SINCRONIZAÇÃO EM SISTEMAS A EVENTOS DISCRETOS

Lucas V. R. Alves*, Patrícia N. Pena†

*Colégio Técnico
Universidade Federal de Minas Gerais
Belo Horizonte, Minas Gerais, Brasil

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

Emails: lucasvra@ufmg.br, ppena@ufmg.br

Abstract— Synchronizing Automata properties are very useful in the study of Discrete Event Systems. The ability to lead the system to a known state is important in industrial systems susceptible to external attacks and failures. In this paper we show under which conditions a set of synchronizing plants and specifications lead to a synchronizing supervisor obtained by the Supervisory Control Theory.

Keywords— Discrete Event Systems, Synchronizing Automata, Supervisory Control Theory.

Resumo— Propriedades de Autômatos Sincronizáveis são muito úteis no estudo de Sistemas a Eventos Discretos. A habilidade de levar o sistema a um estado conhecido é de grande importância em sistemas industriais suscetíveis a ataques externos e falhas. Neste artigo, mostramos sob quais condições um conjunto de plantas e especificações levam à sintetização de um supervisor sincronizável, usando a Teoria de Controle Supervisório.

Palavras-chave— Sistemas a Eventos Discretos, Autômatos Sincronizáveis, Teoria de Controle Supervisório

1 Introdução

No cotidiano, a palavra sincronização é utilizada nos mais variados sentidos. No contexto científico, sincronização guarda semelhanças com seu significado cotidiano, mas é um pouco mais geral. A sincronização começa a ser estudada pela física no século XVII, quando Huygens observa que dois relógios de pêndulo, fracamente acoplados, se tornam sincronizados (Pikovsky et al., 2001).

Em sistemas dinâmicos, o estudo da sincronização tem se desenvolvido muito na área de sistemas caóticos onde dois ou mais sistemas podem ser considerados sincronizados quando ajustam alguma propriedade a um comportamento comum devido ao acoplamento ou a uma força (Boccaletti et al., 2002). Dessa definição ampla, surgiram diversos tipos diferentes de sincronização como, por exemplo, sincronização idêntica, de fase, de atraso, e sincronização generalizada, entre diversas outras.

Apesar de, em geral, estar associada ao tempo, o conceito de sincronização também pode ser aplicado a sistemas orientados à ocorrência de eventos. No contexto dos Autômatos Finitos Determinísticos (AFD), dois autômatos idênticos estão sincronizados quando estão no mesmo estado. Nesse contexto, o maior desafio está em determinar uma Palavra de Sincronização para esse autômato, ou seja, uma palavra tal que, ao ser executada, sempre leva dois autômatos idênticos ao mesmo estado, independentemente dos estados em que se encontram. Essa palavra nem sempre existe, de forma que a sua existência nos permite classificar autômatos como sincronizáveis ou não sincronizáveis.

A sincronização de autômatos apresenta diversas aplicações práticas, como por exemplo na recuperação de falhas. Em sistemas modelados por autômatos, em geral, o estado do sistema é, muitas vezes, difícil de ser identificado, principalmente quando não é acompanhado desde o início da operação do sistema.

Grande parte dos sistemas atuais têm sua dinâmica regida pela ocorrência de eventos assíncronos no tempo, os Sistemas a Eventos Discretos (SED). Uma das abordagens existentes para tratar o problema de controle de um SED é a Teoria do Controle Supervisório (TCS) (Wonham and Ramadge, 1988), onde os sistemas são modelados por Autômatos Finitos Determinísticos e o controle permite todos os comportamentos seguros do sistema.

Na TCS, a planta e o supervisor (controlador) são tratados como entidades distintas e o supervisor tem como papel observar os eventos executados na planta e proibir a ocorrência de certos eventos, chamados eventos controláveis, de acordo com a dinâmica observada de modo a não violar especificações.

Com o crescimento da complexidade dos sistemas industriais e do número de equipamentos envolvidos em cada processo, aumenta o risco de perda de sincronia (Bergagård and Fabian, 2014). Esta perda pode ser causada por falhas nos equipamentos ou mesmo ataques de agentes maliciosos, cada vez mais comuns nos tempos atuais. Nesse sentindo a sincronização é uma ferramenta importante que permite levar o sistema a um estado conhecido e recuperar a sincronização entre

planta e supervisor.

A existência de uma palavra de sincronização em um autômato tem aplicação em diversas áreas, como na área da robótica, no contexto da automação industrial, relacionado ao carregamento, montagem e empacotamento de produtos (Natarajan, 1986; Natarajan, 1989), inicialmente de maneira independente da literatura de sincronização. A partir desses trabalhos originais surgiram desenvolvimentos teóricas na área de sincronização de autômatos no contexto de automação industrial (Eppstein, 1988; Goldberg, 1993; Chen and Ierardi, 1995).

