A TEORIA DE CONTROLE SUPERVISÓRIO APLICADA NA SELEÇÃO DO LAYOUT DE UM CLUSTER TOOL

Márcio J. Nunes*, Patrícia N. Pena†

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

†Departamento de Engenharia Eletrônica - Universidade Federal de Minas Gerais, Belo Horizonte, MG, Brasil

Emails: marcio-junior@hotmail.com, ppena@ufmg.br

Abstract— This paper presents a way to determine the best layout in a cluster tool using two performance metrics: the makespan and the Relative Accumulated Parallelism, the latter being an index proposed in this work. Four different layouts are considered, varying the times of process modules from 1 to 200 seconds. The values of theses metrics are obtained applying the Maximum Parallelism with Time Constraints algorithm to the supervisor responsible for the control of the plant. The analysis is performed comparing the makespan and accumulated parallelism of each sequence for the four layouts, the makespan being a measure of quickness of the cluster tool, and the Relative Accumulated Parallelism being a measure of efficiency. The radial layout with 1 robot presented the best performance when the cluster tool is robot bound, and the radial layout with 2 robots when the cluster tool is process bound.

Keywords— Discrete Event Systems, Supervisory Control Theory, Cluster Tool, Manufacturing Systems

Resumo— Este artigo propõe uma forma de determinar o layout com melhor desempenho para um *cluster tool* a partir de duas métricas de desempenho: o *makespan* e o Paralelismo Acumulado Relativo, sendo este último um indicador proposto neste trabalho. Consideram-se quatro layouts diferentes, variando os tempos das câmaras de processo de 1 a 200 segundos. Os valores destas métricas são obtidos aplicando o algoritmo de Máximo Paralelismo com Restrições Temporais ao supervisor responsável pelo controle da planta. A análise é feita comparando o makespan e o paralelismo acumulado de cada sequência retornada pelo algoritmo para os quatro layouts, sendo o *makespan* uma medida de velocidade de produção do *cluster tool*, e o Paralelismo Acumulado Relativo uma medida de eficiência. Verificou-se que o layout radial com 2 robôs apresenta melhor desempenho quando o cluster tool é *robot bound*, enquanto o layout radial com 1 robô se destaca quando o cluster tool é *process bound*.

Palavras-chave— Sistemas a Eventos Discretos, Teoria de Controle Supervisório, Cluster Tool, Sistemas de Manufatura

1 Introdução

O processo de fabricação de wafers é responsável por mais de 75% do tempo de ciclo na manufatura de circuitos integrados, e é também um dos maiores componentes de custo (Mönch et al., 2011). Por esses motivos, há um esforço constante pela melhoria operacional nesta etapa de produção.

Neste contexto, o *cluster tool* têm sido um tipo de equipamento muito utilizado neste processo. Segundo a norma SEMI E21-94 (*Semiconductor Equipment and Materials International*) (2009), o *cluster tool* é "um sistema de manufatura integrado, ambientalmente isolado, consistindo de processos, transporte e módulos cassete mecanicamente ligados em um mesmo conjunto".

Cluster tools têm diversos layouts e requisitos de escalonamento. O layout pode ser radial, linear ou multi-cluster, de acordo com a configuração dos módulos de processo (*PM*) (Lee e Lee, 2010). Devido à essa diversidade de parâmetros, a escolha do modelo mais adequado antes da compra de um novo equipamento se torna difícil. Há alguns modelos que apoiam a tomada de decisão, geralmente fornecidos pelos próprios fabricantes. Porém, na maioria das vezes, não é possível comparar a performance de diferentes modelos, devido às diferenças na modelagem dos sistemas.

O uso da TCS em *Cluster Tools* tem pouco respaldo na literatura, principalmente para comparação de vários layouts. Uma aplicação da TCS para controle e escalonamento de um *cluster tool* utilizando supervisores modulares locais é apresentada em Schafaschek et al. (2015). Já Ware e Su (2017) utilizaram um algoritmo baseado na projeção de linguagens e poda de um supervisor para encontrar uma sequência temporal ótima em *cluster tools* lineares.

Diferentes técnicas foram utilizadas no estudo de parâmetros específicos de *cluster tools* lineares ou radiais separadamente, como redes de Petri (Lu et al., 2018; Zuberek, 2001) e programação linear (Yang et al., 2017), simulação (LeBaron e Hendrickson, 2000), máquinas de estado finito (Shin et al., 2001) e formulação matemática utilizando o método de decomposição (Yi et al., 2007). Alguns outros, consideraram os layouts radial e linear simultaneamente. Em Park e Morrison (2011), usou-se simulação para estudar o efeito de setup e lavagem em ambos os layouts. Já Van Der Meulen (2007) aponta alguns fatores que interferem no tempo de processamento de lo-

tes de produção menores que 25 wafers para estes *cluster tools*. Lee e Lee (2010) apresentaram uma arquitetura de escalonamento aberta baseada em redes de Petri que pode ser usada em diferentes layouts. Dessa forma, observa-se que nestes trabalhos destacam-se principalmente duas abordagens: a análise de um único layout e parâmetros específicos utilizando outras ferramentas de SEDs, como por exemplo as redes de Petri, ou a utilização de ferramentas não relacionadas a eventos discretos aplicadas a mais de um tipo de layout.

Uma vez que a diferente disposição dos componentes em cada layout resulta em diferentes desempenhos operacionais, este artigo apresenta uma análise de desempenho do *cluster tool* de acordo com o layout, comparando o *makespan* e o Paralelismo Acumulado Relativo de quatro layouts diferentes, mostrados na Figura 1. A Figura 1 também mostra a receita para cada layout, sendo a sequência do processamento do wafer identificada pelos números ao lado de cada *PM*.

