass of Chordal graphs without changing the input graph. We then get the following corollary.

Corollary 5. Steiner Tree *is* NP*-complete when restricted to undirected path graphs of diameter at most 3.*

In [79], the author proves that deciding whether an undirected path graph has a dominating clique of size at most k can be done in polynomial time. We apply his result to get a dichotomy for Steiner Tree restricted to Undirected Path graphs in terms of the diameter of the input graph. For this, we first need some definitions and tool lemmas, which are presented below.

Let G be a connected graph and $X \subseteq V(G)$ be a non-empty set. We denote by $ST(G, X)$ the minimum size of a subset $S \subseteq V(G) \setminus X$ such that $S \cup X$ induces a connected subgraph of G. Throughout the remainder of this section, we assume without loss of generality that $|X| \geq 3$ and that X does not induce a connected subgraph of G, as otherwise $ST(G, X)$ would be easily determined: if $|X| = 1$ or $G[X]$ is connected, then trivially $ST(G, X) = 0$; and, if $X = \{u, v\}$, then $ST(G, X)$ is equal to the number of vertices in any minimum path between u and v in G.

We say that two distinct vertices $u, v \in V(G)$ are *twins* in G if they have the same neighborhood in G, *i.e.* either $N_G(u) = N_G(v)$ or $N_G[u] = N_G[v]$. We prove in the next lemma that we can suppose without loss of generality that G has no twins. Observe that the hypothesis involving u and v can be assumed without loss of generality. It is included in order to write the equation in a more concise way.

Lemma 6. *Let G be a connected graph, containing twin vertices u and v, and $X \subseteq V(G)$ with $|X| \geq 3$. Also, suppose that $u \in X$ implies $v \in X$. Then,*

$$ST(G, X) = ST(G - u, X - u).$$

Proof. First, let $S \subseteq V(G - u) \setminus X$ be a minimum set such that $S \cup (X - u)$ induces a connected subgraph of $G - u$. We want to prove that $S \cup X$ induces a connected subgraph of G, in which case we get $ST(G, X) \leq ST(G - u, X - u)$. Suppose first that $v \in S$. Since $X \setminus \{u\} \neq \emptyset$, v must have some neighbor $w \in S \cup X$ in $G - u$. Then, it follows from the hypothesis that u and v are twins in G that w is also a neighbor of u in G. This implies that $S \cup X$ induces a connected subgraph of G. A similar argument can be applied when $v \in X$. Indeed, since $|X| \geq 3$, $X \setminus \{u, v\} \neq \emptyset$. Thus, v must have some neighbor

$w \in S \cup (X - u)$, which is also a neighbor of u in G, and consequently $S \cup X$ induces a connected subgraph of G. Finally, if $v \notin S \cup X$, then by hypothesis we also know that $u \notin X$, in which case trivially $S \cup X$ induces a connected subgraph of G.

Now, let $S \subseteq V(G) \setminus X$ be a minimum set such that $S \cup X$ induces a connected subgraph of G. Since S is minimum, $|S \cap \{u, v\}| \leq 1$. If $u \in S$, then, by the minimality of S, $v \notin S$ and $(S \setminus \{u\}) \cup \{v\}$ witnesses $ST(G - u, X - u) \leq ST(G, X)$. On the other hand, if $u \notin S$, then we trivially get that $S \cup (X - u)$ induces a connected subgraph of $G - u$, and therefore $ST(G - u, X - u) \leq ST(G, X)$. □

Based on Lemma 6, we assume from now on that the input graph G has no twin vertices. Also, in the remainder of the text, $(T, \{T_u\}_{u \in V(G)})$ is a tree model of G. Moreover, given a node $t \in V(T)$, we denote by V_t the set $\{u \in V(G) \colon t \in V(T_u)\}$. We say that $u \in V(G)$ is a *leafy vertex* if $V(T_u) = \{\ell_u\}$ and ℓ_u is a leaf in T; denote by $\mathcal{L}$ the set of leafy vertices, and for every $u \in \mathcal{L}$, denote by ℓ_u the unique node in T_u. We also say that $(T, \{T_u\}_{u \in V(G)})$ is *minimal* if there are no two adjacent nodes $t, t' \in V(T)$ such that $V_t \subseteq V_{t'}$. It is known that such a tree model can be computed in polynomial time [56]. We prove in the following lemma that, for any minimal tree model $(T, \{T_u\}_{u \in V(G)})$, there is a one-to-one correspondence between the leaves of T and the leafy vertices associated with T.

Lemma 7. *Let G be a connected undirected path graph without twin vertices, and $(T, \{T_u\}_{u \in V(G)})$ be a minimal tree model for G. Then, for every leaf ℓ of T, there exists a unique $u \in \mathcal{L}$ such that $\mathcal{L} \cap V_\ell = \{u\}$.*

Proof. Since G has no twin vertices, for every leaf t of T, there exists at most one leafy vertex u of G associated with T such that $\ell_u = t$. On the other hand, suppose that there exists a leaf t of T such that there is no leafy vertex of G associated with T corresponding to t, i.e., for every leafy vertex u of G associated with T, we have that $\ell_u \neq t$. Then, let t' be the parent of t in T. One can readily verify that $V_t \subseteq V_{t'}$, contradicting the fact that we are on a minimal tree model. □

A vertex u is called *simplicial* if $N_G(u)$ is a clique in G. Note that every leafy vertex is simplicial. Moreover, note that a simplicial vertex that is not in X certainly is not contained in any minimum Steiner tree for X; therefore we can suppose that X contains every simplicial vertex of G and, in particular, $\mathcal{L} \subseteq X$. In the next lemma, we prove that we can suppose that every $x \in X$ is either leafy, or is such that T_x contains no leaf of T.

Lemma 8. *Let G be a connected undirected path graph without twin vertices. Also, let $(T, \{T_u\}_{u \in V(G)})$ be a minimal tree model of G, $\mathcal{L}$ be the set of leafy vertices associated with T, and let $X \subseteq V(G)$ be a set of terminals such that $\mathcal{L} \subseteq X \subseteq V(G)$. Suppose that $x \in X \setminus \mathcal{L}$ is such that $\ell \in V(T_x)$ for some leaf ℓ of T. Then, $ST(G, X) = ST(G - u, X - u)$, where $\mathcal{L} \cap V_\ell = \{u\}$.*

Proof. By Lemma 7, $G - u$ is the graph related to the tree model $T - \ell$. Suppose that there exists a set $S \subseteq V(G - u) \setminus X$ such that $S \cup (X - u)$ induces a connected subgraph of $G - u$. Since $\ell \in T_u \cap T_x$, $ux \in E(G)$. Thus, $S \cup X$ induces a connected subgraph of G, and consequently $ST(G, X) \leq ST(G - u, X - u)$. Conversely, suppose that there exists a set $S \subseteq V(G) \setminus X$ such that $S \cup X$ induces a connected subgraph of G. Since u is a simplicial vertex of G, we obtain that $S \cup (X - u)$ induces a connected subgraph of $G - u$, and therefore $ST(G, X) \geq ST(G - u, X - u)$. □

In the proof, we modify a subset S that gives a solution in order to ensure that the new set is a clique. The following lemma will help us do that.

Lemma 9. *Let G be a connected undirected path graph, $(T, \{T_u\}_{u \in V(G)})$ be a tree model of G, $X \subseteq V(G)$ be a set of terminals, $S \subseteq V(G) \setminus X$ be a set such that $G[S \cup X]$ is connected, and let $u, v \in S$. If $y, z \in V(G)$ are such that $N_G(u) \cup N_G(v) \subseteq N_G(y) \cup N_G(z)$, then $S' \cup X$ is connected, where $S' = ((S \setminus \{u, v\}) \cup \{y, z\}) \setminus X$.*

Proof. Suppose otherwise, and let H, H' be distinct components of $G[S' \cup X]$. This means that there exist $w \in V(H)$ and $w' \in V(H')$ such that every path P between w and w' goes through u and/or v; but since $N_G(u) \cup N_G(v) \subseteq N_G(y) \cup N_G(z)$, it means that u and/or v can be replaced by y and/or z. □

We are now ready to prove our theorem. In [79], the author proves that deciding whether an undirected path graph has a dominating clique of size at most k can be done in polynomial time. We make a polynomial reduction from Connected Dominating Set to Dominating Clique, thus getting the desired polynomial algorithm by the equivalence given in [117]. This and Theorem 4 give a dichotomy of both Connected Dominating Set and Steiner Tree in terms of the diameter of the input graph G. We observe that, in order to prove that Dominating Clique is polynomial-time solvable for Undirected Path graphs, it is used in [79] an alternative notion of tree model called *characteristic tree*, where the nodes of the model are the maximal cliques of the graph G (see [56,91]).

Theorem 10. Steiner Tree *and* Connected Dominating Set *can be solved in polynomial time when restricted to undirected path graphs of diameter at most 2.*

Proof. In view of the reduction from Connected Dominating Set to Steiner Tree presented in [117], it suffices to prove that Steiner Tree can be solved in polynomial time. Thus, let G be a connected undirected path graph with diameter at most 2, $X \subseteq V(G)$ be a terminal set such that $|X| \geq 3$, and let κ be a positive integer.

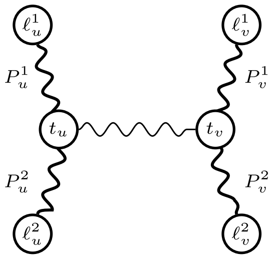


Fig. 5. The bold lines represent the paths T_u and T_v.

As usual, we consider a minimal tree model $(T, \{T_u\}_{u\in V(G)})$ of G. There is no loss of generality making these assumptions, since, as previously mentioned, it is known that such a tree model can be computed in polynomial time [56]. Let $\mathcal{L}$ denote the set of leafy vertices associated with T, and assume that $\mathcal{L} \subseteq X$.

In what follows, we prove that: $ST(G, X) \leq \kappa$ if and only if there exists a clique $S \subseteq V(G) \setminus X$ in G of size at most κ that dominates $\mathcal{L}$. Our theorem follows since this is exactly what is computed in the algorithm presented in [79].

For the sufficiency part of our claim, we just note that if S is a clique of G that dominates $\mathcal{L}$, then $\bigcup_{u\in S} V(T_u) = V(T)$, which means that in fact S is a dominating clique of G and therefore $S \cup X$ induces a connected subgraph of G. Because $|S| \leq \kappa$, it follows that $ST(G, X) \leq \kappa$.

Now, to prove necessity, suppose first that $ST(G, X) \leq \kappa$, and let $S \subseteq V(G)\setminus X$ be a set such that $S\cup X$ induces a connected subgraph of G. Suppose that S is minimum and that, among all such subsets of minimum cardinality, S maximizes the number of edges in $E(G[S])$. In addition, by Lemma 8, we can suppose that there are no edges in G between the vertices belonging to $X \setminus \mathcal{L}$ and the vertices belonging to $\mathcal{L}$. Moreover, note that $\mathcal{L}$ is an independent set. Thus, since $G[S \cup X]$ is connected, we get that every vertex in $\mathcal{L}$ must be adjacent to some vertex in S; in other words, S dominates $\mathcal{L}$. Thus, it remains to prove that S is a clique of G. Suppose for the sake of contradiction that there exist two distinct vertices $u, v \in S$ such that $uv \notin E(G)$. We prove that one can find a pair x, y of adjacent vertices in G such that $N_G(u)\cup N_G(v) \subseteq N_G(x)\cup N_G(z)$. Then, based on Lemma 9, by letting $S' = ((S\setminus\{u, v\})\cup\{y, z\})\setminus X$, we obtain a contradiction, since in this case either $|S'| < |S|$, or $|E(G[S'])| > |E(G[S])|$.

Let $t_u \in V(T_u)$ and $t_v \in V(T_v)$ be nodes whose distance from each other in T is the smallest possible (observe Fig. 5). Note that $t_u \neq t_v$ since $V(T_u) \cap V(T_v) = \emptyset$. Also, let P_u^1, P_u^2 be the two subpaths defined by t_u in T_u, and define P_v^1, P_v^2 similarly. For each $i \in \{1, 2\}$, let ℓ_u^i be the end vertex of P_u^i different from t_u, if it exists; otherwise, let ℓ_u^i be equal to t_u. Define ℓ_v^1, ℓ_v^2 similarly. Note that, if $\ell_u^1 \neq t_u$, then we can suppose that there exist $w_u^1 \in V(G)$ such that $\ell_u^1 \in T_{w_u^1}$, and $t \notin T_{w_u^1}$, where t is the neighbor of ℓ_u^1 in P_u^1 (otherwise, we could contract the edge $\ell_u^1 t$ and still have a tree model of G). Define w_u^2, w_v^1, w_v^2 similarly. There are two possible cases to be considered.

Case 1. Suppose that all the vertices $w_u^1, w_u^2, w_v^1, w_v^2$ exist and are well-defined. Since G has diameter at most 2, there must exist vertices $y \in N_G(w_u^1) \cap N_G(w_v^1)$ and $z \in N_G(w_u^2) \cap N_G(w_v^2)$. Clearly, the path between t_u and t_v in T is contained in the paths T_y and T_z (which means that y and z are adjacent in G), and $V(P_u^1 \cup P_v^1) \subseteq V(T_y)$, and $V(P_u^2 \cup P_v^2) \subseteq V(T_z)$. Thus, $N_G(u) \cup N_G(v) \subseteq N_G(y) \cup N_G(z)$, as desired.

Case 2. Now, suppose that some of the vertices $w_u^1, w_u^2, w_v^1, w_v^2$ do not exist or are not well-defined. Note that, since $S\cup X$ induces a connected subgraph of G, there must exist a path in $G[S \cup X]$ between u and v, which means that there must exist $w \in (S \cup X) \setminus \{u\}$ such that $t_u \in V(T_w)$. This implies that at least one of the vertices w_u^1, w_u^2 is well-defined, as otherwise $V(T_u) = \{t_u\} \subseteq V(T_w)$ and we could just remove u from S. The same argument can be applied with respect to w_v^1, w_v^2. Thus, suppose without loss of generality that w_u^1, w_v^1 are well-defined, and that w_u^2 is not well-defined (which means that $V(P_u^2) = \{t_u\}$). Pick y as before, and note that $V(T_u) \subseteq V(T_y)$, and that $yv \in E(G)$ since $\ell_v^1 \in V(T_y)\cap V(T_v)$. We can then apply Lemma 9 to $\{u, v\}$ and $\{y, v\}$ since $N_G(u)\cup N_G(v) \subseteq N_G(y)\cup N_G(v)$. □

5. Stubborn puzzles 35 years later

After 40 years, two open problems from [GJ] are still unsolved, namely: Open1 Graph Isomorphism, and Open8 Precedence constrained 3-processor schedule. In STOC 2016, László Babai announced that Graph Isomorphism could be solved in Quasipolynomial Time. Only one O entry from [OG] remains stubbornly open for 35 years: the complexity of Chromatic Index for Planar graphs. It is a puzzle to understand why still today the Chromatic Index column has the majority of thirteen O? entries, for instance Chromatic Index for Cographs is a long-standing open problem, as mentioned in [75]. We invite the reader to find a reference or a proof for the underlined [OG] entry in Table 1 corresponding to a "private communication" which would classify as polynomial Steiner Tree restricted to Circle graphs. Our proposed Table 2 leaves as open the parameterized complexity classification of Partition into Cliques for Line graphs. We invite the reader to further study the eight XP entries, observing that five of them belong to our target Steiner Tree column. In particular, we highlight that, even though the closely related problem Connected Dominating Set is known to be

FPT for Claw-Free graphs [61], it is open whether κ-Steiner Tree is also FPT for Line and Claw-Free graphs. Regarding the obtained second dichotomy for the Steiner Tree problem restricted to Undirected Path graphs, according to the diameter of the input graph, we should mention that Connected Dominating Set was proven to be NP-complete and W[2]-hard even when restricted to Split graphs of diameter 2 [83]. A straightforward modification of their proof leads to the NP-completeness of Steiner Tree (and to the W[2]-hardness of κ-Steiner Tree) when restricted to Split graphs of diameter 2.

Acknowledgments

We would like to thank anonymous reviewers for their thorough reading and numerous insightful suggestions and comments; in particular, we would like to thank one of the reviewers for the references related to mim-width. We are grateful to Vinicius F. Santos for sharing Ref. [1], for the NP-completeness proof of MaxCut for Interval graphs. This work was partially supported by the Brazilian agencies CNPq, Brazil (Grant Nos.: 407635/2018-1, 140399/2017-8, 407430/2016-4, 437841/2018-9, 401519/2016-3, and 304576/2017-4), CAPES, Brazil (Finance Code 001), FAPERJ, Brazil (Grant Nos.: CNE E-26/202.793/2017), and CNPq/FUNCAP, Brazil (PNE-0112-00061.01.00/16).

References

[1] R. Adhikary, K. Bose, S. Mukherjee, B. Roy, Complexity of maximum cut on interval graphs, 2020, arXiv:2006.00061.
[2] T. Akiyama, T. Nishizeki, N. Saito, NP-completeness of the Hamiltonian cycle problem for bipartite graphs, J. Inf. Process. 3 (2) (1980) 73–76.
[3] N. Alon, S. Gutner, Linear time algorithms for finding a dominating set of fixed size in degenerated graphs, Algorithmica 54 (4) (2009) 544.
[4] S. Arnborg, D.G. Corneil, A. Proskurowski, Complexity of finding embeddings in a k-tree, SIAM J. Algebr. Discrete Methods 8 (2) (1987) 277–284.
[5] S. Arnborg, A. Proskurowski, Linear time algorithms for NP-hard problems restricted to partial k-trees, Discrete Appl. Math. 23 (1) (1989) 11–24.
[6] L. Babai, E.M. Luks, Canonical labeling of graphs, in: Proceedings of the Fifteenth Annual ACM Symposium on Theory of Computing, 1983, pp. 171–183.
[7] L. Babel, I. Ponomarenko, G. Tinhofer, The isomorphism problem for directed path graphs and for rooted directed path graphs, J. Algorithms 21 (3) (1996) 542–564.
[8] B. Baker, Approximation algorithms for NP-complete problems on planar graphs, J. ACM 41 (1) (1994) 153–180.
[9] F. Barahona, On the Complexity of Max Cut, Technical Report, Université Scientifique et Medicale et Institut National Polytechnique de Grenoble, France, 1980.
[10] R. Belmonte, M. Vatshelle, Graph classes with structured neighborhoods and algorithmic applications, Theoret. Comput. Sci. 511 (2013) 54–65.
[11] B. Bergougnoux, M.M. Kanté, More applications of the d-neighbor equivalence: Connectivity and acyclicity constraints, in: 27th Annual European Symposium on Algorithms, ESA 2019, Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2019.
[12] F. Berman, D. Johnson, T. Leighton, P.W. Shor, L. Snyder, Generalized planar matching, J. Algorithms 11 (2) (1990) 153–184.
[13] A.A. Bertossi, M.A. Bonuccelli, Hamiltonian circuits in interval graph generalizations, Inform. Process. Lett. 23 (4) (1986) 195–200.
[14] M.D. Biasi, Polynomial problems in graph classes defined by forbidden induced cyclic subgraphs, Theoretical Computer Science Stack Exchange. https://cstheory.stackexchange.com/q/24882 (version: 2014-06-15).
[15] A. Björklund, T. Husfeldt, P. Kaski, M. Koivisto, Fourier meets Möbius: fast subset convolution, in: Proceedings of the Thirty-Ninth Annual ACM Symposium on Theory of Computing, 2007, pp. 67–74.
[16] M. Blanchette, E. Kim, A. Vetta, Clique cover on sparse networks, in: 2012 Proceedings of the Fourteenth Workshop on Algorithm Engineering and Experiments, ALENEX, SIAM, 2012, pp. 93–102.
[17] H.L. Bodlaender, Polynomial algorithms for graph isomorphism and chromatic index on partial k-trees, J. Algorithms 11 (4) (1990) 631–643.
[18] H. Bodlaender, A partial k-arboretum of graphs with bounded treewidth, Theoret. Comput. Sci. 209 (1–2) (1998) 1–45.
[19] H. Bodlaender, N. Brettell, M. Johnson, G. Paesani, D. Paulusma, E.J. van Leeuwen, Steiner trees for hereditary graph classes: a treewidth perspective, 2020, arXiv:2004.07492.
[20] H.L. Bodlaender, K. Jansen, On the complexity of the maximum cut problem, in: Annual Symposium on Theoretical Aspects of Computer Science, Springer, 1994, pp. 769–780.
[21] É. Bonnet, N. Bousquet, P. Charbit, S. Thomassé, R. Watrigant, Parameterized complexity of independent set in H-free graphs, in: 13th International Symposium on Parameterized and Exact Computation, 2019.
[22] K.S. Booth, C.J. Colbourn, Problems Polynomially Equivalent to Graph Isomorphism, Technical Report CS-77-04, Computer Science Department, University of Waterloo, Waterloo, Ont., 1979, Available on https://cs.uwaterloo.ca/research/tr/1977/CS-77-04.pdf.
[23] K. Booth, J. Johnson, Dominating sets in chordal graphs, SIAM J. Comput. 11 (1) (1982) 191–199.
[24] N. Bousquet, D. Gonçalves, G.B. Mertzios, C. Paul, I. Sau, S. Thomassé, Parameterized domination in circle graphs, Theory Comput. Syst. 54 (1) (2014) 45–72.
[25] A. Brandstädt, C. Hundt, F. Mancini, P. Wagner, Rooted directed path graphs are leaf powers, Discrete Math. 310 (4) (2010) 897–910.
[26] C. Buchheim, L. Zheng, Fixed linear crossing minimization by reduction to the maximum cut problem, in: International Computing and Combinatorics Conference, Springer, 2006, pp. 507–516.
[27] B.-M. Bui-Xuan, J.A. Telle, M. Vatshelle, Fast dynamic programming for locally checkable vertex subset and vertex partitioning problems, Theoret. Comput. Sci. 511 (2013) 66–76.
[28] L. Cai, J.A. Ellis, NP-completeness of edge-colouring some restricted graphs, Discrete Appl. Math. 30 (1) (1991) 15–27.
[29] M. Cerioli, L. Faria, T. Ferreira, C. Martinhon, F. Protti, B. Reed, Partition into cliques for cubic graphs: Planar case, complexity and approximation, Discrete Appl. Math. 156 (12) (2008) 2270–2278.
[30] J. Chen, I.A. Kanj, L. Perković, E. Sedgwick, G. Xia, Genus characterizes the complexity of certain graph problems: Some tight results, J. Comput. System Sci. 73 (6) (2007) 892–907.
[31] J. Chen, I.A. Kanj, G. Xia, Improved parameterized upper bounds for vertex cover, in: International Symposium on Mathematical Foundations of Computer Science, Springer, 2006, pp. 238–249.
[32] B.N. Clark, C.J. Colbourn, D.S. Johnson, Unit disk graphs, Discrete Math. 86 (1–3) (1990) 165–177.
[33] C.J. Colbourn, L. Stewart, Dominating cycles in series-parallel graphs, Ars Combin. 19 (1985) 107–112.
[34] G. Cornuejols, X. Liu, K. Vuskovic, A polynomial algorithm for recognizing perfect graphs, in: Proceedings 44th Annual IEEE Symposium on Foundations of Computer Science, 2003, 2003, pp. 20–27.

[35] B. Courcelle, J.A. Makowsky, U. Rotics, Linear time solvable optimization problems on graphs of bounded clique-width, Theory Comput. Syst. 33 (2) (2000) 125–150.

[36] B. Courcelle, S. Olariu, Upper bounds to the clique width of graphs, Discrete Appl. Math. 101 (1–3) (2000) 77–114.

[37] M. Cygan, F.V. Fomin, Ł. Kowalik, D. Lokshtanov, D. Marx, M. Pilipczuk, M. Pilipczuk, S. Saurabh, Parameterized Algorithms, Vol. 4, Springer, 2015.

[38] M. Cygan, G. Philip, M. Pilipczuk, M. Pilipczuk, J.O. Wojtaszczyk, Dominating set is fixed parameter tractable in claw-free graphs, Theoret. Comput. Sci. 412 (50) (2011) 6982–7000.

[39] P. Damaschke, The Hamiltonian circuit problem for circle graphs is NP-complete, Inform. Process. Lett. 32 (1) (1989) 1–2.

[40] S. Datta, P. Nimbhorkar, T. Thierauf, F. Wagner, Graph Isomorphism for $K_{3,3}$-free and K_5-free graphs is in Log-space, in: IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, Vol. 4, 2009, pp. 145–156.

[41] H.N. de Ridder, et al., Information system on graph classes and their inclusions (ISGCI), https://www.graphclasses.org.

[42] E.D. Demaine, M.T. Hajiaghayi, D.M. Thilikos, Exponential speedup of fixed-parameter algorithms on $K_{3,3}$-minor-free or K_5-minor-free graphs, in: Algorithms and Computation, Springer Berlin Heidelberg, 2002, pp. 262–273.

[43] E.D. Demaine, M.T. Hajiaghayi, D.M. Thilikos, Exponential speedup of fixed-parameter algorithms for classes of graphs excluding single-crossing graphs as minors, Algorithmica 41 (4) (2004) 245–267.

[44] J.S. Deogun, G. Steiner, Polynomial algorithms for Hamiltonian cycle in cocomparability graphs, SIAM J. Comput. 23 (3) (1994) 520–552.

[45] A.K. Dewdney, Fast Turing Reductions Between Problems in NP: Chapter 4: Reductions Between NP-Complete Problems, Department of Computer Science, University of Western Ontario, 1981.

[46] R. Diestel, Graph Theory, Springer Berlin Heidelberg, 2017.

[47] R.G. Downey, M.R. Fellows, Parameterized computational feasibility, in: P. Clote, J.B. Remmel (Eds.), Feasible Mathematics II, Birkhäuser Boston, Boston, MA, 1995, pp. 219–244.

[48] R.G. Downey, M.R. Fellows, Parameterized Complexity, in: Monographs in Computer Science, Springer Verlag, 1999.

[49] R.G. Downey, M.R. Fellows, Fundamentals of Parameterized Complexity, in: Texts in Computer Science, Springer, 2013.

[50] S.E. Dreyfus, R.A. Wagner, The Steiner problem in graphs, Networks 1 (3) (1971) 195–207.

[51] J. Ellis, H. Fan, M. Fellows, The dominating set problem is fixed parameter tractable for graphs of bounded genus, J. Algorithms 52 (2) (2004) 152–168.

[52] F.V. Fomin, D. Lokshtanov, S. Saurabh, M. Zehavi, Kernelization: Theory of Parameterized Preprocessing, Cambridge University Press, 2019.

[53] M.R. Garey, D.S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, W. H. Freeman & Co., New York, 1979.

[54] M.R. Garey, D.S. Johnson, G.L. Miller, C.H. Papadimitriou, The complexity of coloring circular arcs and chords, SIAM J. Algebr. Discrete Methods 1 (2) (1980) 216–227.

[55] M. Garey, D. Johnson, L. Stockmeyer, Some simplified NP-complete graph problems, Theoret. Comput. Sci. 1 (1976) 237–267.

[56] F. Gavril, A recognition algorithm for the intersection graphs of paths in trees, Discrete Math. 23 (3) (1978) 211–227.

[57] E.M. Gurari, I.H. Sudborough, Improved dynamic programming algorithms for bandwidth minimization and the mincut linear arrangement problem, J. Algorithms 5 (4) (1984) 531–546.

[58] F. Gurski, The behavior of clique-width under graph operations and graph transformations, Theory Comput. Syst. 60 (2) (2017) 346–376.

[59] V. Guruswami, Maximum cut on line and total graphs, Discrete Appl. Math. 92 (2–3) (1999) 217–221.

[60] T.W. Haynes, S. Hedetniemi, P. Slater, Fundamentals of Domination in Graphs, CRC Press, 1998.

[61] D. Hermelin, M. Mnich, E.J.V. Leeuwen, G. Woeginger, Domination when the stars are out, ACM Trans. Algorithms 15 (2) (2019) 1–90.

[62] P. Hliněnỳ, S.-i. Oum, Finding branch-decompositions and rank-decompositions, SIAM J. Comput. 38 (3) (2008) 1012–1032.

[63] I. Holyer, The NP-completeness of edge-coloring, SIAM J. Comput. 10 (4) (1981) 718–720.

[64] A. Itai, C.H. Papadimitriou, J.L. Szwarcfiter, Hamilton paths in grid graphs, SIAM J. Comput. 11 (4) (1982) 676–686.

[65] L. Jaffke, O.-j. Kwon, T.J. Strømme, J.A. Telle, Mim-width III. Graph powers and generalized distance domination problems, Theoret. Comput. Sci. 796 (2019) 216–236.

[66] D.S. Johnson, The NP-completeness column: an ongoing guide, J. Algorithms 6 (3) (1985) 434–451.

[67] M. Jones, D. Lokshtanov, M.S. Ramanujan, S. Saurabh, O. Suchý, Parameterized complexity of directed Steiner tree on sparse graphs, SIAM J. Discrete Math. 31 (2) (2017) 1294–1327.

[68] V. Kalisz, P. Klavík, P. Zeman, Circle graph isomorphism in almost linear time, 2019, arXiv:1908.09151v1.

[69] M. Kamiński, V.V. Lozin, M. Milanič, Recent developments on graphs of bounded clique-width, Discrete Appl. Math. 157 (12) (2009) 2747–2761.

[70] R. Karp, Reducibility among combinatorial problems, in: R. Miller, J. Thatcher, J. Bohlinger (Eds.), Chapter Complexity of Computer Computations, Plenum, New York, 1972, pp. 85–103.

[71] J. Keil, The complexity of domination problems in circle graphs, Discrete Appl. Math. 42 (1) (1993) 51–63.

[72] J. Keil, R. Laskar, P. Manuel, The vertex clique cover problem and some related problems in chordal graphs, in: Abstract Presented at SIAM Conference on Discrete Mathematics, 1994, Albuquerque, New Mexico, submitted for publication in 1997.

[73] J. Keil, L. Stewart, Approximating the minimum clique cover and other hard problems in subtree filament graphs, Discrete Appl. Math. 154 (14) (2006) 1983–1995.

[74] S. Khot, V. Raman, Parameterized complexity of finding subgraphs with hereditary properties, in: Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2000, pp. 137–147.

[75] D. Kobler, U. Rotics, Edge dominating set and colorings on graphs with fixed clique-width, Discrete Appl. Math. 126 (2–3) (2003) 197–221.

[76] E. Korach, N. Solel, Linear Time Algorithm for Minimum Weight Steiner Tree in Graphs with Bounded Treewidth, Technical Report 632, Israel Institute of Technology, 1990.

[77] R. Kothari, (https://cstheory.stackexchange.com/users/206/robin-kothari). Citation showing minors are topological minors for subcubic graphs. Theoretical Computer Science Stack Exchange, https://cstheory.stackexchange.com/q/7331, URL: https://cstheory.stackexchange.com/q/7331 (version: 2011-07-12).

[78] D. Král', J. Kratochvíl, Z. Tuza, G.J. Woeginger, Complexity of coloring graphs without forbidden induced subgraphs, in: Graph-Theoretic Concepts in Computer Science, Springer Berlin Heidelberg, 2001, pp. 254–262.

[79] D. Kratsch, Finding dominating cliques efficiently, in strongly chordal graphs and undirected path graphs, Discrete Math. 86 (1–3) (1990) 225–238.

[80] T. Krawczyk, Testing isomorphism of circular-arc graphs – Hsu's approach revisited, 2019, arXiv:1904.04501v3.

[81] R. Laskar, J. Pfaff, S.M. Hedetniemi, S.T. Hedetniemi, On the algorithmic complexity of total domination, SIAM J. Algebr. Discrete Methods 5 (3) (1984) 420–425.

[82] M.C. Lin, F.J. Soulignac, J.L. Szwarcfiter, A simple linear time algorithm for the isomorphism problem on proper circular-arc graphs, in: Scandinavian Workshop on Algorithm Theory, Springer, 2008, pp. 355–366.

[83] D. Lokshtanov, N. Misra, G. Philip, M.S. Ramanujan, S. Saurabh, Hardness of r-dominating set on graphs of diameter $(r+1)$, in: G. Gutin, S. Szeider (Eds.), Parameterized and Exact Computation, Springer International Publishing, 2013, pp. 255–267.

[84] G.S. Lueker, K.S. Booth, A linear time algorithm for deciding interval graph isomorphism, J. ACM 26 (2) (1979) 183–195.

[85] F. Maffray, M. Preissmann, On the NP-completeness of the k-colorability problem for triangle-free graphs, Discrete Math. 162 (1) (1996) 313–317.
[86] M. Mahajan, V. Raman, Parameterizing above guaranteed values: MaxSat and MaxCut, J. Algorithms 31 (2) (1999) 335–354.
[87] S. Malitz, Genus g graphs have pagenumber $O(\sqrt{g})$, J. Algorithms 17 (1) (1994) 85–109.
[88] D. Mölle, S. Richter, P. Rossmanith, Enumerate and expand: Improved algorithms for connected vertex cover and tree cover, Theory Comput. Syst. 43 (2) (2008) 234–253.
[89] B. Monien, How to find long paths efficiently, in: G. Ausiello, M. Lucertini (Eds.), Analysis and Design of Algorithms for Combinatorial Problems, in: North-Holland Mathematics Studies, Elsevier, 1985, pp. 239–254.
[90] B. Monien, I.H. Sudborough, Bandwidth constrained NP-complete problems, Theoret. Comput. Sci. 41 (1985) 141–167.
[91] C.L. Monma, V.K. Wei, Intersection graphs of paths in a tree, J. Combin. Theory Ser. B 41 (2) (1986) 141–181.
[92] E. Mujuni, F. Rosamond, Parameterized complexity of the clique partition problem, in: Proceedings of the Fourteenth Symposium on Computing: The Australasian Theory-Volume 77, 2008, pp. 75–78.
[93] H. Müller, Hamiltonian circuits in chordal bipartite graphs, Discrete Math. 156 (1–3) (1996) 291–298.
[94] H. Müller, A. Brandstädt, The NP-completeness of steiner tree and dominating set for chordal bipartite graphs, Theoret. Comput. Sci. 53 (2–3) (1987) 257–265.
[95] A. Munaro, Bounded clique cover of some sparse graphs, Discrete Math. 340 (9) (2017) 2208–2216.
[96] R. Niedermeier, Invitation to Fixed-Parameter Algorithms, Oxford University Press, 2006.
[97] S.-I. Oum, Approximating rank-width and clique-width quickly, ACM Trans. Algorithms 5 (1) (2008) 1–20.
[98] S.-i. Oum, P. Seymour, Approximating clique-width and branch-width, J. Combin. Theory Ser. B 96 (4) (2006) 514–528.
[99] B.S. Panda, D. Pradhan, NP-Completeness of Hamiltonian cycle problem on rooted directed path graphs, 2008, arXiv:0809.2443v1.
[100] G. Philip, V. Raman, S. Sikdar, Solving dominating set in larger classes of graphs: FPT algorithms and polynomial kernels, in: European Symposium on Algorithms, Springer, 2009, pp. 694–705.
[101] M. Pilipczuk, M. Pilipczuk, P. Sankowski, E.J.V. Leeuwen, Network sparsification for Steiner problems on planar and bounded-genus graphs, ACM Trans. Algorithms 14 (4) (2018) 1–73.
[102] R.V. Pocai, The complexity of SIMPLE MAX-CUT on comparability graphs, Electron. Notes Discrete Math. 55 (2016) 161–164, 14th Cologne-Twente Workshop on Graphs and Combinatorial Optimization (CTW16).
[103] S. Poljak, A note on stable sets and colorings of graphs, Comment. Math. Univ. Carolin. 15 (2) (1974) 307–309.
[104] V. Raman, S. Saurabh, Short cycles make W-hard problems hard: FPT algorithms for W-hard problems in graphs with no short cycles, Algorithmica 52 (2) (2008) 203–225.
[105] M. Rao, MSOL partitioning problems on graphs of bounded treewidth and clique-width, Theoret. Comput. Sci. 377 (1–3) (2007) 260–267.
[106] W. Shih, T. Chern, W.-L. Hsu, An $O(n^2 \log n)$ algorithm for the Hamiltonian cycle problem on circular-arc graphs, SIAM J. Comput. 21 (6) (1992) 1026–1046.
[107] J. Spinrad, Efficient Graph Representations, American Mathematical Society, Fields Institute, 2003.
[108] L. Stewart, Cographs - A Class of Tree Representable Graphs (Master's thesis), University of Toronto, Canada, 1978, also a Technical Report, 126/78, Department of Computer Science, University of Toronto.
[109] R. Sucupira, L. Faria, S. Klein, A complexidade do problema corte máximo para grafos fortemente cordais, in: Anais do XLV Simpósio Brasileiro de Pesquisa Operacional, 2013, pp. 2979–2988.
[110] M. Syslo, NP-complete problems on some tree-structured graphs: a review, in: Proc. WG'83 International Workshop on Graph Theoretic Concepts in Computer Science, Univ. Verlag Rudolf Trauner, Linz, West Germany, 1983.
[111] R. Uehara, S. Toda, T. Nagoya, Graph isomorphism completeness for chordal bipartite graphs and strongly chordal graphs, Discrete Appl. Math. 145 (3) (2005) 479–482.
[112] W. Unger, On the k-colouring of circle-graphs, in: R. Cori, M. Wirsing (Eds.), Proceedings of the Annual Symposium on Theoretical Aspects of Computer Science STACS 88, in: Lecture Notes in Computer Science, vol. 294, Springer, Berlin, Heidelberg, 1988, pp. 61–72.
[113] M. Vatshelle, New Width Parameters of Graphs (Ph.D. thesis), The University of Bergen, 2012.
[114] J.A. Wald, C.J. Colbourn, Steiner trees in outerplanar graphs, in: Proc. 13th Southeastern Conference on Combinatorics, Graph Theory, and Computing, 1982, pp. 15–22.
[115] J.A. Wald, C.J. Colbourn, Steiner trees, partial 2–trees, and minimum IFI networks, Networks 13 (2) (1983) 159–167.
[116] E. Wanke, k-NLC graphs and polynomial algorithms, Discrete Appl. Math. 54 (2–3) (1994) 251–266.
[117] K. White, M. Farber, W. Pulleyblank, Steiner trees, connected domination and strongly chordal graphs, Networks 15 (1985) 109–124.
[118] P. Winter, Steiner problem in Halin networks, Discrete Appl. Math. 17 (3) (1987) 281–294.
[119] M. Yannakakis, Node-and edge-deletion NP-complete problems, in: Proceedings of the Tenth Annual ACM Symposium on Theory of Computing, STOC'78, ACM Press, 1978, pp. 253–264.

Appendix B

Manuscript: A Multivariate Analysis of the Strict Terminal Connection Problem

This appendix contains the manuscript:
Alexsander A. de Melo, Celina M. H. de Figueiredo, Uéverton S. Souza. A Multivariate Analysis of the Strict Terminal Connection Problem. Published in *Journal of Computer and System Sciences* (2020) [96].

Journal of Computer and System Sciences 111 (2020) 22–41

Contents lists available at ScienceDirect

Journal of Computer and System Sciences

www.elsevier.com/locate/jcss

A multivariate analysis of the strict terminal connection problem [☆]

Alexsander A. Melo [a,*], Celina M.H. Figueiredo [a], Uéverton S. Souza [b]

[a] *Universidade Federal do Rio de Janeiro, Rio de Janeiro, Brazil*
[b] *Universidade Federal Fluminense, Niterói, Brazil*

ARTICLE INFO

Article history:
Received 30 August 2018
Received in revised form 29 January 2020
Accepted 4 February 2020
Available online 10 February 2020

Keywords:
Terminal vertices
Connection tree
Steiner tree
Bounded degree
Parameterized complexity

ABSTRACT

A strict connection tree of a graph G for a set W is a tree subgraph of G whose leaf set equals W. The Strict terminal connection problem (S-TCP) is a network design problem whose goal is to decide whether G admits a strict connection tree T for W with at most ℓ vertices of degree 2 and r vertices of degree at least 3. We establish a Poly vs. NP-c dichotomy for S-TCP with respect to ℓ and $\Delta(G)$. We prove that S-TCP parameterized by r is W[2]-hard even if ℓ is bounded by a constant; we provide a kernelization for S-TCP parameterized by ℓ, r and $\Delta(G)$, and we prove that such a version of the problem does not admit a polynomial kernel, unless NP $\subseteq$ coNP/poly. Finally, we analyze S-TCP on split graphs and cographs.

1. Introduction

Network design problems are combinatorial questions of great practical and theoretical interest. Indeed, such problems are challenging tasks closely related to real-world applications. In this paper, we investigate the computational complexity of a network design problem called Strict terminal connection.

A *connection tree* T of a graph $G = (V, E)$ for a *terminal set* $W \subseteq V$ is a tree subgraph of G such that $W \subseteq V(T)$ and every leaf of T belongs to W *cf.* [1,2]. In a connection tree T for W, the vertices belonging to $V(T) \setminus W$ are called *non-terminal* and are classified into two types according to their respective degrees in T, namely: the non-terminal vertices with degree exactly equal to 2 in T are called *linkers* and the non-terminal vertices with degree at least 3 in T are called *routers cf.* [1,2]. Thus, there exists a partition $\mathcal{V}_T = \{W, \mathsf{L}(T), \mathsf{R}(T)\}$ of the vertex set of a connection tree T into terminal vertices, linkers and routers, where $\mathsf{L}(T)$ and $\mathsf{R}(T)$ denote the linker and router sets of T, respectively.

In some applications, the terminal vertices must be leaves. For example, in telecommunications, the message senders and receivers, which correspond to the terminal vertices, are not allowed to behave as transmitters [3], which correspond to the vertices with degree greater than 1. A connection tree T for W is said *strict* if all vertices belonging to W are leaves of T, i.e. the leaf set of T coincides with the terminal set W. Based on that and also motivated by applications in information security and network routing, Dourado et al. [2] introduced the Strict terminal connection problem (S-TCP), which has as

[☆] This work was partially supported by the Brazilian agencies CNPq (Grant numbers: 132890/2015-1, 140399/2017-8, 407635/2018-1, 303726/2017-2), CAPES (Finance Code 001) and FAPERJ (Grant numbers: CNE E-26/202.793/2017, JCNE E-26/203.272/2017).

[*] Corresponding author.

E-mail addresses: aamelo@cos.ufrj.br (A.A. Melo), celina@cos.ufrj.br (C.M.H. Figueiredo), ueverton@ic.uff.br (U.S. Souza).

https://doi.org/10.1016/j.jcss.2020.02.001
0022-0000/© 2020 Elsevier Inc. All rights reserved.

Table 1
Contributions of this work (in bold) and known results for S-TCP.

Graph class	Parameters				
	–	ℓ	r	ℓ, r	ℓ, r, Δ
General	NP-c [2]	NP-c [2]	Poly for $r \in \{0, 1\}$ [29] but Open for $r \geq 2$, and **W[2]-h** Theorem 4	XP [2] but **W[2]-h** Theorem 4	FPT [2] (and Theorem 5) but **No-poly kernel** Theorem 6
$\Delta = 4$	**NP-c** Theorem 1	**NP-c** Theorem 1	Poly for $r \in \{0, 1\}$ [29] but Open for $r \geq 2$	FPT [2] (and Theorem 5)	FPT [2] (and Theorem 5)
$\Delta = 3$	**NP-c** Theorem 2	**XP** Theorem 3	Poly for $r \in \{0, 1\}$ [29] but Open for r ≥ 2	FPT [2] (and Theorem 5)	FPT [2] (and Theorem 5)
Split	**NP-c** Theorem 7	**NP-c** Theorem 7	**XP** Theorem 7 but **W[2]-h** Theorem 7	XP [2] (and Theorem 7) but **W[2]-h** Theorem 7	FPT [2] (and Theorem 5)
Cographs	**Poly** Theorem 8	**Poly** Theorem 8	**Poly** Theorem 8	**Poly** Theorem 8	**Poly** Theorem 8

input a graph $G = (V, E)$, a non-empty subset $W \subseteq V$ and two non-negative integers ℓ and r, and asks for the existence of a strict connection tree T of G for W such that $|\mathsf{L}(T)| \leq \ell$ and $|\mathsf{R}(T)| \leq r$.

Besides the practical point of view, S-TCP is strongly related to classical network design problems, such as vertex-disjoint path problems and integral network flow problems. Furthermore, S-TCP can be viewed as a close variant of the unweighted version of the Steiner tree problem in graphs, in which we are given a graph $G = (V, E)$, a terminal set $W \subseteq V$ and a positive integer k, and we aim to decide whether G contains a connected subgraph T such that $W \subseteq V(T)$ and $|E(T)| \leq k$. Since every minimal solution T for a given instance of Steiner tree is necessarily a connection tree for W, the constraints on T being a tree and its leaf set being a subset of W can be omitted without loss of generality from the definition of Steiner tree. However, for our target problem, S-TCP, neither constraint can be ignored. Indeed, as a result of the number of non-terminal vertices with degree 2 being bounded, there exist instances $I = (G, W, \ell, r)$ that would be considered Yes instances of S-TCP although all connected subgraphs of G containing the vertices in W, and with at most ℓ non-terminal vertices with degree 2 and at most r non-terminal vertices with degree at least 3, have cycles or non-terminal vertices that are leaves.

Steiner tree is a classical NP-complete problem [4], and it has been extensively studied from distinct classes of algorithmic paradigms, such as structured graph classes [5–9] and parameterized complexity [10–14].

Additionally, several variants of Steiner tree have been investigated over the years. One of the most well-known variants is the so-called Full Steiner tree (or Terminal Steiner tree), in which the terminal vertices are further constrained to be leaves of the sought connection tree T, i.e. T must be strict [15]. The original motivation to study the Full Steiner tree problem was to use it as a building block to solve the Steiner tree problem itself, provided the fact that any connection tree can be decomposed into strict connection trees [16,17]. Full Steiner tree was proved to be NP-complete [3,17,18]. On the other hand, Fernau et al. [19] proved that the problem is in FPT when parameterized by k, the maximum size of the sought strict connection tree T, but that it does not admit a strict polynomial kernel unless P = NP. It is also known that, unless NP $\subseteq$ coNP/poly, Full Steiner tree parameterized by k does not admit a polynomial kernel *cf.* [14,12]. In addition, many approximation algorithms and approximation lower bounds for the problem have been proposed [3,17,20–25] in the last years.

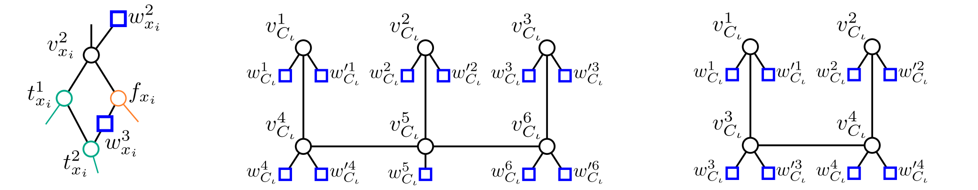
Motivated by applications in optical networks and bandwidth consumption minimization, another variant of Steiner tree that has been investigated is the one in which the number of *branching nodes*, i.e. vertices with degree at least 3 in T (not necessarily non-terminal), is bounded. In [26–28], the authors address the undirected and directed cases of this variant, for which they devise approximation and parameterized polynomial-time algorithms, apart from obtaining some intractability results.

Nevertheless, there is no variant of Steiner tree requiring simultaneously *full Steiner trees* and bounded number of *branching nodes* that has been investigated. Therefore, we emphasize that S-TCP certainly has its own merit to be studied. Thus, this paper aims to provide a multivariate analysis of S-TCP with respect to the input aspects: ℓ, number of linkers; r, number of routers; and Δ, the maximum degree of the input graph.

S-TCP was proved to be polynomial-time solvable if ℓ and r are bounded by constants [2], or if ℓ is unbounded but $r \in \{0, 1\}$ [29]; and it was proved to be in FPT when ℓ, r and Δ are parameters [2]. On the other hand, for every $\ell \geq 0$, S-TCP was proved to be NP-complete when r is unbounded, even if Δ is bounded by a constant [2].

In this paper, we extend the results described above by presenting several contributions to the complexity of S-TCP. More specifically, in Section 2, we establish a *Poly vs. NP-c* dichotomy for S-TCP with respect to ℓ and Δ; and in Section 3, we provide further complexity results for S-TCP parameterized by ℓ, r or Δ. Additionally, in Sections 4 and 5, we investigate the problem on split graphs and cographs, respectively. Finally, in Section 6, we present some directions for future work. Table 1 summarizes the contributions of this work.

Throughout this work we denote by n, m, and Δ the number of vertices, the number of edges, and the maximum degree of the input graph G, respectively.



(a) Gadget G_{x_i}.

(b) Gadget G_{C_ι}, for $|C_\iota| = 3$.

(c) Gadget G_{C_ι}, for $|C_\iota| = 2$.

Fig. 1. Gadgets G_{x_i} and G_{C_ι}.

2. Bounded maximum degree dichotomy

In this section, we address the analysis of the computational complexity of S-TCP when it is restricted to graphs with bounded maximum degree. More specifically, we prove that S-TCP is NP-complete even if ℓ is bounded by a constant and $\Delta = 4$, or ℓ is unbounded and $\Delta = 3$. On the other hand, we give a polynomial-time algorithm for the problem when ℓ is bounded by a constant and $\Delta = 3$. Observe that S-TCP is easily solvable if $\Delta \leq 2$. Thus, our results establish a *Poly vs. NP-c* dichotomy for S-TCP with respect to ℓ and Δ.

2.1. Hardness results

We first prove the NP-completeness of S-TCP with ℓ bounded by a constant and $\Delta = 4$. Our proof consists in a polynomial-time reduction from the NP-complete (*cf.* [30]) variant of 3-SAT, called 3-SAT(3), which has as input a set X of boolean variables and a set $\mathcal{C}$ of clauses over X such that: (1) each clause in $\mathcal{C}$ has two or three distinct literals; and (2) each variable in X appears exactly twice positive and once negative in the clauses belonging to $\mathcal{C}$; and asks for the existence of a truth assignment for the variables in X such that every clause in $\mathcal{C}$ has at least one true literal.

Theorem 1. *For every $\ell \geq 0$, S-TCP remains NP-complete even if $\Delta = 4$.*

Proof. Let $I = (X, \mathcal{C})$ be an instance of 3-SAT(3), where $X = \{x_1, x_2, \ldots, x_p\}$ and $\mathcal{C} = \{C_1, C_2, \ldots, C_q\}$. We construct from I an instance $f(I) = (G, W, r)$ of S-TCP with ℓ bounded by a constant, such that $\Delta = 4$, as follows (see Fig. 2):

- first, we create ℓ vertices $u_1, u_2, \ldots, u_\ell$ and, for each $i \in \{1, 2, \ldots, \ell - 1\}$, add the edges $u_i u_{i+1}$; moreover, we create the vertices w_I and v_I and add the edges $w_I u_1$ and $u_\ell v_I$, originating the path $P_I = \langle w_I, u_1, \ldots, u_\ell, v_I \rangle$;
- for each variable $x_i \in X$, we create the gadget G_{x_i} (see Fig. 1a) such that
  - $V(G_{x_i}) = \{v^2_{x_i}, w^2_{x_i}, w^3_{x_i}, t^1_{x_i}, t^2_{x_i}, f_{x_i}\}$ and
  - $E(G_{x_i}) = \{w^2_{x_i} v^2_{x_i}, v^2_{x_i} t^1_{x_i}, t^1_{x_i} t^2_{x_i}, t^2_{x_i} w^3_{x_i}, w^3_{x_i} f_{x_i}, f_{x_i} v^2_{x_i}\}$;

  moreover, we create the vertices $w^1_{x_i}$ and $v^1_{x_i}$, and we add the edges $w^1_{x_i} v^1_{x_i}$ and $v^1_{x_i} v^2_{x_i}$;
- we create a complete strict binary tree T_I, rooted at v_I, whose leaves are the vertices $v^1_{x_1}, v^1_{x_2}, \ldots, v^1_{x_p}$;
- for each clause $C_\iota \in \mathcal{C}$, we create the gadget G_{C_ι} such that, if $|C_\iota| = 3$, then (see Fig. 1b)
  - $V(G_{C_\iota}) = \{v^\kappa_{C_\iota}, w^\kappa_{C_\iota}, w'^\kappa_{C_\iota} \mid \kappa \in \{1, 2, 3, 4, 6\}\} \cup \{v^5_{C_\iota}, w^5_{C_\iota}\}$ and
  - $E(G_{C_\iota}) = \{v^\kappa_{C_\iota} w^\kappa_{C_\iota}, v^\kappa_{C_\iota} w'^\kappa_{C_\iota} \mid \kappa \in \{1, 2, 3, 4, 6\}\} \cup \{v^5_{C_\iota} w^5_{C_\iota}\} \cup \{v^1_{C_\iota} v^4_{C_\iota}, v^4_{C_\iota} v^5_{C_\iota}, v^5_{C_\iota} v^2_{C_\iota}, v^5_{C_\iota} v^6_{C_\iota}, v^6_{C_\iota} v^3_{C_\iota}\}$,

  and if $|C_\iota| = 2$, then (see Fig. 1c)
  - $V(G_{C_\iota}) = \{v^\kappa_{C_\iota}, w^\kappa_{C_\iota}, w'^\kappa_{C_\iota} \mid \kappa \in \{1, \ldots, 4\}\}$ and
  - $E(G_{C_\iota}) = \{v^\kappa_{C_\iota} w^\kappa_{C_\iota}, v^\kappa_{C_\iota} w'^\kappa_{C_\iota} \mid \kappa \in \{1, \ldots, 4\}\} \cup \{v^1_{C_\iota} v^3_{C_\iota}, v^3_{C_\iota} v^4_{C_\iota}, v^4_{C_\iota} v^2_{C_\iota}\}$;
- for each clause $C_\iota \in \mathcal{C}$, we add the edge $t^j_{x_i} v^\kappa_{C_\iota}$ if the κ-th literal belonging to C_ι corresponds to the j-th occurrence in I of the positive literal x_i, for $x_i \in X$, $j \in \{1, 2\}$ and $\kappa \in \{1, \ldots, |C_\iota|\}$; on the other hand, we add the edge $f_{x_i} v^\kappa_{C_\iota}$ if the κ-th literal belonging to C_ι corresponds to the (only) occurrence in I of the negative literal $\overline{x}_i$, for $x_i \in X$ and $\kappa \in \{1, \ldots, |C_\iota|\}$;
- we define $W = W_{\mathcal{C}} \cup \{w_I\} \cup \{w^1_{x_i}, w^2_{x_i}, w^3_{x_i} \mid x_i \in X\}$, where $W_{\mathcal{C}} = \bigcup_{C_\iota \in \mathcal{C}} W_{C_\iota}$ and
  - $W_{C_\iota} = \begin{cases} \{w^\kappa_{C_\iota}, w'^\kappa_{C_\iota} \mid \kappa \in \{1, 2, 3, 4, 6\}\} \cup \{w^5_{C_\iota}\} \text{ if } |C_\iota| = 3 \\ \{w^\kappa_{C_\iota}, w'^\kappa_{C_\iota} \mid \kappa \in \{1, \ldots, 4\}\} \text{ if } |C_\iota| = 2; \end{cases}$
- finally, we define $r = |V \setminus W| - \ell$.

One may verify that the maximum degree of G is 4.

Fig. 2 exemplifies the graph G and the terminal set W of $f(I)$.

Now, we prove that I is a Yes instance of 3-SAT(3) if and only if $f(I)$ is a Yes instance of S-TCP with ℓ bounded by a constant.

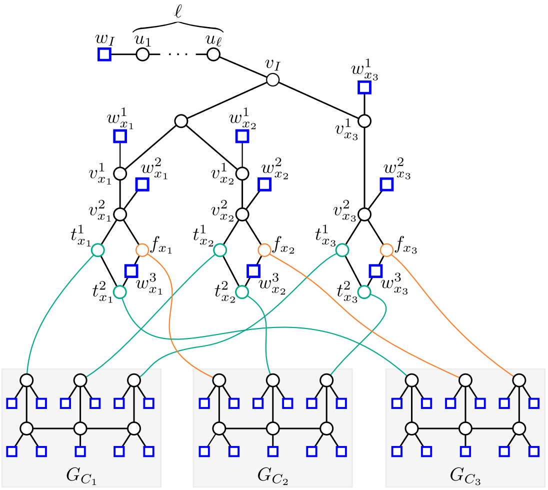


Fig. 2. Graph G and terminal set W (blue square vertices) of $f(I)$ obtained from the instance $I = (X, \mathcal{C})$ of 3-SAT(3), where $X = \{x_1, x_2, x_3\}$ and $\mathcal{C} = \{C_1 = \{x_1, x_2, x_3\}, C_2 = \{\bar{x}_1, x_2, x_3\}, C_3 = \{x_1, \bar{x}_2, \bar{x}_3\}\}$. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

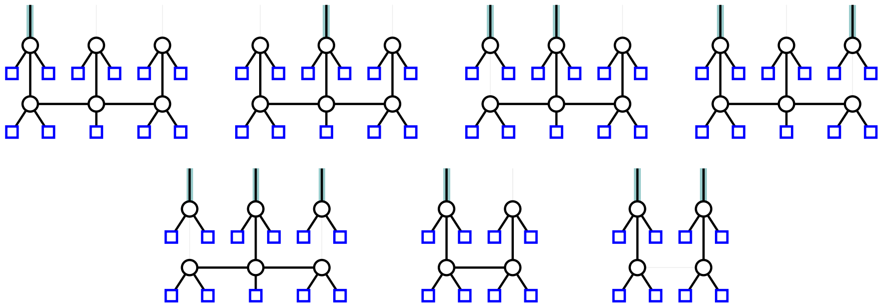


Fig. 3. Patterns used to connect the vertices of G_{C_ι} to T.

First, suppose that I is a Yes instance of 3-SAT(3). Hence, there exists a truth assignment $\alpha \colon X \to \{true, false\}$ that satisfies all clauses in $\mathcal{C}$. Based on α, we construct a strict connection tree T of G for W as follows:

- we add the path P_I to T;
- we add the complete strict binary tree T_I to T along with the vertices $w^1_{x_i}$ and the edges $w^1_{x_i} v_{x_i}$ for every $x_i \in X$;
- for each variable $x_i \in X$, we add the vertices $v^2_{x_i}$, $w^2_{x_i}$ and $w^3_{x_i}$ and the edges $v^1_{x_i} v^2_{x_i}$ and $v^2_{x_i} w^2_{x_i}$ to T; furthermore, if $\alpha(x_i) = true$, then we add the vertices $t^1_{x_i}$ and $t^2_{x_i}$ to T along with all of their neighbors and incident edges in G; on the other hand, if $\alpha(x_i) = false$, then we add the vertex f_{x_i} to T along with all of its neighbors and incident edges in G.

Since by hypothesis α satisfies all clauses in $\mathcal{C}$, for each $C_\iota \in \mathcal{C}$, there exists at least one vertex either $t^j_{x_i}$ (where $j \in \{1, 2\}$) or f_{x_i}, for some $x_i \in X$, which is adjacent to one of the vertices $v^1_{C_\iota}$, $v^2_{C_\iota}$ (and $v^3_{C_\iota}$ if $|C_\iota| = 3$) in T. Thus, we can connect all the other vertices of the gadget G_{C_ι} to T by following one of the patterns (or their symmetrical cases) depicted in Fig. 3, concluding the construction of T.

Fig. 4 exemplifies the strict connection tree T of G for W, referring to the instance $f(I)$ described in Fig. 2, obtained from a truth assignment α.

Observe that, T is indeed a strict connection tree of G for W and, besides that, $\mathsf{L}(T) = \{u_1, u_2, \ldots, u_\ell\}$ and $\mathsf{R}(T) = V(T_I) \cup \left(\bigcup_{C_\iota \in \mathcal{C}} V(G_{C_\iota}) \setminus W_C\right) \cup \{v^2_{x_i} \mid x_i \in X\} \cup \{t^1_{x_i}, t^2_{x_i} \mid \alpha(x_i) = true, x_i \in X\} \cup \{f_{x_i} \mid \alpha(x_i) = false, x_i \in X\}$. Hence, $|\mathsf{L}(T)| \le \ell$ and, obviously, $|\mathsf{R}(T)| \le |V \setminus W| - \ell$. Therefore, $f(I)$ is a Yes instance of S-TCP with ℓ bounded by a constant.

Conversely, suppose that $f(I)$ is a Yes instance of S-TCP with ℓ bounded by a constant. Hence, there exists a strict connection tree T of G for W such that $|\mathsf{L}(T)| \le \ell$ and $|\mathsf{R}(T)| \le |V \setminus W| - \ell$. Since the path P_I is necessarily contained in T and there are precisely ℓ vertices in P_I with degree 2, the vertices belonging to $V(T) \setminus (W \cup V(P_I))$ are routers of T. However, we have $d_G(t^1_{x_i}) = d_G(t^2_{x_i}) = d_G(f_{x_i}) = 3$. Thus, if $t^1_{x_i} \in V(T)$, then $N_T(t^1_{x_i}) = N_G(t^1_{x_i})$; if $t^2_{x_i} \in V(T)$, then $N_T(t^2_{x_i}) = N_G(t^2_{x_i})$; and, if $f_{x_i} \in V(T)$, then $N_T(f_{x_i}) = N_G(f_{x_i})$. Hence, if $t^1_{x_i} \in V(T)$ or $t^2_{x_i} \in V(T)$, then $f_{x_i} \notin V(T)$, otherwise T would have a cycle (and the degree of the terminal $w^3_{x_i}$ would be greater than 1 in T). Analogously, if $f_{x_i} \in V(T)$, then $t^1_{x_i}, t^2_{x_i} \notin V(T)$. Thus, we can define a truth assignment $\alpha \colon X \to \{true, false\}$ in the following way: $\alpha(x_i) = true$ if and only if $f_{x_i} \notin$

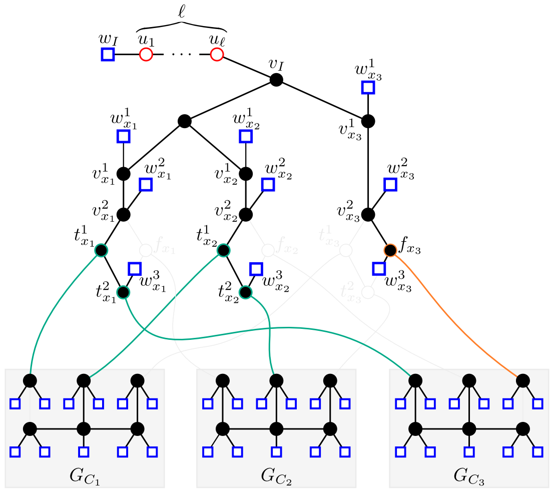


Fig. 4. Strict connection tree of G for W obtained from the truth assignment $\alpha(x_1) = true$, $\alpha(x_2) = true$ and $\alpha(x_3) = false$.

$V(T)$. Since by hypothesis $W_{\mathcal{C}} \subset W \subseteq V(T)$ and, for every clause $C_\iota \in \mathcal{C}$, the path in T between any terminal $w_{C_\iota} \in W_{C_\iota}$ and any other terminal $w \in W \setminus W_{C_\iota}$ must contain either $t^j_{x_i}$ (where $j \in \{1, 2\}$) or f_{x_i}, for some $x_i \in X$, we have that all clauses in $\mathcal{C}$ are satisfied by α. Therefore, I is a Yes instance of 3-SAT(3). □

Now, we analyze S-TCP when $\Delta = 3$. First, by a polynomial-time reduction from the 1-in-3-SAT(3) problem — which has the same input of 3-SAT(3) but asks whether there exists a truth assignment such that every clause has exactly (instead of at least) one true literal — we prove that, if ℓ is unbounded, then S-TCP remains NP-complete even if $\Delta = 3$.

The next proposition shows that 1-in-3-SAT(3) is an NP-complete problem and that, unless ETH fails, it does not admit a $2^{o(|X|+|\mathcal{C}|)}$-time algorithm. To prove such a result, we first present a polynomial-time reduction from 3-SAT (where each clause has exactly three distinct literals) to 1-in-3-SAT, which has the same input of 3-SAT and the same question of 1-in-3-SAT(3); then, we present a polynomial-time reduction from 1-in-3-SAT to 1-in-3-SAT(3).

For simplicity, in the context of 1-in-3-SAT and 1-in-3-SAT(3), we say that a truth assignment α *satisfies* a given clause C if there is exactly (instead of at least) one true literal in C under α.

Proposition 1. *1-in-3-SAT(3) is NP-complete and cannot be solved in time $2^{o(|X|+|\mathcal{C}|)}$, unless ETH fails.*

Proof. Polynomial-time reduction from 3-SAT to 1-in-3-SAT.

Let $I = (X', \mathcal{C}')$ be an instance of 3-SAT. We construct from I an instance $f(I) = (X'', \mathcal{C}'')$ of 1-in-3-SAT, as follows: $X'' = X' \cup \{y^1_j, y^2_j, y^3_j, y^4_j \mid C_j \in \mathcal{C}'\}$ and $\mathcal{C}'' = \{C^1_j, C^2_j, C^3_j \mid C_j \in \mathcal{C}'\}$, where, for $C_j = \{z^1_j, z^2_j, z^3_j\}$, with $z^i_j \in \{x^i_j, \overline{x}^i_j\}$ for some variable $x^i_j \in X'$, we have $C^1_j = \{\overline{z}^1_j, y^1_j, y^2_j\}$, $C^2_j = \{z^2_j, y^2_j, y^3_j\}$ and $C^3_j = \{\overline{z}^3_j, y^3_j, y^4_j\}$. Note that, $|X''| = |X'| + 4|\mathcal{C}'|$ and $|\mathcal{C}''| = 3|\mathcal{C}'|$.

Suppose that I is a Yes instance of 3-SAT, and let $\alpha\colon X' \to \{true, false\}$ be a truth assignment that satisfies all clauses in $\mathcal{C}'$. Then, from α, we defined a truth assignment $\beta\colon X'' \to \{true, false\}$, as follows: $\beta(x_i) = \alpha(x_i)$ for all $x_i \in X'$, and

- $\beta(y^1_j) = false$, $\beta(y^2_j) = true$, $\beta(y^3_j) = false$ and $\beta(y^4_j) = false$ if $\alpha(z^1_j) = true$, $\alpha(z^2_j) = false$ and $\alpha(z^3_j) = false$;
- $\beta(y^1_j) = false$, $\beta(y^2_j) = false$, $\beta(y^3_j) = false$ and $\beta(y^4_j) = false$ if $\alpha(z^1_j) = false$, $\alpha(z^2_j) = true$ and $\alpha(z^3_j) = false$;
- $\beta(y^1_j) = false$, $\beta(y^2_j) = false$, $\beta(y^3_j) = true$ and $\beta(y^4_j) = false$ if $\alpha(z^1_j) = false$, $\alpha(z^2_j) = false$ and $\alpha(z^3_j) = true$;
- $\beta(y^1_j) = true$, $\beta(y^2_j) = false$, $\beta(y^3_j) = false$ and $\beta(y^4_j) = false$ if $\alpha(z^1_j) = true$, $\alpha(z^2_j) = true$ and $\alpha(z^3_j) = false$;
- $\beta(y^1_j) = false$, $\beta(y^2_j) = true$, $\beta(y^3_j) = false$ and $\beta(y^4_j) = true$ if $\alpha(z^1_j) = true$, $\alpha(z^2_j) = false$ and $\alpha(z^3_j) = true$;
- $\beta(y^1_j) = false$, $\beta(y^2_j) = false$, $\beta(y^3_j) = false$ and $\beta(y^4_j) = true$ if $\alpha(z^1_j) = false$, $\alpha(z^2_j) = true$ and $\alpha(z^3_j) = true$;
- $\beta(y^1_j) = true$, $\beta(y^2_j) = false$, $\beta(y^3_j) = false$ and $\beta(y^4_j) = true$ if $\alpha(z^1_j) = true$, $\alpha(z^2_j) = true$ and $\alpha(z^3_j) = true$;

for all $C_j \in \mathcal{C}'$, where $\alpha(z^i_j) = \alpha(x^i_j)$ if $z^i_j = x^i_j$, and $\alpha(z^i_j) = \overline{\alpha(x^i_j)}$ otherwise. It is easy to see that β satisfies all clauses in $\mathcal{C}''$. Therefore, $f(I)$ is a Yes instance of 1-in-3-SAT.

Conversely, suppose that $f(I)$ is a Yes instance of 1-in-3-SAT, and let $\beta\colon X'' \to \{true, false\}$ be a truth assignment that satisfies all clauses in $\mathcal{C}''$. We let $\alpha\colon X' \to \{true, false\}$ be the truth assignment such that, for each $x_i \in X'$, $\alpha(x_i) = \beta(x_i)$. For the sake of contradiction, suppose that there exists a clause $C_j = \{z^1_j, z^2_j, z^3_j\} \in \mathcal{C}'$ that is false under α, i.e. $\alpha(z^1_j) = false$, $\alpha(z^2_j) = false$ and $\alpha(z^3_j) = false$, where $z^i_j \in \{x^i_j, \overline{x}^i_j\}$ for some variable $x^i_j \in X'$. Then, $\beta(y^1_j) = false$ and $\beta(y^2_j) = false$, otherwise

C_j^1 would have more than one true literal under β. Consequently, $\beta(y_j^3) = \mathit{true}$, otherwise C_j^2 would have no true literal under β. However, this contradicts the hypothesis that C_j^3 has exactly one true literal under β, since in this case $\overline{z}_j^3$ and y_j^3 are both true literals under β. As a result, we obtain that α is indeed a truth assignment that satisfies all clauses in $\mathcal{C}'$. Therefore I is Yes instance of 3-SAT.

Polynomial-time reduction from 1-in-3-SAT to 1-in-3-SAT(3).

