57º SBPO
SIMPÓSIO BRASILEIRO DE PESQUISA OPERACIONAL
Gramado • RS
DE 5 A 9 DE OUTUBRO DE 2025

Revenimento Paralelo Aplicado ao Sequenciamento de Tarefas em Máquinas Flexíveis Paralelas Não Idênticas

André Luís Barroso Almeida
Universidade Federal de Ouro Preto
Campus Morro do Cruzeiro, s/n, CEP 35400-000 - Ouro Preto/MG
andre.almeida@ifmg.edu.br

Marco Antonio Moreira de Carvalho
Universidade Federal de Ouro Preto
Campus Morro do Cruzeiro, s/n, CEP 35400-000 - Ouro Preto/MG
mamc@ufop.edu.br

RESUMO

O problema de sequenciamento de tarefas e trocas de ferramentas em máquinas paralelas não idênticas é uma variação do clássico problema de minimização de trocas de ferramentas. Nesse contexto, diferentes tarefas exigem conjuntos distintos de ferramentas para seu processamento, e, como as máquinas não comportam todas as ferramentas simultaneamente, interrupções são necessárias para trocas, as quais impactam significativamente o desempenho do sistema. O problema, de natureza $\mathcal{NP}$-difícil, é tratado considerando a minimização do *makespan*. É proposta a aplicação do algoritmo revenimento paralelo (ou *parallel tempering*), cujos resultados são avaliados em um conjunto abrangente de 640 instâncias *benchmark* extraídas da literatura. O método proposto superou consistentemente a abordagem estado da arte, alcançando todos os valores ótimos conhecidos e apresentando melhorias de até 14,19% em relação às melhores soluções publicadas e redução de até 98,45% no tempo computacional. Novas melhores soluções foram geradas para 37,96% das instâncias da literatura.

PALAVRAS CHAVE. Sequenciamento. Manufatura Flexível. Revenimento Paralelo.

Tópicos (POI, MH, AD&GP)

ABSTRACT

The job-sequencing and tool-switching problem on non-identical parallel machines is a variation of the classic job sequencing and tool switching problem. In this context, different jobs require different sets of tools for their processing, and, since machines cannot support all tools simultaneously, interruptions are necessary for tool switches. These interruptions can significantly impact system performance. The problem is $\mathcal{NP}$-hard and is approached considering the makespan minimization. We propose the application of the parallel tempering algorithm. We evaluate the results using a comprehensive set of 640 benchmark instances drawn from existing literature. Our proposed method consistently outperformed state-of-the-art approaches, achieving all known optimal values and demonstrating improvements of up to 14.79% over the best solutions published so far, with reductions of up to 98.45% in running time. New best solution values were generated for 37.96% of the literature instances.

KEYWORDS. Sequencing. Flexible Manufacturing. Parallel Tempering.

Paper topics (POI, MH, AD&GP)

DOI: 10.59254/sbpo-2025-212298
http://live-proceedings.pantheonsite.io/p/212298

1. Introdução

O modelo tradicional de produção, baseado na criação de linhas dedicadas para cada produto, torna-se cada vez mais inviável devido a limitações logísticas, altos custos de implementação e à volatilidade dos interesses do mercado. Como alternativa, surgiram a Indústria 4.0 e as *smart factories*, a partir das quais passou-se a adotar sistemas produtivos mais adaptáveis e inteligentes, incluídos os *sistemas de manufatura flexíveis* (SMFs).

Os SMFs são compostos por máquinas de controle numérico computadorizado (CNC), ou seja, programáveis, interligadas de maneira a operar de forma coordenada [Yadav e Jayswal, 2018]. Adicionalmente, as máquinas CNC em um SMF são ditas flexíveis por possuírem a capacidade de serem configuradas com diferentes ferramentas para produzir diferentes tipos de produtos. Genericamente, a fabricação de um produto é considerada uma *tarefa* a ser processada pela máquina flexível. Diferentes tarefas requerem que diferentes *ferramentas* estejam carregadas na máquina CNC para realizar operações como perfurar, fresar, serrar e cortar, entre outras.

Os SMF são particularmente úteis no contexto do modelo de manufatura conhecido como *high-mix*, *low-volume* (HMLV) [Gan et al., 2023], relacionado a processos industriais com alto grau de controle e repetibilidade, como na produção de semicondutores, e na manufatura de precisão. No HMLV, a fabricação ocorre em lotes pequenos, permitindo a flexibilidade necessária para atender às demandas do mercado em constante mudança. Cada produto único implica em tempos de produção distintos e ajustes específicos de maquinário, tornando a programação da produção um desafio complexo.

