TÉCNICAS DE CAUSALIZAÇÃO DE CONTROLADORES MAX-PLUS: VALIDAÇÃO EM UM SIMULADOR

Marcelo Xavier Reis Souza*, Vinícius Mariano Gonçalves*

*Programa de Pós-Graduação em Engenharia Elétrica - Universidade Federal de Minas Gerais - Av. Antônio Carlos 6627, 31270-901, Belo Horizonte, MG, Brasil

Emails: mxrs@ufmg.br, mariano@cpdee.ufmg.br

Abstract— The theory of linear systems in the so-called *Max-Plus Algebra* has gained importance in the recent years in the modeling of certain types of discrete events systems. In view of the few solutions available in the literature for the causation of controllers, this text aims to propose a new technique of causation that is more efficient and robust. In addition, through the development of a realistic simulator of a plant, we also seek in this work the validation of this new technique and to test the controller in situations of perturbations.

Keywords— Discrete Event Systems, Max-Plus, Petri Nets, Causalization, Simulator

Resumo— A teoria de sistemas lineares na chamada *Álgebra Max-Plus* tem ganhado notório destaque nos últimos anos na modelagem de certos tipos de sistemas a eventos discretos. Em face às poucas soluções disponíveis na literatura para a causalização de controladores, este texto tem como objetivo propor uma nova técnica de causalização mais eficiente e robusta. Além disso, através do desenvolvimento de um simulador realista de uma planta, busca-se também neste trabalho a validação dessa nova técnica e o teste do controlador em situações de perturbações.

Palavras-chave— Sistemas a Eventos Discretos, Max-Plus, Redes de Petri, Causalização, Simulador

1 Introdução

Uma teoria relativamente nova, baseada na álgebra Max-Plus, encontra-se em amplo desenvolvimento e permite o controle de alguns sistemas a eventos discretos, mais especificamente Grafos a Eventos Temporizados (GETs, veja (Baccelli et al., 1992), uma revisão da literatura de aplicações pode ser vista em (Gonçalves et al., 2017)). Tendo como $x[k]$ o vetor dos tempos de disparo dos eventos de estado pela k-ésima vez e $u[k]$ o mesmo para os eventos de controle, esses sistemas têm uma representação dada pela seguinte equação:

$$x[k+1] = Ax[k] \oplus Bu[k] \tag{1}$$

onde os produtos e somas matriciais são realizados na álgebra Max-Plus, melhor detalhada na Seção 2 deste trabalho.

Devido à estrutura peculiar da Álgebra Max-Plus, poucos resultados da desenvolvida teoria para sistemas na álgebra convencional podem ser transpostos para essa nova álgebra. Por exemplo, os conceitos de controlabilidade e observabilidade ainda não são bem compreendidos, carecendo maior aprofundamento.

Em um dos problemas relacionados à Equação (1), estamos procurando por um controlador $u[k] = f(x[k])$ que garanta a convergência em regime permanente para um conjunto $\mathbb{S}$, que representa características desejadas, e sua permanência nesse. Em muitos trabalhos como (Katz, 2007; Maia et al., 2011; Amari et al., 2012; Gonçalves et al., 2016; Kim and Lee, 2016; Gonçalves et al., 2017) esse controlador é obtido como uma realimentação linear de estados, ou seja, $u[k] = Fx[k]$ para uma matriz fixa F. Em relação a essas técnicas temos dois problemas práticos importantes que ainda não foram amplamente discutidos na literatura e serão o foco deste trabalho:

- Causalização - um dos problemas encontrados na implementação de um controlador, expressado pela lei $u[k] = Fx[k]$, é quando o controlador é *não causal*, isso é, demanda em um determinado momento o conhecimento de um valor que só será obtido no futuro. Portanto, há a necessidade de técnicas de causalisação desses controladores, isso é, transformar o controlador em algo equivalente que possa ser implementado. Do conhecimento dos autores, o trabalho (Maia et al., 2013) é o único na literatura que trata de técnicas de causalização para controladores por realimentação de estado genéricos. Neste trabalho, iremos abordar e discutir uma nova técnica e comprovar que ela é mais robusta na medida que leva menos eventos para convergir ao conjunto desejado.
- Robustez - Poucos trabalhos (veja (Gonçalves et al., 2017) para uma revisão bibliográfica) discutem o que acontece com o controlador quando ocorrem perturbações no sistema. Por exemplo, desvios dos parâmetros nominais da planta como o tempo de completude de uma máquina. Como perturbações são comuns na prática, é muito importante saber se o controlador vai conseguir continuar controlando mesmo se depois de um determinado número de eventos houverem pequenas perturbações. Nesse sentido, para este trabalho, foi desenvolvido um programa em

