ON DIGRAPHS WITH A ROOTED TREE STRUCTURE

Jayme L. Szwarcfiter

NCE-04/83

Dezembro 1983

Jayme Luiz Szwarcfiter
Universidade Federal do Rio de Janeiro
Núcleo de Computação Eletrônica
Caixa Postal 2324
20.001, Rio de Janeiro, RJ, Brasil

R E S U M O

Uma classe especial de digrafos redutíveis é caracterizada e algoritmos polinomiais são descritos para o seu reconhecimento, isomorfismo e determinanar digrafos equivalentes mínimos. Um algoritmo aproximativo é também apresentado para resolver este último problema, em seu caso geral. O tamanho da aproximação obtida é sempre menor do que o dobro da solução exata. Em adição, o isomorfismo de busca em profundidade é resolvido como um caso especial do isomorfismo dessa classe.

A B S T R A C T

A special class of reducible digraphs is characterized and polynomial time algorithms are described for their recognition, isomorphism and finding minimum equivalent digraphs. An approximative algorithm is also given for solving this last problem in its general case. The size of the approximation is always less than twice the exact solution. In addition, isomorphism of depth first search is solved as a special case of isomorphism of this class.

1. Introduction

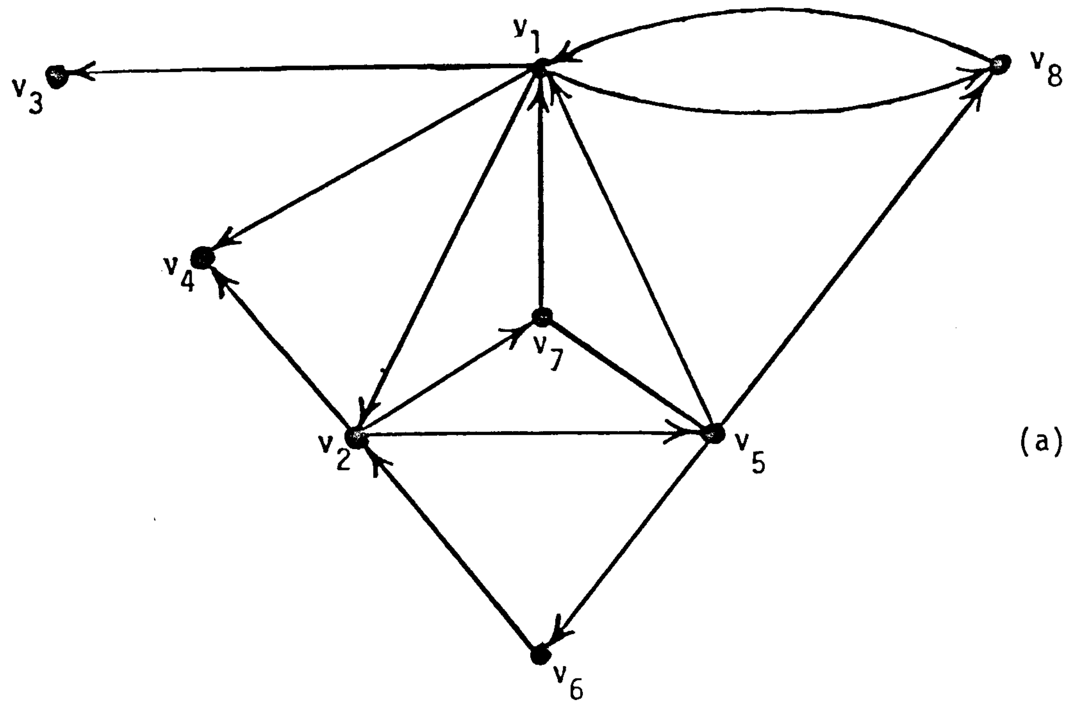
We examine a special class of reducible digraphs named tree reducible (TR) which are closely related to depth first search (DFS). A characterization of the class is first considered. Based on it we describe algorithms for the problems of recognition, finding a minimum equivalent (MEQ) and isomorphism of TR digraphs. The time bounds of these three algorithms is the same as recognizing reducible digraphs, i.e. almost linear [9]. In contrast, the MEQ problem is NP-hard for general digraphs [3,8], whereas isomorphism of reducible digraphs is complete, i.e. equivalent to the general case (because reducible digraphs contain acyclic digraphs, whose isomorphism is known to be complete). In addition, an approximative algorithm is proposed for finding the MEQ of a general digraph. The size (number of edges) of the approximation obtained is always less than twice that of the optimal solution. Finally, we consider the problem of verifying whether two DFS's of an undirected graph are isomorphic. This is shown to be a special case of TR digraph isomorphism and solvable in linear time in the size of the graphs.

