MAX HERING DE QUEIROZ

CONTROLE SUPERVISÓRIO MODULAR DE SISTEMAS DE GRANDE PORTE

FLORIANÓPOLIS
2000

UNIVERSIDADE FEDERAL DE SANTA CATARINA

PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

CONTROLE SUPERVISÓRIO MODULAR DE SISTEMAS DE GRANDE PORTE

Dissertação submetida à
Universidade Federal de Santa Catarina
como parte dos requisitos para a
obtenção do grau de Mestre em Engenharia Elétrica.

MAX HERING DE QUEIROZ

Florianópolis, Março de 2000.

CONTROLE SUPERVISÓRIO MODULAR DE SISTEMAS DE GRANDE PORTE

Max Hering de Queiroz

'Esta Dissertação foi julgada adequada para obtenção do Título de Mestre em Engenharia Elétrica, Área de Concentração em *Controle e Automação*, e aprovada em sua forma final pelo Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal de Santa Catarina.'

Prof. José Eduardo Ribeiro Cury, Dr. d'Etat
Orientador

Prof. Ildemar Cassana Decker, D. Sc.
Coordenador do Programa de Pós-Graduação em Engenharia Elétrica

Banca Examinadora:

Prof. José Eduardo Ribeiro Cury, Dr. d'Etat
Presidente

Prof. Jean-Marie Farines, Dr. Ing.

Prof. Guilherme Bittencourt, Dr. Ing.

Prof. Roberto Mário Ziller, M. Eng.

"Enquanto dois peixes, dois pássaros,

dois homens,

fizerem do sonho

o sonho de vários peixes, vários pássaros,

vários homens

permanecerá vivo o sonho de solidariedade

entre os homens."

Lindolf Bell (1938 – 1998)

AGRADECIMENTOS

Gostaria de agradecer sinceramente a todos aqueles que direta ou indiretamente contribuíram para a elaboração desta pesquisa. De maneira particular, expresso minha gratidão:

- ao professor José Eduardo Ribeiro Cury, pela dedicação, competência e amizade que marcaram o seu papel de orientador durante todo o desenvolvimento desta pesquisa;

- aos Professores, Funcionários e Colegas do LCMI que fazem deste laboratório um ambiente de pesquisa próspero, cooperativo e amigável;

- ao contribuinte brasileiro e à CAPES, pelo apoio financeiro;

- aos revisores, dentre os quais se destaca meu Pai, pela disposição em corrigir a redação deste trabalho;

- aos meus Familiares e Amigos, pelo amor e amizade que me sustentam;

- aos meus queridos Pais e Irmãos que sempre apoiaram e incentivaram a conquista de meus sonhos;

- a Deus, pelo Dom da Vida.

Resumo da Dissertação apresentada à UFSC como parte dos requisitos necessários para a obtenção do grau de Mestre em Engenharia Elétrica.

CONTROLE SUPERVISÓRIO MODULAR DE SISTEMAS DE GRANDE PORTE

Max Hering de Queiroz

Março/2000

Orientador: Prof. José Eduardo Ribeiro Cury, Dr. d'Etat.
Área de Concentração: Controle e Automação.
Palavras-chave: controle supervisório; sistemas a eventos discretos; controle modular; sistemas de grande porte; sistemas de manufatura.
Número de Páginas: 61.

A presente pesquisa contribui para a teoria de controle de sistemas a eventos discretos com a proposta de uma abordagem mais eficiente para a síntese de supervisores modulares para sistemas de grande porte. A abordagem desenvolvida baseia-se na modelagem da dinâmica do sistema em malha aberta (planta) e da especificação do comportamento desejado por linguagens controláveis. O modelo de Ramadge e Wonham é utilizado, então, para propiciar um processo automático de síntese de controladores minimamente restritivos. Entretanto, a modelagem de sistemas por linguagens controláveis gera um crescimento exponencial do número de estados do modelo com a agregação de subsistemas. Isso pode inviabilizar a construção de supervisores para sistemas mais complexos, que caracteristicamente envolvem múltiplas especificações sobre um grande número de subsistemas concorrentes. Para resolver esse problema, o presente trabalho faz um refinamento do controle modular clássico explorando aspectos de modularidade da planta e das especificações, no sentido de agregar o mínimo necessário de subsistemas em cada módulo de controle. Como resultado, obtém-se uma estrutura de controle naturalmente descentralizada em que cada módulo supervisiona apenas os subsistemas diretamente afetados pela respectiva especificação. Com base nessa abordagem, propõe-se ainda uma nova metodologia para a síntese de controladores para sistemas de manufatura. Essa metodologia é elucidada pelo cálculo de supervisores para um exemplo de sistema integrado de manufatura.

Abstract of Dissertation presented to UFSC as a partial fulfillment of the requirements for the degree of Master in Electrical Engineering.

MODULAR SUPERVISORY CONTROL OF LARGE SCALE SYSTEMS

Max Hering de Queiroz

March /2000

Advisor: Prof. José Eduardo Ribeiro Cury, Dr. d'Etat.
Area of Concentration: Control and Automation.
Keywords: discrete event systems; distributed control; modeling; manufacturing automation.
Number of Pages: 61.

The current research contributes to the theory of control of discrete-event systems by proposing a more efficient approach to the synthesis of modular supervisors for large scale systems. In the developed approach, dynamics of the open loop system (plant) and of the desired behavior specification are modeled by controllable languages. The model of Ramadge and Wonham is used then to allow an automatic process of synthesis of minimally restrictive controllers. However, the modeling by controllable languages generates an exponential increase in the number of states of the model with the aggregation of subsystems. This issue can make the supervisors' synthesis unfeasible for more complex systems, which characteristically involve multiple specifications over a large number of concurrent subsystems. To solve this problem, the current work refines the classical modular control by exploiting aspects of modularity of plant and specifications, aiming to incorporate the least necessary number of subsystems in each control module. As a result, a naturally decentralized control structure, in which each module supervises only the subsystems directly affected by the respective specification, is obtained. Taking into account that approach, a new methodology to synthesize controllers for manufacturing systems is still proposed. This methodology is elucidated by computing supervisors for an example of integrated manufacturing system.

ÍNDICE

1. INTRODUÇÃO

A complexa organização da sociedade moderna e a crescente escassez de recursos naturais têm justificado um enorme esforço de otimização e automação dos sistemas criados pelo homem. Nesse contexto, a teoria de controle de sistemas a eventos discretos se destaca pela sua larga aplicação aos sistemas modernos e pelo seu potencial para desenvolvimento de ferramentas teóricas.

Um sistema a eventos discretos (SED) é um sistema dinâmico cuja mudança de estado ocorre em pontos discretos do tempo, em decorrência de eventos isolados. O domínio de SED's abrange uma série de sistemas modernos entre os quais se incluem sistemas de manufatura, protocolos de comunicação, sistemas automatizados de tráfego, sistemas de gerenciamento de base de dados e sistemas computacionais. São sistemas dirigidos a eventos, em geral complexos e que não têm um tratamento adequado pelas abordagens clássicas de equações diferenciais e a diferenças.

O estudo de SED's baseia-se num modelo matemático que descreve o sistema real. Esse modelo proporciona, além de uma melhor compreensão do sistema, o uso de técnicas formais de análise. Na literatura há várias abordagens distintas para a modelagem de SED's, dentre as quais se destacam Linguagens Controláveis [20], Redes de Petri [16], Redes de Petri Controladas [13], Cadeias de Markov [5] e Teoria das Filas [12]. Uma apresentação resumida desses modelos pode ser encontrada em [28].

A Teoria de Controle de SED's proposta por Ramadge e Wonham [20] baseia-se na modelagem da dinâmica do sistema em malha aberta (planta) e na especificação do comportamento desejado por linguagens controláveis. O supervisor tem a função de exercer uma ação de controle restritiva sobre a planta, de modo a confinar seu comportamento ao especificado.

Diferentemente dos outros modelos que, em sua maioria, limitam-se à análise de sistemas, esta abordagem tem a vantagem de permitir a síntese automática de controladores de forma mais independente da experiência e inspiração do projetista. Além disso, quando a especificação não pode ser cumprida, tem-se a garantia de construir um supervisor que satisfaz parte do comportamento desejado de forma minimamente restritiva.

Embora os cálculos para a maioria das soluções de controle exijam esforço polinomial no número de estados do sistema, a modelagem de sistemas por linguagens controláveis gera um crescimento exponencial do número de estados com a agregação de subsistemas. Para sistemas mais complexos, a síntese de controladores pode, então, ser inviabilizada pela explosão de estados, o que é um forte incentivo para refinar a abordagem de Ramadge e Wonham.

O presente trabalho tem o objetivo de contribuir para a Teoria de Controle de Sistemas a Eventos Discretos, no sentido de reduzir a complexidade computacional em problemas de síntese de supervisores para sistemas de grande porte. Para isso, duas estratégias sistêmicas universais têm sido utilizadas. A primeira explora regularidades internas da estrutura algébrica ou aritmética do sistema. Este é o caso da abordagem baseada em aspectos de simetria proposta por Eyzell e Cury [9, 10] e da abordagem de BDD [1, 3]. A segunda estratégia baseia-se na criação de arquiteturas adequadas. A arquitetura de controle hierárquico [27] permite a síntese de controladores baseada em diferentes níveis de abstração do sistema. Outras arquiteturas exploram a modularidade no nível da planta (controle descentralizado) e das especificações (controle modular). Pelo controle descentralizado[15, 21 , 22], a natureza modular da planta impõe restrições na estrutura de informações disponíveis, de modo que é necessária a definição de um grupo de supervisores, cada qual observando e controlando apenas uma parte do processo, para a realização de uma tarefa global.

Na abordagem de controle modular [26], ao invés de se projetar um único supervisor que satisfaça todas as especificações, procura-se construir um controlador para cada especificação, de forma que, atuando em conjunto, os supervisores satisfaçam todas as especificações. Neste caso, deseja-se que os supervisores resultantes sejam modulares, isto é, que a ação conjunta dos supervisores modulares tenha o mesmo desempenho que a do supervisor monolítico. Quando essa propriedade é verificada, a abordagem de controle modular é bastante vantajosa no sentido de promover maior flexibilidade, maior eficiência computacional e segurança na aplicação do controle.

Entretanto, as abordagens clássicas de controle modular e centralizado requerem a construção de um modelo único para o funcionamento global do sistema por linguagens controláveis, o que pode ser inviável no projeto de sistemas de grande porte. Por outro

lado, grande parte dos sistemas de grande porte pode ser representada pela composição de diversos subsistemas menores, normalmente modelando operações concorrentes. Este é o caso do modelo estruturado chamado Sistema Produto [17, 18, 20].

Este trabalho faz um refinamento do controle modular clássico explorando a estrutura descentralizada de Sistemas Produto, no sentido de agregar o mínimo necessário de subsistemas em cada módulo de controle. Como resultado, é apresentada uma nova abordagem que possibilita a síntese de múltiplos supervisores modulares evitando a explosão de estados no processo de modelagem e construção de controladores. Com base nessa abordagem, propõe-se ainda uma metodologia para a síntese de supervisores para sistemas de manufatura.

Esta dissertação é estruturada da seguinte forma: o Capítulo 2 introduz de forma sintética os conceitos da Teoria de Controle de Sistemas a Eventos Discretos baseada em Linguagens Controláveis. Também nele são demonstradas algumas propriedades de linguagens que fundamentam o desenvolvimento matemático dos capítulos seguintes.

O Capítulo 3 apresenta os principais resultados da abordagem clássica de controle modular proposta por Wonham e Ramadge. Inclui, também, uma extensão desses resultados para problemas envolvendo mais de duas especificações.

No Capítulo 4, propõe-se uma nova abordagem para a modelagem de sistemas de grande porte por linguagens controláveis, baseada na noção de Sistema Produto. Tirando proveito dessa modelagem, como principal contribuição do presente trabalho, faz-se o refinamento da teoria de controle modular de forma a evitar a explosão de estados na síntese de supervisores de sistemas de grande porte. Enfim, estes resultados são generalizados para a síntese de múltiplos controladores.

Baseado nos resultados do Capítulo 4, o Capítulo 5 propõe uma metodologia para a síntese automática de controladores modulares para sistemas discretos de manufatura. Essa metodologia é ilustrada pelo cálculo de supervisores para um modelo hipotético de sistema integrado de manufatura.

Finalmente, apresentam-se conclusões da pesquisa, bem como sugestões para trabalhos futuros.

2. CONTROLE DE SISTEMAS A EVENTOS DISCRETOS

Este capítulo apresenta a teoria de linguagens controláveis introduzida por Ramadge e Wonham para o controle de sistemas a eventos discretos. Esta apresentação não é exaustiva, mas introduz os principais conceitos sobre os quais este documento se baseia, bem como as propriedades que são fundamentais para o desenvolvimento de seus resultados.

É importante ressaltar que a representação matemática apresentada a seguir não admite a ocorrência simultânea de dois ou mais eventos distintos, de forma que os sistemas tratados apresentam evolução seqüencial. Além disso, a modelagem apresentada não considera os instantes de tempo em que os eventos ocorrem, mas apenas a ordem em que acontecem. Em contrapartida, a abordagem de Ramadge e Wonham pode ser estendida para suportar um modelo temporizado [2].

Na primeira seção é apresentada uma abordagem para a representação matemática de sistemas a eventos discretos, que pode ser aprofundada em [11]. A seção seguinte mostra os principais resultados da estrutura de controle monolítico proposta por Ramadge e Wonham. Um estudo mais detalhado sobre este assunto pode ser feito a partir de [14, 20, 25].

2.1 Teoria de Linguagens

A mudança de estados de um SED acontece em pontos discretos do tempo pela ocorrência de eventos específicos. São exemplos de eventos o início e o término de uma tarefa e a percepção de uma mudança de estado em um sensor. Na Teoria de Linguagens, o comportamento de um SED é modelado por um conjunto de cadeias de símbolos representando todas as possíveis seqüências de eventos admitidas pelo sistema.

2.1.1 Linguagens

Seja Σ o conjunto finito de etiquetas de eventos que ocorrem num SED a controlar, também chamado de *planta*. Seja Σ^* o conjunto de todas as cadeias finitas de elementos do

conjunto Σ, incluindo a cadeia vazia ε. Representa-se o comprimento de uma cadeia $s \in \Sigma^*$ por $|s|$. A cadeia $u \in \Sigma^*$ é um prefixo de $v \in \Sigma^*$ se, para alguma cadeia $s \in \Sigma^*$, $v = us$.

Uma linguagem sobre o alfabeto Σ é um subconjunto de Σ^*. Assim, o comportamento de um SED de alfabeto Σ pode ser modelado por uma linguagem $L \subseteq \Sigma^*$, representando todas as cadeias finitas de eventos que o SED pode gerar, ou seja, que são fisicamente realizáveis.

O prefixo fechamento, ou simplesmente fechamento, de L é dado por $\overline{L} = \{u: \exists v \in \Sigma^* \wedge uv \in L\}$. Diz-se, então, que uma linguagem é prefixo fechada sempre que $L = \overline{L}$.

2.1.2 *Autômatos*

Se o comportamento de um SED pode ser descrito por uma linguagem regular [11], então há um dispositivo chamado *autômato* capaz de gerar esta mesma linguagem de acordo com regras bem definidas. Formalmente, um autômato de estados finitos é uma quíntupla $(\Sigma, X, \delta, x_0, X_m)$, onde Σ é um alfabeto de eventos, X é um conjunto de estados, $x_0 \in X$ é o estado inicial, $X_m \subseteq X$ é o conjunto de estados marcados (ou estados finais) e $\delta: \Sigma \times X \rightarrow X$ é a função de transição de estados.

Os autômatos podem ser ilustrados por *diagramas de transição de estado*, que são grafos direcionados onde os nós representam os estados e os ramos representam os eventos. Nesses diagramas, os estados marcados são caracterizados por nós desenhados com linhas duplas e o estado inicial é identificado por uma seta.

2.1.3 *Geradores*

O gerador é um autômato $G = (\Sigma, Q, \delta, q_0, Q_m)$, onde Σ é um alfabeto de eventos, Q é um conjunto de estados, $q_0 \in Q$ é o estado inicial, $Q_m \subseteq Q$ é o conjunto de estados marcados e $\delta: \Sigma \times Q \rightarrow Q$, a função de transição, é uma função parcial definida em cada estado de Q para um subconjunto de Σ.

A função de transição δ pode ser naturalmente estendida para cadeias de eventos. Assim, define-se a *função de transição estendida* como a função $\hat{\delta}: \Sigma^* \times Q \rightarrow Q$ tal que, para

$q \in Q$, $s \in \Sigma^*$ e $\sigma \in \Sigma$, $\hat{\delta}(\varepsilon,q) = q$ e $\hat{\delta}(s\sigma,q) = \delta(\sigma,\hat{\delta}(s,q))$, sempre que $q' = \hat{\delta}(s,q)$ e $\delta(\sigma,q')$ estiverem ambas definidas.

Na abordagem de linguagens controláveis proposta por Ramadge e Wonham [20], a planta a controlar é modelada por um gerador G, cujo alfabeto de eventos é particionado em dois subconjuntos $\Sigma = \Sigma_c \cup \Sigma_u$. Eventos controláveis $\Sigma_c \subseteq \Sigma$ são aqueles que podem ser ativados ou desativados por agentes externos. Eventos não controláveis $\Sigma_u \subseteq \Sigma$ são aqueles cuja ocorrência não pode ser evitada e por isso são considerados permanentemente habilitados. Os eventos controláveis são representados no diagrama de transição de estado por ramos interceptados.