C# que simula uma planta de esteiras presente na universidade de Angers-França (veja (Gonçalves, 2014)) de modo a testar o controlador e a técnica de causalização proposta em uma situação com perturbações mais realistas. Mais detalhes desse simulador serão dados posteriormente neste trabalho.

Desta forma, este artigo tem como principal foco desenvolver uma nova técnica de causalização e validá-la através de um simulador. É importante mencionar que a aplicação prática desse tipo de controlador em um sistema real, pelo conhecimento dos autores, foi feita pouquíssimas vezes, sendo, do conhecimento dos autores, o trabalho de (Gonçalves, 2014) pioneiro. Assim, este trabalho vem agregar nesse sentido, uma vez que a implementação do controlador em um ambiente bem próximo do real (simulador realista) permite avaliar as dificuldades e características de tal implementação.

2 Notações Matemáticas

A *Algebra Max-Plus*, também conhecida como *Álgebra Tropical*, é um exemplo de uma estrutura algébrica denominada *dióide* ou semi-anel idempotente (veja (Baccelli et al., 1992)). Define-se $\varepsilon = -\infty$, e também como ε um vetor de tamanho apropriado formado apenas por esses elementos (vetor nulo). Denota-se por $\mathbb{Z}_{max}$ o conjunto $\mathbb{Z} \cup \{\varepsilon\}$, onde $\mathbb{Z}$ é o conjunto dos números inteiros. Para os elementos $a, b \in \mathbb{Z}_{max}$ definem-se operações $\oplus$ e $\otimes$ como:

$$a \oplus b = max(a, b) \quad e \quad a \otimes b = a + b \tag{2}$$

onde $\oplus$ é o máximo e $\otimes$ é a soma tradicional. É usual omitir o símbolo de $\otimes$ para o produto. Assim, xy, no contexto da álgebra Max-Plus, é lido como "x vezes y". Também consideramos os produtos e somas matriciais no contexto Max-Plus, onde trocamos as somas e produtos nessas operações por máximos e somas, respectivamente.

3 Causalização do Controlador

3.1 O problema e técnica anterior

Considere o sistema Max-Plus

$$x[k+1] = Ax[k] \oplus Bu[k] \tag{3}$$

com $A \in \mathbb{Z}_{max}^{n \times n}$, $B \in \mathbb{Z}_{max}^{n \times n}$, em que $x_i[k]$ e $u_i[k]$ representam o tempo do k-ésimo disparo da i-ésima transição de estado e controlador, respectivamente. O objetivo é encontrar um controlador $u[k]$ para fazer com que o estado $x[k]$ convirja e fique em um conjunto $\mathbb{S}$ descrito como

$$\mathbb{S} = \{x \mid Ex = Dx\} \tag{4}$$

em que $E, D, \in \mathbb{Z}_{max}^{r \times n}$ são matrizes dadas.

Mostra-se que sob certas condições, existe um $F \in \mathbb{Z}_{max}^{m \times n}$ tal que o controlador $u[k] = Fx[k]$ resolve o problema proposto. Existem várias formas de obter esse controlador, como verificado em alguns trabalhos como (Katz, 2007; Maia et al., 2011; Amari et al., 2012; Gonçalves et al., 2016; Kim and Lee, 2016; Gonçalves et al., 2017), mas é no artigo (Gonçalves et al., 2017) que obtemos condições matemáticas que são sempre necessárias (se violar as equações não há solução para o problema de controle) e suficientes (se a equação tiver solução, o problema de controle tem solução) para uma ampla gama de problemas. É também o primeiro trabalho, do conhecimento do autores, a garantir propriedades matemáticas de robustez para o controlador. O controlador desenvolvido garante convergência para o conjunto $\mathbb{S}$ a partir de *qualquer* condição inicial $x[0]$ em um número finito de eventos k. Será essa a técnica utilizada no presente trabalho.

