Opacidade de sistemas a eventos discretos modelados por uma classe de autômatos temporizados ⋆

Mariana Guimarães Marques * **João Carlos Basilio** *

* *COPPE - Programa de Engenharia Elétrica, Universidade Federal do Rio de Janeiro, 21.949-900, Rio de Janeiro, RJ, Brasil (e-mails: mariana.marques@coppe.ufrj.br, basilio@dee.ufrj.br)*

Abstract: This paper addresses the problem of language opacity for a class of discrete-event systems whose transitions occur within some known time interval after the event becomes enabled. For this purpose, a new class of timed automata is proposed, the so-called time-interval automata, in which a time interval is associated with each event that labels a transition. For such automata, the operations of complement, projection and product are redefined without using refinement by partition, as usually used in the literature. The usual language-based definition of opacity is extended to this class of automata and an algorithm for its verification is proposed.

Resumo: Este artigo aborda o problema de opacidade da linguagem para uma classe de sistemas de eventos discretos cujas transições ocorrem dentro de um intervalo de tempo conhecido após o evento ser habilitado. Para tanto, é proposta uma nova classe de autômatos temporizados, os chamados autômatos com intervalos de tempo, nos quais um intervalo de tempo é associado a cada evento que rotula uma transição. Para tais autômatos, as operações de complemento, projeção e produto são redefinidas sem o uso do chamado refinamento por partição, que é usualmente utilizado na literatura. A definição usual de opacidade baseada em linguagem é estendida para esta classe de autômatos e um algoritmo para sua verificação é proposto.

Keywords: Discrete event systems, opacity, time interval automata

Palavras-chaves: Sistemas a eventos discretos, opacidade, autômatos com intervalos de tempo

1. INTRODUÇÃO

Um sistema a eventos discretos (SED) é dito opaco quando o seu comportamento secreto não é revelado a um observador externo, usualmente um intruso com intenções maliciosas. Presume-se que o intruso tenha conhecimento do funcionamento da planta e de seu comportamento. Por meio de estimativas com base na observação, o intruso tenta estimar o estado atual de funcionamento do sistema.

Diferentes noções de opacidade podem ser encontradas na literatura (Wu e Lafortune, 2013; Basilio et al., 2021), e.g., opacidade de estado atual, opacidade com base em linguagem, opacidade de estado inicial e opacidade de estados inicial-final, cujos algoritmos para verificação estão descritos em Wu e Lafortune (2013). As diferentes noções de opacidade representam um único comportamento secreto a ser escondido de intrusos e podem ser convertidas umas nas outras como descrito em Wu e Lafortune (2013).

Entretanto, existem sistemas que não são naturalmente opacos e, quando a opacidade é exigida, faz-se necessário construir um forçador de opacidade, o qual muda a saída observável do sistema de forma a impedir que o intruso estime o seu comportamento secreto. Podem-se encontrar na literatura diferentes formas de forçar a opacidade. Wu e Lafortune (2014) usam função de inserção, que insere observações falsas. Ji e Lafortune (2017) usam funções de edição, que alteram a saída do sistema a partir da inserção de eventos fictícios ou deletando-se observações. Barcelos e Basilio (2021) forçam a opacidade embaralhando e deletando observações de eventos.

Os artigos citados anteriormente consideram que a ocorrência de eventos seja assíncrona. Contudo, em muitos sistemas, os eventos têm tempos específicos para ocorrerem. Em Alves et al. (2020) e Viana et al. (2021) considera-se um tempo mínimo no qual cada transição pode ocorrer; contudo, não ná uma limitação para um tempo máximo para ocorrência dos eventos. Neste artigo será usada uma definição similar à proposta em Wang et al. (2018), onde há um relógio que é reinicializado a cada transição cujos tempos mínimo e máximo para a ocorrência dos eventos são definidos pelos limites inferior e superior de um intervalo real. Nesse contexto, as operações usuais envolvendo autômatos (Cassandras e Lafortune, 2008), serão redefinidas para essa nova classe de SEDs temporizados. A forma aqui proposta das operações envolvendo os autômatos temporizados é diferente daquela proposta em Wang et al. (2018).

Este artigo está estruturado da seguinte forma. Na seção 2 são revistos os conceitos fundamentais de SEDs necessários para tornar o artigo autocontido. Na seção 3 é apresentada, por meio de um exemplo, a motivação por trás do estudo

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

feito neste artigo. Na seção 4 é apresentada a definição da nova classe de autômatos temporizados e são propostos métodos para realizar em tais autômatos as operações existentes na literatura para autômatos não temporizados. Na seção 5 é definida a opacidade com base em linguagem e é mostrado um método para sua verificação, em autômatos temporizados. Por fim, a seção 6 resume a contribuição do artigo.

2. FUNDAMENTOS TEÓRICOS

Com o objetivo de modelar os sistemas a eventos discretos, é utilizada neste artigo a abordagem por autômatos determinísticos, conforme definidos em Cassandras e Lafortune (2008) pela quíntupla $G = (X, \Sigma, f, x_0, X_m)$, em que X é o conjunto de estados, Σ é o conjunto de eventos, $f : X \times \Sigma \to X$ é a função de transição, x_0 é o estado inicial e $X_m \subseteq X$ os estados marcados. Além disso, $f(x, \sigma) = y$ significa que há uma transição do estado x para o estado y por meio do evento σ. É possível que nem todas as transições que se originam em um estado do autômato sejam rotuladas por todos os eventos de Σ, portanto f é uma função parcial em relação ao seu domínio Σ.

Associado ao autômato G, podemos definir os seguintes conjuntos: *(i)* conjunto das transições: $\Delta = \{(x, \sigma, y) \in X \times \Sigma \times X | y = f(x, \sigma)\}$; *(ii)* conjunto de eventos ativos: $\Gamma(x) = \{\sigma \in \Sigma | \exists y \in X, f(x, \sigma) = y\} \subseteq 2^{\Sigma}$; *(iii)* o conjunto de todos os eventos que chegam no estado x, i.e., $\Upsilon(x) = \{\sigma \in \Sigma | \exists x' \in X, f(x', \sigma) = x\}$.

O conjunto de eventos Σ define um alfabeto e uma sequência em Σ é dada por $w = \sigma_1 ... \sigma_n$, tal que $\sigma_i \in \Sigma$, para $i = 1, ..., n$. O tamanho de w é dado por $\|w\|$ e a sequência vazia ϵ possui tamanho nulo. O Fecho de Kleene de Σ, denominado por Σ^*, é o conjunto de todas as sequências finitas formadas com os elementos de Σ. O domínio de f é estendido para $X \times \Sigma^*$ recursivamente, de forma que $f(x, \epsilon) = x$ e $f(x, se) = f(f(x, s), e)$ para $s \in \Sigma^*$ e $e \in \Sigma$. A linguagem gerada por G é definida como $\mathcal{L}(G) = \{s \in \Sigma^* | f(x_0, s)!\}$. A notação "!" significa "é definida", isto é, existe $y \in X$ tal que $f(x, s) = y$. $\mathcal{L}_m(G) = \{s \in \mathcal{L}(G) | f(x_0, s) \in X_m\}$ é a linguagem marcada de G.

