ABSTRAÇÃO DO SUPERVISOR PARA APLICAÇÃO NA SOLUÇÃO DE PROBLEMAS DE PLANEJAMENTO EM SISTEMAS DE MANUFATURA

JULIANA N. VILELA*, PATRÍCIA N. PENA†

*Programa de Pós-Graduação em Engenharia Elétrica
Universidade Federal de Minas Gerais
Belo Horizonte, MG, Brasil

†Departamento de Engenharia Eletrônica
Universidade Federal de Minas Gerias
Belo Horizonte, MG, Brasil

Emails: julianavilela@gmail.com, ppena@ufmg.br

Abstract— Finite-state automata and Supervisory Control Theory have been used to model and solve job-shop scheduling and planning problems. However, even if it seems to be easier to work with DFA and SCT, this solution will suffer with a state explosion when the system becomes bigger and more complex, what turns difficult the use of this solution to solve scheduling problems. This paper presents a set of sufficient conditions that allows to work with an abstraction of the supervisor into the set of controllable events, instead of the supervisor itself, as the search universe to solve a planning problem. Also, we present a set of conditions for the model of the system and specifications that will results on the satisfaction of the previous conditions by the supervisor abstraction.

Keywords— Planning, Supervisor, Abstraction, Observer Property.

Resumo— Autômatos de estados-finitos determinísticos (ADF) e a Teoria de Controle Supervisório (TCS) têm sido muito utilizados para modelar e resolver problemas de escalonamento de *job-shop* e de planejamento. Contudo, esta solução sofre de uma explosão de estados quando o sistema se torna maior e mais complexo, o que torna o problema de buscar soluções de problemas de escalonamento computacionalmente caro. Este artigo apresenta um conjunto de condições suficientes que permitem trabalhar com uma abstração do supervisor sobre os eventos controláveis, ao invés do próprio supervisor, como universo de busca para a solução de problemas de planejamento. Também é apresentado um conjunto de características para modelar o sistema e suas especificações, de tal forma que a abstração do supervisor satisfaz as condições apresentadas diretamente.

Palavras-chave— Planejamento, Supervisor, Abstração, Propriedade do Observador.

1 Introdução

Devido à crescente competitividade, as indústrias desejam produzir cada vez mais, mais rápido e utilizando todos os recursos disponíveis. De acordo com Baker and Trietsch (2009), o problema relacionado a essa necessidade já é conhecido e é definido como sendo um problema de escalonamento de *job-shop*. Tal problema tem como objetivo organizar um conjunto de tarefas sobre um conjunto de recursos de tal forma que algum critério seja otimizado, como o *makespan* ou custo, por exemplo. Os problemas de escalonamento podem ser vistos como problemas de otimização com restrições, e também, conforme Ghallab et al. (2004), como problemas de planejamento.

À medida que as indústrias se tornam cada vez mais complexas, modelá-las nas linguagens matemáticas comumente usadas pelas técnicas clássicas para solução dos problemas de escalonamento passou a ser uma tarefa extremamente difícil. Logo, a procura por um novo método de modelagem se tornou uma proeminente área de pesquisa. Os autômatos de estados-finitos determinísticos (ADF) e a Teoria de Controle Supervisório (TCS) proposta por Ramadge and Wonham (1989) vêm ganhando espaço nessa área de pesquisa. Tais técnicas têm sido utilizadas tanto para modelar o comportamento de sistemas, como pode ser visto em Kobetski and Fabian (2006), Abdedda and Maler (2001), Pena et al. (2016); quanto para resolver todo o problema de planejamento, como visto em Su (2012), Fabre and Jezequel (2009) e Nishi et al. (2009).

De acordo com Ghallab et al. (2004), o problema de planejamento tem como objetivo encontrar um caminho entre o estado inicial do sistema e um estado de interesse, ordenando as tarefas de tal forma que um determinado critério seja otimizado. A dinâmica do sistema é modelada por meio de "eventos", que simplesmente acontecem, e "ações", que podem ser controladas (desabilitadas, forçadas). Essa divisão dos eventos é bem similar à partição dos eventos da TCS, que os divide em controláveis e não-controláveis. Logo, se torna direto associar o modelo de Ghallab et al. (2004) para a solução de problemas de planejamento com a TCS de Ramadge and Wonham (1989).