O comportamento da planta G é caracterizado por dois subconjuntos de Σ^*: o *comportamento gerado* de G, denotado por $L(G)$, representa todas as seqüências de eventos que a planta pode gerar; o *comportamento marcado* de G, denotado por $L_m(G)$, caracteriza as seqüências representando tarefas completas. Formalmente, a linguagem gerada $L(G)$ é definida por $L(G) = \{s: s \in \Sigma^* \text{ e } \hat{\delta}(s,q_0) \text{ esteja definida}\}$ e a linguagem marcada $L_m(G)$ é definida por $L_m(G) = \{s: s \in L(G) \text{ e } \hat{\delta}(s,q_0) \in Q_m\}$.

2.1.4 *Projeções*

Sejam Σ e Σ_i conjuntos de eventos com $\Sigma_i \subset \Sigma$. P_i: $\Sigma^* \rightarrow \Sigma_i^*$, a projeção natural de Σ^* para Σ_i^*, é definida recursivamente por:

$$P_i(\varepsilon) = \varepsilon$$

$$P_i(e) = \begin{cases} \varepsilon & \text{se } e \notin \Sigma_i \\ e & \text{se } e \in \Sigma_i \end{cases}$$

$$P_i(ue) = P_i(u)P_i(e) \text{ onde } u \in \Sigma^*;\ e \in \Sigma$$

O conceito de projeção natural pode ser estendido para linguagens regulares como: $P_i(L) = \{u' \in \Sigma_i^* \mid u' = P_i(u) \text{ para algum } u \in L\}$. A projeção inversa P_i^{-1}:$\Sigma_i^* \rightarrow \Sigma^*$, é, então, definida como $P_i^{-1}(L) = \{u \in \Sigma^* \mid P_i(u) \in L\}$.

As propriedades introduzidas a seguir, relativas às projeções natural e inversa, serão consideradas válidas ao longo da dissertação. Como a publicação anterior dessas propriedades não foi identificada na literatura, suas demonstrações são apresentadas.

Propriedade 1: Sejam $L \subseteq \Sigma^*$ e $P_i:\Sigma^* \rightarrow \Sigma_i^*$. Sejam t e u cadeias de eventos tais que $tu \in L$. Então, $P_i(tu) = P_i(t)P_i(u)$.

Prova:

Como $t, u \in L$ são cadeias finitas, a cadeia u pode ser representada como $u_1u_2...u_{|u|}$. Então, $P_i(tu) = P_i(tu_1u_2...u_{|u|}) = P_i(tu_1u_2...u_{|u|-1})P_i(u_{|u|}) = P_i(t)P_i(u_1)P_i(u_2)...P_i(u_{|u|-1})P_i(u_{|u|}) = P_i(t)P_i(u_1u_2...u_{|u|-1}u_{|u|}) = P_i(t)P_i(u)$. ♦

Propriedade 2: Sejam $L \subseteq \Sigma^*$ e $P_i:\Sigma^* \rightarrow \Sigma_i^*$. Então, $\overline{P_i(L)} = P_i(\overline{L})$.

Prova:

i) $P_i(\overline{L}) \subseteq \overline{P_i(L)}$

Seja $s \in P_i(\overline{L})$. Seja $t \in \overline{L}$ com $P_i(t) = s$ e v tal que $tv \in L$. Seja $u = P_i(v)$. Então, pela definição de prefixo fechamento, $s \in \overline{su} = \overline{P_i(t)P_i(v)} = \overline{P_i(tv)} \in \overline{P_i(L)}$.

ii) $\overline{P_i(L)} \subseteq P_i(\overline{L})$

Seja agora $s \in \overline{P_i(L)}$. Então, $\exists u \mid su \in P_i(L)$, i.e., $\exists t, v \mid tv \in L \wedge P_i(t) = s \wedge P_i(v) = u$. Assim, $t \in \overline{L}$ e, portanto, $s = P_i(t) \subseteq P_i(t) \cup P_i(\overline{L}) = P_i(\overline{L})$. ♦

Propriedade 3: Sejam $L_i \subseteq \Sigma_i^*$ e $P_i:\Sigma^* \rightarrow \Sigma_i^*$. Sejam t e u cadeias de eventos tais que $tu \in L_i$. Então, $P_i^{-1}(tu) = P_i^{-1}(t)P_i^{-1}(u)$.

Prova:

i) $P_i^{-1}(tu) \subseteq P_i^{-1}(t)P_i^{-1}(u)$:

Seja $v \in P_i^{-1}(tu)$. Então $P_i(v) = tu$ e, pela definição de projeção, $\exists\ v_1, v_2$ tais que $v = v_1v_2$, $P_i(v_1)=t$ e $P_i(v_2)=u$. Assim, $v_1 \in P_i^{-1}(t)$ e $v_2 \in P_i^{-1}(u) \rightarrow v = v_1v_2 \in P_i^{-1}(t)P_i^{-1}(u)$.

ii) $P_i^{-1}(t)P_i^{-1}(u) \subseteq P_i^{-1}(tu)$:

Seja $v \in P_i^{-1}(t)P_i^{-1}(u)$. Então v pode ser decomposta em $v_1 \in P_i^{-1}(t)$ e $v_2 \in P_i^{-1}(u)$. Assim, $P_i(v_1v_2) = P_i(v_1)P_i(v_2) = tu \rightarrow v_1v_2 \in P_i^{-1}(tu)$. ♦

Propriedade 4: Sejam $L_i \subseteq \Sigma_i^*$ e $P_i:\Sigma^* \rightarrow \Sigma_i^*$. Então, $P_i^{-1}(\overline{L_i}) = \overline{P_i^{-1}(L_i)}$.

Prova:

i) $\overline{P_i^{-1}(L_i)} \subseteq P_i^{-1}(\overline{L_i})$

Seja $s \in \overline{P_i^{-1}(L_i)}$. Então, $\exists u \mid su \in P_i^{-1}(L_i)$, i.e., $P_i(su) = P_i(s)P_i(u) \in L_i$. Assim, $P_i(s) \in \overline{L_i}$ e, portanto, $P_i^{-1}(P_i(s)) \in P_i^{-1}(\overline{L_i})$. Por conseguinte, $s \in P_i^{-1}(P_i(s)) \subseteq P_i^{-1}(\overline{L_i})$.

ii) $P_i^{-1}(\overline{L_i}) \subseteq \overline{P_i^{-1}(L_i)}$

Seja $s \in P_i^{-1}(\overline{L_i})$. Seja $t = P(s)$, então, $t \in \overline{L_i} \Rightarrow \exists v \mid tv \in L_i$. Assim, $P_i^{-1}(t)P_i^{-1}(v) = P_i^{-1}(tv) \subseteq P_i^{-1}(L_i)$ e, portanto, $P_i^{-1}(t) \subseteq \overline{P_i^{-1}(L_i)}$. Por conseguinte, $s \in P_i^{-1}(P_i(s)) = P_i^{-1}(t) \subseteq \overline{P_i^{-1}(L_i)}$. ♦

Propriedade 5: Seja $L_1, L_2 \subseteq \Sigma_1^*$, $P_1:\Sigma \rightarrow \Sigma_1$. Então, $P_1^{-1}(L_1) \cap P_1^{-1}(L_2) = P_1^{-1}(L_1 \cap L_2)$.

Prova:

$P_1^{-1}(L_1) \cap P_1^{-1}(L_2) = \{u \in \Sigma^* \mid P_1(u) \in L_1\} \cap \{u \in \Sigma^* \mid P_1(u) \in L_2\} = \{u \in \Sigma^* \mid P_1(u) \in L_1 \wedge P_1(u) \in L_2\} = \{u \in \Sigma^* \mid P_1(u) \in L_1 \cap L_2\} = P_1^{-1}(L_1 \cap L_2)$ ♦

2.1.5 *Combinação de Sistemas a Eventos Discretos*

Sejam $L_i \subseteq \Sigma_i^*$, $i=1,...,n$. Seja $\Sigma = \bigcup_{i=1}^{n} \Sigma_i$ e $P_i:\Sigma^* \rightarrow \Sigma_i^*$ a projeção natural de Σ em Σ_i, $i=1,...,n$. Define-se o produto síncrono de linguagens $\|_{i=1}^{n} L_i \subseteq \Sigma^*$ como: $\|_{i=1}^{n} L_i = \bigcap_{i=1}^{n} P_i^{-1}(L_i) = \{u \in \Sigma^* \mid \bigwedge_{i=1}^{n} P_i(u) \in L_i\}$.

A composição de plantas modeladas por $G_i = (\Sigma_i, Q_i, \delta_i, q_{0i}, Q_{mi})$, $i = 1,...,n$, gera uma planta global $G = \|_{i=1}^{n} G_i$, definida sobre o alfabeto $\Sigma = \bigcup_{i=1}^{n} \Sigma_i$ como um gerador com

comportamento global dado por $L_m(G) = \|_{i=1}^{n} L_m(G_i)$ e $L(G) = \|_{i=1}^{n} L(G_i)$. Pode-se observar que, numa análise de pior caso, o número de estados de um gerador G cresce exponencialmente com a composição de subsistemas G_i, assim, se cada planta G_i tiver m estados, planta global $G = \|_{i=1}^{n} G_i$ será formada, no pior caso, por m^n estados.

As seguintes propriedades, extraídas de [25], são válidas para o produto síncrono de linguagens.

Propriedade 6: Para alfabetos $\Sigma=\Sigma_1\cup\Sigma_2$, sejam $L_1 \subseteq \Sigma_1^*$ e $L_2 \subseteq \Sigma_2^*$. Seja $P_1:\Sigma^*\rightarrow\Sigma_1^*$. Então, $P_1^{-1}(L_1) = L_1\|\Sigma^* = L_1\|(\Sigma-\Sigma_1)^* = L_1\|\Sigma_2^* = L_1\|(\Sigma_2-\Sigma_1)^*$.

Propriedade 7: Para alfabetos $\Sigma=\Sigma_1\cup\Sigma_2$, sejam $L_1 \subseteq \Sigma_1^*$ e $L_2 \subseteq \Sigma_2^*$. Então, $L_1\|L_2 = (L_1\|(\Sigma_2-\Sigma_1)^*)\cap (L_2\|(\Sigma_1-\Sigma_2)^*)$.

Propriedade 8: Para alfabetos $\Sigma=\Sigma_1\cup\Sigma_2\cup\Sigma_3$, sejam $L_1 \subseteq \Sigma_1^*$, $L_2 \subseteq \Sigma_2^*$ e $L_3 \subseteq \Sigma_3^*$. Então, $(L_1\|L_2)\|L_3 = L_1\|(L_2\|L_3)$.

Propriedade 9: Para alfabetos $\Sigma_0 \subseteq \Sigma_1 \cup \Sigma_2 = \Sigma$, sejam $L_1 \subseteq \Sigma_1^*$ e $L_2 \subseteq \Sigma_2^*$. Seja $P_0:\Sigma^*\rightarrow\Sigma_0^*$. Então, $P_0(L_1\|L_2) \subseteq P_0(L_1)\|P_0(L_2)$ e a igualdade é verificada sempre que $\Sigma_1 \cap \Sigma_2 \subseteq \Sigma_0$.

Quando $\Sigma_1 \cap \Sigma_2 = \varnothing$, o produto síncrono $L_1 \| L_2 \subseteq \Sigma^*$ gera o conjunto de todas as possíveis combinações de cadeias de L_1 e L_2. Como nesse caso especial não há sincronização de eventos das duas linguagens, define-se o produto assíncrono $L_1 \|_{as} L_2$ como:

$$L_1 \|_{as} L_2 = \{u \in \Sigma^* | P_1(u) = L_1 \wedge P_2(u) = L_2\}.$$

Propriedade 10: Sejam $L_1 \subseteq \Sigma_1^*$ e $L_2 \subseteq \Sigma_2^*$. Se $\Sigma_1 \cap \Sigma_2 = \varnothing$, então $\overline{L_1 \| L_2} = \overline{L_1} \| \overline{L_2}$.

Prova:

i) $\overline{L_1 \| L_2} \subseteq \overline{L_1} \| \overline{L_2}$ é sempre verdade, pois $\overline{P_1^{-1}(L_1) \cap P_2^{-1}(L_2)} \subseteq \overline{P_1^{-1}(L_1)} \cap \overline{P_2^{-1}(L_2)} = P_1^{-1}(\overline{L_1}) \cap P_2^{-1}(\overline{L_2})$.

ii) $\overline{L_1} \,||\, \overline{L_2} \subseteq \overline{L_1 \,||\, L_2}$, se $\Sigma_1 \cap \Sigma_2 = \varnothing$.

Seja $s \in \overline{L_1} \,||\, \overline{L_2}$. Então, $P_1(s) \in \overline{L_1} \wedge P_2(s) \in \overline{L_2}$, i.e., $\exists (t \in \Sigma_1 \wedge u \in \Sigma_2) \mid P_1(s)t \in L_1 \wedge P_2(s)u \in L_2$. Como $\Sigma_1 \cap \Sigma_2 = \varnothing$, $P_1(u) = \varepsilon \wedge P_2(t) = \varepsilon$ e $\exists v \in (\Sigma_1 \cup \Sigma_2)^* \mid P_1(v) = t \wedge P_2(v) = u$ (por exemplo, $v = tu$). Assim, $P_1(sv) = P_1(s)t \in L_1$ e $P_2(sv) = P_2(s)u \in L_2$ implicam que $sv \in L_1 \,||\, L_2$. Dessa forma, $s \in \overline{sv} \in \overline{L_1 \,||\, L_2}$. Portanto, $\overline{L_1} \,||\, \overline{L_2} \subseteq \overline{L_1 \,||\, L_2}$. ♦

2.2 Controle Monolítico

O objetivo do controle supervisório monolítico é projetar um único controlador cuja função é habilitar ou desabilitar eventos controláveis, conforme a seqüência de eventos observados na planta, de forma que o sistema em malha fechada obedeça a algumas regras operacionais especificadas. A figura abaixo ilustra o funcionamento de um supervisor monolítico.

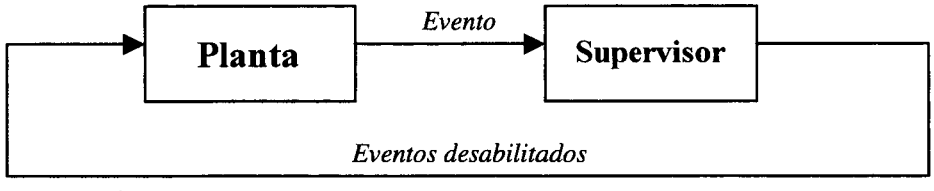


***Fig. 1:** Funcionamento de supervisor monolítico.*

2.2.1 *Supervisores*

Formalmente, o *supervisor* (ou controlador) é um par $h = (R, \psi)$, onde $R = (\Sigma, X, \xi, x_0, X_m)$ é um autômato que reconhece uma linguagem sobre o mesmo conjunto Σ de eventos da planta G e ψ é um mapeamento entre o par (estados de R, eventos de Σ) e o conjunto { desabilitado, habilitado}. O comportamento em malha fechada de um sistema controlado por um supervisor $h = (R, \psi)$ é representado por um autômato h/G, cujo comportamento fechado, denotado por $L(h/G)$, permite a ocorrência de uma cadeia de eventos se esta for aceita por G e R e se cada elemento da cadeia estiver habilitado por ψ. O comportamento marcado original que sobrevive com a supervisão é denotado por $L_m(h/G) = L(h/G) \cap L_m(G)$.

Na prática, um supervisor h pode ser representado por um gerador S, cujas mudanças de estado são ditadas pela ocorrência de eventos na planta, de modo que a ação de controle sobre G esteja implícita na estrutura de transição de S. Para isso, o supervisor S é construído de forma que as transições não habilitadas por h não apareçam na estrutura de transição de estados de S e que as transições habilitadas por h, e fisicamente possíveis, apareçam na estrutura de S. Portanto, a ação de controle do supervisor S, definida para cada estado do gerador correspondente, é desabilitar na planta os eventos que não possam ocorrer em S após uma cadeia de eventos observada. O funcionamento do sistema controlado *S/G,* equivalente a *h/G,* pode ser descrito pelo SED correspondente à composição síncrona de S e G, isto é, $S \parallel G$. Pela simplicidade oferecida, este modelo de supervisor é adotado na seqüência do documento.

Diz-se que um supervisor S é *próprio* (ou não bloqueante) para G se garantir o não bloqueio do sistema em malha fechada, isto é, se $\overline{L_m(S/G)} = L(S/G)$.

2.2.2 *Condições para a síntese de supervisores*

A solução do problema de síntese de um controlador monolítico pode ser convenientemente descrita usando as noções de controlabilidade e de $L_m(G)$-fechamento.

Uma linguagem $K \subseteq \Sigma^*$ é uma *linguagem controlável* de $L(G)$, ou controlável em relação a (e.r.a) G, ou apenas controlável, quando a G associada está implícita, se $\overline{K}\Sigma_u \cap L(G) \subseteq \overline{K}$, i.e., se $\forall s \in \overline{K}, s\Sigma_u \subset L(G) \rightarrow s\Sigma_u \subset \overline{K}$, ou ainda, se $\forall s \in \overline{K}, \exists u \in \Sigma_u^* \mid su \in L(G) \wedge su \notin \overline{K}$. Isso significa que a ocorrência de um evento controlável e fisicamente possível, após uma cadeia de $\overline{K}$, mantém a seqüência no conjunto $\overline{K}$. A classe de linguagens controláveis contidas numa linguagem M é denotada por $C(M, G) = \{ K \mid K \subseteq M$ e K é controlável e.r.a $G\}$ e é fechada por união. Sendo assim, $C(M, G)$ contém um único elemento supremo, chamado de $SupC(M, G)$.

Diz-se que uma linguagem K é $L_m(G)$-*fechada* se $K = \overline{K} \cap L_m(G)$, isto é, se todos os seus prefixos que são palavras de $L_m(G)$ forem também palavras de K.

As condições para a existência de um supervisor próprio que atenda a uma dada especificação podem ser resumidas pela Proposição 1 [20].