Um autômato não determinístico é descrito pela quíntupla $G_{nd} = (X, \Sigma \cup \{\epsilon\}, f_{nd}, X_0, X_m)$, em que $X_0 \subseteq X$ e $f_{nd}(x, \sigma) \in 2^X, \sigma \in \Sigma \cup \{\epsilon\}$. Assim, um autômato G será determinístico se as seguintes condições forem satisfeitas: *(i)*$X_0 = \{x_0\}$; *(ii)* nenhuma transição é rotulado por ϵ; *(iii)* $\forall \sigma \in \Gamma(x)$, $y = f(x, \sigma)$ é único.

Considere dois conjuntos de eventos Σ_l e Σ_s, tais que $\Sigma_s \subset \Sigma_l$. A projeção de uma sequência a partir conjunto maior, Σ_l, no conjunto menor, Σ_s, é uma operação denotada por $P : {\Sigma_l}^* \to {\Sigma_s}^*$ e definida recursivamente como: $P(\epsilon) = \epsilon$; $P(\sigma) = \sigma$, se $\sigma \in \Sigma_s$; $P(\sigma) = \epsilon$, se $\sigma \in \Sigma_l \setminus \Sigma_s$; $P(s\sigma) = P(s)P(\sigma)$, para $s \in {\Sigma_l}^*$ e $\sigma \in \Sigma_l$. A operação de projeção é estendida para linguagens aplicando-a a todas as sequências da linguagem. Desta forma, dada $L \subseteq {\Sigma_l}^*$ tem-se que $P(L) = \{t \in {\Sigma_s}^* | (\exists s \in L), P(s) = t\}$.

A parte acessível de um autômato, denotada por $Ac(G)$, é obtida deletando-se de G os estados que não podem ser alcançados a partir de x_0 por sequências em $\mathcal{L}(G)$. Um estado $x \in X$ é coacessível se há um caminho composto por transições em G que leva de x para um estado marcado.

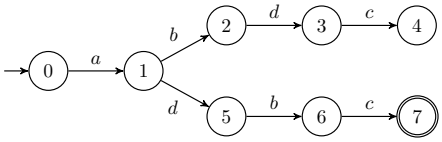

Figura 1. Autômato G para o exemplo de motivação.

O autômato $G^c = (X \cup \{x_d\}, \Sigma, f_{tot}, x_0, \{X \cup \{x_d\}) \setminus X_m\}$ que marca o complemento da linguagem marcada de G, $\Sigma^* \setminus \mathcal{L}_m(G)$, é tal que $f_{tot}(x, \sigma) = f(x, \sigma)$ se $\sigma \in \Gamma(x)$, e $f_{tot}(x, \sigma) = x_d$ caso contrário.

Dado um autômato G, cujo conjunto de eventos é Σ, e um conjunto menor $\Sigma_s \subset \Sigma$, o autômato que gera $P[\mathcal{L}(G)]$ e marca $P[\mathcal{L}_m(G)]$ é obtido substituindo-se as transições rotuladas por eventos em $\Sigma \setminus \Sigma_s$ por transições-ϵ. Tal procedimento resulta em um autômato não determinístico, G_{nd}, cujo correspondente autômato determinístico é obtido calculando-se o seu observador (Cassandras e Lafortune, 2008).

Considere dois autômatos determinísticos $G_1 = (X_1, \Sigma_1, f_1, x_{01}, X_{m1})$ e $G_2 = (X_2, \Sigma_2, f_2, x_{02}, X_{m2})$, cujos conjuntos de eventos ativos são dados respectivamente por $\Gamma_1(x)$ e $\Gamma_2(x)$. A operação de produto entre eles gera $G_1 \times G_2 = Ac(X_1 \times X_2, \Sigma_1 \cup \Sigma_2, f, (x_{01}, x_{02}), X_{m1} \times X_{m2})$, tal que $f((x_1, x_2), \sigma) = (f_1(x_1, \sigma), f_2(x_2, \sigma))$, se $\sigma \in \Gamma_1(x_1) \cap \Gamma_2(x_2)$, e não é definida caso contrário. Essa operação sincroniza os autômatos em seus eventos comuns, de forma que $\mathcal{L}(G_1 \times G_2) = \mathcal{L}(G_1) \cap \mathcal{L}(G_2)$.

2.1 Opacidade com base em linguagem

Como descrito em Wu e Lafortune (2013), um sistema é considerado opaco se o seu comportamento secreto não puder ser revelado a um observador externo, o qual presume-se que tenha conhecimento do funcionamento da planta. Em relação à observação de eventos é então considerado que o conjunto de eventos de uma planta é igual à união dos conjuntos disjuntos Σ_o e Σ_{uo}, de eventos observáveis e não observáveis, respectivamente. Portanto a projeção observável P_o é definida de forma que $P_o : \Sigma^* \to \Sigma_o^*$. Neste artigo, iremos considerar somente a opacidade com base em linguagem (Lin, 2011), que é definida da seguinte forma.

Definição 1. (Opacidade de linguagem). Considere um autômato $G = (X, \Sigma, f, x_o, X_m)$, cujas linguagens secreta e não secreta são denotadas por $L_S \subset \mathcal{L}(G)$ e $L_{NS} = \mathcal{L}(G) \setminus L_S$, respectivamente. O sistema representado por G é dito opaco em linguagem se para toda sequência $s \in L_S$ existir uma sequência $s' \in L_{NS}$, tal que $P_o(s) = P_o(s')$.

3. EXEMPLO DE MOTIVAÇÃO

Considere o autômato G representado na figura 1 que marca a linguagem secreta e é tal que $\Sigma_{uo} = \{d\}$. Assim a linguagem secreta é $L_S = \{adbc\}$ e a não secreta é $L_{NS} = \mathcal{L}(G) \setminus L_S$. Para a sequência $s = adbc \in L_S$, existe $s' = abdc \in L_{NS}$, cujas projeções observáveis são iguais, i.e., $P_o(s) = P_o(s') = abc$. Então, de acordo com a definição 1, o sistema é opaco.

Entretanto, considerando-se limites de tempo para a ocorrência das transições, por exemplo, poder-se-ia supor que quando o sistema está no estado 1, a ocorrência do evento b daria-se no intervalo real $[0, 1]$ segundos. Assim, é possível

imaginar que o caminho secreto leve entre 1 e 3 segundos para sair do estado 0 e chegar no 7 e que o caminho não secreto leve de 4 a 5 segundos para ir do estado inicial 0 ao estado 4. Note que um intruso seria capaz de estimar se o sistema está no estado secreto ou não a partir da medição do tempo que as sequências levam para ocorrer dado que $[1,3] \cap [4,5] = \emptyset$. Portanto, ainda que as projeções observáveis coincidam, o sistema temporizado descrito pode ser considerado como não opaco.

Assim, limites de tempo associados a transições influenciam no modo como deve ser definida a opacidade em comparação com autômatos não temporizados.

4. AUTÔMATO TEMPORIZADO POR INTERVALOS

Neste trabalho iremos considerar uma classe de autômatos temporizados cujos disparos ocorrem dentro de um intervalo de tempo.

Definição 2. (Autômato temporizado por intervalos). Um autômato temporizado por intervalos (ATI) é uma sêxtupla $G_T = (X, \Sigma, f, x_0, X_m, \mu)$ na qual X é o conjunto de estados, Σ é o conjunto de eventos, $f : X \times \Sigma \to 2^X$ é a função de transição, x_0 é o estado inicial e X_m é o conjunto de estados marcados. A função $\mu : X \times \Sigma \times X \to 2^{\mathbb{R}_+} \setminus \{\emptyset\}$ é a função de rotulamento, de forma que para um par $(x, y) \in X \times X$ e $\sigma \in \Sigma$, $\mu(x, \sigma, y) = I \subseteq 2^{\mathbb{R}_+} \setminus \{\emptyset\}$, se $f(x, \sigma) = y$, e não definida caso contrário.

