CONTROLE SUPERVISÓRIO ROBUSTO DE SISTEMAS A EVENTOS DISCRETOS SUJEITOS A PERDAS INTERMITENTES DE OBSERVAÇÃO

Marcos Vinícius Silva Alves

Dissertação de Mestrado apresentada ao Programa de Pós-graduação em Engenharia Elétrica, COPPE, da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Mestre em Engenharia Elétrica.

Orientadores: João Carlos dos Santos Basilio
Lilian Kawakami Carvalho

Rio de Janeiro
Fevereiro de 2014

CONTROLE SUPERVISÓRIO ROBUSTO DE SISTEMAS A EVENTOS DISCRETOS SUJEITOS A PERDAS INTERMITENTES DE OBSERVAÇÃO

Marcos Vinícius Silva Alves

DISSERTAÇÃO SUBMETIDA AO CORPO DOCENTE DO INSTITUTO ALBERTO LUIZ COIMBRA DE PÓS-GRADUAÇÃO E PESQUISA DE ENGENHARIA (COPPE) DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS EM ENGENHARIA ELÉTRICA.

Examinada por:

Prof. João Carlos dos Santos Basilio, Ph.D.

Prof. Lilian Kawakami Carvalho, D. Sc.

Prof. Antonio Eduardo Carrilho da Cunha, D. Eng.

Prof. José Eduardo Ribeiro Cury, Docteur d'Etat

RIO DE JANEIRO, RJ – BRASIL
FEVEREIRO DE 2014

Alves, Marcos Vinícius Silva

Controle supervisório robusto de sistemas a eventos discretos sujeitos a perdas intermitentes de observação/Marcos Vinícius Silva Alves. – Rio de Janeiro: UFRJ/COPPE, 2014.

XIV, 63 p.: il.; 29, 7cm.

Orientadores: João Carlos dos Santos Basilio

Lilian Kawakami Carvalho

Dissertação (mestrado) – UFRJ/COPPE/Programa de Engenharia Elétrica, 2014.

Referências Bibliográficas: p. 61 – 63.

1. Sistemas a eventos discretos. 2. Controle supervisório. 3. Robustez. 4. Autômatos. I. Basilio, João Carlos dos Santos *et al.* II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia Elétrica. III. Título.

A minha mãe, Noêmia, ao meu pai, Marcelo, e aos meus irmãos Marcelo, Charlles e Carlos.

Agradecimentos

Gostaria de agradecer primeiramente a Deus e à minha família, pelo amparo nos momentos difíceis.

Em especial, à minha mãe, Noêmia, pelo amor e imenso zelo ao cuidar dos seus filhos.

Ao meu pai, Marcelo, pelo esforço e luta para nos dar conforto.

Aos meus irmãos, Carlos, Charlles e Marcelo, verdadeiros companheiros em todos os desafios da vida.

À minha namorada, Mayne, que sempre me motivou e apoiou nos momentos mais complicados durante todo o curso de mestrado. Obrigado pelo seu amor e carinho.

Aos meus orientadores, Professor Basilio e Professora Lilian, pela dedicação e paciência ao guiarem meus passos neste trabalho.

Ao Professor Carrilho, que participou ativamente da elaboração desse trabalho.

Ao Dr. Kurt Rohloff, por nos dar acesso às suas anotações referentes ao seu trabalho em controle supervisório robusto seguro em presença de falhas permanentes de sensores.

Aos colegas do Laboratório de Controle e Automação da UFRJ, em especial: Carlos Eduardo, Gustavo, Felipe e Félix.

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

CONTROLE SUPERVISÓRIO ROBUSTO DE SISTEMAS A EVENTOS DISCRETOS SUJEITOS A PERDAS INTERMITENTES DE OBSERVAÇÃO

Marcos Vinícius Silva Alves

Fevereiro/2014

Orientadores: João Carlos dos Santos Basilio
Lilian Kawakami Carvalho

Programa: Engenharia Elétrica

Na área de sistemas a eventos discretos, o controle supervisório robusto foi inicialmente formulado como o problema de se projetar um supervisor capaz de prover o comportamento desejado ao atuar sobre um conjunto de modelos (autômatos), ao invés de um único modelo nominal. Uma abordagem diferente foi recentemente proposta levando ao chamado problema de controle supervisório seguro em presença de falhas de sensores. Nessa abordagem, os sensores podem falhar a qualquer momento, contudo, quando uma falha ocorre, ela é permanente e o evento associado permanece não-observável para sempre. Neste trabalho, é considerado o problema de controle supervisório robusto supondo perdas intermitentes de observação. Essa é uma abordagem mais geral do que a anterior, uma vez que perdas permanentes de observabilidade podem ser vistas como um caso particular de perdas intermitentes de observação. Para tanto, foi usada a operação de dilatação de autômatos, proposta na literatura no contexto de diagnose de falhas, permitindo estender a definição usual de observabilidade para a observabilidade robusta. Nesse contexto, duas definições de observabilidade robusta são apresentadas: a observabilidade robusta forte e a observabilidade robusta fraca. A principal contribuição do trabalho é um método para projetar supervisores robustos em presença de perdas intermitentes de observação, de tal forma que a linguagem admissível permaneça robusta (no sentido fraco).

Abstract of Dissertation presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Master of Science (M.Sc.)

ROBUST SUPERVISORY CONTROL OF DISCRETE EVENT SYSTEMS AGAINST INTERMITTENT LOSS OF OBSERVATIONS

Marcos Vinícius Silva Alves

February/2014

Advisors: João Carlos dos Santos Basilio
Lilian Kawakami Carvalho

Department: Electrical Engineering

In the area of discrete event systems, the robust supervisory control problem was initially formulated as the problem of designing a robust supervisor with the view to guaranteeing the desired behavior assuming that the discrete event system is modeled by a class of automata and not as a single nominal one. A different approach has been recently proposed leading to the so-called safe supervisory control in the presence of sensor faults. In this approach, it is assumed that sensors can fail any time but once the fault occurs, it is permanent and the related event remains unobservable forever. In this work we consider the problem of robust supervisory control assuming intermittent loss of observations. This is a more general approach than the previous one, since permanent loss of observations can be seen as a particular case of intermittent ones. The dilation operation, proposed in literature in the context of fault diagnosis, is used to extend the definitions of observability to robust observability. In this context, two definitions of robust observability are proposed here: the strong robust observability and the weak robust observability. The main contribution of this work is a method to design robust supervisors that keep the admissible language robust (in the weak sense).

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Símbolos

Σ	Conjunto de eventos (alfabeto)
Σ_c	Conjunto de eventos controláveis
Σ_{uc}	Conjunto de eventos não-controláveis
Σ_o	Conjunto de eventos observáveis
Σ_{uo}	Conjunto de eventos não-observáveis
Σ_{ilo}	Conjunto de eventos observáveis sujeitos a perdas intermitentes de observação
Σ_{nilo}	Conjunto de eventos sempre observáveis
Σ_{dil}	Conjuntos de eventos dilatados
Σ^*	Fecho de Kleene de um conjunto de eventos Σ
ε	sequência nula
$\emptyset$	Conjunto (linguagem) vazio
$\|s\|$	Comprimento de uma sequência s
K^C	Complemento da linguagem K, definida sobre Σ, em relação a Σ
$K_1 \cup K_2$	Operação de união sobre as linguagens K_1 e K_2
$K_1 \cap K_2$	Operação de interseção sobre as linguagens K_1 e K_2
$K_1 \setminus K_2$	Operação de subtração de conjuntos entre as linguagens K_1 e K_2
K_1K_2	Operação de concatenação entre as linguagens K_1 e K_2
$\overline{K}$	Fecho do prefixo da linguagem K
K^*	Fecho de Kleene da da linguagem K
$D(\cdot)$	Operação de dilatação
$R_D^{-1}(\cdot)$	Operação de contração
G	Autômato determinístico de estados finitos
X	conjunto de estados de um autômato
$f(\cdot,\cdot)$	Função de transição de um autômato
$\Gamma(\cdot)$	Conjunto de eventos ativos de em um estado de um autômato
x_o	Estado inicial de um autômato
X_m	Conjunto de estados marcados de um autômato
$L(G)$	Linguagem gerada por um autômato G
L	$L = L(G)$
$L_m(G)$	Linguagem marcada por um autômato G

L_m	$L_m = L_m(G)$
$Ac(\cdot)$	Operação de coacessibilidade
$Trim(\cdot)$	Operação *trim*
$P(\cdot)$	Projeção natural
$P^{-1}(\cdot)$	Projeção inversa
P_o	Projeção do conjunto de eventos Σ para o conjunto Σ_o
$P_{dil,o}$	Projeção do conjunto de eventos Σ_{dil} para o conjunto Σ_o
P_{nilo}	Projeção do conjunto de eventos Σ para o conjunto Σ_{nilo}
$P_{o,nilo}$	Projeção do conjunto de eventos Σ_o para o conjunto Σ_{nilo}
$P_{dil,nilo}$	Projeção do conjunto de eventos Σ_{dil} para o conjunto Σ_{nilo}
$UR(\cdot)$	Alcance não observável de um estado
$Obs(G, \Sigma_o)$	Observador do autômato G em relação a Σ_o
$G_1 \times G_2$	Composição produto entre os autômatos G_1 e G_2
$G_1 \|\| G_2$	Composição paralela entre os autômatos G_1 e G_2
S	Supervisor sob observação completa
S_P	Supervisor sob observação parcial (supervisor-P)
S_{rob}	Supervisor robusto (supervisor-R)
S/G	Sistema em malha fechada, S controlando G
K	Linguagem especificada para um SED
$K^{\uparrow C}$	Sublinguagem controlável suprema de K
$K^{\uparrow N}$	Sublinguagem normal suprema de K
$K^{\uparrow CO}$	Sublinguagem controlável e observável suprema de K
$K^{\uparrow CN}$	Sublinguagem controlável e normal suprema de K
$K_{sto}^{\uparrow CN}$	Sublinguagem de K controlável (em relação a L e Σ_{uc}) e normal (em relação a L e P_{nilo}) suprema
$D(K)^{\uparrow CN}$	Sublinguagem de $D(K)$ controlável (em relação a $D(L)$ e $\Sigma_{uc,dil}$) e normal (em relação a $D(L)$ e $P_{dil,o}$) suprema
$\exists$	Operador de existência
$\wedge$	Operador *e* lógico
$\vee$	Operador *ou* lógico
$A \subseteq B$	A está contido ou é igual a B
$A \supseteq B$	A contém ou é igual a B
$A \subset B$	A é um subconjunto próprio de B
$A \supset B$	B é um subconjunto próprio de A

Capítulo 1

Introdução

Nas últimas décadas, o crescimento do processo de automatização de sistemas industriais tem aumentado rapidamente a demanda por fundamentos teóricos que se apliquem a sistemas cada vez mais complexos e extensos de forma prática e adequada. Nos primórdios da teoria de controle, o enfoque dos trabalhos estava relacionado ao comportamento dinâmico de baixo nível do sistema estudado. Com o crescimento da complexidade dos sistemas, surgiu a necessidade de se estudar o comportamento dinâmico do sistema a um nível mais alto, que possibilitasse uma melhor compreensão do sistema como um todo e de como cada uma de suas partes interagem. Para uma classe considerável de sistemas, quando seu comportamento dinâmico é estudado a um nível mais alto, esse apresenta uma natureza lógica, na qual a dinâmica se torna praticamente regida pela ocorrência de eventos. Os sistemas cujas dinâmicas são comandadas pela ocorrência de eventos e que possuem espaços de estados discretos são denominados *Sitemas a Eventos Discretos* (SED).

Quando um SED é estudado a nível lógico, ou seja, quando o interesse principal está relacionado às sequências de eventos executadas pelo SED, esse sistema pode ter sua dinâmica modelada pelo conjunto de todas as sequências de eventos possíveis de serem geradas por ele. Um conjunto de sequências é denominado uma *linguagem*. Por sua vez, quando a linguagem gerada pelo SED não satisfizer algum interesse, a dinâmica do SED, ou seja, a linguagem gerada por ele, pode ser alterada por meio da inclusão de um controlador. O Controle Supervisório, inicialmente proposto por RAMADGE e WONHAM [1], exerce um controle realimentado sobre o SED, no sentido de restringir a linguagem gerada pelo SED a um subconjunto da linguagem gerada em malha aberta.

Ao se tratar de controle realimentado, a robustez é um dos principais objetivos. Dado um determinado sistema, às vezes, devido a incertezas nos parâmetros, imperícia no processo de modelagem ou presença de perturbações, não é possível determinar um modelo nominal suficientemente preciso para o sistema. O objetivo ao se projetar um controle supervisório robusto é alcançar o comportamento de-

sejado mesmo em presença de incertezas no modelo ou de perturbações. Na área de Sistemas a Eventos Discretos, o problema de robustez tem sido abordado sob diversos enfoques em controle supervisório [2–11] e diagnose de falhas [12–18].

O controle supervisório robusto de SED foi inicialmente proposto por LIN [2], que formulou o problema de se projetar um supervisor robusto capaz de prover o comportamento desejado, supondo observação parcial e considerando que o sistema a eventos discretos não é modelado por um único modelo nominal (autômato), mas por um conjunto de modelos (autômatos). O problema formulado em [2] foi estendido por BOURDON *et al.* [6] que consideraram uma especificação para cada possível modelo, mas supondo observação completa. Em seguida, SABOORI e HASHTRUDI-ZAD [9] revisitaram o trabalho de [6] no contexto de observação parcial.

Em um contexto diferente, CURY e KROGH [3] formularam um novo problema de controle supervisório robusto que consiste em projetar um supervisor não bloqueante, maximamente permissivo para o modelo nominal e que maximiza o conjunto de plantas que, sob a ação do controle supervisório projetado, atende às especificações desejadas. O problema proposto por [3] foi revisitado por TAKAI [5, 7] supondo, respectivamente, observação completa e parcial.

Ao invés de considerar um conjunto de plantas definidas sobre um alfabeto conhecido de eventos, como em [2, 3, 7], PARK e LIM [4, 19] propõem um novo caminho para modelar incertezas no comportamento do sistema, associando as incertezas a eventos desconhecidos e não observáveis, que são incluídos no modelo da planta pela adição de transições rotuladas por um evento Δ.

Mais recentemente, SANCHEZ e MONTOYA [10] e ROHLOFF [8, 11] consideraram o problema de controle supervisório seguro em presença de falhas de sensores, supondo que os sensores podem falhar a qualquer momento, mas quando uma falha ocorre ela é permanente, ou seja, quando um evento torna-se não-observável devido a uma falha do sensor que detecta a sua ocorrência, ele permanece não-observável para sempre.

Neste trabalho, será considerado o problema de controle supervisório robusto supondo a ocorrência de falhas intermitentes nos sensores associados às observações de um subconjunto dos eventos da planta. As perdas de observação podem ser decorrentes de falhas em sensores ou em canais de comunicação entre sensor e controlador.

Um sistema sujeito a perdas intermitentes de observação pode ser modelado por um conjunto de modelos da planta, como nos problemas propostos em [2] e [3]. No entanto, a suposição da ocorrência de perdas intermitentes gera um conjunto preestabelecido de modelos, o que diferencia o problema de perdas intermitentes de observação do problema proposto em [3], uma vez que nesse último deseja-se maximizar o conjunto de modelos. Por sua vez, no problema proposto em [2] a linguagem que modela o comportamento desejado em malha fechada deve estar contida

na linguagem de cada um dos supostos modelos da planta. No problema de controle supervisório em presença de perdas intermitentes de observação, essa imposição tornaria a solução excessivamente restritiva, uma vez que, ao se considerar a hipótese de que a linguagem gerada desejada em malha fechada deve estar contida na linguagem gerada por cada modelo, a linguagem gerada desejada em malha fechada não poderá conter sequências nas quais podem ocorrer perdas de observação. Embora a suposição feita em [2] tenha sido relaxada em [9], torna-se ainda necessário em [9] que sejam determinados os comportamentos desejados em malha fechada para cada possível planta, procedimento esse que implica em um acréscimo desnecessário na complexidade, quando o problema tratado é o de controle supervisório em presença de perdas intermitentes de observação. Por sua vez, na abordagem proposta em [4] e [19], quando se supõe que um estado possui incertezas associadas, para cada transição originada nesse estado é criada uma nova transição rotulada com Δ com mesmo estado de destino da transição original. Portanto, não é previsto em [4] e [19] o caso no qual somente uma parcela das transições de um estado possuem incertezas associadas.

O problema de controle supervisório robusto supondo perdas intermitentes de observação, considerado neste trabalho, é mais geral do que o considerado em [8, 10, 11], uma vez que perdas permanentes de observabilidade podem ser vistas como um caso particular de perdas intermitentes de observação.

Na teoria proposta neste trabalho, a definição de observabilidade será estendida para o conceito de observabilidade robusta em presença de perdas intermitentes de observação, usando o modelo proposto por CARVALHO *et al.* [16] no contexto de diagnose de falhas. Com a ajuda da operação *Dilatação* (também proposta em [16]), duas definições de observabilidade robusta serão propostas. A primeira delas, denominada *observabilidade robusta forte*, será feita de forma análoga à definição de diagnosticabilidade robusta proposta em [16]. A segunda definição, denominada *observabilidade robusta fraca*, levará em consideração a possível observação de um evento (após uma perda anterior da observação desse evento) para aumentar a permissividade da linguagem gerada pelo sistema em malha fechada. Com base na definição de controlabilidade e observabilidade robusta fraca, determinar-se-ão as condições necessárias e suficientes para a existência de um supervisor robusto em presença de perdas intermitentes de observação.

Esta dissertação está organizada da seguinte forma. Nos capítulos 2 e 3 serão apresentados os fundamentos teóricos aplicados no desenvolvimento desse trabalho. Especificamente, no capítulo 2 serão apresentados conceitos da teoria de sistemas a eventos discretos, como, por exemplo, linguagens e autômatos. Ainda no capítulo 2, será apresentada a modelagem proposta em [16] para acrescentar falhas intermitentes de observação em modelos de SEDs. No capítulo 3 serão apresentados con-

ceitos da teoria de controle supervisório de SEDs. No capítulo 4 serão apresentadas as contribuições desse trabalho. Inicialmente, o problema de controle supervisório robusto em presença de perdas intermitentes de observação será formulado e, em seguida, solucionado. Adicionalmente, serão apresentadas propriedades adicionais da operação de dilatação. Por fim, no capítulo 5 serão apresentadas as conclusões e serão sugeridas possíveis questões a serem abordadas em trabalhos futuros.

Capítulo 2

Sistemas a eventos discretos

Neste capítulo é apresentada uma breve revisão dos fundamentos da teoria de sistemas a eventos discretos. Inicialmente, na seção 2.1, o conceito de sistemas a eventos discretos é introduzido. Em seguida, na seção 2.2, são apresentados alguns conceitos da teoria de linguagens com o objetivo de aplicá-los para descrever o comportamento de sistemas a eventos discretos a nível lógico. As seções 2.3 e 2.4 são dedicadas à descrição da teoria de autômatos. Por fim, na seção 2.5, é apresentada uma forma de se modelar sistemas a eventos discretos sujeitos a perdas intermitentes de observações.

2.1 Modelos a eventos discretos

O desenvolvimento contínuo das áreas do conhecimento humano cria uma demanda de tarefas, cada vez mais complexas, que precisam ser executadas. Desta forma, a busca por ferramentas mais sofisticadas é constante nos diversos campos da ciência. Intuitivamente, um sistema pode ser definido como um conjunto de partes que atuam juntas na execução de uma tarefa; tarefa essa, que nenhuma das partes conseguiria executar separadamente. O conceito de sistema não é restrito a algo físico, podendo ser aplicado a fenômenos abstratos como, por exemplo, transações econômicas.

Em engenharia, ao se estudar um sistema, normalmente, deseja-se compreender as relações existentes entre um conjunto de variáveis mensuráveis. Essas variáveis são classificadas, usualmente, como variáveis de entrada ou de saída. As variáveis de entrada, agrupadas no vetor coluna $\mathbf{u}(t) = [u_1(t) \ \ldots \ u_p(t)]^T$, são aquelas que podem ser manipuladas diretamente. Por sua vez, as variáveis de saída, agrupadas no vetor coluna $\mathbf{y}(t) = [y_1(t) \ \ldots \ y_m(t)]^T$, são aquelas que, normalmente, podem ser medidas, mas não podem ser diretamente manipuladas.

Os sistemas nos quais os valores das saídas em um instante de tempo t_0 não dependem dos valores das entradas no período anterior a t_0 são chamados de sistemas estáticos, ou sistemas sem memória. Por sua vez, os sistemas nos quais os valores das saídas são consequência não só dos valores atuais das entradas, mas também

dos valores passados, são denominados sistemas dinâmicos. A informação do comportamento passado do sistema, necessária para determinar a saída em um instante de tempo, é sintetizada no conceito de estado.

O estado de um sistema dinâmico, em um instante de tempo t_0, é o menor conjunto de variáveis x_i, $i = 1, ..., n$, tais que o conhecimento dessas variáveis, uma vez conhecidas as entradas $\mathbf{u}(t)$, $\forall t \geq t_0$, determina completamente o comportamento do sistema para qualquer $t \geq t_0$. Por sua vez, o espaço de estados de um sistema, normalmente denotado por X, é o conjunto de todos os possíveis valores que o estado do sistema, $\mathbf{x}(t) = [x_1(t) \ \ldots \ x_n(t)]^T$, pode assumir.

Definição 1 (Sistemas a eventos discretos [20]) *Um sistema a eventos discretos (SED) é um sistema dinâmico de espaço de estados discreto, cujas transições de estado se dão por meio da ocorrência, em geral assíncrona, de eventos.*

Um evento pode ser visto como uma ocorrência instantânea que causa a transição do estado do sistema. Os eventos podem estar associados a uma ação específica como, por exemplo, o apertar de um botão ou um sensor de fim de curso, ou associados a uma ocorrência espontânea da natureza como, por exemplo, um computador ser desligado por alguma razão não descrita.

Na teoria de controle moderno [21], são estudados os chamados sistemas dinâmicos de variáveis contínuas (SDVC), ou seja, sistemas nos quais o espaço de estados é contínuo. Normalmente, a dinâmica de um SDVC é descrita por equações diferenciais (sistemas de tempo contínuo), ou por equações a diferença (sistemas de tempo discreto). Por sua vez, um sistema a eventos discretos (SED) é um sistema dinâmico de espaço de estados discreto e, portanto, requer uma modelagem distinta.

2.2 Linguagens

A nível lógico, ao se considerar a evolução dos estados de um SED, preocupa-se, principalmente, com a sequência de estados visitados e com os eventos que causaram as correspondentes transições de estado, isto é, o modelo de um SED é composto basicamente de dois elementos: estados e eventos. Todas as sequências de eventos possíveis de serem geradas por um SED caracterizam a linguagem desse sistema. Dessa forma, uma linguagem pode ser vista como uma maneira formal de descrever, a nível lógico, o comportamento de um SED. Com isso em mente, nesta seção, serão apresentados conceitos fundamentais da teoria de linguagens com o intuito de, posteriormente, aplicá-los ao controle supervisório de SEDs. A seguir, o conceito de linguagem é definido formalmente.

Definição 2 (Linguagem[20]) *Uma linguagem definida sobre um conjunto de eventos Σ (alfabeto) é um conjunto de sequências finitas (palavras), de comprimento arbitrariamente longo, formadas por eventos pertencentes a Σ.*

A operação de encadeamento de eventos e/ou sequências, denominada *concatenação*, é a operação básica de criação de uma sequência e, consequentemente, de uma linguagem. Por exemplo, a sequência abb é originada da concatenação da sequência ab com o evento b. Por sua vez, a sequência ab é resultado da concatenação dos eventos a e b. A sequência nula, simbolizada por ε, é o elemento neutro da concatenação, ou seja, $s\varepsilon = \varepsilon s = s$, para toda sequência s.

O *fecho de Kleene de um conjunto de eventos* Σ, simbolizado por Σ^*, é o conjunto de todas as sequências de comprimento finito formadas por eventos pertencentes a Σ, incluindo a sequência nula, ε. Qualquer linguagem definida sobre Σ é um subconjunto de Σ^*. Particularmente, $\emptyset$, Σ e Σ^* são linguagens definidas sobre Σ. Como exemplo ilustrativo, considere o conjunto de eventos $\Sigma = \{a, b\}$. Então, tem-se que:

$$\Sigma^* = \{\varepsilon, a, b, aa, ab, ba, bb, aaa, aab, aba, abb, baa, bab, bba, bbb, ...\}.$$

Antes da apresentação de algumas operações com linguagens, é necessário definir algumas terminologias relacionadas a sequências. Com essa finalidade, considere uma sequência s, particionada arbitrariamente da forma $s = tuv$, em que $t, u, v \in \Sigma^*$. As sequências t, u e v são *sub-sequências* de s. A sub-sequência t é um *prefixo* de s e, por sua vez, a sub-sequência v é um *sufixo* de s. O símbolo $|s|$ representa o *comprimento* de s, ou seja, o número de eventos que compõem a sequência s. Pode-se observar que ε e s são ambas prefixos, sub-sequências e sufixos de s.