Now, let $I = (X'', \mathcal{C}'')$ be an instance of 1-in-3-SAT. For a given variable $x_i \in X''$, let γ_i denote the number of positive occurrences of x_i in I, and let $\overline{\gamma}_i$ denote the number of negative occurrences of x_i in I. We construct from I an instance $f(I) = (X, \mathcal{C})$ of 1-in-3-SAT(3), as follows:

- for each $x_i \in X''$ such that $\gamma_i \geq 2$ and each $j \in \{2, \ldots, \gamma_i\}$, we create the variable μ_i^j and replace the j-th positive occurrence of x_i with the positive literal μ_i^j; moreover, if $\overline{\gamma}_i \geq 1$, then we create the variable $\mu_i^{\gamma_i+1}$;
- for each $x_i \in X''$ such that $\overline{\gamma}_i \geq 1$ and each $j \in \{1, \ldots, \overline{\gamma}_i\}$, we create the variable ν_i^j and replace the j-th negative occurrence of x_i with the positive literal ν_i^j; moreover, we create the variable $\nu_i^{\overline{\gamma}_i+1}$;
- for each $x_i \in X''$ such that $\gamma_i = 0$ and $\overline{\gamma}_i \geq 1$, we create the clauses $\{x_i, \overline{x}_i\}$, $\{x_i, \nu_i^1\}$, $\{\overline{\nu}_i^1, \nu_i^2\}$, …, $\{\overline{\nu}_i^{\overline{\gamma}_i-1}, \nu_i^{\overline{\gamma}_i}\}$, $\{\overline{\nu}_i^{\overline{\gamma}_i}, \nu_i^{\overline{\gamma}_i+1}\}$, $\{\nu_i^{\overline{\gamma}_i+1}, \overline{\nu}_i^{\overline{\gamma}_i+1}\}$;
- for each $x_i \in X''$ such that $\gamma_i = 1$ and $\overline{\gamma}_i \geq 1$, we create the clauses $\{x_i, \nu_i^1\}$, $\{\overline{\nu}_i^1, \nu_i^2\}$, …, $\{\overline{\nu}_i^{\overline{\gamma}_i-1}, \nu_i^{\overline{\gamma}_i}\}$, $\{\overline{\nu}_i^{\overline{\gamma}_i}, \overline{x}_i\}$;
- for each $x_i \in X''$ such that $\gamma_i \geq 2$ and $\overline{\gamma}_i = 0$, we create the clauses $\{\overline{x}_i, \mu_i^2\}$, $\{\overline{\mu}_i^2, \mu_i^3\}$, …, $\{\overline{\mu}_i^{\gamma_i-1}, \mu_i^{\gamma_i}\}$, $\{\overline{\mu}_i^{\gamma_i}, x_i\}$;
- finally, for each $x_i \in X''$ such that $\gamma_i \geq 2$ and $\overline{\gamma}_i \geq 1$, we create the clauses $\{\overline{x}_i, \mu_i^2\}$, $\{\overline{\mu}_i^2, \mu_i^3\}$, …, $\{\overline{\mu}_i^{\gamma_i}, \mu_i^{\gamma_i+1}\}$, $\{\overline{\mu}_i^{\gamma_i+1}, x_i\}$, $\{\mu_i^{\gamma_i+1}, \nu_i^1\}$, $\{\overline{\nu}_i^1, \nu_i^2\}$, …, $\{\overline{\nu}_i^{\overline{\gamma}_i-1}, \nu_i^{\overline{\gamma}_i}\}$, $\{\overline{\nu}_i^{\overline{\gamma}_i}, \nu_i^{\overline{\gamma}_i+1}\}$, $\{\nu_i^{\overline{\gamma}_i+1}, \overline{\nu}_i^{\overline{\gamma}_i+1}\}$.

It is easy to see that every variable in X appears exactly twice positive and once negative in $f(I)$. Additionally, note that $|X| \leq 2|X''| + \sum_{x_i \in X''}(\gamma_i + \overline{\gamma}_i) \leq 2|X''| + 3|\mathcal{C}''|$ and $|\mathcal{C}| \leq |\mathcal{C}''| + 3|X''| + \sum_{x_i \in X''}(\gamma_i + \overline{\gamma}_i) \leq 4|\mathcal{C}''| + 3|X''|$.

Suppose that I is a Yes instance of 1-in-3-SAT, and let $\alpha\colon X'' \to \{\mathit{true}, \mathit{false}\}$ be a truth assignment that satisfies all clauses in $\mathcal{C}''$. From α, we defined a truth assignment $\beta\colon X \to \{\mathit{true}, \mathit{false}\}$, as follows: $\beta(x_i) = \alpha(x_i)$ for all $x_i \in X''$; $\beta(\mu_i^j) = \alpha(x_i)$ for all $j \in \{2, \ldots, \gamma_i\} \cup \{\gamma_i + 1 \mid \gamma_i \geq 2, \overline{\gamma}_i \geq 1\}$ and all $x_i \in X''$; and $\beta(\nu_i^j) = \overline{\alpha(x_i)}$ for all $j \in \{1, \ldots, \overline{\gamma}_i\} \cup \{\overline{\gamma}_i + 1 \mid \overline{\gamma}_i \geq 1\}$ and all $x_i \in X''$. One may verify that β satisfies all clauses in $\mathcal{C}''$. Therefore, $f(I)$ is a Yes instance of 1-in-3-SAT(3).

Conversely, suppose that $f(I)$ is a Yes instance of 1-in-3-SAT(3), and let $\beta\colon X \to \{\mathit{true}, \mathit{false}\}$ be a truth assignment that satisfies all clauses in $\mathcal{C}$. Note that, the truth assignment $\alpha\colon X'' \to \{\mathit{true}, \mathit{false}\}$, where $\alpha(x_i) = \beta(x_i)$ for all $x_i \in X''$, satisfies all clauses in $\mathcal{C}''$. Therefore I is Yes instance of 1-in-3-SAT.

To conclude this proof, note that, based on the above reductions, the existence of a $2^{o(|X|+|\mathcal{C}|)}$-time algorithm for 1-in-3-SAT(3) implies the existence of a $2^{o(|X'|+|\mathcal{C}'|)}$-time algorithm for 3-SAT. Thus, unless ETH fails, 1-in-3-SAT(3) cannot be solved in time $2^{o(|X|+|\mathcal{C}|)}$. □

Theorem 2. *S-TCP remains NP-complete and, unless ETH fails, cannot be solved in time $2^{o(\ell+n)}$ even if $\Delta = 3$.*

Proof. Let $I = (X, \mathcal{C})$ be an instance of 1-in-3-SAT(3), where $X = \{x_1, x_2, \ldots, x_p\}$ and $\mathcal{C} = \{C_1, C_2, \ldots, C_q\}$. We construct from I an instance $f(I) = (G, W, \ell, r)$ of S-TCP, such that $\Delta = 3$, as follows (see Fig. 5):

- first, we create the vertices v_I and w_I and add the edge $v_I w_I$;
- for each variable $x_i \in X$, we create the gadget G_{x_i} such that
  - $V(G_{x_i}) = \{v_{x_i}^2, u_{x_i}^1, u_{x_i}^2, t_{x_i}^1, t_{x_i}^2, f_{x_i}, w_{x_i}\}$ and
  - $E(G_{x_i}) = \{v_{x_i}^2 u_{x_i}^1, u_{x_i}^1 t_{x_i}^1, t_{x_i}^1 t_{x_i}^2, t_{x_i}^2 w_{x_i}, w_{x_i} f_{x_i}, f_{x_i} u_{x_i}^2, u_{x_i}^2 v_{x_i}^2\}$;

  we also create the vertices w'_{x_i} and $v_{x_i}^1$ and add the edges $w'_{x_i} v_{x_i}^1$ and $v_{x_i}^1 v_{x_i}^2$;
- we create a complete strict binary tree T_I, rooted at v_I, whose leaves are the vertices $v_{x_1}^1, v_{x_2}^1, \ldots, v_{x_p}^1$;
- for each clause $C_\iota \in \mathcal{C}$, we create the vertex w_{C_ι}; moreover, we add the edge $t_{x_i}^j w_{C_\iota}$ if one of the literals in C_ι corresponds to the j-th occurrence in I of the positive literal x_i, for $x_i \in X$ and $j \in \{1, 2\}$; on the other hand, we add the edge $f_{x_i} w_{C_\iota}$ if one of the literals in C_ι corresponds to the (only) occurrence in I of the negative literal $\overline{x}_i$;
- we define $W = \{w_I\} \cup \{w_{x_i}, w'_{x_i} \mid x_i \in X\} \cup \{w_{C_\iota} \mid C_\iota \in \mathcal{C}\}$;
- finally, we define $\ell = 2p$ and $r = 2p + |V(T_I)| = 4p - 1$.

One may verify that the maximum degree of G is 3. Furthermore, note that, G has $n = 10p + q$ vertices.

Fig. 5 exemplifies the graph G and the terminal set W of $f(I)$.

Now, we prove that I is a Yes instance of 1-in-3-SAT(3) if and only if $f(I)$ is a Yes instance of S-TCP.

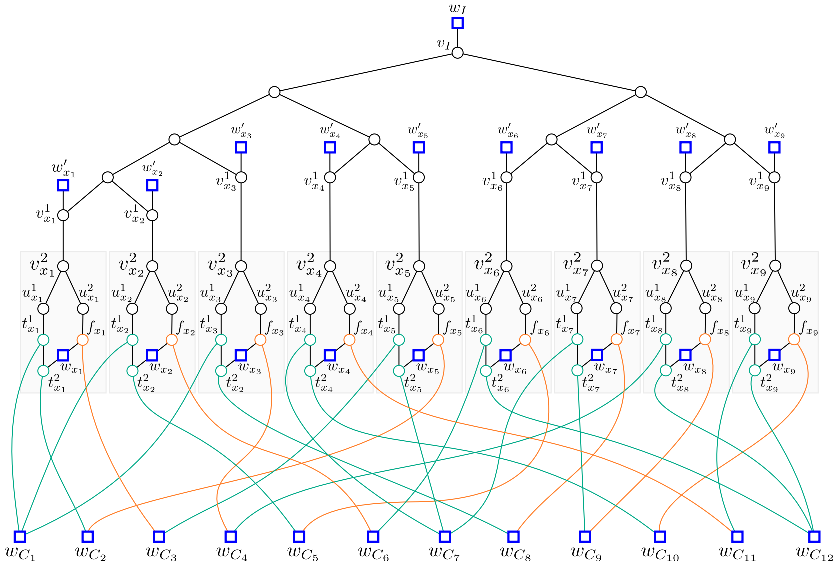

w_I
v_I
w'_{x_1} w'_{x_2} w'_{x_3} w'_{x_4} w'_{x_5} w'_{x_6} w'_{x_7} w'_{x_8} w'_{x_9}
$v^1_{x_1}$ $v^1_{x_2}$ $v^1_{x_3}$ $v^1_{x_4}$ $v^1_{x_5}$ $v^1_{x_6}$ $v^1_{x_7}$ $v^1_{x_8}$ $v^1_{x_9}$
$v^2_{x_1}$ $v^2_{x_2}$ $v^2_{x_3}$ $v^2_{x_4}$ $v^2_{x_5}$ $v^2_{x_6}$ $v^2_{x_7}$ $v^2_{x_8}$ $v^2_{x_9}$
$u^1_{x_1}$ $u^2_{x_1}$ $u^1_{x_2}$ $u^2_{x_2}$ $u^1_{x_3}$ $u^2_{x_3}$ $u^1_{x_4}$ $u^2_{x_4}$ $u^1_{x_5}$ $u^2_{x_5}$ $u^1_{x_6}$ $u^2_{x_6}$ $u^1_{x_7}$ $u^2_{x_7}$ $u^1_{x_8}$ $u^2_{x_8}$ $u^1_{x_9}$ $u^2_{x_9}$
$t^1_{x_1}$ f_{x_1} $t^1_{x_2}$ f_{x_2} $t^1_{x_3}$ f_{x_3} $t^1_{x_4}$ f_{x_4} $t^1_{x_5}$ f_{x_5} $t^1_{x_6}$ f_{x_6} $t^1_{x_7}$ f_{x_7} $t^1_{x_8}$ f_{x_8} $t^1_{x_9}$ f_{x_9}
w_{x_1} w_{x_2} w_{x_3} w_{x_4} w_{x_5} w_{x_6} w_{x_7} w_{x_8} w_{x_9}
$t^2_{x_1}$ $t^2_{x_2}$ $t^2_{x_4}$ $t^2_{x_5}$ $t^2_{x_6}$ $t^2_{x_7}$ $t^2_{x_8}$ $t^2_{x_9}$
w_{C_1} w_{C_2} w_{C_3} w_{C_4} w_{C_5} w_{C_6} w_{C_7} w_{C_8} w_{C_9} $w_{C_{10}}$ $w_{C_{11}}$ $w_{C_{12}}$

Fig. 5. Graph G and terminal set W (blue square vertices) of $f(I)$ obtained from the instance $I = (X, \mathcal{C})$ of 1-IN-3-SAT(3), where $X = \{x_1, x_2, \ldots, x_9\}$ and $\mathcal{C} = \{C_1 = \{x_1, x_2, x_3\}, C_2 = \{x_1, \overline{x}_5\}, C_3 = \{\overline{x}_1, x_5\}, C_4 = \{\overline{x}_3, x_8\}, C_5 = \{x_2, \overline{x}_6\}, C_6 = \{\overline{x}_2, x_6\}, C_7 = \{x_4, x_5, x_7\}, C_8 = \{x_3, \overline{x}_7\}, C_9 = \{x_7, \overline{x}_8\}, C_{10} = \{x_4, \overline{x}_9\}, C_{11} = \{\overline{x}_4, x_9\}, C_{12} = \{x_6, x_8, x_9\}\}$.

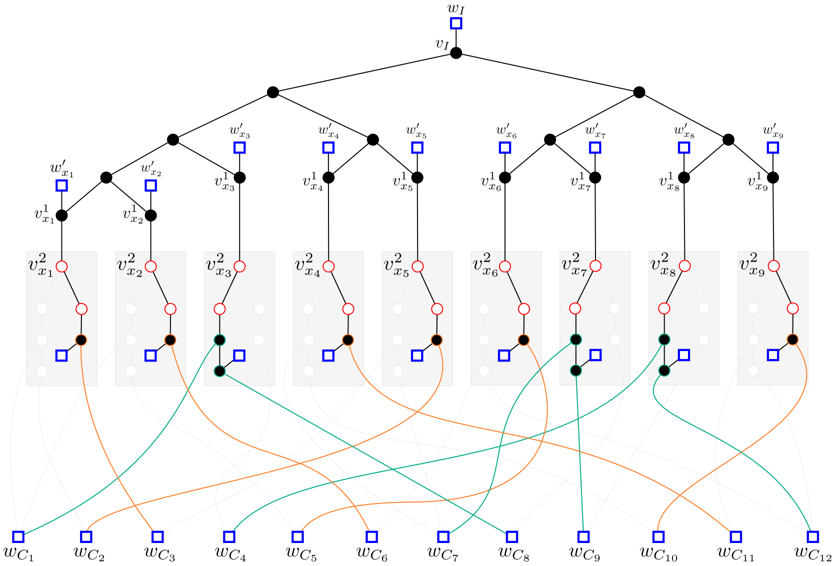

w_I
v_I
w'_{x_1} w'_{x_2} w'_{x_3} w'_{x_4} w'_{x_5} w'_{x_6} w'_{x_7} w'_{x_8} w'_{x_9}
$v^1_{x_1}$ $v^1_{x_2}$ $v^1_{x_3}$ $v^1_{x_4}$ $v^1_{x_5}$ $v^1_{x_6}$ $v^1_{x_7}$ $v^1_{x_8}$ $v^1_{x_9}$
$v^2_{x_1}$ $v^2_{x_2}$ $v^2_{x_3}$ $v^2_{x_4}$ $v^2_{x_5}$ $v^2_{x_6}$ $v^2_{x_7}$ $v^2_{x_8}$ $v^2_{x_9}$
w_{C_1} w_{C_2} w_{C_3} w_{C_4} w_{C_5} w_{C_6} w_{C_7} w_{C_8} w_{C_9} $w_{C_{10}}$ $w_{C_{11}}$ $w_{C_{12}}$

Fig. 6. Strict connection tree of G for W obtained from the truth assignment $\alpha(x_1) = \mathit{false}$, $\alpha(x_2) = \mathit{false}$, $\alpha(x_3) = \mathit{true}$, $\alpha(x_4) = \mathit{false}$, $\alpha(x_5) = \mathit{false}$, $\alpha(x_6) = \mathit{false}$, $\alpha(x_7) = \mathit{true}$, $\alpha(x_8) = \mathit{true}$ e $\alpha(x_9) = \mathit{false}$. (For simplicity, some vertex labels are omitted.)

First, suppose that I is a YES instance of 1-IN-3-SAT(3). Hence, there exists a truth assignment $\alpha\colon X \to \{\mathit{true}, \mathit{false}\}$ that satisfies all clauses in $\mathcal{C}$. Based on α, we construct a strict connection tree T of G for W as follows:

- we add the complete strict binary tree T_I to T along with the terminal vertex w'_{x_i} and the edges $w'_{x_i}v^1_{x_i}$ for every $x_i \in X$; we also add the vertex w_I and the edge $w_I v_I$ to T;
- for each variable $x_i \in X$, we add the vertices $v^2_{x_i}$ and w_{x_i} and the edge $v^1_{x_i}v^2_{x_i}$ to T; moreover, if $\alpha(x_i) = \mathit{true}$, then we add the vertices $u^1_{x_i}$, $t^1_{x_i}$ and $t^2_{x_i}$ to T along with all of their neighbors and incident edges in G; on the other hand, if $\alpha(x_i) = \mathit{false}$, then we add the vertices $u^2_{x_i}$ and f_{x_i} to T along with all of their neighbors and incident edges in G.

Fig. 6 exemplifies the strict connection tree T of G for W, referring to the instance $f(I)$ described in Fig. 5, obtained from a truth assignment α.

Clearly, for every clause $C_\iota \in \mathcal{C}$, $w_{C_\iota} \in V(T)$. Furthermore, observe that these terminals are necessarily leaves of T, since by hypothesis every clause $C_\iota \in \mathcal{C}$ has exactly one true literal under the truth assignment α. Thus, it is easy to see that T is a strict connection tree of G for W. Additionally, note that $\mathsf{L}(T) = \{v^2_{x_i} \mid x_i \in X\} \cup \{u^1_{x_i} \mid \alpha(x_i) = \mathit{true}, x_i \in X\} \cup \{u^2_{x_i} \mid \alpha(x_i) = \mathit{false}, x_i \in X\}$ and $\mathsf{R}(T) = V(T_I) \cup \{t^1_{x_i}, t^2_{x_i} \mid \alpha(x_i) = \mathit{true}, x_i \in X\} \cup \{f_{x_i} \mid \alpha(x_i) = \mathit{false}, x_i \in X\}$. Consequently, $|\mathsf{L}(T)| = 2p = \ell$ and $|\mathsf{R}(T)| \leq 2p + |V(T_I)| = r$. Therefore, $f(I)$ is a Yes instance of S-TCP.

Conversely, suppose that $f(I)$ is a Yes instance of S-TCP. Hence, there exists a strict connection tree T of G for W such that $|\mathsf{L}(T)| \leq \ell$ and $|\mathsf{R}(T)| \leq r$. Note that, for every variable $x_i \in X$, the path in T between the terminals w_{x_i} and w'_{x_i}, say P_{x_i}, necessarily consists in one of the following possibilities: either $P_{x_i} = \langle w_{x_i}, t^2_{x_i}, t^1_{x_i}, u^1_{x_i}, v^2_{x_i}, v^1_{x_i}, w'_{x_i} \rangle$ or $P_{x_i} = \langle w_{x_i}, f_{x_i}, u^2_{x_i}, v^2_{x_i}, v^1_{x_i}, w'_{x_i} \rangle$. Hence, $f_{x_i} \in V(T)$ if and only if $t^1_{x_i} \notin V(T)$. Indeed, if f_{x_i} and $t^1_{x_i}$ simultaneously belonged to $V(T)$, then $u_{x^1_i}$, $u_{x^2_i}$, and at least one of the vertices f_{x_i}, $t^1_{x_i}$ or $t^2_{x_i}$ would be linkers of T, which would imply $|\mathsf{L}(T)| > 2p = \ell$. Thus, we can define a truth assignment $\alpha \colon X \to \{\mathit{true}, \mathit{false}\}$ in the following way: $\alpha(x_i) = \mathit{true}$ if $f_{x_i} \notin V(T)$, and $\alpha(x_i) = \mathit{false}$ otherwise.

Let $W_{\mathcal{C}} = \{w_{C_\iota} \mid C_\iota \in \mathcal{C}\}$. Since $W_{\mathcal{C}} \subset W \subseteq V(T)$ and every path in T between the terminal $w_{C_\iota} \in W_{\mathcal{C}}$ and any other terminal $w \in W$ must contain one of the edges $w_{C_\iota} t^1_{x_i}$, $w_{C_\iota} t^2_{x_i}$ or $w_{C_\iota} f_{x_i}$, for some $x_i \in X$, all clauses in $\mathcal{C}$ have at least one true literal under α. Indeed, for every clause $C_\iota \in \mathcal{C}$, if $w_{C_\iota} t^1_{x_i} \in E(T)$ or $w_{C_\iota} t^2_{x_i} \in E(T)$, then $f_{x_i} \notin V(T)$, and so the assignment $\alpha(x_i) = \mathit{true}$ satisfies C_ι; on the other hand, if $w_{C_\iota} f_{x_i} \in E(T)$, then obviously $f_{x_i} \in V(T)$, and so the assignment $\alpha(x_i) = \mathit{false}$ satisfies C_ι. Furthermore, observe that, each clause in $\mathcal{C}$ has no more than one true literal under α, since by hypothesis all terminals belonging to $W \supset W_{C_\iota}$ are leaves of T and, for every vertex $\rho_{x_i} \in \{t^1_{x_i}, t^2_{x_i}, f_{x_i} \mid x_i \in X\} \cap V(T)$, we have that $N_T(\rho_{x_i}) = N_G(\rho_{x_i})$ — otherwise, $|\mathsf{L}(T)| > 2p = \ell$ or some non-terminal vertex would be a leaf of T. Therefore, I is a Yes instance of 1-in-3-SAT(3).

To conclude this proof, note that, since $n = 10p + q$ and $\ell = 2p$, the existence of a $2^{o(\ell+n)}$-time algorithm for S-TCP, even if $\Delta = 3$, implies the existence of a $2^{o(p+q)}$-time algorithm for 1-in-3-SAT(3). Therefore, based on Proposition 1, unless ETH fails, S-TCP cannot be solved in time $2^{o(\ell+n)}$ even if $\Delta = 3$. □

2.2. *Tractable case: maximum degree* 3 *and bounded number of linkers*

Motivated by Theorems 1 and 2, which state that S-TCP remains NP-complete when $\Delta = 4$ — even if ℓ is bounded by a constant — and when $\Delta = 3$, respectively, we now prove that, if ℓ is bounded by a constant and $\Delta = 3$, then S-TCP is polynomial-time solvable. More specifically, we show that S-TCP can be solved in time $2^{\mathcal{O}(\ell \log n)}$ when $\Delta = 3$. The following proposition and the following lemmas provide the basis of this result.

Let G be a graph, and let $W \subseteq V$ such that $|W| \geq 3$. Given a strict connection tree T of G for W, we denote by $\mathsf{L}'(T)$ the subset of $\mathsf{L}(T)$ defined as follows: $v \in \mathsf{L}'(T)$ if and only if $v \in \mathsf{L}(T)$ and v belongs to a path in T whose (two) endpoints are routers of T; and we denote by T^* the subgraph of T induced by the vertices belonging to $\mathsf{L}'(T) \cup \mathsf{R}(T)$. Observe that, since $|W| \geq 3$, $\mathsf{R}(T) \neq \varnothing$, and thus we have that T^* is well-defined, containing at least one vertex. Furthermore, observe that T^* is a tree.

Proposition 2. *Let $G = (V, E)$ be a graph, and let $W \subseteq V$ such that $|W| \geq 3$. If T is a strict connection tree of G for W, then* $\left\lceil \frac{|W|-2}{\Delta-2} \right\rceil \leq |\mathsf{R}(T)| \leq |W| - 2$.

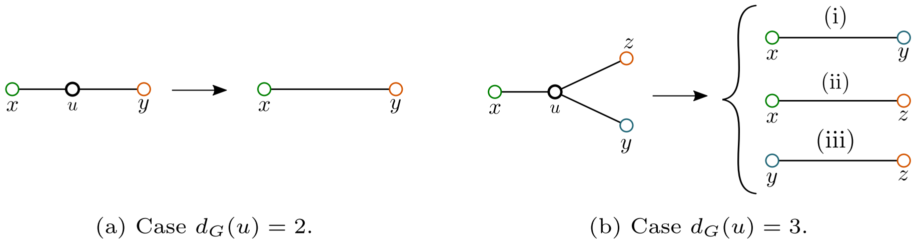
Proof. Note that, $|W| = \sum_{v \in \mathsf{R}(T)} (d_T(v) - d_{T^*}(v))$. Furthermore, we have that

$$\begin{aligned}\sum_{v \in \mathsf{R}(T)} d_{T^*}(v) &= \sum_{v \in V(T^*) \setminus \mathsf{L}'(T)} d_{T^*}(v) = \sum_{v \in V(T^*)} d_{T^*}(v) - \sum_{v \in \mathsf{L}'(T)} d_{T^*}(v) \\ &= 2|E(T^*)| - 2|\mathsf{L}'(T)| = 2\left(|\mathsf{L}'(T)| + |\mathsf{R}(T)| - 1\right) - 2|\mathsf{L}'(T)| \\ &= 2|\mathsf{R}(T)| - 2.\end{aligned}$$

Thus, $|W| = \sum_{v \in \mathsf{R}(T)} d_T(v) - 2|\mathsf{R}(T)| + 2$. Finally, observe that $3|\mathsf{R}(T)| \leq \sum_{v \in \mathsf{R}(T)} d_T(v) \leq \Delta|\mathsf{R}(T)|$. Therefore, $\left\lceil \frac{|W|-2}{\Delta-2} \right\rceil \leq |\mathsf{R}(T)| \leq |W| - 2$. □

By Proposition 2, we can assume without loss of generality that $r = |W| - 2$ whenever $\Delta = 3$. Thus, in the remainder of this section, we omit the input aspect r in the description of the instances of S-TCP. Furthermore, we also assume that G does not contain edges whose endpoints are both terminals, i.e. W is an independent set of G.

Lemma 1. *Let $G = (V, E)$ be a graph such that $\Delta = 3$. Then, G admits a strict connection tree T for W with $\mathsf{L}(T) = \varnothing$ if and only if there exists a subset $V' \subseteq V$ such that $T' = G[V']$ is a strict connection tree for W with $\mathsf{L}(T') = \varnothing$.*



(a) Case $d_G(u) = 2$.

(b) Case $d_G(u) = 3$.

Fig. 7. Operations corresponding to the possible configurations of $u \in L$ being a linker in a strict connection tree of G for W.

Proof. First, suppose that G admits such a tree T. Since $\mathsf{L}(T) = \varnothing$, all non-terminal vertices of T are routers, i.e. $V(T) \setminus W = \mathsf{R}(T)$. Thus, if $\rho \in V(T) \setminus W$, then $N_T(\rho) = N_G(\rho)$, otherwise $\rho \notin \mathsf{R}(T)$ since $\Delta = 3$. Finally, since W is an independent set of G, we have $G[V'] = T$, where $V' = V(T)$.

Conversely, it is immediate that if there exists a subset V' such that $T' = G[V']$ is a strict connection tree for W with $\mathsf{L}(T') = \varnothing$, then G admits a strict connection tree T for W with $\mathsf{L}(T) = \varnothing$. Indeed, T' is itself such a tree T. □

Given a terminal vertex $w \in W$ and a non-terminal vertex $v \in N_G(w)$, we denote by $\mathcal{H}_v$ the subgraph of G induced by the vertices belonging to $V(\mathcal{H}'_v) \cup W$, where $\mathcal{H}'_v$ is the component of $G - W$ that contains v.

Lemma 2. *Let $G = (V, E)$ be a graph such that $\Delta = 3$, and let $w \in W$ be an arbitrary terminal. Then, there exists a subset $V' \subseteq V$ such that $T' = G[V']$ is a strict connection tree for W with $\mathsf{L}(T') = \varnothing$ if and only if, for some non-terminal vertex $v \in N_G(w)$, $\mathcal{H}_v$ is a strict connection tree for W with $\mathsf{L}(\mathcal{H}_v) = \varnothing$.*

Proof. First, suppose that there exists such a subset $V' \subseteq V$. Let $v \in N_{T'}(w)$ be the only neighbor of w in T', where $T' = G[V']$. Since by hypothesis all terminals in W are leaves of T', we have that $T' - W = G[V' \setminus W]$ is connected. Consequently, all non-terminal vertices $\rho \in V' \setminus W$ (including v) belong to a same component of $G - W$. Thus, $V' \setminus W \subseteq V(\mathcal{H}'_v)$. On the other hand, since $\Delta = 3$ and $|\mathsf{L}(T')| = \varnothing$, $N_{T'}(\rho) = N_G(\rho)$ for every $\rho \in V' \setminus W$. Consequently, we have that $V' \setminus W \supseteq V(\mathcal{H}'_v)$. Thus, $V' = V(\mathcal{H}'_v) \cup W$ and, therefore, $\mathcal{H}_v$ is a strict connection tree for W such that $\mathsf{L}(\mathcal{H}_v) = \varnothing$.

Conversely, suppose that, for some non-terminal vertex $v \in N_G(w)$, $\mathcal{H}_v$ is a strict connection tree for W with $\mathsf{L}(\mathcal{H}_v) = \varnothing$. Therefore, for $V' = V(\mathcal{H}_v)$, we have that $T' = G[V']$ is a strict connection tree for W such that $\mathsf{L}(T') = \varnothing$. □

Corollary 1. *S-TCP is linear-time solvable when $\ell = 0$ and $\Delta = 3$.*

Proof. Let $I = (G, W)$ be a given instance of S-TCP with $\ell = 0$ and $\Delta = 3$. Let $w \in W$ be an arbitrary terminal vertex. It is easy to see that, for every non-terminal vertex $v \in N_G(w)$, the graph $\mathcal{H}_v$ can be constructed in time linear in the size of I; for instance, we can obtain $\mathcal{H}_v$ by running the variant of the breadth-first search rooted at v on which the terminal vertices are not explored (i.e. they must be leaves in the resulting search tree). Moreover, we can also verify in linear-time whether $\mathcal{H}_v$ is a strict connection tree for W such that $\mathsf{L}(\mathcal{H}_v) = \varnothing$. Therefore, it follows from Lemmas 1 and 2 that S-TCP is linear-time solvable if $\Delta = 3$ and $\ell = 0$. □

Theorem 3. *S-TCP can be solved in time $2^{\mathcal{O}(\ell \log n)}$ when $\Delta = 3$, but assuming ETH there is no $2^{o(\ell+n)}$-time algorithm for the problem.*

Proof. We first prove that S-TCP can be solved in time $2^{\mathcal{O}(\ell \log n)}$ when $\Delta = 3$. Let $I = (G, W, \ell)$ be an instance of S-TCP such that $\Delta = 3$. For each subset $L \subseteq V \setminus W$ such that $|L| \leq \ell$, we generate all combinations of graphs G'_L obtained from G by successively applying, for each vertex $u \in L$, the operation depicted in Fig. 7a if $d_G(u) = 2$, or one of the three operations depicted in Fig. 7b if $d_G(u) = 3$. These operations simulate the possible configurations of the vertex $u \in L$ being a linker in a strict connection tree of G for W. In the end, after all vertices belonging to L have been processed as described above, we verify whether the resulting graphs G'_L admit a strict connection tree T' for W such that $\mathsf{L}(T') = \varnothing$. Then, the algorithm returns that I is a Yes instance of S-TCP if and only, for some subset $L \subseteq V \setminus W$, with $|L| \leq \ell$, there exists a graph G'_L that admits such a tree T'. Algorithm 1 presents this Turing reduction formally, where the function Get-tree-without-linkers, with input (G, W), denotes a linear-time procedure for solving S-TCP with $\ell = 0$, when $\Delta = 3$, that returns either a strict connection tree T of G for W such that $\mathsf{L}(T) = \varnothing$ or *null* if such a tree does not exist.

The correctness of the algorithm follows from the fact that all possible relevant configurations for the existence of a strict connection tree of G for W with at most ℓ linkers are analyzed. In fact, if there exists a graph G'_L that admits a strict connection tree T'_L for W such that $\mathsf{L}(T'_L) = \varnothing$, for some subset $L \subseteq V \setminus W$, with $|L| \leq \ell$, then by construction G admits a strict connection tree T for W such that $\mathsf{L}(T) \subseteq L$, and thus we have that $I = (G, W, \ell)$ is a Yes instance of S-TCP. Conversely, if G admits a strict connection tree T for W such that $|\mathsf{L}(T)| \leq \ell$, then clearly, for $L = \mathsf{L}(T)$, there exists a graph G'_L that admits a strict connection tree T'_L for W such that $\mathsf{L}(T'_L) = \varnothing$, and thus we have that $I' = (G'_L, W)$ is a Yes instance of S-TCP with $\ell = 0$.

Algorithm 1: Turing reduction from S-TCP to S-TCP with $\ell = 0$, restricted to graphs with maximum degree 3.

```
Input: A graph $G = (V, E)$ such that $\Delta = 3$, a terminal set $W \subseteq V$ and a non-negative integer $\ell$.
Output: A strict connection tree $T$ of $G$ for $W$ such that $|\mathsf{L}(T)| \leq \ell$ and $|\mathsf{R}(T)| \leq r$, or null if such a tree does not exist.
Let $\mathcal{L}$ be a collection of all subsets $L \subseteq V \setminus W$, with $|L| \leq \ell$, such that the sets in $\mathcal{L}$ are ordered according to their inclusion order.
foreach $L \in \mathcal{L}$ do
    $T$ := Get-tree-given-linker-superset($G, W, L, 1$)
    if $T \neq$ null then return $T$
return null
Function Get-tree-given-linker-superset($G, W, L, i$)
    if $i > |L|$ then return Get-tree-without-linkers($G, W$)
    $u := L[i]$                                          // $L[i]$ denotes the $i$-th element in $L$
    if $d_G(u) = 1$ then return null
    foreach $x, y \in N_G(u)$ such that $x \neq y$ do
        Let $G'_L$ be the graph defined as follows: $V(G'_L) := V \setminus \{u\}$ and $E(G'_L) := (E \setminus \{uv \mid v \in N_G(u)\}) \cup \{xy\}$
        $T$ := Get-tree-given-linker-superset($G'_L, W, L, i + 1$)
        if $T \neq$ null then
            $E(T) := (E(T) \setminus \{xy\}) \cup \{ux, uy\}$
            return $T$
    return null
```

Regarding the running time of the algorithm, in the worst case the number of recursive calls to Get-tree-given-linker-superset is

$$\sum_{\substack{L \subseteq V \setminus W \\ |L| \leq \ell}} \prod_{u \in L} \binom{d_G(u)}{2} \leq \sum_{\substack{L \subseteq V \setminus W \\ |L| \leq \ell}} 3^{|L|} \leq \sum_{i=0}^{\ell} \binom{n}{i} 3^i = \mathcal{O}\left(3^\ell \cdot n^\ell\right).$$

Thus, the total time spent by the algorithm is $\mathcal{O}\left(3^\ell \cdot n^{\ell+1}\right)$, since Get-tree-without-linkers runs in time linear in n and all the other operations of the algorithm can be performed in constant time. Therefore, S-TCP can be solved in time $2^{\mathcal{O}(\ell \log n)}$ when $\Delta = 3$.

The proof that S-TCP cannot be solved in time $2^{o(\ell+n)}$, even when $\Delta = 3$, unless ETH fails, follows directly from Theorem 2. □

3. Using ℓ, r and Δ as parameters

In the present section, we investigate the parameterized complexity of S-TCP when ℓ, r and Δ are parameters. We remark that, as a result of Theorem 1, S-TCP parameterized by ℓ and Δ is para-NP-complete; consequently, the problem does not even admit an XP-time algorithm, unless P = NP. On the other hand, Dourado et al. [2] showed that S-TCP parameterized by ℓ and r is in XP.

Nevertheless, we now prove that S-TCP parameterized by ℓ and r is W[2]-hard. Particularly, we show that, for every $\ell \geq 0$, S-TCP parameterized by r is W[2]-hard. Thus, unless FPT = W[2], S-TCP does not admit an algorithm with running time $g(r) \cdot n^{h(\ell)}$, for any computable functions g and h.

Theorem 4. *For every $\ell \geq 0$, S-TCP parameterized by r is W[2]-hard.*

Proof. Let $I = (U, \mathcal{F}, k)$ be an instance of Set cover, a classical W[2]-hard problem [14], where U is the universe, $\mathcal{F}$ is the collection of non-empty sets over U, and k is the parameter of the problem, a non-negative integer. We construct an instance $f(I) = (G, W, r)$ of S-TCP with ℓ bounded by a constant as follows:

- for each $i \in \{1, \ldots, \ell\}$, we create the vertices u_i and w'_i, and we add the edge $u_i w'_i$; let $L = \{u_1, u_2, \ldots, u_\ell\}$;
- for each set $F_i \in \mathcal{F}$, we create the vertex v_{F_i}; moreover, for each pair $F_i, F_j \in \mathcal{F}$ with $i \neq j$, we add the edge $v_{F_i} v_{F_j}$; let $K_{\mathcal{F}} = \{v_{F_i} \mid F_i \in \mathcal{F}\}$;
- we create the vertices w_a and w_b and, for each set $F_i \in \mathcal{F}$, we add the edges $w_a v_{F_i}$ and $w_b v_{F_i}$;
- for each pair of vertices u_i, v_{F_j}, where $u_i \in L$ and $v_{F_j} \in K_{\mathcal{F}}$, we add the edge $u_i v_{F_j}$;
- for each element $x_i \in U$, we create the vertex w_i;
- for each set $F_j \in \mathcal{F}$ and each element $x_i \in F_j$, we add the edge $v_{F_j} w_i$;
- finally, we define $W = \{w_a, w_b\} \cup \{w'_i \mid i \in \{1, \ldots, \ell\}\} \cup \{w_i \mid x_i \in U\}$ and $r = k$.

Now we prove that I is a Yes instance of Set cover if and only if $f(I)$ is a Yes instance of S-TCP with ℓ bounded by a constant.

First, suppose that I is a Yes instance of Set cover, and let $\mathcal{F}' = \{F'_1, F'_2, \dots, F'_z\}$ be a subcollection of $\mathcal{F}$ such that $\bigcup_{F' \in \mathcal{F}'} F' = U$ and $z \leq k$. Assume without loss of generality that $\mathcal{F}'$ is minimal with respect to the property of covering all elements of U, i.e. for any set $F' \in \mathcal{F}'$, $\mathcal{F}' \setminus \{F'\}$ does not cover all elements of U. Based on $\mathcal{F}'$, we construct a strict connection tree T of G for W as follows:

- for each $i \in \{1, 2, \dots, \ell\}$, we add the vertices u_i and w'_i to T along with the edges $u_i w'_i$;
- for each set $F'_j \in \mathcal{F}'$, we add the vertex $v_{F'_j}$ to T;
- for each $i \in \{1, 2, \dots, z-1\}$, we add the edge $v_{F'_i} v_{F'_{i+1}}$ to T;
- moreover, for each $i \in \{1, 2, \dots, \ell\}$, we add the edge $u_i v_{F'_1}$ to T; we also add the edges $w_a v_{F'_1}$ and $w_b v_{F'_z}$ to T;
- finally, for each element $x_i \in U$, we add the vertex w_i to T along with the edge $v_{F'_j} w_i$, where $j = \min\{1, \dots, z\}$ such that $x_i \in F'_j$ and $F'_j \in \mathcal{F}'$.

It is easy to verify that T is a strict connection tree of G for W. Now, we prove that $|\mathsf{L}(T)| \leq \ell$ and $|\mathsf{R}(T)| \leq r$. First, note that the vertices $u_1, u_2, \dots, u_\ell$ have degree exactly 2 in T. Thus, $\mathsf{L}(T) \supseteq L$. On the other hand, it follows from the minimality of $\mathcal{F}'$ that, for every set $F'_j \in \mathcal{F}$, the vertex $v_{F'_j}$ is adjacent to at least one terminal $w_i \in W$, since F'_j covers at least one element $x_i \in U$ which is not covered by any other set in $\mathcal{F}'$. Consequently, every vertex $v_{F'_j}$ has degree at least 3 in T, for $F'_j \in \mathcal{F}'$. Thus, $\mathsf{L}(T) = L$ and $\mathsf{R}(T) = \{v_{F'_j} \mid F'_j \in \mathcal{F}'\}$, which implies $|\mathsf{L}(T)| \leq \ell$ and $|\mathsf{R}(T)| = z \leq k = r$. Therefore, $f(I)$ is a Yes instance of S-TCP with ℓ bounded by a constant.

Conversely, suppose that G admits a strict connection tree T for W such that $|\mathsf{L}(T)| \leq \ell$ and $|\mathsf{R}(T)| \leq r = k$. Note that, for every $i \in \{1, 2, \dots, \ell\}$, the path in T between the terminal w'_i and any other terminal belonging to W necessarily contains the vertex $u_i \in L$. Hence, $V(T) \supseteq L$. Furthermore, since the vertices in $W \supset \{w_i \mid x_i \in U\}$ are leaves of T, there is a vertex $v_{F_j} \in K_{\mathcal{F}}$ such that $N_T(w_i) = \{v_{F_j}\}$ for every $x_i \in U$. However, the vertices v_{F_j} are non-terminal and T contains at most $\ell + r$ non-terminal vertices. Thus, since $V(T) \supseteq L$ and $|L| = \ell$, there are at most r vertices belonging to $K_{\mathcal{F}}$ in T, i.e. $|V(T) \cap K_{\mathcal{F}}| \leq r = k$. Then, $\mathcal{F}' = \{F_j \mid v_{F_j} \in V(T) \cap K_{\mathcal{F}}\}$ is a subcollection of $\mathcal{F}$ such that $|\mathcal{F}'| \leq k$. Finally, it follows from the fact that $W \subseteq V(T)$ that $\bigcup_{F' \in \mathcal{F}'} F' = U$. Therefore, I is a Yes instance of Set cover. □

Based on the technique called bounded search tree, Dourado et al. [2] provided an $\mathcal{O}\left((2^{\Delta-1})^{\ell+r}\Delta n\right)$-time algorithm for S-TCP. As an immediate result, they proved that if, besides ℓ and r, the maximum degree Δ of G is also considered as a parameter, then S-TCP is in FPT. We now present an alternative, but substantially simpler, proof for the tractability of S-TCP parameterized by ℓ, r and Δ, which consists in a kernelization algorithm for the problem derived from the following reduction rules.

Reduction rule 1. For any two terminals $w_1, w_2 \in W$, if the distance between them in $G - (W \setminus \{w_1, w_2\})$ is greater than $\ell + r + 1$, then conclude that G does not admit a strict connection for W with at most ℓ linkers and at most r routers.

Reduction rule 2. Let $v \in V \setminus W$ and $w \in W$. If the distance between v and w in $G - (W \setminus \{w\})$ is greater than $\ell + r + 1$, then remove v from G.

Lemma 3. *Reduction rules 1 and 2 are safe.*

Proof. Suppose that there exist $w_1, w_2 \in W$ such that the distance between w_1 and w_2 in $G - (W \setminus \{w_1, w_2\})$ is greater than $\ell + r + 1$. Thus, every strict connection tree of G for $W \supseteq \{w_1, w_2\}$ has more than ℓ linkers or more than r routers. Therefore, we are dealing with a No instance of the problem in this case, and so Reduction rule 1 is indeed safe. Now, suppose that there exists a non-terminal vertex $v \in V \setminus W$ such that, for some terminal $w \in W$, the distance between v and w in $G - (W \setminus \{w\})$ is greater than $\ell + r + 1$. Note that, v does not belong to any strict connection tree T of G for $W \supset \{w\}$, otherwise T would have more than ℓ linkers or more than r routers. Thus, G admits a strict connection tree for W with at most ℓ linkers and at most r routers if and only if $G - v$ admits a strict connection tree for W with at most ℓ linkers and at most r routers. Therefore, we have that Reduction rule 2 is also safe. □

Theorem 5. *S-TCP admits a kernel with $\mathcal{O}\left(\Delta^{2(\ell+r+1)}\right)$ vertices.*

Proof. Based on Reduction rule 1, suppose that, for every pair of terminal vertices $w_1, w_2 \in W$, the distance between them in $G - (W \setminus \{w_1, w_2\})$ is at most $\ell + r + 1$. Moreover, while it is possible, apply Reduction rule 2 successively. Let G' denote the resulting graph. Note that, the distance between any non-terminal vertex $v \in V(G') \setminus W$ and any terminal vertex $w \in W$ in $G' - (W \setminus \{w\})$ is at most $\ell + r + 1$. Moreover, the distance between any two non-terminal vertices $u, v \in V \setminus W$ in G' is at most $2(\ell + r + 1)$. Thus, the diameter of G' is at most $2(\ell + r + 1)$ and, therefore, the number of vertices in G' is $\mathcal{O}\left(\Delta^{2(\ell+r+1)}\right)$. □

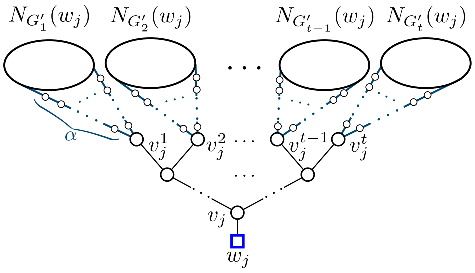


Fig. 8. Complete strict binary tree T_j and paths $P(v_j^i, x)$, where $x \in N_{G'_i}(w_j)$, for $i \in \{1, 2, \dots, t\}$ and $j \in \{1, 2, \dots, k\}$.

At this point, a natural question that arises is whether there exists a polynomial kernel for S-TCP parameterized by ℓ, r and Δ. However, we show that this does not seem to be the case, the existence of such a kernel is unlikely. By using a framework developed by Bodlaender *et al.* [31,32], called *cross-decomposition*, we prove that S-TCP parameterized by ℓ, r and Δ does not admit a polynomial kernel, unless NP $\subseteq$ coNP/poly.

By Proposition 2, if a graph G with maximum degree Δ admits a strict connection tree T for a terminal set W with at most r routers, then $\left\lceil \frac{|W|-2}{\Delta-2} \right\rceil \le r$, which implies $|W| \le r(\Delta - 2) + 2$. Thus, if r and Δ are parameters, then, without loss of generality, $|W|$ can be considered as a parameter as well.

Theorem 6. *S-TCP parameterized by ℓ, r, Δ and $|W|$ does not admit a polynomial kernel, unless NP $\subseteq$ coNP/poly.*

Proof. We prove this theorem by showing that S-TCP cross-composes into S-TCP parameterized by ℓ, r, Δ and $|W|$. We first need to present a polynomial equivalence relation. Thus, let $\mathcal{R}$ be the equivalence relation defined as follows: all bitstrings which do not encode a valid instance of S-TCP belong to a same equivalence class; and two well-formed instances (G_1, W_1, ℓ_1, r_1) and (G_2, W_2, ℓ_2, r_2) of S-TCP belong to a same equivalence class if and only if $|V(G_1)| = |V(G_2)|$, $\Delta(G_1) = \Delta(G_2)$, $|W_1| = |W_2|$, $\ell_1 = \ell_2$ and $r_1 = r_2$. One may verify that any set of well-formed instances of S-TCP on at most n vertices each can be partitioned into $\mathcal{O}\left(n^5\right)$ equivalence classes. Therefore, $\mathcal{R}$ is a polynomial equivalence relation.

Let $I'_1, I'_2, \dots, I'_t$ be $t \ge 1$ input instances which are equivalent under $\mathcal{R}$. If such instances are not well-formed, then we output a single trivial No instance of S-TCP parameterized by ℓ, r, Δ and $|W|$. Thus, assume that all of the input instances $I'_1, I'_2, \dots, I'_t$ are well-formed and encode structures $(G'_1, W'_1, \ell'_1, r'_1)$, $(G'_2, W'_2, \ell'_2, r'_2)$, $\dots$, $(G'_t, W'_t, \ell'_t, r'_t)$, respectively. For simplicity, we also assume without loss of generality that, for every $i \in \{1, 2, \dots, t\}$, $\ell'_i = \ell' \le |V(G'_i)|$, $r'_i = r' \le |V(G'_i)|$, $\Delta(G'_i) = \gamma \ge 3$ and W'_i is an independent set of G'_i such that $W'_i = \{w_1, w_2, \dots, w_k\}$, where $k \ge 3$. Then, we compose $I'_1, I'_2, \dots, I'_t$ into a single instance $I = (G, W, \ell, r)$ of S-TCP parameterized by ℓ, r, Δ and $|W|$, as follows:

- we add the vertices $w_1, w_2, \dots, w_k$ to G;
- for each $j \in \{1, 2, \dots, k\}$, we create the vertices $v_j, v_j^1, v_j^2, \dots, v_j^t$ and a complete strict binary tree T_j, rooted at v_j, whose leaves are the vertices $v_j^1, v_j^2, \dots, v_j^t$; moreover, we add the edge $w_j v_j$ to G;
- for each $i \in \{1, 2, \dots, t\}$, we add all the vertices and all the edges of the graph $G'_i - W'_i$ to G;
- for each $i \in \{1, 2, \dots, t\}$, for each $j \in \{1, 2, \dots, k\}$ and for each vertex $x \in N_{G'_i}(w_j)$, we add the edge $v_j^i x$ to G and subdivide this edge into α new vertices, where $\alpha = \ell' + k\lceil \log_2 t \rceil + 1$; in other words, for each vertex $x \in N_{G'_i}(w_j)$, we create the vertices $x_1^*, x_2^*, \dots, x_\alpha^*$ and the path $P(v_j^i, x) = \langle v_j^i, x_1^*, x_2^*, \dots, x_\alpha^*, x \rangle$ (see Fig. 8);
- finally, we define $W = \{w_1, w_2, \dots, w_k\}$, $\ell = (k+1)\alpha - 1$ and $r = r'$.

Fig. 9 illustrates the overall structure of G and W.

One may easily verify that I can be constructed in time polynomial in $\sum_{i=1}^{t} |I'_i|$, and that $\Delta(G) \le \gamma + 1$, $|W| = k$, $\ell = \mathcal{O}\left(k\ell' + k^2 \log_2 t\right)$ and $r = r'$.

We now proof that there exists $i \in \{1, 2, \dots, t\}$ such that I'_i is a Yes instance of S-TCP if and only if I is a Yes instance of S-TCP parameterized by ℓ, r, Δ and $|W|$.

First, suppose that, for some $i \in \{1, 2, \dots, t\}$, I'_i is a Yes instance of S-TCP, and let T' be a strict connection tree of G'_i for W'_i such that $|\mathsf{L}(T')| \le \ell'$ and $|\mathsf{R}(T')| \le r'$. Then, consider the subgraph T of G defined as follows:

$$V(T) = V(T') \cup \bigcup_{w_j \in W} V(P(w_j, v_j^i)) \cup \bigcup_{\substack{x \in N_{T'}(w_j) \\ w_j \in W}} V(P(v_j^i, x)) \text{ and}$$

$$E(T) = \left(E(T') \setminus \{w_j x \mid w_j \in W\}\right) \cup \bigcup_{w_j \in W} E(P(w_j, v_j^i)) \cup \bigcup_{\substack{x \in N_{T'}(w_j) \\ w_j \in W}} E(P(v_j^i, x)),$$

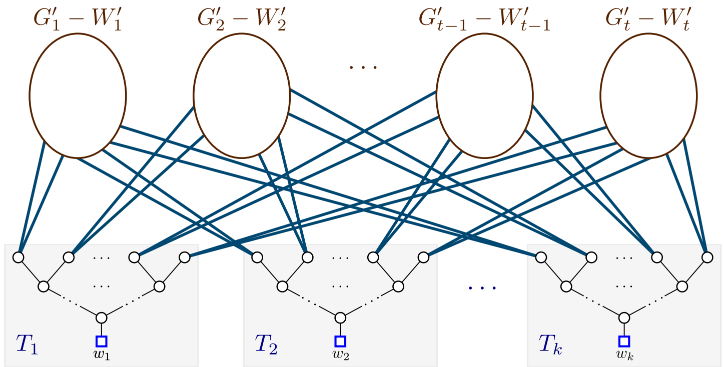


Fig. 9. Graph G and terminal set W. For simplicity, the vertices $x^*_1, x^*_2, \ldots, x^*_\alpha$ corresponding to the subdivision of the edges $v^i_j x$ are omitted, where $x \in N_{G'_i}(w_j)$.

where $P(w_j, v^i_j)$ denotes the only path in T_j between the vertices w_j and v^i_j. It is easy to see that T is a strict connection tree of G for W such that $|\mathsf{L}(T)| \le |\mathsf{L}(T')| + k\lceil \log_2 t\rceil + k\alpha \le (k+1)\alpha - 1 = \ell$ and $|\mathsf{R}(T)| \le r' = r$, since $\mathsf{L}(T) = \mathsf{L}(T') \cup \bigcup_{\substack{x \in N_{T'}(w_j)\\ w_j \in W}} \left(V(P(v^i_j, x)) \setminus \{x\}\right) \cup \bigcup_{w_j \in W} \left(V(P(w_j, v^i_j)) \setminus \{w_j\}\right)$ and $\mathsf{R}(T) = \mathsf{R}(T')$. Therefore, I is a Yes instance of S-TCP.

Conversely, suppose that I is a Yes instance of S-TCP, and let T be a strict connection tree of G for W such that $|\mathsf{L}(T)| \le \ell$ and $|\mathsf{R}(T)| \le r$. Note that, for each $w_j \in W$, all paths in G between w_j and any other terminal must contain a path $P(v^i_j, x)$ as a subpath, for some $x \in N_{G'_i}(w_j)$ and some $i \in \{1, 2, \ldots, t\}$; so, since $W \subseteq V(T)$, for each $w_j \in W$, T contains a path $P(v^i_j, x)$. Thus, $|\mathsf{L}(T)| \ge k\alpha$. Consequently, for every $j \in \{1, 2, \ldots, k\}$, the intersection between T and the complete binary tree T_j can only contain the path $P(w_j, v^i_j)$ of T_j whose endpoints are w_j and v^i_j, otherwise: T would have a leaf which is not a terminal; or, besides $P(v^i_j, x)$, T would have a further path $P(v^\iota_j, y)$, for some $y \in N_{G'_\iota}(w_j)$ and some $\iota \in \{1, 2, \ldots, t\}$ with $\iota \ne i$, which would imply $|\mathsf{L}(T)| \ge (k+1)\alpha > \ell$. By similar reasons, we have that the degree of v^i_j in T must be equal to 2. Hence, for every $j \in \{1, 2, \ldots, k\}$, the subgraph of T_j in T – i.e. the path $P(w_j, v^i_j)$ – can be viewed as a leaf of T (in the sense that its only purpose in T is connecting the terminal w_j, as a leaf of T). Consequently, there exists precisely one index $i \in \{1, 2, \ldots, t\}$ such that $V(T) \cap V(G'_i - W'_i) \ne \varnothing$, otherwise T would be disconnected. Therefore, I'_i is a Yes instance of S-TCP. Indeed, the graph T', where $V(T') = \left(V(T) \cap V(G'_i - W'_i)\right) \cup W'_i$ and $E(T') = \left(E(T) \cap E(G'_i - W'_i)\right) \cup \{w_j x \mid v^i_j x \in E(T)\}$, is a strict connection tree for W'_i such that $|\mathsf{L}(T')| \le \ell'$ and $|\mathsf{R}(T')| \le r'$, since $\mathsf{L}(T') = \mathsf{L}(T) \cap V(G'_i - W'_i)$ and $\mathsf{R}(T') = \mathsf{R}(T)$. □

4. The split graph case

A *split graph* is a graph whose vertex set can be partitioned into a clique and an independent set.

S-TCP on split graphs may have interesting applications in IoT (*Internet of Things*), where devices with high communicating/processing power (such as wireless routers) are modeled as a cluster, while devices with low communicating/processing power (such as wireless printers) are modeled as an independent set, being able to send (receive, resp.) messages just to (from, resp.) devices of the cluster. Thus, motivated by applications in IoT and by the fact that it is well-known that Steiner tree is NP-complete on split graphs [6], we analyze in this section the complexity of S-TCP restricted to split graphs.

More specifically, we prove that S-TCP restricted to split graphs can be solved in time $n^{\mathcal{O}(r)}$, implying thereby that S-TCP on split graphs is polynomial-time solvable when r is bounded by a constant. On the other hand, we extend Theorem 4 by showing that S-TCP parameterized by r remains W[2]-hard even if it is restricted to split graphs and ℓ is bounded by a constant; furthermore, we show that, for any computable function g, there is no $g(r) \cdot n^{o(r)}$-time algorithm for the problem, unless ETH fails.

Given an instance $I = (G, W, \ell, r)$ of S-TCP, where $G = (V, E)$ is a split graph. We assume throughout this section that $V = K \cup S$, where K is a maximal clique and S is a maximal independent set of G. We also assume that $r \ge 1$ and $|W| \ge 3$.

Fact 1. If $K \subseteq W$, then G does not admit a strict connection tree for W.

Fact 2. If $K \setminus W \ne \varnothing$ and $W \cap S = \varnothing$ (i.e. $W \subset K$), then G admits a strict connection tree T for W such that $|\mathsf{L}(T)| = 0$ and $|\mathsf{R}(T)| = 1$.

Lemma 4. *Suppose that $K \setminus W \neq \emptyset$ and $W \cap S \neq \emptyset$. If G admits a strict connection tree T' for W, then there exists a strict connection tree T of G for W such that $|\mathsf{L}(T)| \leq |\mathsf{L}(T')|$, $|\mathsf{R}(T)| \leq |\mathsf{R}(T')|$, and $\mathsf{R}(T) \subseteq K$.*

Proof. Since S is an independent set of G, $N_{T'}(S) \subseteq K$. Moreover, it follows from the assumptions $W \cap S \neq \emptyset$ and $|W| \geq 3$ that $\big(V(T') \setminus W\big) \cap K \neq \emptyset$. Thus, let v be an arbitrary vertex in $\big(V(T') \setminus W\big) \cap K$. Since K is a clique of G, we have that $v \in N_G(v')$ for all $v' \in N_{T'}(S) \setminus \{v\}$. Then, consider the graph T defined as follows: $V(T) = V(T') \setminus \big(\mathsf{R}(T') \cap S\big)$ and $E(T) = \big(E(T') \setminus \{\rho v' \mid v' \in N_{T'}(\rho),\ \rho \in \mathsf{R}(T') \cap S\}\big) \cup \{vv' \mid v' \in N_{T'}(\mathsf{R}(T') \cap S)\}$. One may easily verify that T is a strict connection tree of G for W such that $|\mathsf{L}(T)| \leq |\mathsf{L}(T')|$, $|\mathsf{R}(T)| \leq |\mathsf{R}(T')|$, and $\mathsf{R}(T) \subseteq K$. □

Lemma 5. *Let T' be a strict connection tree of G for W such that $\mathsf{R}(T') \subseteq K$. There exists a strict connection tree T of G for W, with $|\mathsf{L}(T)| \leq |\mathsf{L}(T')|$ and $\mathsf{R}(T) \subseteq \mathsf{R}(T')$, which holds the following properties:*

(i) $\mathsf{L}(T) \subseteq K$*;*
(ii) each vertex in $L(T)$ is adjacent to exactly one vertex in $\mathsf{R}(T)$ and exactly one vertex $w \in W$, where $w \in S$ and $w \notin N_G(\mathsf{R}(T))$;
(iii) $T[R(T)]$ is a path.

Proof. (i). Note that, for every vertex $u \in \mathsf{L}(T') \cap S$, if x_u and y_u are the two distinct neighbors of u in T', then $x_u, y_u \in K$. Thus, the graph obtained from T' by removing all the vertices in $\mathsf{L}(T') \cap S$ and adding all the edges in $\{x_u y_u \mid x_u, y_u \in N_{T'}(u), u \in \mathsf{L}(T') \cap S\}$ is a strict connection tree of G for W with linker set $\mathsf{L}(T') \setminus S \subseteq K$ and router set $\mathsf{R}(T')$. Thus, for simplicity, we assume hereinafter that $\mathsf{L}(T') \subseteq K$.

(ii). Since $|W| \geq 3$, for every vertex $u \in \mathsf{L}(T')$, if x_u and y_u are the two distinct neighbors of u in T', then $x_u \notin W$ or $y_u \notin W$, otherwise T' would not be a strict connection tree for W. If $x_u, y_u \notin W$, then $x_u, y_u \in K$. Hence, we can remove u from T' and add the edge $x_u y_u$. Thus, suppose that $x_u \in W$ and $y_u \notin W$. If $y_u \in \mathsf{L}(T')$, then y_u has exactly one neighbor in T' in addition to u. Let z be this second neighbor of y_u in T'. Since $|W| \geq 3$ and we are supposing that $x_u \in W$, we have $z \notin W$, which implies $z \in K$. As a result, the graph obtained from T' by removing y_u and adding the edge uz is a strict connection tree of G for W with linker set $\mathsf{L}(T') \setminus \{y_u\}$ and router set $\mathsf{R}(T')$. Suppose now that $y_u \in \mathsf{R}(T')$ but there exists a vertex $\rho \in \mathsf{R}(T')$, possibly $\rho = y_u$, such that $\rho x_u \in E(G)$. Consequently, the graph H obtained from T' by removing u and adding the edge ρx_u is a strict connection tree of G for W such that $\mathsf{L}(H) = \mathsf{L}(T') \setminus \{u\}$ and $\mathsf{R}(H) = \mathsf{R}(T')$ if $d_{T'}(y_u) > 3$ or $\rho = y_u$, and $\mathsf{L}(H) = \big(\mathsf{L}(T') \setminus \{u\}\big) \cup \{y_u\}$ and $\mathsf{R}(H) = \mathsf{R}(T') \setminus \{y_u\}$ otherwise. Therefore, one may verify that, by applying successively the steps described above, it is always possible to obtain a strict connection tree of G for W which holds property (ii).

(iii). If $|\mathsf{R}(T')| \leq 1$, then trivially T' holds property (iii). Thus, assume that $|\mathsf{R}(T')| \geq 2$. Additionally, assume that T' holds property (ii). Consequently, $H_R = T[\mathsf{R}(T')]$ is a tree. Note that, H_R contains at least two leaves. Let R^* be the set defined in the following way: $\rho^* \in R^*$ if and only if $\rho^* \in \mathsf{R}(T')$ and there is at least one terminal vertex $w \in W$ such that $\mathrm{dist}_{T'}(w, \rho^*) = \mathrm{dist}_{T'}(w, \mathsf{R}(T'))$, i.e. the path between w and ρ^* in T' does not contain any other router. Note that, every leaf of H_R necessarily belongs to R^*; more specifically, for every leaf ρ^* of H_R, there exists at least two distinct terminal vertices $w^1_{\rho^*}, w^2_{\rho^*} \in W$ such that $\mathrm{dist}_{T'}(w^i_{\rho^*}, \rho^*) = \mathrm{dist}_{T'}(w^i_{\rho^*}, \mathsf{R}(T'))$ for $i \in \{1, 2\}$, otherwise the degree of ρ^* in T' would be less than 3. Let $\langle \rho_1, \ldots, \rho_k \rangle$ be an arbitrary ordering of the vertices in R^* such that ρ_1 and ρ_k are leaves of H_R, where $k = |R^*|$. Then, consider the graph T defined as follows: $V(T) = V(T') \setminus \big(\mathsf{R}(T') \setminus R^*\big)$ and $E(T) = \big(E(T') \setminus E(H_R)\big) \cup \{\rho_i \rho_{i+1} \mid i \in \{1, \ldots, k-1\}\}$. One may verify that T is a strict connection tree of G for W such that $\mathsf{L}(T) = \mathsf{L}(T')$, $\mathsf{R}(T) = R^* \subseteq \mathsf{R}(T')$, and $T[\mathsf{R}(T)]$ is a path. □

Proposition 3. *Suppose that $K \setminus W \neq \emptyset$ and $W \cap S \neq \emptyset$. Given two non-negative integers ℓ and r, with $r \geq 1$, we can in time $n^{\mathcal{O}(r)}$ obtain a strict connection tree T of G for W such that $|\mathsf{L}(T)| \leq \ell$ and $|\mathsf{R}(T)| \leq r$, or conclude that such a tree does not exist.*

Proof. Since S-TCP can be solved in polynomial-time when $r \leq 1$ [29], for simplicity, we assume that G does not admit a strict connection tree T for W such that $|\mathsf{L}(T)| \leq \ell$ and $|\mathsf{R}(T)| \leq 1$. Based on Lemmas 4 and 5, our strategy consists in enumerating all possible subsets $R \subseteq K \setminus W$, with $2 \leq |R| = k \leq r$, and all possible unordered pairs $\{\rho_1, \rho_k\} \subseteq R$ of distinct vertices in order to try to obtain a strict connection T of G for W such that $|\mathsf{L}(T)| \leq \ell$, $\mathsf{R}(T) = R$ and $T[\mathsf{R}(T)]$ is a path with endpoints ρ_1 and ρ_k. Hence, let $R \subseteq K \setminus W$, with $2 \leq |R| = k \leq r$, and ρ_1 and ρ_k be two distinct vertices belonging to R.

Let $W_R = W \cap N_G(R)$ and $\overline{W_R} = W \setminus W_R$. Note that, if $|\overline{W_R}| > \ell$, then G does not admit a strict connection tree for W such that $|\mathsf{L}(T)| \leq \ell$ and $\mathsf{R}(T) = R$. Thus, assume $|\overline{W_R}| \leq \ell$. Let H_1 be the bipartite graph defined as follows: $V(H_1) = X_1 \cup Y_1$ and $E(H_1) = \{xy \in E(G) \mid x \in X_1, y \in Y_1\}$, where $X_1 = \overline{W_R}$ and $Y_1 = (V(G) \setminus (R \cup W)) \cap N_G(X_1)$.

Claim 1. *If $X_1 \neq \emptyset$ and G admits a strict connection tree T for W such that $|\mathsf{L}(T)| \leq \ell$ and $\mathsf{R}(T) = R$, then there exists a matching M_1 in H_1 that saturates all vertices belonging to X_1.*

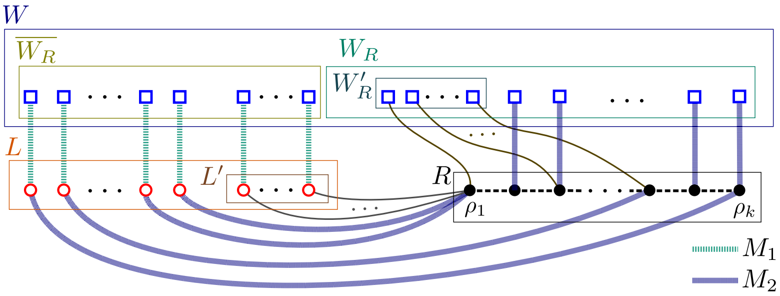


Fig. 10. Strict connection tree of G for W obtained from a matching M_2 in H_2 that saturates all vertices belonging to X_2.

Proof of claim. Assume that T holds properties (i)–(iii) described in Lemma 5. Thus, each linker $u \in \mathsf{L}(T)$ is adjacent to exactly one vertex in $\mathsf{R}(T)$ and exactly one vertex $w \in W$ such that $w \notin N_G(\mathsf{R}(T))$. As a consequence, the set of terminal vertices which are adjacent to a linker of T coincides with X_1. In addition to that, note that, since $|W| \geq 3$, each vertex belonging to $\mathsf{L}(T)$ is adjacent in T to at most one vertex belonging to X_1. Therefore, the set $M_1 = \{uw \in E(T) \mid u \in \mathsf{L}(T),\ w \in W\}$ is a matching in H_1 that saturates all vertices belonging to X_1. □

Based on Claim 1, we assume that $X_1 = \varnothing$, or that there exists a matching M_1 in H_1 that saturates all vertices belonging to X_1. From such a matching M_1 (if any), we let $L = \varnothing$ if $X_1 = \varnothing$, and $L = \{u \in Y_1 \mid uw \in M_1, w \in X_1\}$ otherwise; and we let H_2 be the bipartite graph such that $V(H_2) = X_2 \cup Y_2$ and

$$\begin{aligned} E(H_2) = \{&xy \in E(G) \mid x \in R \setminus \{\rho_1, \rho_k\}, y \in Y_2\} \\ &\cup \{\rho_i^j y \mid \rho_i y \in E(G), y \in Y_2, i \in \{1,k\}, j \in \{1,2\}\}, \end{aligned}$$

where $X_2 = (R \setminus \{\rho_1, \rho_k\}) \cup \{\rho_1^1, \rho_1^2, \rho_k^1, \rho_k^2\}$, $Y_2 = W_R \cup L$ and $\rho_1^1, \rho_1^2, \rho_k^1, \rho_k^2$ are new auxiliary vertices, not belonging to G.

Claim 2. *G admits a strict connection tree T for W such that $|\mathsf{L}(T)| \leq \ell$, $\mathsf{R}(T) = R$ and $T[\mathsf{R}(T)]$ is path with endpoints ρ_1 and ρ_k if and only if there exists a matching M_2 in H_2 that saturates all vertices belonging to X_2.*

Proof of claim. First, suppose that such a tree T exists. Additionally, assume that T holds properties (i)–(iii) described in Lemma 5. As a result, we have $|L| = |X_1| = |\mathsf{L}(T)|$. Let $\phi \colon \mathsf{L}(T) \to L$ be an arbitrary bijection. Since all routers of T have degree at least 3, each endpoint of the path $T[\mathsf{R}(T)]$ — i.e. the vertices ρ_1 and ρ_k — must be adjacent to at least two distinct vertices in $W_R \cup \mathsf{L}(T)$; thus, for $i \in \{1,k\}$, let $v_i^1, v_i^2 \in W_R \cup \mathsf{L}(T)$ be two arbitrary distinct neighbors of ρ_i in T. Furthermore, we have that each internal vertex of $T[\mathsf{R}(T)]$ must be adjacent to at least one vertex in $W_R \cup \mathsf{L}(T)$; thus, for $i \in \{2,\ldots,k-1\}$, let $v_i \in W_R \cup \mathsf{L}(T)$ be an arbitrary neighbor of ρ_i in T. Let $Y_{M_2} = \{v_i^j \mid i \in \{1,k\},\ j \in \{1,2\}\} \cup \{v_i \mid i \in \{2,\ldots,k-1\}\}$. We remark that $Y_{M_2} \setminus W_R \subseteq \mathsf{L}(T)$. Therefore, one may verify that

$$\begin{aligned} M_2 = \{&\rho_i^j v_i^j \mid \rho_i v_i^j \in E(T),\ v_i^j \in Y_{M_2} \cap W_R,\ i \in \{1,k\},\ j \in \{1,2\}\} \\ &\cup \{\rho_i v_i \in E(T) \mid v_i \in Y_{M_2} \cap W_R,\ i \in \{2,\ldots,k-1\}\} \\ &\cup \{\rho_i^j \phi(v_i^j) \mid \rho_i v_i^j \in E(T),\ v_i^j \in Y_{M_2} \setminus W_R,\ i \in \{1,k\},\ j \in \{1,2\}\} \\ &\cup \{\rho_i \phi(v_i) \mid \rho_i v_i \in E(T),\ v_i \in Y_{M_2} \setminus W_R,\ i \in \{2,\ldots,k-1\}\} \end{aligned}$$

is a matching in H_2 that saturates all vertices belonging to X_2.

