Diagnosticabilidade de falhas repetidas de sistemas a eventos discretos: novos algoritmos para verificação da diagnosticabilidade-κ ⋆

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

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

Abstract: Studies carried out in real plants show that repeated and/or intermittent failures frequently occur during the operation of a system. When it comes to diagnosing the occurrence of repeated faults, the problem becomes one of determining the number of occurrences of the fault, i.e., for a given $\kappa \in \mathbb{Z}_+^*$, ensure from observation of events that the failure has occurred at least κ times. In this work, the problem of diagnosability of repeated failures will be revisited, and algorithms for diagnosability checking-κ based both in the construction of test automata and in the construction of verifiers will be proposed and, based on these automata, necessary and sufficient conditions for the κ-diagnosability will be presented.

Resumo: Estudos realizados em plantas reais mostram que as falhas repetidas e/ou intermitentes ocorrem com frequência durante a operação de um sistema. Quando se trata de diagnosticar a ocorrência de falhas repetidas, o problema passa a ser o de determinar o número de ocorrências da falha, i.e., para um dado $\kappa \in \mathbb{Z}_+^*$, assegurar, a partir da observação de eventos, que a falha ocorreu, ao menos κ vezes. Neste artigo, o problema da diagnosticabilidade de falhas repetidas será revisitado, e algoritmos para a verificação da diagnosticabilidade-κ baseados tanto na construção de autômato de teste quanto na construção de verificadores serão propostos e, com base nesses autômatos, condições necessárias e suficientes para a diagnosticabilidade-κ serão apresentadas.

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

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

1. INTRODUÇÃO

Em sistemas a eventos discretos (SEDs), a diagnose de falhas desempenha um papel importante no funcionamento seguro e adequado de sistemas industriais, uma vez que falhas podem afetar equipamentos que variam desde pequenos componentes a sistemas complexos. Uma falha representa qualquer desvio do comportamento normal ou esperado de um sistema e, nesse contexto, introduz-se a noção de diagnosticabilidade de falhas, i.e., se é possível assegurar que uma falha ocorreu dada uma sequência observável de eventos.

O problema da diagnose/diagnosticabilidade tem recebido bastante atenção na literatura, sendo introduzido primeiramente em Sampath et al. (1995). Esse conceito foi estendido para codiagnosticabilidade em Debouk et al. (2000) e para diagnosticabilidade modular em Garcia et al. (2005) e Contant et al. (2006). Algoritmos para verificar a diagnosticabilidade de falhas foram propostos por Jiang et al. (2001), Yoo e Lafortune (2002), Qiu e Kumar (2006) Moreira et al. (2011) e Viana e Basilio (2019).

Falhas podem ocorrer mais de uma vez durante a operação de um sistema (Boussif et al., 2021), caracterizando assim, o conceito de *falhas repetidas*. Além disso, tal processo pode ser periódico ou não, fazendo com que o sistema alterne entre o comportamento normal e o de falha. Esse padrão é chamado de *falha intermitente*. Assim, torna-se necessário estender o conceito de diagnosticabilidade de falha para falhas repetidas e para falhas intermitentes.

No contexto de falhas repetidas e intermitentes, destaca-se Jiang et al. (2003), que elaborou uma proposta de modelo para a diagnosticabilidade de eventos de falha repetidos. Em Yoo e Garcia (2009) e Zhou e Kumar (2009), foram desenvolvidos novos algoritmos com o objetivo de verificar diversas noções de falhas intermitentes. Mais recentemente, Boussif et al. (2021) apresenta um tutorial/survey onde revisita as noções de falhas intermitentes existentes na literatura e os correspondentes métodos de verificação para SEDs modelados por autômatos.

Neste artigo, revisitaremos o problema da diagnosticabilidade falhas repetidas. Em Jeron et al. (2006), tal problema é tratado como um tipo de padrão de supervisão aplicado

⋆ Este trabalho foi parcialmente financiado pelo Conselho Nacional de Desenvolvimento Científico e Tecnológico, CNPq, processo número 316881/2021-0, e pela Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES), Código de Financiamento 001.

a sistemas de transição, e apresenta uma sugestão de autômato contador de falhas. Em Liang et al. (2022), são empregados padrões semelhantes, porém utiliza verificadores para a diagnosticabilidade. Algoritmos para a verificação da diagnosticabilidade-κ baseados tanto na construção de autômato de teste (Viana e Basilio, 2019) quanto na construção de verificadores (Moreira et al., 2011) serão propostos e, com base nesses autômatos, condições necessárias e suficientes para da diagnosticabilidade-κ serão apresentadas. A proposição de um método de verificação baseado no autômato teste proposto em Viana e Basilio (2019) é justificada em Clavijo e Basilio (2017), que elabora uma comparação entre as complexidades computacionais na construção de diagnosticadores e verificadores. Será também mostrado que a complexidade do algoritmo aqui proposto baseado na construção de verificadores possui melhor complexidade computacional que os existentes na literatura.