Na área de Sistemas a Eventos Discretos, existe uma série de artigos tratando do problema de sincronização em Redes de Petri (Pocci et al., 2013; Pocci et al., 2014a; Pocci et al., 2014b; Pocci et al., 2016), no sentido de tratar problemas de recuperação de erros, utilizando palavras de sincronização para levar o sistema a uma condição conhecida.

Pretende-se, neste trabalho, estudar o papel que a teoria de sincronização pode realizar no sentido de ajudar na recuperação de um sistema cujas partes perderam sincronia.

2 Conceitos Preliminares

2.1 Sistemas a Eventos Discretos

Sistemas a eventos discretos são sistemas dinâmicos que mudam de estado por meio de estímulos, chamados eventos. Esses eventos podem ser o início ou fim de uma tarefa, o acionamento de um sensor, além de eventos internos, como o fim de uma temporização. Em geral, define-se estado como a configuração do sistema entre a ocorrência de dois eventos consecutivos, ou seja, a ocorrência de um evento acarreta em uma transição de estado no sistema (Cassandras and Lafortune, 2009).

Seja Σ um conjunto finito de símbolos, usualmente referido como alfabeto. Seja Σ^+ o conjunto de todas as sequências finitas de símbolos, na forma $\sigma_1\sigma_2...\sigma_k$ onde $k > 0$ e $\sigma_i \in \Sigma$. A sequência vazia, com nenhum símbolo, é representada por ϵ e:

$$\Sigma^* = \{\epsilon\} \cup \Sigma^+$$

Uma sequência $s \in \Sigma^*$ é, usualmente, chamada de palavra ou cadeia sobre o alfabeto Σ. O comprimento de uma sequência $|s|$ é definido como:

$$|\epsilon| = 0$$
$$|\sigma_1\sigma_2...\sigma_k| = k.$$

A concatenação de duas cadeias $s, u \in \Sigma^*$ forma a palavra $t = su$, onde $t \in \Sigma^*$. Neste caso, diz-se que s é um prefixo de t e u é um sufixo de t.

Um subconjunto qualquer $L \subseteq \Sigma^*$ é chamado de linguagem e, sobre elas, é possível definir diversas operações em conjuntos (união, complemento, interseção e diferença) e a concatenação.

Neste trabalho, a fim de simplificar a notação, uma linguagem composta por apenas uma sequência $\{u\}$ será representada apenas por u, o símbolo da união $\cup$ será substituído pelo símbolo $+$, a concatenação é representada pela justaposição de duas linguagens, a linguagem vazia é representada por $\emptyset$.

A concatenação de duas linguagens L_1 e L_2 é dada por:

$$L_1L_2 = \{u_1u_2 \,|\, u_1 \in L_1 \wedge u_2 \in L_2\}$$

A concatenação é uma operação associativa que admite como identidade a linguagem composta pela sequência vazia ϵ, ou seja, $L\epsilon = \epsilon L$ e, além disso, é uma operação distributiva sobre a união, logo, $L(L_1 + L_2) = LL_1 + LL_2$.

A potência de uma linguagem é definida como $L^0 = \epsilon$, $L^1 = L$ e, por indução, $L^n = L^{n-1}L$. A estrela de Kleene de uma linguagem L, denotada por L^*, é a união de todas as potências de L:

$$L^* = \sum_{n \geq 0} L^n$$

O prefixo fechamento de uma linguagem L, denotado por $\overline{L}$ é uma linguagem que contém todos os prefixos de sequências pertencentes a L, incluindo a sequência vazia ϵ:

$$\overline{L} = \{s \in \Sigma^* \,|\, su \in L\}$$

Outra operação comum, aplicada em sequências e linguagens, é a projeção natural, partindo de um alfabeto Σ_1 para um alfabeto menor Σ_2, tal que $\Sigma_2 \subseteq \Sigma_1$. A projeção para uma sequência é definida como:

$$P_{\Sigma_1 \to \Sigma_2}(\epsilon) = \epsilon$$
$$P_{\Sigma_1 \to \Sigma_2}(\sigma) = \begin{cases} \sigma & \text{se } \sigma \in \Sigma_2 \\ \epsilon & \text{se } \sigma \in \Sigma_1 \setminus \Sigma_2 \end{cases}$$
$$P_{\Sigma_1 \to \Sigma_2}(s\sigma) = P_{\Sigma_2 \to \Sigma_1}(s)P_{\Sigma_2 \to \Sigma_1}(\sigma)$$
para $s \in \Sigma_1^*, \sigma \in \Sigma_1$.