Proposição 1: Seja um SED G para o qual é especificado um comportamento desejado modelado pela linguagem K.

1. Para $K \subseteq L(G)$ não vazia existe um supervisor S tal que $L(S/G) = K$ se e somente se K é prefixo fechada e controlável.
2. Para $K \subseteq L_m(G)$ não vazia existe um supervisor próprio S tal que $L_m(S/G) = K$ se e somente se K é $L_m(G)$-fechada e controlável.

Nem sempre é possível construir um supervisor que restrinja o comportamento do sistema a uma linguagem especificada de forma exata. Quando um comportamento especificado não atende às condições para a síntese monolítica, é possível projetar um supervisor próprio que atenda às especificações de forma minimamente restritiva. Desde que a eventual restrição no comportamento desejado seja aceitável, o controlador é calculado para que o sistema em malha fechada obedeça à máxima linguagem controlável contida na especificação. Neste caso, o controle monolítico objetiva sintetizar um supervisor S para uma linguagem especificada $K \subseteq \Sigma^*$, tal que $L_m(S/G) = SupC(K, G)$. Se a restrição da linguagem $SupC(K, G)$ não for aceitável, diz-se que o problema não tem solução.

A abordagem de controle monolítico é elucidada pelo seguinte exemplo, extraído de [25]:

2.2.3 *Exemplo*

Consideram-se dois usuários para dois recursos compartilhados. Para cumprir sua respectiva tarefa, cada usuário $i=1,2$ precisa simultaneamente de ambos recursos A e B, sendo que os recursos são adquiridos em qualquer ordem. O evento de o usuário i tomar o Recurso A é representado por α_i e tomar o Recurso B, β_i. O evento liberação de ambos recursos pelo usuário i é representado pelo evento γ_i. Considerando $\Sigma_c = \{\alpha_1, \beta_1, \alpha_2, \beta_2\}$ e $\Sigma_u = \{\gamma_1, \gamma_2\}$, pode-se modelar a planta G pela composição dos geradores para o Usuário 1 e Usuário 2, cujos diagramas de transição são apresentados de forma simplificada na figura abaixo.

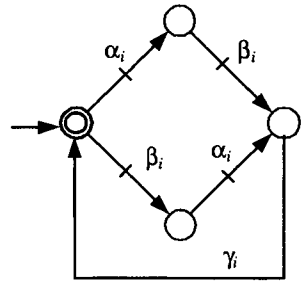


Fig. 2: Gerador para o Usuário i.

As especificações R_A e R_B para o Recurso A e Recurso B podem ser expressas de maneira simples pelas linguagens geradas pelos geradores abaixo:

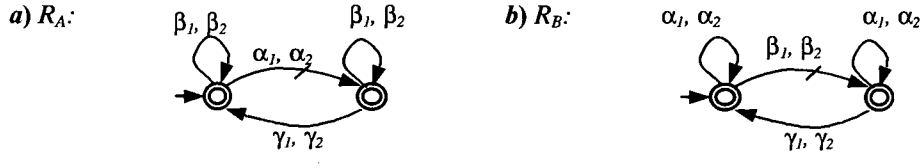


Fig. 3: Gerador para as especificações sobre a) Recurso A e b) Recurso B.

A especificação global K é formada pela composição das linguagens R_A, R_B e G. Então, sintetiza-se o supervisor monolítico ótimo S que atende à máxima linguagem controlável $SupC(K, G)$, modelada pelo gerador abaixo.

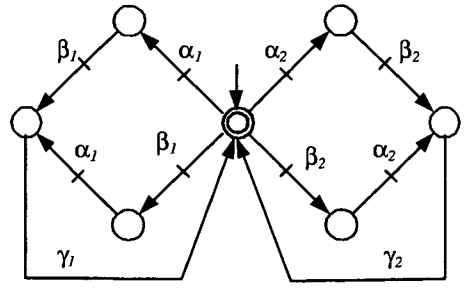


Fig. 4: Gerador para SupC(K, G).

Finalmente, a ação do supervisor S considera que inicialmente os usuários estão sem recursos. Assim que um usuário adquire um recurso, S impede o outro usuário de adquirir qualquer recurso até que o primeiro complete sua tarefa.

Os conceitos básicos de linguagens e autômatos introduzidos neste capítulo são fundamentais para a compreensão da abordagem de Linguagens Controláveis. Da mesma

forma, as propriedades sobre projeção e composição de linguagens apresentadas servem de ferramenta para o desenvolvimento dos resultados desta dissertação.

A teoria clássica de controle proposta por Ramadge e Wonham possui a vantagem de permitir a síntese automática de supervisores, ao invés do método comum de tentativa e erro. Igualmente, a noção de máxima linguagem controlável garante a síntese de controladores de forma minimamente restritiva.

No entanto, quando um grande número de tarefas deve ser executado pelo sistema de controle, a abordagem monolítica pode ter um desempenho computacional bastante desfavorável. Isso porque a composição síncrona das especificações gera um crescimento exponencial no número de estados do modelo e, por conseguinte, na complexidade computacional do problema. Nesse contexto, a teoria clássica de controle modular é apresentada no capítulo seguinte como alternativa para reduzir a complexidade computacional na síntese de supervisores.

3. CONTROLE MODULAR CLÁSSICO

Uma forma de diminuir a complexidade da síntese de controladores é dividir a tarefa de controle em várias subtarefas, que são resolvidas usando a teoria clássica de controle proposta por Ramadge e Wonham, e combinar os subcontroladores resultantes de modo a solucionar o problema original. Esta construção é chamada de *síntese modular* e os controladores resultantes de *supervisores modulares*. O ação conjunta de supervisores modulares desabilita um evento sempre que este for desabilitado por algum dos subcontroladores. Este funcionamento é ilustrado na figura abaixo.

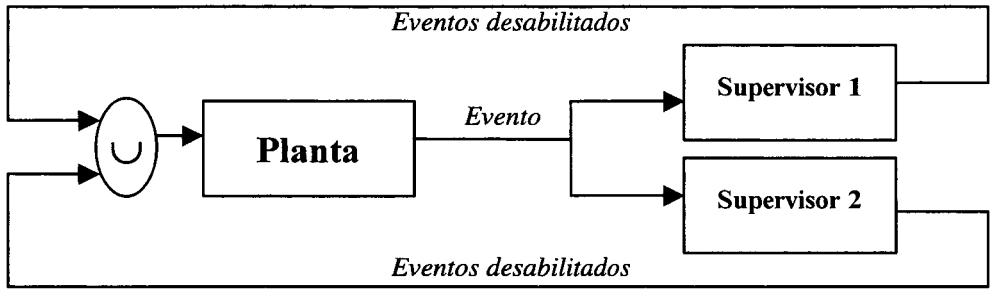


Fig. 5: *Funcionamento de supervisores modulares.*

A síntese modular permite, assim, que problemas complexos possam ser decompostos em módulos mais simples, de forma a atribuir maior flexibilidade ao controlador resultante. Além de ser mais facilmente construído, um supervisor modular costuma ser mais facilmente modificado, atualizado e corrigido. Por exemplo, se uma subtarefa for mudada, só é preciso reconstruir o subcontrolador correspondente ao invés de se refazer todo o sistema supervisor.

Em contrapartida, os controladores modulares têm suas ações de controle baseadas numa versão parcial do estado de funcionamento do sistema global. Por conseguinte, a síntese modular é, em geral, degradada em relação à solução monolítica, podendo em muitos casos gerar conflitos na ação de controle. As condições para que isso não ocorra são apresentadas na seqüência deste documento.

Este capítulo apresenta a teoria clássica proposta por Wonham e Ramadge [19, 26] para a síntese de supervisores modulares para sistemas a eventos discretos, segundo a abordagem de linguagens controláveis. Numa primeira seção, são apresentados os

principais resultados sobre o controle modular clássico para problemas com duas especificações. Um estudo mais profundo sobre este assunto pode ser feito a partir de [14, 19, 20, 25, 26].

Uma vez que normalmente os problemas de controle para sistemas de grande porte envolvem mais de duas especificações, a segunda seção deste capítulo apresenta a extensão do controle modular clássico para múltiplas tarefas. Essa extensão foi extraída principalmente de [2, 26].

3.1 Controle Modular para duas Especificações

Sejam S_i, $i = 1,...,m$, supervisores próprios para G. Então, a *composição* de S_i, $i = 1,...,m$, é denotada por $\bigwedge_{i=1,...,m} S_i$, cuja ação supervisória é habilitar um evento somente quando este for habilitado por S_1, S_2, ... e S_m simultaneamente.

Apesar da composição de supervisores ser uma operação associativa e comutativa, a ação conjunta de dois supervisores não bloqueantes não é necessariamente não bloqueante. O exemplo a seguir ilustra um caso de controle modular bloqueante.

Exemplo: Seja o problema de dois recursos compartilhados entre dois usuários apresentado na Seção 2.2.3. Ao invés de construir um único controlador para ambos os recursos, sintetizam-se controladores que atendam a cada linguagem R_A e R_B de forma que as correspondentes linguagens controladas sejam dadas por $K_A = R_A \| L_m(G)$ e $K_B = R_B \| L_m(G)$. Como cada linguagem é controlável e não bloqueante e.r.a G, os respectivos controladores S_A e S_B são próprios. Entretanto, as linguagens K_A e K_B são conflitantantes. É fácil ver que a operação conjunta de K_A e K_B pode levar a um bloqueio: se o Usuário 1 tomar um recurso (evento α_1 ou β_1) e o Usuário 2 tomar o segundo recurso (evento β_2 ou α_2, respectivamente), nenhum dos dois usuários poderá completar sua tarefa e, por conseguinte, o sistema permanecerá bloqueado.

A chave para garantir o não bloqueio entre controladores é a propriedade de modularidade.

3.1.1 *Modularidade*

Para L_1, $L_2 \subseteq \Sigma^*$ é sempre verdade que $\overline{L_1 \cap L_2} \subseteq \overline{L_1} \cap \overline{L_2}$, isto é, o prefixo de uma cadeia comum a L_1 e a L_2 é também um prefixo de L_1 e de L_2. Diz-se que L_1 e L_2 são *modulares* (ou não conflitantes) se $\overline{L_1 \cap L_2} = \overline{L_1} \cap \overline{L_2}$. Isso quer dizer que duas linguagens são modulares se, toda vez que compartilharem um prefixo, compartilharem uma palavra contendo esse prefixo. Por exemplo, quaisquer linguagens prefixo fechadas são modulares entre si.

3.1.2 *Controle Modular*

Proposição 2: [26] Sejam S e T supervisores para G. Então, o supervisor $R=S \wedge T$ tem comportamento $L(R/G)=L(S/G) \cap L(T/G)$ e $L_m(R/G)=L_m(S/G) \cap L_m(T/G)$.

Considerando-se agora que os supervisores S e T da Proposição 2 sejam próprios para G ($\overline{L_m(S/G)} = L(S/G)$ e $\overline{L_m(T/G)} = L(T/G)$), a condição para que R seja não bloqueante é que $\overline{L_m(R/G)} = L(R/G)$. Mas essa condição é equivalente a $\overline{L_m(S/G) \cap L_m(T/G)} = L(S/G) \cap L(T/G) = \overline{L_m(S/G)} \cap \overline{L_m(T/G)}$, ou seja, a $L_m(S/G)$ e $L_m(T/G)$ serem modulares. Com base nesta observação, a proposição seguinte apresenta a modularidade como condição para o não bloqueio da síntese modular.

Proposição 3: [26] Seja G uma planta e K_1, $K_2 \subseteq \Sigma^*$ linguagens $L_m(G)$-fechadas e controláveis. Sejam S e T supervisores próprios para G tais que $L_m(S/G) = K_1$ e $L_m(T/G) = K_2$. Então, o supervisor $R=S \wedge T$ tem comportamento $L_m(R/G)= K_1 \cap K_2$. Além disso, R é próprio para G se e somente se as linguagens K_1 e K_2 são modulares.

Enquanto que a Proposição 3 fundamenta o controle modular, a seguinte proposição mostra que, dadas duas especificações, a modularidade das máximas linguagens controláveis calculadas localmente para cada uma das especificações garante que a síntese modular seja ótima, isto é, não apresente perda de desempenho em relação ao controle monolítico.

Proposição 4: [26] Sejam K_1, $K_2 \subseteq L(G) \subseteq \Sigma^*$. Então, se $SupC(K_1, G)$ e $SupC(K_2, G)$ são modulares, $SupC(K_1, G) \cap SupC(K_2, G) = SupC(K_1 \cap K_2, G)$.

Assim, o resultado anterior em conjunto com a Proposição 3 indica que se forem construídos supervisores para $SupC(K_1, G)$ e para $SupC(K_2, G)$, garantida a modularidade dessas linguagens, a ação conjunta dos supervisores gera a máxima linguagem controlável de $K_1 \cap K_2$.

3.1.3 *Análise de Complexidade*

Considerando-se apenas dependência da complexidade com o número de estados dos geradores para as linguagens envolvidas, é feita uma análise de pior caso da complexidade computacional envolvida na síntese modular. Uma análise mais detalhada pode ser encontrada em [26].

Para isso, assume-se que o gerador G tem m estados. Então, para linguagens K_1 e K_2 especificadas por autômatos de no máximo n estados, a complexidade da verificação da controlabilidade de K_i, $i=1,2$, é $O(nm)$. A interseção $K_1 \cap K_2$, assim como a verificação da modularidade, é uma operação $O(n^2)$ e o cálculo de $SupC(K_i, G)$ é $O(n^2m^2)$. Portanto, a síntese de controladores modulares tem complexidade computacional de ordem $O(n^2m^2)$. Por outro lado, o cálculo de um controlador pela abordagem monolítica tem complexidade $O(n^4m^2)$.

3.1.4 *Exemplo*

Para exemplificar a metodologia proposta nesta seção, sintetizam-se controladores modulares para o modelo simplificado de "linha de transferência industrial" ilustrado na Figura 6. A linha é composta de três máquinas M_1, M_2 e M_3 trabalhando em série e ligadas pelos *buffers* B_1 e B_2 de capacidade 1. Como especificações, são consideradas a não ocorrência de *underflow* e *overflow* em B_1 e B_2.

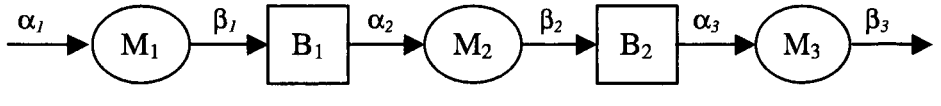


***Fig. 6:** Linha de Transferência Industrial*

O comportamento G_i de cada máquina M_i, i = 1,2,3, e as especificações genéricas E_a e E_b para B_i, i = 1,2, podem ser descritos respectivamente pelos autômatos das figuras 7 e 8. Consideram-se os eventos α_i como o início de operação da máquina M_i, com a retirada

de uma peça de B_{i-1} (quando existir), e β_i como o fim de operação da máquina M_i com o depósito de uma peça em B_i (quando existir).

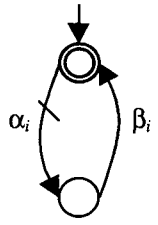


Fig. 7:. G_i, i = 1,2,3

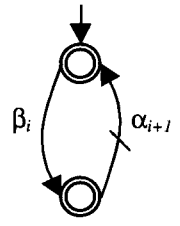


Fig. 8:. Autômato para E_x, x= a(i=1), b(i=2)

O comportamento em malha aberta da linha de transferência pode ser representado por um gerador $G = G_1 || G_2 || G_3$ formado pela composição dos geradores das três máquinas. O gerador G é representado pelo diagrama de transição de estados da figura abaixo.

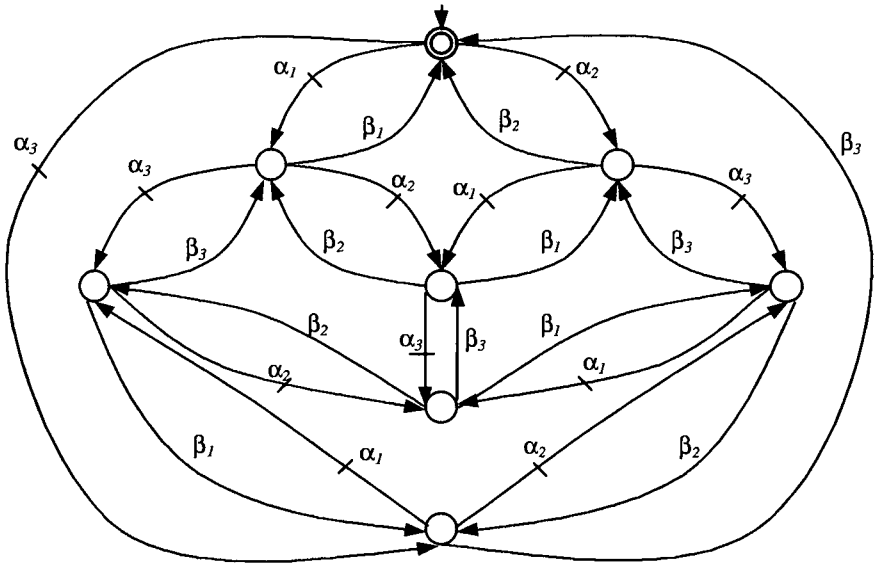


Fig. 9:. Autômato para G.

Assim, as linguagens K_A e K_B especificadas para a planta G são obtidas pelo produto síncrono das especificações genéricas E_a e E_b com a planta G, ou seja, $K_A = E_a \,||\, L_m(G)$ e $K_B = E_b \,||\, L_m(G)$. Os autômatos correspondentes a K_A e K_B têm 16 estados cada.

Como as especificações K_A e K_B não são controláveis em relação a G, é preciso calcular as máximas linguagens controláveis $SupC(K_A, G)$ e $SupC(K_B, G)$, ambas representadas por autômatos de 18 estados e 32 transições.