Este artigo está organizado da seguinte forma: na seção 2, a teoria básica sobre sistemas a eventos discretos é apresentada, incluindo as principais definições relevantes ao trabalho; na seção 3, está apresentado o principal objetivo deste artigo: novos algoritmos são detalhados, bem como seus teoremas relacionados, suas provas e exemplos ilustrativos; na seção 4, é exposta uma discussão sobre a complexidade computacional dos algoritmos aqui propostos, bem como comparações com algoritmos já publicados; finalmente, na seção 5, apresenta-se a conclusão deste artigo.

2. FUNDAMENTOS TEÓRICOS

O formalismo adotado neste artigo para a modelagem de SEDs é o autômato finito (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 conjuntos de eventos ativos, definida como $\Gamma(x) = \{\sigma \in \Sigma : f(x,\sigma)!\}$, em que $f(x,\sigma)!$ indica que $f(x,\sigma)$ é definida, i.e., existe $y \in X$ tal que $f(x,\sigma) = y$, $x_0 \in X$ o estado inicial e $X_m \subseteq X$ o conjunto de estado marcados. A função de transição de estados pode ser estendida para $f : X \times \Sigma^* \to X$, em que Σ^* representa o fecho de Kleene de Σ, por meio da seguinte recursão: $f(x,\varepsilon) = x$; e $f(x, s\sigma) = f(f(x,s),\sigma)$, $\sigma \in \Sigma, s \in \Sigma^*$. A linguagem gerada por G é definida como $\mathcal{L}(G) = \{s \in \Sigma^* : f(x_0, s)!\}$, sendo denotada por L. O conjunto de eventos Σ é aqui particionado em $\Sigma = \Sigma_o \dot{\cup} \Sigma_{uo}$, em que Σ_o e Σ_{uo} são os conjuntos de eventos observáveis e não-observáveis, respectivamente.

No decorrer do texto, o *fecho de prefixo* de uma sequência fica representado por $Pre(s)$, sendo definido por $Pre(s) = \{u \in \Sigma^* : (\exists v \in \Sigma^*)[uv = s]\}$. O *pós-linguagem* de L após uma sequência s é definida por $L/s = \{t \in \Sigma^* : st \in L\}$. Dados dois conjuntos de eventos Σ_a e Σ_b ($\Sigma_b \subset \Sigma_a$), a *projeção natural* $P_{a,b} : \Sigma_a^* \to \Sigma_b^*$, ou simplesmente projeção, é definida como (Ramadge e Wonham, 1989): $P_{a,b}(\sigma) = \sigma$, se $\sigma \in \Sigma_b$, ou $P_{a,b}(\sigma) = \varepsilon$, se $\sigma \in \Sigma_a \setminus \Sigma_b$; $P_{a,b}(\varepsilon) = \varepsilon$; $P_{a,b}(s\sigma) = P_{a,b}(s)P_{a,b}(\sigma)$, $s \in \Sigma_a^*$, $\sigma \in \Sigma_a$ e ε é a sequência vazia. A *projeção inversa* é definida como $P_{a,b}^{-1} : \Sigma_b^* \to 2^{\Sigma_a^*}$, sendo $P_{a,b}^{-1}(t) = \{s \in \Sigma_a^* : P_{a,b}(s) = t\}$ e $\Sigma_b \subset \Sigma_a$. Os conceitos de *fecho de prefixo*, *projeção* e *projeção inversa* podem ser estendidos sobre uma linguagem L, aplicando-os a cada sequência de L.

A operação que calcula a *parte acessível* de um autômato G exclui todos os estados que não são alcançáveis por alguma sequência que parte do estado inicial, sendo definida como: $Ac(G) = (X_{Ac}, \Sigma, f_{Ac}, \Gamma_{Ac}, x_0, X_{Ac,m})$, sendo $X_{Ac} = \{x \in X : (\exists s \in \Sigma^*)[f(x_0, s) = x]\}$, $X_{Ac,m} = X_m \cap X_{Ac}$ e $f_{Ac} : X_{Ac} \times \Sigma^* \to X_{Ac}$ 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. A operação que calcula a *parte coacessível* exclui todos os estados que não fazem parte de um caminho que parte deste estado e 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 indefinida, caso contrário, e $f_{Coac} : X_{Coac} \times \Sigma^* \to X_{Coac}$, sendo $f_{Coac}(x,\sigma) = f(x,\sigma)$, se $f(x,\sigma) \in X_{Coac}$, ou $f(x,\sigma)$ indefinida, caso contrário.

A operação de *composição paralela* entre autômatos cria um novo autômato no qual os comportamentos em comum dos autômatos originais são sincronizados e suas características individuais fluem de modo independente, sendo formalmente definida por: $G_1||G_2 = Ac(X_1 \times X_2, \Sigma_1 \cup \Sigma_2, f_{1||2}, \Gamma_{1||2}, (x_{0,1}, x_{0,2}), X_{m,1} \times X_{m,2})$, em que: $f_{1||2}((x_1,x_2),\sigma) = (f_1(x_1,\sigma), x_2)$, 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 indefinida, caso contrário.

