CONTROLE SUPERVISÓRIO MODULAR DE SISTEMAS DE MANUFATURA

Max H. de Queiroz, José E. R. Cury*
max,cury@lcmi.ufsc.br

*LCMI-DAS-UFSC, CEP 88040-900 – Florianópolis SC

ABSTRACT

This paper proposes a methodology for the synthesis of optimal supervisors for automated manufacturing facilities. The methodology is based on local modular supervisory control, an extension of the Supervisory Control Theory with the goal of avoiding states explosion in the modeling and synthesis of supervisors. The modeling consider that manufacturing systems are characteristically formed by several concurrent subsystems that should be synchronized in way to obey a series of specifications. The synthesis process exploits modularity of specifications, as well as the natural decentralized structure of manufacturing systems, to generate minimally restrictive supervisors with reduced number of states. The proposed methodology is applied to a hypothetical example of integrated manufacturing system. The results show considerable reduction of the computational complexity and of the size of supervisors.

KEYWORDS: Supervisory control, manufacturing systems, discrete-event systems, control system synthesis, decentralized control.

RESUMO

Este artigo propõe uma metodologia para a síntese de supervisores ótimos para sistemas automatizados de manufatura. A metodologia se baseia na abordagem de controle supervisório modular local, uma extensão da Teoria de Controle Supervisório que tem o objetivo de evitar a explosão de estados na modelagem e na síntese de supervisores. A modelagem considera que os sistemas de manufatura sejam formados por diversos subsistemas concorrentes que devem ser sincronizados de forma a atender a uma série de especificações. O processo de síntese explora a modularidade das especificações, bem como a estrutura descentralizada natural de sistemas de manufatura, para gerar supervisores minimamente restritivos e de tamanhos reduzidos. A metodologia proposta é aplicada a um exemplo hipotético de sistema integrado de manufatura. Os resultados obtidos mostram considerável redução da complexidade computacional e do tamanho dos supervisores.

PALAVRAS-CHAVE: Controle supervisório, sistemas de manufatura, sistemas a eventos discretos, síntese de sistemas de controle, controle decentralizado.

1 INTRODUÇÃO

Os modernos sistemas automatizados de manufatura são compostos por múltiplos subsistemas, tais como fabricação, montagem, transporte e armazenagem. O principal objetivo do controle supervisório nesse domínio é a coordenação desses subsistemas de forma que atendam a uma série de tarefas individuais e conjuntas, garantindo o bom funcionamento global do sistema.

Na automação de sistemas de manufatura, diversas abordagens formais têm sido utilizadas para o desenvolvimento da lógica de controle, dentre as quais se incluem Controle Supervisório (Ramadge e Wonham, 1989), Redes de Petri (Murata, 1989), Redes de Petri Controladas (Krogh e Holloway, 1991), Cadeias de Markov (Çinlair, 1975) e Teoria das Filas

Artigo submetido em 20/12/00
1a. Revisão em 17/05/01
Aceito sob recomendação do Ed. Assoc. Prof. Paulo E. Miyagi

(Kleinrock, 1975). A maior parte dessas abordagens limita-se à análise de soluções propostas, que são geradas com base na experiência e inspiração do projetista. Em contrapartida, a Teoria de Controle Supervisório (TCS) utiliza a modelagem da planta e das especificações por linguagens de forma a permitir a síntese automática de controladores ótimos.

Embora na TCS os cálculos para a solução dos problemas de controle exijam esforço polinomial no número de estados dos modelos da especificação e da planta, o número de estados desses modelos cresce exponencialmente com a agregação de especificações e subsistemas. Este fator limitante tem sido considerado por vários autores que procuram explorar diferentes aspectos do problema, como modularidade (Wonham e Ramadge, 1988) e simetria (Eyzell e Cury, 1998a e 1998b), no sentido de superar dificuldades computacionais.

Como o projeto de sistemas de manufatura envolve uma grande quantidade de especificações, a abordagem de controle modular é freqüentemente usada para a síntese de controladores. Ao invés de se projetar um único supervisor monolítico que satisfaça todas as especificações, procura-se construir um supervisor para cada especificação (Wonham e Ramadge, 1988). 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.

Por outro lado, o processo de modelagem da planta por uma única linguagem pode induzir uma explosão no número de estados do modelo pela composição dos diversos subsistemas envolvidos. Esse fator pode inviabilizar o controle supervisório em grande parte dos sistemas de manufatura. Em Queiroz (2000) e em Queiroz e Cury (2000a e 2000b), a abordagem de controle modular apresentada por Wonham e Ramadge (1988) foi estendida de forma a permitir implementar os supervisores a partir de modelos de menor tamanho, tirando proveito da característica descentralizada da planta. Trabalhando-se, assim, com modelos mais reduzidos, diminui-se a complexidade computacional da síntese de controladores.

Neste artigo, apresenta-se uma metodologia para a síntese de controladores para sistemas de manufatura, baseada na abordagem de controle supervisório modular local proposta por Queiroz e Cury (2000b). Essa metodologia é, então, aplicada a um modelo hipotético de linha de montagem. A seqüência do artigo é a seguinte: a Seção 2 propõe uma abordagem para a modelagem de sistemas de manufatura por linguagens; na Seção 3 é apresentada uma metodologia de síntese de controladores modulares locais, a qual é ilustrada por um exemplo na Seção 4; os resultados são discutidos na Seção 5.