Conversely, suppose that there exists a matching M_2 in H_2 that saturates all vertices belonging to X_2. Let W'_R and L' be the subsets of W_R and L, respectively, composed by the vertices which are not saturated by M_2. Also, let $\varphi \colon W'_R \to R$ be a mapping such that, for each $w \in W'_R$, if $\varphi(w) = \rho$, then $w \in N_G(\rho)$. Consider the graph T defined as follows: $V(T) = W \cup L \cup R$ and

$$\begin{aligned} E(T) = M_1 &\cup (M_2 \setminus \{\rho_i^j v_i \mid v_i \in Y_2,\ i \in \{1,k\},\ j \in \{1,2\}\}) \\ &\cup \{\rho_i v_i \mid \rho_i^j v_i \in M_2,\ v_i \in Y_2, i \in \{1,k\},\ j \in \{1,2\}\} \\ &\cup \{\rho_1 v \mid v \in L'\} \cup \{\varphi(w)w \mid w \in W'_R\} \cup \{\rho_i \rho_{i+1} \mid i \in \{1,\ldots,k-1\}\}. \end{aligned}$$

Fig. 10 illustrates the graph T. One may verify that T is a strict connection tree of G for W such that $\mathsf{L}(T) = L$, $\mathsf{R}(T) = R$ and $T[\mathsf{R}(T)]$ is a path with endpoints ρ_1 and ρ_k. □

To conclude the proof of this proposition, we remark that, based on Lemmas 4 and 5, there exists a strict connection tree of G for W with at most ℓ linkers and at most r routers if and only if, for some set $R \subseteq K \setminus W$, with $2 \leq |R| = k \leq r$, and some unordered pair $\{\rho_1, \rho_k\} \subseteq R$ of distinct vertices, there exists a strict connection tree T of G for W such that $|\mathsf{L}(T)| \leq \ell$, $\mathsf{R}(T) = R$ and $T[\mathsf{R}(T)]$ is a path with endpoints ρ_1 and ρ_k.

Furthermore, based on the previous claims, we have that, for a given set $R \subseteq K \setminus W$, with $2 \leq |R| = k \leq r$, and a given unordered pair $\{\rho_1, \rho_k\} \subseteq R$ of distinct vertices, we can obtain a strict connection tree T of G for W such that $|\mathsf{L}(T)| \leq \ell$, $\mathsf{R}(T) = R$ and $T[\mathsf{R}(T)]$ is a path with endpoints ρ_1 and ρ_k, or conclude that such a tree T does not exist, in time polynomial in n. Therefore, since all unordered pair $\{\rho_1, \rho_k\} \subseteq R$ of distinct vertices can be enumerated in time $\mathcal{O}\left(n^2\right)$ and all subsets $R \subseteq K \setminus W$, with $2 \leq |R| \leq r$, can be enumerated in time $n^{\mathcal{O}(r)}$, the total running time of the algorithm is $n^{\mathcal{O}(r)}$. □

Theorem 7. *S-TCP restricted to split graphs can be solved in time* $n^{\mathcal{O}(r)}$ *but, assuming FPT* $\neq$ *W*[2], *cannot be solved in time* $g(r) \cdot n^{h(\ell)}$ *and, assuming ETH, cannot be solved in time* $g(r) \cdot n^{o(r)}$, *for any computable functions g and h.*

Proof. It follows immediately from Facts 1 and 2 and Proposition 3 that S-TCP restricted to split graphs can be solved in time $n^{\mathcal{O}(r)}$.

On the other hand, to see that S-TCP restricted to split graphs does not admit a $g(r) \cdot n^{h(\ell)}$-time algorithm, unless FPT = W[2], note that the proof of Theorem 4 can be easily adapted so that the constructed graph G becomes a split graph. Indeed, it is enough to add to G the edge $u_i u_j$ for each $i, j \in \{1, \ldots, \ell\}$ with $i \neq j$. In this case, $\{K = L \cup K_{\mathcal{F}}, S = W\}$ is a partition of the vertex set V of G into a clique and an independent set, respectively. Therefore, S-TCP remains W[2]-hard even if it is restricted to split graphs and ℓ is bounded by a constant.

Finally, to show that S-TCP restricted to split graphs does not admit a $g(r) \cdot n^{o(r)}$-time algorithm, unless ETH fails, consider the following claim.

Claim 3. *Assuming ETH,* Set cover *cannot be solved in time* $g(k) \cdot n^{o(k)}$, *for any computable function* g.

Proof of claim. We present a polynomial-time reduction from Dominating set, another classical W[2]-complete problem, which under ETH was proved not to admit a $g(k) \cdot n^{o(k)}$-time algorithm, for any computable function g, where k is the parameter of the problem *cf.* [14]. Let $I = (G', k')$ be an instance of Dominating set. We construct an instance $f(I) = (U, \mathcal{F}, k)$ of Set cover, as follows: $U = V(G')$, $\mathcal{F} = \{N_{G'}[u'] \mid u' \in V(G')\}$ and $k = k'$.

It is easy to see that, if a set $S' \subseteq V(G')$ is a dominating set of G', then $\mathcal{F}' = \{N_{G'}[u'] \mid u' \in V'\} \subseteq \mathcal{F}$ is a vertex cover of G such that $|\mathcal{F}'| = |S'|$.

Conversely, if $\mathcal{F}' \subseteq \mathcal{F}$ is a vertex cover of G, then $S' = \{u' \mid N_{G'}[u'] \in \mathcal{F}'\} \subseteq V(G')$ is a dominating set of G' such that $|S'| = |\mathcal{F}'|$.

Thus, I is a Yes instance of Dominating set if and only if $f(I)$ is a Yes instance of Set cover. Consequently, the existence of a $g(k) \cdot |V(G)|^{o(k)}$-time algorithm for Set cover implies the existence of a $g(k') \cdot |V(G')|^{o(k')}$-time algorithm for Dominating set. □

Therefore, we obtain from the proof of Theorem 4 that the existence of such an algorithm for S-TCP implies the failure of ETH. □

5. The cograph case

A *cograph* is a graph that does not contain any induced path of length 3. Alternatively, cographs can be characterized by the following recursive definition given by Corneil et al. [33]: G is a cograph if and only if $G = K_1$ or there exist two other cographs G_1 and G_2 such that either $G = G_1 \cup G_2$ or $G = G_1 \wedge G_2$, where K_1 denotes the trivial graph with a single vertex, and $G_1 \cup G_2$ and $G_1 \wedge G_2$ respectively denote the *disjoint union* and the *join* of G_1 and G_2, i.e. $V(G_1 \cup G_2) = V(G_1 \wedge G_2) = V(G_1) \cup V(G_2)$, $E(G_1 \cup G_2) = E(G_1) \cup E(G_2)$ and $E(G_1 \wedge G_2) = E(G_1) \cup E(G_2) \cup \{uv \mid u \in V(G_1), v \in V(G_2)\}$.

In complex networks, *cograph communities* are defined as the connected components of a network such that the underlying graph is a cograph. According to [34], as a whole community, cograph communities reveal more intensive social roles or biological functions than those obtained by general communities. Thus, motivated by the relevance of cographs in complex networks, we analyze in this section the complexity of S-TCP restricted to cographs.

More specifically, we prove that S-TCP on cographs is polynomial-time solvable. Although this can be an expected result (for instance, it is known that Steiner tree on cographs is polynomial-time solvable [9]), since cographs have strong structural properties that are useful for the development of polynomial-time algorithms, our proof is not trivial whatsoever, consisting in providing a sophisticated dynamic programming algorithm for the problem.

Since S-TCP can be easily solved in linear-time when $r < 1$ or $|W| < 3$, we assume that $r \geq 1$ and $|W| \geq 3$. Next, we analyze all the other possible cases, and then we finally summarize in Theorem 8 the recurrence relation of our algorithm.

Fact 3. Let $G = (V, E)$ be a cograph such that $G = G_1 \cup G_2$. Then, G admits a strict connection tree for W with at most $\ell \geq 0$ linkers and at most $r \geq 1$ routers if and only if $V(G_j) \cap W = \varnothing$ and G_i admits a strict connection tree for W with at most ℓ linkers and at most r routers, where $V(G_i) \cap W \neq \varnothing$, $i, j \in \{1, 2\}$ and $i + j = 3$.

Fact 4. Let $G = (V, E)$ be a cograph such that $G = G_1 \wedge G_2$, $V(G_1) \subseteq W$ and $|V(G_2) \cap W| \leq 1$. Then, there exists a strict connection tree of G for W if and only if $W = V(G_1)$, or $V(G_2) \cap W = \{w\}$ and $N_{G_2}(w) \neq \varnothing$. In particular, if such a tree exists, then G admits a strict connection tree T for W such that $|\mathsf{L}(T)| = 0$ and $|\mathsf{R}(T)| = 1$.

Lemma 6. *Let $G = (V, E)$ be a cograph such that $G = G_1 \wedge G_2$, $V(G_1) \subset W$ and $|V(G_2) \cap W| = 2$. Let w'_1 and w'_2 be the two vertices belonging to $V(G_2) \cap W$. Then, G admits a strict connection tree for W with at most $\ell \geq 0$ linkers and at most $r \geq 1$ routers if and only if the distance in $G'_2 = G_2 - w'_1 w'_2$ between w'_1 and w'_2 is at most $\ell + \min\{r, n_1\} + 1$, where $n_1 = |V(G_1)|$.*

Proof. First, suppose that G admits a strict connection tree T for W such that $|\mathsf{L}(T)| \leq \ell$ and $|\mathsf{R}(T)| \leq r$. Let P be the path in T between w'_1 and w'_2. Observe that, the length of P is at most $|\mathsf{L}(T)| + |\mathsf{R}(T)| + 1$, otherwise T would have more than $|\mathsf{L}(T)|$ linkers or more than $|\mathsf{R}(T)|$ routers. Since by hypothesis $V(G_1) \subset W$ and $|V(G_2) \cap W| = 2$, we have that $n_1 = |V(G_1)| = |W| - 2$. Furthermore, we know that $|\mathsf{R}(T)| \leq |W| - 2$ (see Proposition 2), which implies $|\mathsf{R}(T)| \leq n_1$. Thus, $|\mathsf{R}(T)| \leq \min\{r, n_1\}$, and so the length of P is at most $|\mathsf{L}(T)| + \min\{r, n_1\} + 1 \leq \ell + \min\{r, n_1\} + 1$. Finally, since $V(G_1) \subset W$ and $|W| \geq 3$, P is contained in G'_2, otherwise T would have some terminal with degree greater than 1. Therefore, the distance in G'_2 between w'_1 and w'_2 is at most $\ell + \min\{r, n_1\} + 1$.

Conversely, suppose that the distance in G'_2 between w'_1 and w'_2 is at most $\ell + \min\{r, n_1\} + 1$. Let $P = \langle w'_1, u'_1, u'_2, \ldots, u'_z, w'_2 \rangle$ be a shortest path in G'_2 between w'_1 and w'_2, and let $\alpha = \langle w_1, w_2, \ldots, w_{n_1} \rangle$ be an arbitrary ordering of the vertices in $V(G_1)$. We define from P and α the subgraph T of G as follows: $V(T) = V(P) \cup V(G_1)$ and $E(T) = E(P) \cup \{u'_i w_i \mid i \in \{1, \ldots, \min\{r, n_1, z\}\}\} \cup \{u'_1 w_i \mid i \in \{\min\{r, n_1, z\} + 1, \ldots, n_1\}\}$. Observe that, T is a strict connection tree of G for W such that $\mathsf{L}(T) = \{u'_i \mid i \in \{\min\{r, n_1\} + 1, \ldots, z\}\}$ and $\mathsf{R}(T) = \{u'_i \mid i \in \{1, \ldots, \min\{r, n_1, z\}\}\}$. Since by hypothesis the length of P is at most $\ell + \min\{r, n_1\} + 1$, $z \leq \ell + \min\{r, n_1\}$, and consequently $|\mathsf{L}(T)| \leq \ell$. Finally, note that $|\mathsf{R}(T)| = \min\{r, n_1, z\} \leq r$. Therefore, G admits a strict connection tree for W with at most ℓ linkers and at most r routers. □

Lemma 7. *Let $G = (V, E)$ be a cograph such that $G = G_1 \wedge G_2$, $V(G_1) \subset W$ and $|V(G_2) \cap W| \geq 3$. Then, G admits a strict connection tree for W with at most $\ell \geq 0$ linkers and at most $r \geq 1$ routers if and only if G_2 admits a strict connection tree for $V(G_2) \cap W$ with at most $\ell + \lambda$ linkers and at most $r - \lambda$ routers, for some $\lambda \in \{0, 1, \ldots, \min\{r - 1, n_1\}\}$, where $n_1 = |V(G_1)|$.*

Proof. First, suppose that G admits a strict connection tree T for W such that $|\mathsf{L}(T)| \leq \ell$ and $|\mathsf{R}(T)| \leq r$. Since $|V(G_2) \cap W| \geq 3$ and all vertices in $V(G_1)$ are terminal, there exists at least one router $\rho \in \mathsf{R}(T)$ such that $\rho \in V(G_2)$.

Suppose that there is a vertex $v \in V(G_2) \setminus W$ with $N_T(v) \cap V(G_1) \neq \varnothing$, such that $|N_{G_2}(v) \cap V(T)| < 2$. Then, let H be the graph obtained from T by removing the vertex v and by adding, for each $w \in N_T(v) \cap V(G_1)$, the edge ρw. We remark that H possibly contains non-terminal vertices that became leaves. Then, let T_H be the graph obtained from H by successively removing such non-terminal vertices. One can easily verify that T_H is a strict connection tree of G for W such that $\mathsf{L}(T_H) \subseteq \mathsf{L}(T)$ and $\mathsf{R}(T_H) \subseteq \mathsf{R}(T)$. Thus, hereinafter, assume without loss of generality that $|N_{G_2}(v) \cap V(T)| \geq 2$ for every vertex $v \in V(G_2) \setminus W$ with $N_T(v) \cap V(G_1) \neq \varnothing$.

Let $T' = T - V(G_1)$. Since by hypothesis $V(G_1) \subset W$, the vertices belonging to $V(G_1)$ are leaves of T. Thus, T' is a connected graph. More specifically, it follows from the assumption described in the previous paragraph that T' is a strict connection tree of G_2 for $V(G_2) \cap W$. Let

$$R' = \{v \in V(G_2) \setminus W \mid N_T(v) \cap V(G_1) \neq \varnothing, d_{T'}(v) < 3\}.$$

Note that, the vertices belonging to R' are routers of T and linkers of T'. Thus, $|\mathsf{L}(T')| = |\mathsf{L}(T)| + |R'|$ and $|\mathsf{R}(T')| = |\mathsf{R}(T)| - |R'|$. Moreover, since the vertices belonging to $V(G_1)$ are leaves of T and $|W| \geq 3$, each vertex $w \in V(G_1)$ is adjacent in T to exactly one vertex $v \in V(G_2) \setminus W$. Consequently, $|R'| \leq n_1$. On the other hand, since by hypothesis $|V(G_2) \cap W| \geq 3$, T' has at least one router. Thus, $|R'| \leq r - 1$, which implies $|R'| \leq \min\{r - 1, n_1\}$. Therefore, we can define $\lambda = |R'|$, and so we have that G_2 admits a strict connection tree for $V(G_2) \cap W$ with at most $\ell + \lambda$ linkers and at most $r - \lambda$ routers.

For the converse, suppose now that G_2 admits a strict connection tree T' for $V(G_2) \cap W$ such that $|\mathsf{L}(T')| \leq \ell + \lambda$ and $|\mathsf{R}(T')| \leq r - \lambda$, for some $\lambda \in \{0, 1, \ldots, \min\{r - 1, n_1\}\}$. Since $|V(G_2) \cap W| \geq 3$, $\mathsf{R}(T')$ is non-empty. Then, let $\rho \in \mathsf{R}(T')$, arbitrarily chosen. Let R' be a subset of $\mathsf{L}(T')$ such that $|R'| = z$, where $z = \min\{|\mathsf{L}(T')|, \lambda\}$; and let $\langle u'_1, u'_2, \ldots, u'_z \rangle$ be an arbitrary ordering of the vertices belonging to R'. Also, let $\langle w_1, w_2, \ldots, w_{n_1} \rangle$ be an arbitrary ordering of the vertices belonging to $V(G_1)$. Then, the graph T, defined as follows: $V(T) = V(T') \cup V(G_1)$ and $E(T) = E(T') \cup \{u'_i w_i \mid i \in \{1, \ldots, z\}\} \cup \{\rho w_i \mid i \in \{z+1, \ldots, n_1\}\}$, is a strict connection tree for W such that $\mathsf{L}(T) = \mathsf{L}(T') \setminus \{u'_i \mid i \in \{1, \ldots, z\}\}$ and $\mathsf{R}(T) = \mathsf{R}(T') \cup \{u'_i \mid i \in \{1, \ldots, z\}\}$. Therefore, G admits a strict connection tree T for W such that $|\mathsf{L}(T)| = |\mathsf{L}(T')| - z = |\mathsf{L}(T')| - \min\{|\mathsf{L}(T')|, \lambda\} \leq \ell$ and $|\mathsf{R}(T)| = |\mathsf{R}(T')| + z = |\mathsf{R}(T')| + \min\{|\mathsf{L}(T')|, \lambda\} \leq r$. □

Proposition 4. *Let $G = (V, E)$ be a cograph such that $G = G_1 \wedge G_2$, $V(G_1) \not\subseteq W$, $V(G_1) \cap W \neq \varnothing$, $V(G_2) \not\subseteq W$ and $V(G_2) \cap W \neq \varnothing$. Given $\ell \geq 0$ and $r \geq 1$, we can in polynomial-time obtain a strict connection tree T for W such that $|\mathsf{L}(T)| \leq \ell$ and $|\mathsf{R}(T)| \leq r$, or conclude that such a tree does not exist.*

Proof. First, consider $r = 1$. By a Turing reduction to the problem of finding vertex-disjoint paths with minimum total cost, we can in polynomial-time obtain a strict connection tree T of G for W such that $|\mathsf{L}(T)| \leq \ell$ and $|\mathsf{R}(T)| = 1$, or conclude that such a tree does not exist [29].

Thus, hereinafter, assume that $r \geq 2$ and that G does not admit a strict connection tree T for W such that $|\mathsf{L}(T)| \leq \ell$ and $|\mathsf{R}(T)| \leq 1$. Consequently, assume also that $|W| \geq 4$. There are two cases to be analyzed.

Case 1: $|V(G_1) \cap W| = 1$.

In this case, we can additionally assume that $\ell = 0$ and $N_{G_1}(w) = \varnothing$, where w is the only vertex in $V(G_1) \cap W$. Indeed, suppose that $\ell \geq 1$. Then, for $v \in V(G_1) \setminus W$ and $v' \in V(G_2) \setminus W$, we have that the graph T, where $V(T) = W \cup \{v, v'\}$ and $E(T) = \{vv', v'w\} \cup \{vw' \mid w' \in V(G_2) \cap W\}$, is a strict connection tree of G for W such that $\mathsf{L}(T) = \{v'\}$ and $\mathsf{R}(T) = \{v\}$. Now, suppose that $N_{G_1}(w) \neq \varnothing$, and let $v \in N_{G_1}(w)$. Note that, $N_G(v) \supseteq W$. Consequently, G admits a strict connection tree for W without linkers and with at most one router. For example, the graph T, where $V(T) = W \cup \{v\}$ and $E(T) = \{vw \mid w \in W\}$, is a strict connection tree of G for W such that $\mathsf{L}(T) = \varnothing$ and $\mathsf{R}(T) = \{v\}$. Thus, we assume hereinafter that $\ell = 0$ and $N_{G_1}(w) = \varnothing$.

If $N_{G_2}(w') \setminus W \neq \varnothing$ for some terminal $w' \in V(G_2) \cap W$, then G admits a strict connection tree T for W such that $|\mathsf{L}(T)| = 0$ and $|\mathsf{R}(T)| = 2$. Indeed, for $v' \in N_{G_2}(w') \setminus W$ and $v \in V(G_1) \setminus W$, we have that the graph T, where $V(T) = W \cup \{v, v'\}$ and $E(T) = \{vv', v'w, v'w'\} \cup \{vw' \mid w' \in (V(G_2) \cap W) \setminus \{w'\}\}$, is a strict connection tree for W such that $\mathsf{L}(T) = \varnothing$ and $\mathsf{R}(T) = \{v, v'\}$. Thus, assume that, for every vertex $w' \in V(G_2) \cap W$, $N_{G_2}(w') \setminus W = \varnothing$.

Note that, if $r = 2$ or $|V(G_2) \cap W| = 3$, then G does not admit a strict connection tree for W without linkers and with at most r routers. Suppose for purposes of contradiction that it is not true, and let T be such a tree. Since $N_{G_1}(w) = \varnothing$, the only neighbor of w in T is a non-terminal vertex $v' \in V(G_2) \setminus W$. However, if $r = 2$, then $|\mathsf{R}(T)| = 2$; and if $|V(G_2) \cap W| = 3$, then $|\mathsf{R}(T)| \leq |W| - 2 = 2$ (see Proposition 2). Consequently, T has at most two non-terminal vertices, being v' one of those vertices. Thus, v' is adjacent to at most one non-terminal vertex in T, and so its degree in T is at most 2, since $N_{G_2}(v') \cap W = \varnothing$. Therefore, G does not admit a strict connection tree for W without linkers and with at most r routers. Similarly, it is easy to see that G does not admit such a tree if $|V(G_1) \setminus W| = 1$. Hence, assume that $r \geq 3$, $|V(G_2) \cap W| \geq 4$ and $|V(G_1) \setminus W| \geq 2$.

Let $v_1, v_2 \in V(G_1) \setminus W$ and $v' \in V(G_2) \setminus W$. Also, let $\langle w'_1, w'_2, \ldots, w'_{n_2} \rangle$ be an arbitrary ordering of the vertices belonging to $V(G_2) \cap W$, where $n_2 = |V(G_2) \cap W|$. Then, the graph T, where $V(T) = W \cup \{v_1, v_2, v'\}$ and $E(T) = \{v_1v', v_2v', v_1w'_1, v_2w'_2, v'w'\} \cup \{v_2w'_i \mid i \in \{3, \ldots, n_2\}\}$, is a strict connection tree for W such that $\mathsf{L}(T) = \varnothing$ and $\mathsf{R}(T) = \{v_1, v_2, v'\}$. Therefore, G admits a strict connection tree T for W such that $|\mathsf{L}(T)| = 0 = \ell$ and $|\mathsf{R}(T)| = 3 \leq r$.

Case 2: $|V(G_1) \cap W| \geq 2$ and $|V(G_2) \cap W| \geq 2$.

Let $v \in V(G_1) \setminus W$ and $v' \in V(G_2) \setminus W$. Then, the graph T, where $V(T) = W \cup \{v, v'\}$ and $E(T) = \{vv'\} \cup \{vw' \mid w' \in V(G_2) \cap W\} \cup \{v'w \mid w \in V(G_1) \cap W\}$, is a strict connection tree for W such that $\mathsf{L}(T) = \varnothing$ and $\mathsf{R}(T) = \{v, v'\}$. Therefore, in this case, G always admits a strict connection tree T for W such that $|\mathsf{L}(T)| = 0$ and $|\mathsf{R}(T)| = 2$.

To conclude the proof of this proposition, note that all operations described above can be computed in time polynomial in n. □

An important property of cographs is that every cograph G can be uniquely represented by a rooted tree $\mathcal{T}_G$, called *cotree*, such that (1) the leaves of $\mathcal{T}_G$ correspond to the vertices of G; and, (2) each internal node u of $\mathcal{T}_G$ corresponds to either the disjoint union or the join of the cographs induced by the leaves of the subtrees of T_G rooted at each child of u [33]. Throughout this section, we assume without loss of generality that a cotree is a binary tree. Thus, the cotree $\mathcal{T}_G$ of a cograph G can be viewed as the tree corresponding to the unique decomposition of G as the trivial graph K_1, or either the join or the union of two other cographs. Another important property is the fact that the recognition of a given graph G as a cograph, as well as obtaining its respective cotree (if any), can be performed in time linear in n and m [35].

Theorem 8. *S-TCP is polynomial-time solvable if it is restricted to cographs.*

Proof. Let $I = (G, W, \ell, r)$ be an instance of S-TCP, where $G = (V, E)$ is a cograph, and let $\mathcal{T}_G$ be the cotree of G. We define a dynamic programming table M such that: for each node u' of $\mathcal{T}_G$ and for each pair of non-negative integers ℓ' and r', with $\ell' + r' < |V(G_{u'}) \setminus W|$, there is an entry $M[G_{u'}, \ell', r']$ which is set *true* if and only if $V(G_{u'}) \cap W \neq \varnothing$ and $G_{u'}$ admits a strict connection tree for $V(G_{u'}) \cap W$ with at most ℓ' linkers and at most r' routes, where $G_{u'}$ denotes the cograph associated with the subtree of $\mathcal{T}_G$ rooted at u'.

Facts 3 and 4, Lemmas 6 and 7 and Proposition 4 are used to fill M and, thus, decide whether G admits a strict connection tree for W with at most ℓ linkers and at most r routers. More specifically, $M[G,\ell,r]$ is defined on the basis of the following rules:

$$M[G,\ell,r] := \begin{cases}
\left[\begin{array}{l}
\textbf{case 1. } |V\cap W|\le 2 \text{ or } r=0 : \\
\quad \mathit{true} \;\; \text{if } |V\cap W|=1, \\
\quad \mathit{true} \;\; \text{if } V\cap W=\{w_1,w_2\} \text{ and } \mathrm{dist}_G(w_1,w_2)\le \ell+1, \\
\quad \mathit{false} \;\; \text{otherwise;}
\end{array}\right. \\[2ex]
\left[\begin{array}{l}
\textbf{case 2. } G=G_1\cup G_2 : \\
\quad M[G_1,\ell,r] \;\; \text{if } V(G_2)\cap W=\varnothing, \\
\quad \mathit{false} \;\; \text{otherwise;}
\end{array}\right. \\[2ex]
\left[\begin{array}{l}
\textbf{case 3. } G=G_1\wedge G_2 \text{ and } V(G_1)=W : \\
\quad \mathit{true};
\end{array}\right. \\[2ex]
\left[\begin{array}{l}
\textbf{case 4. } G=G_1\wedge G_2,\, V(G_1)\subset W \text{ and } V(G_2)\cap W=\{w\} : \\
\quad \mathit{true} \;\; \text{if } N_{G_2}(w)\neq\varnothing, \\
\quad \mathit{false} \;\; \text{otherwise;}
\end{array}\right. \\[2ex]
\left[\begin{array}{l}
\textbf{case 5. } G=G_1\wedge G_2,\, V(G_1)\subset W \text{ and } V(G_2)\cap W=\{w'_1,w'_2\} : \\
\quad \mathit{true} \;\; \text{if } \mathrm{dist}_{G'_2}(w'_1,w'_2) < \ell+\min\{r,n_1\}+1, \\
\quad \mathit{false} \;\; \text{otherwise,} \\
\text{where } G'_2=G_2-w'_1w'_2 \text{ and } n_1=|V(G_1)|;
\end{array}\right. \\[2ex]
\left[\begin{array}{l}
\textbf{case 6. } G=G_1\wedge G_2,\, V(G_1)\subset W \text{ and } |V(G_2)\cap W|\ge 3 : \\
\quad \displaystyle\bigvee_{\lambda=0}^{\min\{r-1,n_1\}} M[G_2,\ell+\lambda,r-\lambda], \\
\text{where } n_1=|V(G_1)|;
\end{array}\right. \\[2ex]
\left[\begin{array}{l}
\textbf{case 7. } G=G_1\wedge G_2,\, V(G_i)\not\subset W,\, V(G_i)\cap W\neq\varnothing, \forall i\in\{1,2\} : \\
\quad \textsc{Alg}(G,W,\ell,r), \\
\text{where } \textsc{Alg} \text{ denotes the algorithm described in Proposition 4.}
\end{array}\right.
\end{cases}$$

Note that, the size of M is $\mathcal{O}(n^3)$. Furthermore, one may verify that each entry of M can be computed in time polynomial in n, in a bottom-up manner according to the post-order traversal of $\mathcal{T}_G$. Regarding the correctness of the dynamic programming algorithm, case 1 can be easily verified *cf.* [29]; case 2 derives from Fact 3; case 3 and 4 derive from Fact 4; case 5 derives from Lemma 6; case 6 derives from Lemma 7; and case 7 clearly derives from Proposition 4. □

6. Conclusions and open problems

We have presented several complexity results for S-TCP (see Table 1). Nonetheless, the complexity of the problem remains unknown on some particular cases. Thus, to conclude this work, three open questions are highlighted.

(i) Is S-TCP parameterized by r in XP?
(ii) Is S-TCP parameterized by ℓ in FPT when $\Delta = 3$?
(iii) Is S-TCP parameterized by $|W|$ in FPT? And if r and Δ are parameters?

Although Steiner tree parameterized by $|W|$ is in FPT [10], it is not clear that S-TCP parameterized by $|W|$, or even parameterized by r and Δ, is also in FPT.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] M.C. Dourado, R.A. Oliveira, F. Protti, U.S. Souza, Design of connection networks with bounded number of non-terminal vertices, in: Proceedings of V Latin-American Workshop on Cliques in Graphs, in: Matemática Contemporânea, vol. 42, SBM, Buenos Aires, 2014, pp. 39–47.
[2] M.C. Dourado, R.A. Oliveira, F. Protti, U.S. Souza, Conexão de terminais com número restrito de roteadores e elos, in: Proccedings of XLVI Simpósio Brasileiro de Pesquisa Operacional, 2014, pp. 2965–2976.
[3] G. Lin, G. Xue, On the terminal Steiner tree problem, Inf. Process. Lett. 84 (2) (2002) 103–107.
[4] R.M. Karp, Reducibility Among Combinatorial Problems, Springer US, Boston, MA, 1972, pp. 85–103.
[5] M.R. Garey, D.S. Johnson, The rectilinear Steiner tree problem is NP-complete, SIAM J. Appl. Math. 32 (4) (1977) 826–834.
[6] K. White, M. Farber, W. Pulleyblank, Steiner trees, connected domination and strongly chordal graphs, Networks 15 (1) (1985) 109–124.
[7] H. Müller, A. Brandstädt, The NP-completeness of Steiner tree and dominating set for chordal bipartite graphs, Theor. Comput. Sci. 53 (2–3) (1987) 257–265.
[8] A. D'Atri, M. Moscarini, Distance-hereditary graphs, Steiner trees, and connected domination, SIAM J. Comput. 17 (3) (1988) 521–538.
[9] C.J. Colbourn, L.K. Stewart, Permutation graphs: connected domination and Steiner trees, Discrete Math. 86 (1–3) (1990) 179–189.
[10] S.E. Dreyfus, R.A. Wagner, The Steiner problem in graphs, Networks 1 (3) (1971) 195–207.
[11] A. Björklund, T. Husfeldt, P. Kaski, M. Koivisto, Fourier meets Möbius: fast subset convolution, in: Proceedings of the Thirty-Ninth Annual ACM Symposium on Theory of Computing, STOC '07, Association for Computing Machinery, New York, NY, USA, 2007, pp. 67–74.
[12] M. Cygan, M. Pilipczuk, M. Pilipczuk, J.O. Wojtaszczyk, Kernelization hardness of connectivity problems in d-degenerate graphs, Discrete Appl. Math. 160 (15) (2012) 2131–2141.
[13] J. Nederlof, Fast polynomial-space algorithms using inclusion-exclusion, Algorithmica 65 (4) (2013) 868–884.
[14] M. Cygan, F.V. Fomin, Ł. Kowalik, D. Lokshtanov, D. Marx, M. Pilipczuk, M. Pilipczuk, S. Saurabh, Parameterized Algorithms, Springer, 2015.
[15] F.K. Hwang, D.S. Richards, P. Winter, The Steiner Tree Problem, Annals of Discrete Mathematics, vol. 53, Elsevier, 1992.
[16] F. Hwang, A linear time algorithm for full Steiner trees, Oper. Res. Lett. 4 (5) (1986) 235–237.
[17] C.L. Lu, C.Y. Tang, R.C.-T. Lee, The full Steiner tree problem, Theor. Comput. Sci. 306 (1–3) (2003) 55–67.
[18] A. Biniaz, A. Maheshwari, M. Smid, On the hardness of full Steiner tree problems, J. Discret. Algorithms 34 (2015) 118–127.
[19] H. Fernau, T. Fluschnik, D. Hermelin, A. Krebs, H. Molter, R. Niedermeier, Diminishable parameterized problems and strict polynomial kernelization, in: Sailing Routes in the World of Computation. Proceedings of 4th Conference on Computability in Europe, in: Lecture Notes in Computer Science, vol. 10936, Springer, 2018, pp. 161–171.
[20] R. Khandekar, G. Kortsarz, Z. Nutov, On some network design problems with degree constraints, J. Comput. Syst. Sci. 79 (5) (2013) 725–736.
[21] B. Fuchs, A note on the terminal Steiner tree problem, Inf. Process. Lett. 87 (4) (2003) 219–220.
[22] D.E. Drake, S. Hougardy, On approximation algorithms for the terminal Steiner tree problem, Inf. Process. Lett. 89 (1) (2004) 15–18.
[23] F.V. Martinez, J.C. de Pina, J. Soares, Algorithms for terminal Steiner trees, Theor. Comput. Sci. 389 (1) (2007) 133–142.
[24] Y.H. Chen, An improved approximation algorithm for the terminal Steiner tree problem, in: Proceedings of International Conference on Computational Science and Its Applications, Springer, Berlin, Heidelberg, 2011, pp. 141–151.
[25] A. Biniaz, A. Maheshwari, M. Smid, On full Steiner trees in unit disk graphs, Comput. Geom. 48 (6) (2015) 453–458.
[26] L. Gargano, M. Hammar, P. Hell, L. Stacho, U. Vaccaro, Spanning spiders and light-splitting switches, Discrete Math. 285 (1) (2004) 83–95.
[27] D. Watel, M.-A. Weisser, C. Bentz, D. Barth, Steiner problems with limited number of branching nodes, in: Proceedings of 20th International Colloquium on Structural Information and Communication Complexity, in: Lecture Notes in Computer Science, vol. 8179, Springer-Verlag Inc., New York, 2013, pp. 310–321.
[28] D. Watel, M.-A. Weisser, C. Bentz, D. Barth, Directed Steiner trees with diffusion costs, J. Comb. Optim. 32 (4) (2016) 1089–1106.
[29] A.A. Melo, C.M.H. Figueiredo, U.S. Souza, Connecting terminals using at most one router, in: Proceedings of VII Latin-American Workshop on Cliques in Graphs, in: Matemática Contemporânea, vol. 45, SBM, 2017, pp. 49–57.
[30] H. Müller, Hamiltonian circuits in chordal bipartite graphs, Discrete Math. 156 (1–3) (1996) 291–298.
[31] H.L. Bodlaender, B.M.P. Jansen, S. Kratsch, Cross-composition: a new technique for kernelization lower bounds, CoRR, arXiv:1011.4224 [abs].
[32] H.L. Bodlaender, B.M.P. Jansen, S. Kratsch, Kernelization lower bounds by cross-composition, SIAM J. Discrete Math. 28 (1) (2014) 277–305.
[33] D.G. Corneil, H. Lerchs, S.L. Burlingham, Complement reducible graphs, Discrete Appl. Math. 3 (3) (1981) 163–174.
[34] S. Jia, L. Gao, Y. Gao, J. Nastos, Y. Wang, X. Zhang, H. Wang, Defining and identifying cograph communities in complex networks, New J. Phys. 17 (1) (2015) 013044.
[35] D.G. Corneil, Y. Perl, L.K. Stewart, A linear recognition algorithm for cographs, SIAM J. Comput. 14 (4) (1985) 926–934.

Appendix C

Manuscript: The Strict Terminal Connection Problem on Chordal Bipartite Graphs

This appendix contains the manuscript:
Alexsander A. de Melo, Celina M. H. de Figueiredo, Uéverton S. Souza. The Strict Terminal Connection Problem on Chordal Bipartite Graphs. Published in *Matemática Contemporânea* (2021) [38].

Matemática Contemporânea, Vol. 48, 137–145
http://doi.org/10.21711/231766362021/rmc4814

The Strict Terminal Connection Problem on Chordal Bipartite Graphs

Alexsander Andrade de Melo
Celina Miraglia Herrera de Figueiredo
Uéverton dos Santos Souza

Abstract

A *strict connection tree* T of a graph G for a non-empty subset $W \subseteq V(G)$, called *terminal set*, is a tree subgraph of G whose leaf set coincides with W. A *non-terminal* vertex $v \in V(T) \setminus W$ is called *linker* if its degree in T is exactly 2, and it is called *router* if its degree in T is at least 3. Given a graph G, a terminal set $W \subseteq V(G)$ and two non-negative integers ℓ and r, the Strict terminal connection problem (S-TCP) asks whether G admits a strict connection tree for W with at most ℓ linkers and at most r routers. In the present extended abstract, we prove that S-TCP is NP-complete on chordal bipartite graphs even if ℓ is bounded by a constant.

1 Introduction

Steiner tree is one of the most fundamental problems in graph theory

2000 AMS Subject Classification: 68R10, 68Q17.

Keywords and Phrases: Connection tree, Terminal vertices, Steiner tree, Chordal bipartite graphs, NP-completeness.

This research was supported by CAPES (Finance code 001), CNPq (140399/2017-8, 407635/2018-1, 303726/2017-2), and FAPERJ (CNE E-26/202.793/2017, JCNE E-26/203.272/2017)

and combinatorial optimization, being related to many real-world applications. In this work, we study the complexity of the so-called Strict terminal connection problem, which is a natural variation of Steiner tree introduced by Dourado et al. [1] motivated by questions in information security, network routing and telecommunication.

Let G be a graph and $W \subseteq V(G)$ be a non-empty set, called *terminal* set. A *strict connection tree* of G for W is a tree subgraph of G whose leaf set is equal to W. A non-terminal vertex of a strict connection tree T is called *linker* if its degree in T is exactly 2, and it is called *router* if its degree in T is at least 3. We remark that the vertex set of every strict connection tree can be partitioned into terminal vertices, linkers and routers. For each strict connection tree T, we let $\mathsf{L}(T)$ denote the linker set of T and $\mathsf{R}(T)$ denote the router set of T. Next, we formally define the Strict terminal connection problem.

Strict Terminal Connection (S-TCP)

Input: A graph G, a non-empty terminal set $W \subseteq V(G)$ and two non-negative integers ℓ and r.

Question: Does there exist a strict connection tree T of G for W, such that $|\mathsf{L}(T)| \leq \ell$ and $|\mathsf{R}(T)| \leq r$?

Table 1 summarises the known complexity results of S-TCP with respect to the parameters $\ell, r, \Delta(G)$, and the classes of split graphs and cographs. In addition to these results, in [4], S-TCP was studied from the perspective of disjoint paths and integral commodity flow problems.

	Parameters				
Graph class	–	ℓ	r	ℓ, r	$\ell, r, \Delta(\mathbf{G})$
General	NPC [1]	NPC [1]	P for $r \in \{0,1\}$ [2] but W[2]h [3]	XP [1] but W[2]h [3]	FPT [1, 3] but No-poly kernel [3]
$\Delta = 4$	NPC [3]	NPC[3]	P for $r \in \{0,1\}$ [2]	FPT [1, 3]	FPT [1, 3]
$\Delta = 3$	NPC [3]	XP [3]	P for $r \in \{0,1\}$ [2]	FPT [1, 3]	FPT [1, 3]
Split	NPC [3]	NPC [3]	XP [3] but W[2]h [3]	XP [1, 3] but W[2]h [3]	FPT [1, 3]
Cographs	P [3]	P [3]	P [3]	P [3]	P [3]

Table 1: Computational complexity of S-TCP. (Adapted from [3].)

Contribution. In this work, we prove that S-TCP remains NP-complete when restricted to *chordal bipartite graphs*, even if $\ell \geq 0$ is bounded by a constant.

2 S-TCP on Chordal Bipartite Graphs

A graph G is called *chordal bipartite* if every induced cycle of G has length 4. Equivalently, a graph G is chordal bipartite if G is bipartite and every cycle of G of length at least 6 has a *chord*, i.e. an edge between two non-consecutive vertices of the cycle.

To prove that S-TCP is NP-complete on chordal bipartite graphs, we present a polynomial-time reduction from Vertex Cover, which has as input a graph G and a positive integer k and asks whether there is a subset $S \subseteq V(G)$ such that $|S| \leq k$ and every edge of G has an endpoint in S. The proposed reduction, described next, is based on the polynomial-time reduction given by Müller and Brandstädt [5] so as to prove that Steiner tree is NP-complete on chordal bipartite graphs.

Construction. Let $I = (G, k)$ be an instance of Vertex cover and $c \geq 0$ be a constant. Assume that $V(G) = \{v_1, \ldots, v_n\}$ for some positive integer $n \geq 2$. Moreover, assume that G has at least one edge, i.e. $m = |E(G)| \geq 1$. We let $f(I, c) = (H, W, \ell = c, r)$ be the instance of S-TCP defined as follows.

1. For each $v_i \in V(G)$, create the gadget H_i as illustrated in Figure 1.

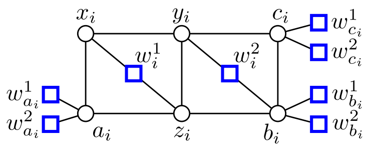
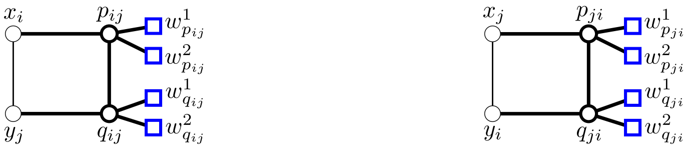


Figure 1: Gadget H_i.

2. Subdivide the edge $w^1_{a_1}a_1$ of H_1 into ℓ new vertices $u_1, u_2, \ldots, u_\ell$, creating the induced path $\langle w^1_{a_1}, u_1, \ldots, u_\ell, a_1\rangle$.

3. For each pair $v_i, v_j \in V(G)$, with $i \neq j$, add the edges x_iy_j and z_iy_j, making the subgraph of H induced by $X \cup Y \cup Z$ a complete bipartite graph with bipartition $(X \cup Z, Y)$, where $X = \{x_i \mid v_i \in V(G)\}$, $Y = \{y_i \mid v_i \in V(G)\}$ and $Z = \{z_i \mid v_i \in V(G)\}$.

4. For each $v_iv_j \in E(G)$, create the gadgets H_{ij} and H_{ji} as illustrated in Figure 2.



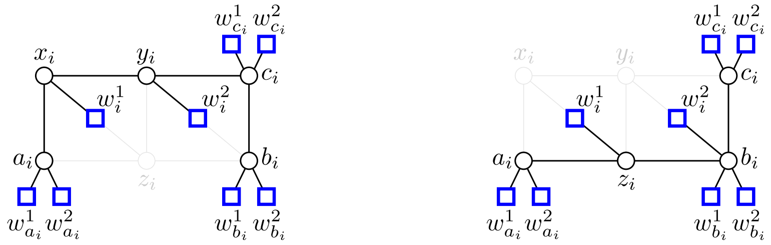
(a) H_{ij} (b) H_{ji}

Figure 2: Gadgets H_{ij} and H_{ji}, respectively.

5. Finally, define $W = W_1 \cup W_2 \cup W_3$ and $r = k + 4n + 4m$, where $W_1 = \{w^1_i, w^2_i \mid v_i \in V(G)\}$, $W_2 = \{w^1_{a_i}, w^2_{a_i}, w^1_{b_i}, w^2_{b_i}, w^1_{c_i}, w^2_{c_i} \mid v_i \in V(G)\}$, and $W_3 = \{w^1_{p_{ij}}, w^2_{p_{ij}}, w^1_{q_{ij}}, w^2_{q_{ij}} \mid v_iv_j \in E(G)\}$.

Lemma 2.1. *Let $I = (G, k)$ be an instance of* Vertex Cover, *such that G has at least one edge. For every $c \geq 0$, the graph H of $f(I, c)$ is chordal bipartite.*

Proof. First, we note that H is chordal bipartite if and only if the graph $G' = H - (W_2 \cup W_3)$ is chordal bipartite. Indeed, the vertices belonging to $W_2 \cup W_3$ are vertices of degree 1 of H, and therefore they do not belong to any cycle of H. Consequently, in order to prove this lemma, it is sufficient to show that G' is chordal bipartite. Note that, for every $v_i \in V(G)$, w^1_i is a false twin of a_i in G', i.e. $N_{G'}(w^1_i) = N_{G'}(a_i)$. Similarly, for every $v_i \in V(G)$, w^2_i is a false twin of c_i in G', i.e. $N_{G'}(w^2_i) = N_{G'}(c_i)$. As a result, if w^1_i or w^2_i belongs to an odd cycle or to an induced cycle of length greater than or equal to 6 in G', then certainly a_i or c_i, respectively, also belongs



(a) $H_i - z_i$

(b) $H_i - \{x_i, y_i\}$

Figure 3: Subgraphs $H_i - z_i$ and $H_i - \{x_i, y_i\}$, respectively.

to an odd cycle or to an induced cycle of length greater than or equal to 6 in G'. Therefore, it follows from the fact that $H - (W_1 \cup W_2 \cup W_3)$ is chordal bipartite [5] that G' (and, thus, H) is chordal bipartite as well. ■

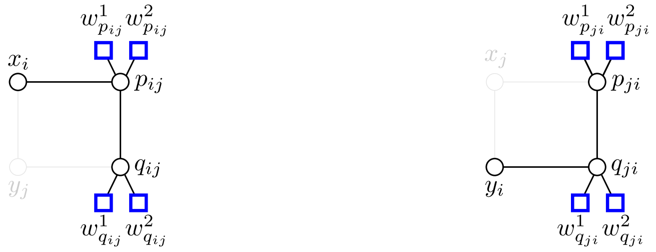
Lemma 2.2. *Let $I = (G, k)$ be an instance of* Vertex Cover*, such that G has at least one edge. For every $c \geq 0$, I is a* yes*-instance of* Vertex cover *if and only if $f(I, c)$ is a* yes*-instance of S-TCP.*

Proof. First, suppose that I is a yes-instance of Vertex cover, and let $S \subseteq V(G)$ be a vertex cover of G such that $|S| \leq k$. Based on S, we construct a strict connection tree T of H for W as described below.

1. For each $v_i \in V(G)$, if $v_i \in S$, then add the subgraph $H_i - z_i$ (see Figure 3a) to T; on the other hand, if $v_i \notin S$, then add the subgraph $H_i - \{x_i, y_i\}$ (see Figure 3b) to T.

2. For each $v_iv_j \in E(G)$ with $v_i \in S$, if $v_j \notin S$ or $i < j$, then add the subgraphs $H_{ij} - y_j$ (see Figure 4a) and $H_{ji} - x_j$ (see Figure 4b) to T. We remark that, possibly, the pairs of vertices x_i and y_j, and y_i and x_j, simultaneously belong to $V(T)$. However, if $x_ip_{ij} \in E(T)$ or $y_iq_{ji} \in E(T)$, then $x_iy_j, y_jq_{ij} \notin E(T)$ and $x_jy_i, x_jp_{ji} \notin E(T)$.

One can verify that, until the last step, T is an acyclic subgraph of H that contains all the terminal vertices belonging to W. Thus, in order to conclude the construction of T, we only need to connect the connected components of T in such a way that the resulting graph is still an acyclic



(a) $H_{ij} - y_j$

(b) $H_{ji} - x_j$

Figure 4: Subgraphs $H_{ij} - y_j$ and $H_{ji} - x_j$, respectively.

subgraph of H. Since $|E(G)| \geq 1$, $|S| \geq 1$. As a result, $Y_S = \{y_i \mid v_i \in S\}$ is non-empty. Then, let y_α be a vertex in Y_S, arbitrarily chosen. It follows from the construction of G that y_α is adjacent in G to all vertices belonging to $X_S \cup Z_S$, where $X_S = \{x_i \mid v_i \in S\}$ and $Z_S = \{z_i \mid v_i \notin S\}$. Moreover, note that all connected components of T necessarily have at least one vertex in $X_S \cup Z_S$. Thus, to conclude the construction of T, we perform the following operation:

3. For each connected component T' of T which does not contain the vertex y_α, select arbitrarily a vertex $v \in (X_S \cup Z_S) \cap V(T')$ and, then, add the edge vy_α to T.

Then, we have finally obtained a subgraph T of H which is a tree and contains all the terminal vertices belonging to W. Moreover, note that $\mathsf{L}(T) = \{u_1, u_2, \ldots, u_\ell\}$ and

$$\begin{aligned}\mathsf{R}(T) = \{&x_i, y_i \mid v_i \in S\} \cup \{z_i \mid v_i \notin S, v_i \in V(G)\} \\ &\cup \{a_i, b_i, c_i \mid v_i \in V(G)\} \cup \{p_{ij}, p_{ji}, q_{ij}, q_{ji} \mid v_iv_j \in E(G)\}.\end{aligned}$$

Hence, T is a strict connection tree of G for W such that $|\mathsf{L}(T)| = \ell$ and $|\mathsf{R}(T)| = 2|S| + (n - |S|) + 3n + 4m \leq k + 4n + 4m = r$. Therefore, $f(I, c)$ is a yes-instance of S-TCP.

Conversely, suppose that G admits a strict connection tree T for W such that $|\mathsf{L}(T)| \leq \ell$ and $|\mathsf{R}(T)| \leq r = k + 4n + 4m$. Note that, by construction of H, the only path in H between the terminal vertices $w^1_{a_1}$

and $w^2_{a_1}$ contains the non-terminal vertices $u_1, u_2, \ldots, u_\ell$. Besides that, $d_H(u_i) = 2$ for every $i \in \{1, \ldots, \ell\}$. As a result, $\mathsf{L}(T) = \{u_1, u_2, \ldots, u_\ell\}$. This implies that all the other non-terminal vertices of T must be routers. In addition, for every $v_i \in V(G)$, we have that $a_i \in \mathsf{R}(T)$, since a_i is the only neighbour in H of the terminal vertices $w^1_{a_i}$ and $w^2_{a_i}$ and, thus, a_i necessarily belongs to $V(T)$. Analogously, we have that $b_i, c_i \in \mathsf{R}(T)$.

Claim 2.1. Let $v_i \in V(G)$ and T_i be the subgraph of T induced by $V(H_i)$. If $y_i \in V(T)$, then we can assume that the degree of y_i in T_i is at least 3.

Proof. First, we note that every path in H between y_i and a_i contains x_i or z_i. Consequently, x_i or z_i must belong to the path P in T between y_i and a_i. It is not hard to verify that, if $x_i \in V(P)$, then we can assume that $x_i, w^2_i, c_i \in N_T(y_i)$. On the other hand, if $z_i \in V(P)$, then we can assume that $z_i, w^2_i, c_i \in N_T(y_i)$. ■

Claim 2.2. Let $v_i \in V(G)$. If $x_i \in V(T)$, then we can assume that $y_i \in V(T)$. Analogously, if $y_i \in V(T)$, then we can assume that $x_i \in V(T)$.

Proof. Suppose that $x_i \in V(T)$ but $y_i \notin V(T)$. The case in which $y_i \in V(T)$ but $x_i \notin V(T)$ is analogous. Note that, the path in T between a_i and c_i must contain z_i, which must be a router of T. Furthermore, by the previous claim, we can assume that, for every vertex $y_j \in N_T(z_i)$ with $j \neq i$, the degree of y_j in T_j is at least 3, where T_j denotes the subgraph of T induced by $V(H_j)$. Thus, let T' be the graph with vertex set $V(T') = V(T) \setminus \{z_i\} \cup \{y_i\}$ and edge set

$$\begin{aligned} E(T') = E(T) \setminus \big(\{vz_i \mid v \in N_T(z_i)\} \cup \{w^2_i b_i\}\big) \\ \cup\, E(H_i - z_i) \cup \{y_j x_i \mid y_j \in N_T(z_i), j \neq i\}. \end{aligned}$$

One can verify that T' is a strict connection tree of H for W that simultaneously contains the vertices x_i and y_i and satisfies the constraints $|\mathsf{L}(T')| \leq \ell$ and $|\mathsf{R}(T')| \leq r$; more precisely, $\mathsf{L}(T') = \mathsf{L}(T)$ and $\mathsf{R}(T') = (\mathsf{R}(T) \setminus \{z_i\}) \cup \{y_i\}$. ■

Thus, consider the subset $S = \{v_i \in V(G) \mid x_i, y_i \in V(T)\}$. We claim that S is a vertex cover of G. For the sake of contradiction, suppose that there exists an edge $e = v_iv_j \in E(G)$ such that $S \cap \{v_i, v_j\} = \emptyset$. Consequently, $x_i, y_i \notin V(T)$ and $x_j, y_j \notin V(T)$. Moreover, we have that $w^1_{p_{ij}}, w^2_{p_{ij}}, w^1_{q_{ij}}, w^2_{q_{ij}} \notin V(T)$ (as well as $w^1_{p_{ji}}, w^2_{p_{ji}}, w^1_{q_{ji}}, w^2_{q_{ji}} \notin V(T)$), since T is connected and the only path in T between such terminals and any other terminal belonging to W — for example, the terminals in $W_1 \cup W_2$ — necessarily contains x_i or y_j (x_j or y_i, respectively). However, this contradicts the hypothesis that $W_3 \subseteq W \subseteq V(T)$. As a result, such an edge e cannot exist. In other words, S is a vertex cover of G. Finally, note that, if $|S| = k'$, then

$$|\mathsf{R}(T)| = 2k' + (n - k') + 3n + 4m = k' + 4n + 4m \leq r = k + 4n + 4m,$$

which implies $|S| \leq k$. Therefore, I is a yes-instance of Vertex cover. ■

Theorem 2.1. *S-TCP remains* NP*-complete when restricted to chordal bipartite graphs, even if ℓ is bounded by a constant.*

Proof. This result follows from Lemmas 2.1 and 2.2 and from the fact that the construction f can be computed in polynomial-time over the input size of the given instance I of Vertex cover and the parameter ℓ. ■

3 Concluding Remarks

In the present extended abstract, we have proved that S-TCP is NP-complete on chordal bipartite graphs even if ℓ is bounded by a constant. On the other hand, it remains unknown whether S-TCP can be solved in polynomial-time on chordal bipartite graphs if r is bounded by a constant (and ℓ is arbitrarily large). More generally, one of the main questions concerning S-TCP is whether the problem parameterized by r is in XP.

References

[1] M. C. Dourado, R. A. Oliveira, F. Protti, and U. S. Souza, *Conexão de terminais com número restrito de roteadores e elos*, In Proceedings of XLVI Simpósio Brasileiro de Pesquisa Operacional (2014), pp. 2965–2976.

[2] A. A. Melo, C. M. H. de Figueiredo, and U. S. Souza, *Connecting terminals using at most one router*, Matemática Contemporânea **45** (2017), SBM, pp. 49–57.

[3] A. A. Melo, C. M. H. de Figueiredo, and U. S. Souza, *A multivariate analysis of the strict terminal connection problem*, Journal of Computer and System Sciences **111** (2020), pp. 22–41.

[4] A. A. Melo, C. M. H. de Figueiredo, and U. S. Souza, *On undirected two-commodity integral flow, disjoint paths and strict terminal connection problems*, Networks **77** (2021), pp. 559–571.

[5] H. Müller and A. Brandstädt, *The NP-completeness of Steiner tree and dominating set for chordal bipartite graphs*, Theoretical Computer Science **53** (1987), pp. 257–265.

Alexsander Andrade de Melo
Federal University of Rio de Janeiro
Rio de Janeiro, Brazil.
`aamelo@cos.ufrj.br`

Celina Miraglia Herrera de Figueiredo
Federal University of Rio de Janeiro
Rio de Janeiro, Brazil.
`celina@cos.ufrj.br`

Uéverton dos Santos Souza
Fluminense Federal University
Niterói, Brazil.
`ueverton@ic.uff.br`

Appendix D

Manuscript: On the Computational Difficulty of the Terminal Connection Problem

This appendix contains the manuscript:
Alexsander A. de Melo, Celina M. H. de Figueiredo, Uéverton S. Souza. On the Computational Difficulty of the Terminal Connection Problem. Presented in the *47th International Conference on Current Trends in Theory and Practice of Computer Science* (SOFSEM 2021) [41], and submitted in March 2022 to *RAIRO - Theoretical Informatics and Applications.*

Theoretical Informatics and Applications Will be set by the publisher
Informatique Théorique et Applications

ON THE COMPUTATIONAL DIFFICULTY OF THE TERMINAL CONNECTION PROBLEM [*]

Alexsander A. de Melo[1], Celina M. H. de Figueiredo[1]
and Uéverton S. Souza[2]

Abstract. A *connection tree* of a graph G for a *terminal set* W is a tree subgraph T of G such that leaves$(T) \subseteq W \subseteq V(T)$. A non-terminal vertex is called *linker* if its degree in T is exactly 2, and it is called *router* if its degree in T is at least 3. The Terminal connection problem (TCP) asks whether G admits a connection tree for W with at most ℓ linkers and at most r routers, while the Steiner tree problem asks whether G admits a connection tree for W with at most k non-terminal vertices. We prove that, if $r \geq 1$ is fixed, then TCP is polynomial-time solvable when restricted to split graphs. This result separates the complexity of TCP from the complexity of Steiner tree, which is known to be NP-complete on split graphs. Additionally, we prove that TCP is NP-complete on strongly chordal graphs, even if $r \geq 0$ is fixed, whereas Steiner tree is known to be polynomial-time solvable. We also prove that, when parameterized by clique-width, TCP is W[1]-hard, whereas Steiner tree is known to be in FPT. On the other hand, agreeing with the complexity of Steiner tree, we prove that TCP is linear-time solvable when restricted to cographs (i.e. graphs of clique-width 2). Finally, we prove that, even if either $\ell \geq 0$ or $r \geq 0$ is fixed, TCP remains NP-complete on graphs of maximum degree 3.

2020 Mathematics Subject Classification. 68Q17, 68Q25, 68R10, 05C40, 05C85.

Keywords and phrases: Computational difficulty of problems, Parameterized complexity, Terminal vertices, Connection tree, Steiner tree, Split graphs, Strongly chordal graphs, Cographs, Bounded degree

[*] *This work was supported by the Brazilian agencies CAPES (Finance Code: 001), CNPq (Grant Numbers: 140399/2017-8, 407635/2018-1, 303726/2017-2) and FAPERJ (Grant Numbers: E-26/202.793/2017 and E-26/203.272/2017).*

[1] Federal University of Rio de Janeiro, Rio de Janeiro, Brazil; e-mail: `aamelo@cos.ufrj.br` & `celina@cos.ufrj.br`

[2] Fluminense Federal University, Niterói, Brazil; e-mail: `ueverton@ic.uff.br`

1. Introduction

Steiner tree is one of the most fundamental network design problems, proved to be NP-complete by Karp in his seminal paper [20]. Besides being related to several real-world applications, Steiner tree is of great theoretical interest, and it has been extensively studied from the perspective of graph theory [4,9,16,29,33] and computational complexity [2, 8, 12, 30]. The Steiner tree problem has as input a connected graph G, a non-empty terminal set $W \subseteq V(G)$, and a non-negative integer k, and it asks whether there exists a connected subgraph T of G such that $W \subseteq V(T)$ and $|V(T) \setminus W| \leq k$. Such a connected subgraph T admits a spanning tree with at most k non-terminal vertices. In this paper, we analyse the computational complexity of a network design problem closely related to Steiner tree, called Terminal connection.

Let G be a graph and $W \subseteq V(G)$ be a non-empty set. A *connection tree* T of G for W is a tree subgraph of G such that $\text{leaves}(T) \subseteq W \subseteq V(T)$, where $\text{leaves}(T)$ denotes the leaf set of T. In a connection tree T for W, the vertices belonging W are called *terminal*, and the vertices belonging to $V(T) \setminus W$ are called *non-terminal* and are classified into two types according to their respective degrees in T, namely: the non-terminal vertices of degree exactly 2 in T are called *linkers* and the non-terminal vertices of degree at least 3 in T are called *routers cf.* [10]. We remark that the vertex set of every connection tree can be partitioned into terminal vertices, linkers and routers. For each connection tree T, we let $\mathsf{L}(T)$ denote the linker set of T and $\mathsf{R}(T)$ denote the router set of T. Next, we present a formal definition for the Terminal connection problem.

Terminal Connection (TCP)

Input:	A connected graph G, a non-empty terminal set $W \subseteq V(G)$ and two non-negative integers ℓ and r.
Question:	Does there exist a connection tree T of G for W such that $\lvert\mathsf{L}(T)\rvert \leq \ell$ and $\lvert\mathsf{R}(T)\rvert \leq r$?

TCP was introduced by Dourado *et al.* [10], having as motivation applications in information security and network routing, and it was proved to be polynomial-time solvable when the parameters ℓ and r are both fixed [10]. Nevertheless, it was proved to be NP-complete even if either $\ell \geq 0$ or $r \geq 0$ is fixed [10]. In particular, the problem was proved to be NP-complete even if $\ell \geq 0$ is fixed and the input graph has constant maximum degree [11].

There is a straightforward Turing reduction from Steiner tree to TCP, namely: (G, W, k) is a yes-instance of Steiner tree if and only if (G, W, ℓ, r) is a yes-instance of TCP for some pair $\ell, r \in \{0, \ldots, k\}$ such that $\ell + r = k$. An interesting aspect of this Turing reduction is the fact that it preserves the structure of the input graph. Consequently, if TCP is polynomial-time solvable on some graph class $\mathcal{G}$, then so is Steiner tree. Analogously, if Steiner tree is NP-complete on some graph class $\mathcal{G}$, then TCP cannot be solved in polynomial-time on $\mathcal{G}$, unless P=NP. Nevertheless, if either $\ell \geq 0$ or $r \geq 0$ is fixed, then possibly TCP is polynomial-time solvable on a graph class $\mathcal{G}$, while STEINER TREE remains NP-complete on $\mathcal{G}$. In addition, there might exist a graph class $\mathcal{G}$ on which STEINER TREE is polynomial-time solvable whereas TCP remains NP-complete.

In this work, we confirm the existence of such complexity separating classes. In Section 2, we prove that, on *split* graphs, TCP is polynomial-time solvable if $r \geq 1$ is fixed, whereas STEINER TREE is known to be NP-complete [33]. Besides, we prove that, on *strongly chordal* graphs, TCP remains NP-complete even if $r \geq 0$ is fixed, whereas STEINER TREE is known to be polynomial-time solvable [33]. Also, we prove in Section 3.1 that, parameterized by clique-width, TCP is W[1]-hard, whereas STEINER TREE is known to be in FPT [1].

On the other hand, in Section 3.2, we prove that TCP can be solved in linear-time on *cographs* (i.e. graphs of clique-width 2), agreeing with the computational complexity of STEINER TREE [4]. Additionally, in Section 4, we prove that TCP remains NP-complete on graphs of maximum degree 3 even if either $\ell \geq 0$ or $r \geq 0$ is fixed. It is worth mentioning that, although STEINER TREE is known to be NP-complete on graphs of maximum degree 3 [22], our NP-completeness results of TCP with either $\ell \geq 0$ or $r \geq 0$ fixed do not immediately follow from the NP-completeness of STEINER TREE.

Table 1 summarises the mentioned results.

	Problem			
Graph class/Parameter	TCP	TCP fixed ℓ	TCP fixed r	STEINER TREE
Split	NP-c Thm. 2	NP-c Thm. 2	Poly, for $r \geq 1$ Thm. 1	NP-c [33]
Strongly chordal	NP-c Thm. 3	Open	NP-c Thm. 3	Poly [33]
Clique-width	W[1]-h Thm. 4	Open	W[1]-h Thm. 4	FPT [1]
Cographs	Poly Thm. 5	Poly Thm. 5	Poly Thm. 5	Poly [4]
Maximum degree 3	NP-c Thms. 6 and 7	NP-c Thm. 6	NP-c Thm. 7	NP-c [22]

TABLE 1. Comparison between the computational complexity of TCP with the computational complexity of STEINER TREE.

Related works. Motivated by applications in optical networks and bandwidth consumption minimization, another variant of STEINER TREE that has been investigated is the one in which the number of *branching nodes* in the sought tree T, i.e. vertices (which not necessarily are non-terminal) of degree at least 3 in T, is bounded. In [17, 31, 32], the authors addressed the undirected and directed cases of this variant, for which they devised approximation and parameterized tractable algorithms, apart from obtaining some intractability results.

In addition, Dourado *et al.* introduced in [11] the *strict* variant of TCP, called STRICT TERMINAL CONNECTION problem (S-TCP), which has the same input of TCP but further requires that the sought connection tree T satisfies leaves$(T) = W \subseteq V(T)$. It is worth mentioning that, just as TCP can be seen as a generalization of Steiner tree, S-TCP can be seen as a generalization of Full Steiner tree, which is a widely studied variant of Steiner tree [18, 21, 23]. Similarly to TCP, it was proved that S-TCP is polynomial-time solvable when the parameters $\ell \geq 0$ and $r \geq 0$ are both fixed [11], and that the problem is still NP-complete if $\ell \geq 0$ is fixed [11]. Nevertheless, except for the case $r \in \{0, 1\}$, which was shown to be polynomial-time solvable [24], the complexity of S-TCP for fixed $r \geq 2$ has remained open. Motivated by this question, S-TCP was also investigated in [25,26]. In particular, in [26], S-TCP was proved to be NP-complete (and W[2]-hard when parameterized by r), even if $\ell \geq 0$ is constant and the input graph is restricted to split graphs. An interesting fact of this proof is that it can be easily adapted to TCP. Consequently, we obtain that TCP is also NP-complete (and W[2]-hard when parameterized by r) on split graphs even if $\ell \geq 0$ is constant. Besides this result, it was analysed in [26] the complexity of S-TCP when restricted to graphs of bounded maximum degree, and it was also proved that S-TCP is polynomial-time solvable on cographs.

A previous version of this work appeared as an extended abstract at SOFSEM 2021 conference [27]. Besides the full proofs omitted in [27], the present paper contains further contributions, such as the tractability of TCP on split graphs and the W[1]-hardness of TCP parameterized by clique-width.

Graph notation. Now, we present some basic notation and terminologies of graph theory that are used throughout this paper. For any missing definition or terminology, we refer to [3].

In this work, all graphs are finite, simple and undirected. Let G be a graph. We let $V(G)$ and $E(G)$ denote the vertex set and the edge set of G, respectively. For every vertex $u \in V(G)$, we let $N_G(u)$ and $N_G[u] = N_G(u) \cup \{u\}$ denote the *(open) neighbourhood* and the *closed neighbourhood* of u in G, respectively; and we let $d_G(u) = |N_G(u)|$ denote the *degree* of u in G. Two distinct vertices $u, v \in V(G)$ are said to be *false twins* (resp. *true twins*) in G if in G if $N_G(u) = N_G(v)$ (resp. $N_G[u] = N_G[v]$). The *length* of a path P is defined as the number of edges of P. The *distance* between two vertices $u, v \in V(G)$ is the length of a path of G between u and v of minimum length. For every non-empty subset $S \subseteq V(G)$, we let $G[S]$ denote the *subgraph of G induced by S*.

Let $G_1, \ldots, G_k$ be $k \geq 2$ graphs. The *disjoint union* of $G_1, \ldots, G_k$ is the graph H, denoted by $G_1 \cup \cdots \cup G_k$, with vertex set $V(H) = V(G_1) \uplus \cdots \uplus V(G_k)$ and edge set $E(H) = E(G_1) \uplus \cdots \uplus E(G_k)$. The *join* of $G_1, \ldots, G_k$ is the graph H, denoted by $G_1 \wedge \cdots \wedge G_k$, with vertex set $V(H) = V(G_1 \cup \cdots \cup G_k)$ and edge set

$$E(H) = E(G_1 \cup \cdots \cup G_k) \uplus \{uv \mid u \in V(G_i),\ v \in V(G_j),\ i, j \in \{1, \ldots, k\},\ i \neq j\}.$$

2. Separating classes: Split and Strongly chordal

In this section, we present the main results of this work. First, we prove that, when restricted to split graphs, TCP is polynomial-time solvable if $r \geq 1$ is fixed.

Second, we prove that, when restricted to strongly chordal graphs, TCP is NP-complete even if $r \geq 0$ is fixed. Such results separate the complexity of TCP from the complexity of Steiner tree, since Steiner tree is known to be NP-complete on split graphs [33] and polynomial-time solvable on strongly chordal graphs [33].

2.1. Split graphs

A *split* graph is a graph whose vertex set can be partitioned into a clique and a stable set. In what follows, we prove the following theorem.

Theorem 1. *For $r \geq 1$, TCP can solved in time $n^{\mathcal{O}(r)}$ on split graphs.*

To prove Theorem 1, we propose a polynomial-time reduction from TCP, with $r \geq 1$, to its strict variant S-TCP, in which the terminal vertices are further required to coincide with the leaf set of the sought tree. S-TCP was shown to admit an $n^{\mathcal{O}(r)}$-time algorithm on split graphs [26]. In a nutshell, the algorithm presented in [26] enumerates each possible candidate router set $R \subseteq V(G) \setminus W$, with $|R| \leq r$, and then decides through matching techniques whether the input graph G admits a connection tree T for the terminal set W, such that $|\mathsf{L}(T)| \leq \ell$, $\mathsf{R}(T) = R$ and leaves$(T) = W$. Thus, combining our polynomial-time reduction with this algorithm, we obtain that TCP can be solved in time $n^{\mathcal{O}(r)}$ on split graphs for $r \geq 1$.

It is worth mentioning that this result is optimum, i.e. the $n^{\mathcal{O}(r)}$-time complexity cannot be considerably improved. Indeed, the following theorem immediately comes from a trivial adaptation of a parameterized polynomial-time reduction from the Set cover problem to S-TCP presented in [26] (see Theorem 7 of [26]).

Theorem 2 ([26]). *For any computable functions f and h, TCP cannot be solved in time $f(r) \cdot n^{h(\ell)}$, unless $\mathsf{FPT} = \mathsf{W}[2]$, and cannot be solved in time $f(r) \cdot n^{o(r)}$, unless ETH fails.*

In what follows, we write $G\langle K, S\rangle$ to refer a split graph G and explicitly denote that $K \cup S$ is a partition of the vertex set of G into a clique K and a stable set S.