2.2.1 Operações com linguagens

Por definição, as linguagens são conjuntos cujos elementos são sequências. Portanto, operações usuais da teoria de conjuntos, tais como, união, interseção, diferença e complemento em relação a Σ^* são aplicáveis às linguagens.

Exemplo 1 *Para ilustrar a aplicação das operações da teoria de conjuntos a linguagens, considere o alfabeto $\Sigma = \{a, b, c\}$ e as linguagens, definidas sobre esse alfabeto, $K_1 = \{\varepsilon, a, ab, abc\}$ e $K_2 = \{a, bc\}$. Dessa forma, tem-se que o complemento de K_2 em relação a Σ^*, a união e a interseção de K_1 e K_2 e a diferença de K_1 em relação*

a K_2 são, respectivamente,

$$\begin{aligned} K_2^C &= \{\varepsilon, b, c, aa, ab, ac, ba, bb, ca, cb, cc, aaa, ...\}, \\ K_1 \cup K_2 &= \{\varepsilon, a, ab, bc, abc\}, \\ K_1 \cap K_2 &= \{a\}, \\ K_1 \setminus K_2 &= \{\varepsilon, ab, abc\}. \end{aligned}$$

Além das operações usuais de conjuntos, é comum a aplicação de outras operações a linguagens, como, por exemplo, as operações de concatenação, fecho do prefixo e fecho de Kleene. Essas operações são descritas a seguir.

Concatenação. Sejam as linguagens K_1 e K_2 definidas sobre Σ. A concatenação de K_1 e K_2 é definida como

$$K_1 K_2 := \{s \in \Sigma^* : (\exists s_1 \in K_1)(\exists s_2 \in K_2)[s = s_1 s_2]\}.$$

Ou seja, uma sequência pertence a $K_1 K_2$ se essa sequência puder ser formada por meio da concatenação de uma sequência pertencente a K_1 e uma sequência pertencente a K_2.

Fecho do prefixo. Seja uma linguagem $K \subseteq \Sigma^*$. O fecho do prefixo de K é a linguagem

$$\overline{K} := \{s \in \Sigma^* : (\exists t \in \Sigma^*)[st \in K]\}.$$

Em palavras, $\overline{K}$ é formada por todos os prefixos de todas as sequências pertencentes a K. Dessa forma, $K \subseteq \overline{K}$. A linguagem K é dita ser prefixo fechada se $K = \overline{K}$.

Fecho do Kleene. Seja uma linguagem $K \subseteq \Sigma^*$. O fecho de Kleene de K é a linguagem

$$K^* = \{\varepsilon\} \cup K \cup KK \cup KKK \cup \cdots$$

Dessa forma, o fecho de Kleene de K é o conjunto formado por todas as possíveis concatenações entre as sequências pertencentes a K.

Exemplo 2 *Considere o alfabeto Σ e as linguagens K_1 e K_2 dados no exemplo 1. Dessa forma, pode-se observar que, nesse caso, K_1 é prefixo fechada, ou seja, $K_1 = \overline{K}_1$. Além disso, a concatenação de K_1 com K_2, o fecho do prefixo e o fecho de Kleene de K_2 são, respectivamente,*

$$\begin{aligned} K_1 K_2 &= \{a, aa, bc, aba, abc, abca, abbc, abcbc\}, \\ \overline{K}_2 &= \{\varepsilon, a, b, bc\}, \\ K_2^* &= \{\varepsilon, a, aa, bc, aaa, abc, bca, aaaa, aabc, abca, bcaa, bcbc, ...\}. \end{aligned}$$

Em adição às operações já descritas nesta seção, a projeção natural e a projeção inversa são outras operações frequentemente aplicadas a sequências e linguagens. A seguir, essas operações são, inicialmente, definidas para sequências e, em seguida, estendidas para linguagens.

Projeção natural [1]. Sejam dois conjuntos de eventos Σ_p e Σ_g, tais que $\Sigma_p \subset \Sigma_g$. A projeção natural P para uma sequência $s \in \Sigma_g^*$ é o mapeamento:

$$\begin{array}{rcl} P : \Sigma_g^* & \longrightarrow & \Sigma_p^* \\ s & \longmapsto & P(s) \end{array}$$

definido da seguinte forma:

$$\begin{array}{rcl} P(\varepsilon) & := & \varepsilon; \\ P(\sigma) & := & \begin{cases} \sigma, & \text{se } \sigma \in \Sigma_p; \\ \varepsilon, & \text{se } \sigma \in \Sigma_g \setminus \Sigma_p; \end{cases} \\ P(s\sigma) & := & P(s)P(\sigma), \text{ para } s \in \Sigma_g^* \text{ e } \sigma \in \Sigma_g. \end{array}$$

Por sua vez, a projeção natural P de uma linguagem $K \subseteq \Sigma_g^*$ é o conjunto formado pelas projeções de todas as sequências pertencentes a K, ou seja:

$$P(K) := \left\{ t \in \Sigma_p^* : (\exists s \in K)[P(s) = t] \right\}.$$

No decorrer deste trabalho, os termos projeção e projeção natural serão utilizados indiscriminadamente.

Projeção inversa. Sejam os conjuntos de eventos Σ_p e Σ_g definidos anteriormente. A projeção inversa P^{-1} para um sequência $t \in \Sigma_p^*$ é o mapeamento:

$$\begin{array}{rcl} P^{-1} : \Sigma_p^* & \longrightarrow & \Sigma_g^* \\ t & \longmapsto & P^{-1}(t) \end{array}$$

definido como:

$$P^{-1}(t) := \left\{ s \in \Sigma_g^* : P(s) = t \right\}.$$

Por fim, a projeção inversa P^{-1} de uma linguagem $K_p \subseteq \Sigma_p^*$ é a união das projeções inversas de todas as sequências pertencentes a K_p, ou seja:

$$P^{-1}(K_p) := \left\{ s \in \Sigma_g^* : (\exists t \in K_p)[P(s) = t] \right\}.$$

Exemplo 3 *Para ilustrar a aplicação das projeções direta e inversa, considere os conjuntos de eventos* $\Sigma_g = \{a, b, c\}$ *e* $\Sigma_p = \{a\}$ *e as linguagens* $K_g = \{a, b, c, aab, aba, aca, aaabc\}$ *e* $K_p = \{a\}\{a\}^*$ *definidas sobre* Σ_g *e* Σ_p*, respectiva-*

mente. Dessa forma,

$$\begin{aligned} P(K_g) &= \{\varepsilon, a, aa, aaa\}, \\ P^{-1}(K_p) &= \{b, c\}^*\{a\}\{a, b, c\}^*. \end{aligned}$$

As operações de projeção e projeção inversa são exaustivamente utilizadas no estudo de controle supervisório de SEDs. Por esse motivo, são listadas, a seguir, algumas propriedade relacionadas a essas duas operações. Sendo K,K_1 e K_2 linguagens, as seguintes propriedades são válidas [20]:

1. $P[P^{-1}(K)] = K$,
   $K \subseteq P^{-1}[P(K)]$,

2. Se $K_1 \subseteq K_2$, então $P(K_1) \subseteq P(K_2)$ e $P^{-1}(K_1) \subseteq P^{-1}(K_2)$,

3. $P(K_1 \cup K_2) = P(K_1) \cup P(K_2)$,
   $P(K_1 \cap K_2) \subseteq P(K_1) \cap P(K_2)$,

4. $P^{-1}(K_1 \cup K_2) = P^{-1}(K_1) \cup P^{-1}(K_2)$,
   $P^{-1}(K_1 \cap K_2) = P^{-1}(K_1) \cap P^{-1}(K_2)$,

5. $P(K_1K_2) = P(K_1)P(K_2)$,
   $P^{-1}(K_1K_2) = P^{-1}(K_1)P^{-1}(K_2)$.

2.2.2 Representações de linguagens

Como dito anteriormente, o uso de linguagens é uma maneira formal de descrever o comportamento de um SED. Por meio de uma linguagem é possível especificar todas as sequências de eventos capazes de serem geradas pelo SED. No entanto, nem sempre é fácil lidar diretamente com linguagens. Por exemplo, a linguagem $K_p = \{a\}\{a\}^*$, do exemplo anterior, pode ser descrita como o conjunto de todas as sequências pertencentes a $\Sigma_p^* = \{a\}^*$ com exceção da sequência nula. Porém, não é possível listar todas as sequências de K_p. Essa incapacidade evidencia a necessidade de um conjunto de ferramentas que permitam representar linguagens e que possam ser manipuladas por meio de operações bem definidas, de forma que seja possível construir, manipular e analisar linguagens arbitrariamente complexas.

Existem vários formalismos que podem ser usados para representar as linguagens de um SED, dentre os quais os mais difundidos são os Autômatos [20] e Redes de Petri [22, 23]. Na seção a seguir, são apresentados alguns tópicos da teoria de autômatos.

2.3 Autômatos

Autômatos são ferramentas com as quais é possível representar uma classe importante de linguagens de uma maneira formal e manipulável por meio de operações bem definidas.

Uma forma simples de descrever o comportamento de um autômato é a utilização de sua representação gráfica, também chamada de diagrama de transição de estados. O diagrama de transições de estados é um grafo no qual cada nó está associado a um estado do sistema e as transições estão associadas aos eventos que promovem as mudanças de estado, como ilustrado na figura 2.1.

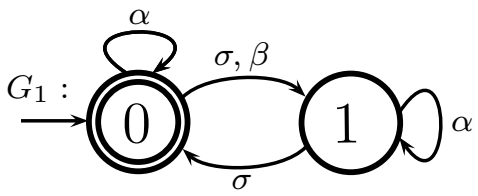


Figura 2.1: Exemplo de diagrama de transição de estados de um autômato.

Um autômato é dito ser determinístico se não há, partindo de um de seus estados, mais de uma transição rotulada com o mesmo evento. Por outro lado, autômatos não determinísticos possuem duas ou mais transições com mesmo rótulo que se originam em um mesmo estado. As definições formais de autômatos determinístico e não determinístico são apresentadas a seguir.

Definição 3 (Autômatos determinísticos e não determinísticos [24]) *Um autômato determinístico G é uma sêxtupla*

$$G = (X, \Sigma, f, \Gamma, x_0, X_m),$$

em que X é o conjunto de estados, Σ o conjunto finito de eventos associados a G, $f : X \times \Sigma \to X$ é a função de transição, $f(x, \sigma) = y$ significa que existe uma transição rotulada pelo evento σ do estado x para o estado y, $\Gamma : X \to 2^{\Sigma}$ é o conjunto de eventos ativos ($\Gamma(x)$ é o conjunto formado por todos os eventos $\sigma \in \Sigma$, para os quais $f(x, \sigma)$ é definida), x_0 é o estado inicial e X_m é o conjunto de estados marcados.

Um autômato não determinístico G_{nd} é uma sêxtupla

$$G_{nd} = (X, \Sigma \cup \{\varepsilon\}, f_{nd}, \Gamma, x_0, X_m),$$

em que, f_{nd} é a função $f_{nd} : X \times \Sigma \cap \{\varepsilon\} \to 2^X$, tal que, $f_{nd}(x, \sigma) \subseteq X$ quando definida e o estado inicial, x_0, é um subconjunto de X.

A definição 3 não determina que o conjunto de estados, X, seja finito. Quando X é um conjunto finito, o autômato é denominado *autômato de estados finitos*. Quando um autômato determinístico G possui um conjunto de estados, X, finito, G é denominado *autômato determinístico de estados finitos*. No decorrer deste trabalho, exceto quando indicado, o termo *autômato* será utilizado ao se referir a *autômatos determinísticos de estados finitos*.

Um autômato G opera da seguinte forma: ao inicializar, o autômato está no estado inicial x_0; quando ocorre algum evento $\sigma \in \Gamma(x_0) \subseteq \Sigma$, G faz uma transição para o estado $f(x_0, \sigma)$ e esse processo se repete de acordo com a forma como f é definida.

Por meio do diagrama de transição de estados de um autômato é possível inferir bastante informação relacionada aos elementos, $X, \Sigma, f, \Gamma, x_0, X_m$, que compõem esse autômato, conforme ilustrado no exemplo a seguir.

Exemplo 4 *Considere o autômato G_1, cujo diagrama de estados está representado na figura 2.1. O conjunto de estados de G_1 é $X = \{0, 1\}$, o estado inicial (indicado no diagrama de transição de estados por uma flecha) é $x_0 = 0$ e o conjunto de estados marcados (indicados por duplos círculos) é $X_m = \{0\}$. Pode-se inferir também que $\Sigma \supseteq \{\alpha, \beta, \sigma\}$. Além dos eventos presentes na figura 2.1, Σ pode conter outros eventos que não interferem diretamente na dinâmica de G_1, mas que podem influenciar quando G_1 for submetido a operações de composição de autômatos, apresentadas mais adiante. Por sua vez, de acordo com a figura 2.1, a função de transição de estados é definida para os seguintes pares pertencentes a $X \times \Sigma$: $f(0, \alpha) = 0$, $f(0, \sigma) = f(0, \beta) = 1$, $f(1, \alpha) = 1$ e $f(1, \sigma) = 0$ e, por fim, $\Gamma(0) = \{\alpha, \beta, \sigma\}$ e $\Gamma(1) = \{\alpha, \sigma\}$.*

2.3.1 Linguagens representadas por autômatos

Dado que o interesse é aplicar autômatos no estudo de linguagens de sistemas a eventos discretos, é importante estabelecer como autômatos e linguagens estão relacionados.

A linguagens representadas por um autômato podem ser analisadas por meio do diagrama de transição de estados desse autômato. O conjunto de todas as sequências de eventos que partem do estado inicial, x_0, é denominado *linguagem gerada* pelo autômato e, caso o autômato seja usado para representar o comportamento de um SED, constitui o conjunto de todas as possíveis sequências que o sistema pode realizar. Por sua vez, o subconjunto da linguagem gerada que corresponde às sequências que atingem um estado marcado $x_m \in X_m$ é denominado *linguagem marcada*. A decisão de quais estados devem ser marcados é um critério de projeto. Normalmente, os estados marcados simbolizam o término de uma tarefa ou uma situação crítica.

Quando uma linguagem pode ser marcada por um autômato de estados finitos, ela é denominada *regular*. Formalmente, as linguagens geradas e marcadas são definidas da forma a seguir.

Definição 4 (Linguagens gerada e marcada por um autômato) *Seja o autômato* $G = (X, \Sigma, f, \Gamma, x_0, X_m)$. *A* linguagem gerada *por* G *é*

$$L(G) := \{s \in \Sigma^* : f(x_0, s) \text{ é definida } \}.$$

A linguagem marcada *por* G *é*

$$L_m(G) := \{s \in L(G) : f(x_0, s) \in X_m\}.$$

Na definição 4, a função de transição f foi estendida do domínio $X \times \Sigma$ para o domínio $X \times \Sigma^*$, da seguinte maneira: *(i)* $f(x, \varepsilon) := x$; *(ii)* $f(x, s\sigma) := f(f(x, s), \sigma)$, para todo $s \in \Sigma^*$ e $\sigma \in \Sigma$. Uma consequência disso é que, para qualquer G com conjunto de estados X não vazio, $\varepsilon \in L(G)$.

Por meio do diagrama de transição de estados de um autômato, é possível caracterizar completamente suas linguagens gerada e marcada. Por exemplo, para o autômato G_2, cujo diagrama de transição de estados é apresentado na figura 2.2,

$$L(G_2) = [\{\beta\}\{\alpha\}^* \cup \{\beta\}\{\alpha\}^*\{\beta\}]^*$$

e

$$L_m(G_2) = [\{\beta\}\{\alpha\}^*\{\beta\}]^*.$$

2.3.2 Autômatos com bloqueio

Por definição, $L(G)$ é prefixo fechada. Assim, em conjunto com a definição de $L_m(G)$, tem-se que $\overline{L_m(G)} \subseteq L(G)$. Dessa forma, duas situações são possíveis: $L_m(G)$ pode ser um subconjunto próprio de $L(G)$, ou seja, $\overline{L_m(G)} \subset L(G)$ ou $\overline{L_m(G)} = L(G)$. A primeira situação, $\overline{L_m(G)} \subset L(G)$, ocorre por dois motivos:

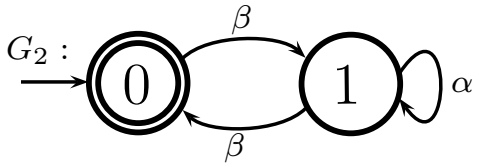


Figura 2.2: Diagrama de transição de estados do autômato G_2.

- G possui um estado x, para o qual $\Gamma(x) = \emptyset$ e $x \notin X_m$. Nesse caso, quando esse estado x é atingido, nenhum outro evento poderá ocorrer e, consequentemente, o sistema fica bloqueado e sem a possibilidade de alcançar estados marcados. Esse tipo de bloqueio é chamado de bloqueio definitivo (*deadlock*);

- G possui estados não marcados que formam um componente fortemente conexo (isto é, um conjunto de estados no qual, a partir de qualquer um desses estados, é possível alcançar todos os outros estados desse conjunto), mas sem nenhuma transição para fora desse conjunto. Se o sistema alcança um desses estados, ele ficará restrito aos estados pertencentes ao componente fortemente conexo e, portanto, não poderá mais alcançar estados marcados. Esse tipo de bloqueio é chamado de bloqueio ativo (*livelock*).

Os estados que, uma vez alcançados, não possibilitam que o sistema atinja estados marcados são chamados de *estados de bloqueio*. Com base no que foi apresentado acima, um autômato G possui estados de bloqueio se

$$\overline{L_m(G)} \subset L(G)$$

e G não possui estados de bloqueio quando

$$\overline{L_m(G)} = L(G).$$

2.4 Operações com autômatos

Nesta seção, são apresentadas as operações com autômatos mais usuais e seus efeitos sobre as linguagens representadas por esses. Inicialmente, são apresentadas operações que se aplicam a um único autômato com o objetivo de modificar o seu diagrama de transição de estados. Em seguida, são descritas as operações aplicadas a dois, ou mais, autômatos com o objetivo de combiná-los, de maneira que o resultado seja a composição dos comportamentos modelados por cada autômato individual. Essas operações são bastante úteis quando se deseja descrever sistema complexos por meio da composição de sistemas mais simples. Por fim, é apresentado um algoritmo com o qual é possível obter um autômato observador de $G = (X, \Sigma, f, \Gamma, x_0, X_m)$ em relação a um conjunto de eventos Σ_p ($\Sigma_p \subset \Sigma$).

2.4.1 Operações unárias

As operações unárias são aplicadas a um único autômato. Tais operações modificam o diagrama de transição de estados do autômato original, porém, não alteram o

conjunto de eventos, Σ, relacionado ao autômato. Nesse trabalho, são consideradas as operações unárias descritas a seguir.

Acessibilidade. Dado um autômato $G = (X, \Sigma, f, \Gamma, x_0, X_m)$, um estado $x \in X$ é *acessível* se existe uma sequência $s \in \Sigma^*$, tal que $f(x_0, s) = x$. Do contrário x é *não acessível*. A operação de acessibilidade elimina todos os estados não acessíveis de G. Formalmente, o autômato obtido pela operação de acessibilidade, denominado parte acessível de G, é definido da seguinte forma:

$$Ac(G) := (X_{ac}, \Sigma, f_{ac}, x_0, X_{m,ac}),$$

no qual,

$$\begin{aligned} X_{ac} &= \{x \in X : (\exists s \in \Sigma^*)[f(x_0, s) = x]\}; \\ X_{m,ac} &= X_m \cap X_{ac}; \\ f_{ac} &= f|_{X_{ac} \times \Sigma \to X_{ac}}. \end{aligned}$$

A notação $f|_{X_{ac} \times \Sigma \to X_{ac}}$ indica que f foi restringida ao domínio referente aos estados acessíveis, X_{ac}.

Claramente, a operação de acessibilidade não afeta as linguagens gerada e marcada pelo autômato.

Coacessibilidade. Dado um autômato $G = (X, \Sigma, f, \Gamma, x_0, X_m)$, um estado $x \in X$ é *coacessível* se existe uma sequência $s \in \Sigma^*$, tal que $f(x, s) \in X_m$. Do contrário, x é não *coacessível*. A operação de coacessibilidade elimina todos os estados não coacessíveis de G. Formalmente, o autômato obtido pela operação de coacessibilidade, denominado parte co-acessível de G, é definido da seguinte forma:

$$CoAc(G) := (X_{coac}, \Sigma, f_{coac}, x_{0,coac}, X_m),$$

no qual,

$$\begin{aligned} X_{coac} &= \{x \in X : (\exists s \in \Sigma^*)[f(x, s) \in X_m]\}; \\ x_{0,coac} &= \begin{cases} x_0, \text{ se } x_0 \in X_{coac}; \\ \text{não definido, caso contrário;} \end{cases} \\ f_{coac} &= f|_{X_{coac} \times \Sigma \to X_{coac}}. \end{aligned}$$

A operação de coacessibilidade reduz $L(G)$, pois estados acessíveis a partir de x_0 podem ser deletados. No entanto, a operação de coacessibilidade não afeta $L_m(G)$.

Se $G = CoAc(G)$, então G é dito ser coacessível e, nesse caso, $L(G) = \overline{L_m(G)}$. A coacessibilidade está intimamente ligada à existência de estados de bloqueio. Como apresentado na seção 2.3, um autômato G possuirá estados de bloqueio quando

$\overline{L_m(G)}$ for um subconjunto próprio de $L(G)$, ou seja, $\overline{L_m(G)} \subset L(G)$, mas $\overline{L_m(G)} \neq L(G)$. Claramente, os estados de bloqueio são estados acessíveis e não coacessíveis.

Operação de redução (trim). A operação *trim* nada mais é do que a aplicação sequencial das operações Ac e $CoAc$, ou seja, dado um autômato G,

$$Trim(G) := CoAc[Ac(G)] = Ac[CoAc(G)].$$

Quando $G = Trim(G)$, G é dito ser um autômato *trim*.

Projeções e projeções inversas. Dado um autômato $G = (X, \Sigma, f, \Gamma, x_0, X_m)$, um conjunto de eventos Σ_p, tal que $\Sigma_p \subset \Sigma$, e a projeção $P_p : \Sigma^* \to \Sigma_p^*$, um autômato que gera $P_p[L(G)]$ e marca $P_p[L_m(G)]$ pode ser obtido a partir de G, por meio da substituição de todas as transições rotuladas com eventos pertencentes a $\Sigma \setminus \Sigma_p$ por transições rotuladas com ε. No entanto, o autômato resultante desse procedimento é *não determinístico*. Uma outra forma de encontrar um autômato que gera $P_p[L(G)]$ e marca $P_p[L_m(G)]$ será apresentada na seção 2.4.3.

Considere, agora, as linguagens $L_p = L(G) \subseteq \Sigma_p^*$, $L_{m,p} = L_m(G)$, o conjunto de eventos Σ_g, tal que $\Sigma_g \supset \Sigma_p$, e a projeção $P : \Sigma_g^* \to \Sigma_p^*$. Então, o autômato que gera a projeção inversa $P^{-1}(L_p)$ e marca $P^{-1}(L_{m,p})$ pode ser obtido adicionando-se autolaços rotulados por todos os eventos pertencentes a $\Sigma_g \setminus \Sigma_p$ a todos os estados de G.

Exemplo 5 *Para ilustrar a aplicação das operações de acessibilidade, coacessibilidade e trim, considere o autômato G_3, cujo diagrama de transição de estados está representado na figura 2.3(a). Como pode ser facilmente observado, o estado* 6 *de G_3 é não acessível e os estados* 3, 4 *e* 5 *são não coacessíveis. Dessa forma, os autômatos $Ac(G_3)$ e $CoAc(G_3)$ são aqueles ilustrados nas figuras 2.3(b) e 2.3(c), respectivamente. Por sua vez, o autômato $trim(G_3)$ terá o diagrama de estados apresentado na figura 2.3(d). Como esperado, os estados* 3, 4, 5 *e* 6 *não estão presentes em $trim(G_3)$.*

2.4.2 Operações de composição

As operações de composição são aplicadas a dois ou mais autômatos que operam concorrentemente com o objetivo de combinar os seus comportamentos. São apresentadas, aqui, duas operações de composição de autômatos: a composição produto e a composição paralela.