2 MODELAGEM DE SISTEMAS DE MANUFATURA

Para fins de síntese da lógica de controle, os sistemas de manufatura podem ser classificados como Sistemas a Eventos Discretos (SEDs), que são sistemas dinâmicos cuja mudança de estado ocorre em pontos discretos do tempo, em decorrência de eventos isolados, como por exemplo comandos para operação de máquinas ou sinais de ativação de sensores. Em contrapartida aos sistemas dirigidos pelo tempo, que são classicamente modelados por equações diferenciais e a diferenças, o comportamento dessa classe de sistemas pode ser matematicamente modelado por linguagens, que são conjuntos de cadeias finitas de símbolos representando todas as seqüências de eventos admitidas pelo sistema (Hopcroft e Ullman, 1979). Nesta seção, são introduzidos os conceitos preliminares sobre linguagens, bem como uma abordagem para a modelagem de sistemas de manufatura que fundamenta a metodologia proposta neste artigo.

2.1 Linguagens e Geradores

Na abordagem de Controle Supervisório proposta por Ramadge e Wonham (1989), o funcionamento em malha aberta de um sistema de manufatura é modelado por linguagens sobre um conjunto de eventos Σ, que é particionado em eventos controláveis e não controláveis. 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 que não podem ser evitados de ocorrer e por isso são considerados permanentemente habilitados. Essas linguagens, se forem regulares, são associadas a geradores (Hopcroft e Ullman, 1979). Um gerador é uma quíntupla G = (Σ, Q, δ, q_0, Q_m), onde Σ é o alfabeto de eventos, Qé um conjunto de estados, $q_0 \in Q$ é o estado inicial, $Q_m \subseteq Q$. é o conjunto de estados marcados e δ:$\Sigma \times Q \rightarrow Q$, a função de transição, é uma função parcial definida em cada estado de Q para um subconjunto de Σ.

Seja Σ* o conjunto de todas as cadeias finitas de Σ, incluindo a cadeia nula ε. Então, G é caracterizado por dois subconjuntos de Σ* chamados de *linguagem gerada* de G(todas as seqüências de eventos que a planta pode gerar), denotado por $L(G)$, e de *linguagem marcada* de G(seqüências representando tarefas completas), denotado por $L_m(G)$.

Os geradores 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. Os eventos controláveis são representados por ramos interceptados.

Por exemplo, o funcionamento simplificado de uma máquina pode ser representado pelo gerador G da figura 1, cujos eventos α e β representam respectivamente o início de operação (controlável) e o final de operação (não controlável). O comportamento da máquina nesse modelo é $L_m(G)$ = $\{\varepsilon, \alpha\beta, \alpha\beta\alpha\beta, ...\}$ e $L(G)$ = $\{\varepsilon, \alpha, \alpha\beta, \alpha\beta\alpha, \alpha\beta\alpha\beta, ...\}$.

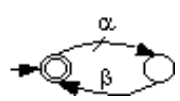

Figura 1: Gerador para uma máquina.

A função de transição δ pode ser naturalmente estendida para cadeias de eventos 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.

Um estado $q \in Q$ é chamado de acessível se $\exists s \in \Sigma^*$ tal que $\hat{\delta}(s,q_0) = q$ e de coacessível se $\exists s \in \Sigma^*$ tal que $\hat{\delta}(s,q) \in Q_m$. Diz-se que um gerador é aparado (ou *trim*) se todos seus estados forem acessíveis e coacessíveis.

O prefixo fechamento, ou simplesmente fechamento, de uma linguagem L é dado por $\bar{L} = \{u : \exists v \in \Sigma* \wedge uv \in L\}$.

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: $P_i(L)$=$\{u_i \in \Sigma_i* | u_i = P_i(u)$ para algum $u \in L\}$. A projeção inversa é, então, definida como

$$P_i^{-1}(L_i) = \{u \in \Sigma* | P_i(u) \in L_i\}.$$

Sejam$L_i \subseteq \Sigma_i^*$, *i=1,...,n.* Seja $\Sigma = \cup_{i=1}^n \Sigma_i$ e P_i:$\Sigma^* \rightarrow \Sigma_i^*$. Define-se o produto síncrono $||_{i=1}^n L_i \subseteq \Sigma^*$ como:

$$||_{i=1}^n L_i = \cap_{i=1}^n P_i^{-1}(L_i) = \{u \in \Sigma* \mid \wedge_{i=1}^n P_i(u) \in L_i\}.$$

Sejam geradores G_i, *i=1,...,n.* A composição síncrona $G = ||_{i=1}^n G_i$ é obtida fazendo-se a evolução em paralelo dos n geradores G_i, na qual um evento comum a múltiplos geradores só é executado se todos os geradores que contiverem este evento o executarem simultaneamente. As linguagens resultantes da composição síncrona são caracterizadas por:

$$L(G) = ||_{i=1}^n L(G_i); L_m(G) = ||_{i=1}^n L_m(G_i).$$

2.2 Modelagem por Sistema Produto

Os sistemas automatizados de manufatura são geralmente compostos por um grande número de subsistemas concorrentes interagindo entre si, como produção, movimentação e armazenamento de material. No projeto de sistemas de maior complexidade, a modelagem das diversas partes envolvidas G_i', *i = 1,...,n'*, geralmente é um passo intermediário na representação do comportamento conjunto do sistema. Isso porque a modelagem de sistemas de menor porte exige menor esforço computacional, menos memória e costuma ser mais compreensível ao 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 único gerador $G = ||_{i=1}^{n'} G_i'$ acaba sendo proibitiva na maioria dos sistemas de manufatura. Para contornar essa barreira, o sistema de manufatura é representado como um Sistema Produto (Ramadge e Wonham, 1989; Ramadge, 1989), que consiste de um conjunto de subsistemas completamente assíncronos entre si.