A TCS possui uma característica conhecida como *"a maldição da dimensionalidade"*, que é a explosão de estados quando o sistema cresce e se torna mais complexo. Uma crescente área de estudos existe com o objetivo de buscar meios para se contornar esse problema. Quando o supervisor é utilizado como universo de busca por algum algoritmo de otimização, como é o caso em algumas soluções do problema de planejamento, é interessante usar uma abstração do supervisor que mantenha as "boas" propriedades do supervisor e possua um espaço de estados menor. Assim, a redução do espaço de estados do supervisor implica na redução do universo de busca utilizado pelo algo-

ritmo, podendo acarretar em uma melhora no desempenho deste. Além disso, uma vez que só é possível agir sobre os eventos controláveis, os eventos não-controláveis se tornam dispensáveis. Surge então uma importante questão: Como obter uma abstração do supervisor que mantenha todos os seus eventos controláveis e também as "boas" propriedades deste? Por "boas" propriedades se entende a execução completa na planta de qualquer plano escolhido na abstração do supervisor. Isto pode ser traduzido como a verificação da controlabilidade da linguagem construída a partir do plano.

Este artigo mostra que se a propriedade do observador, proposta em Wong et al. (1998), é verificada para a projeção do supervisor sobre os eventos controláveis, então o plano obtido sempre poderá ser executado sobre o sistema. Como um resultado adicional, também foi estabelecido um conjunto de características para a modelagem do sistema que garante que a projeção do supervisor sobre os eventos controláveis sempre terá a propriedade do observador.

O artigo é organizado da seguinte maneira. Seção 2 introduz os conhecimentos necessários para o presente trabalho. Seção 3 apresenta o resultado principal e a demonstração, além de dois exemplos para ilustração das ideias. No primeiro exemplo, a propriedade do observador é satisfeita e a controlabilidade da linguagem é verificada. O exemplo seguinte mostra que a controlabilidade não é sempre possível de ser obtida quando a propriedade do observador não é satisfeita. Seção 4 mostra um conjunto de características propostas para a modelagem do sistema que irá resultar na satisfação da propriedade do observador na projeção do supervisor. Seção 5 traz alguns exemplos de aplicações dos resultados obtidos no artigo. Finalmente, comentários finais e conclusões são apresentados na Seção 6.

2 Preliminares

Esse artigo foi desenvolvido sob o formalismo da Teoria de Controle Supervisório de Ramadge and Wonham (1989). Para uma revisão detalhada da teoria é sugerida uma consulta a Cassandras and Lafortune (2008). O comportamento de um SED é modelado usando cadeias de eventos pertencentes a um alfabeto finito Σ. Σ^* é o conjunto de todas as cadeias finitas formadas por eventos de Σ, incluindo a cadeia vazia ε. A operação de concatenação de duas cadeias $s,u \in \Sigma^*$ é escrita como su. Uma cadeia $s \in \Sigma^*$ é considerada prefixo de outra cadeia $t \in \Sigma^*$, $s \leq t$, se existe $u \in \Sigma^*$ tal que $su = t$. Um subconjunto $L \subseteq \Sigma^*$ é chamado de linguagem. O prefixo-fechamento $\overline{L}$ de uma linguagem $L \subseteq \Sigma^*$ é o conjunto de todos os prefixos das cadeias que compõem L. Uma linguagem regular pode ser representada por um autômato de estados-finitos. Um autômato de estados-finitos é uma tupla $G = (\Sigma, Q, \rightarrow, Q_m, q_0)$, onde Σ é um conjunto finito de eventos, Q é um conjunto finito de estados, $\rightarrow \subseteq Q \times \Sigma \times Q$ é a função de transição, $Q_m \subseteq Q$ é o conjunto finito de estados marcados, e $q_0 \in Q$ é o estado inicial. G é determinístico, se $x \xrightarrow{\sigma} y_1$ e $x \xrightarrow{\sigma} y_2$ sempre implica em $y_1 = y_2$. Finalmente, a linguagem gerada de um autômato G é definida como $\mathscr{L}(G) = \{s \in \Sigma^* \mid G \xrightarrow{s}\}$ e a linguagem marcada é $\mathscr{L}_m(G) = \{s \in \Sigma^* \mid G \xrightarrow{s} Q_m\}$.