Após o cômputo do controlador deve-se verificar se F é uma *matriz causal*, ou seja, todos os seus elementos são ≥ 0 ou ε. Caso contrário uma lei de controle do tipo $u[k] = Fx[k]$ pode exigir predições. Para mostrar essa situação, tomemos como exemplo um estado e controle, x, u respectivamente escalares e a lei de controle $u[k] = (-2)x[k]$, gerada pela matriz $F = (-2)$ (não causal). Essa lei de controle gera a seguinte regra: dispare u pela k-ésima vez 2 unidades de tempo *antes* de x disparar pela k-ésima vez. Nesse caso u nunca será disparado, já que quando ocorrer o disparo de x o tempo de disparo de u já terá se passado. Ou seja, para que essa regra fosse implementada fisicamente, para o disparo de u, seria necessário *prever* qual o tempo de disparo de x.

Sendo F uma matriz *não-causal*, uma forma de tornar implementável o controlador seria usar uma *técnica de causalização*. Do conhecimento dos autores, na literatura atual temos apenas uma técnica utilizada, (Maia et al., 2013), para a causalização de controladores por realimentação de estados genéricos. Ela funciona da seguinte maneira: se existe uma lei de controle não realizável $u_{nc}[k] = F_{nc}x[k]$ (onde a matriz F_{nc} é *não-causal*), considerando que o controle fosse aplicado na planta:

$$x[k] = (A \oplus BF_{nc})x[k-1] \tag{5}$$

então temos que a lei de controle pode ser reescrita como:

$$u_{nc}[k] = F_{nc}(A \oplus BF_{nc})^m x[k-m] \tag{6}$$

para um m qualquer e todo $k \geq m$. É demonstrado em (Maia et al., 2013) que, incrementando-se m quantas vezes preciso, é possível fazer com que $F_{mc} = F_{nc}(A \oplus BF_{nc})^{m-1}$ seja causal. É importante ressaltar que temos que "arbitrar" con-

dições iniciais $x[k]$ para $k = -m, -m+1, ..., -1$. Arbitramos aqui todas elas como ε.

Como é possível verificar em (6), essa técnica utiliza-se de todas as informações de estado atrasadas, o que pode tornar o controlador menos robusto. Visando dar mais robustez ao controlador, este trabalho propõe uma nova técnica de causalização que utiliza-se, além de informações atrasadas, de informações atuais dos estados. Ela será demonstrada a seguir.

3.2 Técnica de Causalização Proposta

Toda matriz não-causal pode ser desmembrada em sua parte causal e não causal. Como exemplo, suponhamos a matriz H, não causal:

$$H = \begin{bmatrix} -1 & 2 \\ \varepsilon & 3 \end{bmatrix}.$$

Definindo-se um operador causal $[.]_c$ que irá desmebrar a matriz com sua parte causal (todos os elementos ≥ 0 ou iguais a ε) e o operador não-causal $[.]_n$ que irá desmebrar a matriz com sua parte não-causal (todos os elementos < 0 ou iguais a ε), temos:

$$[H]_c = \begin{bmatrix} \varepsilon & 2 \\ \varepsilon & 3 \end{bmatrix}, \; [H]_n = \begin{bmatrix} -1 & \varepsilon \\ \varepsilon & \varepsilon \end{bmatrix}$$

de forma que $[H]_c \oplus [H]_n = H$.

