

# IMPLEMENTAÇÃO DE UMA REDE DE PETRI PARA DIAGNOSE ONLINE DE FALHAS EM CONTROLADORES LÓGICOS PROGRAMÁVEIS

MARCOS V. MOREIRA\* FELIPE G. CABRAL\* OUMAR DIENE\* JOÃO C. BASILIO\*

\*COPPE-Programa de Engenharia Elétrica, Universidade Federal do Rio de Janeiro, Cidade Universitária, Ilha do Fundão, Rio de Janeiro, 21.945-970, RJ, Brasil

Email: moreira.mv@poli.ufrj.br, felipecabral@poli.ufrj.br, oumar@poli.ufrj.br, basilio@dee.ufrj.br

**Abstract**— In this paper we present methods for obtaining Sequential Function Charts (SFCs) and ladder diagrams for the implementation of a Petri net diagnoser (PND) on a programmable logic controller (PLC). Since the PND is a binary Petri net, the conversion into SFC is almost straightforward. The method for the conversion of PND into ladder diagrams allows the implementation of the diagnoser on CLPs that do not support SFC language, and provides a well-structured ladder diagram that allows the visualization of the structure of the Petri net as well as the evolution of the tokens in the diagnoser.

**Keywords**— Ladder diagram, SFC, Petri net, Automaton, Fault diagnosis, Discrete event systems.

**Resumo**— Neste artigo são apresentados métodos para obtenção de diagramas funcionais sequenciais (SFC - Sequential Function Chart) e diagramas ladder para implementação de uma rede de Petri para diagnose online de falhas (RPD) em um controlador lógico programável (CLP). Uma vez que a RPD é uma rede de Petri binária, a conversão para SFC é quase direta. O método para conversão de RPD em diagramas ladder permite a implementação do diagnosticador em CLPs que não suportam programação em SFC e fornece códigos de programação bem estruturados, que permitem fácil visualização da estrutura da rede de Petri para diagnose de falhas, bem como o acompanhamento da evolução das fichas no diagnosticador.

**Palavras-chave**— Diagrama ladder, SFC, Rede de Petri, Autômato, Diagnose de falhas, Sistemas a eventos discretos.

## 1 INTRODUÇÃO

Diversas técnicas têm sido apresentadas na literatura para a obtenção de diagnosticadores de falhas em Sistemas a Eventos Discretos (SEDs) modelados por autômatos finitos (Sampath et al., 1995; Sampath et al., 1996; Qiu e Kumar, 2006). Em Moreira et al. (2012a) e Moreira et al. (2012b) é proposto um método para diagnose online de SEDs modelados por autômatos finitos computacionalmente mais eficiente do que os demais métodos existentes na literatura. O método consiste na obtenção de uma rede de Petri para diagnose que faz uma estimativa dos estados do comportamento normal do sistema com relação a uma partição do conjunto de eventos de falha, e verifica se é possível que uma dada sequência observada esteja associada a uma sequência normal com relação a algum tipo de falha do sistema. Caso isso não seja verificado, então ocorreu um evento de falha e o diagnosticador detecta essa ocorrência.

Apesar de existirem diversos métodos para obtenção de diagnosticadores de falhas, apenas alguns trabalhos tratam da implementação de um diagnosticador online em um CLP. O CLP é a ferramenta mais importante para o controle discreto de sistemas de manufatura automatizados e pode ser programado usando cinco linguagens diferentes definidas na norma internacional IEC 61131-3 (ISO/IEC, 2001): (i) diagrama ladder; (ii) diagrama de blocos de função; (iii) texto estruturado; (iv) lista de instruções; e (v) diagrama funcional sequencial (SFC - sequential function chart). En-

tre essas cinco linguagens, o diagrama ladder é o mais utilizado pela indústria e está disponível em quase todos os CLPs.

Um CLP pode ser usado exclusivamente para diagnose ou, dependendo das especificações do sistema em malha fechada, o diagnosticador online pode ser implementado no mesmo CLP utilizado no controle realimentado. A principal vantagem desse último esquema de implementação é a redução do hardware necessário para diagnose. Note que, nesse caso, todos os eventos de comando se tornam observáveis para o diagnosticador, sem a necessidade de sensores adicionais ou barramentos de comunicação.