De acordo com a definição 2, a cada transição $y = f(x, \sigma)$, é associado um intervalo de tempo I, medido em unidade de tempo (UT), que determina o intervalo no qual o sistema poderá realizar σ desde que chegou no estado x. Note que, de acordo com a definição 2, para um dado estado $x \in X$ é possível que existam transições com o mesmo evento, rotuladas por intervalos distintos e que levem a estados diferentes de G_T. Assim como nos autômatos não temporizados, é definido o conjunto de eventos ativos $\Gamma(x) = \{\sigma \in \Sigma | \exists y \in X, f(x, \sigma) = y\} \subseteq 2^{\Sigma}$. Ao longo do texto, iremos denotar por Δ o conjunto de transições, definido como $\Delta = \{(x, \sigma, y) | (x, y \in X)(\sigma \in \Sigma), f(x, \sigma) = y\}$. As definições de acessibilidade e coacessibilidade descritas na seção 2 para autômatos não temporizados se mantém para autômatos temporizados.

Seja $\sigma^T = (\sigma, [l, u]) \in \Sigma \times 2^{\mathbb{R}_+} \setminus \{\emptyset\}$. Define-se Δ_T, como o conjunto de transições temporizadas $\Delta_T = \{(x, \sigma^T, y) | (\sigma^T = (\sigma, [l, u])) \wedge ((x, \sigma, y) \in \Delta) \wedge (\mu(x, \sigma, y) = [l, u])\}$. A função de eventos ativos pode ser estendida para $\Gamma_T(x) = \{\sigma^T | (x, \sigma^T, y) \in \Delta_T\}$, o que nos permite definir $\Delta_T(x) = \{(x, \sigma^T, y) | \sigma^T \in \Gamma_T(x)\}$.

Um ATI G_T será determinístico se as seguintes condições forem satisfeitas: *(i)* $X_0 = \{x_0\}$; *(ii)* não há transições rotuladas por ϵ; *(iii)* para todo $\delta \in \Delta_T(x)$, se existirem $\delta_1 = (x, \sigma_1^T, x_1)$ e $\delta_2 = (x, \sigma_2^T, x_2)$ tais que $(x_1 \neq x_2) \wedge (\sigma_1^T = (\sigma, I_1) \wedge \sigma_2^T = (\sigma, I_2))$, então $I_1 \cap I_2 = \emptyset$. Quando, pelos menos uma das condições *(i)–(iii)* acima não for satisfeita, o autômato será dito não determinístico. Nesse caso, a palavra nula ϵ deverá ser acrescentada ao conjunto dos eventos de G_T.

Assim como para os autômatos não temporizados, pode-se estender o domínio de f recursivamente para $X \times \Sigma^*$, de forma que $f(x, \epsilon) = x$ e $f(x, s\sigma) = f(f(x, s), \sigma)$, para $s \in \Sigma^*$ e $\sigma \in \Sigma$. Por convenção, a transição (x, ϵ, x) estará associada ao intervalo $\emptyset$.

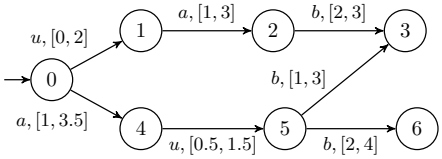

Figura 2. Autômato temporizado G_T.

Dada uma sequência $s = (\sigma_1, I_1)(\sigma_2, I_2) \dots (\sigma_n, I_n)$ em $(\Sigma \times 2^{\mathbb{R}_+} \setminus \emptyset)^*$, define-se a componente de s em relação a Σ^* como $\pi(s, \Sigma) = \sigma_1 \sigma_2 \dots \sigma_n$. A linguagem temporizada gerada por G_T é definida por $\mathcal{L}_T(G_T) = \{s_t = (\sigma_1, I_1) \dots (\sigma_n, I_n) \in (\Sigma \times 2^{\mathbb{R}_+} \setminus \emptyset)^* | (f(x_0, \sigma_1 \dots \sigma_n)!) \wedge (I_i = \mu(x_{i-1}, \sigma_i, x_i), i = 1, \dots, n) \wedge (x_0 \in X_0) \wedge (x_i = f(x_{i-1}, \sigma_i), i = 1, \dots, n)\} \cup \{(\epsilon, \emptyset)\}$. A linguagem temporizada marcada de G_T é dada por $\mathcal{L}_{mT}(G_T) = \{s \in \mathcal{L}_T(G_T) | f(x_0, \pi(s)) \in X_m\}$. A linguagem não temporizada gerada (ou simplesmente linguagem gerada) por G_T é definida como $\mathcal{L}(G_T) = \{\pi(s), \forall s \in \mathcal{L}_T(G_T)\}$ e a linguagem marcada não temporizada é $\mathcal{L}_m(G_T) = \{\pi(s), \forall s \in \mathcal{L}_{mT}(G_T)\}$.

Exemplo 1. No autômato da figura 2, o conjunto de estados é $X = \{0, 1, 2, 3, 4, 5, 6\}$, $x_0 = 0$, o conjunto dos eventos $\Sigma = \{a, b, u\}$. Os intervalos de tempo atribuídos a cada transição estão especificados na figura 2 ao lado de seu evento correspondente. Por exemplo, na transição $\delta = (1, a, 2)$, o evento a acontecerá entre 1 e 3 UTs, representado na figura como $a, [1, 3]$. A linguagem temporizada gerada por G_T é igual a $\mathcal{L}_T(G_T) = \{(\epsilon, \emptyset); (u, [0, 2]); (a, [1, 3.5]); (u, [0, 2])(a, [1, 3]); (a, [1, 3.5])(u, [0.5, 1.5]); (u, [0, 2])(a, [1, 3])(b, [2, 3]); (a, [1, 3.5])(u, [0.5, 1.5])(b, [2, 4]); (a, [1, 3.5])(u, [0.5, 1.5])(b, [1, 3])\}$.

Definição 3. (Interseção de sequências temporizadas). Sejam $s_1 = (\sigma_1, I_1)(\sigma_2, I_2) \dots (\sigma_n, I_n)$ e $s_2 = (\sigma'_1, I'_1)(\sigma'_2, I'_2) \dots (\sigma'_n, I'_n)$ duas sequências temporizadas em $\Sigma \times 2^{\mathbb{R}_+} \setminus \emptyset$, tais que $\|\pi(s_1)\| = \|\pi(s_2)\|$. Define-se a interseção de s_1 com s_2 da seguinte forma:

$$s_1 \cap s_2 = \begin{cases} (\sigma_1, I_1 \cap I'_1)(\sigma_2, I_2 \cap I'_2) \dots (\sigma_n, I_n \cap I'_n), \\ \text{se } (\pi(s_1, \Sigma) = \pi(s_2, \Sigma)) \wedge (I_i \cap I'_i \neq \emptyset, \\ i = 1, 2, \dots, n); \\ (\epsilon, \emptyset), \text{ caso contrário.} \end{cases}$$