The following describes the terminology.

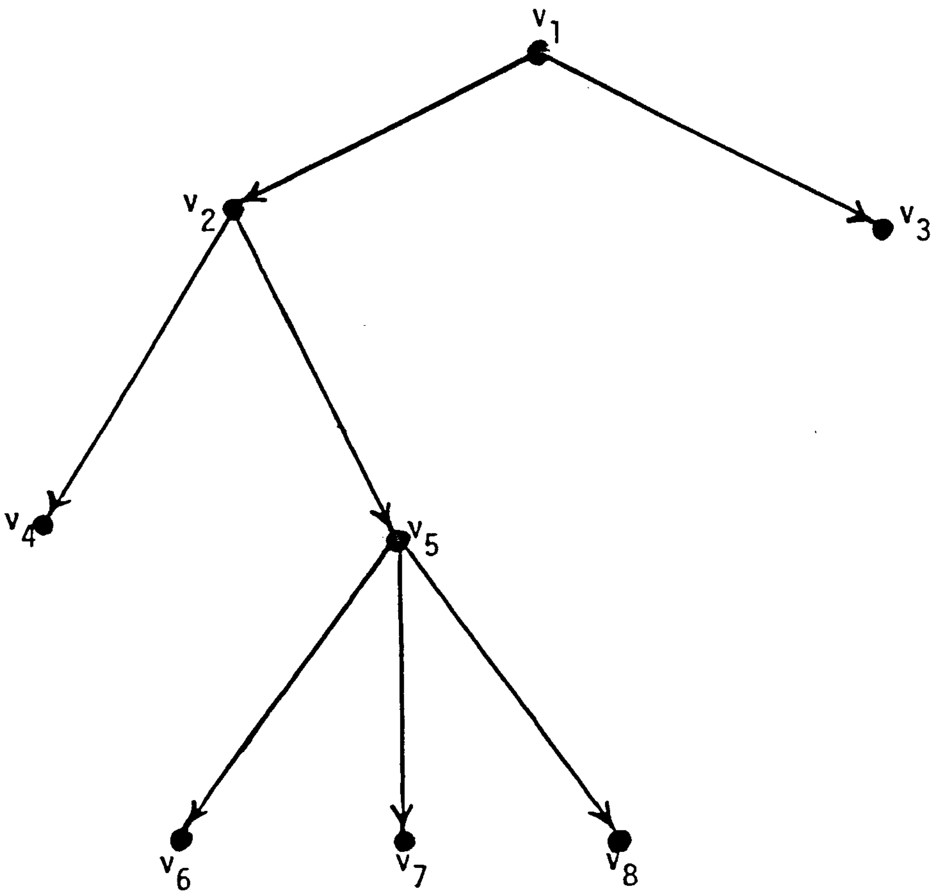
The vertices of a graph can be traversed according to predefined rules, such as those of depth first search (DFS). A DFS of an undirected graph divides its edges into two disjoint subsets, tree edges and fronds, respectively. For a digraph it produces four disjoint subsets, the tree, forward, back and cross edges, respectively. The initial vertex of a DFS is the start of the search. The positive integer indicating the order in which each vertex v has been first considered is the DFS-number of v. A description of DFS can be found in [1], for instance.

