Diagnosticabilidade de falhas repetidas de sistemas a eventos discretos com estrutura descentralizada ⋆

Guilherme M. O. Silva * **João C. Basilio** *

* *Departamento de Engenharia Elétrica, Universidade Federal do Rio de Janeiro, 21949-900, Rio de Janeiro, Brazil (e-mails: guilherme.ottoni@coppe.ufrj.br,basilio@dee.ufrj.br.*

Abstract: This paper extends the results about repeated fault diagnosability of monolithic discrete-event systems to systems with decentralized structure. To this end, it was introduced the notion of κ-codiagnosability, which basically consists of ensuring that at least one partially-observed diagnoser is able to accurately determine that a fault event has occurred at least κ times. Moreover, a necessary and sufficient condition for κ-codiagnosability is presented and an algorithm is proposed for its verification in discrete-event systems modeled by regular languages. This proposed algorithm is based on a verifier automaton and, thus, its computational complexity is polynomial.

Resumo: Este artigo estende os resultados existentes sobre diagnosticabilidade de falhas repetidas de um sistema a eventos discretos monolítico a sistemas com estrutura descentralizada. Para tanto, foi introduzida a definição de codiagnosticabilidade-κ, que consiste basicamente em assegurar que ao menos um diagnosticador que tenha observação parcial dos eventos do sistema seja capaz de determinar com exatidão que um evento de falha ocorreu pelo menos κ vezes. Além disso, é apresentada uma condição necessária e suficiente para a codiagnosticabilidade-κ e é proposto um algoritmo para a sua verificação em sistemas a eventos discretos modelados por linguagens regulares. O algoritmo proposto é baseado em um autômato verificador e, por isso, possui complexidade computacional polinomial.

Keywords: Discrete event systems, Automata, Repeated faults, Diagnosability

Palavras-chaves: Sistemas a eventos discretos, Autômatos, Falhas repetidas, Diagnosticabilidade

1. INTRODUÇÃO

O diagnóstico de falhas é fundamental para assegurar um bom funcionamento dos sistemas industriais. Nesse sentido, a verificação se uma falha é diagnosticável representa um passo fundamental para o diagnóstico de falhas, uma vez que assegura que os diagnosticadores projetados para detectar a ocorrência da falha, de fato, irão detectá-la.

Uma falha representa qualquer desvio do comportamento normal ou esperado de um sistema (Basilio et al., 2021). O conceito de diagnosticabilidade de falhas de sistemas de eventos discretos (SEDs) foi introduzido por Sampath et al. (1995) e estendido para codiagnosticabilidade em Debouk et al. (2000). Os algoritmos propostos em Sampath et al. (1995) e Debouk et al. (2000) foram generalizados em Viana e Basilio (2019), onde a hipótese de não existência de ciclos de estados conectados por eventos não observáveis foi relaxada; além disso, a busca por ciclos é substituída por busca por componentes fortemente conectadas, que possui complexidade polinomial. Algoritmos para verificação da diagnosticabilidade de SEDs com complexidade polinomial foram propostos em Yoo e Lafortune (2002), Qiu e Kumar (2006) e Moreira et al. (2011).

Uma mesma falha pode ocorrer múltiplas vezes durante a operação de um sistema, ampliando o contexto da diagnosticabilidade de falhas para o da diagnosticabilidade de falhas repetidas (Boussif et al., 2021), no qual o objetivo é determinar o número de ocorrências da falha. Algoritmos para a verificação de diagnosticabilidade de falhas repetidas foram propostos em Jiang et al. (2003), Jeron et al. (2006), Yoo e Garcia (2009) e Zhou e Kumar (2009).

Neste artigo, o problema da diagnosticabilidade de falhas repetidas de SEDs é revisitado e estendido para SEDs com estrutura descentralizada. A estrutura considera neste artigo é idêntica à considerada em Debouk et al. (2000), na qual múltiplos locais de observação fazem a leitura parcial dos eventos do sistema (observação parcial) que os informa a um diagnosticador local, que, por sua vez, informa a um coordenador central se, baseado na sua observação parcial, a falha ocorreu ou não. A decisão sobre a ocorrência da falha é tomada pelo coordenador de forma disjuntiva, isto

⋆ Este trabalho foi parcialmente financiado pela Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES), código de financiamento 001. O trabalho de J. C. Basilio teve também o apoio financeiro do Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq), processo número 316881/2021-0, e da Fundação Carlos Chagas Filho de Amparo á Pesquisa do Estado do Rio de Janeiro, FAPERJ, processo número E-26/201.144-2022.

é, o coordenador declara que a falha ocorreu se, pelo menos, um dos diagnosticadores locais declara a ocorrência da falha. Usando essa mesma estrutura descentralizada, este trabalho introduz a noção de codiagnosticabilidade-κ, que assegura que ao menos um dos diagnosticadores locais é capaz de garantir que um evento de falha ocorreu, pelo menos κ vezes. Uma condição necessária e suficiente para a codiagnosticabilidade-κ para SEDs com linguagens regulares e um algoritmo baseado na construção de um autômato verificador que utiliza o padrão de supervisão para falhas repetidas (Jeron et al., 2006) são apresentados. O verificador aqui proposto estende, para falhas repetidas, aquele proposto em Moreira et al. (2011).

Este artigo está estruturado da seguinte forma: na Seção 2 são apresentadas as noções básicas sobre diagnosticabilidade de falhas de SEDs; na Seção 3 é introduzida a noção de codiagnosticabilidade-κ e são apresentados uma condição necessária e suficiente para essa propriedade e um algoritmo para a verificação baseado na construção de um autômato verificador; na Seção 4 é feita a análise da complexidade computacional do algoritmo proposto; e na Seção 5, são apresentados os comentários finais.

