UNIVERSIDADE FEDERAL DE SANTA CATARINA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA DE AUTOMAÇÃO E SISTEMAS

Germano Schafaschek

UMA ABORDAGEM LOCAL PARA O CONTROLE SUPERVISÓRIO MODULAR DE SISTEMAS A EVENTOS DISCRETOS TEMPORIZADOS

Florianópolis

2014

Germano Schafaschek

UMA ABORDAGEM LOCAL PARA O CONTROLE SUPERVISÓRIO MODULAR DE SISTEMAS A EVENTOS DISCRETOS TEMPORIZADOS

Dissertação submetida ao Programa de Pós-Graduação em Engenharia de Automação e Sistemas para a obtenção do Grau de Mestre em Engenharia de Automação e Sistemas.
Orientador: Max Hering de Queiroz
Coorientador: José Eduardo Ribeiro Cury

Florianópolis

2014

Ficha de identificação da obra elaborada pelo autor,
através do Programa de Geração Automática da Biblioteca Universitária da UFSC.

Schafaschek, Germano
Uma abordagem local para o controle supervisório modular de sistemas a eventos discretos temporizados / Germano Schafaschek ; orientador, Max Hering de Queiroz ; coorientador, José Eduardo Ribeiro Cury. - Florianópolis, SC, 2014.
94 p.

Dissertação (mestrado) - Universidade Federal de Santa Catarina, Centro Tecnológico. Programa de Pós-Graduação em Engenharia de Automação e Sistemas.

Inclui referências

1. Engenharia de Automação e Sistemas. 2. Controle Supervisório. 3. Sistemas a Eventos Discretos Temporizados. 4. Controle Modular. 5. Sistemas Compostos. I. Queiroz, Max Hering de. II. Cury, José Eduardo Ribeiro. III. Universidade Federal de Santa Catarina. Programa de Pós-Graduação em Engenharia de Automação e Sistemas. IV. Título.

Germano Schafaschek

UMA ABORDAGEM LOCAL PARA O CONTROLE SUPERVISÓRIO MODULAR DE SISTEMAS A EVENTOS DISCRETOS TEMPORIZADOS

Esta Dissertação foi julgada aprovada para a obtenção do Título de "Mestre em Engenharia de Automação e Sistemas", e aprovada em sua forma final pelo Programa de Pós-Graduação em Engenharia de Automação e Sistemas.

Florianópolis, 24 de novembro de 2014.

Prof. Rômulo Silva de Oliveira
Coordenador do Programa de Pós-Graduação em Engenharia de Automação e Sistemas

---

Prof. Max Hering de Queiroz
Orientador

Prof. José Eduardo Ribeiro Cury
Coorientador

Banca Examinadora:

Prof. Max Hering de Queiroz
Presidente

Prof.ª Patrícia Nascimento Pena

Prof. Rômulo Silva de Oliveira

Prof. Eduardo Camponogara

Dr. Fabio Baldissera

Ao meu avô, Germano.

AGRADECIMENTOS

Aos Professores Max e Cury, agradeço pelo comprometimento com este trabalho, pelas inestimáveis contribuições através de ideias e de frutíferas discussões e pela cuidadosa revisão do presente documento.

Agradeço, também, à CAPES e ao CNPq pelo suporte financeiro ao longo do projeto.

Pelo apoio e incentivo incondicionais, agradeço à minha família, em especial à minha mãe, Liliane, e à Ana, minha namorada, quase noiva, mais do que esposa e para sempre companheira.

The time you enjoy wasting is not wasted time.

Bertrand Russell

RESUMO

Nesta dissertação, propomos uma abordagem para a síntese local de supervisores modulares no contexto de sistemas a eventos discretos temporizados. O objetivo é reduzir o custo computacional na aplicação da teoria de controle supervisório a essa classe de sistemas, haja vista a dificuldade imposta pela explosão no número de estados dos modelos em sistemas de grande porte. Em grande parte dos problemas complexos envolvendo sistemas temporizados, a planta a ser controlada é composta de subsistemas que operam de maneira assíncrona a menos do compartilhamento de um relógio global. Ademais, o comportamento desejado é normalmente colocado na forma de especificações de controle elementares, cada uma das quais visa sincronizar e restringir o comportamento de apenas alguns dos subsistemas da planta. A ideia da metodologia de controle aqui apresentada é explorar tanto a modularidade das especificações quanto a do próprio sistema. Nossos supervisores são calculados com base em modelos locais, construídos pela agregação dos subsistemas que são afetados por cada especificação. Isso leva a módulos de controle nos quais a ação dos supervisores é relativamente simples, baseada apenas em informações locais, o que facilita sua compreensão, implementação e modificação. Apresentamos condições necessárias e suficientes sob as quais a ação conjunta dos supervisores locais leva o sistema a um comportamento global não bloqueante e que cumpre as especificações de forma ótima (minimamente restritiva). Mostramos, ainda, que a abordagem proposta reduz o custo computacional quando comparada a outras existentes. Por fim, um exemplo de interesse prático e com rígidas restrições temporais é resolvido, ilustrando a aplicabilidade da metodologia proposta.

Palavras-chave: Controle supervisório. Sistemas a eventos discretos temporizados. Controle modular. Sistemas compostos. Sistemas de grande porte.

ABSTRACT

In this thesis, an approach is proposed for the local synthesis of modular supervisors in the context of timed discrete-event systems. The objective is to reduce computational costs for the application of timed supervisory control, in face of the hindrances imposed by state explosion in the models of large scale systems. In a wide variety of complex problems involving timing issues, the plant to be controlled is composed of subsystems that work asynchronously except for the sharing of a global clock. Moreover, the desired behavior for the plant is usually represented by a number of elementary control specifications, each of which attempts to restrict and synchronize the behavior of only some of the system's components. The idea of our control methodology is to explore the modularity of both the specifications and the system itself. Our supervisors are designed over local models, which are obtained by aggregating the subsystems affected by each specification. This results in control modules with relatively simple supervisory actions, based only on local information, which makes the supervisors easier to comprehend, implement, and modify. We present necessary and sufficient conditions under which the concurrent action of the local supervisors leads the system to a nonblocking global behavior that complies with the specifications in an optimal (minimally restrictive) way. We also show that the proposed strategy reduces computational efforts in comparison with existing ones. Finally, a practical problem with critical time restrictions is solved to exemplify an application of the proposed control methodology.

Keywords: Supervisory control. Timed discrete-event systems. Modular control. Composed systems. Large scale systems.

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

Sistemas projetados e construídos pelo ser humano, tais como os computacionais, de manufatura, de tráfego e de comunicação, vêm tendo crescente preponderância na sociedade moderna. Compartilham entre si a propriedade de que suas dinâmicas não são diretamente regidas pela passagem contínua do tempo, mas pela ocorrência de *eventos* que causam mudanças abruptas de estado em instantes discretos. Sistemas com essas características são chamados de *sistemas a eventos discretos* (SED) (CASSANDRAS; LAFORTUNE, 2010).

Ao contrário de fenômenos tratados em problemas clássicos de controle, como a evolução de um processo químico ou o deslocamento de um corpo no espaço, o comportamento de sistemas a eventos discretos não pode ser descrito por equações diferenciais ou a diferenças com base nas leis da física ou da química. Um dos paradigmas mais difundidos para sua modelagem é a teoria de linguagens e autômatos (HOPCROFT; MOTWANI; ULLMAN, 2007). Com base nessa teoria, foi iniciada por Ramadge e Wonham a *teoria de controle supervisório* (RAMADGE; WONHAM, 1987), na qual tanto a planta a ser controlada quanto a especificação do comportamento desejado podem ser modeladas por autômatos de estados finitos. De posse de tais modelos, a teoria de controle supervisório permite a síntese automática de supervisores, cuja ação de controle consiste em proibir (desabilitar) a ocorrência de eventos no sistema de modo a restringir o seu comportamento e levá-lo a respeitar o especificado.

Em grande parte dos sistemas complexos encontrados na prática, o comportamento global é resultante da operação paralela e independente (assíncrona) de subsistemas mais simples. Consequentemente, o autômato que representa esse comportamento é naturalmente obtido pela composição de autômatos menores e que não compartilham eventos entre si, resultando na chamada representação por *sistema produto* (RAMADGE; WONHAM, 1989; RAMADGE, 1989). É o caso, por exemplo, de máquinas trabalhando em uma linha de montagem ou célula de manufatura, computadores enviando e recebendo mensagens em uma rede de comunicação e semáforos em um sistema de controle de tráfego. Na ausência de uma ação coordenadora, cada componente desses sistemas trabalha alheio aos demais, cabendo, portanto, ao supervisor a incumbência de sincronizar suas operações. Essa sincronização é ditada pelo comportamento especificado para o sistema, o qual, por sua vez, é normalmente colocado na forma de especificações de controle

elementares, cada uma das quais visa sincronizar e restringir o comportamento de apenas alguns dos subsistemas da planta. Com a agregação de componentes da planta e de especificações, o número de estados dos modelos cresce em ritmo exponencial. O consequente aumento, por vezes exacerbado, na complexidade computacional dos procedimentos de síntese de supervisores é uma das principais limitações para a aplicação da teoria de controle supervisório em sistemas reais.

A incorporação do tempo no modelo de um SED aumenta a capacidade de modelagem e é essencial em aplicações temporalmente críticas. Possibilita a representação de limitações físicas, como atrasos de comunicação, bem como a imposição de prazos para a conclusão de tarefas, além de permitir a avaliação do desempenho do sistema. Em (BRANDIN; WONHAM, 1994), é proposto um modelo para *sistemas a eventos discretos temporizados* (SEDT), o qual considera que o tempo é medido por meio de um relógio digital global. Esse modelo herda os conceitos de controlabilidade e de máxima sublinguagem controlável da teoria de controle supervisório, tendo possibilitado que os principais resultados dessa teoria fossem estendidos ao contexto temporizado. No entanto, a inclusão de um evento adicional representando a passagem do tempo pode causar um aumento considerável no tamanho dos modelos, agravando ainda mais o problema da explosão de estados. Isso motivou o surgimento de diversos trabalhos buscando reduzir a complexidade computacional; em particular, muitos dos resultados desenvolvidos originalmente para SED não temporizados com esse mesmo propósito foram estendidos para o contexto temporizado.

Em (BRANDIN; WONHAM, 1993), os autores ampliam seus resultados anteriores, apresentando versões temporizadas das abordagens modular (WONHAM; RAMADGE, 1988) e descentralizada (LIN; WONHAM, 1988). Um método para o cálculo de supervisores reduzidos para SEDT é apresentado em (GOHARI; WONHAM, 2003); é utilizada uma abstração do modelo da planta, medindo o tempo com um relógio "mais lento". Apesar de simplificar a síntese de supervisores, essa proposta abre mão da otimidade. Saadatpoor e Wonham propõem uma síntese baseada em estados para supervisores temporizados, utilizando *binary decision diagrams* (BDD) (SAADATPOOR; WONHAM, 2007). Foi provado que explorar a estrutura da informação para obter uma representação adequada em BDD pode reduzir significativamente o custo computacional. Mais recentemente, uma abordagem chamada de *localização de supervisores* (CAI; WONHAM, 2010) foi estendida ao contexto temporizado (ZHANG et al., 2013). Seu foco está voltado ao controle de sistemas com uma arquitetura puramente distribuída, e os resultados

precisam ainda ser combinados com uma estratégia modular eficiente para que sejam aplicáveis a SEDT de maior porte.

Nesta dissertação, propomos uma abordagem modular local para o controle de SED temporizados, com o objetivo de reduzir a complexidade computacional na síntese de supervisores. Exploramos o fato de que, em muitos casos de interesse prático, a planta a ser controlada é composta por subsistemas assíncronos a menos do compartilhamento de um relógio global. Nossa abordagem consiste em obter modelos locais pela composição apenas dos subsistemas afetados por cada especificação. Supervisores são, então, calculados sobre esses modelos, impondo localmente o comportamento especificado. Buscamos, com isso, refinar a abordagem modular temporizada (BRANDIN; WONHAM, 1993), evitando computações custosas que envolvam o modelo global da planta. Além disso, é desejável que os supervisores resultantes possam ser representados por autômatos relativamente simples, em contraste com os modelos monolíticos e menos intuitivos obtidos, por exemplo, pela abordagem utilizando BDD (SAADATPOOR; WONHAM, 2007). Apresentamos condições que garantem que a ação conjunta dos supervisores locais leve o sistema à otimidade global e à ausência de bloqueio. Este trabalho consiste na extensão ao contexto temporizado dos resultados propostos por Queiroz e Cury para SED não temporizados (QUEIROZ, 2000; QUEIROZ; CURY, 2000, 2002).

A organização do restante do documendo se dá como segue.

No **Capítulo 2**, é revisado o modelo de Brandin e Wonham para SED temporizados, iniciando com uma breve apresentação de algumas definições básicas de linguagens e autômatos. São formalmente definidos os conceitos de supervisor e de controlabilidade de linguagens.

O **Capítulo 3** apresenta uma visão intuitiva do conceito de supervisão modular local, contextualizando e comparando com outras abordagens.

No **Capítulo 4**, são apresentadas as contribuições desta dissertação. O problema de controle é formulado, e os resultados formais que sedimentam o controle modular local de SEDT são enunciados e demonstrados. Uma análise de complexidade é realizada para avaliar a eficiência da abordagem proposta, e é sugerido um método para a resolução de conflitos.

No **Capítulo 5**, um exemplo envolvendo um tipo de sistema bem difundido na indústria é resolvido para ilustrar a aplicabilidade da metodologia apresentada.

Por fim, no **Capítulo 6** são colocadas as conclusões e os comentários finais, bem como perspectivas para futuros trabalhos na área.

2 SISTEMAS A EVENTOS DISCRETOS TEMPORIZADOS

Neste capítulo, é revisado o modelo de SEDT proposto em (BRANDIN; WONHAM, 1994). Uma vez que esse modelo se baseia na teoria de linguagens e autômatos, são brevemente apresentadas algumas definições; uma introdução mais didática e detalhada ao assunto pode ser encontrada em (CASSANDRAS; LAFORTUNE, 2010; HOPCROFT; MOTWANI; ULLMAN, 2007).

2.1 LINGUAGENS E AUTÔMATOS

Um *alfabeto* Σ é um conjunto não vazio e finito de símbolos. Uma sequência finita de símbolos de um alfabeto é chamada de *cadeia*. Diz-se que uma cadeia s tem *comprimento* $n \in \mathbb{N}$, denotado por $|\, s \,| = n$, caso seja formada por n símbolos (não necessariamente distintos). A *cadeia vazia* ε é aquela que possui comprimento nulo. Define-se Σ^+ como o conjunto de todas as cadeias não vazias formadas por símbolos de Σ, e $\Sigma^* = \Sigma^+ \cup \{\varepsilon\}$. Dadas cadeias $s, u \in \Sigma^*$, u é dita ser *prefixo* de s se $\exists v \in \Sigma^* \,|\, s = uv$.

Uma *linguagem* L sobre Σ é qualquer subconjunto de Σ^*. O *prefixo-fechamento* de L é $\overline{L} = \{u \in \Sigma^* \mid \exists v \in \Sigma^*, uv \in L\}$. Para qualquer $s \in \overline{L}$, define-se o conjunto $\Sigma_L(s) = \{\sigma \in \Sigma \mid s\sigma \in \overline{L}\}$.

Dado um alfabeto $\Sigma_i \subseteq \Sigma$, a *projeção natural* $P_i : \Sigma^* \to \Sigma_i^*$ é definida recursivamente da seguinte maneira: $\forall \sigma \in \Sigma$ e $\forall s \in \Sigma^*$,

$$P_i(\varepsilon) = \varepsilon\,;$$
$$P_i(s\sigma) = \begin{cases} P_i(s) & \text{se } \sigma \notin \Sigma_i\,, \\ P_i(s)\sigma & \text{se } \sigma \in \Sigma_i\,. \end{cases}$$

Essa definição pode ser estendida para linguagens. Denotando por 2^Y o conjunto das partes de Y, $P_i : 2^{\Sigma^*} \to 2^{\Sigma_i^*}$ é definida por

$$P_i(L) = \{s_i \in \Sigma_i^* \mid \exists s \in L, P_i(s) = s_i\}\,.$$

Define-se, ainda, a projeção inversa $P_i^{-1} : 2^{\Sigma_i^*} \to 2^{\Sigma^*}$ como

$$P_i^{-1}(L_i) = \{s \in \Sigma^* \mid P_i(s) \in L_i\}\,.$$

Considere uma coleção de alfabetos Σ_i, $i \in \{1, \ldots, n\}$, e seja

$\Sigma = \bigcup_{i=1}^{n} \Sigma_i$. Dadas linguagens $L_i \subseteq \Sigma_i^*$, seu *produto síncrono* é definido como

$$\Big|\Big|_{i=1}^{n} L_i = \bigcap_{i=1}^{n} P_i^{-1}(L_i)\,.$$

Um *autômato* (de estados finitos) é uma quíntupla

$$\mathbf{A} = (X, \Sigma, \xi, x_0, X_m)\,,$$

na qual X é um conjunto finito de estados, Σ é um alfabeto, $\xi : X \times \Sigma \to X$ é uma função parcial de transição de estados, definida em cada $x \in X$ para um subconjunto de Σ, x_0 é o estado inicial e $X_m \subseteq X$ é o conjunto de estados marcados. Escreve-se $\xi(x,\sigma)!$ caso $\xi(x,\sigma)$ esteja definida e $\xi(x,\sigma)\not!$ caso contrário. A função ξ pode ser generalizada para cadeias, sendo então dada por $\xi : X \times \Sigma^* \to X$ e definida recursivamente como: $\forall x \in X$, $\forall s \in \Sigma^*$ e $\forall \sigma \in \Sigma$,

$$\begin{aligned} &\xi(x,\varepsilon) = x\,; \\ &\xi(x,s\sigma) = \xi\big(\xi(x,s),\sigma\big)\,. \end{aligned}$$

Naturalmente, $\xi(x,\varepsilon)!$ para qualquer $x \in X$ e

$$\xi(x,s\sigma)! \;\Leftrightarrow\; \xi(x,s)! \;\&\; \xi\big(\xi(x,s),\sigma\big)!\,.$$

Definem-se a *linguagem gerada*

$$L(\mathbf{A}) = \{s \in \Sigma^* \,|\, \xi(x_0,s)!\}$$

e a *linguagem marcada*

$$L_m(\mathbf{A}) = \{s \in \Sigma^* \,|\, \xi(x_0,s) \in X_m\}\,.$$

Por simplicidade, para qualquer cadeia $s \in L(\mathbf{A})$ e para $x = \xi(x_0,s) \in X$, denotaremos $\Sigma_{\mathbf{A}}(s) = \Sigma_{\mathbf{A}}(x) = \Sigma_{L(\mathbf{A})}(s)$.

Um estado $x \in X$ é dito *acessível* se $\exists s \in \Sigma^* \,|\, \xi(x_0,s) = x$, e é *coacessível* se $\exists s \in \Sigma^* \,|\, \xi(x,s) \in X_m$. A *componente acessível* de $\mathbf{A}$, $Ac(\mathbf{A})$, é obtida eliminando-se todos os seus estados não acessíveis; obtém-se a *componente coacessível* $CoAc(\mathbf{A})$ de forma análoga. A *componente trim* é dada por

$$Trim(\mathbf{A}) = CoAc\big(Ac(\mathbf{A})\big) = Ac\big(CoAc(\mathbf{A})\big)\,.$$

O autômato $\mathbf{A}$ é trim se $Trim(\mathbf{A}) = \mathbf{A}$.

Para quaisquer dois autômatos $\mathbf{A}_1 = (X_1, \Sigma_1, \xi_1, x_{1_0}, X_{1_\mathrm{m}})$ e $\mathbf{A}_2 = (X_2, \Sigma_2, \xi_2, x_{2_0}, X_{2_\mathrm{m}})$, define-se a sua *composição síncrona* como

$$\mathbf{A}_1 \,||\, \mathbf{A}_2 = Ac\big(X_1 \times X_2\,,\ \Sigma_1 \cup \Sigma_2\,,\ \xi_{12}\,,\ (x_{1_0}, x_{2_0})\,,\ X_{1_\mathrm{m}} \times X_{2_\mathrm{m}}\big)\,;$$

a função ξ_{12} é definida, para quaisquer $x = (x_1, x_2) \in X_1 \times X_2$ e $\sigma \in \Sigma_1 \cup \Sigma_2$, como

$$\xi_{12}(x,\sigma) = \begin{cases} \big(\xi_1(x_1,\sigma), \xi_2(x_2,\sigma)\big) & \text{se } \sigma \in \Sigma_{\mathbf{A}_1}(x_1) \cap \Sigma_{\mathbf{A}_2}(x_2)\,; \\ \big(\xi_1(x_1,\sigma), x_2\big) & \text{se } \sigma \in \Sigma_{\mathbf{A}_1}(x_1) - \Sigma_2\,; \\ \big(x_1, \xi_2(x_2,\sigma)\big) & \text{se } \sigma \in \Sigma_{\mathbf{A}_2}(x_2) - \Sigma_1\,; \\ \text{indefinida} & \text{caso contrário}\,. \end{cases}$$

Dessa maneira, tem-se $L(\mathbf{A}_1 \,||\, \mathbf{A}_2) = L(\mathbf{A}_1) \,||\, L(\mathbf{A}_2)$ e $L_m(\mathbf{A}_1 \,||\, \mathbf{A}_2) = L_m(\mathbf{A}_1) \,||\, L_m(\mathbf{A}_2)$. A generalização da composição para um número arbitrário de autômatos é imediata. Note que, aqui e ao longo desta dissertação, o símbolo $||$ é utilizado para representar tanto a operação de produto síncrono de linguagens quanto a de composição síncrona de autômatos, ficando clara a distinção de acordo com o contexto.

2.2 O MODELO BRANDIN-WONHAM

Conforme (BRANDIN; WONHAM, 1994), para definir um sistema a eventos discretos temporizado $\mathbf{G}$ parte-se de um autômato

$$\mathbf{G}_\mathrm{at} = (A, \Sigma_\mathrm{at}, \delta_\mathrm{at}, a_0, A_m)\,,$$

que corresponde ao modelo de um SED não temporizado. Aqui, A é chamado de conjunto de *atividades*, e os elementos do alfabeto Σ_at correspondem aos *eventos* do sistema. Para introduzir o tempo em $\mathbf{G}_\mathrm{at}$, imagina-se medi-lo por meio de um relógio digital global. Um novo evento *tick* é então introduzido para representar a passagem de uma unidade de tempo do relógio, o qual assume, portanto, valores temporais discretos, determinados pelo número de ocorrências de *tick*. O conjunto completo de eventos fica dado por $\Sigma = \Sigma_\mathrm{at} \cup \{tick\}$.

Com o intuito de relacionar a passagem do tempo com os eventos "normais" do sistema, associa-se a cada $\sigma \in \Sigma_\mathrm{at}$ um *limite temporal inferior* $\ell_\sigma \in \mathbb{N}$ e um *limite temporal superior* $u_\sigma \in \mathbb{N} \cup \{\infty\}$; esses correspondem, respectivamente, aos tempos mínimo e máximo de espera antes da ocorrência de σ, contados a partir da sua habilitação em $\mathbf{G}_\mathrm{at}$. Tipicamente, ℓ_σ representa um atraso de comunicação ou

de resposta a um comando, enquanto u_σ representa um prazo determinado por uma especificação de controle ou por uma restrição física do próprio sistema. De acordo com o valor de seus limites temporais superiores, os eventos são divididos em dois grupos: eventos *remotos* $\Sigma_{\text{rem}} = \{\sigma \in \Sigma_{\text{at}} \,|\, u_\sigma = \infty\}$ (aqueles que podem ser indefinidamente postergados) e eventos esperados $\Sigma_{\text{esp}} = \{\sigma \in \Sigma_{\text{at}} \,|\, u_\sigma < \infty\}$ (aqueles que, uma vez habilitados, são limitados a ocorrer dentro de um horizonte temporal finito). Isso resulta na partição $\Sigma_{\text{at}} = \Sigma_{\text{rem}} \dot{\cup} \Sigma_{\text{esp}}$.