Let D(V,E) be a digraph. If there exists a path from $v \in V$ to $w \in V$ then v is said to reach w. If v, w are such that neither of them reaches the other then v, w are incomparable. If every vertex of D is reachable from a vertex $s \in V$ then s is a root of D. If any DFS of D starting at some fixed root s determines the same set B of back edges then D is (fully) reducible. The digraph $D_A(V, E-B)$ is the directed acyclic graph (dag) associated to D. If D is reducible then D_A is unique. See [2, 4-5, 7] for other characterizations of reducible digraphs.

Figure 1: A TR digraph of root v_1 and the transitive reduction of its dag.



(a)



(b)

(2) ==>(3): Suppose the transitive closure of D_A contains figure 2 as an induced subgraph. Then v_1, v_2 reach w while v_1, v_2 are incomparable in D_A. Let Δ be a DFS of D and v_1 the first vertex among v_1, v_2, w to be considered in Δ. Then any path from v_2 to w in D_A contains a cross edge of Δ. The same applies if we

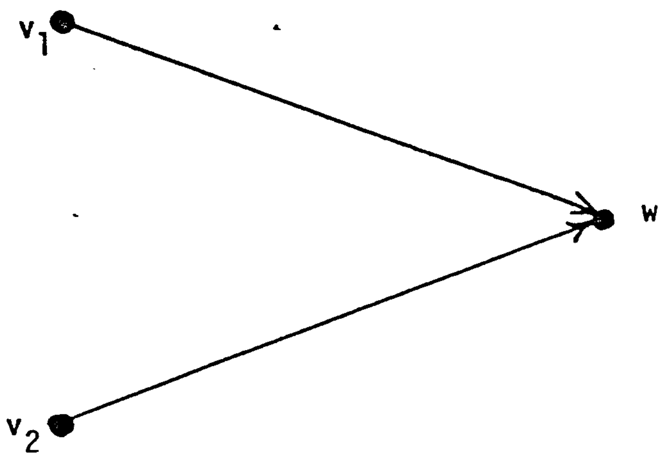


Figure 2: The forbidden induced subdigraph for the transitive closure of the dag of a TR digraph