Para obter uma representação por sistema produto (RSP) mais refinada possível a partir de uma modelagem inicial de todos os subsistemas, faz-se a composição dos subsistemas síncronos originais (que têm eventos em comum), criando-se um conjunto com o maior número possível de subsistemas assíncronos distintos, cada qual com o mínimo de estados. A RSP de um sistema composto requer, então, um mínimo esforço computacional em operações de produto para obter um conjunto de subsistemas assíncronos que modelam o sistema global em malha aberta. Este modelo representa a estrutura descentralizada natural de operações concorrentes para um sistema de manufatura. Embora os resultados deste artigo sejam válidos para qualquer RSP, a abordagem de controle modular local apresentada na seqüência será tão vantajosa quanto mais refinada for a RSP.

O funcionamento de um sistema de manufatura em malha aberta inclui uma série de cadeias de eventos indesejáveis, resultantes da interação descoordenada dos diversos subsistemas na ausência de controle. Assim, para expressar matematicamente o comportamento desejado ao sistema, definem-se as especificações de controle como linguagens representando o ordenamento necessário de subconjuntos de eventos da planta.

Seja a RSP de um sistema G formada por subsistemas G_i = (Σ_i, Q_i, δ_i, q_{0i}, Q_{mi}), $i \in I$ =$\{1, ..., n\}$. Para *j=1,...,m*, sejam agora as especificações genéricas locais $E_{gen,j}$ definidas respectivamente em subconjuntos de eventos $\Sigma_{gen,j} \subseteq \Sigma$.

Para *j=1,...,m*, a planta local $G_{loc,j}$ associada à especificação $E_{gen,j}$ é definida por:

$G_{loc,j} = ||_{i \in I_{loc,j}} G_i$, com $I_{loc,j} = \{k \in I | \Sigma_k \cap \Sigma_{gen,j} \neq \emptyset\}$.

Assim, a planta local $G_{loc,j}$ é composta apenas pelos subsistemas da modelagem original que estão diretamente (e indiretamente) restringidos por $E_{gen,j}$. A planta enxuta $G_e = ||_{j=1}^{m} G_{loc,j}$ engloba apenas os subsistemas relevantes ao problema de controle.

Por exemplo, seja um sistema composto pelo conjunto de subplantas $\{G'_i = (\Sigma'_i, Q'_i, \delta'_i, q'_{0i}, Q'_{mi}),\ i = 1,...,5\}$. Sejam duas especificações genéricas $E_{gen,a} \subseteq \Sigma_a$* e $E_{gen,b} \subseteq \Sigma_b$*. A relação entre os conjuntos de eventos é ilustrada pela figura 2. 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 $G_{loc,a} = G_1||G_2$ e $G_{loc,b} = G_2||G_3$. A planta enxuta é calculada como $G_e = G_1||G_2||G_3$ e a planta global $G = G_1||G_2||G_3||G_4$.

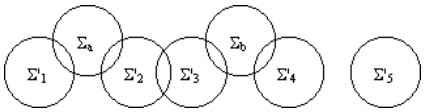

Figura 2: Relação entre alfabetos do sistema composto.

As especificações genéricas $E_{gen,i}$ podem, então, ser expressas em termos da planta local $G_{loc,i}$ da planta enxuta G_e e da planta global G, respectivamente como: $E_{loc,j} = E_{gen,j}||L_m(G_{loc,j})$, $K_{glo-e,j} = E_{gen,j}||L_m(G_e)$ e $K_{glo,j} = E_{gen,j}||L_m(G)$.

Para simplificar a apresentação deste artigo, assume-se que sejam modelados no sistema produto apenas subsistemas restringidos por alguma especificação, ou seja, que $G = G_e$.

3 CONTROLE MODULAR DE SISTEMAS DE MANUFATURA

Um dos grandes problemas no projeto de sistemas a eventos discretos é a síntese de supervisores que, controlando o sistema modelado, restrinjam seu funcionamento a um comportamento especificado. A teoria de Ramadge e Wonham (1989) tem se mostrado uma ferramenta bastante poderosa para essa tarefa.

3.1 Controle Supervisório

O objetivo do controle supervisório monolítico é projetar um único supervisor 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 3 ilustra a estrutura de controle supervisório monolítico.

Figura 3: Esquema de controle monolítico.

Um supervisor pode ser representado por um gerador S, cujas mudanças de estado são ditadas pela ocorrência de eventos na planta G. A ação de controle de S, definida para cada estado do gerador correspondente, é desabilitar em G os eventos que não possam ocorrer em S após uma cadeia de eventos observada. O funcionamento do sistema controlado S/Gpode ser descrito pelo SED resultante da composição síncrona de S e G, isto é, $S||G$. Desse modo, além de restringir o comportamento da planta, o supervisor assim definido tem a função de desmarcar estados, ou seja, uma tarefa do sistema em malha fechada é considerada completa somente se for marcada pela planta e pelo supervisor (chamado de desmarcador).