A projeção natural $\theta_i : \Sigma^* \rightarrow \Sigma_i^*$ é uma operação sobre linguagens definidas sobre conjuntos de eventos Σ e Σ_i, em que $\Sigma_i \subseteq \Sigma$. A projeção natural mapeia as cadeias de Σ^* para cadeias em Σ_i^*, apagando todos os eventos que não estão em Σ_i. A projeção natural pode ter uma propriedade conhecida como propriedade do observador definida por Wong et al. (1998) apresentada na Definição 1.

Definição 1 *(Wong et al. (1998)): Seja uma linguagem $L \subseteq \Sigma^*$, um alfabeto $\Sigma_i \subseteq \Sigma$ e $\theta : \Sigma^* \longrightarrow \Sigma_i^*$ a projeção natural de cadeias em Σ^* para cadeias em Σ_i^*. Se $(\forall a \in \overline{L})$ $(\forall b \in \Sigma_i^*)$, $\theta(a)b \in \theta(L) \Rightarrow (\exists c \in \Sigma^*)$ $\theta(ac) = \theta(a)b$ e $ac \in L$,então, a projeção natural $\theta(L)$ tem a propriedade do observador.*

Além disso, se um autômato possui a propriedade do observador, de acordo com Wong and Wonham (1996), o autômato resultante da projeção sempre será menor ou no máximo do mesmo tamanho do autômato original (caso $\Sigma_i = \Sigma$).

2.1 Teoria de Controle Supervisório

O conjunto de eventos usado para modelar um sistema G (o qual será referido como planta) pode ser dividido em dois subconjuntos, $\Sigma = \Sigma_c \cup \Sigma_{uc}$, sendo Σ_c o subconjunto de eventos controláveis, e Σ_{uc}, o subconjunto de eventos não-controláveis. O supervisor atua sobre os eventos controláveis, restringindo o comportamento da planta para um subconjunto de $\mathscr{L}(G)$. A linguagem desejada K é obtida realizando a composição paralela das especificações E com $\mathscr{L}_m(G)$. Entretanto, nem sempre a implementação de K é possível de ser feita, uma vez que pode ser necessário desabilitar eventos não-controláveis para obter K. Se esse é o caso, então K é dita ser não-controlável em relação a $\mathscr{L}(G)$. Contudo, existe uma máxima sublinguagem controlável, $Sup\mathscr{C}(K,G)$, que implementa o comportamento mais permissivo e não-bloqueante do sistema que respeita as especificações. O supervisor S implementa $S = Sup\mathscr{C}(K,G)$.

Para que uma linguagem $K \subseteq \Sigma^*$ seja controlável em relação a uma dada planta $G \subseteq \Sigma^*$, é necessário que $\overline{K}\Sigma_{uc} \cap \mathscr{L}(G) \subseteq \overline{K}$. Ou seja, se K desabilita algum evento não-controlável quando este está habiltado na planta G, então K é não-controlável em relação a G.

3 Resultado Principal

O supervisor obtido pela TCS desabilita eventos controláveis para evitar a ocorrência de comportamentos ilegais na planta, contudo, ele não escolhe entre os eventos habilitados qual deve ser executado primeiro

ou em qual ordem estes devem ser executados. Logo, o supervisor garante o não-bloqueio e a segurança do sistema, mas não a sua performance. No problema de planejamento, a ideia é escolher uma ordem de execução dos eventos de tal maneira, que algum critério seja otimizado. Se as seguintes ideias forem unidas, (i) a manipulação apenas dos eventos controláveis; (ii) definição de uma ordem de execução dos eventos; é sensato considerar que a solução do problema de planejamento pode ser definida apenas sobre o conjunto de eventos controláveis.

Seja o alfabeto $\Sigma = \Sigma_c \cup \Sigma_{uc}$, Σ_c o subconjunto de eventos controláveis e Σ_{uc} o subconjunto de eventos não-controláveis. O autômato G modela o sistema $\mathscr{L}(G) \subseteq \Sigma^*$, e o autômato S é um supervisor que implementa a máxima sublinguagem controlável da linguagem desejada K, $S = Sup\mathscr{C}(K,G)$ (S é usado tanto para a linguagem quanto para o autômato que implementa a linguagem). $\theta : \Sigma^* \to \Sigma_c^*$ é uma projeção natural.