A maneira pela qual os limites temporais restringem a ocorrência de um evento deve, naturalmente, ser interpretada em termos do relógio do sistema, ou seja, em termos do número de ocorrências de *tick*. Para tal, é utilizado um *cronômetro* τ_σ para cada $\sigma \in \Sigma_{\text{at}}$, o qual assume valores em um intervalo $T_\sigma \subseteq \mathbb{N}$ dado por

$$T_\sigma = \begin{cases} [0, \ell_\sigma] & \text{se } \sigma \in \Sigma_{\text{rem}}\,; \\ [0, u_\sigma] & \text{se } \sigma \in \Sigma_{\text{esp}}\,. \end{cases}$$

O valor inicial de τ_σ é determinado pelo limite superior de T_σ, ou seja,

$$\tau_\sigma^0 = \begin{cases} \ell_\sigma & \text{se } \sigma \in \Sigma_{\text{rem}}\,; \\ u_\sigma & \text{se } \sigma \in \Sigma_{\text{esp}}\,. \end{cases}$$

Mediante a habilitação de um evento em $\mathbf{G}_{\text{at}}$, o cronômetro correspondente inicia uma contagem regressiva a partir do seu valor inicial, decrementando uma unidade a cada ocorrência de *tick*. Essa contagem prosseguirá enquanto o evento continuar habilitado. Tanto a desabilitação quanto a ocorrência de um evento fazem com que seu cronômetro volte para o valor inicial. Qualquer evento σ torna-se *elegível* a ocorrer no sistema caso permaneça habilitado durante ℓ_σ ocorrências de *tick*, i. e., caso seu cronômetro seja decrementado ℓ_σ vezes. Isso significa que um evento $\sigma \in \Sigma_{\text{rem}}$ passa a estar elegível quando $\tau_\sigma = 0$, enquanto para $\sigma \in \Sigma_{\text{esp}}$ o mesmo ocorre quando $\tau_\sigma = u_\sigma - \ell_\sigma$. Um evento remoto pode permanecer elegível indefinidamente; seu cronômetro, tendo atingido o valor nulo, interrompe a contagem e mantém esse valor enquanto o evento esteja habilitado e não ocorra. Em contrapartida, um evento esperado σ torna-se *iminente* caso permaneça habilitado por u_σ unidades de tempo ou, equivalentemente, caso se mantenha elegível por um período igual a $u_\sigma - \ell_\sigma$, o que significa que seu cronômetro atingiu valor nulo. Nesse caso, o evento deverá ocorrer antes do próximo *tick* do relógio, a menos que seja desabilitado nesse ínterim pela ocorrência de outro evento de Σ_{at}.

A discussão acima esboça o comportamento dinâmico do sistema

temporizado. Para torná-la formal, é necessário definir a função de transição de estados de $\mathbf{G}$. Cada *estado* possui a forma

$$q = \left(a, \{\tau_\sigma \,|\, \sigma \in \Sigma_{\text{at}}\}\right),$$

com $a \in A$. Note que um estado contém, além de uma atividade a, os valores dos cronômetros associados a cada um dos eventos de Σ_{at}. Em particular, o estado inicial é

$$q_0 = \left(a_0, \{\tau_\sigma^0 \,|\, \sigma \in \Sigma_{\text{at}}\}\right).$$

Dessa forma, o conjunto de estados é dado por

$$Q = A \times \prod \{T_\sigma \mid \sigma \in \Sigma_{\text{at}}\}. \tag{2.1}$$

É definida, então, a função $\delta : Q \times \Sigma \to Q$ de modo que, para quaisquer $q \in Q$ e $\sigma \in \Sigma$, $\delta(q, \sigma)!$ se e somente se

i. $\sigma = \textit{tick}$ e nenhum evento esperado está iminente, i. e., $\forall \rho \in \Sigma_{\text{esp}}$, $\delta_{\text{at}}(a, \rho)! \Rightarrow \tau_\rho > 0$; ou

ii. $\sigma \in \Sigma_{\text{rem}}$ e σ está elegível em q, i. e., $\delta_{\text{at}}(a, \sigma)!$ e $\tau_\sigma = 0$; ou

iii. $\sigma \in \Sigma_{\text{esp}}$ e σ está elegível em q, i. e., $\delta_{\text{at}}(a, \sigma)!$ e $0 \leq \tau_\sigma \leq u_\sigma - \ell_\sigma$.

Assim como em um autômato qualquer, essa definição pode ser estendida para cadeias de eventos (ver Seção 2.1). O estado de destino $q' = \delta(q, \sigma) = \left(a', \{\tau'_\sigma \,|\, \sigma \in \Sigma_{\text{at}}\}\right)$, alcançado mediante a ocorrência de um evento $\sigma \in \Sigma$, é definido da seguinte maneira:

i. se $\sigma = \textit{tick}$, então $a' = a$ e, $\forall \rho \in \Sigma_{\text{at}}$,

$$\text{se } \rho \in \Sigma_{\text{rem}}, \qquad \tau'_\rho = \begin{cases} \tau_\rho^0 = \ell_\rho & \text{se } \delta_{\text{at}}(a, \rho)!\,; \\ \tau_\rho - 1 & \text{se } \delta_{\text{at}}(a, \rho)! \text{ e } \tau_\rho > 0\,; \\ 0 & \text{se } \delta_{\text{at}}(a, \rho)! \text{ e } \tau_\rho = 0\,; \end{cases}$$

$$\text{se } \rho \in \Sigma_{\text{esp}}, \qquad \tau'_\rho = \begin{cases} \tau_\rho^0 = u_\rho & \text{se } \delta_{\text{at}}(a, \rho)!\,; \\ \tau_\rho - 1 & \text{se } \delta_{\text{at}}(a, \rho)! \text{ e } \tau_\rho > 0\,. \end{cases}$$

(Lembre que, se $\rho \in \Sigma_{\text{esp}}$, não é necessário considerar o caso $\tau_\rho = 0$, pois isso implicaria que $\delta(q, \textit{tick})!$.)

ii. se $\sigma \in \Sigma_{\text{at}}$, então $a' = \delta_{\text{at}}(a, \sigma)$, $\tau'_\sigma = \tau_\sigma^0$ e, $\forall \rho \in \Sigma_{\text{at}} - \{\sigma\}$,

$$\tau'_\rho = \begin{cases} \tau_\rho^0 & \text{se } \delta_{\text{at}}(a', \rho)!\,; \\ \tau_\rho & \text{se } \delta_{\text{at}}(a', \rho)!\,. \end{cases}$$

Finalmente, define-se o conjunto de estados marcados como

$$Q_m \subseteq A_m \times \prod \{T_\sigma \mid \sigma \in \Sigma_{\mathrm{at}}\}\,.$$

Veja que o fato de uma atividade a ser marcada em $\mathbf{G}_{\mathrm{at}}$ não implica que um estado da forma $q = (a, \_)$ também o seja em $\mathbf{G}$, sendo que a escolha de quais combinações de valores dos τ_σ resultam em estados marcados fica como critério de modelagem.

Com base nas definições apresentadas, um *sistema a eventos discretos temporizado* é definido como um autômato de estados finitos

$$\mathbf{G} = (Q, \Sigma, \delta, q_0, Q_m)\,.$$

Tal sistema é *não bloqueante* caso $L(\mathbf{G}) = \overline{L_m(\mathbf{G})}$, ou seja, se qualquer cadeia por ele gerada puder ser continuada até levar a um estado marcado.

Um SEDT $\mathbf{G}$ pode ser representado graficamente através de um *grafo de transição de atividades* (GTA), que consiste no grafo de transição relacionado a $\mathbf{G}_{\mathrm{at}}$, ou de um *grafo de transição temporizado* (GTT), correspondente ao grafo de transição de $\mathbf{G}$ (com a representação explícita de *tick*). Na Figura 1 é mostrado um exemplo de GTA; os limites dos intervalos próximos aos eventos representam os limites temporais a eles relacionados. O respectivo GTT aparece na Figura 2, em que o evento *tick* é denotado por t.

A partir desse exemplo, pode-se perceber que os cronômetros associados aos eventos do sistema não aparecem explicitamente em nenhuma das representações. Isso é um reflexo do fato de que tais cronômetros são mecanismos "internos", cuja única função, no que concerne ao modelo, é servir de base para a definição do conjunto de estados Q e da função de transição δ. Essa, uma vez definida, conterá todas as informações sobre as restrições temporais para a ocorrência dos eventos, e a contagem dos cronômetros ficará implícita na dinâmica do sistema. Para tornar a ideia mais clara, a Tabela 1 mostra os valores dos cronômetros referentes ao sistema das Figuras 1 e 2 para algumas das cadeias por ele geradas. Note, por exemplo, que a ocorrência de *tick* logo após β faz com que τ_α e τ_γ sejam decrementados, uma vez que α e γ são os eventos habilitados em $\mathbf{G}_{\mathrm{at}}$. Como isso leva a $\tau_\alpha = u_\alpha - \ell_\alpha$ e $\tau_\gamma = 0$, esses dois eventos tornam-se, então, elegíveis em $\mathbf{G}$. Esse mesmo fato está embutido na definição da função δ, pois pela Figura 2 fica claro que $\delta(q_0, \beta\alpha)!$ e $\delta(q_0, \beta\gamma)!$, enquanto $\delta(q_0, \beta t\alpha)!$ e $\delta(q_0, \beta t\gamma)!$. Pelas definições apresentadas, é evidente que cadeias que levem ao mesmo estado resultarão em valores iguais dos cronômetros.

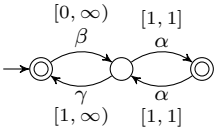

Figura 1 – Exemplo de GTA para um SEDT.

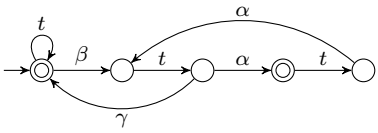

Figura 2 – GTT para o mesmo SEDT da Figura 1.

Tabela 1 – Valores dos cronômetros do sistema das Figuras 1 e 2.

Cadeia	$\{\tau_\alpha, \tau_\beta, \tau_\gamma\}$
ε	{1,0,1}
t	{1,0,1}
β	{1,0,1}
βt	{0,0,0}
$\beta t\gamma$	{1,0,1}
$\beta t\alpha$	{1,0,1}
$\beta t\alpha t$	{0,0,1}
$\beta t\alpha t\alpha$	{1,0,1}

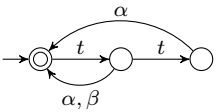

Figura 3 – Exemplo de GTT para o qual não há GTA equivalente.

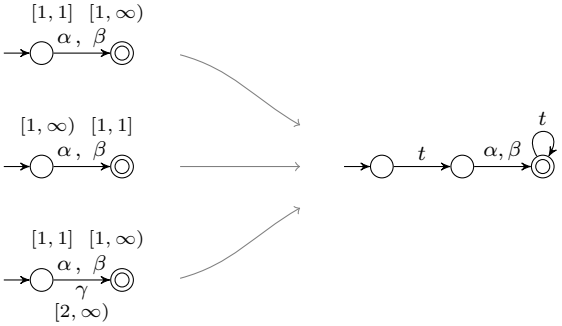

Figura 4 – Exemplo de GTT com múltiplos GTA equivalentes.

Qualquer GTA pode ser transformado em um único GTT equivalente (a menos de marcação), porém a recíproca não é verdadeira: existem GTT para os quais não há GTA equivalente — vide exemplo na Figura 3 — e GTT com múltiplos GTA equivalentes — Figura 4. Nesta dissertação, consideraremos que todo SEDT $\mathbf{G}$ é obtido a partir de um modelo não temporizado $\mathbf{G}_{\text{at}}$, de modo que para cada GTT haverá um GTA previamente especificado.

Observação 1. Em geral, para um SEDT $\mathbf{G}$, o conjunto de estados definido como em (2.1) possui estados não acessíveis, ainda que $\mathbf{G}_{\text{at}}$ seja um autômato trim. Isso ocorre porque nem todas as combinações de valores dos cronômetros são possíveis para cada atividade $a \in A$. Por exemplo, de acordo com a definição, para o SEDT da Figura 1 o estado $q = \big(a, \{\tau_\alpha, \tau_\beta, \tau_\gamma\}\big) = \big(a_0, \{1, 0, 0\}\big)$ pertence a Q, enquanto é facilmente verificado que não existe cadeia que leve a ele a partir do estado inicial $q_0 = \big(a_0, \{1, 0, 1\}\big)$. Entretanto, uma vez que para fins teóricos são de interesse apenas as linguagens gerada e marcada por $\mathbf{G}$, esse fato não terá relevância nos resultados desta dissertação. É evidente que, em termos computacionais, bem como em exemplos ilustrativos, é vantajoso que sejam considerados apenas os estados acessíveis.

Observação 2. De modo a manter a coerência física, a definição de um SEDT deve excluir a possibilidade fictícia de que uma sequência de

eventos seja repetida indefinidamente sem qualquer ocorrência de *tick*, ou seja, no intervalo de uma unidade de tempo. Em outras palavras, $\mathbf{G}$ deve ser *livre de loops de atividade* (BRANDIN; WONHAM, 1994), ou

$$\forall q \in Q,\ s \in \Sigma_{\text{at}}^{+} \ \Rightarrow\ \delta(q,s) \neq q\,. \tag{2.2}$$

Além disso, de acordo com a definição da função δ, se $\Sigma_{\mathbf{G}}(s) \cap \Sigma_{\text{at}} = \emptyset$ para algum $s \in L(\mathbf{G})$, então $tick \in \Sigma_{\mathbf{G}}(s)$. Esse fato, combinado com (2.2) e com o fato de Q ser um conjunto finito, implica que o relógio do sistema nunca para, o que pode ser formalizado como

$$\forall s \in L(\mathbf{G}), \exists u \in \Sigma^{*} \,|\, \Sigma_{\mathbf{G}}(su) = \{tick\}\,.$$

Observação 3. Como mencionado em (LIN; WONHAM, 1995), se a ocorrência de *tick* não for possível em $\mathbf{G}$ como continuação a uma cadeia s, então deverá haver pelo menos um evento esperado que esteja iminente após s. De acordo com a Observação 2, pode haver no máximo um número *finito* de eventos na sequência de s antes que *tick* possa ocorrer. Em suma, é possível concluir que, $\forall s \in L(\mathbf{G})$,

$$tick \notin \Sigma_{\mathbf{G}}(s) \ \Rightarrow\ \exists v \in \Sigma_{\text{esp}}^{+} \,\big|\, sv \in L(\mathbf{G}) \ \&\ tick \in \Sigma_{\mathbf{G}}(sv)\,.$$

2.3 COMPOSIÇÃO DE SEDT

A operação *comp* de composição de SEDT é definida em (BRANDIN; WONHAM, 1994) como segue. Dados dois SEDT $\mathbf{G}_1$ e $\mathbf{G}_2$,

$$\mathbf{G}_{12} = comp(\mathbf{G}_1, \mathbf{G}_2)$$

é o SEDT tal que

$$\mathbf{G}_{12_{\text{at}}} = \mathbf{G}_{1_{\text{at}}} \,||\, \mathbf{G}_{2_{\text{at}}}\,.$$

Os limites temporais dos eventos de $\mathbf{G}_{12}$ são determinados pela seguinte regra:

- se $\sigma \in (\Sigma_{1_{\text{at}}} - \Sigma_{2_{\text{at}}}) \cup (\Sigma_{2_{\text{at}}} - \Sigma_{1_{\text{at}}})$, ℓ_{12_σ} e u_{12_σ} permanecem inalterados;
- caso contrário, se $\sigma \in \Sigma_{1_{\text{at}}} \cap \Sigma_{2_{\text{at}}}$, então $\ell_{12_\sigma} = \max\{\ell_{1_\sigma}, \ell_{2_\sigma}\}$ e $u_{12_\sigma} = \min\{u_{1_\sigma}, u_{2_\sigma}\}$, desde que $\ell_{12_\sigma} \leq u_{12_\sigma}$.

Caso essa última condição seja violada, i. e., se a regra resultar em $\ell_{12_\sigma} > u_{12_\sigma}$ para algum evento σ, a composição é considerada indefinida. Uma maneira de interpretar essa definição é: (i) a ocorrência de

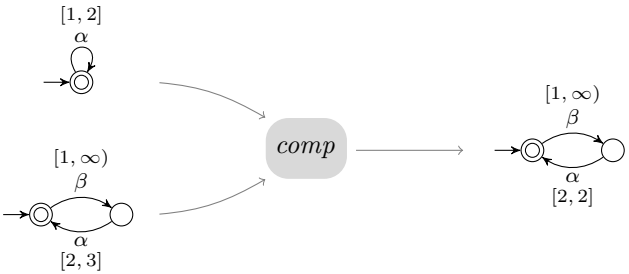

Figura 5 – Ilustração da operação *comp*.

um evento que seja exclusivo a um dos sistemas não é afetada pelo comportamento do outro sistema; (ii) o cronômetro de um evento compartilhado por $\mathbf{G}_1$ e $\mathbf{G}_2$ realiza sua contagem somente se o evento estiver habilitado em $\mathbf{G}_{1_{\text{at}}}$ e em $\mathbf{G}_{2_{\text{at}}}$; (iii) para que um evento compartilhado se torne elegível, é necessário que permaneça habilitado simultaneamente em *ambos* os sistemas por tempo suficiente para que fique elegível tanto em $\mathbf{G}_1$ quanto em $\mathbf{G}_2$; (iv) um evento $\sigma \in \Sigma_{1_{\text{esp}}} \cap \Sigma_{2_{\text{esp}}}$ torna-se iminente caso permaneça habilitado em $\mathbf{G}_{1_{\text{at}}}$ e em $\mathbf{G}_{2_{\text{at}}}$ por um período que o torne iminente em *pelo menos um* dos sistemas. A definição da operação *comp* vai, portanto, de acordo com o princípio de que, na sincronização de dois sistemas, um evento compartilhado pode ocorrer somente se as condições impostas para sua ocorrência em ambos os sistemas forem satisfeitas.

Como ilustração, na Figura 5 é mostrado o resultado da composição de dois SEDT. Chamemos os sistemas na parte esquerda da figura de $\mathbf{G}_1$ (superior) e $\mathbf{G}_2$ (inferior), e o sistema resultante (à direita) de $\mathbf{G}_{12}$. Observe que os limites temporais do evento β não se alteram, visto que esse é exclusivo a $\mathbf{G}_2$. Em contrapartida, o evento α é compartilhado, e sua ocorrência depende das condições dos dois sistemas. É necessária a passagem de no mínimo uma unidade de tempo para que ele esteja elegível em $\mathbf{G}_1$ e duas em $\mathbf{G}_2$, portanto sua eligibilidade somente será verificada em *ambos* os sistemas após duas unidades, e seu limite inferior em $\mathbf{G}_{12}$ será $\ell_{12_\alpha} = 2$. Além disso, α se torna iminente em $\mathbf{G}_1$ após duas ocorrências de *tick* e em $\mathbf{G}_2$ após três; como a sua iminência em $\mathbf{G}_1$ impede que se espere por mais um *tick* antes de sua ocorrência, a regra determina que esse evento seja considerado iminente também em $\mathbf{G}_{12}$ após duas unidades de tempo, ou seja, $u_{12_\alpha} = 2$. Dado que $\ell_{12_\alpha} = u_{12_\alpha}$, a composição é bem definida.

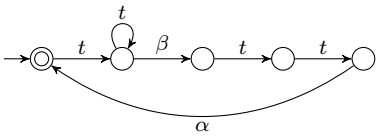

Figura 6 – GTT para o GTA resultante da operação *comp* na Figura 5.

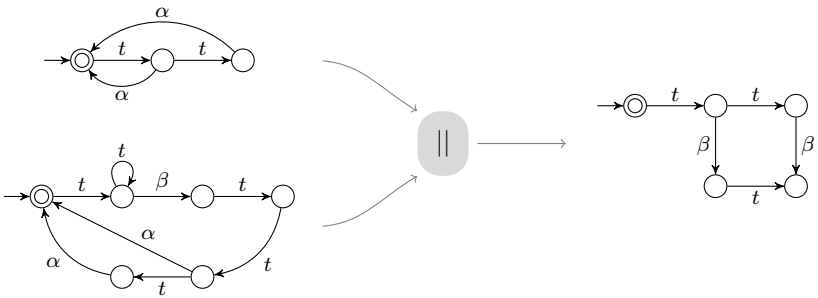

Figura 7 – Composição síncrona dos mesmos SEDT da Figura 5.

É importante ressaltar que, de modo geral, o resultado da composição de dois SEDT através de *comp* difere daquele obtido pela composição síncrona dos seus GTT. Para exemplificar, na Figura 6 pode ser visto o GTT corresponde ao GTA obtido pela operação *comp* da Figura 5, enquanto na Figura 7 é mostrada a composição síncrona dos sistemas da Figura 5. Vê-se claramente que os GTT resultantes das duas operações são distintos; ademais, observe que o resultado da composição síncrona sequer é consistente com a definição de um SEDT, já que o relógio é parado após a ocorrência de uma das cadeias $tt\beta$ ou $t\beta t$. Contudo, a seguinte proposição estabelece uma condição sob a qual a equivalência das duas operações é garantida. Esse resultado será de particular utilidade para a estratégia de controle proposta no Capítulo 4. Sua demonstração é colocada como o Exercício 9.6.1 em (WONHAM, 2013), e uma possível solução será aqui apresentada.

Proposição 1. *Sejam* $\mathbf{G}_1, \mathbf{G}_2$ *SEDT. Se* $\Sigma_1 \cap \Sigma_2 = \{tick\}$*, então* $comp(\mathbf{G}_1, \mathbf{G}_2) \approx \mathbf{G}_1 \,||\, \mathbf{G}_2$*, em que* $\approx$ *denota que as linguagens geradas e marcadas coincidem.*

Demonstração. Sejam $\mathbf{G}_1$ e $\mathbf{G}_2$ SEDT como acima, e seja $\Sigma = \Sigma_1 \cup \Sigma_2$. Denotaremos

$$\mathbf{G}_{\mathrm{c}} = comp(\mathbf{G}_1, \mathbf{G}_2) = \big(Q_{\mathrm{c}}, \Sigma, \delta_{\mathrm{c}}, q_{\mathrm{c}_0}, Q_{\mathrm{c}_m}\big)\,;$$
$$\mathbf{G}_{\mathrm{s}} = \mathbf{G}_1 \,||\, \mathbf{G}_2 = \big(Q_{\mathrm{s}}, \Sigma, \delta_{\mathrm{s}}, q_{\mathrm{s}_0}, Q_{\mathrm{s}_m}\big)\,.$$

Defina a projeção $P_{\mathrm{at}} : \Sigma^* \to \Sigma_{\mathrm{at}}^*$ e, para $i \in \{1,2\}$, $P_i : \Sigma^* \to \Sigma_i^*$ e $P_{i_{\mathrm{at}}} : \Sigma^* \to \Sigma_{i_{\mathrm{at}}}^*$. Defina ainda $P_t : \Sigma^* \to \{tick\}^*$ e, para cada $\sigma \in \Sigma_{\mathrm{at}}$, $P_\sigma : \Sigma^* \to \{\sigma\}^*$. Primeiramente, mostraremos que para qualquer cadeia $s \in \Sigma^*$,

$$s \in L(\mathbf{G}_{\mathrm{s}}) \;\Leftrightarrow\; s \in L(\mathbf{G}_{\mathrm{c}})\,. \qquad (\dagger)$$

A prova será feita por indução sobre o comprimento de s. Seja, portanto, $s \in \Sigma^*$ tal que $|\,s\,| = 0$, ou seja, $s = \varepsilon$. Temos

$$\begin{aligned}
\varepsilon \in L(\mathbf{G}_{\mathrm{s}}) &\Leftrightarrow Q_{\mathrm{s}} \neq \emptyset \\
&\Leftrightarrow Q_i \neq \emptyset,\ \forall\, i \in \{1,2\} \\
&\Leftrightarrow A_i \neq \emptyset,\ \forall\, i \in \{1,2\} \\
&\Leftrightarrow A \neq \emptyset \\
&\Leftrightarrow Q_{\mathrm{c}} \neq \emptyset \\
&\Leftrightarrow \varepsilon \in L(\mathbf{G}_{\mathrm{c}})\,.
\end{aligned}$$

Como hipótese indutiva, dado $k > 0$, suponha que (†) é válido para qualquer $s \in \Sigma^*$ tal que $|\,s\,| = k$.

Seja, agora, uma cadeia $r \in \Sigma^*$, $|\,r\,| = k+1$. Podemos escrever qualquer cadeia assim definida na forma $r = v\alpha$, com $v \in \Sigma^*$, $|\,v\,| = k$ e $\alpha \in \Sigma$. Pela hipótese, ou $v \notin L(\mathbf{G}_{\mathrm{s}}) \cup L(\mathbf{G}_{\mathrm{c}})$, ou $v \in L(\mathbf{G}_{\mathrm{s}}) \cap L(\mathbf{G}_{\mathrm{c}})$. No primeiro caso, temos $r \notin L(\mathbf{G}_{\mathrm{s}}) \cup L(\mathbf{G}_{\mathrm{c}})$, portanto (†) é verificado. Analisaremos agora o segundo caso.