Em Luca et al. (2011), uma plataforma particular de CLPs, o softPLC Orchestra, é utilizada para diagnose online de falhas em um SED modelado por um autômato finito. Nesse caso, o diagnosticador é uma tarefa do CLP, escrita em linguagem C, que amostra as variáveis globais do CLP e acompanha a evolução do sistema através da evolução dos estados do autômato diagnosticador. Em Luca et al. (2011) o diagnosticador é obtido seguindo o método de Sampath et al. (1995). Embora esse esquema de implementação seja aplicado com sucesso em Luca et al. (2011) para um exemplo, a extensão desse método para outras plataformas de CLP que não suportam a linguagem C não é uma tarefa simples. Não é do conhecimento dos autores a existência de algum trabalho que trate da implementação de um diagnosticador em um CLP utilizando uma das cinco linguagens

definidas na norma IEC 61131-3.

Apesar da escassez de trabalhos sobre a implementação de diagnosticadores online em CLPs, vários métodos para conversão de códigos complexos de controle em diagramas ladder têm sido apresentados (Uzam e Jones, 1998). Em Fabian e Hellgren (1998) e Hellgren et al. (2005), dois problemas importantes relacionados à implementação de controladores modelados por autômatos e SFCs em diagramas ladder são introduzidos: (i) o efeito avalanche; (ii) a necessidade de escolha entre transições envolvidas em um conflito. Em Moreira et al. (2009) é apresentado um método para conversão de uma rede de Petri interpretada para controle em um diagrama ladder para implementação em um CLP. O diagrama ladder é organizado de forma que o efeito avalanche é evitado e a resolução de conflitos é alcançada associando-se prioridades entre as transições envolvidas em um conflito efetivo.

Neste artigo, um método para a conversão de redes de Petri para diagnose (RPD) em SFC é apresentado. Uma vez que a RPD é uma rede binária (Alayan e Newcomb, 1987), a conversão em SFC é quase direta. Além disso, é apresentado um método para a conversão de RPD em diagramas ladder para implementação em CLPs que não suportam programação em SFC. O método de conversão, baseado em Moreira et al. (2009), evita o efeito avalanche e fornece um código de programação em diagrama ladder bem estruturado que permite fácil visualização da estrutura da rede de Petri para diagnose, bem como o acompanhamento da evolução das fichas no diagnosticador.

Este artigo está estruturado da seguinte forma: na seção 2 é apresentada a definição de redes de Petri para diagnose de falhas; na seção 3 alguns aspectos práticos da implementação de diagnosticadores em CLPs são abordados; na seção 4 são apresentados os métodos de conversão de RPD em SFC e diagramas ladder. Por fim, na seção 5, são apresentadas as conclusões.

## 2 Rede de Petri para diagnose de falhas

Seja  $G = (Q, \Sigma, \Gamma, f, q_0)$  o autômato determinístico que modela um sistema a eventos discreto parcialmente observado, ou seja, o conjunto de eventos pode ser particionado como  $\Sigma = \Sigma_o \dot{\cup} \Sigma_{uo}$ , em que  $\Sigma_o$  e  $\Sigma_{uo}$  denotam os conjuntos de eventos observáveis e não-observáveis, respectivamente. Seja  $\Sigma_f \subseteq \Sigma_{uo}$  o conjunto de eventos de falha e suponha que  $\Sigma_f$  possa ser particionado como  $\Sigma_f = \bigcup_{k=1}^r \Sigma_{f_k}$ , em que  $r$  denota o número de tipos de falha,  $\Sigma_{f_k}$  representa um conjunto de eventos de falha do mesmo tipo e  $\Pi_f$  denota essa partição. Considere que  $G_{N_k}$  modela o subautômato de  $G$  que representa o comportamento normal do sistema com relação ao conjunto de eventos de falha  $\Sigma_{f_k}$ .