Com o objetivo de definir apropriadamente o *observador* de um autômato G: $Obs(G, \Sigma_o)$, torna-se necessário definir, inicialmente, o *alcance não observável de um estado* x, *i.e.*, $UR(x, \Sigma_{uo}) = \{x' \in X : (\exists s \in \Sigma_{uo}^*)[f(x, s) = x']\}$. O alcance não observável de um estado pode ser estendido a um conjunto de estados $Y \in 2^X$ da seguinte forma: $UR(Y, \Sigma_{uo}) = \cup_{x \in Y} UR(x, \Sigma_{uo})$. A partir dessa definição, o *observador* de G é definido como $G_{obs} = Obs(G, \Sigma_o) = (X_{obs}, \Sigma_o, f_{obs}, \Gamma_{obs}, x_{0,obs}, X_{m,obs})$, em que $X_{obs} \in 2^X$, $f_{obs}(x_{obs}, \sigma) = \cup_{x \in x_{obs} \wedge f(x,\sigma)!} UR(f(x, \sigma), \Sigma_{uo})$, $\Gamma_{obs}(x_{obs}) = \cup_{x \in x_{obs}} \Gamma(x) \cap \Sigma_o$, $x_{0,obs} = UR(x_0, \Sigma_{uo})$ e $X_{m,obs} = \{x_{obs} \in X_{obs} : (\exists x \in x_{obs})[x \in X_m]\}$. Finalmente, $\mathcal{L}(G_{obs}) = P_o(\mathcal{L}(G))$, sendo denotado por L_{obs}, em que $P_o : \Sigma^* \to \Sigma_o^*$.

Dado um grafo orientado $D = (V, E)$, em que V e E denotam os conjuntos de vértices e arestas de D, respectivamente, dizemos que um conjunto de vértices $V_{scc} \subseteq V$ é uma componente fortemente conexa (SCC) de D se todos os pares de vértices u e v em V_{scc} são alcançáveis entre si, *i.e.*, se $\forall u, v \in V_{scc}, (u \rightsquigarrow v) \wedge (v \rightsquigarrow u)$, e V_{scc} é maximal, no sentido de que não há outros vértices que possam ser incluídos em V_{scc}, e que satisfaçam a condição de alcançabilidade descrita acima. Segundo Cormen et al. (2007), o cálculo de SCCs é, no pior caso, linear com relação ao número de arestas e vértices de D, *i.e.*, $O(V + E)$. Observa-se que é possível que V_{scc} tenha um único elemento, *i.e.*, $V_{scc} = \{u\}$. Nesse caso, existem duas possibilidades: *(i)* $u \rightsquigarrow u$ por meio de uma aresta (autolaço); *(ii)* não existe um autolaço em u. Para distinguir estes dois tipos de SCCs que contêm apenas um elemento, as SCCs que satisfazem *(i)* serão denominadas, "SCCs não triviais".

2.1 Diagnosticabilidade de falhas intermitentes

Um *conjunto de eventos de falha* $\Sigma_f \subseteq \Sigma$, é um conjunto formados por todos os *eventos de falha*. Claramente, os eventos de falha são não observáveis, i.e., $\Sigma_f \subseteq \Sigma_{uo}$. Adicionalmente, sem perda de generalidade (Santoro et al., 2017), vamos considerar que a $|\Sigma_f| = 1$, i.e. $\Sigma_f = \{\sigma_f\}$. Caso exista mais de um tipo de evento de falha, cada evento de falha deverá ser tratado separadamente, considerando os demais eventos de falha como eventos não observáveis ordinários.

A primeira definição de diagnosticabilidade aplicada a SEDs foi introduzida por Sampath et al. (1995), segundo a qual, para que uma linguagem gerada por um autômato seja diagnosticável, é necessário que a ocorrência do evento falha seja diagnosticada após a ocorrência de um número finito de eventos após a falha.

Para se definir formalmente a diagnosticabilidade de uma linguagem, é necessário acrescentar aqui a definição do conjunto formado por todas as sequências de L que terminam com o evento de falha: $\Psi(\sigma_f) = \{s \in L : \exists u \in (\Sigma \setminus \Sigma_f)^*, s = u\sigma_f\}$. Além disso, suponha que $|s|$ denote o comprimento da sequência s, *i.e.*, o número de eventos de s.

Definição 1. (Diagnosticabilidade). Um linguagem L é diagnosticável com relação a $\Sigma_f = \{\sigma_f\}$ e $P_o : \Sigma^* \to \Sigma_o$ se a seguinte condição for verificada:

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

sendo a condição de diagnosticabilidade D expressa por:

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

A definição 1, que corresponde à mesma contida em Sampath et al. (1995), não leva em conta repetições da ocorrência de eventos de falha. Quando se desejar que múltiplas ocorrências de falhas sejam detectadas, torna-se necessário utilizar as definições de falhas repetidas propostas em Jiang et al. (2003) e revisitadas em Boussif et al. (2021). Nesse contexto, dada uma sequência $s \in \Sigma^*$, iremos denotar por N_s^F o número de ocorrências de eventos de falha σ_f em s.