Supondo que $v \in L(\mathbf{G}_{\mathrm{s}}) \cap L(\mathbf{G}_{\mathrm{c}})$, sejam $q_{\mathrm{s}} = \delta_{\mathrm{s}}(q_{\mathrm{s}_0}, v) \in Q_{\mathrm{s}}$ e $q_{\mathrm{c}} = \delta_{\mathrm{c}}(q_{\mathrm{c}_0}, v) \in Q_{\mathrm{c}}$. Temos que $q_{\mathrm{c}} = \big(a, \{\tau_\sigma \,|\, \sigma \in \Sigma_{\mathrm{at}}\}\big)$ e

$$q_{\mathrm{s}} = (q_1, q_2) = \big((a_1, a_2)\,,\, \{\tau_{1_\sigma} \,|\, \sigma \in \Sigma_{1_{\mathrm{at}}}\} \cup \{\tau_{2_\sigma} \,|\, \sigma \in \Sigma_{2_{\mathrm{at}}}\}\big)\,,$$

com $a = \delta_{\mathrm{at}}\big(a_0, P_{\mathrm{at}}(v)\big)$ e $a_i = \delta_{i_{\mathrm{at}}}\big(a_{i_0}, P_{i_{\mathrm{at}}}(v)\big)$. Para um dado evento $\sigma \in \Sigma_{\mathrm{at}}$, seja $i \in \{1,2\}$ tal que $\sigma \in \Sigma_{i_{\mathrm{at}}}$. Provaremos agora que

$$\tau_\sigma = \tau_{i_\sigma}\,. \qquad (2.3)$$

Faremos a prova apenas para o caso $\sigma \in \Sigma_{i_{\mathrm{esp}}}$, sendo o caso $\sigma \in \Sigma_{i_{\mathrm{rem}}}$

completamente análogo. Perceba que isso equivale a mostrar que, para qualquer $n \in T_\sigma = T_{i_\sigma}$, $\tau_\sigma = n \Leftrightarrow \tau_{i_\sigma} = n$. Temos, então,

$$\begin{aligned}
\tau_\sigma = n \;\Leftrightarrow\; & \exists w, y \in \Sigma^* \;\big|\; v = wy \;\&\; P_\sigma(y) = \varepsilon \\
& \qquad \&\; \forall z \in \overline{\{y\}},\; P_{\mathrm{at}}(wz)\sigma \in L(\mathbf{G}_{\mathrm{at}}) \\
& \qquad \&\; P_t(y) = u_\sigma - n \\
\overset{(*)}{\Leftrightarrow}\; & \exists w_i, y_i \in \Sigma_i^* \;\big|\; P_i(v) = w_i y_i \;\&\; P_\sigma(y_i) = \varepsilon \\
& \qquad \&\; \forall z_i \in \overline{\{y_i\}},\; P_{i_{\mathrm{at}}}(w_i z_i)\sigma \in L(\mathbf{G}_{i_{\mathrm{at}}}) \\
& \qquad \&\; P_t(y_i) = u_{i_\sigma} - n \\
\Leftrightarrow\; & \tau_{i_\sigma} = n\,,
\end{aligned}$$

onde em $(*)$, para $(\Rightarrow)$ tome, por exemplo, $w_i = P_i(w)$ e $y_i = P_i(y)$, e para $(\Leftarrow)$ tome $w \in P_i^{-1}(w_i)$ e $y \in P_i^{-1}(y_i)$ tais que $v = wy$. Com isso, prova-se (2.3). Agora, consideraremos cada caso possível para α. Os passos marcados com $(\star)$ são justificados por (2.3).

i. $\alpha = tick$:

$$\begin{aligned}
\delta_{\mathrm{s}}(q_{\mathrm{s}}, \alpha)! \;\Leftrightarrow\; & \delta_i(q_i, \alpha)!,\; \forall\, i \in \{1,2\} \\
\Leftrightarrow\; & \big[\forall \rho \in \Sigma_{i_{\mathrm{esp}}},\; \delta_{i_{\mathrm{at}}}(a_i, \rho)! \Rightarrow \tau_{i_\rho} > 0\,\big],\; \forall\, i \in \{1,2\} \\
\overset{(\star)}{\Leftrightarrow}\; & \forall \rho \in \Sigma_{\mathrm{esp}},\; \delta_{\mathrm{at}}(a, \rho)! \Rightarrow \tau_\rho > 0 \\
\Leftrightarrow\; & \delta_{\mathrm{c}}(q_{\mathrm{c}}, \alpha)!\,.
\end{aligned}$$

ii. $\alpha \in \Sigma_{i_{\mathrm{esp}}}$ para um dado $i \in \{1,2\}$:

$$\begin{aligned}
\delta_{\mathrm{s}}(q_{\mathrm{s}}, \alpha)! \;&\Leftrightarrow \delta_i(q_i, \alpha)! \\
&\Leftrightarrow \delta_{i_{\mathrm{at}}}(a_i, \alpha)! \;\&\; 0 \leq \tau_{i_\alpha} \leq u_{i_\alpha} - \ell_{i_\alpha} \\
&\overset{(\star)}{\Leftrightarrow} \delta_{\mathrm{at}}(a, \alpha)! \;\&\; 0 \leq \tau_\alpha) \leq u_\alpha - \ell_\alpha \\
&\Leftrightarrow \delta_{\mathrm{c}}(q_{\mathrm{c}}, \alpha)!\,.
\end{aligned}$$

iii. $\alpha \in \Sigma_{i_{\mathrm{rem}}}$ para um dado $i \in \{1,2\}$:

$$\begin{aligned}
\delta_{\mathrm{s}}(q_{\mathrm{s}}, \alpha)! \;&\Leftrightarrow \delta_i(q_i, \alpha)! \\
&\Leftrightarrow \delta_{i_{\mathrm{at}}}(a_i, \alpha)! \;\&\; \tau_{i_\alpha} = 0 \\
&\overset{(\star)}{\Leftrightarrow} \delta_{\mathrm{at}}(a, \alpha)! \;\&\; \tau_\alpha = 0 \\
&\Leftrightarrow \delta_{\mathrm{c}}(q_{\mathrm{c}}, \alpha)!\,.
\end{aligned}$$

Isso conclui a indução e prova (†). Em seguida, mostraremos que

$$s \in L_m(\mathbf{G}_\mathrm{s}) \;\Leftrightarrow\; s \in L_m(\mathbf{G}_\mathrm{c}) \qquad (\ddagger)$$

para qualquer $s \in \Sigma^*$. Por (†), temos que ou $s \notin L(\mathbf{G}_\mathrm{s}) \cup L(\mathbf{G}_\mathrm{c})$, ou $s \in L(\mathbf{G}_\mathrm{s}) \cap L(\mathbf{G}_\mathrm{c})$. No primeiro caso, evidentemente também $s \notin L_m(\mathbf{G}_\mathrm{s}) \cup L_m(\mathbf{G}_\mathrm{c})$, portanto (‡) vale. Caso $s \in L(\mathbf{G}_\mathrm{s}) \cap L(\mathbf{G}_\mathrm{c})$, temos

$$\begin{aligned}
s \in L_m(\mathbf{G}_\mathrm{s}) &\Leftrightarrow \delta_\mathrm{s}(q_{\mathrm{s}_0}, s) \in Q_{\mathrm{s}_m} \\
&\Leftrightarrow \delta_i\big(q_{i_0}, P_i(s)\big) \in Q_{i_m},\ \forall\, i \in \{1,2\} \\
&\Leftrightarrow \big(a_1, \{\tau_{1_\sigma} \,|\, \sigma \in \Sigma_{1_\mathrm{at}}\}\big) \in Q_{1_m}\ \ \& \\
&\quad \big(a_2, \{\tau_{2_\sigma} \,|\, \sigma \in \Sigma_{2_\mathrm{at}}\}\big) \in Q_{2_m}\,, \\
&\quad \text{onde}\ \ a_i = \delta_{i_\mathrm{at}}\big(a_{i_0}, P_{i_\mathrm{at}}(s)\big) \\
&\Leftrightarrow \big((a_1,a_2), \{\tau_{1_\sigma} \,|\, \sigma \in \Sigma_{1_\mathrm{at}}\} \cup \{\tau_{2_\sigma} \,|\, \sigma \in \Sigma_{2_\mathrm{at}}\}\big) \\
&\quad = \delta_\mathrm{c}(q_{\mathrm{c}_0}, s) \in Q_{\mathrm{c}m} \\
&\Leftrightarrow s \in L_m(\mathbf{G}_\mathrm{c})\,,
\end{aligned}$$

provando (‡) e concluindo a demonstração.

□

2.4 CONTROLE SUPERVISÓRIO DE SEDT

Há duas possíveis ações de controle que um supervisor é capaz de realizar sobre um SED temporizado. Primeiramente, ele pode *desabilitar* eventos no sistema, sendo os eventos passíveis de desabilitação aqueles que podem ser indefinidamente proibidos de ocorrer. Naturalmente, esse não é o caso para todos os eventos. Em particular, se um evento esperado pudesse ser desabilitado, isso significaria que ele poderia ser proibido de ocorrer mesmo em um estado no qual estivesse iminente e fosse o único evento elegível. Tal situação levaria o sistema a um bloqueio temporal, o que contradiz as definições da Seção 2.2. Isso fica claro se imaginarmos, por exemplo, que o evento α possa ser desabilitado no sistema da Figura 2 após a cadeia $\beta t \alpha t$. Para evitar semelhantes incoerências, define-se o conjunto de eventos *proibíveis* como $\Sigma_\mathrm{pro} \subseteq \Sigma_\mathrm{rem}$.

Em segundo lugar, existe a possibilidade de *preemptar* o *tick* do relógio, e o conjunto dos eventos capazes de fazê-lo, chamados de eventos *forçáveis*, é dado por $\Sigma_\mathrm{for} \subseteq \Sigma_\mathrm{at}$. Apesar de o evento *tick* não poder ser propriamente desabilitado (já que o controlador não é capaz

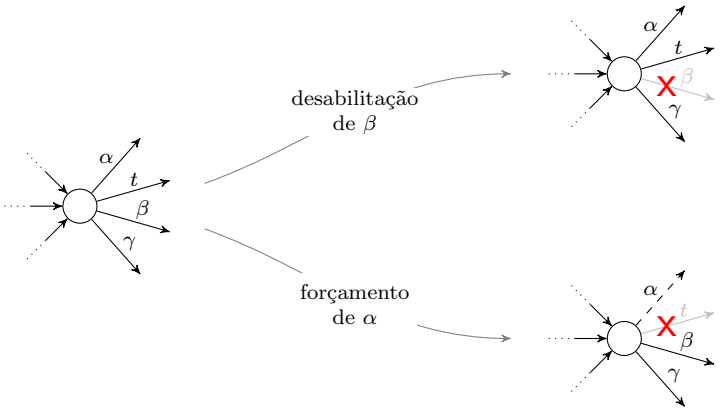

Figura 8 – Efeito das ações de controle sobre os eventos de um SEDT.

de parar o relógio), caso um ou mais eventos forçáveis estejam elegíveis ele pode ser efetivamente preemptado, assegurando-se que o próximo a ocorrer seja algum dos eventos de Σ_{at} (mas não necessariamente de Σ_{for}). Por outras palavras, se nenhum dos demais eventos de Σ_{at} ocorrer, em última instância pode-se contar com um evento forçável para "ser mais rápido que o relógio", i. e., para ocorrer antes de *tick*.

Definem-se, então, o conjunto de eventos *controláveis*

$$\Sigma_c = \Sigma_{\text{pro}} \cup \{tick\}$$

e o conjunto de eventos *não controláveis*

$$\Sigma_{nc} = \Sigma_{\text{at}} - \Sigma_{\text{pro}}\,.$$

É importante enfatizar que, apesar de aqui designado como controlável, *tick* somente pode ser preemptado na presença de um evento forçável, ao contrário dos eventos proibíveis, que podem ser desabilitados em qualquer estado. Ressalva-se, ainda, que Σ_{for} não carrega nenhuma relação particular com Σ_{pro}, de modo que um evento forçável pode ser, ou não, controlável.

O efeito das ações de um supervisor sobre os eventos de um SEDT é ilustrado na Figura 8. Supõe-se que α seja o único evento forçável, que β seja controlável e γ não controlável. Uma seta tracejada é usada para denotar que o evento correspondente está sendo forçado. Perceba, em particular, que o forçamento de α não garante sua ocorrência, apenas a preempção de *tick*. O não determinismo — inerente a um SED —

quanto à ocorrência dos eventos que permanecem elegíveis é, portanto, mantido. Obviamente, existe a possibilidade de que as duas ações sejam tomadas em um mesmo estado, o que, neste exemplo, resultaria em ter-se apenas α e γ como elegíveis. Se α for proibível, também é possível desabilitá-lo juntamente com β, perdendo-se, no entanto, a capacidade de preemptar *tick*, haja vista que deixaria de haver eventos forçáveis elegíveis. Finalmente, note que a ocorrência de γ não pode ser impedida por nenhuma das ações de controle.

Dado um SEDT $\mathbf{G}$, considere uma função $\mathcal{S} : L(\mathbf{G}) \to 2^{\Sigma}$ e defina a linguagem $L(\mathcal{S}/\mathbf{G})$ por

- $\varepsilon \in L(\mathcal{S}/\mathbf{G})$;
- $s\sigma \in L(\mathcal{S}/\mathbf{G}) \;\Leftrightarrow\; s \in L(\mathcal{S}/\mathbf{G}) \;\&\; \sigma \in \mathcal{S}(s) \;\&\; s\sigma \in L(\mathbf{G})$.

Diz-se que $\mathcal{S}$ é um *supervisor* para $\mathbf{G}$ se, $\forall s \in L(\mathcal{S}/\mathbf{G})$, forem satisfeitas as seguintes condições:

(s_1) $\Sigma_{nc} \subseteq \mathcal{S}(s)$;

(s_2) $\mathcal{S}(s) \cap \Sigma_{\mathbf{G}}(s) \cap \Sigma_{\mathrm{for}} = \emptyset \;\&\; tick \in \Sigma_{\mathbf{G}}(s) \;\Rightarrow\; tick \in \mathcal{S}(s)$.

Para $s \in L(\mathcal{S}/\mathbf{G})$, $\mathcal{S}(s) \in 2^{\Sigma}$ representa o subconjunto de eventos permitidos a ocorrer em $\mathbf{G}$ pelo supervisor após s, i. e., no estado $q = \delta(q_0, s) \in Q$. Equivalentemente, $\Sigma_{\mathbf{G}}(s) - \mathcal{S}(s)$ é o conjunto de eventos desabilitados em q pela ação de controle. Assim, (s_1) dita que jamais um evento não controlável seja desabilitado. Por sua vez, (s_2) formaliza o modo pelo qual os eventos forçáveis podem ser utilizados pelo supervisor, permitindo a preempção de *tick* apenas quando $\mathcal{S}(s) \cap \Sigma_{\mathbf{G}}(s) \cap \Sigma_{\mathrm{for}} \neq \emptyset$.

Uma linguagem $M \subseteq L_m(\mathbf{G})$ pode ser associada a um supervisor $\mathcal{S}$, e o par $(\mathcal{S}, M)$ é chamado de *supervisor marcador* para $\mathbf{G}$. O papel de M é determinar quais das cadeias marcadas de $\mathbf{G}$ permanecerão marcadas sob supervisão de $\mathcal{S}$. Deste ponto em diante, sempre que nos referirmos a um supervisor $\mathcal{S}$ entenda-se um supervisor marcador $(\mathcal{S}, M)$, sendo que M ficará implícita na notação.

Utiliza-se $\mathcal{S}/\mathbf{G}$ para denotar $\mathbf{G}$ sob supervisão de $\mathcal{S}$, cuja linguagem gerada é $L(\mathcal{S}/\mathbf{G}) \subseteq L(\mathbf{G})$ como acima e marcada é

$$L_m(\mathcal{S}/\mathbf{G}) = L(\mathcal{S}/\mathbf{G}) \cap M\,.$$

Um supervisor $\mathcal{S}$ é *não bloqueante* para $\mathbf{G}$ caso $\mathcal{S}/\mathbf{G}$ seja não bloqueante, i. e., se $L(\mathcal{S}/\mathbf{G}) = \overline{L_m(\mathcal{S}/\mathbf{G})}$.

Uma linguagem $K \subseteq L(\mathbf{G})$ é *controlável* com respeito a $\mathbf{G}$ se, $\forall s \in \overline{K}$, as seguintes condições forem válidas:

(c_1) $\sigma \in \Sigma_{nc} \cap \Sigma_{\mathbf{G}}(s) \;\Rightarrow\; \sigma \in \Sigma_K(s)$;

(c_2) $tick \in \Sigma_{\mathbf{G}}(s) \;\;\&\;\; \Sigma_K(s) \cap \Sigma_{\text{for}} = \emptyset \;\Rightarrow\; tick \in \Sigma_K(s)$.

A condição (c_1) é idêntica à definição de controlabilidade para o caso não temporizado, a qual exige que qualquer evento não controlável que esteja elegível em $\mathbf{G}$ também o esteja em K. Já (c_2) introduz a ideia de que, se *tick* puder ocorrer em $\mathbf{G}$ e não houver eventos forçáveis elegíveis em K, então *tick* também deverá poder ocorrer em K. Para fins práticos, K normalmente terá o papel de um comportamento desejado para o sistema, determinado por uma dada especificação de controle. Nesse caso, as condições acima servirão para garantir que esse comportamento possa ser atingido mediante a ação de um supervisor, dentro das limitações impostas pelas condições (s_1) e (s_2).

De fato, em (BRANDIN; WONHAM, 1994) mostrou-se que, para uma linguagem K que obedeça a $\emptyset \neq K \subseteq L_m(\mathbf{G})$, existe um supervisor não bloqueante $\mathcal{S}$ tal que $L_m(\mathcal{S}/\mathbf{G}) = K$ se e somente se K for controlável com respeito a $\mathbf{G}$. Caso a controlabilidade não seja verificada, foi também provado que o conjunto de sublinguagens controláveis de K, escrito $\mathcal{C}(K)$, é sempre não vazio e possui um único elemento supremo $sup\mathcal{C}(K)$, o qual representa o comportamento minimamente restritivo possível de ser imposto por supervisão sobre $\mathbf{G}$ de modo a respeitar K. É evidente que, se K for controlável, então $sup\mathcal{C}(K) = K$.

Observação 4. Pode-se mostrar de maneira simples que uma linguagem controlável $K \subseteq L(\mathbf{G})$ nunca para o relógio de $\mathbf{G}$, no sentido de que ela nunca desabilita *tick* indefinidamente. De fato, $\forall s \in \overline{K}$ temos uma das seguintes situações:

i. $tick \in \Sigma_{\mathbf{G}}(s) - \Sigma_K(s)$;

ii. $tick \in \Sigma_{\mathbf{G}}(s) \cap \Sigma_K(s)$;

iii. $tick \notin \Sigma_{\mathbf{G}}(s)$.

Pela controlabilidade de K, o caso (i) implica que $\Sigma_K(s) \cap \Sigma_{\text{for}} \neq \emptyset$, o que leva a $\Sigma_{\mathbf{G}}(s) \cap \Sigma_{\text{for}} \neq \emptyset$ pois $K \subseteq L(\mathbf{G})$. Além disso, pela Observação 3, o caso (iii) implica que $\exists v \in \Sigma^+_{\text{esp}} \,|\, sv \in L(\mathbf{G})$. Em particular, note que tanto para (i) como para (iii) temos $\Sigma_{\mathbf{G}}(s) \cap \Sigma_{\text{at}} \neq \emptyset$. A Observação 2 então garante que uma sucessão de (i) ou (iii) não pode ser repetida indefinidamente para cadeias que venham na sequência de s, portanto (ii) deve eventualmente ocorrer. Formalmente, conclui-se que

$$\forall s \in \overline{K},\; \exists u \in \Sigma^* \,|\, tick \in \Sigma_K(su)\,.$$

Observação 5. Embora conveniente do ponto de vista teórico, a definição abstrata de um supervisor como uma função $\mathcal{S}$ não é favorável à implementação prática. Não seria razoável listar $\mathcal{S}(s)$ para cada cadeia $s \in L(\mathcal{S}/\mathbf{G})$; é necessário obter uma *representação* adequada. Já que os modelos dos sistemas aqui tratados são dados por autômatos, uma escolha natural é utilizar autômatos também para a representação dos supervisores. Para um SEDT $\mathbf{G}$, diz-se que um autômato $\mathbf{S}$ *implementa* o supervisor $\mathcal{S}$ se $L(\mathcal{S}/\mathbf{G}) = L(\mathbf{S} \,||\, \mathbf{G})$ e $L_m(\mathcal{S}/\mathbf{G}) = L_m(\mathbf{S} \,||\, \mathbf{G})$. Uma discussão mais detalhada sobre implementação de supervisores por autômatos encontra-se em (WONHAM, 2013) ou, sob o nome de *realização*, em (CASSANDRAS; LAFORTUNE, 2010).

3 O CONCEITO DE SUPERVISÃO MODULAR LOCAL

O objetivo deste capítulo é esclarecer e motivar o conceito de *supervisão modular local*, conforme proposto por Queiroz e Cury em (QUEIROZ; CURY, 2000, 2002). Esses trabalhos estão voltados para SED não temporizados; embora a consideração do tempo venha a trazer significativas diferenças aos resultados teóricos do Capítulo 4, por ora manteremos a discussão em um nível mais alto de abstração, no qual os conceitos fundamentais são basicamente os mesmos para sistemas temporizados e não temporizados. Para fins de comparação e contextualização, antes de ser introduzido o controle modular local serão brevemente revisadas três abordagens de controle supervisório desenvolvidas originalmente no contexto não temporizado e posteriormente estendidas para SEDT, quais sejam: monolítica, modular e descentralizada.

Como um comentário introdutório, gostaríamos de salientar que muitos SED complexos podem ser naturalmente modelados como a composição de subsistemas menores e assíncronos entre si, os quais trabalham em paralelo de forma independente. Tal representação, chamada de *sistema produto*, foi estudada em (RAMADGE; WONHAM, 1989; RAMADGE, 1989). Além disso, é comum que, das múltiplas especificações normalmente impostas sobre um tal sistema, cada uma restrinja o comportamento de apenas uma parte da planta global e vise sincronizar o funcionamento de somente alguns de seus subsistemas. Com base nesses fatos, ao longo da discussão consideraremos um SED arbitrário $\mathbf{G}$ (temporizado ou não) modelado como um sistema produto com três componentes, $\mathbf{G}_1$, $\mathbf{G}_2$ e $\mathbf{G}_3$. Para esse sistema, suponha que sejam colocadas duas especificações de controle, $E_a \subseteq \Sigma_a^*$ e $E_b \subseteq \Sigma_b^*$, das quais a primeira afeta os subsistemas $\mathbf{G}_2$ e $\mathbf{G}_3$ e a segunda afeta $\mathbf{G}_1$ e $\mathbf{G}_2$, ou seja, $\Sigma_a \subseteq \Sigma_2 \cup \Sigma_3$ e $\Sigma_b \subseteq \Sigma_1 \cup \Sigma_2$.

3.1 CONTROLE SUPERVISÓRIO MONOLÍTICO

Na abordagem monolítica, desenvolvida em (RAMADGE; WONHAM, 1987) para SED e em (BRANDIN; WONHAM, 1994) para SEDT, os subsistemas são todos compostos para formar o modelo da planta $\mathbf{G}$, o qual é então composto com as especificações para obter uma linguagem que representa o comportamento desejado para o sistema como um todo. Um único supervisor $\mathcal{S}$ é calculado para impor esse com-

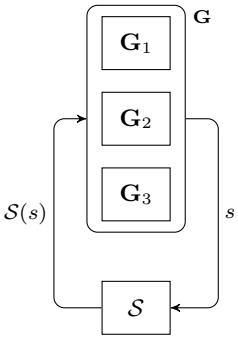

Figura 9 – Arquitetura do controle supervisório monolítico.

portamento sobre o sistema de maneira minimamente restritiva. A arquitetura de controle sobre a qual se baseia essa abordagem pode ser vista na Figura 9. O supervisor observa uma cadeia de eventos s executada pela planta e retorna um conjunto $\mathcal{S}(s)$ de eventos habilitados, determinado de acordo com as especificações E_a e E_b.

Embora o cálculo do supervisor monolítico tenha complexidade polinomial no número de estados do modelo global, esse número pode crescer exponencialmente com a quantidade de componentes agregados. Em um caso geral, com um número arbitrário de subsistemas e especificações, isso pode impor severas limitações para a aplicação desta metodologia.

3.2 CONTROLE SUPERVISÓRIO MODULAR

Uma alternativa para reduzir a complexidade do procedimento de síntese de supervisores é a abordagem modular, proposta no contexto não temporizado em (WONHAM; RAMADGE, 1988) e estendida para o temporizado em (BRANDIN; WONHAM, 1993). A ideia principal é dividir a tarefa de controle em subtarefas, calculando um supervisor separado para cada especificação. No exemplo aqui considerado, isso significa que teríamos dois supervisores, $\mathcal{S}_A$ e $\mathcal{S}_B$, impondo separadamente as especificações E_a e E_b. Condições foram apresentadas para que a conjunção desses supervisores, denotada pelo símbolo $\wedge$, resulte em um comportamento global não bloqueante que respeite ambas as especificações de forma minimamente restritiva. A arquitetura de controle é mostrada na Figura 10.

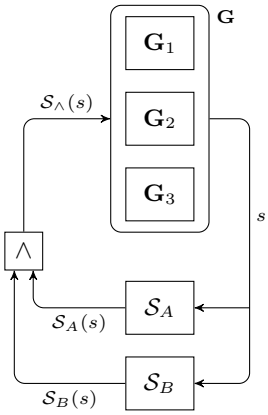

Figura 10 – Arquitetura do controle supervisório modular.