Suponha agora que se deseja, a partir do co-

nhecimento de uma seqüência de eventos observáveis  $\nu \in \Sigma_o^*$ , diagnosticar a ocorrência de um evento de falha, ou seja, detectar a ocorrência do evento e identificar a qual conjunto  $\Sigma_{f_k}$ , para  $k = 1, \dots, r$ , o evento de falha pertence. Para tanto, supondo que a linguagem do sistema é diagnosticável (Moreira et al., 2011), é necessário construir um diagnosticador de falhas. Em Moreira et al. (2012a) é apresentado um algoritmo de obtenção de uma rede de Petri para diagnose de falhas  $\mathcal{N}_D$  para um sistema modelado por um autômato finito. A principal característica desse diagnosticador é a capacidade de fornecer uma estimativa dos estados de cada um dos autômatos  $G_{N_k}$ , que modela o comportamento normal do sistema com relação a um evento de falha do conjunto  $\Sigma_{f_k}$ . A partir dessa estimativa, a ocorrência da falha é verificada observando se é possível alcançar algum estado de  $G_{N_k}$ , após a ocorrência da seqüência de eventos observáveis  $\nu$ . Caso não seja possível, então uma falha do conjunto  $\Sigma_{f_k}$  ocorreu. A rede de Petri para diagnose  $\mathcal{N}_D$  pode ser definida da seguinte forma.

**Definição 1** *Uma rede de Petri para diagnose de falhas é uma rede de Petri binária  $\mathcal{N}_D = (P_D, T_D, Pre_D, Post_D, In_D, x_{0,D}, \Sigma_D, l_D)$ , em que  $P_D$  é o conjunto finito de lugares,  $T_D$  é o conjunto finito de transições,  $Pre_D : (P_D \times T_D) \rightarrow \{0, 1\}$  e  $Post_D : (T_D \times P_D) \rightarrow \{0, 1\}$  são as funções dos arcos que ligam lugares a transições e transições a lugares, respectivamente,  $In_D : (P_D \times T_D) \rightarrow \{0, 1\}$  é a função dos arcos inibidores que ligam lugares a transições,  $x_{0,D}$  é o estado inicial da rede de Petri,  $\Sigma_D = \Sigma_o \cup \{\lambda\}$ , em que  $\lambda$  denota o evento sempre ocorrente, é o conjunto de eventos utilizados para rotulação das transições e  $l_D : T_D \rightarrow 2^{\Sigma_D}$  é a função de rotulação que atribui a cada transição de  $T_D$  um subconjunto de eventos de  $\Sigma_D$ .*

O conjunto de transições  $T_D$  pode ser particionado como  $T_D = T_{SO} \dot{\cup} T_f$ , em que  $T_{SO}$  denota o conjunto de transições associadas a uma rede de Petri observadora a partir da qual  $\mathcal{N}_D$  é obtida (Moreira et al., 2012a), e  $T_f$  denota o conjunto de transições associadas aos eventos de falha do sistema. Portanto,  $|T_f| = r$ , em que  $|\cdot|$  denota a cardinalidade de um conjunto. O disparo de uma transição  $t_{f_k} \in T_f$  indica que um evento de falha do conjunto  $\Sigma_{f_k}$  foi diagnosticado. A seguir, um exemplo da obtenção de uma rede de Petri para diagnose e da sua utilização no diagnóstico de uma falha é apresentado.

**Exemplo 1** *Considere o sistema modelado pelo autômato  $G$  apresentado na figura 1, em que  $\Sigma = \{a, b, c, \sigma_u, \sigma_{f_1}, \sigma_{f_2}\}$ ,  $\Sigma_o = \{a, b, c\}$ ,  $\Sigma_{uo} = \{\sigma_u, \sigma_{f_1}, \sigma_{f_2}\}$ , e  $\Sigma_f = \{\sigma_{f_1}, \sigma_{f_2}\}$ . Suponha que o conjunto de eventos de falha possa ser particionado como  $\Sigma_f = \Sigma_{f_1} \dot{\cup} \Sigma_{f_2}$ , sendo  $\Sigma_{f_1} = \{\sigma_{f_1}\}$  e  $\Sigma_{f_2} = \{\sigma_{f_2}\}$ . Seguindo os passos para obtenção da rede de Petri observadora apresentado em*

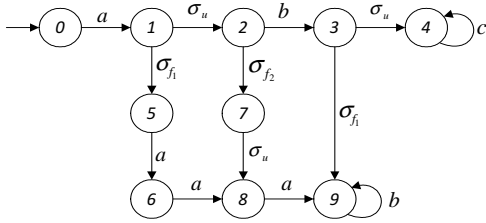


Figura 1: Autômato  $G$  de um SED.

Moreira et al. (2012a), obtém-se a rede  $\mathcal{N}_{SO}$  da figura 2. A partir dessa rede, a rede de Petri para diagnose  $\mathcal{N}_D$  pode ser obtida acrescentando-se a  $\mathcal{N}_{SO}$  transições associadas aos conjuntos de eventos de falha e arcos inibidores, como mostrado na figura 3.