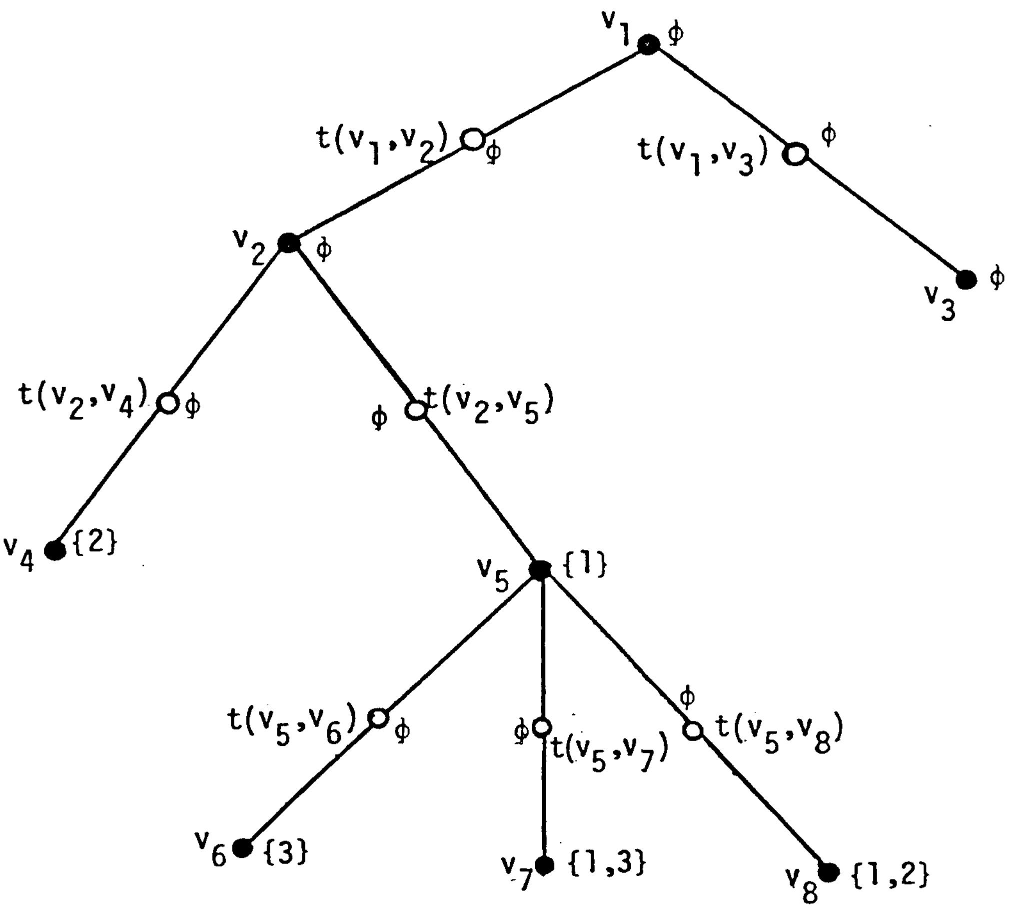


Figure 3: The labelled rooted tree $L(\alpha(D))$

preserved D_M must contain a simple path $v_1, \ldots, v_{k-1}, v_k$, $k \geq 3$, from $v \equiv v_1$ to $w \equiv v_k$. Consider now a canonical DFS of D and examine edge (v_{k-1}, v_k). It can not be a tree edge since $(v,w) \equiv (v_1, v_k)$ is one, by hypothesis. (v_{k-1}, v_k) is not a back edge because it would violate reducibility: there would be a path in D from s to v_{k-1} avoiding v_k (namely, the path from s to $v \equiv v_1$ in T followed by the path $v_1, \ldots, v_{k-1}$ in D_M). There are no cross edges. The only possibility is therefore (v_{k-1}, v_k) to be a forward edge and $v \neq s$. In this case the digraph $(V, [E_M-(v_{k-1}, v_k)] \cup \{(v,w)\})$ is also a MEQ of D and contains one more edge of E_T than D_M. This completes the proof▲

From the above lemma we conclude that every TR digraph D(V,E) has a MEQ of the form $(V, E_T \cup E')$, where E_T is the set of tree edges of a canonical DFS of D and E' a subset of back edges. The computation of E' can be done by iteratively selecting the back edge (w,b(w)) from a leaf w of T to its oldest ancestor b(w) and then collapsing into b(w) the path in T from b(w) to w. The collapsing operation becomes simpler when the leaf w is chosen so as to maximize the level in T of b(w). The formulation below describes the process.

<u>Initial step</u>:

<u>let</u> D(V,E) be a given TR digraph
    Δ a canonical DFS of D
    $T(V, E_T)$ the spanning tree of Δ
    and W(T) the set of leaves of T
E' := ϕ
<u>for</u> each $v \in V$ <u>define</u>
    actual(v) := v
    level(v) := level of v in T
    $B(v) := \{w \in V | (v,w)$ is a back edge$\}$
    <u>if</u> $B(v) = \phi$ <u>then</u> b(v) := v
    <u>otherwise</u> b(v) := w, where $w \in B(v)$ satisfies
        $level(w) \leq level(w')$, for all $w' \in B(v)$

<u>General</u> <u>step</u>:

<u>if</u> $W(T) = \phi$ <u>then</u> <u>stop</u>: $(V, E_T \cup E')$ is a MEQ of D
<u>otherwise</u> choose $w \in W(T)$ such that
    $level(b(w)) \geq level(b(w'))$, for all $w' \in W(T)$
  <u>if</u> $b(w) = w$ <u>then</u> remove w from T
  <u>otherwise</u> let $v_1, \ldots, v_k$ be the path P in T from $v_1 = b(w)$ to $v_k = w$
    $E' := E' \cup \{(actual(v_k), v_1)\}$
    find a vertex v_j in P such that
      $level(b(v_j)) \leq level(b(v_i))$, for all $1 \leq i \leq k$
    $b(v_1) := b(v_j)$
    $actual(v_1) := actual(v_j)$
    collapse the path $v_1, \ldots, v_k$ into v_1 in T
      (i.e. remove $v_2, \ldots, v_k$ from T
        and set $A_{v_1}(T) := \bigcup_{1 \leq i \leq k} A_{v_i}(T) - \{v_2, \ldots, v_k\})$
  <u>repete</u> the general step ▲

Next is described an approximative algorithm for finding a MEQ of a general digraph.