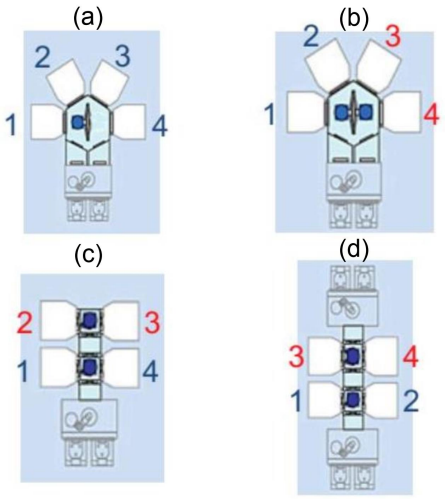

Figura 1: Layouts considerados:
(a) Radial com 1 robô, (b) Radial com 2 robôs, (c) Linear com entrada e saída única e (d) Linear com entrada e saída separada

Nessa abordagem, um sistema de controle denominado supervisor é construído para controlar e evitar estados indesejáveis do *cluster tool* e situações de bloqueio, utilizando a Teoria de Controle Supervisório (TCS) (Ramadge e Wonham, 1989) de Sistemas a Eventos Discretos (SED). A TCS é a base para o uso do algoritmo de Máximo Paralelismo com Restrições Temporais (MPT) (Alves et al., 2016), escolhido para avaliar os layouts.

O uso desse algoritmo, que é executado sobre o comportamento legal da planta obtido pela TCS, se apresenta como uma alternativa simples de controle e sequenciamento para este tipo de aplicação. Assim, podemos aplicá-lo às diferentes configurações de *cluster tools*, a fim de comparar o desempenho do sistema. Dessa forma, temos uma ferramenta que pode ser utilizada ao mesmo tempo para controlar e escalonar o *cluster tool*, e ainda comparar diferentes arquiteturas variando seus parâmetros de operação.

Este artigo está organizado em 6 seções, sendo a primeira esta Introdução. Na Seção 2 descrevem-se brevemente os conceitos preliminares relacionados à TCS e ao MPT. A Seção 3 apresenta a descrição e modelagem do *cluster tool*, e a Seção 4 mostra a aplicação do algoritmo MPT ao problema. A Seção 5 traz os principais resultados e a análise dos dados obtidos. As observações finais são apresentadas na Seção 6.

2 Conceitos Preliminares

Os sistemas a eventos discretos (SEDs) são sistemas dinâmicos regidos por estímulos com o mundo externo, chamados eventos, que ocorrem em instantes de tempo não determinados. Um evento pode indicar o instante de início ou fim de uma tarefa, além de eventos internos. Exemplos de eventos são o início ou fim de uma tarefa do sistema, o acionamento de um sensor em um sistema de manufatura, a chegada de um cliente em uma fila, recepção de mensagens em uma rede de comunicação ou o fim de um evento interno, como uma temporização. Entre a ocorrência de dois eventos consecutivos, o sistema permanece no mesmo estado. São os eventos que causam a mudança destes estados.

Dentre os diversos modelos formais utilizados em SEDs, o modelo proposto por Ramadge e Wonham (Ramadge e Wonham, 1989), baseado na Teoria de Linguagens e Autômatos e denominado "modelo RW", foi utilizado como forma de representação de SEDs neste artigo.

Para um conjunto de eventos finitos Σ formando um *alfabeto*, dizemos que uma *palavra* s sobre Σ é uma sequência qualquer de eventos deste conjunto, e o *fechamento de Kleene* Σ^* é o conjunto de todas as palavras formadas por eventos de Σ, incluindo a palavra vazia ε. Um subconjunto $L \subseteq \Sigma^*$ é chamado linguagem. A concatenação das palavras u, v forma a palavra $t = uv$, sendo u o prefixo de t, denotado por $u \leq t$, e v o seu sufixo. O prefixo fechamento $\overline{L}$ de uma linguagem $L \subseteq \Sigma^*$ é o conjunto de todos os prefixos de cadeias em L, isto é, $\overline{L} = \{u \in \Sigma^* \mid u \leq t \text{ para algum } t \in L\}$.

Definição 1 *Um autômato é uma quíntupla* $G = (Q, \Sigma, \hat{\delta}, q_0, Q_m)$*, sendo* Q *o conjunto finito e não-vazio de estados,* Σ *o conjunto de eventos,* $\hat{\delta} : Q \times \Sigma \to Q$ *a função de transição,* q_0 *o estado inicial e* $Q_m \subseteq Q$ *o conjunto de estados marcados.*

A função de transição pode ser estendida para lidar com cadeias sobre Σ^* como $\delta : Q \times \Sigma^* \to Q$. Neste caso, $\delta(q, \epsilon) = q$ e $\delta(q, s\sigma) = \hat{\delta}(\delta(q, s), \sigma)$.

A linguagem gerada por $G = (Q, \Sigma, \delta, q_0, Q_m)$ é $\mathcal{L}(G) := \{s \in \Sigma^* : \delta(q_0, s) \text{ é definida}\}$, e a linguagem marcada é $\mathcal{L}_m(G) := \{s \in \mathcal{L}(G) : \delta(q_0, s) \in Q_m\}$.