Sendo $s_{ot} \in \theta(S) \subseteq \Sigma_c^*$ uma solução para o problema de planejamento, para que essa sequência realmente seja uma solução possível, é necessário que ela possa ser implementada no sistema real. Para tanto reconstrói-se a linguagem $A = \theta^{-1}(s_o t) \cap \mathscr{L}(S) \subseteq \mathscr{L}(S)$, que é o subconjunto de cadeias de S que projetam em s_{ot}. Assim, para que seja possível implementar A na planta G, A precisa ser controlável em relação a $\mathscr{L}(G)$. O teorema a seguir mostra sob quais condições sempre é possível garantir isso.

Teorema 1 *Seja G um sistema, S a máxima sublinguagem controlável contida na linguagem desejada K e $\theta : \Sigma^* \to \Sigma_c^*$ a projeção natural. Para toda sequência $s_{ot} \in \theta(S)$ e $A = \theta^{-1}(s_{ot}) \cap S$, se $\theta(S)$ possui a propriedade do observador, então A sempre será controlável em relação a $\mathscr{L}(G)$.*

Prova: Para mostrar que A é controlável em relação a $\mathscr{L}(G)$, é preciso mostrar que $\forall a \in \overline{A}$ e $\forall \sigma \in \Sigma_{uc}$ com $a\sigma \in \mathscr{L}(G)$, então $a\sigma \in \overline{A}$.

Uma cadeia qualquer $s_{ot} \in \theta(S) \subseteq \Sigma_c^*$ é escolhida. Para $a \in \overline{S}$ tal que $\theta(a) \leq s_{ot}$ e $a\sigma \in \mathscr{L}(G)$, com $\sigma \in \Sigma_{uc}$, por construção, tem-se que $a \in \overline{A}$.

Como S é controlável, tem-se que:

$$\forall \sigma \in \Sigma_u, a\sigma \in \mathscr{L}(G) \Rightarrow a\sigma \in \overline{S}. \quad (1)$$

Da equação (1), conclui-se que $\theta(a\sigma) \in \theta(\overline{S})$. Pela premissa inicial $\theta(a) \leq s_{ot}$, então $\exists b \in \Sigma_c^*$ tal que:

$$\theta(a)b = \theta(a\sigma)b = s_{ot} \in \theta(S). \quad (2)$$

Uma vez que $\theta(S)$ tem a propriedade do observador, Definição 1, $\exists t \in \Sigma^*$ tal que

$$\theta(a\sigma t) = \theta(a\sigma)b \text{ e} \quad (3)$$

$$a\sigma t \in S. \quad (4)$$

De (2) e (3) tem-se que a $\theta(a\sigma t) = s_{ot}$. Então:

$$a\sigma t \in \theta^{-1}(s_{ot}). \quad (5)$$

A partir de $A = \theta^{-1}(s_{ot}) \cap S$ e das equações (4) e (5), conclui-se que $a\sigma t \in A$, logo:

$$a\sigma \in \overline{A}.$$

□

A verificação da propriedade do observador pode ser feita usando o OP-verificador definido em Pena et al. (2014) e que possui complexidade quadrática no número de estados do autômato original.

3.1 Exemplos

Para ilustrar a ideia exposta anteriormente, dois exemplos são apresentados. O primeiro exemplo mostra um caso em que a controlabilidade é verificada, já o segundo exemplo mostra um caso em que a verificação da controlabilidade falha.