Para garantir que os supervisores para $SupC(K_A, G)$ e para $SupC(K_B, G)$ sejam não conflitantes, testa-se a modularidade entre estas duas linguagens. Isso é feito calculando-se

$\overline{SupC(K_A, G)} \cap \overline{SupC(K_B, G)}$ e $\overline{SupC(K_A, G) \cap SupC(K_B, G)}$ e verificando-se a sua igualdade. Dessa forma, garante-se que não há nenhuma perda de eficiência entre a ação dos controladores modulares resultantes e a melhor solução centralizada.

3.2 Controle Modular para Múltiplas Especificações

Os problemas de síntese envolvendo sistemas de grande porte costumam apresentar grande número de tarefas a serem cumpridas. Parece natural que, quanto maior for o número de módulos envolvidos na solução do problema, mais vantajosa será a abordagem modular em relação à monolítica. Sendo assim, esta seção apresenta os resultados do controle modular clássico estendidos para múltiplas especificações.

3.2.1 Modularidade

O conceito de modularidade pode ser estendido da seguinte forma: seja I um conjunto de índices. Sejam $K_i \subseteq \Sigma^*$, $i \in I$. O conjunto $\{K_i, i \in I\}$ é modular quando $\bigcap_{i \in I} \overline{K_i} = \overline{\bigcap_{i \in I} K_i}$ [2]. Essa propriedade implica que um prefixo comum a todas as linguagens é necessariamente um prefixo da composição das mesmas linguagens.

Nota-se que a modularidade das linguagens duas a duas não implica necessariamente a modularidade do conjunto. Assim, a linguagem representando a interseção de todas as linguagens do conjunto deve ser calculada. Isso pode representar um custo muito elevado em termos computacionais.

3.2.2 Controle Modular

A partir do conceito estendido de modularidade, os resultados de controle modular clássico podem ser naturalmente generalizados conforme os corolários apresentados a seguir.

Corolário 1: [2] Seja I um conjunto de índices. Sejam S_i, $i \in I$, supervisores próprios para G. Então, o supervisor $R = \bigwedge_{i \in I} S_i$ tem comportamento $L_m(R/G) = \bigcap_{i \in I} L_m(S_i/G)$. Além disso, o supervisor R é próprio se e somente se o conjunto $\{L_m(S_i/G), i \in I\}$ é modular.

Corolário 2: [2] Seja I um conjunto de índices. Seja o conjunto modular de linguagens $\{K_i \subseteq \Sigma^*, i \in I\}$. Se K_i, $i \in I$, são todas $L_m(G)$-fechadas e controláveis, então $\bigcap_{i \in I} K_i$ é $L_m(G)$-fechada e controlável.

Corolário 3: [2] Seja I um conjunto de índices. Sejam $K_i \subseteq L(G) \subseteq \Sigma^*$, $i \in I$. Então, se $\{SupC(K_i, G), i \in I\}$ é modular, $\bigcap_{i \in I} SupC(K_i, G) = SupC(\bigcap_{i \in I} K_i, G)$.

3.2.3 *Análise de Complexidade*

Para analisar a complexidade computacional da síntese de múltiplos controladores modulares, assume-se uma planta cujo gerador G tem m estados. Então, para linguagens K_i, $i=1,...,q$, especificadas por autômatos de no máximo n estados, a complexidade da verificação da controlabilidade de cada linguagem K_i é $O(nm)$. A interseção $\bigcap_{i=1,...,q} K_i$, assim como a verificação da modularidade, é uma operação $O(n^q)$ e o cálculo de $SupC(K_i, G)$ é $O(n^2m^2)$. Assim, numa análise de pior caso, a síntese de controladores modulares tem complexidade computacional de ordem $O(max(n^q + n^2m^2))$. Dessa forma, a complexidade da síntese modular clássica cresce exponencialmente com o número de especificações, o que é uma desvantagem para sistemas de grande porte.

Por outro lado, o cálculo de um controlador que atenda às q linguagens especificadas pela abordagem monolítica tem complexidade $O(n^{2q}m^2)$.

Quando a condição de modularidade entre as tarefas de controle é verificada, o controle modular mostra-se bastante vantajoso em relação ao monolítico em termos da implementação da estrutura de controle e da complexidade do processo de síntese.

No entanto, a modelagem por autômatos pode induzir a uma explosão do número de estados da planta à medida em que subsistemas vão sendo agregados a ela. Apesar de cada supervisor ser concebido para uma especificação isolada, a abordagem clássica considera que cada módulo de controle observa e controla a planta inteira. Dessa forma, o controle modular clássico pode ser inviável para sistemas de grande porte.

No capítulo a seguir, a abordagem modular é estendida de forma a considerar a estrutura de módulos ao nível da planta e não apenas das especificações.

4. CONTROLE MODULAR DE SISTEMAS COMPOSTOS

Uma das grandes críticas que se faz à abordagem de Ramadge e Wonham é que a modelagem por autômatos finitos não permite uma boa representação do paralelismo, pois apenas admite o "interleaving" de eventos e não o paralelismo verdadeiro. Como conseqüência, a composição de subsistemas na síntese de controle supervisório pode levar a uma explosão de estados, o que é um fator prejudicial para problemas mais complexos.

Em sistemas de grande porte, é comum a existência de diversos subsistemas funcionando em paralelo. A modelagem desses sistemas por um único autômato é, então, de enorme dificuldade computacional. Neste capítulo, é utilizada a abordagem modular para apresentar uma solução alternativa para a síntese de controle supervisório modular, que explora aspectos de modularidade da planta. Propõe-se uma arquitetura de controle em que cada módulo atua somente sobre os subsistemas atingidos. Essa idéia é ilustrada pela figura abaixo.

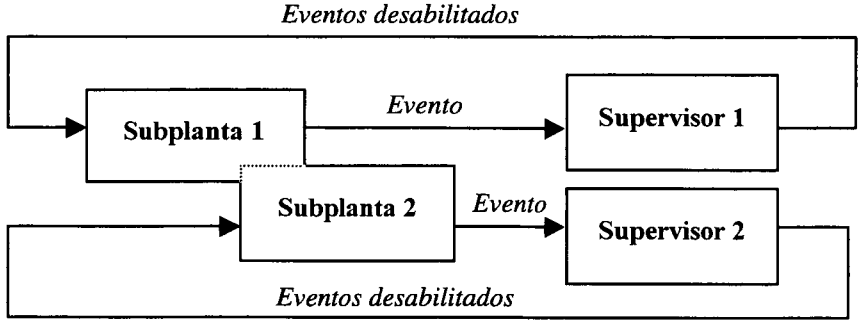


***Fig. 10:**. Funcionamento de supervisores localmente modulares.*

Numa primeira seção, é apresentada a abordagem utilizada para a modelagem de sistemas de grande porte por linguagens controláveis. As seções 4.2 e 4.3 desenvolvem os resultados de controle modular clássico para que cada supervisor modular possa ser sintetizado isoladamente a partir de modelos reduzidos da planta e para que a modularidade possa ser testada de forma mais eficiente. Finalmente, a última seção deste capítulo apresenta sinteticamente algumas soluções alternativas para os casos em que as condições para a síntese modular não são verificadas.

Os resultados deste capítulo podem ser encontrados em [6] e em [7].

4.1 Modelagem de Sistemas Compostos

No projeto de sistemas de maior complexidade, a modelagem das diversas partes envolvidas é geralmente um passo intermediário na representação do comportamento conjunto do sistema. Isso porque a modelagem dessas partes exige menor esforço computacional , menos memória e costuma ser mais compreensível para o projetista.

Como a composição de subsistemas assíncronos provoca a explosão do número de estados do sistema global, a representação do comportamento global do sistema por um gerador G e o uso desse modelo para cálculos computacionais acabam sendo proibitivos na maioria dos sistemas de grande porte. Por essa razão, esses sistemas são normalmente modelados pela composição de diversos subsistemas de menor porte. Assim, de acordo com o nível de composição das subplantas originalmente modeladas, diferentes representações para o sistema podem ser formuladas.

4.1.1 Representação por Sistemas Compostos (RSC)

Uma *Representação por Sistemas Compostos* (*RSC*) é apresentada aqui como qualquer modelagem da planta global $G = (\Sigma, Q, \delta, q_0, Q_m)$ obtida pela combinação de subplantas $G'_i = (\Sigma'_i, Q'_i, \delta'_i, q'_{0i}, Q'_{mi})$, $i \in N' = \{1,...,n'\}$. Assim, tem-se $G = \prod_{i=1}^{n'} G'_i$, com alfabeto de eventos $\Sigma = \bigcup_{i=1}^{n'} \Sigma'_i$.

4.1.2 Representação por Sistemas Produto (RSP)

Um Sistema Produto é um sistema que pode ser modelado pela composição de subsistemas completamente assíncronos entre si. Esses sistemas foram estudados por (Ramadge e Wonham, 1989) (Ramadge, 1989). Assim, chama-se de Representação por Sistemas Produto (RSP) qualquer RSC cujas subplantas não tenham eventos síncronos em comum.

Considerando que cada subsistema de uma RSP representa uma parte isolada de um sistema em malha aberta, pode-se afirmar que esta modelagem representa a estrutura descentralizada natural de operações concorrentes para sistemas de maior porte. Embora os resultados desta dissertação sejam válidos para qualquer RSP, a abordagem de controle

modular local apresentada na seqüência será tanto mais vantajosa quanto mais refinada for a RSP.

Na verdade, pode-se obter a mais refinada Representação por Sistema Produto equivalente a uma RSC qualquer, pela composição dos geradores síncronos deste último. Para isso, faz-se a composição dos subsistemas síncronos originais (que têm eventos em comum), criando-se o maior número possível de subsistemas assíncronos, cada qual com o mínimo de estados. Formalmente, a mais refinada RSP obtida a partir de um sistema composto G'_i, $i \in N'$, é formada por plantas assíncronas $G_i = (\Sigma_i, Q_i, \delta_i, q_{0i}, Q_{mi})$, $i \in N = \{1,...,n\}$, com $G_i = \|_{j \in K_i} G'_j$, onde K_i, $i \in N$, é o limite de uma seqüência definida por

$K^0_i = \{k\}$ $(k \in N', k \notin K_l, l \neq i)$;

$K^j_i = K^{j-1}_i \cup \{k \in N' | \Sigma_k \cap \Sigma_{K^{j-1}_i} \neq \emptyset\}$.

Pode-se mostrar facilmente que o conjunto $\{G_i\}$ obtido desta forma é único sob permutação de índices.

Seja, então, a RSP de um sistema G formada por subsistemas $G_i = (\Sigma_i, Q_i, \delta_i, q_{0i}, Q_{mi})$, $i \in N = \{1,...,n\}$. Para $j=1,...,m$, sejam agora especificações genéricas modeladas por E_{xj} definidas respectivamente em $\Sigma_{xj} \subseteq \Sigma$. Para $j=1,...,m$, a *planta local* $G_{Xj} = (\Sigma_{Xj}, Q_{Xj}, \delta_{Xj}, q_{0Xj}, Q_{mXj})$ associada à especificação E_{xj} é definida por $G_{Xj} = \|_{i \in N_{Xj}} G_i$, com $N_{Xj} = \{k \in N \mid \Sigma_k \cap \Sigma_{xj} \neq \emptyset\}$. Assim, a planta local G_{Xj} é composta apenas pelos subsistemas da modelagem original que estão diretamente (e indiretamente) restringidos por E_{xj}.

Para a demonstração dos resultados deste capítulo, são definidas ainda a *planta enxuta* $G_e = \|_{j=1}^{m} G_{Xj}$, $\Sigma_e = \cup_{j=1}^{m} \Sigma_{Xj}$, $N_e = \cup_{j=1}^{m} N_{Xj}$; as plantas complementares $G^c_{Xj} = \|_{i \in N - N_{Xj}} G_i$, $G^{ce}_{Xj} = \|_{i \in N_e - N_{Xj}} G_i$, $G^c_e = \|_{i \in N - N_e} G_i$; e as projeções naturais $P_{Xj} : \Sigma_e^* \rightarrow \Sigma_{Xj}^*$ e $P_e : \Sigma^* \rightarrow \Sigma_e^*$ $(j=1,...,m)$.

4.1.3 *Modelagem das Especificações*

As *especificações genéricas* E_{xi} consideram apenas os eventos fundamentais para a descrição do comportamento desejado. As especificações podem, então, ser expressas em

termos da planta local G_{Xi}, da planta enxuta G_e e da planta global G, respectivamente como: $E_{Xj} = E_{xj} \parallel L_m(G_{Xj})$, $K_{Xje} = E_{xj} \parallel L_m(G_e)$ e $K_{Xj} = E_{xj} \parallel L_m(G)$. Pode ser notado que, enquanto a *especificação local* E_{Xj} envolve apenas os eventos das subplantas que são afetadas pela respectiva especificação, a *especificação enxuta* K_{Xje} considera o funcionamento conjunto de todas as subplantas que são afetadas por alguma das especificações e a *especificação global* K_{Xj} representa o comportamento do sistema inteiro que satisfaz a especificação.

4.1.4 *Exemplo*

Seja a RSC de um sistema formada pelo conjunto de subplantas $\{ G'_i = (\Sigma'_i, Q'_i, \delta'_i, q'_{0i}, Q'_{mi}), i = 1,...,5 \}$. Sejam duas especificações genéricas $E_a \subseteq \Sigma_a^*$ e $E_b \subseteq \Sigma_b^*$. A relação entre os conjuntos de eventos é ilustrada pela figura a seguir.

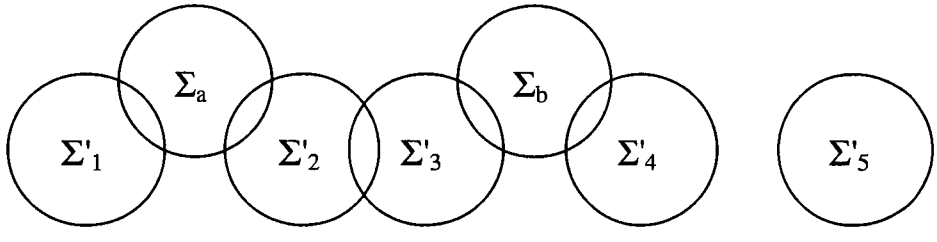


Fig. 11:. *Relação entre alfabetos do sistema composto*

Esse sistema tem uma RSP mais refinada dada pelo conjunto de plantas assíncronas $\{ G_i = (\Sigma_i, Q_i, \delta_i, q_{0i}, Q_{mi}), i = 1,...,4 \}$, onde $G_1 = G'_1$, $G_2 = G'_2 \| G'_3$, $G_3 = G'_4$ e $G_4 = G'_5$. Assim, as plantas locais são dadas por $G_A = G_1 \| G_2$ e $G_B = G_2 \| G_3$. A planta "enxuta" é calculada como $G_e = G_1 \| G_2 \| G_3$ e a planta global $G = G_1 \| G_2 \| G_3 \| G_4$. Enfim, as plantas complementares são definidas por $G_e^c = G_4$, $G_A^c = G_3 \| G_4$, $G_B^c = G_1 \| G_4$, $G_A^{ce} = G_3$ e $G_B^{ce} = G_1$.

Deste modo podem ser calculadas as especificações locais $E_A = E_a \parallel L_m(G_A)$ e $E_B = E_b \parallel L_m(G_B)$, as especificações "enxutas" $K_{Ae} = E_a \parallel L_m(G_e)$ e $K_{Be} = E_b \parallel L_m(G_e)$ e as especificações globais $K_A = E_a \parallel L_m(G)$ e $K_B = E_b \parallel L_m(G)$.

4.2 Controle Modular Local para duas Especificações

4.2.1 Modularidade Local

No caso de sistemas com RSP, a modularidade entre duas especificações genéricas pode ser testada em diferentes níveis de representação da planta. Parece natural que se duas especificações expressas em termos da planta global G forem modulares, então as mesmas especificações, quando expressas em termos da planta enxuta G_e, serão modulares. O inverso também é esperado. Isso porque a adição ou subtração de um sistema assíncrono G_e^c a ambas especificações altera de um modo equivalente ambos os comportamentos. Esse resultado é apresentado no Teorema 1.

Teorema 1: $\overline{K_A \cap K_B} = \overline{K_A} \cap \overline{K_B} \Leftrightarrow \overline{K_{Ae} \cap K_{Be}} = \overline{K_{Ae}} \cap \overline{K_{Be}}$.