A projeção inversa de uma sequência mapeia uma sequência de um alfabeto menor Σ_2 em um alfabeto maior Σ_1, sendo definida como:

$$P^{-1}_{\Sigma_2 \to \Sigma_1}(t) = \{s \in \Sigma_1^* \,|\, P(s) = t\}.$$

É importante observar que a projeção inversa retorna todas as sequências do alfabeto Σ_1 que projetadas formam t, ou seja, $s \in P^{-1}(P(s))$. Ambas as definições, de projeção e projeção inversa podem ser estendidas para linguagens como mostrado a seguir:
Para $L \subseteq \Sigma_1^*$:

$$P_{\Sigma_1 \to \Sigma_2}(L) = \{t \in \Sigma_2^* \,|\, (\exists s \in L)\,[P_{\Sigma_1 \to \Sigma_2}(s) = t]\}.$$

Para $L \subseteq \Sigma_2^*$:

$$P^{-1}_{\Sigma_2 \to \Sigma_1}(L) = \{s \in \Sigma_1^* \,|\, (\exists t \in P(L))\,[P_{\Sigma_1 \to \Sigma_2}(s) = t]\}.$$

Dado um alfabeto Σ, existe um conjunto de linguagens, chamadas linguagens regulares, que podem ser representadas como autômatos ou expressões regulares. Uma expressão regular é uma expressão formal definida recursivamente como:

1. As linguagens $\emptyset$ e ϵ são expressões regulares.
2. Qualquer símbolo $\sigma \in \Sigma$ é uma expressão regular.
3. Se e_1 e e_2 são expressões regulares, então $(e_1 + e_2)$, $(e_1 e_2)$ e e^* também são expressões regulares.

2.2 Autômatos Finitos Determinísticos

Um autômato finito determinístico é definido por uma 5-tupla $G = (Q, \Sigma, \delta, q_0, Q_m)$, onde Q é um conjunto finito de estados, Σ é o conjunto finito de símbolos (alfabeto), $\delta : Q \times \Sigma \to Q$ é a função de transição de estados, $q_0 \in Q$ é o estado inicial e $Q_m \subseteq Q$ é o conjunto de estados marcados de Q (Hopcroft et al., 2006). A função de transição pode ser estendida para reconhecer cadeias sobre Σ^* como $\delta(q, \sigma s) = q'$ se $\delta(q, \sigma) = x$, $\delta(x, s) = q'$ e $\delta(q, \epsilon) = q$.

Um caminho, em um autômato, é denotado por:

$$c : q_0 \xrightarrow{\sigma_1} q_1 ... q_{n-1} \xrightarrow{\sigma_n} q_n$$

ou, de uma forma compacta, como $c : q_0 \xrightarrow{\sigma_1 ... \sigma_n} q_n$. O conjunto de todas as sequências de símbolos do alfabeto Σ que, partindo do estado inicial, formam um caminho num autômato G compõem a linguagem gerada pelo autômato, representada por $\mathcal{L}(G)$. Além disso, a linguagem marcada de um autômato é definida como um subconjunto da linguagem gerada, $\mathcal{L}_m(G) \subseteq \mathcal{L}(G)$, composto por todas as sequências que terminam em um estado marcado. É possível, também, analisar o conjunto de sequências factíveis partindo de um estado qualquer, $q \in Q$, do autômato denotada por $\mathcal{L}_G(q) = \{s \in \Sigma^* | \delta(q, s) = q' \wedge q, q' \in Q\}$ tal que $\mathcal{L}_G(q_0) = \mathcal{L}(G)$.

A composição paralela entre dois autômatos $G_1 = (Q_1, \Sigma_1, \delta_1, q_{01}, Q_{m1})$ e $G_2 = (Q_2, \Sigma_2, \delta_2, q_{02}, Q_{m2})$ permite a representação do comportamento conjunto dos autômatos originais e é dada por $G_{1||2} = (Q_1 \times Q_2, \Sigma_1 \cup \Sigma_2, \delta_{1||2}, (q_{01} \times q_{02}), Q_{m1} \times Q_{m2})$ onde:

$$\delta_{1||2}((q_1, q_2), \sigma) = \begin{cases} (\delta_1(q_1, \sigma), \delta_2(q_2, \sigma)) \text{ se } \delta_1(q_1, \sigma)! \text{ e } \delta_2(q_2, \sigma)! \\ (\delta_1(q_1, \sigma), q_2) \text{ se } \delta_1(q_1, \sigma)! \text{ e } \sigma \notin \Sigma_2 \\ (q_1, \delta_2(q_2, \sigma)) \text{ se } \delta_2(q_2, \sigma)! \text{ e } \sigma \notin \Sigma_1 \\ \text{indefinido caso contrário} \end{cases}$$

Do ponto de vista de linguagens, a composição paralela entre dois autômatos pode ser representada como $\mathcal{L}(G_{1||2}) = P^{-1}_{\Sigma_2 \to \Sigma_1 \cup \Sigma_2}(\mathcal{L}(G_1)) \cap P^{-1}_{\Sigma_1 \to \Sigma_1 \cup \Sigma_2}(\mathcal{L}(G_2))$.

