METODOLOGIA E FERRAMENTA DE APOIO AO TESTE DE NÃO-CONFLITO NO CONTROLE MODULAR DE SISTEMAS A EVENTOS DISCRETOS

Patrícia N. Pena[*]
ppena@ufmg.br

Antonio E.C. Cunha[†]
carrilho@ime.eb.br

José E. R. Cury[‡]
cury@das.ufsc.br

Stéphane Lafortune[§]
stephane@eecs.umich.edu

[*]DELT-Universidade Federal de Minas Gerais, Belo Horizonte, MG, Brasil

[†]SE/3-Instituto Militar de Engenharia, Rio de Janeiro, RJ, Brasil.

[‡]DAS-CTC-Universidade Federal de Santa Catarina, Florianópolis, SC, Brasil

[§]Department of EECS-The University of Michigan, Ann Arbor, MI, USA

ABSTRACT

Methodology and Computational Tool for Testing Nonconflict On Modular Control of Discrete Event Systems

This paper presents contributions on the use of abstractions in modular control of discrete event systems. First, the $W \circ T$-strategy is presented for the construction of abstractions for the nonconflict test of local modular supervisors. Second, an efficient computational tool is introduced for the problem of verifying if given abstractions of modular supervisors satisfy the so-called observer property (OP); the observer property plays an important role in the validity of the nonconflict test over abstractions. Finally, this new computational tool is used to validate the $W \circ T$ strategy in the context of a case study of a comprehensive manufacturing example.

KEYWORDS: supervisory control, discrete event systems; modular control; observer property.

RESUMO

Este artigo apresenta contribuições ao uso de abstrações no controle modular de sistemas a eventos discretos. Inicialmente, propõe-se a estratégia $W \circ T$ para a construção de abstrações para o teste de não-conflito. Em seguida, apresenta-se uma ferramenta computacional eficiente para o problema de verificar se uma abstração satisfaz a propriedade de observador (OP); esta propriedade tem um papel importante na abordagem de verificação de não-conflito de supervisores modulares baseada em abstrações. Esta ferramenta é então utilizada para validar a estratégia $W \circ T$ no contexto de um estudo de caso correspondendo a um exemplo de sistema de manufatura de médio porte.

PALAVRAS-CHAVE: controle supervisório, sistemas a eventos discretos; controle modular; propriedade do observador.

1 INTRODUÇÃO

Abstrações têm sido amplamente utilizadas na literatura relacionada à Teoria de Controle Supervisório (Ramadge e Wonham, 1987), como uma forma de tratar a explosão

Artigo submetido em 31/03/2009 (Id.: 00987)
Revisado em 27/06/2009, 23/09/2009
Aceito sob recomendação do Editor Associado Prof. Alexandre Bazanella

de estados inerente à síntese de supervisores para Sistemas a Eventos Discretos. No controle hierárquico, abstrações são usadas como modelos para o nível gerencial (alto-nível), permitindo a síntese de leis de controle que satisfaçam especificações deste nível da estrutura hierárquica de controle (Wong e Wonham, 1996), (Torrico e Cury, 2004), (Schmidt et al., 2004), (Leduc et al., 2005), (Cunha e Cury, 2007), (Schmidt et al., 2007), (Hill, Cury, de Queiroz e Tilbury, 2008). Nesse caso, o modelo de alto-nível esconde eventos que correspondem a detalhes operacionais do comportamento do sistema, não-relevantes no mundo gerencial. Abstrações têm sido ainda usadas no controle modular (Wonham e Ramadge, 1988), (de Queiroz e Cury, 2000), (de Queiroz e Cury, 2002), como um meio de aliviar a carga computacional do teste de não-conflito entre supervisores sintetizados por uma abordagem de controle modular. Nesse caso, o comportamento em malha fechada decorrente da ação de cada supervisor local é abstraído, e o teste de não-conflito é realizado sobre o conjunto de abstrações obtidas (Feng e Wonham, 2006), (Pena et al., 2006a), (Pena et al., 2006b), (Hill e Tilbury, 2006), (Hill, Tilbury e Lafortune, 2008). Em ambos os problemas acima descritos, é necessário que os modelos abstratos sejam de algum modo consistentes com os modelos originais, de modo a garantir que eles cumpram com os objetivos a que se propõem. Uma propriedade chave para a consistência das abstrações é a chamada propriedade de observador (OP) (Wong e Wonham, 1996). OP-abstrações não são somente necessárias para garantir consistência entre modelos, mas também para garantir que os autômatos correspondentes às abstrações tenham espaço de estados de tamanho reduzido quando comparado àquele dos autômatos correspondentes aos modelos originais. É sabido que, no pior caso, a complexidade computacional do cálculo de abstrações baseadas na projeção natural, é exponencial no número de estados do autômato original. No entanto, se a projeção é uma OP-abstração, essa complexidade torna-se polinomial. Além disso, nesse caso a abstração obtida tem um número de estados menor ou igual ao do modelo original (Wong, 1998).

Este artigo apresenta três contribuições. A primeira é a proposta de uma estratégia eficiente para a verificação do não-conflito em um sistema composto; essa estratégia, aqui denominada $W \circ T$, é baseada numa combinação das estratégias propostas em (Pena et al., 2006a) and (Pena et al., 2006b), e torna possível obter maiores reduções para os modelos abstraídos. A segunda corresponde à introdução de um tipo especial de autômato, denominado OP-verificador, e sua implementação computacional. O OP-verificador é usado para verificação, em tempo polinomial, da propriedade do observador a partir de um autômato e um conjunto de eventos a serem preservados. Esta verificação é feita sem a necessidade de se calcular a projeção que leva ao modelo abstraído, evitando-se assim o problema da complexidade exponencial do cálculo de projeções que não possuem a propriedade do observador. O OP-verificador é inspirado no verificador introduzido em (Yoo e Lafortune, 2002), para teste de diagnosticabilidade de linguagens. Finalmente, utiliza-se a ferramenta computacional introduzida no trabalho, para validar a abordagem $W \circ T$ no contexto do estudo de caso de um sistema de manufatura de médio porte. No exemplo, o teste de não-conflito é realizado usando as abordagens propostas em (Pena et al., 2006a), (Pena et al., 2006b) e a estratégia combinada proposta no artigo. Versão preliminar deste artigo foi apresentada em (Pena et al., 2008).

O artigo é organizado como segue. A Seção 2 retoma alguns resultados preliminares relativos à teoria de controle supervisório e os conceitos de projeção natural e observadores; a Seção 3 apresenta o teste de não-conflito baseado em abstrações, obtido pela combinação de estratégias previamente propostas; a Seção 4 apresenta detalhes do OP-verificador e de sua implementação como ferramenta computacional; o estudo de caso de um sistema de manufatura é desenvolvido na Seção 5.