Suponha, agora, que a sequência de falha  $s = a\sigma_{f_1}aa$  seja executada pelo sistema. Então, a sequência observada é  $\nu = aaa$ . Após a ocorrência do primeiro evento  $a$ , a rede de Petri para diagnose informa a estimativa de estados do sistema  $\{(7N_1, F_2), (2N_1, 2N_2), (F_1, 5N_2), (1N_1, 1N_2), (8N_1, F_2)\}$ . Quando, o segundo evento  $a$  é observado, as transições  $t_{SO_2}, t_{SO_4}, t_{SO_5}, t_{SO_7}, t_{SO_8}$  disparam e o conjunto de lugares com uma ficha é dado por  $\{(F_1, 6N_2), (9N_1, F_2)\}$ . Após a ocorrência do terceiro evento  $a$ , as transições  $t_{SO_{12}}, t_{SO_{14}}$  disparam e o único lugar que permanece com uma ficha é  $(F_1, 8N_2)$ . Uma vez que todos os lugares rotulados com  $N_1$  não possuem ficha, então a transição  $t_{f_1}$ , rotulada com o evento  $\lambda$ , é habilitada e dispara, removendo a ficha do lugar  $p_{N_1}$  e adicionando uma ficha a  $p_{F_1}$ , indicando a ocorrência do evento de falha  $\sigma_{f_1}$ .

### 3 Aspectos práticos sobre implementação de diagnosticadores em CLPs

Controladores lógicos programáveis são geralmente utilizados para implementação de códigos de controle de sistemas a eventos discretos. Contudo, é possível utilizar CLPs também para a implementação de diagnosticadores online de falhas. Para tanto, pode-se utilizar um CLP dedicado exclusivamente para a diagnose da falha ou então implementar o código de diagnose junto do código de controle. A vantagem de se implementar o diagnosticador no mesmo CLP do controlador está na redução do número de sensores e/ou de barramentos de comunicação necessários para informar a ocorrência dos eventos associados a ações. Neste trabalho é suposto que o código do diagnosticador é implementado no mesmo CLP que o código do controlador a eventos discreto.

Bordas de subida ou de descida dos sinais de sensores são utilizadas para identificar a ocorrência de eventos provenientes da planta, enquanto que os eventos associados a ações executadas pelo código de controle, implementado no CLP, podem ser identificadas internamente no código do diagnosticador. Para tanto, as linhas do código do di-

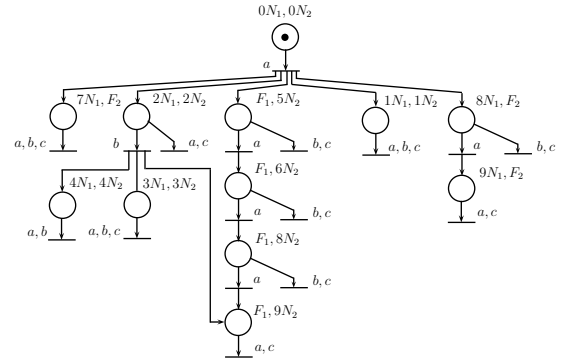


Figura 2: Rede de Petri observadora  $\mathcal{N}_{SO}$ .

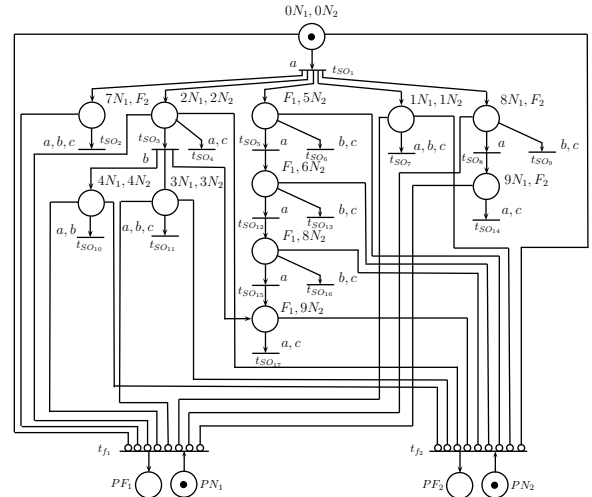


Figura 3: Rede de Petri para diagnose  $\mathcal{N}_D$ .

agnosticador devem preceder as linhas do código de controle. Isso evita que o diagnosticador tenha como entrada um evento da planta, simultaneamente com um evento de ação gerado pelo controlador como resposta a esse evento da planta. É importante ressaltar que o diagnosticador é um elemento passivo que apenas observa as ocorrências de eventos e informa a detecção dos eventos de falha, tendo como hipótese que dois eventos não são observados simultaneamente pelo diagnosticador. O caso em que dois eventos podem ser observados simultaneamente não é trivialmente resolvido e requer um estudo da resolução de conflitos no sistema. Neste trabalho é suposto que dois eventos não ocorrem simultaneamente.