Para dois autômatos G_1 e G_2 cujos conjuntos de eventos são Σ_1 e Σ_2, respectivamente, as composições produto $(G_1 \times G_2)$ e paralela $(G_1||G_2)$ são formas distintas de se interconectar esses autômatos. A diferença básica entre as composições produto e paralela está relacionada à evolução dos *eventos particulares* de cada autômato,

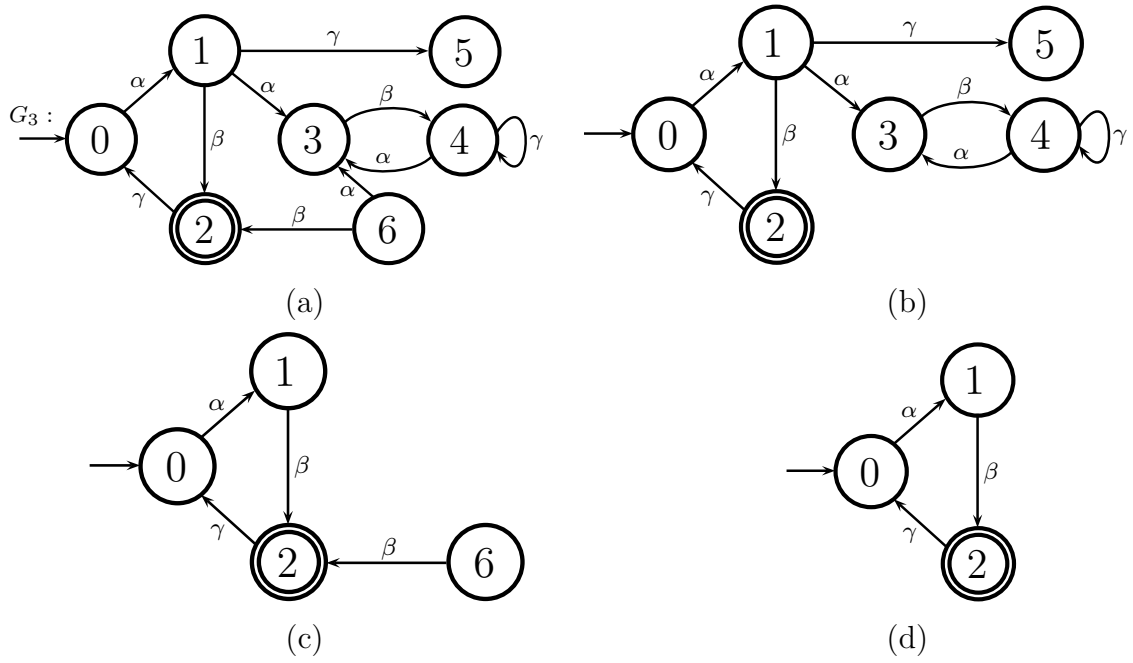


Figura 2.3: Diagrama de transição de estados do autômato G_3 (a); $Ac(G_3)$ (b); $CoAc(G_3)$ (c); e $trim(G_3)$ (d).

isto é, os eventos que não pertencem a $\Sigma_1 \cap \Sigma_2$. Na composição produto, o eventos particulares são impedidos de ocorrer, enquanto que, na composição paralela, esses eventos ocorrem livremente. Por sua vez, as ocorrências dos *eventos comuns*, isto é, aqueles que pertencem a $\Sigma_1 \cap \Sigma_2$, são sincronizadas pelas duas operações de composição.

Composição produto. Na composição produto, as transições dos dois autômatos devem sempre ser sincronizadas por um *evento comum*, isto é, um evento ocorre se, e somente se, ele ocorrer em ambos os autômatos. Dados os autômatos $G_1 = (X_1, \Sigma_1, f_1, \Gamma_1, x_{0_1}, X_{m_1})$ e $G_2 = (X_2, \Sigma_2, f_2, \Gamma_2, x_{0_2}, X_{m_2})$, tem-se que a composição produto é definida da seguinte forma:

$$G_1 \times G_2 := Ac\left(X_1 \times X_2, \Sigma_1 \cup \Sigma_2, f_{1\times 2}, \Gamma_{1\times 2}, (x_{0_1}, x_{0_2}), X_{m_1} \times X_{m_2}\right),$$

em que,

$$f_{1\times 2}((x_1, x_2), \sigma) := \begin{cases} (f_1(x_1, \sigma), f_2(x_2, \sigma)), \text{ se } \sigma \in \Gamma_1(x_1) \cap \Gamma_2(x_2) \\ \text{não definido, caso contrário.} \end{cases}$$

Portanto, $\Gamma_{1\times 2}(x_1, x_2) = \Gamma_1(x_1) \cap \Gamma_2(x_2)$. Por fim, pode-se inferir que as linguagens marcadas e geradas pelo produto são, respectivamente:

$$\begin{aligned} L(G_1 \times G_2) &= L(G_1) \cap L(G_2); \\ L_m(G_1 \times G_2) &= L_m(G_1) \cap L_m(G_2). \end{aligned}$$

Composição paralela. Na composição paralela, um *evento comum* só pode ser executado se ambos os autômatos o executarem de maneira síncrona. Por outro lado, os *eventos particulares* não estão sujeitos a tal restrição e podem ser executados quando possíveis no autômato ao qual estão relacionados. A composição paralela de dois autômatos $G_1 = (X_1, \Sigma_1, f_1, \Gamma_1, x_{0_1}, X_{m_1})$ e $G_2 = (X_2, \Sigma_2, f_2, \Gamma_2, x_{0_2}, X_{m_2})$ é definida da seguinte forma:

$$G_1||G_2 := Ac\left(X_1 \times X_2, \Sigma_1 \cup \Sigma_2, f_{1||2}, \Gamma_{1||2}, (x_{0_1}, x_{0_2}), X_{m_1} \times X_{m_2}\right),$$

em que,

$$f_{1||2}((x_1, x_2), \sigma) := \begin{cases} (f_1(x_1, \sigma), f_2(x_2, \sigma)), \text{ se } \sigma \in \Gamma_1(x_1) \cap \Gamma_2(x_2) \\ (f_1(x_1, \sigma), x_2), \text{ se } \sigma \in \Gamma_1(x_1) \setminus \Sigma_2 \\ (x_1, f_2(x_2, \sigma)), \text{ se } \sigma \in \Gamma_2(x_2) \setminus \Sigma_1 \\ \text{não definido, caso contrário.} \end{cases}$$

Não é difícil observar que $\Gamma_{1||2}(x_1, x_2) = [\Gamma_1(x_1) \cap \Gamma_2(x_2)] \cup [\Gamma_1(x_1) \setminus \Sigma_2] \cup [\Gamma_2(x_2) \setminus \Sigma_1]$. Por sua vez, pode-se inferir que as linguagens marcadas e geradas pela composição paralela dos autômatos são, respectivamente:

$$\begin{aligned} L(G_1||G_2) &= P_1^{-1}[L(G_1)] \cap P_2^{-1}[L(G_2)], \\ L_m(G_1||G_2) &= P_1^{-1}[L_m(G_1)] \cap P_2^{-1}[L_m(G_2)], \end{aligned}$$

em que $P_i : (\Sigma_1 \cup \Sigma_2)^* \to \Sigma_i^*$, para $i = 1, 2$.

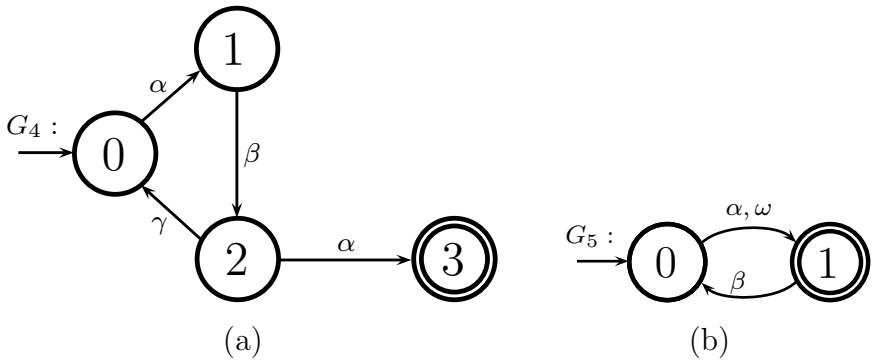
Por fim, diretamente das definições das composições produto e paralela, pode-se observar que essas operações são associativas e, portanto, pode-se definir que:

$$\begin{aligned} G_1 \times G_2 \times G_3 &:= (G_1 \times G_2) \times G_3 = G_1 \times (G_2 \times G_3); \\ G_1||G_2||G_3 &:= (G_1||G_2)||G_3 = G_1||(G_2||G_3). \end{aligned}$$

Exemplo 6 *Para ilustrar a aplicação das composição produto e paralela, considere os autômatos* $G_4 = (X_4, \Sigma_4, f_4, \Gamma_4, x_{0_4}, X_{m_4})$*, ilustrado na figura 2.4(a), e* $G_5 = (X_5, \Sigma_5, f_5, \Gamma_5, x_{0_5}, X_{m_5})$*, ilustrado na figura 2.4(b), sendo que* $\Sigma_4 = \{\alpha, \beta, \gamma\}$ *e* $\Sigma_5 = \{\alpha, \beta, \omega\}$*. Os resultados das composições produto e paralela desses autômatos são apresentados nas figuras 2.5(a) e 2.5(b), respectivamente.*

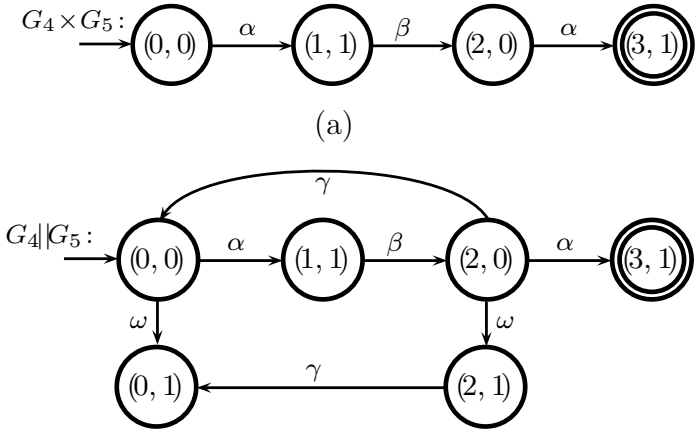
2.4.3 Autômato observador

Dado um autômato $G = (X, \Sigma, f, \Gamma, x_0, X_m)$. O *observador de* G em relação a um conjunto de eventos Σ_o ($\Sigma_o \subset \Sigma$), denotado por $Obs(G, \Sigma_o)$, é um autômato



(a) (b)

Figura 2.4: Diagramas de transição de estados dos autômatos G_4 (a) e G_5 (b).



(a)

(b)

Figura 2.5: Diagramas de transição de estados dos autômatos $G_4 \times G_5$ (a) e $G_4||G_5$ (b).

determinístico que gera e marca, respectivamente, $P_o[L(G)]$ e $P_o[L_m(G)]$, em que P_o é a projeção $P_o : \Sigma^* \to \Sigma_o^*$. O autômato $Obs(G, \Sigma_o)$ é definido da seguinte forma:

$$Obs(G, \Sigma_o) = (X_{obs}, \Sigma_o, f_{obs}, \Gamma_{obs}, x_{0,obs}, X_{m,obs})$$

sendo $X_{obs} \in 2^X$ e $X_{m,obs} = \{B \in X_{obs} : B \cap X_m \neq \emptyset\}$. Por sua vez, para definir $x_{0,obs}$, f_{obs} e Γ_{obs} é necessário introduzir o conceito de *alcance não-observável de um estado* $x \in X$, denotado por $UR(x, \Sigma_o)$:

$$UR(x, \Sigma_o) := \{y \in X : (\exists t \in (\Sigma \setminus \Sigma_o)^*)[f(x, t) = y]\} \tag{2.1}$$

e, analogamente, para todo $B \in 2^X$, tem-se que:

$$UR(B, \Sigma_o) := \bigcup_{x \in B} UR(x, \Sigma_o). \tag{2.2}$$

Utilizando (2.1) e (2.2), pode-se definir $x_{0,obs}$, f_{obs} e Γ_{obs} de acordo com o algoritmo a seguir.

Algoritmo 1 (Construção de observadores)

Passo 1: *Defina* $x_{0,obs} = UR(x_0, \Sigma_o)$. *Faça* $X_{obs} = \{x_{0,obs}\}$ *e* $\tilde{X}_{obs} = X_{obs}$.

Passo 2: *Faça* $\hat{X}_{obs} = \tilde{X}_{obs}$ *e* $\tilde{X}_{obs} = \emptyset$.

Passo 3: *Para cada* $B \in \hat{X}_{obs}$,

- $\Gamma_{obs}(B) = \left(\bigcup_{x \in B} \Gamma(x)\right) \cap \Sigma_o$;
- *Para cada* $\sigma \in \Gamma_{obs}(B)$,

$$f_{obs}(B, \sigma) = UR\left(\{x \in X : (\exists y \in B)[x = f(y, \sigma)]\}, \Sigma_o\right);$$

- $\tilde{X}_{obs} \leftarrow \tilde{X}_{obs} \cup f_{obs}(B, \sigma)$.

Passo 4: $X_{obs} \leftarrow \tilde{X}_{obs} \cup X_{obs}$.

Passo 5: *Repita os passos 2 a 4 até que toda a parte acessível de* $Obs(G, \Sigma_p)$ *tenha sido construída.*

Passo 6: $X_{m,obs} = \{B \in X_{obs} : B \cap X_m \neq \emptyset\}$.

Exemplo 7 *Para ilustrar a utilização do algoritmo 1, considere o autômato* G_4 *representado na figura 2.4(a) e* $\Sigma_o = \{\beta, \gamma\}$. *Para se calcular* $Obs(G_4, \Sigma_o)$ *utilizando o algoritmo 1, procede-se da seguinte forma:*
Passo 1: $x_{0,obs} = (0,1)$, $X_{obs} = \{(0,1)\}$ *e* $\tilde{X}_{obs} = X_{obs}$;
Passo 2: $\hat{X}_{obs} = \{(0,1)\}$ *e* $\tilde{X}_{obs} = \emptyset$;
Passo 3: $\Gamma_{obs}((0,1)) = \{\beta\}$, $f_{obs}((0,1), \beta) = (2,3)$ *e* $\tilde{X}_{obs} = \{(2,3)\}$;
Passo 4: $X_{obs} = \{(0,1), (2,3)\}$;
Passo 5: *Repita os passos 2 a 4;*
Passo 2: $\hat{X}_{obs} = \{(2,3)\}$ *e* $\tilde{X}_{obs} = \emptyset$;
Passo 3: $\Gamma_{obs}((2,3)) = \{\gamma\}$, $f_{obs}((2,3), \gamma) = (0,1)$ *e* $\tilde{X}_{obs} = \{(0,1)\}$;
Passo 4: $X_{obs} = \{(0,1), (2,3)\}$;
Passo 6: $X_{m,obs} = \{(2,3)\}$.
O autômato $Obs(G_4, \Sigma_o)$ *obtido pela aplicação do algoritmo 1 está representado na figura 2.6.*

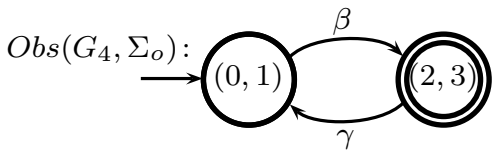


Figura 2.6: Diagramas de transição de estados do autômato $Obs(G_4, \Sigma_o)$.

2.5 Modelagem de sistemas a eventos discretos sujeitos a perdas intermitentes de observação

Nesta seção será considerada a abordagem proposta por CARVALHO *et al.* [16], com a qual se pode levar em conta possíveis perdas de observação decorrentes de falhas em sensores ou em canais de comunicação entre sensor e controlador em SEDs modelados por autômatos.

Nesse contexto, com intuito de acrescentar a modelagem de perdas intermitentes de observações diretamente ao modelo nominal (sem falhas) do sistema, CARVALHO *et al.* [16] definiu o seguinte operador:

Definição 5 (Dilatação [16]) *Seja o conjunto de eventos, Σ, particionado da forma $\Sigma = \Sigma_{ilo} \dot{\cup} \Sigma_{nilo} \dot{\cup} \Sigma_{uo}$, em que Σ_{ilo} é o conjunto de eventos passíveis de perdas intermitentes de observações, Σ_{nilo} é o conjunto de eventos sempre observáveis e Σ_{uo} é o conjunto de eventos não observáveis. Além disso, seja o conjunto de eventos não observáveis $\Sigma'_{ilo} = \{\sigma' : \sigma \in \Sigma_{ilo}\}$ e $\Sigma_{dil} = \Sigma \cup \Sigma'_{ilo}$. A dilatação, D, é o mapeamento:*

$$\begin{aligned} D : \Sigma^* &\longrightarrow 2^{\Sigma^*_{dil}} \\ s &\longmapsto D(s) \end{aligned}$$

no qual:

$$\begin{aligned} D(\varepsilon) &:= \{\varepsilon\}, \\ D(\sigma) &:= \begin{cases} \{\sigma\}, \text{ se } \sigma \in \Sigma \backslash \Sigma_{ilo}, \\ \{\sigma, \sigma'\}, \text{ se } \sigma \in \Sigma_{ilo}, \end{cases} \\ D(s\sigma) &:= D(s)D(\sigma),\ s \in \Sigma^*,\ \sigma \in \Sigma. \end{aligned}$$

A extensão do mapeamento D para uma linguagem $L \subseteq \Sigma^*$ é dada por:

$$D(L) := \bigcup_{s \in L} D(s).$$

O autômato determinístico G_{dil}, que modela o comportamento do sistema considerando possíveis ocorrências de perdas intermitentes de observação pode ser obtido a partir do autômato $G = (X, \Sigma, f, \Gamma, x_0, X_m)$, que modela o comportamento nomi-

nal da planta, adicionando-se, em paralelo às transições de G rotuladas com eventos pertencentes a Σ_{ilo}, novas transições rotuladas com os respectivos eventos pertencentes a Σ'_{ilo}. G_{dil} é formalmente definido como:

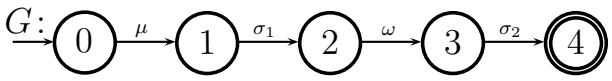
$$G_{dil} = (X, \Sigma_{dil}, f_{dil}, \Gamma_{dil}, x_0, X_m),$$

em que:

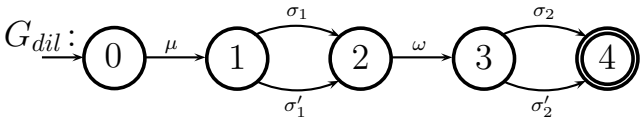
$$\begin{aligned} \Gamma_{dil}(x) &= D[\Gamma(x)] \\ f_{dil}(x, \sigma_{dil}) &= f(x, \sigma), \forall \sigma_{dil} \in \Gamma_{dil}(x) : \sigma_{dil} \in D(\sigma), \sigma \in \Sigma. \end{aligned}$$

Como provado em CARVALHO *et al.* [16], $L(G_{dil}) = D(L)$ e $L_m(G_{dil}) = D(L_m)$.

Note que o operador dilatação consiste em acrescentar transições rotuladas por σ' em paralelo com as transições rotuladas por eventos $\sigma \in \Sigma_{ilo}$. Por exemplo, considere o conjunto de eventos $\Sigma = \{\mu, \omega, \sigma_1, \sigma_2\}$, particionado de tal forma que $\Sigma_{uo} = \{\mu\}$, $\Sigma_{ilo} = \{\sigma_1, \sigma_2\}$ e $\Sigma_{nilo} = \{\omega\}$, e as linguagens L e L_m gerada e marcada, respectivamente, pelo autômato G representado na figura 2.7(a). O autômato G_{dil} (representado na figura 2.7(b)) que gera e marca, respectivamente, $D(L)$ e $D(L_m)$ pode ser obtido do autômato G acrescentando-se as transições $f_{dil}(1, \sigma'_1) = 2$ e $f_{dil}(3, \sigma'_2) = 4$.



(a)



(b)

Figura 2.7: Autômato G que gera e marca, respectivamente, L e L_m (a) e autômato G_{dil} que gera e marca, respectivamente, $D(L)$ e $D(L_m)$ (b).

Capítulo 3

Controle supervisório

Na teoria de controle clássica, sistemas de controle realimentados são comumente utilizados para modificar o comportamento de uma determinada planta. De forma semelhante, na teoria de sistemas a eventos discretos, um controle realimentado pode ser utilizado para alterar o comportamento de um SED. Seja o comportamento não controlado do SED modelado por um autômato G. Quando a linguagem $L(G)$ contiver sequências de eventos que violam *especificações* (condições) que se deseja impor ao comportamento do sistema, será possível restringir o comportamento do SED a um subconjunto de $L(G)$, por meio da aplicação de um controle realimentado. Esse controle realimentado recebe o nome de *supervisor* (supervisor S).

As especificações são definidas de forma a impedir a ocorrência de sequências indesejadas, tais como, sequências que levam a situações de risco, estados de bloqueio, estados que são fisicamente inadmissíveis, como, por exemplo, a superlotação de um *buffer* ou a colisão de máquinas; ou, até mesmo, para se estabelecer a ordem correta da execução de tarefas. Dessa forma, é estabelecida uma sublinguagem de $L(G)$ que representa o comportamento máximo permitido, denominado *comportamento legal*, L_a, para o sistema controlado. Quando o comportamento controlado está contido no comportamento legal, ele é denominado *seguro*. Em adição à busca por um comportamento controlado seguro, normalmente, deseja-se que esse comportamento seja o mais *permissivo* possível, ou seja, seja a maior sublinguagem de L_a possível.

Os fundamentos da teoria de controle supervisório foram desenvolvidos inicialmente por W. M. Wonham e P. J. Ramadge, e seus co-autores (veja [25] e suas referências) e, desde então, vários outros pesquisadores têm contribuído no desenvolvimento desse campo de pesquisa em SEDs. No paradigma de controle supervisório, é considerado que o supervisor S observa alguns eventos (possivelmente todos) executados por G. Então, S determina quais eventos, dentre os eventos ativos de G, podem ser executados por G. Em outras palavras, S tem a capacidade de desabilitar alguns eventos (mas não necessariamente todos) possíveis de serem executados por G. Duas considerações importantes podem ser feitas a partir desse paradigma.

Primeiro, S pode estar limitado a observar um subconjunto dos eventos relacionados a G (eventos observáveis). Segundo, S pode estar limitado a desabilitar somente um subconjunto dos eventos relacionados a G (eventos controláveis).

O restante deste capítulo está organizado da seguinte forma. Na seção 3.1 o problema de controle supervisório é definido formalmente e é apresentado o conceito de controlabilidade de uma linguagem. Na seção 3.2 são apresentados o problema de controle supervisório sob observação parcial e os conceitos de observabilidade e normalidade de uma linguagem. Finalmente, na seção 3.3 é demonstrado o processo de realização de um supervisor.

3.1 Problema de controle supervisório

Neste trabalho, iremos considerar o problema de controle supervisório sem bloqueio, isto é, desconsiderando a linguagem marcada pelo autômato. Por conseguinte, os conjuntos de estados marcados dos autômatos serão desconsiderados e, dessa forma, um autômato G será definido pela quíntupla $G = (X, \Sigma, f, \Gamma, x_0)$.

Considere um SED modelado por uma linguagem L definida sobre um conjunto de eventos Σ, em que $L = \overline{L}$ é o conjunto de todas as sequências que o SED pode gerar. Sem perda de generalidade, suponha que L seja a linguagem gerada pelo autômato $G = (X, \Sigma, f, \Gamma, x_0)$, ou seja, $L = L(G)$. Então, o problema de controle supervisório pode ser formulado da forma a seguir. Dado um autômato G, obter um supervisor S para interagir com G de forma realimentada, como ilustrado na figura 3.1, de tal forma que o sistema realimentado, S/G (lê-se S controlando G) gere a linguagem $L(S/G) = L_a \subseteq L$.

Formalmente, um supervisor S é uma função da linguagem gerada por G no conjunto potência de Σ, definida da forma a seguir:

$$\begin{aligned} S : L(G) &\longrightarrow 2^{\Sigma} \\ s &\longmapsto S(s) \end{aligned}$$

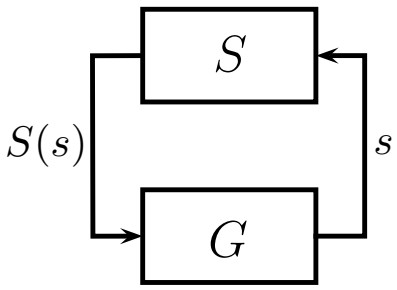


Figura 3.1: Estrutura realimentada do controle supervisório.

tal que o novo conjunto de eventos ativos $\Gamma_N[f(x_0, s)]$ que G pode executar no estado $f(x_0, s)$ seja $\Gamma_N[f(x_0, s)] = \Gamma[f(x_0, s)] \cap S(s)$. Em palavras, G não pode executar um evento no estado $f(x_0, s)$ se esse evento não pertencer a $S(s)$. O conjunto $S(s)$ é a ação de controle do supervisor S para a sequência s, enquanto S é a lei de controle.

O sistema controlado, S/G, é um autômato e sua linguagem gerada pode ser definida recursivamente da seguinte forma:

(i) $\varepsilon \in L(S/G)$;

(ii) $s\sigma \in L(S/G) \Leftrightarrow (s \in L(S/G)) \wedge (s\sigma \in L(G)) \wedge (\sigma \in S(s))$.