Exemplo 1 *Considere um pequeno sistema de manufatura apresentado nas Fig. 1 e Fig. 2. O sistema funciona, conforme especificação (Fig. 2d). Os eventos a_i são controláveis e os b_i são não-controláveis, $i = \{1,2,3\}$. Para ilustrar a linguagem A, uma cadeia qualquer que produz um produto foi escolhida no supervisor projetado $\theta(S)$, Fig. 3 (que tem a propriedade do observador) $s_{ot} = a_1a_2a_3$. A linguagem A é reconhecida pelo autômato da Fig. 4. A estrutura de desabilitação de A em relação à planta $G = M_1||M_2||M_3$ é mostrada com setas pontilhadas originadas nos estados, e apenas eventos controláveis são desabilitados, logo, pode-se concluir que A é controlável em relação a G.*

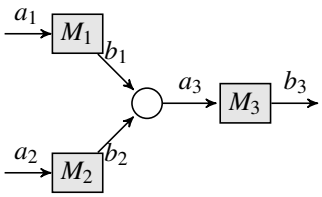

Figura 1: Exemplo 1: Diagrama.

Exemplo 2 *Considere o problema da pequena fábrica, adaptado de Wonham (2015). A máquina M_2 não tem os eventos de quebra e reparo representados em seu modelo. Os autômatos que modelam as máquinas e a especificação do sistema estão na Fig. 6, com $a_i \in \Sigma_c$ e $b_i \in \Sigma_{uc}$, $i = \{1,2,3\}$. Como no exemplo anterior, uma sequência qualquer que completa a produção de um produto é escolhida, $s = a_1a_3 \in \theta(S)$, Fig. 7, $\theta(S)$ não atende as condições do Teorema 1. O autômato que implementa a linguagem recuperada A está na Fig. 8, juntamente com a estrutura de desablilitação. Neste caso, existem eventos não-controláveis entre os eventos que são desabilitados, especificamente, o evento b_2 é desabilitado no estado q_1. Logo, a linguagem A não é controlável em relação à planta $G = M_1||M_2$.*

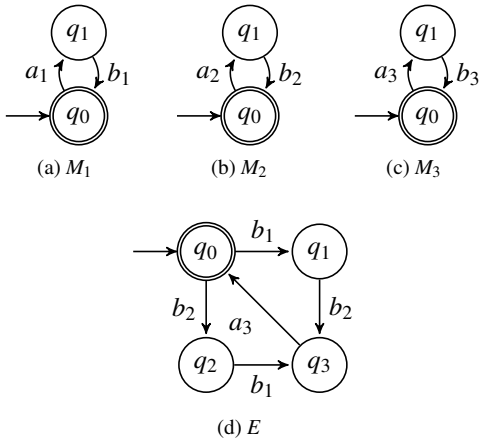

Figura 2: Exemplo 1: Autômatos para M_1, M_2, M_3 e E

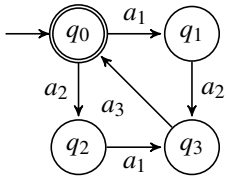

Figura 3: Exemplo 1: Autômato de $\theta(S)$.

4 Outro resultado

Na seção anterior, foi demonstrado que é possível trabalhar com a projeção do supervisor nos eventos controláveis se a propriedade do observador é verificada. Contudo, se o problema de controle é modelado seguindo algumas características apresentadas abaixo, a propriedade do observador sempre será atendida quando o supervisor for projetado nos eventos controláveis.

A seguir são apresentados o sistema de produção e a especificação de coordenação.

Definição 2 *Um sistema de produção G é definido como um sistema composto por m máquinas modeladas pelos autômatos* $M_i = (\Sigma_i, Q_i, \rightarrow_i, Q_i^m, q_i^0)$, $i \in I$, $I = \{1,...,m\}$ *onde:*

i) $G = \|_{i=1}^{m} M_i$, $\Sigma = \bigcup_{i=1}^{m} \Sigma_i$;

ii) $\Sigma_i \cap \Sigma_j = \emptyset$, $i, j \in I$ *e* $i \neq j$;

iii) M_i *é caracterizada por:*

- *a-* $\forall \sigma \in \Sigma_i$ *tal que* $q_0 \xrightarrow{\sigma}_i$ *então* $\sigma \in \Sigma_{ic}$, Σ_{ic} *é o subconjunto de eventos controláveis de* Σ_i;
- *b-* *Para a projeção natural* $\theta : \Sigma_i^* \rightarrow (\Sigma_i \cap \Sigma_{ic})^*$, $\theta(M_i)$ *possui a propriedade do observador.*

Um *sistema de produção* é composto por máquinas com conjuntos de eventos disjuntos. Estas máquinas sempre são iniciadas com um evento controlável,