Seja um sistema como em (3) e um controlador $u[k] = Fx[k]$, possivelmente não causal pois F pode ser não causal. Temos também a matriz de malha fechada $M = (A \oplus BF)$. Note em (3) que as sequências $x_i[k], \forall i$ são não-decrescentes e, portanto, $A \succeq I$. Deste modo, é imediato observar que $M = (A \oplus BF) \succeq 0$. Da mesma forma que na técnica anterior (Maia et al., 2013) é necessário que M seja uma matriz irredutível com autovalor positivo. Então, separando-se F em sua parte causal $[F]_c$ e não causal $[F]_n$, temos:

$$u[k] = [F]_c x[k] \oplus [F]_n x[k]. \tag{7}$$

Note que a parte $[F]_c x[k]$ do controlador é implementável, pois não depende de previsões do futuro. A parte $[F]_n x[k]$ não, ao menos que ela seja uma matriz toda de ε. Se for o caso dela ser ε, podemos parar. Caso não, note que assumindo que o controlador $u[k] = Fx[k]$ foi implementado temos

$$\begin{aligned} x[k] &= (A \oplus BF)x[k-1] \\ x[k] &= Mx[k-1]. \end{aligned} \tag{8}$$

Então substituindo (8) somente na parte não causal de (7), temos:

$$u[k] = [F]_c x[k] \oplus ([F]_n M)x[k-1]. \tag{9}$$

Note que a matriz $([F]_n M)$ pode ter alguns elementos ≥ 0, e de fato, tende a ter mais elementos positivos, uma vez que M tem raio espectral ≥ 0. Separando a parte causal e não causal dessa matriz teremos:

$$\begin{aligned} u[k] &= [F]_c x[k] \oplus [([F]_n M)]_c x[k-1] \\ &\quad \oplus [([F]_n M)]_n x[k-1]. \end{aligned} \tag{10}$$

Note que agora temos os termos: $[F]_c x[k]$ e $[([F]_n M)]_c x[k-1]$ que são implementáveis pois são causais. O termo $[([F]_n M)]_n x[k-1]$ não é causal. Mas temos através de (8) que: $x[k-1] = Mx[k-2]$. Então, substituindo na parte não-causal de (10), temos:

$$\begin{aligned} u[k] &= [F]_c x[k] \oplus [([F]_n M)]_c x[k-1] \\ &\quad \oplus [([([F]_n M)]_n M)]_c x[k-2] \\ &\quad \oplus [([([F]_n M)]_n M)]_n x[k-2]. \end{aligned} \tag{11}$$

E assim, continuamos com esse procedimento. Temos duas situações possíveis: a parte não-causal pode ser eventualmente ε ou isso pode nunca acontecer. Se acontecer, então nosso controlador será causal e da forma:

$$u[k] = \bigoplus_{i=0}^{s} F_i x[k-i] \tag{12}$$

com matrizes F_i causais e um s finito, em que:

$$\begin{aligned} F_0 &= [F]_c \\ F_1 &= [([F]_n M)]_c \\ F_2 &= [([([F]_n M)]_n M)]_c \\ F_3 &= [([([([F]_n M)]_n M)]_n M)]_c \; \ldots \end{aligned}$$

e assim por diante. Entretanto, a sequência F_n pode nunca terminar. Tome como exemplo

$$F = \begin{bmatrix} -1 & \varepsilon \end{bmatrix}, \; M = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}.$$

O leitor pode verificar que $[F]_n = F$ e $[FM]_n = F$. Então $F_0 = [F]_c = \varepsilon$ e $F_1 = F_2 = F_3 = ...$ Nesse caso, a sequência nunca termina. Sobre esse fenômeno, enunciaremos dois resultados sem prova. Esses resultados serão detalhados em trabalhos futuros:

- Embora possa não terminar, a sequência F_n é cíclica após um transiente, isso é, existem inteiros τ e σ tal que $F_{\tau+k\sigma} = F_\tau$ para todo k. Assim, só existem matrizes novas até $F_{\tau+\sigma-1}$.
- Nesse caso, como os estados são naturalmente não decrescentes ($x[k+1] \geq x[k]$), podemos desconsiderar sem nenhuma perda as matrizes após $F_{\tau+\sigma-1}$ e implementar o controlador com as demais, como se a sequência tivesse terminado.

