Redução do Makespan para Sistemas com Ponderações Temporais Usando um Algoritmo de Busca Clonal

Gabriel Laport Vargas * **Antonio Eduardo Carrilho da Cunha** **

* *Faculdade de Engenharia de Defesa, Instituto Militar de Engenharia, RJ, (e-mail: glvargas@ime.eb.br).*
** *Faculdade de Engenharia de Defesa, Instituto Militar de Engenharia, RJ (e-mail: carrilho@ime.eb.br)*

Abstract: The time-optimal control of Time-Weighted Systems (TWS) is an approach that can be used to makespan reduction of Computer-Integrated Manufacturing (CIM) Systems. Unfortunately, the complexity of the TWS algorithms turns even suboptimal solutions computationally intractable when dealing with realistic CIM systems with multiple components, stations, and different productions batches. In this paper, we propose some strategies to alleviate the complexity cost of makespan reduction computations using TWS. The strategies consist of: relaxing the need of symmetric imposed mutual exclusions relations between events; using a hashtable algorithm for makespan computations instead of a heap of pieces algorithm; avoiding computation of batch supervisors using a scalable synthesis method; and applying a Clonal Search heuristic for makespan reduction. With this, we obtain the reduction of the computational cost while maintaining the advantages of the TWS approach regarding the reduction of models by representing one task by only one event. The performance of the proposed algorithm is compared with some of the known efficient approaches by means of a realistic CIM example.

Resumo: O controle ótimo do tempo de Sistemas com Ponderações Temporais (SPT) é uma abordagem que pode ser usada para reduzir o makespan de Sistemas de Manufatura Integrada por Computador (CIM – Computer-Integrated Manufacturing). Infelizmente, a complexidade dos algoritmos SPT torna até mesmo soluções abaixo do ideal computacionalmente intratáveis ao lidar com sistemas CIM realistas com vários componentes, estações e lotes de produção diferentes. Neste artigo, propomos algumas estratégias para aliviar o custo computacional e diminuir a complexidade de cálculos de redução do makespan usando SPT. As estratégias consistem em: afrouxar a necessidade da simetria entre os eventos das relações de exclusões mútuas impostas; usar um algoritmo de hashtable para cálculos de makespan em vez de um algoritmo de pilha de peças; evitar a computação de supervisores de lote usando um método de síntese escalável; e aplicar uma heurística de Busca Clonal para redução de makespan. Com isso, obtemos a redução do custo computacional, mantendo as vantagens da abordagem SPT quanto à redução de modelos ao representar uma tarefa por apenas um evento. O desempenho do algoritmo proposto é comparado com algumas das abordagens eficientes conhecidas através de um exemplo realista de CIM.

Keywords: Discrete event systems, Time-weighted systems, Supervisory control, Makespan reduction, Manufacturing.

Palavras-chaves: Sistemas a Eventos Discretos; Sistemas com Ponderações Temporais; Controle Supervisório; Redução do Makespan; Fabricação

1. INTRODUÇÃO

Diversas abordagens para resolver problemas de controle de produção, como programação linear e multiobjetivo, são amplamente adotadas na Pesquisa Operacional. No entanto, o foco deste trabalho é a análise de abordagens no contexto de Sistemas a Eventos Discretos (SED). Os autores Ramadge e Wonham propuseram em (Ramadge and Wonham, 1989) a Teoria do Controle Supervisório (TCS) para SED. Com base nessa teoria, foram desenvolvidos Sistemas com Eventos Discretos Temporizados (Brandin and Wonham, 1994) e Sistemas com Intervalos de Tempo (Lin et al., 2019), além dos Sistemas com Ponderações Temporais (SPT) (Su et al., 2012).

Neste trabalho, a abordagem SPT proposta em Tuxi and da Cunha (2018) será comparada com as técnicas utilizadas em Alves et al. (2021) e Pena et al. (2021) que utilizam o TCS para controlar o sistema com segurança e posteriormente buscar o resultado com menor tempo de produção. Será proposta uma nova abordagem que permite o uso de SPT, herdando boas práticas utilizadas em Alves et al. (2021) e Pena et al. (2021), possibilitando a redução do makespan em um tempo computacional

viável, através do relaxamento da simetria na relação de exclusão mútua forçada. O exemplo da Pequena Fábrica (Cassandras and Lafortune, 2010) será usado para ilustrar a abordagem, e o problema de redução do makespan de um sistema CIM realista é aplicado para avaliar a complexidade computacional da abordagem proposta e compará-la com as demais.

Este artigo está organizado da seguinte forma. A seção 2 mostra conceitos preliminares e compara as teorias através de um exemplo. A Seção 3 apresenta a ideia conceitual para este artigo e a Seção 4 os principais resultados. Na Seção 5 aplicamos as contribuições da seção anterior em um estudo de caso e comparamos os resultados com três abordagens. Por fim, a conclusão e os comentários finais são apresentados na seção 6.

2. PRELIMINARES

Nesta seção, é apresentado o material de referência sobre controle supervisório de SED e SPT e redução de makespan. O leitor deve consultar Ramadge and Wonham (1989), Su et al. (2012) e Tuxi and da Cunha (2018) e as referências neles contidas para uma introdução detalhada. Também fornecemos um exemplo ilustrativo para comparar as duas abordagens.

2.1 Linguagem e Autômatos

Tanto no SED quanto no SPT, os eventos são associados a símbolos em um alfabeto Σ. Σ^* denota o conjunto de todas as cadeias de eventos (strings) de comprimento finito formadas pela justaposição de símbolos em Σ, mais a cadeia vazia ε. Dadas as strings $s, t \in \Sigma^*$, afirma-se que s é prefixo de t, denotado $s \leq t$, se existir $u \in \Sigma^*$ tal que $su = t$.