A Teoria de Controle Supervisório (TCS) proposta por Ramadge e Wonham (1989), possibilita, baseado na teoria de linguagens e autômatos, um método formal para o cálculo de supervisores não bloqueantes e minimamente restritivos, tal que a ação de controle sobre a planta seja mínima. Na TCS, a partir de especificações de segurança e do sistema a ser controlado, denominado planta, gera-se o supervisor, que é o agente controlador, agindo no sistema desabilitando eventos controláveis para limitar o comportamento da planta apenas àquele desejado. A planta é modelada por um autômato $G = (Q, \Sigma, \delta, q_0, Q_m)$, com $\Sigma = \Sigma_c \cup \Sigma_u$, sendo Σ_c o conjunto de eventos controláveis, que podem ser desabilitados pelo supervisor, e Σ_u o conjunto de eventos não controláveis, que não podem ser desabilitados.

As linguagens gerada e marcada de uma planta G sobre a ação de um supervisor S são, respectivamente, $\mathcal{L}$(S/G) e $\mathcal{L}_m(S/G) \subseteq \mathcal{L}(S/G)$. Um supervisor S é dito não bloqueante quando $\overline{\mathcal{L}_m(S/G)} = \mathcal{L}(S/G)$, ou seja, se o prefixo-fechamento da linguagem marcada pelo supervisor S controlando a planta G é igual à linguagem gerada pelo supervisor S controlando a planta G, então o supervisor S é não bloqueante.

Sendo G a planta, E uma especificação e K o comportamento desejado, a condição necessária e suficiente para que exista um supervisor S não bloqueante para G, tal que $\mathcal{L}_m$(S/G) = $\mathcal{L}(G) \mid\mid E = K$, é que K seja controlável em relação a $\mathcal{L}(G)$ e Σ_u. Diz-se que K é controlável se $\overline{K}\,\Sigma_u \cap \mathcal{L}(G) \subseteq \overline{K}$. Caso não seja, deve-se calcular a máxima sublinguagem controlável da linguagem desejada K em relação a G, denotada por $Sup\mathcal{C}(K, G)$.

No algoritmo MPT (Alves et al., 2016), representa-se o número de tarefas ativas em um estado por meio da função de tarefas ativas $f_{ta} : Q \to \mathbb{Z}^*$, que associa a cada estado $q \in Q$ de um autômato um número inteiro não negativo de tarefas. Para avaliar o paralelismo de uma sequência finita, é necessário definir a função acumulativa de tarefas ativas.

Definição 2 *(Alves et al., 2016) Seja $S = (Q, \Sigma, \delta, q_0, Q_m)$ um supervisor. A função acumulativa de tarefas ativas, $F_{TA} : Q \times \Sigma^* \to \mathbb{Z}^*$ é definida como:*

$$\begin{cases} F_{TA}(q, \epsilon) = f_{ta}(q) \\ F_{TA}(q, \sigma s) = f_{ta}(q) + F_{TA}(\delta(q, \sigma), s) \end{cases}$$

onde $\sigma s \in \Sigma^*$.

Para duas sequências s_1 e s_2, tais que $|s_1| = |s_2|$, aquela com maior valor de F_{TA} tem maior paralelismo.

A função temporal f_T avalia o tempo para ocorrência de um evento no supervisor, dada a sequência já executada.

Definição 3 *(Alves et al., 2016) Seja $S = (Q, \Sigma, \delta, q_0, Q_m)$ um supervisor. A função temporal $f_T^{'} : \Sigma^* \times \Sigma \to \mathbb{R}^*$, de S é definida como:*

$$f_T^{'}(s, \sigma) = \begin{cases} t & \text{se } \delta(\delta(q_0, s), \sigma) \text{ está definido} \\ \infty & \text{caso contrário} \end{cases}$$

tal que, para um evento $\sigma \in \Sigma$ e uma sequência $s \in \mathcal{L}(S/G)$, t é o tempo até que o evento σ ocorra, dado que a sequência s acabou de ser executada.

A função temporal é implementada no algoritmo utilizando a simulação de sistemas a eventos discretos, avaliando a diferença entre o tempo de execução do último evento da sequência s e o tempo de s. Para a avaliação da informação temporal de uma sequência de eventos, a função temporal deve ser expandida:

Definição 4 *(Alves et al., 2016) Seja $S = (Q, \Sigma, \delta, q_0, Q_m)$ um supervisor. A função temporal expandida $f_T : \Sigma^* \to \mathbb{R}^*$, de S é definida como:*

$$\begin{cases} f_T(\epsilon) = 0 \\ f_T(s\sigma) = f_T(s) + f_T^{'}(s, \sigma) \end{cases}$$

Assim, busca-se a sequência da linguagem marcada do supervisor que ao mesmo tempo maximize o paralelismo e tenha um tempo de execução menor que infinito, indicando que ela é temporalmente factível.

Sendo n o número de eventos para produção de um lote de produtos e o universo de busca $L = \{s \in \mathcal{L}(S/G) \mid |s| = n \ \wedge \ f_T(s) \neq \infty\}$, o problema de otimização pode ser definido como o de encontrar o caminho s^* no autômato do supervisor S que maximiza o número acumulado de tarefas ativas F_{TA}:

$$s^* = \underset{s \in L}{\text{argmax}}\ F_{TA}(s).$$

O algoritmo MPT resolve esse problema caminhando sobre o supervisor, associando uma agenda de eventos a cada estado. Quando um estado é alcançado na mesma profundidade por mais de um caminho, sendo a agenda de eventos diferente, somente é mantido o caminho que apresenta maior paralelismo acumulado.