Definição 4. (Interseção entre linguagens temporizadas). Sejam L_1 e L_2 duas linguagens temporizadas. A interseção entre as linguagens L_1 e L_2 é definida como:

$$L_1 \cap L_2 = \begin{cases} \{s | (\exists u \in L_1) \wedge (\exists v \in L_2), s = u \cap v\} \setminus \{(\epsilon, \emptyset)\}, \\ \text{se } (\epsilon, \emptyset) \notin L_1 \wedge (\epsilon, \emptyset) \notin L_2; \\ \{s | (\exists u \in L_1) \wedge (\exists v \in L_2), s = u \cap v\}, \text{ caso} \\ \text{contrário.} \end{cases}$$

Exemplo 2. Sejam as linguagens temporizadas $L_1 = \{(a, [0, 2]); (a, [0, 2])(b, [2, 5])\}$ e $L_2 = \{(a, [1, 2]); (a, [1, 2])(b, [3, 6])\}$ definidas em $\Sigma = \{a, b\}^*$. A interseção $L_1 \cap L_2$ é obtida da seguinte forma. Primeiramente são comparadas as sequências de mesmo tamanho $s_1 = (a, [0, 2]) \in L_1$ e $s_2 = (a, [1, 2]) \in L_2$, pois $\pi(s_1) = \pi(s_2)$. A interseção entre seus intervalos de tempo é $[0, 2] \cap [1, 2] = [1, 2]$, portanto de acordo com a definição 3 tem-se que $s_1 \cap s_2 = (a, [1, 2])$. Analogamente, são comparadas $(a, [0, 2])(b, [2, 5]) \in L_1$ e $(a, [1, 2])(b, [3, 6]) \in L_2$. Para o evento a é feita a interseção dos intervalos de tempo $[0, 2] \cap [1, 2] = [1, 2]$ e para o evento b, $[2, 5] \cap [3, 6] = [3, 5]$. Portanto, a sequência resultante da interseção é $(a, [1, 2])(b, [3, 5])$. Por fim, utilizando a

definição 4, é obtida a interseção entre as linguagens $L_1 \cap L_2 = \{(a,[1,2]);(a,[1,2])(b,[3,5])\}$.

Na projeção observável de sequências temporizadas, os eventos não observáveis, embora não apareçam na projeção, afetam o tempo de observação do próximo evento observável.

Definição 5. (Soma de intervalos de tempo). A soma de n intervalos $I_i = [l_i, u_i]$, tais que $l_i, u_i \in \mathbb{R}$ para $i = 1, \ldots, n$, é definida como $\sum_{i=1}^{n} I_i = [\sum_{i=1}^{n} l_i, \sum_{i=1}^{n} u_i]$.

Definição 6. (Projeção). Seja Σ_s um subconjunto de um conjunto de eventos Σ_l. Defina $\Sigma_\epsilon = (\Sigma_l \setminus \Sigma_s) \cup \{\epsilon\}$. Então, a projeção de uma sequência $s \in (\Sigma_l \times 2^{\mathbb{R}_+} \setminus \emptyset)^*$ é o mapeamento $\Pi : (\Sigma_l \times 2^{\mathbb{R}_+} \setminus \emptyset)^* \to (\Sigma_s \cup \{\epsilon\} \times 2^{\mathbb{R}_+} \setminus \emptyset)^*$ definido recursivamente como:

(i) $\Pi[(\epsilon, I)] = (\epsilon, I)$;

(ii) $\Pi[(\sigma, I)] = \begin{cases} (\sigma, I), \sigma \in \Sigma_s, \\ (\epsilon, I), \sigma \in \Sigma_\epsilon; \end{cases}$

(iii) $\Pi[(\sigma_1, I_1)(\sigma_2, I_2)] = \begin{cases} (\sigma_1, I_1)(\sigma_2, I_2), \sigma_1, \sigma_2 \in \Sigma_s, \\ (\sigma_2, I_1 + I_2), (\sigma_1 \in \Sigma_\epsilon) \wedge \\ \quad (\sigma_2 \in \Sigma_s); \\ (\sigma_1, I_1)(\epsilon, I_2), (\sigma_1 \in \Sigma_s) \wedge \\ \quad (\sigma_2 \in \Sigma_\epsilon); \\ (\epsilon, I_1 + I_2), \sigma_1, \sigma_2 \in \Sigma_\epsilon; \end{cases}$

(iv) $\Pi[s(\sigma, I)] = \Pi[\Pi(s)(\sigma, I)]$, $s \in (\Sigma_l \times 2^{\mathbb{R}_+} \setminus \emptyset)^*$, $|s| \geq 2$, e $\sigma \in \Sigma_l \times 2^{\mathbb{R}_+} \setminus \emptyset$.

Note que a projeção de uma sequência elimina os eventos não pertencentes ao conjunto menor e adiciona seus tempos de ocorrência ao próximo evento em Σ_s. De acordo com a definição 6, a projeção de uma sequência formada apenas por eventos em Σ_ϵ retorna (ϵ, I), sendo I a soma de todos os intervalos associados aos eventos da sequência.

Exemplo 3. Considere a sequência temporizada $s_t = (u,[0,2])(a,[1,3])(b,[2,3])$ pertencente à linguagem temporizada do autômato da figura 2, definida no conjunto de eventos $\Sigma = \{a, b, u\}$. Deseja-se obter sua projeção no conjunto menor de eventos $\Sigma_s = \{a, b\} \subset \Sigma$. Portanto, utilizando a definição 6, tem-se que $\Pi(s_t) = (a,[1,5])(b,[2,3])$.

4.1 Operações com autômatos temporizados

Para a definição de operações envolvendo autômatos temporizados, a seguinte hipótese é necessária:

H1. Os autômatos temporizados não possuem ciclos de estados conectados por transições-ϵ.

O motivo dessa restrição é que ciclos e autolaços como os descritos acima podem atrasar a observação de um evento indefinidamente, resultando, de acordo com a definição 6, em infinitas possibilidades de projeções de uma sequência.

De acordo com a definição 2, eventos em autômatos temporizados não ocorrem necessariamente de forma assíncrona, portanto as operações definidas na seção 2 de complemento, observador e produto para autômatos não temporizados, devem ser generalizadas para autômatos temporizados. Em Wang et al. (2018) é definida a operação de refinamento por partição, na qual um ATI é transformado em um equivalente não temporizado por meio de um processo baseado em divisão de intervalos de tempo. A metodologia a ser utilizada neste artigo evita a necessidade desse passo intermediário.

• *Autômato projeção*

Considere um ATI $G_{T,\epsilon} = (X, \Sigma \cup \epsilon, f, X_0, X_m, \mu)$, e suponha que o conjunto de eventos Σ seja particionado da seguinte forma: $\Sigma = \Sigma_1 \dot{\cup} \Sigma_2$. Seja $\Pi_1 : (\Sigma \times 2^{\mathbb{R}_+} \setminus \emptyset)^* \to (\Sigma_1 \cup \{\epsilon\} \times 2^{\mathbb{R}_+} \setminus \emptyset)^*$. Então, o autômato que gera e marca $\Pi_1[\mathcal{L}_T(G_T)]$ e $\Pi_1[\mathcal{L}_{mT}(G_T)]$, respectivamente, é obtido substituindo-se as transições rotuladas por eventos em $\Sigma_2 = \Sigma \setminus \Sigma_1$, por transições-$\epsilon$, porém mantendo-se os seus respectivos intervalos de tempo e, em seguida, eliminando-se as transições-ϵ do autômato temporizado não determinístico resultante por meio da projeção. O autômato obtido ao final dessas operações é, em geral, um autômato não determinístico sem transições-ϵ, denominado autômato projeção e denotado por G_P.