Uma linguagem em Σ é qualquer subconjunto de Σ^*. O fechamento de prefixo de uma linguagem $L \subseteq \Sigma^*$ é $\overline{L} = \{s \in \Sigma^* : (\exists t \in L)\ s \leq t\}$. Uma linguagem é fechada por prefixo se $\overline{L} = L$.

Um autômato é uma tupla $G = (X, \Sigma, \delta, x_0, X_m)$, onde Σ é o alfabeto, X é o conjunto de estados, $\delta : X \times \Sigma \to X$ é a função de transição, $x_0 \in X$ é o estado inicial e $X_m \subseteq X$ é o conjunto de estados marcados. A função de transição é parcial, e escrevemos $\delta(x, \sigma)!$ se δ for definido para $x \in X$ e $\sigma \in \Sigma$. Estendendo a função de transição para traços em Σ^*, definimos para um autômato a linguagem gerada $L(G) = \{s \in \Sigma^* : \delta(x_0, s)!\}$ e a linguagem marcada $L_m(G) = \{s \in \Sigma^* : \delta(x_0, s) \in X_m\}$. A linguagem gerada é fechada por prefixo, e quando $\overline{L_m(G)} = L(G)$, o autômato é dito não bloqueante. Dados os estados $x, y \in X$, x é alcançado a partir de y quando existe uma string $s \in \Sigma^*$ tal que $y = \delta(x, s)$. Um estado que é alcançado a partir do estado inicial e que atinge um estado marcado é dito *trim*. Dado um autômato G a operação $trim(G)$ dá o subautômato de G onde todos os estados são trim. G é dito trim quando $trim(G)$ e G são isomórficos e, neste caso, G também é não bloqueante. Outra operação relevante em autômatos é a composição paralela. A composição paralela dos autômatos G_i, com $i = 1 \ldots n$, é o autômato $G = \|_{i=1}^{n} G_i$ que representa a evolução paralela dos autômatos n, onde um símbolo comum a vários autômatos só pode ser executado, caso todos os autômatos que contêm este símbolo possam executá-lo simultaneamente.

2.2 Controle de supervisão e otimização de tempo de SED

SED são sistemas dinâmicos de estado discreto cuja evolução é conduzida por eventos com ocorrência instantânea. No controle supervisório do SED, o sistema a ser controlado, ou planta, é representado pelo autômato G. Os eventos em um SED são representados pelo alfabeto do autômato e os estados em um SED são representados pelos estados dos autômatos. As ocorrências espontâneas de eventos que impulsionam a evolução do SED são representadas pelas transições dos autômatos. O autômato da planta G pode ser obtido pela composição paralela de autômatos modelando o comportamento dos componentes da planta.

O mecanismo de controle é definido pelo particionamento do conjunto de eventos da planta em subconjuntos controláveis e incontroláveis, ou seja, $\Sigma = \Sigma_c \cup \Sigma_u$. Os eventos controláveis são Σ_c, enquanto os eventos incontroláveis são Σ_u. O outro componente do TCS do SED é o supervisor, que também é representado por um autômato S. O comportamento em malha fechada é representado pela composição paralela $S||G$. O problema de controle de supervisão (SCP) é definido por: dado G com um mecanismo de controle e uma linguagem desejada $K \subseteq L_m(G)$, encontre um supervisor S para G tal que $\emptyset \neq L_m(S||G) \subseteq K$. Em geral, K é obtido definindo primeiro um autômato que representa o comportamento em malha fechada desejado, digamos E, e compondo-o com G, obtendo um autômato $H = E||G$, tal que $L_m(H) = K$. O primeiro resultado é que: existe tal supervisor S se K é controlável em relação a $L(G)$ e Σ_u, dado pela fórmula $\overline{K}\Sigma u \cap L(G) \subseteq \overline{K}$. Além disso, se K não é controlável, sempre é possível encontrar uma linguagem $supC(K) \subseteq K$, denominada suprema sublinguagem controlável de K, obtida cortando estados ruins e transições de H. A linguagem $supC(K)$ é tal que todas as possíveis sublinguagens controláveis de K estão contidas nela e, se não estiver vazia, a solução do problema de controle de supervisão (SCP) é o autômato Z, resultante de cortes sucessivos de H, tal que $L_m(Z) = supC(K)$.

2.3 Controle de supervisão e otimização de tempo de SPT

Um *Sistema com Ponderações Temporais* (SPT) é uma tripla (G, f, h) onde, $G = (X, \Sigma, \delta, x_0, X_m)$ é um autômato, $f : X \times \Sigma \to R^+$ é uma função de peso, e $h \subseteq \Sigma \times \Sigma$ é uma relação de exclusão mútua intrínseca.

O autômato G é interpretado da maneira padrão, como gerando e marcando strings em Σ^*. Dado um evento $\sigma \in \Sigma$, $t_\sigma \in R^+ \cup \{0\}$ denota o momento inicial de disparo de σ. A ordem dos eventos em uma string $s \in L(G)$ denota a ordem de seus respectivos momentos iniciais de disparo. Por exemplo, para $a, b \in \Sigma$, o traço $ab \in L(G)$ é interpretado como $t_a \leq t_b$.

Ao contrário do controle SED, onde uma transição é instantânea, a função de peso define o tempo para que a transição seja concluída. Estendendo o conceito, pode-se definir o tempo de execução de um evento em G pela função peso, sendo dependente de qual transição de G o evento figura. Dado $s \in \Sigma^*$ e $\sigma \in \Sigma$ tais que $s\sigma \in L(G)$,

a duração de σ é dada por $f(\delta(x_0), s), \sigma)$. Neste caso, a execução de σ começa em t_σ e dura até $t_\sigma + f(\delta(x_0, s), \sigma)$.