2. FUNDAMENTAÇÃO TEÓRICA

2.1 SED modelados por autômatos

O formalismo aqui utilizado para modelagem de SEDs, é o autômato de estados finitos (AEF) (Hopcroft et al., 2006; Cassandras e Lafortune, 2008; Basilio et al., 2021), que é definido pela sêxtupla $G = (X, \Sigma, f, \Gamma, x_0, X_m)$, sendo X o conjunto finito de estados, Σ o conjunto finito de eventos, $f : X \times \Sigma \to X$ a função de transição de estados, $\Gamma : X \to 2^{\Sigma}$ o conjunto de eventos ativos, definido por $\Gamma(x) = \{\sigma \in \Sigma : f(x,\sigma)!\}$ ($f(x,\sigma)!$ denota que $f(x,\sigma)$ é definida, i.e., existe $y \in X$ tal que $f(x,\sigma) = y$), x_0 é o estado inicial e X_m o conjunto de estados marcados. A função de transição pode ser estendida para $f : X \times \Sigma^* \to X$, em que Σ^* denota o o Fecho de Kleene de Σ [1], e ε a sequência nula, por meio da seguinte recursão: $f(x,\varepsilon) = x$; e $f(x, s\sigma) = f(f(x,s),\sigma)$, $\sigma \in \Sigma, s \in \Sigma^*$. O conjunto de eventos Σ pode ser particionado como $\Sigma = \Sigma_o \dot{\cup} \Sigma_{uo}$, em que Σ_o e Σ_{uo} são os conjuntos de eventos observáveis e não-observáveis, respectivamente.

A linguagem gerada por um autômato G é definida por $\mathcal{L}(G) = \{s \in \Sigma^* : f(x_0, s)!\}$, sendo denotada por $\mathcal{L}$. Uma linguagem é dita regular se puder ser representada por um AEF. Uma linguagem regular é viva se, para todo estado que a representa existir uma transição definida por algum evento de Σ.

No decorrer do texto, o fecho de prefixo de uma sequência s será denotado por $Pre(s)$, sendo definido por $Pre(s) = \{u \in \Sigma^* : (\exists v \in \Sigma^*)[uv = s]\}$. A pós-linguagem de $\mathcal{L}$ após uma sequência s é definida como $\mathcal{L}/s = \{t \in \Sigma^* : st \in \mathcal{L}\}$. Dados dois conjuntos de eventos Σ_a e Σ_b, em que $\Sigma_b \subset \Sigma_a$, a projeção natural $P_{a,b} : \Sigma_a^* \to \Sigma_b^*$, ou simplesmente projeção, é definida conforme Ramadge e Wonham (1989) pela seguinte recursão: $P_{a,b}(\varepsilon) = \varepsilon$; $P_{a,b}(\sigma) = \sigma$, se $\sigma \in \Sigma_b$, ou $P_{a,b}(\sigma) = \varepsilon$, se $\sigma \in \Sigma_a \setminus \Sigma_b$; $P_{a,b}(s\sigma) = P_{a,b}(s)P_{a,b}(\sigma)$, $s \in \Sigma_a^*$, $\sigma \in \Sigma_a$. A projeção inversa é definida por $P_{a,b}^{-1} : \Sigma_b^* \to 2^{\Sigma_a^*}$, em que $P_{a,b}^{-1}(t) = \{s \in \Sigma_a^* : P_{a,b}(s) = t\}$ e $\Sigma_b \subseteq \Sigma_a$. As definições de fecho de prefixo, projeção e projeção inversa podem ser estendidas para uma linguagem, aplicando-as em cada sequência.

A acessibilidade é uma operação que exclui todos os estados de um autômato que não são alcançáveis por nenhuma sequência que parte do estado inicial, sendo definida por $Ac(G) = (X_{Ac}, \Sigma, f_{Ac}, \Gamma_{Ac}, x_0, X_{m,Ac})$, sendo: $X_{Ac} = \{x \in X : (\exists s \in \Sigma^*)[f(x_0, s) = x]\}$; $f_{Ac} : X_{Ac} \times \Sigma^* \to X_{Ac}$, em que $f_{Ac}(x,\sigma) = f(x,\sigma)$, se $f(x,\sigma) \in X_{Ac}$, ou $f(x,\sigma)$ não é definida, caso contrário, e $\Gamma_{Ac}(x) = \{\sigma \in \Sigma : f_{Ac}(x,\sigma)!\}$. A coacessibilidade é uma operação que exclui todos os estados de um autômato a partir dos quais não existe um caminho que conduz a um estado marcado, sendo definida por $Coac(G) = (X_{Coac}, \Sigma, f_{Coac}, \Gamma_{Coac}, x_{0,Coac}, X_{Coac,m})$, sendo $X_{Coac} = \{x \in X : (\exists s \in E^*)[f(x,s) \in X_m]\}$, $x_{0,Coac} = x_0$, se $x_0 \in X_{Coac}$, ou não definida caso contrário, e $f_{Coac} : X_{Coac} \times \Sigma^* \to X_{Coac}$ em que $f_{Coac}(x,\sigma) = f(x,\sigma)$, se $f(x,\sigma) \in X_{Coac}$, ou $f(x,\sigma)$ não é definida, caso contrário.

Sejam $G_1 = (X_1, \Sigma_1, f_1, \Gamma_1, x_{0,1}, X_{m,1})$ e $G_2 = (X_2, \Sigma_2, f_2, \Gamma_2, x_{0,2}, X_{m,2})$ dois AEFs. A composição paralela entre eles é definida por $G_1||G_2 = Ac(X_1 \times X_2, \Sigma_1 \cup \Sigma_2, f_{1||2}, \Gamma_{1||2}, (x_{0,1}, x_{0,2}), (X_{m,1} \times X_{m,2}))$, sendo $f_{1||2}((x_1,x_2),\sigma) = (f_1(x_1,\sigma), x_2)$, se $\sigma \in \Gamma_1(x_1) \setminus \Sigma_2$; $f_{1||2}((x_1,x_2),\sigma) = (x_1, f_2(x_2,\sigma))$, se $\sigma \in \Gamma_2(x_2) \setminus \Sigma_1$; $f_{1||2}((x_1,x_2),\sigma) = (f_1(x_1,\sigma), f_2(x_2,\sigma))$, se $\sigma \in \Gamma_1(x_1) \cap \Gamma_2(x_2)$, ou não definida, caso contrário.