Definição 7. (Autômato projeção). Dado um autômato temporizado não determinístico com transições-ϵ $G_{T,\epsilon} = (X, \Sigma \cup \{\epsilon\}, f, X_0, X_m, \mu)$, o autômato projeção de $G_{T,\epsilon}$ é o ATI $G_{T_P} = (X_P, \Sigma, f_P, X_{0_P}, X_{m_P}, \mu_P)$, em que $X_P = \{x \in X | \exists \sigma \in \Sigma, \sigma \in \Upsilon(x)\} \cup X_0$, $f_P(x_P, \sigma) = x'_P \in X_P$, se $\exists s = \epsilon \ldots \epsilon\sigma, x'_P = f(x_P, s)$, e não definida, caso contrário, $X_{0_P} = X_0$, $X_{m_P} = X_m \cap X_P$ e $\mu_P(x, \sigma, y)$ é definida como $\mu_P(x, \sigma, y) = I + \sum_{i=1}^{n} I_i$, para todo $x \in X_P$ e $\sigma \in \Sigma$ que satisfazem a seguinte condição: $\exists s_t = (\epsilon, I_1) \ldots (\epsilon, I_n)(\sigma, I), f(x, \pi(s_t)) = y$.

É importante ressaltar que, de acordo com a definição 7, é possível que, para um dado $x_P \in X_P$, existam transições rotuladas pelo mesmo evento, porém com intervalos de tempo diferentes, que levem a estados distintos em G_{T_P}.

Definição 8. (Caminho). Considere um estado $y_0 \in X$ de um ATI não determinístico $G_{T,\epsilon}$ e seja Δ_ϵ conjunto de transições de $G_{T,\epsilon}$. Então o conjunto de caminhos que se iniciam em y_0 é definido da seguinte forma:

$$C(y_0) = \{(y_0, \sigma_1, y_1)(y_1, \sigma_2, y_3) \ldots (y_{n-1}, \sigma_n, y_n) \\ | (y_{i-1}, \sigma_i, y_i) \in \Delta_\epsilon, i = 1, 2, \ldots, n\}$$

Note que $C(y_0)$ é composto por todas as sequências finitas de transições definidas em $G_{T,\epsilon}$ que começam em y_0.

Definição 9. (Caminho detectável). Dado um ATI não determinístico $G_{T,\epsilon}$, define-se caminho detectável que começa no estado y_0 aos caminhos pertencentes ao seguinte subconjunto de $C(y_0)$:

$$CD(y_0) = \{(y_0, \epsilon, y_1)(y_1, \epsilon, y_2) \ldots (y_{n-1}, \epsilon, y_n)(y_n, \sigma, y') \\ \in C(y_0) | \sigma \in \Sigma\} \cup \{(y_0, \sigma, y') \in C(y_0) | \sigma \in \Sigma\}$$

Note que um caminho detectável é formado por uma sequência de transições-ϵ definidas a partir do estado y_0 e que termina com uma única transição rotulada por $\sigma \in \Sigma$; ou uma única transição (y_0, σ, y'), $\sigma \in \Sigma$.

Exemplo 4. Considere o autômato G_T da figura 2 e substitua as transições rotuladas por u por transições-ϵ, de forma a obter-se $G_{T,\epsilon}$. O conjunto de caminhos do autômato $G_{T,\epsilon}$ resultante que começa no estado 1 é dado por $C(1) = \{(1, a, 2); (1, a, 2)(2, b, 3)\}$, e o conjunto de caminhos detectáveis a partir do estado 0 em G_T, ϵ é $CD(0) = \{(0, \epsilon, 1)(1, a, 2); (0, a, 4)\}$.

Portanto, o seguinte algoritmo (algoritmo 1) pode ser utilizado para a obtenção do autômato projeção equivalente.

Algoritmo 1. Obtenção do autômato projeção

Entrada: $G_{T,\epsilon} = (X, \Sigma \cup \{\epsilon\}, f, x_0, X_m, \mu)$, Δ_ϵ

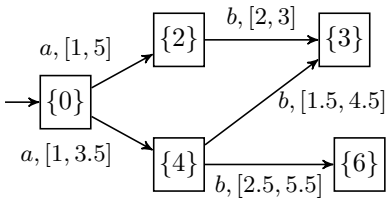

Figura 3. Autômato projeção de G_T da figura 2.

Saída: $G_{T_P} = (X_P, \Sigma, f_P, X_{0_P}, X_{m_P}, \mu_P)$, Δ_P

1: Faça $X_P = \{x \in X | (\exists \sigma \in \Sigma)[\sigma \in \Upsilon(x)]\} \cup \{X_0\}$, $X_{0_P} = X_0$, $\Delta_P = \emptyset$, $\mu_P = \emptyset$
2: Para cada $x \in X_P$
3: Calcule $CD(x)$
4: Para $p \in CD(x)$
5: Faça $\delta_P = (x, \sigma, x')$ em que $\sigma \in \Sigma$ e x' são tais que ou $p = (x, \epsilon, y_1)(y_1, \epsilon, y_2), \ldots (y_n, \sigma, x')$ ou $p = (x, \sigma, x')$
6: $\mu_P(\delta_P) = \sum_{\delta \in p} \mu(\delta)$ ▷ *Por abuso de notação consideramos p como um conjunto formado pelas transições* $(x_i, \sigma_i, x_{i+1}) \in p$.
7: $\Delta_P = \Delta_P \cup \{\delta_P\}$
8: Faça $X_{m_P} = X_m \cap X_P$
9: Definir f_P a partir de Δ_P

Exemplo 5. Considere o ATI não determinístico G_T , representado na figura 2, e forme o autômato $G_{T,\epsilon}$ a partir de G_T substituindo-se as transições rotuladas pelo evento u por transições-ϵ e mantendo-se os intervalos de tempo originais. Suponha que se deseje obter o autômato projeção G_{T_P} de $G_{T,\epsilon}$. Assim, $G_{T,\epsilon}$ deve ser fornecido como entrada do algoritmo 1. O primeiro passo do algoritmo é calcular X_P, que é composto pelos estados de $G_{T,\epsilon}$ que recebem uma transição $\sigma \in \Sigma$ e pelo estado inicial; portanto $X_P = \{0, 2, 3, 4, 6\}$. O próximo passo é analisar todos os caminhos detectáveis dos estados de X_P. Começando por $x = 0$, tem-se $CD(0) = \{(0, \epsilon, 1)(1, a, 2); (0, a, 4)\}$ e, portanto, as transições resultantes para o observador são $(0, a, 2)$ e $(0, a, 4)$, cujos respectivos intervalos de tempo são $\mu_P(0, a, 2) = \mu(0, u, 1) + \mu(1, a, 2) = [1, 5]$ e $\mu_P(0, a, 4) = \mu(0, a, 4) = [1, 3.5]$. Na próxima iteração é calculado $CD(2) = \{(2, b, 3)\}$ e a nova transição $(2, b, 3)$, sendo $\mu_P(2, b, 3) = \mu(2, b, 3) = [2, 3]$, adicionada a G_{T_P}. Analogamente teremos $CD(4) = \{(4, \epsilon, 5)(5, b, 6); (4, \epsilon, 5)(5, b, 3)\}$ que resulta nas transições $(4, b, 6)$ e $(4, b, 3)$, respectivamente, rotuladas por $\mu_P(4, b, 6) = \mu(4, \epsilon, 5) + \mu(5, b, 6) = [2.5, 5.5]$ e $\mu_P(4, b, 3) = \mu(4, \epsilon, 5) + \mu(5, b, 3) = [1.5, 4.5]$. Finalmente, $CD(3) = CD(6) = \emptyset$. O autômato G_{T_P} resultante está representado pela figura 3.