Finalmente, a relação de exclusão mútua h é uma relação binária, reflexiva e simétrica definida de tal forma que se $(\sigma, \sigma') \in h$, suas execuções são mutuamente exclusivas, ou seja, se uma estiver em execução, o outro evento não será disparado e vice-versa.

Um *requisito* para um SPT (G, f, h) é um par (E, h_E), onde E é um autômato com o alfabeto $\Delta \subseteq \Sigma$, chamado *especificação*, e $h_E \subseteq \Delta \times \Delta$ é a *relação de exclusão mútua forçada*. A especificação E é interpretada como o comportamento desejado para (G, f, h), como nas outras abordagens de controle SED. A exclusão mútua forçada h_E é uma relação binária simétrica que expressa a desejada intercalação de execuções de eventos que não podem ser expressas em E. Enquanto a relação de exclusão mútua intrínseca h é uma propriedade do sistema, a relação de exclusão mútua imposta h_E é uma propriedade desejada para o sistema, a ser imposta por um supervisor.

Para garantir a existência de um supervisor que imponha um determinado requisito a um SPT, as noções de controlabilidade da linguagem, como na abordagem SED, e harmonia da linguagem são apresentadas em Su et al. (2012). Dado um SPT (G, f, h) e um requisito (E, h_E), a suprema linguagem controlável e harmoniosa de (G, f, h) sob (E, h_E), digamos $\sup CH(G, f, h, E, h_E)$, é a sublinguagem controlável e harmoniosa menos restritiva de $K = L_m(G) \| L_m(E)$ que pode ser aplicada pelo controle de supervisão em (G, f, h).

Seguindo as premissas simplificadoras de Tuxi (2018), consideramos que todos os eventos são controláveis e que suas durações dependem apenas do evento em si e não das transições de G onde ocorrem. Esta é uma suposição razoável e intuitiva quando se trata de componentes padrão de um SPT, como máquinas que podem executar diferentes processos, operações ou tarefas, cada uma associada a um evento controlável que representa o comando para iniciar a tarefa. O feedback da conclusão da referida tarefa está implícito no tempo necessário para completar a tarefa (o peso do evento). Além de ser uma suposição de modelagem intuitiva, existem alguns ganhos computacionais relevantes. A linguagem de destino $K = L_m(G) \| L_m(E)$ torna-se controlável e harmonioso, ou seja, $K = \sup CH(G, f, h, E, h_E)$. Como consequência, o autômato trim $Z = trim(G \| E)$ torna-se um reconhecedor de K.

2.4 Exemplo da pequena fábrica

O exemplo da pequena fábrica consiste em duas máquinas com um buffer unitário no meio. Para que um produto seja finalizado, ele deve seguir a sequência: primeiro, ele vai para a Máquina 1, depois para o buffer e, por fim, para a Máquina 2. Este exemplo nos permite analisar o fluxo de trabalho de uma pequena linha de produção nas abordagens TCS e SPT. Primeiramente, será explorado o problema de coordenação e, em seguida, será avaliado o tempo ótimo. Neste artigo, o tempo ótimo é o menor tempo de produção para um lote de produtos, ou seja, o menor makespan.

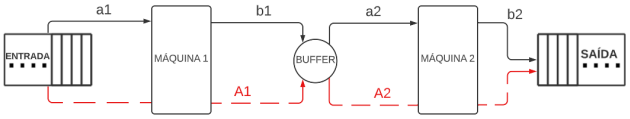

Figura 1. Diagrama da Pequena Fábrica

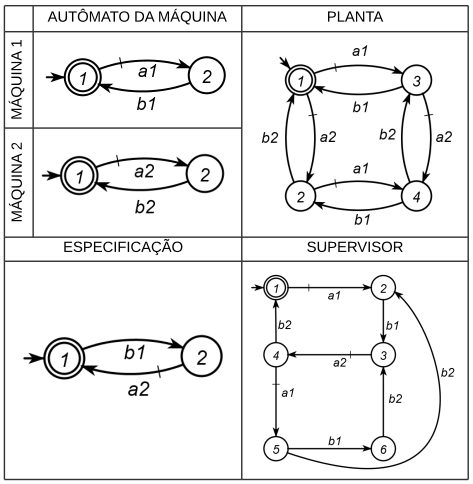

Figura 2. Modelos na TCS

Tabela 1. Valores dos Eventos

Abordagem	Eventos			
TCS	Delays			
	a1	b1	a2	b2
	0	5	0	10
SPT	Duração			
	A1		A2	
	5		10	

Na TCS, o exemplo pode ser observado na Figura 1. A máquina 1 possui um evento controlável a_1 que representa o início do processo e um evento não controlável b_1 que marca o término do processo; a mesma lógica se aplica à máquina 2 em relação a a_2 e b_2. Na Figura 1, todas as transições são representadas por setas pretas de traço contínuo. O autômato de cada máquina, a especificação e o supervisor são mostrados na Figura 2.

Na teoria SPT, os modelos se tornam mais diretos porque consideramos apenas os eventos controláveis, representados pelas setas vermelhas tracejadas na Figura 1. A forma como observamos a passagem do tempo no SPT nos permite adotar essa simplificação na parte de modelagem do problema, conforme apresentado na Figura 3. No entanto, para manter as propriedades desejadas do sistema, precisamos definir as relações de exclusões mútuas, tais que $h = \{(A_1, A_1), (A_2, A_2)\}$ e $he = \{(A_1, A_2), (A_2, A_1)\}$. Os valores dos eventos em unidades de tempo (t.u.) estão descritos na tabela 1.