Com o objetivo de se definir o observador $Obs(G, \Sigma_o)$ de um autômato G com relação a um conjunto de eventos observáveis Σ_o, é necessário definir o alcance não observável de um estado $x \in X$ da seguinte forma: $UR(x, \Sigma_{uo}) = \{x' \in X : (\exists s \in \Sigma_{uo}^*)[f(x,s) = x']\}$; portanto $x \in UR(x, \Sigma_{uo})$. O alcance não observável de um estado pode ser estendido a um conjunto de estados $Y \in 2^X$ da seguinte forma: $UR(Y, \Sigma_{uo}) = \bigcup_{x \in Y} UR(x, \Sigma_{uo})$. Partindo desta definição, o observador de um autômato G é um AEF formalmente definido por: $G_{obs} = Obs(G, \Sigma_o) = (X_{obs}, \Sigma_o, f_{obs}, \Gamma_{obs}, x_{0,obs}, X_{m,obs})$, sendo $X_{obs} \in 2^X$, $f_{obs}(x_{obs},\sigma) = \bigcup_{x \in x_{obs} : f(x,\sigma)!} UR(f(x,\sigma), \Sigma_{uo})$, $\Gamma_{obs}(x_{obs}) = \bigcup_{x \in x_{obs}} \Gamma(x) \cap \Sigma_o$, e $x_{0,obs} = UR(x_0, \Sigma_{uo})$. Note que $\mathcal{L}(G_{obs}) = P_o(\mathcal{L}(G))$, em que $P_o : \Sigma^* \to \Sigma_o^*$, e para simplificar, será denotada por $\mathcal{L}_{obs}$.

Por fim, dados um grafo $D = (V, E)$, em que V e E representam os conjuntos de vértices e arestas, respectivamente, um conjunto de vértices $V_{scc} \subseteq V$, dizemos que V_{scc} é uma componente fortemente conectada (CFC) de D se (i) todos os pares de vértices u e v em V_{scc} são alcançáveis entre si, i.e., $\forall u, v \in V_{scc}((u \rightsquigarrow v) \wedge (v \rightsquigarrow u))$ [2]; (ii) V_{scc} é máximo, no sentido em que não há outros vértices a serem adicionados a V_{scc} que satisfazem a condição de alcançabilidade. Conforme mostrado em Cormen et al. (2007), a complexidade computacional de se calcular uma CFC é, no pior caso, linear com relação ao número de arestas e vértices de D, i.e, $O(V + E)$. Note que é possível que V_{scc} contenha apenas um elemento, sendo representado

[1] Dado um conjunto de eventos Σ, o seu fecho de Kleene é um conjunto formado por todas as sequências possíveis de comprimento finito formadas com os eventos de Σ mais a sequência nula ε.

[2] O símbolo $u \rightsquigarrow v$ significa que existe um caminho que conecta os vértices u e v.

por $V_{scc} = \{u\}$. Neste caso, existem duas possibilidades: $u \rightsquigarrow u$ por meio de autolaço, ou u é vértice de D. Para diferenciar estes dois casos, CFCs que contêm apenas um elemento que satisfazem o segundo caso são considerados CFCs triviais.

2.2 Noções sobre diagnosticabilidade de falhas

Seja $\Sigma_f \subseteq \Sigma_{uo}$ o conjunto formado pelos eventos de falha, i.e. eventos que conduzem o sistema a um comportamento indesejado, sendo que não necessariamente acarretam em interrupção de seu funcionamento. Por esse motivo, suas ocorrências devem ser diagnosticadas. Conforme mostrado em Santoro et al. (2017), pode-se supor, sem perda de generalidade, que o conjunto de eventos de falha possui um único evento, i.e., $\Sigma_f = \{\sigma_f\}$; quando há mais de um evento de falha, cada um deles deve ser considerado separadamente, com os eventos de falha restantes tratados como eventos não observáveis. A definição de diagnosticabilidade de falhas foi introduzida em Sampath et al. (1995), que considera que uma linguagem gerada por um autômato é diagnosticável se, após a ocorrência do evento de falha, é possível assegurar a falha ocorreu após a ocorrência de um número finito de eventos.

Com o objetivo de se definir formalmente a diagnosticabilidade de linguagens, faz-se necessário introduzir a definição do conjunto de todas as sequências em $\mathcal{L}$ que terminam com o evento de falha: $\Psi(\sigma_f) = \{\Sigma \setminus \Sigma_f\}^*\{\sigma_f\} \cap \mathcal{L}$. Além disso, suponha que $\|s\|$ denote o número de eventos em s.

Definição 1. (Diagnosticabilidade). (Sampath et al., 1995) Uma linguagem viva e prefixo-fechada $\mathcal{L}$ é diagnosticável com relação a $\Sigma_f = \{\sigma_f\}$ e $P_o : \Sigma^* \to \Sigma_o^*$, se a seguinte condição for satisfeita:

$$(\exists n \in \mathbb{N})(\forall s \in \Psi(\sigma_f))(\forall t \in \mathcal{L}/s, \|t\| \geq n \to D),$$

em que a condição para a diagnosticabilidade D é dada por:

$$D : \forall \omega \in P_o^{-1}(P_o(st)) \cap \mathcal{L}, \sigma_f \in \omega.$$

Observe que a condição de diagnosticabilidade é violada caso exista uma sequência arbitrariamente longa $s_Y \in \mathcal{L}$ que contenha o evento de falha σ_f e outra sequência não necessariamente arbitrariamente longa $s_N \in \mathcal{L}$ que não contenha o evento de falha, tal que $P_o(s_Y) = P_o(s_N)$. Consequentemente, o observador externo não será capaz de afirmar que o evento de falha σ_f ocorreu ou não.