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

A condição necessária e suficiente para a existência de um supervisor (desmarcador) próprio S que atenda a uma dada especificação $K \subseteq L_m(G)$ ($L_m(S/G) = K$) é a controlabilidade da linguagem especificada (Ramadge e Wonham, 1989). Uma linguagem $K \subseteq L(G)$ é uma sublinguagem controlável de $L(G) \subset \Sigma^*$ (ou controlável e.r.a G) se $\overline{K}\Sigma_u \cap L(G) \subseteq \overline{K}$. Isso quer dizer que a ocorrência de um evento não-controlável e fisicamente possível, após uma cadeia de $\bar{K}$, mantém a seqüência no conjunto $\bar{K}$. A classe de linguagens controláveis contidas na linguagem gerada por G é denotada por $C(M, G) = \{K | K \subseteq M$ e K é controlável e.r.a $G\}$ e, por ser fechada sobre união e não vazia ($\emptyset \in C(M, G)$), contém um (único) elemento supremo, chamado *SupC*(M, G).

Nem sempre é possível construir um supervisor que restrinja o comportamento do sistema a uma linguagem especificada de forma exata. Entretanto, quando um comportamento especificado não é controlável, é possível projetar um supervisor próprio que atenda às especificações de forma minimamente restritiva Neste caso, o controle monolítico objetiva sintetizar um supervisor S para uma linguagem especificada $K \subseteq L(G)$, 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 de controle não tem solução.

Portanto, na teoria de controle supervisório (Ramadge e Wonham, 1989), o passo mais importante para a síntese de supervisores é o cálculo da máxima linguagem controlável contida em uma linguagem que representa o comportamento desejado. A complexidade desse cálculo, embora polinomial no número de estados do modelo da planta e da especificação, é um fator limitante em aplicações, pois o número de estados que representa o sistema cresce exponencialmente com o número de elementos componentes do sistema.

3.2 Controle Modular

Quando a especificação global se compõe de mais de uma especificação ($K = \cap_{i=1}^{n} K_{glo,i}$), dois tipos de abordagens são consideradas para a síntese de controladores. Por um lado, pode-se projetar um único supervisor monolítico que satisfaça todas as especificações. Por outro lado, pode-se construir um supervisor modular para cada especificação, de forma que, atuando em conjunto, os supervisores satisfaçam todas as especificações (Wonham e Ramadge, 1988). A ação conjunta de supervisores modulares desabilita um evento controlável da planta sempre que este for desabilitado por algum dos subcontroladores. Este funcionamento é ilustrado na figura 4.

Figura 4: Esquema de controle modular.

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. 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.

Sejam linguagens $L_i \subseteq \Sigma^*$, *i = 1,...,n*. Diz-se que o conjunto de linguagens $\{L_i,\ i = 1,...,n\}$ é modular (ou não-conflitante) se $\cap_{i=1}^{n} \overline{L_i} = \overline{\cap_{i=1}^{n} L_i}$. Isso quer dizer que, sempre que um prefixo for aceito por todo o conjunto de linguagens, todo o conjunto deve aceitar uma palavra contendo esse prefixo, ou seja, as linguagens não geram conflito.

A condição necessária e suficiente para que o resultado do controle modular seja equivalente ao monolítico é a modularidade das linguagens marcadas pelas ações dos supervisores (Wonham e Ramadge, 1988).

3.3 Controle Modular Local

A abordagem de controle modular local proposta por Queiroz e Cury (2000b) permite explorar, além da modularidade das especificações, a estrutura naturalmente descentralizada dos sistemas de manufatura automatizados, de forma a proporcionar um menor esforço computacional no processo de síntese e de implementação do controle. O conceito apresentado a seguir, extensão da modularidade para linguagens sobre alfabetos distintos, é fundamental para a aplicação dessa abordagem.

Sejam linguagens $L_i \subseteq \Sigma_i^*$, *i = 1,...,n*. Diz-se que o conjunto de linguagens $\{L_i,\ i = 1,...,n\}$ é localmente modular se $\cap_{i=1}^{n} \overline{P_i^{-1}(L_i)} = \overline{\cap_{i=1}^{n} P_i^{-1}(L_i)}$, ou seja, se $||_{i=1}^{n} \overline{L_i} = \overline{||_{i=1}^{n} L_i}$. Se, para *i = 1,...,n*, as linguagens L_i forem marcadas por geradores aparados G_i, é fácil provar que a modularidade local é verificada se e somente se $G = ||_{i=1}^{n} G_i$ for um gerador aparado.

É importante observar que a modularidade local de todos os subconjuntos de linguagens não garante a modularidade local do conjunto completo. Por exemplo, sejam as linguagens L_1=$\{\alpha\beta,\ \alpha\gamma,\ \mu\}$, L_2=$\{\alpha\gamma,\ \alpha\mu,\ \beta\}$ e L_3=$\{\alpha\mu,\ \alpha\beta,\ \gamma\}$ definidas em $\Sigma_1 = \Sigma_2 = \Sigma_3 = \{\alpha,\ \beta,\ \gamma,\ \mu\}$. 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 possa ser decomposta e que requeira, portanto, o cálculo da composição síncrona do conjunto completo de linguagens.

O seguinte resultado mostra que a síntese da máxima linguagem controlável de múltiplas especificações pode ser executada diretamente a partir das especificações locais sem perda de performance em relação à solução monolítica (e, por conseguinte, à solução modular clássica), desde que a modularidade local seja válida.

Teorema 1: (Queiroz e Cury, 2000b) Dados um sistema com RSP formada por G_j, *j = 1,...,m*, e as especificações $E_{gen,i}$, *i=1,...,n*. Sejam $E_{loc,i}, K_{glo,i}, G_{loc,i}$ e Gdefinidos como na Seção 2. Se $\{SupC(E_{loc,i}, G_{loc,i}),\ i = 1,...,n\}$ for localmente modular, então, $SupC(\cap_{i=1}^{n} K_{glo,i}, G) = ||_{i=1}^{n} SupC(E_{loc,i}, G_{loc,i})$.