A primeira parte do problema é resolvida pelos supervisores da Figura 2 e Figura 3. Para o problema de tempo ótimo, será aplicada a abordagem apresentada em Pena et al. (2021) para TCS e em Tuxi and da Cunha (2018) para SPT. A Figura 4 permite comparar a passagem do tempo e os resultados em cada abordagem. No TCS, os eventos são discretos e, por isso, utiliza-se um par (controlável/não controlável), resultando em um makespan de

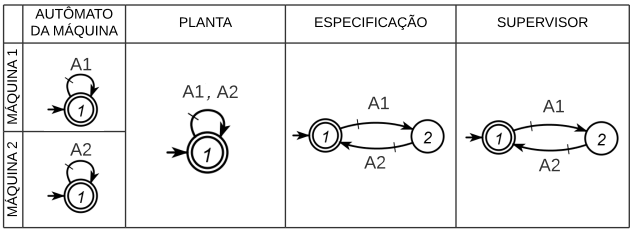

Figura 3. Modelos no SPT

35s. Para a parte SPT, a imagem se assemelha a um gráfico de Gantt, com eventos tendo uma duração, resultando em um makespan de 45s. Apesar de lidar com o mesmo problema, a última abordagem tem um resultado pior. Na próxima seção, será apresentada a ideia conceitual para melhorar a abordagem SPT em relação ao makespan.

3. IDEIA CONCEITUAL

Trabalhar com o controle de problemas industriais tende a crescer em complexidade ao longo do tempo. Para solucionar o problema de redução do makespan na Manufatura Integrada por Computador, composta por múltiplos componentes, estações e diferentes lotes de produção, é importante planejar as etapas para encontrar os resultados necessários em um tempo computacional compatível com a realidade industrial. A ideia é compor diferentes estratégias para que o método de controle tenha um supervisor com baixa complexidade, boa escalabilidade, e possa ser compilado em um único software.

Para a parte de modelagem, utilizamos o SPT porque o autômato com ponderação temporal nos permite utilizar a simplificação no modelo utilizando apenas um evento sem perder a análise do tempo. Esta técnica é usada em Tuxi (2018) para reduzir o número de estados e transições, portanto, a complexidade geral do problema.

A ferramenta utilizada para modelar o sistema e resolver o problema de busca foi *ULTRADES* (Alves et al., 2017). Esta ferramenta é uma biblioteca *C#* para modelagem, análise e controle de SED. A vantagem de utilizar uma única ferramenta é a praticidade durante a codificação e posteriormente a comunicação entre o modelo e o algoritmo de busca, dispensando a conversão através de linguagens como TCT e GRAIL.

Ao lidar com problemas complexos que ilustram cenários do mundo real, o tempo necessário para encontrar a melhor solução possível geralmente inviabiliza a aplicação. Para mitigar isso, ao invés de tentar resolver o problema de minimização do makespan, focamos na redução do makespan usando uma heurística com parâmetros de parada pré-definidos, que permite a escalabilidade dos lotes de produção. A heurística foi adaptada de Pena et al. (2021) para trabalhar com SPT desconsiderando a propriedade de simetria da relação de exclusão mútua imposta h_e.

O problema encontrado com a relação h_e é que a simetria torna-se muito restritiva para nosso estudo de caso. A interpretação da especificação E no SPT diz: "Dada uma string, digamos $ab \in L_m(E)$, podemos interpretá-la de duas maneiras: 1) o momento inicial de a deve preceder o momento inicial de b; ou 2) o momento inicial de b é posterior ao momento em que o disparo de a é concluído."(Su et al., 2012). Esta afirmação origina a simetria em h_e para lidar com ambas as sentenças, mas se considerarmos apenas a segunda interpretação, então é possível obter resultados melhores no escalonamento da produção. As suposições feitas na seção 2.3, as técnicas de modelagem e avaliação do makespan adotadas, garantem a segurança do sistema e viabilidade da solução.

Seguindo o exemplo da Pequena Fábrica demonstrado na última seção, se ao invés de usar $he = \{(A_1, A_2), (A_2, A_1)\}$, considerarmos apenas $he = \{(A_2, A_1)\}$, então é possível alcançar o mesmo makespan obtido pela abordagem TCS usando a simplificação da modelagem SPT (SPT MODIFICADO), conforme ilustrado na Figura 4.

4. PRINCIPAIS RESULTADOS

Nesta seção, propomos um algoritmo que calcula o makespan de uma cadeia de eventos desconsiderando a propriedade de simetria da relação de exclusão mútua imposta em um SPT.

4.1 Método de síntese do supervisor

Um supervisor pode ser visto como um domínio que contém todas as soluções possíveis para um problema de planejamento de produção representado em um autômato cíclico. Portanto, inclui as sequências de eventos que levam à produção de lotes de todos os tamanhos. Para abstrair a solução desejada sem criar uma nova especificação e compor um novo supervisor, como em Pena et al. (2021), usamos a técnica aplicada em Alves et al. (2021) que utiliza um autômato de desenrolamento para criar um grafo acíclico *on-the-fly*. A técnica consiste em conhecer a fórmula de produção e então, buscar no supervisor os caminhos que correspondem ao comportamento desejado. A fórmula é o número de vezes que cada evento precisa ocorrer para completar um ciclo de produção. Com essas informações, podemos pesquisar através do supervisor acompanhando os eventos.