• *Autômato temporizado determinístico equivalente a um ATI não determinístico*

Considere um ATI não determinístico $G_T = (X, \Sigma, f, X_0, X_m, \mu)$ e suponha, sem perda de generalidade, que G_T não tenha transições-ϵ. Seu autômato determinístico equivalente $G_{T_D} = (X_D, \Sigma, f_D, x_{0_D}, X_{m_D}, \mu_D)$, tal que $\mathcal{L}(G_T)$ e $\mathcal{L}_m(G_T)$, pode ser obtido de acordo com o algoritmo 2.

Algoritmo 2. Obtenção de um equivalente determinístico

Entrada: $G_T = (X, \Sigma, f, X_0, X_m, \mu)$, Δ

Saída: $G_{T_D} = (X_D, \Sigma, f_D, x_{0_D}, X_{m_D}, \mu_D)$, Δ_D

1: Definir $x_{0_D} = X_0$, $X_D = \{x_{0_D}\}$, $X_{aux} = \{x_{0_D}\}$, $\Delta_D = \emptyset$, $\mu_D = \emptyset$
2: Enquanto $X_{aux} \neq \emptyset$
3: $X_{aux} = X_D$
4: Para cada $x_D \in X_D$
5: Para cada $\sigma \in \Sigma$
6: ${x_D}^\sigma = \{x \in x_D | \exists y = f(x, \sigma)\}$
7: $T^\sigma = \{I | \exists x \in {x_D}^\sigma | \mu(x, \sigma, f(x, \sigma)) = I\} = \{I_1, I_2, \ldots, I_n\}$
8: Para $1 \leq k \leq n$
9: $C_k = \{I_{i_1} \cap \cdots \cap I_{i_k}, 1 \leq i_1 < \ldots < i_k \leq n\}$
10: Se $k \geq 2$, $U_k = \bigcup_{I \in C_k} I$
11: Se $k = 1$, $U_1 = \emptyset$
12: $I_C = \emptyset$
13: Para $1 \leq k \leq n$
14: Para cada $C_{kj} \in C_k$
15: $I_{C_{kj}} = C_{kj} \setminus U_{k+1} \setminus U_{k+2} \setminus \cdots \setminus U_n$
16: $I_C = I_C \cup \{I_{C_{kj}}\}$
17: $I_C = (I_C \cup \{U_n\}) \setminus \{\emptyset\}$
18: Para cada $I \in I_C$
19: $x'_D = \{y \in X | (x \in x_D), I \subseteq \mu(x, \sigma, y)\}$
20: $\delta^\sigma = (x_D, \sigma, x'_D)$
21: $\mu_D(\delta^\sigma) = I$
22: $\Delta_D = \Delta_D \cup \{\delta^\sigma\}$
23: $X_D = X_D \cup \{x'_D\}$
24: $X_{m_D} = X_{m_D} \cup \{x'_D\}$, se $(\exists x \in x'_D) | x \in X_m$
25: $X_{aux} = X_D \setminus X_{aux}$
26: Obter f_D a partir de Δ_D

Ressaltamos que quando for dado um ATI $G_{T,\epsilon}$, não determinístico que possua transições-ϵ, o seu autômato determinístico correspondente, G_{T_D}, será obtido aplicando-se os algoritmos 1 e 2, nessa ordem.

Exemplo 6. Considere o ATI não determinístico G_T, sem transições-ϵ, representado na figura 3 e renomeie seus estados $\{0\}, \{2\}, \{3\}, \{4\}, \{6\}$ respectivamente para $0, 2, 3, 4, 6$. De acordo com o algoritmo 2, o primeiro passo é obter o estado inicial de G_{T_D} que é dado por $x_{0_D} = \{0\}$, conforme mostrado na figura 4. O passo seguinte (linhas 3–8) é obter as transições equivalentes àquelas definidas para os estados de X_0. Para autômato G_T, têm-se transições com o evento a, cujos intervalos de tempo são $[1, 5]$ e $[1, 3.5]$. Portanto tem-se $T^a = \{[1, 5]; [1, 3.5]\}$ e consequentemente $n = 2$. A partir da linha 8 do algoritmo 2, os conjuntos formados pelas interseções são $C_1 = \{[1, 5]; [1, 3.5]\}$ e $C_2 = \{[1, 3.5]\}$, que geram o conjunto das uniões $U_2 = \{[1, 3.5]\}$. Em seguida são calculados os conjuntos disjuntos $I_{C_{11}} = \{[1, 5] \setminus [1, 3.5]\} = \{(3.5, 5]\}$ e $I_{C_{12}} = \{[1, 3.5] \setminus [1, 3.5]\} = \emptyset$. Desta forma os intervalos disjuntos que determinarão as novas transições com o evento a que partem de x_{0_D}são $I_C = \{[1, 3.5], (3.5, 5]\}$. Em seguida (linhas 18 – 24) analisamos cada um deles para determinar o próximo estado de suas respectivas transições. O intervalo $[1, 3.5] \subseteq [1, 3.5] \subseteq [1, 5]$, portanto $a, [1, 3.5]$ leva ao estado $\{2, 4\}$ em G_{T_D}. O intervalo $(3.5, 5] \subseteq [1, 5]$, portanto $a, (3.5, 5]$ leva ao estado $\{2\}$ em G_{T_D}. Os novos estados de G_{T_D} são portanto $\{2\}$ e $\{2, 4\}$. Para os próximos estados é feito procedimento análogo, cujo resultado final está representado na figura 4.

• *Complementar de um ATI*

Considere, sem perda de generalidade, um automato temporizado determinístico $G_T = (X, \Sigma, \Delta, x_0, X_m, \mu)$ e considere o problema de se calcular o autômato $G_T^c = (X \cup$

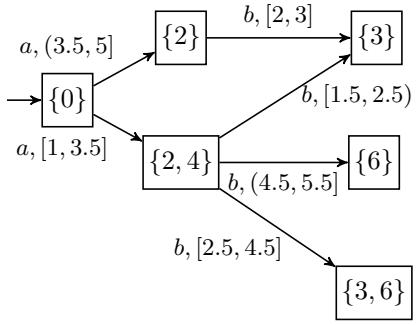

Figura 4. Autômato temporizado determinístico.

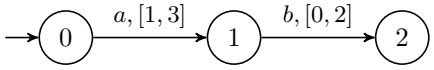

Figura 5. Autômato G_{1_T} utilizado nos exemplos 7 e 8.

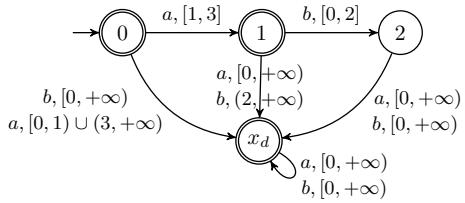

Figura 6. $G^c_{1_T}$ obtido no exemplo 7.