Em resumo, diante às duas técnicas de causalização apresentadas, temos evidências, que serão apresentadas neste trabalho, que a técnica de (Maia et al., 2013) tende a ser menos robusta se

comparada com a técnica proposta neste trabalho. Essa dedução é induzida comparando-se as equações dos controladores para as duas técnicas. Na equação para o controlador utilizando a técnica de (Maia et al., 2013), representada em (6), podemos verificar que todas as informações utilizadas são atrasadas. Já pela equação (12), que representa o controlador para a técnica proposta, podemos verificar que além de informações atrasadas o controlador utiliza-se de informações mais recentes, e temos evidências, que serão apresentadas neste trabalho, que ela proporciona uma resposta mais rápida do controlador quando o sistema é perturbado. Os testes e conclusões para essas duas técnicas serão apresentados detalhadamente na Seção 5 deste trabalho.

4 Simulador

A implementação do simulador proposto para este trabalho baseia-se em um sistema real localizado no *Laboratório Angevin de Pesquisa em Engenharia de Sistemas (LARIS)* da Universidade de Angers-França. (ver (Gonçalves, 2014)). A Figura 1 mostra o desenho esquemático do sistema. Esse sistema é composto de dois circuitos independentes, dez botões (B1 a B10) e dez sensores de proximidade (um sensor um pouco antes de cada botão). O circuito superior e inferior possuem três paletes cada. Os paletes no circuito superior se movem no sentido horário enquanto os do circuito inferior se movem no sentido anti-horário. Os botões são programados de modo que, quando acionados (isto é, o botão está abaixado e o palete pode mover), eles voltam automaticamente (para cima) no tempo necessário para a passagem de apenas um palete por vez. Para cada trecho entre dois botões sucessivos (por exemplo, o trecho B1 $\rightarrow$ B2), existe uma capacidade associada de paletes. Para que um dado botão do sistema dispare são necessárias três condições satisfeitas:

(I) deve haver pelo menos um palete esperando um pouco antes do botão.
(II) deve haver pelo menos um espaço vazio no trecho para onde o palete está se movimentando (conforme a capacidade do trecho).
(III) deve haver uma *ficha de controle*, o que representa uma ação externa do sistema.

Uma diferença importante que deve ser mencionada é que na planta real não existem alguns sensores utilizados no simulador e que são necessários para fazer controle por realimentação de estado $u[k] = Fx[k]$. Esses sensores virtuais medem o tempo de chegada em cada posição do palete antes de cada botão do sistema, onde não existem sensores físicos instalados. Por exemplo, no trecho $B1 \rightarrow B2$, que possui três paletes de capacidade,

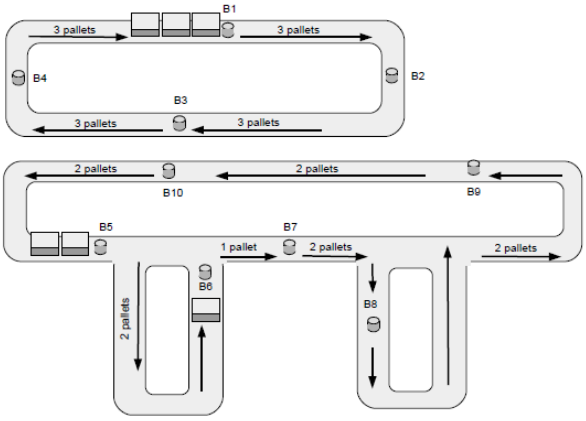

Figura 1: Figura esquemática do sistema de transporte localizado no *Laboratório Angevin de Pesquisa em Engenharia de Sistemas (LARIS)* da Universidade de Angers-França.

cada palete pode se situar em três posições diferentes antes do botão $B2$, que são:

- Posição 1 - palete parado em primeiro lugar na linha um pouco antes de $B2$.
- Posição 2 - palete parado em segundo lugar na linha um pouco antes de $B2$.
- Posição 3 - palete parado em terceiro lugar na linha um pouco antes de $B2$.

Nesse caso, o tempo de chegada na Posição 1 é medido por um sensor físico (sensor de proximidade) instalado um pouco antes do botão B2. Já os tempos de chegada nas Posições 2 e 3 são medidos através de sensores virtuais programados no simulador. Na planta real, onde esses sensores virtuais não existem, é necessário fazer uma realimentação por saída (apenas alguns dos estados são medidos) ou um observador (ver (Gonçalves et al., 2014)). Essas situações ficam como oportunidade de trabalhos futuros, uma vez que neste trabalho usaremos os sensores virtuais (que, novamente, não existem na planta real) e portanto faremos uma realimentação de estados $u[k] = Fx[k]$.