Esta abordagem pode apresentar consideráveis vantagens com relação à monolítica no que diz respeito a custo computacional — para uma discussão detalhada, o leitor pode referir-se a (WONHAM; RAMADGE, 1988). Ainda assim, o fato de que o cálculo de cada supervisor modular leva em conta o comportamento de toda a planta pode resultar em um custo excessivamente alto nos casos em que o modelo do sistema apresenta um número elevado de estados.

3.3 CONTROLE SUPERVISÓRIO DESCENTRALIZADO

Em (LIN; WONHAM, 1988) foi proposta uma abordagem de controle descentralizado para SED, a qual foi estendida para sistemas temporizados em (BRANDIN; WONHAM, 1993). Essa abordagem foca no fato de que, em grande parte dos casos, cada especificação de contole restringe de maneira direta a ocorrência de apenas parte dos eventos da planta. Em outras palavras, as especificações não são definidas sobre o alfabeto completo do sistema, mas sobre subalfabetos locais. Isso permite, além de calcular um supervisor separado para cada especificação, que esse cálculo seja realizado com base em uma versão parcial da planta, obtida através da projeção do comportamento global sobre o respectivo alfabeto local. Para exemplificar, no corrente exemplo os supervisores $\mathcal{S}_A$ e $\mathcal{S}_B$ seriam calculados com base nos modelos $P_a\big(L(\mathbf{G})\big)$ e $P_b\big(L(\mathbf{G})\big)$, respectivamente. A ideia é ilustrada pela arquitetura mostrada na Figura 11.

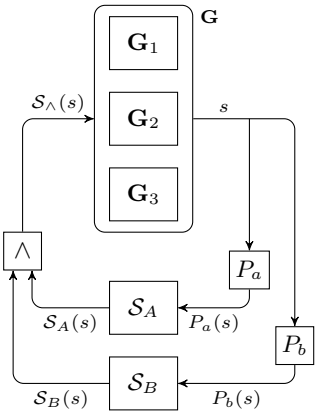

Figura 11 – Arquitetura do controle supervisório descentralizado.

Uma das motivações desta abordagem é tratar situações nas quais cada supervisor não tem acesso a informações sobre a ocorrência de todos os eventos da planta, devido, por exemplo, à distribuição física dos subsistemas. Outra motivação é aumentar a eficiência do processo de síntese, o qual, idealmente, passaria a envolver modelos mais simples e com espaços de estados reduzidos. Entretanto, a utilização de projeções naturais para a obtenção desses modelos locais pode ter um custo bastante elevado, e o tamanho dos mesmos pode na verdade *crescer* exponencialmente com o número de estados de $\mathbf{G}$, comumente tornando-se maior que o do próprio modelo global (RUDIE, 1988; LIN; WONHAM, 1990; RUDIE; WILLEMS, 1995). Em (WONG, 1998) foi mostrado que, se P_a e P_b tiverem a propriedade de serem *observadores* para $L_m(\mathbf{G})$, então os modelos obtidos por projeção serão garantidamente não mais complexos que o sistema original, e o cálculo dos supervisores com observação parcial tende a ter um custo menor quando comparado às abordagens anteriores. No mesmo trabalho, é apresentado um algoritmo capaz de computar as projeções naturais com complexidade polinomial (no número de estados de $\mathbf{G}$). Não obstante, em geral o esforço total requerido para a aplicação da abordagem descentralizada ainda é superior ao da abordagem monolítica.

No caso particular em que as especificações são definidas precisamente sobre os alfabetos dos subsistemas, por exemplo se $\Sigma_a = \Sigma_2 \cup \Sigma_3$ e $\Sigma_b = \Sigma_1 \cup \Sigma_2$, o uso de projeções naturais torna-se desnecessário, já que $P_a\big(L(\mathbf{G})\big)$ é, então, igual à composição de $L(\mathbf{G}_2)$ com $L(\mathbf{G}_3)$, e analogamente para $P_b\big(L(\mathbf{G})\big)$. Isso pode fazer com que esta aborda-

gem se torne vantajosa em termos computacionais. Ainda assim, os resultados nela estabelecidos são limitados ao caso de especificações prefixo-fechadas; além disso, as condições para garantir a ausência de bloqueio, estabelecidas em (LIN; WONHAM, 1991), aplicam-se apenas aos casos em que os alfabetos dos supervisores locais são disjuntos dois a dois, ou seja, em que dois supervisores jamais exercem controle sobre partes coincidentes da planta.

3.4 CONTROLE SUPERVISÓRIO MODULAR LOCAL

Um fator comum às três estratégias de controle acima apresentadas é a utilização do modelo global da planta (diretamente para o cálculo dos supervisores nas duas primeiras e para a obtenção dos modelos locais na última). O fato de que, para sistemas de grande porte, a mera construção desse modelo pode ser uma tarefa árdua, ou mesmo infactível, aponta para a necessidade de um método mais eficiente.

A estratégia de controle modular local proposta em (QUEIROZ; CURY, 2000) para SED não temporizados proporcionou um refinamento da abordagem modular, explorando tanto a modularidade das especificações quanto a do próprio sistema. São considerados para a síntese de cada supervisor apenas os subsistemas diretamente afetados pela especificação correspondente. No exemplo desta discussão, duas plantas locais, $\mathbf{G}_A$ e $\mathbf{G}_B$, são assim construídas, a primeira pela composição dos subsistemas afetados por E_a e a segunda daqueles afetados por E_b. Supervisores modulares $\mathcal{S}_A$ e $\mathcal{S}_B$ são, então, calculados *localmente*, considerando-se que cada um deles observa e controla apenas os eventos da respectiva planta local. A arquitetura por trás desta abordagem aparece na Figura 12.

Assim como no caso descentralizado, o objetivo é simplificar o procedimento de síntese pelo fato de utilizarem-se modelos reduzidos. Porém, enquanto a descentalizada é uma abordagem *top-down* (de cima para baixo), na qual os modelos locais são obtidos por projeções do modelo global, aqui as plantas locais são construídas de maneira *bottom-up* (de baixo para cima), pela composição apenas dos subsistemas que devem ser controlados pelo supervisor correspondente. O uso do modelo global $\mathbf{G}$ durante o processo de síntese é com isso evitado, o que representa uma expressiva vantagem quando o sistema envolvido for de grande porte. Condições, similares àquelas do caso modular, foram estabelecidas para garantir que os supervisores locais, quando operando em paralelo, levem o sistema a ter um comportamento global não blo-

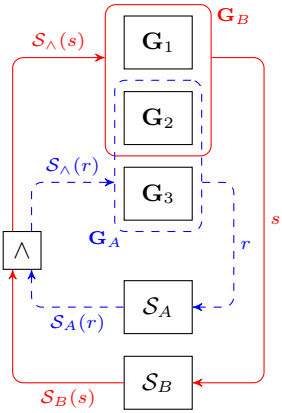

Figura 12 – Arquitetura do controle supervisório modular local.

queante e ótimo. O teste de tais condições deve, no entanto, ser realizado no nível global, envolvendo todos os modelos locais sob supervisão. Consequentemente, apesar desta abordagem ser mais eficiente que as anteriores, o número de componentes do sistema permanece como a principal limitação para a sua aplicação. Nesta dissertação, objetivamos trazer os benefícios do controle modular local para o contexto de SED temporizados.

4 CONTROLE SUPERVISÓRIO MODULAR LOCAL DE SEDT

As contribuições desta dissertação, que consistem na extensão do controle modular local para o contexto de sistemas a eventos discretos temporizados, são apresentadas neste capítulo. Por questão de simplicidade, os resultados são expostos considerando apenas duas especificações. Esses podem, todavia, ser facilmente adaptados para o caso mais geral.

4.1 FORMULAÇÃO DO PROBLEMA

Conforme mencionado no Capítulo 3, muitos SED podem ser modelados como a composição de subsistemas assíncronos, que operam de maneira independente. No caso não temporizado, isso equivale a dizer que os alfabetos de eventos dos subsistemas são disjuntos dois a dois. Já no contexto temporizado o mesmo não é possível, pois assumimos que os relógios de todos os subsistemas são sincronizados, o que significa que todos compartilham o mesmo evento *tick*. É possível, no entanto, considerar que as atividades realizadas pelos subsistemas sejam assíncronas, ou seja, que seus alfabetos de eventos regulares (diferentes de *tick*) sejam disjuntos dois a dois. Portanto, definimos um *sistema produto temporizado* (SPT) como qualquer SEDT dado por

$$\mathbf{G} = comp(\mathbf{G}_1, \ldots, \mathbf{G}_N) \tag{4.1}$$

tal que

$$\Sigma_j \cap \Sigma_k = \{tick\}, \quad \forall\, j, k \in \{1, \ldots, N\},\ j \neq k\,. \tag{4.2}$$

A partir da Proposição 1, a propriedade (4.2) implica que (4.1) pode ser reescrito como

$$\mathbf{G} = \mathop{\Big|\Big|}_{i=1}^{N} \mathbf{G}_i\,.$$

Embora essa expressão represente o sistema completo como a composição de todas as suas partes, lembre que o intuito da abordagem modular local é precisamente *evitar* a construção desse modelo global.

Doravante no presente capítulo, seja $\mathbf{G} = (Q, \Sigma, \delta, q_0, Q_m)$ um SPT composto de subsistemas $\mathbf{G}_i$, $i \in I = \{1, \ldots, N\}$. Considere ainda duas especificações para esse sistema, representadas pelas linguagens $E_a \subseteq \Sigma_a^*$ e $E_b \subseteq \Sigma_b^*$, com $\Sigma_a, \Sigma_b \subseteq \Sigma$. O comportamento global

desejado é, portanto, determinado pela linguagem $E_a \,||\, E_b \,||\, L_m(\mathbf{G})$. Nosso principal objetivo é obter, de uma maneira modular e local, supervisores $\mathcal{S}_A$ e $\mathcal{S}_B$ — um para cada especificação — cuja conjunção $\mathcal{S}_A \wedge \mathcal{S}_B$ garanta que o sistema controlado não viole esse comportamento e seja não bloqueante. O modo como a operação de conjunção é definida consiste, evidentemente, em um aspecto fundamental da estratégia de controle, e será apresentado na Seção 4.4. O *problema de controle supervisório modular local* (PCSML) pode ser formalmente enunciado como:

PCSML: Dadas duas especificações de controle E_a e E_b, encontre supervisores locais $\mathcal{S}_A$ e $\mathcal{S}_B$ tais que $\mathcal{S}_A \wedge \mathcal{S}_B$ é um supervisor não bloqueante para $\mathbf{G}$ *e* $L_m(\mathcal{S}_A \wedge \mathcal{S}_B/\mathbf{G}) \subseteq E_a \,||\, E_b \,||\, L_m(\mathbf{G})$.

Por *supervisor local* entendemos um supervisor que atua localmente sobre $\mathbf{G}$, ou seja, sobre apenas um subconjunto de $\{\mathbf{G}_i \,|\, i \in I\}$. Efetivamente, gostaríamos de garantir que cada supervisor agisse sobre o subconjunto formado apenas pelos subsistemas afetados pela especificação correlata. Dizemos que uma especificação *afeta* o subsistema $\mathbf{G}_i$ se ela influencia de maneira direta a ocorrência de algum dos eventos de $\Sigma_{i_{\mathrm{at}}}$.

Com base nessas ideias, em um primeiro passo definimos os conjuntos

$$I_A = \left\{ i \in I \,\middle|\, (\Sigma_i \cap \Sigma_a) - \{tick\} \neq \emptyset \right\},$$
$$I_B = \left\{ i \in I \,\middle|\, (\Sigma_i \cap \Sigma_b) - \{tick\} \neq \emptyset \right\}.$$

Em seguida, definimos as *plantas locais*

$$\mathbf{G}_A = \mathop{||}_{i \in I_A} \mathbf{G}_i \qquad \text{e} \qquad \mathbf{G}_B = \mathop{||}_{i \in I_B} \mathbf{G}_i \,,$$

cada qual compreendendo precisamente os subsistemas que são afetados pela respectiva especificação. Os comportamentos desejados para essas plantas locais são, então, dados por

$$K_A = E_a \,||\, L_m(\mathbf{G}_A) \qquad \text{e} \qquad K_B = E_b \,||\, L_m(\mathbf{G}_B)\,.$$

Nossa abordagem consistirá em obter supervisores locais $\mathcal{S}_A$ e $\mathcal{S}_B$ para $\mathbf{G}_A$ e $\mathbf{G}_B$ de modo a satisfazer K_A e K_B, respectivamente, e estabelecer condições sob as quais o comportamento global, com ambos os supervisores agindo em paralelo, seja não bloqueante e obedeça a $E_a \,||\, E_b \,||\, L_m(\mathbf{G})$ de forma minimamente restritiva. Isso equivale a di-

zer que buscamos uma solução *ótima* para o PCSML, de modo que $L_m(\mathcal{S}_A \wedge \mathcal{S}_B/\mathbf{G}) = sup\mathcal{C}\big(E_a \,||\, E_b \,||\, L_m(\mathbf{G})\big)$.

Para o cálculo dos supervisores locais, trata-se cada módulo (planta local + especificação relacionada) como um problema de controle separado, solucionado por meio de uma síntese monolítica. Apesar de não considerarmos aqui o procedimento de síntese de maneira explícita, assumiremos que $sup\mathcal{C}(K_A)$ e $sup\mathcal{C}(K_B)$ são ambas não vazias, o que garante, pelos resultados de (BRANDIN; WONHAM, 1994), a existência de uma solução ótima para cada problema local.

Denotaremos a parte controlada da planta por

$$\mathbf{G}_{AB} = \mathbf{G}_A \,||\, \mathbf{G}_B \,.$$

Por questão de brevidade, denotaremos também o comportamento desejado para $\mathbf{G}_{AB}$ por

$$K_{AB} = K_A \,||\, K_B = E_a \,||\, E_b \,||\, L_m(\mathbf{G}_{AB})$$

e o comportamento desejado global por

$$K = E_a \,||\, E_b \,||\, L_m(\mathbf{G}) \,.$$

Para $\Phi \in \{A, B, AB\}$, definimos a *planta complementar*

$$\mathbf{G}^c_\Phi = \Big|\Big| \limits_{i \in (I - I_\Phi)} \mathbf{G}_i \,.$$

O alfabeto de $\mathbf{G}^c_\Phi$ é denotado por Σ^c_Φ, e a projeção natural correspondente por P^c_Φ.

Vale ressaltar que nem todos os subsistemas $\mathbf{G}_i$ estão necessariamente envolvidos em alguma das plantas locais. Com isso, a parte não controlada, formada pelos subsistemas que não são afetados por nenhuma especificação, é a planta complementar $\mathbf{G}^c_{AB}$. Note que $K = K_{AB} \,||\, L_m(\mathbf{G}^c_{AB})$.

Sem perda de generalidade, trabalharemos sob a hipótese de que $\mathbf{G}^c_{AB}$ é não bloqueante. Caso isso não fosse verdade, seria natural projetar um supervisor independente para $\mathbf{G}^c_{AB}$, digamos $\mathcal{S}^c_{AB}$, de modo a resolver qualquer situação de bloqueio. Nesse caso, para o problema de controle a parte não controlada da planta seria tomada como o subsistema em malha fechada $\mathcal{S}^c_{AB}/\mathbf{G}^c_{AB}$.

4.2 EXEMPLO ILUSTRATIVO

Para tornar mais palpáveis as ideias introduzidas na Seção 4.1, considere um SPT $\mathbf{G}$ formado por três subsistemas, $\mathbf{G}_1$, $\mathbf{G}_2$ e $\mathbf{G}_3$, cujos GTA e GTT são mostrados nas Figuras 13 e 14, respectivamente. A princípio, não daremos qualquer interpretação física a esse sistema; este exemplo servirá apenas como uma ilustração simples e abstrata da nossa metodologia de controle. Os alfabetos de eventos dos subsistemas são dados por $\Sigma_1 = \{\omega, \theta, tick\}$, $\Sigma_2 = \{\alpha, \beta, \lambda, tick\}$ e $\Sigma_3 = \{\gamma, tick\}$, podendo ser facilmente observado que a propriedade (4.2) é satisfeita. Sendo $\Sigma = \Sigma_1 \cup \Sigma_2 \cup \Sigma_3$ o alfabeto de $\mathbf{G}$, assumiremos que $\Sigma_{\text{pro}} = \{\omega, \theta, \lambda\}$ e $\Sigma_{\text{for}} = \{\omega, \theta, \beta\}$. Duas especificações de controle, E_a e E_b, são colocadas para $\mathbf{G}$, representadas pelos autômatos da Figura 15. A primeira delas impõe que ω ocorra imediatamente após β e que, na sequência, θ ocorra sempre imediatamente após α. A segunda estabelece que β deve ou ocorrer antes do primeiro *tick*, ou não ocorrer jamais. Note que E_a afeta $\mathbf{G}_1$ e $\mathbf{G}_2$, enquanto E_b afeta apenas $\mathbf{G}_2$. Sendo assim, as plantas locais ficam definidas como $\mathbf{G}_A = \mathbf{G}_1 \,||\, \mathbf{G}_2$ e $\mathbf{G}_B = \mathbf{G}_2$.

Para obter as linguagens K_A e K_B, referentes aos comportamentos desejados para $\mathbf{G}_A$ e $\mathbf{G}_B$, cada especificação é composta com a respectiva planta local, conforme esquematizado na Figura 16. Vê-se que a parte não controlada do sistema, não afetada por nenhuma das especificações, é dada por $\mathbf{G}_{AB}^{c} = \mathbf{G}_3$. Na Figura 17 são mostrados autômatos que reconhecem as linguagens K_A e K_B. Uma solução modular local para este problema, impondo esses comportamentos desejados locais de maneira a garantir a otimidade global, será apresentada na Seção 4.7.

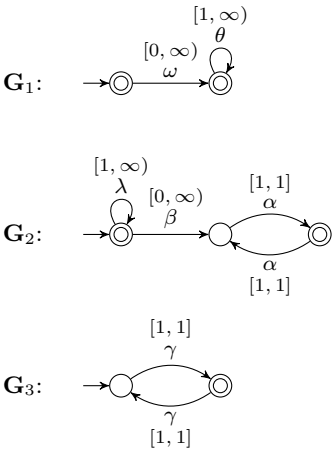

Figura 13 – GTA para os subsistemas $\mathbf{G}_1$, $\mathbf{G}_2$ e $\mathbf{G}_3$.

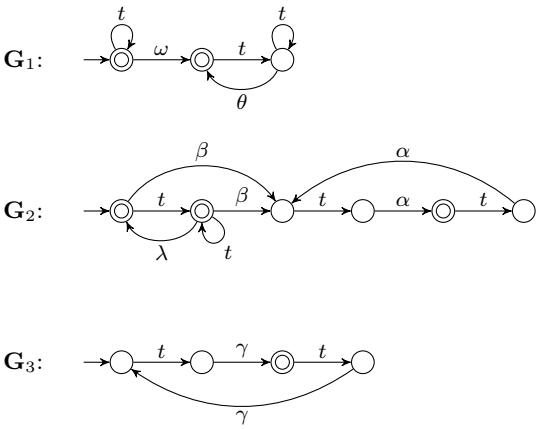

Figura 14 – GTT para os subsistemas da Figura 13.

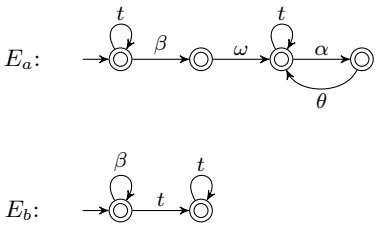

Figura 15 – Autômatos que representam as especificações E_a e E_b.

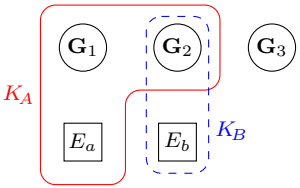

Figura 16 – Representação esquemática da abordagem modular local.

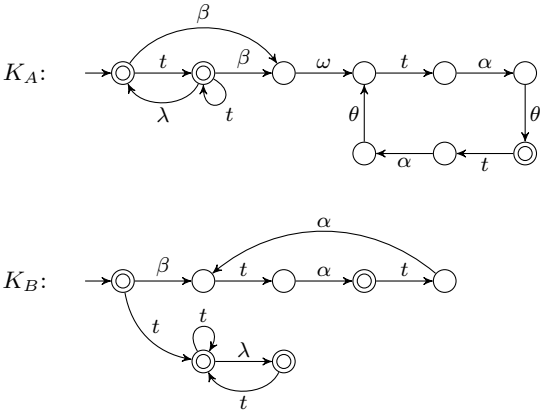

Figura 17 – Comportamentos desejados locais K_A e K_B.

4.3 PROPRIEDADES IMPORTANTES

Duas propriedades terão papéis fundamentais nos resultados do restante deste capítulo. No produto síncrono de linguagens definidas sobre alfabetos que compartilham um ou mais eventos, é possível que cadeias sejam mutuamente excluídas, de modo que a linguagem resultante contenha uma versão mais restrita de cada uma das linguagens originais. Em síntese, estas podem ser *conflitantes*. O *não conflito* entre linguagens L_i, $i \in \{1, \ldots, n\}$, é definido por

$$\left|\left|_{i=1}^{n} \overline{L_i} \right. \right. \;=\; \overline{\left|\left|_{i=1}^{n} L_i \right. \right.} \; .$$

Se as linguagens L_i representam comportamentos marcados de plantas locais, a condição determina que, para cada cadeia global s, se todas as suas versões locais $P_i(s)$ podem ser continuadas até estados marcados locais, então também s pode ser continuada até um estado marcado global.

A segunda é uma propriedade que emerge no contexto do controle modular de SEDT (BRANDIN; WONHAM, 1993). Uma linguagem $L \subseteq L(\mathbf{G})$ é *coerciva* com respeito a $\mathbf{G}$ se, $\forall s \in \overline{L}$,

$$tick \in \Sigma_{\mathbf{G}}(s) \;\;\&\;\; \Sigma_L(s) \cap \Sigma_{\text{for}} = \emptyset \;\;\Rightarrow\;\; tick \in \Sigma_L(s) \, .$$

Observe que L é coerciva exatamente quando a condição (c_2) da definição de controlabilidade for satisfeita. Intuitivamente, se L representa um comportamento desejado para $\mathbf{G}$ a ser imposto por supervisão, quando quer que *tick* esteja elegível em $L(\mathbf{G})$ ele também deve estar elegível em L, a menos que haja um evento forçável elegível em L que possa preemptá-lo.

Duas linguagens $L_A \subseteq L(\mathbf{G}_A)$ e $L_B \subseteq L(\mathbf{G}_B)$ são ditas *juntamente coercivas* com respeito a $\mathbf{G}_{AB}$ se $L_A \,||\, L_B$ for coerciva com respeito a $\mathbf{G}_{AB}$. Portanto, se L_A e L_B são comportamentos desejados que devem ser simultaneamente impostos sobre $\mathbf{G}_A$ e $\mathbf{G}_B$, a coercividade conjunta significa que *tick* somente pode ser preemptado por qualquer das duas linguagens caso exista algum evento forçável σ que "sobreviva" à sua sincronização, i. e., ou σ está elegível em uma dentre L_A e L_B e não pertence ao alfabeto da outra, ou $\sigma \in \Sigma_A \cup \Sigma_B$ e está elegível em ambas L_A e L_B.

4.4 CONJUNÇÃO DE SUPERVISORES

A possibilidade de que dois ou mais supervisores locais atuem sobre o mesmo subsistema torna necessária a existência de uma regra que defina qual será a ação de controle resultante. Por exemplo, se a ocorrência de um dado evento é proibida por um supervisor enquanto outro a permite, é preciso que essa regra determine qual das duas ações será efetivamente transmitida ao sistema.

Considere, então, dois supervisores $\mathcal{S}_A$ e $\mathcal{S}_B$ para os SEDT $\mathbf{G}_A$ e $\mathbf{G}_B$, respectivamente, definidos conforme a Seção 2.4. A *conjunção* de $\mathcal{S}_A$ e $\mathcal{S}_B$ é o mapa

$$\mathcal{S}_A \wedge \mathcal{S}_B : L(\mathbf{G}_A) \,||\, L(\mathbf{G}_B) \;\rightarrow\; 2^{\Sigma_A \cup \Sigma_B}$$

definido, $\forall s \in L(\mathbf{G}_A) \,||\, L(\mathbf{G}_B)$ e $\forall \sigma \in \Sigma_A \cup \Sigma_B$, por:

- se $\sigma \in \Sigma_A \cap \Sigma_B$, então $\sigma \in (\mathcal{S}_A \wedge \mathcal{S}_B)(s) \;\Leftrightarrow\; \sigma \in \mathcal{S}_A\big(P_A(s)\big) \cap \mathcal{S}_B\big(P_B(s)\big)$;
- se $\sigma \in \Sigma_A - \Sigma_B$, então $\sigma \in (\mathcal{S}_A \wedge \mathcal{S}_B)(s) \;\Leftrightarrow\; \sigma \in \mathcal{S}_A\big(P_A(s)\big)$;
- se $\sigma \in \Sigma_B - \Sigma_A$, então $\sigma \in (\mathcal{S}_A \wedge \mathcal{S}_B)(s) \;\Leftrightarrow\; \sigma \in \mathcal{S}_B\big(P_B(s)\big)$.