Tabela 2. Formula da Produção

Eventos	a_1	b_1	a_2	b_2
Número de Repetições	1	1	1	1

Usando o supervisor Figura 2 como exemplo; se dissermos que a fórmula segue a tabela 2, então a única solução possível é a string $s = a_1b_1a_2b_2$ para o lote de um produto. A fórmula deve ser multiplicada pelo número de ciclos (lotes) necessários, então em um lote com 2 produtos as strings possíveis seriam $s_1 = a_1b_1a_2b_2a_1b_1a_2b_2$, $s_2 = a_1b_1a_2a_1b_2b_1a_2b_2$ e $s_3 = a_1b_1a_2a_1b_1b_2a_2b_2$. A vantagem desta técnica é que podemos encontrar possíveis strings sem compor todas as soluções, ao invés de fazer um supervisor para cada ciclo de produção, o que se torna essencial na busca heurística de lotes maiores.

4.2 Busca Clonal / Busca Heurística

A metodologia adotada para resolver o problema de otimização é o Algoritmo de Busca Clonal (CSA - Clonal Search Algorithm). O Algoritmo de Busca Clonal proposto em de Castro and Von Zuben (2002) é inspirado no comportamento do sistema imunológico de mamíferos, onde células

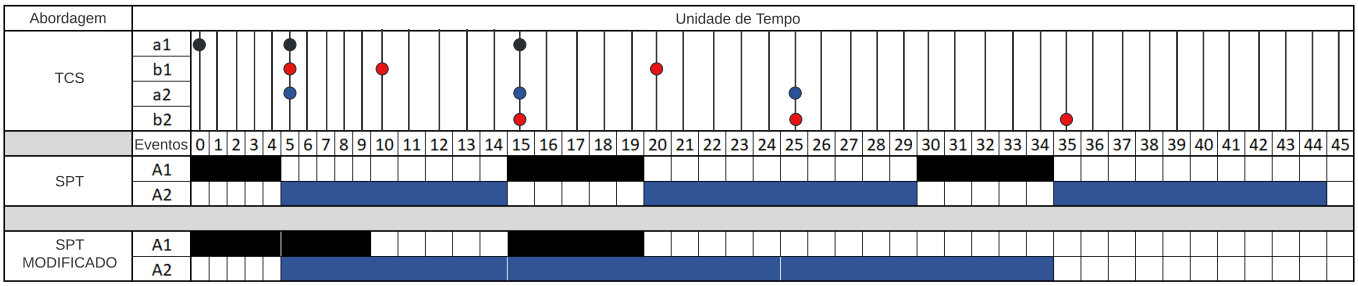

Figura 4. Passagem do Tempo

que podem ter uma melhor resposta contra o antígeno são selecionadas para sofrer replicação e mutação, fortalecendo assim o sistema imunológico.

Neste trabalho, adaptaremos a versão CSA utilizada em Rafael (2018). Na versão original, as abstrações e o conceito de estados divergentes são usados para reduzir a complexidade do supervisor e auxiliar o uso do algoritmo clonal na resolução de problemas usando TCS. No entanto, não há necessidade de usar abstrações, pois todos os eventos são controláveis. Além disso, é essencial adaptar o algoritmo para trabalhar com a abordagem SPT.

Primeiramente, é necessário introduzir o conceito de evento ativo e depois os estados divergentes. Eventos ativos $\Gamma(x)$ são todos os eventos que podem ocorrer no estado x. Estados Divergentes (ED) são estados onde há mais de um evento ativo, conforme definido abaixo:

Definição *1.* Seja $G = (X, \Sigma, \delta, x_0, X_m)$ um autômato determinístico e $\Gamma(x) = \{\sigma \in \Sigma \mid \delta(x, \sigma)!\}$ sua função de evento ativo correspondente. Um estado x é chamado Estado Divergente (ED) se $|\Gamma(x)| > 1$. O conjunto de todos os estados ED é $X_{ED} = \{x \in X | \Gamma(x) > 1\}$.

A Busca Clonal é um processo heurístico porque usa parâmetros de parada para interromper o algoritmo sem avaliar todas as possibilidades. Inicialmente, uma quantidade N de sequências aleatórias do supervisor (Geração 0) é escolhida, conforme mostrado na Figura 5. As sequências são avaliadas e classificadas com base no makespan. O algoritmo de avaliação será apresentado na subseção seguinte.

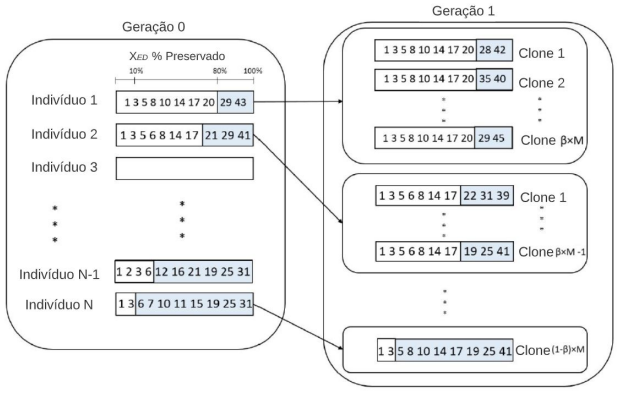

Figura 5. Diagrama do Processo Clonal Rafael (2018)

Com a geração 0 classificada, o algoritmo inicia o processo de mutação. A lógica da mutação é que os melhores indivíduos geram mais clones e sofrem menos mutação, enquanto os piores indivíduos fazem o contrário. A mutação altera a porcentagem de estados divergentes preservados; uma sequência com 20% de mutação terá 80% de X_{ED} preservado. O algoritmo procura aleatoriamente rotas a partir do último estado preservado que atingem o estado marcado, criando a próxima Geração.