$\{x_d\}, \Sigma, \Delta^c, x_0, \{X \cup \{x_d\}\} \backslash X_m, \mu^c)$ cuja linguagem marcada temporizada $\mathcal{L}_{mT}(G^c_T) = (\Sigma \times 2^{\mathbb{R}_+} \setminus \emptyset)^* \setminus \mathcal{L}_{mT}(G_T)$.

A partir de G_T é feito o seguinte procedimento para obter-se o seu complementar G^c_T. Primeiro é criado um estado novo x_d e para cada $x \in X$ e $\sigma \in \Sigma$ são criadas transições $(x, (\sigma, I), x_d)$, tais que $I = [0, +\infty) - \mu(x, \sigma, \Gamma(x, \sigma))$, se $\Gamma(x, \sigma)!$ ou $I = [0 + \infty)$, caso contrário. O segundo passo consiste em, para cada $\sigma \in \Sigma$, criar autolaços $(x_d, (\sigma, [0, +\infty)), x_d)$. Finalmente marca-se todos os estados originalmente não marcados, incluindo x_d, desmarca-se todos os estados originalmente marcados.

As transições temporizadas de G^c_T são definidas como $\Delta^c_T = \Delta_T \cup \Delta^c_I \cup \Delta_{[0,+\infty)} \cup \Delta_{x_d}$, sendo $\Delta^c_I = \{(x, (\sigma, [0, +\infty) - I), x_d) | (x, (\sigma, I), y) \in \Delta_T\}$ e $\Delta_{[0,+\infty)} = \{(x, (\sigma, [0, +\infty), x_d)) | (x, \sigma, y) \notin \Delta\}$ e $\Delta_{x_d} = \{(x_d, (\sigma, [0 + \infty)), x_d) | \sigma \in \Sigma\}$.

Exemplo 7. Seja o autômato G_{1_T} da figura 5. Considerando o estado marcado $X_m = \{2\}$, a linguagem temporizada marcada será $\mathcal{L}_{mT}(G_{1_T}) = \{(a, [1, 3])(b, [0, 2])\}$. Para obter o autômato que marca o complemento de $\mathcal{L}_{mT}(G)$ primeiro é criado x_d. Em seguida, para o estado 0 temos o evento $(a, [1, 3])$, cujo complemento em relação a Σ e ao intervalo de tempo é o conjunto $\{(b, [0, +\infty)); (a, [0, 1) \cup (3, +\infty))\}$. Para cada intervalo deste conjunto é criada uma nova transição com o evento a de 0 para x_d. Analogamente para o estado 1, os eventos que complementam $(b, [0, 2])$ são $(a, [0, +\infty))$ e $(b, (2, +\infty))$, que por sua vez definirão novas transições de 1 para x_d. Finalmente são criados autolaços com os eventos $(a, [0, +\infty))$ e $(b, [0, +\infty))$ em x_d. O estado 2 é desmarcado e 0, 1 e x_d são marcados, o autômato resultante $G^c_{1_T}$ está representado na figura 6.

• *Produto de dois autômatos temporizados*

A definição do produto entre dois autômatos temporizados é similar à versão para autômatos não temporizados, descrita na seção 2, que sincronizam apenas em eventos comuns, exceto que no caso temporizado consideram-se também os intervalos de tempo associados às transições.

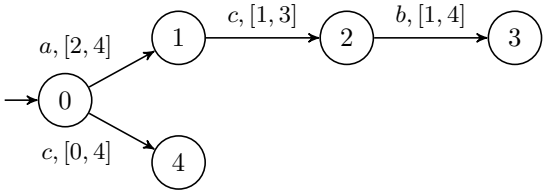

Figura 7. Autômato G_{2_T} utilizado no exemplo 8.

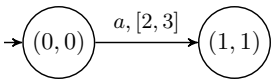

Figura 8. Produto dos autômatos das figuras 5 e 7.

Definição 10. (Operação Produto). O produto entre dois autômatos temporizados $G_{1_T} = (X_1, \Sigma_1, f_1, x_{0_1}, X_{m_1}, \mu_1)$ e $G_{2_T} = (X_2, \Sigma_2, f_2, , x_{0_2}, X_{m_2}, \mu_2)$, tais que Δ_{T_1} e Δ_{T_2} são respectivamente seus conjuntos de transições temporizadas, e Γ_1 e Γ_2 são respectivamente seus conjuntos de eventos ativos, é dado pelo autômato $G_{1_T} \times G_{2_T} = Ac(X_1 \times X_2, \Sigma_1 \cup \Sigma_2, f_{1\times2}, \Gamma_{1\times2}, (x_{0_1}, x_{0_2}), X_{m_1} \times X_{m_2}, \mu_{1\times2})$ em que para $x_1 \in X_1$ e $x_2 \in X_2$, define-se $f_{1\times2}((x_1, x_2)) = (f_1(x_1, \sigma), f_2(x_2, \sigma))$ se $\sigma \in \Gamma_1(x_1) \cap \Gamma_2(x_2)$, caso contrário a função não é definida. A função de rotulamento é tal que $\mu_{1\times2}((x_1, x_2), \sigma, (y_1, y_2)) = \mu_1(x_1, \sigma, y_1) \cap \mu_2(x_2, \sigma, y_2)$.

Exemplo 8. Sejam os autômatos $G_{1_T} = (X_1, \Sigma_1, \Delta_1, x_{01}, X_{m_1}, \mu_1)$ (figura 5) e $G_{2_T} = (X_2, \Sigma_2, \Delta_2, x_{02}, X_{m_2}, \mu_2)$ (figura 7), cujos conjuntos de transições temporizadas são respectivamente Δ_{T1} e Δ_{T2}. O método para obtenção do produto $G_{P_T} = G_{1_T} \times G_{2_T}$ começa no estado $(0, 0)$ no qual ocorre a sincronização entre $(0, (a, [1, 3]), 1) \in \Delta_{T_1}$ e $(0, (a, [2, 4]), 1) \in \Delta_{T_2}$, que resulta na transição temporizada $((0, 0), (a, [2, 3]), (1, 1))$ em G_{P_T}. Como no estado $\{1, 1\}$ não há mais eventos de G_{1_T} e G_{2_T} que sincronizem, o autômato resultante é mostrado na figura 8.

5. OPACIDADE EM AUTÔMATOS TEMPORIZADOS

Aqui serão considerados eventos observáveis, Σ_o, e eventos não observáveis, Σ_{uo}, como na seção 2, e portanto será utilizada a projeção observável $\Pi_o : (\Sigma \times 2^{\mathbb{R}_+} \setminus \emptyset)^* \rightarrow (\Sigma_o \cup \{\epsilon\} \times 2^{\mathbb{R}_+} \setminus \emptyset)^*$. Seja $G_T = (X, \Sigma, f, x_0, X_m, \mu)$, cuja linguagem temporizada secreta é representada por $L_S \subseteq \mathcal{L}(G_T)$ e a linguagem não secreta por $L_{NS} = \mathcal{L}(G_T) \backslash L_S$. Se para toda sequência temporizada $s \in L_S$ existir $s' \in L_{NS}$, tal que $\Pi_o(s') \cap \Pi_o(s) \neq \emptyset$, o sistema é opaco, caso contrário ele não o é. Além disso, para $\Pi_o(s') \cap \Pi_o(s) = \Pi_o(s')$ o sistema é considerado fortemente opaco. Para a verificação de opacidade, os eventos não observáveis são sempre substituídos por ϵ nos autômatos utilizados, com o objetivo de realizar as operações definidas anteriormente nos algoritmos 1 e 2.