Assim, um evento compartilhado por $\mathbf{G}_A$ e $\mathbf{G}_B$ pertence a $(\mathcal{S}_A \wedge \mathcal{S}_B)(s)$ se e somente se for habilitado por ambos $\mathcal{S}_A$ e $\mathcal{S}_B$, enquanto um evento exclusivo a um dos sistemas pertence a $(\mathcal{S}_A \wedge \mathcal{S}_B)(s)$ quando sua ocorrência for permitida pelo supervisor correspondente. Essa definição é uma adaptação da versão não temporizada apresentada em (WONHAM; RAMADGE, 1988), aqui alterada de modo a abranger o caso de supervisão local, em que cada supervisor é definido sobre uma planta local distinta (e portanto com um alfabeto de eventos também distinto).

Lembre que linguagens marcadoras M_A e M_B estão implicitamente associadas a $\mathcal{S}_A$ e $\mathcal{S}_B$. A linguagem marcadora para a sua conjunção é, então, definida como $M_A \,||\, M_B \subseteq L_m(\mathbf{G}_A) \,||\, L_m(\mathbf{G}_B)$.

Para que a nossa definição de conjunção faça sentido, é desejável que a ação de controle imposta por cada um dos supervisores não seja "enfraquecida", ou seja, que os eventos proibidos por cada um deles na respectiva planta local permaneçam sendo proibidos globalmente. Caso contrário, o cumprimento de uma especificação mediante a síntese local não garantiria que ela fosse respeitada pelo sistema global sob supervisão. Em suma, deve-se questionar se a ação conjunta dos supervisores locais sobre o sistema controlado resulta em um comportamento

equivalente à operação paralela (síncrona) das plantas locais em malha fechada. O seguinte resultado fornece a resposta.

Proposição 2.

a) $L(\mathcal{S}_A \wedge \mathcal{S}_B/\mathbf{G}_{AB}) = L(\mathcal{S}_A/\mathbf{G}_A) \,||\, L(\mathcal{S}_B/\mathbf{G}_B)$;

b) $L_m(\mathcal{S}_A \wedge \mathcal{S}_B/\mathbf{G}_{AB}) = L_m(\mathcal{S}_A/\mathbf{G}_A) \,||\, L_m(\mathcal{S}_B/\mathbf{G}_B)$.

Demonstração. Para (a), mostraremos que, $\forall s \in L(\mathbf{G}_{AB})$,

$$s \in L(\mathcal{S}_A \wedge \mathcal{S}_B/\mathbf{G}_{AB}) \;\Leftrightarrow\; s \in L(\mathcal{S}_A/\mathbf{G}_A) \,||\, L(\mathcal{S}_B/\mathbf{G}_B)\,, \qquad (4.3)$$

por indução sobre o comprimento de s. Primeiramente, ambos os lados de (4.3) são verdadeiros para ε, por definição. Agora, assuma que (4.3) seja válido para qualquer $s \in L(\mathbf{G}_{AB})$ tal que $|\, s \,| = k \geq 0$, e tome $r \in L(\mathbf{G}_{AB})$ com $|\, r \,| = k+1$. Podemos escrever $r = v\sigma$, com $v \in L(\mathbf{G}_{AB})$, $|\, v \,| = k$ e $\sigma \in \Sigma_A \cup \Sigma_B$. Temos, então, três possibilidades distintas:

i. $\sigma \in \Sigma_A \cap \Sigma_B$:

$$\begin{aligned}
& v\sigma \in L(\mathcal{S}_A \wedge \mathcal{S}_B/\mathbf{G}_{AB}) \\
\Leftrightarrow\; & v \in L(\mathcal{S}_A \wedge \mathcal{S}_B/\mathbf{G}_{AB}) \;\;\&\;\; \sigma \in (\mathcal{S}_A \wedge \mathcal{S}_B)(v) \\
& \quad \&\;\; v\sigma \in L(\mathbf{G}_{AB}) \\
\Leftrightarrow\; & v \in L(\mathcal{S}_A/\mathbf{G}_A) \,||\, L(\mathcal{S}_B/\mathbf{G}_B) \;\;\&\;\; \sigma \in \mathcal{S}_A\big(P_A(v)\big) \cap \mathcal{S}_B\big(P_B(v)\big) \\
& \quad \&\;\; v\sigma \in L(\mathbf{G}_A) \,||\, L(\mathbf{G}_B) \\
\Leftrightarrow\; & P_A(v\sigma) \in L(\mathcal{S}_A/\mathbf{G}_A) \;\;\&\;\; P_B(v\sigma) \in L(\mathcal{S}_B/\mathbf{G}_B) \\
\Leftrightarrow\; & v\sigma \in L(\mathcal{S}_A/\mathbf{G}_A) \,||\, L(\mathcal{S}_B/\mathbf{G}_B)\,;
\end{aligned}$$

ii. $\sigma \in \Sigma_A - \Sigma_B$:

$$\begin{aligned}
& v\sigma \in L(\mathcal{S}_A \wedge \mathcal{S}_B/\mathbf{G}_{AB}) \\
\Leftrightarrow\; & v \in L(\mathcal{S}_A \wedge \mathcal{S}_B/\mathbf{G}_{AB}) \;\;\&\;\; \sigma \in (\mathcal{S}_A \wedge \mathcal{S}_B)(v) \\
& \quad \&\;\; v\sigma \in L(\mathbf{G}_{AB}) \\
\Leftrightarrow\; & v \in L(\mathcal{S}_A/\mathbf{G}_A) \,||\, L(\mathcal{S}_B/\mathbf{G}_B) \;\;\&\;\; \sigma \in \mathcal{S}_A\big(P_A(v)\big) \\
& \quad \&\;\; v\sigma \in L(\mathbf{G}_A) \,||\, L(\mathbf{G}_B) \\
\Leftrightarrow\; & P_A(v\sigma) \in L(\mathcal{S}_A/\mathbf{G}_A) \;\;\&\;\; P_B(v\sigma) = P_B(v) \in L(\mathcal{S}_B/\mathbf{G}_B) \\
\Leftrightarrow\; & v\sigma \in L(\mathcal{S}_A/\mathbf{G}_A) \,||\, L(\mathcal{S}_B/\mathbf{G}_B)\,;
\end{aligned}$$

iii. $\sigma \in \Sigma_B - \Sigma_A$: segue por analogia direta com o caso (ii).

Isso conclui a indução e prova (a). Para (b), temos

$$\begin{aligned} L_m(\mathcal{S}_A \wedge \mathcal{S}_B/\mathbf{G}_{AB}) &= L(\mathcal{S}_A \wedge \mathcal{S}_B/\mathbf{G}_{AB}) \cap M_A \,||\, M_B \\ &= L(\mathcal{S}_A/\mathbf{G}_A) \,||\, L(\mathcal{S}_B/\mathbf{G}_B) \cap M_A \,||\, M_B \\ &= L(\mathcal{S}_A/\mathbf{G}_A) \cap M_A \,||\, L(\mathcal{S}_B/\mathbf{G}_B) \cap M_B \\ &= L_m(\mathcal{S}_A/\mathbf{G}_A) \,||\, L_m(\mathcal{S}_B/\mathbf{G}_B)\,. \end{aligned}$$

□

A ação conjunta de dois supervisores locais garante, portanto, que o sistema controlado tenha o comportamento esperado. Surge então uma nova questão, ainda mais fundamental: a conjunção de dois supervisores resulta de fato em um novo supervisor? Mais especificamente, o resultado da operação $\mathcal{S}_A \wedge \mathcal{S}_B$ respeita as condições (s_1) e (s_2) da definição de supervisor (Seção 2.4)? E, em caso afirmativo, o não bloqueio dos supervisores locais garante que sua conjunção seja não bloqueante? Ocorre que, em geral, a resposta é negativa. Para ver o que pode dar errado, suponha que, para uma dada cadeia $s \in L(\mathbf{G}_{AB})$, tenhamos $\mathit{tick} \in \Sigma_{\mathbf{G}_{AB}}(s)$, e que $\mathcal{S}_A$ preempte tick em $\mathbf{G}_A$ forçando um evento $\sigma \in (\Sigma_A \cap \Sigma_B)_{\mathrm{for}}$, de modo que $\mathit{tick} \notin \mathcal{S}_A\big(P_A(s)\big)$. Como $\mathit{tick} \in \Sigma_A \cap \Sigma_B$, da definição de conjunção isso implicaria que $\mathit{tick} \notin (\mathcal{S}_A \wedge \mathcal{S}_B)(s)$. Se, além disso, assumirmos que σ seja proibível e seja desabilitado por $\mathcal{S}_B$ em $\mathbf{G}_B$, significando que $\sigma \notin \mathcal{S}_B\big(P_B(s)\big)$, seria também o caso de que $\sigma \notin (\mathcal{S}_A \wedge \mathcal{S}_B)(s)$. Não havendo nenhum outro evento forçável elegível, isso resultaria em $(\mathcal{S}_A \wedge \mathcal{S}_B)(s) \cap (\Sigma_A \cup \Sigma_B)_{\mathrm{for}} = \emptyset$, violando a condição ($\mathrm{s}_2$). Colocando em poucas palavras, a conjunção "falha" porque um dos supervisores desabilita o evento forçável do qual o outro depende para preemptar *tick*.

Aliás, mesmo que a conjunção obedeça às condições da definição de supervisor, e mesmo que $\mathcal{S}_A$ e $\mathcal{S}_B$ sejam não bloqueantes, sua conjunção pode ser bloqueante. Um raciocínio similar ao empregado acima pode ser usado para mostrar que, por meio da desabilitação de eventos compartilhados, todas as cadeias permitidas por cada supervisor e que levam a estados marcados poderiam ser proibidas pelo outro supervisor, fazendo com que nenhum estado marcado pudesse ser alcançado quando ambas as ações fossem simultaneamente aplicadas.

Portanto, é interessante estabelecer condições que garantam que $\mathcal{S}_A \wedge \mathcal{S}_B$ seja um supervisor para $\mathbf{G}_{AB}$ e que sua ação resulte em um comportamento global não bloqueante. Revisitando a definição de coercividade conjunta (Seção 4.3), fica intuitivamente claro que a verificação

dessa condição para dois supervisores impede que sua conjunção viole (s_2) como na situação hipotética acima. Ademais, a propriedade de não conflito desponta como uma candidata natural para evitar o bloqueio. O próximo teorema formaliza essas ideias. Antes, porém, uma consideração e um resultado preliminar se fazem necessários.

Observação 6. A conjunção $\mathcal{S}_\wedge = \mathcal{S}_A \wedge \mathcal{S}_B$ pode ser estendida para atuar sobre $\mathbf{G} = \mathbf{G}_{AB} \,||\, \mathbf{G}^c_{AB}$, com a regra natural de que, $\forall s \in L(\mathbf{G})$, $\mathcal{S}_\wedge(s) = \mathcal{S}_\wedge\big(P_{AB}(s)\big) \dot\cup \big(\Sigma^c_{AB} - \{tick\}\big)$. Pode ser visto facilmente que isso resulta em $L(\mathcal{S}_\wedge/\mathbf{G}) = L(\mathcal{S}_\wedge/\mathbf{G}_{AB}) \,||\, L(\mathbf{G}^c_{AB})$, de maneira que $\mathcal{S}_\wedge$ não afeta diretamente o comportamento de $\mathbf{G}^c_{AB}$, como esperado. É também natural assumir que a linguagem marcadora M seja substituída por $M \,||\, L_m(\mathbf{G}^c_{AB})$, garantindo que a marcação de $\mathbf{G}^c_{AB}$ não seja alterada, i. e., $L_m(\mathcal{S}_\wedge/\mathbf{G}) = L_m(\mathcal{S}_\wedge/\mathbf{G}_{AB}) \,||\, L_m(\mathbf{G}^c_{AB})$.

Proposição 3. *A conjunção $\mathcal{S}_A \wedge \mathcal{S}_B$ é um supervisor para $\mathbf{G}_{AB}$ se e somente se a sua extensão, feita pelas regras da Observação 6, for um supervisor para $\mathbf{G}$.*

Demonstração. (somente se) Seja $\mathcal{S}_\wedge = \mathcal{S}_A \wedge \mathcal{S}_B$. Verificaremos as condições (s_1) e (s_2) da definição de supervisor. Para (s_1), note que, $\forall s \in L(\mathcal{S}_\wedge/\mathbf{G})$, $(\Sigma_A \cup \Sigma_B)_{nc} \subseteq \mathcal{S}_\wedge\big(P_{AB}(s)\big) \subseteq \mathcal{S}_\wedge(s)$ (já que $\mathcal{S}_\wedge$ é um supervisor para $\mathbf{G}_{AB}$), e $(\Sigma^c_{AB})_{nc} \subseteq \mathcal{S}_\wedge(s)$ (pela Observação 6), logo $\Sigma_{nc} \subseteq \mathcal{S}_\wedge(s)$.

Para (s_2), $\forall s \in L(\mathcal{S}_\wedge/\mathbf{G})$ temos

$$\begin{aligned}
&\mathcal{S}_\wedge(s) \cap \Sigma_{\mathbf{G}}(s) \cap \Sigma_{\text{for}} = \emptyset \;\;\&\;\; tick \in \Sigma_{\mathbf{G}}(s) \\
\Rightarrow\; &\mathcal{S}_\wedge\big(P_{AB}(s)\big) \cap \Sigma_{\mathbf{G}_{AB}}\big(P_{AB}(s)\big) \cap (\Sigma_A \cup \Sigma_B)_{\text{for}} = \emptyset \\
&\quad \&\;\; tick \in \Sigma_{\mathbf{G}_{AB}}\big(P_{AB}(s)\big) \\
\Rightarrow\; &tick \in \mathcal{S}_\wedge\big(P_{AB}(s)\big) \\
\Rightarrow\; &tick \in \mathcal{S}_\wedge(s)\,.
\end{aligned}$$

(se) Primeiramente, mostraremos que, quando quer que *tick* esteja sendo preemptado, existe um evento forçável de $\Sigma_A \cup \Sigma_B$ que está elegível, i. e., $\forall s \in L(\mathcal{S}_\wedge/\mathbf{G})$,

$$tick \in \Sigma_{\mathbf{G}}(s) - \mathcal{S}_\wedge(s) \;\Rightarrow\; \mathcal{S}_\wedge\big(P_{AB}(s)\big) \cap \Sigma_{\mathbf{G}_{AB}}\big(P_{AB}(s)\big) \cap (\Sigma_A \cup \Sigma_B)_{\text{for}} \;\neq\; \emptyset\,. \quad (4.4)$$

Isso significa que $\mathcal{S}_\wedge$ nunca depende de qualquer evento de Σ^c_{AB} para efetuar uma preempção. Para mostrar esse fato, veja que da Observação 2 temos que $\forall s \in L(\mathcal{S}_\wedge/\mathbf{G})$, $\exists u \in (\Sigma^c_{AB})^*_{\text{at}} \mid \Sigma_{\mathbf{G}^c_{AB}}\big(P^c_{AB}(su)\big) = \{tick\}$.

Como $P_{AB}(su) = P_{AB}(s)$ e portanto $tick \in \mathcal{S}_\wedge(s) \Leftrightarrow tick \in \mathcal{S}_\wedge(su)$, segue que

$$\begin{aligned} & tick \in \Sigma_{\mathbf{G}}(s) - \mathcal{S}_\wedge(s) \\ \Rightarrow\ & tick \in \Sigma_{\mathbf{G}}(su) - \mathcal{S}_\wedge(su) \\ \overset{(*)}{\Rightarrow}\ & \mathcal{S}_\wedge(su) \cap \Sigma_{\mathbf{G}}(su) \cap (\Sigma_A \cup \Sigma_B)_{\text{for}} \neq \emptyset \\ \Rightarrow\ & \mathcal{S}_\wedge\big(P_{AB}(s)\big) \cap \Sigma_{\mathbf{G}_{AB}}\big(P_{AB}(s)\big) \cap (\Sigma_A \cup \Sigma_B)_{\text{for}} \neq \emptyset\,, \end{aligned}$$

com $(*)$ resultando de $\Sigma_{\mathbf{G}}(su) \cap (\Sigma^c_{AB})_{\text{act}} = \emptyset$ e da hipótese de que $\mathcal{S}_\wedge$ é um supervisor para $\mathbf{G}$. Isso prova (4.4).

Mostraremos ainda que

$$\forall r \in L(\mathcal{S}_\wedge/\mathbf{G}_{AB}),\ \exists s \in L(\mathcal{S}_\wedge/\mathbf{G}) \mid r = P_{AB}(s)\,. \tag{4.5}$$

Defina a projeção $P_t : \Sigma^* \to \{tick\}^*$. Como consequência da Observação 2 temos que $\{tick\}^* \subseteq P_t\big(L(\mathbf{G}^c_{AB})\big)$, o que implica que

$$\forall r \in L(\mathcal{S}_\wedge/\mathbf{G}_{AB}),\ \exists v \in L(\mathbf{G}^c_{AB}) \mid P_t(v) = P_t(r)\,.$$

Como

$$\emptyset \ \neq\ P^{-1}_{AB}(r) \cap (P^c_{AB})^{-1}(v) \ \subseteq\ L(\mathcal{S}_\wedge/\mathbf{G})\,,$$

a afirmação (4.5) está provada.

Agora, seja $r \in L(\mathcal{S}_\wedge/\mathbf{G}_{AB})$, e tome $s \in L(\mathcal{S}_\wedge/\mathbf{G})$ tal que $r = P_{AB}(s)$. Assim como anteriormente, verificaremos as condições. Para (s_1), segue diretamente que $\Sigma_{nc} \subseteq \mathcal{S}_\wedge(s) \Rightarrow (\Sigma_A \cup \Sigma_B)_{nc} \subseteq \mathcal{S}_\wedge(r)$. Para (s_2), provaremos a condição equivalente

$$tick \in \Sigma_{\mathbf{G}_{AB}}(r) - \mathcal{S}_\wedge(r)\ \Rightarrow\ \mathcal{S}_\wedge(r) \cap \Sigma_{\mathbf{G}_{AB}}(r) \cap (\Sigma_A \cup \Sigma_B)_{\text{for}} \neq \emptyset\,. \tag{4.6}$$

Se $tick \in \Sigma_{\mathbf{G}_{AB}}(r) - \mathcal{S}_\wedge(r)$, então $tick \notin \mathcal{S}_\wedge(s)$ e temos um dos seguintes casos:

i. $tick \in \Sigma_{\mathbf{G}}(s)$: Verifica-se (4.6) como consequência direta de (4.4).

ii. $tick \notin \Sigma_{\mathbf{G}}(s)$: Neste caso $tick \notin \Sigma_{\mathbf{G}^c_{AB}}\big(P^c_{AB}(s)\big)$, portanto $\exists u \in (\Sigma^c_{AB})^+_{\text{spe}} \mid tick \in \Sigma_{\mathbf{G}^c_{AB}}\big(P^c_{AB}(su)\big)$ (da Observação 2). Como isso significa que $tick \in \Sigma_{\mathbf{G}}(su) - \mathcal{S}_\wedge(su)$, e sendo que $P_{AB}(su) = P_{AB}(s) = r$, retornamos ao caso (i) trocando s por su.

□

Teorema 1. *Sejam $\mathcal{S}_A$ e $\mathcal{S}_B$ supervisores não bloqueantes para $\mathbf{G}_A$ e $\mathbf{G}_B$, respectivamente. $\mathcal{S}_A \wedge \mathcal{S}_B$ define um supervisor para $\mathbf{G}_{AB}$ se e somente se $L(\mathcal{S}_A/\mathbf{G}_A)$ e $L(\mathcal{S}_B/\mathbf{G}_B)$ forem juntamente coercivas com respeito a $\mathbf{G}_{AB}$. Além disso, esse supervisor é não bloqueante para $\mathbf{G}$ se e somente se $L_m(\mathcal{S}_A/\mathbf{G}_A)$, $L_m(\mathcal{S}_B/\mathbf{G}_B)$ e $L_m(\mathbf{G}^c_{AB})$ forem não conflitantes.*

Demonstração. Para a primeira afirmação, verificaremos as condições da definição de supervisor. Para (s_1), $\forall s \in L(\mathcal{S}_A \wedge \mathcal{S}_B/\mathbf{G}_{AB})$ temos

$$\begin{aligned}\sigma \in (\Sigma_A \cup \Sigma_B)_{nc} \;&\Rightarrow\; \big[\sigma \in \Sigma_\Psi \Rightarrow \sigma \in \mathcal{S}_\Psi\big(P_\Psi(s)\big)\big],\;\; \forall\; \Psi \in \{A, B\} \\ &\Rightarrow\; \sigma \in (\mathcal{S}_A \wedge \mathcal{S}_B)(s)\,.\end{aligned}$$

Isso mostra que (s_1) é sempre válida.

Basta, portanto, mostrar que (s_2) é verificada se e somente se $L(\mathcal{S}_A/\mathbf{G}_A)$ e $L(\mathcal{S}_B/\mathbf{G}_B)$ forem juntamente coercivas. Primeiramente, note que $\forall s \in L(\mathcal{S}_A \wedge \mathcal{S}_B/\mathbf{G}_{AB})$ e $\forall \sigma \in (\Sigma_A \cup \Sigma_B)_{\mathrm{for}}$ temos

$$\sigma \in (\mathcal{S}_A \wedge \mathcal{S}_B)(s)\;\&\; s\sigma \in L(\mathbf{G}_{AB}) \;\;\Leftrightarrow\;\; s\sigma \in L(\mathcal{S}_A \wedge \mathcal{S}_B/\mathbf{G}_{AB})\,,$$

o que implica que

$$\begin{gathered}(\mathcal{S}_A \wedge \mathcal{S}_B)(s) \cap \Sigma_{\mathbf{G}_{AB}}(s) \cap (\Sigma_A \cup \Sigma_B)_{\mathrm{for}} = \emptyset \\ \Updownarrow \\ \Sigma_{L(\mathcal{S}_A \wedge \mathcal{S}_B/\mathbf{G}_{AB})}(s) \cap (\Sigma_A \cup \Sigma_B)_{\mathrm{for}} = \emptyset\,.\end{gathered} \tag{4.7}$$

Além disso, $\forall s \in L(\mathcal{S}_A \wedge \mathcal{S}_B/\mathbf{G}_{AB})$, se $tick \in \Sigma_{\mathbf{G}_{AB}}(s)$ temos

$$tick \in \Sigma_{L(\mathcal{S}_A \wedge \mathcal{S}_B/\mathbf{G}_{AB})}(s) \;\;\Leftrightarrow\;\; tick \in (\mathcal{S}_A \wedge \mathcal{S}_B)(s) \tag{4.8}$$

diretamente da definição de $L(\mathcal{S}_A \wedge \mathcal{S}_B/\mathbf{G}_{AB})$. Podemos então concluir, recordando a parte (a) da Proposição 2, que a combinação de (4.7) e (4.8) resulta em

$$\begin{gathered}\big[\,(\mathcal{S}_A \wedge \mathcal{S}_B)(s) \cap \Sigma_{\mathbf{G}_{AB}}(s) \cap (\Sigma_A \cup \Sigma_B)_{\mathrm{for}} = \emptyset \;\;\&\;\; tick \in \Sigma_{\mathbf{G}_{AB}}(s)\,\big] \\ \Rightarrow\;\; tick \in (\mathcal{S}_A \wedge \mathcal{S}_B)(s) \\ \Updownarrow \\ \big[\,\Sigma_{L(\mathcal{S}_A/\mathbf{G}_A)||L(\mathcal{S}_B/\mathbf{G}_B)}(s) \cap (\Sigma_A \cup \Sigma_B)_{\mathrm{for}} = \emptyset \;\;\&\;\; tick \in \Sigma_{\mathbf{G}_{AB}}(s)\,\big] \\ \Rightarrow\;\; tick \in \Sigma_{L(\mathcal{S}_A/\mathbf{G}_A)||L(\mathcal{S}_B/\mathbf{G}_B)}(s)\end{gathered}$$

$\forall s \in L(\mathcal{S}_A \wedge \mathcal{S}_B/\mathbf{G}_{AB})$, como desejado.