Este processo é repetido até que um critério de parada seja atendido. Esses critérios são o número máximo de gerações ou o número de gerações sem melhoria. A Figura 6 demonstra o comportamento do desvio padrão (DP) ao longo das gerações. As flutuações no DP são justificadas pelas mutações das gerações, sendo clara a tendência para um $DP = 0$ com o avanço das gerações. No entanto, $DP = 0$ não garante que o aplicativo atingiu o mínimo global, mas sim local. Por esta razão, o algoritmo é repetido um número específico de vezes em favor de um melhor resultado.

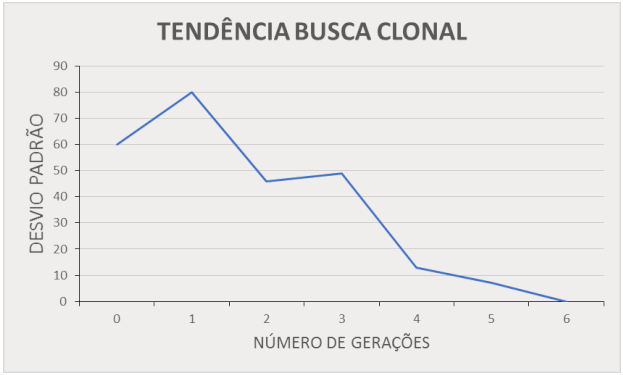

Figura 6. Tendência da Busca Clonal

4.3 Algoritmo de Avaliação do Makespan

Nesta subseção, será apresentado o algoritmo que avalia uma cadeia de eventos (string) e gera o makespan correspondente. O algoritmo avalia o tempo necessário para completar uma sequência de eventos respeitando as relações (h e he) no SPT.

A Figura 7 mostra um fluxograma do algoritmo. Existem quatro entradas essenciais: a String, a Duração de cada evento e as tabelas com as relações (h e he). Como as características do SPT dizem respeito à passagem do tempo, o algoritmo atribui os momentos de disparo (t_σ) de cada evento e avalia o tempo necessário para completar a sequência. Assumimos que o CSA fornece a sequência de eventos.

Primeiro, criamos o List_Control, que é uma tabela com três colunas, Tempo Inicial (T.I.), Evento e Tempo Final (T.F.). Sua finalidade é manter registrada a última aparição de cada evento. Inicialmente, os valores de T.I. e T.F. são zero. Então, para cada evento σ na String, o código encontrará o evento de relação (h ou he) que ocorreu mais recentemente, ou seja, o maior T.F. entre as relações. Esse tempo é chamado Tempo Final Máximo das Relações (TFMR). Uma vez obtido o TFMR, o próximo passo é analisar o seu valor. Se o TFMR for igual a zero, então o T.I. do evento σ assume o valor zero; isso representa quando um evento pode acontecer a qualquer momento ou é o primeiro evento da sequência. A outra possibilidade é que o TFMR seja maior que zero, neste caso, o T.I. de σ assume o valor TFMR. O último caso ocorre quando temos dois eventos que têm relação ocorrendo sequencialmente. Após o T.I. de σ ser definido, calculamos o T.F. usando a Duração e atualizamos o List_Control. Quando o algoritmo passa por todos os eventos da String, o makespan é obtido do maior T.F. na Lista_Control. O algoritmo engloba relações de exclusão intrínsecas e nos permite usar relações de exclusão impostas sem simetria.

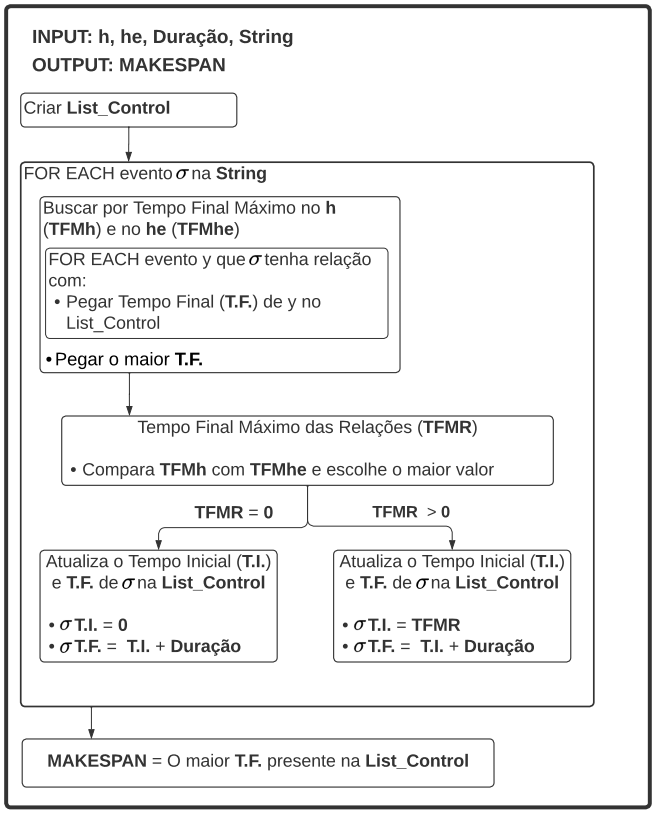

Figura 7. Algoritmo

5. ESTUDO DE CASO

Esta seção está dividida em duas partes. Primeiramente, apresentamos a MecatrIME. Em seguida, analisamos os resultados. A MecatrIME é uma plataforma de Manufatura Integrada por Computador (Computer Integrated Manufacturing - CIM) composta por 9 estações diferentes: ASRS, Torno, Fresagem, Soldagem, Metrologia, Montagem, Gravação a Laser, Esteira e Gerenciador. Este trabalho analisará a produção de dois produtos envolvendo o ASRS (Automated Storage and Retrieving System), a esteira e a estação de torno.