Prova:

i) $K_A \cap K_B = K_A \| K_B = K_{Ae} \| L_m(G_e^c) \| K_{Be} \| L_m(G_e^c) =$

$= (K_{Ae} \| K_{Be}) \| L_m(G_e^c) = (K_{Ae} \cap K_{Be}) \| L_m(G_e^c)$;

ii) $P_e(\overline{K_A \cap K_B}) = \overline{P_e(K_A \cap K_B)} = \overline{P_e((K_{Ae} \cap K_{Be}) \| L_m(G_e^c))} =$

$= \overline{P_e(K_{Ae} \cap K_{Be}) \| P_e(L_m(G_e^c))} = \overline{(K_{Ae} \cap K_{Be}) \| \varnothing} = \overline{(K_{Ae} \cap K_{Be})}$;

iii) $\overline{K_A} \cap \overline{K_B} = \overline{K_A} \| \overline{K_B} = \overline{K_{Ae} \| L_m(G_e^c)} \| \overline{K_{Be} \| L_m(G_e^c)} =$

$= \overline{K_{Ae}} \| \overline{L_m(G_e^c)} \| \overline{K_{Be}} \| \overline{L_m(G_e^c)} = (\overline{K_{Ae}} \| \overline{K_{Be}}) \| \overline{L_m(G_e^c)} = (\overline{K_{Ae}} \cap \overline{K_{Be}}) \| \overline{L_m(G_e^c)}$;

iv) $P_e(\overline{K_A} \cap \overline{K_B}) = P_e((\overline{K_{Ae}} \cap \overline{K_{Be}}) \| \overline{L_m(G_e^c)}) = P_e(\overline{K_{Ae}} \cap \overline{K_{Be}}) \| P_e(\overline{L_m(G_e^c)}) =$

$= \overline{K_{Ae}} \cap \overline{K_{Be}} \| \varnothing = \overline{K_{Ae}} \cap \overline{K_{Be}}$;

v) $\overline{K_A \cap K_B} = \overline{K_A} \cap \overline{K_B} \Rightarrow \overline{K_{Ae} \cap K_{Be}} = P_e(\overline{K_A \cap K_B}) =$

$= P_e(\overline{K_A} \cap \overline{K_B}) = \overline{K_{Ae}} \cap \overline{K_{Be}}$;

vi) $\overline{K_{Ae} \cap K_{Be}} = \overline{K_{Ae}} \cap \overline{K_{Be}} \Rightarrow \overline{K_A \cap K_B} = \overline{(K_{Ae} \cap K_{Be}) \| L_m(G_e^c)} =$

$$= \overline{(K_{Ae} \cap K_{Be}) \| L_m(G_e^c)} = (\overline{K_{Ae}} \cap \overline{K_{Be}}) \| \overline{L_m(G_e^c)} = \overline{K_A} \cap \overline{K_B} \,. \quad \blacklozenge$$

O resultado apresentado no Teorema 1 apenas demonstra o processo intuitivo de se economizar esforço computacional ao desconsiderar no modelo da planta todos os subsistemas que não são afetados por nenhuma especificação (G_e^c). Contudo, é esperado que um ganho computacional adicional possa ser obtido verificando-se a modularidade de duas especificações expressas apenas em termos de suas respectivas plantas locais G_A e G_B, i.e., os subsistemas que são afetados por cada especificação. Este resultado será demonstrado na seqüência, mas para isso é necessária a definição de um novo conceito de modularidade.

Diz-se que as linguagens $E_A \subseteq \Sigma_A^*$ e $E_B \subseteq \Sigma_B^*$ são *localmente modulares* se $\overline{E_A \| E_B} = \overline{E_A} \| \overline{E_B}$, ou seja, se $\overline{P_A^{-1}(E_A) \cap P_B^{-1}(E_B)} = \overline{P_A^{-1}(E_A)} \cap \overline{P_B^{-1}(E_B)}$.

É importante lembrar que a projeção inversa pode ser obtida adicionando *self loops* a cada estado da representação por autômatos da linguagem original, não introduzindo nenhum novo estado. Assim, a verificação da modularidade local é feita diretamente sobre as especificações locais, não exigindo a composição da planta enxuta com as especificações.

Os Lemas 1 e 2 a seguir são utilizados na demonstração do Teorema 2, o qual garante que duas especificações locais são localmente modulares se e somente se suas representações em termos da planta global são modulares. Isto é, através da modularidade local pode-se verificar a modularidade entre duas especificações compondo-se apenas os subsistemas relacionados a elas.

Lema 1: $\quad K_{Ae} \cap K_{Be} = P_A^{-1}(E_A) \cap P_B^{-1}(E_B)$.

Prova: $K_{Xe} = E_x \| L_m(G_e) = E_x \| L_m(G_A) \| L_m(G_B) = E_X \| L_m(G_{X^c})$

Então:

$$K_{Ae} \cap K_{Be} = K_{Ae} \| K_{Be} = E_A \| L_m(G_B) \| E_B \| L_m(G_A)$$
$$= E_A \| E_B = P_A^{-1}(E_A) \cap P_B^{-1}(E_B) \quad \blacklozenge$$

Lema 2: $\quad \overline{K_{Ae}} \cap \overline{K_{Be}} = \overline{P_A^{-1}(E_A)} \cap \overline{P_B^{-1}(E_B)}$.

Prova: $K_{Xe} = E_x \parallel L_m(G_e) = E_x \parallel L_m(G_X) \parallel L_m(G_X^{ce}) = E_X \parallel L_m(G_X^{ce})$

$\Sigma_X \cap \Sigma_X^{ce} = \varnothing$

Então:

$$\begin{aligned}
\overline{K_{Ae}} \cap \overline{K_{Be}} &= \overline{K_{Ae}} \parallel \overline{K_{Be}} = \overline{E_A \parallel L_m(G_A^{ce})} \parallel \overline{E_B \parallel L_m(G_B^{ce})} \\
&= \overline{E_A} \parallel \overline{L_m(G_A^{ce})} \parallel \overline{E_B} \parallel \overline{L_m(G_B^{ce})} \\
&= (\overline{E_A} \parallel \overline{L_m(G_B^{ce})}) \parallel (\overline{E_B} \parallel \overline{L_m(G_A^{ce})}) \\
&= \overline{E_A} \parallel \overline{E_B} = P_A^{-1}(\overline{E_A}) \cap P_B^{-1}(\overline{E_B}) \\
&= \overline{P_A^{-1}(E_A)} \cap \overline{P_B^{-1}(E_B)}
\end{aligned}$$

♦

Teorema 2: Para especificações locais $E_A \subseteq \Sigma_A^*$ e $E_B \subseteq \Sigma_B^*$ e especificações globais K_A, $K_B \subseteq \Sigma^*$ definidas conforme a discussão anterior, K_A, K_B são modulares se e somente se E_A, E_B são localmente modulares. Ou seja:

$$\overline{K_A \cap K_B} = \overline{K_A} \cap \overline{K_B} \Leftrightarrow \overline{P_A^{-1}(E_A) \cap P_B^{-1}(E_B)} = \overline{P_A^{-1}(E_A)} \cap \overline{P_B^{-1}(E_B)}.$$

Prova: Deduz-se diretamente dos Lemas 1 e 2 e do Teorema 1. ♦

4.2.2 *Controle Modular Local*

Os Teoremas 1 e 2 permitem que, em sistemas compostos, se reduza a complexidade da verificação de modularidade. Entretanto, os cálculos envolvidos na síntese da máxima linguagem controlável pela abordagem clássica são realizados sempre a partir das especificações expressas em termos do comportamento da planta global. Isso pode implicar operações sobre geradores de muitos estados, no caso de sistemas de maior porte.

O Teorema 3, apresentado na seqüência, demonstra que a composição assíncrona de subsistemas não afeta sua controlabilidade. Os lemas seguintes são necessários para a demonstração desse teorema.

Lema 3: Seja $L_1 \subseteq \Sigma_1^*$, $\Sigma_1 \subset \Sigma$ e a projeção natural $P_1{:}\Sigma \rightarrow \Sigma_1$. Se $K_1 \subset \Sigma_1^*$ é controlável em relação a L_1, então $P_1^{-1}(K_1)$ é controlável em relação a $P_1^{-1}(L_1)$.

Prova:

$\Sigma_u \subseteq P_1^{-1}(P_1(\Sigma_u)) = P_1^{-1}(\Sigma_{1u} + \varepsilon)$

$\overline{P_1^{-1}(K_1)}\Sigma_u \cap P_1^{-1}(L_1) \subseteq \overline{P_1^{-1}(K_1)}P_1^{-1}(\Sigma_{1u} + \varepsilon) \cap P_1^{-1}(L_1) =$

$P_1^{-1}(\overline{K_1}(\Sigma_{1u} + \varepsilon)) \cap P_1^{-1}(L_1) = P_1^{-1}(\overline{K_1}(\Sigma_{1u} + \varepsilon) \cap L_1) = P_1^{-1}((\overline{K_1}\Sigma_{1u} \cap L_1) \cup (\overline{K_1} \cap L_1)) \subseteq$

$P_1^{-1}(\overline{K_1} \cup \overline{K_1}) = \overline{P_1^{-1}(K_1)}$. ♦

Lema 4: Sejam L_1, $K_1 \subseteq \Sigma_1^*$ e L_2, $K_2 \subseteq \Sigma_2^*$, $\Sigma_1 \cap \Sigma_2 = \varnothing$. Então $K = K_1 \|_{as} K_2$ é controlável e.r.a $L = L_1 \|_{as} L_2$ se e somente se K_1 é controlável e.r.a L_1 e K_2 é controlável e.r.a L_2.

Prova:

i) (Se) $\overline{K_1}\Sigma_{1u} \cap L_1 \subseteq \overline{K_1} \wedge \overline{K_2}\Sigma_{2u} \cap L_2 \subseteq \overline{K_2} \rightarrow \overline{K}\Sigma_u \cap L \subseteq \overline{K}$:

$\overline{K}\Sigma_u \cap L = \overline{K_1 \|_{as} K_2}\Sigma_u \cap L_1 \|_{as} L_2 = \overline{K_1} \|_{as} \overline{K_2}\Sigma_u \cap L_1 \|_{as} L_2 =$

$(P_1^{-1}(\overline{K_1}) \cap P_2^{-1}(\overline{K_2}))\Sigma_u \cap P_1^{-1}(L_1) \cap P_2^{-1}(L_2) =$

$P_1^{-1}(\overline{K_1})\Sigma_u \cap P_2^{-1}(\overline{K_2})\Sigma_u \cap P_1^{-1}(L_1) \cap P_2^{-1}(L_2) =$

$P_1^{-1}(\overline{K_1})\Sigma_u \cap P_1^{-1}(L_1) \cap P_2^{-1}(\overline{K_2})\Sigma_u \cap P_2^{-1}(L_2) \subseteq P_1^{-1}(\overline{K_1}) \cap P_2^{-1}(\overline{K_2}) = \overline{K_1} \|_{as} \overline{K_2} = \overline{K_1 \|_{as} K_2}$

$= \overline{K}$

ii) (Somente se) $\overline{K}\Sigma_u \cap L \subseteq \overline{K} \rightarrow \overline{K_1}\Sigma_{1u} \cap L_1 \subseteq \overline{K_1} \wedge \overline{K_2}\Sigma_{2u} \cap L_2 \subseteq \overline{K_2}$:

a) $\overline{K}\Sigma_u \cap L \subseteq \overline{K} \leftrightarrow \forall s \in K, \bar{s}\Sigma_u \cap L \subseteq \overline{K} \rightarrow \forall s \in K, \bar{s}\Sigma_{1u} \cap L \subseteq \overline{K} \leftrightarrow$

$\forall t \in \overline{K}, (t\Sigma_{1u} \subset L(G) \rightarrow t\Sigma_{1u} \subset \overline{K} \rightarrow P_1(t)\Sigma_{1u} \subset \overline{K_1}) \rightarrow$

$\forall s \in K, (\bar{s}\Sigma_{1u} \subset L(G) \rightarrow \overline{P_1(s)}\Sigma_{1u} \subset \overline{K_1})$.

Por outro lado, para qualquer cadeia $u \in K$, considerando $u_1 = P_1(u)$ e $u_2 = P_2(u)$, $u_i \in K_i \subset L_i$, temos que $\overline{u_1}\Sigma_{1u} \subset L_1 \rightarrow \bar{u}\Sigma_{1u} \subset \overline{(u_1 \|_{as} u_2)}\Sigma_{1u} = (\overline{u_1} \|_{as} \overline{u_2})\Sigma_{1u} \subset \overline{u_1}\Sigma_{1u} \|_{as} \overline{u_2} \subset \overline{u_1}\Sigma_{1u} \|_{as} L_2 \subset L_1 \|_{as} L_2 = L$.

Como $P_1(K) = K_1$, qualquer elemento u_1 de K_1 pode ser obtido pela projeção em Σ_1 de alguma cadeia $u \in K$. Assim, pela demonstração acima:

$\overline{K\Sigma_u} \cap L \subseteq \overline{K} \rightarrow \forall u_1 \in K_1,\ \overline{u_1\Sigma_{1u}} \subset L_1 \rightarrow \overline{u\Sigma_{1u}} \subset L \rightarrow \overline{P_1(u)\Sigma_{1u}} \subset \overline{K_1} \leftrightarrow \overline{u_1\Sigma_{1u}} \subset \overline{K_1}$. Isso significa que: $\overline{K\Sigma_u} \cap L \subseteq \overline{K} \rightarrow \overline{K_1\Sigma_{1u}} \cap L_1 \subseteq \overline{K_1}$.

b) Por simetria, $\overline{K\Sigma_u} \cap L \subseteq \overline{K} \rightarrow \overline{K_2\Sigma_{2u}} \cap L_2 \subseteq \overline{K_2}$. ♦

Lema 5: Sejam $L_1 \subseteq \Sigma_1^*$ e $L_2 \subseteq \Sigma_2^*$, com $\Sigma_1 \cap \Sigma_2 = \varnothing$, e as projeções naturais $P_i:(\Sigma_1 \cup \Sigma_2) \rightarrow \Sigma_i$, $i=1,2$. Se a linguagem $K \subset \Sigma^*$ é controlável e.r.a $L_1 \|_{as} L_2$, então $P_1(K)$ é controlável e.r.a L_1 e $P_2(K)$ é controlável e.r.a L_2.

Prova:

i) Mostrar-se-á que $\overline{P_1(K)}\Sigma_{1u} \cap L_1 \subset \overline{P_1(K)}$:

Sabendo-se que $\overline{P_1(K)} = P_1(\overline{K})$, a relação acima pode ser escrita como $P_1(\overline{K})\Sigma_{1u} \cap L_1 \subset P_1(\overline{K})$.

Supõe-se que $P_1(K)$ não seja controlável e.r.a L_1. Então, existem $\sigma_1' \in \Sigma_{1u}$ e $s \in \overline{K}$ tais que $P_1(s)\sigma_1' \in L_1$ e $P_1(s)\sigma_1' \notin P_1(\overline{K})$. Mas, daí, $s\sigma_1' \in (P_1(s) \|_{as} P_2(s))\sigma_1' \subset P_1(s)\sigma_1' \|_{as} P_2(s) \subset L_1 \|_{as} L_2$ e $s\sigma_1' \notin \overline{K}$, pois $s\sigma_1' \in \overline{K} \rightarrow P_1(s\sigma_1') \in P_1(\overline{K}) \rightarrow P_1(s)\sigma_1' \in P_1(\overline{K})$. Mas isso quer dizer que existem $\sigma_1' \in \Sigma_u$ e $s \in \overline{K}$ tais que $s\sigma_1' \cap L_1 \|_{as} L_2 \notin \overline{K}$, i.e., K não é controlável e.r.a $L_1 \|_{as} L_2$, e o resultado é provado por contradição.

ii) Por simetria, $\overline{P_2(K)}\Sigma_{2u} \cap L_2 \subset \overline{P_2(K)}$. ♦

Teorema 3: Sejam K_1, $L_1 \subseteq \Sigma_1^*$ e K_2, $L_2 \subseteq \Sigma_2^*$, com $\Sigma_1 \cap \Sigma_2 = \varnothing$. Então, $SupC(K_1, L_1) \|_{as} SupC(K_2, L_2) = SupC(K_1 \|_{as} K_2, L_1 \|_{as} L_2)$.

Prova:

i) $SupC(K_1, L_1) \|_{as} SupC(K_2, L_2) \subseteq SupC(K_1 \|_{as} K_2, L_1 \|_{as} L_2)$:

Como $SupC(K_1, L_1)$ é controlável e.r.a L_1 e $SupC(K_2, L_2)$ é controlável e.r.a L_2, pelo Lema 4 é sabido que $SupC(K_1, L_1) \|_{as} SupC(K_2, L_2)$ é controlável e.r.a $L_1 \|_{as} L_2$. Portanto, $SupC(K_1, L_1) \|_{as} SupC(K_2, L_2) \subseteq SupC(K_1 \|_{as} K_2, L_1 \|_{as} L_2)$.

ii) $SupC(K_1 \|_{as} K_2, L_1 \|_{as} L_2) \subseteq SupC(K_1, L_1) \|_{as} SupC(K_2, L_2)$:

Seja $K' = SupC(K_1 \|_{as} K_2, L_1 \|_{as} L_2)$. Pelo Lema 5, $P_1(K')$ é controlável e.r.a L_1 e $P_2(K')$ é controlável e.r.a L_2. Isso implica que $P_i(K') \subseteq SupC(K_i, L_i)$, $i=1,2$. Então, $SupC(K_1 \|_{as} K_2, L_1 \|_{as} L_2) = K' \subseteq P_1(K') \|_{as} P_2(K') \subseteq SupC(K_1, L_1) \|_{as} SupC(K_2, L_2)$. ♦

O resultado anterior indica uma forma alternativa de se fazer a síntese de controladores para Sistemas Produtos sem passar pela composição de todos os subsistemas com cada especificação. Um primeiro resultado intuitivo é o de que os subsistemas que não são restringidos por nenhuma especificação não afetam a controlabilidade do sistema. Esse resultado é expresso pelo Corolário 4.

Corolário 4: Dado um Sistema Composto G_i, $i = 1,...,n$ e as especificações E_a e E_b, sejam K_{Ae}, K_{Be}, K_A, K_B, G_e, G_e^c e G definidos como na seção anterior. Então, $SupC(K_{Ae} \cap K_{Be}, L_m(G_e)) \|_{as} L_m(G_e^c) = SupC(K_A \cap K_B, L_m(G))$.

Finalmente, o principal resultado da seção pode ser apresentado como segue.

Teorema 4: Dado um Sistema Composto G com subplantas G_i, $i = 1,...,n$ e as especificações E_a e E_b, sejam E_A, E_B, K_{Ae}, K_{Be}, G_A, G_B, G_A^{ce}, G_B^{ce} e G_e definidos como na seção anterior. Se $SupC(E_A, L_m(G_A))$ e $SupC(E_B, L_m(G_B))$ são localmente modulares, então, $SupC(K_{Ae} \cap K_{Be}, L_m(G_e)) = SupC(E_A, L_m(G_A)) \| SupC(E_B, L_m(G_B))$.