Para a segunda afirmação, a Proposição 3 garante que $\mathcal{S}_A \wedge \mathcal{S}_B$ é também um supervisor para $\mathbf{G}$, e temos

$$\begin{aligned} L(\mathcal{S}_A \wedge \mathcal{S}_B/\mathbf{G}) &= L(\mathcal{S}_A \wedge \mathcal{S}_B/\mathbf{G}_{AB}) \,||\, L(\mathbf{G}_{AB}^c) \\ &= L(\mathcal{S}_A/\mathbf{G}_A) \,||\, L(\mathcal{S}_B/\mathbf{G}_B) \,||\, L(\mathbf{G}_{AB}^c) \\ &= \overline{L_m(\mathcal{S}_A/\mathbf{G}_A)} \,||\, \overline{L_m(\mathcal{S}_B/\mathbf{G}_B)} \,||\, \overline{L_m(\mathbf{G}_{AB}^c)}\,, \end{aligned}$$

onde a primeira igualdade segue da Observação 6, a segunda da Proposição 2, e a terceira do fato de que $\mathcal{S}_A$ e $\mathcal{S}_B$ são não bloqueantes, bem como $\mathbf{G}_{AB}^c$. Juntamente com

$$\overline{L_m(\mathcal{S}_A \wedge \mathcal{S}_B/\mathbf{G})} = \overline{L_m(\mathcal{S}_A/\mathbf{G}_A) \,||\, L_m(\mathcal{S}_B/\mathbf{G}_B) \,||\, L_m(\mathbf{G}_{AB}^c)}\,,$$

isso permite concluir que

$$\begin{aligned} L(\mathcal{S}_A \wedge \mathcal{S}_B/\mathbf{G}) &= \overline{L_m(\mathcal{S}_A \wedge \mathcal{S}_B/\mathbf{G})} \\ &\Updownarrow \\ \overline{L_m(\mathcal{S}_A/\mathbf{G}_A)} \,||\, \overline{L_m(\mathcal{S}_B/\mathbf{G}_B)} \,||\, \overline{L_m(\mathbf{G}_{AB}^c)} \\ = \overline{L_m(\mathcal{S}_A/\mathbf{G}_A) \,||\, L_m(\mathcal{S}_B/\mathbf{G}_B) \,||\, L_m(\mathbf{G}_{AB}^c)}\,, \end{aligned}$$

completando a demonstração. □

O Teorema 1 estabelece condições necessárias e suficientes sob as quais a conjunção de dois supervisores não bloqueantes locais é um supervisor não bloqueante global. Esse é, claramente, um primeiro e importante passo em direção à solução do nosso problema de controle. Resta garantir que o supervisor assim obtido confira o comportamento desejado ao sistema, sendo esse o objetivo da próxima seção.

Cabe aqui destacar uma relevante diferença com relação ao caso não temporizado. Se o tempo não for considerado, nunca pode haver conflito entre a planta controlada e a planta complementar. Isso porque, não havendo o compartilhamento de nenhum evento, a dinâmica das duas passa a ser completamente independente. Sendo assim, em tal caso apenas o não conflito entre as plantas locais precisa ser verificado. Em um sistema temporizado, por outro lado, o compartilhamento de *tick* pode levar a um resultado global bloqueante mesmo se $\mathcal{S}_A \wedge \mathcal{S}_B$ for não bloqueante para $\mathbf{G}_{AB}$. Isso torna necessáro que se inclua $L_m(\mathbf{G}_{AB}^c)$ no teste de não conflito. No exemplo da Seção 4.10 será mostrada uma situação na qual ocorre esse tipo de conflito.

4.5 OTIMIDADE GLOBAL POR SUPERVISÃO LOCAL

Nesta seção, apresentaremos condições para a solução ótima (minimamente restritiva) do PCSML. Mais precisamente, a questão que abordaremos é: dado que os supervisores $\mathcal{S}_A$ e $\mathcal{S}_B$ são localmente ótimos, i. e., são supervisores não bloqueantes tais que $L_m(\mathcal{S}_A/\mathbf{G}_A) = sup\mathcal{C}(K_A)$ e $L_m(\mathcal{S}_B/\mathbf{G}_B) = sup\mathcal{C}(K_B)$, sob que condições sua ação conjunta resulta na otimidade global?

O Teorema 2, a seguir, determina que a ação conjunta de dois supervisores locais ótimos nunca resulta em um comportamento mais restritivo que a de um supervisor monolítico ótimo para as mesmas especificações. Conforme será discutido na Seção 4.9, essa é uma condição fundamental para que eventuais conflitos possam ser resolvidos, i. e., para que a solução ótima possa ser calculada a partir do comportamento resultante da ação conjunta dos supervisores locais. Para a demonstração desse teorema, o Lema 1 será necessário. Perceba que, para um comportameto desejado representado por uma linguagem L, o SEDT sobre o qual $sup\mathcal{C}(L)$ é calculado é deixado implícito na notação, ficando claro de acordo com a própria definição de L. Por exemplo, recordando a notação da Seção 4.1, temos que $sup\mathcal{C}(K_A)$ se refere a $\mathbf{G}_A$, $sup\mathcal{C}(K_{AB})$ a $\mathbf{G}_{AB}$ e $sup\mathcal{C}(K)$ a $\mathbf{G}$.

Lema 1. *Denote* $\mathbf{G}_C = \mathbf{G}^c_{AB}$. *Então,* $P_\Psi\big[sup\mathcal{C}(K)\big]$ *é controlável com respeito a* $\mathbf{G}_\Psi$, $\forall\ \Psi \in \{A, B, C\}$.

Demonstração. Por questão de brevidade, escreva $K^\uparrow = sup\mathcal{C}(K)$. Será verificada apenas a controlabilidade de $P_A(K^\uparrow)$, sendo os demais casos inteiramente similares. Para a condição (c_1), temos

$$
\begin{aligned}
& s \in \overline{P_A(K^\uparrow)} \ \ \& \ \ \sigma \in \Sigma_{A_{nc}} \ \ \& \ \ s\sigma \in L(\mathbf{G}_A) \\
\Rightarrow\ & s = P_A(r), r \in \overline{K^\uparrow} \ \ \& \ \ \sigma \in \Sigma_{nc} \ \ \& \ \ \underbrace{r\sigma \in L(\mathbf{G})}_{(\dagger)} \\
\Rightarrow\ & r\sigma \in \overline{K^\uparrow} \\
\Rightarrow\ & P_A(r\sigma) = s\sigma \in P_A(\overline{K^\uparrow}) = \overline{P_A(K^\uparrow)}\,,
\end{aligned}
$$

com (†) justificado por $L(\mathbf{G}) = L(\mathbf{G}_A) \,||\, L(\mathbf{G}^c_A)$, $P_A(r\sigma) \in L(\mathbf{G}_A)$ e $P^c_A(r\sigma) = P^c_A(r) \in P^c_A(\overline{K^\uparrow}) \subseteq L(\mathbf{G}^c_A)$.

Para (c_2), veja que de

$$
s \in \overline{P_A(K^\uparrow)} \ \ \& \ \ tick \in \Sigma_{\mathbf{G}_A}(s) \ \ \& \ \ \Sigma_{P_A(K^\uparrow)}(s) \cap \Sigma_{A_{\text{for}}} = \emptyset
$$

podemos escrever $s = P_A(r)$, com $r \in \overline{K^\uparrow}$, e portanto temos um dos três seguintes casos:

i. $tick \in \Sigma_{\mathbf{G}}(r) \cap \Sigma_{K^\uparrow}(r)$:
Segue diretamente que $P_A(r\,tick) = s\,tick \in \overline{P_A(K^\uparrow)}$.

ii. $tick \in \Sigma_{\mathbf{G}}(r) - \Sigma_{K^\uparrow}(r)$:
Como $K^\uparrow$ é controlável, isso implica que $\Sigma_{K^\uparrow}(r) \cap \Sigma_{\text{for}} \neq \emptyset$. Uma vez que $\Sigma_{K^\uparrow}(r) \cap \Sigma_{A_{\text{for}}} = \emptyset$, devemos ter $\Sigma_{K^\uparrow}(r) \cap (\Sigma - \Sigma_A)_{\text{for}} \neq \emptyset$. Nesse caso, o fato de que $tick \in \Sigma_{\mathbf{G}_A}(s)$ juntamente com a Observação 4 implicam que

$$\exists u \in (\Sigma - \Sigma_A)^*_{\text{at}} \,\big|\, tick \in \Sigma_{K^\uparrow}(ru)\,,$$

o que, por sua vez, significa que $P_A(ru\,tick) = s\,tick \in \overline{P_A(K^\uparrow)}$.

iii. $tick \notin \Sigma_{\mathbf{G}}(r)$:
Como $P_A(r\,tick) \in L(\mathbf{G}_A)$, é verdade que $P^c_A(r\,tick) \notin L(\mathbf{G}^c_A)$. Portanto, da Observação 3

$$\exists x \in (\Sigma - \Sigma_A)^+_{\text{esp}} \,\big|\, P^c_A(rx) \in L(\mathbf{G}^c_A)\ \&\ tick \in \Sigma_{\mathbf{G}^c_A}(rx),$$

o que significa que $rx \in L(\mathbf{G})$ e $tick \in \Sigma_{\mathbf{G}}(rx)$. Agora, como $x \in \Sigma^+_{nc}$ e $K^\uparrow$ é controlável com respeito a $\mathbf{G}$, $rx \in \overline{K^\uparrow}$. Daí, se $\Sigma_{K^\uparrow}(rx) \cap \Sigma_{\text{for}} = \emptyset$, retornamos ao caso (i) trocando r por rx. Caso contrário, fazemos a mesma troca para o caso (ii).

□

Teorema 2.

$$sup\mathcal{C}(K_A) \,||\, sup\mathcal{C}(K_B) \,||\, L_m(\mathbf{G}^c_{AB}) \supseteq sup\mathcal{C}(K).$$

Demonstração. Usando a notação $K^\uparrow_A = sup\mathcal{C}(K_A)$, $K^\uparrow_B = sup\mathcal{C}(K_B)$ e $K^\uparrow = sup\mathcal{C}(K)$, do Lema 1 temos que $P_A(K^\uparrow)$ é controlável com respeito a $\mathbf{G}_A$. Além disso,

$$\begin{aligned} P_A(K^\uparrow) \subseteq\ & P_A(K) \\ &= P_A\big[P_A^{-1}(K_A) \cap P_B^{-1}(K_B) \cap (P^c_{AB})^{-1}\big(L_m(\mathbf{G}^c_{AB})\big)\big] \\ &\subseteq P_A\big[P_A^{-1}(K_A)\big] \\ &= K_A\,, \end{aligned}$$

de modo que $P_A(K^\uparrow) \subseteq K^\uparrow_A$. Similarmente, temos $P_B(K^\uparrow) \subseteq K^\uparrow_B$ e $P^c_{AB}(K^\uparrow) \subseteq L_m(\mathbf{G}^c_{AB})$. Portanto,

$$\begin{aligned} K^{\uparrow} &\subseteq P_A^{-1}\big(P_A(K^{\uparrow})\big) \cap P_B^{-1}\big(P_B(K^{\uparrow})\big) \cap (P_{AB}^c)^{-1}\big(P_{AB}^c(K^{\uparrow})\big) \\ &\subseteq P_A^{-1}(K_A^{\uparrow}) \cap P_B^{-1}(K_B^{\uparrow}) \cap (P_{AB}^c)^{-1}\big(L_m(\mathbf{G}_{AB}^c)\big) \\ &= K_A^{\uparrow} \,||\, K_B^{\uparrow} \,||\, L_m(\mathbf{G}_{AB}^c)\,. \end{aligned}$$

□

Para que seja verificada a inclusão reversa e assim obtido o comportamento global desejado, algumas condições são necessárias. O seguinte lema sugere quais são essas condições, e será de vital importância para a demonstração do teorema que o segue.

Lema 2. *Se sup$\mathcal{C}(K_A)$, sup$\mathcal{C}(K_B)$ e $L_m(\mathbf{G}_{AB}^c)$ forem não conflitantes, e se sup$\mathcal{C}(K_A)$ e sup$\mathcal{C}(K_B)$ forem juntamente coercivas com respeito a* $\mathbf{G}$*, então sup$\mathcal{C}(K_A) \,||\, sup\mathcal{C}(K_B) \,||\, L_m(\mathbf{G}_{AB}^c)$ é controlável com respeito a* $\mathbf{G}$*.*

Demonstração. Denotando $K_A^{\uparrow}$ e $K_B^{\uparrow}$ como no Teorema 2, verificamos as condições de controlabilidade. Para (c_1), veja que de

$$s \in \overline{K_A^{\uparrow} \,||\, K_B^{\uparrow} \,||\, L_m(\mathbf{G}_{AB}^c)} \;\;\&\;\; \sigma \in \Sigma_{nc} \;\;\&\;\; s\sigma \in L(\mathbf{G}) \tag{4.9}$$

devemos mostrar que

$$s\sigma \in \overline{K_A^{\uparrow} \,||\, K_B^{\uparrow} \,||\, L_m(\mathbf{G}_{AB}^c)} = \overline{K_A^{\uparrow}} \,||\, \overline{K_B^{\uparrow}} \,||\, \overline{L_m(\mathbf{G}_{AB}^c)}\,,$$

a igualdade sendo consequência da hipótese de não conflito. Como supõe-se que $\mathbf{G}_{AB}^c$ seja não bloqueante, $s\sigma \in L(\mathbf{G})$ significa que

$$P_{AB}^c(s\sigma) \in L(\mathbf{G}_{AB}^c) = \overline{L_m(\mathbf{G}_{AB}^c)}\,,$$

portanto é suficiente mostrar que $P_A(s\sigma) \in \overline{K_A^{\uparrow}}$ e $P_B(s\sigma) \in \overline{K_B^{\uparrow}}$. Existem quatro possibilidades a partir de (4.9):

i. $\sigma \in (\Sigma_A \cap \Sigma_B)_{nc}$:

$$\begin{aligned} &P_{\Psi}(s) \in \overline{K_{\Psi}^{\uparrow}} \;\;\&\;\; \sigma \in \Sigma_{\Psi_{nc}} \;\;\&\;\; P_{\Psi}(s\sigma) \in L(\mathbf{G}_{\Psi}),\; \forall\, \Psi \in \{A, B\} \\ &\Rightarrow P_{\Psi}(s\sigma) \in \overline{K_{\Psi}^{\uparrow}},\;\; \forall\; \Psi \in \{A, B\}\,, \end{aligned}$$

já que $K_A^{\uparrow}$ e $K_B^{\uparrow}$ são controláveis.

ii. $\sigma \in (\Sigma_A - \Sigma_B)_{nc}$:

$$P_A(s) \in \overline{K_A^\uparrow} \;\&\; \sigma \in \Sigma_{A_{nc}} \;\&\; P_A(s\sigma) \in L(\mathbf{G}_A) \;\Rightarrow\; P_A(s\sigma) \in \overline{K_A^\uparrow}\,.$$

Ademais, temos $P_B(s\sigma) = P_B(s) \in \overline{K_B^\uparrow}$.

iii. $\sigma \in (\Sigma_B - \Sigma_A)_{nc}$:
Segue por analogia direta com (ii).

iv. $\sigma \in (\Sigma_{AB}^c)_{nc}$:
Temos que $P_\Psi(s\sigma) = P_\Psi(s) \in \overline{K_\Psi^\uparrow},\ \forall\ \Psi \in \{A, B\}$.

Isso prova que (c_1) é satisfeita.

Para (c_2), note primeiramente que

$$\begin{aligned} &\Sigma_{K_A^\uparrow || K_B^\uparrow || L_m(\mathbf{G}_{AB}^c)}(s) \cap \Sigma_{\text{for}} = \emptyset \\ \Rightarrow\; &\Sigma_{K_A^\uparrow || K_B^\uparrow}\big(P_{AB}(s)\big) \cap (\Sigma_A \cup \Sigma_B)_{\text{for}} = \emptyset\,. \end{aligned} \tag{4.10}$$

Assim,

$$\begin{aligned} & s \in \overline{K_A^\uparrow \,||\, K_B^\uparrow \,||\, L_m(\mathbf{G}_{AB}^c)} \;\;\&\;\; s\,tick \in L(\mathbf{G}) \\ &\quad \&\;\; \Sigma_{K_A^\uparrow || K_B^\uparrow || L_m(\mathbf{G}_{AB}^c)}(s) \cap \Sigma_{\text{for}} = \emptyset \\ \overset{(*)}{\Rightarrow}\; & P_{AB}(s) \in \overline{K_A^\uparrow \,||\, K_B^\uparrow} \;\;\&\;\; P_{AB}(s\,tick) \in L(\mathbf{G}_{AB}) \\ &\quad \&\;\; \Sigma_{K_A^\uparrow || K_B^\uparrow}\big(P_{AB}(s)\big) \cap (\Sigma_A \cup \Sigma_B)_{\text{for}} = \emptyset \\ &\quad \&\;\; P_{AB}^c(s\,tick) \in L(\mathbf{G}_{AB}^c) = \overline{L_m(\mathbf{G}_{AB}^c)} \\ \overset{(**)}{\Rightarrow}\; & P_{AB}(s\,tick) \in \overline{K_A^\uparrow \,||\, K_B^\uparrow} \;\;\&\;\; P_{AB}^c(s\,tick) \in \overline{L_m(\mathbf{G}_{AB}^c)} \\ \Rightarrow\; & s\,tick \in \overline{K_A^\uparrow \,||\, K_B^\uparrow \,||\, L_m(\mathbf{G}_{AB}^c)}\,, \end{aligned}$$

com $(*)$ seguindo de (4.10) e $(**)$ da coercividade conjunta de $K_A^\uparrow$ e $K_B^\uparrow$. Finalmente, como $K_A^\uparrow$, $K_B^\uparrow$ e $L_m(\mathbf{G}_{AB}^c)$ são não conflitantes e

$$\overline{K_A^\uparrow \,||\, K_B^\uparrow} \,||\, \overline{L_m(\mathbf{G}_{AB}^c)} \subseteq \overline{K_A^\uparrow} \,||\, \overline{K_B^\uparrow} \,||\, \overline{L_m(\mathbf{G}_{AB}^c)}\,,$$

temos $s\,tick \in \overline{K_A^\uparrow \,||\, K_B^\uparrow \,||\, L_m(\mathbf{G}_{AB}^c)}$, como desejado. □

Apresentamos, agora, o principal resultado deste trabalho.

Teorema 3. *Sejam $\mathcal{S}_A$ e $\mathcal{S}_B$ supervisores não bloqueantes tais que $L_m(\mathcal{S}_A/\mathbf{G}_A) = sup\mathcal{C}(K_A)$ e $L_m(\mathcal{S}_B/\mathbf{G}_B) = sup\mathcal{C}(K_B)$. Então, $\mathcal{S}_A \wedge \mathcal{S}_B$ é uma solução ótima para o PCSML se e somente se $L_m(\mathcal{S}_A/\mathbf{G}_A)$, $L_m(\mathcal{S}_B/\mathbf{G}_B)$ e $L_m(\mathbf{G}^c_{AB})$ forem não conflitantes, e $L(\mathcal{S}_A/\mathbf{G}_A)$ e $L(\mathcal{S}_B/\mathbf{G}_B)$ forem juntamente coercivas com respeito a $\mathbf{G}_{AB}$.*

Demonstração. (se) Do Teorema 1 sabemos que, sob as condições colocadas, $\mathcal{S}_A \wedge \mathcal{S}_B$ é um supervisor não bloqueante para $\mathbf{G}$. Consequentemente, o que resta mostrarmos é que

$$L_m(\mathcal{S}_A \wedge \mathcal{S}_B/\mathbf{G}) = sup\mathcal{C}\big(E_a \,||\, E_b \,||\, L_m(\mathbf{G})\big)\,. \tag{4.11}$$

Novamente, denotaremos $K^{\uparrow}$, $K^{\uparrow}_A$ e $K^{\uparrow}_B$ como no Teorema 2. Veja que

$$\begin{aligned} & L_m(\mathcal{S}_A \wedge \mathcal{S}_B/\mathbf{G}) \\ (*) \quad & = L_m(\mathcal{S}_A \wedge \mathcal{S}_B/\mathbf{G}_{AB}) \,||\, L_m(\mathbf{G}^c_{AB}) \\ (**) \quad & = L_m(\mathcal{S}_A/\mathbf{G}_A) \,||\, L_m(\mathcal{S}_B/\mathbf{G}_B) \,||\, L_m(\mathbf{G}^c_{AB}) \\ & = K^{\uparrow}_A \,||\, K^{\uparrow}_B \,||\, L_m(\mathbf{G}^c_{AB})\,, \end{aligned} \tag{4.12}$$

onde $(*)$ vem da Observação 6 e $(**)$ da Proposição 2. Agora, do Lema 2 temos que $K^{\uparrow}_A \,||\, K^{\uparrow}_B \,||\, L_m(\mathbf{G}^c_{AB})$ é controlável com respeito a $\mathbf{G}$. Além disso,

$$\begin{aligned} & K^{\uparrow}_A \,||\, K^{\uparrow}_B \,||\, L_m(\mathbf{G}^c_{AB}) \\ & = P_A^{-1}(K^{\uparrow}_A) \cap P_B^{-1}(K^{\uparrow}_B) \cap (P^c_{AB})^{-1}\big(L_m(\mathbf{G}^c_{AB})\big) \\ & \subseteq P_A^{-1}(K_A) \cap P_B^{-1}(K_B) \cap (P^c_{AB})^{-1}\big(L_m(\mathbf{G}^c_{AB})\big) \\ & = K\,, \end{aligned}$$

de modo que $K^{\uparrow}_A \,||\, K^{\uparrow}_B \,||\, L_m(\mathbf{G}^c_{AB}) \subseteq K^{\uparrow}$. Isso, juntamente com o Teorema 2, mostra que $K^{\uparrow}_A \,||\, K^{\uparrow}_B \,||\, L_m(\mathbf{G}^c_{AB}) = K^{\uparrow}$, que por (4.12) leva a (4.11).

(somente se) Suponha que $\mathcal{S}_A \wedge \mathcal{S}_B$ seja uma solução para o PCSML. Em particular, isso significa que é um supervisor não bloqueante para $\mathbf{G}$ e, portanto, de acordo com a Proposição 3, também é um supervisor para $\mathbf{G}_{AB}$. O resultado segue, então, diretamente do Teorema 1. □

Pelo Teorema 3 fica estabelecido que, dados dois supervisores locais ótimos, as condições do Teorema 1 são necessárias e suficientes para garantir que a sua conjunção seja um supervisor não bloqueante que impõe sobre o sistema o comportamento especificado de forma minimamente restritiva e, assim, define uma solução ótima para o problema

de controle apresentado. Esse resultado sedimenta a metodologia de controle modular local para SED temporizados. É interessante notar que as condições aqui colocadas são as mesmas da abordagem modular (BRANDIN; WONHAM, 1993), apenas generalizadas para o caso da supervisão local.

Observação 7. Os resultados desta seção, assim como os da Seção 4.4, podem ser facilmente generalizados para o caso de múltiplas especificações. Isso pode ser feito por indução natural sobre o número de especificações, tomando como base da indução o caso de duas especificações aqui demonstrado. Essa mesma estratégia foi utilizada para o controle modular local não temporizado (QUEIROZ, 2000).

4.6 SINOPSE DA METODOLOGIA DE CONTROLE

Para a aplicação da metodologia de controle proposta neste capítulo, podem ser seguidos os passos abaixo:

1. obtenha uma representação por sistema produto temporizado para a planta, i. e., modele cada subsistema como um autômato de estados finitos de maneira que os alfabetos de eventos regulares (diferentes de *tick*) sejam disjuntos dois a dois;

2. modele cada especificação como um autômato, incluindo apenas os eventos cuja ocorrência se queira de alguma forma restringir;

3. construa as plantas locais relacionadas a cada especificação, compondo os subsistemas que compartilham eventos com ela;

4. obtenha o comportamento local desejado compondo as plantas locais com as especificações correspondentes;

5. calcule a máxima sublinguagem controlável de cada comportamento local desejado;

6. verifique o não conflito e a coercividade conjunta das linguagens resultantes, conforme o Teorema 3;

   (a) caso ambas as condições sejam satisfeitas, projete supervisores não bloqueantes que imponham as máximas sublinguagens controláveis sobre as respectivas plantas locais; sua ação conjunta resultará num comportamento global ótimo;

   (b) caso qualquer das condições seja violada, resolva o conflito conforme a Seção 4.9.

No passo 6(a), os próprios autômatos não bloqueantes que modelam as máximas sublinguagens controláveis obtidas no passo 5 podem ser inicialmente tomados como implementações para os supervisores locais, estando claramente de acordo com a definição dada na Observação 5 (Seção 2.4). Contudo, na maior parte dos casos é possível representar a mesma ação de controle utilizando autômatos mais simples, já que os modelos acima sugeridos incorporam, além das restrições que levam ao cumprimento das especificações, todas aquelas inerentes ao comportamento original das próprias plantas locais. Em (SU; WONHAM, 2004), é mostrado que o problema de encontrar um autômato *mínimo* cuja ação de controle seja equivalente à de um supervisor dado é NP-complexo. No entanto, o mesmo trabalho propõe um algoritmo para redução de supervisores com complexidade polinomial no número de estados da planta e do supervisor a ser reduzido; embora evidentemente não se possa garantir que o modelo resultante seja mínimo, em muitos casos a simplificação é bastante significativa.