De modo geral, associamos neste trabalho os eventos controláveis ao início de tarefas no *cluster tool* e os eventos não controláveis ao fim dessa tarefa, sendo estes respostas da planta ao eventos controláveis. Um problema que pode ocorrer no *cluster tool* é o aparecimento de situações que causem o bloqueio no sistema, chamadas de *deadlocks*. Essas situações devem ser evitados pelo supervisor.

3 Descrição e Modelagem do *Cluster Tool*

Os layouts dos *cluster tools* considerados são apresentados na Figura 1. Na Figura 1(a) é represen-

tado um layout radial com um robô manipulador *single-armed*, enquanto na Figura 1(b) tem-se um layout radial com dois robôs manipuladores *single-armed*. Na Figura 1(c), um layout linear com entrada e saída única é mostrado, ou seja, com apenas um *Equipment Front-End Module (EFEM)* para inserção e retirada de wafers no *cluster tool*. Já a Figura 1(d) representa um layout linear com entrada e saída separadas, sendo um *EFEM* para inserção e outro para retirada de wafers. Cada *EFEM* tem um robô manipulador e duas *load ports* para armazenar o *FOUP (Front Opening Unified Pod)*, onde os wafers são transportados. Considerou-se, para efeito de comparação, a existência de quatro *PMs*, com um slot único em cada um. Optou-se por trabalhar com lotes de 25 wafers, conforme os trabalhos de Mönch et al. (2011), Van Der Meulen (2007) e Hendrickson (1997).

A receita do wafer, que é a sequência de passos a ser seguida no *cluster tool*, é única para todos os layouts, considerando um fluxo serial, no qual o wafer visita cada *PM* exatamente uma vez de modo sequencial. Cada *PM* é responsável por uma etapa do processamento.

A operação do *cluster tool* pode ser classificada de acordo com o equipamento que representa o gargalo de produção. Quando o *cluster tool* opera na região denominada *process bound*, o tempo de processamento dos *PMs* domina o tempo de ciclo e o robô deve esperar algum *PM* finalizar a operação. Por outro lado, na região denominada *robot bound*, o robô está sempre ocupado, transferindo wafers entre os *PMs* (Yi et al., 2007).

3.1 Modelagem e Síntese de Supervisores

Na modelagem de cada layout as ações dos robôs e as operações nos *PMs* são representadas por eventos. Os eventos rotulados por números ímpares formam o conjunto de eventos controláveis Σ_c, e indicam o início de alguma tarefa, já aqueles rotulados por números pares formam o conjunto de eventos não controláveis Σ_u, e indicam o fim da tarefa correspondente. Deste modo, no MPT, a duração de cada atividade é a diferença temporal entre o evento controlável que indica seu início e o evento não-controlável que indica seu fim. O modelo do layout linear com entrada e saída única, é mostrado na Figura 2, com seus respectivos eventos identificados sobre as setas que indicam a movimentação do wafer. Neste caso, $\Sigma_c = \{11, 13, 103, 113, 115, 121, 123, 125, 211, 213, 215, 223, 225\}$ e $\Sigma_u = \{12, 14, 112, 114, 116, 122, 124, 132, 212, 214, 222, 224, 226\}$.

Inicialmente, para possibilitar a síntese de um supervisor monolítico capaz de controlar a planta de modo minimamente restritivo, foram obtidos os modelos em autômatos dos equipamentos que compõem cada layout considerado, e das restrições

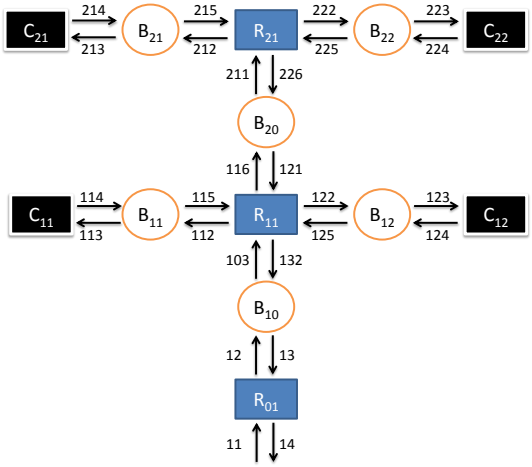

Figura 2: Exemplo de modelo do Layout Linear com entrada e saída única

a serem consideradas no funcionamento desejado. Como exemplo, alguns dos autômatos que modelam os componentes e as especificações do layout linear com entrada e saída única são mostrados na Figura 3. Os autômatos dos robôs manipuladores são indiciados pela letra R, das câmaras de processamento *(PM)* pela letra C e das especificações pela letra E.

Nos modelos em autômatos mostrados, o estado inicial dos autômatos dos robôs e *PMs* é o estado marcado, uma vez que este indica que alguma tarefa foi completada. No caso dos *PMs*, essa tarefa é o processamento do wafer, e para os robôs manipuladores, essa tarefa é o transporte do wafer.

O autômato de cada *PM* possui apenas dois estados, que são o estado 0 (estado inicial e marcado), o qual indica que o *PM* está aguardando um wafer, e o estado 1, que indica que o *PM* está em operação.