2 NOÇÕES PRELIMINARES

Este artigo foi desenvolvido sob o formalismo iniciado por Ramadge e Wonham (1987). Sob esse formalismo, um Sistema a Eventos Discretos (SED) é modelado por um autômato $G = (Q, \Sigma, \delta, q_0, Q_m)$, onde Q é o conjunto de estados, Σ é o conjunto de eventos, δ é a função de transição, que é uma função parcial e que pode ser naturalmente estendida para cadeias, q_0 é o estado inicial e Q_m é o conjunto de estados marcados. Σ^* é o conjunto de todas as cadeias finitas de elementos em Σ, incluindo a cadeia vazia ϵ. Usa-se $En^G(q)$ para representar o conjunto de eventos factíveis no estado $q \in Q$ de G.

Um estado $q \in Q$ é chamado de acessível se $\exists s \in \Sigma^*$ tal que $\delta(q_0, s) = q$, ou seja, se existir uma cadeia no autômato G que, partindo do estado inicial q_0, alcance o estado q. Um estado $q \in Q$ é chamado de coacessível se $\exists s \in \Sigma^*$ tal que $\delta(q, s) \in Q_m$, isto é, se houver uma cadeia em G que, partindo do estado q, alcance um estado marcado. Diz-se que um autômato é aparado se todos seus estados forem acessíveis e coacessíveis.

Uma linguagem é um subconjunto de Σ^*. O comportamento de um autômato G é modelado por duas linguagens: $\mathcal{L}(G) = \{s \in \Sigma^* | \delta(q_0, s) \text{ definida}\}$ que é o conjunto de cadeias finitas que G gera e $\mathcal{L}_m(G) = \{s \in \mathcal{L}(G) | \delta(q_0, s) \in Q_m\}$, que é o conjunto de cadeias que

representam tarefas completas.

O prefixo-fechamento de uma linguagem, denotado por $\overline{L}$, é dado por $\overline{L} = \{s \in \Sigma^* | st \in L \text{ para algum } t \in \Sigma^*\}$. Uma linguagem L é dita prefixo-fechada se $\overline{L} \subseteq L$.

O conjunto de eventos usados para modelar a planta é dividido em dois subconjuntos: o conjunto de eventos controláveis e o conjunto de eventos não-controláveis, Σ_{uc}. A ação do supervisor sobre a planta consiste em inibir a ocorrência de eventos controláveis de forma a alcançar o comportamento desejado. Às vezes não é possível implementar a linguagem desejada e a especificação (também chamada de linguagem desejada) é dita não-controlável. Formalmente, uma linguagem K é não-controlável em relação a (e.r.a) uma dada linguagem $\mathcal{L}(G)$ se $\overline{K}\Sigma_{uc} \cap \mathcal{L}(G) \nsubseteq \overline{K}$. Nestes casos, o supervisor implementa a máxima sublinguagem controlável da linguagem desejada, chamada $Sup\mathcal{C}(K, G)$, onde K, a linguagem desejada, é obtida pela composição síncrona da especificação genérica dada por E, e $\mathcal{L}_m(G)$, denotada por $E \| \mathcal{L}_m(G)$.

A projeção natural $P_i : \Sigma^* \to \Sigma_i^*$ é uma operação definida sobre dois conjuntos de eventos, Σ e Σ_i, onde $\Sigma_i \subseteq \Sigma$. Esta operação mapeia cadeias em Σ^* em cadeias em Σ_i^* apagando as ocorrências de eventos em $\Sigma \setminus \Sigma_i$. A projeção natural é definida a seguir:

$$\begin{aligned} P_i(\epsilon) &= \epsilon \\ P_i(s\sigma) &= \begin{cases} P_i(s) & \text{se } s \in \Sigma^*, \sigma \notin \Sigma_i \\ P_i(s)\sigma & \text{se } s \in \Sigma^*, \sigma \in \Sigma_i. \end{cases} \end{aligned}$$

O conceito de projeção natural pode ser estendido para linguagens como $P_i(L) = \{u_i \in \Sigma_i^* \,|\, u_i = P_i(u) \text{ para algum } u \in L\}$.

No contexto deste trabalho, as abstrações dos supervisores são autômatos que possuem espaço de estados reduzido, obtido a partir do supervisor original, com certas propriedades. Neste caso, utiliza-se a operação de projeção natural para obter as abstrações. Estas abstrações precisam ser consistentes com os supervisores originais em relação às possibilidades de conflito. Para garantir que um autômato resultante da operação de projeção natural tenha espaço de estados reduzido, é necessário que a abstração tenha a *propriedade do observador*.

A propriedade das projeções conhecida como *propriedade do observador* foi introduzida por Wong e Wonham (1996) para linguagens prefixo-fechadas a depois estendida para linguagens não prefixo-fechadas (Wong et al., 2000) e é apresentada na Definição 1.

Definição 1 Sejam $L \subseteq \Sigma^*$ uma linguagem, $\Sigma_r \subseteq \Sigma$ um conjunto de eventos e $\theta : \Sigma^* \to \Sigma_r^*$ a projeção natural de cadeias em Σ^* em cadeias em Σ_r^*. Se $(\forall m \in \overline{L})(\forall n \in \Sigma_r^*)$ tal que $\theta(m)n \in \theta(L)$ é verdade que $(\exists p \in \Sigma^*)$ tal que $\theta(mp) = \theta(m)n$ e $mp \in L$, então $\theta(L)$ possui a propriedade do observador.

Em palavras, $\theta(L)$ possui a propriedade de observador se, para toda cadeia m do prefixo de L tal que sua projeção $\theta(m)$ seja prefixo de uma cadeia $\theta(m)n$ da linguagem $\theta(L)$, seja possível estender a cadeia m para uma cadeia mp em L tal que $\theta(mp)$ seja igual a $\theta(m)n$.

Diz-se que $\theta(L)$ é uma "OP-abstração" se possuir a propriedade do observador.

A próxima seção apresenta o contexto em que a nova metodologia e a ferramenta de apoio ao teste de não-conflito foram desenvolvidos.

3 TESTE DE NÃO-CONFLITO EM SUPERVISORES MODULARES

Visando a redução da complexidade de síntese de supervisores, a abordagem modular (Wonham e Ramadge, 1988) é proposta. Trata-se de uma extensão do controle monolítico em que há vários supervisores, um para cada especificação. O controle supervisório modular local (de Queiroz e Cury, 2000) é uma extensão de (Wonham e Ramadge, 1988) em que cada supervisor é projetado com uma visão parcial da planta. Nesta abordagem, cada supervisor é aplicado sobre uma planta local (G_j) e cada planta pode ter interferência com outras, significando que elas compartilham eventos. Esta interferência pode causar bloqueio quando mais de um supervisor é aplicado simultaneamente sobre a planta. Neste caso, diz-se que os supervisores são conflitantes. Para as linguagens implementadas pelos supervisores S_j, obtidas pelo cálculo de $Sup\mathcal{C}(K_j, G_j)$, em que $K_j = E_j \| \mathcal{L}_m(G_j)$ e E_j é a especificação genérica, pode-se dizer que as linguagens são não-conflitantes se:

$$\|_{j=1}^{m} \overline{S_j} = \overline{\|_{j=1}^{m} S_j}. \qquad (1)$$

Devido à necessidade de compor todos os supervisores, este teste possui alta complexidade computacional.