4.7 EXEMPLO ILUSTRATIVO — SOLUÇÃO

Para ilustrar os resultados apresentados, resolveremos agora o problema introduzido na Seção 4.2. O passo seguinte é obter a máxima sublinguagem controlável dos comportamentos locais desejados. Pode ser verificado que K_A é controlável com respeito a $\mathbf{G}_A$, de modo que $K_A^{\uparrow} = sup\mathcal{C}(K_A) = K_A$, mas o mesmo não se verifica para K_B. De fato, K_B tenta desabilitar o evento não controlável β após a primeira ocorrência de *tick*. Assim sendo, buscamos calcular $K_B^{\uparrow} = sup\mathcal{C}(K_B)$. A única restrição "ilegal" feita por K_B é aquela mencionada acima, portanto deve-se garantir que o estado no qual β deveria ser proibido jamais seja alcançado. Para tal, a medida menos restritiva é fazer com que β ocorra imediatamente assim que o sistema for inicializado, forçando-o a preemptar *tick* no estado inicial. A linguagem $K_B^{\uparrow}$, resultante dessa alteração, é representada na Figura 18.

Neste exemplo simples, é possível verificar por inspeção que $K_A^{\uparrow}$ e $K_B^{\uparrow}$ são juntamente coercivas com respeito a $\mathbf{G}_{AB} = \mathbf{G}_1 \,||\, \mathbf{G}_2$ e que $K_A^{\uparrow}$, $K_B^{\uparrow}$ e $L_m(\mathbf{G}_{AB}^c)$ são não conflitantes. Logo, do Teorema 3 sabemos que se supervisores locais não bloqueantes $\mathcal{S}_A$ e $\mathcal{S}_B$ forem projetados de maneira que $L_m(\mathcal{S}_A/\mathbf{G}_A) = K_A^{\uparrow}$ e $L_m(\mathcal{S}_B/\mathbf{G}_B) = K_B^{\uparrow}$, sua conjunção levará o sistema a um comportamento global ótimo e não bloqueante. Tal comportamento é mostrado na Figura 19. Pode-se observar que as restrições impostas pelas especificações de controle não são violadas.

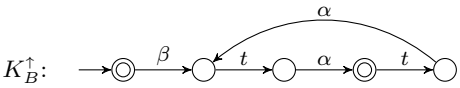

Figura 18 – Máxima sublinguagem controlável $K_B^{\uparrow}$ para K_B da Figura 17.

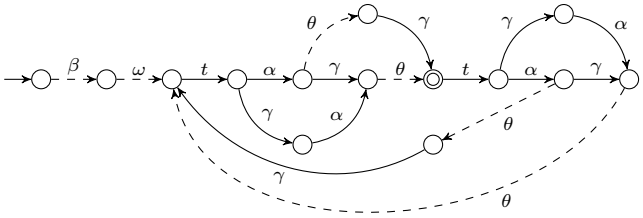

Figura 19 – Comportamento não bloqueante ótimo $L_m(\mathcal{S}_A \wedge \mathcal{S}_B/\mathbf{G})$.

A exigência de que α seja sempre imediatamente seguido por θ, por exemplo, proibe apenas a ocorrência de *tick* entre esses dois eventos, e por conseguinte não é descumprida pela possibilidade de ocorrer a sequência $\alpha \to \gamma \to \theta$.

4.8 ANÁLISE DE COMPLEXIDADE

Conforme mencionado no Capítulo 1, a complexidade computacional é um dos principais fatores limitantes para a aplicação da teoria de controle supervisório. Esta seção é dedicada a avaliar o desempenho da metodologia proposta neste trabalho em comparação às abordagens monolítica e modular clássica.

Como em (RUDIE, 1988), a notação $O(\cdot)$ é utilizada conforme a definição de (HOROWITZ; SAHNI, 1983), qual seja: para um dado parâmetro n, a complexidade da função $f(n)$ é dada por $O\big(g(n)\big)$ se existem $N, M \in \mathbb{N}$ tais que $\forall n > N,\ |f(n)| \leq M|g(n)|$. Se $f(n)$ representa o tempo requerido para a execução de um algoritmo em função do parâmetro n, intuitivamente a definição significa que, a partir de um certo valor de n, f não crescerá mais rápido que g. Essa notação fornece, portanto, um limite superior para a velocidade de crescimento de um algoritmo. Aqui, assim como em (WONHAM; RAMADGE, 1988) e (QUEIROZ, 2000), consideraremos uma análise de pior caso e tomaremos como parâmetro apenas o número de estados dos modelos.

De acordo com os resultados de (WONHAM; RAMADGE, 1988),

dados o modelo de um sistema (com x estados) e um autômato que reconhece uma linguagem representando o comportamento desejado (y estados), o cálculo da máxima sublinguagem controlável tem complexidade $O(x^2y^2)$. Já o teste de não conflito para duas linguagens representadas por autômatos de z estados é de complexidade $O(z^2)$.

Considere um sistema produto temporizado composto por k subsistemas, cada qual com n estados. Sejam duas especificações modeladas por autômatos com m estados, cada uma afetando l subsistemas. Por simplicidade, suponha que todos os subsistemas da planta sejam afetados por pelo menos uma das especificações, de modo que $k/2 \leq l \leq k$.

Para a síntese monolítica, o comportamento desejado global é obtido pela composição de todos os subsistemas da planta e ambas as especificações, o que resulta em um modelo com $n^k m^2$ estados. O modelo da planta possui n^k estados. Então, a complexidade do cálculo da máxima sublinguagem controlável global é de ordem $O(n^{4k}m^4)$.

No caso modular clássico, os comportamentos desejados são obtidos pela composição separada de cada especificação com todos os subsistemas da planta, gerando dois modelos com $n^k m$ estados. O cálculo de cada uma das máximas sublinguagens controláveis tem complexidade $O(n^{4k}m^2)$, enquanto a verificação de não conflito e de coercividade conjunta é de ordem $O(n^{2k}m^2)$. Portanto, a síntese modular tem uma complexidade total de $O(n^{4k}m^2)$.

Para o caso modular local, são construídas duas plantas locais com n^l estados. Os comportamentos desejados locais são, então, obtidos pela composição de cada especificação com a respectiva planta local, resultando em dois modelos de tamanho $n^l m$. Para calcular as máximas sublinguagens controláveis, a complexidade é, portanto, $O(n^{4l}m^2)$, e para o teste de não conflito e coercividade conjunta é $O(n^{2l}m^2)$. Logo, a complexidade geral da síntese modular local é dada por $O(n^{4l}m^2)$.

Um primeiro fato a ser notado é que, como $l \leq k$, a abordagem modular local tem complexidade computacional menor ou igual à abordagem modular clássica, e estritamente menor que a monolítica para $m > 1$.

Além disso, de acordo com (QUEIROZ, 2000), em um caso mais geral com q especificações de m estados, a complexidade da síntese modular clássica é $O(n^{4k}m^2 + n^{kq}m^q)$, enquanto a da modular local é $O(n^{4l}m^2 + n^{lq}m^q)$. Os segundos termos dessas expressões originam-se do teste de não conflito, o qual passa a determinar a complexidade de ambas as abordagens para valores altos de q. Apesar de o crescimento ser exponencial em ambos os casos, é importante observar que o fa-

Tabela 2 – Complexidade computacional das abordagens analisadas.

Abordagem	Complexidade	
	2 especificações	q especificações
Monolítica	$O(n^{4k}m^4)$	$O(n^{4k}m^{2q})$
Modular	$O(n^{4k}m^2)$	$O(n^{4k}m^2 + n^{kq}m^q)$
Modular Local	$O(n^{4l}m^2)$	$O(n^{4l}m^2 + n^{lq}m^q)$

tor l, ao contrário de k, não cresce necessariamente com o número de componentes do sistema, o que pode trazer um ganho computacional relevante. Finalmente, para esse caso geral a complexidade do controle monolítico é $O(n^{4k}m^{2q})$, sendo fácil ver que, se $l < 4k/q$, a abordagem modular local leva a uma redução significativa na complexidade. Em um problema no qual $k = 10$ e $q = 8$, por exemplo, isso significaria termos $l < 5$, ou seja, cada especificação afetando no máximo quatro dos dez subsistemas, o que em muitos casos é uma condição bastante razoável.

As expressões referentes à complexidade de cada abordagem aparecem organizadas na Tabela 2.

4.9 RESOLUÇÃO DE CONFLITOS

Um dos desafios enfrentados na aplicação do controle modular provém da dificuldade em prever o surgimento de conflitos. No caso temporizado, o mesmo pode ser dito com respeito à coercividade conjunta. Salvo em casos excepcionalmente simples, não se consegue detectar de antemão se essas duas propriedades serão satisfeitas para dois ou mais supervisores, sendo possível a verificação apenas mediante a conclusão do processo de síntese. Ao mesmo tempo, no Teorema 3 mostramos que ambas são indispensáveis para que a abordagem modular local leve à solução ótima do PCSML. Por isso, para que não se corra o risco de que o esforço despendido na síntese dos supervisores locais seja desperdiçado e que a resolução do problema em questão retorne à "estaca zero", é importante que eventuais conflitos possam ser resolvidos, ou seja, que o comportamento global desejado (não bloqueante e minimamente restritivo) possa ser alcançado *a posteriori* a partir de um resultado que não respeite as duas condições.

Existem na literatura diferentes abordagens para a resolução de

conflitos no contexto não temporizado (WONG; WONHAM, 1998; CHEN; LAFORTUNE; LIN, 2000; WONG et al., 2000). Contudo, nossa pesquisa durante o desenvolvimento desta dissertação não encontrou propostas de tais métodos para SED temporizados. Uma alternativa óbvia é a realização de uma síntese monolítica sobre o sistema em malha fechada resultante, utilizando a componente trim do modelo obtido como comportamento desejado. O sucesso dessa estratégia depende, no entanto, da condição de que o comportamento ótimo esteja contido na linguagem global obtida. Isso significa que é necessário que a ação conjunta dos supervisores locais ótimos não seja mais restritiva que a de um supervisor monolítico ótimo para as mesmas especificações. O Teorema 2 assevera que essa condição é satisfeita para o controle modular local temporizado, de modo que, pela notação desse teorema, temos

$$sup\,\mathcal{C}\big(K_A^{\uparrow} \,||\, K_B^{\uparrow} \,||\, L_m(\mathbf{G}_{AB}^{c})\big) = sup\,\mathcal{C}\big(E_a \,||\, E_b \,||\, L_m(\mathbf{G})\big)\,.$$

Uma vez obtidos os supervisores locais $\mathcal{S}_A$ e $\mathcal{S}_B$, em caso de conflito é possível fazer o cálculo de $sup\,\mathcal{C}\big(K_A^{\uparrow} \,||\, K_B^{\uparrow} \,||\, L_m(\mathbf{G}_{AB}^{c})\big)$ considerando a planta em malha fechada $\mathbf{G}_{\text{conf}} = \mathcal{S}_A \wedge \mathcal{S}_B/\mathbf{G}$, ao invés do modelo original em malha aberta $\mathbf{G}$. Essa alternativa apresenta a vantagem de que, como parte considerável do comportamento gerado pelo sistema terá sido eliminado pela ação dos supervisores $\mathcal{S}_A$ e $\mathcal{S}_B$, um supervisor $\mathcal{S}_{\text{conf}}$ projetado para resolver o conflito sobre $\mathbf{G}_{\text{conf}}$, ainda que monolítico, exercerá uma ação de controle relativamente simples, ao contrário de um supervisor monolítico calculado para impor todas as especificações sobre $\mathbf{G}$. Essa simplicidade refletirá na possibilidade de representação através de um modelo mais compacto, facilitando sua compreensão e manipulação.

Porém, no caso temporizado o uso de eventos forçáveis pode fazer com que a linguagem $sup\,\mathcal{C}\big(K_A^{\uparrow} \,||\, K_B^{\uparrow} \,||\, L_m(\mathbf{G}_{AB}^{c})\big)$, calculada sobre

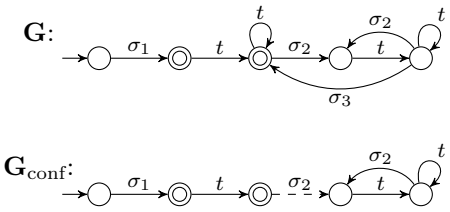

Figura 20 – Exemplo simples de uma planta: em malha aberta ($\mathbf{G}$) e bloqueante em malha fechada ($\mathbf{G}_{\text{conf}}$).

$\mathbf{G}_{\text{conf}}$, não seja controlável com respeito a $\mathbf{G}$. Para visualizar como isso ocorre, imaginemos um simples exemplo no qual $\mathbf{G}$ e $\mathbf{G}_{\text{conf}}$ são dados como na Figura 20, em que σ_2 e σ_3 são eventos controláveis e σ_2 é forçável. É fácil ver que o comportamento minimamente restritivo que resolve o conflito e é controlável com respeito a $\mathbf{G}_{\text{conf}}$ é dado pela linguagem marcada $\{\sigma_1, \sigma_1 t\}$, a qual claramente é não controlável com respeito a $\mathbf{G}$. Assim, para que a técnica de resolução de conflito acima exposta leve, de fato, o sistema ao comportamento global ótimo, é necessário testar a controlabilidade de $sup\mathcal{C}\big(K_A^{\uparrow} \,||\, K_B^{\uparrow} \,||\, L_m(\mathbf{G}_{AB}^c)\big)$ com respeito a $\mathbf{G}$. Se ela for verificada, a ação de $\mathcal{S}_{\text{conf}}$, juntamente com os supervisores locais, levará o sistema ao comportamento desejado, ou seja, teremos

$$L_m\big(\mathcal{S}_{\text{conf}} \,/\, (\mathcal{S}_A \wedge \mathcal{S}_B/\mathbf{G})\big) = sup\mathcal{C}\big(K_A^{\uparrow} \,||\, K_B^{\uparrow} \,||\, L_m(\mathbf{G}_{AB}^c)\big)\,.$$

Caso contrário, resta a alternativa de resolver o conflito através de uma síntese monolítica sobre $\mathbf{G}$.

4.10 EXEMPLO ILUSTRATIVO — RESOLUÇÃO DE CONFLITO

Suponha que a especificação E_b do exemplo das Seções 4.2 e 4.7 seja alterada para uma nova versão $\widetilde{E}_b$, a qual dita que se *tick* ocorrer duas vezes sem nenhuma ocorrência de β, então, e somente então, λ deve ocorrer, e na sequência β é sempre proibido. Essa especificação, juntamente com a máxima sublinguagem controlável $\widetilde{K}_B^{\uparrow}$ do novo comportamento desejado $\widetilde{K}_B = \widetilde{E}_b \,||\, L_m(\mathbf{G}_B)$, são mostradas na Figura 21. Seja $\widetilde{\mathcal{S}}_B$ um supervisor não bloqueante tal que $L_m(\widetilde{\mathcal{S}}_B/\mathbf{G}_B) = \widetilde{K}_B^{\uparrow}$.

Embora $K_A^{\uparrow}$ e $\widetilde{K}_B^{\uparrow}$ sejam novamente juntamente coercivas, as linguagens $K_A^{\uparrow}$, $\widetilde{K}_B^{\uparrow}$ e $L_m(\mathbf{G}_{AB}^c)$ são agora conflitantes, o que significa que $\mathcal{S}_A \wedge \widetilde{\mathcal{S}}_B/\mathbf{G}$ é bloqueante. Esse fato fica claro ao observarmos o comportamento global resultante (Figura 22). Podemos ver que $K_A^{\uparrow} \,||\, \widetilde{K}_B^{\uparrow} \,||\, L_m(\mathbf{G}_{AB}^c) = L_m(\mathcal{S}_A \wedge \widetilde{\mathcal{S}}_B/\mathbf{G})$ não é controlável com respeito a $\mathbf{G}$, pois o evento $\beta \in \Sigma_{nc}$ teria que ser desabilitado após as cadeias t e $t\gamma$. Como curiosidade, neste caso pode ser verificado que $\mathcal{S}_A \wedge \widetilde{\mathcal{S}}_B/\mathbf{G}_{AB}$ é não bloqueante, sendo que o conflito surge da interação com $\mathbf{G}_{AB}^c$.

Mas o Teorema 2 garante que o comportamento assim obtido jamais pode ser mais restritivo que o comportamento global ótimo para as mesmas especificações, o qual, de acordo com a discussão da Seção 4.9, pode ser obtido calculando-se $sup\mathcal{C}\big(K_A^{\uparrow} \,||\, \widetilde{K}_B^{\uparrow} \,||\, L_m(\mathbf{G}_{AB}^c)\big)$ sobre $\mathcal{S}_A \wedge \widetilde{\mathcal{S}}_B/\mathbf{G}$. A forma minimamente restritiva de obter uma su-

blinguagem controlável de $L_m(\mathcal{S}_A \wedge \widetilde{\mathcal{S}}_B/\mathbf{G})$ é proibir a ocorrência das cadeias t e $t\gamma$, o que pode ser feito, em termos de ação de controle, forçando β a preemptar *tick* no estado inicial. Então, um supervisor $\mathcal{S}_{\text{conf}}$ projetado para impor essa restrição, ao atuar juntamente com $\mathcal{S}_A$ e $\widetilde{\mathcal{S}}_B$, leva o sistema ao comportamento ótimo

$$L_m\big(\mathcal{S}_{\text{conf}} \,/\, (\mathcal{S}_A \wedge \widetilde{\mathcal{S}}_B/\mathbf{G})\big) = sup\mathcal{C}\big(K_A^{\uparrow} \,||\, \widetilde{K}_B^{\uparrow} \,||\, L_m(\mathbf{G}_{AB}^{c})\big)\,,$$

o qual, neste caso, é controlável com respeito a $\mathbf{G}$ e coincide com aquele obtido anteriormente na Seção 4.7 (Figura 19).

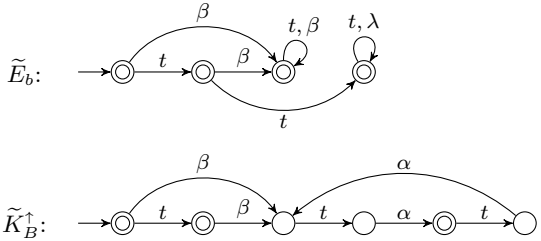

Figura 21 – Nova especificação $\widetilde{E}_b$ e máxima sublinguagem controlável $\widetilde{K}_B^{\uparrow}$ do novo comportamento desejado $\widetilde{K}_B = \widetilde{E}_b \,||\, L_m(\mathbf{G}_B)$.

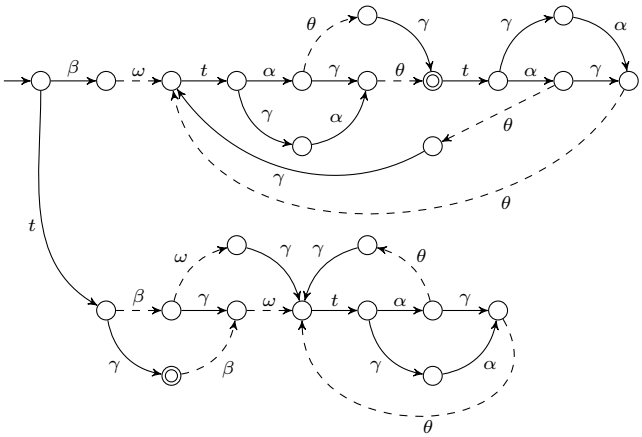

Figura 22 – Comportamento global bloqueante de $\mathcal{S}_A \wedge \widetilde{\mathcal{S}}_B/\mathbf{G}$.

5 EXEMPLO DE APLICAÇÃO

Uma das principais motivações da abordagem desenvolvida nesta dissertação é possibilitar a aplicação do controle supervisório a sistemas encontrados na prática, cujos modelos comumente apresentam um número elevado de estados e em muitos dos quais o cumprimento de restrições temporais é um fator determinante. Neste capítulo, visamos apresentar a solução de um problema envolvendo um sistema que exibe essas características e que se encontra bastante presente no meio industrial moderno. Trata-se do que é conhecido como *cluster tool*.

Para as operações computacionais sobre os modelos, utilizaremos a ferramenta TTCT[1].

5.1 DESCRIÇÃO DO SISTEMA

Cluster tools são ferramentas utilizadas em diversos processos industriais, tendo se proliferado durante as últimas décadas na manufatura de semicondutores (WOOD; SARASWAT, 1991). São responsáveis pelo processamento dos chamados *wafers*, e consistem na integração, em um ambiente fechado e isolado, de um robô transportador e uma série de módulos de processamento (MP). Além disso, possuem um ou mais compartimentos (*loadlocks*) por onde entram e saem os *wafers* antes e depois de serem processados. Tanto o robô quanto os MP possuem capacidade para apenas um *wafer*. Embora existam *cluster tools* com arquitetura linear, a mais comum é a radial, como aquela mostrada na Figura 23.

Após ser apanhado pelo robô no compartimento de entrada, cada *wafer* é transportado e carregado em um dos MP, descarregado após o processamento, transportado entre diferentes MP (de acordo com uma sequência de produção predeterminada), e finalmente retornado ao compartimento inicial. A ausência de *buffers* entre os módulos complexifica a coordenação e sincronização das tarefas desempenhadas por eles e pelo robô. Pode haver diferenças significativas entre os tempos de processamento de cada MP, o que pode tornar vantajoso o uso de módulos redundantes, i.e., dois ou mais deles operando em paralelo e desempenhando a mesma função. Evita-se, com isso, a presença de gargalos no sistema. Para uma descrição mais detalhada, o leitor pode recorrer a (KIM; LEE, 2008; LEE, 2008).

[1] Disponível online em `http://www.control.toronto.edu/DES`.

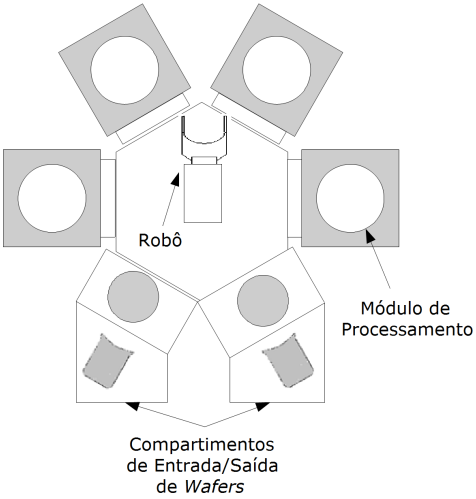

Figura 23 – *Cluster tool* com arquitetura radial, contendo quatro MP e dois compartimentos de entrada/saída. Adaptado de (WU et al., 2008).

Um fator crítico que motiva o emprego de modelos temporizados para a solução de problemas envolvendo *cluster tools*, conforme mencionado em (KIM; LEE, 2008), é a presença de calor e gases na atmosfera interior dos MP. Assim que um *wafer* é carregado a um MP e adentra esse ambiente, é imediatamente exposto a tais intempéries, e qualquer demora no início da operação pode levar a alterações na sua estrutura, comprometendo a qualidade do produto final. Por consequência, é imprescindível que o processamento de um *wafer* por um MP inicie imediatamente após a sua entrada.

No presente exemplo, trataremos de uma ferramenta com arquitetura radial e com três módulos de processamento que chamaremos de M_j, $j \in \{1, 2, 3\}$, sendo que M_2 e M_3 são redundantes. Os *wafers* devem ser processados primeiramente por M_1 e em seguida por M_2 ou M_3. A planta é formada, portanto, por esses três módulos juntamente com um robô. Consideraremos, ainda, que existe apenas um compartimento de entrada e saída de *wafers*. Por simplicidade, assumimos que sempre existem wafers disponíveis na entrada e que sempre há espaço na saída. Assim, como não há restrições entre o robô e o compartimento de entrada/saída, não é necessário incluir este último no modelo. Os GTA para o robô ($\mathbf{R}$) e para os módulos ($\mathbf{M}_j$) são mostrados na Figura 24. Em $\mathbf{R}$, o estado inicial representa o robô vazio parado na

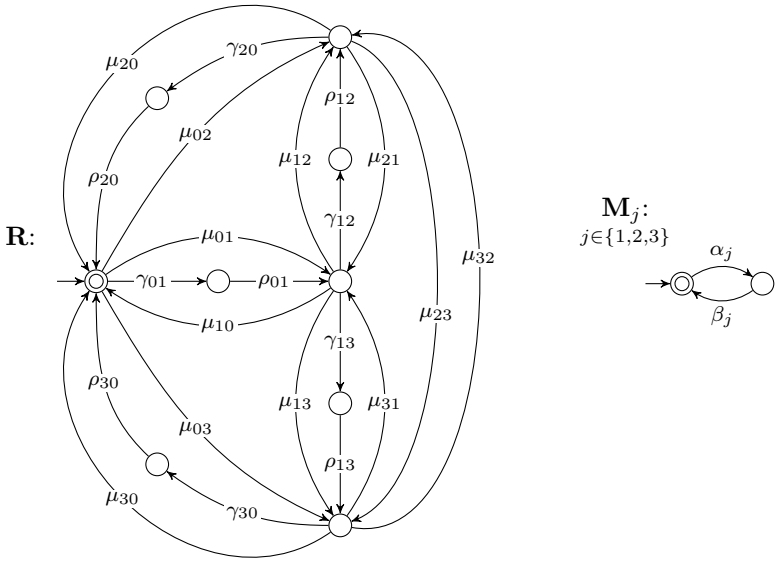

Figura 24 – GTA para o robô ($\mathbf{R}$) e para os módulos ($\mathbf{M}_j$, $j \in \{1, 2, 3\}$).