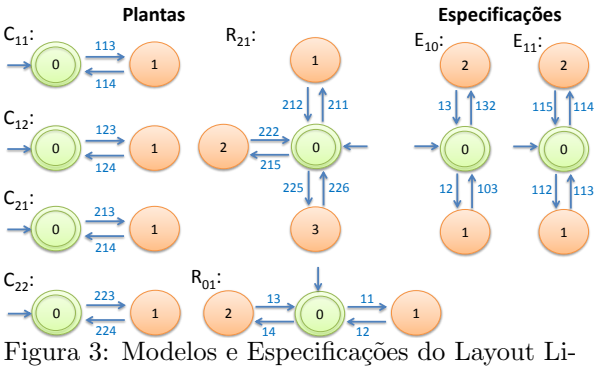

Figura 3: Modelos e Especificações do Layout Linear

No caso do robô manipulador do *cluster tool*, o número de estados depende da quantidade de movimentos que o mesmo executa na movimentação do wafer. O autômato do robô foi implementado com a sequência que o robô deve executar na receita, de tal forma que, quando um robô pegar um wafer, o único evento possível é depositar o wafer na posição correta. Como os robôs manipu-

ladores podem mover o wafer para vários pontos dentro do *cluster tool*, o autômato correspondente geralmente apresenta um maior número de estados quando comparado ao autômato do *PM*.

Os autômatos das restrições são responsáveis por estabelecer as regras de produção a serem respeitadas, e evitar o *overflow* e *underflow* nos equipamentos do *cluster tool*. Dessa forma, os autômatos das especificações evitam que wafers sejam retirados do *PM* antes de serem processados, que o mesmo wafer seja processado mais de uma vez e que mais de um wafer seja colocado no *PM*. As características descritas se mantêm em todos os layouts.

Uma vez obtidos os modelos em autômatos da planta e das especificações, utilizou-se a TCS para calcular os supervisores monolíticos de cada um dos layouts apresentados. O supervisor é responsável pelo controle lógico dos equipamentos no *cluster tool*, garantindo o comportamento desejado de modo e livre de bloqueio, atuando sobre a planta de modo minimamente restritivo. Esse cálculo foi feito na ferramenta computacional UltraDES (Alves et al., 2017). Inseriu-se os modelos em autômatos dos *PMs*, robôs e especificações, no UltraDES, que calculou e retornou o supervisor monolítico para cada layout mostrado.

Tabela 1: Estados e Transições dos Supervisores

	Supervisor	
Layout	Estados	Transições
Radial 1 Robô	3.855	12.951
Radial 2 Robôs	4.684	16.381
Linear E/S Única	3.255	9.715
Linear E/S Separada	12.696	51.470

É objetivo do trabalho comparar o desempenho de diferentes layouts. No entanto, cada layout demanda um número diferente de equipamentos. Apesar da receita ser a mesma em todos eles, devido à diferença entre o número de equipamentos, observa-se uma diferença também no número de eventos de uma sequência para produzir um mesmo lote de wafers. A Tabela 2 mostra o número de equipamentos e a quantidade de eventos necessários para produzir um wafer em cada layout.

Tabela 2: Número de eventos e equipamentos

Layout	Ev.	Equip.
Radial - 1 Robô	22	6
Radial - 2 Robôs	22	7
Linear - E/S Única	26	7
Linear - E/S Separada	24	8

Estas informações serão utilizadas no cálculo do PAR, que é uma das métricas escolhidas para comparação dos layouts.

4 Aplicação do MPT ao Problema

Para aplicar o MPT nos supervisores encontrados em cada layout, é necessário determinar o tempo de duração das tarefas no *cluster tool*, além de avaliar o tempo até que determinado evento ocorra em algum estado do supervisor. Neste trabalho, para calcular os tempos de movimentação dos robôs manipuladores, foram utilizados os mesmos valores descritos por Yi et al. (2007) para rotações com ângulos de 90° e 180°, comuns no *cluster tool* linear. O tempo de rotação do robô é função do ângulo do movimento e do índice que indica a temperatura do wafer transportado. Já o tempo de pegar/depositar o wafer no *PM* ou nos *load locks* e *ASLs* é função apenas da temperatura do wafer transportado. No layout radial, utilizou-se uma regressão polinomial para encontrar a função que relaciona o tempo de rotação do robô com o ângulo, pois nesse caso há movimentos em ângulos diferentes de 90° e 180°. Detalhes desta implementação foram omitidos por questão de espaço.

O tempo de operação total do robô é a soma dos tempos de deslocamento até a posição do wafer, o tempo para pegá-lo, executar o movimento de rotação até o destino, colocar o wafer e retornar à posição inicial, sendo que cada um desses tempos devem ser calculados considerando o respectivo parâmetro de temperatura do wafer. Considerou-se que o robô sempre volta à posição inicial após depositar o wafer.

O tempo de todos os *PMs* (em todos os layouts) é igual, e varia em cada uma das 30 execuções do algoritmo no conjunto de valores {1, 3, 5, 7, 10, 13, 15, 17, 20, 23, 25, 30, 35, 40, 45, 50, 55, 60, 65, 70, 75, 80, 85, 90, 95, 100, 125, 150, 175, 200}. No MPT, a duração de cada atividade é a diferença temporal entre o evento que indica seu início e o evento que indica o fim.

Para avaliar o tempo até a ocorrência de um evento, dada uma sequência s no supervisor, é utilizada a função temporal, implementada por meio de simulação de sistemas a eventos discretos.

Após a determinação desses valores, o algoritmo de escalonamento MPT foi aplicado. Foram feitas 30 execuções, de forma aleatória, e em cada uma alterou-se o tempo de processo dos *PMs*, variando-o de 1 a 200 segundos, registrando os valores de Paralelismo Acumulado e *makespan* retornados pelo algoritmo.