A Definição 1 considera uma estrutura simples, contendo apenas um diagnosticador capaz de ter acesso a apenas um conjunto de eventos observáveis. Entretanto, é possível que as observações de eventos por parte do sistema sejam descentralizadas, existindo múltiplos diagnosticadores e locais de observação, fazendo com que haja múltiplos padrões de observação de eventos. Neste caso, cada diagnosticador local possui seu próprio conjunto de eventos observáveis. Isto conduz à definição de codiagnosticabilidade, que foi introduzida em Debouk et al. (2000). Na estrutura descentralizada, existem m diagnosticadores com diferentes conjuntos de eventos observáveis, denotados por $\Sigma_{o,i}$, $1 \leq i \leq m$. A linguagem será codiagnosticável se ao menos um diagnosticador for capaz de afirmar com precisão a ocorrência de falha para toda sequência possível.

Definição 2. (Codiagnosticabilidade).(Debouk et al., 2000) Uma linguagem viva e prefixo-fechada $\mathcal{L}$ é codiagnosticável com relação a $\Sigma_f = \{\sigma_f\}$ e $P_{o,i} : \Sigma^* \to \Sigma_{o,i}^*$, $i = 1, 2, \ldots, m$ se a seguinte condição for satisfeita:

$$(\exists n \in \mathbb{N})(\forall s \in \Psi(\sigma_f))(\forall t \in \mathcal{L}/s, \|t\| \geq n \to CoDiag),$$

em que a condição para a codiagnosticabilidade $CoDiag$ é dada por:

$$CoDiag : (\exists i \in \{1, \ldots m\})$$
$$(\forall \omega \in P_{o,i}^{-1}(P_{o,i}(st)) \cap \mathcal{L}, \sigma_f \in \omega).$$

As definições 1 e 2 não fazem distinção entre diagnosticar uma única ou múltiplas ocorrências da falha. Quando o objetivo for diagnosticar múltiplas ocorrências da falha, faz-se necessário introduzir a noção de diagnosticabilidade de falhas repetidas. Para tanto, suponha que N_s^F denote o número de vezes em que o evento de falha σ_f ocorreu em uma sequência $s \in \Sigma^*$ e seja $\mathbb{Z}_+^*$ o conjunto dos números inteiros positivos.

Definição 3. (Diagnosticabilidade-κ). (Jiang et al., 2003) Uma linguagem viva e prefixo-fechada $\mathcal{L}$ é diagnosticável-κ, $\kappa \in \mathbb{Z}_+^*$ fixo, com relação a $P_o : \Sigma^* \to \Sigma_o^*$ e Σ_f se as seguintes condições forem verdadeiras:

$$(\exists n_\kappa \in \mathbb{N})(\forall s \in \mathcal{L}, N_s^F \geq \kappa)(\forall t \in \mathcal{L}/s, \|t\| \geq n_\kappa \to D_\kappa),$$

em que a condição de diagnosticabilidade-κ D_κ é dada por:

$$D_\kappa : \forall \omega \in P_o^{-1}(P_o(st)) \cap \mathcal{L}, N_\omega^F \geq \kappa.$$

De acordo com a definição 3, uma linguagem $\mathcal{L}$ não é diagnosticável-κ se existir uma sequência s contendo ao menos κ ocorrências de evento de falha, uma continuação arbitrariamente longa t de s em $\mathcal{L}$, e uma sequência $\omega \in \mathcal{L}$, que satisfaça as seguintes condições: *(i)* $P_o(st) = P_o(\omega)$; *(ii)* $N_\omega^F < \kappa$. Observe que quando $\kappa = 1$, a definição de diagnosticabilidade-κ se reduz à definição 1.

É importante ressaltar que a diagnosticabilidade-κ não é monótona, i.e, diagnosticabilidade-κ não necessariamente implica em diagnosticabilidade-$(\kappa - 1)$ ou diagnosticabilidade-$(\kappa + 1)$.

3. CODIAGNOSTICABILIDADE-κ

Considere a mesma arquitetura descentralizada contendo m diagnosticadores locais considerada na definição 2 e suponha que desejamos diagnosticar múltiplas ocorrências de eventos da falha. Para tanto, deve-se estender a definição 2 para falhas repetidas, conduzindo à noção de codiagnosticabilidade-κ.

Definição 4. (Codiagnosticabilidade-κ). Uma linguagem viva e prefixo-fechada $\mathcal{L}$ é codiagnosticável-κ com relação a $\Sigma_f = \{\sigma_f\}$ e $P_{o,i} : \Sigma^* \to \Sigma_{o,i}^*$, se a seguinte condição for satisfeita:

$$(\exists n_\kappa \in \mathbb{N})(\forall s \in \mathcal{L}, N_s^F \geq \kappa)(\forall t \in \mathcal{L}/s, \|t\| \geq n \to CoDiag_\kappa),$$

em que a condição de codiagnosticabilidade-κ $CoDiag_\kappa$ é dada por:

$$CoDiag_\kappa : (\exists i \in \{1, ..., m\})(\forall \omega \in P_{o,i}^{-1}(P_{o,i}(st)) \cap \mathcal{L}, N_\omega^F \geq \kappa).$$

A definição 4 indica que, para a estrutura descentralizada disjuntiva com m diagnosticadores proposta por Debouk et al. (2000), uma linguagem $\mathcal{L}$ é codiagnosticável-κ se ao menos um dos diagnosticadores locais for capaz de diagnosticar corretamente que o evento de falha ocorreu ao menos κ vezes. Caso contrário, i.e, se todos os diagnosticadores parciais permanecerem em dúvida, a linguagem não será codiagnosticável-κ.