Definição 2. (Diagnosticabilidade-κ) Dado um número fixo $\kappa \in \mathbb{Z}_+^*$, diz-se que uma linguagem viva e prefixo-fechada L é diagnosticável-κ com respeito a $P_o : \Sigma^* \to \Sigma_o^*$ e Σ_f se a seguinte proposição for verdadeira:

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

sendo a condição de diagnosticabilidade-κ D_κ expressa por:

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

De acordo com a definição 2, uma linguagem L não será diagnosticável-κ se existir uma sequência de eventos s que contenha, ao menos, κ eventos de falha e uma continuação ilimitada t de s em L, e uma sequência $\omega \in L$ indistinguível de st, i.e., $P_o(st) = P_o(\omega)$, tal que ω tenha um número menor que κ de ocorrências de eventos de falha. É importante ressaltar que, quando $\kappa = 1$, a definição de diagnosticabilidade-κ se reduz à definição 1, introduzida em Sampath et al. (1995).

3. ALGORITMOS PARA A VERIFICAÇÃO DA DIAGNOSTICABILIDADE-κ

Neste artigo, iremos propor dois algoritmos distintos para a verificação da diagnosticabilidade-κ: o primeiro utiliza um autômato teste com base no proposto por Viana e Basilio (2019); e o segundo utiliza um verificador similar ao proposto por Moreira et al. (2011). Em seguida, com base nesses autômatos, iremos apresentar condições necessárias e suficientes para a diagnosticabilidade-κ de linguagens regulares.

Seja $G = (X, \Sigma, f, \Gamma, x_0, X_m)$ o autômato que modela o sistema e suponha que o conjunto de eventos Σ está particionado em $\Sigma = \Sigma_o \dot{\cup} \Sigma_{uo}$ e que o evento de falha σ_f pode ocorrer repetidas vezes. Como autômato contador, iremos utilizar a estrutura proposta por Jeron et al. (2006), que é dada por:

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

em que $X_\kappa = \{0, 1, ..., \kappa\}$, $x_0 = 0$, $f_\kappa(i, \sigma_f) = i + 1, i = 0, \ldots, \kappa - 1$ e $f_\kappa(\kappa, \sigma_f) = \kappa$. Como exemplo ilustrativo, a figura 1 mostra o autômato contador para $\kappa = 2$.

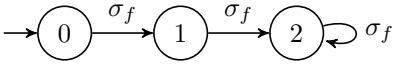

Figura 1. Autômato contador $\mathcal{A}_\kappa$ para $\kappa = 2$.

3.1 Algoritmo baseado no autômato teste de Viana e Basilio (2019)

O primeiro algoritmo proposto neste artigo baseia-se na construção de um autômato teste G_{test} proposto em Viana e Basilio (2019), sendo a verificação da diagnosticabilidade-κ realizada de acordo com o algoritmo 1.

Algoritmo 1. Verificação da diagnosticabilidade-κ utilizando autômato teste

- *Parâmetros de entrada*: $G = (X, \Sigma, f, \Gamma, x_0, X_m)$, Σ_o, $\Sigma_f = \{\sigma_f\}$ e $\kappa \in Z_+^*$.
- *Parâmetros de saída*: Autômato teste G_{test}, diagnosticabilidade-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 (1).
2. Calcule $G_\kappa = G || \mathcal{A}_k$.
3. Calcule $G_{\kappa,obs} = Obs(G_\kappa, \Sigma_o)$.
4. Calcule $G_{test} = G_{\kappa,obs} || G_\kappa$.
5. Calcule o conjunto SCC formado por todas as componentes fortemente conexas (SCC) não triviais em G_{test}.
6. Faça diagnosticabilidade-κ = Sim.
7. Enquanto (diagnosticabilidade-κ = Sim):
   Para cada $scc \in SCC$:
   Se $\exists x = (x_{\kappa,obs}, x_\kappa) \in SCC, (x_\kappa = (x, \kappa)) \wedge (\exists y = (x, \tilde{\kappa}) \in x_{\kappa,obs}, \tilde{\kappa} < \kappa)$, então:
   Faça diagnosticabilidade-κ = Não.

De acordo com o algoritmo 1, a verificação da diagnosticabilidade-κ é baseada na busca por SCC em G_{test} formadas por estados $x = (x_{\kappa,obs}, x_\kappa)$ tais que a segunda componente possua contador igual a κ e o primeira componente possua

ao menos um estado cujo contador seja $\tilde{\kappa} < \kappa$. Caso exista uma SCC com tais parâmetros, então $L(G)$ não é diagnosticável-κ.