Cada máquina CNC em um SMF possui um compartimento denominado *magazine* com capacidade limitada para armazenar as ferramentas necessárias para o processamento das tarefas. No entanto, considerando-se a variedade e a quantidade de ferramentas disponíveis, é impossível acomodá-las todas simultaneamente no *magazine*, originando a necessidade de *trocas de ferramentas* entre o processamento sequencial de tarefas diferentes. As trocas de ferramentas representam frequentemente os maiores gargalos de tempo em um FMS [Van Hop, 2005], uma vez que são realizadas com as máquinas CNC desativadas, o que gera impactos financeiros relevantes devido à ociosidade das máquinas.

O *job sequencing and tool switching problem* (SSP) [Tang e Denardo, 1988] modela o sequenciamento do processamento das tarefas e o planejamento das trocas de ferramentas em um SMF. O sequenciamento das tarefas é um problema $\mathcal{NP}$-difícil [Crama et al., 1994], enquanto o plano de trocas de ferramentas pode ser determinado em tempo determinístico polinomial [Tang e Denardo, 1988; Qiu et al., 2025]. Na versão *uniforme* do SSP, considera-se uma única máquina flexível e tanto o tempo necessário para trocas de ferramentas quanto o tamanho das ferramentas são uniformes. O objetivo é o de minimizar o número de trocas de ferramentas necessário para processar todas as tarefas.

Neste artigo é considerado um ambiente industrial com múltiplas máquinas paralelas não idênticas, uma variante do SSP denominada *job sequencing and tool switching problem with non-identical parallel machines* (SSP-NPM). Este problema de otimização foi formalizado por Calmels [2022] como segue. Considera-se um FMS com um conjunto de ferramentas $F = \{1, \ldots, f\}$, um conjunto de máquinas paralelas não idênticas $M = \{1, \ldots, m\}$, cada uma equipada com um *magazine* com capacidade limitada c_i ($i \in M$) e um conjunto de tarefas $T = \{1, \ldots, t\}$. A heterogeneidade entre as máquinas se expressa tanto na capacidade dos *magazines* quanto nos tempos de processamento p_{ij} ($i \in M, j \in T$) de cada tarefa e nos tempos de troca sw_i ($i \in M$). O SSP-NPM busca definir simultaneamente a alocação e o sequenciamento das tarefas em cada máquina. Os objetivos a serem alcançados variam entre minimizar o *makespan*, minimizar o *flowtime* total e minimizar o número de trocas de ferramentas. Neste estudo, considera-se o objetivo de

DOI: 10.59254/sbpo-2025-212298
http://live-proceedings.pantheonsite.io/p/212298

minimizar o *makespan*.

A Tabela 1 apresenta os detalhes de uma instância do SSP-NPM que $t = 6$, $f = 9$, $m = 2$, $c_1 = 3$, $c_2 = 4$, $sw_1 = 3$, e $sw_2 = 2$. Tempos de processamento iguais a ∞ indica que uma tarefa não pode ser processada por uma máquina devido a uma restrição de elegilibidade.

Tabela 1: Instância SSP-NPM. Adaptado de Soares e Carvalho [2024].

Tarefas	1	2	3	4	5	6
Ferramentas	1	2	1	3	3	2
	3	4	2	5	5	3
	7	8	5	7	9	4
	9			8		
p_{1j}	∞	5	8	∞	2	7
p_{2j}	1	4	7	5	1	9

Uma solução para uma instância do SSP-NPM é caracterizada pelo sequenciamento de tarefas em cada máquina e pelo plano de trocas de ferramentas correspondente. As trocas de ferramentas em uma máquina $i \in M$ podem ser representadas por uma matriz binária $R^i = \{r^i_{tj}\}$, com $t \in F$ e $j \in T$. Em R^i, as linhas representam as ferramentas e as colunas representam as tarefas atribuídas à máquina i na ordem de processamento. Uma entrada $r^i_{tj} = 1$ indica que a ferramenta t está carregada na máquina i antes do processamento da tarefa j, caso contrário, $r^i_{tj} = 0$. O número de trocas de ferramentas em uma única máquina é calculado usando a Equação (1), que conta o número de inversões de 0 a 1, representando a inserção da ferramenta no *magazine*.

$$Z_{SSP}(R^i) = \sum_{j \in J} \sum_{t \in T} r^i_{tj}(1 - r^i_{tj-1}) \tag{1}$$