Let D be a strongly connected digraph. In the <u>initial</u> step, label all vertices <u>uncovered</u>. Then choose an arbitrary cycle c_1 and label <u>covered</u> each vertex of c_1. In the <u>general</u> <u>step</u>, assume the cycles $c_1, \ldots, c_j$, $j \geq 1$, have already been chosen, with the union of them defining a strongly connected subdigraph D_j and that a vertex v of D is covered iff v is in D_j. Now, if all vertices are covered the process terminates: D_j is the approximative MEQ. Otherwise, find a path $P(v,w)$ from v to w in D, such that

(i) v and w are covered and not necessarily distinct vertices, and
(ii) except for its ends v and w, $P(v,w)$ contains solely vertices labeled uncovered.

This path is easy to find and since D is strongly connected it necessarily exists. Then define cycle c_{j+1} as consisting of $P(v,w)$ followed by an arbitrary (simple) path from w to v in D_j. Finally, label <u>covered</u> each vertex of $P(v,w)$ and repeat the general step. It is clear that the required assumptions are all met.

This process takes linear time in the size of D.

If D is strongly connected apply standard techniques: use the above algorithm for each strongly connected component and then add the edges corresponding to the transitive reduction of the condensation digraph D_c of D (D_c has one vertex for each component S_i of D and an edge (S_i, S_j) if there is some edge in D from a vertex of S_i to another of S_j).

To evaluate the quality of the proposed approximation, suppose first that the given digraph D is strongly connected. Let us compute the number of edges of the approximative MEQ D_k constructed by the algorithm. Recall that D_k is a union of cycles $c_1, \ldots, c_k$. The first cycle c_1 covers $n_1 > 1$ vertices and has n_1 edges. Each subsequent cycle c_j, $1<j\leq k$, covers $n_j > 0$ new vertices (not covered by any of the preceding $c_1, \ldots, c_{j-1}$) and has precisely n_j+1 edges not belonging to any of $c_1, \ldots, c_{j-1}$. Since $\Sigma n_j = |V|$, we conclude that D_k has at most $2|V|-2$ edges. On the other hand, the actual MEQ of D has at least $|V|$ edges. Therefore the number of edges of the approximation is less than twice that of the actual MEQ. If D is not strongly connected the edges connecting different components occur in the approximation with the same frequency as in the actual MEQ. Therefore the bound applies in general.

4. Isomorphism

In this section is described an algorithm for isomorphism of TR digraphs.

Let D(V,E) be a TR digraph. Define $\alpha(D)$ to be the digraph obtained as follows:

(i) Perform a canonical DFS of D. Let T be the corresponding DFS spanning tree.

(ii) Find a subdivision $\alpha(D)$ of T, i.e. define $\alpha(D) := T$ and then replace each edge (v,w) of $\alpha(D)$ by the pair of directed edges $(v, t(v, w))$ and $(t(v,w),w)$, where $t(v,w)$ is a newly introduced vertex.

(iii) For each forward edge $(v_1, v_2) \in E$ add to $\alpha(D)$ the edge $(t(v_1, w),v_2)$, where w is the son of v_1 in the path from v_1 to v_2 in T.

(iv) For each back edge $(v_1, v_2) \in E$ add to $\alpha(D)$ the edge (v_2, v_1).

$\alpha(D)$ is clearly TR and acyclic. Furthermore it preserves isomorphism, i.e. if D_1, D_2 are TR digraphs then $D_1 \cong D_2$ iff $\alpha(D_1) \cong \alpha(D_2)$.

Let H be a general acyclic TR digraph. Define L(H) to be the labelled rooted tree obtained as follows:

(i) Perform a canonical DFS of H. Let L(H) be the corresponding DFS spanning tree.

(ii) Compute <u>level</u>(v), the level in L(H) of each vertex v of L(H).