Prova:

Como $L_m(G_A) = L_m(G_A) \| L_m(G_B^{ce})$ e $SupC(E_A, L_m(G_A)) = SupC(E_A, L_m(G_A)) \| L_m(G_A)$, é verdade que $SupC(E_A, L_m(G_A)) = SupC(E_A, L_m(G_A)) \| L_m(G_B^{ce})$. Da mesma forma, pode-se dizer que $SupC(E_B, L_m(G_B)) = SupC(E_B, L_m(G_B)) \| L_m(G_A^{ce})$. Daí, $SupC(E_A, L_m(G_A)) \| SupC(E_B, L_m(G_B)) = (SupC(E_A, L_m(G_A)) \| L_m(G_B^{ce})) \| (SupC(E_B, L_m(G_B)) \| L_m(G_A^{ce})) = (SupC(E_A, L_m(G_A)) \| L_m(G_A^{ce})) \| (SupC(E_B, L_m(G_B)) \| L_m(G_B^{ce}))$.

Por outro lado, como $\Sigma_A \cap \Sigma_A^{ce} = \varnothing$ e $L_m(G_A^{ce}) = SupC(L_m(G_A^{ce}), L_m(G_A^{ce}))$, o teorema anterior assegura que $SupC(E_A, L_m(G_A)) \| L_m(G_A^{ce}) = SupC(E_A \| L_m(G_A^{ce}), L_m(G_A) \| L_m(G_A^{ce})) = SupC(K_{Ae}, L_m(G_e))$. Do mesmo modo, pode-se provar que $SupC(E_B, L_m(G_B)) \| L_m(G_B^{ce}) = SupC(K_{Be}, L_m(G_e))$. Portanto, $SupC(E_A, L_m(G_A)) \| SupC(E_B, L_m(G_B)) = (SupC(E_A, L_m(G_A)) \| L_m(G_A^{ce})) \| (SupC(E_B, L_m(G_B)) \| L_m(G_B^{ce})) = SupC(K_{Ae}, L_m(G_e)) \| SupC(K_{Be}, L_m(G_e)) = SupC(K_{Ae}, L_m(G_e)) \cap SupC(K_{Be}, L_m(G_e))$.

Como $SupC(E_A, L_m(G_A))$ e $SupC(E_B, L_m(G_B))$ são localmente modulares, Lema 2 (combinado com o Teorema 4) garante que $SupC\ K_{Ae}, L_m(G_e))$ e $SupC(K_{Be}, L_m(G_e))$ são modulares. Então, pelo Teorema 3, $SupC(E_A, L_m(G_A)) \parallel SupC(E_B, L_m(G_B)) = SupC(K_{Ae}, L_m(G_e)) \cap SupC(K_{Be}, L_m(G_e)) = SupC(K_{Ae} \cap K_{Be}, L_m(G_e))$. ♦

O Teorema 4 fundamenta uma nova abordagem para a síntese de controladores modulares para sistemas compostos. Dadas duas especificações genéricas E_a e E_b sobre um sistema produto, é necessário apenas expressá-las em termos dos subsistemas G_A e G_B afetados por elas. Calculam-se assim as máximas linguagens controláveis contidas nas mesmas $SupC(E_A, L_m(G_A))$ e $SupC(E_B, L_m(G_B))$, para as quais são gerados controladores locais correspondentes. A condição de modularidade local dessas duas linguagens, pelo teorema anterior, garante que a ação conjunta dos supervisores locais resultantes é ótima, isto é, não resulta em nenhuma perda de performance em relação ao controle centralizado.

4.2.3 *Análise de Complexidade*

Para avaliar o desempenho da metodologia de síntese de controladores apresentada nesta seção, é feita uma comparação entre a complexidade computacional desta abordagem, e as abordagens modular clássica e monolítica. Supõe-se um sistema composto formado por $p + e$ subsistemas com n estados cada. Assim, o comportamento global do sistema pode ser representado por um gerador G, com n^{p+e} estados. Sejam duas especificações genéricas E_a e E_b modeladas por autômatos de l estados, restringindo respectivamente k_a e k_b dos $p + e$ subsistemas de forma que e subsistemas não são afetados por nenhuma das especificações. Seja ainda $k = max(k_a, k_b)$.

Conforme as definições para sistemas compostos, feitas na Seção 4.1, a análise de complexidade considera geradores para a planta enxuta G_e, com n^p estados, para as plantas locais G_A, com n^{ka} estados e G_B, com n^{kb} estados e autômatos descrevendo as especificações locais E_A e E_B, com respectivamente ln^{ka} e ln^{kb} estados. As especificações ainda podem ser expressas em termos da planta enxuta como K_e, representada por um autômato com l^2n^p estados, ou isoladamente como K_{Ae} e K_{Be}, cujos modelos de autômatos têm ln^p estados cada.

Considerando apenas os fatores relativos ao número de estados, como em [26], numa análise de pior caso, a verificação da modularidade entre K_{Ae} e K_{Be} tem

complexidade $O(l^2n^{2p})$. Entretanto, a complexidade da verificação da modularidade local entre E_A e E_B é de ordem $O(l^2n^{ka + kb})$, o que pode representar um ganho computacional considerável.

A complexidade do processo de síntese de um controle centralizado $SupC(K_e, L_m(G_e))$ é de ordem $O(l^4n^{4p})$. Já a síntese clássica de controladores modulares $SupC(K_{Ae}, L_m(G_e))$ e $SupC(K_{Be}, L_m(G_e))$ tem complexidade $O(l^2n^{4p})$. Pode-se perceber que o ganho do controle modular não é tão grande em sistemas onde $l << n^p$, ou seja, em sistemas de grande porte onde as especificações genéricas possam ser representadas por geradores de poucos estados.

Por outro lado, os cálculos de $SupC(E_A, L_m(G_A))$ e de $SupC(E_B, L_m(G_B))$, como propostos nesta seção, têm complexidades $O(l^2n^{4ka})$ *e* $O(l^2n^{4kb})$, respectivamente. Além disso, a verificação da modularidade local entre os dois resultados é, no pior dos casos, uma operação de complexidade computacional de ordem $O(l^2n^{ka + kb})$. Portanto, a síntese de controladores localmente modulares para sistemas compostos tem complexidade $O(l^2n^{4k})$. Como $p/2 \leq k \leq p$, esta abordagem terá sempre complexidade computacional menor ou igual à abordagem modular clássica.

4.2.4 Exemplo

Para exemplificar a metodologia proposta nesta seção, sintetizam-se controladores locais para o exemplo de "linha de transferência industrial" apresentado na seção 3.1.4.

Como o alfabeto das especificações genéricas contém eventos de G_i e G_{i+1} e como as máquinas não têm eventos em comum, as plantas locais para E_a e E_b são dadas respectivamente por $G_A = G_1 \parallel G_2$ e $G_B = G_2 \parallel G_3$ e seus comportamentos são ilustrados pelo autômato da figura a seguir.

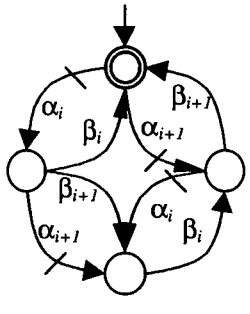


Fig. 12:. *G_X, $X = A(i=1), B(i=2)$*

Assim, as especificações B_i podem ser expressas em termos das plantas locais G_X, $X= A,B$, por $E_X = E_x||L_m(G_X)$ (ver Figura 13). Com isso, as máximas linguagens controláveis contidas em E_X, isto é, $SupC(E_X, G_X)$, podem ser calculadas (ver figura 14).

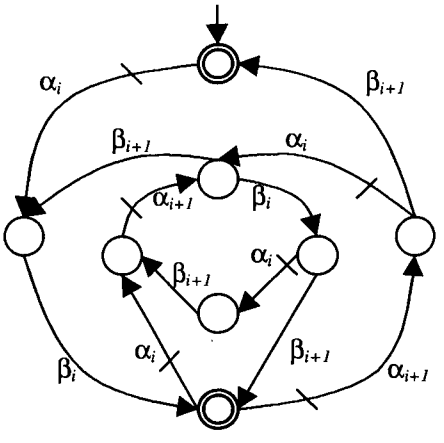


Fig. 13:. Autômato para E_X, $X = A(i=1), B(i=2)$

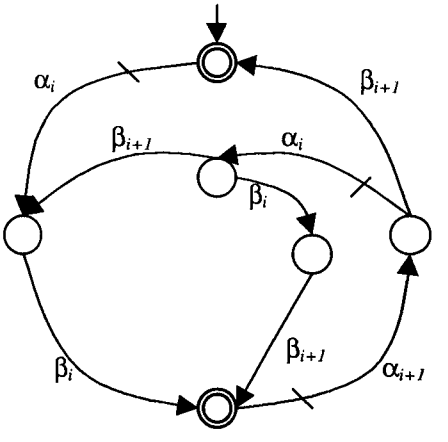


Fig. 14:. Autômato para $SupC(E_X, G_X)$, $X = A(i=1), B(i=2)$

Por meio de ferramenta computacional adequada, pode-se verificar que as especificações controláveis são localmente modulares, isto é, pode-se calcular e verificar a igualdade de $\overline{SupC(E_A, G_A)} \,||\, \overline{SupC(E_B, G_B)}$ e $\overline{SupC(E_A, G_A) \,||\, SupC(E_B, G_B)}$. Pelo Teorema 4, a modularidade garante que não há qualquer perda de eficiência entre esta solução modular apresentada e a melhor solução centralizada.

De acordo com a discussão desenvolvida na Seção 4.2.3, a complexidade computacional desse procedimento de síntese é de ordem $O(2^{10})$. Por outro lado, pela abordagem modular clássica, esse cálculo teria ordem $O(2^{14})$, enquanto a síntese de um supervisor monolítico teria complexidade de ordem $O(2^{16})$.

4.3 Controle Modular Local para Múltiplas Especificações

4.3.1 Modularidade Local

Da mesma forma que para problemas de síntese de controladores para duas especificações, é esperado que a adição ou subtração de um sistema assíncrono G_e^c a todas especificações altere de um modo equivalente todos os comportamentos do ponto de vista da modularidade. Esse resultado, generalizado do Teorema 1, é apresentado no teorema abaixo.

Teorema 5: Seja I um conjunto de índices. Para $K_{Xie} \subseteq \Sigma_e^*$, i ∈ I, e $K_{Xi} \subseteq \Sigma^*$, i ∈ I, definidos como anteriormente, $\{K_{Xi}, i \in I\}$ é modular se e somente se $\{K_{Xie}, i \in I\}$ é modular, isto é:

$$\bigcap_{i\in I} \overline{K_{Xi}} = \overline{\bigcap_{i\in I} K_{Xi}} \Leftrightarrow \bigcap_{i\in I} \overline{K_{Xie}} = \overline{\bigcap_{i\in I} K_{Xie}}\,.$$

Prova: Semelhante à demonstração do Teorema 1.

O conceito de modularidade local introduzido na Seção 4.2.1 pode ser estendido da seguinte forma. Seja I um conjunto de índices. Sejam $L_i \subseteq \Sigma_i^*$, $i \in I$. Diz-se que o conjunto $\{L_i, i \in I\}$ é *localmente modular* sempre que $\|_{i\in I} \overline{L_i} = \overline{\|_{i\in I} L_i}$, ou seja, $\bigcap_{i\in I} \overline{P_i^{-1}(L_i)} = \overline{\bigcap_{i\in I} P_i^{-1}(L_i)}$.

Uma observação importante para a generalização da modularidade é que a modularidade de todos os subconjuntos de linguagens não garante a modularidade do conjunto completo. Por exemplo, sejam as linguagens $L_1=\{\alpha\beta, \gamma\}$, $L_2=\{\alpha\mu, \alpha\mu\mu\beta, \alpha\gamma\}$ e $L_3=\{\mu\}$ definidas respectivamente em $\Sigma_1=\{\alpha, \beta, \gamma\}$, $\Sigma_2=\{\alpha, \beta, \gamma, \mu\}$ e $\Sigma_3=\{\mu\}$. É fácil verificar que, embora $\{L_1, L_2\}$, $\{L_2, L_3\}$ e $\{L_1, L_3\}$ sejam localmente modulares, o conjunto $\{L_1, L_2, L_3\}$ não o é. Essa afirmação implica que a verificação da modularidade nem sempre pode ser decomposta e requerer, portanto, o cálculo da composição do conjunto completo de linguagens.

O teorema seguinte garante para o caso de múltiplas especificações que a modularidade local é verificada se e somente se o conjunto completo de especificações é modular.

Teorema 6: Seja I um conjunto de índices. Para $E_{Xi} \subseteq \Sigma_{Xi}^*$, i∈ I, e $K_{Xi} \subseteq \Sigma^*$, i∈ I definidos como anteriormente, $\{K_{Xi}, i\in I\}$ é modular se e somente se $\{E_{Xi}, i\in I\}$ é localmente modular, ou seja:

$$\bigcap_{i\in I} \overline{K_{X_i}} = \overline{\bigcap_{i\in I} K_{X_i}} \Leftrightarrow \bigcap_{i\in I} \overline{P_{X_i}^{-1}(E_{X_i})} = \overline{\bigcap_{i\in I} P_{X_i}^{-1}(E_{X_i})}\,.$$

A prova do Teorema 6 é baseada nos seguintes lemas.

Lema 6: Sejam $L_1 \subset \Sigma_1^*$ e $L_2 \subset L_3 \subset \Sigma_2^*$. Então $\overline{L_1 \| L_2} \| \overline{L_3} = \overline{L_1 \| L_2}$.

Prova:

Seja P_2: $\Sigma^* \rightarrow \Sigma_2^*$, com $\Sigma = \Sigma_1 \cup \Sigma_2$.

1) $\overline{L_1 \| L_2} = \overline{L_1 \| L_2 \| L_3} = \overline{(L_1 \| L_2) \cap P_2^{-1}(L_3)} \subseteq \overline{L_1 \| L_2} \cap \overline{P_2^{-1}(L_3)} = \overline{L_1 \| L_2} \cap P_2^{-1}(\overline{L_3}) = \overline{L_1 \| L_2} \| \overline{L_3}$.

2) $\overline{L_1 \| L_2} \| \overline{L_3} = \overline{L_1 \| L_2} \cap P_2^{-1}(\overline{L_3}) \subseteq \overline{L_1 \| L_2}$. ♦

Lema 7: Seja I um conjunto de índices. Seja a RSP de um sistema G dada por subplantas G_j, $j \in N = \{1, ..., n\}$. Dadas as especificações E_{xi}, $i \in I$, define-se, para cada $i \in I$, E_{Xi}, K_{Xie}, G_{Xi} e G_e como na Seção 4.1. Então, $\bigcap_{i \in I} K_{Xie} = \bigcap_{i \in I} P_{Xi}^{-1}(E_{Xi})$.

Prova:

$$\bigcap_{i \in I} K_{Xie} = \|_{i \in I} K_{Xie} = \|_{i \in I} (E_{xi} \| L_m(G_e)) = (\|_{i \in I} E_{xi}) \| L_m(G_e) = (\|_{i \in I} E_{xi}) \| (\|_{i \in I} L_m(G_{Xi})) = \|_{i \in I} (E_{xi} \| L_m(G_{Xi})) = \|_{i \in I} E_{Xi} = \bigcap_{i \in I} P_{Xi}^{-1}(E_{Xi}).$$ ♦

Lema 8: Seja I um conjunto de índices. Seja a RSP de um sistema G dada por subplantas G_j, $j \in N = \{1, ..., n\}$. Dadas as especificações E_{xi}, $i \in I$, define-se, para cada $i \in I$, E_{Xi}, K_{Xie}, G_{Xi}, G_{Xi}^{ce} e G_e como na Seção 4.1. Então, $\bigcap_{i \in I} \overline{K_{Xie}} = \|_{i \in I} \overline{E_{Xi}}$.

Prova:

$$\bigcap_{i \in I} \overline{K_{Xie}} = \|_{i \in I} \overline{K_{Xie}} = \|_{i \in I} \overline{(E_{Xi} \| L_m(G_{Xi}^{ce}))} = \|_{i \in I} (\overline{E_{Xi}} \| \overline{L_m(G_{Xi}^{ce})}) = (\|_{i \in I} \overline{E_{Xi}}) \| (\|_{i \in I} \overline{L_m(G_{Xi}^{ce})}) = (\|_{i \in I} \overline{E_{Xi}}) \| (\|_{i \in I} (\overline{\|_{j \in N_e - N_{Xi}} L_m(G_j)})) = (\|_{i \in I} \overline{E_{Xi}}) \| (\|_{i \in I} (\overline{\|_{j \in N_e - N_{Xi}} L_m(G_j)})) = (\|_{i \in I} \overline{E_{xi} \| \|_{k \in N_{Xi}} L_m(G_k)}) \| (\|_{j \in N_e - \bigcap_{i \in I} N_{Xi}} \overline{L_m(G_j)}) =^* \|_{i \in I} \overline{E_{xi} \| \|_{k \in I_{Xi}} L_m(G_k)} = \|_{i \in I} \overline{E_{Xi}}.$$

* como $N_e - \bigcap_{i \in I} N_{Xi} \subset N_e$, cada $i \in N_e - \bigcap_{i \in I} N_{Xi}$ está contido em algum N_{Xi}, $i \in I$, de forma que Lema 4 pode ser aplicado sucessivamente à expressão acima. ♦

Prova do Teorema 6: Segue diretamente dos Lemas 6 e 7 e do Teorema 5. ♦

Assim, a modularidade local pode ser considerada como uma extensão da propriedade de modularidade para conjuntos de linguagens cujos alfabetos são distintos. No caso de Sistemas Produtos, a verificação da modularidade local entre especificações locais evita a composição de especificações globais, cujas representações por autômatos costumam ter números de estados de ordem muito elevada para sistemas de grande porte.