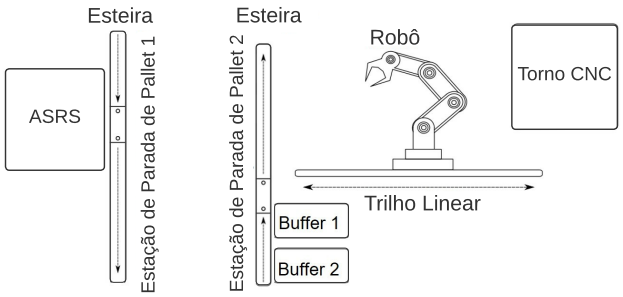

Figura 8. MecatrIME

5.1 Processo de Produção

O ASRS é um sistema de armazenamento cartesiano que contém 72 slots de templates. O template é uma base plástica ou metálica que aloca o material. Possui um engate que permite que o robô e o ASRS movam a base pelo sistema. A esteira possui 12 pallets, cada um com um ID único verificado na EPP (Estação de Parada de Pallet). Como a esteira possui uma estrutura em malha fechada com um único caminho, é comum ver uma fila de pallets quando um pallet entra na EPP. O sistema se assemelha a uma linha de metrô. O template é um passageiro que precisa ir das estações A a D. O metrô (pallet) parará nas estações B, C e D sequencialmente. Então, apesar do passageiro só descer na parada D, ele tem que esperar o metrô passar pelas outras estações. Esta configuração tende a criar uma fila de pallets visto que a operação de carga/descarga de um pallet com um template bloqueia a passagem de outros pallets pela EPP. Neste estudo, um processo sempre se inicia com uma matéria-prima do ASRS sendo enviada para a Estação Torno, e então a peça acabada é armazenada no ASRS.

A Estação de Torno consiste em um braço robótico (Yaskawa Motman MHSF) montado em um Trilho Linear, dois buffers e um torno CNC (EMCO CONCEPT TURN 250). Visto que o pallet correto (aquele que carrega o template com a matéria-prima) chega à EPP 2, o robô se move para uma posição próxima e transfere o template do pallet para o buffer. Feita a transferência, o pallet é liberado, o robô pega a matéria-prima e a coloca no CNC. Com a operação de torneamento concluída, o robô realiza o processo inverso, passando do CNC para o buffer e depois do buffer para um pallet livre na EPP. Como não há controle sobre onde os pallets estão na esteira, o sistema deve esperar até que um pallet livre chegue à EPP. Após o carregamento do pallet, o mesmo é enviado para ser armazenado no ASRS, finalizando o processo. (https://sites.google.com/ime.eb.br/mecatrime)

5.2 Resultados

Nossa produção consiste em lotes de dois produtos; o primeiro produto A tem um tempo de usinagem de 14 segundos e o produto B dura 23 segundos. Os lotes seguem a relação de 2 produtos A para cada produto B, portanto, no lote 10, são 20 A e 10 B (30 produtos no total). A produção em tempo real será medida até o lote quatro

Tabela 3. Resultados (Tempo em segundos)

	Tempo Real	SPT		BUSCA CLONAL		MÁXIMO PARALELISMO		**Clonal + SPT (MODIFICADO)**	
Lotes	Makespan	Makespan	Runtime	Makespan	Runtime	Makespan	Runtime	**Makespan**	**Runtime**
1	720	582	1,5448	588	3,381	594	0,003600	**582**	**7,715**
2	1324	999	22,5988	972	6,667	972	0,009833	**972**	**7,906**
3	1860	1416	62,0089	1350	10,027	1350	0,016367	**1390**	**13,575**
4	2563	1833	125,9196	1728	13,492	1728	0,024700	**1771**	**23,511**
5	3133	2250	202,3739	2106	17,907	2106	0,033400	**2209**	**26,718**
6	3739	2667	311,1392	2484	20,762	2484	0,042233	**2593**	**28,065**
7	4346	3084	416,0932	2862	23,394	2862	0,049600	**3171**	**31,86**
8	4952	3501	1117,9555	3240	28,045	3240	0,059233	**3588**	**34,858**
9	5559	3918	1198,9526	3618	27,413	3618	0,067200	**4068**	**50,257**
10	6165	4335	-	3996	30,496	3996	0,074900	**4502**	**51,193**

Tabela 4. Modelagem do Autômato

Abordagem	**Planta**		**Especificação**	
	Estados	Transições	Estados	Transições
SPT	9	108	315	862
	Supervisor			
	Estados		Transições	
	315		862	
TCS	**Planta**		**Especificação**	
	Estados	Transições	Estados	Transições
	2700	17640	315	1724
	Supervisor			
	Estados		Transições	
	2072		5658	

devido às limitações de matéria-prima. O restante do lote será projetado com base no estado permanente do sistema.

Os resultados desta subseção foram obtidos comparando-se quatro abordagens. O conceito de Máximo Paralelismo proposto em Alves et al. (2021), que consiste em buscar o comportamento do sistema (supervisor) que representa o maior número de dispositivos trabalhando simultaneamente (paralelo) para resolver um problema de escalonamento de tarefas utilizando TCS. A Busca Clonal aplicada em Pena et al. (2021) e explicada na seção anterior. O trabalho feito em Tuxi and da Cunha (2018) e Tuxi (2018) usa um Breadth First Search (BFS) que procura por sequências com o menor makespan enquanto faz modificações nos autômatos de busca *on-the-fly* para reduzir a complexidade computacional em SPT. Além disso, o algoritmo proposto neste artigo. Devido ao uso de duas abordagens (SPT e SED), foi necessário realizar uma modelagem de autômatos para cada um dos casos tabela 4. É necessário compor os supervisores de lote para utilizar a técnica apresentada em Tuxi (2018).