Dependendo da forma com que um sistema a eventos discreto é implementado em um CLP, pode ocorrer que duas ou mais transições rotuladas por um mesmo evento  $\sigma$  sejam simultaneamente transpostas quando  $\sigma$  ocorre, mesmo que inicialmente não estejam todas habilitadas. Esse efeito é conhecido na literatura como efeito avalanche (Fabian e Hellgren, 1998). Visando contornar esse problema, em Fabian e Hellgren (1998) são apresentadas algumas formas para evitar o efeito avalanche em sistemas descritos por autômatos e SFCs. Embora essas formas tenham sido apli-

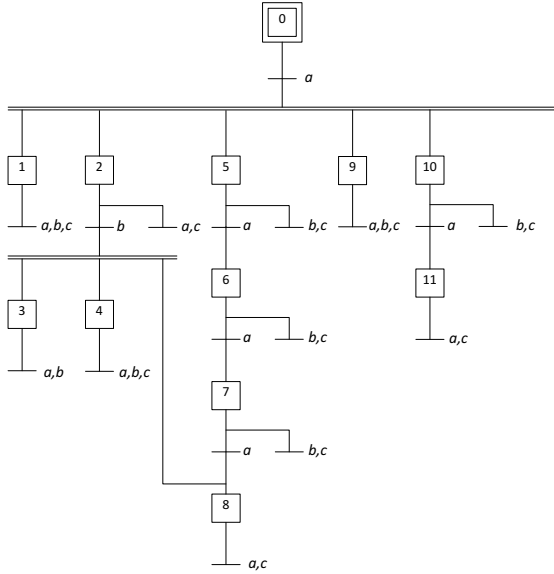


Figura 4: SFC de  $\mathcal{N}_{SO}$ .

cadadas com sucesso em alguns exemplos, não são facilmente empregadas quando o sistema contém laços.

Um método de conversão que estabelece regras simples de transformação de redes de Petri interpretadas para controle para diagramas ladder é apresentado em Moreira et al. (2009). O método proposto em Moreira et al. (2009) preserva a estrutura de rede de Petri e evita o efeito avalanche mesmo que a rede de Petri contenha laços. Esse método será estendido neste trabalho para a implementação de redes de Petri para diagnose.

#### 4 Implementação em CLP de uma rede de Petri para diagnose de falhas

##### 4.1 Conversão de redes de Petri para diagnose de falhas em SFC

Uma vez que a rede de Petri para diagnose é uma rede binária, o diagrama em SFC pode ser obtido de forma quase direta. O código do diagnosticador pode ser dividido em  $r + 1$  SFCs parciais, em que um SFC parcial corresponde à rede de Petri observadora de estados, e os outros  $r$  SFCs representam os testes para verificação da ocorrência dos eventos de falha para cada um dos  $r$  tipos de falha. Na figura 4, o SFC do observador de estados  $\mathcal{N}_{SO}$  da figura 2 é apresentado. Cada lugar de  $\mathcal{N}_{SO}$  é transformado em uma etapa do SFC, e as transições permanecem inalteradas. A tabela 1 apresenta a correspondência entre cada lugar da rede de Petri observadora e a etapa associada do SFC. Nas figuras 5 e 6 os testes para verificação das falhas são realizados para os dois tipos de falha. Esses SFCs possuem apenas duas etapas associadas aos lugares  $p_{N_k}$  e  $p_{F_k}$  e a única transição  $t_{f_k}$  tem receptividade rotulada com uma expressão Booleana que representa o comportamento dos arcos inibidores da RPD. Quando essa expressão

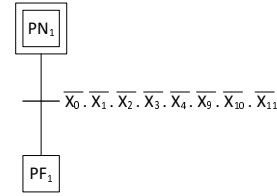


Figura 5: SFC para verificação da ocorrência do evento de falha  $\sigma_{f_1}$ .