3.1 Verificação de codiagnosticável-κ

Nesta seção será proposto um algoritmo para a verificação da codiagnosticabilidade-κ, baseado na construção de um autômato verificador. Para tanto, seja $G = (X, \Sigma, f, \Gamma, x_0)$ o autômato que modela a planta e considere as seguintes partições de Σ: $\Sigma_i = \Sigma_{o,i} \dot{\cup} \Sigma_{uo,i}$, $i = 1, \ldots, m$, sendo $\Sigma_{o,i}$ e $\Sigma_{uo,i}$ os conjuntos de eventos observáveis e não-observáveis do local de observação i. Além disso, suponha que a linguagem $\mathcal{L}(G)$ seja viva e que o evento de falha σ_f possa ocorrer repetidas vezes.

Considere o autômato contador (Jeron et al., 2006):

$$\mathcal{A}_\kappa = (X_{A_\kappa}, \Sigma_f, \Gamma_{A_\kappa}, f_{A_\kappa}, x_0), \tag{1}$$

em que $X_\kappa = \{0, 1, ..., \kappa\}$, $\Sigma_f = \{\sigma_f\}$, $f_{A_\kappa}(i, \sigma_f) = i + 1$, para $i = 0, \ldots, \kappa - 1$, $f_{A_\kappa}(\kappa, \sigma_f) = \kappa$, $\Gamma_{A_\kappa}(x) = \sigma_f, \forall x \in X_\kappa$ e $x_0 = 0$. Observe que o autômato $\mathcal{A}_\kappa$ possui $\kappa + 1$ estados e que seu único evento é a falha σ_f, que conecta todos os estados de zero até κ com um autolaço no estado κ. A figura 1 ilustra o autômato contador A_κ para $\kappa = 2$.

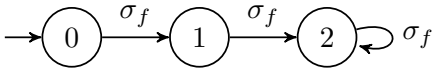

Figura 1. Autômato contador $\mathcal{A}_\kappa$ para $\kappa = 2$.

A seguir, calcula-se o autômato rotulado da seguinte forma:

$$G_\kappa = G \parallel \mathcal{A}_k = (X_\kappa, \Sigma, f_\kappa, \Gamma_\kappa, x_{0,\kappa}). \tag{2}$$

Note que os estados de G_κ são os estados de G rotulados pelo número de falhas ocorridas nas sequências que ligam o estado inicial ao estado considerado, i.e., os estados de G_κ têm a forma $x_\kappa = (x, k)$, em que $x \in X$ e $k \in \{0, \ldots, \kappa\}$. Não é difícil mostrar que $\mathcal{L}(G_\kappa) = \mathcal{L}(G)$.

Após o cálculo de G_κ, vamos calcular o autômato $G_L = (X_L, \Sigma, f_L, \Gamma_L, x_{0,\kappa})$, a partir de G_κ, que gera somente as sequências de $\mathcal{L}(G)$ que possuem menos de κ falhas da seguinte forma: *(i)* marcam-se todos os estados $x_L = (x, \kappa_\ell)$, sendo $\kappa_\ell < \kappa$; *(ii)* calcula-se a parte coacessível do autômato obtido no item *(i)* e; *(iii)* desmarcam-se os estados marcados. Note que G_L não terá estados rotulados com κ.

Vamos agora, definir o operador que renomeia o eventos não observáveis de cada local de observação.

Definição 5. (Função de renomeação). A função de renomeação R_i é uma operação que renomeia os eventos de G_L, sendo definida da seguinte forma: para $i = 1, 2, \ldots, m$,

$$R_i(\sigma) = \begin{cases} \sigma, & \sigma \in \Sigma_{o,i}, \\ \sigma_i, & \sigma \in \Sigma_{uo,i}. \end{cases} \tag{3}$$

Seja, então, $\Sigma_{LR,i} = R_i(\Sigma)$ e defina os seguintes autômatos para $i = 1, 2, \ldots, m$:

$$G_{LR,i} = (X_L, \Sigma_{LR,i}, f_{LR,i}, \Gamma_{LR,i}, x_{0,\kappa}), \tag{4}$$

sendo $f_{LR,i}(x, R_i(\sigma)) = f_L(x, \sigma)$ e $\Gamma_{LR,i} = R_i(\Gamma_L)$.

A ideia de se criar os autômatos $G_{LR,i}$ é fazer com que os eventos não observáveis, sejam eventos particulares de cada um dos autômatos $i \in \{1, 2, \ldots, m\}$, assim como feito em Moreira et al. (2011). A principal diferença é que aqui a falha também é renomeada para assegurar que não haja sincronização entre os eventos de falha de cada um dos autômatos locais individuais.

Com o objetivo de extrair de $\mathcal{L}(G)$ as sequências de falha com pelo menos κ ocorrências da falha, constrói-se o autômato $G_E = (X_E, \Sigma, f_E, \Gamma_E, x_{0,\kappa})$ a partir de G_κ, da seguinte forma: *(i)* marcam-se todos os estados $x_E = (x, \kappa)$; *(ii)* calcula-se a sua parte coacessível e; *(iii)* desmarcam-se os estados marcados. Note que a linguagem gerada pelo autômato G_E possui todas as sequências da linguagem gerada por G com a ocorrência da falha pelo menos κ vezes.

O autômato verificador é definido da seguinte forma:

$$G_v = (||_{i=1}^{m} G_{LR,i}) \,||\, G_E. \tag{5}$$

Note que os estados do verificador $G_v = (X_v, \Sigma_v, f_v, \Gamma_v, x_{0,v})$ possuem $m+1$ componentes, i.e., $x_v = (x_{LR,1}, x_{LR,2}, \ldots, x_{LR,m}, x_E)$, sendo $x_{LR,i} = (\tilde{x}_i, \kappa_{\ell_i})$, $\kappa_{\ell_i} < \kappa$, $i = 1, \ldots, m$, e $x_E = (x, k)$, $k \in \{0, 1, \ldots, \kappa\}$, com $x, \tilde{x}_i \in X$.

O seguinte resultado pode ser enunciado.