O Teorema 1 fundamenta a abordagem para a síntese de con-

troladores modulares proposta por Queiroz e Cury (2000b). Dadas n especificações locais $E_{gen,i}$, *i=1,...,n,* sobre um sistema de manufatura, é necessário apenas expressá-las em termos dos subsistemas $G_{loc,i,}$ *i=1,...,n,* afetados por elas. Calcula-se assim a máxima linguagem controlável *SupC*$(E_{loc,i}, G_{loc,i}),$ *i=1,...,n,* contida nas mesmas. Para cada especificação, pode-se então construir um supervisor não bloqueante que, desabilitando eventos controláveis da respectiva planta local, gere em malha fechada a máxima linguagem controlável obtida. A condição de modularidade local, pelo teorema anterior, garante que este procedimento não resulte em perda de performance em relação ao controle monolítico.

Para a construção de um supervisor não bloqueante a partir de uma linguagem *SupC*$(E_{loc,i}, G_{loc,i}),$ *i=1,...,n,* resultante do processo de síntese, toma-se um gerador aparado $S_{loc,i}$ que marque esta linguagem. A ação de controle é atribuída a cada estado de $S_{loc,i}$ como um conjunto de eventos que devem ser obrigatoriamente desabilitados, ou seja, que no respectivo estado possam ocorrer na planta livre G_i e não possam em $S_{loc,i}$.

O funcionamento conjunto dos supervisores localmente modulares é ilustrado pela figura 5. Pode-se observar que, apesar de nenhuma restrição na estrutura de informações ter sido previamente imposta (Rudie e Wonham, 1992), os controladores locais observam e controlam apenas o comportamento dos subsistemas afetados. Assim, pode-se afirmar que a abordagem de controle modular local induz uma estrutura de controle decentralizada que surge naturalmente do processo de síntese.

Figura 5: Esquema de controle modular local.

3.4 Resolução de Conflitos

É importante observar que, quando a modularidade das especificações controláveis não é verificada, existe um conjunto de supervisores conflitantes que impedem o controle modular de ser diretamente aplicado. Quando isso ocorre, é possível utilizar diferentes abordagens para a resolução de conflitos. Uma alternativa evidente é a aplicação de controle monolítico para o menor subconjunto de especificações conflitantes.

Entretanto, outras abordagens mais elaboradas são encontradas na literatura. Wong *et alii* (1995) propõem evitar conflitos introduzindo uma interface entre um dos supervisores conflitantes e a planta. Essa interface suspende o controlador modular quando o conflito aparece e o reativa no momento apropriado. Chen *et alii* (1995) apresentam um esquema para resolução de conflitos baseado na atribuição de prioridades a supervisores individuais. Já o trabalho de Wong e Wonham (1998) propõe a resolução de conflitos através de coordenação hierárquica.

3.5 Redução de Supervisores

A quantidade de memória empregada pelo programa de controle depende do número de estados dos supervisores e, por ser um recurso limitado, muitas vezes impede a implementação do sistema de controle nos dispositivos usuais, como controladores lógicos programáveis. Considerando também que um menor número de estados torna a lógica de controle de um supervisor mais compreensível ao programador, pode-se afirmar que a redução dos supervisores sintetizados é um procedimento a ser considerado.

Um algoritmo formal para minimização de supervisores é apresentado por Vaz e Wonham (1986). Nesse algoritmo, todos os estados do supervisor original são agrupados no menor número possível de blocos (não necessariamente disjuntos), de forma que as desabilitações dos estados de um bloco não entrem em conflito com os eventos habilitados em outros estados do mesmo bloco e que o supervisor reduzido, representando a estrutura de transição entre os blocos, seja determinístico. Esse algoritmo, entretanto, tem complexidade exponencial no tempo e, assim, é realizável apenas para geradores de pequeno porte. Como esse é geralmente o caso dos supervisores resultantes da síntese modular local, é possível minimizá-los antes da implementação final do sistema de controle.

3.6 Metodologia Proposta

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

1. modelar cada componente elementar do sistema isoladamente, evitando incluir eventos e estados desnecessários;

2. calcular a mais refinada RSP, fazendo a composição dos subsistemas síncronos;

3. modelar cada especificação isoladamente, considerando apenas os eventos relevantes;

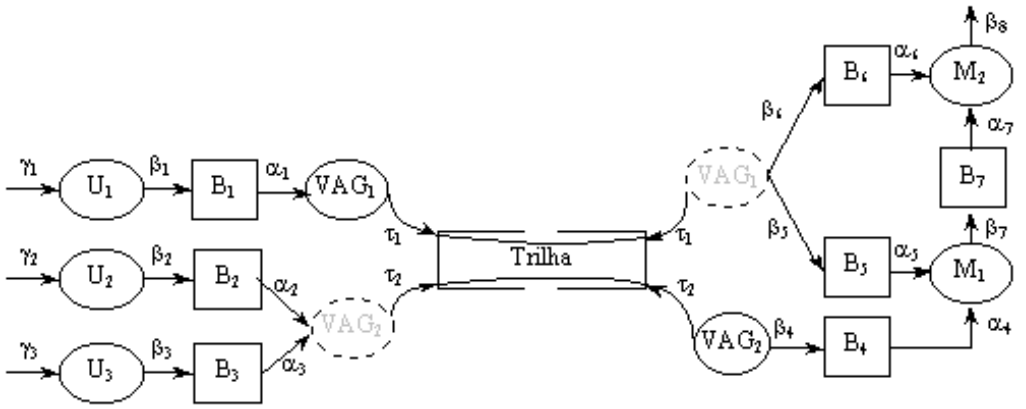

Figura 6: Sistema Integrado de Manufatura.