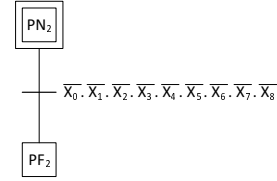


Figura 6: SFC para verificação da ocorrência do evento de falha  $\sigma_{f_2}$ .

torna-se verdadeira, a etapa associada ao lugar  $p_{F_k}$  é ativada e as ações associadas a essa etapa são realizadas.

Tabela 1: Correspondência entre os lugares do observador de estados  $\mathcal{N}_{SO}$  e as etapas da sua implementação em SFC.

Lugares	Etapas
$0N_10N_2$	0
$7N_1F_2$	1
$2N_12N_2$	2
$4N_14N_2$	3
$3N_13N_2$	4
$F_15N_2$	5
$F_16N_2$	6
$F_18N_2$	7
$F_19N_2$	8
$1N_11N_2$	9
$8N_1F_2$	10
$9N_1F_2$	11

##### 4.2 Conversão de redes de Petri para diagnose de falhas em diagramas Ladder

O método consiste em dividir o código de programação da RPD em quatro módulos. O primeiro módulo é associado à inicialização da rede de Petri, ou seja, define o estado inicial da rede. O segundo módulo é associado à observação de eventos a partir da mudança no estado lógico de sensores da planta ou de variáveis associadas a comandos enviados pelo controlador para a planta. O terceiro módulo descreve as condições necessárias para o disparo das transições e o quarto módulo descreve a evolução das fichas na rede de Petri. O quarto módulo pode ser utilizado também para definir os alarmes que podem ser acionados quando ocorre a detecção da falha.

A técnica de conversão proposta neste trabalho fornece uma completa visualização da RPD no diagrama ladder, permitindo que a rede seja facil-

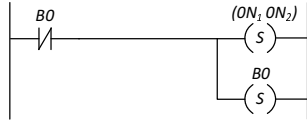


Figura 7: Módulo de inicialização.

mente extraída a partir do código de programação.

Nas próximas subseções, cada um dos quatro módulos é descrito em detalhes e a rede de Petri para diagnóstico  $\mathcal{N}_D$  do exemplo 1 é utilizada para ilustrar o método.

#### 4.3 Módulo de inicialização

A primeira linha do diagrama ladder descreve a inicialização da rede de Petri. Essa linha contém um contato normalmente fechado (NF) associado a uma variável binária interna que, no primeiro ciclo de varredura, energiza logicamente as bobinas associadas aos lugares que possuem uma ficha no estado inicial. Após o primeiro ciclo de varredura, o contato NF é aberto.

Na figura 7 a primeira linha do diagrama ladder para o diagnosticador  $\mathcal{N}_D$  da figura 3 é mostrada. A variável binária interna associada ao contato NF é  $B0$ . Note que, de acordo com o estado inicial de  $\mathcal{N}_D$ , apenas o lugar  $(0N1, 0N2)$  deve possuir uma ficha. Assim, uma bobina com a instrução  $SET$  é associada a uma variável interna relacionada com o lugar  $(0N1, 0N2)$ .

#### 4.4 Módulo dos eventos

Eventos da planta são associados à borda de subida ou de descida de sinais de sensores, enquanto que eventos relacionados a ações são associados a mudanças nos níveis lógicos de variáveis de saída do controlador. Para detectar essas transições de nível de variáveis binárias, alguns CLPs possuem contatos especiais que são fechados quando a expressão booleana anterior ao contato, na mesma linha de programação, muda seu estado lógico, e mantém o contato fechado por apenas um ciclo de varredura. Em alguns CLPs existem contatos sensíveis a transições positivas (P) e negativas (N). Neste trabalho esse tipo de contato é utilizado para observar a ocorrência de um evento, como mostrado na figura 8, em que há três eventos a serem observados. Note que as entradas são associadas às variáveis  $S_a$ ,  $S_b$ , e  $S_c$  que representam sinais de sensores da planta ou variáveis de saída do controlador, enquanto que as bobinas são utilizadas para representar a ocorrência do evento. As variáveis associadas a essas bobinas é que indicam que o evento ocorreu no restante do código do diagnosticador.