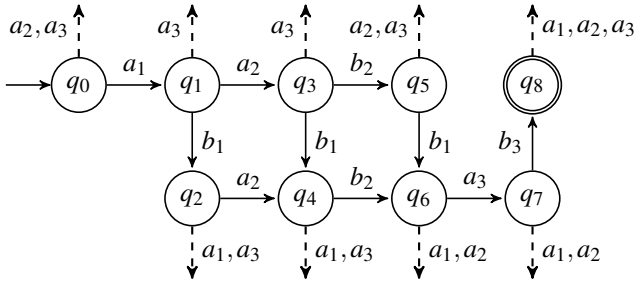

Figura 4: Exemplo 1: Autômato que implementa A.

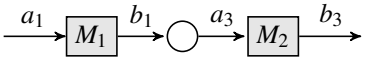

Figura 5: Exemplo 2: Pequena Fábrica com Quebra.

e suas projeções sobre os eventos controláveis têm a propriedade do observador.

Para a sequência $s \in L$, $T_L(s) = \{t \in \Sigma^* | st \in L\}$ é o conjunto de sufixos de s que pertencem a linguagem L.

Definição 3 *A especificação* $E = (\Sigma_E, Q_E, \rightarrow_E, Q_E^m, q_E^0)$ *é uma especificação de coordenação se E é controlável em relação a* $\mathscr{L}(G)$ *e* $\forall s \in \Sigma_E^*$, $t \in (\Sigma_E \cap \Sigma_{uc})^*$ *e* $\sigma \in (\Sigma_E \cap \Sigma_c)$ *onde* $q_0 \xrightarrow{s}_E q \xrightarrow{t}_E q' \xrightarrow{\sigma}_E$. *Se* $\exists u \in (\Sigma_E \cap \Sigma_{uc})^* \cap T_{\mathscr{L}(E)}(s)$ *e* $u \neq t$, *e u é uma permutação dos eventos de t, então necessariamente* $q_0 \xrightarrow{su}_E q'' \xrightarrow{\sigma}_E$.

Em palavras, se um evento controlável σ está habilitado após uma cadeia $st \in \mathscr{L}(E)$ sendo que t é uma cadeia composta apenas por eventos não-controláveis, se existe uma outra cadeia u que seja uma permutação dos eventos de t, com $su \in \mathscr{L}(E)$ então σ necessariamente estará habilitado após a cadeia su.

Com estas definições é possível estabelecer o segundo resultado.

Proposição 1 *Seja a planta* $G = (\Sigma, Q, \rightarrow_G, Q_m, q^0)$, $\Sigma = \Sigma_c \cup \Sigma_{uc}$, *um sistema de produção de acordo com a Definição 2. Seja* $E = (\Sigma_E, Q_E, \rightarrow_E, Q_E^m, q_E^0)$ *uma especificação de coordenação conforme a Definição 3 e* $\theta : \Sigma^* \rightarrow \Sigma_c^*$ *uma projeção natural. Como* $K = G||E$ *é controlável (*$S = Sup(K,G) = K$*) então* $\theta(S)$ *tem a propriedade do observador.*

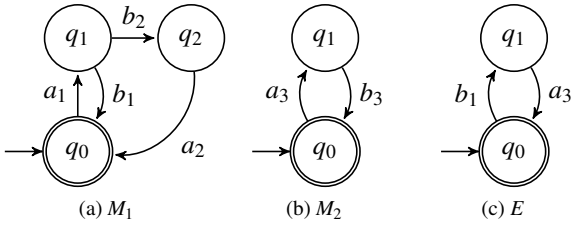

Figura 6: Exemplo 2: Autômatos para M_1, M_2 e E.

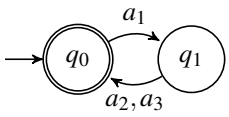

Figura 7: Exemplo 2: Autômato de $\theta(S)$.

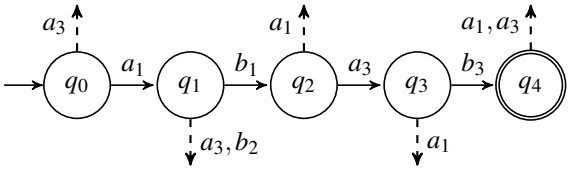

Figura 8: Exemplo 2: Autômato que implementa A.