4.1 Métricas Utilizadas

Além da sequência de maior paralelismo, o algoritmo MPT retorna também o *makespan*, que é o tempo total de produção do lote, e o Paralelismo Acumulado (PA) da sequência, dado pela função acumulativa de tarefas ativas. Para efeito de comparação dos layouts, o valor do Paralelismo

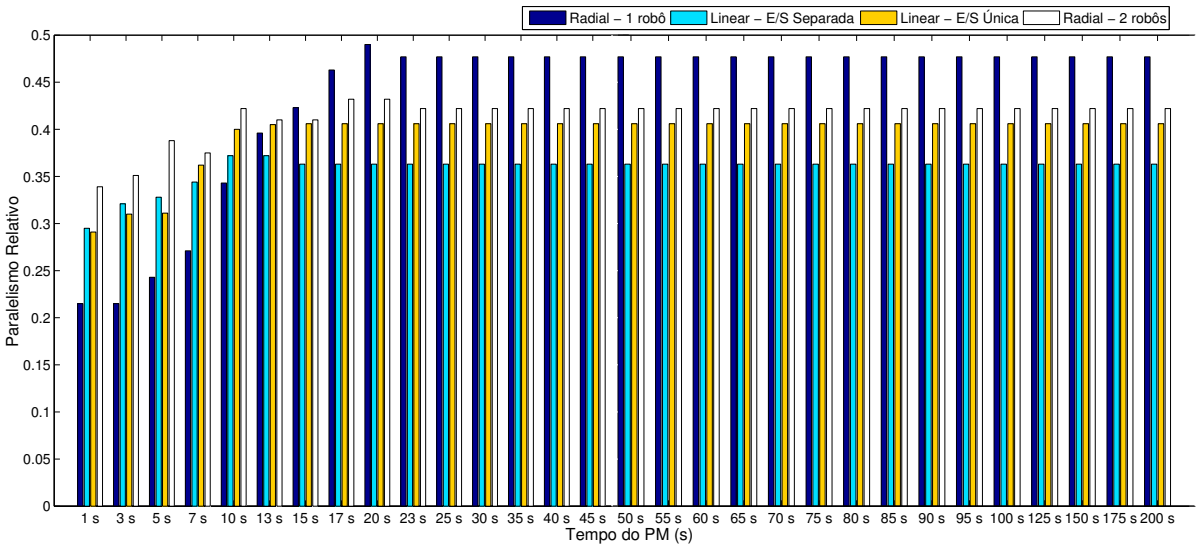

Figura 4: Paralelismo Relativo - Algoritmo MPT - Lote de 25 wafers

Acumulado (PA) de cada sequência não caracteriza uma boa métrica, já que seu valor é tanto maior quanto maior o número de eventos ou maior o número de equipamentos para produzir o mesmo wafer. Dessa forma, um novo parâmetro denominado Paralelismo Acumulado Relativo (PAR) é proposto, dado por (1), para comparar os layouts. Este parâmetro normaliza o Paralelismo Acumulado (PA) de acordo com o número de eventos da sequência para produzir um wafer (Ev) e equipamentos do layout ($Equip$).

$$PAR = \frac{PA}{Ev \times Equip} \tag{1}$$

O PAR é usado como um indicador de eficiência do layout, já que, o layout com maior valor para esse parâmetro produz a mesma quantidade de produtos com um número menor de equipamentos, ou com menor número de eventos.

Já o *makespan* é usado como um indicador de velocidade de produção, uma vez que, comparando dois layouts diferentes, aquele com menor valor de *makespan* irá produzir o lote de wafers em menor tempo.

5 Resultados Experimentais e Análise

Os algoritmos foram executados em um notebook com processador Intel Core I5-3210M, 12 GB de memória RAM e sistema operacional de 64 bits.

O MPT executa uma busca sobre o supervisor, restringindo-o às sequências temporalmente factíveis que produzem o lote de tamanho desejado, no caso 25 wafers. Este algoritmo é aplicado sobre os quatro layouts e o PAR da melhor sequência obtida é apresentado na Figura 4, em função dos tempos de *PM*, que variam entre 1 e 200 s. Quando o tempo do *PM* é menor que o tempo de movimentação do robô, este último sempre estará em operação, o que levará os wafers processados a aguardar pelo transporte. Nesta situação, o *cluster tool* é *robot bound*, uma vez que o gargalo do *cluster tool* é o robô. Por este motivo, observa-se que para tempos do *PM* até 13 s, o layout radial com 2 robôs apresenta maior valor de PAR, o que significa que é o layout com maior número de operações em paralelo, considerando o número de equipamentos e eventos, ou seja, o mais eficiente do ponto de vista do paralelismo. De modo análogo, observa-se que o layout radial com 1 robô tem o menor valor de PAR. Os layouts linear com entrada e saída única e linear com entrada e saída separada apresentam valores de PAR próximos para tempos do *PM* até 7 s, indicando que a eficiência destes layouts é aproximadamente a mesma nestes casos.

Porém, para tempos dos *PMs* maiores ou iguais a 15 s, o tempo de movimentação do robô é menor que o tempo de processamento do wafer, e o robô ficará ocioso aguardando o fim do processamento dos wafers. Assim, o *cluster tool* é *process bound*, uma vez que o *PM* se torna o gargalo. Nestas condições, o layout radial com 1 robô passa a ter o maior valor de PAR, indicando que este é o layout mais eficiente do ponto de vista do paralelismo, enquanto o layout radial com 2 robôs terá o segundo maior valor de PAR, como apresentado na Figura 4. Observa-se também que o layout linear com entrada e saída separada apresenta o menor valor de PAR, e que a diferença deste para o layout linear com entrada e saída única não mais apresenta valores próximos. Isso ocorre porque a utilização de dois robôs no layout radial ou dois *EFEMs* no layout linear não traz um paralelismo maior à sequência, já que os equipamentos adicionais ficam ociosos aguardando o processamento no *PM*. Para tempos dos *PMs* maiores que 25 s, percebe-se que os valores de PAR de todos os layouts se mantêm constantes, pois o aumento do tempo do *PM* apenas aumenta o tempo de espera do robô.