O algoritmo 1 funciona da seguinte forma. O autômato contador $\mathcal{A}_\kappa$ construído no Passo 1 tem a propriedade de contar os eventos de falha de 1 até κ. No Passo 2, calcula-se a composição paralela $G_\kappa = G||\mathcal{A}_k$. Note que os estados de G_κ têm o formato (x, x_κ), em que $x \in X$ e $x_\kappa \in X_{A_\kappa}$ (contagem de eventos de falha). No Passo 3, calcula-se o autômato observador $G_{\kappa,obs}$ com respeito ao conjunto de eventos observáveis Σ_o e, no Passo 4, o autômato teste G_{test} é calculado por meio da composição paralela $G_{\kappa,obs}||G_\kappa$. Observe que os estados de G_{test} têm o formato $(x_{\kappa,obs}, x_\kappa)$, em que $x_{\kappa,obs}$ é um estado do observador $G_{\kappa,obs}$, sendo formado por estados de G_κ, os quais têm o formato descrito no Passo 2. No Passo 5, após computar G_{test}, encontra-se o conjunto de suas componentes fortemente conexas não triviais SCC. No Passo 6, a decisão sobre a diagnosticabilidade-κ é estabelecida para "Sim" como um parâmetro inicial. Finalmente, no Passo 7, verifica-se se há algum estado contido em SCC que satisfaça as duas condições simultaneamente: (i) se o contador for igual a κ em seu estado local x_κ; (ii) e se algum de seus estados em seu componente observador $x_{\kappa,obs}$ possuir contador com menor valor que κ. Caso ambas as condições sejam satisfeitas simultaneamente, a linguagem L gerada por G não é diagnosticável-κ e a decisão sobre a diagnosticabilidade-κ é atualizada para "Não". Caso contrário, esta decisão permanece em "Sim" até que se encerrem os estados a serem testados. Ressalta-se que, caso haja, em algum ponto, uma mudança para diagnosticabilidade-κ = "Não", a busca imposta pelo Passo 7 será interrompida.

O seguinte teorema prova a correção do algoritmo 1.

Teorema 1. Seja L uma linguagem regular e seja $G = (X, \Sigma, f, \Gamma, x_0, X_m)$ um autômato tal que $L(G) = L$. Sejam ainda $\kappa \in \mathbb{Z}_+^*$ fixo, G_{test} o autômato teste construído de acordo com o algoritmo 1 e SCC o conjunto de componentes fortemente conexas não triviais de G_{test}. Então a linguagem L não será diagnosticável-κ com relação a $P_o : \Sigma^* \to \Sigma_o^*$, $\Sigma_f = \sigma_f$ e ao κ fixo se, e somente se, a seguinte condição for verificada:

$$(\exists scc \in SCC)(\exists x = (x_{\kappa,obs}, x_\kappa) \in scc),$$
$$(x_\kappa = (x, \kappa)) \wedge (\exists y = (x, \tilde{\kappa}) \in x_{\kappa,obs}, \tilde{\kappa} < \kappa).$$

O exemplo a seguir ilustra a aplicação do algoritmo 1 e do teorema 1.

Exemplo 1. Considere o autômato G cujo diagrama de transição de estados está representado na figura 2. Suponha que o objetivo seja determinar se $L(G)$ é diagnosticável-2. Para tanto, vamos utilizar o algoritmo 1 para $\kappa = 2$. Pelo Passo 1, constrói-se o autômato $\mathcal{A}_\kappa$, que é o mesmo da figura 1, e, em seguida, calcula-se $G_\kappa = G||\mathcal{A}_\kappa$, que está representado na figura 3 (Passo 2). Em seguida, calculando-se o autômato observador $G_{\kappa,obs} = Obs(G_\kappa, \Sigma_o)$ (Passo 3), é possível, então, obter o autômato teste $G_{test} = G_{\kappa,obs}||G_\kappa$ (Passo 4), mostrado na figura 4.

No Passo 5, obtém-se o conjunto de todas as SCC não triviais em G_{test}. Por meio de uma inspeção da figura 4, identifica-se três conjuntos diferentes de SCC não triviais (dois são constituídos de estados únicos com autolaços no

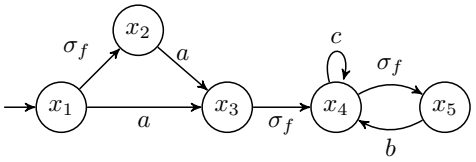

Figura 2. Modelo G para o Exemplo 1.

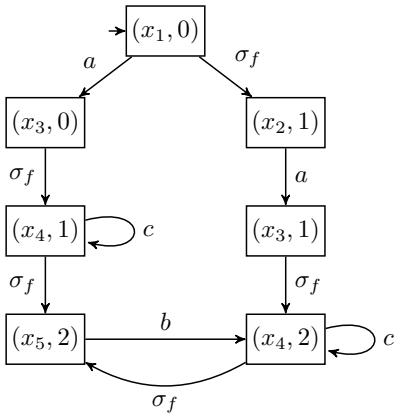

Figura 3. Composição paralela $G_\kappa = G||\mathcal{A}_\kappa$ para o Exemplo 1.