2.3 Teoria de Controle Supervisório

A Teoria de Controle Supervisório (TCS) (Wonham, 2014) é uma abordagem que permite, a partir de plantas e especificações de segurança modeladas por autômatos finitos determinísticos, encontrar um agente de controle, capaz de desabilitar certos eventos a fim de garantir um comportamento seguro e não bloqueante, em malha fechada.

Na TCS, $\Sigma = \Sigma_c \cup \Sigma_{nc}$, o conjunto de eventos de um sistema, é particionado em Σ_c, o conjunto de eventos controláveis, e Σ_{nc}, o conjunto de eventos não controláveis, que não podem ser desabilitados pelo agente de controle.

Sejam G uma planta e E uma especificação, a condição necessária e suficiente para a existência de um supervisor não bloqueante S para G sendo o sistema sobre controle do supervisor denotado por S/G, tal que $\mathcal{L}_m(S/G) = \mathcal{L}(G)||E = K$, é que K seja controlável em relação a $\mathcal{L}(G)$ e Σ_{nc}, ou seja, $\overline{K}\Sigma_{nc} \cap \mathcal{L}(G) \subseteq \overline{K}$. Caso o supervisor não seja controlável, então deve-se calcular a máxima sub linguagem controlável da linguagem desejada, denotada por $Sup\mathcal{C}(K, G)$.

2.4 Sincronização em Autômatos

Dado um autômato finito determinístico (AFD) na forma $G = (Q, \Sigma, \delta, q_0, Q_m)$, dizemos que G é sincronizável quando existe, ao menos, uma cadeia $s \in \Sigma^*$, tal que, quando executada pelo autômato leva o autômato a um determinado estado $q_s \in Q$ independentemente do estado no qual o autômato se encontrava, ou seja, $\delta(q, s) = \delta(q', s)$ para todo $q, q' \in Q$. No contexto de autômatos sincronizáveis, é comum a utilização da notação $A.s$, para denotar o conjunto de estados alcançados com a execução da palavra s partindo de cada estado em A.

Definição 1 *Seja $G = (Q, \Sigma, \delta, q_0, Q_m)$ um autômato finito determinístico, dizemos que G é sincronizável se existe ao menos uma cadeia w tal que $Q.w = \{q'\}$ e $q' \in Q$. Tal cadeia w é chamada palavra de sincronização de G e o conjunto de todas as palavras de sincronização de G é denotado por $Syn(G)$.*

3 Principais Resultados

No contexto de sistemas a eventos discretos (SED), principalmente com relação à Teoria de Controle Supervisório, é convencional modelar os sistemas utilizando autômatos incompletos, de forma que a ideia de sincronização fica melhor definida quando a palavra de sincronização leva sempre ao estado inicial do autômato. Neste caso, denotamos o conjunto de palavras de sincronização ao estado inicial de uma autômato qualquer $Syn_{q_0}(G)$ onde $Syn_{q_0}(G) \subseteq Syn(G)$.

Definição 2 *Seja $G = (Q, \Sigma, \delta, q_0, Q_m)$ um autômato finito determinístico, dizemos que G é um autômato sincronizável com relação ao estado inicial se existe uma cadeia w, chamada palavra de sincronização, tal que $Q.w = \{q_0\}$.*

Nesse caso para todo $q \in Q$ temos que $\delta(q, w) = q_0$. Além disso, $I = Syn_{q_0}(G)$ o conjunto das infinitas palavras de sincronização ao estado inicial do autômato G.

No caso da Definição 2, é fácil observar que $\mathcal{L}(G)I\mathcal{L}(G) \subset \mathcal{L}(G)$ e, da mesma forma, que $\mathcal{L}(G)I\mathcal{L}_m(G) \subset \mathcal{L}_m(G)$ e $\mathcal{L}_m(G)I\mathcal{L}_m(G) \subset \mathcal{L}_m(G)$.

É importante perceber que uma linguagem regular L qualquer, que apresente a propriedade $LIL \subset L$, quando $|I| > 0$ e $\epsilon \notin I$, pode ser modelada por um autômato finito determinístico sincronizável com relação ao estado inicial e toda a cadeia $w \in I$ é uma palavra de sincronização desse autômato.