A Figura 5 apresenta a evolução do *makes-*

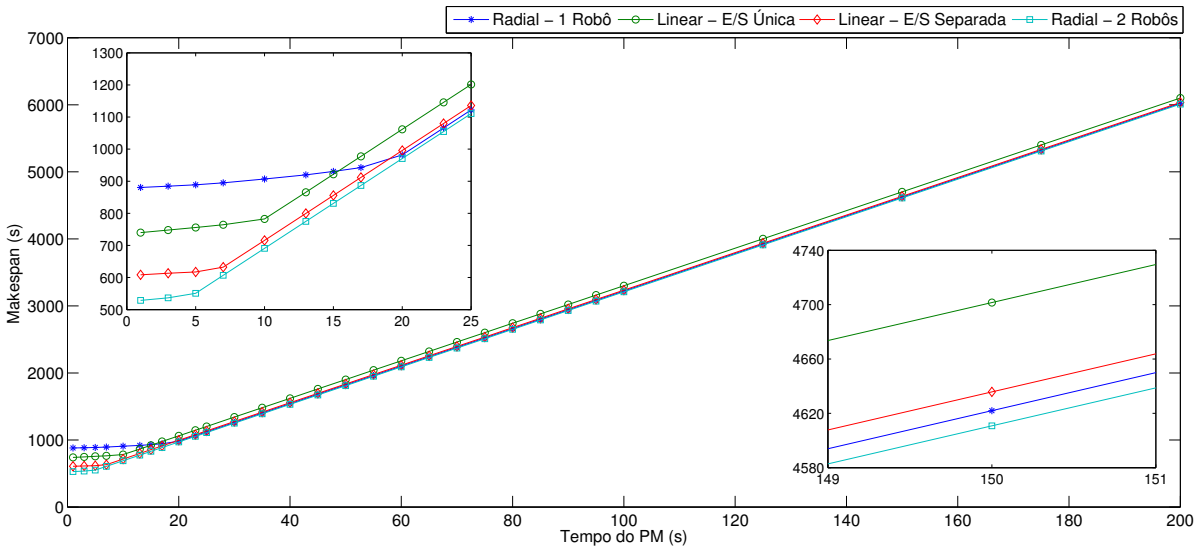

Figura 5: Algoritmo MPT - *Makespan* para vários tempos do *PM* - Lote de 25 wafers

pan em função do tempo do *PM*. Observa-se que para qualquer valor de tempo do *PM*, o layout radial com 2 robôs apresenta o menor *makespan*. Para tempos dos *PMs* menores ou iguais a 15 s, quando o processo é *robot bound*, o layout radial com 1 robô tem o pior desempenho em termos de *makespan*, o que pode ser percebido no detalhe superior da Figura 5 pelo valor dessa métrica próximo de 900 s, enquanto o layout radial com 2 robôs consegue produzir com *makespan* pouco acima de 500 s. Isso se justifica pelo fato do robô ser o gargalo do processo para estes tempos de *PM*. Nesta situação o layout linear com entrada e saída separada tem menor valor de *makespan* na comparação com o layout linear com entrada e saída única, uma vez que o wafer não retorna ao *EFEM* de entrada do *cluster tool* ao fim do processamento. Já para tempos dos *PMs* maiores que 20 s, o layout radial com 1 robô tem o segundo menor valor de *makespan*. Neste caso, como o *cluster tool* é *process bound*, há pouca influência do tempo de movimentação do robô no *makespan*, sendo a diferença dessa métrica entre o layout radial com 1 robô e radial com 2 robôs de 11,17 s, conforme detalhe inferior da Figura 5.

Observa-se também que o *makespan* cresce linearmente, com duas inclinações diferentes. Em todos os layouts, a menor inclinação ocorre para os menores valores de tempo do *PM*. Nesta situação, os tempos dos *PMs* são menores que o tempo de movimentação do robô, e enquanto essa condição se manter, um aumento neste parâmetro tem pouca influência no *makespan*. O ponto de inflexão, que é diferente em cada layout, variando entre 5 s e 17 s, representa o ponto no qual o *cluster tool* deixa de ser *robot bound* para se tornar *process bound*. Após este ponto, um aumento no tempo do *PM* gera um incremento proporcional no *makespan*, pois o tempo do *PM* será maior que o tempo de operação do robô.

É interessante observar que o tempo do *PM* influencia no *makespan* de cada layout, como pode ser visto no caso do layout radial com 1 robô, que, para tempos de *PM* menores que 15 s, tem o maior valor de *makespan*, e para tempos maiores que 20 s tem o segundo menor valor.

6 Conclusões

Neste artigo foi apresentada uma metodologia para realizar a comparação entre diferentes layouts do *cluster tool*. Utilizou-se o algoritmo MPT, baseado na TCS, para gerar duas métricas que são utilizadas para avaliar e comparar o desempenho do layout. A metodologia permite caracterizar o sistema, classificando-o como *process bound* ou *robot bound*. O conhecimento dos tempos de *PM* e de movimentação do robô são fundamentais para a definição de um layout mais adequado.