Teorema 1. Dado $\kappa \in \mathbb{Z}_+^*$, então, a linguagem gerada por $G = (X, \Sigma, f, \Gamma, x_0, X_m)$, $\mathcal{L}(G)$, não é codiagnosticável-κ com relação às projeções $P_{o,i} : \Sigma^* \to \Sigma_{o,i}^*$, $\forall i \in \{1, 2, \ldots, m\}$ e $\Sigma_f = \{\sigma_f\}$ se, e somente se, o verificador G_v obtido de acordo com a equação (5) tiver ao menos uma CFC não trivial, denotada por *cfc*, que contenha um estado $x_v = (x_{LR,1}, x_{LR,2}, \ldots, x_{LR,m}, x_E)$ que satisfaça as seguintes condições:

(1) $x_E = (x, \kappa)$;
(2) $(\forall i \in \{1, 2, \ldots, m\})\ (x_{LR,i} = (\tilde{x}, \kappa_\ell)), \kappa_\ell < \kappa$;
(3) $(\exists \sigma \in \Sigma)(f(x_v, \sigma) = y_v \in cfc)$.

O algoritmo 1, a seguir, sintetiza os passos para a obtenção do verificador G_v e para a verificação da codiagnosticabilidade-κ de acordo com o teorema 1.

Algoritmo 1. Verificação da codiagnosticabilidade-κ

- *Parâmetros de entrada*: $G = (X, \Sigma, f, \Gamma, x_0, X_m)$, $\Sigma_{o,i}$, $\forall i \in \{1, 2, \ldots, m\}$, $\Sigma_f = \{\sigma_f\}$ e $\kappa \in Z_+^*$.
- *Parâmetros de saída*: Verificador G_v, e decisão sobre codiagnosticabilidade-κ (Sim/Não).

1. Construa o autômato contador $\mathcal{A}_\kappa = (X_{A_\kappa}, \Sigma_f, \delta_k, x_0, X_m = \emptyset)$, de acordo com a equação (1).
2. Calcule o autômato $G_\kappa = G \parallel \mathcal{A}_k$.
3. Calcule os autômatos $G_{LR,i}, i = 1, 2, \ldots, m$ da seguinte forma:
   - Marque todos os estados $x_\kappa = (x, \kappa_\ell) \in X_\kappa$ de G_κ que satisfaçam a seguinte condição: $\kappa_\ell < \kappa$;
   - Calcule $G_L = Coac(G_\kappa)$;
   - Desmarque os estados marcados, encontrando $G_L = (X_L, \Sigma, f_L, \Gamma_L, x_{0,\kappa})$;
   - Para cada $i \in \{1, 2, \ldots, m\}$, $\forall \sigma \in \Sigma$, aplique o operador $R_i(\sigma)$ conforme a equação 3, e construa os autômatos $G_{LR,i} = (X_L, \Sigma_{LR,i}, f_{LR,i}, \Gamma_{LR,i}, x_{0,\kappa})$ de acordo com a equação (4).
4. Calcule o autômato $G_E = (X_E, \Sigma, f_E, \Gamma_E, x_{0,\kappa})$ da seguinte forma:
   - Marque todos os estados $x_\kappa = (x, \kappa_i) \in X_\kappa$ de G_κ que satisfaçam a seguinte condição: $\kappa_i = \kappa$;
   - Calcule $G_E = Coac(G_\kappa)$;
   - Desmarque os estados marcados.

5. Calcule o verificador $G_v = (||_{i=1}^{m} G_{LR,i}) \,||\, G_E = (X_v, \Sigma_v, f_v, \Gamma_v, x_{0,v})$.
6. Calcule o conjunto CFC formado por todos os componentes fortemente conectadas de G_v.
7. Faça codiagnosticabilidade-κ = Sim.
8. Para cada $cfc \in CFC$:
   Se $(\exists x_v = (x_{LR,1}, x_{LR,2}, \ldots, x_{LR,m}, x_E) \in cfc)$ $(\forall i \in \{1, 2, \ldots, m\}[(x_E = (x, \kappa)) \wedge (\exists (y_\kappa = (\tilde{x}, \kappa_\ell) \in x_{LR,i}, \kappa_\ell < \kappa)]) \wedge (\exists \sigma \in \Sigma, f(x_v, \sigma) = y_v \in cfc)$
   Então
      Faça codiagnosticabilidade-κ = Não
   Retorne

Exemplo 1. Considere uma estrutura descentralizada contendo dois diagnosticadores locais que se comunicam com um coordenador e seja a planta $G = (X, \Sigma, f, \Gamma, x_0, X_m)$, exibida na figura 2, em que $\Sigma = \{a, b, c, \sigma_f\}$, $\Sigma_{o,1} = \{a, c\}$, $\Sigma_{o,2} = \{b, c\}$ e $\Sigma_f = \{\sigma_f\}$. Suponha que desejamos verificar se a linguagem gerada por G é codiagnosticável-2, i.e., $\kappa = 2$.

De acordo com o algoritmo 1, o passo 1 é calcular o autômato contador $\mathcal{A}_\kappa$ para $\kappa = 2$, que é o mesmo exibido na figura 1. No passo 2, o autômato G_κ é calculado, que é mostrado na figura 3. No passo 3, os autômatos $G_{LR,1}$ e $G_{LR,2}$ são construídos primeiramente com a construção de G_L, em que apenas os estados rotulados com contadores iguais a $\kappa = 2$ são eliminados, bem como suas transições associadas. Após isso, os eventos não observáveis para cada diagnosticador são renomeados e, como resultado, os autômatos $G_{LR,1}$ e $G_{LR,2}$ são obtidos, conforme mostrados na figura 4. No passo 4, o autômato G_E é construído; porém neste caso particular, ele é idêntico a G_κ (figura 3), pois não há a remoção de nenhum estado ou transição.

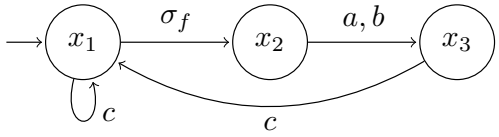

Figura 2. Planta G para o Exemplo 1.