Lemma 1. *Let $G\langle K, S\rangle$ be a split graph and $W \subseteq V(G)$. If $|W| \geq 3$, $W \cap K = \emptyset$ and there exists a connection tree T of G for W such that $\mathsf{R}(T) = \emptyset$, then there exists a connection tree T' of G for W such that $\mathsf{L}(T') \subseteq |\mathsf{L}(T)|$ and $|\mathsf{R}(T')| = 1$.*

Proof. Since $|W| \geq 3$ and $\mathsf{R}(T) = \emptyset$, there exists a terminal vertex $w \in W$ whose degree in T is at least 2. Then, let u and u' be two distinct neighbours of w in T. Since $W \cap K = \emptyset$, $u, u' \in \mathsf{L}(T) \cap K$. Let T' be the graph obtained from T by removing the edge wu' and adding the edge uu'. Clearly, T' is a connection tree of G for W such that $\mathsf{L}(T') = \mathsf{L}(T) \setminus \{u\}$ and $\mathsf{R}(T') = \mathsf{R}(T) \cup \{u\}$. □

Lemma 2. *Let $G\langle K, S\rangle$ be a split graph and $W \subseteq V(G)$ be a non-empty set. Suppose that G admits a connection tree T for W. There exists a connection tree T' of G for W, with $\mathsf{L}(T') \subseteq \mathsf{L}(T)$ and $|\mathsf{R}(T')| \leq |\mathsf{R}(T)|$, that simultaneously satisfies the following conditions:*

(1) $\mathsf{L}(T') \subseteq K$ *and* $\mathsf{R}(T') \subseteq K$;

(2) *If* $\mathsf{R}(T) \cap K \neq \emptyset$ *or* $W \cap K \neq \emptyset$*, then every vertex in* $W \cap S$ *is a leaf of* T'*.*

Proof. (1) Suppose that $(\mathsf{L}(T) \cup \mathsf{R}(T)) \cap S \neq \emptyset$. Then, there exists a vertex $u \in V(T) \cap K$. Let T' be the graph obtained from T as follows:

- Remove all vertices belonging to $(\mathsf{L}(T) \cup \mathsf{R}(T)) \cap S$ and their incident edges;
- For each $u' \in \mathsf{L}(T) \cap S$, add the edge vv', where $N_T(u') = \{v, v'\}$;
- For each $u' \in N_T(\mathsf{R}(T) \cap S)$, add the edge uu'.

Clearly, T' is a connection tree of G for W such that $\mathsf{L}(T') \subseteq K$ and $\mathsf{R}(T') \subseteq K$. Moreover, note that $\mathsf{L}(T') \subseteq \mathsf{L}(T) \setminus S$, $\mathsf{R}(T') = \mathsf{R}(T)$ if $\mathsf{R}(T) \cap S = \emptyset$, and $\mathsf{R}(T') \subseteq (\mathsf{R}(T) \cup \{u\}) \setminus S$ otherwise.

(2) Suppose that $W \cap S \neq \emptyset$ and that there exists a vertex $u \in (\mathsf{R}(T) \cup W) \cap K$. Note that, in this case, every vertex $w \in W \cap S$ has at least one neighbour, say $\alpha(w)$, in T. Then, let T' be the graph obtained from T as follows:

- For each $w \in W \cap S$, remove all edges of T that are incident to w except for $w\alpha(w)$; additionally, for each $v \in N_T(w)$, add the edge uv.

Clearly, T' is a connection tree of G for W such that every vertex in $W \cap S$ is a leaf of T'. Furthermore, one can verify that $\mathsf{L}(T') = \mathsf{L}(T)$ and $\mathsf{R}(T') = \mathsf{R}(T)$. □

Next, we present our polynomial-time reduction to S-TCP.

Construction 1 (Reduction from TCP to S-TCP on split graphs)**.** Let $G\langle K, S\rangle$ be a split graph and $I = (G, W, \ell, r)$ be an instance of TCP. If $W \cap K = \emptyset$, then we define our reduction instance of S-TCP as simply $g(I) = I$. Otherwise, let $\rho \in W \cap K$ and consider the graph G' obtained from G as follows (see Figure 1):

- Add all vertices and all edges of G;
- For each $u \in W \cap S \cap N_G(W \cap K) \setminus N_G(\rho)$, add the edge ρu;
- Add three new vertices w'_1, w'_2 and w'_3, and make them adjacent to ρ.

Note that G' is a split graph, and that $K \cup S'$ is a partition of $V(G')$ into a clique and a stable set, where $S' = S \cup \{w'_1, w'_2, w'_3\}$. We then define our reduction instance of S-TCP as $g(I) = (G', W', \ell, r+1)$, where $W' = (W \setminus \{\rho\}) \cup \{w'_1, w'_2, w'_3\}$.

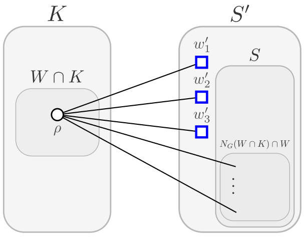


FIGURE 1. Split graph $G'\langle K, S'\rangle$ of the instance $g(I)$ of S-TCP described in Construction 1, obtained from a split graph $G\langle K, S\rangle$ of an instance I of TCP, with $K \cap W \neq \emptyset$.

The following lemma concludes the proof of Theorem 1.

Lemma 3. *Let $G\langle K,S\rangle$ be a split graph and $I=(G,W,\ell,r)$ be an instance of TCP such that $|W|\geq 3$. Also, let $g(I)$ be the instance of S-TCP obtained from I, as described in Construction 1. If $r\geq 1$ or $W\cap K\neq\emptyset$, then I is a* yes*-instance of TCP if and only if $g(I)$ is a* yes*-instance of S-TCP.*

Proof. First, suppose that I is a yes-instance of TCP. Then, there exists a connection tree T of G for W such that $|\mathsf{L}(T)|\leq\ell$ and $|\mathsf{R}(T)|\leq r$. By Lemma 1, we can assume that $\mathsf{R}(T)\neq\emptyset$ or $W\cap K\neq\emptyset$. Furthermore, by Lemma 2, we can assume that every vertex in $W\cap S$ is a leaf of T. This implies $W\setminus \text{leaves}(T)\subseteq K$. If $W\cap K=\emptyset$, then $\text{leaves}(T)=W$ and, therefore, we immediately obtain that $g(I)=I$ is a yes-instance of S-TCP. Thus, suppose that $W\cap K\neq\emptyset$. Additionally, by Lemma 2, assume that $\mathsf{L}(T)\subseteq K$ and $\mathsf{R}(T)\subseteq K$. Note that every vertex in $V(T)\cap S$ is a leaf of T. Since T is a tree and $\rho\in V(T)$, for each vertex $w\in W\cap K\setminus\{\rho\}$, there exists a single path between ρ and w in T and a single vertex in this path, say $\alpha(w)$, that belongs to $N_T(w)\cap K$. Thus, let T' be the graph obtained from T as follows:

- For each $w\in W\cap K\setminus\{\rho\}$ and each $w'\in N_T(w)\setminus\{\alpha(w)\}$, remove the edge ww' and add the edge $\rho w'$;
- For each $i\in\{1,2,3\}$, add the vertex w'_i and the edge $\rho w'_i$.

One can verify that T' is a connection tree of G' for W', such that $\text{leaves}(T')=W'$, $\mathsf{L}(T')=\mathsf{L}(T)$ and $\mathsf{R}(T)=\mathsf{R}(T')\cup\{\rho\}$.

Conversely, suppose that $g(I)$ is a yes-instance of S-TCP. If $W\cap K=\emptyset$, then $g(I)=I$ and, therefore, I is a yes-instance of TCP. Thus, suppose that $W\cap K\neq\emptyset$, and let T' be a connection tree of G' for W', such that $\text{leaves}(T')=W'$, $|\mathsf{L}(T')|\leq\ell$ and $|\mathsf{R}(T')|\leq r+1$. Since the only neighbour of the terminal vertices w'_1, w'_2 and w'_3 in G' is the vertex ρ, we have that ρ necessarily belongs to T' and, besides that, is a router of T'. Moreover, by construction of G', if a vertex w is a neighbour of ρ in T' but is not a neighbour of ρ in G, then $w\in W\cap S$ and there exists a vertex in $W\cap K$, say $\beta(w)$, which is a neighbour of w in G. Then, let T be the graph obtained from T' as follows:

- Remove the vertices w'_1, w'_2 and w'_3 and their incident edges;
- For each $w\in N_{T'}(\rho)\setminus N_G(\rho)$, remove the edge ρw and add the edge $\beta(w)w$.

One can verify that T is a connection tree of G for W, such that $\mathsf{L}(T)=\mathsf{L}(T')$ and $\mathsf{R}(T)=\mathsf{R}(T')\setminus\{\rho\}$. □

2.2. Strongly chordal graphs

A *chord* of a cycle C is an edge between any two non-consecutive vertices of C. A graph G is called *chordal* if every cycle of G of length at least 4 has a chord. In other words, a graph G is chordal if every induced cycle of G has length 3. An *even cycle* is a cycle of even length. A chord uv of an even cycle C is called an *odd chord* if the distance between u and v in C is odd. A graph G is called *strongly chordal* if it is chordal and every even cycle of G of length at least 6 has an odd chord.

A vertex u of a graph G is called a *simple vertex* if, for any two vertices $v, v' \in N_G(u)$, $N_G[v] \subseteq N_G[v']$ or $N_G[v'] \subseteq N_G[v]$. In other words, a vertex u of a graph G is simple if the collection $\{N_G[v] \mid v \in N_G(u)\}$ can be linearly ordered by set inclusion. Farber [13] proved that a graph G is strongly chordal if and only if there exists a linear ordering $(u_1, \ldots, u_n)$ of the vertices of G, called *simple elimination ordering*, such that u_i is a simple vertex of $G[\{u_i, \ldots, u_n\}]$ for each $i \in \{1, \ldots, n\}$.

We prove that TCP remains NP-complete on strongly chordal graphs:

Theorem 3. *For each $r \geq 0$, TCP remains* NP*-complete when restricted to strongly chordal graphs.*

In order to prove Theorem 3, we provide a polynomial-time reduction from the Hamiltonian path problem, which was shown to be NP-complete on strongly chordal graphs by Müller [28]. The Hamiltonian path problem has as input a graph G and asks whether G admits a *Hamiltonian path*, i.e. a path that contains all vertices of G.

The next lemma presents some important properties of the class of strongly chordal graphs, which are used in our reduction.

Lemma 4. *The class of strongly chordal graphs is closed under the following operations:*

(1) *Adding true twin vertices;*
(2) *For any pair of true twin vertices v and v', adding a new vertex w and adding the edges vw and wv'.*

Proof. Let G be a strongly chordal graph and $(u_1, \ldots, u_n)$ be a simple elimination ordering of G. For each $i \in \{1, \ldots, n\}$, let G_i denote $G[\{u_i, \ldots, u_n\}]$.

(1) Let H be the graph obtained from G by adding a true twin v of u_i, for some $i \in \{1, \ldots, n\}$. We claim that

$$(u_1, \ldots, u_i, v, u_{i+1} \ldots, u_n)$$

is a simple elimination ordering of H. First, we show that v is a simple vertex of H_v, where H_v denotes $H[\{v, u_{i+1}, \ldots, u_n\}]$. Since u_i and v are true twins in H, $N_{H_v}[x] = (N_{G_i}[x] \setminus \{u_i\}) \cup \{v\}$ and $N_{H_v}[y] = (N_{G_i}[y] \setminus \{u_i\}) \cup \{v\}$ for every pair $x, y \in N_{H_v}(v)$. Moreover, since u_i is a simple vertex of G_i, we have that $N_{G_i}[x] \subseteq N_{G_i}[y]$ or $N_{G_i}[y] \subseteq N_{G_i}[x]$ for every pair $x, y \in N_{G_i}(v)$. Finally, we remark that $N_{H_v}(v) = N_{G_i}(u_i)$. Then, let $x, y \in N_{H_v}(v)$ and assume without loss of generality that $N_{G_i}[x] \subseteq N_{G_i}[y]$. One can verify that

$$N_{H_v}[x] = (N_{G_i}[x] \setminus \{u_i\}) \cup \{v\} \subseteq (N_{G_i}[y] \setminus \{u_i\}) \cup \{v\} = N_{H_v}[y].$$

Therefore, v is indeed a simple vertex of H_v.

Now, let $j \in \{1, \ldots, n\}$. We prove that v_j is a simple vertex of H_j, where H_j denotes $H[\{u_j, \ldots, u_i, v, u_{i+1}, \ldots, u_n\}]$ if $j \leq i$, and $H[\{u_j, \ldots, u_n\}]$ otherwise. Note that, if $j \geq i + 1$, then u_j is trivially a simple vertex of H_j, since in this case

$H_j = G_j$. Thus, assume that $j \leq i$. One can verify that, for every $x \in V(H_j)\backslash\{v\}$,

$$N_{H_j}[x] = \begin{cases} N_{G_j}[x] \cup \{v\} & \text{if } u_i \in N_G(x) \\ N_{G_j}[x] & \text{otherwise.} \end{cases}$$

Let $x, y \in N_{H_j}(u_j)$. We prove that $N_{H_j}[x] \subseteq N_{H_j}[y]$ or $N_{H_j}[y] \subseteq N_{H_j}[x]$, implying that u_j is indeed a simple vertex of H_j. First, suppose that $y = v$. Note that, if $N_{G_j}[x] \subseteq N_{G_j}[u_i]$, then

$$N_{H_j}[x] = N_{G_j}[x] \cup \{v\} \subseteq N_{G_j}[u_i] \cup \{v\} = N_{H_j}[u_i] = N_{H_j}[v].$$

On the other hand, if $N_{G_j}[u_i] \subseteq N_{G_j}[x]$, then

$$N_{H_j}[v] = N_{H_j}[u_i] = N_{G_j}[u_i] \cup \{v\} \subseteq N_{G_j}[x] \cup \{v\} = N_{H_j}[x].$$

Now, suppose that $x \neq v$ and $y \neq v$. Assume without loss of generality that $N_{G_j}[x] \subseteq N_{G_j}[y]$. Note that, if $u_i \in N_{G_j}(x)$, then $u_i \in N_{G_j}(y)$ and, thus,

$$N_{H_j}[x] = N_{G_j}[x] \cup \{v\} \subseteq N_{G_j}[y] \cup \{v\} = N_{H_j}[y].$$

On the other hand, if $u_i \notin N_G(x)$, then

$$N_{H_j}[x] = N_{G_j}[x] \subseteq N_{G_j}[y] \subseteq N_{H_j}[y].$$

(2) Let H be the graph obtained from G by adding a new vertex w and adding the edges vw and $v'w$, where v and v' are true twins of G. Since $N_H(w) = \{v, v'\}$ and $N_H[v] = N_H[v']$, it is immediate that w is a simple vertex of $H[\{w, u_1, \ldots, u_n\}]$. Moreover, for every $i \in \{1, \ldots, n\}$, u_i is a simple vertex of $G[\{u_i, \ldots, u_n\}]$. Thus,

$$(w, u_1, \ldots, u_i, \ldots, u_n)$$

is a simple elimination ordering of H, and therefore, H is strongly chordal. □

Construction 2 (Gadget H_r and Terminal Set W_r)**.** Let r be a positive integer. We define the gadget H_r as the graph such that (see Figure 2)

$$V(H_r) = \{\rho_1, \ldots, \rho_r\} \cup \{x_1^1, x_1^2\} \cup \{x_i \mid i \in \{2, \ldots, r\}\} \text{ and}$$
$$E(H_r) = \{\rho_i\rho_{i+1} \mid i \in \{1, \ldots, r-1\}\} \cup \{x_1^1\rho_1, x_1^2\rho_1\} \cup \{x_i\rho_i \mid i \in \{2, \ldots, r\}\}.$$

Moreover, we let $W_r = \{x_1^1, x_1^2\} \cup \{x_2, \ldots, x_r\}$ be the terminal set of H_r.

Construction 3 (Reduction from Hamiltonian path to TCP)**.** Let G be a graph, with vertex set $V(G) = \{u_1, \ldots, u_n\}$, and r be a non-negative integer. We let G' be the graph obtained from G and r as follows (see Figure 3):

- Add all vertices and all edges of G to G';

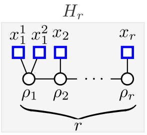


FIGURE 2. Gadget H_r for $r \geq 1$, described in Construction 2.

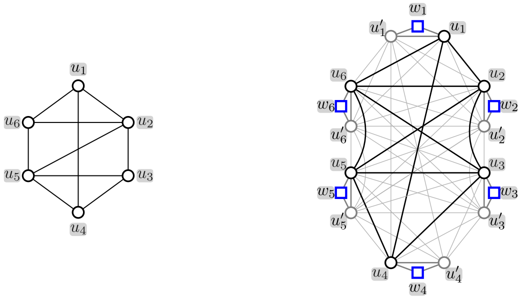


FIGURE 3. A graph G and the graph G' obtained from G (and $r = 0$) as described in Construction 3.

- For each vertex $u_i \in V(G)$, add a true twin u_i' of u_i, in such a way that $N_{G'}[u_i'] = N_{G'}[u_i]$;
- For each vertex $u_i \in V(G)$, add a new vertex w_i and add the edges u_iw_i and $u_i'w_i$, where u_i' denotes the true twin of u_i added in the last step;
- If $r \geq 1$, create the gadget H_r and define the terminal set W_r as described in Construction 2, besides adding the edge $\rho_r w_1$; otherwise, if $r = 0$, define $W_r = \emptyset$.

We then define our reduction instance of TCP as $g(G, r) = (G', W, \ell, r)$, where $W = \{w_1, \ldots, w_n\} \cup W_r$ and $\ell = 2n - 2$.

We remark that, the graph G' described in Construction 3 is similar to the one constructed in [10] to prove the NP-completeness of TCP on general graphs for fixed $r \geq 0$. The main difference is the fact that, in the graph constructed in [10], for each $u_i \in V(G)$, it is added a false twin, instead of a true twin, of u_i. However, this makes the original graph not be strongly chordal, even if the input graph is strongly chordal; for instance, a cycle C_3 of length 3 is strongly chordal, but the graph resulting from adding a false twin for each vertex of C_3 is not strongly chordal, since it contains an induced cycle of length 4. The next lemma, which states that, whenever the input graph is strongly chordal, our constructed graph is strongly chordal as well, immediately follows from Lemma 4 and from the fact that the vertices of H_r are not contained in any cycle of G'.

Lemma 5. *Let G be a graph and r be a non-negative integer. Also, let G' be the graph of the instance $g(G, r)$ of TCP obtained from G and r, as described in Construction 3. If G is strongly chordal, then so is G'.*

Lemma 6. *Let G be a graph and r be a non-negative integer. Also, let $g(G, r)$ be the instance of TCP obtained from G and r, as described in Construction 3. Then, G admits a Hamiltonian path if and only if $g(G, r)$ is a* yes*-instance of TCP.*

Proof. Assume that $V(G) = \{u_1, \ldots, u_n\}$ and that $g(G, r) = (G', W, \ell, r)$. Additionally, for simplicity, consider $W_r = V(H_r) = E(H_r) = \emptyset$ if $r = 0$.

First, suppose that there exists in G a Hamiltonian path $(u_{j_1}, \ldots, u_{j_n})$. Then, let T be the graph with vertex set

$$V(T) = V(H_r) \cup V(P) \cup \{w_{j_1}, u'_{j_1}, u_{j_n}, w_{j_n}\} \cup \{u_{j_i}, w_{j_i}, u'_{j_i} \mid i \in \{2, \ldots, n-1\}\}$$

and edge set

$$\begin{aligned} E(T) = {} & E(H_r) \cup \{\rho_r w_1, w_1 u'_1\} \\ & \cup \{u'_{j_{i-1}} u_{j_i}, u_{j_i} w_{j_i}, w_{j_i} u'_{j_i} \mid i \in \{2, \ldots, n-1\}\} \cup \{u'_{j_{n-1}} u_{j_n}, u_{j_n} w_{j_n}\}, \end{aligned}$$

where u'_{j_i} denotes the true twin of u_{j_i} added in the construction of G'. Note that T is a connection tree of G' for W with $\mathsf{L}(T) = \{u'_{j_1}, u_{j_n}\} \cup \{u_{j_2}, u'_{j_2}, \ldots, u_{j_{n-1}}, u'_{j_{n-1}}\}$ and $\mathsf{R}(T) = \{\rho_1, \ldots, \rho_r\}$. Therefore, $g(G, r)$ is a yes-instance of TCP.

Conversely, suppose that $g(G, r)$ is a yes-instance of TCP. Let T be a connection tree of G' for W such that $|\mathsf{L}(T)| \leq 2n-2$ and $|\mathsf{R}(T)| \leq r$. We remark that ρ_1 is the only neighbour of the terminal vertices $x_1^1, x_1^2 \in W_r$ and, for each $i \in \{2, \ldots, r\}$, ρ_i is the only neighbour of the terminal vertex $x_i \in W_r$. As a result, T must contain all the vertices $\rho_1, \ldots, \rho_r$. More specifically, such vertices must be routers of T. This implies that $T' = T - H_r$ cannot contain any router, and all non-terminal vertices of T' must be linkers. Hence, T' is a path, since, by construction of G', w_i has degree at most 2 in T' for every $i \in \{1, \ldots, n\}$. Then, let $P' = (w_{j_1}, \ldots, w_{j_n})$ be a sequence of distinct vertices such that, for each $i \in \{1, \ldots, n-1\}$, the path in T' between w_{j_i} and $w_{j_{i+1}}$ does not contain any other terminal vertex. Note that, since $|\mathsf{L}(T)| \leq \ell = 2n-2$, every path in T' between any two consecutive vertices w_{j_i} and $w_{j_{i+1}}$ in P' must be of one of the forms: $(w_{j_i}, u'_{j_i}, u'_{j_{i+1}}, w_{j_{i+1}})$, $(w_{j_i}, u'_{j_i}, u_{j_{i+1}}, w_{j_{i+1}})$, $(w_{j_i}, u_{j_i}, u_{j_{i+1}}, w_{j_{i+1}})$, or $(w_{j_i}, u_{j_i}, u'_{j_{i+1}}, w_{j_{i+1}})$. As a result, it follows from the construction of G' that, for each $i \in \{1, \ldots, n-1\}$, u_{j_i} and $u_{j_{i+1}}$ are adjacent in G. Therefore, $(u_{j_1}, \ldots, u_{j_n})$ is a Hamiltonian path of G. □

3. Graphs of bounded clique-width

In this section, we prove that TCP parameterized by the clique-width of the input graph is W[1]-hard. Similarly to the results presented in Section 2, this contrasts with the complexity Steiner tree, since Steiner tree is known to be in FPTwhen parameterized by clique-width [1]. On the other hand, agreeing with the complexity of Steiner tree, we prove that TCP is linear-time solvable on cographs, which are precisely the graphs of clique-width 2.

The notion of clique-width was introduced by Courcelle, Engelfriet and Rozenberg [7], and it is one of the most studied graph parameters. Next, we present the definition of this notion *cf.* [14,15].

Let k be a positive integer. A graph is called a *k-graph* if its vertices are labelled with integers in $\{1,\ldots,k\}$. An *initial k-graph* is a k-labelled graph on a single vertex. The *clique-width* of a graph G, denoted by $\mathsf{cwd}(G)$, is the smallest positive integer k such that G can be constructed by repeated application of the following four operations:

(1) *introducing* (denoted by $\mathsf{int}(u,i)$): construction of an initial k-graph, whose single vertex u is labelled by an integer $i \in \{1,\ldots,k\}$ and has not been introduced yet;
(2) *disjoint union* (here, denoted by $\oplus$);
(3) *relabelling* (denoted by $\mathsf{rel}_{i,j}$): changing all labels i to j, for $i,j \in \{1,\ldots,k\}$;
(4) *join* (denoted by $\eta_{i,j}$): connecting all vertices labelled by i with all vertices labelled by j, for $i,j \in \{1,\ldots,k\}$, $i \neq j$.

A construction of a graph G using the operations (1)-(4) described above can be represented by an algebraic term, called *cwd-expression* defining G, composed of int, $\oplus$, $\mathsf{rel}_{i,j}$, and $\eta_{i,j}$ *cf.* [14], where i and j are distinct positive integers. Note that cwd-expressions define a tree language, where each expression can be represented by a rooted tree T *cf.* [15], where each $\mathsf{int}(u,i)$ of the expression is associated with a leaf of T, and each vertex of G is introduced exactly once. A *k-expression* is a cwd-expression that contains at most k distinct labels *cf.* [14]. Thus, one can verify that, a graph G has clique-width at most k if and only if there exists a k-expression defining G.

3.1. Parameterization by clique-width

Now, we prove the following theorem.

Theorem 4. *For each $r \geq 0$, TCP parameterized by clique-width is* $\mathsf{W}[1]$*-hard.*

More specifically, we show that, if a graph G has clique-width at most k for some $k \geq 2$, then the graph G' obtained from G as described in Construction 3 has clique-width at most $k+1$. This, along with Lemma 6 and the fact that Hamiltonian path is $\mathsf{W}[1]$-hard parameterized by clique-width [15], implies the $\mathsf{W}[1]$-hardness of TCP.

The following lemma is a well-known fact, and it can be immediately verified by an inductive argument on the number of vertices of the tree.

Lemma 7. *Every tree has clique-width at most* 3*. Moreover, if T is a tree and u is a leaf of T, then there exists a* 3*-expression defining a construction of T in which at the root all vertices but u have the same label.*

Lemma 8. *Let G be a graph. For each $r \geq 0$, if $\mathsf{cwd}(G) = k$ for some $k \geq 2$, then $\mathsf{cwd}(G') \leq k+1$, where G' denotes the graph obtained from G and r as described in Construction 3.*

Proof. Assume that $V(G) = \{u_1, \ldots, u_n\}$ and $\mathsf{cwd}(G) \leq k$. Then, let γ_G be a k-expression defining G. Also, let H' be the subgraph of G' induced by $V(H_r) \cup \{w_1\}$. Note that H' is a tree. Thus, by Lemma 7, there exists a construction (3-expression) of a vertex-labelled copy of H' (for short γ'_H) in which all vertices but w_1 have the same label. Assume, without loss of generality, that w_1 is labelled by 1 at the root of $\gamma_{H'}$, and that all the other vertices of $\gamma_{H'}$ are labelled by 2. In what follows, we show that we can obtain from γ_G and $\gamma_{H'}$ a $(k+1)$-expression $\gamma_{G'}$ defining our constructed graph G'. We recall that each vertex $u_i \in V(G)$ has a true twin u'_i in G', and that $N_{G'-H_r}(w_i) = \{u_i, u'_i\}$. Consider $b = k+1$. We define $\gamma_{G'}$ as the cwd-expression obtained from γ_G as follows:

- Let $\mathsf{int}(u_1, i)$ be the leaf term of u_1 in γ_G, for $i \in \{2, \ldots, k\}$. Replace the occurrence of $\mathsf{int}(u_1, i)$ in γ_G with

$$\mathsf{rel}_{1,b}\Big(\eta_{i,1}\Big(\mathsf{rel}_{b,i}\big(\eta_{i,b}(\mathsf{int}(u_1,i), \mathsf{int}(u'_1,b))\big), \mathsf{rel}_{2,b}(\gamma_{H'})\Big)\Big),$$

  For $i = 1$ is similar (just replace the occurrences of 1 by 2 and vice versa).
- For each $u_j \in V(G) \setminus \{u_1\}$, if $\mathsf{int}(u_j, i)$ is the leaf term of u_j in γ_G, replace the occurrence of $\mathsf{int}(u_j, i)$ with

$$\eta_{i,b}\Big(\mathsf{rel}_{b,i}\big(\eta_{i,b}(\mathsf{int}(u_j,i), \mathsf{int}(u'_j,b))\big), \mathsf{int}(w_j,b)\Big).$$

We recall that, besides being represented by leaves, each vertex is introduced exactly once in a expression tree. Moreover, we note that the operations described above consists in local replacements in the corresponding leaves of the expression tree associated to γ_G. Thus, one can verify that $\gamma_{G'}$ defines G'. In addition, it is straightforward that $\gamma_{G'}$ is a $(k+1)$-expression, whenever $k \geq 2$. Therefore, $\mathsf{cwd}(G') \leq k+1$. □

3.2. Cographs

A *cograph* is a graph that does not contain a path of length 3 as an induced subgraph. Alternatively, cographs are characterized by the following recursive definition, given by Corneil *et al.* [5]:

- A graph on a single vertex is a cograph;
- If $G_1, \ldots, G_k$ are cographs, then so is their *disjoint union* $G_1 \cup \cdots \cup G_k$;
- If G is a cograph, then so is its complement $\overline{G}$.

We note that, if G is a connected cograph on more than one vertex, then there exist $k \geq 2$ cographs $G_1, \ldots, G_k$ such that G is their *join* $G_1 \wedge \cdots \wedge G_k$. Moreover, it is straightforward that a graph is a cograph if and only if its clique-width is exactly 2.

A key algorithmic property of cographs is the fact that, up to isomorphism, each cograph G can be uniquely represented by a rooted tree $\mathcal{T}_G$, called *cotree* [5], which can be seen as a specialization of a 2-expression defining G. The leaves of $\mathcal{T}_G$ correspond to the vertices of G, and each internal node u of $\mathcal{T}_G$ represents either the disjoint union or the join operation of the respective cographs induced by the leaves of the subtrees of $\mathcal{T}_G$ rooted at each child of u. Another important property is that, given a graph G, recognising G as a cograph, as well as obtaining its respective cotree (if any), can be performed in time linear in the number of vertices and the number of edges of G [6].

Let $I = (G, W, \ell, r)$ be an instance of TCP, where G is a cograph. Since TCP can be easily solved in linear-time if $|W| < 3$ or $G[W]$ is connected, we assume throughout this section that $|W| \geq 3$ and $G[W]$ is not connected. Moreover, we assume that G is connected and, therefore, is the join of $k \geq 2$ cographs $G_1, \ldots, G_k$.

Lemma 9. *Let G be a cograph that is the join of $k \geq 2$ cographs $G_1, \ldots, G_k$, and let $W \subseteq V(G)$ be a terminal set such that $|W| \geq 3$ and $G[W]$ is not connected. There exists a unique $i \in \{1, \ldots, k\}$ such that $V(G_i) \cap W \neq \emptyset$. Moreover, G admits a connection tree for W that contains exactly one router and no linker.*

Proof. For the sake of contradiction, suppose that, for some $i, j \in \{1, \ldots, k\}$ with $i \neq j$, $V(G_i) \cap W \neq \emptyset$ and $V(G_j) \cap W \neq \emptyset$. Then, let $u \in V(G_i) \cap W$, $v \in V(G_j) \cap W$, and let T be the graph with vertex set $V(T) = W$ and edge set $E(T) = \{uw \mid w \in W \setminus V(G_i)\} \cup \{vw \mid w \in V(G_i) \cap W\}$. Clearly, T is a connected subgraph of $G[W]$. Therefore, there exists a unique $i \in \{1, \ldots, k\}$ such that $V(G_i) \cap W \neq \emptyset$. This implies that $V(G_j) \cap W = \emptyset$ for some $j \in \{1, \ldots, k\} \setminus \{i\}$. Then, let $u' \in V(G_j)$ and T' be the graph with vertex set $V(T') = \{u'\} \cup W$ and edge set $E(T') = \{u'w \mid w \in W\}$. One can verify that T' is a connection tree of G for W such that $\mathsf{L}(T') = \emptyset$ and $\mathsf{R}(T') = \{u'\}$. □

Considering the input graph G as the join of $k \geq 2$ cographs $G_1, \ldots, G_k$, it follows from Lemma 9 that TCP can be trivially solved if $r \geq 1$, or $V(G_i) \cap W \neq \emptyset$ and $V(G_j) \cap W \neq \emptyset$ for some $i, j \in \{1, \ldots, k\}$, with $i \neq j$. Thus, we dedicate the remainder of this section to resolve the case in which $r = 0$ and there exists a unique $i \in \{1, \ldots, k\}$ such that $V(G_i) \cap W \neq \emptyset$.

Lemma 10. *Let G be a cograph and $W \subseteq V(G)$ be a non-empty terminal set. If T is a connection tree of G for W such that $\mathsf{R}(T) = \emptyset$ and $|\mathsf{L}(T)|$ is minimum, then $N_T(u) \subseteq W$ for each $u \in \mathsf{L}(T)$.*

Proof. For the sake of contradiction, suppose that $N_T(u) \not\subseteq W$ for some linker $u \in \mathsf{L}(T)$. Since $\mathsf{R}(T) = \emptyset$ and leaves$(T) \subseteq W$, u belongs to a path P of T between two terminal vertices $w, w' \in W$, such that $(V(P) \setminus \{w, w'\}) \cap W = \emptyset$. Thus, it follows from the assumption $N_T(u) \not\subseteq W$ that $|V(P)| \geq 4$. Since cographs do not contain paths of length 3 as induced subgraphs, there exists a path P' of G between w and w' such that $|V(P')| \leq 3$ and $V(P') \subseteq V(P)$. Then, let T' be the graph with vertex set $V(T') = (V(T) \setminus V(P)) \cup V(P')$ and edge set $E(T') = (E(T) \setminus E(P)) \cup E(P')$. One can easily verify that T' is a connection tree of G for W such that $\mathsf{R}(T) = \emptyset$ and $\mathsf{L}(T') \subsetneq \mathsf{L}(T)$, which contradicts the minimality of $|\mathsf{L}(T)|$. □

For each graph G, we let $\mathsf{cc}(G)$ denote the set of connected components of G, and we let $o(G) = |\mathsf{cc}(G)|$ denote the number of connected components of G.

Corollary 1. *Let G be a cograph, $W \subseteq V(G)$ be a non-empty terminal set, and let T be a connection tree of G for W such that $\mathsf{R}(T) = \emptyset$. If $|\mathsf{L}(T)|$ is minimum, then $|\mathsf{L}(T)| = o(G[W]) - 1$.*

Proof. Since $\mathsf{R}(T) = \emptyset$, it is straightforward that $|\mathsf{L}(T)| \geq o(G[W]) - 1$. On the other hand, it follows from Lemma 10 that, for each $u \in \mathsf{L}(T)$, $N_T(u) \subseteq W$. In addition, we note that, if $u \in \mathsf{L}(T)$ and $N_T(u) = \{w, w'\}$, then w and w' belong to distinct connected components of $G[W]$, otherwise the path (w, u, w') of T could be replaced by a shortest path of $G[W]$ between w and w', yielding a connection tree T' of G for W such that $\mathsf{L}(T') \subsetneq \mathsf{L}(T)$. Therefore, $|\mathsf{L}(T)| \leq o(G[W]) - 1$. □

Corollary 1 establishes that, whenever a cograph G admits a connection tree for a non-empty terminal set $W \subseteq V(G)$ that does not contain routers, G admits a connection tree T for W such that $\mathsf{R}(T) = \emptyset$ and $\mathsf{L}(T) = o(G[W]) - 1$. More importantly, it establishes that $o(G[W]) - 1$ is the minimum possible number of linkers that such a tree T can have. Therefore, if $I = (G, W, \ell, r)$ is an instance of TCP such that G is a cograph and $r = 0$, then ℓ must be at least $o(G[W]) - 1$, otherwise I is certainly a no-instance of the problem.

A *connection forest* of a graph G for a non-empty terminal set W is a subgraph F of G such that F is a forest and $\bigcup_{T \in \mathsf{cc}(F)} \text{leaves}(T) \subseteq W \subseteq V(F)$. A connection forest F is said to be *routerless* if $\bigcup_{T \in \mathsf{cc}(F)} \mathsf{R}(T) = \emptyset$. For each graph G and each non-empty terminal $W \subseteq V(G)$, we let

$$\lambda[G, W] = \min\{o(F) \mid F \text{ is a routerless connection forest of } G \text{ for } W\}.$$

As a degenerate case, we define $\lambda[G, \emptyset] = 0$.

We note that $\lambda[G, W] = 1$ if and only if G admits a connection tree of G for W such that $\mathsf{R}(T) = \emptyset$.

Lemma 11. *Let G be a cograph and $W \subseteq V(G)$ be a terminal set. If G is the disjoint union of $k \geq 2$ cographs $G_1, \ldots, G_k$, then*

$$\lambda[G, W] = \sum_{i \in \{1, \ldots, k\}} \lambda[G_i, V(G_i) \cap W].$$

Proof. Since G is the disjoint union of $G_1, \ldots, G_k$, there is no edge between the vertices of G_i and the vertices of G_j for any $i, j \in \{1, \ldots, k\}$, with $i \neq j$. Thus, $\lambda[G, W] \geq \sum_{i \in \{1, \ldots, k\}} \lambda[G_i, V(G_i) \cap W]$. On the other hand, for each $i \in \{1, \ldots, k\}$ with $V(G_i) \cap W \neq \emptyset$, let F_i be a routerless connection forests of G_i for $V(G_i) \cap W$ with the minimum number of connected components. One can readily verify that $F = F_1 \cup \cdots \cup F_k$ is a routerless connection forests of G for W. Therefore, $\lambda[G, W] \leq \sum_{i \in \{1, \ldots, k\}} \lambda[G_i, V(G_i) \cap W]$. □

Lemma 12. *Let G be a cograph and $W \subseteq V(G)$ be a terminal set. If G is the join of $k \geq 2$ cographs $G_1, \ldots, G_k$ and there exists a unique $i \in \{1, \ldots, k\}$ such that $V(G_i) \cap W \neq \emptyset$, then*

$$\lambda[G, W] = \max\{1, \lambda[G_i, W] - n + n_i\},$$

where $n = |V(G)|$ and $n_i = |V(G_i)|$.

Proof. Let F be a routerless connection forest of G for W. Since $W \subseteq V(G_i)$, $d_F(u) = 2$ for each $u \in V(F) \setminus V(G_i)$. This implies that, for each $u \in V(G) \setminus V(G_i)$, there at most two distinct connected components of G_i that are connected in F by u. In other words, if T is the connected component of F that contains $u \in V(G) \setminus V(G_i)$, then $o(T - u) \leq 2$. Thus,

$$\lambda[G, W] \geq \max\{1, \lambda[G_i, W] - |V(G) \setminus V(G_i)|\}.$$

On the other hand, let F_i be a routerless connection forest of G_i for W with the minimum number of connected components, i.e. $o(F_i) = \lambda[G_i, W]$, and let $S \subseteq V(G) \setminus V(G_i)$ such that $|S| = \min\{|V(G) \setminus V(G_i)|, o(F_i) - 1\}$. Also, let $T \in \mathsf{cc}(F_i)$, $w_T \in V(T) \cap W$ and $\alpha \colon S \to \mathsf{cc}(F_i) \setminus \{T\}$ be an injective map. Additionally, let $w_H \in V(H) \cap W$ for each $H \in \mathsf{cc}(F_i) \setminus \{T\}$. We define F as the graph (see Figure 4) with vertex set $V(F) = V(F_i) \cup S$ and edge set

$$E(F) = E(F_i) \cup \{w_T u, u w_H \mid u \in S, H \in \mathsf{cc}(F_i) \setminus \{T\}, \alpha(u) = H\}.$$

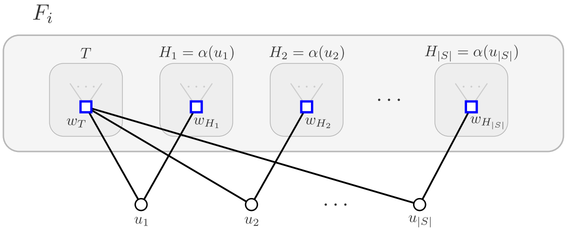


FIGURE 4. Graph F, with $S = \{u_1, \ldots, u_{|S|}\}$ and $\alpha(u_l) = H_l$ for each $l \in \{1, \ldots, |S|\}$.

One can verify that F is as routerless connection forest of G for W such that $o(F) = \lambda[G_i, W] - |S| = \max\{1, \lambda[G_i, W] - |V(G) \setminus V(G_i)|\}$. This implies that

$$\lambda[G, W] \leq \max\{1, \lambda[G_i, W] - n + n_i\},$$

concluding the proof. □

Theorem 5. *TCP is linear-time solvable on cographs.*

Proof. Let $I = (G, W, \ell, r)$ be an instance of TCP, where G is a cograph on n vertices and m edges. Assume without loss of generality that $|W| \geq 3$, G is connected but $G[W]$ is not connected. Moreover, based on Lemma 9 and on Corollary 1, assume that $r = 0$ and $\ell \geq o(G[W])$, respectively. Then, compute $\lambda[G, W]$ following the rules described below:

$$\lambda[G,W] = \begin{cases} \left[\begin{array}{l} \textbf{case 1.}\ |V(G)| = 1: \\ \;0 \quad \text{if } V(G) \cap W = \emptyset, \\ \;1 \quad \text{otherwise;} \end{array}\right. \\[2ex] \left[\begin{array}{l} \textbf{case 2.}\ G = G_1 \cup \cdots \cup G_k,\ \text{ for some } k \geq 2: \\ \;\sum_{i \in \{1,\ldots,k\}} \lambda[G_i, V(G_i) \cap W]; \end{array}\right. \\[2ex] \left[\begin{array}{l} \textbf{case 3.}\ G = G_1 \wedge \cdots \wedge G_k,\ \text{ for some } k \geq 2: \\ \;0 \quad \text{if } \forall\, i \in \{1,\ldots,k\},\ V(G_i) \cap W = \emptyset, \\ \;1 \quad \text{if } \exists\, i, j \in \{1,\ldots,k\},\ i \neq j,\ V(G_i) \cap W \neq \emptyset \text{ and } V(G_j) \cap W \neq \emptyset, \\ \;\max\{1, \lambda[G_i, W] - n + n_i\} \quad \text{if } \exists!\, i \in \{1,\ldots,k\},\ V(G_i) \cap W \neq \emptyset, \\ \text{where } n = |V(G)| \text{ and } n_i = |V(G_i)|. \end{array}\right. \end{cases}$$

The correctness of the rules follows from Lemmas 11 and 12. Since G admits a routerless connection tree if and only if $\lambda[G, W] = 1$, we have that I is a yes-instance of TCP if and only if $\lambda[G, W] = 1$.

Now, we analyse the time complexity of this algorithm. First, we note that $\lambda[G, W]$ can be computed in a bottom-up manner, according to the post-order traversal of the cotree $\mathcal{T}_G$ associated with G, using a dynamic programming matrix indexed by the nodes of $\mathcal{T}_G$. Moreover, we recall that $\mathcal{T}_G$ can be obtained in time $\mathcal{O}(n + m)$ *cf.* [6], and that, by definition, the number of nodes of $\mathcal{T}_G$ is $\mathcal{O}(n)$. Additionally, we note that, before computing $\lambda[G, W]$, $\mathcal{T}_G$ can be preprocessed in time $\mathcal{O}(n)$ so that each node u of $\mathcal{T}_G$ is associated with a flag which informs whether or not $V(G_u) \cap W \neq \emptyset$, where G_u denotes the subgraph of G corresponding to the subtree $\mathcal{T}_G^u$ of $\mathcal{T}_G$ rooted at u, i.e. G_u is the subgraph of G induced by the leaves of $\mathcal{T}_G^u$. Thus, one can verify that, for each node u of $\mathcal{T}_G$, the cell related to u of our dynamic programming matrix, which corresponds to $\lambda[G_u, V(G_u) \cap W]$, can be computed in time $\mathcal{O}(d_{\mathcal{T}_G}(u))$. Since $\mathcal{T}_G$ is a tree on $\mathcal{O}(n)$ nodes, we have that $\sum_{u \in V(\mathcal{T}_G)} d_{\mathcal{T}_G}(u) = \mathcal{O}(n)$. Therefore, $\lambda[G, W]$ can be computed in linear time. □

4. Graphs of bounded maximum degree

In this section, we analyse the complexity of TCP when restricted to graphs of bounded maximum degree. More specifically, we prove that TCP remains NP-complete on graphs of maximum degree 3 even if either the parameter $\ell \geq 0$ or the parameter $r \geq 0$ is fixed. In particular, for fixed $r \geq 0$, we show that TCP is NP-complete on graphs of maximum degree 3 that are *planar*.

It is worth mentioning that, if the input graph G is connected and has maximum degree at most 2, then G is either a path or a cycle, and consequently TCP can be trivially solved in polynomial-time, regardless of ℓ or r. Thus, we obtain that our results establish an NP-*complete versus polynomial-time solvable dichotomy* for TCP with respect to the maximum degree of the input graph.

Another interesting fact about our results is that they separate the complexity of TCP from the complexity of its strict variant, S-TCP. Indeed, while we prove that, for each fixed $\ell \geq 0$, TCP is NP-complete on graphs of maximum degree 3, S-TCP was proved to be polynomial-time solvable on graphs of maximum degree 3 even if $\ell \geq 0$ is fixed [26].

4.1. Fixed number of linkers

First, we consider the case in which the parameter $\ell \geq 0$ is fixed:

Theorem 6. *For each $\ell \geq 0$, TCP remains* NP*-complete when restricted to graphs of maximum degree* 3*.*

To prove Theorem 6, we present a polynomial-time reduction from an NP-complete variant of 3-SAT called 3-SAT(3) *cf.* [28]. The 3-SAT(3) problem has as input a set X of boolean variables and a set $\mathcal{C}$ of clauses over X that satisfy the following conditions:

- Each clause in $\mathcal{C}$ has two or three distinct literals;
- Each variable in X appears exactly twice positive and once negative in the clauses belonging to $\mathcal{C}$.

The problem then asks whether there exists a truth assignment $\alpha \colon X \to \{\mathit{false}, \mathit{true}\}$ such that every clause in $\mathcal{C}$ has at least one true literal under α.

Construction 4 (Reduction from 3-SAT(3) to TCP on Graphs of Maximum Degree 3)**.** Let $I = (X, \mathcal{C})$ be an instance of 3-SAT(3), with variable set $X = \{x_1, x_2, \ldots, x_p\}$ and clause set $\mathcal{C} = \{C_1, C_2, \ldots, C_q\}$, and let ℓ be a non-negative integer. We let G be the graph obtained from I and ℓ as follows (see Figure 5a):

- Create the vertices $u_1, u_2, \ldots, u_\ell$ and, for each $i \in \{1, 2, \ldots, \ell - 1\}$, add the edges $u_i u_{i+1}$; moreover, create the vertices w_I and v_I and add the edges $w_I u_1$ and $u_\ell v_I$, originating the path $P_I = (w_I, u_1, \ldots, u_\ell, v_I)$;
- For each variable $x_i \in X$, create the gadget G_i such that

$$V(G_i) = \{w_i^1, w_i^2, t_i^1, t_i^2, f_i\} \text{ and } E(G_i) = \{w_i^1 t_i^1, t_i^1 t_i^2, t_i^2 w_i^2, w_i^2 f_i, f_i w_i^1\};$$

- Create a complete binary tree T_I, rooted at v_I, whose leaves are the vertices $w_1^1, \ldots, w_p^1$;
- For each clause $C_j \in \mathcal{C}$, create the vertices v_j^1, v_j^2 and v_j^3, and add the edges $v_j^1 v_j^2$, $v_j^2 v_j^3$ and $v_j^3 v_j^1$;
- For each clause $C_j \in \mathcal{C}$, add the edge $t_i^a v_j^b$ if the b-th literal belonging to C_j corresponds to the a-th occurrence in I of the positive literal x_i, for $x_i \in X$, $a \in \{1, 2\}$ and $b \in \{1, \ldots, |C_i|\}$; on the other hand, add the edge $f_i v_j^b$ if the b-th literal belonging to C_j corresponds to the (single) occurrence in I of the negative literal $\overline{x}_i$, for $x_i \in X$ and $b \in \{1, \ldots, |C_j|\}$.

Clearly, G is a graph of maximum degree 3. Then, we let $g(I, \ell) = (G, W, \ell, r)$ be the instance of TCP such that $W = \{w_I\} \cup V(T_I) \cup \{w_i^1, w_i^2 \mid x_i \in X\} \cup \{v_j^1, v_j^2, v_j^3 \mid C_j \in \mathcal{C}\}$ and $r = 2p$.

Lemma 13. *Let $I = (X, \mathcal{C})$ be an instance of 3-SAT(3). For each $\ell \geq 0$, I is a* yes*-instance of 3-SAT(3) if and only if the instance $g(I, \ell)$ described in Construction 4 is a* yes *instance of TCP.*

Proof. Assume that $X = \{x_1, x_2, \ldots, x_p\}$ and $\mathcal{C} = \{C_1, C_2, \ldots, C_q\}$. Additionally, assume that $g(I, \ell) = (G, W, \ell, r)$.

First, suppose that there exists a truth assignment $\alpha\colon X \to \{\mathit{false}, \mathit{true}\}$ such that every clause belonging to $\mathcal{C}$ has at least one true literal under α. Then, let S be the vertex set defined as follows

$$\begin{aligned} S = \{t_i^1, t_i^2 \mid x_i \in X, \alpha(x_i) = \mathit{true}\} &\cup \{f_i \mid x_i \in X, \alpha(x_i) = \mathit{false}\} \\ &\cup \{w_i^1, w_i^2 \mid x_i \in X\} \cup \{v_j^1, v_j^2, v_j^3 \mid C_j \in \mathcal{C}\} \cup V(P_I) \cup V(T_I), \end{aligned}$$

and let $G[S]$ be the subgraph of G induced by S. We note that $G[S]$ is connected but may contain cycles. Thus, let T be a spanning tree subgraph of $G[S]$ that contains all edges of $G[S]$ except for possibly not containing some edges between the vertices v_j^1, v_j^2 and v_j^3, for $C_j \in \mathcal{C}$. In other words, T is a spanning tree subgraph of $G[S]$ such that $E(T) \supseteq E(G[S]) \setminus \{v_j^a v_j^b \mid a, b \in \{1, 2, 3\}, C_j \in \mathcal{C}\}$. It is not hard to check that T is a connection tree of G for W with linker set $\mathsf{L}(T) = \{u_1, \ldots, u_\ell\}$ and router set

$$\mathsf{R}(T) = \{t_i^1, t_i^2 \mid x_i \in X, \alpha(x_i) = \mathit{true}\} \cup \{f_i \mid x_i \in X, \alpha(x_i) = \mathit{false}\}.$$

Therefore, $g(I, \ell)$ is a yes-instance of TCP.

Figure 5 depicts the instance $g(I, \ell) = (G, W, \ell, r)$ of TCP, obtained from an instance $I = (X, \mathcal{C})$ of 3-SAT(3) and a non-negative integer ℓ. It also depicts a connection tree T of G for W, obtained from a truth assignment $\alpha\colon X \to \{\mathit{false}, \mathit{true}\}$.

Conversely, suppose that $g(I, \ell)$ is a yes-instance of TCP, and let T be a connection tree of G for W such $|\mathsf{L}(T)| \leq \ell$ and $|\mathsf{R}(T)| \leq 2p$. We note that the path P_I must be in T, since every path of G between the terminal vertex w_I and any other terminal vertex $w \in W \setminus \{w_I\}$ contains all the vertices of P_I. Consequently, the graph $T' = T - P_I$ cannot contain any linker, and all non-terminal vertices of T' must be routers. This, along with the fact that $\Delta(G) = 3$, implies that $N_T(v) = N_G(v)$ for each $v \in V(T') \setminus W$. Hence, if $t_i^1 \in V(T)$ or $t_i^2 \in V(T)$, then $w_i^1, t_i^2 \in N_T(t_i^1)$ and $w_i^2, t_i^1 \in N_T(t_i^2)$. Analogously, if $f_i \in V(T)$, then $w_i^1, w_i^2 \in N_T(f_i)$. Thus, since T is acyclic, we have that, for each $x_i \in X$, either $t_i^1, t_i^2 \in V(T)$ and $f_i \notin V(T)$, or $t_i^1, t_i^2 \notin V(T)$ and $f_i \in V(T)$. Then, we define a truth assignment $\alpha\colon X \to \{\mathit{false}, \mathit{true}\}$ as follows: for each $x_i \in X$, $\alpha(x_i) = \mathit{false}$ if and only if $f_i \in V(T)$. We note that, for each $C_j \in \mathcal{C}$, every path of G between the terminal vertices v_j^1, v_j^2, v_j^3 and any other terminal vertex $w \in W \setminus \{v_j^1, v_j^2, v_j^3\}$ must contain one of the vertices t_i^1, t_i^2, f_i for some $x_i \in X$. Moreover, by supposition,

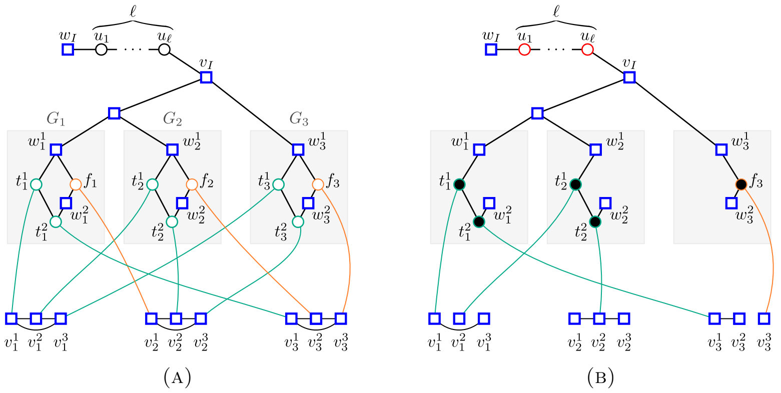


FIGURE 5. (a) Graph G and terminal set W (blue square vertices) of the instance $g(I, \ell)$ of TCP obtained from the instance $I = (X, \mathcal{C})$ of 3-SAT(3), where $X = \{x_1, x_2, x_3\}$ and $\mathcal{C} = \{C_1 = \{x_1, x_2, x_3\}, C_2 = \{\overline{x}_1, x_2, x_3\}, C_3 = \{x_1, \overline{x}_2, \overline{x}_3\}\}$, and from a non-negative integer ℓ. (b) Connection tree T of G for W, obtained from the truth assignment $\alpha\colon X \to \{\mathit{false}, \mathit{true}\}$ such that $\alpha(x_1) = \mathit{true}$, $\alpha(x_2) = \mathit{true}$ and $\alpha(x_3) = \mathit{false}$.

$V(T) \supseteq W \supseteq \{v_j^1, v_j^2, v_j^3 \mid C_j \in \mathcal{C}\}$. Consequently, every clause in $\mathcal{C}$ has at least one true literal under α. Therefore, I is a yes-instance of 3-SAT(3). □

4.2. FIXED NUMBER OF ROUTERS

Now, we consider the case in which the parameter $r \geq 0$ is fixed:

Theorem 7. *For each $r \geq 0$, TCP remains* NP*-complete when restricted to planar graphs of maximum degree* 3.

To prove Theorem 7, we first show through Propositions 1 and 2 that the *st*-HAMILTONIAN PATH problem is NP-complete on planar graphs of maximum degree 3. Then, we present a polynomial-time reduction from this particular case of *st*-HAMILTONIAN PATH to TCP. The *st*-HAMILTONIAN PATH problem is the variant of the HAMILTONIAN PATH problem that asks whether the input graph has a Hamiltonian path between two given vertices s and t.

Next proposition is an intermediate step in order to show the NP-completeness of *st*-HAMILTONIAN PATH on planar graphs of maximum degree 3.

Proposition 1. HAMILTONIAN CYCLE *remains* NP*-complete when restricted to planar graphs of maximum degree* 3 *that have at least two adjacent vertices of degree* 2 *each.*

Proof. Itai *et al.* [19] proved that HAMILTONIAN CYCLE is NP-complete on planar graphs of maximum degree 3. Based on their proof (see Lemma 2.1 [19]), we can suppose without loss of generality that the input graph G has at least one vertex of degree 2. Thus, let $u \in V(G)$ be such a vertex, and let $e = uv$ be an edge that has u and v as endpoints, for some $v \in V(G) \setminus \{u\}$. Then, we define H as the graph obtained from G by *subdividing* e, i.e. by removing e, adding a new vertex u_e and adding the edges uu_e and u_ev. We note that H is a graph of maximum degree 3 that has at least two adjacent vertices of degree 2 each, namely u and u_e. Furthermore, it is immediate that G has a Hamiltonian cycle if and only if H has a Hamiltonian cycle. □

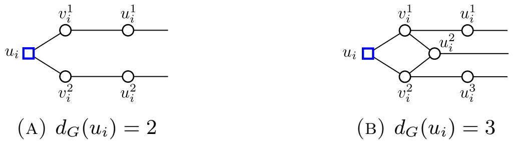
Proposition 2. *st-*HAMILTONIAN PATH *remains* NP*-complete when restricted to planar graphs of maximum degree* 3 *in which s and t have degree* 1 *each.*

Proof. Let G be a planar graph of maximum degree 3. Based on Proposition 1, assume without loss of generality that G contains two vertices $u, v \in V(G)$ such that $uv \in E(G)$ and $d_G(u) = d_G(v) = 2$. Then, let H be the graph obtained from G by adding two new vertices s and t, and by adding the edges su and vt. We note that H is a graph of maximum degree 3 and that s and t have degree 1 in H each. Furthermore, it is straightforward that G has a Hamiltonian cycle if and only if H has a st-Hamiltonian path. □

Below, we finally describe our polynomial-time reduction from st-HAMILTONIAN PATH to TCP. We note that this reduction is slightly similar to the one described in Construction 3 to prove the NP-completeness of TCP on strongly chordal graphs.

Construction 5 (Reduction from st-HAMILTONIAN PATH to TCP on Planar Graphs of Maximum Degree 3)**.** Let G be a planar graph of maximum degree 3 and $s, t \in V(G)$ be distinct vertices of G. Based on Proposition 2, assume without loss of generality that $d_G(s) = d_G(t) = 1$. Moreover, assume that every vertex of G different from s and t has degree at least 2, otherwise G would certainly not admit a *st-Hamiltonian path*, i.e. a Hamiltonian path between s and t. Also, assume that $V(G) = \{u_1, \ldots, u_n\}$, for some positive integer n, where $s = u_1$ and $t = u_n$. Let r be a non-negative integer. For each $u_i \in V(G) \setminus \{s, t\}$, let $\alpha_i \colon N_G(u_i) \to |N_G(u_i)|$ be the bijection such that, for each two distinct vertices $u_{j_1}, u_{j_2} \in N_G(u_i)$, we have that $\alpha_i(u_{j_1}) < \alpha_i(u_{j_2})$ if and only if $j_1 < j_2$. We let G' be the graph obtained from G, s, t and r as follows (see Figure 7):

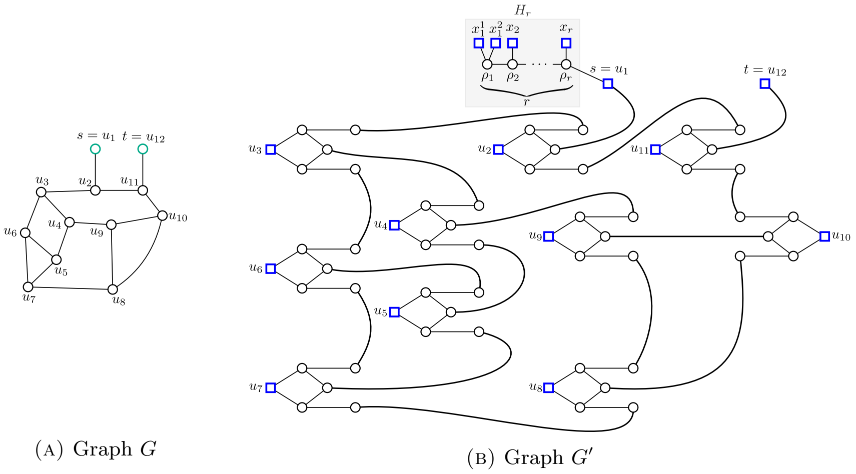
- Add all vertices of G to G';
- For each vertex $u_i \in V(G)$ of degree 2 in G, add new vertices $v_i^1, v_i^2, u_i^1, u_i^2$ and add the edges $u_iv_i^1$, $u_iv_i^2$, $v_i^1u_i^1$ and $v_i^2u_i^2$ (see Figure 6a);
- For each vertex $u_i \in V(G)$ of degree 3 in G, add new vertices $v_i^1, v_i^2, u_i^1, u_i^2, u_i^3$ and add the edges $u_iv_i^1$, $u_iv_i^2$, $v_i^1u_i^2$, $v_i^2u_i^2$, $v_i^1u_i^1$ and $v_i^2u_i^3$; (see Figure 6b)
- For each vertex $u_i \in V(G)$ and each vertex $u_j \in N_G(u_i)$, add the edges $u_i^au_j^b$, where $a = \alpha_i(u_j)$ and $b = \alpha_j(u_i)$;
- If $r \geq 1$, create the gadget H_r and the terminal set W_r described in Construction 2, and add the edge ρ_rs; otherwise, define $W_r = \emptyset$.



(A) $d_G(u_i) = 2$ (B) $d_G(u_i) = 3$

FIGURE 6. (a) Case in which $d_G(u_i) = 2$: vertices $v_i^1, v_i^2, u_i^1, u_i^2$. (b) Case in which $d_G(u_i) = 3$: vertices $v_i^1, v_i^2, u_i^1, u_i^2, u_i^3$.

For each $u_i \in V(G)$, let G_i' be the subgraph of G' illustrated in Figure 6, i.e. the subgraph of G' induced by $\{u_i, v_i^1, v_i^2\} \cup \{u_i^j \mid j \in \{1, \ldots, d_G(u_i)\}\}$. Note that, H_r and, for each $u_i \in V(G)$, G_i' are planar. Additionally, it is not hard to verify that the input graph G is isomorphic to the graph resulting from $G' - H_r$ by identifying every subgraph G_i' into the vertex u_i. Therefore, since G is planar, we have that G' is planar as well. Furthermore, it straightforward that G' is a graph of maximum degree 3. Then, we let $g(G, s, t, r) = (G', W, \ell, r)$ be the instance of TCP such that $W = V(G) \cup W_r$ and $\ell = 4n - 4$.



(A) Graph G (B) Graph G'

FIGURE 7. (a) Graph G of maximum degree 3, with two distinct vertices $s, t \in V(G)$ such that $d_G(s) = d_G(t) = 1$. (b) Graph G' with terminal set W (blue square vertices) of the instance $g(G, s, t, r)$ of TCP described in Construction 5, obtained from G, the vertices s and t, and a non-negative integer r.

Lemma 14. *Let G be a graph of maximum degree 3 and $s, t \in V(G)$ be two distinct vertices of G. Assume that s and t have degree 1 in G each. For each $r \geq 0$, G*

admits a st-Hamiltonian path if and only if the instance $g(G,s,t,r)$ described in Construction 5 is a yes*-instance of TCP.*

Proof. Assume that $V(G) = \{u_1, \ldots, u_n\}$, where $s = u_1$ and $t = u_n$, and that $g(G,s,t,r) = (G', W, \ell, r)$. Additionally, for simplicity, consider $W_r = V(H_r) = E(H_r) = \emptyset$ if $r = 0$.

First, suppose that there exists in G a Hamiltonian path

$$P = (u_{j_1}, u_{j_2}, \ldots, u_{j_{n-1}}, u_{j_n})$$

such that $s = u_{j_1}$ and $t = u_{j_n}$. Then, let S be the vertex set defined as follows:

$$\begin{aligned} S = V(H_r) \cup V(P) \cup \{v_i^1, v_i^2 \mid i \in \{2, \ldots, n-1\}\} \cup \{u_{j_2}^{\alpha_{j_2}(s)}, u_{j_{n-1}}^{\alpha_{j_{n-1}}(t)}\} \\ \cup \{u_{j_i}^a, u_{j_{i+1}}^b \mid a = \alpha_{j_i}(u_{j_{i+1}}), b = \alpha_{j_{i+1}}(u_{j_i}), i \in \{2, \ldots, n-2\}\}, \end{aligned}$$

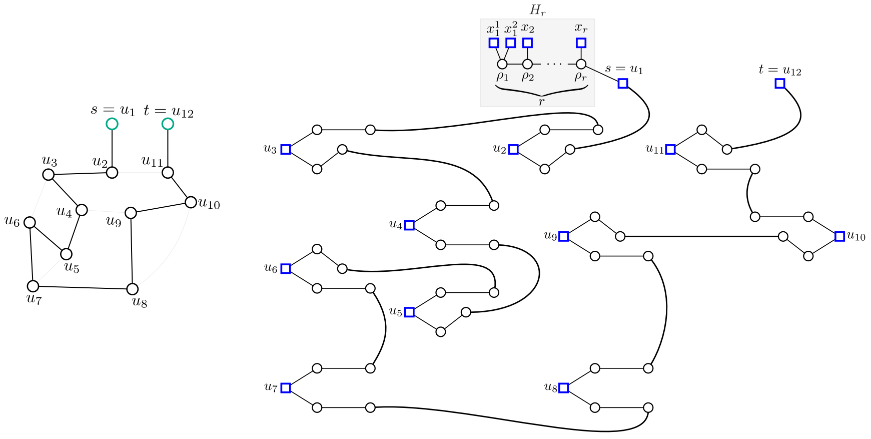
where α_i denotes the bijection from $N_G(u_i)$ to $|N_G(u_i)|$ described in Construction 5. We note that $G'[S]$ is connected but may contain cycles. More precisely, every cycle of $G'[S]$ is of the form $(u_i, v_i^1, u_i^2, v_i^2, u_i)$, and it exists if and only if $d_G(u_i) = 3$ and either $S \supseteq \{u_i^1, u_i^2\}$ or $S \supseteq \{u_i^2, u_i^3\}$, for $u_i \in V(G) \setminus \{s,t\}$. Thus, we let T be the graph obtained from $G'[S]$ by removing, for each vertex $u_i \in V(G) \setminus \{s,t\}$ with $d_G(u_i) = 3$, the edge $v_i^1 u_i^2$ if $S \supseteq \{u_i^1, u_i^2\}$, or the edge $v_i^2 u_i^2$ if $S \supseteq \{u_i^2, u_i^3\}$. One can verify that T is a connection tree of G' for W such that $\mathsf{L}(T) = S \setminus (V(H_r) \cup V(G))$ and $\mathsf{R}(T) = \{\rho_1, \ldots, \rho_r\}$. Therefore, $g(G,s,t,r)$ is a yes-instance of TCP.

Consider the graph G depicted in Figure 7a and the graph G' and the terminal set W depicted in Figure 7b, obtained from G and a non-negative integer r. Figure 8 illustrates a connection tree T of G' for W obtained from the st-Hamiltonian path of G depicted in Figure 8a.

Conversely, suppose that $g(G,s,t,r)$ is a yes-instance of TCP, and let T be a connection tree of G' for W such that $|\mathsf{L}(T)| \leq 4n - 4$ and $|\mathsf{R}(T)| \leq r$. We note that $\mathsf{R}(T) = \{\rho_1, \ldots, \rho_r\}$. Consequently, $T' = T - H_r$ cannot contain any router, and all non-terminal vertices of T' must be linkers. Moreover, by construction, s and t have degree 1 in T' each. This implies that the vertices $u_2, \ldots, u_{n-1}$ have degree exactly 2 in T' each, otherwise T would not be connected or $W \not\subseteq V(T)$. Hence, T' consists in a path P' between s and t of the form

$$P' = (s, u_{j_2}^{a_2}, v_{j_2}^{c_2}, u_{j_2}, v_{j_2}^{c'_2}, u_{j_2}^{b_2}, \ldots, u_{j_{n-1}}^{a_{n-1}}, v_{j_{n-1}}^{c_{n-1}}, u_{j_{n-1}}, v_{j_{n-1}}^{c'_{n-1}}, u_{j_{n-1}}^{b_{n-1}}, t),$$

where, for each $i \in \{2, \ldots, n-2\}$, $a_i = \alpha_{j_i}(u_{j_{i-1}})$, $b_i = \alpha_{j_i}(u_{j_{i+1}})$, and $c_i, c'_i \in \{1,2\}$ with $c_i \neq c'_i$. Therefore, one can verify that $(s, u_{j_2}, \ldots, u_{j_{n-1}}, t)$ is a st-Hamiltonian path of G. □



(A) st-Hamiltonian path P

(B) Connection tree T

Figure 8. (a) st-Hamiltonian path P of the graph G depicted in Figure 7a. (b) Connection tree T of the graph G' for the terminal set W depicted in Figure 7b, obtained from P.

5. Concluding remarks

We conclude this work by posing some open questions. First, we ask about the existence of a graph class $\mathcal{G}$ on which Steiner tree is polynomial-time solvable while TCP remains NP-complete for fixed ℓ. We have shown that, on strongly chordal graphs, TCP is NP-complete for each $r \geq 0$, whereas Steiner tree is known to be polynomial-time solvable. However, the complexity of TCP on strongly chordal graphs for fixed ℓ has not been settled yet. Analogously, we ask whether there exists a graph class $\mathcal{G}$ on which TCP is polynomial-time solvable for fixed ℓ while Steiner tree remains NP-complete. We have shown that, on split graphs, TCP is polynomial-time solvable for fixed $r \geq 1$, whereas Steiner tree is known to be NP-complete. However, up to our knowledge, for fixed ℓ, there is no known example of such a separating class.

In addition, it is worth mentioning that, in our tractability proof of TCP on split graphs, only the cases in which $r \geq 1$ or $W \cap K \neq \emptyset$ are considered. Such hypotheses are imperative in our argumentation so as to ensure the connectivity of the sought connection tree. Thus, we leave as an open question whether TCP can be solved in polynomial-time on split graphs when $r = 0$ and $W \cap K = \emptyset$.

We also leave as an open question whether TCP parameterized by clique-width is in XP. Through a parameterized-reduction from Hamiltonian path, we have shown that TCP parameterized by clique-width is W[1]-hard. Nevertheless, the question whether TCP is in XP remains unsettled.

Finally, we ask about the complexity of TCP parameterized by the number of terminal vertices. Even though it is well-known that Steiner tree parameterized by the number of terminal vertices is in FPT [12], the complexity of the corresponding parameterization of TCP is widely open.

References

[1] Benjamin Bergougnoux and Mamadou Kanté. Fast exact algorithms for some connectivity problems parameterized by clique-width. *Theoretical Computer Science*, 782:30 – 53, 2019.

[2] Andreas Björklund, Thore Husfeldt, Petteri Kaski, and Mikko Koivisto. Fourier meets möbius: Fast subset convolution. In *Proceedings of the Thirty-Ninth Annual ACM Symposium on Theory of Computing*, STOC '07, page 67–74, New York, NY, USA, 2007. Association for Computing Machinery.