#### 4.5 Módulo das condições para o disparo das transições

O módulo das condições para o disparo das transições possui  $|T_D|$  linhas, em que cada linha corres-

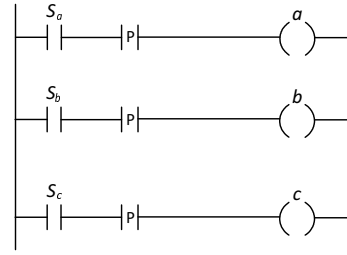


Figura 8: Módulo de eventos.

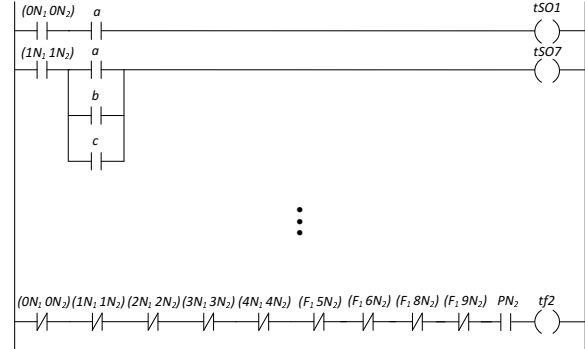


Figura 9: Módulo das condições para o disparo das transições.

ponde às condições para o disparo de uma transição.

A ideia básica é descrever a regra de habilitação de uma transição, ou seja, uma transição  $t_j$  está habilitada quando o número de fichas em um lugar de entrada  $p_i$  é igual a um, para o caso em que  $Pre_D(p_i, t_j) = 1$ , e igual a zero, para o caso em que  $In_D(p_i, t_j) = 1$ <sup>1</sup>. Para tanto, contatos normalmente abertos associados aos lugares  $p_i$  tais que  $Pre_D(p_i, t_j) = 1$  e contatos normalmente fechados associados aos lugares  $p_i$  tais que  $In_D(p_i, t_j) = 1$ , são conectados em série na linha da habilitação da transição  $t_j \in T_D$ . É importante ressaltar que as transições pertencentes a  $T_{SO} \subset T_D$  possuem também contatos associados às variáveis que representam observações das ocorrências de eventos. Além disso, as únicas transições que possuem arcos inibidores são as transições pertencentes a  $T_{f_k}$ .

Na figura 9 são apresentadas apenas três linhas do diagrama ladder de  $\mathcal{N}_D$  da figura 3. As duas primeiras linhas representam as condições para o disparo das transições  $t_{SO1}$  e  $t_{SO7}$  do diagnosticador da figura 3, e a última linha as condições para o disparo de  $t_{j2}$ . Esse módulo completo possui 19 linhas, lembrando que cada linha corresponde às condições para o disparo de uma transição.

#### 4.6 Módulo da dinâmica da rede de Petri

Após o disparo de uma transição  $t_j$ , o número de fichas em uma rede de Petri deve ser atualizado.

<sup>1</sup>É importante lembrar que a rede de Petri para diagnóstico de falhas é uma rede de Petri binária.

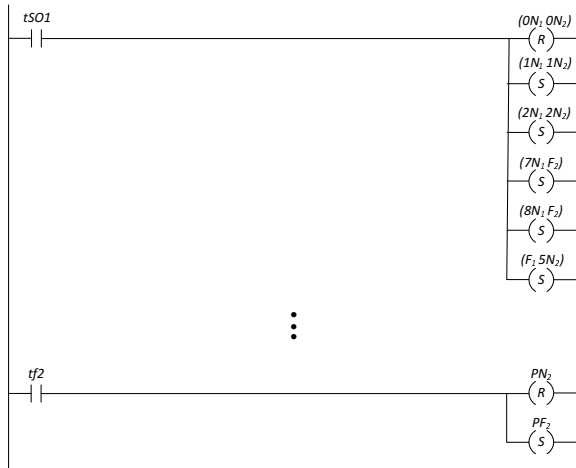


Figura 10: Módulo da dinâmica da rede de Petri.

Esse processo é descrito pela equação de estado da rede de Petri. Esse módulo possui  $|T_D|$  linhas, em que cada linha é associada a uma transição e expressa as mudanças nas marcações dos lugares de entrada e de saída da transição.

Como a rede de Petri é binária, então todos os lugares são seguros, o que permite que apenas bobinas SET e RESET sejam utilizadas para indicar a evolução das fichas.