O valor *makespan* é dado pelo maior tempo de término de processamento de uma máquina do FMS, o que inclui os tempos de processamento das tarefas designadas a ela e também o tempo relacionado às trocas de ferramentas. Seja A_i o conjunto de tarefas atribuídas à máquina $i \in M$. A Equação (2) apresenta a forma de cálculo do *makespan*.

$$C_{max} = \arg\max \left\{ \sum_{j \in A_i} p_{ij} + sw_i \times \sum_{j=2}^{j=|A_i|} \sum_{k \in F} r^i_{kj}(1 - r^i_{kj-1}) \right\}, \forall i \in M \tag{2}$$

A Tabela 2 ilustra uma possível solução para a instância de exemplo ao apresentar as matrizes R^1 e R^2, com as atribuições de tarefas às máquinas m_1 e m_2 respectivamente. Nesta solução hipotética, o valor do *makespan* corresponde ao tempo de término da máquina m_2, igual a 23 unidades de tempo. Segundo Calmels [2022], os três possíveis objetivos para o SSP-NPM são conflitantes entre si e requerem métodos especializados para cada tipo de avaliação.

Este estudo descreve a aplicação do método revenimento paralelo (ou *parallel tempering*, PT) ao SSP-NPM. O desempenho recente deste método a problemas de otimização relacionados à programação da produção em sistemas de manufatura flexível sugere que esta é uma abordagem interessante para o problema abordado Almeida et al. [2025b]. Os experimentos computacionais reportados utilizam instâncias *benchmark* da literatura e comparam os resultados obtidos com aqueles considerados estado-da-arte.

DOI: 10.59254/sbpo-2025-212298
http://live-proceedings.pantheonsite.io/p/212298

Tabela 2: Possível solução para a instância de exemplo.

R^1			
Ferramentas/Tarefas	2	3	5
1	0	1	0
2	1	1	0
3	0	0	1
4	1	0	0
5	0	1	1
6	0	0	0
7	0	0	0
8	1	0	0
9	0	0	1

R^2			
Ferramentas/Tarefas	1	4	6
1	1	0	0
2	0	0	1
3	1	1	1
4	0	0	1
5	0	1	0
6	0	0	0
7	1	1	0
8	0	1	0
9	1	0	0

O restante deste artigo está organizado conforme descrito a seguir. A Seção 2 apresenta uma revisão da curta literatura a respeito do SSP-NPM; a Seção 3 apresenta o método PT brevemente; a Seção 4 reporta os experimentos computacionais realizados e, por fim, a Seção 5 apresenta as conclusões e os trabalhos futuros.

2. Revisão da literatura

O SSP tem sido amplamente explorado na literatura desde sua formulação para ambientes simples até variações mais complexas. A versão básica denominada uniforme foi introduzida por Tang e Denardo [1988], que propuseram o algoritmo denominado *keep tool needed soonest* (KTNS) para resolver o subproblema de determinação do plano de trocas em tempo determinístico polinomial. A complexidade do SSP uniforme foi demonstrada por Crama et al. [1994], que provaram sua natureza $\mathcal{NP}$-difícil, o que estimulou o desenvolvimento de heurísticas e metaheurísticas.

Os primeiros esforços de generalização para ambientes com múltiplas máquinas surgiram na década de 1980, quando Stecke [1983] discutiu desafios relacionados à alocação de tarefas em FMSs. A primeira abordagem explícita ao SSP com múltiplas máquinas foi proposta por Khan et al. [2000], que desenvolveram uma heurística gulosa para o problema com duas máquinas idênticas em paralelo. Desde então, surgiram novos estudos concentrados em ambientes com máquinas paralelas idênticas, como exemplificado por Beezão et al. [2017], que propuseram dois modelos de programação inteira mista e uma metaheurística adaptativa, e por Soares e Carvalho [2020], que aplicaram o algoritmo genético de chaves aleatórias viciadas (ou *biased random-key genetic algorithm*, BRKGA) à mesma versão do SSP.

Para uma visão abrangente sobre o SSP, incluindo variações com tempos de preparação dependentes de sequência, objetivos múltiplos e ambientes diversos, destaca-se as revisão realizada por Calmels [2019]. No entanto, o SSP com máquinas paralelas não idênticas permanece pouco explorado.

Özpeynirci et al. [2016] e Gökgür et al. [2018] foram dos poucos autores a abordar o SSP-NPM, considerando tempos de processamento específicos para cada máquina, sem restrições de elegibilidade, e com número limitado de cópias de ferramentas. Enquanto Özpeynirci et al. [2016] apresentaram modelos matemáticos e uma busca tabu, Gökgür et al. [2018] avançaram com uma abordagem baseada em programação por restrições.