4. obter a planta local para cada especificação compondo os subsistemas da RSP que tenham eventos em comum com a especificação;
5. calcular a linguagem de cada planta local que satisfaça a especificação, através do produto síncrono de cada planta local com sua respectiva especificação genérica;
6. calcular a máxima linguagem controlável contida em cada especificação local;
7. verificar a modularidade local das linguagens resultantes;
8. se não forem modulares, resolver os conflitos;
9. se forem modulares, implementar um supervisor reduzido para cada linguagem controlável.

4 EXEMPLO

Para exemplificar a metodologia proposta neste artigo, sintetizam-se controladores locais para a automatização do modelo hipotético de Sistema Integrado de Manufatura (SIM) ilustrado na figura 6. O SIM em questão é formado por três centros de usinagem U_i, $i = 1, 2, 3$, por dois Veículos Autoguiados (VAGs), denominados VAG_1 e VAG_2, e por duas estações de montagem M_1 e M_2. A função dos VAGs é suprir a linha de montagem com os componentes produzidos nos centros de usinagem, passando por uma trilha que faz a ligação entre as duas áreas.

Fazendo a interface entre as máquinas do SIM há sempre um sistema 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 VAGs 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 os eventos não controláveis 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 geradores G_i, $i = 1, 2, 3$, apresentados na figura 7.

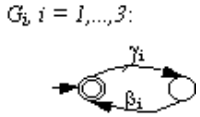

Figura 7: Geradores para $U_i, i = 1, 2, 3$.

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 geradores G_4 e G_5 ilustrados nas figuras 8(a) e 8(b).

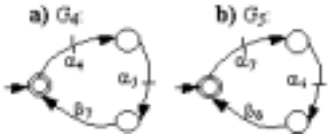

Figura 8: Geradores para: a) M_1; b)M_2.

O VAG_1 desocupa a trilha e retira um componente do *buffer* B_1 através do evento controlável α_1. A seguir, o VAG_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 um componente nos *buffers* B_5 ou B_6, respectivamente. O transportador volta, então, ao estado inicial por τ_1. O comportamento do VAG_1 é modelado pelo gerador G_6, mostrado na figura 9(a).

Por outro lado, o VAG_2 pode inicialmente ocupar a trilha pelo evento controlável τ_2. Na seqüência, o VAG_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 VAG_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 9(b) ilustra a modelagem desse comportamento pelo gerador G_7.

Figura 9: Geradores para: a) VAG_1; b) VAG_2.

As especificações genéricas sobre os *buffers* unitários B_i, *i=1,...,7*, podem ser modeladas respectivamente pelas linguagens marcadas pelos geradores $E_{gen,i}$, *i=1,...,7*, representados na figura 10.

Figura 10: Geradores para $E_{gen,i}, i = 1, ..., 7$.

O modelo da especificação sobre os VAGs é obtido fazendo-se a composição de seus modelos e retirando-se do gerador resultante os estados em que os dois VAGs ocupam a trilha ao mesmo tempo. Assim, a especificação geral de estados proibidos dos VAGs é dada pela linguagem marcada pelo gerador $E_{gen,8}$, o qual é apresentado na figura 11.

Como os subsistemas já estão modelados de forma assíncrona, pode-se afirmar que o sistema encontra-se na mais refinada RSP. Assim, pode-se obter as plantas locais $G_{loc,i}$, *i=1,...,8*, respectivas às especificações genéricas $E_{gen,i}$, *i=1,...,8*, fazendo-se a composição dos modelos que tenham eventos comuns a elas. Geram-se desta forma as seguintes plantas locais:

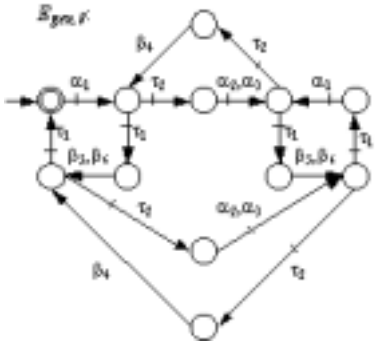

Figura 11: Gerador para $E_{gen,8}$.

$$\diamond\ G_{loc,1} = G_1 \,||\, G_6;$$
$$\diamond\ G_{loc,2} = G_2 \,||\, G_7;$$
$$\diamond\ G_{loc,3} = G_3 \,||\, G_7;$$
$$\diamond\ G_{loc,4} = G_4 \,||\, G_7;$$
$$\diamond\ G_{loc,5} = G_4 \,||\, G_6;$$
$$\diamond\ G_{loc,6} = G_5 \,||\, G_6;$$
$$\diamond\ G_{loc,7} = G_4 \,||\, G_5;$$
$$\diamond\ G_{loc,8} = G_6 \,||\, G_7.$$

Para *i=1,...,8*, calculam-se, então, as especificações locais $E_{loc,i}$ pela composição síncrona das especificações $E_{gen,i}$ com suas respectivas plantas $G_{loc,i}$. O passo seguinte é o cálculo das máximas linguagens controláveis *SupC*($E_{loc,i}$, $G_{loc,i}$), *i=1,...,8*, contidas nas especificações locais. Por exemplo, a linguagem *SupC*($E_{loc,1}, G_{loc,1}$) é marcada pelo gerador da figura 12.