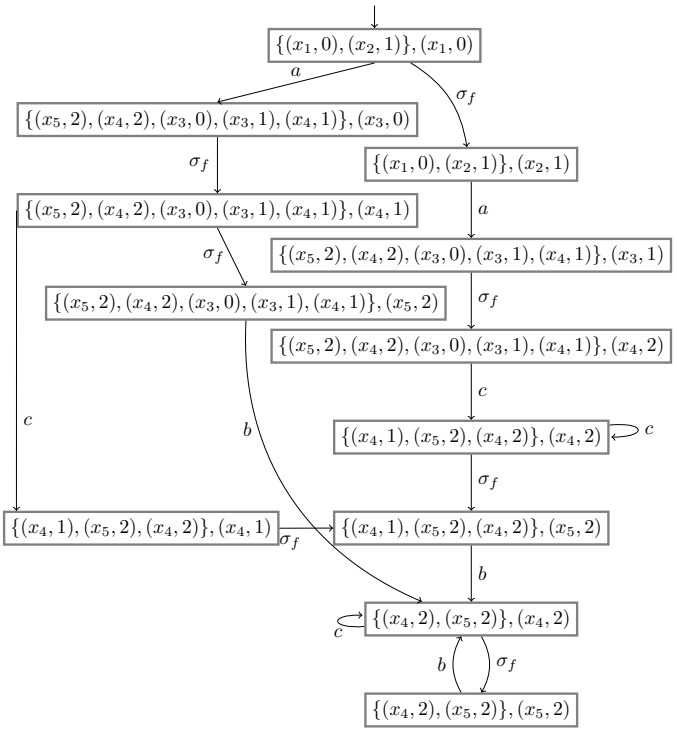

Figura 4. Autômato de teste G_{test} para o Exemplo 1.

evento c e apenas um é composto por dois estados), os quais são dados por: (1) $(\{(x_4,1),(x_5,2),(x_4,2)\}, (x_4,2))$; (2) $(\{(x_4,2),(x_5,2)\}, (x_4,2))$; e (3) $(\{(x_4,2),(x_5,2)\}, (x_4,2))$, $(\{(x_4,2),(x_5,2)\}, (x_5,2))$. De acordo com o Passo 7 do algoritmo, deve-se buscar estados contidos em SCCs cujos contadores são iguais a $\kappa = 2$, e, como resultado, constata-se que todos os estados contidos em SCC atendem tal condição. Esta análise, de acordo com os Passos 6 e 7, prossegue com a verificação da existência de algum estado inserido em seus componentes observadores

com contadores de valor inferior a $\kappa = 2$. É fácil constatar que há um estado que atende a essa condição: $(x_4, 1)$ no autolaço correspondente à primeira SCC (1). Portanto, a linguagem L gerada pelo modelo G não é diagnosticável-2.

3.2 Algoritmo baseado na construção do verificador de Moreira et al. (2011)

O segundo algoritmo proposto neste artigo baseia-se na construção de um autômato verificador G_V proposto em Moreira et al. (2011), sendo a verificação da diagnosticabilidade-κ realizada conforme o algoritmo 2.

Algoritmo 2. Verificação da diagnosticabilidade-κ utilizando verificadores

- *Parâmetros de entrada*: $G = (X, \Sigma, f, \Gamma, x_0, X_m)$, Σ_o, $\Sigma_f = \{\sigma_f\}$ e $\kappa \in Z_+^*$.
- *Parâmetros de saída*: Verificador G_V, e decisão sobre a diagnosticabilidade-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 (1).
2. Calcule $G_\kappa = G||\mathcal{A}_k$.
3. Calcule o autômato apropriado para contagens de valores inferiores a κ denominado G_L, marcando todos os estados nos quais seus contadores são iguais a $\kappa-1$, tomando a sua parte coacessível, e desmarcando os estados marcados.
4. Renomeie os eventos não observáveis em G_L, adicionando a todos eles um sufixo "r", e denominando o autômato resultante de G_{LR}.
5. Calcule G_E, marcando todos os estados nos quais seus contadores são iguais a κ, tomando a sua parte coacessível, e desmarcando os estados marcados.
6. Calcule $G_V = G_{LR}||G_E$.
7. Calcule o conjunto SCC formado pelas componentes fortemente conexas (SCC) não triviais de G_V.
8. Faça diagnosticabilidade-κ = Sim.
9. Enquanto (diagnosticabilidade-κ = Sim):
   Para cada $scc \in SCC$:
   Se $\exists x = (x_{\kappa,1}, x_{\kappa,2}) \in scc, [(x_{\kappa,2} = (x, \kappa)) \wedge (y \in x_{\kappa,1}) \wedge (\tilde{\kappa} < \kappa), y = (x, \tilde{\kappa})] \wedge \exists \tilde{\sigma} \in scc, \tilde{\sigma} \in \Sigma$ então:
   Faça diagnosticabilidade-κ = Não.