Por ser de natureza puramente mecânica, a planta está sujeita à perturbações, especificamente o agarramento do movimento dos paletes na esteira, tipicamente ocasionadas pelo desgaste natural de seus componentes e a falta de lubrificação. Por isso é importante verificar o comportamento do controlador quando elas acontecem, o que será feito posteriormente nos testes com o simulador desenvolvido.

O simulador foi desenvolvido na linguagem C# com uma amigável interface com o usuário (ver Figura 2) e com as mesmas configurações da planta real. Na tela de interface podemos verificar o desenho dos dois circuitos independentes. No canto superior esquerdo o usuário pode definir

alguns ajustes importantes, como a velocidade de simulação e a inclusão de perturbações com sua respectiva taxa. Além disso, o usuário tem a opção de importar as matrizes do controlador e exportar os dados de estado ($x[k]$) e das entradas de controle ($u[k]$).

O desenvolvimento de um simulador para um sistema dessa natureza é de grande utilidade, pois testes realizados na planta real nem sempre são tão simples de se fazer, primeiramente devido a sua localização (Angers - França) e também devido a indisponibilidade dos engenheiros que manuseiam a planta. Já o simulador desenvolvido permite a simulação a qualquer momento com resultados rápidos (por permitir o ajuste da velocidade de simulação) e bastante fiéis à planta real. De fato, como as características de interesse da planta (movimento dos paletes, colisões entre eles, colisões com os botões, detecção da chegada de paletes, etc...) não têm uma natureza física muito complexa (processos químicos, térmicos, elétricos, etc...), é possível desenvolver um simulador bastante realista que é suficiente para a validação das técnicas de controle enquanto traz comodidade.

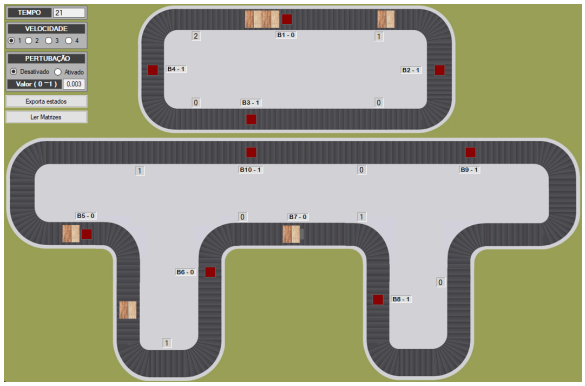

Figura 2: *Screenshot* do simulador desenvolvido.

5 Testes e Resultados

Para a realização dos testes, primeiramente o modelo Max-Plus do sistema foi levantado na forma da equação (3), sendo $x[k]$ o vetor dos tempos de disparo dos eventos de estado pela k-ésima vez e $u[k]$ o mesmo para os eventos de controle (a *ficha de controle*, conforme mencionado na seção anterior, para cada botão). No total, o sistema é composto por 33 eventos de estado e 10 eventos de controle, um para cada botão. O modelo em grafos a eventos temporizados pode ser visto em (Gonçalves, 2014), Capitulo 6, embora os tempos aqui utilizados para cada disparo são diferentes.

O sistema é programado para fazer uma sincronização forçada entre os circuitos superior e inferior nos botões B3 e B10 (ver Figura 1) de forma que B3 apenas dispara se existir pelo menos um palete presente em B10 e B10 apenas dispara se houver pelo menos um palete presente em B3, ou seja, os botões B3 e B10 são sincronizados.