Na figura 10 são mostradas apenas duas linhas do módulo da dinâmica da rede de Petri para diagnóstico  $\mathcal{N}_D$  da figura 3. A primeira linha está associada ao disparo da transição  $t_{SO1}$  e a última ao disparo de  $t_{f2}$ . Note que é possível ainda associar uma bobina de saída na linha da transição que indica a falha  $t_{f_k}$  para gerar, por exemplo, um alarme indicando para o usuário a ocorrência do evento de falha do conjunto  $\Sigma_{f_k}$ .

#### 4.7 Observações sobre o método

Algumas observações importantes sobre o método apresentado nesta seção podem ser feitas: (i) assim como em Moreira et al. (2009), a ordenação das linhas evita o efeito avalanche uma vez que cada marcação da RPD permanece inalterada por pelo menos um ciclo de varredura na sua implementação em diagrama ladder; (ii) uma vez que existem  $|\Sigma|$  eventos distintos associados a mudanças no nível lógico de variáveis binárias, então o número máximo de linhas no programa ladder obtido a partir do método é  $(|\Sigma| + 2 \times |T_D| + 1)$ ; (iii) a forma como é construído o diagrama ladder deixa clara a estrutura da rede de Petri e permite o acompanhamento da evolução das fichas na rede. A principal vantagem desse fato é que a RPD pode ser obtida diretamente do diagrama ladder e vice-versa.

## 5 Conclusões

Neste artigo são apresentados métodos para obtenção de um SFC e de um diagrama ladder para

implementação em controladores lógicos programáveis de redes de Petri para diagnose. Esses métodos são baseados em regras simples e permitem uma completa visualização da estrutura da rede de Petri.

## Referências

- Alayan, H. e Newcomb, R. W. (1987). Binary petri-net relationships, *IEEE transactions on circuits and systems* **CAS-34**: 565–568.
- Fabian, M. e Hellgren, A. (1998). PLC-based implementation of supervisory control for discrete event systems, *37th IEEE Conference on Decision and Control*, Tampa, Florida USA, pp. 3305–3310.
- Hellgren, A., Fabian, M. e Lennartson, B. (2005). On the execution of sequential function charts, *Control Engineering Practice* **13**: 1283–1293.
- ISO/IEC (2001). *International standard IEC 61131-3*, 2nd edn, ISO/IEC.
- Luca, F., Massimo, A. e Alessio, D. (2011). A methodology for fault isolation and identification in automated equipments, *9th IEEE International Conference on Industrial Informatics*, Lisbon, Portugal, pp. 157–162.
- Moreira, M. V., Botelho, D. S. e Basilio, J. C. (2009). Ladder Diagram Implementation of Control Interpreted Petri Nets: a State Equation Approach, *4th IFAC Workshop on Discrete-Event System Design*, Gandia Beach, Spain, pp. 85–90.
- Moreira, M. V., Cabral, F. G. e Diene, O. (2012a). Diagnosticador rede de Petri para um SED modelado por um autômato finito, *XIX Congresso Brasileiro de Automática*, Campina Grande - PB, pp. 3723–3730.
- Moreira, M. V., Cabral, F. G. e Diene, O. (2012b). Petri net diagnoser for DES modeled by finite state automaton, *51st IEEE Conference on Decision and Control*, Maui - HI, pp. 6742–6748.
- Moreira, M. V., Jesus, T. C. e Basilio, J. C. (2011). Polynomial time verification of decentralized diagnosability of discrete event systems, *IEEE Transactions on Automatic Control* pp. 1679–1684.
- Qiu, W. e Kumar, R. (2006). Decentralized failure diagnosis of discrete event systems, *IEEE Transactions on Systems, Man, and Cybernetics Part A: Systems and Humans* **36**(2).
- Sampath, M., Sengupta, R., Lafortune, S., Sinnamohideen, K. e Teneketzis, D. (1995). Diagnosability of discrete-event systems, *IEEE Trans. on Automatic Control* **40**(9): 1555–1575.
- Sampath, M., Sengupta, R., Lafortune, S., Sinnamohideen, K. e Teneketzis, D. (1996). Failure diagnosis using discrete-event models, *IEEE Trans. on Control Systems Technology* **4**(2): 105–124.
- Uzam, M. e Jones, A. H. (1998). Discrete event control system design using automation petri nets and their ladder diagram implementation, *International Journal of Advanced Manufacturing Technology* **14**: 716–728.