Lema 1 *Seja I um conjunto de palavras tal que $LIL \subset L$ e seja G o menor autômato tal que $\mathcal{L}(G) = L$. Então G é um autômato sincronizável com relação ao estado inicial e $I = Syn_{q_0}(G)$.*

Prova: *Qualquer cadeia $s \in LI$ leva a um estado tal que a linguagem a partir desse estado é igual à linguagem do estado inicial, logo, se G é mínimo, $\delta(q_0, s) = q_0$.*

Tomando qualquer cadeia $w \in I$, por definição, temos $\delta(q, w) = q_0$ para qualquer $q \in Q$. Para o estado q, considerando que todos os estados de G são alcançáveis, sabemos que existe uma cadeia $u \in L$ tal que $\delta(q_0, u) = q$. Note que $uw \in L$ leva a um estado cuja linguagem de saída é igual à do estado inicial, então, se G é mínimo, $\delta(q_0, uw) = q_0$. Sabendo que $\delta(q_0, u) = q$ e que $\delta(q, w) = q_0$, temos que $w \in Syn_{q_0}(G)$, logo $I \subseteq Syn_{q_0}(G)$ e G é sincronizável. □

De modo geral, faz pouco sentido, no modelo RW, esperar que um supervisor seja sincronizável quando os autômatos que deram origem a esse supervisor não são. Nesse caso, é importante analisar como a palavra de sincronização sobrevive às operações relacionadas à sintetização de supervisores, ou seja, verificar em quais situações as operações aplicadas nos autômatos das plantas e especificações sincronizáveis resultam em autômatos também sincronizáveis.

A primeira operação analisada é a composição paralela que, por sua vez, pode ser dividida em dois passos, sendo eles a projeção inversa e a interseção.

Quando pensado no contexto de autômatos, fica claro que a projeção inversa, que cria autolaços em todos os estados do autômato para os eventos de $\Sigma_2 \setminus \Sigma_1$, não altera a sincronicidade do autômato, e que existe interseção entre o conjunto de palavras de sincronização de ambos os autômatos.

Lema 2 *Seja $L \subseteq \Sigma_1^*$, e G um autômato tal que $L = \mathcal{L}(G)$ e $I_L = Syn_{q_0}(G) \neq \emptyset$. Considere, também, a projeção inversa $P^{-1}_{\Sigma_2} : \Sigma_1^* \to (\Sigma_1 \cup \Sigma_2)^*$. Nesse caso, a linguagem $P^{-1}(L)$ pode ser modelada por um autômato finito determinístico sincronizável com relação ao estado inicial.*

Prova: *Considerando:*

$$LI_LL \subset L$$

dado que a projeção inversa de uma linguagem é a projeção inversa de cada cadeia da linguagem, temos $P^{-1}(LI_LL) \subset P^{-1}(L)$.

Podemos, então, decompor a projeção inversa do lado esquerdo, obtendo $P^{-1}(L)P^{-1}(I_L)P^{-1}(L) \subset P^{-1}(L)$.

Substituindo $K = P^{-1}(L)$ e $I_K = P^{-1}(I_L)$ temos $KI_KK \subset K$, onde, pelo o Lema 1, K pode ser modelada por um autômato sincronizável com relação ao estado inicial. □

Nesse caso, observamos que a linguagem K pode ser modelada por um autômato finito determinístico sincronizável com relação ao estado inicial e que $I_L \subset I_K$, o que indica que toda palavra de sincronização do autômato que modela a linguagem L também é palavra de sincronização do autômato que modela a linguagem K.

Lema 3 *Sejam G_1 e G_2 autômatos finitos determinísticos tais que $L_1 = \mathcal{L}(G_1)$, $I_1 = Syn_{q_0}(G_1) \neq \emptyset$, $L_2 = \mathcal{L}(G_2)$ e $I_2 = Syn_{q_0}(G_2) \neq \emptyset$. Caso $I_k = I_1 \cap I_2 \neq \emptyset$, sabemos que $L_k = L_1 \cap L_2$ também pode ser modelada por um autômato finito determinístico sincronizável com relação ao estado inicial.*

Prova: *Considerando $L_1I_1L_1 \subset L_1$ e $L_2I_2L_2 \subset L_2$. Dado que as cadeias comuns às duas linguagens são mantidas na interseção temos*

$$(L_1I_1L_1) \cap (L_2I_2L_2) \subset L_1 \cap L_2$$

sendo $\sigma \in (L_1 \cap L_2)$ temos que $\sigma \in L_1$ e sendo $w \in (I_1 \cap I_2)$ temos que $w \in I_1$. Dessa forma, sabemos que $\sigma w \sigma \in (L_1I_1L_1)$ e, com isso, podemos escrever:

$$\begin{aligned}(L_1 \cap L_2)(I_1 \cap I_2)(L_1 \cap L_2) &\\ \subset (L_1I_1L_1) \cap (L_2I_2L_2) &\\ \subset L_1 \cap L_2 &\end{aligned}$$

Substituindo $L_k = L_1 \cap L_2$ e $I_K = I_1 \cap I_2$ temos $L_kI_kL_k \subset L_K$, onde L_k pode ser modelada por um autômato sincronizável com relação ao estado inicial considerando o Lema 1. □