O algoritmo 2 funciona do seguinte modo. Os Passos 1 e 2 são similares aos do algoritmo 1, resultando no cálculo de G_κ. Após o Passo 2, os estados de G_κ têm o formato (x, x_κ), em que $x \in X$ e $x_\kappa \in X_{A_\kappa}$ (contagem de eventos de falha). No Passo 3, constrói-se o autômato G_L com o objetivo de eliminar os estados de G_κ cujos contadores são $x_\kappa = \kappa$. No Passo 4, todos os eventos não observáveis são renomeados com a adição do sufixo "r", gerando G_{LR}. No Passo 5, calcula-se G_E, marcando os estados cujos contadores são iguais a κ, tomando a sua parte coacessível, e desmarcando os estados marcados. No Passo 6, computa-se o verificador G_V por meio da composição paralela $G_{LR}||G_E$. Observa-se que os estados em G_V têm o seguinte formato: um estado oriundo de G_{LR}, cujos contadores têm valores inferiores a κ, e que têm o formato descrito no Passo 2 (primeira componente); e um estado oriundo de G_E, que mantém os estados com contadores iguais a κ, e que também têm o formato obtido após o Passo 2 (segunda componente). No Passo 7, após computar G_V, deve-se encontrar o conjunto das componentes fortemente conexas não triviais SCC existentes no verificador. No Passo 8, a decisão sobre a diagnosticabilidade-κ é estabelecida para "Sim" como parâmetro inicial. Finalmente, no Passo 9, faz-se necessário averiguar se há algum estado contido em SCC que satisfaça as seguintes condições: se o contador é igual a κ em sua segunda componente; se o contador possui valor menor que κ em sua primeira componente; e se há alguma transição $\sigma \in \Sigma$ interna à SCC em questão. Caso estas condições sejam satisfeitas simultaneamente, a linguagem L gerada por G não é diagnosticável-κ e a decisão sobre a diagnosticabilidade-κ é atualizada para "Não". Caso contrário, esta decisão permanece em "Sim" até que se encerrem os estados a serem testados. Observa-se que, caso haja, em algum ponto, uma mudança para diagnosticabilidade-κ = "Não", esta busca imposta pelo Passo 9 será interrompida.

O seguinte resultado demonstra a correção do algoritmo 2.

Teorema 2. Seja L uma linguagem regular e seja $G = (X, \Sigma, f, \Gamma, x_0, X_m)$ um autômato tal que $L(G) = L$. Sejam ainda $\kappa \in \mathbb{Z}_+^*$ fixo, G_V o verificador construído de acordo com o algoritmo 2 e SCC o conjunto de componentes fortemente conexas não triviais de G_V. Então a linguagem L não será diagnosticável-κ com relação a $P_o : \Sigma^* \to \Sigma_o^*$, $\Sigma_f = \sigma_f$ e ao κ fixo se, e somente se, a seguinte condição for verificada:

$$(\exists scc \in SCC)(\exists x = (x_{\kappa,1}, x_{\kappa,2}) \in scc),$$
$$(x_{\kappa,2} = (x, \kappa)) \wedge (x_{\kappa,1} = (x, \tilde{\kappa}))(\tilde{\kappa} < \kappa) \wedge (\exists \sigma \in scc, \sigma < \Sigma).$$

O exemplo a seguir ilustra a aplicação do algoritmo 2 e do teorema 2.

Exemplo 2. Considere o mesmo modelo G utilizado no Exemplo 1 (figura 2) e suponha que o valor de κ a ser aplicado seja também igual a 2 (diagnosticabilidade-2). Será aplicado o algoritmo 2 para tal verificação. Nos Passos 1 e 2, calculam-se o autômato contador (figura 1) e o autômato G_κ (figura 3), respectivamente. Nos Passos 3 e 4, constrói-se o autômato G_{LR}, eliminando os estados cujos contadores são iguais a κ e suas transições associadas, e renomeando os eventos não observáveis, sendo mostrado na figura 5. No Passo 5, calcula-se G_E, que neste caso é igual a G_κ.

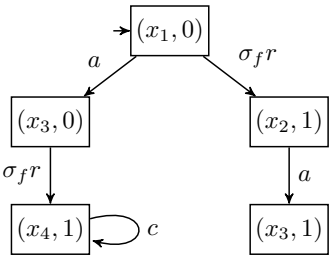

Figura 5. Autômato G_{LR} para o Exemplo 2.

O verificador G_V é obtido por meio da execução de $G_{LR}||G_E$ e é mostrado na figura 6 (Passo 6). No Passo 7, obtém-se o conjunto dos SCC não triviais de G_V. Nos Passos 8 e 9, são examinados todos os scc de G_V, obtendo dois conjuntos de autolaços: (1) $(x_4, 1), (x_4, 1)$ e (2) $(x_4, 1), (x_4, 2)$. O autolaço (1) não tem relação com as condições iniciais para a diagnosticabilidade-κ com

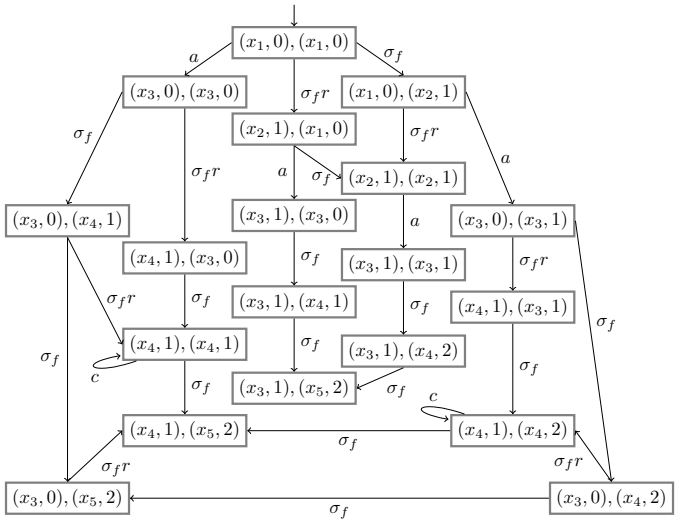

Figura 6. Autômato verificador G_V para o Exemplo 2.