Colocamos então uma especificação que não é garantida se não houver um controle: que se um palete chega um pouco antes de B3 ele não espere mais do que 20 (vinte) unidades de tempo até que um palete chegue um pouco antes de B10 e vice-e-versa. Desta forma espera-se que o controlador consiga atender a essas restrições. Matematicamente, considerando $x_6[k]$ o estado de chegada de um palete um pouco antes de B3 e $x_{22}[k]$ o estado de chegada de um palete um pouco antes de B10 e atendendo a especificação de que um palete chegando um pouco antes do botão B3 não aguarde mais do que 20 (vinte) unidades de tempo por um palete antes de B10, temos o seguinte equacionamento para o circuito superior:

$$\begin{aligned} x_6[k] - x_{22}[k] &\leq 20 \text{ ou} \\ x_6[k] \oplus 20x_{22}[k] &= 20x_{22}[k]. \end{aligned} \tag{13}$$

Para o circuito inferior, da mesma forma, considerando-se agora que um palete chegando um pouco antes do botão B10 não aguarde mais do que 20 (vinte) unidades de tempo por um palete antes de B3:

$$\begin{aligned} x_{22}[k] - x_6[k] &\leq 20 \text{ ou} \\ x_{22}[k] \oplus 20x_6[k] &= 20x_6[k]. \end{aligned} \tag{14}$$

Verificamos, portanto, que as especificações equacionadas em (13) e em (14), podem ser escritas na forma $Ex[k] = Dx[k]$, conforme Equação (4). De posse das matrizes do sistema modelado, juntamente com as matrizes de restrição, o controlador foi calculado conforme proposto em (Gonçalves et al., 2017) e causalizado conforme as duas técnicas de causalização, a de (Maia et al., 2013) e a proposta neste artigo .

Após implementação do controlador no simulador, alguns testes foram realizados de forma a verificar a eficiência da técnica de causalização de (Maia et al., 2013) comparada com a técnica de causalização proposta neste trabalho. Esses testes consistiram em verificar o atendimento às especificações do sistema, (representadas em (13) e em (14)), em malha aberta e em malha fechada, em todos os casos com perturbação. Dois tipos de perturbações foram simuladas, sendo a primeira automática, com uma taxa configurada pelo usuário de 0.5% e a segunda manualmente, que consistiu em parar um palete um pouco depois do botão B4 por 400 unidades de tempo nos intervalos 1500, 3400 e 5300 unidades de tempo. A janela de simulação para todos os testes foi de 6500 unidades de tempo. A Figura 3 mostra os resultados para todos os testes realizados, onde o eixo das ordenadas indica a diferença de tempo (em segundos) entre a chegada de um palete um pouco antes do botão $B3$ (representada pelo estado $x_{6)}$) e a chegada de um palete um pouco antes do botão $B10$

(representada pelo estado x_{22}) e o eixo das abscissas indica o k-ésimo evento. Note que o eixo das ordenadas está definido em escala logarítmica para facilitar a visualização.

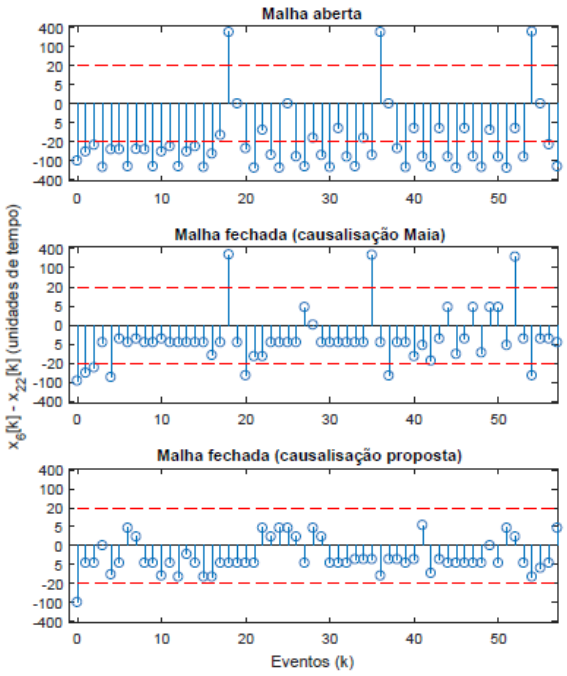

Figura 3: Testes com perturbação realizados em malha aberta, malha fechada com a causalização (Maia et al., 2013) e malha fechada com a causalização proposta

Como exemplo na Figura 3, considerando o sistema em malha aberta, se tomarmos o 20º evento, temos que a diferença de chegada de um palete um pouco antes de $B3$ e a chegada de um palete um pouco antes de $B10$ é de -20 segundos, ou seja quando um palete chega em $B3$ somente 20 segundos após um palete chega em $B10$.