Suponha agora, que o conjunto de eventos Σ seja particionado em dois subconjuntos disjuntos, da seguinte forma:

$$\Sigma = \Sigma_c \dot{\cup} \Sigma_{uc},$$

em que, Σ_c é o *conjunto de eventos controláveis*, ou seja, os eventos que podem ser desabilitados por S e Σ_{uc} é o *conjunto de eventos não-controláveis*, que, por sua vez, não podem ter suas ocorrências impedidas pelo supervisor S. Levando-se em consideração essa partição de Σ, um supervisor S é denominado *admissível* se, para todo $s \in L(G)$

$$\Sigma_{uc} \cap \Gamma[f(x_0, s)] \subseteq S(s).$$

Em palavras, o supervisor S não pode desabilitar eventos não-controláveis ativos. De agora em diante, serão considerados somente supervisores admissíveis.

3.1.1 Controlabilidade de uma linguagem

Nesta subseção é apresentada uma condição necessária e suficiente para que um determinado comportamento possa ser obtido em malha fechada, quando um SED, modelado por um autômato G, é controlado por um supervisor S, considerando que todos os eventos executados por G são observados por S.

Definição 6 (Controlabilidade) *Sejam as linguagens K e L definidas sobre um conjunto de eventos Σ, em que $K \subseteq L$ e $L = \overline{L}$, e suponha que o conjunto de eventos seja particionado como $\Sigma = \Sigma_c \dot{\cup} \Sigma_{uc}$. K é controlável em relação a L e Σ_{uc} se*

$$\overline{K}\Sigma_{uc} \cap L \subseteq \overline{K}.$$

Em palavras, para toda sequência s pertencente a $\overline{K}$ e um evento σ_{uc} pertencente a Σ_{uc}, se K for controlável, então, se a sequência $s\sigma_{uc}$ pertencer a L implica que ela também pertencerá a $\overline{K}$. Por definição, a controlabilidade é um propriedade do fecho do prefixo, ou seja, K é controlável se, e somente se, $\overline{K}$ for controlável.

Quando as linguagens K e L são regulares, é possível verificar se K é controlável por meio de operações com autômatos da forma a seguir. Sejam K e L as linguagens geradas pelos autômatos H e G, respectivamente. Calculado o produto $H \times G$, a verificação da controlabilidade é feita comparando os eventos ativos de cada estado de $H \times G$ com os do seu respectivo estado em G (dado pelo segundo componente do estado de $H \times G$). Se houver algum evento não-controlável que aparecer em um estado de G e não aparecer no respectivo estado de $H \times G$, então K não será controlável.

A verificação da controlabilidade de uma linguagem torna-se importante, pelo fato da existência de um supervisor que gere essa linguagem estar condicionada à controlabilidade, conforme mostrado no teorema a seguir.

Teorema 1 (Teorema da controlabilidade [20]) *Considere um SED descrito pelo autômato* $G = (X, \Sigma, f, \Gamma, x_0)$*, no qual* $\Sigma = \Sigma_c \dot{\cup} \Sigma_{uc}$*. Seja* K *uma sublinguagem de* $L(G)$*, em que* $K \neq \emptyset$*. Então, existe um supervisor* S*, tal que* $L(S/G) = \overline{K}$ *se, e somente se,* K *é controlável em relação a* $L(G)$ *e* Σ_{uc}*.*

Quando uma linguagem K não é controlável, uma alternativa possível é buscar a "maior" sublinguagem de K que seja controlável, em que "maior" está relacionado à inclusão de conjuntos. Essa sublinguagem é denominada *sublinguagem controlável suprema* e é denotada por $K^{\uparrow C}$. Pode-se provar que a sublinguagem controlável suprema sempre existe, ao se considerar a seguinte propriedade da controlabilidade:

- Considere duas sublinguagens K_1 e K_2 de uma linguagem L ($L = \overline{L}$), todas definidas sobre um conjunto de eventos Σ, tal que $\Sigma = \Sigma_c \dot{\cup} \Sigma_{uc}$. Se K_1 e K_2 forem controláveis em relação a L e Σ_{uc}, então $K_1 \cup K_2$ também será controlável em relação a L e Σ_{uc}.

No pior caso $K^{\uparrow C} = \emptyset$. Por outro lado, se K for controlável, então $K^{\uparrow C} = K$.

3.2 Controle supervisório sob observação parcial

Na seção 3.1 foi considerado que as ocorrências de todos os eventos executados pelo SED eram observadas pelo supervisor S. Agora, considere que o conjunto de eventos Σ seja particionado, em relação a observabilidade, em dois subconjuntos disjuntos $\Sigma = \Sigma_o \dot{\cup} \Sigma_{uo}$, em que Σ_o é o *conjunto de eventos observáveis*, ou seja, os eventos que podem ser observados por S e Σ_{uo} é o *conjunto de eventos não-observáveis*, que, por sua vez, não podem ter suas ocorrências diretamente observadas pelo supervisor S. A principal causa de não-observabilidade de um evento é a ausência de um sensor capazes de detectar a ocorrência desse evento.

Levando em consideração a partição de Σ em relação à observabilidade, quando o sistema executa uma sequência s, o supervisor observa a projeção $P_o(s)$, em que $P_o : \Sigma^* \to \Sigma_o^*$. Nesse caso, quando duas sequências s_1 e s_2 possuem a mesma projeção, ou seja, $P_o(s_1) = P_o(s_2)$, o supervisor não tem como distinguir qual das duas sequências ocorreu e, portanto, deve tomar a mesma decisão para ambas. Dessa forma, ao invés do supervisor tomar sua decisão com base em $L(G)$, agora, a decisão deve ser tomada com base em $P_o[L(G)]$, como é ilustrado na figura 3.2. Formalmente, um supervisor sob observação parcial, denominado *supervisor-P* ou S_P, é uma função:

$$\begin{aligned} S_P : P_o[L(G)] &\longrightarrow 2^{\Sigma} \\ P_o(s) &\longmapsto S_P[P_o(s)] \end{aligned}$$

em que $S_P[P_o(s)]$ é tal que o novo conjunto de eventos ativos será $\Gamma_N[f(x_0, s)] = \Gamma[f(x_0, s)] \cap S_P[P_o(s)]$.

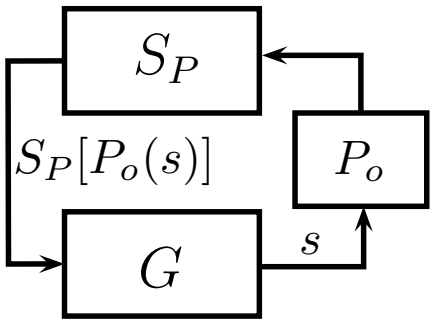


Figura 3.2: Estrutura realimentada do controle supervisório sob observação parcial.

A condição de admissibilidade para o supervisor-P é mais elaborada devido à existência de sequências de eventos não-observáveis; uma nova decisão é tomada pelo supervisor-P somente quando um evento observável ocorre e essa decisão não se altera até outra ocorrência de um evento observável. Seja, conforme definido em [20],

$$L_t = P_o^{-1}(t')\{\sigma\}(S_P(t) \cap \Sigma_{uo})^* \cap L,$$

em que $t = t'\sigma \in P_o(L)$ e $\sigma \in \Sigma_o$. Então, S_P será admissível se

$$\Sigma_{uc} \cap \left[\bigcup_{s \in L_t} \Gamma[f(x_0, s)] \right] \subseteq S_P(t).$$

O comportamento em malha fechada, isto é, a linguagem gerada por S_P/G é definida recursivamente como:

(i) $\varepsilon \in L(S_P/G)$;

(ii) $s\sigma \in L(S_P/G) \Leftrightarrow (s \in L(S_P/G) \wedge (s\sigma \in L(G)) \wedge (\sigma \in S_P[P_o(s)])$.

3.2.1 Observabilidade de uma linguagem

Como será apresentado nesta subseção, sob observação parcial, a existência de um supervisor que leve a um determinado comportamento em malha fechada não está associada somente à controlabilidade, mas, também, à observabilidade da linguagem desejada. A observabilidade é definida formalmente da forma a seguir.

Definição 7 (Observabilidade) *Sejam as linguagens K e L definidas sobre um conjunto de eventos Σ, em que $K \subseteq L$ e $L = \overline{L}$, e suponha que o conjunto de eventos pode ser particionado como $\Sigma = \Sigma_c \dot{\cup} \Sigma_{uc} = \Sigma_o \dot{\cup} \Sigma_{uo}$. Seja ainda a projeção $P_o : \Sigma^* \to \Sigma_o^*$. Então, K será observável em relação a L, P_o e Σ_c se, para todo $s \in \overline{K}$ e todo $\sigma \in \Sigma_c$,*

$$(s\sigma \in L \setminus \overline{K}) \Rightarrow (\nexists s' \in \overline{K})[(P_o(s) = P_o(s')) \wedge (s'\sigma \in \overline{K})]$$

ou, de forma equivalente,

$$(s\sigma \in L \setminus \overline{K}) \Rightarrow P_o^{-1}[P_o(s)]\sigma \cap \overline{K} = \emptyset.$$

A verificação da observabilidade pode ser testada por algoritmos de complexidade polinomial [20, 26]. Sob observação parcial, a observabilidade, em conjunto com a controlabilidade, compõe a condição de existência de um supervisor capaz de gerar uma determinada linguagem. Essa condição é apresentada formalmente no teorema a seguir.

Teorema 2 (Teorema da controlabilidade e observabilidade[20]) *Considere um SED descrito pelo autômato $G = (X, \Sigma, f, \Gamma, x_0)$, no qual $\Sigma = \Sigma_c \dot{\cup} \Sigma_{uc} = \Sigma_o \dot{\cup} \Sigma_{uo}$. Seja ainda a projeção $P_o : \Sigma^* \to \Sigma_o^*$ e uma sublinguagem K de $L(G)$, em que $K \neq \emptyset$. Então, existirá um supervisor-P S_P, tal que $L(S_P/G) = \overline{K}$ se, e somente se, K for controlável em relação a $L(G)$ e Σ_{uc} e observável em relação $L(G)$, P_o e Σ_c.*

Ao contrário da controlabilidade, a observabilidade não é fechada em relação à união, ou seja, dadas duas linguagens K_1 e K_2 observáveis, $K_1 \cup K_2$ não é obrigatoriamente observável. Por conseguinte, nem sempre existirá uma sublinguagem observável suprema.

Na literatura, pode-se encontrar vários trabalho voltados ao cálculo de sublinguagens controláveis e observáveis menos restritivas, como, por exemplo [27–31].

3.2.2 Normalidade de uma linguagem

Quando a linguagem desejada, K, não é controlável ou observável e uma vez que a observabilidade não é fechada em relação à união, a propriedade da *normalidade*

pode ser útil para solucionar o problema de se encontrar um supervisor-P que gere uma sublinguagem de K. No entanto, a solução não necessariamente será maximamente permissiva.

Definição 8 (Normalidade) *Sejam as linguagens K e L definidas sobre um conjunto de eventos Σ, em que $K \subseteq L$ e $L = \overline{L}$, e a projeção $P_o : \Sigma^* \to \Sigma_o^*$. Então, K será normal em relação a L e P_o se*

$$P_o^{-1}[P_o(\overline{K})] \cap L = \overline{K}.$$

Em palavras, uma linguagem K é normal quando $\overline{K}$ pode ser recuperada a partir da sua projeção, $P_o(\overline{K})$, e de L. Da mesma forma que a controlabilidade e a observabilidade, a normalidade é uma propriedade do fecho do prefixo de uma linguagem. Então, K será normal se, e somente se, $\overline{K}$ for normal. Além disso, como desejado, a normalidade é fechada em relação à união, ou seja, se $K_1, K_2 \subseteq L$ são normais com relação a L e P_o, então $K_1 \cup K_2$ também será. Como consequência, sempre existirá uma *sublinguagem controlável e normal suprema*, $K^{\uparrow CN}$. Em [26], é proposto um método de complexidade polinomial para o cálculo da sublinguagem controlável e normal suprema. No pior caso, $K^{\uparrow CN} = \emptyset$. Quando K for controlável e observável, tem-se que $K^{\uparrow CN} = K$.

Pode-se provar que, se K for normal em relação a L e P_o, então K será observável em relação a L, P_o e Σ_c, embora o contrário nem sempre seja verdadeiro. Portanto, de acordo com o teorema 2, se K for controlável e normal, então existirá um supervisor-P S_P, tal que $L(S_P/G) = \overline{K}$. Por fim, quando imposta a condição de que $\Sigma_c \subseteq \Sigma_o$, a normalidade torna-se equivalente à observabilidade, como assegurado pelo teorema a seguir.

Teorema 3 (Equivalência entre observabilidade e normalidade) *Suponha que $\Sigma_c \subseteq \Sigma_o$. Dessa forma, se K for controlável (em relação a L e Σ_{uc}) e observável (em relação a L, P_o e Σ_c), então K será normal (em relação a L e P_o).*

De acordo com o teorema 3, quando $\Sigma_c \subseteq \Sigma_o$, existirá a sublinguagem controlável e observável suprema, e esta será igual a sublinguagem controlável e normal suprema.

3.3 Realização de supervisores-P

Construído o autômato que modela uma especificação K controlável e observável, em que $K \subset L(G)$, torna-se necessário representar o supervisor-P de uma forma conveniente, uma vez que é impraticável listar a ação de controle $S_P(s_o)$ para toda sequência $s_o \in P_o[L(S_P/G)]$.

Dado que foram usados autômatos para representar o sistema (autômato G) e a especificação, é interessante considerar o uso de um autômato para representar, também, o supervisor. Quando K e $L(G)$ forem linguagens regulares, a lei de controle $S_P(s_o)$ poderá ser representada por um autômato de estados finitos. Além disso, a operação de composição paralela, apresentada na subseção 2.4.2, poderá ser aplicada para determinar o comportamento de S_P controlando G.

O autômato que representa o supervisor-P S_P, denominado *realização de* S_P e denotado pelo símbolo R_S, pode ser construído por meio do algoritmo 2. O autômato R_S resultante desse procedimento é tal que $L(R_S||G) = L(S_P/G) = \overline{K}$.

Algoritmo 2 (Realização de um supervisor-P)

Passo 1: *Construa um autômato* trim $R = (X_R, \Sigma, f_R, \Gamma_R, x_{0,R})$*, que gera* $\overline{K}$*.*

Passo 2: *Construa o observador* $R_{obs} = Obs(R, \Sigma_o)$ *(por meio do algoritmo 1, apresentado na subseção 2.4.1).*

Passo 3: *Em cada estado* x_{obs} *de* R_{obs}*, acrescente autolaços rotulados com os eventos não-observáveis pertencentes a* $\bigcup_{x \in x_{obs}} \Gamma_R(x)$*.*

Capítulo 4

Controle supervisório robusto em presença de perdas intermitentes de observação

Neste capítulo, o problema tratado na tese será formulado e resolvido. Os resultados serão apresentados, assim como, exemplos para ilustrar a eficácia da metodologia. Inicialmente, na seção 4.1, será proposto um exemplo com o objetivo de expor as motivações desse trabalho. Em seguida, na seção 4.2, o problema estudado será formulado, aplicando-se o formalismo adequado. Na seção 4.3, são apresentados alguns resultados preliminares, na sua maioria relacionados com a operação de dilatação. Em seguida, nas seções 4.4 e 4.5 são apresentados os principais resultados obtidos. Por fim, na seção 4.6, a abordagem proposta será aplicada a um exemplo de maior complexidade.

4.1 Exemplo de motivação

Nesta seção será apresentado um exemplo com o objetivo de expor as motivações de estudar o problemas de controle supervisório em presença de perdas intermitentes de observação.

Considere o autômato G, cujo diagrama de transição de estados é apresentado na figura 4.1(a), em que $\Sigma = \{\alpha, \beta, \gamma, \delta\}$ é o conjunto de eventos e os conjuntos de eventos observáveis e controláveis são, respectivamente, $\Sigma_o = \Sigma$ e $\Sigma_c = \{\alpha, \delta\}$. Suponha que a linguagem gerada por G, $L(G)$, deve ser modificada de tal maneira que satisfaça a especificação K gerada pelo autômato H, cujo diagrama de transição de estados está representado na figura 4.1(b).

Primeiramente considere o projeto de um supervisor supondo observação completa. Como K é controlável em relação a $L(G)$ e Σ_{uc}, então existe um supervisor

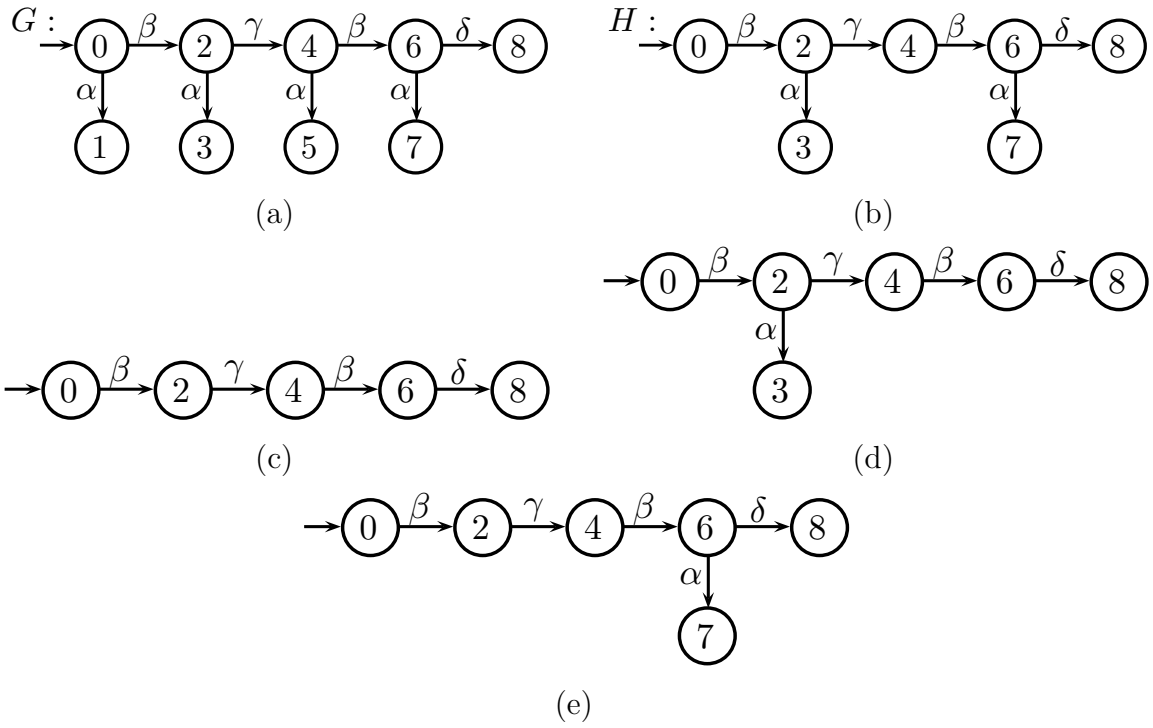


Figura 4.1: Autômato do sistema, G (a); Autômato da especificação, H (b); Comportamento obtido supondo β como não-observável (c); comportamentos possíveis de serem obtidos quando somente uma das observações de β falha (d) e (e).

S tal que $L(S/G) = L(H)$. Além disso, uma vez que β e γ são não controláveis, eles não podem ser desabilitados. Pode-se verificar que as exigências de projeto são satisfeitas pelo supervisor S, com a seguinte ação de controle:

(i) $s = \varepsilon \rightarrow S(s) = \{\beta, \gamma\}$;
(ii) $s = \beta \rightarrow S(s) = \{\alpha, \beta, \gamma\}$;
(iii) $s = \beta\gamma \rightarrow S(s) = \{\beta, \gamma\}$;
(iv) $s = \beta\gamma\beta \rightarrow S(s) = \{\alpha, \beta, \gamma, \delta\}$.

Considere, agora, que o evento β esteja sujeito a perdas intermitentes de observação. Inicialmente, suponha que a sequência $s = \beta\gamma$ ocorreu e o sensor que detecta a ocorrência do evento β falhou. Dessa forma, o supervisor deve tomar a decisão com base em $s' = \gamma$, a qual não pertence ao domínio de S. Como consequência, dependendo da forma como o supervisor foi implementado, ou ele permitirá que β ocorra novamente, ou o sistema irá travar. Em nenhuma dessas duas possibilidades os estados 3, 7 e 8 serão alcançados.

Ao se considerar que β está sujeito a perdas intermitentes de observação:

- Uma possível abordagem para projetar o supervisor é considerar β como um evento não-observável, ou seja, projetar um supervisor-P supondo $\Sigma_{uo} = \{\beta\}$. Uma vez que α deve ser desabilitado no início e $S_p(s)$ não muda até uma outra ocorrência de um evento suposto observável (nesse caso γ), então a nova ação

de controle para $s = \beta\gamma$ será $S_p(\gamma) = \{\beta, \delta\}$, pois α deve ser desabilitado depois da ocorrência de $s = \beta\gamma$. Isso implica que o comportamento desejado não será alcançado, como pode ser observado na figura 4.1(c).

- Outra possibilidade consiste em projetar o supervisor usando a abordagem proposta por ROHLOFF [8], na qual falhas de sensores podem ocorrer a qualquer momento, mas, uma vez ocorrida um falha, o respectivo sensor não mais volta a funcionar. No entanto, o uso da abordagem de Rohloff permite somente alcançar o mesmo comportamento obtido quando β é considerado não-observável.

Ao se considerar a natureza intermitente da falha, é possível projetar um supervisor capaz de alcançar resultados melhores do que os obtidos pelos dois métodos anteriores. No caso desse exemplo, deve-se projetar um supervisor que possa lidar com as situações a seguir:

(1) Não ocorrência de perdas de observação do evento β;

(2) A primeira observação de β é perdida, mas a segunda ocorrência de β é observada;

(3) A primeira ocorrência de β é observada enquanto a segunda é perdida;

(4) Ambas as ocorrências de β não são observadas.

Seja S_{rob} o supervisor com o qual é possível lidar com as situações *(1)–(4)* e $\hat{s}$ a sequência observada por S_{rob}. Então, S_{rob} pode ser definido da seguinte forma:

(i) $\hat{s} = \varepsilon \rightarrow S_{rob}(\hat{s}) = \{\beta, \gamma\}$;

(ii) $\hat{s} = \beta \rightarrow S_{rob}(\hat{s}) = \{\alpha, \beta, \gamma\}$;

(iii) $\hat{s} \in \{\gamma, \beta, \gamma\} \rightarrow S_{rob}(\hat{s}) = \{\beta, \gamma, \delta\}$;

(iv) $\hat{s} \in \{\beta\gamma\beta, \gamma\beta\} \rightarrow S_{rob}(\hat{s}) = \{\alpha, \beta, \gamma, \delta\}$.

Pode-se verificar que o supervisor S_{rob} alcança o comportamento desejado, dado na figura 4.1(b), quando ambas as ocorrências de β são observadas, o comportamento apresentado na figura 4.1(c) quando ambas as observações de β são perdidas e os comportamentos mostrados nas figuras 4.1(d) e 4.1(e) quando somente é observada, respectivamente, a primeira ou a segunda ocorrência de β.

É importante mencionar que, embora nem sempre o comportamento desejado seja alcançado nesse exemplo, a lei de controle proposta não só permite que o supervisor continue a trabalhar quando há perdas de observação, mas também a linguagem do sistema controlado por S_{rob} é mais permissiva que as linguagens obtidas pelas abordagens anteriores que se aplicam a situações em que falhas de sensores são possíveis.

4.2 Formulação do problema

Considere uma planta representada por um autômato $G = (X, \Sigma, f, \Gamma, x_0)$, cuja linguagem gerada é L. Seja $K = \overline{K}$ uma linguagem definida sobre Σ, tal que $K \subseteq L$, que especifica o comportamento máximo desejado.

Sejam as projeções $P_{dil,o} : \Sigma^*_{dil} \rightarrow \Sigma^*_o$ e $P_o : \Sigma^* \rightarrow \Sigma^*_o$. Como mostrado em CARVALHO *et al.* [16], quando G está sujeito a perdas intermitentes de observação, a linguagem observável é $P_{dil,o}[D(L)]$, ao invés de $P_o(L)$. Por esse motivo, sob perdas intermitentes de observação, o supervisor deve ser projetado para tomar decisões com base em $P_{dil,o}[D(L)]$ e não $P_o(L)$, conforme mostrado na figura 4.2.

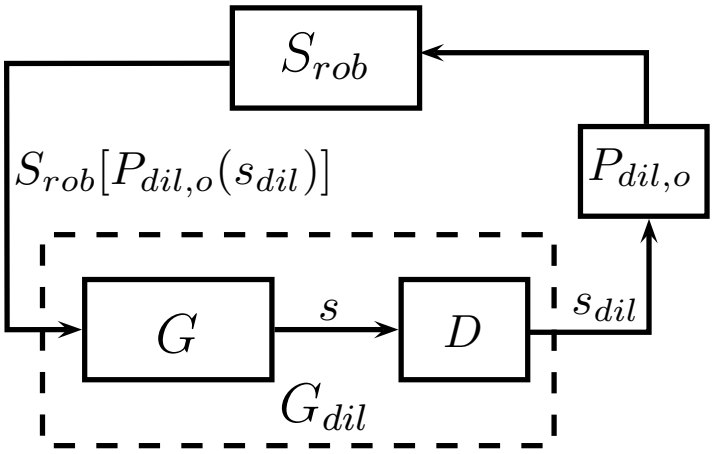


Figura 4.2: Estrutura realimentada para o problema de controle supervisório robusto sob perdas intermitentes de observações.