Tabela 3 – Limites temporais para os eventos dos GTA da Figura 24.

Evento	Limites Temporais
α_j	$[0, \infty)$
β_1	$[9, 9]$
β_2, β_3	$[13, 14]$
μ_{kl}	$[1, \infty)$
γ_{kl}	$[0, \infty)$
ρ_{01}	$[3, 4]$
$\rho_{12}, \rho_{13}, \rho_{20}, \rho_{30}$	$[2, 2]$

posição do compartimento de entrada/saída. A posição ocupada pelo robô ao parar no módulo M_j será chamada de *estação* j, sendo que a estação 0 corresponde à posição inicial. Denominaremos, além disso, de *operação de transporte da estação* k *para a estação* l o ato de retirar um *wafer* da primeira, levá-lo até a segunda e lá descarregá-lo. A interpretação dos eventos da planta se dá como segue:

α_j : início de operação do módulo M_j ;

β_j : fim de operação do módulo M_j ;

μ_{kl} : movimento do robô vazio da estação k para a estação l ;

γ_{kl} : início da operação de transporte da estação k para a estação l ;

ρ_{kl} : fim da operação de transporte da estação k para a estação l .

Consideraremos que todos os eventos remotos (α_j, μ_{kl} e γ_{kl}) são controláveis e forçáveis, e que nenhum dos eventos esperados (β_j e ρ_{kl}) é forçável.

Em um primeiro momento, pode-se ter a impressão de que o sistema descrito é de pequeno porte, uma vez que a composição dos GTA da Figura 24 resulta em um modelo com $9 \times 2^3 = 72$ estados. No entanto, conforme destacado em (CURY; MARTINEZ; QUEIROZ, 2013), uma dificuldade inerente ao modelo Brandin-Wonham é o crescimento vertiginoso do número de estados dos GTT quando se trata de sistemas que possuem operações com tempos de duração altamente discrepantes. No caso de *cluster tools*, essa discrepância aparece porque os movimentos do robô entre as estações têm duração muito menor que as operações dos módulos de processamento. Na Tabela 3 são listados os valores dos limites temporais para os eventos da planta, baseados em (WU et al., 2008). Na realidade, os valores aqui retratados são adaptados para facilitar a manipulação dos modelos; em *cluster tools* industriais, a divergência encontrada é usualmente ainda maior. Obtém-se, com esses valores, um GTT para $\mathbf{R}$ com 25 estados e 48 transições, para $\mathbf{M}_1$ com 11 estados e 12 transições, e para $\mathbf{M}_2$ e $\mathbf{M}_3$ cada um com 16 estados e 18 transições. O GTT global, resultante da composição desses quatro GTT, possui 70400 estados e 163048 transições, ou seja, quase mil vezes o número de estados do GTA correspondente. Essa explosão deixa clara a necessidade de técnicas computacionalmente eficientes de controle para esse tipo de sistema.

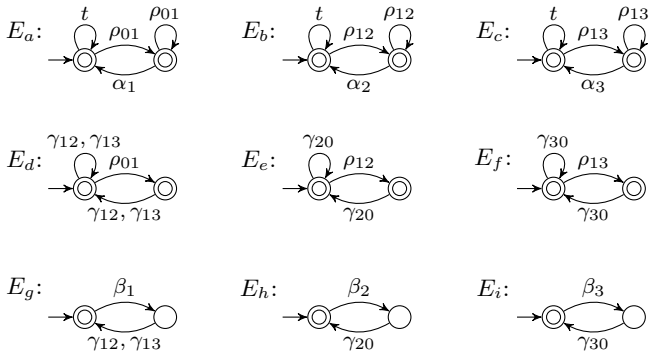

Figura 25 – Especificações para a planta da Figura 24.

5.2 ESPECIFICAÇÕES DE CONTROLE

Os objetivos gerais a serem alcançados na solução do problema de controle aqui abordado podem ser sintetizados como: (i) sincronizar as operações dos diferentes componentes da planta (robô e módulos), de modo a respeitar suas limitações de capacidade e a levar ao correto sequenciamento das etapas de processamento de cada *wafer*; e (ii) garantir que as restrições temporais do processo, resultantes das suas características físicas e químicas, sejam respeitadas. A seguir, detalharemos as especificações que devem ser impostas para o cumprimento dessas metas.

Naturalmente, uma primeira restrição desejável é a de que nenhum MP inicie sua operação sem ter recebido um *wafer* do robô, o que pode ser visto como evitar o *underflow* dos módulos M_j. Além disso, levaremos em conta o fator crítico referente ao início de operação, mencionado na Seção 5.1. Dizer que o processamento de um *wafer* por um módulo M_j deve iniciar imediatamente após a sua entrada significa, em termos dos eventos do sistema, que o início da operação (α_j) deve ocorrer antes do próximo *tick* do relógio. As especificações E_a, E_b e E_c, representadas por autômatos na Figura 25, traduzem as restrições acima em termos dos eventos da planta. A primeira delas, por exemplo, impõe que, após o robô realizar o transporte de um *wafer* da entrada até M_1 (evento ρ_{01}), esse módulo deve iniciar o processamento (α_1) antes da ocorrência do próximo *tick*.

É necessário, também, respeitar a capacidade dos MP, evitando seu *overflow*. Para tal, uma vez que um *wafer* seja levado até um MP,

não deve ser permitida a chegada de um novo *wafer* até que o primeiro tenha sido processado e retirado. Isso é imposto através das especificações E_d, E_e e E_f (Figura 25). Ao mesmo tempo, o robô jamais deve retirar um *wafer* de um MP antes que o seu processamento tenha sido concluído, levando a E_g, E_h e E_i na mesma figura. Essas especificações, em combinação com as três primeiras, também definem o roteamento dos *wafers* entre os módulos M_1, M_2 e M_3, exigindo, através da marcação, que todo *wafer* que entre no sistema seja devidamente processado e levado até a saída.

5.3 SOLUÇÃO DO PROBLEMA

Seguindo os passos da metodologia apresentada na Seção 4.6, constroem-se as plantas locais compondo os subsistemas afetados por cada especificação, conforme mostrado na Tabela 4. Compondo as especificações com as plantas locais correspondentes, são obtidos os comportamentos locais desejados e então calculadas as suas máximas sublinguagens controláveis. Verifica-se que essas são, de fato, não conflitantes e juntamente coercivas, ou seja, as condições do Teorema 3 são satisfeitas. Portanto, são finalmente obtidos supervisores para impor cada um dos comportamentos ótimos locais, os quais, de acordo com os comentários da Seção 4.6, podem ser inicialmente tomados como os próprios autômatos que representam esses comportamentos. Através do algoritmo de redução de supervisores proposto em (SU; WONHAM, 2004) e implementado na ferramenta TTCT, é possível obter modelos reduzidos para esses supervisores, sem com isso alterar sua ação de controle. Sua ação conjunta garante, pelos resultados do Capítulo 4, que o sistema se comporte de maneira não bloqueante e cumpra as especificações de forma minimamente restritiva. Os resultados são mostrados na Tabela 5, onde os valores entre parênteses representam (nº de estados, nº de transições) para o respectivo modelo e $K_\Phi^\uparrow = sup\mathcal{C}(K_\Phi)$.

Vale lembrar que, para a solução do mesmo problema através da abordagem modular clássica, no lugar das plantas locais da Tabela 4 seria utilizado o modelo global da planta para a síntese de cada supervisor. Ao tentarmos aplicar essa metodologia com o propósito de comparação, para a especificação E_a encontra-se uma máxima sublinguagem controlável com 54272 estados e 123447 transições. A ferramenta TTCT foi incapaz de computar um supervisor reduzido para este caso, por falta de memória. Portanto, vê-se que os supervisores obtidos pela abordagem modular local podem ser implementados por

Tabela 4 – Plantas locais referentes às especificações da Figura 25.

Especificações	Planta Local	Dados (estados, transições)
E_a, E_g	$\mathbf{R} \,\|\|\, \mathbf{M}_1$	$(275, 558)$
E_b, E_h	$\mathbf{R} \,\|\|\, \mathbf{M}_2$	$(400, 823)$
E_c, E_i	$\mathbf{R} \,\|\|\, \mathbf{M}_3$	$(400, 823)$
E_d, E_e, E_f	$\mathbf{R}$	$(25, 48)$

Tabela 5 – Dados dos supervisores locais obtidos.

Comportamentos Desejados	Máximas Sublinguagens Controláveis	Supervisores Reduzidos
K_A $(252, 471)$	$K_A^{\uparrow}$ $(212, 421)$	$\mathcal{S}_A$ $(11, 122)$
K_B $(379, 731)$	$K_B^{\uparrow}$ $(324, 663)$	$\mathcal{S}_B$ $(26, 259)$
K_C $(379, 731)$	$K_C^{\uparrow}$ $(324, 663)$	$\mathcal{S}_C$ $(26, 259)$
K_D $(44, 88)$	$K_D^{\uparrow}$ $(39, 82)$	$\mathcal{S}_D$ $(2, 42)$
K_E $(47, 92)$	$K_E^{\uparrow}$ $(44, 88)$	$\mathcal{S}_E$ $(2, 43)$
K_F $(47, 92)$	$K_F^{\uparrow}$ $(44, 88)$	$\mathcal{S}_F$ $(2, 43)$
K_G $(502, 1003)$	$K_G^{\uparrow}$ $(448, 905)$	$\mathcal{S}_G$ $(11, 218)$
K_H $(761, 1527)$	$K_H^{\uparrow}$ $(677, 1371)$	$\mathcal{S}_H$ $(15, 310)$
K_I $(761, 1527)$	$K_I^{\uparrow}$ $(677, 1371)$	$\mathcal{S}_I$ $(15, 310)$

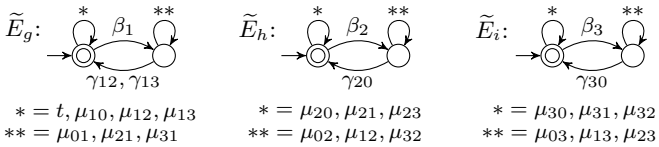

Figura 26 – Novas especificações para o sistema.

autômatos relativamente simples, tornando mais fácil a interpretação das ações de controle bem como sua modificação e atualização.

5.4 ESPECIFICAÇÕES ADICIONAIS

Além das especificações colocadas na Seção 5.2, alguns processos de fabricação possuem estritas restrições quanto ao atraso na retirada de um *wafer* após o seu processamento por um dado MP. Nesses casos, o robô deve apanhar o *wafer* daquele módulo dentro de um intervalo limitado de tempo; caso contrário, a superfície do *wafer* pode sofrer sérios problemas de qualidade por permanecer em contato com o calor e com os gases residuais dentro da câmara de processameto do MP (KIM; LEE, 2008). Aqui, consideraremos que cada *wafer* processado por M_1 deve ser retirado imediatamente, ou seja, nenhum *tick* pode ocorrer após β_1 antes da sua retirada.

Como uma última medida, buscamos também minimizar a movimentação do robô, reduzindo desgastes e custos de energia supérfluos. Nas especificações da Figura 25, nenhuma restrição é feita quanto aos eventos μ_{kl}, que representam o movimento do robô vazio. Logo, o comportamento resultante da ação conjunta dos supervisores obtidos permite que o robô transite vazio entre as estações sem que esteja, com isso, desempenhando nenhuma função útil. A única situação em que tais movimentos devem ocorrer é quando o robô tiver o objetivo de buscar um *wafer* que esteja pronto para ser transportado de uma estação a outra, seja para retirá-lo do compartimento de entrada e levá-lo até M_1, para retirá-lo de M_1 e levá-lo até M_2 ou M_3, ou para retirá-lo de um desses dois módulos e entregá-lo novamente à estação inicial.

As duas restrições acima descritas podem ser traduzidas em especificações de controle através da modificação de E_g, E_h e E_i. Suas novas versões, chamadas de $\widetilde{E}_g$, $\widetilde{E}_h$ e $\widetilde{E}_i$, são mostradas na Figura 26.

5.5 NOVA SOLUÇÃO

As plantas locais para as novas especificações são as mesmas das suas versões anteriores. Os dados dos novos comportamentos desejados, máximas sublinguagens controláveis e autômatos que implementam os supervisores reduzidos são mostrados na Tabela 6.

Verifica-se que as linguagens $K_A^{\uparrow}$ a $K_F^{\uparrow}$, juntamente com $\widetilde{K}_G^{\uparrow}$, $\widetilde{K}_H^{\uparrow}$ e $\widetilde{K}_I^{\uparrow}$, são agora conflitantes, implicando que a ação conjunta de supervisores que imponham localmente esses comportamentos será bloqueante no nível global. Pela composição de todas as plantas locais em malha fechada, obtemos o modelo global conflitante

$$\mathbf{G}_{\text{conf}} = \mathcal{S}_A \wedge \mathcal{S}_B \wedge \mathcal{S}_C \wedge \mathcal{S}_D \wedge \mathcal{S}_E \wedge \mathcal{S}_F \wedge \widetilde{\mathcal{S}}_G \wedge \widetilde{\mathcal{S}}_H \wedge \widetilde{\mathcal{S}}_I / \mathbf{R} || \mathbf{M}_1 || \mathbf{M}_2 || \mathbf{M}_3 \,,$$

que possui 1742 estados e 2869 transições e cuja linguagem marcada é dada por

$$L_m(\mathbf{G}_{\text{conf}}) = K_A^{\uparrow} \,||\, K_B^{\uparrow} \,||\, K_C^{\uparrow} \,||\, K_D^{\uparrow} \,||\, K_E^{\uparrow} \,||\, K_F^{\uparrow} \,||\, \widetilde{K}_G^{\uparrow} \,||\, \widetilde{K}_H^{\uparrow} \,||\, \widetilde{K}_I^{\uparrow} \,.$$

Para resolver o conflito, obtém-se a máxima sublinguagem controlável $sup\mathcal{C}\big(L_m(\mathbf{G}_{\text{conf}})\big)$, calculada com respeito a $\mathbf{G}_{\text{conf}}$, que resulta em 1643 estados e 2642 transições. Essa linguagem, então, é verificada como sendo controlável com respeito à planta gobal em malha aberta $\mathbf{R} \,||\, \mathbf{M}_1 \,||\, \mathbf{M}_2 \,||\, \mathbf{M}_3$. Assim sendo, de acordo com a Seção 4.9 é implementado um supervisor $\mathcal{S}_{\text{conf}}$ que resolve o conflito sobre $\mathbf{G}_{\text{conf}}$, cujo modelo reduzido apresenta 32 estados e 230 transições. Observa-se que mesmo sendo um supervisor global, no sentido de que é computado considerando todos os eventos da planta, sua ação pode ser sintetizada em um modelo de tamanho bastante reduzido.

Em contraste, obtendo um supervisor monolítico para as mesmas especificações a partir da planta global em malha aberta, consegue-se para ele um modelo reduzido de 262 estados e 950 transições. Isso significa que, mesmo que uma solução deste problema através da abordagem monolítica seja possível, o supervisor resultante é representado por um modelo complexo e difícil de ser compreendido, implementado ou modificado.

Tabela 6 – Dados dos novos supervisores locais.

Comportamentos Desejados	Máximas Sublinguagens Controláveis	Supervisores Reduzidos
$\widetilde{K}_G$ $(265, 496)$	$\widetilde{K}_G^{\uparrow}$ $(214, 396)$	$\widetilde{\mathcal{S}}_G$ $(13, 179)$
$\widetilde{K}_H$ $(761, 1431)$	$\widetilde{K}_H^{\uparrow}$ $(677, 1284)$	$\widetilde{\mathcal{S}}_H$ $(15, 268)$
$\widetilde{K}_I$ $(761, 1431)$	$\widetilde{K}_I^{\uparrow}$ $(677, 1284)$	$\widetilde{\mathcal{S}}_I$ $(15, 268)$

6 CONCLUSÕES

A metodologia de controle supervisório para sistemas a eventos discretos temporizados proposta nesta dissertação permite a síntese local de supervisores modulares, considerando, em cada módulo de controle, apenas os componentes do sistema diretamente afetados pela respectiva especificação. Os resultados demonstrados estabelecem condições necessárias e suficientes para garantir que a ação conjunta dos supervisores locais confira ao sistema um comportamento global não bloqueante e minimamente restritivo com relação ao especificado.

A principal contribuição da abordagem apresentada é a redução no custo computacional da síntese de supervisores para sistemas temporizados complexos, em comparação a técnicas já existentes. Ademais, como os supervisores são projetados levando em conta apenas informações locais, sua ação de controle tende a ser relativamente simples. Isso permite que eles sejam representados por autômatos de pequeno porte, o que facilita sua compreensão, implementação e modificação. Em caso de alterações na planta, por exemplo, apenas os supervisores cuja ação se baseia na parte modificada precisam ser reprojetados, garantindo maior flexibilidade.

Com base na análise de complexidade realizada na Seção 4.8, pode-se concluir que a abordagem modular local soluciona o problema da explosão de estados no que concerne ao cálculo do comportamento minimamente restritivo para cada módulo de controle. Isso porque a complexidade para a solução de cada problema local cresce com o número de estados dos subsistemas afetados pela especificação correspondente, o qual, por sua vez, não tende a crescer com o aumento do sistema global. No entanto, o teste da condição de não conflito, aspecto mais custoso da síntese modular, permanece como um potencial obstáculo para a aplicação da teoria de controle supervisório a sistemas reais. Para o caso não temporizado, encontram-se na literatura propostas no sentido de aumentar a eficiência nesse quesito, a exemplo de (PENA; CURY; LAFORTUNE, 2009). Entretanto, no contexto temporizado essa questão fica em aberto, bem como a de resolução de conflitos (Seção 4.9), ambas servindo como sugestões para futuros trabalhos na área.

Uma primeira versão dos resultados deste trabalho foi publicada em (SCHAFASCHEK; QUEIROZ; CURY, 2014), e a versão apresentada nesta dissertação foi submetida para publicação na revista *IEEE Transactions on Automatic Control*.

REFERÊNCIAS

BRANDIN, B. A.; WONHAM, W. M. Modular supervisory control of timed discrete-event systems. In: *Proc. 32nd IEEE Conference on Decision and Control*. San Antonio, TX: IEEE, 1993. p. 2230–2235.

BRANDIN, B. A.; WONHAM, W. M. Supervisory control of timed discrete-event systems. *IEEE Transactions on Automatic Control*, v. 39, n. 2, p. 329–342, 1994.

CAI, K.; WONHAM, W. M. Supervisor localization: a top-down approach to distributed control of discrete-event systems. *IEEE Transactions on Automatic Control*, v. 55, n. 3, p. 605–618, 2010.

CASSANDRAS, C. G.; LAFORTUNE, S. *Introduction to Discrete Event Systems*. 2. ed. New York: Springer, 2010.

CHEN, Y.-L.; LAFORTUNE, S.; LIN, F. Design of nonblocking modular supervisors using event priority functions. *IEEE Transactions on Automatic Control*, v. 45, n. 3, p. 432–452, 2000.

CURY, J. E. R.; MARTINEZ, C.; QUEIROZ, M. H. de. Scheduling cluster tools with supervisory control theory. In: *Proc. 11th IFAC Workshop on Intelligent Manufacturing Systems*. São Paulo, Brasil: IFAC, 2013. p. 312–317.

GOHARI, P.; WONHAM, W. M. Reduced supervisors for timed discrete-event systems. *IEEE Transactions on Automatic Control*, v. 48, n. 7, p. 1187–1198, 2003.

HOPCROFT, J. E.; MOTWANI, R.; ULLMAN, J. D. *Introduction to Automata Theory, Languages, and Computation*. 3. ed. Boston, MA: Pearson/Addison Wesley, 2007.

HOROWITZ, E.; SAHNI, S. *Fundamentals of Data Structures*. Rockville, MD: Computer Science Press, 1983.

KIM, J.-H.; LEE, T.-E. Schedulability analysis of time-constrained cluster tools with bounded time variation by an extended petri net. *IEEE Transactions on Automation Science and Engineering*, v. 5, n. 3, p. 490–503, 2008.

LEE, T.-E. A review of scheduling theory and methods for semiconductor manufacturing cluster tools. In: *Proc. 2008 Winter Simulation Conference*. Austin, TX: IEEE, 2008. p. 2127–2135.

LIN, F.; WONHAM, W. M. Decentralized supervisory control of discrete-event systems. *Information Science*, v. 44, n. 3, p. 199–224, 1988.

LIN, F.; WONHAM, W. M. Decentralized supervisory control and coordination of discrete-event systems with partial observation. *IEEE Transactions on Automatic Control*, v. 35, n. 12, p. 1330–1337, 1990.

LIN, F.; WONHAM, W. M. Verification of nonblocking in decentralized supervision. *Control – Theory and Advanced Technology*, v. 7, n. 1, p. 19–29, 1991.

LIN, F.; WONHAM, W. M. Supervisory control of timed discrete-event systems under partial observation. *IEEE Transactions on Automatic Control*, v. 40, n. 3, p. 558–562, 1995.

PENA, P. N.; CURY, J. E. R.; LAFORTUNE, S. Verification of nonconflict of supervisors using abstractions. *IEEE Transactions on Automatic Control*, v. 54, n. 12, p. 2803–2815, 2009.

QUEIROZ, M. H. de. *Controle Supervisório Modular de Sistemas de Grande Porte*. Dissertação (Mestrado) — Universidade Federal de Santa Catarina, Florianópolis, Brazil, 2000.

QUEIROZ, M. H. de; CURY, J. E. R. Modular supervisory control of large scale discrete-event systems. In: *Proc. 5th International Workshop on Discrete Event Systems (WODES'2000)*. Ghent, Belgium: Kluwer Academic Publishers, 2000. p. 103–110.

QUEIROZ, M. H. de; CURY, J. E. R. Controle supervisório modular de sistemas de manufatura. *Sociedade Brasileira de Automática: Controle & Automação*, v. 13, n. 2, p. 123–133, 2002.

RAMADGE, P. J. Some tractable supervisory control problems for discrete-event systems modeled by Büchi automata. *IEEE Transactions on Automatic Control*, v. 34, n. 1, p. 10–19, 1989.

RAMADGE, P. J.; WONHAM, W. M. Supervisory control of a class of discrete-event processes. *SIAM Journal on Control and Optimization*, v. 25, n. 1, p. 206–230, 1987.

RAMADGE, P. J.; WONHAM, W. M. The control of discrete event systems. *Proc. IEEE, Special Issue on Discrete Event Dynamic Systems*, v. 77, n. 1, p. 81–98, 1989.

RUDIE, K. *Software for the control of discrete-event systems: a complexity study*. Dissertação (Mestrado) — University of Toronto, Toronto, Canada, 1988.

RUDIE, K.; WILLEMS, J. C. The computational complexity of decentralized discrete-event control problems. *IEEE Transactions on Automatic Control*, v. 40, n. 7, p. 1313–1319, 1995.

SAADATPOOR, A.; WONHAM, W. M. State based control of timed discrete event systems using binary decision diagrams. *Systems and Control Letters*, v. 56, p. 62–74, 2007.

SCHAFASCHEK, G.; QUEIROZ, M. H. de; CURY, J. E. R. Local modular supervisory control of timed discrete-event systems. In: *Proc. 12th International Workshop on Discrete Event Systems (WODES'14)*. Paris: IFAC, 2014. p. 271–277.

SU, R.; WONHAM, W. M. Supervisor reduction for discrete-event systems. *Discrete Event Dynamic Systems*, v. 14, n. 1, p. 31–53, 2004.

WONG, K. C. On the complexity of projections of discrete-event systems. In: *Proc. 4th International Workshop on Discrete Event Systems (WODES'98)*. Cagliari, Italy: IEE Publisher, 1998. p. 201–206.

WONG, K. C. et al. Supervisory control of distributed systems: Conflict resolution. *Discrete Event Dynamic Systems*, v. 10, n. 1–2, p. 131–186, 2000.

WONG, K. C.; WONHAM, W. M. Modular control and coordination of discrete-event systems. *Discrete Event Dynamic Systems*, v. 8, n. 3, p. 247–297, 1998.

WONHAM, W. M. *Supervisory Control of Discrete-Event Systems*. Toronto, Canada: Systems and Control Group, Dept. of Electrical and Computer Eng., University of Toronto, 2013.

WONHAM, W. M.; RAMADGE, P. J. Modular supervisory control of discrete-event systems. *Maths. of Control, Signals and Systems*, v. 1, n. 1, p. 13–30, 1988.

WOOD, S. C.; SARASWAT, K. C. Modeling the performance of cluster-based fabs. In: *Proc. IEEE/SEMI International Semiconductor Manufacturing Science Symposium*. Burlingame, CA: IEEE, 1991. p. 8–14.

WU, N. et al. Petri net modeling and real-time control of dual-arm cluster tools with residency time constraint and activity time variations. In: *Proc. 4th IEEE Conference on Automation Science and Engineering*. Washington DC: IEEE, 2008. p. 109–114.

ZHANG, R. et al. Supervision localization of timed discrete-event systems. *Automatica*, v. 49, n. 9, p. 2786–2794, 2013.