Tem havido um interesse recente em reduzir a complexidade deste teste. Pena et al. (2006a) e Feng (2006) desenvolveram independentemente um conjunto de condições sobre as abstrações obtidas pela projeção natural tal que o teste de não-conflito pode ser aplicado sobre elas gerando o mesmo resultado que o teste original.

Duas metodologias para obtenção do conjunto de eventos relevantes para o conflito, ou seja o conjunto de eventos que não podem ser apagados pela operação de proje-

ção, foram determinadas em (Pena et al., 2006a) e (Pena et al., 2009). Seguindo a nomenclatura utilizada nestes trabalhos, este subconjunto de eventos é chamado de conjunto de eventos relevantes ou Σ_r. Além da abstrações serem obtidas a partir de um conjunto mínimo de eventos, deve-se ter abstrações com a propriedade do observador. Quando não é possível atender as duas condições simultaneamente, o conjunto de eventos relevantes Σ_r pode ser estendido para que OP-abstrações sejam obtidas. Os Teoremas 1 e 2 apresentam dois conjuntos de condições sobre as abstrações tais que elas podem ser utilizadas para verificação de conflito.

Teorema 1 *(Pena et al., 2006a)* Sejam $\theta_j : \Sigma_j^* \rightarrow (\Sigma_r \cap \Sigma_j)^*$ projeções naturais. Se $\theta_j(S_j)$, $j \in J$, são OP-abstrações e se $\Sigma_s \subseteq \Sigma_r$ com $\Sigma_s = \bigcup_{k,l \in J}^{k \neq l} (\Sigma_k \cap \Sigma_l)$, então:

$$\|_{j=1}^{m} \overline{\theta_j(S_j)} = \overline{\|_{j=1}^{m} \theta_j(S_j)} \Longleftrightarrow \|_{j=1}^{m} \overline{S_j} = \overline{\|_{j=1}^{m} S_j}.$$

O Teorema 1 estabelece que todos os eventos comuns devem estar contidos em Σ_r, ou seja, devem ser considerados relevantes. Esta é uma primeira condição sobre os eventos das abstrações que garante, juntamente com a propriedade do observador, que o conjunto de abstrações obtido pela projeção natural dos supervisores sobre os subconjuntos de Σ_r que fazem parte de cada supervisor, é consistente com o conjunto de supervisores originais em relação às suas possibilidades de conflito.

O resultado apresentado no Teorema 1 é usado para desenvolver um procedimento, chamado de W-abordagem, para obtenção de OP-abstrações para o teste do não-conflito. A W-abordagem é detalhada a seguir.

W-abordagem

1. Supervisores locais são obtidos;
2. Conjunto inicial de eventos relevantes é obtido, $\Sigma_{rW}^I = \Sigma_s$;
3. Conjuntos locais de eventos relevantes são obtidos, $\Sigma_j'^I = \Sigma_j \cap \Sigma_{rW}^I$, $j \in J$;
4. Extensão de Σ_{rW}^I de forma a obter OP-abstrações para cada supervisor S_j;
5. Teste de não-conflito sobre o conjunto de abstrações obtidas, $\theta_j(S_j)$, $j \in J$.

Outro conjunto de condições para as abstrações foi obtido em (Pena et al., 2006b) e posteriormente relaxado em (Pena et al., 2009). Esta última versão é apresentada no Teorema 2 a seguir.

Teorema 2 *(Pena et al., 2009)* Usando as definições apresentadas anteriormente, se as projeções naturais $\theta_j(S_j)$, $j \in J$, são OP-abstrações e se:

- K_j é controlável e.r.a $\mathcal{L}(G_j)$;
- $\bigcup_{j=1}^{m} (\Sigma_{E_j}) \subseteq \Sigma_r$;

então:

$$\|_{j=1}^{m} \overline{\theta_j(S_j)} = \overline{\|_{j=1}^{m} \theta_j(S_j)} \Longleftrightarrow \|_{j=1}^{m} \overline{S_j} = \overline{\|_{j=1}^{m} S_j}.$$

O Teorema 2 indica outro conjunto de condições sobre o conjunto de eventos Σ_r que pode ser usado para desenvolver uma solução análoga à W-abordagem apresentada anteriormente, chamada de T-abordagem. Nem sempre as especificações de um problema são controláveis. Neste caso, obtém-se o supervisor como sendo a máxima sublinguagem controlável contida em K_j. Para obter a especificação controlável, pode-se usar algoritmos de redução de supervisores (Su e Wonham, 2004) e (Sivolella et al., 2006). O supervisor reduzido faz o papel de especificação genérica E_j', indicando quais os eventos que devem estar em Σ_r. A seguir, detalha-se a T-abordagem.

T-abordagem

1. Supervisores locais são obtidos;
2. Conjunto inicial de eventos relevantes é obtido, Σ_{rT}^I;
   a. Supervisores reduzidos são obtidos;
   b. Supervisores reduzidos S_{Rj} são renomeados para E_j';
   c. $\Sigma_{rT}^I = \bigcup_{j=1}^{m} \Sigma_{E_j'}$.
3. Conjuntos locais de eventos relevantes são obtidos, $\Sigma_j'^I = \Sigma_j \cap \Sigma_{rT}^I$, $j \in J$;
4. Extensão de Σ_{rT}^I de forma a obter OP-abstrações para cada supervisor S_j;
5. Teste de não-conflito sobre o conjunto de abstrações obtidas, $\theta_j(S_j)$, $j \in J$.

Os resultados apresentados nos Teoremas 1 e 2 permitem prever que a aplicação combinada das estratégias apresentadas nas abordagens W e T possa levar a ganhos maiores na redução das abstrações. Uma terceira estratégia, a $W \circ T$-abordagem é então proposta a seguir. Esta estratégia consiste na aplicação das duas abordagens, em sequência. A ordem de aplicação das duas abordagens, sendo T aplicada antes de W, é justificada pelo fato de na T-abordagem ser necessário utilizar informação da estrutura do sistema, mais precisamente, das especificações que geraram os supervisores. Esta estrutura é perdida quando a W-abordagem é aplicada, tornando impossível a aplicação da T-abordagem sobre as OP-abstrações obtidas pela W-abordagem.