4.3.2 Controle Modular Local

O resultado a seguir, generalizado do Teorema 4, garante que a síntese da máxima linguagem controlável para um conjunto de especificações locais não apresente perda de performance em relação a uma solução monolítica, sempre que o conjunto de linguagens resultantes for localmente modular.

Teorema 7: Seja $I = \{1, ..., m\}$ um conjunto de índices. Seja uma RSP de um sistema G dada por subplantas G_j, $j \in N = \{1, ..., n\}$. Sejam as especificações genéricas E_{Xi}, $i \in I$, e , para cada $i \in I$, as linguagens E_{Xi} e K_{Xie} e os geradores G_{Xi} e G_e definidos como na Seção 4.1. Se o conjunto $\{SupC(E_{Xi}, L_m(G_{Xi})), i \in I\}$ for localmente modular, então, $SupC(\bigcap_{i=1,...,m} K_{Xie}, L_m(G_e)) = \|_{i=1,...,m} SupC(E_{Xi}, L_m(G_{Xi}))$.

Prova: (Indução sobre m.)

1) ($m=1$): $G_{X1} = G_e$, $K_{X1e} = E_{X1} \Rightarrow SupC(K_{X1e}, L_m(G_e)) = SupC(E_{X1}, L_m(G_{X1}))$;

2) ($m=2$): Teorema 4 $\Rightarrow SupC(K_{X1e} \cap K_{X2e}, L_m(G_e)) = SupC(E_{X1}, L_m(G_{X1})) \| SupC(E_{X2}, L_m(G_{X2}))$;

3) ($m=k$): Supõe-se que $SupC(\bigcap_{i=1,...,k} K_{Xie}, L_m(G_e)) = \|_{i=1,...,k} SupC(E_{Xi}, L_m(G_{Xi}))$;

4) ($m=k+1$): $SupC(\bigcap_{i=1,...,k+1} K_{Xie}, L_m(G_e)) = SupC((\bigcap_{i=1,...,k} K_{Xie}) \cap K_{Xk+1e}, L_m(G_e))$ = (Pelo Teorema 4) $SupC((\bigcap_{i=1,...,k} K_{Xie}), L_m(\|_{i=1,k} G_{Xi})) \| SupC(E_{Xk+1}, L_m(G_{Xk+1}))$ = (Pela suposição anterior) $(\|_{i=1,...,k} SupC(E_{Xi}, L_m(G_{Xi}))) \| SupC(E_{Xk+1}, L_m(G_{Xk+1})) = \|_{i=1,...,k+1} SupC(E_{Xi}, L_m(G_{Xi}))$. ♦

Os resultados desta seção mostram que a síntese de supervisores modulares pode ser feita ao nível das plantas locais, indicando um modo mais eficiente de verificação da condição de não bloqueio.

4.3.3 *Análise de Complexidade*

Da mesma forma que na Seção 4.2.3, esta análise de complexidade considera um sistema composto formado por $p + e$ subsistemas com n estados cada. Assim, o comportamento global do sistema pode ser representado por um gerador G, com n^{p+e} estados. Sejam as especificações genéricas E_{xi}, $i = 1,...,q$, modeladas por autômatos de m estados, cada uma restringindo k dos $p + e$ subsistemas de forma que e subsistemas não são afetados por nenhuma das especificações.

Conforme as definições da Seção 4.1, são considerados geradores para a planta enxuta G_e, com n^p estados, para as plantas locais G_{Xi}, $i = 1,...,q$, com n^k estados, e autômatos descrevendo as especificações locais E_{Xi}, $i = 1,...,q$, com mn^k estados. As especificações ainda podem ser expressas em termos da planta enxuta como K_e, representada por um autômato com $m^q n^p$ estados, ou isoladamente como K_{Xie}, $i = 1,...,m$, cujos modelos de autômatos têm mn^p estados cada.

De acordo com [26], numa análise de pior caso, a verificação da modularidade de $\{K_{Xie}, i = 1,...,q\}$ tem complexidade $O(m^q n^{qp})$. Entretanto, a complexidade da verificação da modularidade local de $\{E_{Xi}, i = 1,...,m\}$ é de ordem $O(m^q n^{qk})$, o que pode representar um ganho computacional considerável, pois o fator k, ao contrário de p, não cresce necessariamente com o porte do sistema.

A complexidade do processo de síntese de um controle centralizado $SupC(K_e, L_m(G_e))$ é de ordem $O(m^{2q} n^{4p})$. Já o cálculo de $\{SupC(K_{Xie}, L_m(G_e)), i = 1,...,q\}$ tem complexidade $O(m^2 n^{4p})$. Assim, a complexidade da síntese clássica de controladores modulares é de ordem $O(m^2 n^{4p} + m^q n^{qp})$. Assumindo que $m = n$, pode ser mostrado que o ganho de complexidade do controle modular clássico em relação ao monolítico é positivo apenas para $q < 4p/(p-1) \approx 4$. Isso implica que, numa análise de pior caso, quando as especificações genéricas podem ser representadas por autômatos do mesmo porte das subplantas ($m \approx n$), a abordagem modular clássica só é computacionalmente vantajosa para um pequeno número q de especificações.

Por outro lado, o cálculo de $\{SupC(E_{Xi}, L_m(G_{Xi})), i = 1,...,q\}$, como proposto nesta seção, tem complexidade de ordem $O(m^2 n^{4k})$. Além disso, a verificação da modularidade local para o conjunto de resultados é, no pior dos casos, uma operação de complexidade

computacional de ordem $O(m^q n^{qk})$. Portanto, a síntese de múltiplos controladores localmente modulares para sistemas compostos tem complexidade $O(m^2 n^{4k} + m^q n^{qk})$. Como $p/q \leq k \leq p$, esta abordagem terá sempre complexidade computacional menor ou igual à abordagem modular clássica. É fácil verificar que, para $k < 4p/q$, a síntese modular local oferece uma significativa redução na complexidade computacional em relação à abordagem monolítica.

4.3.4 Exemplo

O controle modular local para múltiplas especificações é ilustrado por uma "linha de transferência industrial" composta por seis máquinas M_1, M_2, M_3, M_4, M_5, e M_6 ligadas por *buffers* B_A, B_B, B_C e B_D de capacidade 1, trabalhando de acordo com a Figura 15. Como no exemplo da Seção 3.1.4, desejam-se sintetizar supervisores que garantam a não ocorrência de *underflow* e *overflow* em todos os *buffers*.

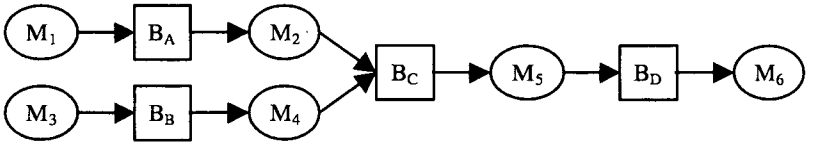
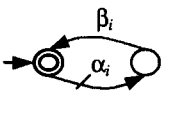


***Fig. 15:**. Linha de Transferência Industrial*

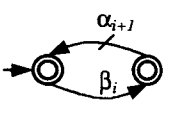
O comportamento em malha aberta G_i de cada máquina M_i, i = 1,...,6 é representado pelo gerador da Figura 16a. As especificações genéricas E_a para B_A, E_b para B_B e E_d para B_D podem ser descritas pelo autômato da Figura 16b. Do mesmo modo, a especificação E_c para o *buffer* B_C é ilustrada na Figura 16c. Consideram-se os eventos α_i como o início de operação da máquina M_i, com a retirada de uma peça de B_{i-1} (quando existir), e β_i como o fim de operação da máquina M_i com o depósito de uma peça em B_i (quando existir).

Como o alfabeto das especificações genéricas contém eventos dos subsistemas anteriores e posteriores e como os subsistemas não têm eventos em comum, as plantas locais a E_a, E_b, E_c e E_d são dadas respectivamente por $G_A = G_1 \| G_2$, $G_B = G_3 \| G_4$, $G_C = G_2 \| G_4 \| G_5$ e $G_D = G_5 \| G_6$.

a) G_i, $i = 1,...,6$:



b) $E_a(i=1)$, $E_b(i=3)$ e $E_d(i=5)$:



c) E_c :

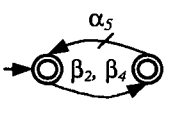
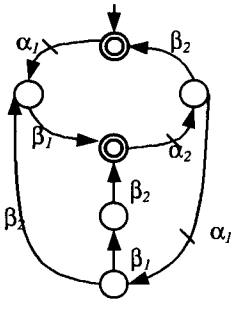


***Fig. 16:**. Autômatos para: a) G_i, $i = 1,...,6$; b) E_a, E_b e E_d; c) E_c.*

Assim, as especificações E_x, x = *a, b, c* e *d*, podem ser expressas em termos das plantas locais G_X, X= *A, B, C* e *D,* por $E_X = E_x \| L_m(G_X)$. Com isso, as máximas linguagens controláveis contidas em E_X, isto é, $SupC(E_X, G_X)$, podem ser calculadas (ver figuras 17a e 17b).

a) $SupC(E_X, G_X)$, $X = A(i=1)$, $B(i=3)$ *e* $D(i=5)$:



b) $SupC(E_C, G_C)$:

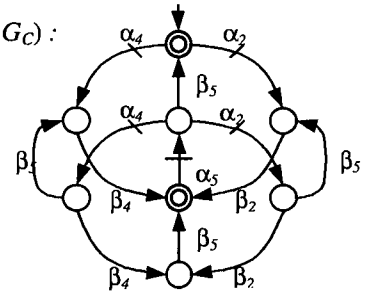


Fig. 17:. *Autômatos para: a)* $SupC(E_X, G_X)$, X = A, B *e* D; *b)* $SupC(E_C, G_C)$.

Por meio de uma ferramenta computacional adequada[1], pode-se verificar que as especificações controláveis são localmente modulares, isto é, pode-se calcular e verificar a igualdade de $\|_{X=A,B,C,D} \overline{SupC(E_X, G_X)}$ e $\overline{\|_{X=A,B,C,D} SupC(E_X, G_X)}$. Pelo Teorema 7, a modularidade garante que não há nenhuma perda de eficiência entre esta solução modular apresentada e a melhor solução centralizada.

De acordo com a discussão desenvolvida na Seção 4.3.3, a complexidade computacional desse procedimento de síntese é de ordem $O(10^4)$. Por outro lado, pela abordagem modular clássica, esse cálculo teria ordem $O(10^7)$, enquanto a síntese de um supervisor monolítico teria complexidade de ordem $O(10^9)$.

4.3.5 Interação de Múltiplos Controladores

A interação de múltiplos controladores introduz novas questões em termos da flexibilidade do sistema. Como visto anteriormente, mesmo que todas as interações parciais de supervisores sejam não conflitantes, o comportamento global em malha fechada pode ser bloqueante. Em muitos desses casos, o bloqueio pode ser solucionado pela desativação de um supervisor local.

Por outro lado, pode haver um grupo de supervisores parcialmente conflitantes, cujo comportamento global em malha fechada é não bloqueante. Nesse caso, desativar um

[1]Nesta pesquisa, utilizou-se o programa TCT, desenvolvido e cedido pelo Systems Control Group do Departamento de Engenharia Elétrica da Universidade de Toronto, Canadá.

controlador modular pode induzir um bloqueio no sistema. Por exemplo, sejam as linguagens $L_1=\{\alpha\beta, \gamma\}$, $L_2=\{\mu, \mu\mu\beta, \mu\mu\alpha\gamma\}$ e $L_3=\{\mu\}$ definidas respectivamente em $\Sigma_1=\{\alpha, \beta, \gamma\}$, $\Sigma_2=\{\alpha, \beta, \gamma, \mu\}$ e $\Sigma_3=\{\mu\}$. Pode ser mostrado que L_1 e L_2 são conflitantes enquanto o conjunto $\{ L_1, L_2, L_3\}$ é localmente modular. Então, se supervisores para L_1 e L_2 são ativados e um supervisor que atenda a L_3 é desativado, o sistema pode bloquear-se.

Portanto, a modularidade de múltiplos controladores não garante flexibilidade, a menos que essa propriedade seja verificada para todas as possíveis combinações de subconjuntos de controladores. Além do mais, essa particularidade instiga a propor que a anexação de supervisores locais próprios, desativando linguagens bloqueantes, pode resolver o problema de conflito global.

4.4 Resolução de conflitos

É importante ressaltar que, quando a modularidade das especificações não é verificada, o controle modular não pode ser diretamente aplicado. Nesse caso, podem-se utilizar diferentes abordagens para a resolução de conflitos. Uma alternativa evidente é a aplicação de controle centralizado para as especificações conflitantes. Porém, na literatura são encontradas diversas abordagens distintas, como é o caso da abordagem de controle-modular-e-coordenação apresentada por [24], do controle modular com prioridades [4] e do esquema para resolução de conflitos apresentado em [23].

Enfim, quando o controle modular é aplicável, a abordagem apresentada neste capítulo explora, além da modularidade das especificações, aspectos de modularidade da planta. Ao contrário do controle descentralizado, em que a estrutura de informações é imposta pela planta, a abordagem de controle modular local induz a uma estrutura de controle descentralizada que surge naturalmente do processo de síntese.

Com isso, o principal resultado da presente dissertação alia as vantagens da abordagem modular aos benefícios do controle descentralizado. Assim, a abordagem proposta nesta dissertação permite uma redução da complexidade computacional tanto no processo de síntese quanto no funcionamento da estrutura de controle, uma vez que os controladores locais consideram apenas o comportamento dos subsistemas afetados.

Entretanto, apesar de mais reduzida, a complexidade do processo de verificação da modularidade local apresentado cresce exponencialmente com o número de tarefas de controle o que aponta ainda para a necessidade de métodos mais eficientes.

Finalmente, a síntese de controladores localmente modulares tem aplicação em diversos sistemas de grande porte. O Capítulo 5 ilustra a relevância desta abordagem propondo uma metodologia para a síntese de controle para sistemas de manufatura.

5. APLICAÇÃO A SISTEMAS DE MANUFATURA

Uma das grandes aplicações do controle supervisório de sistemas a eventos discretos é o controle de sistemas de manufatura. Verifica-se na prática que a síntese de supervisores para estes sistemas é feita por processos de tentativa e erro, baseados na experiência e inspiração do projetista. Apesar de haver grande número de ferramentas para a análise de projetos de controle, em sistemas mais complexos a síntese manual de controladores pode ter elevado grau de dificuldade.

Em [13] é apresentada uma abordagem para a síntese de controladores usando redes de Petri controladas. Porém, essa abordagem é aplicável apenas em problemas cujo objetivo de controle possa ser especificado em termos de estados proibidos.

A teoria de Ramadge e Wonham permite a síntese automática de controladores ótimos, no sentido de satisfazer as especificações da maneira menos restritiva possível. Entretanto, a modelagem de sistemas mais complexos por um autômato a estados finitos acaba sendo inviável, tamanho o custo computacional envolvido. Para superar essa dificuldade, neste capítulo é apresentada uma metodologia para a síntese de supervisores para sistemas de manufatura, baseada na abordagem de controle modular local. A metodologia proposta é, então, ilustrada por um problema de controle para um sistema de manufatura hipotético.

Os resultados deste capítulo podem ser encontrados em [8].

5.1 Metodologia Proposta

Os modernos sistemas automatizados de manufatura são caracteristicamente compostos por um grande número de subsistemas responsáveis pela realização de tarefas particulares do processo global. Entre esses subsistemas se encontram tipicamente sistemas de produção, como usinagem, montagem, soldagem, pintura, entre outros; de movimentação, como esteiras e Veículos Auto Guiados (AGVs – *Automated Guided Vehicles*); e de armazenamento de material. O funcionamento em malha aberta dos sistemas discretos de manufatura pode, então, ser modelado como um conjunto de sistemas

a eventos discretos concorrentes. Portanto, é viável a obtenção de uma Representação por Sistema Produto, conforme discutido na Seção 4.1.

Cabe ao controle supervisório a sincronização adequada dos subsistemas de modo que o sistema em malha fechada funcione da forma desejada. Supondo que o funcionamento global desejado possa ser representado por um conjunto de especificações sobre a interação dos diversos subsistemas, propõe-se sintetizar o sistema de controle supervisório segundo a abordagem de controle modular local apresentada no Capítulo 4.

A metodologia proposta para a síntese de supervisores localmente modulares para sistemas automatizados de manufatura pode ser descrita sinteticamente pela seqüência de passos abaixo:

1. modelar o funcionamento em malha aberta de cada subsistema por um gerador da forma mais sintética possível;
2. calcular a mais refinada RSP, fazendo-se a composição dos subsistemas síncronos;
3. modelar cada especificação isoladamente, considerando apenas os eventos relevantes;
4. obter a planta local para cada especificação compondo-se os subsistemas da RSP que tenham eventos em comum com ela;
5. calcular a linguagem de cada planta local que satisfaça a especificação, através do produto síncrono de cada planta local com sua respectiva especificação;
6. calcular a máxima linguagem controlável contida em cada especificação local;
7. verificar a modularidade local das linguagens resultantes;
8. se não forem modulares, procurar resolver o problema de não modularidade por outra abordagem (ver seção 4.4);
9. se forem modulares, implementar um supervisor local para cada linguagem controlável.

5.2 Exemplo de Sistema de Manufatura

Para exemplificar a metodologia proposta neste capítulo, sintetizam-se controladores locais para a automatização do modelo hipotético de Sistema Integrado de Manufatura (SIM) ilustrado na Figura 18. O SIM em questão é formado por três centros de usinagem U_i, i=1,2,3, por dois Veículos Auto Guiados, denominados AGV_1 e AGV_2, e por 2 estações de montagem M_1 e M_2. A função dos AGVs é alimentar a linha de montagem com os componentes produzidos nos centros de usinagem, passando por uma trilha que faz a ligação entre as duas áreas.