Dessa forma, é possível definir em quais situações a composição paralela mantém a sincronicidade dos autômatos originais, caso ambos sejam autômatos sincronizáveis com relação ao estado inicial:

Teorema 1 *Sejam os autômatos* $G_1 = (Q_1, \Sigma_1, \delta_1, q_{01}, Q_{m1})$ *e* $G_2 = (Q_2, \Sigma_2, \delta_2, q_{02}, Q_{m2})$ *sincronizáveis com relação ao estado inicial e seja* $\Sigma = \Sigma_1 \cup \Sigma_2$. *O autômato resultante da composição paralela* $G = G_1||G_2$ *é sincronizável com relação ao estado inicial se* $P^{-1}_{\Sigma_1\to\Sigma}(I_1) \cap P^{-1}_{\Sigma_2\to\Sigma}(I_2) \neq \emptyset$.

Prova: *Sabemos, pelo Lema 2, que as linguagens* $P^{-1}_{\Sigma_1\to\Sigma}(\mathcal{L}(G_1))$, $P^{-1}_{\Sigma_1\to\Sigma}(\mathcal{L}_m(G_1))$, $P^{-1}_{\Sigma_2\to\Sigma}(\mathcal{L}(G_2))$ *e* $P^{-1}_{\Sigma_2\to\Sigma}(\mathcal{L}_m(G_2))$ *podem ser modeladas por autômatos finitos determinísticos sincronizáveis com relação ao estado inicial.*

Considerando que:

$$\mathcal{L}(G_1||G_2) = P^{-1}_{\Sigma_1\to\Sigma}(\mathcal{L}(G_1)) \cap P^{-1}_{\Sigma_2\to\Sigma}(\mathcal{L}(G_2))$$

temos, pelo Lema 3, que a linguagem $\mathcal{L}(G_1||G_2)$ *também pode ser modelada por um autômato finito determinístico sincronizável com relação ao estado inicial, desde que* $P^{-1}_{\Sigma_1\to\Sigma}(I_1) \cap P^{-1}_{\Sigma_2\to\Sigma}(I_2)$. □

Levando em consideração a Teoria de Controle Supervisório, sabemos que o conjunto de eventos Σ é particionado como $\Sigma_c \cup \Sigma_{nc}$ onde Σ_c é o conjunto de eventos controláveis e Σ_{nc} é o conjunto de eventos não controláveis. Sabemos que o supervisor não pode desabilitar eventos não controláveis, logo, caso $\Sigma^*_{nc} \cap I \neq \emptyset$ sabemos que existem palavras de sincronização compostas apenas por eventos não controláveis e, dessa forma, no processo de obtenção da máxima sublinguagem controlável, ao menos essas palavras continuarão sendo palavras de sincronização do resultado ou a máxima sublinguagem controlável será a linguagem vazia.

Teorema 2 *Sejam* G *a planta e* E *a especificação, ambas modeladas por autômatos sincronizáveis com respeito ao estado inicial e* $S = Sup\,C(G||E, G)$. *O supervisor* S, *também é sincronizável com respeito ao estado inicial se* $\Sigma^*_{nc} \cap I \neq \emptyset$, *onde* $I = Syn_{q_0}(G \parallel E)$.

Prova: *Seja* $K = G \parallel E = (Q, \Sigma, \delta, q_0, Q_m)$ *e* $S = (Q_s, \Sigma, \delta_s, q_0, Q_{ms})$,*onde* $Q_s \subseteq Q$ *e* $Q_{ms} \subseteq Q_m$. *Com relação à controlabilidade, cada estado* $q_f \in Q \setminus Q_s$ *é um mau estado, dado que falha no princípio da controlabilidade e, portanto, foi eliminado de* Q_s.

Considerando $\Sigma^*_{nc} \cap I \neq \emptyset$, *existe ao menos uma palavra* $w = \sigma_1\sigma_2\ldots\sigma_n \in \Sigma^*_{nc} \cap I$. *Dessa forma, existem duas possibilidades a serem observadas.*

a) a sequência w *executada a partir de qualquer estado* $q \in Q$ *não visita maus estados. Se este é o caso, como todos os estados em* K *que são visitados são estado bons, eles permanecerão em* S. *Logo,* $w \in I$ *é, também, uma palavra de sincronização de* S.

b) a sequência w *executada a partir de qualquer estado* $q \in Q$ *visita um mau estado. Ao obter* S, *estados do autômato que implementa* K *são removidos caso sejam maus estados. Se existe uma palavra* $w = \sigma_1\sigma_2\ldots\sigma_n \in \Sigma^*_{nc} \cap I$, *onde* $I = Syn_{q_0}(K)$, *então cada estado* $q' \in Q$ *onde* $q' \xrightarrow{\sigma_1\ldots\sigma_p} q_f \xrightarrow{\sigma_{p+1}\ldots\sigma_n} q_0$ *também é um mau estado e não estará em* Q_s. *Dessa forma, todo o estado* q *que leva a um mau estado por meio de eventos não controláveis, também é removido, logo* w *é completamente removido, indicando que* $\mathcal{L}(S) = \emptyset$ *ou apenas um subconjunto de* Q *é removido, tal que* $Q_s.w = \{q_0\}$, *então,* w *é uma palavra de sincronização de* S.