A demonstração deste resultado encontra-se em Vilela and Pena (2016).

Nota 1 *Caso E não seja controlável, o supervisor S é obtido e o supervisor reduzido é utilizado ao invés da especificação E, conforme estabelecido em Pena et al. (2009). Supervisores reduzidos podem ser obtidos pelo algoritmo de complexidade polinomial proposto por Su and Wonham (2004) de tal forma que $S = S_{red}||G$.*

Exemplo 3 *A planta do Exemplo 1 pode ser classificada como um sistema de produção. A especificação, apresentada na Fig. 2d, quando composta com $\mathscr{L}_m(G)$, resulta em uma linguagem não-controlável K. Para que a Definição 3 seja verificada na especificação é necessário calcular o supervisor reduzido S_{red} (Fig. 9) e, assim utilizar $E_c = S_{red}$. Conclui-se que a especificação E_c é uma especificação de coordenação.*

Nota 2 *A planta G do Exemplo 2 não é um sistema de produção, já que M_1 não segue a Definição 2.iii-b), ou seja $\theta(M_1)$ não possui a propriedade do observador.*

5 Aplicações

O uso apenas dos eventos controláveis na solução de problemas de escalonamento de *job-shop* e de planejamento é uma prática comum em diversas abordagens que utilizam ADF e a TCS. Os trabalhos de Kobetski and Fabian (2006), Fabre and Jezequel (2009) e Pena et al. (2016) são exemplos.

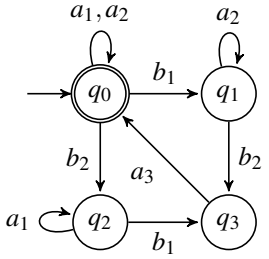

Figura 9: Exemplo 3: Autômato que implementa $E_c = S_{red}$.

Kobetski and Fabian (2006) trabalham com células de manufatura robotizadas, modelando seus comportamentos com ADF e a TCS. Por uma questão de adaptação com o procedimento de otimização utilizado, todos os eventos são modelados como controláveis. Contudo, tal premissa faz com que informações sobre o sistema sejam perdidas e viola o paradigma da TCS, simplificando o problema demasiadamente.

Já Fabre and Jezequel (2009) buscam a solução para o problema de planejamento fatorado. O problema de planejamento é dividido em problemas menores, chamados de "agentes". Cada "agente" é modelado por um ADF, com o conjunto de eventos, Σ, composto apenas pelas "ações" do sistema, as quais de acordo com Ghallab et al. (2004) são controláveis, logo equivalentes aos eventos controláveis da TCS. Apesar do problema de planejamento buscar ordenar tarefas, as quais são representadas pelas "ações", não considerar os eventos não-controláveis na modelagem do sistema gera uma perda de informação grande sobre a dinâmica do mesmo.

Portanto, conclui-se que o apropriado é modelar o problema considerando os eventos não-controláveis e controláveis, e realizar o processo de otimização apenas sobre os eventos controláveis. Ou seja, modelar o sistema sob controle com um supervisor obtido pela TCS, sendo $\Sigma = \Sigma_c \cup \Sigma_{uc}$, e fazer a busca pela solução do problema de planejamento na abstração do supervisor sobre os eventos controláveis. Nas seções anteriores foram apresentadas sob quais condições é possível realizar tal abordagem.

Pena et al. (2016) já fazem uso dessa abordagem, propondo a combinação de técnicas de otimização baseadas em metaheurísticas com a TCS, para a solução de problemas de escalonamento de *job-shop*. Neste trabalho, os autores modelam o sistema por meio de um supervisor obtido pela TCS, utilizando tanto eventos controláveis quanto eventos não-controláveis. A partir desse supervisor é obtida uma sequência qualquer que completa a produção de um lote de produtos. Essa sequência então é entregue ao algoritmo de otimização que trabalha apenas com os eventos controláveis gerando novas sequências, que são permutações desses eventos. Este procedimento é repetido até que alguma condição de parada seja alcançada.