Considere também as seguintes hipóteses:

A1. Existe um supervisor-P S_P, tal que $L(S_P/G) = K$;

A2. $\Sigma_{ilo} \cap \Sigma_c = \emptyset$.

A hipótese **A1** exige que K seja controlável em relação a Σ_{uc} e L e observável em relação a Σ_c, P_o e Σ_{uo}. Vale ressaltar que a hipótese **A1** não implica em perda de generalidade, uma vez que o problema de controle supervisório robusto só faz sentido se existir algum supervisor para o caso nominal. A hipótese **A2** é razoável pelo fato dos eventos controláveis estarem, em muitos problemas de controle supervisório, associados a comandos. Dessa forma, por serem eventos de comando, suas observações não envolvem sensores ou canais de comunicação. O problema em que a ação de controle imposta pelo supervisor não é executada requer uma formulação diferente da abordada neste trabalho.

O problema considerado neste trabalho pode ser formulado da seguinte forma. Encontre um supervisor robusto S_{rob} tal que:

E1. $L(S_{rob}/G) = K$;

E2. $L(S_{rob}/G_{dil}) \subseteq D(K)$.

A exigência **E1** garante que quando não ocorrerem perdas de observação, o supervisor robusto irá produzir o mesmo comportamento que o supervisor-P, enquanto que a exigência **E2** determina que, na presença de perdas intermitentes de observações, S_{rob} deve gerar um comportamento que não ultrapasse $D(K)$. O supervisor assim projetado será referido aqui como um supervisor robusto a perdas intermitentes de observação (supervisor-R ou S_{rob}).

4.3 Resultados Preliminares

4.3.1 Resultados adicionais da Dilatação

Nesta seção, são apresentados alguns resultados preliminares relacionados à operação de dilatação. Tais resultados são cruciais na obtenção dos resultados principais a serem apresentados nas seções 4.4 e 4.5. Inicialmente, é definida uma nova operação, com a qual é possível recuperar uma sequência s a partir de qualquer sequência pertencente a $D(s)$.

Definição 9 (Contração) *A contração é o mapeamento:*

$$\begin{array}{rccc} R_D^{-1}: & \Sigma_{dil}^* & \longrightarrow & \Sigma^* \\ & s_{dil} & \longmapsto & R_D^{-1}(s_{dil}) \end{array}$$

no qual:

$$\begin{aligned} R_D^{-1}(\varepsilon) &:= \varepsilon, \\ R_D^{-1}(\sigma) &:= \sigma, \text{ se } \sigma \in \Sigma, \\ R_D^{-1}(\sigma') &:= \sigma, \text{ se } \sigma' \in \Sigma'_{ilo}, \\ R_D^{-1}(s_{dil}\sigma) &:= R_D^{-1}(s_{dil})R_D^{-1}(\sigma), \text{ para todo } s_{dil} \in \Sigma_{dil}^* \text{ e } \sigma \in \Sigma_{dil}. \end{aligned}$$

A extensão do mapeamento R_D^{-1} para uma linguagem $L_{dil} \subseteq \Sigma_{dil}^*$ é dada por:

$$R_D^{-1}(L_{dil}) = \bigcup_{s_{dil} \in L_{dil}} R_D^{-1}(s_{dil}).$$

Diretamente da definição de R_D^{-1}, é possível observar que, para qualquer linguagem $K \subseteq \Sigma^*$, $R_D^{-1}\left[D(K)\right] = K$.

Serão apresentadas, agora, novas propriedades da dilatação. Para tanto, considere um conjunto de eventos Σ, $\sigma \in \Sigma$, $s, s_1, s_2 \in \Sigma^*$, e as linguagens K, K_1 e K_2 definidas sobre Σ.

Propriedade 1 (P1) $K \subseteq D(K)$.

Demonstração: Direta da definição de dilatação. ■

Propriedade 2 (P2) $D(K_1K_2) = D(K_1)D(K_2)$.

Demonstração: Note que

$$\begin{aligned} D(K_1)D(K_2) &= [\cup_{s_1\in K_1} D(s_1)]\,[\cup_{s_2\in K_2} D(s_2)] \\ &= [\cup_{s_1\in K_1, s_2\in K_2} D(s_1)D(s_2)] \\ &= [\cup_{s_1\in K_1, s_2\in K_2} D(s_1s_2)] \\ &= D(K_1K_2). \end{aligned}$$

■

Propriedade 3 (P3) $s_1 \neq s_2 \Leftrightarrow D(s_1) \cap D(s_2) = \emptyset$

Demonstração: ($\Leftarrow$) Da propriedade **P1**, $\{s_1\} \subseteq D(s_1)$ e $\{s_2\} \subseteq D(s_2)$. Portanto, $\{s_1\} \cap \{s_2\} \subseteq D(s_1) \cap D(s_2) = \emptyset$, implica que $s_1 \neq s_2$.

($\Rightarrow$) Suponha que $D(s_1) \cap D(s_2) \neq \emptyset$. Então, existe $s_{dil} \in D(s_1) \cap D(s_2)$. Por sua vez, da definição de contração, para todo $s_{dil} \in D(s)$, $R_D^{-1}(s_{dil}) = s$. Portanto, $R_D^{-1}(s_{dil}) = s_1 = s_2$. ■

Propriedade 4 (P4) $D(K_1) \cup D(K_2) = D(K_1 \cup K_2)$.

Demonstração: É fácil observar que,

$$\begin{aligned} D(K_1) \cup D(K_2) &= [\cup_{s_1\in K_1} D(s_1)] \cup [\cup_{s_2\in K_2} D(s_2)] \\ &= \cup_{s\in K_1\cup K_2} D(s) \\ &= D(K_1 \cup K_2). \end{aligned}$$

■

Propriedade 5 (P5) $D(K_1) \cap D(K_2) = D(K_1 \cap K_2)$.

Demonstração: Utilizando as propriedades **P3** e **P4** é possível particionar $D(K_1)$ e $D(K_2)$ como $D(K_1) = S_1 \dot\cup S_{12}$ e $D(K_2) = S_2 \dot\cup S_{12}$, em que $S_1 = D(K_1 \setminus K_2)$, $S_{12} = D(K_1 \cap K_2)$, e $S_2 = D(K_2 \setminus K_1)$. Dessa forma,

$$\begin{aligned} D(K_1) \cap D(K_2) &= (S_1 \cup S_{12}) \cap (S_2 \cup S_{12}) \\ &= [(S_1 \cup S_{12}) \cap S_2] \cup [(S_1 \cup S_{12}) \cap S_{12}] \\ &= (S_1 \cap S_2) \cup (S_{12} \cap S_2) \cup (S_1 \cap S_{12}) \cup (S_{12} \cap S_{12}) \end{aligned}$$

Por sua vez, da propriedade **P3** da dilatação, pode-se deduzir que $S_1 \cap S_2 = \emptyset$, $S_{12} \cap S_2 = \emptyset$ e $S_1 \cap S_{12} = \emptyset$. Portanto, $D(K_1) \cap D(K_2) = S_{12} = D(K_1 \cap K_2)$. ■

Propriedade 6 (P6) $\overline{D(K)} = D(\overline{K})$.

Demonstração: Inicialmente, será provado, por indução no comprimento de uma sequência $s \in \Sigma^*$, que $\overline{D(s)} = D(\overline{s})$.

(i) (Caso base) $s = \varepsilon$. $D(\varepsilon) = \{\varepsilon\} \Rightarrow \overline{D(s)} = \{\varepsilon\} = D(\overline{s})$.

(ii) (Hipótese da indução) Suponha que $\overline{D(s_n)} = D(\overline{s_n})$, $\forall s_n : |s_n| \leq n$.

(iii) Seja $s_{n+1} = s_n\sigma$, para algum $\sigma \in \Sigma$. Então,

$$\overline{s_{n+1}} = \overline{s_n\sigma} = \overline{\{s_n\}} \cup \{s_n\sigma\},$$

o que implica

$$\begin{aligned} D(\overline{s_{n+1}}) &= D(\overline{s_n}) \cup D(s_n\sigma) = \overline{D(s_n)} \cup D(s_n)D(\sigma) \\ &= \overline{D(s_n)D(\sigma)} = \overline{D(s_n\sigma)} = \overline{D(s_{n+1})}. \end{aligned}$$

Aplicando o resultado acima para todas as sequências pertencentes a $D(K)$, pode-se escrever que:

$$\overline{D(K)} = \cup_{s\in K}\overline{D(s)} = \cup_{s\in K} D(\overline{s}) = \cup_{s\in\overline{K}} D(s) = D(\overline{K}).$$

■

Propriedade 7 (P7) $K_1 \subseteq K_2 \Leftrightarrow D(K_1) \subseteq D(K_2)$.

Demonstração: ($\Rightarrow$) Imediato a partir da propriedade **P1**.

($\Leftarrow$) Note que, $D(K_1) \subseteq D(K_2) \Rightarrow D(K_1) = D(K_1) \cap D(K_2) = D(K_1 \cap K_2)$, na qual a última igualdade é decorrente da propriedade **P5**. Dessa forma,

$$R_D^{-1}[D(K_1)] = R_D^{-1}[D(K_1 \cap K_2)] \Rightarrow K_1 = K_1 \cap K_2 \Rightarrow K_1 \subseteq K_2.$$

■

Propriedade 8 (P8) $K = D(K) \cap \Sigma^*$.

Demonstração: Pela definição de dilatação, é possível particionar $D(K)$ da forma $D(K) = K\dot{\cup}K'$, em que $K' = \{s \in \Sigma^*_{dil} : (\exists\sigma' \in \Sigma'_{ilo} \text{ e } s_1, s_2 \in \Sigma^*_{dil})[s = s_1\sigma' s_2]\}$. Por consequência, $K' \cap \Sigma^* = \emptyset$, o que implica que $D(K) \cap \Sigma^* = (K \cup K') \cap \Sigma^* = (K \cap \Sigma^*) \cup (K' \cap \Sigma^*) = (K \cap \Sigma^*) \cup \emptyset = K$. ■

Propriedade 9 (P9) $D(s) \cap D(K) \neq \emptyset \Leftrightarrow D(s) \subseteq D(K)$.

Demonstração: ($\Leftarrow$) Imediata.

($\Rightarrow$) Suponha que existe uma sequência $s_{dil} \in D(s) \cap D(K)$. Uma vez que $s_{dil} \in D(K)$, então $R_D^{-1}(s_{dil}) = s \in K$, que por sua vez, implica que $D(s) \subseteq D(K)$.

■

Propriedade 10 (P10) $D(K_1 \setminus K_2) = D(K_1) \setminus D(K_2)$.

Demonstração: Note que,

$$\begin{aligned}
& K_1 = (K_1 \setminus K_2) \cup (K_1 \cap K_2) \\
\overset{\mathbf{P5}}{\Longrightarrow} \; & D(K_1) = D(K_1 \setminus K_2) \cup D(K_1 \cap K_2) \\
\Longrightarrow \; & D(K_1) \subseteq D(K_1 \setminus K_2) \cup D(K_2) \\
\Longrightarrow \; & D(K_1) \setminus D(K_2) \subseteq [D(K_1 \setminus K_2) \cup D(K_2)] \setminus D(K_2) \\
\Longrightarrow \; & D(K_1) \setminus D(K_2) \subseteq D(K_1 \setminus K_2).
\end{aligned}$$

Por outro lado, seja $s_{dil} \in D(K_1 \setminus K_2)$. Então,

$$\begin{aligned}
& s = R_D^{-1}(s_{dil}) \in K_1 \setminus K_2 \\
\Longrightarrow \; & s \in K_1 \text{ e } s \notin K_2 \\
\overset{\mathbf{P3}}{\Longrightarrow} \; & s_{dil} \in D(K_1) \text{ e } s_{dil} \notin D(K_2) \\
\Longrightarrow \; & D(K_1 \setminus K_2) \subseteq D(K_1) \setminus D(K_2).
\end{aligned}$$

■

Em adição às propriedades da dilatação **P1**-**P10**, serão apresentadas duas propriedade que relacionam as operações de dilatação e contração. Para tanto, considere uma linguagem $K_{dil} \subseteq \Sigma_{dil}^*$, não necessariamente obtida pela dilatação de alguma linguagem $K \in \Sigma^*$.

Propriedade 11 (P11) $D\left[R_D^{-1}(K_{dil})\right] \supseteq K_{dil}$.

Demonstração: Sejam as sequências $s_{dil} \in K_{dil}$ e $s = R_D^{-1}(s_{dil})$. Então, $s_{dil} \in D(s)$ e $s \in R_D^{-1}(K_{dil})$, o que implica que $s_{dil} \in D(s) \subseteq D\left[R_D^{-1}(K_{dil})\right]$.

Para demonstrar que a igualdade, $D\left[R_D^{-1}(K_{dil})\right] = K_{dil}$, nem sempre é verdadeira, considere, por exemplo, $K_{dil} = \{a\sigma'\}$, com $\Sigma = \{a, \sigma\}$ e $\Sigma_{ilo} = \{\sigma\}$. É fácil verificar que $D\left[R_D^{-1}(K_{dil})\right] = \{a\sigma, a\sigma'\}$. ■

Propriedade 12 (P12) $D\left[R_D^{-1}(K_{dil})\right] = K_{dil} \Leftrightarrow (\exists K \subseteq \Sigma^*)\,[D(K) = K_{dil}]$.

Demonstração: ($\Rightarrow$) Imediata.

($\Leftarrow$) Suponha que existe uma linguagem $K \subseteq \Sigma^*$, tal que $D(K) = K_{dil}$. Portanto, da definição de contração, $K = R_D^{-1}\left[D(K)\right] = R_D^{-1}(K_{dil})$, que implica que $D(K) = K_{dil} = D\left[R_D^{-1}(K_{dil})\right]$. ■

Para completar o conjunto de propriedades da operação de dilatação, serão apresentadas três outras propriedades, as quais relacionam a operação de dilatação com algumas projeções.

Propriedade 13 (P13) *Sejam as projeções* $P_{dil,o} : \Sigma_{dil}^* \to \Sigma_o^*$, $P_{nilo} : \Sigma^* \to \Sigma_{nilo}^*$ *e* $P_{o,nilo} : \Sigma_o^* \to \Sigma_{nilo}^*$. *Para qualquer linguagem* K *definida sobre* Σ,

$$P_{dil,o}\left[D(K)\right] \subseteq P_{o,nilo}^{-1}\left[P_{nilo}(K)\right].$$

Demonstração: Considere uma sequência $s \in \Sigma^*$. Sem perda de generalidade, suponha que s possa ser escrita como: $s = w\sigma t$, em que $w, t \in (\Sigma \setminus \Sigma_{ilo})^*$ e $\sigma \in \Sigma_{ilo} \cup \{\varepsilon\}$. Dessa forma, observa-se que:

$$P_{dil,o}\left[D(s)\right] = P_o(w)\{\sigma, \varepsilon\}P_o(t).$$

Por outro lado,

$$P_{o,nilo}^{-1}\left[P_{nilo}(s)\right] = P_{o,nilo}^{-1}\left[P_{nilo}(w)\right] P_{o,nilo}^{-1}\left[P_{nilo}(t)\right].$$

Note que $P_{o,nilo}^{-1}\left[P_{nilo}(w)\right] \supseteq \{P_o(w), P_o(w)\sigma\}$ e $P_{o,nilo}^{-1}\left[P_{nilo}(t)\right] \supseteq \{P_o(t)\}$, o que implica que:

$$P_{o,nilo}^{-1}\left[P_{nilo}(s)\right] \supseteq \{P_o(w), P_o(w)\sigma\}\{P_o(t)\} = P_{dil,o}\left[D(s)\right].$$

Portanto, para qualquer linguagem $K \subseteq \Sigma^*$:

$$\begin{aligned} P_{dil,o}\left[D(K)\right] &= P_{dil,o}\left[\bigcup_{s\in L} D(s)\right] = \bigcup_{s\in L}\left\{P_{dil,o}\left[D(s)\right]\right\} \\ &\subseteq \bigcup_{s\in L}\left\{P_{o,nilo}^{-1}\left[P_{nilo}(s)\right]\right\} = P_{o,nilo}^{-1}\left[P_{nilo}(K)\right]. \end{aligned}$$

■

Propriedade 14 (P14) *Sejam as projeções* $P_{dil,nilo} : \Sigma_{dil}^* \to \Sigma_{nilo}^*$ *e* $P_{nilo} : \Sigma^* \to \Sigma_{nilo}^*$. *Para toda linguagem* $K_{nilo} \subseteq \Sigma_{nilo}^*$,

$$P_{dil,nilo}^{-1}\left(K_{nilo}\right) = D\left[P_{nilo}^{-1}\left(K_{nilo}\right)\right].$$

Demonstração: Defina $\Sigma_1 = \Sigma_{dil} \setminus \Sigma_{nilo} = \Sigma_{ilo} \cup \Sigma_{uo} \cup \Sigma'_{ilo}$ e $\Sigma_2 = \Sigma \setminus \Sigma_{nilo} = \Sigma_{ilo} \cup \Sigma_{uo}$ e seja $s = \sigma_1\sigma_2...\sigma_n \in K_{nilo}$, com $\sigma_i \in \Sigma_{nilo}$ e $i = 1...n$. Dessa forma,

$$P_{dil,nilo}^{-1}(s) = \Sigma_1^*\sigma_1\Sigma_1^*\sigma_2\Sigma_1^*...\Sigma_1^*\sigma_n\Sigma_1^*$$

e

$$P_{nilo}^{-1}(s) = \Sigma_2^*\sigma_1\Sigma_2^*\sigma_2\Sigma_2^*...\Sigma_2^*\sigma_n\Sigma_2^*.$$

Note que $\Sigma_1 = \Sigma_2 \cup \Sigma'_{ilo}$ e, portanto:

$$\begin{aligned} D[P^{-1}_{nilo}(s)] &= (\Sigma_2 \cup \Sigma'_{ilo})^* \sigma_1 (\Sigma_2 \cup \Sigma'_{ilo})^* \sigma_2 (\Sigma_2 \cup \Sigma'_{ilo})^* ... (\Sigma_2 \cup \Sigma'_{ilo})^* \sigma_n (\Sigma_2 \cup \Sigma'_{ilo})^* \\ &= P^{-1}_{dil,nilo}(s). \end{aligned}$$

■

Propriedade 15 (P15) *Sejam as projeções* $P_{dil,o} : \Sigma^*_{dil} \to \Sigma^*_o$ *e* $P_{nilo} : \Sigma^* \to \Sigma^*_{nilo}$. *Para qualquer linguagem* K *definida sobre* Σ*:*

$$P^{-1}_{dil,o} \left\{ P_{dil,o} \left[D(K) \right] \right\} \subseteq D \left\{ P^{-1}_{nilo} \left[P_{nilo}(K) \right] \right\}.$$

Demonstração: Usando a propriedade **P13**, é possível escrever:

$$P_{dil,o} \left[D(K) \right] \subseteq P^{-1}_{o,nilo} \left[P_{nilo}(K) \right].$$

Portanto,

$$\begin{aligned} P^{-1}_{dil,o} \left\{ P_{dil,o} \left[D(K) \right] \right\} &\subseteq P^{-1}_{dil,o} \{ P^{-1}_{o,nilo} \left[P_{nilo}(K) \right] \} \\ &= P^{-1}_{dil,nilo} \left[P_{nilo}(K) \right] \\ &= D \{ P^{-1}_{nilo} \left[P_{nilo}(K) \right] \}, \end{aligned}$$

na qual a última igualdade é uma aplicação direta da propriedade **P14**. ■

Com o objetivo de facilitar futuras consultas às propriedades **P1** a **P15**, essas são apresentadas na tabela 4.1.

Tabela 4.1: Propriedades adicionais da dilatação e contração.

P1	$K \subseteq D(K)$
P2	$D(K_1 K_2) = D(K_1) D(K_2)$
P3	$s_1 \neq s_2 \Leftrightarrow D(s_1) \cap D(s_2) = \emptyset$
P4	$D(K_1) \cup D(K_2) = D(K_1 \cup K_2)$
P5	$D(K_1) \cap D(K_2) = D(K_1 \cap K_2)$
P6	$\overline{D(K)} = D(\overline{K})$
P7	$K_1 \subseteq K_2 \Leftrightarrow D(K_1) \subseteq D(K_2)$
P8	$K = D(K) \cap \Sigma^*$
P9	$D(s) \cap D(K) \neq \emptyset \Leftrightarrow D(s) \subseteq D(K)$
P10	$D(K_1 \setminus K_2) = D(K_1) \setminus D(K_2)$
P11	$D[R^{-1}_D(K_{dil})] \supseteq K_{dil}$
P12	$D[R^{-1}_D(K_{dil})] = K_{dil} \Leftrightarrow (\exists K \subseteq \Sigma^*)[D(K) = K_{dil}]$
P13	$P_{dil,o}[D(K)] \subseteq P^{-1}_{o,nilo}[P_{nilo}(K)]$
P14	$P^{-1}_{dil,nilo}(K_{nilo}) = D[P^{-1}_{nilo}(K_{nilo})]$
P15	$P^{-1}_{dil,o}\{P_{dil,o}[D(K)]\} \subseteq D\{P^{-1}_{nilo}[P_{nilo}(K)]\}$

4.3.2 Controlabilidade em presença de perdas intermitentes de observação

Nessa seção, é mostrado que a controlabilidade de uma linguagem não é afetada por perdas intermitentes de observações.

Proposição 1 (Controlabilidade) *K é controlável em relação a L e Σ_{uc} se, e somente se, $D(K)$ é controlável em relação a $D(L)$ e $\Sigma_{uc,dil} = D(\Sigma_{uc})$, ou seja,*

$$\overline{K}\Sigma_{uc} \cap L \subseteq \overline{K} \Leftrightarrow \overline{D(K)}D(\Sigma_{uc}) \cap D(L) \subseteq \overline{D(K)}. \tag{4.1}$$

Demonstração: A prova é obtida por meio da aplicação direta das propreidades **P7**, **P5**, **P2** e **P6**, da seguinte forma:

$$\begin{aligned}
\overline{K}\Sigma_{uc} \cap L \subseteq \overline{K} &\overset{\mathbf{P7}}{\Longleftrightarrow} D(\overline{K}\Sigma_{uc} \cap L) \subseteq D(\overline{K}) \\
&\overset{\mathbf{P5}}{\Longleftrightarrow} D(\overline{K}\Sigma_{uc}) \cap D(L) \subseteq D(\overline{K}) \\
&\overset{\mathbf{P2}}{\Longleftrightarrow} D(\overline{K})D(\Sigma_{uc}) \cap D(L) \subseteq D(\overline{K}) \\
&\overset{\mathbf{P6}}{\Longleftrightarrow} \overline{D(K)}D(\Sigma_{uc}) \cap D(L) \subseteq \overline{D(K)}.
\end{aligned}$$

■

4.4 Observabilidade robusta em presença de perdas intermitentes de observação

4.4.1 Observabilidade robusta forte

Inicialmente, será apresentada uma definição de observabilidade robusta em presença de perdas intermitentes de observação seguindo as mesmas premissas que a definição de diagnosticabilidade robusta, proposta por CARVALHO *et al.* [16].

Definição 10 (Observabilidade robusta forte) *Uma linguagem observável $K \subseteq L$, em que $L = \overline{L}$, é observável robusta forte quando sujeita a perdas intermitentes de observação ou, equivalentemente, em relação a L, D, $P_{dil,o} : \Sigma^*_{dil} \to \Sigma^*_o$ e Σ_c se, e somente se, $\forall s \in \overline{K}$ e $\forall \sigma \in \Sigma_c$:*

$$[s\sigma \in L \setminus \overline{K}] \Longrightarrow (\nexists s' \in \overline{K}) \left[(P_{dil,o}[D(s)] \cap P_{dil,o}[D(s')] \neq \emptyset) \wedge (s'\sigma \in \overline{K}) \right],$$

ou, de maneira equivalente:

$$[s\sigma \in L \setminus \overline{K}] \Longrightarrow P^{-1}_{dil,o}\left[P_{dil,o}[D(s)]\right]\{\sigma\} \cap \overline{K} = \emptyset.$$

A definição de observabilidade robusta forte pode ser analisada da seguinte forma. Dado que uma sequência $s \in \overline{K}$ ocorreu e supondo que essa sequência seja sucedida por um evento σ que deve ser desabilitado, ou seja, $s\sigma \in L \setminus \overline{K}$ e, além disso, supondo que possam ocorrer perdas intermitentes de observação, então o supervisor deve tomar sua decisão não unicamente a partir de $P_o(s)$, mas com base em $P_{dil,o}[D(s)]$. Assim, se existir uma sequência $s' \in \overline{K}$, tal que $s'\sigma \in \overline{K}$ e $P_{dil,o}[D(s)] \cap P_{dil,o}[D(s')] \neq \emptyset$, então, a sequência s' possui também uma continuação com σ e, portanto, não deverá ser desabilitada, gerando, assim, uma decisão ambígua.