Algoritmo $W \circ T$

1. Aplicar a T-abordagem sobre os supervisores S_j, $j \in J$;
2. Aplicar a W-abordagem sobre as abstrações resultantes da aplicação da T-abordagem.

Ao combinar as duas soluções, há mais chances de se obterem abstrações menores e também de se abstrair mais supervisores. No entanto, as melhores escolhas para extensões do conjunto Σ_r não são conhecidas uma vez que este problema possui complexidade NP (Feng, 2006). Então, busca-se por boas abstrações, não necessariamente mínimas, pela escolha de eventos a serem adicionados a Σ_r. Na próxima seção, o OP-verificador, um algoritmo polinomial para verificar se uma abstração possui a propriedade do observador, é introduzido.

4 OP-VERIFICADOR

Para lidar com problemas maiores, torna-se necessário o uso de uma ferramenta computacional para assistir na busca das abstrações que levam à redução de complexidade no teste do não-conflito. Portanto, torna-se crítico ser capaz de verificar se uma abstração possui a propriedade do observador. Como a busca pela OP-abstração é tentativa-erro, é desejável ter a capacidade de classificar, dados os conjuntos de eventos envolvidos na projeção, se a projeção possui a propriedade do observador, sem calculá-la. Apresenta-se a seguir, um algoritmo inspirado no diagnosticador introduzido por (Yoo e Lafortune, 2002), e que realiza a classificação acima mencionada.

Dado um autômato aparado $G = (Q^G, \Sigma, \delta^G, q_0^G, F^G)$ o primeiro passo é substituir os estados marcados de G por estados com autolaços rotulados com o evento τ. Este novo autômato é denominado M e é definido pela quíntupla $M = (Q^M, \Sigma^\tau, \delta^\tau, q_0^M, F^M)$, onde $\Sigma^\tau = \Sigma \cup \{\tau\}$. O autômato M é usado para gerar o verificador $V_G = (Q, \Sigma^\tau, \delta, q_0)$, onde:

- $Q \subseteq [(Q^M \times Q^M) \cup \{Dead\}]$ é o conjunto de estados;
- $\Sigma^\tau = \Sigma \cup \{\tau\}$;
- δ é a função de transição, cuja obtenção é apresentada no procedimento $\delta(q)$;
- $q_0 = (q_0^M, q_0^M)$ é o estado inicial.

Os estados do OP-verificador representam pares de estados de M que possuem a mesma projeção. A violação da propriedade do observador é detectada pela alcançabilidade de um estado especial chamado *Dead* em V_G. A única restrição imposta ao autômato M, e que pode ser estendida para o autômato G, para que se possa aplicar o algoritmo é que ele não tenha ciclos de eventos não-relevantes, ou seja, que o autômato G não possua ciclos de eventos que não estão em Σ_r.

De forma geral, o procedimento para verificação da propriedade do observador é realizado em 4 passos, dados G e Σ_r.

Procedimento para Verificação da Propriedade do Observador

1. Obter M pela inclusão de auto-laços rotulados com τ nos estados marcados;
2. Aparar M.
3. Verificar que não há ciclos de eventos não-relevantes em M.
4. Construir o OP-verificador.
5. Classificar $\theta(G)$ como uma OP-abstração no caso do estado *Dead* não ser alcançável em V_G ou como não sendo uma OP-abstração caso contrário.

O algoritmo para construção do verificador (item 4. do procedimento acima) é apresentado a seguir.

MainAlgorithm

```
1  Q_{T+1} = {(0,0)}
2  Q_T = { }
3  ∀q ∈ (Q_{T+1} − Q_T)
4  Q_T = Q_T ∪ q
5  δ(q)
6  if Dead ∈ Q_{T+1}
7      quit
8  end
9  end
10 V_G = (Q_T, Σ^τ, δ, q_0)
```

A função de transição $\delta(q)$ é apresentada a seguir.

```
δ(q)
11 q_1 = q(1)
12 q_2 = q(2)
13 ∀σ ∈ En(q) = En^M(q_1) ∪ En^M(q_2)
14 if σ ∈ Σ_r
15   if δ^τ(q_1,σ)! & δ^τ(q_2,σ)!
16     δ((q_1,q_2),σ) = (δ^τ(q_1,σ), δ^τ(q_2,σ))
17     Q_{T+1} = Q_{T+1} ∪ {(δ^τ(q_1,σ), δ^τ(q_2,σ))}
18   elseif δ^τ(q_1,σ)! & En^M(q_2) ∩ Σ_u = ∅ or
            δ^τ(q_2,σ)! & En^M(q_1) ∩ Σ_u = ∅
19     δ((q_1,q_2),σ) = (Dead)
20     Q_{T+1} = Q_{T+1} ∪ {(Dead)}
21   end
22 else
23   if δ^τ(q_1,σ)!
24     δ((q_1,q_2),σ) = (δ^τ(q_1,σ), q_2)
25     Q_{T+1} = Q_{T+1} ∪ {(δ^τ(q_1,σ), q_2)}
26   end
27   if δ^τ(q_2,σ)!
28     δ((q_1,q_2),σ) = (q_1, δ^τ(q_2,σ))
29     Q_{T+1} = Q_{T+1} ∪ {(q_1, δ^τ(q_2,σ))}
30   end
31 end
32 end
```

A demonstração de que o procedimento acima descrito verifica a condição de OP-abstração se baseia na definição de um teste, o OP-Teste, que verifica se para qualquer par de cadeias do supervisor S que tenham a mesma projeção, a projeção do conjunto de sufixos das duas cadeias é igual. A demonstração da corretude do procedimento inclui duas partes: (i) provar que, se o par (supervisor, Σ_r) passa no OP-teste, então a abstração obtida pela projeção natural do supervisor no conjunto Σ_r é OP-abstração; (ii) provar que o estado *Dead* não é alcançável no OP-verificador se e somente se o supervisor passa no OP-teste. Destas duas demonstrações, decorre que *Dead* não é alcançável se e somente se é possível obter um OP-abstração a partir do par (supervisor,Σ_r). Detalhes destas demonstrações formais estão disponíveis em (Pena et al., 2007).

Implementou-se o OP-verificador utilizando-se as funções do Grail para Controle Supervisório (Reiser et al., 2006), que é uma ferramenta computacional elaborada por pesquisadores da Universidade Federal de Santa Catarina onde estão disponíveis alguns dos principais algoritmos relacionados à Teoria de Controle Supervisório e suas extensões. O Grail para Controle Supervisório foi desenvolvido sobre o Grail, que é um ambiente computacional simbólico destinado à manipulação de autômatos e expressões regulares (Raymond e Wood, 1994), e é composto por um núcleo básico, correspondendo à implementação dos algoritmos fundamentais da Teoria de Controle Supervisório, e por *toolboxes* que lidam com diferentes extensões da teoria. Estão implementados *toolboxes* para o Controle Supervisório de Sistemas Condição/Evento (Garcia, 2002; Leal, 2005), Controle Supervisório Multitarefa (de Queiroz et al., 2005) e Controle Hierárquico com Marcação Flexível (Cunha e Cury, 2007). Um dos pontos positivos do Grail para Controle Supervisório é a possibilidade de criação de novas funções usando os métodos disponíveis na biblioteca de classes C++ do Grail e dos toolboxes desenvolvidos. Isso permite uma rápida prototipação de novas funções para novos domínios de aplicação, evitando que o desenvolvedor perca tempo em detalhes de programação das estruturas básicas de manipulação. O Grail para Controle Supervisório está disponível para download em (Cury, 2009).