Após a remoção dos maus estados, a parte acessível do autômato resultante é sempre coacessível e, dessa forma, não bloqueante. □

Na próxima seção, apresenta-se um exemplo de aplicação da teoria.

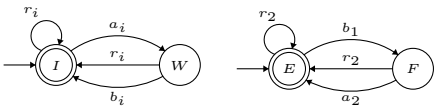
4 Exemplo de Aplicação

A existência de uma palavra de sincronização é incomum na área de sistemas a eventos discretos, dessa forma, aqui utilizaremos eventos não controláveis de *reset* similares aos que são apresentados em (Bergagård and Fabian, 2014). A ideia consiste na criação de transições de *reset* visando a recuperação de sincronização entre a planta e o supervisor após a ocorrência de uma falha.

O sistema conhecido como Pequena Fábrica (Wonham and Ramadge, 1988) é composto por duas máquinas (M_1 e M_2) e um *buffer* unitário entre elas. A especificação de funcionamento é evitar o *overflow* e o *underflow* do *buffer*. Nesse caso, tanto as plantas quanto a especificação foram modeladas como autômatos sincronizáveis, utilizando eventos de *reset* (r_1 e r_2). No caso da especificação, não há necessidade de esvaziar o *buffer* quando M_1 executa um *reset*, logo somente está definido o evento r_2.

Na Figura 1 o modelo de cada máquina do sistema e da especificação é apresentado. A menor palavra de sincronização é dada pela sequencia r_i para cada máquina e r_2 para a especificação. Elas não são únicas no entanto, $r_1\ r_1$ também é palavra de sincronização para M_1, assim como $r_1\ a_1\ r_1$.

A composição paralela dos autômatos que modelam as máquinas também é um autômato sincronizável, como mostrado na Figura 2, e apresenta duas palavras de sincronização mínimas, sendo elas $w_1 = r_1r_2$ e $w_2 = r_2r_1$. É fácil observar que qualquer uma das palavras de sincronização leva cada um dos autômatos originais, quando



(a) Autômato da Máquina M_i (b) Autômato da especificação

Figura 1: Plantas e especificação modeladas como autômatos sincronizáveis

projetados no mesmo alfabeto, para seus respectivos estados iniciais.

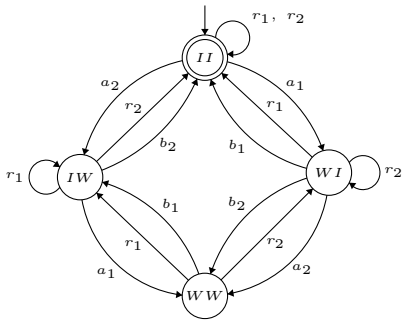


Figura 2: Composição paralela das Máquinas 1 e 2, com menores palavras de sincronização $w_1 = r_1\ r_2$ e $w_2 = r_2\ r_1$.

Como os eventos r_1 e r_2 são não controláveis, dado que são relacionados à segurança do sistema e não devem ser desabilitados, o supervisor resultante será sincronizável com relação ao estado inicial e a menores palavras de sincronização iguais às de $M_1 \parallel M_2$.

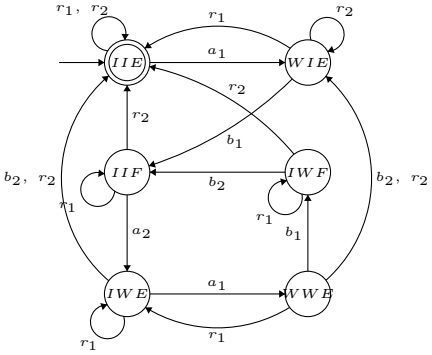


Figura 3: Supervisor resultante com menores palavras de sincronização dadas por $w_1 = r_1\ r_2$ e $w_2 = r_2\ r_1$, as mesmas da planta.

O compartilhamento de palavras de sincronização entre supervisor e planta pode ser usado na recuperação de sistemas que sofreram ataques de agentes externos ou falhas que atrapalhem o sincronismo entre o supervisor (Figura 2) e a planta (Figura 3), ou seja, o supervisor aplica uma ação de controle, desabilitando eventos, considerando que a planta está em determinado estado, porém ela não está no estado previsto.