[3] A. Bondy and U.S.R. Murty. *Graph Theory*. Graduate Texts in Mathematics. Springer London, 2008.

[4] Charles J. Colbourn and Lorna K. Stewart. Permutation graphs: connected domination and Steiner trees. *Discrete Mathematics*, 86(1-3):179–189, 1990.

[5] Derek G. Corneil, Helmut Lerchs, and Stewart L. Burlingham. Complement reducible graphs. *Discrete Applied Mathematics*, 3(3):163–174, 1981.

[6] Derek G. Corneil, Yehoshua Perl, and Lorna K. Stewart. A linear recognition algorithm for cographs. *SIAM Journal on Computing*, 14(4):926–934, 1985.

[7] Bruno Courcelle, Joost Engelfriet, and Grzegorz Rozenberg. Handle-rewriting hypergraph grammars. *Journal of Computer and System Sciences*, 46(2):218–270, 1993.

[8] Marek Cygan, Marcin Pilipczuk, Michał Pilipczuk, and Jakub Onufry Wojtaszczyk. Kernelization hardness of connectivity problems in d-degenerate graphs. *Discrete Applied Mathematics*, 160(15):2131 – 2141, 2012.

[9] Alessandro D'Atri and Marina Moscarini. Distance-hereditary graphs, Steiner trees, and connected domination. *SIAM Journal on Computing*, 17(3):521–538, 1988.

[10] M. C. Dourado, R. A. Oliveira, F. Protti, and U. S. Souza. Design of connection networks with bounded number of non-terminal vertices. In *Proceedings of V Latin-American Workshop on Cliques in Graphs*, volume 42 of *Matemática Contemporânea*, pages 39–47, Buenos Aires, 2014. SBM.

[11] Mitre C. Dourado, Rodolfo A. Oliveira, Fábio Protti, and Uéverton S. Souza. Conexão de terminais com número restrito de roteadores e elos. In *Proceedings of XLVI Simpósio Brasileiro de Pesquisa Operacional*, pages 2965–2976, 2014.

[12] Stuart E. Dreyfus and Robert A. Wagner. The Steiner problem in graphs. *Networks*, 1(3):195–207, 1971.

[13] Martin Farber. Characterizations of strongly chordal graphs. *Discrete Mathematics*, 43(2):173 – 189, 1983.

[14] Michael R Fellows, Frances A Rosamond, Udi Rotics, and Stefan Szeider. Clique-width is np-complete. *SIAM Journal on Discrete Mathematics*, 23(2):909–939, 2009.

[15] Fedor V Fomin, Petr A Golovach, Daniel Lokshtanov, and Saket Saurabh. Clique-width: on the price of generality. In *Proceedings of the twentieth annual ACM-SIAM symposium on Discrete algorithms*, pages 825–834. SIAM, 2009.

[16] Michael R. Garey and David S. Johnson. The rectilinear Steiner tree problem is NP-complete. *SIAM Journal on Applied Mathematics*, 32(4):826–834, 1977.

[17] Luisa Gargano, Mikael Hammar, Pavol Hell, Ladislav Stacho, and Ugo Vaccaro. Spanning spiders and light-splitting switches. *Discrete Mathematics*, 285(1):83 – 95, 2004.

[18] Frank K. Hwang, Dana S. Richards, and Pawel Winter. *The Steiner tree problem*, volume 53 of *Annals of Discrete Mathematics*. Elsevier, 1992.

[19] Alon Itai, Christos H. Papadimitriou, and Jayme Luiz Szwarcfiter. Hamilton paths in grid graphs. *SIAM Journal on Computing*, 11(4):676–686, nov 1982.

[20] Richard M. Karp. *Reducibility among Combinatorial Problems*, pages 85–103. Springer US, Boston, MA, 1972.

[21] Guohui Lin and Guoliang Xue. On the terminal Steiner tree problem. *Information Processing Letters*, 84(2):103–107, 2002.

[22] Giordano Da Lozzo and Ignaz Rutter. Strengthening hardness results to 3-connected planar graphs. *arXiv preprint arXiv:1607.02346*, 2016.

[23] Chin Lung Lu, Chuan Yi Tang, and Richard Chia-Tung Lee. The full Steiner tree problem. *Theoretical Computer Science*, 306(1-3):55–67, 2003.

[24] Alexsander A. Melo, Celina M. H. Figueiredo, and Uéverton S. Souza. Connecting terminals using at most one router. In *Proceedings of VII Latin-American Workshop on Cliques in Graphs*, volume 45 of *Matemática Contemporânea*, pages 49–57. SBM, 2017.

[25] Alexsander A. Melo, Celina M. H. Figueiredo, and Uéverton S. Souza. On undirected two-commodity integral flow, disjoint paths and strict terminal connection problems. *Networks*, 77(4):559–571, 2021.

[26] Alexsander A. Melo, Celina M. H. Figueiredo, and Uéverton S. Souza. A multivariate analysis of the strict terminal connection problem. *Journal of Computer and System Sciences*, 111:22–41, 2020.

[27] Alexsander A. Melo, Celina M. H. Figueiredo, and Uéverton S. Souza. On the terminal connection problem. In *Proceedings of 47th International Conference on Current Trends in Theory and Practice of Computer Science*, volume 12607 of *Lecture Notes in Computer Science*, pages 278–292. Springer-Verlag New York, Inc., 2021.

[28] Haiko Müller. Hamiltonian circuits in chordal bipartite graphs. *Discrete Mathematics*, 156(1-3):291–298, 1996.

[29] Haiko Müller and Andreas Brandstädt. The NP-completeness of Steiner tree and dominating set for chordal bipartite graphs. *Theoretical Computer Science*, 53(2-3):257–265, 1987.

[30] Jesper Nederlof. Fast polynomial-space algorithms using inclusion-exclusion. *Algorithmica*, 65(4):868–884, 2013.

[31] Dimitri Watel, Marc-Antoine Weisser, Cédric Bentz, and Dominique Barth. Steiner problems with limited number of branching nodes. In *Proceedings of 20th International Colloquium on Structural Information and Communication Complexity*, volume 8179 of *Lecture Notes in Computer Science*, pages 310–321. Springer-Verlag New York, Inc., 2013.

[32] Dimitri Watel, Marc-Antoine Weisser, Cédric Bentz, and Dominique Barth. Directed Steiner trees with diffusion costs. *Journal of Combinatorial Optimization*, 32(4):1089–1106, 2016.

[33] Kevin White, Martin Farber, and William Pulleyblank. Steiner trees, connected domination and strongly chordal graphs. *Networks*, 15(1):109–124, 1985.

Communicated by (The editor will be set by the publisher).
...

Appendix E

Manuscript: Maximum Cut on Interval Graphs of Interval Count Four is NP-complete

This appendix contains the manuscript:
Celina M. H. de Figueiredo, Alexsander A. de Melo, Fabiano de Oliveira, Ana Silva. Maximum Cut on Interval Graphs of Interval Count Four is NP-complete. Presented in the *46th International Symposium on Mathematical Foundations of Computer Science (MFCS 2021)* [35], and submitted in December 2021 to *Discrete & Computational Geometry.*

Maximum cut on interval graphs of interval count four is NP-complete

Celina M.H. de Figueiredo[1], Alexsander A. de Melo[1], Fabiano S. Oliveira[2] and Ana Silva[3*†]

[1]Department, Universidade Federal do Rio de Janeiro, Street, Rio de Janeiro, 100190, Rio de Janeiro, Brazil.
[2]Department, Universidade do Estado do Rio de Janeiro, Street, City, 10587, Rio de Janeiro, Brazil.
[3*]Departamento de Matemática - Centro de Ciências, Universidade Federal do Ceará, Av. Mister Hull s/n - Bloco 914, Fortaleza, 60455-760, Ceará, Brazil.

*Corresponding author(s). E-mail(s): anasilva@mat.ufc.br;
Contributing authors: celina@cos.ufrj.br; aamelo@cos.ufrj.br; fabiano.oliveira@ime.uerj.br;
†These authors contributed equally to this work.

Abstract

The computational complexity of the MaxCut problem restricted to interval graphs has been open since the 80's, being one of the problems proposed by Johnson on his *Ongoing Guide to NP-completeness*, and has been settled as NP-complete only recently by Adhikary, Bose, Mukherjee and Roy. On the other hand, many flawed proofs of polynomiality for MaxCut on the more restrictive class of unit/proper interval graphs (or graphs with interval count 1) have been presented along the years, and the classification of the problem is still unknown. In this paper, we present the first NP-completeness proof for MaxCut when restricted to interval graphs with bounded interval count, namely graphs with interval count 4.

Keywords: maximum cut, interval graphs, interval lengths, interval count, NP-complete.

MSC Classification: 68Q17 , 68Q25 , 68R10 , 05C62

1 Introduction

A *cut* is a partition of the vertex set of a graph into two disjoint parts and the *maximum cut problem* (denoted MaxCut for short) aims to determine a cut with the maximum number of edges for which each endpoint is in a distinct part. The decision problem MaxCut is known to be NP-complete since the seventies [1], and only recently its restriction to interval graphs has been announced to be hard [2], settling a long-standing open problem that appeared in the 1985 column of the *Ongoing Guide to NP-completeness* by David S. Johnson [3]. We refer the reader to a revised version of the table in [4], where one can also find a parameterized complexity version of said table.

An *interval model* is a family of closed intervals of the real line. A graph is an *interval graph* if there exists an interval model, for which each interval corresponds to a vertex of the graph, such that distinct vertices are adjacent in the graph if and only if the corresponding intervals intersect. Ronald L. Graham proposed in the 80's the study of the *interval count* of an interval graph as the smallest number of interval lengths used by an interval model of the graph. Interval graphs having interval count 1 are called *unit intervals* (these are also called proper interval, or indifference). Understanding the interval count, besides being an interesting and challenging problem by itself, can be also of value for the investigation of problems that are hard for general interval graphs, and easy for unit interval graphs (e.g. geodetic number [5, 6], optimal linear arrangement [7, 8], sum coloring [9, 10]). The positive results for unit interval graphs usually take advantage of the fact that a representation for these graphs can be found in linear time [11, 12]. Surprisingly, the recognition of interval graphs with interval count k is open, even for $k = 2$ [13]. Nevertheless, another generalization of unit interval graphs has been recently introduced which might be more promising in this aspect. These graphs are called *k-nested interval graphs*, for which an efficient recognition algorithm has firstly appeared in [14]. Recently, a linear time algorithm has been devised in [15].

In the same way that MaxCut on interval graphs has evaded being solved for so long, the community has been puzzled by the restriction to unit interval graphs. Indeed, two attempts at solving it in polynomial time were proposed in [16, 17] just to be disproved closely after [18, 19]. In this paper, we give the first classification that bounds the interval count, namely, we prove that MaxCut is NP-complete when restricted to interval graphs of interval count 4. This also implies NP-completeness for the newly generalized class of 4-nested graphs, and opens the search for a full polynomial/NP-complete dichotomy classification in terms of the interval count. It can still happen that the problem is hard even on graphs of interval count 1. We contribute towards filling the complexity gap between interval and unit interval graphs. We have communicated the result at the MFCS 2021 conference [20], and previous versions of the full proof appeared in the ArXiv [21]. The present paper contains the improved and much shorter full proof.

Next, we establish basic definitions and notation. Section 2 describes our reduction and Section 3 discusses the interval count of the interval graph constructed in [2].

1.1 Preliminaries

In this work, all graphs considered are simple. For missing definitions and notation of graph theory, we refer to [22]. For a comprehensive study of interval graphs, we refer to [23].

Let G be a graph. Let X and Y be two disjoint subsets of $V(G)$. We let $E_G(X,Y)$ be the set of edges of G with an endpoint in X and the other endpoint in Y. A *cut* of G is a partition of $V(G)$ into two parts $A, B \subseteq V(G)$, denoted by $[A,B]$; the edge set $E_G(A,B)$ is called the *cut-set* of G associated with $[A,B]$. For each two vertices $u, v \in V(G)$, we say that u and v *are in a same part of* $[A,B]$ if either $\{u,v\} \subseteq A$ or $\{u,v\} \subseteq B$; otherwise, we say that u and v *are in opposite parts of* $[A,B]$. Denote by $\mathsf{mc}(G)$ the maximum size of a cut-set of G. The MaxCut problem has as input a graph G and a positive integer k, and it asks whether $\mathsf{mc}(G) \geq k$.

Let $I \subseteq \mathbb{R}$ be a closed interval of the real line. We let $\ell(I)$ and $r(I)$ denote respectively the minimum and maximum points of I, which we call the *left* and the *right endpoints* of I, respectively. We denote a closed interval I by $[\ell(I), r(I)]$. We say that an interval I *precedes* an interval I' if $r(I) < \ell(I')$. The *length* of an interval I is defined as $|I| = r(I) - \ell(I)$. An *interval model* is a finite multiset $\mathcal{M}$ of intervals. The *interval count* of an interval model $\mathcal{M}$, denoted by $\mathsf{ic}(\mathcal{M})$, is defined as the number of distinct lengths of the intervals in $\mathcal{M}$. Let G be a graph and $\mathcal{M}$ be an interval model. An $\mathcal{M}$*-representation* of G is a bijection $\phi\colon V(G) \to \mathcal{M}$ such that, for every two distinct vertices $u, v \in V(G)$, we have that $uv \in E(G)$ if and only if $\phi(u) \cap \phi(v) \neq \emptyset$. If such an $\mathcal{M}$-representation exists, we say that $\mathcal{M}$ is an *interval model of* G. We note that a graph may have either no interval model or arbitrarily many distinct interval models. A graph is called an *interval graph* if it has an interval model. The *interval count of an interval graph* G, denoted by $\mathsf{ic}(G)$, is defined as $\mathsf{ic}(G) = \min\{\mathsf{ic}(\mathcal{M})\colon \mathcal{M} \text{ is an interval model of } G\}$. An interval graph is called a *unit interval graph* if its interval count is equal to 1.

Note that, for every interval model $\mathcal{M}$, there exists a unique (up to isomorphism) graph that admits an $\mathcal{M}$-representation. Thus, for every interval model $\mathcal{M} = \{I_1, \ldots, I_n\}$, we let $\mathbb{G}_{\mathcal{M}}$ be the graph with vertex set $V(\mathbb{G}_{\mathcal{M}}) = \{1, \ldots, n\}$ and edge set $E(\mathbb{G}_{\mathcal{M}}) = \{ij\colon I_i, I_j \in \mathcal{M},\ I_i \cap I_j \neq \emptyset,\ i \neq j\}$. Since $\mathbb{G}_{\mathcal{M}}$ is uniquely determined (up to isomorphism) from $\mathcal{M}$, in what follows we may make an abuse of language and use graph terminologies to describe properties related to the intervals in $\mathcal{M}$. Two intervals $I_i, I_j \in \mathcal{M}$ are said to be *true twins* in $\mathbb{G}_{\mathcal{M}}$ if they have the same closed neighborhood in $\mathbb{G}_{\mathcal{M}}$.

2 Our reduction

The following theorem is the main contribution of this work:

Theorem 1 MaxCut *is NP-complete on interval graphs of interval count* 4*.*

This result is a stronger version of that of Adhikary et al. [2]. To prove Theorem 1, we present a polynomial-time reduction from MaxCut on cubic graphs, which is known to be NP-complete [24]. Since our proof is based on that of Adhikary et al., we start by presenting some important properties of their key gadget.

2.1 Grained gadget

The interval graph constructed in the reduction of [2] is strongly based on two types of gadgets, called *V-gadgets* and *E-gadgets*. In fact, these gadgets have the same structure except for the number of intervals of certain kinds contained in each of them. In this subsection, we present a generalization of such gadgets, rewriting their key properties to suit our purposes. In order to discuss the interval count of the reduction of [2], we describe it in details in Section 3.

Let x and y be two positive integers. An (x, y)*-grained gadget* is an interval model $\mathcal{H}$ formed by y long intervals (called *left long*) intersecting in their right endpoint with other y long intervals (called *right long)*, together with $2x$ short intervals, x of which (called *left short*) intersect exactly the y left long ones, and x of which (called *right short*) intersect exactly the y right long ones; see Figure 1. We write $\mathcal{LS}(\mathcal{H})$, $\mathcal{LL}(\mathcal{H})$, $\mathcal{RS}(\mathcal{H})$ and $\mathcal{RL}(\mathcal{H})$ to denote the left short, left long, right short and right long intervals of $\mathcal{H}$, respectively. And we omit $\mathcal{H}$ when it is clear from the context.

Note that, if $\mathcal{H}$ is an (x, y)-grained gadget, then $\mathbb{G}_{\mathcal{H}}$ is a split graph such that $\mathcal{LS} \cup \mathcal{RS}$ is an independent set of size $2x$, $\mathcal{LL} \cup \mathcal{RL}$ is a clique of size $2y$, $N_{\mathbb{G}_{\mathcal{H}}}(\mathcal{LS}) = \mathcal{LL}$ and $N_{\mathbb{G}_{\mathcal{H}}}(\mathcal{RS}) = \mathcal{RL}$. Moreover, the intervals in $\mathcal{LL}$ are true twins in $\mathbb{G}_{\mathcal{H}}$; similarly, the intervals in $\mathcal{RL}$ are true twins in $\mathbb{G}_{\mathcal{H}}$.

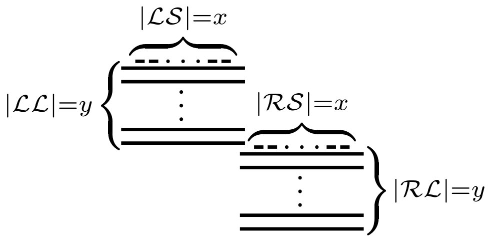
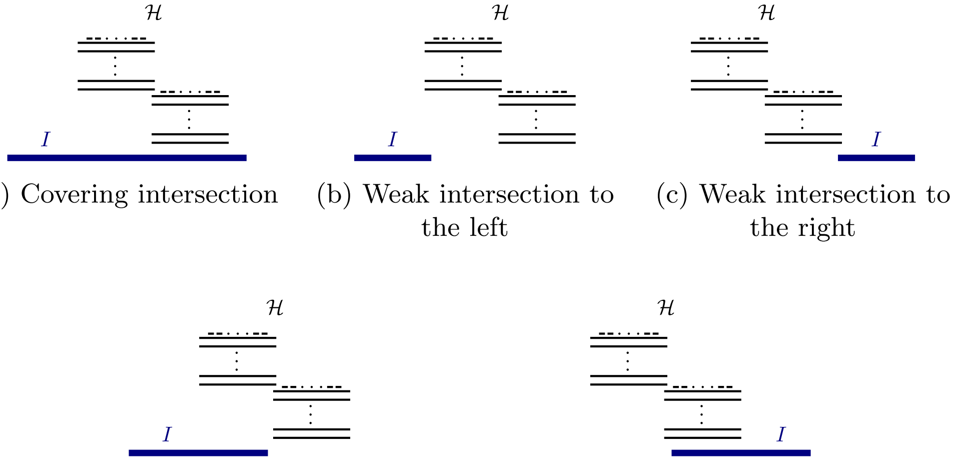


Fig. 1: General structure of an (x, y)-grained gadget.

Let $\mathcal{M}$ be an interval model containing an (x, y)-grained gadget $\mathcal{H}$. The possible types of intersections between an interval $I \in \mathcal{M} \setminus \mathcal{H}$ and $\mathcal{H}$ in our construction are depicted in Figure 2, with the used nomenclature. More specifically, the *cover intersection* intersects all the intervals, the *weak intersection to the left* (*right*) intersects exactly the left (right) long intervals, while the *strong intersection to the left* (*right*) intersects exactly the left (right) long and short intervals. We say that $\mathcal{M}$ *respects the structure* of $\mathcal{H}$ if, for every interval $I \in \mathcal{M} \setminus \mathcal{H}$, we have that I either does not intersect $\mathcal{H}$, or intersects $\mathcal{H}$ as one of the described possible types in Figure 2.



(a) Covering intersection (b) Weak intersection to the left (c) Weak intersection to the right

(d) Strong intersection to the left (e) Strong intersection to the right

Fig. 2: Interval $I \in \mathcal{M} \setminus \mathcal{H}$ (a) covering $\mathcal{H}$, (b-c) weakly intersecting $\mathcal{H}$ to the left and to the right, and (d-e) strongly intersecting $\mathcal{H}$ to the left and to the right.

The advantage of this gadget is that, by manipulating the values of x and y, we can ensure that, in a maximum cut, the left long and right short intervals are placed in the same part, opposite to the part containing the left short and right long intervals, as proved in the following lemma.

Lemma 1 *Let x and y be positive integers, $\mathcal{H}$ be an (x,y)-grained gadget and $\mathcal{M}$ be an interval model that respects the structure of $\mathcal{H}$. Also, let $[A,B]$ be a maximum cut of $\mathbb{G}_{\mathcal{M}}$, t be the number of intervals in $\mathcal{M} \setminus \mathcal{H}$ intersecting $\mathcal{H}$, ℓ be the number of intervals in $\mathcal{M}$ intersecting the left short intervals of $\mathcal{H}$, and r be the number of intervals in $\mathcal{M}$ intersecting the right short intervals of $\mathcal{H}$. If ℓ and r are odd, $y > 2t$ and $x > t + 2y$, then the following hold:*

1. *$\mathcal{LS}(\mathcal{H}) \subseteq A$ and $\mathcal{LL}(\mathcal{H}) \subseteq B$, or vice versa;*
2. *$\mathcal{RS}(\mathcal{H}) \subseteq A$ and $\mathcal{RL}(\mathcal{H}) \subseteq B$, or vice versa; and*
3. *$\mathcal{LL}(\mathcal{H}) \subseteq A$ and $\mathcal{RL}(\mathcal{H}) \subseteq B$, or vice versa.*

Proof First, we prove that all the left short intervals are in the same part of $[A,B]$. Denote by $\mathcal{N}$ the set of intervals in $\mathcal{M}$ that intersect the left short intervals. Suppose, without loss of generality, that B contains more than half of the intervals in $\mathcal{N}$ (it must occur for either A or B since ℓ is odd). Since $\mathcal{LS}$ is an independent set, this implies that it is more advantageous to put every left short interval in A. Because r is also odd, a similar argument holds for the right short intervals.

Now consider the left long intervals and suppose, without loss of generality, that all the left short intervals are contained in A. Observe that the number of intervals in $\mathcal{M} \setminus \mathcal{LS}$ intersecting a left long interval is less than $t + 2y < x$. Therefore, it is more advantageous to put all the left long intervals in the same part of $[A, B]$ opposite to the left short intervals. This also holds for the right long intervals, analogously.

Finally, let $\mathcal{L}$ denote the set of long intervals of $\mathcal{H}$ and suppose that $\mathcal{L} \subseteq A$. Since the short intervals are always opposite to the corresponding long intervals, the number of edges in the cut incident to $\mathcal{L}$ is at most the number of edges between $\mathcal{L}$ and $\mathcal{M} \setminus \mathcal{H}$, plus the number of edges between $\mathcal{LL}$ and $\mathcal{LS}$, plus the number of edges between $\mathcal{RL}$ and $\mathcal{RS}$. This amounts to at most $2yt + 2xy$ edges. Now, if instead we have $\mathcal{LL} \subseteq A$ and $\mathcal{RL} \subseteq B$, then the number of edges in the cut incident to $\mathcal{L}$ is at least the number of edges between $\mathcal{LL}$ and $\mathcal{LS}$, plus the number of edges between $\mathcal{RL}$ and $\mathcal{RS}$, plus the number of edges between $\mathcal{LL}$ and $\mathcal{RL}$, which amounts to at least $y^2 + 2xy$. Again the latter is more advantageous since $y > 2t$. □

We say that $(\mathcal{H}, \mathcal{M})$ is *well-valued* if the conditions of Lemma 1 are satisfied. Moreover, we say that the constructed model $\mathcal{M}$ is *well-valued* if all its grained gadgets $\mathcal{H}$ are well-valued with respect to the model $\mathcal{M}$. Finally, we say that $\mathcal{H}$ is *A-partitioned* by $[A, B]$ if $\mathcal{LS}(\mathcal{H}) \cup \mathcal{RL}(\mathcal{H}) \subseteq A$ and $\mathcal{RS}(\mathcal{H}) \cup \mathcal{LL}(\mathcal{H}) \subseteq B$. Define *B-partitioned* analogously.

2.2 Reduction graph

In this subsection, we formally present our construction. We will make a reduction from MaxCut on cubic graphs. So, let G be a cubic graph on n vertices and m edges. Intuitively, we consider an ordering of the edges of G, and we divide the real line into m regions, with the j-th region holding the information about whether the j-th edge is in the cut-set. For this, each vertex u will be related to a subset of intervals traversing all the m regions, bringing the information about which part of the cut contains u. Let $\pi_V = (v_1, \ldots, v_n)$ be an ordering of $V(G)$, $\pi_E = (e_1, \ldots, e_m)$ be an ordering of $E(G)$, and $\mathfrak{G} = (G, \pi_V, \pi_E)$.

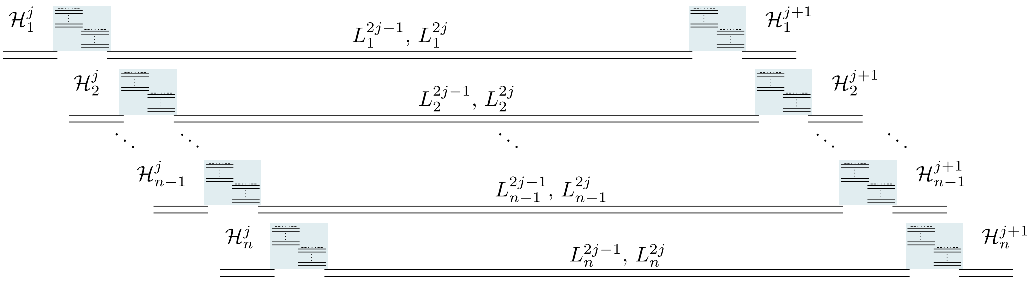


Fig. 3: General structure of a region of an (n, m)-escalator. The shaded rectangles represent the (p, q)-grained gadgets $\mathcal{H}_i^j$.

We first describe the gadgets related to the vertices. Please refer to Figure 3 to follow the construction. The values of p, q used next will be defined later. An (n, m)-*escalator* is an interval model $\mathcal{D}$ formed by $m+1$ (p, q)-grained gadgets for each v_i, denoted by $\mathcal{H}_i^1, \ldots, \mathcal{H}_i^{m+1}$, together with $2m$ *link intervals*, denoted by $L_i^1, \ldots, L_i^{2m}$, such that L_i^{2j-1} and L_i^{2j} weakly intersect $\mathcal{H}_i^j$ to the right and weakly intersect $\mathcal{H}_i^{j+1}$ to the left. Additionally, all the grained gadgets are mutually disjoint, given $j \in \{1, \ldots, m+1\}$ and $i, i' \in \{1, \ldots, n\}$ with $i < i'$, the grained gadget $\mathcal{H}_i^j$ occurs to the left of $\mathcal{H}_{i'}^j$, and H_n^j occurs to the left of H_1^{j+1} for $j \in \{1, \ldots, m\}$.

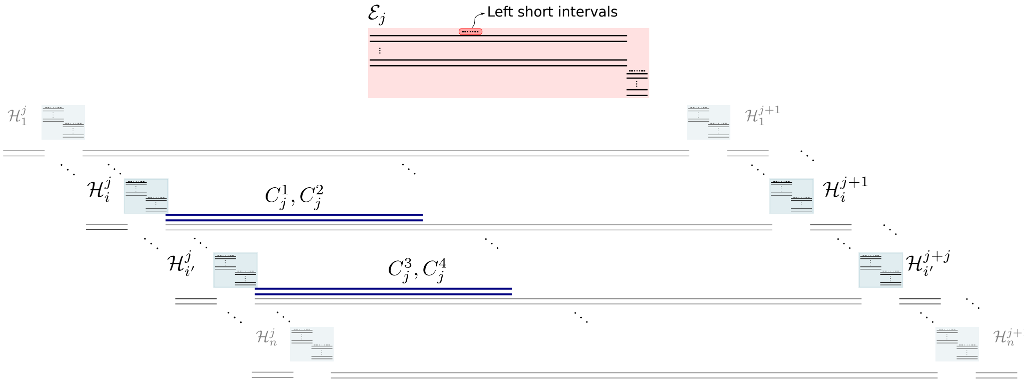


Fig. 4: General structure of the constructed interval model $\mathcal{M}(\mathfrak{G})$ highlighting the intersections between the intervals of the (n, m)-escalator $\mathcal{D}$, the intervals of the (p', q')-grained gadget $\mathcal{E}_j$, and the intervals $C_j^1, C_j^2, C_j^3, C_j^4$.

Now, we add the gadgets related to the edges. Please refer to Figure 4 to follow the construction. The values of p', q' used next will be defined later. For each edge $e_j = v_i v_{i'} \in E(G)$, with $i < i'$, create a (p', q')-grained gadget $\mathcal{E}_j$ and intervals $C_j^1, C_j^2, C_j^3, C_j^4$ in such a way that $\mathcal{E}_j$ is entirely contained in the j-th region (i.e., in the open interval between the right endpoint of $\mathcal{H}_n^j$ and the left endpoint of $\mathcal{H}_1^{j+1}$), C_j^1 and C_j^2 weakly intersect $\mathcal{H}_i^j$ to the right and weakly intersect $\mathcal{E}_j$ to the left, and C_j^3 and C_j^4 weakly intersect $H_{i'}^j$ to the right and strongly intersect $\mathcal{E}_j$ to the left. We call the intervals in $\{C_j^i \mid i \in \{1, \ldots, 4\}, j \in \{1, \ldots, m\}\}$ intervals *of type* C. Denote the constructed model by $\mathcal{M}(\mathfrak{G})$ (or simply by $\mathcal{M}$ when $\mathfrak{G}$ is clear from the context), which defines the reduction graph $\mathbb{G}_{\mathcal{M}(\mathfrak{G})}$.

The following straightforward lemma will be useful in the next section.

Lemma 2 *Let G be a graph, $\pi_V = (v_1, \ldots, v_n)$ and $\pi_E = (e_1, \ldots, e_m)$ be orderings of $V(G)$ and $E(G)$, respectively, and $\mathcal{M}$ be the model constructed as before. The following holds for every grained gadget $\mathcal{H}$:*

- *$\mathcal{M}$ respects the structure of $\mathcal{H}$;*
- *The number of intervals covering $\mathcal{H}$ is even; and*

- *The number of intervals strongly intersecting $\mathcal{H}$ to the left and the number of intervals strongly intersecting $\mathcal{H}$ to the right are both even.*

Observe that Lemma 2 implies that, in order for the values ℓ and r in Lemma 1 to be odd, it suffices to choose odd values for q and q'.

2.3 Proof of Theorem 1: Maximum cut of the reduction graph

Consider a cubic graph G on n vertices and $m = 3n/2$ edges, and let $\pi_V = (v_1, \ldots, v_n)$ be an ordering of $V(G)$, $\pi_E = (e_1, \ldots, e_m)$ be an ordering of $E(G)$ and $\mathfrak{G} = (G, \pi_V, \pi_E)$. We now prove that $\mathsf{mc}(G) \geq k$ if and only if $\mathsf{mc}(\mathbb{G}_{\mathcal{M}(\mathfrak{G})}) \geq f(G, k)$, where f is a function defined at the end of this subsection. As it is usually the case in this kind of reduction, given a cut of G, constructing an appropriate cut of the reduction graph $\mathbb{G}_{\mathcal{M}(\mathfrak{G})}$ is an easy task. On the other hand, constructing an appropriate cut $[X, Y]$ of G, from a given a cut $[A, B]$ of the reduction graph $\mathbb{G}_{\mathcal{M}(\mathfrak{G})}$, requires that the intervals in $\mathcal{M}(\mathfrak{G})$ behave in a way with respect to $[A, B]$ so that $[X, Y]$ can be inferred, a task achieved by appropriately manipulating the values of p, q, p', q', as done in Lemma 1. We start by giving conditions on these values that ensure that the partitioning of the edge gadget related to an edge $e_j = v_i v_{i'}$, with $i < i'$, depends solely on the partitioning of $\mathcal{H}^j_{i'}$.

Lemma 3 *Let G be a cubic graph, $\pi_V = (v_1, \ldots, v_n)$ and $\pi_E = (e_1, \ldots, e_m)$ be orderings of $V(G)$ and $E(G)$, respectively, and $\mathcal{M}(\mathfrak{G})$ be the model constructed as before, where $\mathfrak{G} = (G, \pi_V, \pi_E)$. Also, let $[A, B]$ be a maximum cut of $\mathbb{G}_{\mathcal{M}(\mathfrak{G})}$, and consider $e_j = v_i v_{i'}$, $i < i'$. If $\mathcal{M}(\mathfrak{G})$ is well-valued, and $q > 4n + p' + q'$, then*

1. *If $\mathcal{H}^j_i$ is A-partitioned by $[A, B]$, then $\{C^1_j, C^2_j\} \subseteq B$; otherwise, $\{C^1_j, C^2_j\} \subseteq A$; and*
2. *If $\mathcal{H}^j_{i'}$ is A-partitioned by $[A, B]$, then $\{C^3_j, C^4_j\} \subseteq B$ and $\mathcal{E}_j$ is A-partitioned by $[A, B]$; otherwise, $\{C^3_j, C^4_j\} \subseteq A$ and $\mathcal{E}_j$ is B-partitioned by $[A, B]$.*

Proof Denote $\mathcal{M}(\mathfrak{G})$ by $\mathcal{M}$ for simplicity. Since $\mathcal{M}$ is well-valued, by Lemma 1, we may assume that $\mathcal{H}^j_i$ is A-partitioned by $[A, B]$, i.e., that $\mathcal{LS} \cup \mathcal{RL} \subseteq A$ and $\mathcal{LL} \cup \mathcal{RS} \subseteq B$. We make the arguments for C^1_j and it will be clear that they also hold for C^2_j. Observe first that all the grained gadgets covered by C^1_j have a balanced number of intervals in A and in B. More formally, from the intervals within the gadgets $\bigcup_{\ell=i+1}^{n} \mathcal{H}^j_\ell$, which are all the intervals covered by C^1_j, there are exactly $(n-i)(p+q)$ intervals in A, and $(n-i)(p+q)$ intervals in B. Additionally, there are at most $2(n-i)$ link intervals intersecting the left of C^1_j (these are the link intervals related to $v_{i''}$ for $i'' > i$ in the $(j-1)$-th region, if $j > 1$), exactly $2(n-i)$ link intervals intersecting the right of C^1_j (these are the link intervals related to $v_{i''}$ for $i'' > i$ in the j-th region), and exactly $2i$ link intervals covering C^1_j (these are the link intervals related to $v_{i''}$ for $i'' \leq i$ in the j-th region). This is a total of at most $2(n-i)+2(n-i)+2i = 4n-2i < 4n$ intervals. Therefore, if C_j^1 is in A, then there are less than $(n-i)(p+q)+4n+q'$ edges of the cut incident to C_j^1; while if C_j^1 is in B, then there are at least $(n-i)(p+q)+q$ edges of the cut incident to C_j^1. Because $q > 4n+p'+q' \geq 4n+q'$, if follows that C_j^1 is in B, and so does C_j^2.

Observe that a similar argument can be applied to C_j^3, C_j^4, except that we gain also p' new edges from the left short intervals of $\mathcal{E}_j$. That is, supposing $\mathcal{H}_{i'}^j$ is A-partitioned by $[A,B]$, if C_j^3 is in A, then there are less than $(n-i')(p+q)+4n+p'+q'$ edges of the cut incident to C_j^3; while if C_j^3 is in B, then there are at least $(n-i')(p+q)+q$ edges of the cut incident to C_j^3. It follows again that C_j^3, C_j^4 are in B, since $q > 4n+p'+q'$.

Finally, suppose that $\mathcal{H}_{i'}^j$ is A-partitioned by $[A,B]$, in which case, from the previous paragraph, we get that $\{C_j^3, C_j^4\} \subseteq B$. Observe that, because $\mathcal{E}_j$ is either A-partitioned or B-partitioned, then there will always be exactly $(p'+q')$ intervals of $\mathcal{E}_j$ within A and exactly $(p'+q')$ within B. This means that the number of edges in the cut between this gadget and the link intervals covering it do not change if we just flip sides of every interval within $\mathcal{E}_j$. Therefore, we need to focus only on the gain between $\mathcal{E}_j$ and $\{C_j^1, C_j^2, C_j^3, C_j^4\}$. It is clear then that it is more advantageous for $\mathcal{E}_j$ to be A-partitioned since in this case we gain at least $2p'$ edges (the ones between C_j^3, C_j^4 and the left short intervals of $\mathcal{E}_j$), while otherwise we gain at most $2q'$ (recall that $p' > q'$ since $\mathcal{M}$ is well-valued). □

After ensuring that each grained gadget behaves well individually, we also need to ensure that $\mathcal{H}_i^1$ can be used to decide in which part of $[X,Y]$ we should put v_i, and for this it is necessary that all gadgets related to v_i agree with one another. In other words, for each v_i, we want that the behaviour of the first gadget $\mathcal{H}_i^1$ influence the behaviour of the subsequent gadgets $\mathcal{H}_i^2, \ldots, \mathcal{H}_i^{m+1}$, as well as the behaviour of the gadgets related to edges incident to v_i. Given $v_i \in V(G)$ and a cut $[A,B]$ of $\mathbb{G}_{\mathcal{M}(\mathfrak{G})}$, we say that *the gadgets of* v_i *alternate in* $[A,B]$ if, for every $j \in \{1,\ldots,m\}$, we get that $\mathcal{H}_i^j$ is A-partitioned if and only if $\mathcal{H}_i^{j+1}$ is B-partitioned, while L_i^{2j-1}, L_i^{2j} are opposite to the right long intervals of $\mathcal{H}_i^j$. Also, we say that $[A,B]$ is *alternating partitioned* if the gadgets of v_i alternate in $[A,B]$, for every $v_i \in V(G)$. We add a further condition on the values of p,q,p',q' in order to ensure that every maximum cut is alternating partitioned. After this, we use the good behaviour of the constructed model in order to relate the sizes of the maximum cuts in G and in $\mathbb{G}_{\mathcal{M}(\mathfrak{G})}$.

Lemma 4 *Let G be a cubic graph, $\pi_V = (v_1,\ldots,v_n)$ and $\pi_E = (e_1,\ldots,e_m)$ be orderings of $V(G)$ and $E(G)$, respectively, and $\mathcal{M}(\mathfrak{G})$ be the model constructed as before, where $\mathfrak{G} = (G,\pi_V,\pi_E)$. Also, let $[A,B]$ be a maximum cut of $\mathbb{G}_{\mathcal{M}(\mathfrak{G})}$. If $\mathcal{M}(\mathfrak{G})$ is well-valued, $q > 4n+p'+q'$, and $q > 3(2n^2+n+q'+2)$, then $[A,B]$ is alternating partitioned.*

Proof By hypothesis, the conditions of Lemmas 1 and 3 are satisfied. Thus, we can suppose that the obtained properties of those lemmas hold. Denote $\mathcal{M}(\mathfrak{G})$ by $\mathcal{M}$ for simplicity, and let $\mathcal{M}_i$ be the family of all the intervals related to vertex v_i; more specifically, it contains every interval in some grained gadget $\mathcal{H}_i^j$, $j \in \{1,\dots,m+1\}$, every link interval L_i^j, $j \in \{1,\dots,2m\}$, every interval of type C_j^h that intersects $\mathcal{H}_i^j$ to the right (this happens if e_j has v_i as endpoint), and every interval in $\mathcal{E}_j$ for e_j incident to v_i. We count the number f_i of edges of the cut incident to some interval in $\mathcal{M}_i$ and argue that, if the gadgets of v_i do not alternate in $[A,B]$, then we can obtain a bigger cut by rearranging $\mathcal{M}_i$, thus getting a contradiction.

Denote by $\overline{\mathcal{M}}_i$ the set of intervals $\mathcal{M} \setminus \mathcal{M}_i$, and by Λ the set of all link intervals.

In what follows, there are some values that must be added to f_i that remain the same, independently of how $\mathcal{M}_i$ is partitioned; we call these values *irrelevant* and do not add them to f_i. For instance, recall that every (x,y)-grained gadget has exactly $x+y$ intervals in A and $x+y$ in B. Thus, because of Lemmas 1 and 3, the number of edges of the cut between grained gadgets and intervals that cover them is irrelevant. In what follows, we count the other possible edges.

First, consider $j \in \{1,\dots,m\}$; we want to count the maximum number of edges of the cut incident to L_i^{2j} (which holds analogously for L_i^{2j-1}). Denote by ℓ_A^j the number of intervals in $\overline{\mathcal{M}}_i \cap \Lambda \cap A$ that intersect L_i^{2j}; define ℓ_B^j similarly. Observe that $\ell_A^j + \ell_B^j < 4n-2$ since it includes at most $2(n-1)$ link intervals in the j-th region, plus at most $2(n-i)$ link intervals of the $(j-1)$-th region, and at most $2(i-1)$ link intervals of the $(j+1)$-th region. Additionally, let a_j be equal to 1 if L_i^{2j} is opposite to the right long intervals of $\mathcal{H}_i^j$, and 0 otherwise; similarly, let b_j be equal to 1 if L_i^{2j} is opposite to the left long intervals of $\mathcal{H}_i^{j+1}$, and 0 otherwise. Because L_i^{2j} might also be opposite to $C_j^1,\dots,C_j^4$, observe that the relevant number of edges of the cut incident to L_i^{2j} is at most $q(a_j+b_j) + \ell_A^j + \ell_B^j + 4$.

Now, let e_j be an edge incident to v_i and let $v_{i'}$ be the other endpoint of e_j (here i' might be smaller than i). We apply Lemma 3 in order to count the edges incident to $\mathcal{E}_j \cup \{C_j^1,\dots,C_j^4\}$. First observe that, since $\mathcal{E}_j$ is always partitioned according to C_j^3, C_j^4, we have an irrelevant value of $2p'$, namely the edges between C_j^3, C_j^4 and the left short intervals of $\mathcal{E}_j$. Now, suppose, without loss of generality, that $\{C_j^1, C_j^2\} \subseteq A$. If $\{C_j^3, C_j^4\} \subseteq A$, then there are no relevant edges to be added; otherwise, then we get $2q'+4$ edges, those between C_j^1, C_j^2 and C_j^3, C_j^4, and between C_j^1, C_j^2 and the left long intervals of $\mathcal{E}_j$. Finally, observe that the edges between $\{C_j^1,\dots,C_j^4\}$ and $\mathcal{H}_i^j$ are irrelevant because of Lemma 3, and that the edges between $\{C_j^1,\dots,C_j^4\}$ and the link intervals have been counted previously.

In order to put everything together, let e_{j_1}, e_{j_2}, e_{j_3} be all the edges incident to v_i, and for each $h \in \{1,2,3\}$, write e_{j_h} as $\{v_i, v_{i_h}\}$ (we use set notation because i is not necessarily smaller than i_h). For each $h \in \{1,2,3\}$, let c_h be equal to 1 if $\mathcal{H}_i^j$ and $\mathcal{H}_{i_h}^j$ are partitioned differently, and 0 otherwise. We then get that:

$$f_i \le 2\sum_{j=1}^{m}(q(a_j+b_j) + \ell_A^j + \ell_B^j + 4) + \sum_{h=1}^{3} c_h(2q'+4). \tag{1}$$

If L_i^{2j} is on the same side as the right long intervals of $\mathcal{H}_i^j$ and the left long intervals of $\mathcal{H}_i^{j+1}$, we can increase f_i simply by switching the side of L_i^{2j}. Indeed, in this case we would lose at most $\max\{\ell_A^j, \ell_B^j\} + 4 < 4n + 2$ edges, while gaining $2q$, a positive exchange since $q > 4n$. Observe that this implies $a_j + b_j \geq 1$. Note also that this type of argument can be always applied, i.e., whenever in what follows we switch side of the intervals in some vertex gadget, we can suppose that this property still holds.

Consider now j to be minimum such that $\mathcal{H}_i^j$ and $\mathcal{H}_i^{j+1}$ are partitioned in the same way, say they are both A-partitioned. Note that this implies that $a_j + b_j = 1$, since the right long intervals of $\mathcal{H}_i^j$ are in A, while the left long intervals of $\mathcal{H}_i^{j+1}$ are in B. We want to switch sides of $\mathcal{H}_i^{j+1}$, but in order to ensure an increase in the size of the cut, we need to also switch subsequent grained gadgets in case they were alternating. For this, let $j' > j$ be minimum such that $\mathcal{H}_i^{j'+1}$ and $\mathcal{H}_i^{j'}$ are either both A-partitioned or both B-partitioned; if it does not exist, let $j' = m + 1$. For each $h \in \{j+1, \dots, j'\}$, we switch sides of $\mathcal{H}_i^h$, and put L_i^{2h-1}, L_i^{2h} in the side opposite to the right long intervals of $\mathcal{H}_i^h$. Also switch the intervals of type C and intervals in edge gadgets appropriately; i.e., in a way that Lemma 3 continues to hold. We prove that we gain at least $2q$ edges, while losing at most $m(8n+4)+6(q'+2) = 6(2n^2+n+q'+2)$ (recall that $m = 3n/2$); the result thus follows since $q > 3(2n^2 + n + q' + 2)$.

Observe first that, concerning intervals L_i^{2j-1} and L_i^{2j}, because now they are in B, while the right long intervals of $\mathcal{H}_i^j$ and the left long intervals of $\mathcal{H}_i^{j+1}$ are both in A, we gain at least $2q$ edges. Now, we count our losses. Concerning intervals L_i^{2j-1} and L_i^{2j}, we lose at most $2(\ell_B^j+4) \leq 8n+4$, namely the edges between these intervals and link intervals or intervals of type C. As for the intervals L_i^{2h-1}, L_i^{2h} for $h \in \{j+1, \dots, j'\}$, by the definition of j' we know that we lose at most $2(\max\{\ell_A^h, \ell_B^h\}+4) \leq 8n + 4$, while the number of edges of the cut between them and the vertex grained gadgets can only increase. Hence, concerning the link intervals in $\mathcal{M}_i$, in total we lose at most $m(8n + 4) = 6(2n^2 + n)$. Additionally, observe the upper bound given by (1) to see that, in the worst case scenario, we have $\{j_1, j_2, j_3\} \subseteq \{j + 1, \dots, j'\}$ and all the values c_h were previously equal to 1 (i.e. all three edges e_{j_1}, e_{j_2}, e_{j_3} belonged to the cut), and are now equal to 0 (i.e. none of the three edges e_{j_1}, e_{j_2}, e_{j_3} belong to the cut); this leads to a possible loss of at most $6(q' + 2)$ edges, as we wanted to show. □

Now, if $[A, B]$ is an alternating partitioned cut of $\mathbb{G}_{\mathcal{M}(\mathfrak{G})}$, we let $\Phi(A, B) = [X, Y]$ be the cut of G such that, for each vertex $v_i \in V(G)$, we have $v_i \in X$ if and only if $\mathcal{H}_i^1$ is A-partitioned by $[A, B]$. Note that $[X, Y]$ is well-defined and uniquely determined by $[A, B]$ (i.e., Φ is one-to-one). On the other hand, given a cut $[X, Y]$ of G, there is a unique alternating partitioned cut $[A, B] = \Phi^{-1}(X, Y)$ of $\mathbb{G}_{\mathcal{M}(\mathfrak{G})}$ such that $[X, Y] = \Phi(A, B)$ (i.e., Φ is also onto). Therefore, it remains to relate the sizes of these cut-sets. Basically we can use the good behaviour of the cuts in $\mathbb{G}_{\mathcal{M}(\mathfrak{G})}$ to prove that the size of $[A, B]$ grows as a function of the size of $\Phi(A, B)$.

Lemma 5 *Suppose that all the conditions in Lemmas 1-4 hold, and that $q' \geq 13n^2$. Let $\Phi(A, B) = [X, Y]$, and k be a positive integer. Then (below, $\mathbb{G}$ denotes $\mathbb{G}_{\mathcal{M}(\mathfrak{G})}$)*

$$|E_G(X, Y)| \geq k \text{ if and only if } |E_{\mathbb{G}}(A, B)| \geq \gamma + (2q' + 4)k,$$

where γ is a well-defined function on m, n, p, q, p', q' (i.e., does not depend on $[A, B]$).

Proof We use the same notation as before and count the number of edges in $E_{\mathbb{G}}(A,B)$. We will count the number of edges of the cut-set separately in the following groups:

- among intervals of a vertex/edge grained-gadget;
- between intervals of a vertex grained-gadget and link intervals;
- between intervals of an edge grained-gadget and other intervals;
- among intervals of type C;
- among link intervals;
- between link intervals and intervals of type C; and
- between intervals of a vertex grained-gadget and intervals of type C.

First, we compute the number of edges of the cut-set within a given (x,y)-grained gadget. By Lemma 1, we get that this is exactly y^2+2xy. Since there are $(m+1)n$ (p,q)-grained gadgets (the ones related to the vertices), and m (p',q')-grained gadgets (the edge ones), we get a total of:

$$\beta_1 = n(m+1)(q^2+2pq) + m((q')^2 + 2p'q').$$

Now, we count the number of edges of the cut-set between a given vertex grained gadget $\mathcal{H} = \mathcal{H}_i^j$ and link intervals; again, denote the set of link intervals by Λ. If an interval I covers $\mathcal{H}$, then there are exactly $p+q$ edges between I and $\mathcal{H}$, since there are these many intervals of $\mathcal{H}$ in each of A and B. And if I intersects $\mathcal{H}$ either to the left or to the right, then there are exactly q edges between I and $\mathcal{H}$, since $\mathcal{M}$ is alternating partitioned (i.e., I is opposite to the corresponding long intervals of $\mathcal{H}$). It remains to count how many of each type of intervals there are. If $j \in \{2,\ldots,m\}$, then there are exactly $2n-2$ intervals covering $\mathcal{H}$, as well as 2 intervals intersecting $\mathcal{H}$ to the left, and 2 to the right; this gives a total of $(2n-2)(p+q)+4q = 2n(p+q)+2(q-p)$ edges between $\mathcal{H}$ and Λ. If $j=1$, then there are $2(i-1)$ intervals covering $\mathcal{H}$, and 2 intervals intersecting $\mathcal{H}$ to the right, thus giving a total of $2(i-1)(p+q)+2q$. Finally, if $j=m+1$, then there are $2(n-i)$ intervals covering $\mathcal{H}$, and 2 intervals intersecting $\mathcal{H}$ to the left, giving a total of $2(n-i)(p+q)+2q$. Summing up, we get:

$$\begin{aligned}\beta_2 &= \textstyle\sum_{j=2}^{m}\sum_{i=1}^{n}[2n(p+q)+2(q-p)]+\\ &\quad \textstyle\sum_{i=1}^{n}[2(i-1)(p+q)+2q+2(n-i)(p+q)+2q]\\ &= 2(m-1)n[n(p+q)+(q-p)]+2n[(n-1)(p+q)+2q]\\ &= 2n[(m-1)n(p+q)+(m-1)(q-p)+(n-1)(p+q)+2q]\\ &= 2mn[n(p+q)+q-p].\end{aligned}$$

We count now the number of edges of the cut-set between a given edge gadget $\mathcal{E}_j$ and an interval I intersecting it, and among intervals of type C. As before, if I covers $\mathcal{E}_j$, then there are exactly $(p'+q')$ edges between I and $\mathcal{E}_j$ in the cut. If I strongly intersects $\mathcal{E}_j$ to the left, then $I \in \{C_j^3, C_j^4\}$ and by Lemma 3 we get that this amounts to p'. Finally, if I weakly intersects $\mathcal{E}_j$ to the left, then this amounts to q', if e_j is within the cut, or to 0, otherwise. As for the number of edges between intervals of type C, by Lemma 3 one can see that this is equal to $4|E_G(X,Y)|$. Summing up, we get:

$$2nm(p'+q') + 2p'm + (2q'+4)|E_G(X,Y)|.$$

Denote the value $2nm(p'+q')+2p'm$ by β_3, and note that this is independent of $[A,B]$.

Let us now count the number of edges of the cut-set among link intervals. For this, denote by $\mathcal{L}^j$ the set of link intervals in the j-th region, i.e., $\mathcal{L}^j = \{L_i^{2j}, L_i^{2j-1} \mid i \in \{1,\ldots,n\}\}$. Also, denote by V_A^j the set of indices $i \in \{1,\ldots,n\}$ such that $\{L_i^{2j-1}, L_i^{2j}\} \subseteq A$; define V_B^j analogously and let $a = |V_A^j|$ and $b = |V_B^j|$. We count the number of edges of the cut between intervals of $\mathcal{L}^j$, for every $j \in \{1,\ldots,m\}$, and between intervals of $\mathcal{L}^j$ and intervals of $\mathcal{L}^{j+1}$, for every $j \in \{1,\ldots,m-1\}$, and then we sum up. So consider a region $j \in \{1,\ldots,m\}$, and observe that, because $[A,B]$ is alternating partitioned, we get that either j is odd and V_A^j contains exactly the indices of the vertices within Y, while V_B^j contains the indices of the vertices within X, or j is even and the reverse occurs. More formally: if j is odd, then $V_A^j = \{i \in \{1,\ldots,n\} \mid v_i \in Y\}$ and $V_B^j = \{i \in \{1,\ldots,n\} \mid v_i \in X\}$; and if j is even, then $V_A^j = \{i \in \{1,\ldots,n\} \mid v_i \in X\}$ and $V_B^j = \{i \in \{1,\ldots,n\} \mid v_i \in Y\}$. In either case, since for each index in V_A^j (resp. V_B^j), there is a pair of intervals in $\mathcal{L}^j \cap A$ (resp. $\mathcal{L}^j \cap B$), we get that the number of edges of the cut between intervals of $\mathcal{L}^j$ is equal to $4|X||Y| = 4ab$. Now, suppose $j \in \{1,\ldots,m-1\}$; we count the edges of the cut between $\mathcal{L}^j$ and $\mathcal{L}^{j+1}$. Again because $[A,B]$ is alternating partitioned, we know that if $V_A^j = \{i_1,\ldots,i_a\}$, then $V_B^{j+1} = V_A^j$, while $V_A^{j+1} = V_B^j = \{1,\cdots,n\} \setminus V_A^j$. Supposing $i_1 < \ldots < i_a$, this implies that there are exactly 4 edges between $\{L_{i_{a'}}^{2j+1}, L_{i_{a'}}^{2j+2}\}$ and $\{L_{i_{a''}}^{2j-1}, L_{i_{a''}}^{2j}\}$ for each $a', a'' \in \{1,\ldots,a\}$ with $a' < a''$. Summing up we get that there are $4\sum_{a'=1}^{a}(a-a') = 4\frac{a(a-1)}{2} = 2a(a-1)$ edges between $\mathcal{L}^j \cap A$ and $\mathcal{L}^{j+1} \cap B$. Analogously we can conclude that there are $2b(b-1)$ edges between $\mathcal{L}^j \cap B$ and $\mathcal{L}^{j+1} \cap A$. Summing up with the previous $4ab$, we get $2a^2 - 2a + 2b^2 - 2b + 4ab = 2[(a+b)^2 - (a+b)]$ edges of the cut incident to $\mathcal{L}^j$ for every $j \in \{1,\ldots,m-1\}$. Recall that $a+b = |X|+|Y| = n$ to see that this gives us $2n(n-1)$ edges. Finally, summing up for all $j \in \{1,\ldots,m-1\}$ and summing also the edges between link intervals in $\mathcal{L}^m$, we get that the number of edges of the cut incident to link intervals is equal to:

$$\sum_{j=1}^{m-1} 2n(n-1) + 4|X||Y| = n(n-1)(3n-2) + 4|X||Y|$$

Observe that $4(n-1) \leq 4|X||Y| \leq n(n+1)$, and denote the value $n(n-1)(3n-2)$ by β_4.

Now, observe that it remains to count the number of edges of the cut-set between link intervals and intervals of type C, and between intervals of type C and vertex grained gadgets. We start with the latter. Given an edge $e_j = v_i v_{i'}$, with $i < i'$, there are exactly $n-i$ vertex grained gadgets covered by C_j^1, C_j^2, and $n-i'$ vertex grained gadgets covered by C_j^3, C_j^4. Together with the q edges between each of these intervals of type C and the corresponding vertex gadgets (namely, $\mathcal{H}_i^j$ and $\mathcal{H}_{i'}^j$), we get a total of $2(n-i)(p+q) + 2(n-i')(p+q) + 4q$. Even though we cannot give a precise value below, observe that this value can be exactly computed during the construction. The upper bound is given just to make it explicit that this is a polynomial function. Also, below, for $e_j = v_i v_{i'}$, the value ℓ_j denotes i and r_j denotes i'.

$$\begin{aligned}\beta_5 &= \textstyle\sum_{j=1}^{m}[2(n-r_j)(p+q) + 2(n-\ell_j)(p+q) + 4q] \\ &= m[4n(p+q)] - 2\textstyle\sum_{j=1}^{m}(r_j+\ell_j)(p+q) + 4q \\ &\leq 4m[n(p+q)+q].\end{aligned}$$

Finally, we count the number of edges of the cut between link intervals and intervals of type C. This is the only part of the counting that will not be exact.

Again, consider an edge $e_j = v_iv_{i'}$, and first consider the interval C_j^1; we will see that the arguments hold for C_j^2, and that analogous arguments hold for C_j^3, C_j^4. Observe that C_j^1 intersects exactly the following link intervals: $L_{i''}^{2j-1}$ and $L_{i''}^{2j}$ for every $i'' \in \{1, \ldots, n\}$; and $L_{i''}^{2j-2}$ and $L_{i''}^{2j-3}$ for every $i'' \in \{i+1, \ldots, n\}$. This is a total of less than $4n$ link intervals. Because an analogous argument can be applied to C_j^2, C_j^3, C_j^4, we get a total of $16n$ possible edges in the cut-set, for each value of j, totalling $16nm = 24n^2$.

Let $\beta = \sum_{i=1}^{5} \beta_i$, and $\gamma = \beta + 4(n-1)$. We now prove that $|E_G(X,Y)| \geq k$ if and only if $|E_{\mathbb{G}}(A,B)| \geq \gamma + (2q'+4)k$. We have proved that:

$$\begin{aligned}\overbrace{\beta + 4(n-1)}^{\gamma} + (2q'+4)|E_G(X,Y)| \leq {} & |E_{\mathbb{G}}(A,B)| \\ \leq {} & \beta + n(n+1) + 24n^2 + (2q'+4)|E_G(X,Y)|.\end{aligned}$$

If $|E_G(X,Y)| \geq k$, then by the first inequality we have that $|E_{\mathbb{G}}(A,B)| \geq \beta + 4(n-1) + (2q'+4)k$. On the other hand, if $|E_{\mathbb{G}}(A,B)| \geq \beta + 4(n-1) + (2q'+4)k$, then by the second inequality we have that $|E_G(X,Y)| \geq k - \frac{25n^2-3n+4}{2q'+4} \geq k - \frac{26n^2}{2q'+4}$. Since $q' \geq 13n^2$, we get that $|E_G(X,Y)| > k-1$. □

To finish the proof that the reduction works, we simply need to choose appropriate values for p, q, p', q'. Recall all necessary conditions:

- For each (x,y)-grained gadget $\mathcal{H}$ in $\mathcal{M}$, let t be the number of intervals in $\mathcal{M} \setminus \mathcal{H}$ intersecting $\mathcal{H}$, ℓ be the number of intervals in $\mathcal{M}$ intersecting the left short intervals, and r be the number of intervals in $\mathcal{M}$ intersecting the right short intervals. Then we want that ℓ and r are both odd, and that $y > 2t$ and $x > t + 2y$ (from Lemma 1);
- $q > 4n + p' + q' = \alpha_1$ (from Lemma 3);
- $q > 3(2n^2 + n + q' + 2) = \alpha_2$ (from Lemma 4); and
- $q' \geq 13n^2$ (from Lemma 5).

By Lemma 2, we know that in order for the values r, ℓ in the first item to be odd, it suffices to choose q, q' to be odd. Observe that $n \geq 4$ since G is a cubic graph. For a given edge gadget $\mathcal{E}_j$, we know that there are exactly $2n+4$ intervals in $\mathcal{M} \setminus \mathcal{E}_j$ intersecting it, namely the link intervals and intervals of type C in the j-th region. Hence, if we choose $q' = 13n^2 + 1$, we satisfy all the conditions on q' since $13n^2 + 1 > 4n + 8$ holds for every $n \geq 4$. As for p', it suffices to choose $2n+4+2q'+1 = 26n^2+2n+7$. Now, for a given vertex gadget $\mathcal{H}_i^j$, if t is the number of intervals intersecting it, we get that $t \leq 2(n-1)+8 = 2n+6$. Therefore, in order to satisfy the conditions of Lemma 1, it suffices to ensure $q > 4n+12 \geq 2t$, and $p > 2q+2n+6 \geq 2q+t$. The stronger condition is on q, so we choose it first. We substitute the chosen values of p' and q' above, obtaining $\alpha_1 = 39n^2 + 6n + 8$ and $\alpha_2 = 45n^2 + 3n + 9$. It is straightforward to verify that, for $n \geq 4$, it holds that $47n^2 + 1 > \max\{4n + 12, \alpha_1, \alpha_2\}$. Hence, we choose q to be equal to $47n^2 + 1$, and as consequence we choose p to be equal to $94n^2 + 2n + 9$. To finish the proof of Theorem 1, it remains to prove that the interval count of our reduction graph is exactly four, which is done in the next subsection.

2.4 Proof of Theorem 1: Bounding the interval count

Consider a cubic graph G on n vertices and $m = 3n/2$ edges, and orderings π_V, π_E of the vertex set and edge set of G. Denote the triple (G, π_V, π_E) by $\mathfrak{G}$. First, we want to prove that the interval count of our constructed interval model $\mathcal{M}(\mathfrak{G})$ is at most 4. But observe that the construction of $\mathcal{M}(\mathfrak{G})$ is actually not unique, since the intervals are not uniquely defined; e.g., given such a model, one can obtain a model satisfying the same properties simply by adding $\epsilon > 0$ to all points defining the intervals. In what follows, we provide a construction of a uniquely defined interval model related to $\mathfrak{G}$ that satisfies the desired conditions and has interval count 4.

Consider our constructed interval model $\mathcal{M}(\mathfrak{G})$, and for each $j \in \{1, \ldots, m\}$, denote by $\mathcal{S}_j$ the set of intervals related to the j-th region, i.e., $\mathcal{S}_j = \mathcal{E}_j \cup \bigcup_{\ell=1}^{4} C_j^{\ell} \cup \bigcup_{i=1}^{n} (\mathcal{H}_i^j \cup \{L_i^{2j} \cup L_i^{2j-1}\})$. We show how to accommodate $\mathcal{S}_1$ within the closed interval $[0, 6n-2]$ in such a way that the same pattern can be adopted in the subsequent regions of $\mathcal{M}(\mathfrak{G})$ too, each time starting at multiples of $4n$. More specifically, letting $t = 4n$, we will accommodate $\mathcal{S}_j$ within $[t \cdot (j-1), 6n-2+t \cdot (j-1)]$. Assume $e_1 = v_i v_{i'}$, with $i < i'$. Below, we describe exactly which closed interval of the line corresponds to each interval $I \in \mathcal{S}_1$.

- For each $i \in \{1, \ldots, n\}$, the left long intervals of $\mathcal{H}_i^1$ are equal to $[2i-2, 2i-\frac{3}{2}]$ and the left short intervals are any choice of q distinct points within the open interval $(2i-2, 2i-\frac{3}{2})$, whereas the right long intervals of $\mathcal{H}_i^1$ are equal to $[2i-\frac{3}{2}, 2i-1]$ and the right short intervals are any choice of q distinct points within the open interval $(2i-\frac{3}{2}, 2i-1)$. Note that open intervals are used to locate the closed intervals of length zero, but that the short intervals themselves are not open.
- C_1^1 and C_1^2 are equal to $[2i-1, 2i+2n-2]$.
- C_1^3 and C_1^4 are equal to $[2i'-1, 2i'+2n-2]$.
- The left long intervals of $\mathcal{E}_1$ are equal to $[2n, 4n-1]$.
- The left short intervals of $\mathcal{E}_1$ are any choice of q' distinct points in the open interval $(2i+2n-2, 2i'+2n-2)$. Again, the open interval is used just to locate the closed intervals of length zero.
- The right long intervals of $\mathcal{E}_1$ are equal to $[4n-1, 4n-\frac{1}{2}]$ and the right short intervals are any choice of q' distinct points within the corresponding open interval.
- For each $i \in \{1, \ldots, n\}$, intervals L_i^1, L_i^2 are equal to $[2i-1, 4n+2(i-1)]$.

The suitable chosen lengths of the above defined closed intervals are (see Figure 5, where we denote by Λ the set of link intervals):

1. 0: short intervals of all grained gadgets (dots in Figure 5);
2. 1/2: left long and right long intervals of each $\mathcal{H}_i^1$, and right long intervals of $\mathcal{E}_1$ (red intervals in Figure 5);
3. $2n-1$: intervals $C_1^1, \ldots, C_1^4$, and left long intervals of $\mathcal{E}_1$ (blue intervals in Figure 5);
4. $4n-1$: intervals L_i^1 and L_i^2, for every $i \in [n]$ (orange intervals in Figure 5).

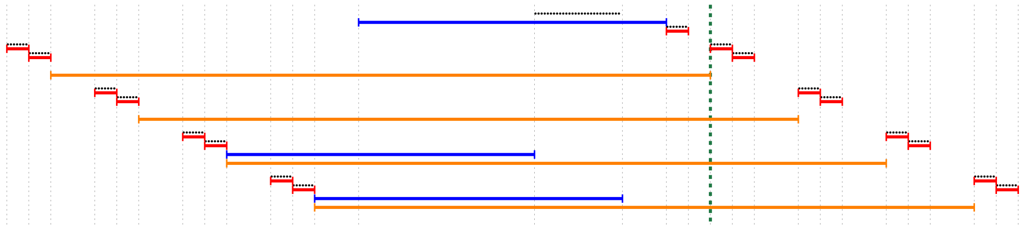


Fig. 5: The closed intervals in $\mathcal{S}_1 \cup \bigcup_{i=1}^{4} \mathcal{H}_i^2$ of a graph on 4 vertices. We consider e_1 to be equal to v_3v_4. Each colour represents a different interval size. The short intervals are represented by the dots located inside the open interval. Vertical lines mark the endpoints of the intervals in $\mathcal{S}_1 \setminus \Lambda$, while the green vertical line marks the beginning of the intervals in $\mathcal{S}_2$.

Now, let $\mathcal{M}'(\mathfrak{G})$ be the interval model where each $\mathcal{S}_j$ is defined exactly as $\mathcal{S}_1$, except that we shift all the intervals to the right in a way that point 0 now coincides with point $t \cdot (j-1)$. More formally, an interval I in $\mathcal{S}_j$ corresponding to the copy of an interval $[\ell, r]$ in $\mathcal{S}_1$ is defined as $[\ell + t \cdot (j-1), r + t \cdot (j-1)]$. Also, we assign the intervals in the $(m+1)$-th grained gadgets to be at the end of this model, using the same sizes of intervals as above; i.e., $\mathcal{H}_i^{m+1}$ is within the interval $[2i-2+t \cdot m, 2i-1+t \cdot m]$.

We have shown above that $\mathcal{M}'(\mathfrak{G})$ has interval count 4. The following lemma shows that the above chosen intervals satisfy the properties imposed in Subsections 2.1 and 2.2 on our constructed interval model $\mathcal{M}(\mathfrak{G})$.

Lemma 6 *Let G be a cubic graph. Then, there exists an interval model $\mathcal{M}(\mathfrak{G})$ with interval count 4 for $\mathfrak{G} = (G, \pi_V, \pi_E)$, for every ordering π_V and π_E of the vertex set and edge set of G, respectively.*

Proof Denote $\mathcal{M}(\mathfrak{G})$ by $\mathcal{M}$. We need to prove that $\mathcal{M}$ satisfies the conditions of our construction, namely:

1. For every $j \in \{1, \ldots, m\}$ and $i \in \{1, \ldots, n\}$, link intervals L_i^{2j}, L_i^{2j-1} weakly intersect $\mathcal{H}_i^j$ to the right and weakly intersects $\mathcal{H}_i^{j+1}$ to the left;
2. For every $j \in \{1, \ldots, m\}$ and $i, i' \in \{1, \ldots, n\}$, $i < i'$, the grained gadget $\mathcal{H}_i^j$ occurs strictly to the left of $\mathcal{H}_{i'}^j$;
3. For every $j \in \{1, \ldots, m\}$, grained gadget $\mathcal{E}_j$ occurs strictly between the right endpoint of $\mathcal{H}_n^j$ and the left endpoint of $\mathcal{H}_1^{j+1}$; and
4. For every $e_j = v_i v_{i'} \in E(G)$, intervals C_j^1, C_j^2 weakly intersect $\mathcal{H}_i^j$ to the right and $\mathcal{E}_j$ to the right, while C_j^3, C_j^4 weakly intersect $\mathcal{H}_{i'}^j$ to the left and stronly intersect $\mathcal{E}_j$ to the right.