O verificador G_v é calculado por meio da composição paralela $G_{LR,1}||G_{LR,2}||G_E$ (passo 5). O autômato verificador G_v possui 56 estados. Contudo, o interesse aqui é ilustrar a perda da codiagnosticabilidade-κ, portanto somente a parte de G_v que satisfaz as condições para violação da codiagnosticabilidade está exibida na figura 5. No passo 6, o conjunto contento todos as CFC não triviais em G_v CFC é obtido e no passo 8, é realizada uma busca em

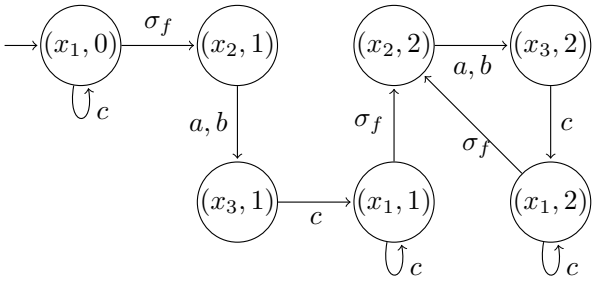

Figura 3. Autômato rotulado $G_\kappa = G||\mathcal{A}_\kappa$ no exemplo 1.

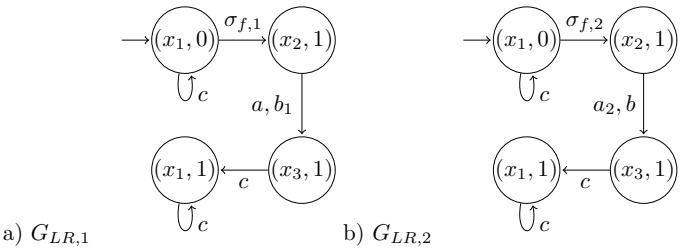

Figura 4. Autômatos $G_{LR,1}$ (a) e $G_{LR,2}$ (b) no exemplo 1.

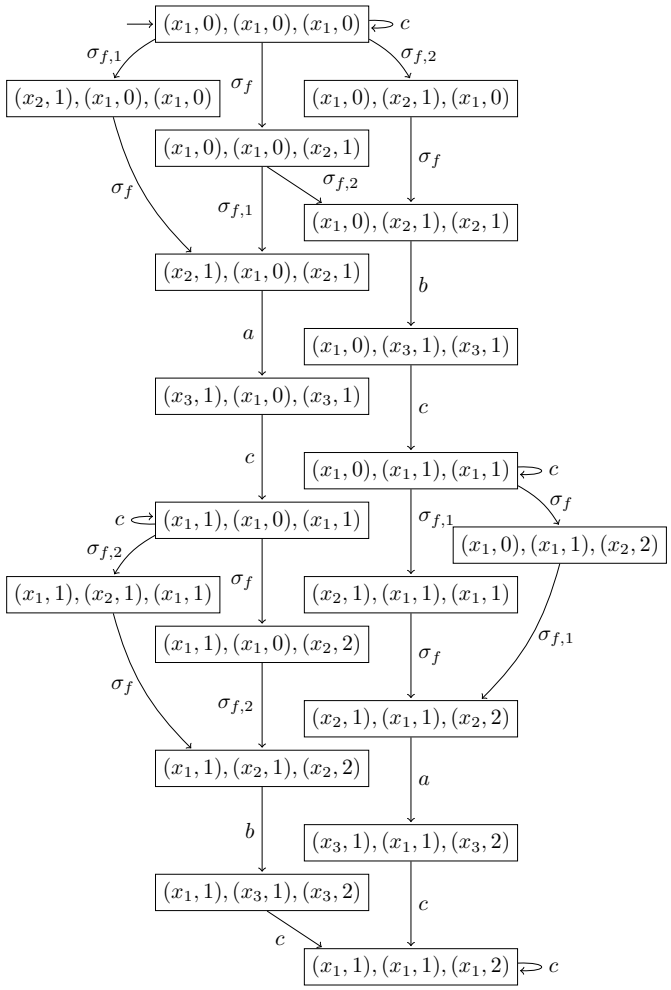

Figura 5. Parte do verificador G_v no exemplo 1 que leva à violação da codiagnosticabilidade-κ.

todas as $cfc \in CFC$ com o objetivo de verificar se há algum estado componente de alguma CFC do verificador que atende às condições (1)–(3) do teorema 1. De fato, é possível verificar que $cfc = \{(x_1, 1), (x_1, 1), (x_1, 2)\}$ é uma CFC não trivial formada pelo estado $(x_1, 1), (x_1, 1), (x_1, 2)$ que satisfaz as condições (1)–(3) do teorema 1. Portanto, a linguagem $\mathcal{L}(G)$ não é codiagnosticável-κ para $\kappa = 2$ com relação às projeções $P_{o,1}$ e $P_{o,2}$ e ao conjunto de eventos de falha Σ_f.

Além disso, é possível identificar as sequências que conduzem à decisão negativa sobre a codiagnosticabilidade-2 por não permitirem que nenhum diagnosticador local tenha certeza de, pelo menos, duas ocorrências da falha: $s_\kappa = c^x \sigma_f bcc^y \sigma_f acc^z$, cujas projeções são $P_{o,1}(s_\kappa) =$

Tabela 1. Complexidade computacional do algoritmo 1.