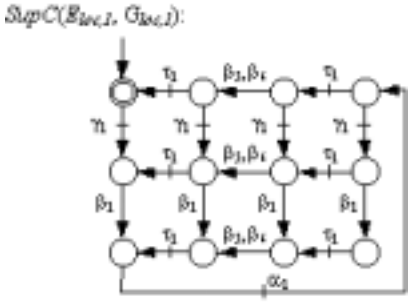

Figura 12: Gerador para *SupC*($E_{loc,1}$, $G_{loc,1}$).

Verifica-se, então, a modularidade das linguagens resultantes calculando e testando a igualdade de $||_{i=1}^{8}\overline{SupC(E_{loc,i}, G_{loc,i})}$ e $\overline{||_{i=1}^{8} SupC(E_{loc,i}, G_{loc,i})}$. Isso garante que não existe bloqueio na ação conjunta dos 8

supervisores e que o resultado da ação modular é o mesmo de uma possível ação centralizada.

Para cada uma dessas linguagens resultantes, elabora-se um supervisor $S_{loc,i}$, *i=1,...,8,* que atue na respectiva planta local $G_{loc,i}$desabilitando eventos controláveis que possam ocorrer na planta e não sejam aceitos por *SupC*($E_{loc,i}, G_{loc,i}$). Por exemplo, o supervisor $S_{loc,1}$, que garante o bom funcionamento do *buffer* B_1, observa e controla o comportamento do centro de usinagem U_1 e do VAG_1, restringindo-o à linguagem *SupC*($E_{loc,1}, G_{loc,1}$). O funcionamento do supervisor $S_{loc,1}$ pode ser ilustrado pelo diagrama de transição de estados da figura 13, cujas setas tracejadas representam as desabilitações de eventos.

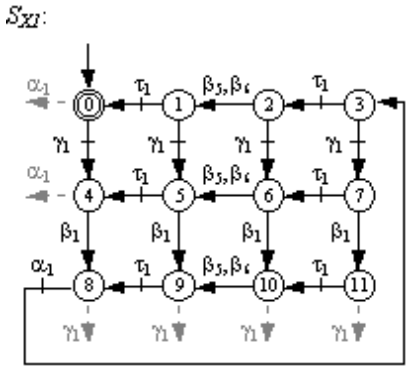

Figura 13: Supervisor para *SupC*($E_{loc,1}$, $G_{loc,1}$).

Agrupando os estados 0,..,7 no bloco X_1 e os estados 8,...,11 em X_2, obtém-se um supervisor reduzido (mínimo) $S_{red,1}$, apresentado na figura 14(a), equivalente ao supervisor $S_{loc,1}$. Os demais supervisores, obtidos pelo processo de redução de estados da seção 3.5, são mostrados na figura 14(b-h).

Como resultado da abordagem modular local, são obtidos 8 supervisores locais com geradores de 2 e 3 estados, que foram reduzidos a partir de supervisores com no máximo 21 estados, conforme a tabela 1. Pela abordagem clássica os geradores para os 8 supervisores modulares teriam no mínimo 864 estados. A abordagem monolítica geraria um supervisor único cujo gerador seria 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}).

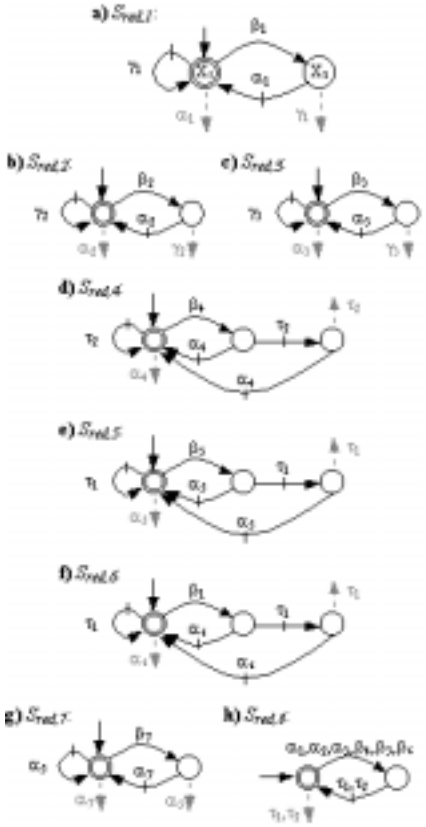

Figura 14: Supervisores reduzidos $S_{red,i}$, *i=1,...,8.*

5 CONCLUSÃO

Os resultados obtidos com o exemplo indicam que a metodologia de controle supervisório modular local proposta é bastante vantajosa na síntese de supervisores para sistemas de manufatura. Especialmente no que diz respeito à complexidade computacional da síntese, o controle modular local se mostrou muito mais eficiente do que a abordagem clássica, uma vez que não há explosão do número de estados do modelo com o aumento do porte do sistema.

Entretanto, é importante notar que a complexidade da verificação da modularidade local, embora vantajosa em relação a outras abordagens encontradas na literatura, acaba crescendo exponencialmente com o número de especificações e subsistemas envolvidos. Isso aponta para a necessidade do desenvolvimento de métodos mais eficientes para a verificação da modularidade. Uma alternativa nesse sentido é a obtenção de condições suficientes que garantam a modularidade local sobre estruturas genéricas de sistemas de manufatura.