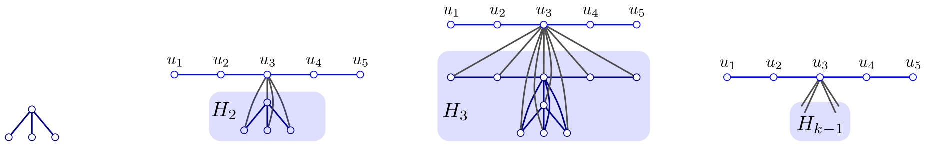
By construction, we know that the right endpoint of $\mathcal{H}_i^j$ is equal to $2i-1+t(j-1)$, which is also equal to the left endpoints of L_i^{2j-1}, L_i^{2j}. Also, the left endpoint of $\mathcal{H}_i^{j+1}$ is equal to $2i-2+tj$, which is also equal to the right endpoints of L_i^{2j-1}, L_i^{2j}

since $t = 4n$; hence Item 1 follows. As for Item 2, just note that the right endpoint of $\mathcal{H}_i^j$, which is equal to $2i - 1 + t(j-1)$, is strictly smaller than the left endpoint of $\mathcal{H}_{i'}^j$, which is equal to $2i' - 2 + t(j-1)$. Indeed, since $i' \geq i + 1$, we get $2i' - 2 \geq 2(i+1) - 2 = 2i > 2i - 1$. Now, observe that $\mathcal{E}_j$ is contained in the closed interval $[2n+t(j-1), 4n-\frac{1}{2}+t(j-1)]$, that the right endpoint of $\mathcal{H}_n^j$ is equal to $2n-1+t(j-1)$, and the the left endpoint of $\mathcal{H}_1^{j+1}$ is equal to $tj = 4n + t(j-1)$. Item 3 thus follows. Finally, as we have seen, the right endpoint of $\mathcal{H}_i^j$ is equal to $2i - 1 + t(j-1)$, which is equal to the left endpoints of C_j^1, C_j^2; hence these weakly intersect $\mathcal{H}_i^j$ to the right. Also, the left endpoint of $\mathcal{E}_j$ is equal to $2n + t(j-1)$, while the right endpoint of C_j^1, C_j^2 is equal to $2(i-1) + 2n + t(j-1)$, and all the left short intervals of $\mathcal{E}_j$ are contained in the open interval $[2(i-1) + 2n + t(j-1), 2(i'-1) + 2n + t(j-1)]$. Therefore we get that C_j^1, C_j^2 weakly intersect $\mathcal{E}_j$ to the left. Analogously, the right endpoint of $\mathcal{H}_{i'}^j$ is equal to $2i' - 1 + t(j-1)$, which is equal to the left endpoints of C_j^3, C_j^4; hence they weakly intersect $\mathcal{H}_{i'}^j$ to the right. Finally, the right endpoint of C_j^3, C_j^4 is equal to $2(i'-1) + 2n + t(j-1)$, and all the left short intervals of $\mathcal{E}_j$ are contained in the open interval $[2(i-1) + 2n + t(j-1), 2(i'-1) + 2n + t(j-1)]$. Also, the left endpoint of the right long intervals of $\mathcal{E}_j$ is equal to $4n - 1 + t(j-1)$, which is strictly bigger than $2(i'-1) + 2n + t(j-1)$ since $i' \leq n$. Therefore, C_j^3, C_j^4 strongly intersect $\mathcal{E}_j$ to the left, finishing the proof of Item 4. □

We have just shown that, for any orderings π_V and π_E, there exists a model $\mathcal{M}(\mathfrak{G})$ of interval count 4, where $\mathfrak{G} = (G, \pi_V, \pi_E)$. On the other hand, we prove in the remainder of this section that any graph isomorphic to $\mathbb{G}_{\mathcal{M}(\mathfrak{G})}$ has interval count at least 4. For this, we show that all such graphs contain as an induced subgraph a certain graph of interval count 4, which we denote by H_4. Next, we define the family $\{H_k\}_{k\geq 2}$ and prove in a more general way that $\mathsf{ic}(H_k) = k$ for every $k \geq 2$.

Let $P_5 = (u_1, \ldots, u_5)$ be a path on 5 vertices. For every graph H', we let $P_5 \circ H'$ be the graph obtained from the disjoint union of P_5 with H' by making u_3, the central vertex of P_5, adjacent to every vertex of H'. In other words, $P_5 \circ H'$ is the graph with vertex set $V(P_5) \cup V(H')$ and edge set $E(P_5) \cup E(H') \cup \{u_3 v \mid v \in V(H')\}$. Then, for every $k \geq 2$, we let H_k be the graph defined recursively as follows (see Figure 6):

- $H_2 = K_{1,3}$;
- $H_k = P_5 \circ H_{k-1}$ for $k > 2$.



(a) $H_2 = K_{1,3}$ (b) $H_3 = P_5 \circ H_2$ (c) $H_4 = P_5 \circ H_3$ (d) $H_k = P_5 \circ H_{k-1}$

Fig. 6: Graph H_k for $k \geq 2$.

Lemma 7 *For every $k \geq 2$, $\mathsf{ic}(H_k) = k$.*

Proof The proof is by induction on k. Since $H_2 = K_{1,3}$ and $\mathsf{ic}(K_{1,3}) = 2$ c.f. [25], we obtain that the lemma holds for $k = 2$. As inductive hypothesis, suppose that $\mathsf{ic}(H_{k'}) = k'$ for some $k' \geq 2$. We prove that $\mathsf{ic}(H_{k'+1}) = k' + 1$.

First, note that, if $\mathcal{M}_{P_5} = \{I_1, \ldots, I_5\}$ is an interval model of a P_5, then the precedence relation among the intervals of $I_1, \ldots, I_5$ is either that of Figure 7 (i.e., I_1 precedes I_3, which precedes I_5, and I_2 precedes I_4), or the reverse of the order presented in the figure c.f. [25]. Let $\mathcal{M}$ be an interval model of $H_{k'+1}$. Since $H_{k'+1}$ contains a P_5 as an induced subgraph, assume without loss of generality that $\mathcal{M} \supset \mathcal{M}_{P_5}$ and that, with respect to $\mathcal{M}$, I_1 precedes I_3, I_3 precedes I_5, and I_2 precedes I_4. This implies that

$$\ell(I_3) \leq r(I_2) < \ell(I_4) \leq r(I_3). \tag{2}$$

By construction, the only vertex of P_5 which is adjacent to the vertices of $H_{k'}$ is its central vertex u_3. Consequently, if $\mathcal{M}_{H_{k'}} \subset \mathcal{M}$ is the interval model of $H_{k'}$, then there cannot be any intersection between $\mathcal{M}_{H_{k'}}$ and $\mathcal{M}_{P_5} \setminus \{I_3\}$, i.e., $I' \cap I_i = \emptyset$ for each $I' \in \mathcal{M}_{H_{k'}}$ and each $i \in \{1, \ldots, 5\}$, with $i \neq 3$. Hence, it follows from (2) that

$$\min\{\ell(I') \mid I' \in \mathcal{M}_{H_{k'}}\} > r(I_2) \text{ and } \max\{r(I') \mid I' \in \mathcal{M}_{H_{k'}}\} < \ell(I_4).$$

Figure 7 illustrates this fact. As a result, $I_3 \supset I'$ for every $I' \in \mathcal{M}_{H_{k'+1}}$. This, along with the inductive hypothesis that $\mathsf{ic}(H_{k'}) = k'$, implies that $\mathsf{ic}(H_{k'+1}) \geq k' + 1$. On the other hand, it is straightforward that $\mathsf{ic}(H_{k'+1}) \leq k' + 1$ (for instance, consider the model illustrated in Figure 7). Therefore, $\mathsf{ic}(H_{k'+1}) = k' + 1$. □

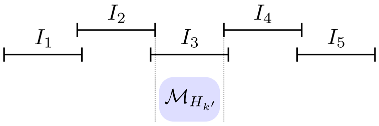


Fig. 7: Interval model $\mathcal{M}_{H_{k'+1}}$ of $H_{k'+1}$.

Now, we finally show that, if $\mathbb{G}'$ is a graph isomorphic to our reduction graph $\mathbb{G}_{\mathcal{M}(\mathfrak{G})}$, then $\mathbb{G}'$ has an H_4 as an induced subgraph. Let $\mathcal{M}'$ be an interval model of $\mathbb{G}'$. Note that $\mathcal{M}'$ holds the same intersection properties of $\mathcal{M}(\mathfrak{G})$ described in Section 2.2, otherwise $\mathbb{G}'$ would not be isomorphic to $\mathbb{G}_{\mathcal{M}(\mathfrak{G})}$. Consequently, there exist orderings π'_V and π'_E of the vertex set and edge set of G, respectively, for which $\mathcal{M}'$ can be defined as a composition of vertex grained gadgets $\mathcal{H}_i^j$ for $i \in \{1, \ldots, n\}$ and $j \in \{1, \ldots, m+1\}$, edge grained gadgets $\mathcal{E}_j$ for $j \in \{1, \ldots, m\}$, link intervals L_i^{2j-1}, L_i^{2j} for $i \in \{1, \ldots, n\}$ and $j \in \{1, \ldots, m\}$, and intervals $C_j^1, C_j^2, C_j^3, C_j^4$ for $j \in \{1, \ldots, m\}$. Additionally, since the input graph G is cubic, there exists an edge $e_j = (v_i, v_{i'}) \in E(G)$ such that, with respect to π'_V and π'_E, $1 < i < i'$. Thus, let (see Figure 4):

- I_1 (resp. I_2) be a right short (resp. long) interval of $\mathcal{H}_1^j$;
- I_3 be the link interval L_1^{2j-1};

- I_4 (resp. I_5) be a left long (resp. short) interval of $\mathcal{H}_1^{j+1}$;
- I'_1 (resp. I'_2) be a right short (resp. long) interval of $\mathcal{H}_i^j$;
- I'_3 be the interval C_j^1;
- I'_4 (resp. I'_5) be a left long (resp. short) interval of $\mathcal{E}_j$;
- J_1, J_2 and J_3 be three left short intervals of $\mathcal{H}_{i+1}^j$; and
- J be a left long interval of $\mathcal{H}_{i+1}^j$.

We note that the intervals described above exist and are well-defined in $\mathcal{M}'$. Moreover, the interval graph related to the model comprised by such intervals is isomorphic to H_4. More specifically, observe first that $\mathcal{J} = \{J, J_1, J_2, J_3\}$ models $K_{1,3}$. Then, notice that $\mathcal{P} = \{I_1, \ldots, I_5\}$ and $\mathcal{P}' = \{I'_1, \ldots, I'_5\}$ model paths on 5 vertices, in this order. Finally observe that I'_3 is adjacent to every $I \in \mathcal{J}$, while there are no edges between $\mathcal{J}$ and $\mathcal{P}' \setminus \{I'_3\}$; hence, $\mathcal{J} \cup \mathcal{P}'$ is a model for H_3. Similarly, I_3 is adjacent to every $I \in \mathcal{J} \cup \mathcal{P}'$, while there are no edges between $\mathcal{J} \cup \mathcal{P}'$ and $\mathcal{P} \setminus \{I_3\}$; hence $\mathcal{J} \cup \mathcal{P}' \cup \mathcal{P}$ is a model for H_4. Therefore, $\mathbb{G}'$ has an H_4 as an induced subgraph, as we wanted to prove.

3 The interval count of Adhikary et al.'s construction

We provided in Section 2 a reduction from the MaxCut problem having as input a cubic graph G into that of MaxCut in an interval graph G' having $\mathsf{ic}(G') \leq 4$. Although our reduction requires the choice of orderings π_V and π_E of respectively $V(G)$ and $E(G)$ in order to produce the resulting interval model, we have established that we are able to construct an interval model with interval count 4 regardless of the particular choices for π_V and π_E (Lemma 6). Our reduction was based on that of [2], strengthened in order to control the interval count of the resulting model.

This section is dedicated to discuss the interval count of the original reduction [2]. Although the interval count was not of concern in [2], in order to contrast the reduction found there with the presented in this work, we investigate how interval count varies in the original reduction considering different vertex/edge orderings. First, we establish that the original reduction yields an interval model corresponding to a graph G' such that $\mathsf{ic}(G') = O(\sqrt[4]{|V(G')|})$. Second, we exhibit an example of a cubic graph G for which a choice of π_V and π_E yields a model $\mathcal{M}'$ with interval count $\Omega(\sqrt[4]{|V(G')|})$, proving that this bound is tight for some choices of π_V and π_E. For bridgeless cubic graphs, we are able in Lemma 9 to decrease the upper bound by a constant factor, but to the best of our knowledge $O(\sqrt[4]{|V(G')|})$ is the tightest upper bound. Before we go further analysing the interval count of the original reduction, it is worthy to note that a tight bound on the interval count of a general interval graph G as a function of its number of vertices n is still open. It is known that $\mathsf{ic}(G) \leq \lfloor (n+1)/2 \rfloor$ and that there is a family of graphs G for which $\mathsf{ic}(G) = (n-1)/3$ [13, 23]. That is, the interval count of a graph can achieve $\Theta(n)$.

In the original reduction, given a cubic graph G, an interval graph G' is defined through the construction of one of its models $\mathcal{M}$, described as follows:

1. let $\pi_V = (v_1, v_2, \ldots, v_n)$ and $\pi_E = (e_1, e_2, \ldots, e_m)$ be arbitrary orderings of $V(G)$ and $E(G)$, respectively;
2. for each $v_i \in V(G)$, $e_j \in E(G)$, let $\mathcal{G}(v_i)$ and $\mathcal{G}(e_j)$ denote respectively a (p, q)-grained gadget and a (p', q')-grained gadget, where:
   - $q = 200n^3 + 1$, $p = 2q + 7n$, and
   - $q' = 10n^2 + 1$, $p' = 2q' + 7n$;
3. for each $v_k \in V(G)$, insert $\mathcal{G}(v_k)$ in $\mathcal{M}$ such that $\mathcal{G}(v_i)$ is entirely to the left of $\mathcal{G}(v_j)$ if and only if $i < j$. For each $e_k \in E(G)$, insert $\mathcal{G}(e_k)$ in $\mathcal{M}$ entirely to the right of $\mathcal{G}(v_n)$ and such that $\mathcal{G}(e_i)$ is entirely to the left of $\mathcal{G}(e_j)$ if and only if $i < j$;
4. for each $e_j = (v_i, v_{i'}) \in E(G)$, with $i < i'$, four intervals $I^1_{i,j}, I^2_{i,j}, I^1_{i',j}, I^2_{i',j}$ are defined in $\mathcal{M}$, called *link* intervals, such that:
   - $I^1_{i,j}$ and $I^2_{i,j}$ (resp. $I^1_{i',j}$ and $I^2_{i',j}$) are true twin intervals that weakly intersect $\mathcal{G}(v_i)$ (resp. $\mathcal{G}(v_{i'})$) to the right;
   - $I^1_{i,j}$ and $I^2_{i,j}$ (resp. $I^1_{i',j}$ and $I^2_{i',j}$) weakly intersect (resp. strongly intersect) $\mathcal{G}(e_j)$ to the left.

   By construction, therefore, $I^1_{i,j}$ and $I^2_{i,j}$ (resp. $I^1_{i',j}$ and $I^2_{i',j}$) cover all intervals in grained gadgets associated to a vertex v_ℓ with $\ell > i$ (resp. $\ell > i'$) or an edge e_ℓ with $\ell < j$.

Note that the number of intervals in $\mathcal{M}$ is invariant under the particular choices of π_V and π_E and, therefore, so is the number of vertices of G'. Let $n' = |V(G')|$. Since G is cubic, $m = \frac{3n}{2}$. By construction,

$$n' = n(2p + 2q) + m(2p' + 2q') + 4m = 1200n^4 + 90n^3 + 25n^2 + 21n$$

and thus $n = \Theta(\sqrt[4]{n'})$. Since the set of intervals covered by any link interval depends on π_V and π_E, distinct sequences yield distinct resulting graphs G' having distinct interval counts.

We show next that $\mathsf{ic}(G') = O(\sqrt[4]{n'})$. Note that

- the intervals of all gadgets $\mathcal{G}(v_i)$ and $\mathcal{G}(e_j)$ can use only two interval lengths (one for all short intervals, another for all the long intervals);
- for each $e_j = v_i v_{i'} \in E(G)$, with $i < i'$, both intervals $I^1_{i,j}$ and $I^2_{i,j}$ may be coincident in any model, and therefore may have the same length. The same holds for both intervals $I^1_{i',j}$ and $I^2_{i',j}$.

Therefore, $\mathsf{ic}(G') \leq 2m+2 = 3n+2 = \Theta(\sqrt[4]{n'})$. Therefore, the NP-completeness result derived from the original reduction in [2] can be strengthened to state that MaxCut is NP-complete for interval graphs G having interval count $O(\sqrt[4]{|V(G)|})$.

Second, we show that there is a resulting model $\mathcal{M}'$ produced in the reduction, defined in terms of particular orderings π_V, π_E for which $\mathsf{ic}(\mathcal{M}') = \Omega(\sqrt[4]{n'})$. Consider the cubic graph G depicted in Figure 8(a) which consists of an even cycle $(v_1, v_2, \ldots, v_n)$ with the addition of the edges $(v_i, v_{i+\frac{n}{2}})$ for all $1 \leq i \leq n/2$. For the ordering $\pi_V = (v_n, v_{n-1}, \ldots, v_1)$ and any ordering π_E in which the first n edges are the edges of the cycle $(v_1, v_2, \ldots, v_n)$, in this order, the reduction yields a model $\mathcal{M}'$ for which there is a chain $I^1_{1,1} \subset I^1_{2,2} \subset \ldots \subset I^1_{n,n}$ of nested intervals (see Figure 8(b)), which shows that $\mathsf{ic}(\mathcal{M}') \geq n$, and thus $\mathsf{ic}(\mathcal{M}') = \Omega(\sqrt[4]{n'})$.

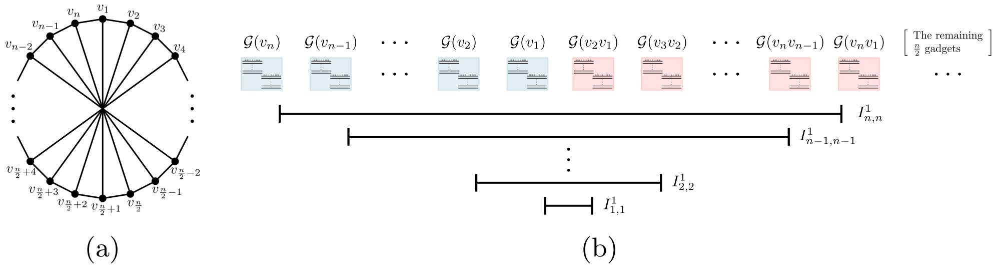


Fig. 8: (a) A cubic graph G, and (b) a chain of nested intervals in the model $\mathcal{M}'$.

It can be argued from the proof of NP-completeness for MaxCut when restricted to cubic graphs [24] that the constructed cubic graph may be assumed to have no bridges. This fact was not used in the original reduction of [2]. In an attempt to obtain a model $\mathcal{M}$ having fewer lengths for bridgeless cubic graphs, we have derived Lemma 9. Although the number of lengths in this new upper bound has decreased by the constant factor of 4/9, it is still $\Theta(n) = \Theta(\sqrt[4]{n'})$.

The proof of Lemma 9 will employ the following result:

Lemma 8 (Petersen, 1891) *Every cubic bridgeless graph admits a perfect matching.*

Lemma 9 *Let G be a cubic bridgeless graph with $n = |V(G)|$. There exist particular orderings π_V of $V(G)$ and π_E of $E(G)$ such that:*

1. *there is a resulting model $\mathcal{M}$ produced in the original reduction of* MaxCut *such that* $\mathsf{ic}(\mathcal{M}) \leq \frac{4n}{3} + 3$.
2. *for all such resulting models $\mathcal{M}$, we have that* $\mathsf{ic}(\mathcal{M}) \geq 5$ *if G is not a Hamiltonian graph.*

Proof Let G be a cubic bridgeless graph with $V(G) = \{v_1, v_2, \ldots, v_n\}$. By Lemma 8, G admits a perfect matching M. Let $H = G \setminus M$. Therefore, H is 2-regular and, therefore, H consists of a disjoint union of cycles $C_1, C_2, \ldots, C_k$, for some $k \geq 1$.

For all $1 \leq i \leq k$, let $\pi_V^i = v_1^i, v_2^i, \ldots, v_{k_i}^i$ be an ordering of the vertices of C_i, with $k_i = |C_i|$, such that $(v_j^i, v_{j+1}^i) \in E(C_i)$ for all $1 \leq j \leq k_i$, where $v_{k_i+1}^i = v_1^i$. Let π_E^i be the ordering $(v_1^i, v_2^i), (v_2^i, v_3^i), \ldots, (v_{k_i-1}^i, v_{k_i}^i), (v_1^i, v_{k_i}^i)$ for all $1 \leq i \leq k$. Let π_M be any ordering of the edges of M such that $(v_i, v_r) < (v_j, v_s)$ in π_M only if $v_i < v_j$ in π_V. Finally, let π_V be the ordering of $V(G)$ obtained from the concatenation of the orderings $\pi_V^1, \pi_V^2, \ldots, \pi_V^k$, and π_E be the ordering of $E(G)$ obtained from the concatenation of the orderings $\pi_E^1, \pi_E^2, \ldots, \pi_E^k, \pi_M$.

In order to prove (2.), assume G is not a Hamiltonian graph. Therefore $k > 1$. Observe that there is the following chain of nested intervals $I_1 \subset I_2 \subset I_3 \subset I_4 \subset I_5$, where

- I_1 is the leftmost interval in $\mathcal{RS}(\mathcal{G}(v_3^2))$,
- I_2 is an interval in $\mathcal{RL}(\mathcal{G}(v_3^2))$,
- I_3 is a link interval corresponding to both $\mathcal{G}(v_2^2)$ and $\mathcal{G}(v_1^2 v_2^2)$,
- I_4 is a link interval corresponding to both $\mathcal{G}(v_1^2)$ and $\mathcal{G}(v_1^2 v_{k_2}^2)$, and
- I_5 is a link interval corresponding to both $\mathcal{G}(v_1^1)$ and $\mathcal{G}(e)$, where e is the edge of M incident to v_1^1,

since $\ell(I_5) < \ell(I_4) < \ell(I_3) < \ell(I_2) < \ell(I_1) < r(I_1) < r(I_2) < r(I_3) < r(I_4) < r(I_5)$. Thus, for all such resulting models $\mathcal{M}$, we have that $\mathsf{ic}(\mathcal{M}) \geq 5$.

In order to prove (1.), we show that there exists an interval model $\mathcal{M}$, produced by the original reduction of MaxCut considering orderings π_V and π_E, such that $\mathsf{ic}(\mathcal{M}) \leq \frac{4n}{3} + 3$, where $n = |V(G)|$. Let L_1 be the set of all link intervals of the grained gadgets corresponding to edges of M, that is, $L_1 = \{I_{i,k}^1, I_{i,k}^2, I_{j,k}^1, I_{j,k}^2 : e_k = (i,j) \in M\}$. Moreover, let L_2 be the set of all link intervals of the grained gadgets corresponding to the edges $(v_1^i, v_{k_i}^i)$ of C_i and the vertex v_1^i for all $1 \leq i \leq k$, that is, $L_2 = \{I_{v_1^i,k}^1, I_{v_1^i,k}^2 : 1 \leq i \leq k\ ,\ e_k = (v_1^i, v_{k_i}^i) \in C_i\}$. Note that $|L_2| = k \leq n/3$ and $|L_1| = 4 \cdot |M| = 2n$. Let $L = L_1 \cup L_2$. Let $\mathcal{M}' = \mathcal{M} \setminus L$. We claim that $\mathsf{ic}(\mathcal{M}') \leq 3$. Since each pair of true twins $I_{j,k}^1, I_{j,k}^2$ and $I_{i,k}^1, I_{i,k}^2$ in L_1 can have the same length in $\mathcal{M}$, it follows from this claim that $\mathsf{ic}(\mathcal{M}) \leq |L_1| + \frac{|L_2|}{2} + \mathsf{ic}(\mathcal{M}') \leq \frac{n}{3} + n + 3 = \frac{4n}{3} + 3$, holding the result. It remains to show that the claim indeed holds.

To prove the claim, let $\mathcal{M}''$ be the interval model obtained from $\mathcal{M}'$ by removing all intervals corresponding to the grained gadgets (or, in other words, by keeping only the intervals corresponding to link intervals). It is easily seen that $\mathcal{M}''$ is a proper interval model, that is, no interval is properly contained in another. Therefore, the interval graph corresponding to $\mathcal{M}''$ is a proper interval graph and $\mathcal{M}''$ can be modified so that their intervals have all a single length. Since it is possible to bring all grained gadgets back to $\mathcal{M}''$ using two more lengths, we have that $\mathsf{ic}(\mathcal{M}') \leq 3$, as claimed. □

As a concluding remark, we note that the interval count of the interval model $\mathcal{M}$ produced in the original reduction is highly dependent on the assumed orderings of $V(G)$ and $E(G)$, and may achieve $\mathsf{ic}(\mathcal{M}) = \Omega(\sqrt[4]{n'})$. The model $\mathcal{M}'$ produced in our reduction enforces that $\mathsf{ic}(\mathcal{M}') = 4$ which is invariant for any such orderings. On the perspective of the problem of interval count 2 and beyond, for which very little is known, our NP-completeness result on a class of bounded interval count graphs is also of interest.

Acknowledgements

We thank Vinicius F. Santos who shared Reference [2], and anonymous referees for many valuable suggestions, including improving the interval count from 5 to 4.

Data availability statement

Data sharing not applicable to this article as no datasets were generated or analysed during the current study.

References

[1] Garey, M.R., Johnson, D.S., Stockmeyer, L.J.: Some simplified NP-complete graph problems. Theor. Comput. Sci. **1**(3), 237–267 (1976). https://doi.org/10.1016/0304-3975(76)90059-1

[2] Adhikary, R., Bose, K., Mukherjee, S., Roy, B.: Complexity of maximum cut on interval graphs. In: Buchin, K., de Verdière, É.C. (eds.) 37th International Symposium on Computational Geometry, SoCG 2021, June 7-11, 2021, Buffalo, NY, USA (Virtual Conference). LIPIcs, vol. 189, pp. 7–1711. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, Dagstuhl, Germany (2021). https://doi.org/10.4230/LIPIcs.SoCG.2021.7

[3] Johnson, D.S.: The NP-completeness column: An ongoing guide. J. Algorithms **6**(3), 434–451 (1985). https://doi.org/10.1016/0196-6774(85)90012-4

[4] de Figueiredo, C.M.H., de Melo, A.A., Sasaki, D., Silva, A.: Revising Johnson's table for the 21st century. Discret. Appl. Math. (2021). https://doi.org/10.1016/j.dam.2021.05.021

[5] Ekim, T., Erey, A., Heggernes, P., van 't Hof, P., Meister, D.: Computing minimum geodetic sets of proper interval graphs. In: Fernández-Baca, D. (ed.) LATIN 2012: Theoretical Informatics - 10th Latin American Symposium, Arequipa, Peru, April 16-20, 2012. Proceedings. Lecture Notes in Computer Science, vol. 7256, pp. 279–290. Springer, Berlin, Heidelberg (2012). https://doi.org/10.1007/978-3-642-29344-3_24

[6] Chakraborty, D., Das, S., Foucaud, F., Gahlawat, H., Lajou, D., Roy, B.: Algorithms and complexity for geodetic sets on planar and chordal graphs. In: Cao, Y., Cheng, S., Li, M. (eds.) 31st International Symposium on Algorithms and Computation, ISAAC 2020, December 14-18, 2020, Hong Kong, China (Virtual Conference). LIPIcs, vol. 181, pp. 7–1715. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, Dagstuhl, Germany (2020). https://doi.org/10.4230/LIPIcs.ISAAC.2020.7

[7] Cohen, J., Fomin, F.V., Heggernes, P., Kratsch, D., Kucherov, G.: Optimal linear arrangement of interval graphs. In: Kralovic, R., Urzyczyn, P. (eds.) Mathematical Foundations of Computer Science 2006, 31st International Symposium, MFCS 2006, Stará Lesná, Slovakia, August 28-September 1, 2006, Proceedings. Lecture Notes in Computer Science, vol. 4162, pp. 267–279. Springer, Berlin, Heidelberg (2006). https://doi.org/10.1007/11821069_24

[8] Jinjiang, Y., Sanming, Z.: Optimal labelling of unit interval graphs. Applied Math. **10**(3), 337–344 (1995). https://doi.org/10.1007/bf02662875

[9] Nicoloso, S., Sarrafzadeh, M., Song, X.: On the sum coloring problem on interval graphs. Algorithmica **23**(2), 109–126 (1999). https://doi.org/10.1007/PL00009252

[10] Marx, D.: A short proof of the NP-completeness of minimum sum interval coloring. Oper. Res. Lett. **33**(4), 382–384 (2005). https://doi.org/10.1016/j.orl.2004.07.006

[11] Corneil, D.G., Kim, H., Natarajan, S., Olariu, S., Sprague, A.P.: Simple linear time recognition of unit interval graphs. Inf. Process. Lett. **55**(2), 99–104 (1995). https://doi.org/10.1016/0020-0190(95)00046-F

[12] de Figueiredo, C.M.H., Meidanis, J., de Mello, C.P.: A linear-time algorithm for proper interval graph recognition. Inf. Process. Lett. **56**(3), 179–184 (1995). https://doi.org/10.1016/0020-0190(95)00133-W

[13] Cerioli, M.R., de S. Oliveira, F., Szwarcfiter, J.L.: The interval count of interval graphs and orders: a short survey. J. Braz. Comput. Soc. **18**(2), 103–112 (2012). https://doi.org/10.1007/s13173-011-0047-1

[14] Cerioli, M.R., Oliveira, F.S., Szwarcfiter, J.L.: On counting interval lengths of interval graphs. Discret. Appl. Math. **159**(7), 532–543 (2011). https://doi.org/10.1016/j.dam.2010.07.006

[15] Klavík, P., Otachi, Y., Sejnoha, J.: On the classes of interval graphs of limited nesting and count of lengths. Algorithmica **81**(4), 1490–1511 (2019). https://doi.org/10.1007/s00453-018-0481-y

[16] Bodlaender, H.L., Kloks, T., Niedermeier, R.: SIMPLE MAX-CUT for unit interval graphs and graphs with few p_4s. Electron. Notes Discret. Math. **3**, 19–26 (1999). https://doi.org/10.1016/S1571-0653(05)80014-9

[17] Boyaci, A., Ekim, T., Shalom, M.: A polynomial-time algorithm for the maximum cardinality cut problem in proper interval graphs. Inf. Process. Lett. **121**, 29–33 (2017). https://doi.org/10.1016/j.ipl.2017.01.007

[18] Bodlaender, H.L., de Figueiredo, C.M.H., Gutierrez, M., Kloks, T., Niedermeier, R.: Simple max-cut for split-indifference graphs and graphs with few p_4's. In: Ribeiro, C.C., Martins, S.L. (eds.) Experimental and Efficient Algorithms, Third International Workshop, WEA 2004, Angra Dos Reis, Brazil, May 25-28, 2004, Proceedings. Lecture Notes in Computer Science, vol. 3059, pp. 87–99. Springer, Berlin, Heidelberg (2004). https://doi.org/10.1007/978-3-540-24838-5_7

[19] Kratochvíl, J., Masarík, T., Novotná, J.: U-bubble model for mixed unit interval graphs and its applications: The maxcut problem revisited. In: Esparza, J., Král', D. (eds.) 45th International Symposium on Mathematical Foundations of Computer Science, MFCS 2020, August 24-28, 2020, Prague, Czech Republic. LIPIcs, vol. 170, pp. 57–15714. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, Dagstuhl, Germany (2020). https://doi.org/10.4230/LIPIcs.MFCS.2020.57

[20] de Figueiredo, C.M.H., de Melo, A.A., de S. Oliveira, F., Silva, A.: Maximum cut on interval graphs of interval count four is NP-complete. In: Bonchi, F., Puglisi, S.J. (eds.) 46th International Symposium on Mathematical Foundations of Computer Science, MFCS 2021, August 23-27, 2021, Tallinn, Estonia. LIPIcs, vol. 202, pp. 38–13815. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, Dagstuhl, Germany (2021). https://doi.org/10.4230/LIPIcs.MFCS.2021.38

[21] de Figueiredo, C.M.H., de Melo, A.A., de S. Oliveira, F., Silva, A.: Maximum cut on interval graphs of interval count five is NP-complete. CoRR **abs/2012.09804** (2020) https://arxiv.org/abs/2012.09804

[22] Bondy, J.A., Murty, U.S.R.: Graph Theory. Graduate Texts in Mathematics. Springer, New York (2008). https://doi.org/10.1007/978-1-84628-970-5

[23] Fishburn, P.C.: Interval graphs and interval orders. Discret. Math. **55**(2), 135–149 (1985). https://doi.org/10.1016/0012-365X(85)90042-1

[24] Berman, P., Karpinski, M.: On some tighter inapproximability results (extended abstract). In: Wiedermann, J., van Emde Boas, P., Nielsen, M. (eds.) Automata, Languages and Programming, 26th International Colloquium, ICALP'99, Prague, Czech Republic, July 11-15, 1999, Proceedings. Lecture Notes in Computer Science, vol. 1644, pp. 200–209. Springer, Berlin, Heidelberg (1999). https://doi.org/10.1007/3-540-48523-6_17

[25] Roberts, F.: Indifference graphs, F. Harary (Ed.), Proof Techniques in Graph Theory. Academic Press, New York (1969)

Appendix F

Manuscript: MaxCut on Permutation Graphs is NP-complete

This appendix contains the manuscript:
Celina M. H. de Figueiredo, Alexsander A. de Melo, Fabiano de Oliveira, Ana Silva. MaxCut on Permutation Graphs is NP-complete. Submitted in March 2022 to *Journal of Graph Theory* [36].

MaxCut on Permutation Graphs is NP-complete

Celina M. H. de Figueiredo[a], Alexsander A. de Melo[a], Fabiano S. Oliveira[b], Ana Silva[c,d]

[a]*Federal University of Rio de Janeiro, Rio de Janeiro, Brazil*
[b]*Rio de Janeiro State University, Rio de Janeiro, Brazil*
[c]*Federal University of Ceará, Ceará, Brazil*
[d]*Universitá degli Studi di Firenze, Italy*

Abstract

In this paper, we prove that the MaxCut problem is NP-complete on permutation graphs, settling a long-standing open problem that appeared in the 1985 column of the *Ongoing Guide to NP-completeness* by David S. Johnson.

1. Introduction

A *cut* is a partition of the vertex set of a graph into two disjoint parts, and the *maximum cut problem* (denoted by MaxCut, for short) aims to determine a cut with the maximum number of edges for which each endpoint is in a distinct part. The decision problem MaxCut is known to be NP-complete since the seventies [1], and only recently its restriction to interval graphs has been announced to be hard by Adhikary, Bose, Mukherjee, and Roy [2]. This settles a long-standing open problem from the *Ongoing Guide to NP-completeness* by David S. Johnson [3].

In his column, David S. Johnson presented a two-page summary table, with a column for each of the ten most famous NP-complete graph problems, and a row for each of thirty selected graph class. Among those graph classes, special emphasis was given to subclasses of perfect graphs and of intersection graphs having broad algorithmic significance. The emphasis was on the restrictions themselves and how they affect the complexity of the considered NP-hard problems. The discussion had focus on the particularly fertile domain of graph theory, where the central open problem at that time was the recognition of perfect graphs.

Many important graph classes are defined or can be characterized by a geometric intersection model. Two particularly well-studied examples are subclasses of perfect graphs: the classes of interval graphs and of permutation graphs [4, 5, 6]. In their respective models, the intersecting objects are line

Email addresses: celina@cos.ufrj.br (Celina M. H. de Figueiredo), aamelo@cos.ufrj.br (Alexsander A. de Melo), fabiano.oliveira@ime.uerj.br (Fabiano S. Oliveira), anasilva@mat.ufc.br (Ana Silva)

Preprint submitted to Journal of Graph Theory *July 12, 2022*

segments in the plane, with different restrictions imposed on their positions. In interval graphs, each line segment must have its endpoints on a single line, while in permutation graphs, their endpoints must lie on two distinct parallel lines.

Besides selecting the recognition of perfect graphs as the famous open problem, in his column, David S. Johnson selected only two others as open and may well be hard problems: Hamiltonian circuit restricted to permutation graphs and edge-coloring restricted to planar graphs. Today, we know that recognition of perfect graphs and Hamiltonian circuit restricted to permutation graphs can both be solved in polynomial time. On the other hand, edge-coloring restricted to planar graphs remains a challenging open problem. Please, refer to [7] for an updated summary table. Surprisingly, after 35 years, the only new resolved entry for permutation graphs is Hamiltonian circuit.

The present paper settles a long-standing open problem proposed by Johnson, by providing the first entry of Johnson's table for permutation graphs resolved as NP-complete.

Theorem 1. MaxCut *is NP-complete on permutation graphs.*

Our proof is based on Adhikary et al.'s construction used to prove the NP-completeness of MaxCut on interval graphs [2]. It is interesting to notice that, among the problems selected by Johnson, MaxCut is the only one classified as NP-complete for interval graphs and for permutation graphs. Despite that, the interval graph constructed by Adhikary et al. is not a permutation graph, and our constructed permutation graph is not an interval graph. Thus, we leave as an open question the complexity of MaxCut on permutation interval graphs.

Our paper is organized as follows. In Section 1.1, we present the basic concepts and notations. In Section 2, we present the main gadget in the reduction of Adhikary et al. [2], which also plays an important role in our reduction. In Section 3, we present the construction of Adhikary et al. [2] and show that it does not lead to a permutation graph. The presentation of their construction is also useful in Section 4, where we finally present the proof of Theorem 1. In Section 5, we prove that our constructed permutation graph is not an interval graph, and propose the complexity of MaxCut on permutation interval graphs as an open problem.

1.1. Preliminaries

In this work, all graphs considered are simple. For missing definitions and notation of graph theory, we refer to [8].

Let G be a graph. We say that a subset $K \subseteq V(G)$ is a *clique* if every two distinct vertices in K are adjacent, and that a subset $S \subseteq V(G)$ is a *stable set* if no two vertices in S are adjacent. Let X and Y be two disjoint subsets of $V(G)$. We say that X is *complete* to Y if every vertex in X is adjacent to every vertex in Y, and that X is *anti-complete* to Y if there are no edges between X and Y. We let $E_G(X, Y)$ be the subset of $E(G)$ with an endpoint in X and the other endpoint in Y. A *cut* of G is a partition of $V(G)$ into two parts $A, B \subseteq V(G)$, denoted by $[A, B]$; the edge set $E_G(A, B)$ is called the *cut-set* of G associated with $[A, B]$. For each two vertices $u, v \in V(G)$, we say that u and v *are in a same part of* $[A, B]$ if either $\{u, v\} \subseteq A$ or $\{u, v\} \subseteq B$; otherwise, we say that u and v *are in opposite parts of* $[A, B]$. Denote by $\mathsf{mc}(G)$ the maximum size of a cut-set of G. The MaxCut problem has as input a graph G and a positive integer k, and it asks whether $\mathsf{mc}(G) \geq k$.

Let π and π' be two permutations of a same set, say V. A graph G is called the *intersection graph related to* $\{\pi, \pi'\}$ if $V(G) = V$ and, for each two vertices $u, v \in V(G)$, $uv \in E(G)$ if and only if $u <_\pi v$ and $v <_{\pi'} u$. In this case, we also say that $\{\pi, \pi'\}$ is a *permutation model* of G. A graph is a *permutation graph* if it is the intersection graph related to a permutation model.

Given two permutations π and γ of disjoint subsets X and Y, respectively, we write $\pi\gamma$ to denote the permutation of $X \cup Y$ given by the *concatenation* of π with γ. Also, we write $\overleftarrow{\pi}$ to denote the reverse of the permutation π, that is, if $\pi = (v_1, \ldots, v_i)$, then $\overleftarrow{\pi} = (v_i, \ldots, v_1)$. In order to simplify the notation, given a set Z, we sometimes use the same symbol, Z, to denote also a chosen permutation of the elements of Z; in such cases, $\overleftarrow{Z}$ represents the reverse of the chosen permutation for Z.

An *interval model* is a finite multiset $\mathcal{M}$ of closed intervals of the real line. Let G be a graph and $\mathcal{M}$ be an interval model. An *$\mathcal{M}$-representation* of G is a bijection $\phi\colon V(G) \to \mathcal{M}$ such that, for every two distinct vertices $u, v \in V(G)$, we have that $uv \in E(G)$ if and only if $\phi(u) \cap \phi(v) \neq \emptyset$. If such an $\mathcal{M}$-representation exists, we say that $\mathcal{M}$ is an *interval model of* G and that G is an *interval graph*.

We write $i \in [n]$ to mean $i \in \{1, \ldots, n\}$.

2. Grained gadget

In this section, we present the notion of *grained gadgets*, which was defined in [9] as a generalization of the so-called *V-gadgets* and *E-gadgets*, these latter introduced by Adhikary et al. [2] in order to prove the NP-completeness of MaxCut on interval graphs.

Let x and y be positive integers. An *(x, y)-grained gadget* is a split graph H formed by a clique $K' \cup K''$ of size $2y$ and a stable set $S' \cup S''$ of size $2x$ with K' being complete to S', K'' being complete to S'', and satisfying $|K'| = |K''| = y$ and $|S'| = |S''| = x$. Figure 1 depicts an interval representation of an (x, y)-grained gadget. One can readily verify that the intersection graph related to the pair of permutations $\{K'S'S''K'', S'\overleftarrow{K''}\overleftarrow{K'}S''\}$ (see Figure 2) is an (x, y)-grained gadget. Thus, grained gadgets are interval graphs and permutation graphs.

Let H be an (x, y)-grained gadget and G be a supergraph of H. For each vertex $u \in V(G) \setminus V(H)$, we say that (see Figure 3): *u covers H* if $V(H) \subseteq N_G(u)$; *u weakly intersects H* if either $N_G(u) \cap V(H) = K'$ or $N_G(u) \cap V(H) = K''$; and that *u strongly intersects H* if either $N_G(u) \cap V(H) = K' \cup S'$ or $N_G(u) \cap V(H) = K'' \cup S''$. Moreover, we say that *G respects the structure* of H if, for each vertex $u \in V(G) \setminus V(H)$, either $N_G(u) \cap V(H) = \emptyset$ or u satisfies one of the previous conditions.

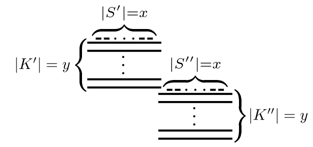


Figure 1: Interval representation of an (x, y)-grained gadget c.f.[9].

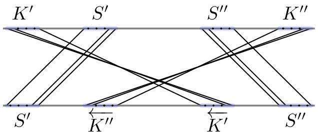


Figure 2: A permutation model of a grained gadget.

The next lemma establishes the key property of grained gadgets with respect to the MaxCut problem. Intuitively, it states that, for suitable values of x and y, if G is a supergraph that respects the structure of an (x, y)-grained gadget, then, in any maximum cut $[A, B]$ of G, the vertices belonging to $K' \cup S''$ are placed in a same part of $[A, B]$, opposite to the part containing the vertices belonging to $K'' \cup S'$.

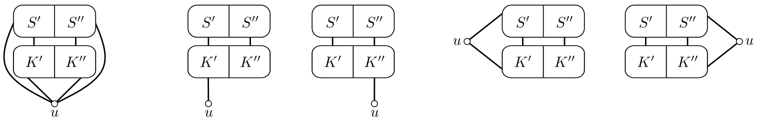
Lemma 1 ([10]). *Let x and y be positive integers, H be an (x, y)-grained gadget and G be a supergraph that respects the structure of H. Also, let $[A, B]$ be a maximum cut of G, t be the number of vertices in $V(G) \setminus V(H)$ adjacent to some vertex of H, ℓ be the number of vertices of G adjacent to some vertex in S', and r be the number of vertices of G adjacent to some vertex in S''. If ℓ and r are odd, $y > 2t$ and $x > t + 2y$, then each of the following holds:*

1. *$S' \subseteq A$ and $K' \subseteq B$, or vice versa;*
2. *$S'' \subseteq A$ and $K'' \subseteq B$, or vice versa;*
3. *$K' \subseteq A$ and $K'' \subseteq B$, or vice versa.*

In the remainder of the text, when a grained gadget H is not clear in the context, we write $S'(H)$, $S''(H)$, $K'(H)$ and $K''(H)$ to denote the stable sets S' and S'' and the cliques K' and K'' of H, respectively.

3. Adhikary et al.'s reduction

In this section, we present the construction given by Adhikary et al. [2] of an interval graph that proves NP-completeness of MaxCut in this class. As we see in Section 4, the general idea behind their construction can also be used to obtain a permutation graph instead. Nevertheless, the question of whether their construction is also permutation might arise. We prove here that this is not the case.



(a) Covering intersection

(b) Weak intersection

(c) Strong intersection

Figure 3: Vertex $u \in V(G) \setminus V(H)$ (a) covering H, (b) weakly intersecting H, and (c) strongly intersecting H. The set $K' \cup K''$ is a clique and the set $S' \cup S''$ is a stable set. A line between sets, or between u and some set, means that all the edges occur.

Given a cubic graph G, let $\pi_V = (v_1, v_2, \ldots, v_n)$ and $\pi_E = (e_1, e_2, \ldots, e_m)$ be arbitrary orderings of $V(G)$ and $E(G)$, respectively. Define the values: $q = 200n^3 + 1$, $p = 2q + 7n$, $q' = 10n^2 + 1$, and $p' = 2q' + 7n$. An interval graph G' is defined through the construction of one of its interval models $\mathcal{M}$, described as follows (observe Figure 4 to follow the construction):

1. Add to $\mathcal{M}$ a (p, q)-grained gadget $\mathcal{H}_i$ for each vertex $v_i \in V(G)$. These gadgets should be pairwise disjoint, with $\mathcal{H}_i$ appearing completely to the left of $\mathcal{H}_{i+1}$ for every $i \in [n-1]$;
2. Add to $\mathcal{M}$ a (p', q')-grained gadget $\mathcal{E}_j$ for each edge $e_j \in E(G)$. Likewise, these gadgets should be pairwise disjoint, with $\mathcal{E}_j$ appearing completely to the left of $\mathcal{E}_{j+1}$ for every $j \in [m-1]$. Additionally, $\mathcal{E}_1$ appears completely to the right of $\mathcal{H}_n$, without intersecting it;
3. Finally, for each edge $e_j = v_i v_{i'} \in E(G)$, with $i < i'$, add four intervals $L^1_{i,j}, L^2_{i,j}, L^1_{i',j}, L^2_{i',j}$, called *link* intervals, such that:
   - $L^1_{i,j}$ and $L^2_{i,j}$ (resp. $L^1_{i',j}$ and $L^2_{i',j}$) weakly intersect $\mathcal{H}_i$ (resp. $\mathcal{H}_{i'}$) to the right of $\mathcal{H}_i$ (resp. $\mathcal{H}_{i'}$);
   - $L^1_{i,j}$ and $L^2_{i,j}$ (resp. $L^1_{i',j}$ and $L^2_{i',j}$) weakly intersect (resp. strongly intersect) $\mathcal{E}_j$ to the left of $\mathcal{E}_j$.

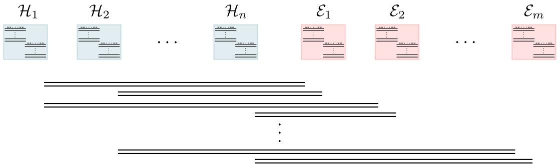
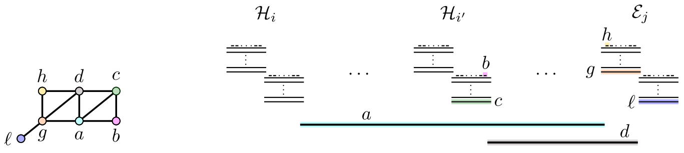


Figure 4: Adhikary et al.'s interval model $\mathcal{M}$, with $e_1 = v_1 v_2$, $e_2 = v_1 v_n$, and $e_m = v_2 v_n$.

As claimed, we show that the constructed graph G' is not a permutation graph. This is because G' contains the graph $\overline{X}_{34}$ depicted in Figure 5a as an induced subgraph, and such a graph is a forbidden subgraph for comparability graphs cf. [11, 12], in turn a known superclass of permutation graphs. To see that this claim holds, observe Figure 5b. Given an edge $e_j = v_i v_{i'} \in E(G)$, with $i < i'$, it shows the intervals in the grained gadgets of v_i, $v_{i'}$ and e_j, as well as some link intervals related to e_j. The adjacencies can be easily checked to be as in the graph of Figure 5a.



(a)

(b)

Figure 5: (a) Forbidden induced subgraph $\overline{X}_{34}$ for comparability graphs cf.[12]. (b) $\overline{X}_{34}$ as an induced subgraph in Adhikary et al.'s construction.

In the next section, we show that a modification of Adhikary et al.'s construction gives us the desired permutation graph.

4. Our reduction

Let G be a cubic graph, and consider $\pi_V = (v_1, v_2, \ldots, v_n)$ and $\pi_E = (e_1, e_2, \ldots, e_m)$ arbitrary orderings of $V(G)$ and $E(G)$, respectively. The values of p, q, p', q' are not the same as in Section 3 and are presented later. Again, for each vertex v_i, create a (p, q)-grained gadget, $\mathcal{H}_i$, and for each edge e_j, create a (p', q')-grained gadget $\mathcal{E}_j$. For simplicity, denote the sets $S'(\mathcal{H}_i)$, $S''(\mathcal{H}_i)$, $K'(\mathcal{H}_i)$ and $K''(\mathcal{H}_i)$ by S'_i, S''_i, K'_i, K''_i, respectively. Similarly, denote the sets $S'(\mathcal{E}_j)$, $S''(\mathcal{E}_j)$, $K'(\mathcal{E}_j)$ and $K''(\mathcal{E}_j)$ by $S_j^{'e}, S_j^{''e}, K_j^{'e}, K_j^{''e}$, respectively.

Recall that for each $i \in [n]$, the permutation model of $\mathcal{H}_i$ consists of the pair of permutations $\{\pi_i^1, \pi_i^2\}$ where $\pi_i^1 = K'_i S'_i S''_i K''_i$ and $\pi_i^2 = S'_i \overleftarrow{K''_i} \overleftarrow{K'_i} S''_i$. Analogously, for each $j \in [m]$, the permutation model of $\mathcal{E}_j$ consists of the pair of permutations $\{\gamma_j^1, \gamma_j^2\}$ where $\gamma_j^1 = K_j^{'e} S_j^{'e} S_j^{''e} K_j^{''e}$ and $\gamma_j^2 = S_j^{'e} \overleftarrow{K_j^{''e}} \overleftarrow{K_j^{'e}} S_j^{''e}$. Now, for each edge $e_j = v_i v_{i'}$, with $i < i'$, add four new vertices $L^1_{i,j}, L^2_{i,j}, L^1_{i',j}, L^2_{i',j}$, called *link* vertices. In what follows, we modify some of the grained gadget permutations in order to make $L^1_{i,j}, L^2_{i,j}$ (resp. $L^1_{i',j}, L^2_{i',j}$) weakly intersect $\mathcal{H}_i$ (resp. $\mathcal{H}_{i'}$) and strongly intersect (resp. weakly intersect) $\mathcal{E}_j$.

If v_i is incident to edges j_1, j_2, j_3, with $j_1 < j_2 < j_3$, then modify one of the permutations defining $\mathcal{H}_i$ to include the link vertices related to v_i as follows:

$$\pi_i^1 = K'_i S'_i S''_i C_i K''_i,$$

where C_i denotes the permutation $L^1_{i,j_1} L^2_{i,j_1} L^1_{i,j_2} L^2_{i,j_2} L^1_{i,j_3} L^2_{i,j_3}$.

Similarly, for each edge $e_j = v_i v_{i'}$, $i < i'$, we modify one of the permutations defining $\mathcal{E}_j$ to include the link vertices related to e_j as follows:

$$\gamma_j^1 = K_j^{'e} L^2_{i',j} L^1_{i',j} S_j^{'e} L^2_{i,j} L^1_{i,j} S_j^{''e} K_j^{''e}.$$

We do not modify π_i^2 and γ_j^2, and keep denoting by π_i^2 the permutation $S'_i \overleftarrow{K''}_i \overleftarrow{K'}_i S''_i$, and by γ_j^2 the permutation $S_j^{'e} \overleftarrow{K''^{e}}_j \overleftarrow{K'^{e}}_j S_j^{''e}$. Finally, let G' be the permutation graph related to $\{\Pi, \Pi'\}$, where:

$$\Pi = \pi_1^1 \dots \pi_n^1 \gamma_1^2, \dots, \gamma_m^2, \text{ and}$$

$$\Pi' = \pi_1^2 \dots \pi_n^2 \gamma_1^1, \dots, \gamma_m^1.$$

Figure 6 illustrates our permutation model $\{\Pi, \Pi'\}$, focusing on the vertex grained gadgets $\mathcal{H}_i$ and $\mathcal{H}_{i'}$, the edge grained gadget $\mathcal{E}_j$, and the link vertices $L^1_{i,j}, L^2_{i,j}$ and $L^1_{i',j}, L^2_{i',j}$ related to an edge $e_j = v_i v_{i'}$, with $i < i'$.

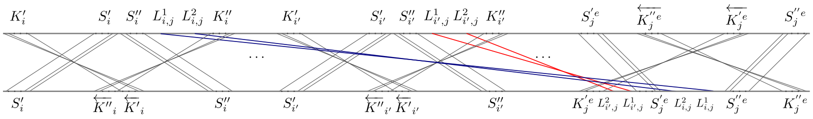


Figure 6: Vertex and edge grained gadgets, and link vertices related to an edge $e_j = v_i v_{i'}$, with $i < i'$, in our permutation model $\{\Pi, \Pi'\}$.

We remark that the main difference of our permutation graph from the Adhikary et al.'s interval graph is the fact that, in Adhikary et al.'s interval graph, the link vertices form a clique, whereas, as we show in Section 5, some link vertices are not adjacent in our permutation graph. Additionally, for an edge $e_j = v_i v_{i'} \in E(G)$, with $i < i'$, the link vertices $L^1_{i,j}, L^2_{i,j}$ (resp. $L^1_{i',j}, L^2_{i',j}$) weakly intersect (resp. strongly intersect) $\mathcal{E}_j$ in Adhikary et al.'s interval graph, whereas in our permutation graph the link vertices $L^1_{i,j}, L^2_{i,j}$ (resp. $L^1_{i',j}, L^2_{i',j}$) strongly intersect (resp. weakly intersect) $\mathcal{E}_j$.

Before our proof, we make some observations about the constructed graph in order to improve the proof's readability. Note that, for each link vertex L and grained gadget H, either the relative order between L and $V(H)$ in Π is the reverse of their relative order in Π', in which case L is complete to $V(H)$, or the relative order is the same in both Π and Π', in which case L is anti-complete to $V(H)$, or L is related to H according to one of the ways described below.

- $L \in \{L^1_{i,j}, L^2_{i,j}\}$ and $H = \mathcal{H}_i$: in this case only the relative orders between L and K''_i are reversed in Π and Π', i.e., L is complete to K''_i and anti-complete to $V(\mathcal{H}_i) \setminus K''_i$;
- $L \in \{L^1_{i,j}, L^2_{i,j}\}$ and $H = \mathcal{E}_j$, with $e_j = v_i v_{i'}$, $i < i'$: in this case the relative orders between L and $K'^e_j \cup S'^e_j$ are reversed in Π and Π', i.e., L is complete to $K'^e_j \cup S'^e_j$ and anti-complete to $V(\mathcal{E}_j) \setminus (K'^e_j \cup S'^e_j)$; or
- $L \in \{L^1_{i',j}, L^2_{i',j}\}$ and $H = \mathcal{E}_j$, with $e_j = v_i v_{i'}$, $i < i'$: in this case only the relative orders between L and K'^e_j are reversed in Π and Π', i.e., L is complete to K'^e_j and anti-complete to $V(\mathcal{E}_j) \setminus K'^e_j$.

Proof of Theorem 1. Consider the reduction graph G' and its permutation model $\{\Pi, \Pi'\}$ as previously defined. For each $e_j = v_i v_{i'} \in E(G)$, let

$$L(e_j) = \{L^1_{i,j}, L^2_{i,j}, L^1_{i',j}, L^2_{i',j}\};$$

and for each $v_i \in V(G)$, let

$$L(v_i) = \{L^1_{i,j}, L^2_{i,j} \mid e_j \text{ is incident to } v_i\}.$$

Also, denote the set of link vertices by $\mathcal{L}$, i.e. $\mathcal{L} = \bigcup_{j=1}^{m} L(e_j)$.

We postpone the assignment of the actual values for p, q, p', q' and, in addition to the conditions necessary for the application of Lemma 1, we also ask that $q > 6n + p'$ and $p' > 2q' > 9n^2$.

In what follows, we prove that there exists a bijective relation f between the maximum cuts of the input graph G and the maximum cuts of our permutation graph G'. Then, we prove that, for each maximum cut $[X, Y]$ of G,

$$|E_G(X, Y)| \geq k \text{ if and only if } |E_{G'}(A, B)| \geq \phi(n, m, k),$$

where $[A, B] = f(X, Y)$ and ϕ is a well-defined function. Theorem 1 immediately follows.

Let $[A, B]$ be a maximum cut of G'. In order to define f, we first prove some properties relating the partitioning of vertex and edge grained gadgets of G' in $[A, B]$ with the partitioning of the link vertices of G' in $[A, B]$. More specifically, we prove that the two following properties hold:

1. For each vertex $v_i \in V(G)$, if $K''_i \subseteq A$, then $\{L^1_{i,j}, L^2_{i,j}\} \subseteq B$ for each edge $e_j \in E(G)$, with $e_j = v_i v_{i'}$ and $i < i'$;
2. For each edge $e_j \in E(G)$, with $e_j = v_i v_{i'}$ and $i < i'$, if $\{L^1_{i,j}, L^2_{i,j}\} \subseteq B$, then $S^{\prime e}_j \subseteq A$.

Proof of Property 1. Let $v_i \in V(G)$ and suppose that $K''_i \subseteq A$. For the sake of contradiction, suppose that there exists a link vertex $L \in L(v_i) \cap A$. Then, let $[A', B']$ be the cut obtained from $[A, B]$ by setting $A' = A \setminus \{L\}$ and $B' = B \cup \{L\}$. Observe that there is a loss of at most $|\mathcal{L}| + \max\{p', q'\} = |\mathcal{L}| + p'$ edges between L and $\mathcal{L}$, and between L and the vertices of the edge grained gadget related to L, say $\mathcal{E}_j$, since $K^{\prime e}_j$ and $S^{\prime e}_j$ are always in opposite parts of the cut. On the other hand we gain all the edges between L and the vertices in K''_i. Therefore, we get an increase of the cut-set of at least q edges, and a decrease of less than $|\mathcal{L}| + \max\{p', q'\} = 6n + p'$ edges. It follows from the hypothesis $q > 6n + p'$ that $|E_{G'}(A', B')|$ is bigger than $|E_{G'}(A, B)|$, contradicting the maximality of $[A, B]$.

Proof of Property 2. Consider an edge $e_j \in E(G)$, with $e_j = v_i v_{i'}$ and $i < i'$, and suppose that $\{L^1_{i,j}, L^2_{i,j}\} \subseteq B$. Observe that, because the relative orders among the edge and vertex grained gadgets themselves are the same in Π and Π', there are no edges between $\mathcal{E}_j$ and any other grained gadgets of G', i.e., the only vertices outside of $\mathcal{E}_j$ that can be adjacent to the vertices of $\mathcal{E}_j$ are those in $\mathcal{L}$. Moreover, Lemma 1 tells us that the vertices belonging to $K^{\prime e}_j \cup S^{\prime\prime e}_j$ are placed in a same part of $[A, B]$, opposite to the part containing the vertices belonging to $K^{\prime\prime e}_j \cup S^{\prime e}_j$. More formally, either $K^{\prime e}_j \cup S^{\prime\prime e}_j \subseteq B$ and $K^{\prime\prime e}_j \cup S^{\prime e}_j \subseteq A$, or $K^{\prime e}_j \cup S^{\prime\prime e}_j \subseteq A$ and $K^{\prime\prime e}_j \cup S^{\prime e}_j \subseteq B$. As a result, switching the vertices of $\mathcal{E}_j$ of part of the cut does not change, and therefore cannot decrease, the number of edges between the vertices of $\mathcal{E}_j$ and the vertices belonging to $\mathcal{L} \setminus L(e_j)$ in the cut-set. Consequently, if $S_j^{'e} \subseteq A$, then we obtain that there are at least $2p'$ edges in the cut-set that are incident to vertices of $\mathcal{E}_j$; these are the edges between $L^1_{i,j}, L^2_{i,j}$ and the vertices belonging to $S_j^{'e}$. On the other hand, if $S_j^{'e} \subseteq B$, then we obtain that there are at most $4q'$ edges in the cut-set that are incident to vertices of $\mathcal{E}_j$; these are the edges between the vertices belonging to $L(e_j)$ and the vertices belonging to $K_j^{'e}$. Therefore, since $p' > 2q'$, we get that $S_j^{'e} \subseteq A$ as we wanted to prove.

We are now ready to prove the existence of the bijective relation f. For each maximum cut $[X, Y]$ of G, let $f(X, Y)$ be the cut $[A, B]$ of G' defined as follows:

- For each vertex $v_i \in V(G)$, if $v_i \in X$, then add $K'_i \cup S''_i \cup L(v_i)$ to A and $K''_i \cup S'_i$ to B; do the opposite otherwise.

- For each $e_j \in E(G)$, with $e_j = v_i v_{i'}$ and $i < i'$, if $L^1_{i,j} \in A$, then add $K_j^{'e} \cup S_j^{''e}$ to A and $K_j^{''e} \cup S_j^{'e}$ to B; and do the opposite otherwise.

Based on Properties 1 and 2, one can readily verify that f is well-defined and is a bijective relation, as desired.

Now, we count the number of edges in $E_{G'}(A, B)$ as a function of n, m, p, q, p', q' and of the size of the cut-set $E_G(X, Y)$. First, consider $v_i \in V(G)$. By construction, we know that there are $2pq + q^2$ edges in the cut-set between the vertices of $\mathcal{H}_i$. Additionally, there are exactly 6 link vertices weakly intersecting $\mathcal{H}_i$, while all other link vertices are either complete or anti-complete to $V(\mathcal{H}_i)$. Observe also that the number of link vertices complete to $V(\mathcal{H}_i)$ is exactly equal to $6(i-1)$; these are the link vertices related to $\{v_1, \ldots, v_{i-1}\}$. This gives us a total of $6[q + (i-1)(p+q)]$ edges between the vertices of $\mathcal{H}_i$ and the vertices belonging to $\mathcal{L}$ in the cut-set. Summing up these values for every $v_i \in V(G)$, we get a total of

$$\alpha_1 = n[2pq + q^2 + 6q] + 6\sum_{i=1}^{n}((i-1)(p+q)) = n[2pq + q^2 + 6q + 3(p+q)(n-1)]$$

edges in the cut-set $E_{G'}(A, B)$ incident to vertex grained gadgets. Now, let $e_j \in E(G)$, with $e_j = v_i v_{i'}$ and $i < i'$. By construction, we know that there are $2p'q' + (q')^2$ edges of the cut-set between vertices of $\mathcal{E}_j$, and $2p'$ edges of the cut-set between $L^1_{i,j}, L^2_{i,j}$ and the vertices of $\mathcal{E}_j$. Additionally, note that there are exactly $4(m-j)$ link vertices that cover and, therefore, are complete to $\mathcal{E}_j$; these are the link vertices related to $\{e_{j+1}, \ldots, e_m\}$. This gives us a total of $4(m-j)(p'+q')$ edges between the vertices of $\mathcal{E}_j$ and the vertices belonging to $\mathcal{L} \setminus L(e_j)$. Finally, suppose without loss of generality that $L^1_{i,j} \in A$ (the count is analogous if it is in B). If $v_{i'} \in X$, then we know that $\{L^1_{i',j}, L^2_{i',j}\} \subseteq A$ and hence there are no edges in the cut-set between vertices $L^1_{i',j}, L^2_{i',j}$ and the vertices of $\mathcal{E}_j$. Otherwise, observe that it follows that $e_j \in E_G(X, Y)$ and $\{L^1_{i',j}, L^2_{i',j}\} \subseteq B$, and hence we get additional $2q'$ edges in the cut-set; these additional edges are between the link vertices $L^1_{i',j}, L^2_{i',j}$ and the vertices belonging to $K_j^{\prime e}$. Summing up these values for every $e_j \in E(G)$, we get a total of $\alpha_2 + 2q'|E_G(X,Y)|$ edges in the cut-set $E_{G'}(A,B)$ incident to edge grained gadgets, where

$$\begin{aligned} \alpha_2 &= m[2p'q' + (q')^2 + 2p'] + \textstyle\sum_{j=1}^{m} (4(m-j)(p'+q')) \\ &= m[2p'q' + (q')^2 + 2p' + 2(p'+q')(m-1)]. \end{aligned}$$

Finally, observe that there are at most $|A \cap \mathcal{L}| \cdot |B \cap \mathcal{L}|$ edges of the cut-set between link vertices. Note also that $|A \cap \mathcal{L}| = 6|Y|$ since each vertex in Y is related to 6 link vertices, which are all placed in A. Similarly, we have $|B \cap \mathcal{L}| = 6|X|$. This gives us at most $36|X| \cdot |Y| \leq 9n^2$ edges in the cut-set between link vertices. Putting everything together, we get:

$$\alpha_1 + \alpha_2 + 2q'|E_G(X,Y)| \leq |E_{G'}(A,B)| \leq \alpha_1 + \alpha_2 + 2q'|E_G(X,Y)| + 9n^2.$$

By setting $\phi(n,m,k)$ to $\alpha_1 + \alpha_2 + 2q'k$, and knowing that p,q,p',q' will be chosen as functions of n and m, we want to prove, as stated in the beginning, that $|E_G(X,Y)| \geq k$ if and only if $|E_{G'}(A,B)| \geq \phi(n,m,k)$. If $|E_G(X,Y)| \geq k$, then the first inequality gives us that $|E_{G'}(X,Y)| \geq \alpha_1 + \alpha_2 + 2q'k = \phi(n,m,k)$. On the other hand, if $|E_{G'}(A,B)| \geq \phi(n,m,k) = \alpha_1 + \alpha_2 + 2q'k$, then the second inequality gives us that $|E_G(X,Y)| \geq k - 9n^2/2q'$. Because we assume that $2q' > 9n^2$, it follows that $k - 9n^2/2q' > k-1$ and hence $|E_G(X,Y)| \geq k$.

It only remains to set the values of p,q,p',q'. Observe that:

- For every grained gadget H, the total number of vertices in $V(G') \setminus H$ adjacent to H is at most $6n$ (these are exactly the link vertices).

- For every vertex grained gadget H, the total number of vertices adjacent to some vertex $u \in S'(H)$ is exactly $q+2h$, for some positive integer h (this is because the number of link vertices adjacent to the vertices in $S'(H)$ is always even). The same holds for the number of vertices adjacent to the vertices in $S''(H)$. We then get that the parity of ℓ and r in the conditions of Lemma 1 applied to H depends only on the parity of q.

- Similarly, if H is an edge grained gadget, then the parity of the total number of vertices adjacent to some $u \in S'(H) \cup S''(H)$ is equal to the parity of q'.

Therefore, the necessary conditions of Lemma 1 translate to: $q > 12n$ and $q' > 12n$; $p > 2q + 6n$ and $p' > 2q' + 6n$; and q and q' are odd. Additionally, we need to ensure: $q > 6n + p'$ and $p' > 2q' > 9n^2$. Hence, consider:

- $q' = 5n^2 + 1$;

- $p' = 11n^2 + 6n$;

- $q = 12n^2 + 12n + 1$;

- $p = 25n^2 + 30n$.

Since $n \geq 4$, one can verify that the values described above satisfy all the required conditions. This, therefore, concludes the proof of Theorem 1. □

5. MaxCut on permutation interval graphs is an open problem

In this paper, we have presented a proof of NP-completeness for the MaxCut problem when constrained to permutation graphs. Surprisingly enough, we found that the main gadget in the reduction recently presented by Adhikary et al. [2] for interval graphs is also a permutation graph. Additionally, in Section 3, we have seen that being permutation is not a property that holds for the full construction of Adhikary et al. [2]. On the other hand, since the grained gadgets play an important role in our reduction too, one could wonder whether our construction instead is in the intersection between interval and permutation graphs. The answer to that is no as we argue next.

Let G be a cubic graph, and consider arbitrary orderings of $V(G)$ and $E(G)$, $(v_1, \ldots, v_n)$ and $(e_1, \ldots, e_m)$, respectively. Let j_1, j_2, j_3 be the indices of the edges incident to v_1, with $j_1 < j_2 < j_3$. Also, let v_i be the other endpoint of e_{j_2}. We present a C_4 in G', the graph constructed in Section 4; it thus follows that G' is not chordal, and hence also not interval [5]. Observe Figure 7 to follow our argument. Let a be equal to L^1_{1,j_1}, b be any vertex in K''_i, c be equal to L^1_{i,j_2}, and d be any vertex in $K'^e_{j_1}$. Since $j_1 < j_2$ and $1 < i$, we know that the relative order between a and c in Π is the same as in Π'; hence $ac \notin E(G')$. Also, the relative order in Π between a and any vertex of $\mathcal{H}_i$ is reversed in Π', the same holds between c and any vertex belonging to K''_i; hence $\{ab, bc\} \subseteq E(G')$. Similarly, the relative order between a and any vertex belonging to $K'^e_{j_1}$ in Π is reversed in Π', and the same holds between c and any vertex of $\mathcal{E}_{j_1}$; hence $\{ad, cd\} \subseteq E(G')$. Finally, since $j_1 < j_2$, the relative order between b and d in Π is the same as in Π', and therefore $bd \notin E(G')$, thus finishing our argument.

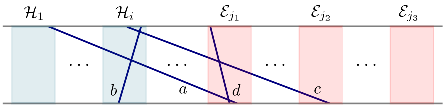


Figure 7: Existence of a $C_4 = (a, b, c, d)$ as an induced subgraph in our permutation graph.

The previous paragraph tells us that for any chosen orderings of $V(G)$ and $E(G)$, the graph constructed in Section 4 contains a C_4. Since it is known that the class of C_4-free co-comparability graphs is precisely the class of interval graphs [13], and that the class of permutation graphs is equal to the class of comparability co-comparability graphs [14], we get that interval permutation graphs are exactly the class of C_4-free permutation graphs.

A good question is whether there is a construction that produces a permutation graph that is also C_4-free (and hence interval). Up to our knowledge, the largest class in the intersection of permutation and interval graphs for which the complexity is known is the class of the trivially perfect graphs, on which MAX-CUT is polynomial-time solvable thanks to the algorithm given for cographs [15], a subclass of permutation graphs that is a superclass of trivially perfect graphs.

References

[1] M. R. Garey, D. S. Johnson, L. J. Stockmeyer, Some simplified NP-complete graph problems, Theor. Comput. Sci. 1 (3) (1976) 237–267. doi:10.1016/0304-3975(76)90059-1.

[2] R. Adhikary, K. Bose, S. Mukherjee, B. Roy, Complexity of maximum cut on interval graphs, in: 37th International Symposium on Computational Geometry, SoCG 2021, Vol. 189 of LIPIcs, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021, pp. 7:1–7:11. doi:10.4230/LIPIcs.SoCG.2021.7.

[3] D. S. Johnson, The NP-completeness column: An ongoing guide, J. Algorithms 6 (3) (1985) 434–451. doi:10.1016/0196-6774(85)90012-4.