$\kappa = 2$, porém o autolaço (2) viola tais condições: a segunda componente possui contador igual a κ enquanto a primeira componente possui contador de menor valor que κ. Além disso, a transição que ocorre neste autolaço é rotulada com $c \in \Sigma$. Portanto, a linguagem L gerada por G não é diagnosticável-2, obtendo os mesmos resultados encontrados no exemplo 1.

4. DISCUSSÃO

4.1 Análise da complexidade computacional

Nesta seção, elabora-se uma discussão sobre a complexidade computacional dos algoritmos propostos. O autômato a ser analisado nesta seção é o verificador G_V, gerado pelo Algoritmo 2. Iniciando pelo autômato contador $\mathcal{A}_\kappa$, conclui-se imediatamente que $|X_{\mathcal{A}_\kappa}| = \kappa + 1$. Logo, a partir da composição paralela $G_\kappa = G||\mathcal{A}_\kappa$, obtém-se $|X_{G_\kappa}| = |X|(\kappa + 1)$. Por fim, a partir do verificador G_V, obtém-se $|X_{G_V}| = (\kappa+1)^2|X|^2$ com crescimento da ordem de $\mathcal{O}(|X|^2)$.

De maneira similar, pode-se concluir que o número de eventos também cresce com a ordem $\mathcal{O}(|\Sigma|)$. Como consequência, considerando ambos os crescimentos dos números de estados e de transições em G_V, a complexidade computacional tem a ordem $\mathcal{O}(|X|^2 \times |\Sigma|^2)$.

Em Jiang et al. (2003) e em Boussif et al. (2021), um autômato verificador utilizado para checagem da diagnosticabilidade-κ foi proposto e sua complexidade computacional foi calculada como $\mathcal{O}(|X|^6)$ pela técnica do grafo de transição. De acordo com os resultados calculados aqui, pode-se concluir que a construção do verificador proposto neste trabalho pode ser realizada com menor custo computacional.

4.2 Outras discussões

Os algoritmos propostos neste trabalho podem ser considerados como uma generalização dos algoritmos anteriormente publicados em Moreira et al. (2011) e em Viana e Basilio (2019) no sentido de que, quando $\kappa = 1$, os algoritmos se reduzem aos anteriormente propostos. Em outras palavras, caso o objetivo do problema seja averiguar a ocorrência de ao menos 1 falha no modelo (diagnosticabilidade-1), a tarefa é reduzida à decisão sobre a simples diagnosticabilidade de falha introduzida em Sampath et al. (1995).

5. CONCLUSÃO

Neste trabalho, foram propostos novos algoritmos para a diagnosticabilidade-κ. Novos teoremas associados a tais algoritmos foram elaborados e suas respectivas provas foram desenvolvidas neste artigo. Finalmente, uma comparação em termos de complexidade computacional com outros modelos anteriormente propostos foi elaborada com o objetivo de comprovar a efetividade dos algoritmos aqui apresentados.

Em termos de possibilidades de trabalhos futuros, algoritmos similares aos propostos aqui podem ser adaptados ou modificados para a checagem de características de temas relacionados à codiagnosticabilidade e à diagnosticabilidade modular. Além disso, alguns procedimentos algorítmicos similares podem ser desenvolvidos com o objetivo de diagnosticar outras definições relacionadas ao tema da diagnose de falhas intermitentes citadas em Boussif et al. (2021) e testar suas respectivas complexidades computacionais.

REFERÊNCIAS

Basilio, J.C., Hadjicostis, C.N., Su, R., et al. (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.

Clavijo, L.B. e Basilio, J.C. (2017). Empirical studies in the size of diagnosers and verifiers for diagnosability analysis. *Discrete Event Dynamic Systems*, 27(4), 701–739.

Contant, O., Lafortune, S., e Teneketzis, D. (2006). Diagnosability of discrete event systems with modular structure. *Discrete Event Dynamic Systems*, 16(1), 9–37.

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.

Garcia, E., Correcher, A., Morant, F., Qulies, E., e Blasco, R. (2005). Modular fault diagnosis based on discrete-event systems. *Discrete Event Dynamic Systems*, 15(3), 237–256.

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.

Jiang, S., Huang, Z., Chandra, V., e Kumar, R. (2001). A polynomial algorithm for testing diagnosability of discrete-event systems. *IEEE Transactions on Automatic Control*, 46(8), 1318–1321.

Liang, Y., Lefebvre, D., e Li, Z. (2022). Fault pattern diagnosis of discrete-event systems by means of logical verifiers. *IFAC Papers Online*, 55, 551–556.

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.