5.1 Verificação de Opacidade em Autômatos Temporizados

A verificação de opacidade de linguagem temporizada segue os mesmos passos de sua versão não-temporizada, usualmente utilizados na literatura. Dados G_{S_T}, que marca a linguagem secreta L_S, e G_{NS_T}, que marca a linguagem não secreta L_{NS}, são obtidos seus autômatos projeção, que em seguida são transformados em determinísticos, gerando $G_{S_T o}$ e $G_{NS_T o}$, respectivamente. Então é feito o complemento $G^c_{NS_T o}$ e finalmente a composição produto

$G_P = G_{S_{To}} \times G^c_{NS_{To}}$. Se $\mathcal{L}_{mT}(G_P) = \emptyset$, então o sistema é completamente opaco. Se $\mathcal{L}_{mT}(G_P) \neq \emptyset$ e $\mathcal{L}_{mT}(G_P) = \mathcal{L}_{mT}(G_{S_{To}})$, então o sistema não é opaco. Caso contrário, o sistema será parcialmente opaco e a janela na qual a opacidade é garantida é determinada pela linguagem marcada $\mathcal{L}_m(G_{S_{To}} \times G_{NS_{To}})$.

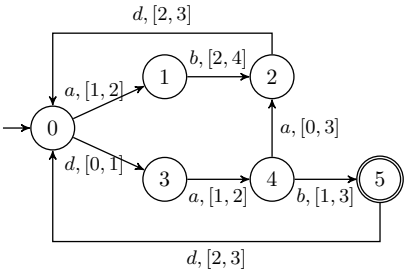

Figura 9. Autômato temporizado G_S do exemplo 9.

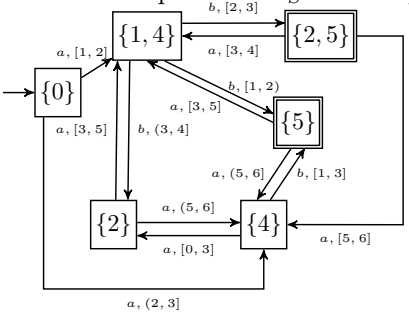

Figura 10. Autômato observador determinístico G_{So} de G_S mostrado na figura 9.

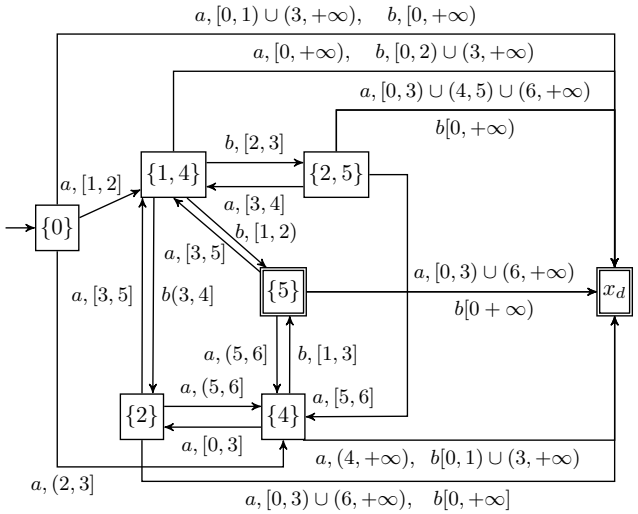

Figura 11. Autômato G^c_{NS} obtido no exemplo 9.

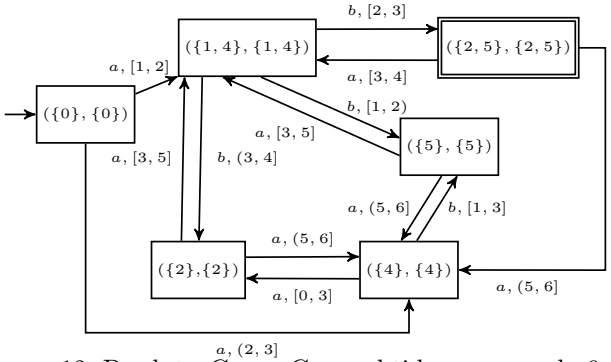

Figura 12. Produto $G_{So} \times G_{NSo}$ obtido no exemplo 9.

Exemplo 9. Seja o ATI G_S na figura 9, que marca a linguagem secreta L_S, e é tal que $\Sigma_o = \{a, b\}$. O ATI G_{NS} que marca a linguagem não-secreta L_{NS} é obtido a partir de G_S desmarcando-se o estado 5 e marcando os outros estados. Para a verificação da opacidade, o primeiro passo é obter os autômatos projeção determinísticos (observadores) de G_S e G_{NS}, respectivamente G_{So} mostrado na figura 10 e G_{NSo} idêntico a G_1 exceto pelos estados marcados, que são $\{\{0\}, \{1, 4\}, \{2\}, \{4\}\}$. Então é calculado o autômato que marca o complemento da linguagem não-secreta G^c_{NSo} mostrado na figura 11. É fácil ver que $\mathcal{L}_{mT}(G_{So} \times G^c_{NSo}) \neq \emptyset$. Além disso $\mathcal{L}_m(G_P) \subset \mathcal{L}_m(G_{So}) = \Pi_o(L_S)$, o que significa que o sistema é parcialmente opaco em relação a L_S e Σ_o. Enfim é possível obter a janela no qual o sistema é considerado opaco por meio da operação $G_{So} \times G_{NSo}$, que marca $\Pi_o(L_S) \cap \Pi_o(L_{NS})$ e seu resultado está na figura 12.

6. COMENTÁRIOS FINAIS

Neste artigo foi apresentada a definição de uma classe de autômatos temporizados nos quais há um intervalo de tempo associado a cada transição. Foram definidas as operações de complemento, projeção (observador) e composição produto. A opacidade baseada em linguagem e sua verificação foram também redefinidas de forma a considerar-se o tempo de cada transição.

REFERÊNCIAS

Alves, M.V., Carvalho, L.K., e Basilio, J.C. (2020). Supervisory control of networked discrete event systems with timing structure. *IEEE Transactions on Automatic Control*, 66(5), 2206–2218.

Barcelos, R.J. e Basilio, J.C. (2021). Enforcing current-state opacity through shuffle and deletions of event observations. *Automatica*, 133, 109836.

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.

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

Ji, Y. e Lafortune, S. (2017). Enforcing opacity by publicly known edit functions. In *56th IEEE Conference on Decision and Control*, 377–383.

Lin, F. (2011). Opacity of discrete event systems and its applications. *Automatica*, 47(3), 496–503.

Viana, G., Alves, M.V.S., e Basilio, J.C. (2021). Codiagnosability of networked discrete event systems with timing structure. *IEEE Transactions on Automatic Control*.

Wang, L., Zhan, N., e An, J. (2018). The opacity of real-time automata. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 37(11), 2845–2856.

Wu, Y.C. e Lafortune, S. (2013). Comparative analysis of related notions of opacity in centralized and coordinated architectures. *Discrete Event Dynamic Systems: Theory and Applications*, 23(3), 307–339.

Wu, Y.C. e Lafortune, S. (2014). Synthesis of insertion functions for enforcement of opacity security properties. *Automatica*, 50(5), 1336–1348.