(iii) To each vertex w of L(H) assign a set L(w) of positive integer labels, defined by:

$L(w) := \{level(v) \mid (v,w) \text{ is a forward edge of } H\}$.

As an example, if D is the TR digraph of figure 1(a) then $L(\alpha(D))$ is the one of figure 3.

It follows that there is a one-to-one correspondence between acyclic TR digraphs and labelled rooted trees such that the labels of each vertex v form a subset of $\{1,2,\ldots, level(v)-2\}$.

<u>Lemma 3</u>: Let D_1, D_2 be TR digraphs. Then $D_1 \cong D_2$ iff $L(\alpha(D_1)) \cong L(\alpha(D_2))$ as labelled rooted trees.

An isomorphism algorithm can be formulated as follows. Let D_1 and D_2 be two given TR digraphs. Construct $\alpha(D_1)$ and $\alpha(D_2)$. Then $L(\alpha(D_1))$ and $L(\alpha(D_2))$. Verify whether or not $L(\alpha(D_1))$ and $L(\alpha(D_2))$ are isomorphic labelled rooted trees, using [6]. By lemma 3, this answers the isomorphism of D_1 and D_2.

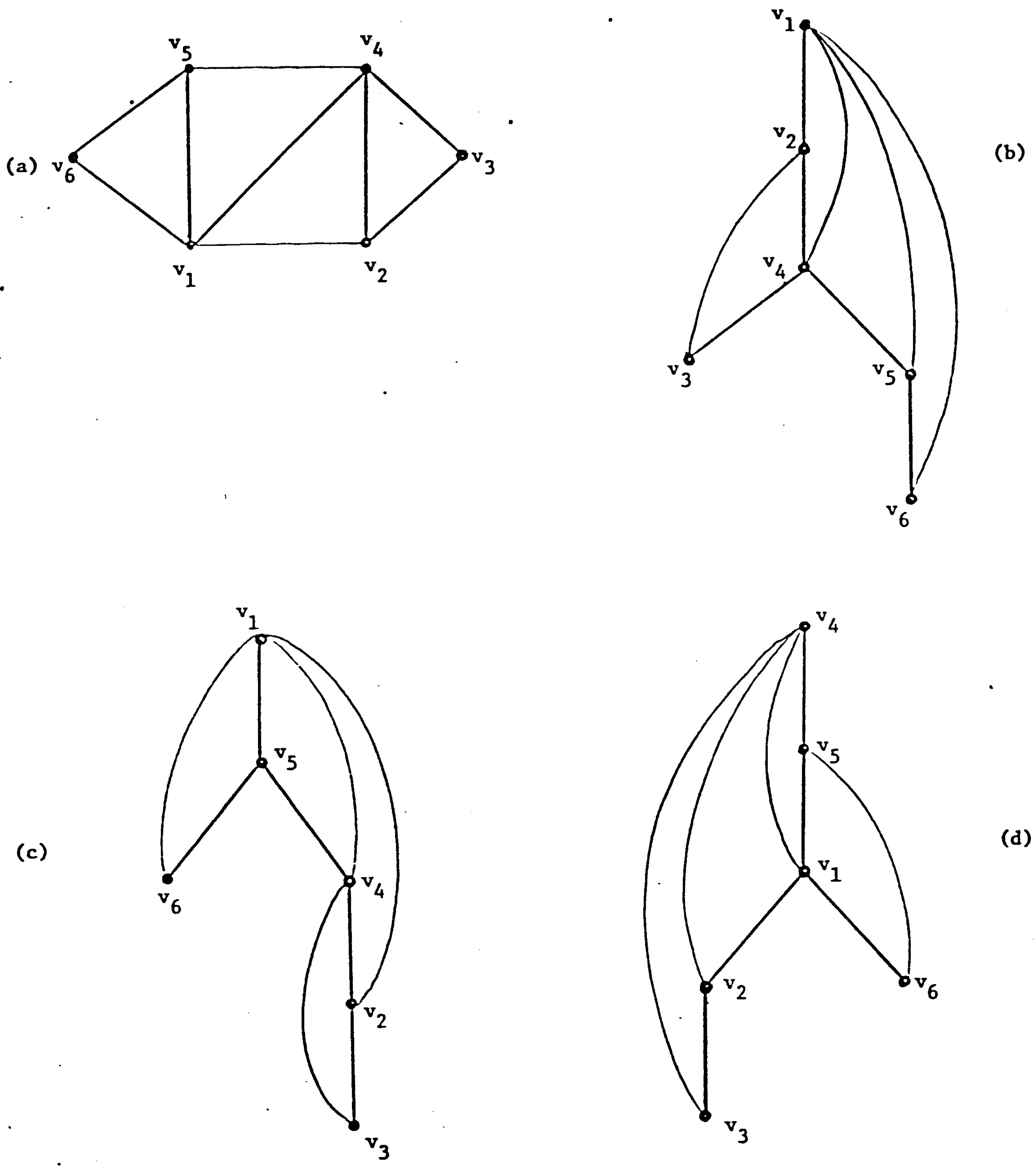


Fig. 4: An undirected graph and three possible DFS

5. Isomorphism of DFS

Let G(V,E) be an undirected graph and Δ_1, Δ_2 two DFS's of G, starting respectively at vertices s_1 and s_2. Δ_1 and Δ_2 are <u>isomorphic</u> when there exists a permutation f of V such that $s_2=f(s_1)$ and for every edge $(v,w) \in E$,

$$(v,w) \text{ is a tree (back) edge of } \Delta_1$$
$$\Updownarrow$$
$$(f(v),f(w)) \text{ is a tree (back) edge of } \Delta_2.$$

For example, figure 4 shows an undirected graph and three DFS's of it. Those of 4(b) and 4(d) are isomorphic, while 4(b) and 4(c) are not.

Consider a DFS Δ of a connected undirected graph G. Denote by $\overrightarrow{G,\Delta}$ the digraph obtained by directing each edge of G from lower to higher DFS-number of its vertices. $\overrightarrow{G,\Delta}$ is acyclic, TR and unique for each DFS.