No exemplo a seguir é apresentada uma situação na qual uma linguagem K é observável no sentido usual e, no entanto, não é observável robusta forte.

Exemplo 8 *Seja o conjunto de eventos $\Sigma = \{\alpha, \beta, \gamma\}$, em que o subconjunto de eventos observáveis é $\Sigma_o = \{\alpha, \beta\}$, o subconjunto de eventos controláveis é $\Sigma_c = \{\alpha\}$ e o subconjunto de eventos passíveis de perdas intermitentes de observação é $\Sigma_{ilo} = \{\beta\}$. Considere as linguagens:*

$$L = \{\varepsilon, \alpha, \beta, \beta\alpha, \beta\gamma, \beta\gamma\alpha\} \quad e \quad K = \{\varepsilon, \beta, \beta\alpha, \beta\gamma, \beta\gamma\alpha\}.$$

Pode-se verificar que K é observável em relação a L, $P_o : \Sigma^* \to \Sigma_o^*$ e Σ_c. No entanto, pode-se verificar, também, que K não é observável robusta forte em relação a L, D, $P_{dil,o} : \Sigma_{dil}^* \to \Sigma_o^*$ e Σ_c. Para tanto, seja $s = \varepsilon$. Portanto $s\alpha = \varepsilon\alpha \in L \setminus \overline{K}$. Seja, agora, $s' = \beta$, note que $s'\alpha = \beta\alpha \in \overline{K}$. Além disso,

$$P_{dil,o}[D(s')] = P_{dil,o}[D(\beta)] = P_{dil,o}[\{\beta, \beta'\})] = \{\varepsilon, \beta\}$$

e

$$P_{dil,o}[D(s)] = P_{dil,o}[D(\varepsilon)] = P_{dil,o}[\{\varepsilon\})] = \{\varepsilon\},$$

ou seja, $P_{dil,o}[D(s)] \cap P_{dil,o}[D(s')] \neq \emptyset$.

Uma condição necessária e suficiente para a observabilidade robusta forte é dada pelo seguinte teorema.

Teorema 4 *Uma linguagem K é observável robusta forte em relação a L, D, $P_{dil,o}$ e Σ_c se, e somente se, $D(K)$ for observável em relação a $D(L)$, $P_{dil,o}$ e Σ_c.*

Demonstração: ($\Rightarrow$) Suponha que $D(K)$ não seja observável em relação a $D(L)$, $P_{dil,o}$ e Σ_c. Então, $\exists s_{dil}, s'_{dil} \in \overline{D(K)}$, $s_{dil} \neq s'_{dil}$, e $\sigma \in \Sigma_c$, tal que $s_{dil}\sigma \in D(L) \setminus \overline{D(K)}$, $s'_{dil}\sigma \in \overline{D(K)}$ e $P_{dil,o}(s_{dil}) = P_{dil,o}(s'_{dil})$.

Seja $s = R_D^{-1}(s_{dil})$ e $s' = R_D^{-1}(s'_{dil})$. Então, usando a propriedade **P3**, não é difícil verificar que $R_D^{-1}(s_{dil}\sigma) \in L \setminus \overline{K}$ e $R_D^{-1}(s'_{dil}\sigma) \in \overline{K}$, o que implica que $s\sigma \in L \setminus \overline{K}$ e

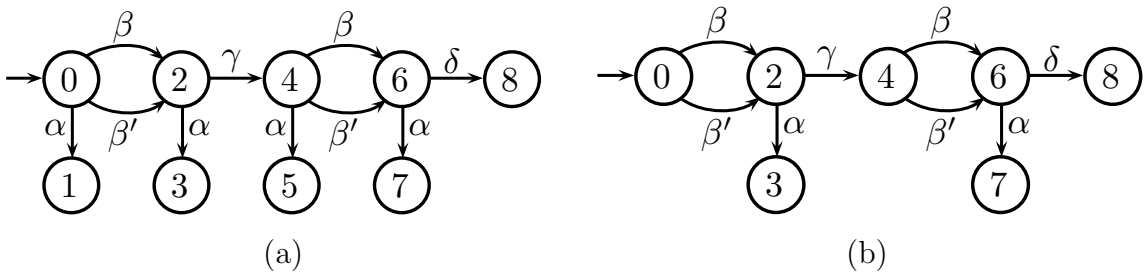


Figura 4.3: Autômatos G_{dil} (a) e H_{dil} (b), cujas linguagens geradas são $D(L)$ e $D(K)$, respectivamente.

$s'\sigma \in \overline{K}$. Além disso,

$$P_{dil,o}[D(s)] \cap P_{dil,o}[D(s')] \supseteq \{P_{dil,o}(s_{dil})\} \cap \{P_{dil,o}(s'_{dil})\} \neq \emptyset.$$

($\Leftarrow$) Suponha, agora, que K não seja observável robusta forte em relação a L, D, $P_{dil,o}$ e Σ_c. Então, $\exists s, s' \in \overline{K}$, $s \neq s'$, e $\sigma \in \Sigma_c$, tal que $s\sigma \in L \setminus \overline{K}$, $s'\sigma \in \overline{K}$ e $P_{dil,o}[D(s)] \cap P_{dil,o}[D(s')] \neq \emptyset$. A última expressão implica que $\exists (s_{dil}, s'_{dil}) \in D(s) \times D(s')$, tal que $P_{dil,o}(s_{dil}) = P_{dil,o}(s'_{dil})$. De acordo com a propriedade **P3**, $s_{dil} \neq s'_{dil}$, uma vez que $s \neq s'$. A prova é concluída ao se usar a definição e as propriedades **P6** e **P10** da dilatação da seguinte forma:

$$\begin{aligned} \{s, s', s'\sigma\} \subseteq \overline{K} &\Rightarrow \{s_{dil}, s'_{dil}, s'_{dil}\sigma\} \subseteq \overline{D(K)}, \\ s\sigma \in L \setminus \overline{K} &\Rightarrow s_{dil}\sigma \in D(L) \setminus \overline{D(K)}. \end{aligned}$$

■

Além de prover uma condição necessária e suficiente para a observabilidade robusta forte, o teorema 4 também fornece uma forma de se testar a observabilidade robusta forte. Por exemplo, a verificação da observabilidade robusta forte para o exemplo da seção 4.1 utilizando o teorema 4 requer a construção dos autômatos G_{dil} e H_{dil}, respectivamente, representados nas figuras 4.3(a) e 4.3(b), cujas linguagens geradas são $D(L)$ e $D(K)$. Note que $D(K)$ não é observável em relação a $D(L)$, $P_{dil,o}$ e Σ_c e, portanto, de acordo com o teorema 4, K não é observável robusta forte em relação a Σ_c, L, D e $P_{dil,o}$. Esse é um exemplo no qual uma linguagem K é observável no sentido usual (em relação a L, P_o e Σ_c), porém não é observável robusta forte.

Suponha, agora, que K seja observável, mas não seja observável robusta forte. Então, um supervisor não poderá gerar K em todas as possíveis combinações de ocorrências de perdas intermitentes de observação. Surge, então, a seguinte questão: qual é a maior sublinguagem de K que é, ao mesmo tempo, observável e observável robusta forte? Essa questão é respondida pelo teorema a seguir.

Teorema 5 *Uma linguagem $K \subseteq L$ será observável robusta forte em relação a L,*

D, $P_{dil,o}$ e Σ_c se, e somente se, K for observável em relação a L, $P_{nilo} : \Sigma^* \to \Sigma^*_{nilo}$ e Σ_c.

Demonstração: ($\Rightarrow$) Suponha que K não seja observável em relação a L, P_{nilo} e Σ_c. Então, existem duas sequências $s, s' \in \overline{K}$, $s \neq s'$ e $\sigma_c \in \Sigma_c$, tais que $s\sigma_c \in L \setminus \overline{K}$, $s'\sigma_c \in \overline{K}$ e $P_{nilo}(s) = P_{nilo}(s')$.

Note que, para todo $\tilde{s} \in \Sigma^*$, existe $\tilde{s}_{dil} \in D(\tilde{s})$ no qual todo evento $\sigma \in \Sigma_{ilo}$ em $\tilde{s}$ foi substituído pelo seu evento correspondente $\sigma' \in \Sigma'_{ilo}$. Dessa forma:

$$P_{dil,o}(\tilde{s}_{dil}) = P_{nilo}(\tilde{s}) \Rightarrow P_{nilo}(\tilde{s}) \in P_{dil,o}[D(\tilde{s})],$$

e, portanto, pode-se afirmar que:

$$P_{nilo}(s) \in P_{dil,o}[D(s)] \text{ e } P_{nilo}(s') \in P_{dil,o}[D(s')],$$

o que implica que $P_{dil,o}[D(s)] \cap P_{dil,o}[D(s')] \neq \emptyset$.

($\Leftarrow$) Agora, suponha que K não seja observável robusta forte em relação a L, D, $P_{dil,o}$ e Σ_c. Então, existem duas sequências $s, s' \in \overline{K}$ e $\sigma \in \Sigma_c$, tais que $s\sigma \in L \setminus \overline{K}$, $s'\sigma \in \overline{K}$ e $P_{dil,o}[D(s)] \cap P_{dil,o}[D(s')] \neq \emptyset$. Usando a propriedade **P13**, é possível escrever:

$$P^{-1}_{o,nilo}\left[P_{nilo}(s)\right] \cap P^{-1}_{o,nilo}\left[P_{nilo}(s')\right] \neq \emptyset.$$

Note que $P_{nilo}(s)$ e $P_{nilo}(s')$ têm unicamente eventos em Σ_{nilo}, e, uma vez que a projeção inversa $P^{-1}_{o,nilo}$ adiciona eventos pertencentes a Σ_{ilo}, $P^{-1}_{o,nilo}\left[P_{nilo}(s)\right] \cap P^{-1}_{o,nilo}\left[P_{nilo}(s')\right] = \emptyset$ se, e somente se, $P_{nilo}(s) \neq P_{nilo}(s')$, o que prova o resultado. ■

Por meio do teorema 5 mostra-se que, para uma linguagem observável K, ser também observável robusta forte com relação a perdas intermitentes de observação de eventos em Σ_{ilo}, é necessário e suficiente que K seja observável em relação a P_{nilo}, ou seja, quando todos os eventos pertencentes a Σ_{ilo} são considerados não observáveis. Isso pode ser explicado da seguinte forma: o problema tratado neste trabalho pode ser definido como o de encontrar um supervisor que controle as $2^{T_{ilo}}$ plantas, em que T_{ilo} é o número de transições em G rotuladas por eventos pertencentes a Σ_{ilo}, geradas por todas as possíveis combinações de transições σ e σ' ($\sigma \in \Sigma_{ilo}$). Dentre essas plantas, o pior caso no que diz respeito à observabilidade, é a planta cujas transições em G rotuladas com eventos pertencentes a Σ_{ilo} foram substituídas por transições rotuladas pelos seus correspondentes eventos em Σ'_{ilo}.

4.4.2 Controlabilidade e observabilidade robusta fraca

A definição de observabilidade robusta forte, apresentada na seção anterior, é muito restritiva, pois está associada à ideia de K ser a linguagem gerada pelo sistema controlado, até mesmo quando ocorrem perdas de observação. Esse fato sugere que a definição 10 pode ser relaxada, de forma que sejam considerados como aceitáveis, controladores que, na ocorrência de perdas de observação, gerem sublinguagens de K, mas que não interrompam sequências de K que possam ser continuadas devido a observações de eventos passíveis de se tornarem não-observáveis intermitentemente.

Antes da apresentar uma segunda definição relacionada à robustez em presença de perdas intermitentes de observação, é importante lembrar que, como dito na seção 4.2, onde foi formulado o problema aqui considerado, devemos supor que K é controlável em relação a Σ_{uc} e L e observável em relação a Σ_c, P_o e Σ_{uo}.

Definição 11 (Controlabilidade e observabilidade robusta fraca) *Uma linguagem $K \subseteq L$ controlável em relação a Σ_{uc} e L e observável em relação a Σ_c, P_o e L, será controlável e observável robusta fraca em relação a perdas intermitentes de observação se, e somente se, existir $K_{dil} \subseteq \Sigma^*_{dil}$ que seja controlável em relação a $\Sigma_{uc,dil}$ e $D(L)$ e observável em relação a $D(L)$, $P_{dil,o}$ e Σ_c, tal que $K \subseteq K_{dil} \subseteq D(K)$.*

A definição 11 tem como base a ideia de que embora o teorema 4 sugira que, no melhor caso, a máxima linguagem alcançada em relação $D(L)$, $P_{dil,o}$ e Σ_c seja $D(K)$, esse resultado só é alcançável quando K for observável robusta forte. Portanto, na definição 11 essa exigência é relaxada. A seguir, serão apresentadas as condições necessárias e suficientes para a existência de um supervisor robusto no sentido da definição 11. Esse resultado requer o seguinte lema.

Lema 1 (CURY e KROGH [3]) *Sejam G_1 e G_2 dois autômatos definidos sobre um mesmo alfabeto Σ. Suponha que $L(G_1) \subseteq L(G_2)$ e que Σ_o seja o conjunto de eventos observáveis de ambos, G_1 e G_2. Então, para qualquer supervisor $S : \Sigma^*_o \rightarrow 2^{\Sigma}$, $L(S/G_1) = L(S/G_2) \cap L(G_1)$.*

Pode-se, agora, apresentar o resultado desejado.

Teorema 6 *Seja K uma linguagem controlável e observável. Então, existe um supervisor-R $S_{rob} : \Sigma^*_o \rightarrow 2^{\Sigma}$, tal que $L(S_{rob}/G) = K$ e $L(S_{rob}/G_{dil}) \subseteq D(K)$ se, e somente se, K for controlável e observável robusta fraca no sentido da definição 11.*

Demonstração: ($\Rightarrow$) Suponha que existe um supervisor S_{rob}, tal que $L(S_{rob}/G) = K$ e $L(S_{rob}/G_{dil}) \subseteq D(K)$, e defina $L(S/G_{dil}) = K_{dil} \subseteq D(K)$. Dessa forma, K_{dil} é controlável em relação a $D(L)$ e $\Sigma_{uc,dil}$ é observável em relação a $D(L)$, $P_{dil,o}$

e Σ_c. Adicionalmente, usando o lema 1, pode-se inferir que $K = L(S_{rob}/G) = L(S_{rob}/G_{dil}) \cap L = K_{dil} \cap L$, que, por fim, implica que $K \subseteq K_{dil}$.

($\Leftarrow$) Suponha que K seja controlável e observável robusta fraca de acordo com a definição 11. Portanto, existe uma linguagem $K_{dil} \subseteq \Sigma_{dil}^*$, controlável em relação a $D(L)$ e $\Sigma_{uc,dil}$ e observável em relação a $D(L)$, $P_{dil,o}$ e Σ_c, tal que $K \subseteq K_{dil} \subseteq D(K)$. Pode-se observar que $K_{dil} \cap \Sigma^* = K$, pelas seguintes razões:
(i) $K \subseteq K_{dil}$ e $K = K \cap \Sigma^*$ implicam que $K \subseteq K_{dil} \cap \Sigma^*$;
(ii) $K_{dil} \subseteq D(K) \Rightarrow K_{dil} \cap \Sigma^* \subseteq D(K) \cap \Sigma^* = K$, na qual a última igualdade é decorrente da propriedade **P8**.

Por outro lado, como K_{dil} é controlável em relação a $D(L)$ e $\Sigma_{uc,dil}$ e observável em relação a $D(L)$, $P_{dil,o}$ e Σ_c, então existe um supervisor S_{rob} tal que $L(S_{rob}/G_{dil}) = K_{dil} \subseteq D(K)$. Usando o lema 1 conclui-se que $L(S_{rob}/G) = K_{dil} \cap L = K_{dil} \cap (L \cap \Sigma^*) = K \cap L = K$. ■

No exemplo de motivação, apresentado na seção 4.1, embora a linguagem desejada K não seja observável robusta forte, essa linguagem é controlável e observável robusta fraca. Esse fato pode ser verificado aplicando algum teste de controlabilidade e de observabilidade usando os autômatos H_{rob} e G_{dil}, cujos diagramas de transição de estados são apresentados, respectivamente, nas figuras 4.4 e 4.3(a). Adicionalmente, não é difícil verificar que $K = L(H) \subseteq K_{dil} = L(H_{rob}) \subseteq D(K) = L(H_{dil})$.

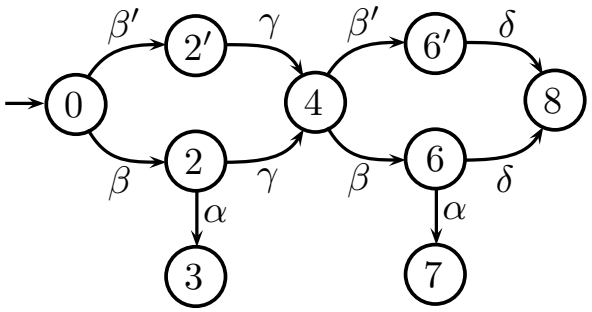


Figura 4.4: Autômato H_{rob}, cuja linguagem gerada é K_{dil}.

4.5 Projeto do supervisor robusto

Nesta seção, será mostrado como se obter um supervisor robusto S_{rob}, tal que $L(S_{rob}/G) = K$ e $L(S_{rob}/G_{dil}) = K_{dil} \subseteq D(K)$, quando a linguagem K não é observável robusta forte. Com esse objetivo, será definida uma normalidade robusta no mesmo sentido da definição de observabilidade robusta fraca.

Definição 12 (Controlabilidade e normalidade robusta fraca) *Uma linguagem $K \subseteq L$ controlável em relação a L e Σ_{uc} e normal em relação a L e P_o, será*

controlável e normal robusta fraca em relação a perdas intermitentes de observação se, e somente se, $K \subseteq D(K)^{\uparrow CN}$.

Como determinado pela proposição a seguir, a observabilidade e a normalidade no sentido robusto fraco estão relacionadas de maneira equivalente à forma como a observabilidade e a normalidade estão relacionadas no sentido usual.

Proposição 2 *Sejam duas linguagens* K *e* L*, tais que* $K \subseteq L$*. As afirmações a seguir são verdadeiras:*
(a) *Se a linguagem* K *for controlável e normal robusta fraca, então* K *será controlável e observável robusta fraca;*
(b) *Supondo que* $\Sigma_c \subseteq \Sigma_o$ *e que* K *seja controlável e observável robusta fraca, então* K *será controlável e normal robusta fraca.*

Demonstração: (a) Suponha que K seja controlável e normal robusta fraca. Então, $K \subseteq D(K)^{\uparrow CN}$ e, por definição, $D(K)^{\uparrow CN} \subseteq D(K)$. Além disso, como $D(K)^{\uparrow CN}$ é controlável (em relação a $D(L)$ e $\Sigma_{uc,dil}$) e normal (em relação a $D(L)$ e $P_{dil,o}$), então, $D(K)^{\uparrow CN}$ também é observável (em relação a $D(L)$, $P_{dil,o}$ e Σ_c). Portanto K é controlável e observável robusta fraca.

(b) Suponha, agora, que K seja controlável e observável robusta fraca no sentido da definição 11. Então, existe $K_{dil} \subseteq \Sigma_{dil}^*$ controlável (em relação a $D(L)$ e $\Sigma_{uc,dil}$) e observável (em relação a $D(L)$, $P_{dil,o}$ e Σ_c), tal que $K \subseteq K_{dil} \subseteq D(K)$. Considerando a suposição de que $\Sigma_c \subseteq \Sigma_o$, tem-se, então, de acordo como o teorema 3, que K_{dil} também é normal (em relação a $D(L)$ e $P_{dil,o}$). Consequentemente, $K \subseteq K_{dil} \subseteq D(K)^{\uparrow CN}$. Portanto, K é controlável e normal robusta fraca. ■

De acordo com o teorema 6 em conjunto com a proposição 2, se uma linguagem K for controlável e normal robusta fraca, então existirá um supervisor S_{rob}, tal que $L(S_{rob}/G) = K$ e $L(S_{rob}/G_{dil}) = K_{dil} \subseteq D(K)$. Além disso, K_{dil} poderá ser qualquer linguagem controlável (em relação a $\Sigma_{uc,dil}$ e $D(L)$) e observável (em relação a $D(L)$, $P_{dil,o}$ e Σ_c), tal que $K \subseteq K_{dil} \subseteq D(K)$, como determinado pelo teorema a seguir.

Teorema 7 *Seja* K *uma linguagem controlável e observável robusta fraca. Para qualquer linguagem* K_{dil} *controlável em relação a* $\Sigma_{uc,dil}$ *e* $D(L)$ *e observável em relação a* $D(L)$*,* $P_{dil,o}$ *e* Σ_c *tal que* $K \subseteq K_{dil} \subseteq D(K)$*, existe um supervisor-R* S_{rob} *tal que* $L(S_{rob}/G) = K$ *e* $L(S_{rob}/G_{dil}) = K_{dil}$.

Demonstração: Seja K_{dil} uma linguagem controlável em relação a $\Sigma_{uc,dil}$ e $D(L)$ e observável em relação a $D(L)$, $P_{dil,o}$ e Σ_c tal que $K \subseteq K_{dil} \subseteq D(K)$. Como K_{dil} é controlável e observável, então existe um supervisor S, tal que:

$$L(S/G_{dil}) = K_{dil} \overset{\textbf{Lema 1}}{\Longrightarrow} L(S/G) = K_{dil} \cap L \Longrightarrow L(S/G) = K_{dil} \cap \Sigma^* \cap L. \quad (4.2)$$

Por outro lado,

$$\begin{aligned} & K \subseteq K_{dil} \subseteq D(K) \\ \Longrightarrow \quad & K \cap \Sigma^* \subseteq K_{dil} \cap \Sigma^* \subseteq D(K) \cap \Sigma^* \\ \Longrightarrow \quad & K \subseteq K_{dil} \cap \Sigma^* \subseteq D(K) \cap \Sigma^* \\ \overset{\mathbf{P8}}{\Longrightarrow} \quad & K \subseteq K_{dil} \cap \Sigma^* \subseteq K \\ \Longrightarrow \quad & K_{dil} \cap \Sigma^* = K. \end{aligned} \tag{4.3}$$

Por fim, subtituindo-se (4.3) em (4.2), tem-se que $L(S/G) = K \cap L = K$ e, dessa forma, S é um supervisor-R no sentido robusto fraco. ■

No corolário a seguir é apresentada uma consequência imediata do teorema 7.

Corolário 1 *Se K for uma linguagem controlável e observável robusta fraca, então existirá um supervisor S_{rob} tal que $L(S_{rob}/G) = K$ e $L(S_{rob}/G_{dil}) = D(K)^{\uparrow CN}$.*

Pode-se verificar que a abordagem baseada na observabilidade robusta forte é mais conservadora do que a baseada na observabilidade robusta fraca, uma vez que a primeira só alcança o mesmo nível de permissividade que a segunda em circunstâncias muito especiais, como será mostrado adiante. Para obter esse resultado, é preciso antes, apresentar o seguinte lema.

Lema 2 *Uma linguagem K é normal em relação a L e P_{nilo} se, e somente se, $D(K)$ é normal em relação a $D(L)$ e $P_{dil,o}$.*

Demonstração: ($\Rightarrow$) Suponha que K seja normal em relação a L e P_{nilo}. Portanto,

$$\begin{aligned} & \overline{K} \supseteq P_{nilo}^{-1}(P_{nilo}(\overline{K})) \cap L \\ \Longrightarrow \quad & D(\overline{K}) \supseteq D(P_{nilo}^{-1}(P_{nilo}(\overline{K})) \cap L) \\ \overset{\mathbf{P5}}{\Longrightarrow} \quad & D(\overline{K}) \supseteq D(P_{nilo}^{-1}(P_{nilo}(\overline{K}))) \cap D(L) \supseteq P_{dil,o}^{-1}(P_{dil,o}(D(\overline{K})) \cap D(L), \end{aligned}$$

na qual a última inclusão é consequência da propriedade **P15**. Finalmente, utilizando a propriedade **P6**, obtém-se $\overline{D(K)} \supseteq P_{nilo}^{-1}(P_{nilo}(\overline{D(K)}) \cap D(L)$, que implica que $D(K)$ é normal em relação a $D(L)$ e $P_{dil,o}$.