[4] P. C. Fishburn, Interval graphs and interval orders, Discret. Math. 55 (2) (1985) 135–149. doi:10.1016/0012-365X(85)90042-1.

[5] M. C. Golumbic, Algorithmic Graph Theory and Perfect Graphs (Annals of Discrete Mathematics, Vol 57), North-Holland Publishing Co., NLD, 2004.

[6] J. P. Spinrad, Efficient Graph Representations, Fields Institute monographs, American Mathematical Society, Providence, RI, 2003.

[7] C. M. de Figueiredo, A. A. de Melo, D. Sasaki, A. Silva, Revising Johnson's table for the 21st century, Discret. Appl. Math. (2021). doi:10.1016/j.dam.2021.05.021.

[8] J. A. Bondy, U. S. R. Murty, Graph Theory, Graduate Texts in Mathematics, Springer, New York, 2008. doi:10.1007/978-1-84628-970-5.

[9] C. M. H. de Figueiredo, A. A. de Melo, F. de S. Oliveira, A. Silva, Maximum cut on interval graphs of interval count four is NP-complete, in: 46th International Symposium on Mathematical Foundations of Computer Science, MFCS 2021, Vol. 202 of LIPIcs, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021, pp. 38:1–38:15. doi:10.4230/LIPIcs.MFCS.2021.38.

[10] C. M. H. de Figueiredo, A. A. de Melo, F. de S. Oliveira, A. Silva, Maximum cut on interval graphs of interval count four is NP-complete (2020). arXiv:2012.09804.

[11] T. Gallai, Transitiv orientierbare graphen, Acta Mathematica Hungarica 18 (1-2) (1967) 25–66.

[12] H. N. de Ridder et al., Graphclass: comparability graphs. information system on graph classes and their inclusions (isgci), https://www.graphclasses.org/classes/gc_72.html, accessed: 2022-02-17.

[13] P. C. Gilmore, A. J. Hoffman, A characterization of comparability graphs and of interval graphs, Canadian Journal of Mathematics 16 (1964) 539–548.

[14] A. Pnueli, A. Lempel, S. Even, Transitive orientation of graphs and identification of permutation graphs, Canadian Journal of Mathematics 23 (1971) 160–175.

[15] H. L. Bodlaender, K. Jansen, On the complexity of the maximum cut problem, in: Annual Symposium on Theoretical Aspects of Computer Science, Springer, 1994, pp. 769–780.

Appendix G

Manuscript: Computing the Zig-Zag Number of Directed Graphs

This appendix contains the manuscript:
Mitre C. Dourado, Celina M. H. de Figueiredo, Alexsander A. de Melo, Mateus de Oliveira Oliveira, Uéverton S. Souza. *Computing the Zig-Zag Number of Directed Graphs*. Published in Discrete Applied Mathematics 312 (2022) [50].

Discrete Applied Mathematics 312 (2022) 86–105

Contents lists available at ScienceDirect

Discrete Applied Mathematics

journal homepage: www.elsevier.com/locate/dam

Computing the zig-zag number of directed graphs

Mitre C. Dourado [a], Celina M.H. de Figueiredo [a], Alexsander A. de Melo [a,*], Mateus de Oliveira Oliveira [c], Uéverton S. Souza [b]

[a] Federal University of Rio de Janeiro, Rio de Janeiro, Brazil
[b] Fluminense Federal University, Niterói, Brazil
[c] University of Bergen, Bergen, Norway

ARTICLE INFO

Article history:
Received 29 June 2020
Received in revised form 12 June 2021
Accepted 15 September 2021
Available online 2 October 2021

Keywords:
Zig-zag number
Directed graphs
Directed width measure
NP-completeness
Computational complexity

ABSTRACT

The notion of zig-zag number was introduced as an attempt to provide a unified algorithmic framework for directed graphs. Nevertheless, little was known about the complexity of computing this directed graph invariant. We prove that deciding whether a directed graph has zig-zag number at most k is in NP for each fixed $k \geq 0$. Although for most of the natural decision problems this is an almost trivial result, settling k-ZIG-ZAG NUMBER in NP is surprisingly difficult. In addition, we prove that 2-ZIG-ZAG NUMBER is already an NP-hard problem.

1. Introduction

Structural graph parameters, such as treewidth, cutwidth and cliquewidth, have been crucial in the development of parameterized complexity theory. Indeed, many problems that are hard on general graphs become tractable when parameterized by such parameters [5,6]. However, one of the limitations of these parameters is the fact that they do not take the direction of edges into account. For instance, directed acyclic graphs (DAGs) in general have unbounded width with respect to any of the parameters mentioned above. Nevertheless, certain problems can be solved efficiently on DAGs by using straightforward algorithms. For instance, DIRECTED HAMILTONIAN PATH can be solved in linear time on DAGs with a depth-first search algorithm.

Building on this observation, Johnson, Robertson, Seymour and Thomas [12] initiated a quest for the development of width measures that explicitly take the direction of edges into consideration. In particular, they defined in [12] the notion of directed treewidth and showed that some linkage problems that are NP-hard on general directed graphs can be solved in polynomial time on directed graphs of constant directed treewidth. Additionally, a directed analog of the notion of pathwidth was also defined by Reed, Thomas, and Seymour around the same time (see for instance *cf.* [1]).

The introduction of directed treewidth and directed pathwidth motivated the development of many other width measures for directed graphs that focus on distinct algorithmic or structural properties [2,3,7,9,10,15,16]. A general algorithmic framework for directed width measures was developed in [13] with the introduction of the notion of *zig-zag* number of a directed graph, and subsequently generalized in [14] with the definition of the notion of *tree-zig-zag* number of a directed graph.

[*] Corresponding author.
E-mail addresses: mitre@dcc.ufrj.br (M.C. Dourado), celina@cos.ufrj.br (C.M.H. de Figueiredo), aamelo@cos.ufrj.br (A.A. de Melo), mateus.oliveira@uib.no (M. de Oliveira Oliveira), ueverton@ic.uff.br (U.S. Souza).

https://doi.org/10.1016/j.dam.2021.09.013
0166-218X/© 2021 Elsevier B.V. All rights reserved.

It was shown in [13] that if $\mathcal{G}$ is a class of directed graphs expressible by a monadic-second order logic formula φ and there is a positive integer p such that each directed graph in $\mathcal{G}$ can be cast as a union of p directed paths, then, given a decomposition of a directed graph G of zig-zag number at most k, one can count in time $f(\varphi, p, k) \cdot |G|^{\mathcal{O}(p\cdot k)}$ the number of subgraphs of G isomorphic to some member of $\mathcal{G}$, for some computable function f. Since directed path decompositions of width d can be efficiently converted into decompositions of zig-zag number $\mathcal{O}(d)$, the counting problem described above can also be solved in time $f(\varphi, p, d) \cdot |G|^{\mathcal{O}(p\cdot d)}$ on directed graphs of directed pathwidth at most d. These results were subsequently generalized in [14] to their respective counterparts for directed graphs of tree-zig-zag number at most k and of directed treewidth at most d. The results in [13] and in [14] were the first algorithmic metatheorems relating the monadic-second order logic of graphs to directed pathwidth and directed treewidth, respectively.

In a seminal paper analyzing the algorithmic potential of directed width measures, Ganian et al. [8] defined a width measure to be *algorithmically useful* if it satisfies the following properties: (1) every graph problem expressible in MSO_1 logic admits an XP-time algorithm when parameterized by the measure and (2) for each constant k, the class of graphs of width at most k is closed under taking directed topological minors. Interestingly, it was shown in [8] that, under standard complexity theoretic assumptions, any width measure satisfying properties (1) and (2) behaves essentially in the same way as the usual notion of undirected treewidth (see Theorems 6.6 and 6.7 of [8] for precise statements). We note that any directed width measure that is constant on DAGs, including zig-zag number, tree-zig-zag number and most of the width measures defined so far, fail to satisfy property (1) since 3-COLORING is MSO_1 definable and NP-complete on DAGs. Despite of this fact, zig-zag number and tree-zig-zag number have been proved to be algorithmically relevant, by establishing through the metatheorems presented in [13,14] a unified algorithmic framework to solve problems on directed graphs of low directed pathwidth and of low directed treewidth, respectively. Another interesting aspect of zig-zag number and tree-zig-zag number is the fact that they can be regarded as graph invariants with challenging theoretical open problems, from the perspectives of computational complexity and graph theory.

In fact, several complexity questions with respect to computing zig-zag number and tree-zig-zag number of a directed graph remain open. In particular, the computational complexity of the problem of determining whether a directed graph has zig-zag number at most k, even for constant k, has remained open since the introduction of this notion in [13].

We show in Section 3 that determining whether a directed graph G has zig-zag number at most k can be solved *non-deterministically* in time $|G|^{\mathcal{O}(k)}$, implying that this problem lies in NP for each fixed k. While the respective statement is almost trivial with respect to other directed width measures, such as directed pathwidth, which is known to be in P [18], our proof settling k-ZIG-ZAG NUMBER in NP turned out to be an interesting quest. This is due to the fact that the definition of zig-zag number, which we formally present in Section 2, involves the alternation of an existential and a universal quantifiers. Thus, a naive application of the definition would only lead to a Σ_2^{P}-upper bound for the problem. To circumvent this, and settle the problem in NP, our proof may be regarded as a way of redefining the property of a directed graph having zig-zag number at most k in a purely existential fashion.

On the other hand, through a polynomial-time reduction from POSITIVE NOT ALL EQUAL 3SAT, we prove in Section 4 that deciding whether a directed graph has zig-zag number at most 2 is an NP-hard problem. It is worth noting that this intractability result does not affect the applicability of the algorithmic metatheorem presented in [13], since for each $k \in \mathbb{N}$, directed path decompositions of width k can be converted efficiently into linear orderings of zig-zag number $\mathcal{O}(k)$ [13], and directed path-decompositions of width k, whenever exist, can be constructed in time $n^{\mathcal{O}(k)}$ [18].

Besides these proposed results, we analyze in Section 5 how zig-zag number and directed treewidth are related to each other. We prove that there are directed graphs of constant directed treewidth but unbounded zig-zag number. As a consequence, with the results of [14], we obtain that the family of directed graphs of constant tree-zig-zag number is strictly richer than the family of directed graphs of constant zig-zag number.

2. Preliminaries

A *directed simple graph* (or, simply *directed graph*) is a pair $G = (V, E)$ comprising a non-empty *vertex set* V and an *edge set* $E \subseteq \{(u, v) : (u, v) \in V \times V, u \neq v\}$. In what follows, we may write n or $|G|$ to denote the number of vertices of G, and we may write $V(G)$ and $E(G)$ to refer to the vertex set and to the edge set of G, respectively.

For each positive integer n, we let $[n] \doteq \{1, 2, \dots, n\}$. Let G be a directed graph on n vertices. For each bijection $\pi : V(G) \to [n]$, we let $<_\pi \subseteq V(G) \times V(G)$ be the linear order associated with π such that, for each $u, v \in V(G)$, $u <_\pi v$ if and only if $\pi(u) < \pi(v)$. Analogously, we let $>_\pi \subseteq V(G) \times V(G)$ be the linear order such that, for each $u, v \in V(G)$, $u >_\pi v$ if and only if $\pi(u) > \pi(v)$. Let $X, Y \subseteq V(G)$ be two non-empty sets. We write $X <_\pi Y$ to denote that $u <_\pi v$ for each $u \in X$ and each $v \in Y$. We define $X >_\pi Y$ similarly. Moreover, for any non-empty set $X \subseteq V(G)$, we write $\min_\pi X$ to denote the unique vertex $u \in X$ such that $\{u\} <_\pi X \setminus \{u\}$. We define $\max_\pi X$ similarly.

The zig-zag number of a directed graph. Let n be a positive integer, G be a directed graph on n vertices and $\pi : V(G) \to [n]$ be a bijection. For simplicity, assume that $V(G) = \{u_1, \dots, u_n\}$ and, for each $u_i, u_j \in V(G)$, $i < j$ if and only if $u_i <_\pi u_j$.

For a proper subset $X \subset V(G)$, we let $E_G(X)$ denote the subset of edges of G with one endpoint in X and another endpoint in $V(G) \setminus X$. An *edge cut* (or simply *cut*) of G is defined as a subset $S \subseteq E(G)$ such that, for some $X \subset V(G)$, $S = E_G(X)$. For each $i \in [n-1]$, we let $S_G(\pi, i) \doteq E_G(\{u_1, \dots, u_i\})$ be the *ith cut of G with respect to* π. Then, the *cutwidth of G with respect to* π is defined as $\text{cw}(G, \pi) \doteq \max_{i\in[n-1]} |S_G(\pi, i)|$, and the *cutwidth of G* is defined as the minimum $\text{cw}(G, \pi)$ over all bijections $\pi : V(G) \to [n]$.

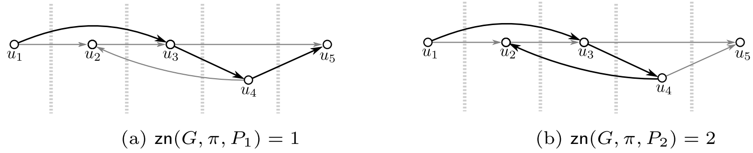

u_1
u_2
u_3
u_4
u_5
(a) $\mathsf{zn}(G, \pi, P_1) = 1$
(b) $\mathsf{zn}(G, \pi, P_2) = 2$

Fig. 1. Directed graph G, bijection $\pi : V(G) \to [|G|]$, where $i < j$ iff $u_i <_\pi u_j$, and directed paths P_1 and P_2 (in bold), such that $\mathsf{zn}(G, \pi, P_1) = 1$ and $\mathsf{zn}(G, \pi, P_2) = 2$, respectively.

Let P be a directed path of G. We let $\mathsf{zn}(G, \pi, P)$ be the maximum number of edges of P that are part of the cut $S_G(\pi, i)$, where the maximum is taken over all $i \in [n-1]$. More formally,

$$\mathsf{zn}(G, \pi, P) \doteq \max_{i \in [n-1]} |E(P) \cap S_G(\pi, i)|.$$

Then, we let $\mathsf{zn}(G, \pi)$ be the maximum $\mathsf{zn}(G, \pi, P)$ over all directed paths P of G. Finally, we define the *zig-zag number of* G, denoted by $\mathsf{zn}(G)$, as the minimum $\mathsf{zn}(G, \pi)$ over all bijections $\pi : V(G) \to [n]$.

Fig. 1 exemplifies a directed graph G and a bijection $\pi : V(G) \to [|G|]$ such that $\mathsf{zn}(G, \pi) = 2$. In fact, one can verify that $\mathsf{zn}(G) = \mathsf{zn}(G, \pi) = 2$.

It is straightforward from the definition of zig-zag number that a directed graph has zig-zag number 0 if and only if it does not contain any edge. Moreover, one can verify that every directed acyclic graph with at least one edge has zig-zag number 1. Indeed, it is known that a directed graph G is directed acyclic if and only if it admits a *topological ordering*, *i.e.* a linear order $<_\pi$ such that $u <_\pi v$ for each $(u, v) \in E(G)$. Thus, one can verify that, if G is a directed acyclic graph and π corresponds to a topological ordering of G, then $\mathsf{zn}(G, \pi) = 1$. In other words, graphs of zig-zag number at least 2 must contain directed cycles. On the other hand, every directed graph G with a directed cycle of length at least 3 necessarily has zig-zag number greater than or equal to 2. Indeed, in this case, for each bijection $\pi : V(G) \to [|G|]$, there always exist three distinct vertices $a, b, c \in V(G)$ such that $\langle a, b, c\rangle$ is a directed path of G, where $a <_\pi b$ and $c <_\pi b$. Intuitively, the zig-zag number of a directed graph measures how much its directed cycles are nested.

Next, we formally define the Zig-zag number problem.

Zig-zag number

Input: A directed graph G and a non-negative integer k.

Question: Is $\mathsf{zn}(G) \le k$? In other words, does there exist a bijection $\pi : V(G) \to [|G|]$ such that, for every directed path P of G,

$$\mathsf{zn}(G, \pi, P) = \max_{i \in [|G|-1]} |E(P) \cap S_G(\pi, i)| \le k?$$

In particular, for each fixed non-negative integer k, we define k-zig-zag number as the decision problem that, given a directed graph G, asks whether $\mathsf{zn}(G) \le k$. More formally:

k-zig-zag number

Input: A directed graph G.

Question: Is $\mathsf{zn}(G) \le k$? In other words, does there exist a bijection $\pi : V(G) \to [|G|]$ such that, for every directed path P of G,

$$\mathsf{zn}(G, \pi, P) = \max_{i \in [|G|-1]} |E(P) \cap S_G(\pi, i)| \le k?$$

3. NP-membership for fixed *k*

In this section, we prove that k-zig-zag number is in NP for each fixed k. We remark that a naive application of the definition of zig-zag number of a directed graph naturally leads to a Σ_2^{P}-upper bound. To circumvent this and settle k-zig-zag number in NP, we show how to replace the inner universal quantifier, which iterates over all directed paths, with an XP-time deterministic computation corresponding to a guessed linear order of the vertices of the input graph and the integer k. More specifically, we prove the following theorem.

Theorem 1. *Let G be a directed graph and k be a non-negative integer. One can non-deterministically decide in time $|G|^{\mathcal{O}(k)}$ whether $zn(G) \le k$.*

In order to prove Theorem 1, we reduce the problem of deciding whether $\mathsf{zn}(G, \pi) \ge k + 1$, for a guessed bijection $\pi : V(G) \to [|G|]$, to the Reachability problem in a suitably defined directed acyclic graph, denoted by $D_G(\pi, k)$, which we call *compatibility graph* of the triple (G, π, k). The formal definition of such a graph is properly given later on. Next, we describe how this section is organized.

In Section 3.1, we define the concept of *compatible subcut sequence* of a directed graph G with respect to a bijection $\pi : V(G) \to [|G|]$. Based on this concept, we provide a necessary and sufficient condition for $\mathsf{zn}(G, \pi) \geq k + 1$. Considering such a condition, we formally define in Section 3.2 the notion of *compatibility graph* and, then, we introduce a characterization relating the existence of the compatible subcut sequences of interest to the existence of directed paths with $|G| - 1$ vertices in the compatibility graph of (G, π, k). The proof of this characterization is presented in Section 3.3.

3.1. Compatible subcut sequence

Let G be a directed graph on n vertices and $\pi : V(G) \to [n]$ be a bijection. For simplicity, assume throughout this section that $V(G) = \{u_1, \dots, u_n\}$ and, for each $u_i, u_j \in V(G)$, $i < j$ if and only if $u_i <_\pi u_j$.

The *cut sequence of G with respect to* π is defined as the sequence

$$\gamma_{G,\pi} \doteq \langle S_G(\pi, 1), \dots, S_G(\pi, n - 1)\rangle.$$

For each $i, j \in [n - 1]$, with $i < j$, and each two *subcuts* $S'_i \subseteq S_G(\pi, i)$ and $S'_j \subseteq S_G(\pi, j)$, we say that S'_i is *compatible* with S'_j, and we denote this fact by $S'_i \prec_{G,\pi} S'_j$, if for each $e = (u, v) \in E(G)$ the two conditions below are observed.

(1) If $e \in S'_i$, and either $\pi(u) > j$ or $\pi(v) > j$, then $e \in S'_j$.
(2) If $e \in S_G(\pi, i) \setminus S'_i$, then $e \notin S'_j$.

Intuitively, condition (1) says that, if e belongs to the subcut S'_i and either the order of u or the order of v, with respect to π, is greater than j, then e must belong to the subcut S'_j. On the other hand, condition (2) says that if e belongs to the cut $S_G(\pi, i)$ but does not belong to the subcut S'_i, then e cannot belong to the subcut S'_j.

A *compatible subcut sequence* of $\gamma_{G,\pi}$ is a sequence of subcuts

$$\gamma' = \langle S'_1, \dots, S'_{n-1}\rangle$$

such that $S'_i \subseteq S_G(\pi, i)$ for each $i \in [n - 1]$, and $S'_j \prec_{G,\pi} S'_{j+1}$ for each $j \in [n - 2]$. A neat idea behind the definition of compatible subcut sequence is to focus on neighboring subcuts S'_j and S'_{j+1} for each $j \in [n-2]$. We let $\boldsymbol{\Gamma}(\gamma_{G,\pi})$ be the set of all compatible subcut sequences of $\gamma_{G,\pi}$.

The next proposition establishes that the compatibility conditions (1) and (2) described above are sufficient to ensure that, if S'_i and S'_j are two subcuts in a same compatible subcut sequence, then there do not exist any inconsistency with respect to the edges that belong to S'_i and to S'_j. More specifically, provided that S'_i and S'_j are two subcuts in a same compatible subcut sequence, for any edge e belonging simultaneously to the cuts $S_G(\pi, i)$ and $S_G(\pi, j)$, we have that e belongs to S'_i if and only if it belongs to S'_j.

Proposition 1. *Let G be a directed graph, $\pi : V(G) \to [|G|]$ be a bijection and $\gamma' = \langle S'_1, \dots, S'_{|G|-1}\rangle \in \boldsymbol{\Gamma}(\gamma_{G,\pi})$. For each edge $e \in E(G)$ and each $i \in [|G| - 1]$, if $e \in S_G(\pi, i) \setminus S'_i$, then $e \notin S'_j$ for any $j \in [|G| - 1]$.*

Proof. Let $i, j \in [|G| - 1]$, with $i \neq j$, and $e = (u, v)$ be an edge in $S_G(\pi, i)$. The proof is split into two cases.

First, assume that $i < j$. Note that $i < |G| - 1$, otherwise $j < i$. Moreover, we have by hypothesis that $S'_l \prec_{G,\pi} S'_{l+1}$ for each $l \in \{i, \dots, |G| - 2\}$. Thus, if $e \in S_G(\pi, l) \setminus S'_l$, then $e \notin S'_{l+1}$ for each $l \in \{i, \dots, |G| - 2\}$. This inductively implies that, if $e \in S_G(\pi, i) \setminus S'_i$, then either $e \notin S_G(\pi, l)$ or $e \in S_G(\pi, l) \setminus S'_l$ for each $l \in \{i + 1, \dots, |G| - 1\}$. In particular, if $e \in S_G(\pi, i) \setminus S'_i$, then $e \notin S'_j$.

Now, assume that $i > j$. We prove that, if $e \in S'_j$, then $e \in S'_i$. Thus, additionally assume that $e \in S'_j$. Since $e \in S_G(\pi, j)$ and $e \in S_G(\pi, i)$, either $\pi(u) > l$ or $\pi(v) > l$ for each $l \in \{j, \dots, i\}$. This and the hypothesis that $S'_l \prec_{G,\pi} S'_{l+1}$ imply that, if $e \in S'_l$, then $e \in S'_{l+1}$ for each $l \in \{j, \dots, i - 1\}$. Therefore, since $e \in S'_j$, we inductively obtain that $e \in S'_l$ for each $l \in \{j, \dots, i\}$. In particular, $e \in S'_i$. □

Let $\gamma' = \langle S'_1, \dots, S'_{n-1}\rangle$ be a compatible subcut sequence in $\boldsymbol{\Gamma}(\gamma_{G,\pi})$. The *width* of γ' is defined as

$$\omega(\gamma') \doteq \max_{i \in [n-1]} |S'_i|.$$

If $E' = \bigcup_{i \in [n-1]} S'_i \neq \emptyset$, then we define $G[\gamma'] \doteq G[E']$ as the directed graph induced by E'. In particular, we remark that $\gamma_{G,\pi}$ is a compatible subcut sequence itself of width $\mathrm{cw}(G, \pi)$ and that $G[\gamma_{G,\pi}]$ consists in the directed graph obtained from G by removing all of its isolated vertices.

The next lemma states that deciding whether $\mathsf{zn}(G, \pi) \geq k+1$ is equivalent to deciding whether there is a compatible subcut sequence of $\gamma_{G,\pi}$ of width at least $k + 1$, whose associated directed graph is a directed path.

Lemma 1. *Let G be a directed graph, $\pi : V(G) \to [|G|]$ be a bijection and k be a non-negative integer. Then, $\mathsf{zn}(G, \pi) \geq k + 1$ if and only if there is a compatible subcut sequence $\gamma' = \langle S'_1, \dots, S'_{|G|-1}\rangle \in \boldsymbol{\Gamma}(\gamma_{G,\pi})$ such that $\omega(\gamma') \geq k + 1$ and $G[\gamma']$ is a directed path.*

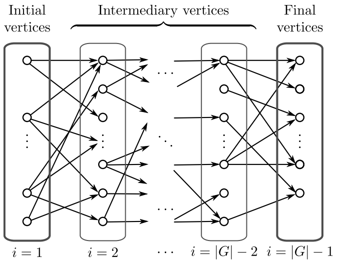


Fig. 2. A compatibility graph.

Proof. First, suppose that $\mathsf{zn}(G, \pi) \geq k + 1$, and let P be a directed path of G such that $\mathsf{zn}(G, \pi, P) \geq k + 1$. Consider the sequence $\gamma' = \langle S'_1, \ldots, S'_{|G|-1}\rangle$ of subcuts such that $S'_i = E(P) \cap S_G(\pi, i)$ for each $i \in [|G| - 1]$. We prove that γ' is a compatible subcut sequence of $\gamma_{G,\pi}$. In other words, we prove that $S'_i \prec_{G,\pi} S'_{i+1}$ for each $i \in [|G| - 2]$. Note that, if for some $i \in [|G| - 2]$ there exists an edge $e \in S_G(\pi, i) \setminus S'_i$, then $e \in E(G) \setminus E(P)$ and, consequently, $e \notin S'_{i+1}$. Now, suppose that for some $i \in [|G| - 2]$ there exists an edge $e = (u, v) \in S'_i$ such that either $\pi(u) > i + 1$ or $\pi(v) > i + 1$. Clearly, $e \in E(P)$. Moreover, note that $e \in S_G(\pi, i + 1)$. Thus, $e \in S'_{i+1}$, otherwise $e \in S_G(\pi, i + 1) \setminus S'_{i+1}$, which would imply that $e \in E(G) \setminus E(P)$. Therefore, γ' is indeed a compatible subcut sequence of $\gamma_{G,\pi}$. Additionally, one can straightforwardly verify that $\omega(\gamma') \geq k + 1$ and $G[\gamma']$ is a directed path.

Conversely, suppose that there exists a compatible subcut sequence $\gamma' = \langle S'_1, \ldots, S'_{|G|-1}\rangle$ of $\gamma_{G,\pi}$ such that $\omega(\gamma') \geq k+1$ and $G[\gamma']$ is a directed path. Thus, there exists $i \in [|G| - 1]$ such that $|S'_i| \geq k + 1$. As a result, if $P = G[\gamma']$, then $\mathsf{zn}(G, \pi, P) \geq |S'_i| \geq k + 1$. Therefore, $\mathsf{zn}(G, \pi) \geq \mathsf{zn}(G, \pi, P) \geq k + 1$. □

3.2. Compatibility graph

In this section, we define the notion of *compatibility graph*. Intuitively, each directed path with $|G| - 1$ vertices of the *compatibility graph* $D_G(\pi, k)$ corresponds to a compatible subcut sequence γ' of $\gamma_{G,\pi}$ satisfying the conditions described in Lemma 1. More specifically, the vertices of $D_G(\pi, k)$ consist in special tuples which, along with the directed edges between them, define a dynamic programming table. This table stores all the information needed to guarantee that, if there is a directed path in $D_G(\pi, k)$ with $|G| - 1$ vertices, then there exists a compatible subcut sequence γ' of $\gamma_{G,\pi}$ such that $\omega(\gamma') \geq k + 1$.

In order to capture the above property, we partition the vertex set of $D_G(\pi, k)$ into $|G|-1$ distinct levels, such that each level $i \in [|G| - 1]$ is associated with the cut $S_G(\pi, i)$ and there is a *directed* edge in $D_G(\pi, k)$ from a vertex $\mathfrak{u}$ to a vertex $\mathfrak{v}$ only if $\mathfrak{u}$ belongs to a level i and $\mathfrak{v}$ belongs to the level $i + 1$, and some additional constraints (described in Section 3.2.2) are satisfied. The vertices in the level $i = 1$ are called *initial*, the vertices in a level $i \in [|G| - 1] \setminus \{1, |G| - 1\}$ are called *intermediary*, and the vertices in the level $i = |G| - 1$ are called *final*. We note that, by definition, the initial vertices of $D_G(\pi, k)$ have in-degree 0 and the final vertices of $D_G(\pi, k)$ have out-degree 0. Fig. 2 illustrates the partitioning of the vertex set of the compatibility graph $D_G(\pi, k)$ into these $|G| - 1$ distinct levels.

One can alternatively regard $D_G(\pi, k)$ as an acyclic finite automaton — with transition set defined by the adjacency relation described in Section 3.2.2. From this perspective, the initial vertices represent the initial states of the automaton and the final vertices represent the final states of the automaton.

The following immediate observation provides the basis for the definition of compatibility graph.

Observation 1. *A directed graph P is a directed path if and only if it satisfies the following four conditions:*

(DP1) P has exactly one vertex, called source vertex, with in-degree 0 *and out-degree* 1*;*
(DP2) P has exactly one vertex, called target vertex, with in-degree 1 *and out-degree* 0*;*
(DP3) All the other vertices of P have in-degree 1 *and out-degree* 1*;*
(DP4) P is weakly connected.

In particular, for a compatible subcut sequence γ' of $\gamma_{G,\pi}$, we have that $G[\gamma']$ is a directed path if and only if it satisfies Conditions (DP1)–(DP4). Based on that, and aiming at devising a dynamic programming method that determines whether there exists a compatible subcut sequence γ' such that $G[\gamma']$ is a directed path and that, more generally, satisfies the conditions described in Lemma 1, we define the set $\boldsymbol{B}_G(\pi, k)$. This set consists of all tuples of the form

$$\mathfrak{u}_i = \left(i, S'_i, \phi_i, \varphi_i, \mathcal{S}_i, \tau_i, \psi_i\right),$$

where $i \in [n - 1]$, S'_i is a subcut of $S_G(\pi, i)$ with cardinality at most $k + 1$, $\mathcal{S}_i$ is either an empty set or a partition of S'_i, $\phi_i, \varphi_i, \tau_i \in \{0, 1, 2\}$ are ternary flags and $\psi_i \in \{0, 1\}$ is a boolean flag. We remark that, for each $i \in [n - 1]$, there are at most $n^{\mathcal{O}(k)}$ distinct tuples $\mathfrak{u}_i \in \boldsymbol{B}_G(\pi, k)$. Indeed, for each $i \in [n - 1]$, the cut $S_G(\pi, i)$ has at most $2i(n - i) \leq n^2/2$ directed

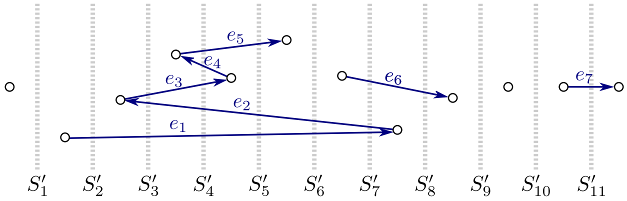

e_1
e_2
e_3
e_4
e_5
e_6
e_7
S'_1 S'_2 S'_3 S'_4 S'_5 S'_6 S'_7 S'_8 S'_9 S'_{10} S'_{11}

Fig. 3. Example of a compatible subcut sequence: $\gamma' = \langle S'_1, \dots, S'_{11}\rangle$, where $S'_1 = \emptyset$, $S'_2 = \{e_1\}$, $S'_3 = \{e_1, e_2, e_3\}$, $S'_4 = \{e_1, e_2, e_3, e_4, e_5\}$, $S'_5 = \{e_1, e_2, e_5\}$, $S'_6 = \{e_1, e_2\}$, $S'_7 = \{e_1, e_2, e_6\}$, $S'_8 = \{e_6\}$, $S'_9 = \emptyset$, $S'_{10} = \emptyset$, and $S'_{11} = \{e_7\}$.

edges, which is the maximum possible number of directed edges between the vertices belonging to $\{u_1, \dots, u_i\}$ and the vertices belonging to $\{u_{i+1}, \dots, u_n\}$. Thus, $S_G(\pi, i)$ has at most $\binom{n^2/2}{k+1} = \mathcal{O}(n^{2k+2})$ distinct subcuts S'_i of cardinality at most $k+1$, and each such a subcut S'_i admits at most $(k+1)^{\mathcal{O}(k)}$ distinct partitions.

Let $i \in [n-1]$ and $\mathfrak{p}_i = \left(1, S'_1, \phi_1, \varphi_1, \mathcal{S}_1, \tau_1, \psi_1\right), \dots, \left(i, S'_i, \phi_i, \varphi_i, \mathcal{S}_i, \tau_i, \psi_i\right)$ be a sequence of tuples, such that S'_j is compatible with S'_{j+1} for each $j \in [i-1]$. Then, let H_i be the subgraph of G with vertex set $V(H_i) = \{u_1, \dots, u_i\} \cup X_i$ and edge set $E(H_i) = S'_1 \cup \dots \cup S'_i$, where X_i denotes the set of endpoints of the edges in $S'_1 \cup \dots \cup S'_i$. Note that H_i may contain isolated vertices.

Intuitively, the ternary flag ϕ_i (the ternary flag φ_i, resp.) informs whether there exist zero, one, or more than one vertices from $\{u_1, \dots, u_i\}$ that are source vertices (target vertices, resp.) of H_i.

The partition $\mathcal{S}_i$ represents the set of all non-trivial weakly connected components of H_i, *restricted to the subcut* S'_i, that are defined by only taking into account the vertices from $\{u_1, \dots, u_i\}$. In other words, two edges $e, e' \in S'_i$ belong to a same part of $\mathcal{S}_i$ if and only if there exists an undirected path of H_i between an endpoint of e and an endpoint of e' that only uses vertices from $\{u_1, \dots, u_i\}$. For instance, consider the compatible subcut sequence γ' illustrated in Fig. 3. In this example, $\mathcal{S}_1 = \emptyset$, $\mathcal{S}_2 = \{\{e_1\}\}$, $\mathcal{S}_3 = \{\{e_1\}, \{e_2, e_3\}\}$, $\mathcal{S}_4 = \{\{e_1\}, \{e_2, e_3\}, \{e_4, e_5\}\}$, $\mathcal{S}_5 = \{\{e_1\}, \{e_2, e_5\}\}$, $\mathcal{S}_6 = \{\{e_1\}, \{e_2\}\}$, $\mathcal{S}_7 = \{\{e_1\}, \{e_2\}, \{e_6\}\}$, $\mathcal{S}_8 = \{\{e_6\}\}$, $\mathcal{S}_9 = \emptyset$, $\mathcal{S}_{10} = \emptyset$, and $\mathcal{S}_{11} = \{\{e_7\}\}$.

The ternary flag τ_i informs whether there exist zero, one, or more than one non-trivial weakly connected components of H_i that do not contain any of the vertices from $\{u_{i+1}, \dots, u_n\}$. For instance, consider again the compatible subcut sequence γ' illustrated in Fig. 3. In this example, $\tau_i = 0$ for each $i \in \{1, \dots, 7\}$, $\tau_8 = 1$, and $\tau_i = 2$ for each $i \in \{9, 10, 11\}$.

Finally, the boolean flag ψ_i informs whether or not there exists a subcut of width $k+1$ among the subcuts $S'_1, \dots, S'_i$.

3.2.1. Initial, final and intermediary tuples

Now, we present the formal definitions of the notions of *initial*, *final* and *intermediary* tuples, which precisely comprise the vertex set of $D_G(\pi, k)$. More specifically, the initial tuples correspond to the initial vertices of $D_G(\pi, k)$, the final tuples correspond to the final vertices of $D_G(\pi, k)$, and the intermediary tuples correspond to the intermediary vertices of $D_G(\pi, k)$.

Let $\mathfrak{u} = \left(i, S'_i, \phi_i, \varphi_i, \mathcal{S}_i, \tau_i, \psi_i\right)$ be a tuple in $\boldsymbol{B}_G(\pi, k)$.

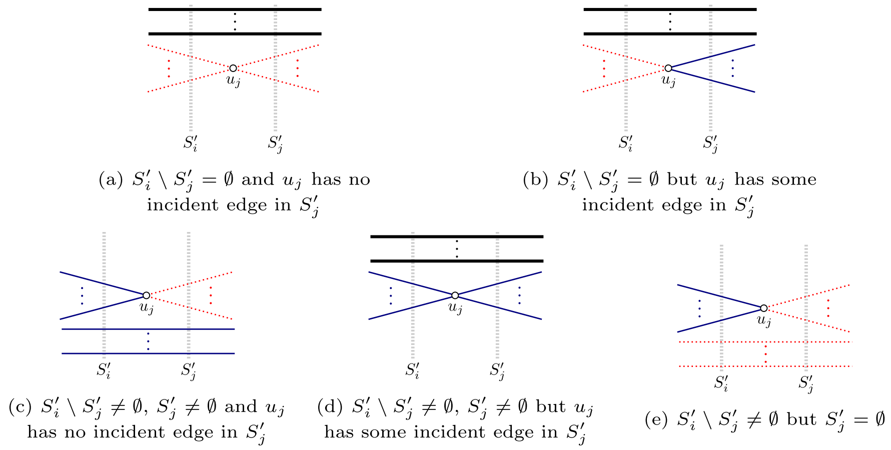
We say that $\mathfrak{u}$ is *initial* if $i = 1$ and the following conditions are satisfied:

1. The vertex u_1 has at most one in-edge and at most one out-edge in S'_1;
2. If u_1 has no in-edge and one out-edge in S'_1, then $\phi_1 = 1$ and $\varphi_1 = 0$;
3. If u_1 has one in-edge and no out-edge in S'_1, then $\phi_1 = 0$ and $\varphi_1 = 1$;
4. If u_1 has one in-edge and one out-edge in S'_1 or does not have any incident edge in S'_1, then $\phi_1 = 0$ and $\varphi_1 = 0$;
5. If $S'_1 = \emptyset$, then $\mathcal{S}_1 = \emptyset$; otherwise, $\mathcal{S}_1 = \left\{S'_1\right\}$;
6. $\tau_1 = 0$;
7. If $|S'_1| = k+1$, then $\psi_1 = 1$; otherwise, $\psi_1 = 0$.

On the other hand, we say that $\mathfrak{u}$ is *final* if $i = n-1$ and the following conditions are satisfied:

1. The vertex u_n has at most one in-edge and at most one out-edge in S'_{n-1};
2. If u_n has no in-edge and one out-edge in S'_{n-1}, then $\phi_{n-1} = 0$ and $\varphi_{n-1} = 1$;
3. If u_n has one in-edge and no out-edge in S'_{n-1}, then $\phi_{n-1} = 1$ and $\varphi_{n-1} = 0$;
4. If u_n has one in-edge and one out-edge in S'_{n-1} or does not have any incident edge in S'_{n-1}, then $\phi_{n-1} = 1$ and $\varphi_{n-1} = 1$;
5. $|\mathcal{S}_{n-1}| \le 2$, and if $|\mathcal{S}_{n-1}| = 1$, then $|S'_{n-1}| = 1$;
6. $\tau_{n-1} \le 1$, and if $\tau_{n-1} = 1$, then $S'_{n-1} = \emptyset$;
7. $\psi_{n-1} = 1$.

Intuitively, the tuple $\mathfrak{u}$ is called initial (final, resp.) if $i = 1$ ($i = n-1$, resp.) and the values assigned to the parameters S'_i, ϕ_i, φ_i, $\mathcal{S}_i$, τ_i and ψ_i establish a valid configuration with respect to the semantic of each parameter itself and with respect to Conditions (DP1)–(DP4).



(a) $S'_i \setminus S'_j = \emptyset$ and u_j has no incident edge in S'_j

(b) $S'_i \setminus S'_j = \emptyset$ but u_j has some incident edge in S'_j

(c) $S'_i \setminus S'_j \neq \emptyset$, $S'_j \neq \emptyset$ and u_j has no incident edge in S'_j

(d) $S'_i \setminus S'_j \neq \emptyset$, $S'_j \neq \emptyset$ but u_j has some incident edge in S'_j

(e) $S'_i \setminus S'_j \neq \emptyset$ but $S'_j = \emptyset$

Fig. 4. Connectedness rules. (Red) dotted lines represent non-edges, (black) thicker lines represent non-mandatory edges, and (blue) normal style lines represent mandatory edges.

Finally, we say $\mathfrak{u}$ is *intermediary* if $i \in [n-1] \setminus \{1, n-1\}$.

3.2.2. Compatibility relation

Let $\mathfrak{u} = \left(i, S'_i, \phi_i, \varphi_i, \mathcal{S}_i, \tau_i, \psi_i\right)$ and $\mathfrak{v} = \left(j, S'_j, \phi_j, \varphi_j, \mathcal{S}_j, \tau_j, \psi_j\right)$ be a pair of tuples from $\boldsymbol{B}_G(\pi, k)$. We say that $\mathfrak{u}$ is *compatible* with $\mathfrak{v}$, and we denote such a fact by $\mathfrak{u} \rightsquigarrow \mathfrak{v}$, if $j = i + 1$, $S'_i \prec_{G,\pi} S'_j$, and $\mathfrak{u}$ and $\mathfrak{v}$ satisfy the Vertex degree, Connectedness and Minimum subcut width rules, which are presented below.

Vertex degree rules.

1. The vertex u_j has at most one in-edge and at most one out-edge in $S'_i \cup S'_j$.
2. If u_j has no in-edge and one out-edge in $S'_i \cup S'_j$, then $\phi_j = \min\{2, \phi_i + 1\}$ and $\varphi_j = \varphi_i$.
3. If u_j has one in-edge and no out-edge in $S'_i \cup S'_j$, then $\phi_j = \phi_i$ and $\varphi_j = \min\{2, \varphi_i + 1\}$.
4. If u_j has one in-edge and one out-edge in $S'_i \cup S'_j$ or does not have any incident edge in $S'_i \cup S'_j$, then $\phi_j = \phi_i$ and $\varphi_j = \varphi_i$.

Connectedness rules.

1. If $S'_i \setminus S'_j = \emptyset$ and u_j has no incident edge in S'_j (see Fig. 4(a)), then $\tau_j = \tau_i$ and $\mathcal{S}_j = \mathcal{S}_i$.
2. If $S'_i \setminus S'_j = \emptyset$ but u_j has some incident edge in S'_j (see Fig. 4(b)), then $\tau_j = \tau_i$ and $\mathcal{S}_j = \mathcal{S}_i \cup \left\{S'_j \setminus S'_i\right\}$.
3. If $S'_i \setminus S'_j \neq \emptyset$ and $S'_j \neq \emptyset$ (see Figs. 4(c) and 4(d)), then $\tau_j = \tau_i$ and $\mathcal{S}_j = (\mathcal{S}_i \setminus \mathcal{Q}'_j) \cup \mathcal{Q}_j$, where $\mathcal{Q}'_j$ denotes the collection of all sets in $\mathcal{S}_i$ that have at least one edge in S'_i with u_j as an endpoint, *i.e.*
$$\mathcal{Q}'_j = \left\{Q \in \mathcal{S}_i : Q \cap (S'_i \setminus S'_j) \neq \emptyset\right\},$$
and $\mathcal{Q}_j$ denotes the singleton collection whose set comprises all edges in S'_j with u_j as an endpoint, along with all edges in S'_j that belong to a set of $\mathcal{Q}'_j$, *i.e.*
$$\mathcal{Q}_j = \left\{(S'_j \setminus S'_i) \cup \left(\bigcup_{Q \in \mathcal{Q}'_j} Q \cap S'_j\right)\right\}.$$
In this case, we further require $\mathcal{Q}_j \neq \emptyset$.
Informally, $\mathcal{Q}'_j$ represents the set of non-trivial weakly connected components *restricted to* S'_i that have at least one edge in S'_i with u_j as an endpoint. Since, when considering the subcut S'_j, all such components contain u_j as a common vertex, they actually form a single non-trivial weakly connected component *restricted to* S'_j. This single component is represented by $\mathcal{Q}_j$, which, besides the edges that are already present in S'_i, contains all the edges in S'_j with u_j as an endpoint.
4. If $S'_i \setminus S'_j \neq \emptyset$ but $S'_j = \emptyset$ (see Fig. 4(e)), then $\tau_j = \min\{2, \tau_i + 1\}$ and $\mathcal{S}_j = \emptyset$.

Minimum subcut width rule.

(1) If $\psi_i = 1$ or $|S'_j| = k + 1$, then $\psi_j = 1$.

We notice that, for any sequence $\langle \mathfrak{u}_1, \ldots, \mathfrak{u}_{n-1} \rangle$ of tuples from $\boldsymbol{B}_G(\pi, k)$, such that $\mathfrak{u}_i$ is compatible with $\mathfrak{u}_{i+1}$ for each $i \in [n-2]$, there exists a unique associated compatible subcut sequence $\gamma' \in \boldsymbol{\Gamma}(\gamma_{G,\pi})$. Thus, the intuition behind the Vertex degree and Connectedness rules is ensuring that, if γ' is the subcut sequence associated with a directed path $\langle \mathfrak{u}_1, \ldots, \mathfrak{u}_{n-1} \rangle$ in $D_G(\pi, k)$, then $G[\gamma']$ satisfies Conditions (DP1)–(DP4). And, the intuition behind the Minimum subcut width rule is ensuring that the width of any such compatible subcut sequences γ' is at least $k+1$.

Now, we are finally able to formally define the notion of *compatibility graph* and then prove Theorem 1.

For each directed graph G, each bijection $\pi : V(G) \to [|G|]$ and each non-negative integer k, we define the *compatibility graph* of the triple (G, π, k) as the directed acyclic graph $D_G(\pi, k)$ with vertex set

$$V = \{\mathfrak{u} \in \boldsymbol{B}_G(\pi, k) \colon \mathfrak{u} \text{ is initial, intermediary or final}\}$$

and edge set

$$E = \{(\mathfrak{u}, \mathfrak{v}) \in V \times V : \mathfrak{u} \rightsquigarrow \mathfrak{v}\}.$$

Lemma 2 states that deciding whether there exists a compatible subcut sequence of $\gamma_{G,\pi}$ of width at least $k+1$ whose associated directed graph is a directed path is equivalent to deciding whether there exists a directed path of $D_G(\pi, k)$ with $n-1$ vertices. Then, based on this characterization and on Lemma 1, we prove in Lemma 3 that deciding whether $\mathsf{zn}(G, \pi) \geq k+1$ is reducible to the Reachability problem in $D_G(\pi, k)$.

Section 3.3 is devoted to present the proof of Lemma 2.

Lemma 2. *Let G be a directed graph, $\pi : V(G) \to [|G|]$ be a bijection and k be a non-negative integer. There exists a compatible subcut sequence $\gamma' \in \boldsymbol{\Gamma}(\gamma_{G,\pi})$ such that $\omega(\gamma') \geq k+1$ and $G[\gamma']$ is a directed path if and only if there exists a directed path of $D_G(\pi, k)$ with $|G|-1$ vertices.*

Lemma 3. *Given a directed graph G, a bijection $\pi : V(G) \to [|G|]$ and a non-negative integer k, one can deterministically decide in time $|G|^{\mathcal{O}(k)}$ whether $zn(G, \pi) \leq k$.*

Proof. First, we construct the directed graph $D_G(\pi, k)$. Note that, for each tuple

$$\mathfrak{u}_i = \left(i, S'_i, \phi_i, \varphi_i, \mathcal{S}_i, \tau_i, \psi_i\right) \in \boldsymbol{B}_G(\pi, k),$$

the subcut S'_i has at most $k+1$ distinct elements. As a result, one can easily check in time polynomial in k if $\mathfrak{u}_i$ is an *initial*, an *intermediary* or a *final* tuple. Moreover, since there are $|G|^{\mathcal{O}(k)}$ distinct tuples in $\boldsymbol{B}_G(\pi, k)$, the vertex set of $D_G(\pi, k)$ can be determined in time $|G|^{\mathcal{O}(k)} \cdot \mathsf{poly}(k) = |G|^{\mathcal{O}(k)}$. Regarding the edge set of $D_G(\pi, k)$, we have by definition that there exists a directed edge from a vertex $\mathfrak{u}_i = \left(i, S'_i, \phi_i, \varphi_i, \mathcal{S}_i, \tau_i, \psi_i\right)$ to a vertex $\mathfrak{u}_j = \left(j, S'_j, \phi_j, \varphi_j, \mathcal{S}_j, \tau_j, \psi_j\right)$ of $D_G(\pi, k)$ if and only if $\mathfrak{u}_i$ is *compatible with* $\mathfrak{u}_j$, *i.e.*, $j = i+1$, $S'_i \prec_{G,\pi} S'_j$, and $\mathfrak{u}$ and $\mathfrak{v}$ satisfy the Vertex degree, Connectedness and Minimum subcut width rules. Since $|S'_i| \leq k+1$ and $|S'_j| \leq k+1$, the satisfaction of the Vertex degree, Connectedness and Minimum subcut width rules by $\mathfrak{u}$ and $\mathfrak{v}$ can be clearly checked in time polynomial in k. In addition, one can verify whether $S'_i \prec_{G,\pi} S'_j$ in time polynomial in $|G|$. Thus, it can be checked in time $\mathsf{poly}(|G|, k)$ whether there should exist in $D_G(\pi, k)$ a directed edge from $\mathfrak{u}$ to $\mathfrak{v}$. This implies that the edge set of $D_G(\pi, k)$ can be determined in time $|G|^{\mathcal{O}(k)} \cdot |G|^{\mathcal{O}(k)} \cdot \mathsf{poly}(|G|, k) = |G|^{\mathcal{O}(k)}$. Therefore, $D_G(\pi, k)$ can be wholly constructed in time $|G|^{\mathcal{O}(k)}$.

Then, by using an algorithm for the Reachability problem, we decide in time linear in the number of vertices and edges of $D_G(\pi, k)$, *i.e.* in time $|G|^{\mathcal{O}(k)}$, whether there is a directed path of $D_G(\pi, k)$ with $|G|-1$ vertices. By Lemmas 1 and 2, such a path exists if and only if $\mathsf{zn}(G, \pi) \geq k+1$. Therefore, we can decide in time $|G|^{\mathcal{O}(k)}$ whether $\mathsf{zn}(G, \pi) \leq k$. □

As a result, we obtain that deciding whether $\mathsf{zn}(G) \leq k$ is in NP for each fixed $k \geq 0$, concluding thereby the proof of Theorem 1.

3.3. Proof of Lemma 2

Assume that $V(G) = \{u_1, \ldots, u_n\}$ and, for each $u_i, u_j \in V(G)$, $i < j$ if and only if $u_i <_\pi u_j$. Consider the following auxiliary claim.

Claim 1. *Let $\mathfrak{p} = \langle \mathfrak{u}_1, \ldots, \mathfrak{u}_{n-1} \rangle$ be a sequence of tuples such that*

1. *for each $i \in [n-1]$, $\mathfrak{u}_i = \left(i, S'_i, \phi_i, \varphi_i, \mathcal{S}_i, \tau_i, \psi_i\right) \in \boldsymbol{B}_G(\pi, k)$;*
2. *$\mathfrak{u}_1$ is initial;*
3. *for each $i \in [n-2]$, $\mathfrak{u}_i \rightsquigarrow \mathfrak{u}_{i+1}$;*

and let $\gamma' = \langle S'_1, \ldots, S'_{n-1} \rangle$ be the compatible subcut sequence corresponding to $\mathfrak{p}$. Then, for each $\ell \in [n-2]$, we have that any two edges $e, e' \in S'_\ell$ belong to a same part of $\mathcal{S}_\ell$ if and only if there exists in $G[\gamma']$ an undirected path between x and y that only contains vertices from $\{u_1, \ldots, u_\ell\}$, where x and y denote the endpoints of e and e', respectively, that belong to $\{u_1, \ldots, u_\ell\}$.

Proof of claim. The proof is by induction on ℓ.

Base case. Suppose that $\ell = 1$. In this case, $x = y$. Then, trivially, there exists in $G[\gamma']$ an undirected path between x and y that only contains vertices from $\{u_1, \ldots, u_\ell\}$. Moreover, it follows from the fact that $\mathfrak{u}_1$ is an initial tuple that $\mathcal{S}_1 = \left\{S'_\ell\right\}$. Thus, e and e' belong to a same part of $\mathcal{S}_1$.

Inductive hypothesis. Suppose that there exists $\iota \in [|G| - 1]$ such that the claim holds for each $\ell \in [\iota - 1]$.

Inductive step. Suppose that $\ell = \iota > 1$. First, consider $x = y$, and let $i = \pi(x) \leq \iota$. Similarly to the base case, there trivially exists an undirected path of $G[\gamma']$ between x and y that only contains vertices from $\{u_1, \ldots, u_\iota\}$. Moreover, since $\mathfrak{u}_{i-1} \rightsquigarrow \mathfrak{u}_i$, it follows from the Connectedness rules that e and e' belong to a same part of $\mathcal{S}_i$. As a result, we obtain that e and e' also belong to a same part of $\mathcal{S}_\iota$. Thus, in what follows, consider $x \neq y$. Additionally, assume without loss of generality that $\pi(x) < \pi(y)$.

First, we prove that, if e and e' belong to a same part of $\mathcal{S}_\iota$, then there is in $G[\gamma']$ an undirected path between x and y that only contains vertices from $\{u_1, \ldots, u_\iota\}$. Thus, suppose e and e' belong to a same part of $\mathcal{S}_\iota$. Note that, if e and e' belong to a same part of $\mathcal{S}_i$ for some $i < \iota$, then the result immediately follows from the inductive hypothesis. Thus, assume that ι is the least integer $j \in \{\pi(y), \ldots, n - 1\}$ such that e and e' belong to a same part of $\mathcal{S}_j$.

Consider $\iota = \pi(y)$. Since $\mathfrak{u}_{\iota-1} \rightsquigarrow \mathfrak{u}_\iota$, it follows from the Connectedness rules that there exists an edge $e'' \in S'_{\iota-1} \setminus S'_\iota$ such that e and e'' belong to a same part of $\mathcal{S}_{\iota-1}$, otherwise e and e' would belong to distinct parts of $\mathcal{S}_\iota$. Then, let z be the endpoint of e'' that belongs to $\{u_1, \ldots, u_{\iota-1}\}$. By the inductive hypothesis, there exists in $G[\gamma']$ an undirected path between x and z that only contains vertices from $\{u_1, \ldots, u_{\iota-1}\}$. Moreover, since $\iota = \pi(y)$ and $e'' \in S'_{\iota-1} \setminus S'_\iota$, we have that y is an endpoint of e''. Therefore, there exists in $G[\gamma']$ an undirected path between x and y that only contains vertices from $\{u_1, \ldots, u_\iota\}$.

Now, consider $\iota > \pi(y)$. By the Connectedness rules, there exist two distinct edges $e''_x, e''_y \in S'_{\iota-1} \setminus S'_\iota$, such that e and e''_x belong to a same part of $\mathcal{S}_{\iota-1}$ and e' and e''_y belong to a same part of $\mathcal{S}_{\iota-1}$, otherwise e and e' would belong to distinct parts of $\mathcal{S}_\iota$. Then, let z_x and z_y be the endpoints of e''_x and e''_y, respectively, that belong to $\{u_1, \ldots, u_{\iota-1}\}$. By the inductive hypothesis and by the minimality of ι, there exist in $G[\gamma']$ two vertex-disjoint undirected paths $P_x = \langle x, \ldots, z_x\rangle$ and $P_y = \langle z_y, \ldots, y\rangle$ that only contain vertices from $\{u_1, \ldots, u_{\iota-1}\}$, where P_x is a path between x and z_x, and P_y is a path between z_y and y. Therefore, since z''_x and z''_y are both neighbors of u_ι, $P_x + \langle u_\iota\rangle + P_y = \langle x, \ldots, z_x, u_\iota, z_y, \ldots, y\rangle$ is an undirected path between x and y that only use vertices from $\{u_1, \ldots, u_\iota\}$.

Now, we prove the converse part, *i.e.* if there is in $G[\gamma']$ an undirected path between x and y that only contains vertices from $\{u_1, \ldots, u_\iota\}$, then e and e' belong to a same part of $\mathcal{S}_\iota$. Thus, suppose that there is in $G[\gamma']$ an undirected path P between x and y that only contains vertices from $\{u_1, \ldots, u_\iota\}$. Also, assume that ι is the least integer in $\{\pi(y), \ldots, n - 1\}$ holding such a property.

Consider $\iota = \pi(y)$. In this case, one can verify that there exists exactly one edge e'' in the set $E(P) \cap S'_{\iota-1} \setminus S'_\iota$. Let z be the endpoint of e'' that belongs to $\{u_1, \ldots, u_{\iota-1}\}$. Note that, $P - y = P - u_\iota$ is an undirected path between x and z that only contains vertices from $\{u_1, \ldots, u_{\iota-1}\}$. By the inductive hypothesis, e and e'' belong to a same part of $\mathcal{S}_{\iota-1}$. Therefore, we obtain by Connectedness rule 3 that e and e' belong to a same part of $\mathcal{S}_\iota$.

Now, consider $\iota > \pi(y)$. In this case, there exist exactly two distinct edges e''_x, e''_y in the set $E(P) \cap S'_{\iota-1} \setminus S'_\iota$. Let z_x and z_y be the endpoints of e''_x and e''_y, respectively, that belong to $\{u_1, \ldots, u_{\iota-1}\}$. Note that, $P - u_\iota$ consists of two undirected path P_x and P_y that only contain vertices from $\{u_1, \ldots, u_{\iota-1}\}$, where P_x is an undirected path between x and z_x, and P_y is an undirected path between z_y and y. Thus, it follows from the inductive hypothesis that e and e''_x belong to a same part of $\mathcal{S}_{\iota-1}$, and that e' and e''_y belong to a same part of $\mathcal{S}_{\iota-1}$. Therefore, by Connectedness rule 3, e and e' belong to a same part of $\mathcal{S}_\iota$. ■

Now, we are finally able to properly prove Lemma 2.

First, suppose that there exists $\gamma' = \langle S'_1, \ldots, S'_{n-1}\rangle \in \boldsymbol{\Gamma}(\gamma_{G,\pi})$ such that $P = G[\gamma']$ is a directed path and $\omega(\gamma') \geq k + 1$. Let $\mathfrak{u}_1 = \left(1, S'_1, \phi_1, \varphi_1, \mathcal{S}_1, \tau_1, \psi_1\right)$ be the initial tuple in $\boldsymbol{B}_G(\pi, k)$ obtained from the subcut S'_1. We notice that, given the subcut S'_1, the parameters ϕ_1, φ_1, $\mathcal{S}_1$, τ_1 and ψ_1 are uniquely determined according to the definition of initial tuple. Thus, $\mathfrak{u}_1$ is well-defined. Additionally, note that, according to the Vertex degree, Connectedness and Minimum subcut width rules, for each $i \in \{2, \ldots, n - 1\}$ and each tuple

$$\mathfrak{u}_{i-1} = \left(i - 1, S'_{i-1}, \phi_{i-1}, \varphi_{i-1}, \mathcal{S}_{i-1}, \tau_{i-1}, \psi_{i-1}\right) \in \boldsymbol{B}_G(\pi, k),$$

there exists exactly one tuple $\mathfrak{u}_i = \left(i, S'_i, \phi_i, \varphi_i, \mathcal{S}_i, \tau_i, \psi_i\right) \in \boldsymbol{B}_G(\pi, k)$ such that $\mathfrak{u}_{i-1}$ is compatible with (*i.e.* $\mathfrak{u}_{i-1} \rightsquigarrow \mathfrak{u}_i$).

Consequently, there exists a unique sequence $\langle \mathfrak{u}_1, \ldots, \mathfrak{u}_{n-1}\rangle$ of tuples from $\boldsymbol{B}_G(\pi, k)$ that can be obtained from γ' and satisfies the conditions of $\mathfrak{u}_1$ being initial and of $\mathfrak{u}_i$ being compatible with $\mathfrak{u}_{i+1}$ for each $i \in [n - 2]$.

We claim that such a sequence $\langle \mathfrak{u}_1, \ldots, \mathfrak{u}_{n-1}\rangle$ corresponds to a directed path of $D_G(\pi, k)$ with $n - 1$ vertices. To prove this, we just need to show that the tuple $\mathfrak{u}_{n-1} = \left(n - 1, S'_{n-1}, \phi_{n-1}, \varphi_{n-1}, \mathcal{S}_{n-1}, \tau_{n-1}, \psi_{n-1}\right)$ is final.

Since by hypothesis P is a directed path, every vertex of P has in-degree at most one and out-degree at most one. Moreover, P contains exactly one source vertex $u_\iota \in V(P)$ for some $\iota \in [n]$. Thus, one can verify that: if $\iota < n$, then $\phi_i = 0$ for each $i \in [\iota - 1]$ and $\phi_j = 1$ for each $j \in \{\iota, \ldots, n - 1\}$; and, if $\iota = n$, then $\phi_i = 0$ for each $i \in [n - 1]$. Similarly, P contains exactly one target vertex $u_{\iota'} \in V(P)$ for some $\iota' \in [n]$. Thus, one can verify that: if $\iota' < n$, then $\varphi_i = 0$ for each $i \in [\iota' - 1]$ and $\varphi_j = 1$ for each $j \in \left\{\iota', \ldots, n - 1\right\}$; and, if $\iota' = n$, then $\varphi_i = 0$ for each $i \in [n - 1]$. As a result, if $\iota \neq n$ and $\iota' \neq n$, then $\phi_{n-1} = 1$ and $\varphi_{n-1} = 1$. On the other hand, note that $\iota \neq \iota'$. Hence, if $\iota = n$ and $\iota' < n$, then $\phi_{n-1} = 0$ and $\varphi_{n-1} = 1$; and if $\iota < n$ and $\iota' = n$, then $\phi_{n-1} = 1$ and $\varphi_{n-1} = 0$.

Moreover, since $\omega(\gamma') \geq |S'_\iota| = k + 1$ for some $\iota \in [n-1]$, one can easily verify that $\psi_i = 0$ for each $i \in [\iota - 1]$ and $\psi_j = 1$ for each $j \in \{\iota, \dots, n-1\}$.

Now, let a and b be the least and the greatest integers in $[n-1]$, respectively, such that $S'_a \neq \emptyset$ and $S'_b \neq \emptyset$. Since by hypothesis $\omega(\gamma') \geq k + 1 \geq 1$, such integers a and b are well-defined. Thus, it follows from the fact that P is weakly connected that the following properties hold.

1. For each $i \in [a-2]$, $S'_i \setminus S'_{i+1} = \emptyset$ and u_{i+1} has no incident edge in S'_{i+1}, which implies $\tau_{i+1} = \tau_i = 0$.
2. $S'_{a-1} \setminus S'_a = \emptyset$ but u_a has some incident edge in S'_a, which implies $\tau_a = \tau_{a-1} = 0$.
3. For each $i \in \{a, \dots, b-1\}$, $S'_{i+1} \neq \emptyset$, which implies $\tau_{i+1} = \tau_i = 0$. In particular, if $b = n-1$, then $\tau_{n-1} = 0$. On the other hand, if $b < n-1$, then $S'_b \setminus S'_{b+1} \neq \emptyset$ and $S'_{b+1} = \emptyset$, which implies $\tau_{b+1} = \tau_b + 1 = 1$; moreover, $S'_{i-1} \setminus S'_i = \emptyset$ and $S'_i = \emptyset$ for each $i \in \{b+2, \dots, n-1\}$, which implies $\tau_i = \tau_{i-1} = 1$.

As a result, we obtain that $\tau_{n-1} \leq 1$, and that $\tau_{n-1} = 1$ implies $S'_{n-1} = \emptyset$.

Since u_n has at most one in-edge and at most one out-edge in S'_{n-1}, it is immediate that $|S'_{n-1}| \leq 2$ and $|\mathcal{S}_{n-1}| \leq 2$. Moreover, it follows from Claim 1 that $|\mathcal{S}_{n-1}| = 1$ implies $|S'_{n-1}| = 1$, otherwise P would not be a directed path.

Now, we prove the converse of Lemma 2. Suppose that there is in $D_G(\pi, k)$ a directed path $\mathfrak{p} = \langle \mathfrak{u}_1, \dots, \mathfrak{u}_{n-1} \rangle$ with $n-1$ vertices. One can verify that, for each $i \in [n-1]$, $\mathfrak{u}_i$ necessarily consists in a tuple in $\boldsymbol{B}_G(\pi, k)$ of the form

$$\mathfrak{u}_i = \left(i, S'_i, \phi_i, \varphi_i, \mathcal{S}_i, \tau_i, \psi_i\right).$$

Note that, $\gamma' = \langle S'_1, \dots, S'_{n-1} \rangle \in \boldsymbol{\Gamma}(\gamma_{G,\pi})$. Moreover, it follows from the definition of $D_G(\pi, k)$ that the tuple $\mathfrak{u}_{n-1}$ is final. As a result, $\psi_{n-1} = 1$, and thus $\omega(\gamma') \geq k + 1$, since by the Minimum subcut width rule we have that $\psi_{n-1} = 1$ if and only if $|S'_i| = k + 1$ for some $i \in [n-1]$. Thus, it just remains to prove that $G[\gamma']$ is a directed path. We prove in the following claims that $G[\gamma']$ satisfies each of Conditions (DP1)–(DP4), respectively.

Claim 2. *$G[\gamma']$ contains exactly one source vertex.*

Proof of claim. Since $\mathfrak{u}_{n-1}$ is final, either $\phi_{n-1} = 0$ or $\phi_{n-1} = 1$.

Suppose that $\phi_{n-1} = 1$. By the Vertex degree rules, $\phi_i \leq \phi_{i+1}$ for each $i \in [n-2]$. Hence, there exists $\iota \in [n-1]$ such that $\phi_i = 0$ for each $i \in [\iota - 1]$ and $\phi_j = 1$ for each $j \in \{\iota, \dots, n-1\}$. Since $\mathfrak{u}_1$ is initial, if $\iota = 1$, then u_ι has no in-edge and one out-edge in S'_1. On the other hand, if $\iota > 1$, then, by the Vertex degree rules, u_ι has no in-edge and one out-edge in $S'_{\iota-1} \cup S'_\iota$. Consequently, u_ι is a source vertex of $G[\gamma']$. Moreover, one can verify that, for any $i \in [n] \setminus \{\iota\}$, there is no vertex u_i that has in-degree 0 and out-degree 1 in $G[\gamma']$, otherwise $\phi_{n-1} = 2$. Therefore u_n is the only source vertex of $G[\gamma']$.

Now, suppose that $\phi_{n-1} = 0$. Then, u_n has no in-edge and one out-edge in S'_{n-1}, otherwise $\mathfrak{u}_{n-1}$ would not be final. Thus, u_n is a source vertex of $G[\gamma']$. In addition, note that $\phi_i = 0$ for each $i \in [n-1]$, otherwise $\phi_{n-1} \neq 0$. As a result, we obtain that, for any $i \in [n-1]$, there is no vertex u_i that has in-degree 0 and out-degree 1 in $G[\gamma']$. Therefore u_n is the only source vertex of $G[\gamma']$. ■

Claim 3. *$G[\gamma']$ contains exactly one target vertex.*

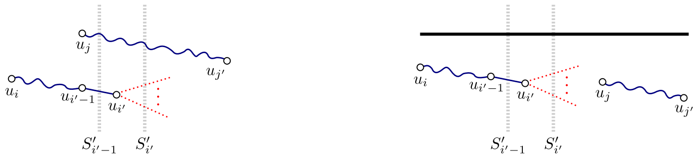
Proof of claim. The proof of this claim is analogous to the proof of Claim 2, following from the fact that $\varphi_i \leq \varphi_{i+1}$ for each $i \in [n-2]$, and from the fact that either $\varphi_{n-1} = 0$ or $\varphi_{n-1} = 1$, since $\mathfrak{u}_{n-1}$ is a final tuple. ■

Claim 4. *Let s and t be the source and target vertices of $G[\gamma']$, respectively, and let $u \in V(G[\gamma']) \setminus \{s, t\}$. Then, u has in-degree* 1 *and out-degree* 1 *in $G[\gamma']$.*

Proof of claim. By hypothesis $\mathfrak{u}_1$ is initial, $\mathfrak{u}_{n-1}$ is final and $\mathfrak{u}_i \rightsquigarrow \mathfrak{u}_{i+1}$ for each $i \in [n-2]$. This implies that every vertex of $G[\gamma']$ has in-degree at most 1 and out-degree at most 1. Moreover, it follows from the uniqueness of s and from the uniqueness of t that the vertex u is neither a source vertex nor a target vertex of $G[\gamma']$. Therefore, since $G[\gamma']$ does not contain isolated vertices, we obtain that u necessarily has in-degree 1 and out-degree 1 in $G[\gamma']$. ■

Claim 5. *$G[\gamma']$ is a weakly connected graph.*

Proof of claim. For the sake of contradiction, suppose that $G[\gamma']$ is not weakly connected. Then, there exist two distinct vertices $u_i, u_j \in V(G[\gamma'])$ such that there is no undirected path between them in $G[\gamma']$. Let $u_{i'}$ be a vertex of $G[\gamma']$ such that there exists in $G[\gamma']$ an undirected path between u_i and $u_{i'}$, for some $i' \in \{1, \dots, n\} \setminus \{i\}$. And, let $u_{j'}$ be a vertex of $G[\gamma']$ such that there exists in $G[\gamma']$ an undirected path between u_j and $u_{j'}$, for some $j' \in \{1, \dots, n\} \setminus \{j\}$. Note that, such vertices $u_{i'}$ and $u_{j'}$ necessarily exist, since by definition $G[\gamma']$ does not contain any isolated vertices. Assume without loss of generality that $i' > i$ and $j' > j$. Additionally, assume that i' is the greatest integer belonging to $\{i+1, \dots, n\}$ that holds the property of existing in $G[\gamma']$ an undirected path between u_i and $u_{i'}$. Analogously, assume that j' is the greatest



(a) Case 1 (b) Case 2

Fig. 5. Proof that $G[\gamma']$ is weakly connected. (Red) dotted lines represent non-edges, (black) thicker lines represent possibly existing edges, (blue) waved lines represent undirected paths, and (blue) normal style lines represent mandatory edges. In these illustrations, we assume that $\iota = i'$ (the case $\iota = j'$ is symmetric).

integer belonging to $\{j+1,\dots,n\}$ that holds the property of existing in $G[\gamma']$ an undirected path between u_j and $u_{j'}$. Let $\iota = \min\left\{i', j'\right\}$. By the maximalities of i' and j', there is no edge in S'_ι that has u_ι as an endpoint, *i.e.*

$$\left\{(x,y) \in S'_\iota : x = u_\iota \text{ or } y = u_\iota\right\} = \emptyset. \tag{1}$$

Moreover, one can readily verify that $S'_{\iota-1} \setminus S'_\iota \neq \emptyset$. We split the remainder of this proof into two cases.