O OP-verificador toma por entradas o autômato G e o conjunto de eventos relevantes para projeção Σ_r, e determina se o par gera ou não uma OP-abstração pela construção do autômato verificador V_G. A complexidade do algoritmo é polinomial no número de estados de G. Diversos testes estão implementados para os argumentos de entrada antes do cálculo de V_G; um dos mais relevantes é o teste que determina se G possui laços de eventos não-relevantes. Este teste baseia-se num algoritmo de detecção de ciclos em grafos, de complexidade polinomial no número de estados de G (Cormen et al., 1990).

Neste trabalho, no contexto da aplicação das T- e $W \circ T$-abordagens, são utilizadas mais duas funções do Grail para Controle Supervisório. A primeira é uma ferramenta de redução de supervisores (Sivolella et al., 2006) que é usada para obtenção de especificações genéricas com a propriedade de serem controláveis em relação às plantas locais. E a segunda é uma ferramenta que remove os eventos dos supervisores reduzidos que não influenciam nas ações de controle sobre as plantas locais.

5 ESTUDO DE CASO: SISTEMA FLEXÍVEL DE MANUFATURA

Ilustra-se o emprego das abordagens W, T e $W \circ T$ e do OP-verificador por intermédio do processo de síntese de supervisores para um Sistema Flexível de Manufatura (SFM), originalmente apresentado em (de Queiroz et al., 2005) e mostrado na Figura 1. O SFM gera dois tipos de produtos a partir de blocos e tarugos brutos: um bloco com um pino cônico no topo (Produto A) e um bloco com um pino cilíndrico pintado (Produto B). O SFM é composto por oito equipamentos, três esteiras C_1, C_2 e C_3, uma Fresa, um Torno, um Robô, uma Máquina de Pintura (MP) e uma Máquina de Montagem (MM), conectados por intermédio de oito depósitos (*buffers*) B_j, $j = 1, \ldots, 8$, cada um dos quais com capacidade para uma peça. A descrição completa deste sistema pode ser encontrada em (de Queiroz et al., 2005). A lógica de controle a ser sintetizada deve permitir o máximo grau de liberdade ao SFM ao mesmo tempo que evita o *overflow* e o *underflow* de peças nos depósitos.

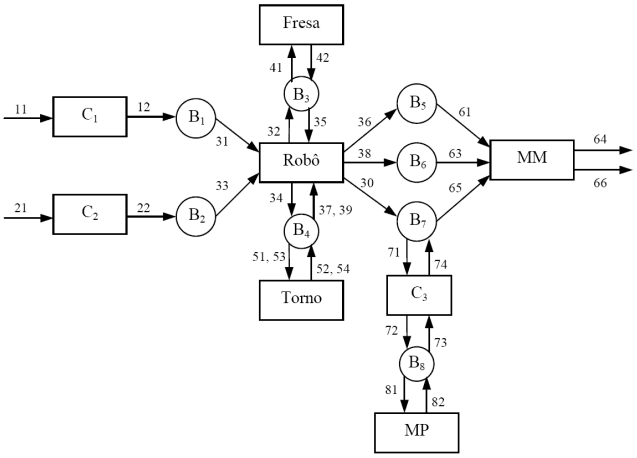

Figura 1: Sistema Flexível de Manufatura.

A modelagem da planta livre, sem controle, é realizada pelo conjunto de oito autômatos assíncronos mostrados na Figura 2, onde os eventos numerados por ímpares são controláveis (de Queiroz et al., 2005). As especificações genéricas de segurança E_j, $j = 1, \ldots, 8$, são expressas pelas linguagens marcadas dos autômatos mostrados na Figura 3. Para cada especificação genérica E_j, $j = 1, \ldots, 8$, obtém-se a respectiva planta local G_j, $j = 1, \ldots, 8$, por composição dos subsistemas afetados. Obtém-se a linguagem desejada $K_j = E_j \| \mathcal{L}_m(G_j)$ e aplica-se o método de síntese modular local para obter os supervisores $S_j = Sup\mathcal{C}(K_j, G_j)$, $j = 1, \ldots, 8$. Apresentam-se os dados relativos ao número de estados de G_j, K_j, S_j, com $j = 1, \ldots, 8$, na Tabela 1, onde $|A|$ denota o número de estados do autômato A. A última coluna da Tabela 1, S_{R_j}, será tratada no contexto da T-abordagem, enquanto que a última linha, com índice $j = 78$, será tratada quando for discutido um esquema de resolução de conflito para o SFM.

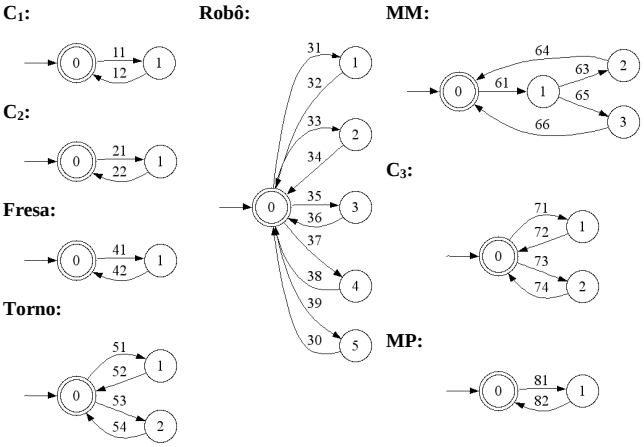

Figura 2: Componentes da planta.

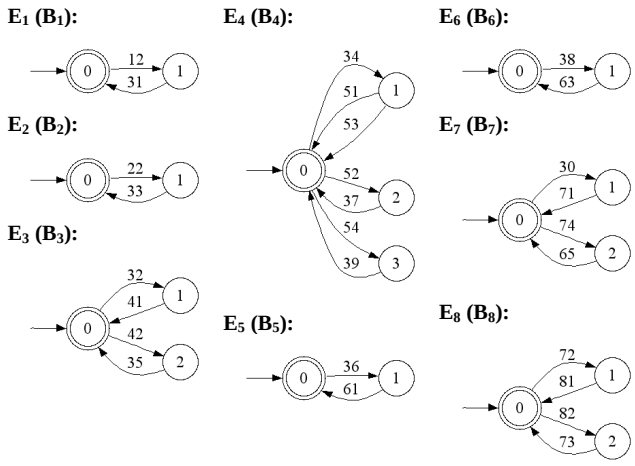

Figura 3: Especificações genéricas.