Planta	Núm. de estados	Núm. de transições
G	$\|X\|$	$\|X\| \times \|\Sigma\|$
$\mathcal{A}_\kappa$	$\kappa+1$	$(\kappa+1)\|\Sigma_f\|$
G_κ	$(\kappa+1)\|X\|$	$(\kappa+1)(\|X\| \times \|\Sigma\|)$
G_E	$(\kappa+1)\|X\|$	$(\kappa+1)(\|X\| \times \|\Sigma\|)$
G_L	$\kappa\|X\|$	$\kappa\|X\|\times(\|\Sigma\| - \|\Sigma_f\|)$
$G_{LR,i}$	$\kappa\|X\|$	$\kappa\|X\|\times(\|\Sigma\| - \|\Sigma_f\|)$
$G_{co,v}$	$(\kappa+1)\kappa^m\|X\|^{m+1}$	$(\kappa+1)\kappa^m\|X\|^{m+1} \times (\|\Sigma\| + m(k\|\Sigma\| - \|\Sigma_f\|))$

$c^{x+y+1}ac^{z+1}$, $P_{o,2}(s_\kappa) = c^xbcc^ycc^z = c^xbc^{y+z+1}$. Seja a sequência $s_{\tilde{\kappa}_1} = c^xcc^y\sigma_f acc^z$, cuja projeção é $P_{o,1}(s_3) = c^{x+y+1}ac^{z+1}$, e portanto, $P_{o,1}(s_1) = P_{o,1}(s_3)$. Seja a sequência $s_{\tilde{\kappa}_2} = c^x\sigma_f bcc^ycc^z$, cuja projeção é $P_{o,2}(s_4) = c^xbcc^ycc^z = c^xbc^{y+z+1}$, e portanto, $P_{o,2}(s_1) = P_{o,2}(s_4)$. Verifica-se que a possibilidade de ocorrências alternadas dos eventos a e b, os quais são observáveis por um dos diagnosticadores e não observáveis pelo outro diagnosticador, impede a codiagnosticabilidade-κ após a ocorrência de dois eventos de falha.

4. ANÁLISE DA COMPLEXIDADE COMPUTACIONAL

A tabela 1 mostra as cardinalidades máximas dos conjuntos de estados e transições para os autômatos envolvidos na construção do verificador G_v. Note que G_v possui, no pior caso, $(\kappa+1)\kappa^m|X|^{m+1}$ estados e $(\kappa+1)\kappa^m|X|^{m+1} \times (|\Sigma| + m(k|\Sigma| - |\Sigma_f|))$ transições. Note que fazendo $k = 1$ (diagnosticabilidade simples de falha), os resultados são: $2|X|^{m+1}$ e $2|X|^{m+1} \times (|\Sigma| + m(|\Sigma| - |\Sigma_f|))$, respectivamente, idênticos aos obtidos em Moreira et al. (2011). Portanto, o método proposto generaliza, para o verificador proposto por Moreira et al. (2011) para falhas repetidas.

5. CONCLUSÃO

Neste artigo, a noção de codiagnosticabilidade-κ foi introduzida por meio da utilização da diagnosticabilidade de falhas repetidas em uma arquitetura que compreende múltiplos diagnosticadores locais. Um algoritmo de complexidade polinomial foi proposto para sua verificação. Além disso, um teorema associado a esta noção foi apresentado juntamente com um exemplo ilustrativo, esclarecendo os passos do algoritmo proposto.

Com relação às possibilidades de pesquisas futuras, é possível desenvolver algoritmos baseados em diagnosticadores para codiagnosticabilidade-κ. Outra possibilidade seria expandir a estrutura descentralizada utilizada aqui para diagnose modular, adaptando o algoritmo para tal configuração.

REFERÊNCIAS

Basilio, J.C., Hadjicostis, C.N., e Su, R. (2021). Analysis and control for resilience of discrete event systems: Fault diagnosis, opacity and cyber security. *Foundations and Trends® in Systems and Control*, 8(4), 285–443.

Boussif, A., Ghazel, M., e Basilio, J.C. (2021). Intermittent fault diagnosability of discrete event systems: an overview of automaton-based approaches. *Discrete Event Dynamic Systems*, 31, 59–102.

Cassandras, C.G. e Lafortune, S. (2008). *Introduction to Discrete Events Systems*. Springer, New York, NY: USA, 2nd edition.

Cormen, T.H., Leiserson, C.E., Rivest, R.L., e Stein, C. (2007). *Introduction to Algorithms*. MIT Press, Cambridge, MA.

Debouk, R., Lafortune, S., e Teneketzis, D. (2000). Coordinated decentralized protocols for failure diagnosis of discrete event systems. *Discrete Event Dynamic Systems: Theory and Applications*, 10, 33–86.

Hopcroft, J.E., Motwani, R., e Ullman, J.D. (2006). *Introduction to Automata Theory, Languages, and Computation*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA: USA.

Jeron, T., Marchand, H., Pinchinat, S., e Cordier, M.O. (2006). Supervision patterns in discrete event systems diagnosis. *8th International Workshop on Discrete Event Systems*, 262–268.

Jiang, S., R, K., e Garcia, H.E. (2003). Diagnosis of repeated /intermittent failures in discrete-event systems. *IEEE Transactions on Robotics and Automaton*, 19, 310–323.

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*, 56, 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), 384–395.

Ramadge, P.J. e Wonham, W.M. (1989). The control of discrete event systems. *Proceedings of the IEEE*, 77(1), 81–98.

Sampath, M., Sengupta, R., e Lafortune, S. (1995). Diagnosability of discrete-event systems. *IEEE Transactions on Robotics and Automaton*, 40, 1555–1575.

Santoro, L.P., Moreira, M.V., e Basilio, J.C. (2017). Computation of minimal diagnosis bases of discrete-event systems using verifiers. *Automatica*, 77, 93–102.

Viana, G.S. e Basilio, J.C. (2019). Codiagnosability of discrete event system revisited: A new necessary and sufficient condition and its applications. *Automatica*, 101, 354–364.

Yoo, T.S. e Garcia, H.E. (2009). Event counting of partially-observed discrete-event systems with uniformly and nonuniformly bounded diagnosis delays. *Discrete Event Dynamic Systems*, 19, 167–187.

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

Zhou, C. e Kumar, R. (2009). Computation of diagnosable fault-occurrence indices for systems with repeatable faults. *IEEE Trans Autom Control*, 54, 1477–1489.