The following is a simple algorithm for DFS isomorphism. Let G(V,E) be a connected graph and Δ_1, Δ_2 two DFS's of it. Construct $\overrightarrow{G,\Delta_1}$ and $\overrightarrow{G,\Delta_2}$. Verify whether these digraphs are isomorphic, using §4. Then $\Delta_1 \cong \Delta_2$ iff $\overrightarrow{G,\Delta_1} \cong \overrightarrow{G,\Delta_2}$.

6. Conclusions

The class of TR digraphs has been considered. Some problems were shown to admit special algorithms for TR digraphs, which are better than those known for the general case. The same applies also for some other problems, such as computing minimal chain decompositions and finding dominators.

R E F E R É N C E S

[1] A.V. Aho, J.E. Hopcroft and J.D. Ullman, The Design and Analysis of Computer Algorithms, Addison-Wesley, Reading, 1974.

[2] J.Cocke and J.T. Schwartz, "Programming Languages and their Compilers: preliminary notes", Courant Institute of Math. Sci. (1970).

[3] M.R. Garey and D.S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, Freeman, San Francisco, 1979.

[4] M.S. Hecht and J.D. Ullman, "Flow Graph Reducibility", SIAM J. Comp., 1, 188-202 (1972).

[5] M.S. Hecht and J.D. Ullman, "Characterizations of Reducible Flow Graphs", J. of the ACM, 21, 367-375 (1974).

[6] J.E. Hopcroft and R.E. Tarjan, "Isomorphism of Planar Graphs", in Complexity of Computer Computations, R.E. Miller and J.W. Thatcher, eds., Plenum Press, New York, 1972.

[7] V.N. Kasyanov, "Some Properties of Fully Reducible Graphs", Inf. Proc. Letters, 2, 113-117 (1973).

[8] S.Sahni, "Computationally Related Problems", SIAM J. Comp., 3, 262-279 (1974).

[9] R.E. Tarjan, "Testing Flow Graph Reducibility", J. Comp. Systems Sciences, 9, 355-365 (1974).