A primeira abordagem para verificar o conflito entre os supervisores locais consiste na aplicação da igualdade (1). Assim, calcula-se o autômato $S_{mod} = \|_{j=1}^{8} S_j$ com 70.272 estados, dos quais 45.504 são não-bloqueantes, 310.452 transições e 31 eventos. Como S_{mod} é bloqueante, o esquema de supervisão modular local proposto também o é. No que segue, realizar-se-á a verificação de conflito de acordo com as abordagens W, T e $W \circ T$. O desempenho das abordagens será então comparado com a abordagem baseada na equação (1), denominada do-

Tabela 1: Dados relativos à Síntese Modular Local.

j	$\lvert E_j\rvert$	$\lvert G_j\rvert$	$\lvert K_j\rvert$	$\lvert S_j\rvert$	$\lvert S_{R_j}\rvert$
1	2	12	24	18	2
2	2	12	24	18	2
3	3	12	18	18	3
4	4	18	21	21	4
5	2	24	48	44	2
6	2	24	48	44	2
7	3	72	168	128	4
8	3	6	6	6	3
78	9	144	192	164	5

ravante abordagem *convencional*. Mais adiante, um esquema de resolução de conflito será apresentado e os desempenhos das quatro abordagens para verificar o conflito para o novo esquema também serão comparados.

Para a W-abordagem, o conjunto inicial de eventos não-relevantes é $\Sigma - \Sigma^I_{rW} = \{11, 12, 21, 22, 41, 42, 51, 52, 53, 54, 81, 82\}$, onde Σ é o conjunto de eventos completo da planta e $\Sigma^I_{rW} = \Sigma_s$, conforme definido no Teorema 1. Para esse conjunto de eventos relevantes, o OP-verificador classifica $\theta^W_1(S_1)$, $\theta^W_2(S_2)$ e $\theta^W_4(S_4)$ como não sendo OP-abstrações. Σ_r é então expandido por adição de eventos em $\Sigma - \Sigma^I_{rW}$, até que todos os supervisores afetados satisfaçam a propriedade de observador. Pode-se verificar que se $\{11\}$, $\{21\}$ e $\{51, 53\}$ forem adicionados a Σ_r, as abstrações são então classificadas como OP-abstrações.

Observa-se que a expansão de Σ_r não afeta os outros supervisores, e que, em todas as execuções do OP-verificador, nenhum laço de eventos não-relevantes foi encontrado. Assim, com a W-abordagem, os eventos 12, 22, 41, 42, 52, 54, 81 e 82 são não-relevantes e podem ser apagados, obtendo-se os autômatos abstraídos $\theta^W_j(S_j)$, $j = 1, \ldots, 8$. Apresentam-se os números de estados dos autômatos abstraídos na Tabela 2. Verifica-se o conflito pelo cálculo do autômato $S^W_{mod} = \|_{j=1}^{8} \theta^W_j(S_j)$, com 9.080 estados (dos quais 6.656 são não-bloqueantes), 33.584 transições e 23 eventos.

Para a T-abordagem, o primeiro passo é verificar se as especificações são controláveis. Neste exemplo, todas as especificações são não-controláveis. Por utilização de uma ferramenta de redução de supervisores (Sivolella et al., 2006), é possível obterem-se supervisores reduzidos S_{R_j}, $j = 1, \ldots, 8$, com, no máximo, quatro estados, como pode ser visto na última coluna da Tabela 1. Apresentam-se os supervisores reduzidos na Figura 4, onde as setas pontilhadas apontam para os eventos desabilitados no estado correspondente, e os eventos listados abaixo do supervisor são os eventos que constavam no supervisor original e que não influenciam na ação de controle, podendo assim serem removidos dos supervisores reduzidos. Utilizam-se assim os supervisores reduzidos como especificações controláveis para a T-abordagem.

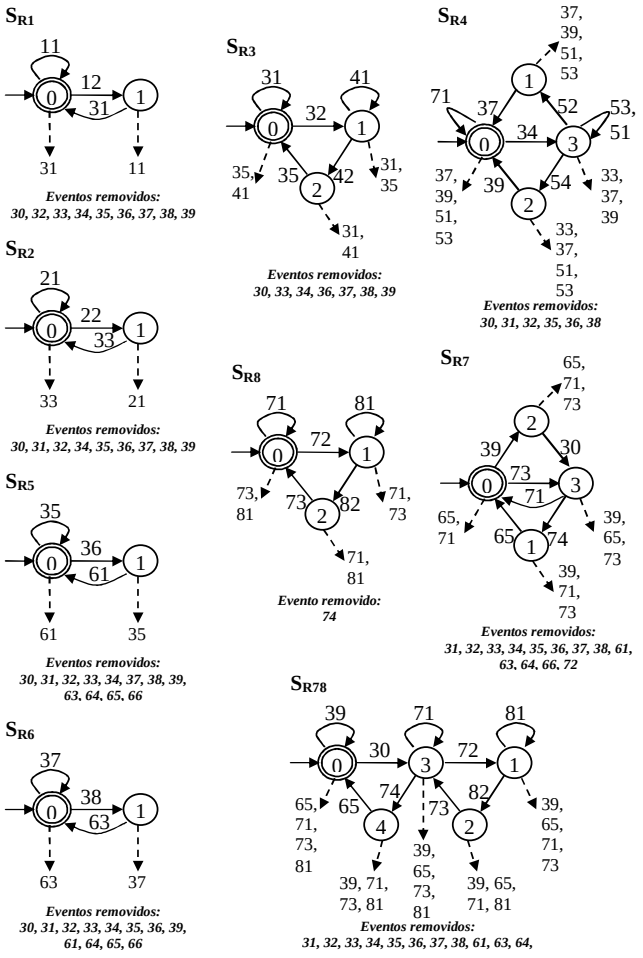

Figura 4: Supervisores Locais Reduzidos.

O conjunto inicial de eventos não-relevantes para a T-abordagem é $\Sigma - \Sigma^I_{rT} = \{64, 65, 66\}$, onde Σ^I_{rT} são os eventos não-relevantes para quaisquer das ações de controle dos supervisores reduzidos (Pena et al., 2006b). Nesse caso, o OP-verificador classifica as abstrações $\theta^T_5(S_5)$, $\theta^T_6(S_6)$ e $\theta^T_7(S_7)$ como OP-abstrações, não sendo necessário expandir o conjunto inicial de eventos relevantes. Para a T-abordagem. obtém-se os autômatos abstraídos $\theta^T_j(S_j)$, $j = 1, \ldots, 8$, onde $\theta^T_j(S_j)$ são as projeções para a T-abordagem, para $j = 1, \ldots, 8$. Apresentam-se os números de estados dos autômatos abstraídos na Tabela 2. Verifica-se o conflito pelo cálculo do autômato $S^T_{mod} = \|_{j=1}^{8} \theta^T_j(S_j)$, com 51.300 es-

tados (dos quais 32.724 são não-bloqueantes), 213.648 transições e 28 eventos.