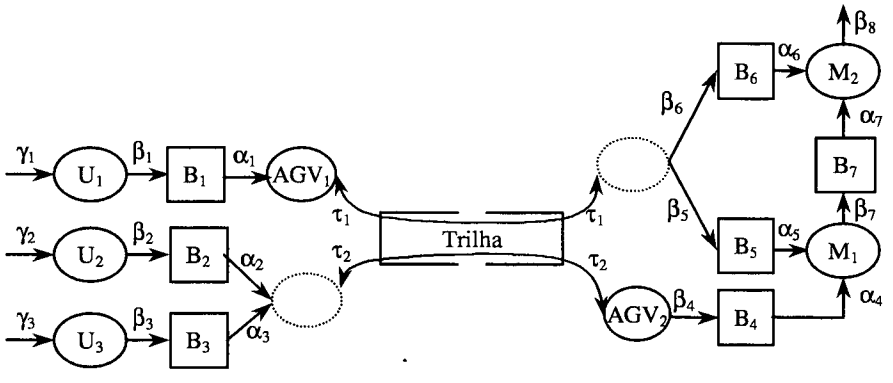


Fig. 18:. Sistema Integrado de Manufatura

Fazendo a interface entre as máquinas do SIM há sempre um sistema unitário de armazenamento temporário (*buffer*), de forma que o SIM do exemplo é composto por 7 *buffers* unitários B_i, i=1,...,7. Deseja-se pelo controle supervisório que não ocorra *overflow* nem *underflow* nos *buffers* unitários e que os dois AGVs não ocupem a trilha ao mesmo tempo.

Num primeiro passo para a síntese de controladores locais deve-se fazer a modelagem dos subsistemas envolvidos. Os centros de usinagem U_i, i=1,2,3, têm suas operações iniciadas respectivamente pelos eventos controláveis γ_i, i=1,2,3, e terminam de operar com o evento não controlável β_i, i=1,2,3, pelo depósito de uma peça no *buffer* seguinte. Assim, as máquinas U_i, i=1,2,3, podem ser modeladas respectivamente pelos autômatos G_i, i=1,2,3, apresentados na Figura 19.

G_i, i = 1,2,3:

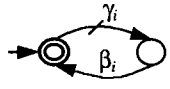
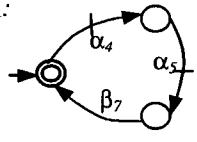


Fig. 19:. Autômatos para U_i, i = 1,2,3

As estações M_1 e M_2 iniciam a montagem respectivamente pelos eventos controláveis α_4 e α_7, os quais retiram uma peça do *buffer* anterior, seguidos da mesma forma por α_5 e α_6, sendo a operação terminada respectivamente por β_7 e β_8. As estações de montagem M_1 e M_2 podem então ser modeladas pelos autômatos G_4 e G_5 ilustrados nas figuras 20a e 20b.

a) G_4:



b) G_5:

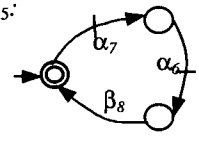
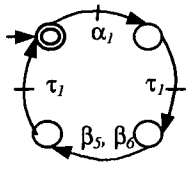


Fig. 20:. Autômatos para: a) M_1; b) M_2.

O AGV_1 desocupa a trilha e retira um componente do *buffer* B_1 através do evento controlável α_1. A seguir, o AGV_1 pode ocupar a trilha pelo evento controlável τ_1, sendo capaz de desocupar a trilha pelos eventos não controláveis β_5 ou β_6, através dos quais deposita-se um componente nos *buffers* B_5 ou B_6, respectivamente. O transportador volta, então, ao estado inicial por τ_1. O comportamento do AGV1 é modelado pelo autômato G_6, mostrado na Figura 21a.

Por outro lado, o AGV_2 pode inicialmente ocupar a trilha pelo evento controlável τ_2. Na seqüência, o AGV_2 desocupa a linha pelos eventos controláveis α_2 ou α_3, através dos quais retira um componente dos *buffers* B_1 ou B_2, respectivamente. Para voltar ao estado inicial, o AGV_2 retoma a trilha por τ_2 e libera-a pelo evento não controlável β_4, depositando uma peça no *buffer* B_4. A Figura 21b ilustra a modelagem desse comportamento pelo autômato G_7.

a) G_6:



b) G_7:

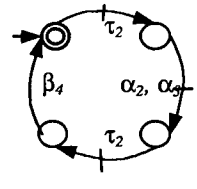


Fig. 21:. Autômatos para: a) AGV_1; b) AGV_2.

As especificações sobre os *buffers* unitários B_i, i=1,...,7, podem ser modeladas respectivamente pelos autômatos E_{xi}, i=1,...,7, representados na Figura 22. O modelo da especificação sobre os AGVs é obtido fazendo-se a composição de seus modelos e retirando-se do autômato resultante os estados em que os dois AGVs ocupam a trilha ao mesmo tempo. Assim, a especificação geral de estados proibidos dos AGVs é dada pelo autômato E_{x8}, o qual é composto por 12 estados.

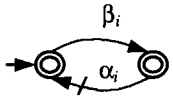


Fig. 22:. Autômatos para E_{xi}, i=1,...,7

Como os subsistemas já estão modelados de forma assíncrona, pode-se observar que o sistema encontra-se na mais refinada RSP. Assim, podem-se obter as plantas locais G_{Xi}, i=1,...,8, respectivas às especificações E_{xi}, i=1,...,8, fazendo-se a composição dos modelos que tenham eventos comuns a elas. Geram-se desta forma os seguintes autômatos: $G_{X1} = G_1 \parallel G_6$; $G_{X2} = G_2 \parallel G_7$; $G_{X3} = G_3 \parallel G_7$; $G_{X4} = G_4 \parallel G_7$; $G_{X5} = G_4 \parallel G_6$; $G_{X6} = G_5 \parallel G_6$; $G_{X7} = G_4 \parallel G_5$ e $G_{X8} = G_6 \parallel G_7$.

Calculam-se, então, as especificações locais E_{Xi}, i=1,...,8, pela composição síncrona das especificações E_{xi}, i=1,...,8, com suas respectivas plantas locais G_{Xi}, i=1,...,8. O passo seguinte é o cálculo das máximas linguagens controláveis $SupC(E_{Xi}, G_{Xi})$, i=1,...,8, contidas nas especificações locais. Para cada uma dessas linguagens resultantes, elabora-se um supervisor que atue na respectiva planta local, ativando ou desativando eventos controláveis, de forma a atender localmente à especificação resultante. Por exemplo, o supervisor que garante o bom funcionamento do *buffer* B_1 observa e controla o funcionamento do centro de usinagem U_1 e do AGV_1, restringindo-o à linguagem $SupC(E_{X1}, G_{X1})$ representada pelo diagrama de transição de estados da figura abaixo.

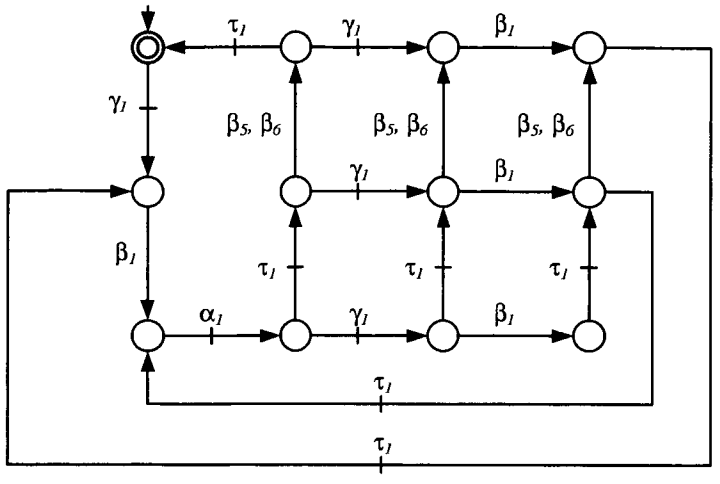


Fig. 23:. Autômato para SupC(E_{X1}, G_{X1})

Para garantir que não exista bloqueio na ação conjunta dos 8 supervisores e que o resultado da ação modular seja o mesmo de uma possível ação centralizada, deve-se verificar a modularidade das linguagens resultantes. Isso é feito calculando-se $\| \overline{SupC(E_{Xi}, G_{Xi})}$ e $\overline{\| SupC(E_{Xi}, G_{Xi})}$, verificando-se, então, a sua igualdade.

Como resultado, são obtidos 8 supervisores locais para a planta com autômatos de no máximo 21 estados. Pela abordagem clássica os geradores para os 8 controladores teriam no mínimo 864 estados. A abordagem monolítica geraria um controlador cujo autômato é formado por 23.490 estados. Essa diferença representa considerável economia de memória e de processamento na execução do controle supervisório.

A complexidade computacional envolvida na síntese de controladores localmente modulares para esse exemplo é de $O(10^5)$, referente à verificação da modularidade. Por outro lado, as complexidades da síntese modular clássica e monolítica para este caso seriam respectivamente da ordem de $O(10^{12})$ e $O(10^{16})$.

A metodologia apresentada neste capítulo é uma alternativa viável para sintetizar supervisores para sistemas de manufatura usando o modelo de Ramadge e Wonham. O exemplo de aplicação mostra que a abordagem proposta neste trabalho permite a solução de problemas de controle de forma modular ao nível das especificações e da planta. Assim, pode-se usar o controle modular para reduzir a complexidade de problemas com um grande número de especificações e de subsistemas agregados.

6. CONCLUSÃO

Esta dissertação contribui para a teoria de controle de sistemas a eventos discretos com a proposta de uma abordagem mais eficiente para a síntese de supervisores modulares para sistemas de grande porte.

A abordagem desenvolvida baseia-se na modelagem por linguagens controláveis. Inicialmente, demonstra-se uma série de propriedades de operações sobre linguagens que, além de fundamentar o presente trabalho, servem como ferramental para futuros desenvolvimentos teóricos. O modelo de Ramadge e Wonham é usado, então, para propiciar um processo automático de síntese de supervisores, ao invés dos usuais procedimentos manuais ou heurísticos. Além desta vantagem, o procedimento de síntese de controladores tem a grande conveniência de ser baseado no modelo da dinâmica do sistema em malha aberta e na especificação do comportamento desejado. Assim, novos controladores podem ser rapidamente e automaticamente projetados quando o sistema é modificado ou os objetivos de controle são trocados. Ademais, a idéia de síntese do supervisor minimamente restritivo, característica da abordagem de linguagens controláveis, pode atribuir maior grau de liberdade ao sistema controlado.

Na abordagem proposta, o problema de explosão do número de estados do modelo de Ramadge e Wonham é tratado pela representação da planta por Sistema Produto e por um refinamento da síntese modular. Assim, os controladores são construídos explorando aspectos de modularidade da planta e das especificações. Como resultado, obtém-se uma estrutura de controle naturalmente descentralizada em que cada módulo supervisiona apenas os subsistemas diretamente afetados pela respectiva especificação. Com isso, em caso de mudanças na planta ou nas especificações, respeitada a condição de não bloqueio, os módulos de controle podem ser reprojetados levando em consideração apenas informações locais. Portanto, quando a estrutura de controle modular local é aplicável, consegue-se um sistema de controle distribuído com maior flexibilidade e maior simplicidade computacional.

Outra contribuição importante deste trabalho é a proposta de uma metodologia para a síntese de supervisores para sistemas de manufatura discretos. Baseada na abordagem

desenvolvida nesta pesquisa, a metodologia proposta incorpora os benefícios da abordagem modular e da abordagem descentralizada na construção de controladores para sistemas de manufatura, com significativa economia computacional na síntese e no funcionamento do controle. As vantagens do método são realçadas pelo exemplo elucidativo de sistema integrado de manufatura.

Além de apresentar contribuições teóricas, esta pesquisa deixa em aberto algumas questões que servem de sugestões para trabalhos futuros. Uma destas questões diz respeito à verificação de modularidade. Apesar do modelo apresentado garantir uma economia computacional, a elevada complexidade do teste de modularidade, aspecto mais custoso da síntese modular, ainda pode ser um obstáculo na solução de problemas de maior porte. Contudo, é esperado que esse processo venha a ser efetuado de forma ainda mais eficiente tanto ao nível das especificações quanto da estrutura da planta.

A solução de conflitos é também um interessante campo de pesquisa no domínio de sistemas a eventos discretos. A discussão sobre a interação de múltiplos controladores, feita na seção 4.3.5, estimula o estudo mais aprofundado da resolução de bloqueio pela composição de novos módulos de supervisão.

Outra indicação para trabalhos futuros decorre da limitação da abordagem de controle modular local para sistemas que possam ser representados como Sistema Produto, ou seja, plantas compostas de subsistemas concorrentes. Espera-se poder refinar ainda mais a síntese modular explorando a estrutura modular de conjuntos de subsistemas síncronos.

7. REFERÊNCIAS BIBLIOGRÁFICAS

[1] BALEMI, S. *Control of Discrete Event Systems: Theory and Application.* Zurich, 1999. Thesis (Doctor of Technical Sciences) - Swiss Federal Institute of Technology.

[2] BRANDIN, B. A.; WONHAM, W. M. Modular Supervisory Control of Timed Discrete-Event Systems. *In: Proceedings of the 32nd IEEE Conference On Decision and Control*, p. 2230-2235, Dec. 1993.

[3] BRYANT, R. E. Graph Based Algorithms for Boolean Function Manipulation. *IEEE Transaction on Computers,* v. 35, n. 8, p. 677-691, 1986.

[4] CHEN, Y.-L.; LAFORTUNE, S.; LIN, F. Modular supervisory control with priorities for discrete event systems. *In: Proceedings of the 34th IEEE Conference On Decision and Control*, p. 409-415, December 1995.

[5] ÇINLAIR, E. *Introduction to Stochastic Processes.* Prentice-Hall, Inc., Englewood Cliffs, N.J., USA, 1975.

[6] DE QUEIROZ, M. H.; CURY, J. E. R. Modular Control of Composed Systems. *In: Proceedings of the American Control Conference*. Chicago, June 2000.

[7] DE QUEIROZ, M. H.; e CURY, J. E. R. Modular Supervisory Control of Large Scale Discrete-Event Systems. *In: Proceedings of the WODES*. Submetido para publicação em 2000.

[8] DE QUEIROZ, M. H.; e CURY, J. E. R. Controle Modular de Sistemas de Manufatura Discretos. *Anais do Congresso Brasileiro de Automática*. Submetido para publicação em 2000.

[9] EYZELL, J. M.; CURY, J. E. R. Symmetry in the Supervisory Control Problem. *In: Proceedings of the Workshop on Discrete Event Systems*, Cagliari, Italy, August 1998

[10] EYZELL, J. M.; CURY, J. E. R. Exploiting Symmetry in the Synthesis of Supervisors for Discrete Event Systems. *In: Proceedings of the American Control Conference*, Philadelphia, USA, June 1998.

[11] HOPCROFT, J. E.; ULLMAN, J. D. *Introduction to Automata Theory, Languages and Computation.* Addison-Wesley, USA, 1979.

[12] KLEINROCK, L. *Queueing Systems*, Volume I: Theory. John Wiley & Sons, Canada, 1975.

[13] KROGH, B. H.; Holloway, L. E. Synthesis of Feedback Control Logic for Discrete Manufacturing Systems. *Automatica*, v. 27, n. 4, p. 641-651, 1991.

[14] KUMAR, R.; GARG, V. K. *Modeling and Control of Logical Discrete Event Systems*. Kluwer Academic Publishers, USA, 1995.

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

[16] MURATA, T. Petri Nets: Properties, Analysis and Applications. *In: Proceedings of the IEEE*, v. 77, n. 4, p. 541-580, 1989.

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

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

[19] RAMADGE, P. J.; WONHAM, W. M. Modular feedback logic for discrete event systems. *SIAM Journal of Control and Optimization*, v. 25, n. 5, p. 1202-1218, 1987.

[20] RAMADGE, P. J.; WONHAM, W. M.. The control of discrete event systems. *In: Proceedings IEEE, Special Issue on Discrete Event Dynamic Systems*, v. 77, n. 1, p. 81-98, Jan. 1989.

[21] RUDIE, K.; WONHAM, W. M. Think globally, act locally: decentralized supervisory control. *IEEE Transactions on Automatic Control*, v. 37, n. 11, p. 1692-1708, 1992.

[22] WILLNER, Y.; HEYMANN, M. Supervisory control of concurrent discrete-event systems. *International Journal on Control*, v. 54, n.5, p. 1143-1169, 1991.

[23] WONG, K. C.; THISTLE, J. G.; HOANG, H.-H.; et al. Conflict resolution in modular control with application to feature interaction. *In: Proceedings of the 34th IEEE Conference On Decision and Control*, p. 416-421, Dec. 1995.

[24] WONG, K. C.; WONHAM, W. M. Modular Control and Coordination of Discrete-Event Systems. *Discrete Event Dynamic Systems*, v. 8, n. 3, p. 241-273, Oct. 1998.

[25] WONHAM, W. M. *Notes on Control of Discrete-Event Systems*. Dept. of Electrical & Computer Engineering, University of Toronto, 1998.

[26] WONHAM, W. M.; RAMADGE, P. J. Modular supervisory control of discrete event systems. *Mathematics of control of discrete event systems*, v. 1, n. 1, p. 13-30, 1988.

[27] ZHONG, H.; WONHAM, W. M. On the Consistency of Hierarchical Supervision in Discrete-Event Systems. *IEEE Transactions on Automatic Control*, v. 35, n. 10, p. 1125-1134, 1990.

[28] ZILLER, R. M. *A abordagem Ramadge-Wonham no controle de sistemas a eventos discretos: contribuições à teoria.* Florianópolis, 1993. Dissertação (Mestrado em Engenharia Elétrica) – Centro Tecnológico, Universidade Federal de Santa Catarina.