($\Leftarrow$) Suponha que K não seja normal em relação a L e P_{nilo}. Então, existem $s_1 \in \overline{K}$ e $s_2 \in L \setminus \overline{K}$, tais que $P_{nilo}(s_1) = P_{nilo}(s_2)$. Usando as propriedades **P6**, **P7** e **P10**, pode-se escrever:

$$D(s_1) \subseteq \overline{D(K)}$$

e

$$D(s_2) \subseteq D(L) \setminus \overline{D(K)}.$$

Considere, agora, o mapeamento $R' : \Sigma^* \rightarrow (\Sigma_{uo} \cup \Sigma_{nilo} \dot{\cup} \Sigma'_{ilo})^*$, definido recursivamente como:
(i) $R'(\varepsilon) = \varepsilon$;
(ii) $R'(\sigma) = \sigma$, se $\sigma \in \Sigma \setminus \Sigma_{ilo}$ e $R'(\sigma) = \sigma' \in \Sigma'_{ilo}$, se $\sigma \in \Sigma_{ilo}$;
(iii) $R'(s\sigma) = R'(s)R'(\sigma)$, para $s \in \Sigma^*$ e $\sigma \in \Sigma$.
Note que o mapeamento $R'(s)$ substitui toda ocorrência de eventos $\sigma \in \Sigma_{ilo}$ pelos seus correspondentes eventos $\sigma' \in \Sigma'_{ilo}$ na sequência $s \in \Sigma^*$.

Sejam $s_{dil,1} \in D(s_1)$ e $s_{dil,2} \in D(s_2)$, tais que $s_{dil,1} = R'(s_1)$ e $s_{dil,2} = R'(s_2)$. Observe que $P_{dil,o}(s_{dil,1}) = P_{nilo}(s_1)$ e $P_{dil,o}(s_{dil,2}) = P_{nilo}(s_2)$. Assim, é possível concluir que $P_{dil,o}(s_{dil,1}) = P_{dil,o}(s_{dil,2})$, $s_{dil,1} \in \overline{D(K)}$ e $s_{dil,2} \in D(L) \setminus \overline{D(K)}$ e, portanto, $D(K)$ não é normal em relação a $D(L)$ e $P_{dil,o}$. ■

Antes de enunciar o próximo teorema, vamos definir as seguintes linguagens:

- $K_{sto}^{\uparrow CN}$: é a sublinguagem de K controlável (em relação a L e Σ_{uc}) e normal (em relação a L e P_{nilo}) suprema;

- $D(K)^{\uparrow CN}$: é a sublinguagem de $D(K)$ controlável (em relação a $D(L)$ e $\Sigma_{uc,dil}$) e normal (em relação a $D(L)$ e $P_{dil,o}$) suprema.

Vale ressaltar que $D(K_{sto}^{\uparrow CN})$ é a linguagem obtida ao se exigir que $L(S_{rob}/G)$ seja controlável e observável robusta forte, enquanto $D(K)^{\uparrow CN}$ é a linguagem obtida ao se exigir que $L(S_{rob}/G)$ seja controlável e observável robusta fraca. Agora, será apresentado o principal resultado dessa seção.

Teorema 8 *Sejam $K_{sto}^{\uparrow CN}$ e $D(K)^{\uparrow CN}$ as linguagens definidas acima. Então:*
(a) $D(K_{sto}^{\uparrow CN}) \subseteq D(K)^{\uparrow CN}$;
(b) $D(K_{sto}^{\uparrow CN}) = D(K)^{\uparrow CN} \iff D\left[R_D^{-1}\left(D(K)^{\uparrow CN}\right)\right] = D(K)^{\uparrow CN}$.

Demonstração: *(a)* Note que $K_{sto}^{\uparrow CN} \subseteq K$. Portanto, de acordo com **P7**,

$$D(K_{sto}^{\uparrow CN}) \subseteq D(K). \tag{4.4}$$

Como, por hipótese, $K_{sto}^{\uparrow CN}$ é controlável em relação a L e Σ_{uc} e normal em relação a L e P_{nilo}, então, de acordo com a proposição 1 e o lema 2, respectivamente, pode-se concluir que $D(K_{sto}^{\uparrow CN})$ é controlável em relação a $D(L)$ e $\Sigma_{uc,dil}$ e normal em relação a $D(L)$ e $P_{dil,o}$, o que implica, juntamente com a equação 4.4, que $D(K_{sto}^{\uparrow CN}) \subseteq D(K)^{\uparrow CN}$.

(b) De acordo com **P12**, $D\left[R_D^{-1}(D(K)^{\uparrow CN})\right] = D(K)^{\uparrow CN}$ se, e somente se, $\exists K_R \subseteq \Sigma^*$, tal que $D(K_R) = D(K)^{\uparrow CN}$. Portanto, é equivalente provar que $D(K_{sto}^{\uparrow CN}) = D(K)^{\uparrow CN}$ se, e somente se, existe $K_R \subseteq \Sigma^*$, tal que $D(K_R) = D(K)^{\uparrow CN}$.

($\Rightarrow$) Suponha que $D(K_{sto}^{\uparrow CN}) = D(K)^{\uparrow CN}$. Nesse caso, $K_R = K_{sto}^{\uparrow CN}$.

($\Leftarrow$) Suponha que existe $K_R \subseteq \Sigma^*$, tal que $D(K_R) = D(K)^{\uparrow CN}$. Então, de acordo com a proposição 1 e o lema 2, K_R é controlável em relação a L e Σ_{uc} e normal em relação a L e P_{nilo}. Adicionalmente,

$$D(K)^{\uparrow CN} \subseteq D(K) \overset{\mathbf{P7}}{\Rightarrow} K_R \subseteq K,$$

então, $K_R \subseteq K_{sto}^{\uparrow CN}$. Aplicando a dilatação em ambos os lados dessa última expressão, obtém-se:

$$D(K)^{\uparrow CN} = D(K_R) \subseteq D(K_{sto}^{\uparrow CN}).$$

Por fim, utilizando a parte *(a)*, $D(K)^{\uparrow CN} = D(K_{sto}^{\uparrow CN})$. ■

Para ilustrar os resultados desta seção, será considerado, novamente, o exemplo de motivação da seção 4.1. Deseja-se projetar um supervisor robusto para o sistema a eventos discretos apresentado na figura 4.1(a), que atenda a linguagem desejada K, gerada pelo autômato H mostrado na figura 4.1(b), e considerando que o evento β esteja sujeito a perdas intermitentes de observações. Como mostrado na seção anterior, $D(K)^{\uparrow CN} = K_{dil}$, em que K_{dil} é a linguagem gerada pelo autômato H_{rob}, apresentado na figura 4.4. Então, de acordo com a proposição 2, existe um supervisor S_{rob} tal que $L(S_{rob}/G) = K$ e $L(S_{rob}/G_{dil}) = K_{dil}$. O autômato que realiza o supervisor S_{rob} é mostrado na figura 4.5. Por meio de uma análise da figura 4.5, é fácil notar que:

(i) Quando ambas as ocorrências de β são observadas, o sistema controlado gera a linguagem desejada K, gerada pelo autômato H, apresentado na figura 4.1(b);

(ii) Quando ambas as ocorrências de β são perdidas, S_{rob} atinge o comportamento mostrado na figura 4.1(c);

(iii) Quando a primeira (respectivamente, segunda) ocorrência de β é perdida, a linguagem gerada pelo autômato mostrado na figura 4.1(d) (resp. figura 4.1(e)) é obtida.

Vale ressaltar que o comportamento alcançado por S_{rob} é mais permissivo do que aquele encontrado ao se buscar uma linguagem observável robusta forte, uma vez que essa última abordagem produz a linguagem $K_{sto}^{\uparrow CN}$, gerada pelo autômato mostrado na figura 4.1(c).

4.6 Aplicação da metodologia proposta a um sistema de controle de tráfego ferroviário

Nesta seção, os resultados apresentados no trabalho serão ilustrados por uma situação de aplicação real. O problema considerado consiste em um sistema de con-

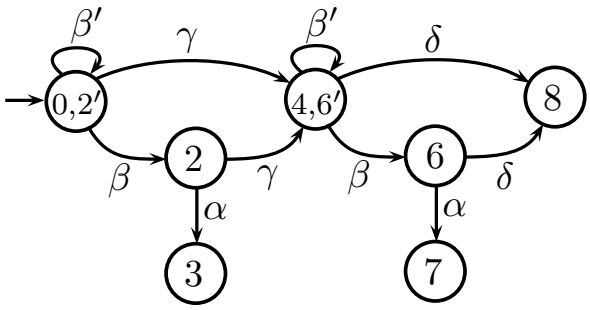


Figura 4.5: Ralização do supervisor S_{rob}, para o qual $L(S_{rob}/G_{dil}) = D(K)^{\uparrow CN}$.

trole de tráfego de trens entre duas estações, o qual foi considerado inicialmente por RAMADGE e WONHAM [25] e também foi analisado em outros trabalhos, como, por exemplo, [26] e [32].

Considere duas estações de trem interligadas por uma ferrovia particionada em quatro setores, conforme ilustrado na figura 4.6. Dois trens, T_1 e T_2, partem da estação A em direção à estação B. Em alguns pontos de transição existem semáforos e sensores. Nas transições em que houver semáforo a passagem de trens pode ser controlada, uma vez que, por meio do uso do semáforo, pode-se ordenar que o trem não passe para o setor seguinte. Por sua vez, nas transições em que houver sensores a passagem de trens pode ser detectada. Suponha que os semáforos (simbolizados por $*$) e os sensores (simbolizados por !) estão dispostos no cenário conforme ilustrado na figura 4.6. Na figura 4.7 é apresentado o diagrama de transição de estados dos autômatos G_{T_i}, $i = 1, 2$, que modelam as movimentações dos trens T_i, $i = 1, 2$. Nesses autômatos, os estados correspondem ao setor da ferrovia, ou estação, no qual o respectivo trem se encontra; e os eventos modelam as transições desse trem de uma das localizações (setores da ferrovia e estações) para outra. Note que a forma como os semáforos e sensores estão dispostos determina, respectivamente, quais eventos são controláveis e observáveis. Então, o conjunto de eventos controláveis é dado por $\Sigma_c = \{11, 12, 14, 21, 22, 24\}$ e o conjunto de eventos observáveis é igual a $\Sigma_o = \{11, 13, 14, 15, 21, 23, 24, 25\}$.

O autômato G que modela o movimento síncrono de T_1 e T_2 é dado por $G = G_{T_1}||G_{T_2}$, e o correspondente diagrama de transição de estados está representado na figura 4.8.

Para evitar colisões entre os trens, deseja-se impedir que ambos estejam simul-

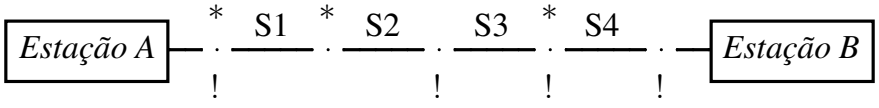


Figura 4.6: Diagrama de controle de uma ferrovia.

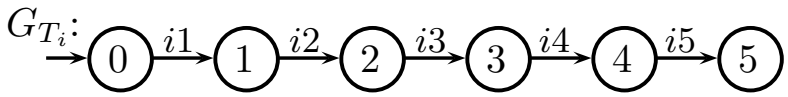


Figura 4.7: Diagrama de transição de estado do autômato G_{T_i} que modela as movimentações do trem T_i, $i = 1, 2$.

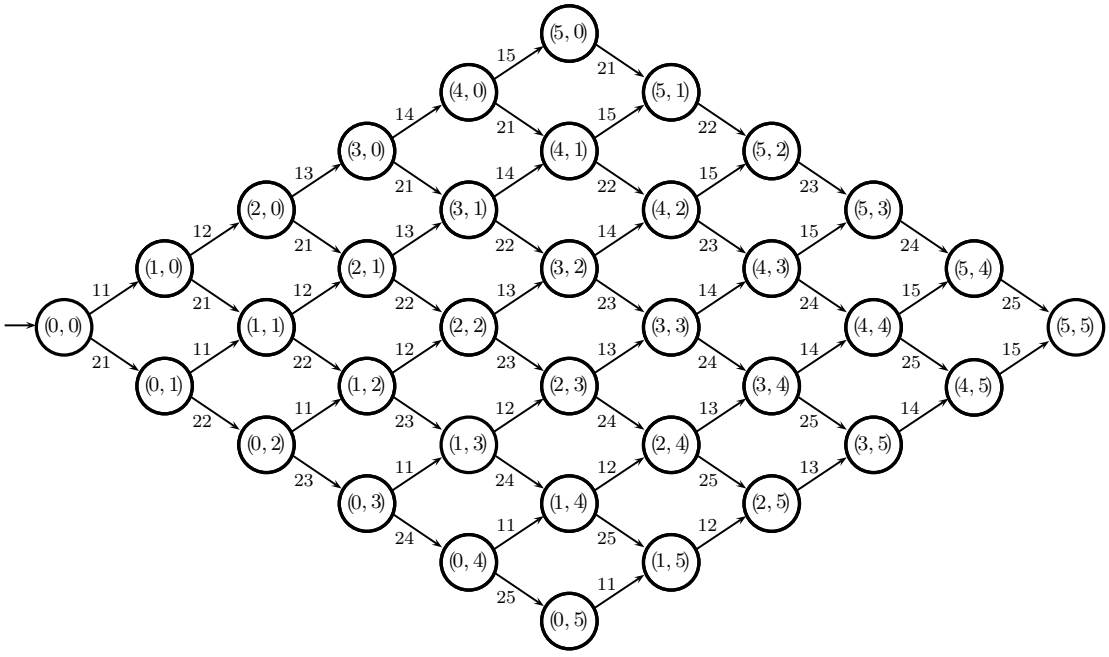


Figura 4.8: Diagrama de transição de estados do autômato $G = G_{T_1}||G_{T_2}$ que modela o comportamento do sistema.

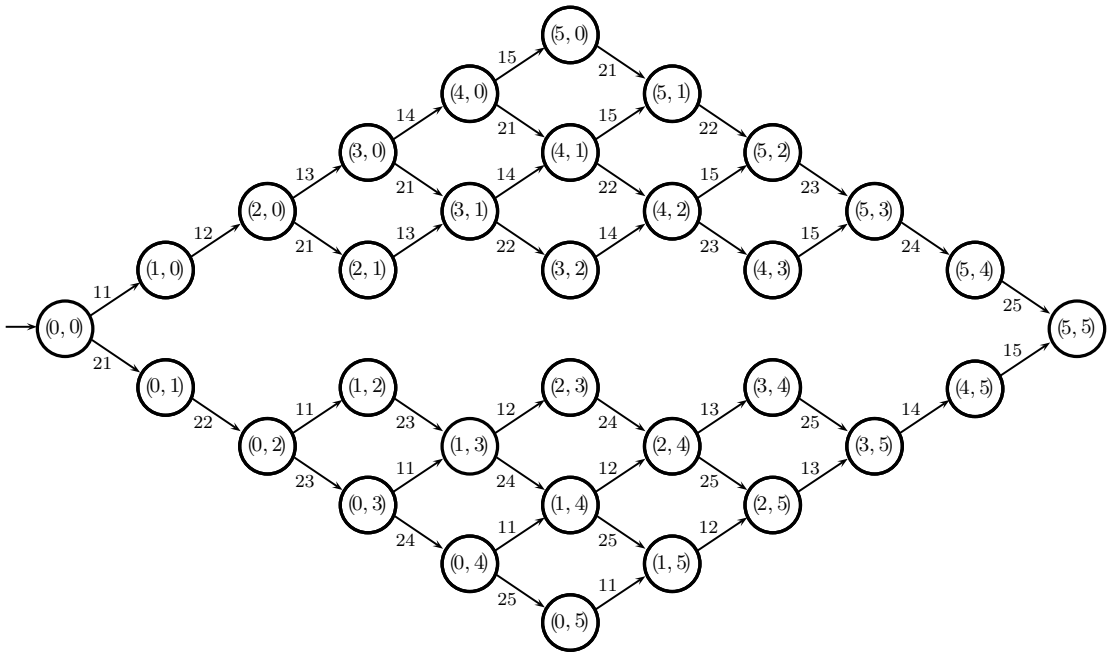


Figura 4.9: Diagrama de transição de estados do autômato H_a cuja linguagem gerada, K_a, modela o comportamento desejado.

taneamente em um mesmo setor da ferrovia, ou seja, deve haver um supervisor que impeça que os estados $(1,1)$, $(2,2)$, $(3,3)$ e $(4,4)$ sejam alcançados. A linguagem K_a ($K_a = \overline{K}_a$) que modela esse comportamento desejado é gerada pelo autômato H_a cujo diagrama de transição de estados é apresentado na figura 4.9.

Utilizando os autômatos G e H_a ilustrados, respectivamente, nas figuras 4.8 e 4.9 é possível verificar que K_a não é controlável e, consequentemente, não é admissível, conforme apresentado na seção 3.1. Uma forma de contornar esse problema consiste em calcular a sublinguagem controlável suprema de K_a, simbolizada aqui por $K_a^{\uparrow C}$. O diagrama de transição de estados do autômato H_C que gera $K_a^{\uparrow C}$ é apresentado na figura 4.10.

Embora a linguagem $K_a^{\uparrow C}$ seja controlável, não é difícil verificar que essa linguagem é não-observável. Portanto, como descrito na seção 3.2, não existe um supervisor-P S_P tal que $L(S_P/G) = K_a^{\uparrow C}$. No entanto, nesse caso específico, uma sublinguagem de $K_a^{\uparrow C}$ que seja, ao mesmo tempo, controlável e observável, pode ser obtida diretamente de $K_a^{\uparrow C}$ por inspeção. O autômato H, apresentado na figura 4.11, gera uma sublinguagem observável de $K_a^{\uparrow C}$ simbolizada, nesta seção, simplesmente por K para aliviar a simbologia. Esse autômato pode ser obtido a partir do autômato H_C (figura 4.10) excluindo-se desse último os estados $(1,2)$ e $(2,1)$. A exclusão do estado $(1,2)$ (respec. $(2,1)$) é consequência de que, para se obter uma sublinguagem observável de $K_a^{\uparrow C}$, é necessário desativar o evento controlável 11 (respec. 21) no estado $(0,2)$ (respec. $(2,0)$), uma vez que esse estado está ligado por um evento não-observável, 12 (respec. 22), ao estado $(1,0)$ (respec. $(0,1)$), no qual 11 (res-

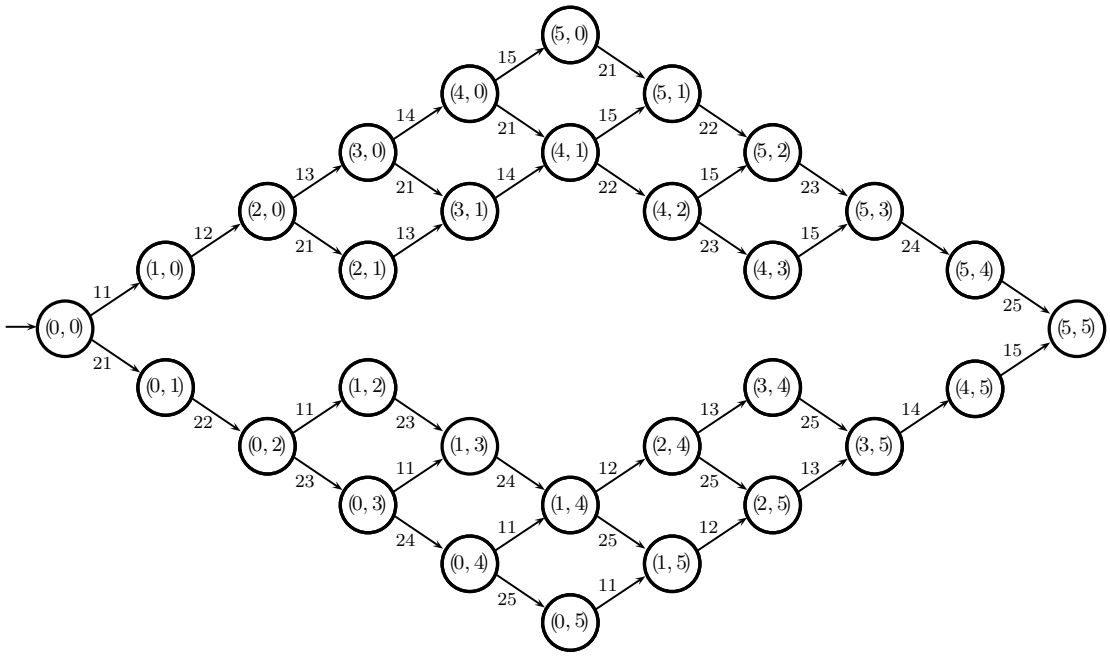


Figura 4.10: Diagrama de transição de estados do autômato H_C que gera a linguagem $K_a^{\uparrow C}$.

pec. 21) foi desativado. Dado que $K_a^{\uparrow C}$ é controlável e que os eventos desativados, 11 e 21, são ambos controláveis, a sublinguagem de $K_a^{\uparrow C}$ gerada pelo autômato H também será controlável. Não é difícil verificar que a linguagem K, gerada por H, é a sublinguagem controlável e observável suprema de $K_a^{\uparrow C}$. No entanto, vale ressaltar que nem sempre existirá uma sublinguagem controlável e observável suprema, a existência dessa linguagem é uma particularidade desse exemplo.

Suponha, agora, que os sensores referentes aos eventos 13 e 23 estejam sujeitos a falhas intermitentes, ou seja, $\Sigma_{ilo} = \{13, 23\}$. Considere o problema de projetar um supervisor robusto em presença perdas intermitentes de observação. O sistema e o comportamento desejado em presença de perdas intermitentes de observação são modelados, respectivamente, pelos autômatos G_{dil} (figura 4.12) e H_{dil} (figura 4.13).

De acordo com a metodologia proposta, são possíveis duas abordagens. Na primeira abordagem, busca-se um supervisor robusto no sentido da observabilidade robusta forte (definição 10), enquanto que, na segunda abordagem, busca-se um supervisor robusto no sentido da controlabilidade e observabilidade robusta fraca (definição 11).

Na primeira abordagem, de acordo com o teorema 5, é necessário que K seja observável em relação a $L(G)$, $P_{nilo} : \Sigma^* \to \Sigma_{nilo}^*$ e Σ_c, em que, no caso desse exemplo $\Sigma_{nilo} = \{11, 14, 15, 21, 24, 25\}$. Utilizando os autômatos G e H representados, respectivamente, nas figuras 4.8 e 4.11, é possível verificar que essa condição (teorema 5) não é satisfeita pela linguagem K. Por sua vez, a sublinguagem de K

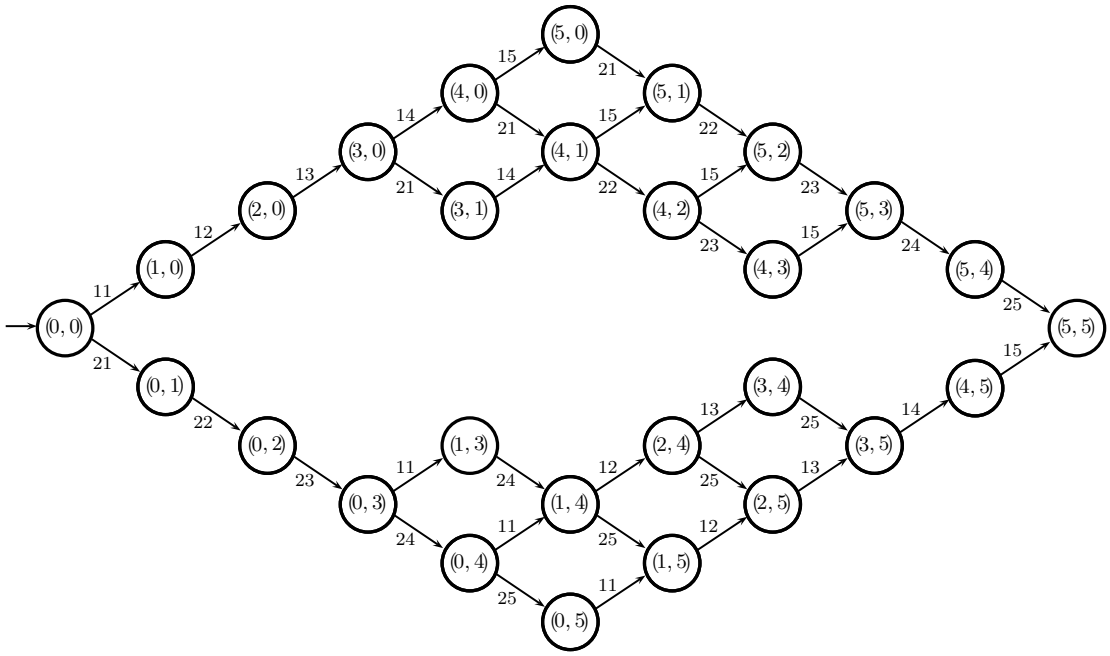


Figura 4.11: Diagrama de transição de estados do autômato H cuja linguagem gerada, K, modela o comportamento desejado admissível.