Case 1. Suppose that $\left\{i,\dots,i'\right\} \cap \left\{j,\dots,j'\right\} \neq \emptyset$ (see Fig. 5(a)). Note that, necessarily $S'_\iota \neq \emptyset$. Then, it follows from Eq. (1) and from Claim 1 that there is no part $Q \in \mathcal{S}_{\iota-1}$ such that $Q \cap (S'_{\iota-1} \setminus S'_\iota) \neq \emptyset$ and $Q \cap S'_\iota \neq \emptyset$, otherwise there would exist in $G[\gamma']$ an undirected path between u_i and u_j. Thus, $\mathcal{Q}_\iota = \emptyset$. Therefore, since $S'_{\iota-1} \setminus S'_\iota \neq \emptyset$ and $S'_\iota \neq \emptyset$, we obtain by Connectedness rule 3 that $\mathfrak{u}_{\iota-1}$ is not compatible with $\mathfrak{u}_\iota$.

Case 2. Suppose that $\left\{i,\dots,i'\right\} \cap \left\{j,\dots,j'\right\} = \emptyset$ (see Fig. 5(b)). It follows from Eq. (1) and from Claim 1 that there is no part $Q \in \mathcal{S}_{\iota-1}$ such that $Q \cap (S'_{\iota-1} \setminus S'_\iota) \neq \emptyset$ and $Q \cap S'_\iota \neq \emptyset$, otherwise i' or j' would not be maximum with respect to the aforementioned properties. Thus, $\mathcal{Q}_\iota = \emptyset$. However, possibly $S'_\iota = \emptyset$. First, suppose that $S'_\iota \neq \emptyset$. Then, as in the previous case, it follows from the fact that $S'_{\iota-1} \setminus S'_\iota \neq \emptyset$ and from the Connectedness rules that the tuple $\mathfrak{u}_{\iota-1}$ is not compatible with the tuple $\mathfrak{u}_\iota$. On the other hand, suppose that $S'_\iota = \emptyset$. Then, $\tau_\iota \geq 1$. As a result, $\tau_l = \tau_{l-1} \geq 1$ for each $l \in \left\{\iota+1,\dots,\max\left\{i',j'\right\}-1\right\}$, and $\tau_l = 2$ for each $l \in \left\{\max\left\{i',j'\right\},\dots,n-1\right\}$. In particular, we obtain that:

1. either $n > \max\left\{i',j'\right\}$, and then $\tau_{n-1} = 2$;
2. or $n = \max\left\{i',j'\right\}$, and then $S'_{n-1} \neq \emptyset$ and $\tau_{n-1} = 1$.

In either case, $\mathfrak{u}_{n-1}$ is not a final tuple. Therefore, $G[\gamma']$ is weakly connected. ■

By the previous claims, we obtain that $G[\gamma']$ is indeed a directed path, and thereby we conclude the proof of Lemma 2.

4. NP-hardness

In this section, we prove that 2-ZIG-ZAG NUMBER is an NP-hard problem. For that, we present a polynomial-time reduction from POSITIVE NOT ALL EQUAL 3SAT, which is a well-known NP-complete problem [17], defined next.

POSITIVE NOT ALL EQUAL 3SAT (PNAE 3SAT)

Input: Set X of variables and a collection $\mathcal{C}$ of clauses over X such that each clause has no negative literal and exactly three positive literals.

Question: Is there a truth assignment $\alpha : X \to \{0,1\}$ such that each clause in $\mathcal{C}$ has at least one true literal and at least one false literal under α?

Construction 1. *Let $I = (X, \mathcal{C})$ be an instance of* PNAE 3SAT *with variable set X and clause set $\mathcal{C}$. We let G_I be the directed graph obtained from I as follows.*

- *For each variable $x_i \in X$, add the vertices u_i^1, u_i^2 and u_i^3, and add the edges $\left(u_i^1, u_i^2\right)$, $\left(u_i^2, u_i^3\right)$ and $\left(u_i^3, u_i^1\right)$.*
- *For each clause $C_j \in \mathcal{C}$, add the vertices v_j^1, v_j^2 and v_j^3, and add the edges $\left(v_j^1, v_j^2\right)$, $\left(v_j^2, v_j^3\right)$ and $\left(v_j^3, v_j^1\right)$. Moreover, assuming $C_j = \left\{x_{l_1}, x_{l_2}, x_{l_3}\right\}$ with $l_1 < l_2 < l_3$, add the edges $\left(u_{l_1}^1, v_j^1\right)$, $\left(u_{l_1}^3, v_j^1\right)$, $\left(u_{l_2}^1, v_j^2\right)$, $\left(u_{l_2}^3, v_j^2\right)$, $\left(u_{l_3}^1, v_j^3\right)$ and $\left(u_{l_3}^3, v_j^3\right)$.*

For each variable $x_i \in X$, we let H_i denote the subgraph of G_I induced by the vertices in $\left\{u_i^1, u_i^2, u_i^3\right\}$. And, for each clause $C_j \in \mathcal{C}$, we let $\widetilde{H}_j$ denote the subgraph of G_I induced by the vertices in $\left\{v_j^1, v_j^2, v_j^3\right\}$. We remark that H_i and $\widetilde{H}_j$ are directed cycles of length 3.

Fig. 6 exemplifies the directed graph G_I, described in Construction 1.

We establish in Lemmas 4 and 6 that there exists a satisfying truth assignment for an instance I of PNAE 3SAT if and only if there exists a linear order of zig-zag number at most 2 for the vertices of G_I. The central idea of our proof

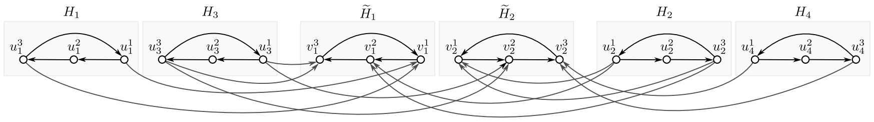
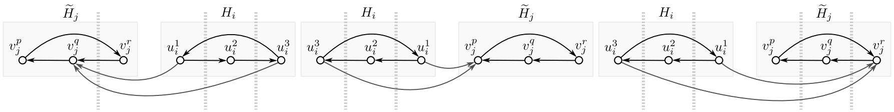


Fig. 6. Directed graph G_I obtained from the instance $I = (X, \mathcal{C})$ of PNAE 3SAT where $X = \{x_1, x_2, x_3, x_4\}$ and $\mathcal{C} = \{C_1 = \{x_1, x_2, x_3\}, C_2 = \{x_2, x_3, x_4\}\}$.



(a) $i = l_q$ (b) $i = l_p$ (c) $i = l_r$

Fig. 7. Case in which the clause $C_j = \left\{x_{l_1}, x_{l_2}, x_{l_3}\right\}$ has exactly one true literal under the truth assignment α, say x_{l_q} for some $q \in \{1, 2, 3\}$.

is to explore the possible internal relative orderings of the vertices of each directed cycle of G_I and, for each clause $C_j = \left\{x_{l_1}, x_{l_2}, x_{l_3}\right\} \in \mathcal{C}$, the possible ordered relative placements among the subgraphs H_{l_1}, H_{l_2}, H_{l_3}, and $\widetilde{H}_j$.

Lemma 4. *Let $I = (X, \mathcal{C})$ be an instance of* PNAE 3SAT. *If I is a yes instance of* PNAE 3SAT, *then* $zn(G_I) \leq 2$.

Proof. Let $\alpha : X \to \{0, 1\}$ be a truth assignment such that each clause in $\mathcal{C}$ has at least one true literal and at least one false literal under α. In what follows, we define from α a linear order $<_\pi$ of the vertices of G_I such that $\mathrm{zn}(G_I, \pi) \leq 2$.

Throughout this proof, consider $X = \left\{x_1, \ldots, x_{|X|}\right\}$ and $\mathcal{C} = \left\{C_1, \ldots, C_{|\mathcal{C}|}\right\}$.

For each variable $x_i \in X$, set

$$\begin{cases} u_i^1 <_\pi u_i^2 <_\pi u_i^3 & \text{if } \alpha(x_i) = 1 \\ u_i^1 >_\pi u_i^2 >_\pi u_i^3 & \text{otherwise.} \end{cases}$$

Let $V_0' \doteq \left\{u_i^1, u_i^2, u_i^3 : \alpha(x_i) = 0\right\}$ and $V_1' \doteq \left\{u_i^1, u_i^2, u_i^3 : \alpha(x_i) = 1\right\}$. Then, for each $y \in V_0'$ and each $z \in V(G_I) \setminus V_0'$, set $y <_\pi z$. And, for each $y \in V_1'$ and each $z \in V(G_I) \setminus V_1'$, set $y >_\pi z$.

Let C_j be a clause in $\mathcal{C}$. Assume that $C_j = \left\{x_{l_1}, x_{l_2}, x_{l_3}\right\}$ with $l_1 < l_2 < l_3$. There are two cases to be considered. First, suppose that C_j has exactly one true literal under α, say l_q for some $q \in \{1, 2, 3\}$. Then, set

$$v_j^p <_\pi v_j^q <_\pi v_j^r,$$

where $p = q \bmod 3 + 1$ and $r = (q + 1) \bmod 3 + 1$. Now, suppose that C_j has exactly two true literals under α. Thus, C_j has exactly one false literal under α, say l_q for some $q \in \{1, 2, 3\}$. Then, set

$$v_j^p >_\pi v_j^q >_\pi v_j^r,$$

where $p = q \bmod 3 + 1$ and $r = (q + 1) \bmod 3 + 1$.

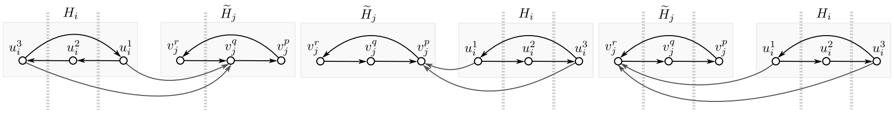
Finally, for each pair of distinct variables $x_i, x_{i'} \in X$ with $i < i'$, such that $\alpha(x_i) = \alpha(x_{i'})$, set $u_i^p <_\pi u_{i'}^q$ for each $p, q \in \{1, 2, 3\}$. And, for each pair of distinct clauses $C_j, C_{j'} \in \mathcal{C}$ with $j < j'$, set $v_j^p <_\pi v_{j'}^q$ for each $p, q \in \{1, 2, 3\}$.

One can readily verify that $<_\pi$ is indeed a linear order of the vertices of G_I.

Now, we prove that $\mathrm{zn}(G_I, \pi) \leq 2$. For the sake of contradiction, suppose that there exists a directed path P in G_I such that $\mathrm{zn}(G_I, \pi, P) \geq 3$. Assume without loss of generality that P is a minimal path with respect to the property that $\mathrm{zn}(G_I, \pi, P) \geq 3$. Recall that, for each variable $x_i \in X$, H_i is a directed cycle of length 3. Similarly, for each clause $C_j \in \mathcal{C}$, $\widetilde{H}_j$ is a directed cycle of length 3. Consequently, P is neither a subgraph of H_i nor a subgraph of $\widetilde{H}_j$, for any $x_i \in X$ and any $C_j \in \mathcal{C}$, otherwise $\mathrm{zn}(G_I, \pi, P) < 3$. Moreover, every edge of G_I is either an edge of one of these subgraphs H_i and $\widetilde{H}_j$ or is an edge from a vertex of H_i to a vertex of $\widetilde{H}_j$, for some $x_i \in X$ and some $C_j \in \mathcal{C}$. As a result, there exists precisely one variable $x_i \in X$ and there exists precisely one clause $C_j \in \mathcal{C}$ such that $V(P) \cap V(H_i) \neq \emptyset$ and $V(P) \cap V(\widetilde{H}_j) \neq \emptyset$. More specifically, P consists in a directed path on at most 4 vertices (by its minimality) from a vertex of H_i to a vertex of $\widetilde{H}_j$ that only contains vertices belonging to $V(H_i) \cup V(\widetilde{H}_j)$. Assume that $C_j = \left\{x_{l_1}, x_{l_2}, x_{l_3}\right\}$ with $l_1 < l_2 < l_3$.

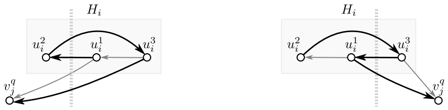
First, consider the case in which C_j has exactly one true literal under α, and let x_{l_q} be such a literal for some $q \in \{1, 2, 3\}$. Thus, $v_j^p <_\pi v_j^q <_\pi v_j^r$, where $p = q \bmod 3 + 1$ and $r = (q + 1) \bmod 3 + 1$. Consequently, if $i = l_q$, then $u_i^1 <_\pi u_i^2 <_\pi u_i^3$ and $V(H_i) >_\pi V(\widetilde{H}_j)$, which implies $\mathrm{zn}(G_I, \pi, P) < 3$ (see Fig. 7(a)). On the other hand, if $i = l_p$ or $i = l_r$, then $u_i^1 >_\pi u_i^2 >_\pi u_i^3$ and $V(H_i) <_\pi V(\widetilde{H}_j)$, which also implies $\mathrm{zn}(G_I, \pi, P) < 3$ (see Figs. 7(b) and 7(c)).

Now, consider the case in which C_j has exactly two true literals under α, and let l_q be the only false literal of C_j under α for some $q \in \{1, 2, 3\}$. Thus, $v_j^p >_\pi v_j^q >_\pi v_j^r$, where $p = q \bmod 3 + 1$ and $r = (q + 1) \bmod 3 + 1$. If $i = l_q$, then $u_i^1 >_\pi u_i^2 >_\pi u_i^3$ and $V(H_i) <_\pi V(\widetilde{H}_j)$, which implies $\mathrm{zn}(G_I, \pi, P) < 3$ (see Fig. 8(a)). On the other hand, if $i = l_p$ or $i = l_r$, then $u_i^1 <_\pi u_i^2 <_\pi u_i^3$ and $V(H_i) >_\pi V(\widetilde{H}_j)$, which also implies $\mathrm{zn}(G_I, \pi, P) < 3$ (see Figs. 8(b) and 8(c)).



(a) $i = l_q$ (b) $i = l_p$ (c) $i = l_r$

Fig. 8. Case in which the clause $C_j = \{x_{l_1}, x_{l_2}, x_{l_3}\}$ has exactly one false literal under the truth assignment α, say x_{l_q} for some $q \in \{1, 2, 3\}$.



(a) (b)

Fig. 9. Case 1: $u_i^2 <_\pi u_i^1 <_\pi u_i^3$. (a) $v_j^q <_\pi u_i^1$. (b) $v_j^q >_\pi u_i^1$.

Therefore, such a path P does not exist in G_I, and consequently we obtain that $\mathsf{zn}(G_I) \le \mathsf{zn}(G_I, \pi) \le 2$. □

Lemma 5. *Let $I = (X, \mathcal{C})$ be an instance of* PNAE 3SAT, *$\pi : V(G_I) \to [|G_I|]$ be a bijection such that $zn(G_I, \pi) \le 2$, and let $x_i \in X$. If $C_j \in \mathcal{C}$ is a clause containing x_i as a literal, then either $V(H_i) <_\pi V(\widetilde{H}_j)$ or $V(H_i) >_\pi V(\widetilde{H}_j)$. Furthermore, if there exists a clause $C_j \in \mathcal{C}$ containing x_i as a literal such that $V(H_i) <_\pi V(\widetilde{H}_j)$, then $V(H_i) <_\pi V(\widetilde{H}_{j'})$ for every other clause $C_{j'} \in \mathcal{C}$ containing x_i as a literal.*

Proof. Let $C_j \in \mathcal{C}$ be a clause containing x_i as a literal. For the sake of contradiction, suppose that neither $V(H_i) <_\pi V(\widetilde{H}_j)$ nor $V(H_i) >_\pi V(\widetilde{H}_j)$. Then, either there exist two vertices $v_j^p, v_j^{p'} \in V(\widetilde{H}_j)$ such that

$$\left\{v_j^p\right\} <_\pi V(H_i) <_\pi \left\{v_j^{p'}\right\}, \tag{2}$$

or there exists a vertex $v_j^p \in V(\widetilde{H}_j)$ such that, for some pair $a, b \in \{1, 2, 3\}$ with $a \neq b$,

$$u_i^a <_\pi v_j^p <_\pi u_i^b. \tag{3}$$

Hence, based on inequalities (2) and (3), one can verify that in either case there exist two (not necessarily distinct) vertices $v_j^p, v_j^{p'} \in V(\widetilde{H}_j)$ such that

$$v_j^p <_\pi \max{}_\pi V(H_i) \;\text{ and }\; v_j^{p'} >_\pi \min{}_\pi V(H_i) \tag{4}$$

In particular, we note that, $v_j^p = v_j^{p'}$ if and only if the second case – the one described by inequality (3) – holds.

Additionally, since by hypothesis x_i is a literal of C_j, there exists a vertex $v_j^q \in V(\widetilde{H}_j)$ such that $\left(u_i^1, v_j^q\right), \left(u_i^3, v_j^q\right) \in E(G_I)$. It is worth mentioning that, possibly, $v_j^p \neq v_j^q$ and $v_j^{p'} \neq v_j^q$. If $v_j^p \neq v_j^q$, then there exists a directed path $P_1' = \langle v_j^q, \ldots, v_j^p \rangle$ in $\widetilde{H}_j$ from v_j^q to v_j^p that has at least one edge. Analogously, if $v_j^{p'} \neq v_j^q$, then there exists a directed path $P_2' = \langle v_j^q, \ldots, v_j^{p'} \rangle$ in $\widetilde{H}_j$ from v_j^q to $v_j^{p'}$ that has at least one edge.

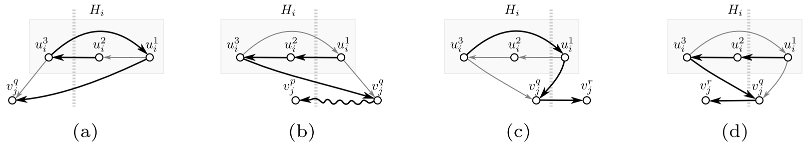
We split the remainder of this proof into three main cases.

(Case 1). Suppose that $u_i^2 <_\pi u_i^1 <_\pi u_i^3$. If $v_j^q <_\pi u_i^1$, then $P = \langle u_i^1, u_i^2, u_i^3, v_j^q \rangle$ is a directed path of G_I such that $\mathsf{zn}(G_I, \pi, P) = 3$ (see Fig. 9(a)). Similarly, if $v_j^q >_\pi u_i^1$, then $P = \langle u_i^2, u_i^3, u_i^1, v_j^q \rangle$ is a directed path of G_I such that $\mathsf{zn}(G_I, \pi, P) = 3$ (see Fig. 9(b)).

For the remainder cases, let $v_j^r \in V(\widetilde{H}_j)$ such that $\left(v_j^q, v_j^r\right) \in E(G_I)$. Note that, by construction, such a vertex v_j^r exists and, besides that, is well-defined.

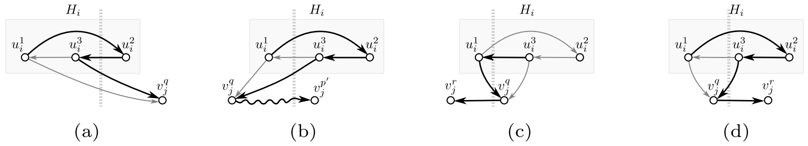
(Case 2). Suppose that $u_i^3 <_\pi u_i^2 <_\pi u_i^1$. If $v_j^q <_\pi u_i^2$, then $P = \langle u_i^2, u_i^3, u_i^1, v_j^q \rangle$ is a directed path of G_I such that $\mathsf{zn}(G_I, \pi, P) = 3$ (see Fig. 10(a)). Assume that $v_j^q >_\pi u_i^2$. If $v_j^q >_\pi u_i^1$, then we obtain from (4) that $v_j^p \neq v_j^q$. Consequently, the path $P = \langle u_i^1, u_i^2, u_i^3, v_j^q, \ldots, v_j^p \rangle$ — obtained by concatenating $P'' = \langle u_i^1, u_i^2, u_i^3, v_j^q \rangle$ with P_1' — is a directed path of G_I such that $\mathsf{zn}(G_I, \pi, P) \ge 3$ (see Fig. 10(b)). Assume that $v_j^q <_\pi u_i^1$. If $v_j^r >_\pi v_j^q$, then $P = \langle u_i^3, u_i^1, v_j^q, v_j^r \rangle$ is a directed path of G_I such that $\mathsf{zn}(G_I, \pi, P) = 3$ (see Fig. 10(c)). Otherwise, if $v_j^r <_\pi v_j^q$, then $P = \langle u_i^1, u_i^2, u_i^3, v_j^q, v_j^r \rangle$ is a directed path of G_I such that $\mathsf{zn}(G_I, \pi, P) = 3$ (see Fig. 10(d)).

(Case 3). Suppose that $u_i^1 <_\pi u_i^3 <_\pi u_i^2$. If $v_j^q >_\pi u_i^3$, then $P = \langle u_i^1, u_i^2, u_i^3, v_j^q \rangle$ is a directed path of G_I such that $\mathsf{zn}(G_I, \pi, P) = 3$ (see Fig. 11(a)). Assume that $v_j^q <_\pi u_i^3$. If $v_j^q <_\pi u_i^1$, then we obtain from (4) that $v_j^{p'} \neq v_j^q$. Consequently, the path $P = \langle u_i^1, u_i^2, u_i^3, v_j^q, \ldots, v_j^{p'} \rangle$ — obtained by concatenating $P'' = \langle u_i^1, u_i^2, u_i^3, v_j^q \rangle$ with P_2' — is a directed path of G_I such that $\mathsf{zn}(G_I, \pi, P) \ge 3$ (see Fig. 11(b)). Assume that $v_j^q >_\pi u_i^1$. If $v_j^r <_\pi v_j^q$, then $P = \langle u_i^3, u_i^1, v_j^q, v_j^r \rangle$ is a directed path



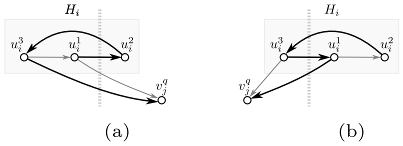
(a) (b) (c) (d)

Fig. 10. Case 2: $u_i^3 <_\pi u_i^2 <_\pi u_i^1$. (a) $v_j^q <_\pi u_i^2$. (b) $v_j^q >_\pi u_i^2$ and $v_j^q >_\pi u_i^1$. (c) $u_i^2 <_\pi v_j^q <_\pi u_i^1$ and $v_j^r >_\pi v_j^q$. (d) $u_i^2 <_\pi v_j^q <_\pi u_i^1$ and $v_j^r <_\pi v_j^q$.



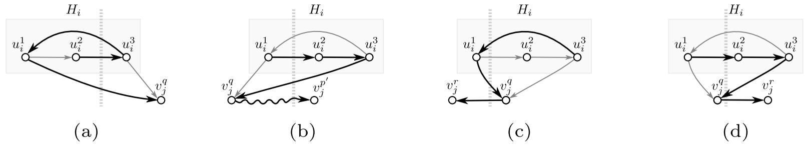
(a) (b) (c) (d)

Fig. 11. Case 3: $u_i^1 <_\pi u_i^3 <_\pi u_i^2$. (a) $v_j^q >_\pi u_i^3$. (b) $v_j^q <_\pi u_i^3$ and $v_j^q <_\pi u_i^1$. (c) $u_i^1 <_\pi v_j^q <_\pi u_i^3$ and $v_j^r <_\pi v_j^q$. (d) $u_i^1 <_\pi v_j^q <_\pi u_i^3$ and $v_j^r >_\pi v_j^q$.



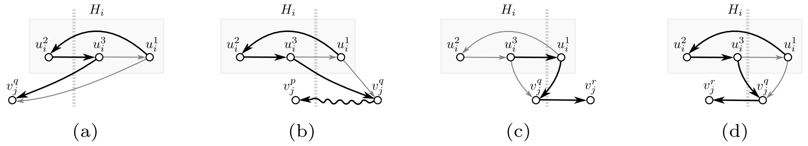
(a) (b)

Fig. 12. Case in which $u_i^3 <_\pi u_i^1 <_\pi u_i^2$. (a) $v_j^q >_\pi u_i^1$. (b) $v_j^q <_\pi u_i^1$.



(a) (b) (c) (d)

Fig. 13. Case in which $u_i^1 <_\pi u_i^2 <_\pi u_i^3$. (a) $v_j^q >_\pi u_i^2$. (b) $v_j^q <_\pi u_i^2$ and $v_j^q <_\pi u_i^1$. (c) $u_i^1 <_\pi v_j^q <_\pi u_i^2$ and $v_j^r <_\pi v_j^q$. (d) $u_i^1 <_\pi v_j^q <_\pi u_i^2$ and $v_j^r >_\pi v_j^q$.



(a) (b) (c) (d)

Fig. 14. Case in which $u_i^2 <_\pi u_i^3 <_\pi u_i^1$. (a) $v_j^q <_\pi u_i^3$. (b) $v_j^q >_\pi u_i^3$ and $v_j^q >_\pi u_i^1$. (c) $u_i^3 <_\pi v_j^q <_\pi u_i^1$ and $v_j^r >_\pi v_j^q$. (d) $u_i^3 <_\pi v_j^q <_\pi u_i^1$ and $v_j^r <_\pi v_j^q$.

of G_I such that $\mathsf{zn}(G_I, \pi, P) = 3$ (see Fig. 11(c)). Otherwise, if $v_j^r >_\pi v_j^q$, then $P = \langle u_i^1, u_i^2, u_i^3, v_j^q, v_j^r \rangle$ is a directed path of G_I such that $\mathsf{zn}(G_I, \pi, P) = 3$ (see Fig. 11(d)).

One can readily verify that the case in which $u_i^3 <_\pi u_i^1 <_\pi u_i^2$ is symmetric to Case 1 (see Fig. 12), the case in which $u_i^1 <_\pi u_i^2 <_\pi u_i^3$ is symmetric to Case 2 (see Fig. 13), and the case in which $u_i^2 <_\pi u_i^3 <_\pi u_i^1$ is symmetric to Case 3 (see Fig. 14). Additionally, note that, regardless of the existence of the vertex v_j^p, Case 1 and consequently the case in which $u_i^3 <_\pi u_i^1 <_\pi u_i^2$ do not consist in valid configurations, otherwise $\mathsf{zn}(G, \pi) \geq 3$ even if $V(H_i) <_\pi V(\widetilde{H}_j)$ or $V(H_i) >_\pi V(\widetilde{H}_j)$.

Thus, $V(H_i) <_\pi V(\widetilde{H}_j)$ or $V(H_i) >_\pi V(\widetilde{H}_j)$, otherwise $\mathsf{zn}(G, \pi) \geq 3$. Particularly, one can further verify that if $u_i^3 <_\pi u_i^2 <_\pi u_i^1$ or $u_i^2 <_\pi u_i^3 <_\pi u_i^1$, then necessarily $V(H_i) <_\pi V(\widetilde{H}_j)$. Analogously, we have that if $u_i^1 <_\pi u_i^3 <_\pi u_i^2$ or $u_i^1 <_\pi u_i^2 <_\pi u_i^3$, then $V(H_i) >_\pi V(\widetilde{H}_j)$. Therefore, if $V(H_i) <_\pi V(\widetilde{H}_j)$, then $V(H_i) <_\pi V(\widetilde{H}_{j'})$ for every other clause $C_{j'} \in \mathcal{C}$ containing x_i as a literal. □

Lemma 6. *Let $I = (X, \mathcal{C})$ be an instance of* PNAE 3SAT. *If $zn(G_I) \leq 2$, then I is a* yes *instance of* PNAE 3SAT.

Proof. Let $\pi : V(G_I) \to [|G_I|]$ be a bijection such that $\mathsf{zn}(G_I, \pi) \leq 2$. It follows from Lemma 5 that, for each variable $x_i \in X$ and each clause $C_j \in \mathcal{C}$, if $V(H_i) >_\pi V(\widetilde{H}_j)$, then $V(H_i) >_\pi V(\widetilde{H}_{j'})$ for each clause $C_{j'} \in \mathcal{C}$ containing x_i as a literal.

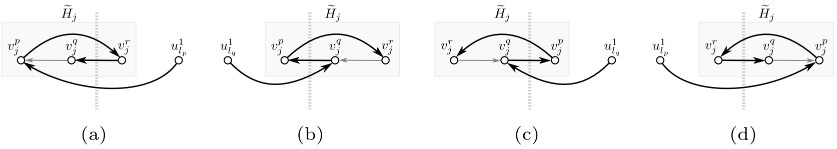


Fig. 15. (a) and (c) $\alpha(x_{l_1}) = \alpha(x_{l_2}) = \alpha(x_{l_3}) = 1$. (b) and (d) $\alpha(x_{l_1}) = \alpha(x_{l_2}) = \alpha(x_{l_3}) = 0$. (a) and (b) $v_j^p <_\pi v_j^q <_\pi v_j^r$. (c) and (d) $v_j^p >_\pi v_j^q >_\pi v_j^r$.

Thus, we let $\alpha : x \to \{0, 1\}$ be the truth assignment defined as follows: for each variable $x_i \in X$, $\alpha(x_i) = 1$ if and only if $V(H_i) >_\pi V(\widetilde{H}_j)$ for each clause $C_j \in \mathcal{C}$.

Now, we prove that each clause in $\mathcal{C}$ has at least one true literal and at least one false literal under α. For the sake of contradiction, suppose that there exists a clause $C_j = \left\{x_{l_1}, x_{l_2}, x_{l_3}\right\}$ in $\mathcal{C}$ such that $\alpha(x_{l_1}) = \alpha(x_{l_2}) = \alpha(x_{l_3})$. Let $q \in \{1, 2, 3\}$, $p = q \bmod 3 + 1$ and $r = (q + 1) \bmod 3 + 1$.

Suppose that $\alpha(x_{l_1}) = \alpha(x_{l_2}) = \alpha(x_{l_3}) = 1$. Thus, $\left\{u_{l_1}^1, u_{l_2}^1, u_{l_3}^1\right\} >_\pi V(\widetilde{H}_j)$. Consequently, if $v_j^p <_\pi v_j^q <_\pi v_j^r$, then $P = \langle u_{l_p}^1, v_j^p, v_j^r, v_j^q \rangle$ is a directed path of G_I such that $\mathsf{zn}(G_I, \pi, P) = 3$ (see Figs. 15(a)); on the other hand, if $v_j^p >_\pi v_j^q >_\pi v_j^r$, then $P = \langle u_{l_q}^1, v_j^q, v_j^p, v_j^r \rangle$ is a directed path of G_I such that $\mathsf{zn}(G_I, \pi, P) = 3$ (see Figs. 15(c)).

Suppose that $\alpha(x_{l_1}) = \alpha(x_{l_2}) = \alpha(x_{l_3}) = 0$. Thus, $\left\{u_{l_1}^1, u_{l_2}^1, u_{l_3}^1\right\} <_\pi V(\widetilde{H}_j)$. Consequently, if $v_j^p <_\pi v_j^q <_\pi v_j^r$, then $P = \langle u_{l_q}^1, v_j^q, v_j^p, v_j^r \rangle$ is a directed path of G_I such that $\mathsf{zn}(G_I, \pi, P) = 3$ (see Figs. 15(b)); on the other hand, if $v_j^p >_\pi v_j^q >_\pi v_j^r$, then $P = \langle u_{l_p}^1, v_j^p, v_j^r, v_j^q \rangle$ is a directed path of G_I such that $\mathsf{zn}(G_I, \pi, P) = 3$ (see Figs. 15(d)).

Therefore, each clause in $\mathcal{C}$ has at least one true literal and at least one false literal under α, and consequently I is a yes instance of PNAE 3SAT. □

Theorem 2. 2-zig-zag Number *is NP-complete.*

Proof. By Theorem 1, 2-zig-zag Number is in NP. It follows from Lemmas 4 and 6 that I is a yes instance of PNAE 3SAT if and only if $\mathsf{zn}(G_I) \leq 2$. Therefore, since G_I can be constructed in time polynomial in $|I|$, 2-zig-zag Number is NP-complete. □

5. Zig-zag number and directed treewidth

It was proved in [13] that directed graphs of constant directed pathwidth have constant zig-zag number, and that there exist directed graphs of constant zig-zag number but unbounded directed pathwidth. Hence, the family of directed graphs of constant zig-zag number properly contains the family of directed graphs of constant directed pathwidth.

Nevertheless, it is unknown whether or not a similar result would hold with respect to zig-zag number and directed treewidth. In this section, we prove that there exist directed graphs of constant directed treewidth but unbounded zig-zag number. More specifically, we prove the following theorem.

Theorem 3. *There exist directed graphs on n vertices of constant directed treewidth but zig-zag number $\Omega(\log n)$.*

We remark that, although it was shown in [2] that there are directed graphs of constant directed treewidth but unbounded directed pathwidth, this result cannot be directly used to conclude the respective statement relating directed treewidth and zig-zag number.

Another interesting aspect of our result follows from the fact that directed graphs of constant directed treewidth have constant tree-zig-zag number [14]. Consequently, there are directed graphs of constant tree-zig-zag number but unbounded zig-zag number. Therefore, considering the fact that directed graphs of constant zig-zag number have constant tree-zig-zag number [14], we obtain that the family of directed graphs of constant tree-zig-zag number is strictly richer than the family of directed graphs of constant zig-zag number.

5.1. Basic definitions

A directed graph G is called *bidirected* if, for each two distinct vertices $u, v \in V(G)$, $(u, v) \in E(G)$ if and only if $(v, u) \in E(G)$. Note that, bidirected graphs may be regarded as undirected graphs. Based on that, we say that a pair of edges (u, v) and (u, v) of a bidirected graph G is a *bidirected edge* between u and v. A directed graph H is called an *undirected minor* of a bidirected graph G if H can be obtained from G by deleting vertices and edges, and by contracting bidirected edges.

An *undirected tree decomposition* of a directed graph G is a pair $\left(T, \{X_t\}_{t \in V(T)}\right)$ satisfying the following conditions:

1. T is a undirected tree;
2. $\bigcup_{t \in V(T)} X_t = V(G)$;

3. for each edge $(u, v) \in E(G)$, there exists a node $t \in V(T)$ such that $\{u, v\} \subseteq X_t$;
4. for each vertex $u \in V(G)$, the graph $T[\{t \in V(T): u \in X_t\}]$ is connected.

In particular, $\big(T, \{X_t\}_{t\in V(T)}\big)$ is called an *undirected path decomposition* of G if T is an undirected path. The *width* of an undirected tree decomposition $\big(T, \{X_t\}_{t\in V(T)}\big)$ is defined as the integer $\max_{t\in V(T)}|X_t| - 1$. The *undirected treewidth* of a directed graph G is defined as the minimum width over all tree decompositions of G, and the *undirected pathwidth* of G is defined as the minimum width over all path decompositions of G.

Throughout this section we are mainly concerned with bidirected graphs. Thus, based on Lemmas 7 and 8, stated next, it suffices to define only the notions of undirected pathwidth and of undirected treewidth. For the definitions of directed pathwidth and directed treewidth, we refer to Refs. [12,15,18].

Lemma 7 (*[1]*)**.** *If G is a bidirected graph, then the directed pathwidth of G is equal to its undirected pathwidth.*

Lemma 8 (*[12]*)**.** *If G is a bidirected graph, then the directed treewidth of G is equal to its undirected treewidth.*

For the sake of simplicity, we also omit the formal definition of tree-zig-zag number, and we refer to Ref. [14]. Informally, the tree-zig-zag number of a directed graph G is defined similarly to the zig-zag number of G except for, instead of linear orders, considering *binary arboreal orders*. That is to say, partial orders $<_\pi \subseteq V(G) \times V(G)$ such that, for each vertex $v \in V(G)$, the following conditions hold: the set $\{u \in V(G): u <_\pi v\}$ is linearly ordered by $<_\pi$ and there are at most two vertices $v' \in V(G)$ with $v <_\pi v'$ such that, for any $u \in V(G)$, $u <_\pi v'$ if and only if $u <_\pi v$.

5.2. Directed vertex separation number

Let G be a directed graph on n vertices and $\pi : V(G) \to [n]$ be a bijection. Assume that $V(G) = \{u_1, \dots, u_n\}$ and, for each $u_i, u_j \in V(G)$, $i < j$ if and only if $u_i <_\pi u_j$. The *directed vertex separation number of G with respect to π* is defined as the maximum number of vertices in $\{u_{i+1}, \dots, u_n\}$ that have some out-neighbor in $\{u_1, \dots, u_i\}$, where the maximum is taken over all $i \in [n-1]$. More formally,

$$\mathsf{dvsn}(G, \pi) \doteq \max_{i\in[n-1]} \left|\left\{v \in \{u_{i+1}, \dots, u_n\} : N^+_G(v) \cap \{u_1, \dots, u_i\} \neq \emptyset\right\}\right|,$$

where $N^+_G(v)$ denotes the out-neighborhood of v in G. The *directed vertex separation number of G*, denoted by $\mathsf{dvsn}(G)$, is defined as the minimum $\mathsf{dvsn}(G, \pi)$ over all bijections $\pi : V(G) \to [n]$.

Lemma 9 (*[19]*)**.** *Let G be a directed graph. The directed pathwidth of G is equal to the directed vertex separation number of G.*

Lemma 10. *If G_1 and G_2 are two directed graphs over a same vertex set X, and $\pi : X \to [|X|]$ is a bijection, then $dvsn(G_1 \cup G_2, \pi) \leq dvsn(G_1, \pi) + dvsn(G_2, \pi)$.*

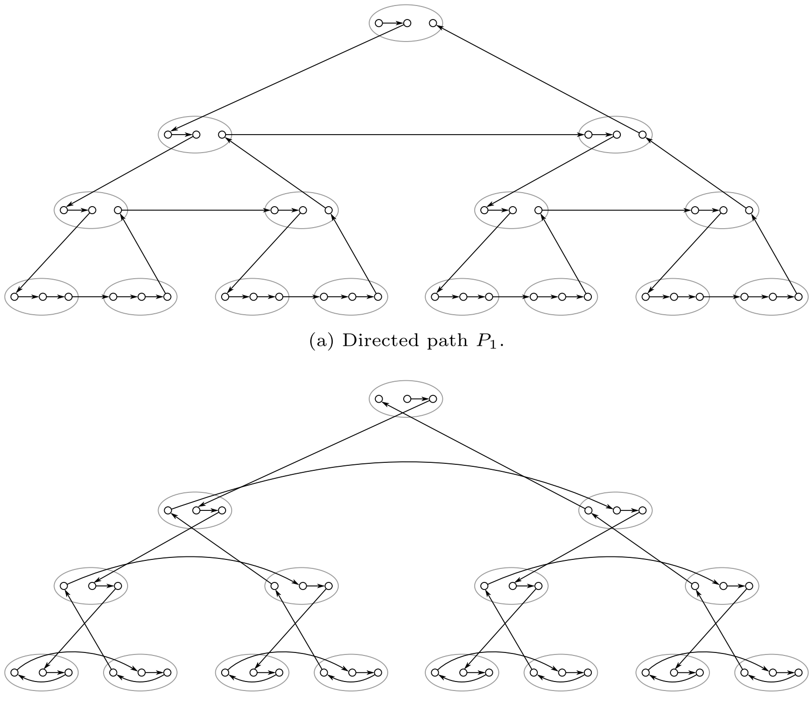
Proof. Assume that $X = \{u_1, \dots, u_n\}$ and, for each $u_i, u_j \in V(G)$, $i < j$ if and only if $u_i <_\pi u_j$. Let $i \in [n-1]$. Suppose that there exist ℓ_1 distinct vertices $v \in \{u_{i+1}, \dots, u_n\}$ such that $N^+_{G_1}(v) \cap \{u_1, \dots, u_i\} \neq \emptyset$. Analogously, suppose that there exist ℓ_2 distinct vertices $v \in \{u_{i+1}, \dots, u_n\}$ such that $N^+_{G_2}(v) \cap \{u_1, \dots, u_i\} \neq \emptyset$. As a result, there exist at most $\ell_1 + \ell_2$ distinct vertices $v \in \{u_{i+1}, \dots, u_n\}$ such that $N^+_{G_1}(v) \cap \{u_1, \dots, u_i\} \neq \emptyset$ or $N^+_{G_2}(v) \cap \{u_1, \dots, u_i\} \neq \emptyset$. In other words, there exist at most $\ell_1 + \ell_2$ distinct vertices $v \in \{u_{i+1}, \dots, u_n\}$ such that $N^+_{G_1\cup G_2}(v) \cap \{u_1, \dots, u_i\} \neq \emptyset$. Therefore, $\mathsf{dvsn}(G_1 \cup G_2, \pi) \leq \mathsf{dvsn}(G_1, \pi) + \mathsf{dvsn}(G_2, \pi)$. □

Lemma 11. *Let G be a directed graph, P be a directed path of G, and let H be the directed graph such that $V(H) = V(G)$ and $E(H) = E(P)$. Then, for each bijection $\pi : V(G) \to [|G|]$, $dvsn(H, \pi) \leq zn(H, \pi) \leq zn(G, \pi)$.*

Proof. The second inequality follows from the fact that H is a subgraph of G. Now, we prove that the first inequality holds. Assume that $V(G) = \{u_1, \dots, u_n\}$ and, for each $u_i, u_j \in V(G)$, $i < j$ if and only if $u_i <_\pi u_j$. Let $i \in [n-1]$. Since the set of edges of H induces a directed path, it follows from the definition of zig-zag number that there exist at most $\mathsf{zn}(H, \pi)$ edges in the cut $S_H(\pi, i)$. As a result, there exist at most $\mathsf{zn}(H, \pi)$ vertices $v \in \{u_{i+1}, \dots, u_n\}$ such that $N^+_H(v) \cap \{u_1, \dots, u_i\} \neq \emptyset$. Therefore, $\mathsf{dvsn}(H, \pi) \leq \mathsf{zn}(H, \pi)$. □

Lemma 12. *Let p be a positive integer, G be a directed graph and $\pi : V(G) \to [|G|]$ be a bijection. If G can be described as the union of p directed paths, then $dvsn(G, \pi) \leq p \cdot zn(G, \pi)$.*

Proof. Suppose that there are p directed paths $P_1, \dots, P_p$ such that $G = \bigcup_{i\in[p]} P_i$. For each $i \in [p]$, let H_i be the directed graph with vertex set $V(H_i) = V(G)$ and edge set $E(H_i) = E(P_i)$. Note that, $G = \bigcup_{i\in[p]} H_i$. It follows from Lemma 11 that, for each $i \in [p]$, $\mathsf{dvsn}(H_i, \pi) \leq \mathsf{zn}(H_i, \pi)$. Therefore, by Lemma 10, $\mathsf{dvsn}(G, \pi) \leq p \cdot \mathsf{zn}(G, \pi)$. □



(a) Directed path P_1.

(b) Directed path P_2.

Fig. 16. Directed paths P_1 and P_2 obtained by Algorithm 1, respectively.

5.3. Proof of Theorem 3

Let B_n be a rooted oriented complete binary tree on n vertices. For each non-leaf vertex $u \in V(B_n)$, we write $\mathsf{left}(u)$ to denote the left child of u in B_n, and we write $\mathsf{right}(u)$ to denote the right child of u in B_n.

We let $\mathbf{B}_n$ be the bidirected graph with vertex set $V(\mathbf{B}_n) = V(B_n) \times \{0, 1, 2\}$ obtained by the union of two suitable directed paths P_1 and P_2, recursively defined in Algorithm 1, and their respective reverse directed paths P_1' and P_2'. More specifically, if r is the root of B_n, then P_1 and P_2 are defined as the directed paths returned by the function calls

- $\texttt{Construct-Path}(B_n, P = \langle\rangle, u = r, \texttt{idx} = 1)$ and
- $\texttt{Construct-Path}(B_n, P = \langle\rangle, u = r, \texttt{idx} = 2)$

of Algorithm 1, respectively, and P_1' is the reverse directed path of P_1 and P_2' is the reverse directed path of P_2. Therefore, $\mathbf{B}_n$ can be decomposed into four directed paths. Fig. 16 illustrates P_1 and P_2.

Algorithm 1: Construction of directed graph $\mathbf{B}_n$.

```
function Construct-Path($B_n, P, u, idx$)
    $a = (\text{idx} - 1) \bmod 3$; $b = \text{idx} \bmod 3$; $c = (\text{idx} + 1) \bmod 3$
    $P := P + \langle (u, a), (u, b) \rangle$          // concatenates $P$ with the sequence $\langle (u, a), (u, b) \rangle$
    if $u$ is not a leaf of $B_n$ then
        $P :=$ Construct-Path($B_n, P, \mathit{left}(u), idx$)
        $P :=$ Construct-Path($B_n, P, \mathit{right}(u), idx$)
    $P := P + \langle (u, c) \rangle$          // concatenates $P$ with the sequence $\langle (u, c) \rangle$
    return $P$
```

Lemma 13. *The complete binary tree B_n is an undirected minor of $\mathbf{B}_n$.*

Proof. First, we note that, for each $i \in \{1, 2\}$, if there exists a directed edge (u, v) in the directed path P_i, then there exists the directed edge (v, u) in the reverse directed path P_i' of P_i. This implies that, whenever (u, v) is a directed edge in P_i for some $i \in \{1, 2\}$, $\mathbf{B}_n$ contains a bidirected edge between the nodes u and v. As a result, in order to show the existence of a bidirected edge in $\mathbf{B}_n$ between nodes u and v, it is enough to show that either (u, v) or (v, u) is a directed edge in P_i for some $i \in \{1, 2\}$.

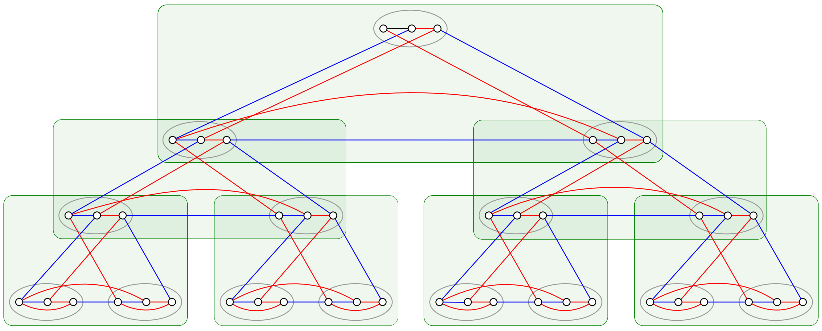


Fig. 17. Tree decomposition $\mathcal{T} = (T, \mathcal{X})$ of $\mathbf{B}_n$, where the rounded squares represent the bags of the nodes of T.

Based on that, we now remark that, for each node $t \in V(B_n)$, there exist in $\mathbf{B}_n$ a bidirected edge between the nodes $(t, 0)$ and $(t, 1)$, and a bidirected edge between the nodes $(t, 1)$ and $(t, 2)$. Indeed, this follows from the facts that $(t, 1)$ immediately succeeds $(t, 0)$ in the directed path P_1, and that $(t, 2)$ immediately succeeds $(t, 1)$ in the directed path P_2. In addition, it follows from construction of P_1 that, for each non-leaf node $t \in V(B_n)$, there exist in $\mathbf{B}_n$ an edge between the nodes $(t, 1)$ and $(\mathsf{left}(t), 0)$, and an edge between the nodes $(\mathsf{right}(t), 2)$ and $(t, 2)$. Similarly, it follows from construction of P_2 that, for each non-leaf node $t \in V(B_n)$, there exist in $\mathbf{B}_n$ an edge between the nodes $(t, 2)$ and $(\mathsf{left}(t), 1)$, and an edge between the nodes $(\mathsf{right}(t), 0)$ and $(t, 0)$.

Thus, let $\mathsf{contr}(\mathbf{B}_n)$ denote the bidirected graph obtained from $\mathbf{B}_n$ by contracting, for each node $t \in V(\mathbf{B}_n)$, the bidirected edge between $(t, 0)$ and $(t, 1)$, and the bidirected edge between $(t, 1)$ and $(t, 2)$. By the discussion above, $\mathsf{contr}(\mathbf{B}_n)$ is a subgraph of B_n. Therefore, B_n is an undirected minor of $\mathbf{B}_n$. □

Lemma 14. *zn*$(\boldsymbol{B}_n) = \Omega(\log n)$.

Proof. By Lemma 13, B_n is an undirected minor of $\mathbf{B}_n$. As a result, we obtain that the undirected pathwidth of $\mathbf{B}_n$ is at least the undirected pathwidth of B_n *cf.* [4]. Moreover, it is well-known that complete binary trees on n vertices have undirected pathwidth $\Theta(\log n)$ *cf.* [2,14]. Thus, since $\mathbf{B}_n$ is a bidirected graph, it follows from Lemma 7 that the directed pathwidth of $\mathbf{B}_n$ is $\Omega(\log n)$. Consequently, by Lemma 9, $\mathsf{dvsn}(\mathbf{B}_n) = \Omega(\log n)$. Moreover, by construction, $\mathbf{B}_n$ can be described as the union of 4 directed paths. Then, it follows from Lemma 12 that $\mathsf{zn}(\mathbf{B}_n) \geq \frac{\mathsf{dvsn}(\mathbf{B}_n)}{4}$. Therefore, $\mathsf{zn}(\mathbf{B}_n) = \Omega(\log n)$. □

Lemma 15. *For each positive integer n,* $\boldsymbol{B}_n$ *has directed treewidth at most* 8.

Proof. Based on Lemma 8, it suffices to prove that $\mathbf{B}_n$ admits a undirected tree decomposition $\mathcal{T} = (T, \mathcal{X})$ of width 8, where $\mathcal{X} = (X_t)_{t \in V(T)}$.

We define T simply as the complete binary tree obtained from B_n by removing all its leaves. Then, for each node $t \in V(T)$, we define the bag of t as the set $X_t = \{t, \mathsf{left}(t), \mathsf{right}(t)\} \times \{0, 1, 2\}$. Fig. 17 illustrates the tree decomposition $\mathcal{T} = (T, \mathcal{X})$. One can verify that $\mathcal{T}$ is a tree decomposition of $\mathbf{B}_n$ of width 8.

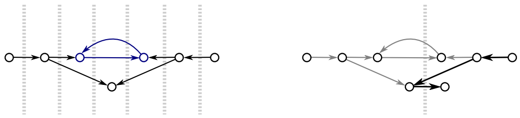
Based on Lemmas 14 and 15, we conclude the proof of Theorem 3.

6. Concluding remarks

We have shed new light on the time complexity of computing the zig-zag number of a directed graph. Nonetheless, some questions still remain open.

More specifically, we have proved that one can non-deterministically decide whether a directed graph G admits zig-zag number at most k in time $|G|^{\mathcal{O}(k)}$, concluding that k-ZIG-ZAG NUMBER is in NP for each fixed $k \geq 0$. Nevertheless, it remains unknown whether k-ZIG-ZAG NUMBER admits a non-deterministic FPT-time algorithm. Another interesting question concerns to determine whether ZIG-ZAG NUMBER is also in NP for non-fixed k. It is worth mentioning that, to settle k-ZIG-ZAG NUMBER in NP, we have actually proved that, given a directed graph G and a bijection $\pi : V(G) \to [|G|]$, deciding whether $\mathsf{zn}(G, \pi) \leq k$ is polynomial-time solvable for fixed k. However, for non-fixed k, deciding whether $\mathsf{zn}(G, \pi) \leq k$ is coNP-complete. As a matter of fact, given a bipartite directed graph G with bipartition $V(G) = X \cup Y$, if $\pi : V(G) \to [|G|]$ is defined in such a way that $x <_\pi y$ for each $x \in X$ and each $y \in Y$, then deciding whether $\mathsf{zn}(G, \pi) \geq |G| - 1$ is equivalent to deciding whether G admits a Hamiltonian path, which is a well-known NP-complete problem [11].

Another intriguing question concerns to determine whether 1-ZIG-ZAG NUMBER is polynomial-time solvable. As already mentioned, every directed acyclic graph has zig-zag number at most 1, and every directed graph containing directed cycles of length at least 3 must have zig-zag number at least 2. However, there exist directed graphs that are not directed acyclic



(a) $\mathsf{zn}(G) = 1$.

(b) $\mathsf{zn}(G) \geq 2$.

Fig. 18. (a) Example of directed graph G that is not directed acyclic and has zig-zag number 1. (b) Example of directed graph G that does not contain directed cycles of length at least 3 and yet has zig-zag number 2.

but still have zig-zag number at most 1 (see Fig. 18(a)). Note that, such graphs can only contain directed cycles that are *digons*, *i.e.* directed cycles of length 2. Nevertheless, this is not a sufficient condition for a directed graph to have zig-zag number at most 1. In fact, there exist directed graphs that only contain directed cycles that are digons and yet have zig-zag number at least 2 (see Fig. 18(b)). A property that seems to be useful to resolve this problem is the fact that, for every directed graph G, $\mathsf{zn}(G) \leq 1$ if and only if there exists a bijection $\pi : V(G) \to [|G|]$ such that, for each three distinct vertices $a, b, c \in V(G)$, with $(a, b), (b, c) \in E(G)$, either $a <_\pi b <_\pi c$ or $c <_\pi b <_\pi a$.

Motivated by the NP-hardness of 2-ZIG-ZAG NUMBER, we additionally ask whether k-ZIG-ZAG NUMBER is NP-hard for $k \geq 3$. In particular, determining whether k-ZIG-ZAG NUMBER is polynomially reducible to $(k + 1)$-ZIG-ZAG NUMBER is an elusive open problem. Generally, such a reduction must consist in constructing a directed graph H from a given directed graph G, such that $\mathsf{zn}(H) = \mathsf{zn}(G) + 1$. However, since for distinct bijections $\pi : V(G) \to [|G|]$ there might exist distinct directed paths P of G such that $\mathsf{zn}(G, \pi, P) = \mathsf{zn}(G, \pi)$, it is not obvious at all how G should be modified so as to produce a directed graph with zig-zag number exactly one unit greater than $\mathsf{zn}(G)$. Indeed, consider for instance the operation of adding a *universal vertex*, *i.e.* a vertex that is an out-neighbor and an in-neighbor of all the other vertices. There exist directed graphs G such that the addition of a universal vertex results in a directed graph with zig-zag number strictly greater than $\mathsf{zn}(G) + 1$; while there also exist directed graphs G such that the addition of a universal vertex results in a directed graph with zig-zag number equal to $\mathsf{zn}(G)$.

It is worth mentioning that, even if k-ZIG-ZAG NUMBER is proved to be NP-hard for every $k \geq 3$, zig-zag number is still a directed width measure of important theoretical and algorithmic interest. Indeed, besides the fact that zig-zag number is asymptotically upper bounded by directed pathwidth, there possibly exist efficient approximation algorithms with constant approximation factors for the k-ZIG-ZAG NUMBER problem, which remains an open question. Motivated by that, we ask for the existence of such approximation algorithms.

Finally, other important questions that are still open concern the establishment of relations between zig-zag number and distinct width measures. We have proved that there exist directed graphs of constant directed treewidth but unbounded zig-zag number. However, it is unknown whether the family of directed graphs of constant directed treewidth contains the family of directed graphs of constant zig-zag number. We remark that a counter-example for such containment would also imply the existence of directed graphs of constant tree-zig-zag number but unbounded directed treewidth, closing an open question from [14]. Related to this, we ask for the existence of a characterization of zig-zag number in terms of pursuit games.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work was supported by CAPES, Brazil (Finance Code: 001 and Grant Number : PDSE 88881.187636/2018-01), CNPq, Brazil (Grant Numbers: 303546/2016-6, 407635/2018-1, 140399/2017-8, and 303726/2017-2), FAPERJ, Brazil (Grant Numbers: E-26/202.793/2017 and E-26/203.272/2017), Bergen Research Foundation, and by the Research Council of Norway (Grant Number: 288761).

References

[1] J. Barát, Directed path-width and monotonicity in digraph searching, Graphs Comb. 22 (2) (2006) 161–172.

[2] D. Berwanger, A. Dawar, P. Hunter, S. Kreutzer, J. Obdrzálek, The DAG-width of directed graphs, J. Combin. Theory Ser. B 102 (4) (2012) 900–923.

[3] D. Berwanger, E. Grädel, L. Kaiser, R. Rabinovich, Entanglement and the complexity of directed graphs, Theoret. Comput. Sci. 463 (2012) 2–25.

[4] H.L. Bodlaender, A partial k-arboretum of graphs with bounded treewidth, Theoret. Comput. Sci. 209 (1–2) (1998) 1–45.

[5] B. Courcelle, The monadic second-order logic of graphs. I. Recognizable sets of finite graphs, Inform. and Comput. 85 (1) (1990) 12–75.

[6] B. Courcelle, J.A. Makowsky, U. Rotics, Linear time solvable optimization problems on graphs of bounded clique-width, Theory Comput. Syst. 33 (2) (2000) 125–150.

[7] R. Ganian, P. Hlinený, J. Kneis, A. Langer, J. Obdrzálek, P. Rossmanith, On digraph width measures in parameterized algorithmics, in: IWPEC, in: LNCS, vol. 5917, 2009, pp. 185–197.

[8] R. Ganian, P. Hliněný, J. Kneis, D. Meister, J. Obdržálek, P. Rossmanith, S. Sikdar, Are there any good digraph width measures? J. Combin. Theory Ser. B 116 (2016) 250–286.
[9] H. Gruber, M. Holzer, Finite automata, digraph connectivity, and regular expression size, in: ICALP, in: LNCS, vol. 5126, 2008, pp. 39–50.
[10] P. Hunter, S. Kreutzer, Digraph measures: Kelly decompositions, games, and orderings, Theoret. Comput. Sci. 399 (3) (2008) 206–219.
[11] A. Itai, C.H. Papadimitriou, J.L. Szwarcfiter, Hamilton paths in grid graphs, SIAM J. Comput. 11 (4) (1982) 676–686.
[12] T. Johnson, N. Robertson, P.D. Seymour, R. Thomas, Directed tree-width, J. Combin. Theory Ser. B 82 (1) (2001) 138–154.
[13] M. de Oliveira Oliveira, Subgraphs satisfying MSO properties on z-topologically orderable digraphs, in: IPEC, in: LNCS, vol. 8246, Springer, 2013, pp. 123–136.
[14] M. de Oliveira Oliveira, An algorithmic metatheorem for directed treewidth, Discrete Appl. Math. 204 (2016) 49–76.
[15] B.A. Reed, Introducing directed tree width, in: CTW, in: ENDM, vol. 3, 1999, pp. 222–229.
[16] M.A. Safari, D-width: A more natural measure for directed tree width, in: MFCS, in: LNCS, vol. 3618, 2005, pp. 745–756.
[17] T.J. Schaefer, The complexity of satisfiability problems, in: Proceedings of the Tenth Annual ACM Symposium on Theory of Computing, STOC'78, ACM, 1978, pp. 216–226.
[18] H. Tamaki, A polynomial time algorithm for bounded directed pathwidth, in: WG, in: LNCS, vol. 6986, 2011, pp. 331–342.
[19] B. Yang, Y. Cao, Digraph searching, directed vertex separation and directed pathwidth, Discrete Appl. Math. 156 (10) (2008) 1822–1837.

Appendix H

Abstracts of Additional Works

This appendix contains one-page abstracts of five additional works published during the doctoral studies. The first abstract corresponds to the work developed during the Master's thesis [39], but presented and fully published during the doctorate. The second abstract corresponds to current work recently presented [37]. The remaining three abstracts register and acknowledge the very fruitful visit to the University of Bergen [42–44].

1. Alexsander A. de Melo, Celina M. H. de Figueiredo, Uéverton S. Souza. On Undirected Two-commodity Integral Flow, Disjoint Paths and Strict Terminal Connection Problems. Presented in the *IX Latin and American Algorithms, Graphs and Optimization Symposium (2017)*, and published in *Networks* (2021) [40].

2. Celina M. H. de Figueiredo, Raul Lopes, Alexsander A. de Melo, Ana Silva. Parameterized algorithms for Steiner Tree and Dominating Set: bounding the leafage by the vertex leafage. Presented in the *16th International Conference and Workshops on Algorithms and Computation (WALCOM 2022)* [37].

3. Alexsander A. de Melo and Mateus de Oliveira Oliveira. On the Width of Regular Classes of Finite Structures. Presented in the *27th International Conference on Automated Deduction (CADE 2021)* as a recipient of the Woody Bledsoe Award [42], and submitted in April 2022 to *Theoretical Computer Science.*

4. Alexsander A. de Melo and Mateus de Oliveira Oliveira. Symbolic Solutions for Symbolic Constraint Satisfaction Problems. Presented in the *17th International Conference on Principles of Knowledge Representation and Reasoning (KR 2020)* [44].

5. Alexsander A. de Melo and Mateus de Oliveira Oliveira. Second-Order Finite Automata. Presented in the *15th International Computer Science Symposium*

in Russia (CSR 2020) as an invited paper [43], and accepted for publication in *Theory of Computing Systems*.

On Undirected Two-commodity Integral Flow, Disjoint Paths and Strict Terminal Connection Problems[*]

Alexsander A. de Melo[1] Celina M. H. de Figueiredo[1] Uéverton dos Santos Souza[2]

[1]Federal University of Rio de Janeiro, Rio de Janeiro, Brazil
{aamelo, celina}@cos.ufrj.br
[2]Federal Fluminense University, Niterói, Brazil
ueverton@ic.uff.br

Abstract

Even, Itai and Shamir (1976) proved simple two-commodity integral flow is NP-complete both in the directed and undirected cases. In particular, the directed case was shown to be NP-complete even if one demand is unitary, which was improved by Fortune, Hopcroft and Wyllie (1980) who proved the problem is still NP-complete if both demands are unitary. The undirected case, on the other hand, was proved by Robertson and Seymour (1995) to be polynomial-time solvable if both demands are constant. Nevertheless, the complexity of the undirected case with exactly one constant demand has remained unknown. We close this forty-year complexity gap, by showing the undirected case is NP-complete even if exactly one demand is unitary. As a by product, we obtain the NP-completeness of determining whether a graph contains $1 + d$ pairwise vertex-disjoint paths, such that one path is between a given pair of vertices and d paths are between a second given pair of vertices. Additionally, we investigate the complexity of another related network design problem called Strict terminal connection.

Keywords. multicommodity integral flow, unitary demand, disjoint paths, terminal vertices, router vertices, connection tree, Steiner tree

[*]The authors acknowledge that this work was partially supported by the Brazilian research agencies CAPES, CNPq and FAPERJ.

Parameterized algorithms for Steiner Tree and Dominating Set: bounding the leafage by the vertex leafage[*]

Celina M. H. de Figueiredo[1] Raul Lopes[2]
Alexsander A. de Melo[1] Ana Silva[2]

[1]Universidade Federal do Rio de Janeiro, Rio de Janeiro, Brazil
{aamelo, celina}@cos.ufrj.br
[2]Universidade Federal do Ceará, Fortaleza, Brazil
raul@alu.ufc.br,anasilva@mat.ufc.br

Abstract

Chordal graphs are intersection graphs of subtrees of a tree, while interval graphs are intersection graphs of subpaths of a path. Undirected path graphs are an intermediate class of graphs, defined as the intersection graphs of paths of a tree. It is known that Dominating Set, Connected Dominating Set, and Steiner Tree are W[2]-hard on chordal graphs, when parameterized by the size of the solution, and are polynomial-time solvable on interval graphs. As for the undirected path graphs, all these problems are known to be NP-complete, but no classification in the parameter tractability complexity theory is known, apart from the trivial XP classification. In this paper, we prove that Dominating Set, Connected Dominating Set, and Steiner Tree are FPT for undirected path graphs when parameterized by the size of the solution. We also prove that they continue to be FPT for general chordal graphs when parameterized by the size of the solution plus the vertex leafage of the graph, provided a tree model with optimal vertex leafage is given. Finally, we show a relation between the parameterization of Min-LC-VSP problems by the leafage of the graph versus the vertex leafage plus the size of a solution.

Keywords. Chordal graphs, Undirected Path graphs, Dominating Set, Steiner Tree, FPT algorithms

[*]Celina M. H. de Figueiredo was partially funded by CNPq grants 302823/2016-6 and 407635/2018-1, CAPES Finance Code 001, and FAPERJ CNE E-26/202.793/2017. Raul Lopes was partially funded by FUNCAP Pronem 4543945/2016. Alexsander A. de Melo was partially funded by CNPq grant 140399/2017-8 and CAPES Finance Code 001. Ana Silva was partially funded by CNPq grants 303803/2020-7 and 437841/2018-9, and FUNCAP/CNPq grant 0112-00061.01.00/16.

On the Width of Regular Classes of Finite Structures*

Alexsander Andrade de Melo[1] Mateus de Oliveira Oliveira[2]

[1]Federal University of Rio de Janeiro, Rio de Janeiro, Brazil
aamelo@cos.ufrj.br
[2]University of Bergen, Bergen, Norway
mateus.oliveira@uib.no

Abstract

In this work, we introduce the notion of decisional width of a finite relational structure and the notion of decisional width of a regular class of finite structures. Our main result states that given a first-order formula ψ over a vocabulary τ, and a finite automaton $\mathcal{F}$ over a suitable alphabet $\mathcal{B}(\Sigma, w, \tau)$ representing a width-w regular-decisional class of τ-structures $\mathcal{C}$, one can decide in time $f(\tau, \Sigma, \psi, w) \cdot |\mathcal{F}|$ whether some τ-structure in $\mathcal{C}$ satisfies ψ. Here, f is a function that depends on the parameters τ, Σ, ψ, w, but not on the size of the automaton $\mathcal{F}$ representing the class. Therefore, besides implying that the first-order theory of any given regular-decisional class of finite structures is decidable, it also implies that when the parameters τ, ψ, Σ and w are fixed, decidability can be achieved in linear time on the size of the input automaton $\mathcal{F}$. Building on the proof of our main result, we show that the problem of counting satisfying assignments for a first-order logic formula in a given structure $\mathfrak{A}$ of width w is fixed-parameter tractable with respect to w, and can be solved in quadratic time on the length of the input representation of $\mathfrak{A}$.

Keywords. Automatic Structures, Width Measures, First Order Logic

*Alexsander Andrade de Melo acknowledges support from the Brazilian National Council for Scientific and Technological Development (CNPQ 140399/2017-8) and from the Brazilian Federal Agency for Support and Evaluation of Graduate Education (CAPES 88881.187636/2018-01). Mateus de Oliveira Oliveira acknowledges support from the Bergen Research Foundation, from the Research Council of Norway (288761) and from the Sigma2 network (NN9535K).

Symbolic Solutions for Symbolic Constraint Satisfaction Problems[*]

Alexsander Andrade de Melo[1] Mateus de Oliveira Oliveira[2]

[1]Federal University of Rio de Janeiro, Rio de Janeiro, Brazil
aamelo@cos.ufrj.br
[2]University of Bergen, Bergen, Norway
mateus.oliveira@uib.no

Abstract

A fundamental drawback that arises when one is faced with the task of deterministically certifying solutions to computational problems in PSPACE is the fact that witnesses may have super-polynomial size, assuming that NP$\neq$PSPACE. Therefore, the complexity of such a deterministic verifier may already be super-polynomially lower-bounded by the size of a witness. In this work, we introduce a new symbolic framework to address this drawback. More precisely, we introduce a PSPACE-hard notion of symbolic constraint satisfaction problem where both instances and solutions for these instances are implicitly represented by ordered decision diagrams (i.e. read-once, oblivious, branching programs). Our main result states that given an ordered decision diagram D of length k and width w specifying a CSP instance, one can determine in time $f(w, w') \cdot k$ whether this instance has a solution which can be encoded by an ODD of width w'. Intuitively, while the parameter w quantifies the complexity of the instance, the parameter w' quantifies the complexity of a prospective solution. We show that CSPs of constant width can be used to formalize natural PSPACE hard problems, such as reachability of configurations for Turing machines working in non-deterministic linear space. For such problems, our main result immediately yields an algorithm that determines the existence of solutions of width w in time $g(w) \cdot n$, where $g : \mathbb{N} \to \mathbb{N}$ is a suitable computable function, and n is the size of the input.

[*]Alexsander Andrade de Melo acknowledges support from the Brazilian National Council for Scientific and Technological Development (CNPQ 140399/2017-8) and from the Brazilian Federal Agency for Support and Evaluation of Graduate Education (CAPES 88881.187636/2018-01). Mateus de Oliveira Oliveira acknowledges support from the Bergen Research Foundation, from the Research Council of Norway (288761) and from the Sigma2 network (NN9535K).

Second-Order Finite Automata*

Alexander Andrade de Melo[1] Mateus de Oliveira Oliveira[2]

[1]Federal University of Rio de Janeiro, Rio de Janeiro, Brazil
aamelo@cos.ufrj.br
[2]University of Bergen, Bergen, Norway
mateus.oliveira@uib.no

Abstract

Traditionally, finite automata theory has been used as a framework for the representation of possibly infinite sets of strings. In this work, we introduce the notion of second-order finite automata, a formalism that combines finite automata with ordered decision diagrams, with the aim of representing possibly infinite *sets of sets* of strings. Our main result states that second-order finite automata can be canonized with respect to the second-order languages they represent. Using this canonization result, we show that sets of sets of strings represented by second-order finite automata are closed under the usual Boolean operations, such as union, intersection, difference and even under a suitable notion of complementation. Additionally, emptiness of intersection and inclusion are decidable.

We provide two algorithmic applications for second-order automata. First, we show that several width/size minimization problems for deterministic and nondeterministic ODDs are solvable in fixed-parameter tractable time when parameterized by the width of the input ODD. In particular, our results imply FPT algorithms for corresponding width/size minimization problems for ordered binary decision diagrams (OBDDs) with a fixed variable ordering. Previously, only algorithms that take exponential time in the size of the input OBDD were known for width minimization, even for OBDDs of constant width. Second, we show that for each k and w one can count the number of distinct functions computable by ODDs of width at most w and length k in time $h(|\Sigma|, w) \cdot k^{O(1)}$, for a suitable $h : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$. This improves exponentially on the time necessary to explicitly enumerate all such functions, which is exponential in both the width parameter w and in the length k of the ODDs.

Keywords: Second-Order Finite Automata, Ordered Decision Diagrams, Fixed-Parameter Tractability

*Alexander Andrade de Melo acknowledges support from the Brazilian National Council for Scientific and Technological Development (CNPQ 140399/2017-8) and from the Brazilian Federal Agency for Support and Evaluation of Graduate Education (CAPES 88881.187636/2018-01). Mateus de Oliveira Oliveira acknowledges support from the Bergen Research Foundation, from the Research Council of Norway (288761) and from the Sigma2 network (NN9535K).