6 Conclusão

A TCS é um método interessante para garantir o não-bloqueio e o comportamento correto do sistema. Contudo, ela não trata problemas de performance do sistema. Ao se resolver um problema de planejamento, buscam-se sequências de comando, de tal forma que algum critério seja otimizado (normalmente, tempo ou custo).

O presente trabalho estabelece condições suficientes que garantem que a busca pelas sequências de solução do problema de planejamento pode ser reali-

zada sobre uma abstração do supervisor. As condições apresentadas garantem que qualquer sequência escolhida na versão abstraída do supervisor poderá ser executada no sistema sem nenhuma violação (a controlabilidade e a propriedade do observador são de grande importância nesse resultado).

A abstração do supervisor possui menos estados que a versão original do mesmo. Como consequência, o problema de otimização tona-se mais fácil de ser resolvido e a solução encontrada mais geral (já que ela é traduzida em uma linguagem onde os eventos não-controláveis não são forçados a acontecer em uma ordem específica).

A proposição estabelecida provê algumas características para a modelagem do sistema, as quais garantem que o a projeção do supervisor nos eventos controláveis irá possuir a propriedade do observador. Tais características são usualmente observadas em modelos de sistemas de manufatura.

Agradecimentos

Este trabalho foi suportado pela Capes - Brasil, CNPq e FAPEMIG.

Referências

Abdedda, Y. and Maler, O. (2001). Job-Shop Scheduling using Timed Automata, *Proc. CAV'01* pp. 478–492.

Baker, K. R. and Trietsch, D. (2009). *Principles of Sequencing and Scheduling*, 1st edn, John Wiley and Sons, Inc.

Cassandras, C. and Lafortune, S. (2008). *Introduction to Discrete Event Systems*, 2nd edn, Springer, New York.

Fabre, E. and Jezequel, L. (2009). Distributed optimal planning: An approach by weighted automata calculus, *Proceedings of the IEEE Conference on Decision and Control and 28th Chinese Control Conference* **section III**: 211–216.

Ghallab, M., Nau, D. and Traverso, P. (2004). *Automated Planning: theory and practice*, 1st edn, Elsevier, San Francisco.

Kobetski, A. and Fabian, M. (2006). Scheduling of discrete event systems using mixed integer linear programming, *Proc. of the 8th International Workshop on Discrete Event Systems* pp. 76–81.

Nishi, T., Wakatake, M. and Inuiguchi, M. (2009). Decomposition of timed automata for solving scheduling problems, *Conference Proceedings - IEEE International Conference on Systems, Man and Cybernetics* pp. 2459–2464.

Pena, P. N., Bravo, H. J., Cunha, A. E. C. d., Malik, R., Lafortune, S. and Cury, J. E. R. (2014). Verification of the Observer Property in Discrete Event Systems, *IEEE Transactions on Automatic Control* **59**(8): 2176–2181.

Pena, P. N., Costa, T. A., Silva, R. S. and Takahashi, R. H. C. (2016). Control of Flexible Manufacturing Systems under model uncertainty using Supervisory Control Theory and evolutionary computation schedule synthesis, *Information Sciences* **329**: 491–502.

Pena, P. N., Cury, J. E. R. and Lafortune, S. (2009). Verification of Nonconflict of Supervisors Using Abstractions, *IEEE Transactions on Automatic Control* **54**: 2803–2815.

Ramadge, P. J. G. and Wonham, W. M. (1989). The Control of Discrete Event Systems, *Proceedings of the IEEE* **1**: 81–98.

Su, R. (2012). Abstraction-based Synthesis of Timed Supervisors for Time-Weighted Systems, *Proc. 11th Int. Workshop on Discrete Event Syst. (WODES 2012)* pp. 128–134.

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

Vilela, J. N. and Pena, P. N. (2016). Supervisor Abstraction to Deal With Planning Problems in Manufacturing Systems, *13th Int. Workshop on Discrete Event Syst. (WODES 2016)* .

Wong, K. C., Thistle, J. G., Malhame, R. P. and Hoang, H. H. (1998). Supervisory Control of Distributed Systems: Conflict Resolution, *Proceedings of the 37th IEEE Conference on Decision and Control* pp. 3275–3280.

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

Wonham, W. M. (2015). Notes on Supervisory Control of Dicrete-Event Systems, *Course Notes for ECE 1636F/1637S* .