Finalmente, uma das grandes vantagens do controle modular local está na implementação distribuída dos supervisores em

Tabela 1: Número de Estados dos Geradores.

	Controle Modular ($\|G\|$=1152)		
i	$\boldsymbol{E}_{gen,i}$	$\boldsymbol{K}_{glo,i}$	***SupC*($\boldsymbol{K}_{glo,i}$,*G*)**
1	2	2304	1728
2	2	2304	1728
3	2	2304	1728
4	2	2304	2016
5	2	2304	2016
6	2	2304	2016
7	2	2304	1920
8	12	864	864

	Controle Modular Local			
i	$\boldsymbol{G}_{loc,i}$	$\boldsymbol{E}_{loc,i}$	$\boldsymbol{S}_{loc,i}$	$\boldsymbol{S}_{red,i}$
1	8	16	12	2
2	8	16	12	2
3	8	16	12	2
4	12	24	21	3
5	12	24	21	3
6	12	24	21	3
7	9	18	15	2
8	16	12	12	2

sistemas de manufatura. Garantidas as condições de modularidade, tem-se maior flexibilidade, maior segurança e economia computacional na aplicação do controle supervisório. Essas boas propriedades são exploradas por Queiroz *et alii* (2001) na implementação em diagrama escada do sistema de controle modular local para uma célula de manufatura real.

AGRADECIMENTOS

Os autores manifestam sua gratidão pelas seguintes fontes de apoio financeiro: o primeiro autor foi apoiado pela CAPES durante o mestrado e pelo CNPq (Processo 140.825/2000-2) no doutorado; o segundo autor teve sua atividade sustentada em parte pelo CNPq (Processo 300.953/93-3 e PRONEX 015/98).

REFERÊNCIAS

Chen, Y.-L., S. Lafortune and F. Lin (1995). Modular supervisory control with priorities for discrete event systems. In *Proceedings of the 34th IEEE Conference On Decision and Control*, pp. 409-415.

Çinlair, E. (1975). *Introduction to Stochastic Processes.* Prentice-Hall, Inc., Englewood Cliffs, N.J., USA.

Eyzell, J.M. and J.E.R. Cury (1998a). Symmetry in the Supervisory Control Problem. In *Proceedings of the Workshop on Discrete Event Systems*, Cagliari, Italy.

Eyzell, J.M. and J.E.R. Cury (1998b). Exploiting Symmetry in the Synthesis of Supervisors for Discrete Event Systems. In *Proceedings of the American Control Conference*, Philadelphia, USA.

Hopcroft, J.E. and J.D. Ullman (1979). *Introduction to Automata Theory, Languages and Computation*. Addison-Wesley, Reading, MA.

Kleinrock, L. (1975). *Queueing Systems*, Volume I: Theory. John Wiley & Sons, Canada.

Krogh, B.H. and L.E. Holloway (1991). Synthesis of Feedback Control Logic for Discrete Manufacturing Systems. *Automatica*, Vol. 27, n° 4, pp. 641-651.

Murata, T. (1989). Petri Nets: Properties, Analysis and Applications. *Proceedings of the IEEE*, Vol. 77, n° 4, pp. 541-580.

Queiroz, M.H. de (2000). Controle supervisório modular de sistemas de grande porte. Dissertação de Mestrado, Universidade Federal de Santa Catarina, Departamento de Engenharia Elétrica, Florianópolis.

Queiroz, M.H. de and J.E.R. Cury (2000a). Modular Control of Composed Systems. In *Proceedings of the American Control Conference*. Chicago, USA.

Queiroz, M.H. de and J.E.R. Cury (2000b). Modular supervisory control of large scale discrete-event systems. In *Discrete Event Systems: Analysis and Control*. Kluwer Academic Publishers, pp. 103-110. (*Proceedings of the WODES 2000*. Ghent, Belgium)

Queiroz, M.H. de, E.A.P. Santos e J.E.R. Cury (2001). Síntese modular do controle supervisório em diagrama escada para uma célula de manufatura. *Anais do V Simpósio Brasileiro de Automação Inteligente*, Gramado RS.

Ramadge, P.J. (1989). Some tractable supervisory control problems for discrete-event systems modeled by Büchi automata. *IEEE Transactions on Automatic Control*, Vol. 34, n° 1, pp. 10-19.

Ramadge, P.J. and W.M. Wonham (1989). The control of discrete event systems. *Proceedings IEEE, Special Issue on Discrete Event Dynamic Systems*, Vol. 77, n° 1, pp. 81-98.

Rudie, K. and W.M. Wonham (1992). Think globally, act locally: decentralized supervisory control. *IEEE Transactions on Automatic Control*, Vol. 37, n° 11, pp. 1692-1708.

Vaz, A.F. and W.M. Wonham (1986). On supervisor reduction in discrete-event systems. *International Journal of Control*, Vol. 44, n° 2, pp. 475-491.

Wonham, W.M. and P.J. Ramadge (1988). Modular supervisory control of discrete event systems. *Mathematics of control of discrete event systems*, Vol. 1, n° 1, pp. 13-30.

Wong, K.C., J.G. Thistle, H.-H Hoang and R.P. Malhamé (1995). Conflict resolution in modular control with application to feature interaction. In *Proceedings of the 34th IEEE Conference On Decision and Control*, pp. 416-421.

Wong, K.C. and W.M. Wonham (1998). Modular Control and Coordination of Discrete-Event Systems. In *Discrete Event Dynamic Systems*, Vol. 8, n° 3, pp. 241-273.