Suponha que, ao tentar executar a sequência $s = a_1b_1a_2a_1b_2b_1a_2b_2$, que produz dois produtos, um agente externo malicioso impeça que o supervisor observe a segunda ocorrência do evento b_1, ou seja, o supervisor executa, de fato, a sequência $s_p = a_1b_1a_2a_1b_2$ e encontra um bloqueio.

Após a execução dos seis primeiros eventos, a planta se encontra no estado II, porém o supervisor não observa o sexto evento, de forma que o supervisor para no estado WIE, que tem todos os eventos controláveis desabilitados pelo supervisor, levando o sistema a uma situação de bloqueio.

Neutralizando o agente malicioso, podemos executar a palavra de sincronização comum entre supervisor e planta, por exemplo $w = r_1r_2$, levando ambos ao estado inicial, ou seja, uma situação onde estão novamente em sincronia.

5 Conclusão

Este artigo apresentou uma introdução ao uso de autômatos sincronizáveis no contexto de sistemas a eventos discretos utilizando a teoria de controle supervisório.

Autômatos sincronizáveis são ferramentas muito úteis na modelagem de sistemas seguros e capazes de se recuperar de falhas, permitindo que o sistema sempre possa retornar a um estado conhecido, mesmo após um ataque externo ou quando um evento se torna temporariamente não observável. Como trabalhos futuros, pretende-se definir uma metodologia de modelagem de plantas e especificações como autômatos sincronizáveis com relação ao estado inicial. Pretende-se também analisar problemas de opacidade em autômatos sincronizáveis e como ambos os conceitos se relacionam.

Agradecimentos

O presente trabalho foi realizado com o apoio financeiro da Fapemig, CAPES - Brasil e CNPq.

Referências

Bergagård, P. and Fabian, M. (2014). Calculating restart states using reset transitions, *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3345–3350.

Boccaletti, S., Kurths, J., Osipov, G., Valladares, D. and Zhou, C. (2002). The synchronization of chaotic systems, *Physics reports* **366**(1): 1–101.

Cassandras, C. G. and Lafortune, S. (2009). *Introduction to discrete event systems*, Springer Science & Business Media.

Chen, Y.-B. and Ierardi, D. (1995). The complexity of oblivious plans for orienting and distinguishing polygonal parts, *Algorithmica* **14**(5): 367–397.

Eppstein, D. (1988). Reset sequences for finite automata with application to design of parts orienters, *Proceedings of the 15th International Col-*

loquium on Automata, Languages and Programming, ICALP '88, Springer-Verlag, London, UK, UK, pp. 230–238.

Goldberg, K. Y. (1993). Orienting polygonal parts without sensors, *Algorithmica* **10**(2): 201–225.

Hopcroft, J. E., Motwani, R. and Ullman, J. D. (2006). Automata theory, languages, and computation, *International Edition* **24**.

Natarajan, B. (1989). Some paradigms for the automated design of parts feeders, *The International Journal of Robotics Research* **8**(6): 98–109.

Natarajan, B. K. (1986). An algorithmic approach to the automated design of parts orienters, *Proceedings of the 27th Annual Symposium on Foundations of Computer Science*, SFCS '86, IEEE Computer Society, Washington, DC, USA, pp. 132–142.

Pikovsky, A., Rosenblum, M. and Kurths, J. (2001). *Synchronization : a universal concept in nonlinear sciences*, The Cambridge nonlinear science series, Cambridge University Press, Cambridge.

Pocci, M., Demongodin, I., Giambiasi, N. and Giua, A. (2013). A new algorithm to compute synchronizing sequences for synchronized petri nets, *TENCON 2013-2013 IEEE Region 10 Conference (31194)*, IEEE, pp. 1–6.

Pocci, M., Demongodin, I., Giambiasi, N. and Giua, A. (2014a). Testing experiments on synchronized petri nets, *IEEE Transactions on Automation Science and Engineering* **11**(1): 125–138.

Pocci, M., Demongodin, I., Giambiasi, N. and Giua, A. (2014b). Testing experiments on unbounded systems: synchronizing sequences using petri nets, *IFAC Proceedings Volumes* **47**(2): 155–161.

Pocci, M., Demongodin, I., Giambiasi, N. and Giua, A. (2016). Synchronizing sequences on a class of unbounded systems using synchronized petri nets, *Discrete Event Dynamic Systems* **26**(1): 85–108.

Wonham, W. M. (2014). *Supervisory Control of Discrete-Event Systems*, Systems Control Group, Department of Electrical & Computer Engineering, University of Toronto, Toronto, Canada.

Wonham, W. M. and Ramadge, P. J. (1988). Modular supervisory control of discrete-event systems, *Mathematics of control, signals and systems* **1**(1): 13–30.