Analisando os resultados obtidos, observa-se que o layout radial com 2 robôs apresenta melhor desempenho quando o *cluster tool* é *robot bound*, pois seu menor valor de *makespan* e maior valor de PAR indicam que ele produz de forma mais rápida e com maior eficiência, respectivamente. Porém, quando o *cluster tool* é *process bound*, o layout radial com 1 robô se destaca, pois seu maior valor de PAR indica um layout mais eficiente, já que o valor de *makespan* em relação ao layout que apresentou o menor valor, no caso o layout radial com 2 robôs, é muito próximo, com uma diferença de 11,17 s. Essa diferença, quando comparada percentualmente, diminui com o aumento do *makespan*, corroborando com essa escolha.

No caso da aplicação do algoritmo MPT especificamente no *cluster tool*, uma vantagem da metodologia desenvolvida é que ela incorpora de maneira simultânea os softwares controlador e escalonador, já que fornece ao mesmo tempo um controle minimamente restritivo por meio da TCS e uma sequência de escalonamento de produção

por meio do MPT, evitando um grande problema do *cluster tool* que é o *deadlock*, além de propor um sequenciamento visando a operação eficiente.

Uma vez que características e vantagens qualitativas de cada layout, como a facilidade de manutenção, operação e de expansão não são consideradas, este método não deve ser usado como forma única de seleção de um *cluster tool*, já que considera apenas o *makespan* e o paralelismo da sequência. Porém, esta abordagem mostrou-se adequada para verificar a influência do layout e dos tempos dos *PMs* no desempenho.

Desdobramentos futuros deste trabalho incluem a expansão do problema para *cluster tools* com maior número de *PMs*, a variação no tamanho dos lotes de produção e a aplicação dessa abordagem com parâmetros reais de um *cluster tool*.

Agradecimentos

O presente trabalho foi realizado com o apoio financeiro da Fapemig, CAPES - Brasil e CNPq.

Referências

Alves, L., Martins, L. e Pena, P. (2017). Ultra-DES - A Library for Modeling, Analysis and Control of Discrete Event Systems, *Accepted at the 20th World Congress of the International Federation of Automatic Control, IFAC 2017*.

Alves, L., Pena, P. e Takahashi, R. (2016). Planejamento da produção baseado no critério do máximo paralelismo com restrições temporais, *XXI Congresso Brasileiro de Automática (CBA)*, pp. 1518–1523.

Hendrickson, R. (1997). Optimizing cluster tool throughput, *Solid State Technology*, **40**(7): 217–221.

International), S. E.-. S. E. M. (2009). Cluster tool module interface: Mechanical interface and wafer transport standard, *SEMI E21-94 (reapproved 0309)* .

LeBaron, H. T. e Hendrickson, R. A. (2000). Using emulation to validate a cluster tool simulation model, *Proceedings of the 2000 Winter Simulation Conference*, Winter Simulation Conference, pp. 1417–1422.

Lee, J.-H. e Lee, T.-E. (2010). An open scheduling architecture for cluster tools, *IEEE Conference on Automation Science and Engineering (CASE), 2010*, IEEE, pp. 420–425.

Lu, Y., Pan, C., Qiao, Y., Wu, N. e Chen, Y. (2018). Petri net-based deadlock avoidance for single-arm cluster tools with concurrently processing two-type wafers, *Networking, Sensing and Control (ICNSC), 2018 IEEE 15th International Conference on*, IEEE, pp. 1–6.

Mönch, L., Fowler, J. W., Dauzère-Pérès, S., Mason, S. J. e Rose, O. (2011). A survey of problems, solution techniques, and future challenges in scheduling semiconductor manufacturing operations, *Journal of Scheduling*, **14**(6): 583–599.

Park, K. e Morrison, J. R. (2011). Cluster tool design comparisons via simulation, *Proceedings of the Winter Simulation Conference*, Winter Simulation Conference, pp. 1877–1887.

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

Schafaschek, G., de Queiroz, M. H. e Cury, J. E. R. (2015). Local modular supervisory control applied to the scheduling of cluster tools, *Automation Science and Engineering (CASE), 2015 IEEE International Conference on*, IEEE, pp. 1381–1388.

Shin, Y.-H., Lee, T.-E., Kim, J.-H. e Lee, H.-Y. (2001). Modeling and implementing a real-time scheduler for dual-armed cluster tools, *Computers in Industry*, **45**(1): 13–27.

Van Der Meulen, P. (2007). Linear semiconductor manufacturing logistics and the impact on cycle time, *Advanced Semiconductor Manufacturing Conference, 2007. ASMC 2007. IEEE/SEMI*, IEEE, pp. 111–116.

Ware, S. e Su, R. (2017). Time optimal synthesis based upon sequential abstraction and its application to cluster tools, *IEEE Transactions on Automation Science and Engineering* **14**(2): 772–784.

Yang, F., Wu, N., Qiao, Y., Zhou, M., Su, R. e Qu, T. (2017). Modeling and optimal cyclic scheduling of time-constrained single-robot-arm cluster tools via petri nets and linear programming, *IEEE Transactions on Systems, Man, and Cybernetics: Systems* pp. 1–13.

Yi, J., Ding, S., Zhang, M. T. e van der Meulen, P. (2007). Throughput analysis of linear cluster tools, *IEEE International Conference on Automation Science and Engineering. CASE 2007.*, IEEE, pp. 1063–1068.

Zuberek, W. M. (2001). Timed petri nets in modeling and analysis of cluster tools, *IEEE Transactions on Robotics and Automation*, **17**(5): 562–575.