A tabela 3 apresenta os resultados obtidos. A coluna "tempo real" foi medida através da realização de testes no sistema real, enquanto as demais foram obtidas através da aplicação dos respectivos algoritmos. Os valores em vermelho foram especulados com base na projeção, seja por limitações físicas ou de software. Os testes foram executados em um notebook com processador Intel Core I7-4700MQ 2,40 GHz com 16,0 GB de RAM. Os parâmetro utilizados na Busca Clonal e na Clonal+TWS são: número de indivíduos na Geração 0 = 100, taxa de mutação de 100 %, indivíduos selecionados por geração = 2, números de clones = 10, máximo de gerações sem melhorias = 5, máximo de gerações = 40 e repetições do código = 10.

A abordagem SPT proposta em Tuxi and da Cunha (2018) não conseguiu computar o décimo lote, e tem no geral, um alto tempo de execução computacional (Runtime). Apesar de ser um algoritmo que encontra o menor tempo, obteve resultados piores quando comparado às heurísticas em TCS devido ao problema de simetria explicado na seção 3. A Busca Clonal e o Máximo Paralelismo conseguiram atingir os menores valores de makespan em pouco tempo; no entanto, sua modelagem é mais complexa. A abordagem proposta neste trabalho conseguiu atingir valores semelhantes aos obtidos na TCS utilizando os modelos compactos do SPT. Além disso, o tempo computacional no nono lote foi aproximadamente 24 vezes mais rápido quando comparado ao SPT e possui um supervisor 6 vezes menor que o TCS. Esses resultados demonstram a vantagem de utilizar a abordagem proposta em problemas complexos que consistem em múltiplos componentes, estações e diferentes lotes de produção.

6. CONCLUSÃO

Este artigo propõe um algoritmo que possibilita o uso do SPT sem a necessidade de atender à simetria da relação de exclusão mútua imposta. Além disso, o algoritmo utiliza boas práticas herdadas de trabalhos como Tuxi and da Cunha (2018), Alves et al. (2021) e Pena et al. (2021). Os resultados são comparados com as técnicas herdadas e testes práticos medidos em um sistema de manufatura integrado por computador.

O algoritmo proposto permite simplificações na parte de modelagem de autômatos, utiliza um método de síntese escalável e uma heurística baseada em Busca Clonal. Os resultados demonstram que foi possível obter valores de makespan semelhantes ao TCS em um pequeno tempo computacional usando SPT, resolvendo assim o problema de simetria.

REFERÊNCIAS

Alves, L., Martins, L., and Pena, P. (2017). Ultrades - a library for modeling, analysis and control of discrete event systems. *IFAC-PapersOnLine*, 50, 5831–5836. doi: 10.1016/j.ifacol.2017.08.540.

Alves, L., Pena, P., and Takahashi, R. (2021). Planning on discrete event systems using parallelism maximization. *Control Engineering Practice*, 112, 104813. doi:10.1016/j.conengprac.2021.104813.

Brandin, B.A. and Wonham, W.M. (1994). Supervisory control of timed discrete-event systems. *IEEE Transactions on Automatic Control*, 39(2), 329–342. doi: 10.1109/9.272327.

Cassandras, C. and Lafortune, S. (2010). *Introduction to Discrete Event Systems*, 800. doi:10.1007/978-0-387-68612-7.

de Castro, L. and Von Zuben, F. (2002). Learning and optimization using the clonal selection principle. *IEEE Transactions on Evolutionary Computation*, 6(3), 239–251. doi:10.1109/TEVC.2002.1011539.

Lin, L., Su, R., Brandin, B.A., Ware, S., Zhu, Y., and Sun, Y. (2019). Synchronous composition of finite interval automata. In *2019 IEEE 15th International Conference on Control and Automation (ICCA)*, 578–583. doi:10.1109/ICCA.2019.8899529.

Pena, P.N., Vilela, J.N., Alves, M.R.C., and Rafael, G.C. (2021). Abstraction of the supervisory control solution to deal with planning problems in manufacturing systems. *IEEE Transactions on Automatic Control*, 67(1), 344–350. doi:10.1109/TAC.2021.3053228.

Rafael, G.C. (2018). Solution of a scheduling problem using an abstraction of the closed loop behaviour of a discrete event system. URL `https://lacsed.eng.ufmg.br/projects`.

Ramadge, P. and Wonham, W. (1989). The control of discrete event systems. *Proceedings of the IEEE*, 77(1), 81–98. doi:10.1109/5.21072.

Su, R., van Schuppen, J., and Rooda, J. (2012). The synthesis of time optimal supervisors by using heaps-of-pieces. *Automatic Control, IEEE Transactions on*, 57, 105 – 118. doi:10.1109/TAC.2011.2157391.

Tuxi, T.M. (2018). *Contribuições ao Controle de Sistemas com Ponderações Temporais*. Master's thesis, Instituto Militar de Engenharia.

Tuxi, T.M. and da Cunha, A.E.C. (2018). Contributions to the control of time-weighted systems. In *2018 IEEE 23rd International Conference on Emerging Technologies and Factory Automation (ETFA)*, volume 1, 418–425. doi: 10.1109/ETFA.2018.8502616.