Para a $W \circ T$-abordagem, aplica-se a W-abordagem ao resultado da T-abordagem, isto é, submetem-se os supervisores locais abstraídos $\theta_j^T(S_j)$, $j = 1, \ldots, 8$, à W-abordagem. Neste segundo passo, os eventos não-relevantes iniciais são 11, 12, 21, 22, 41, 42, 51, 52, 53, 54, 81 e 82, e os supervisores com eventos neste conjunto são $\theta_1^T(S_1) = S_1$, $\theta_2^T(S_2) = S_2$, $\theta_3^T(S_3) = S_3$, $\theta_4^T(S_4) = S_4$ e $\theta_8^T(S_8) = S_8$. Observa-se que os supervisores afetados neste passo são os mesmos que na W-abordagem original acima. Também como na W-abordagem original, o OP-verificador classifica $\theta_3^W(S_3)$ e $\theta_8^W(S_8)$ como OP-abstrações e, se os eventos 11, 21, 51 e 53 forem adicionados a Σ_r, $\theta_1^W(S_1)$, $\theta_2^W(S_2)$ e $\theta_4^W(S_4)$ também são classificados como OP-abstrações pelo OP-verificador. Assim, segundo a $W \circ T$-abordagem, as OP-abstrações a serem usadas para testar o conflito são: $\theta_5^T(S_5)$, $\theta_6^T(S_6)$, $\theta_7^T(S_7)$, $\theta_1^W(S_1)$, $\theta_2^W(S_2)$, $\theta_3^W(S_3)$, $\theta_4^W(S_4)$ e $\theta_8^W(S_8)$, com dados mostrados na Tabela 2. Verifica-se então o conflito pelo cálculo do autômato

$$S_{mod}^{W \circ T} = \|_{j=1}^{4} \theta_j^W(S_j) \|_{j=5}^{7} \theta_j^T(S_j) || \theta_8^W(S_8)$$

com 6.544 estados (dos quais 4.720 são não-bloqueantes), 22.480 transições e 20 eventos.

Tabela 2: Dados referentes às abstrações.

j	$\lvert S_j \rvert$	$\lvert \theta_j^W(S_j) \rvert$	$\lvert \theta_j^T(S_j) \rvert$	$\lvert \theta_j^W(\theta_j^T(S_j)) \rvert$
1	18	12	18	12
2	18	12	18	12
3	18	10	18	10
4	21	15	21	15
5	44	44	22	22
6	44	44	22	22
7	128	128	64	64
8	6	4	6	4
78	164	124	82	62

A parte de cima da Tabela 3 compara os desempenhos das quatro abordagens ao mostrar o número de estados do autômato composto para verificação do conflito. Neste exemplo, a $W \circ T$-abordagem apresenta o melhor desempenho ao reduzir o número de estados a serem testados a 10% do número original de estados.

Propõe-se então um esquema de resolução de conflito para viabilizar o emprego da supervisão modular local. A topologia do sistema sugere que a origem do conflito está localizada nas especificações E_7 e E_8, envolvendo as plantas para o Robô, a esteira C_3, MP e MM. Assim, uma abordagem possível é combinar as duas especificações numa nova especificação local $E_{78} = E_7 \| E_8$ com planta local dada pela composição dos subsistemas afetados. Os dados relativos à síntese do supervisor S_{78} são mostrados na última linha da Tabela 1. A verificação do conflito baseada na igualdade (1) mostra que o autômato $S'_{mod} = \left(\|_{j=1}^{6} S_j \right) \| S_{78}$ é não-bloqueante, com 45.504 estados, 200.124 transições e 31 eventos, confirmando assim a validade do esquema de resolução de conflito proposto.

Tabela 3: Desempenho das abordagens.

Sistema	Abord.	Tamanho	Redução (%)
Bloqueante	Conv.	70.272	0,0
	W-	9.080	87,1
	T-	51.300	27,0
	$W \circ T$-	6.544	90,7
Não-bloqueante	Conv.	45.504	0,0
	W-	6.656	85,4
	T-	32.724	28,1
	$W \circ T$-	4.720	89,6

Para o novo sistema não-bloqueante, o teste de não conflito também foi realizado utilizando-se as abordagens W, T and $W \circ T$. Pode-se verificar que os correspondentes conjuntos de eventos relevantes são os mesmos que no caso do sistema bloqueante. Os dados referentes às abstrações do supervisor S_{78} são apresentados na última linha da Tabela 2. Ainda, a Tabela 3 compara o desempenho das quatro abordagens, em termos do número de estados do autômato composto para o teste de não conflito, agora para o caso do sistema não-bloqueante. Novamente, a $W \circ T$-abordagem proporciona um teste de não conflito com aproximadamente 10% dos estados, quando comparada com o teste convencional.

6 CONCLUSÕES

Este artigo apresenta uma contribuição para contornar o problema de complexidade computacional do teste de não-conflito entre supervisores modulares em Sistemas a Eventos Discretos. A nova estratégia proposta para a construção de OP-abstrações mostra-se eficiente como técnica de obtenção de importantes reduções no tamanho dos autômatos usados no teste. Por outro lado, a ferramenta computacional desenvolvida para a verificação da propriedade de observador auxilia na busca de OP-abstrações. A síntese de supervisores através da abordagem modular local associada à possibilidade de agilização no processo de obtenção de OP-abstrações dão ao projetista a possibilidade de buscar configurações alternativas de controle modular quando soluções prévias levam ao bloqueio. Os autores investigam atual-

mente como o OP-verificador pode ser melhorado para apontar automaticamente para conjuntos de eventos relevantes que levem a OP-abstrações com bons níveis de redução dos autômatos correspondentes.

AGRADECIMENTOS

A pesquisa do primeiro autor é apoiada parcialmente pela PRPq/UFMG. O primeiro autor agradece ainda à FAPEMIG e CNPq. O terceiro autor é apoiado parcialmente pelo CNPq (PQ 300953/93-3). Os três primeiros autores são apoiados parcialmente pela CAPES (Procad 102/2007). A pesquisa do quarto autor é apoiada em parte pelo NSF (grant ECS-0624821).

REFERÊNCIAS

Cormen, T. H., Leiserson, C. E. e Rivest, R. L. (1990). *Introduction to Algorithms*, The MIT Press.