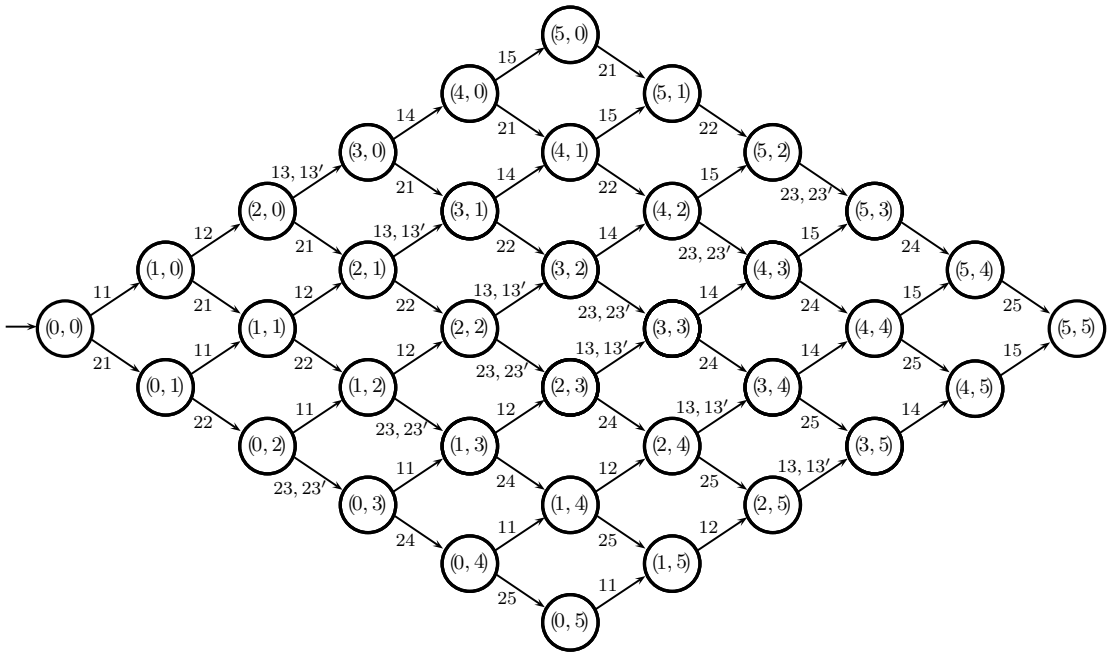


Figura 4.12: Diagrama de transição de estados do autômato G_{dil}.

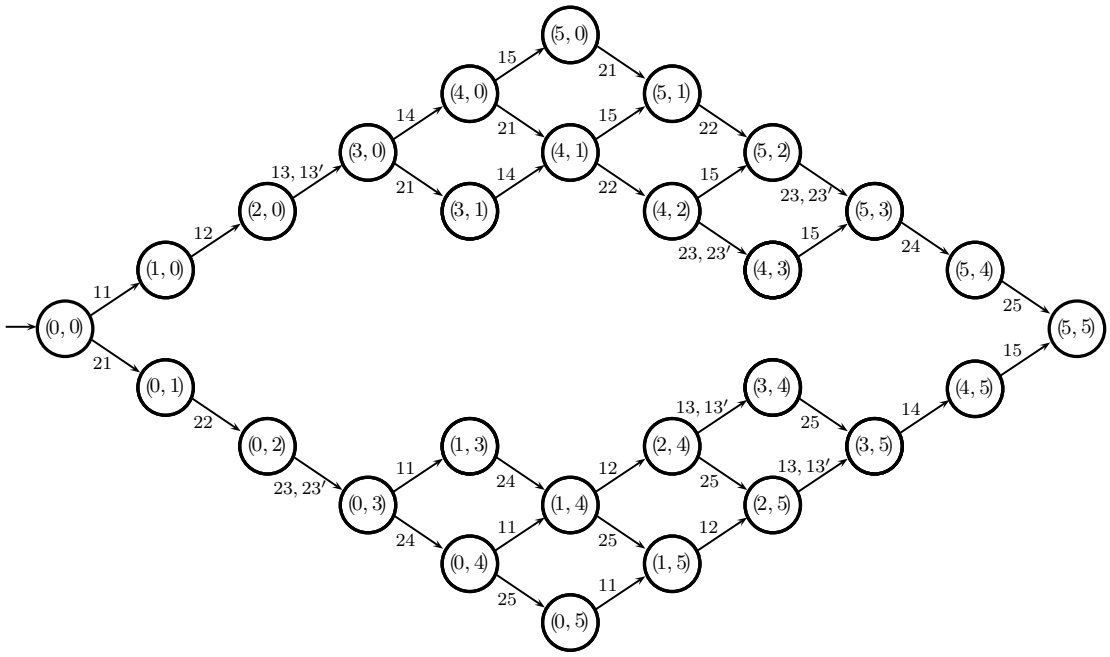


Figura 4.13: Diagrama de transição de estados do autômato H_{dil} que gera a linguagem $D(K)$.

mais permissiva controlável e observável robusta forte, de acordo com a definição 10, simbolizada por $K_{sto}^{\uparrow CO}$, é gerada pelo autômato apresentado na figura 4.14.

Por meio da segunda abordagem, pode-se obter um supervisor robusto S_{rob} que, em presença de perdas intermitentes de observação, produz um comportamento em malha fechada tal que $L(S_{rob}/G) = K$ e $L(S_{rob}/G_{dil}) \subseteq D(K)$. No entanto, de acordo com o teorema 6, para que esse supervisor exista é necessário e suficiente que K seja controlável e observável robusta fraca em relação à definição 11. Por meio de uma análise visual dos autômatos G_{dil} (figura 4.12) e H_{dil} (figura 4.13) é possível verificar que a linguagem $D(K)^{\uparrow CO}$, gerada pelo autômato H_{rob} apresentado na figura 4.15, é a sublinguagem de $D(K)$ controlável (em relação a $L(G_{dil})$ e $D(\Sigma_{uc})$) e observável (em relação a $L(G_{dil})$, $P_{dil,o}$ e Σ_c) mais permissiva. Por fim, pode ser verificado que K é controlável e observável robusta fraca de acordo com a definição 11, ao ser observado por meio dos autômatos H (figura 4.11) e H_{rob} (figura 4.15) que $K \subset D(K)^{\uparrow CO}$ e, portanto, o supervisor S_{rob} desejado existe.

Ao se comparar os autômatos H_{sto} (figura 4.14) e H_{rob} (figura 4.15) que modelam, respectivamente, os comportamentos em malha fechada obtidos com a primeira e a segunda abordagens, pode-se observar que somente nessa última abordagem os estados $(3,1)$ e $(1,3)$ foram conservados. Na segunda abordagem, o estado $(3,1)$ (respectivamente, $(1,3)$) não poderá ser alcançado quando a ocorrência do evento 13 (respec., 23) não for observada no estado $(2,0)$ (respec., $(0,2)$). No entanto, quando essa ocorrência for observada o evento 21 (respec., 11) será liberado pelo supervisor e o estado $(3,1)$ (respec., $(1,3)$) poderá ser alcançado. Na prática isso

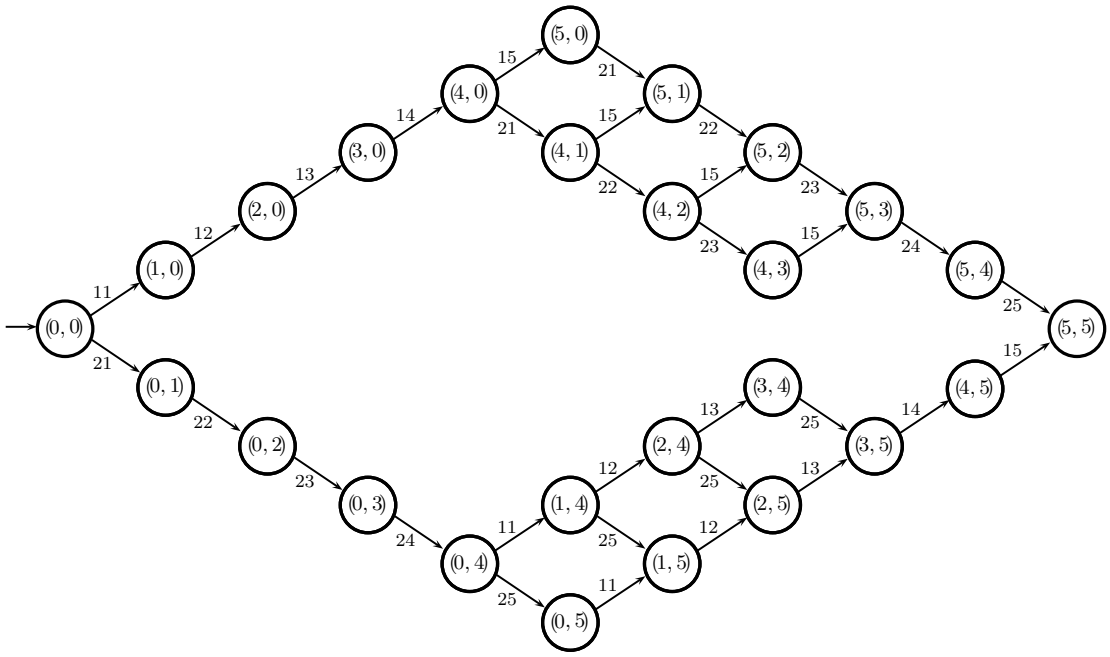


Figura 4.14: Diagrama de transição de estados do autômato H_{sto}, cuja linguagem gerada é $K_{sto}^{\uparrow CO}$.

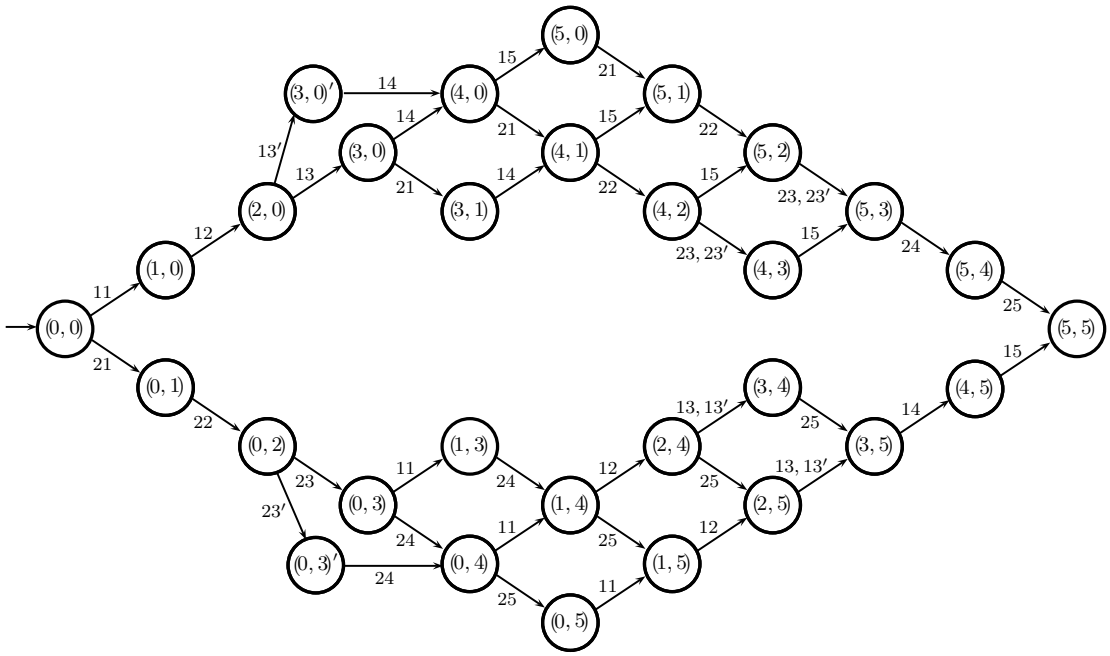


Figura 4.15: Diagrama de transição de estados do autômato H_{rob} que gera a linguagem $D(K)^{\uparrow CO}$.

pode ser interpretado da seguinte forma. Suponha que o trem T_1 saiu da estação A primeiro. Quando a entrada do trem T_1 no setor 3 da ferrovia for observada, o trem T_2 será liberado pelo supervisor para entrar no setor 2 da ferrovia. Caso contrário, ou seja, quando a entrada do trem T_1 no setor 3 não for observada, o trem T_2 deverá esperar o trem T_1 entrar no setor 4 para poder entrar no setor 2. Caso o trem T_2

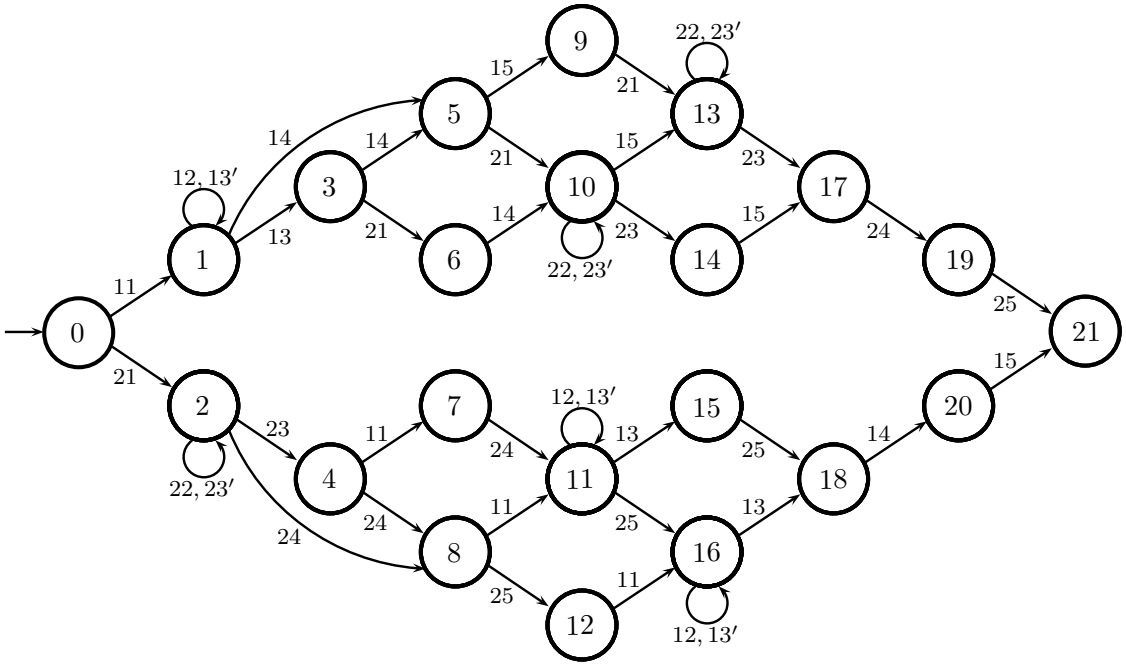


Figura 4.16: Diagrama de transição de estados do autômato que realiza o supervisor S_{rob} para o qual $L(S_{rob}/G_{dil}) = D(K)^{\uparrow CO}$.

saia primeiro da estação A, o comportamento é análogo ao descrito anteriormente.

Por fim, o supervisor S_{rob} obtido pela segunda abordagem, para o qual o comportamento em malha fechada é tal que $L(S_{rob}/G_{dil}) = D(K)^{\uparrow CO}$, é realizado pelo autômato cujo diagrama de transição de estados é apresentado ma figura 4.16. Note que os eventos não observáveis presentes no realizador de S_{rob}, incluindo os eventos pertencentes a Σ'_{ilo}, são decorrentes do algoritmo utilizado para gerar esse realizador (apresentado na seção 3.3).

Observação 1 *Vale ressaltar que, a depender de quais eventos estiverem sujeitos a perdas intermitentes de observação, a hipótese* ***A2*** *($\Sigma_{ilo} \cap \Sigma_c = \emptyset$) pode ser violada. Isso é decorrente do fato de que os eventos controláveis presentes no exemplo apresentado nesta seção não são eventos de comando. Por exemplo, o evento* 11 *simboliza a entrada do trem T_1 no setor* 1 *da ferrovia, enquanto que o comando está relacionado ao acionamento do semáforo nessa transição. Não deixa de ser razoável a suposição de que o comando de acionamento do semáforo da entrada do setor* 1 *sempre funcione bem, porém o sensor que detecta a passagem de trens nessa transição esteja sujeito a perdas intermitentes de observação. Nessa situação, o evento* 11 *seria controlável e sujeito a perdas intermitentes de observação.*

Capítulo 5

Conclusões e trabalhos futuros

Nesse trabalho, foi formulado um novo problema de controle supervisório robusto, no qual a robustez está relacionada a perdas intermitentes de observação. Foram propostas soluções para esse problema e definidas as condições de existência dessas soluções.

Nas primeiras etapas do trabalho, foram propostas contribuições ao formalismo proposto por CARVALHO *et al.* [16] para modelar SED passíveis de perdas de observações (operação de dilatação). Tais contribuições consistiram em quinze novas propriedades da dilatação. Algumas dessas propriedades descrevem a interação da dilatação com projeções e com a operação de contração. A operação de contração foi definida no decorrer desse trabalho, com o objetivo de criar uma função que operasse no sentido inverso ao da dilatação.

Dentre as contribuições principais desse trabalho, foram apresentadas duas novas definições de observabilidade robusta.

A primeira das novas definições de observabilidade robusta, a *observabilidade robusta forte*, é feita nos mesmos moldes da definição de diagnosticabilidade robusta, proposta por CARVALHO *et al.* [16]. Por meio dessa definição, foi possível estabelecer as condições necessárias e suficientes para a existência de um supervisor robusto S_{rob}, tal que $L(S_{rob}/G) = K$ e $L(S_{rob}/G_{dil}) = D(K)$, para uma dada linguagem prefixo fechada K, $K \subseteq L$. No entanto, demonstrou-se que essa observabilidade é consideravelmente restritiva.

A segunda definição, *observabilidade robusta fraca*, foi proposta com o objetivo de relaxar as condições referentes ao supervisor S_{rob} desejado. Com base nessa nova definição, foi possível estabelecer as condições necessárias e suficientes para a existência de um supervisor robusto S_{rob}, tal que $L(S_{rob}/G) = K$ e $L(S_{rob}/G_{dil}) \subseteq D(K)$, para uma dada linguagem prefixo fechada K, $K \subseteq L$. Além disso, também foi apresentado um método para projetar um supervisor robusto no sentido da definição de controlabilidade e observabilidade robusta fraca. A linguagem obtida pela abordagem proposta se mostrou mais permissiva do que as obtidas por abordagens en-

contradas na literatura. A viabilidade da teoria proposta foi ilustrada aplicando-a ao projeto de um supervisor robusto para um sistema simplificado de controle de tráfego ferroviário.

Entre os trabalhos futuros que podem ser considerados, destacamos: *(i)* a extensão da teoria proposta para levar em conta questões relacionadas a bloqueios em presença de perdas intermitentes de observação; *(ii)* problemas em que a ação de controle imposta pelo supervisor não seja executada, o que é equivalente a remover a hipótese **A2**.

Referências Bibliográficas

[1] RAMADGE, P., WONHAM, W. "Supervisory control of a class of discrete event process", *SIAM Journal of Control Optim.*, v. 25, n. 1, pp. 206–230, 1987.

[2] LIN, F. "Robust and Adaptive Supervisory Control of Discrete Event Systems", *IEEE Transactions on Automatic Control*, v. 38, n. 12, pp. 1848–1852, 1993.

[3] CURY, J., KROGH, B. "Robustness of Supervisors for Discrete-Event Systems", *IEEE Transactions on Automatic Control*, v. 44, pp. 376–379, 1999.

[4] PARK, S., LIM, J. "Robust and Nonblocking Supervisor for Discrete-Event Systems with Model Uncertainty Under Partial Observation", *IEEE Transactions on on Automatic Control*, v. 45, n. 12, pp. 2393–2396, 2000.

[5] TAKAI, S. "Synthesis of Maximally Permissive and Robust Supervisors for Prefix-Closed Language Specifications", *IEEE Transactions on Automatic Control*, v. 47, n. 1, pp. 132–136, 2002.

[6] BOURDON, S., LAWFORD, M., WONHAM, W. "Robust Nonblocking Supervisory Control of Discrete-Event Systems". In: *Proceedings of the 2002 American Control Conference*, pp. 730–735, 2002.

[7] TAKAI, S. "Maximazing Robustness of Supervisors for Partially Observed Discrete Event Systems", *Automatica*, v. 40, n. 3, pp. 531–535, 2004.

[8] ROHLOFF, K. R. "Sensor Failure Tolerant Supervisory Control". In: *44th IEEE Conference on Decision and Control and European Control Conference*, pp. 3493–3498, Seville, Spain, 2005.

[9] SABOORI, A., HASHTRUDI-ZAD, S. "Robust Nonblocking Supervisory Control of Discrete-Event Systems Under Partial Observation", *Systems & Control Letters*, v. 55, pp. 839–848, 2006.

[10] SANCHEZ, A. M., MONTOYA, F. J. "Safe supervisory control under observability failure", *Discrete Event Dynamic Systems-Theory And Applications*, v. 16, n. 4, pp. 493–525, 2006.

[11] ROHLOFF, K. R. "Bounded sensor Failure Tolerant Supervisory Control". In: *Preeprints of the 11th Workshop on Discrete Event Systems*, pp. 272–277, Guadalajara, Mexico, 2012.

[12] PAOLI, A., LAFORTUNE, S. "Safe diagnosability for fault-tolerant supervision of discrete-event systems", *Automatica*, v. 41, n. 8, pp. 1335–1347, 2005.

[13] BASILIO, J. C., LAFORTUNE, S. "Robust codiagnosability of discrete event systems". In: *Proc. of the American Control Conference*, pp. 2202–2209, St. Louis, Missouri, 2009.

[14] ATHANASOPOULOU, E., LINGXI, L., HADJICOSTIS, C. "Maximum Likelihood Failure Diagnosis in Finite State Machines Under Unreliable Observations", *IEEE Trans. on Automatic Control*, v. 55, n. 3, pp. 579 –593, 2010.

[15] CARVALHO, L. K., MOREIRA, M. V., BASILIO, J. C. "Generalized robust diagnosability of discrete event systems". In: *Proc. of 18th IFAC World Congress*, pp. 8737–8742, Milan, Italy, 2011.

[16] CARVALHO, L. K., BASILIO, J. C., MOREIRA, M. V. "Robust diagnosis of discrete-event systems against intermittent loss of observations", *Automatica*, v. 48, n. 9, pp. 2068–2078, 2012.

[17] TAKAI, S. "Verification of robust diagnosability for partially observed discrete event systems", *Automatica*, v. 48, n. 8, pp. 1913–1919, 2012.

[18] CARVALHO, L. K., MOREIRA, M. V., BASILIO, J. C., et al. "Robust diagnosis of discrete-event systems against permanent loss of observations", *Automatica*, v. 49, n. 1, pp. 223–231, 2013.

[19] PARK, S., LIM, J. "Non-blocking supervision for uncertain discrete event systems with internal unobservable transitions.pdf". In: *IEE Proc. Control Theory Appl.*, 2005.

[20] CASSANDRAS, C. G., LAFORTUNE, S. *Introduction to Discrete Event Systems*. 2nd ed. New York, Springer, 2008.

[21] OGATA, K. *Engenharia de Controle Moderno*. São Paulo, Brasil, Pearson Prentice Hall, 2010.

[22] MURATA, T. "Petri nets: Properties, analysis, and applications". In: *Proc. IEEE*, v. 77, pp. 541–580, 1989.

[23] DAVID, R., ALLA, H. *Discrete, continuous and hybrid Petri nets*. Springer, 2005.

[24] HOPCROFT, J. E., MOTWANI, R., ULLMAN, J. D. *Introduction to Automata Theory, Languages, and Computation (3rd Edition)*. Boston, MA, USA, Addison-Wesley Longman Publishing Co., Inc., 2006.

[25] RAMADGE, P., WONHAM, W. "The control of discrete event systems", *Proceedings of the IEEE*, v. 77, n. 1, 1989.

[26] BOUZAN, B. P. *Algoritmos em tempo polinomial para verificação da observabilidade e normalidade de linguagens regulares*. Tese de Mestrado, Universidade Federal do Rio de Janeiro, 2013.

[27] FA, J., YANG, X., ZHENG, Y. "Formulas for a class of controllable and observable sublanguages larger than the supremal controllable and normal sublanguage", *Systems & Control Letters*, v. 20, n. 1, pp. 11–18, 1993.

[28] HEYMANN, M., LIN, F. "On-line control of partially observed discrete event systems", *Discrete Event Dynamic Systems: Theory and Applications*, v. 4, n. 3, pp. 221–236, 1994.

[29] BEN HADJ-ALOUANE, N.¨AND LAFORTUNE, S., LIN, F. "Centralized and distributed algorithms for on-line synthesis of maximal control policies under partial observation", *Discrete Event Dynamic Systems: Theory and Applications*, v. 6, pp. 379–427, 1996.

[30] PROSSER, J. H., KAM, M., KWATNY, H. G. "Online supervisor synthesis for partially observed discrete-event systems", *IEEE Transactions on Automatic Control*, v. 43, n. 11, pp. 1630–1634, 1998.

[31] USHIO, T. "On-line control of discrete event systems with a maximally controllable and observable sublanguage", *IEICE Transactions on Fundamentals*, v. A, n. 9, pp. 1965–1970, 1999.

[32] WHONHAM, W. M. "Supervisory control of discrete-event systems". 2013. Disponível em: <`http://www.control.utoronto.ca/cgi-bin/dldes.cgi`>. Acessado em 15/01/2014.