Podemos verificar que em malha aberta (sem controlador) o sistema é instável, não atendendo à especificação conforme já era esperado. No teste em malha fechada usando a causalização de (Maia et al., 2013) o sistema se controla após o transiente inicial, mas quando ocorre a perturbação manual, o sistema leva três disparos subsequentes para estabilizar o controle. Isso ocorre pelo fato desse modelo de causalização, aplicado neste problema, utilizar de informações passadas de 2 disparos anteriores. Já no teste utilizando a técnica de causalização proposta, podemos verificar que o sistema se controla após o transiente inicial e se mantém controlado mesmo após as perturbações automática e manual. Isso ocorre pelo fato desse modelo de causalização utilizar além de informações atrasadas, de informações atuais dos estados, mostrando uma robustez maior desse controlador se comparado com o controlador utilizando a técnica de (Maia et al., 2013).

6 Conclusões

Neste trabalho foi proposta uma nova técnica de causalização para controladores por realimentação de estados genéricos. Diferente da técnica existente na literatura, publicada no trabalho de (Maia et al., 2013), a técnica proposta utiliza-se, além de informações atrasadas, de informações presentes dos estados. Isso oferece maior robustez ao controlador, na medida que leva menos eventos para convergir ao conjunto desejado. O desenvolvimento de um simulador realista de uma planta existente permitiu-nos realizar testes em malha aberta e fechada com perturbações e validar a eficiência dessa nova técnica de causalização. Espera-se em trabalhos posteriores, discutir outras técnicas que leve em consideração também, além de informações passadas e presentes, de informações futuras dos estados, esperando-se assim obter um controlador ainda mais sensível com respostas mais rápidas às perturbações.

Agradecimentos

Os autores agradecem ao CNPQ, CAPES e à FAPEMIG pelo financiamento concedido à esta pesquisa.

Referências

Amari, S., Demongodin, I., Loiseau, J. J. and Martinez, C. (2012). Max-plus control design for temporal constraints meeting in timed event graphs, *IEEE Transactions on Automatic Control* **57**(2): 462–467.

Baccelli, F., Cohen, G., Olsder, G. and Quadrat, J. (1992). *Synchronization and Linearity*, Wiley, New York.

Gonçalves, V. M. (2014). *Tropical Algorithms for Linear Algebra and Linear Event-invariant Dynamical Systems*, PhD thesis, Universidade Federal de Minas Gerais, UFMG, Brazil.

Gonçalves, V. M., Maia, C. A. and Hardouin, L. (2016). On the steady-state control of timed event graphs with firing date constraints, *Automatic Control, IEEE Transactions on* **61**(8): 2187 – 2202.

Gonçalves, V. M., Maia, C. A. and Hardouin, L. (2017). On max-plus linear dynamical system theory: The regulation problem, *Automatica* **75**: 202–209.

Gonçalves, V. M., Maia, C. A., Hardouin, L. and Shang, Y. (2014). An observer for tropical linear event-invariant dynamical systems, *IEEE CDC-ECC, Los Angeles* .

Katz, R. D. (2007). Max-plus (A; B)-invariant spaces and control of timed discrete-event systems, *IEEE Transactions on Automatic Control* **52**(2): 229 – 241.

Kim, C. and Lee, T. E. (2016). Feedback control of cluster tools for regulating wafer delays, *IEEE Transactions on Automation Science and Engineering* **13**(2): 1189–1199.

Maia, C. A., Andrade, C. R. and Hardouin, L. (2011). On the control of maxplus linear system subject to state restriction, *Automatica* **47**(5): 988–992.

Maia, C. A., Hardouin, L. and Cury, J. E. R. (2013). Some results on the feedback control of max-plus linear systems under state constrains, *52nd IEEE Conference on Decision and Control*, pp. 6992–6997.