Cunha, A. e Cury, J. (2007). Hierarchical Supervisory Control Based on Discrete Event Systems with Flexible Marking, *IEEE Transactions on Automatic Control* **52**(12): 2242–2253.

Cury, J. (2009). Grail para Controle Supervisório, `http://www.das.ufsc.br/~cury/grail.html`. Visitada em março de 2009.

de Queiroz, M. e Cury, J. (2000). Modular Supervisory Control of Large Scale Discrete Event Systems, *Proceedings of the 5th Workshop on Discrete Event Systems, WODES'00*, Vol. 1, Ghent, Belgium, pp. 103–110.

de Queiroz, M. e Cury, J. (2002). Controle Supervisório Modular de Sistemas de Manufatura, *Revista Controle & Automação* **13**(2): 115–125.

de Queiroz, M., Cury, J. e Wonham, W. (2005). Multitasking Supervisory Control of Discrete-Event Systems, *Discrete Event Dynamic Systems: Theory and Applications* **15**: 375–395.

Feng, L. (2006). On the Computation of Natural Observers in Discrete-Event Systems, *Technical report*, Systems Control Group Report, University of Toronto.

Feng, L. e Wonham, W. (2006). Computationally Efficient Supervisor Design: Abstraction and Modularity, *Proceedings of the 8th International Workshop on Discrete Event Systems, WODES'06*, Ann Arbor, MI, USA, pp. 3–8.

Garcia, T. (2002). *Controle Supervisório de Sistemas a Eventos Discretos: Uma Abordagem por Modelo Condição/Evento*, Master's thesis, Universidade Federal de Santa Catarina, UFSC, Florianópolis, SC, Brasil.

Hill, R., Cury, J., de Queiroz, M. e Tilbury, D. (2008). Modular Requirements for Hierarchical Interface-Based Supervisory Control with Multiple Levels, *Proceedings of the 2008 American Control Conference, ACC'08*, Seattle, WA, USA, pp. 483–490.

Hill, R. e Tilbury, D. (2006). Modular Supervisory Control of Discrete Event Systems with Abstraction and Incremental Hierarchical Construction, *Proceedings of the 8th International Workshop on Discrete Event Systems, WODES'06*, Ann Arbor, MI, USA, pp. 399–406.

Hill, R., Tilbury, D. e Lafortune, S. (2008). Modular Supervisory Control with Equivalence-based Conflict Resolution, *Proceedings of the 2008 American Control Conference, ACC'08*, Seattle, WA, USA, pp. 491–498.

Leal, A. (2005). *Controle Supervisório Modular de Sistemas Híbridos*, PhD thesis, Universidade Federal de Santa Catarina, UFSC, Florianópolis, SC, Brasil.

Leduc, R., Lawford, M. e Wonham, W. (2005). Hierarchical Interface-Based Supervisory Control - Part II: Parallel Case, *IEEE Transactions on Automatic Control* **50**(9): 1336–1348.

Pena, P., Cunha, A., Cury, J. e Lafortune, S. (2008). Teste de Não-Conflito de Supervisores Modulares Usando o OP-Verificador, *Anais do XVII Congresso Brasileiro de Automática, CBA'08*.

Pena, P., Cury, J. e Lafortune, S. (2006a). Detecção de Conflito na Síntese Modular de Supervisores: Uma Abordagem Baseada em Abstrações, *Anais do XVI Congresso Brasileiro de Automática, CBA'06*, Salvador, Brasil, pp. 977–982.

Pena, P., Cury, J. e Lafortune, S. (2006b). New Results on Testing Modularity of Local Supervisors using Abstractions, *11th IEEE International Conference on Emerging Technologies and Factory Automation*, pp. 950–956.

Pena, P., Cury, J. e Lafortune, S. (2007). Polynomial-Time Verification of the Observer Property in Abstractions, *Technical report*, Discrete Event Systems Group Report, Department of Automation and Systems (DAS) - Federal University of Santa Catarina.

Pena, P., Cury, J. e Lafortune, S. (2009). Verification of Nonconflict of Supervisors Using Abstractions, *Aceito para IEEE Transactions on Automatic Control* .

Ramadge, P. e Wonham, W. (1987). Supervisory Control of a Class of Discrete Event Processes, *SIAM Journal Control and Optimization* pp. 206–230.

Raymond, D. e Wood, D. (1994). GRAIL: A C++ Library for Automata and Expressions„ *Journal of Symbolic Computation,* **17**: 341–350. Artigo sobre o grail.

Reiser, C., da Cunha, A. e Cury, J. (2006). The Environment Grail for Supervisory Control of Discrete Event Systems, *Proceedings of the 8th International Workshop on Discrete Event Systems, WODES'06*, Ann Arbor, MI, USA, pp. 390–391.

Schmidt, K., de Queiroz, M. e Cury, J. (2007). Hierarchical and Decentralized Multitasking Control of Discrete Event Systems, *Proceedings of the 46th IEEE Conference on Decision and Control, CDC07*, New Orleans, USA, pp. 5936–5941.

Schmidt, K., Reger, J. e Moor, T. (2004). Hierarchical Control for Structural Decentralized DES, *Proceedings of the 7th Workshop on Discrete Event Systems, WODES'04*.

Sivolella, L. F., da Cunha, A. E. C. e Ades, R. (2006). Redução de Supervisores como Ferramenta para Implementação de Controladores Discretos, *Anais do XVI Congresso Brasileiro de Automática, CBA'06*, Salvador, Brasil, pp. 2778–2783.

Su, R. e Wonham, W. (2004). Supervisor Reduction for Discrete-Event Systems, *Discrete Event Dynamic Systems: Theory and Applications* **14**: 31–53.

Torrico, C. e Cury, J. (2004). Controle Supervisório Hierárquico Modular por Agregação de Estados, *Revista Controle & Automação* **15**(3): 291–300.

Wong, K. (1998). On the Complexity of Projections of Discrete-Event Systems, *Proceedings of the 4th Workshop on Discrete Event Systems, WODES'98*, Cagliari, Italy.

Wong, K., Thistle, J., Malhamé, R. e Hoang, H.-H. (2000). Supervisory Control of Distributed Systems: Conflict Resolution, *Discrete Event Dynamic Systems: Theory and Applications* **10**: 131–186.

Wong, K. e Wonham, W. M. (1996). Hierarchical Control of Discrete-Event Systems, *Discrete Event Dynamic Systems: Theory and Applications* **6**(3): 241–273.

Wonham, W. e Ramadge, P. (1988). Modular Supervisory Control of Discrete Event Systems, *Math. Control, Signals Systems* **1**: 13–30.

Yoo, T. e Lafortune, S. (2002). Polynomial-time verification of diagnosability of partially observed discrete-event systems, *IEEE Transactions on Automatic Control* **47**(9): 1491– 1495.