Posteriormente, Calmels [2022] propôs uma formulação de programação linear inteira mista, três heurísticas construtivas e três variantes de busca local iterada (*iterated local search*, ILS), avaliadas sobre um conjunto inédito de 640 instâncias. Foram consideradas múltiplas funções

DOI: 10.59254/sbpo-2025-212298
http://live-proceedings.pantheonsite.io/p/212298

objetivo: número de trocas de ferramentas, *makespan* e *flowtime* total. Os resultados mostraram que, enquanto o MILP foi competitivo para instâncias pequenas, as variantes do ILS se destacaram em instâncias maiores, especialmente para objetivos relacionados ao tempo.

Cura [2023] aprimorou o estado da arte com uma abordagem híbrida de algoritmo genético com busca local, testando duas políticas distintas de remoção de ferramentas (KTNS e RRT – *randomly removed tool*). Os experimentos mantiveram o mesmo conjunto de instâncias e funções objetivo de Calmels [2022], e os resultados mostraram melhorias consistentes, com destaque para os casos de maior dimensão. A melhoria média obtida em relação ao ILS anterior foi de 13,34%, consolidando o método proposto.

Mais recentemente, Soares e Carvalho [2024] apresentaram uma análise dos objetivos elencados para o SSP-NPM por Calmels [2022], concluindo que o objetivo de minimizar o número de trocas de ferramentas é incompatível com o ambiente de múltiplas máquinas paralelas não idênticas, à medida em que as soluções geradas atribuem um grande número de tarefas à máquina com *magazine* de maior capacidade, gerando uma ociosidade injustificável nas demais máquinas. Concluiu-se também que, embora a minimização do *makespan* e a minimização do *flowtime* total possuam grande afinidade, não necessariamente há soluções ótimas simultaneamente para ambos objetivos.

No mesmo estudo foi proposta a aplicação do BRKGA paralelizado e hibridizado com buscas locais organizadas na forma de uma descida em vizinhança variável (ou *variable neighbohood search*, VND). Os experimentos computacionais reportados consideraram as mesmas 640 instâncias de Calmels [2022] e os resultados foram comparados com os de Cura [2023]. O método BRKGA+VND foi capaz de atingir todas as soluções ótimas conhecidas e melhorar ou igualar todas as demais soluções anteriores. A melhoria média foi de 7,13% nos valores de *makespan* e de 6,36% nos valores de *flowtime* total, embora o custo computacional tenha sido considerado elevado.

Diante desse contexto, o presente trabalho dá continuidade à linha de pesquisa sobre o SSP-NPM ao propor uma nova estratégia baseada no revenimento paralelo, a fim de gerar novas e melhores soluções em tempo reduzido quando comparado ao estado da arte atual, reafirmando a relevância do SSP-NPM e a necessidade contínua de avanços metodológicos.

3. Revenimento paralelo

O PT, também conhecido como *replica exchange Monte Carlo*, é um método estocástico baseado em *Monte Carlo por Cadeias de Markov* (ou *Markov chain Monte Carlo*, MCMC), projetado para superar as limitações associadas à amostragem em espaços de alta dimensionalidade e distribuições multimodais [Swendsen e Wang, 1986; Geyer, 1991; Hukushima e Nemoto, 1996]. O método foi desenvolvido como uma resposta às dificuldades enfrentadas por algoritmos como o de *Metropolis* [Metropolis et al., 1953] em transitar entre modos distantes no espaço de estados, especialmente em problemas com múltiplos ótimos locais separados por elevadas barreiras de energia.

Os métodos tradicionais de Monte Carlo baseiam-se na geração aleatória de estados para inferir propriedades de distribuições complexas. A combinação desses métodos com cadeias de Markov originou os algoritmos MCMC, nos quais os estados são gerados sequencialmente, com cada novo estado dependendo unicamente do anterior. A transição entre estados é regulada pelo *critério de Boltzmann*, que assegura que a distribuição estacionária da cadeia coincida com a distribuição de interesse. Esse critério é essencial para garantir que o algoritmo realize a amostragem correta em regiões de alta probabilidade, mesmo em espaços altamente não lineares [Jones e Qin, 2022]. No algoritmo de Metropolis, cada nova solução candidata s' é aceita com probabilidade

$$P_{\text{aceite}} = \min\left[1, \exp\left(-\frac{\Delta E}{T}\right)\right],$$

DOI: 10.59254/sbpo-2025-212298
http://live-proceedings.pantheonsite.io/p/212298

em que $\Delta E = E(s') - E(s)$ representa a diferença de energia entre a nova e a atual solução, e T é a *temperatura* do sistema. A temperatura atua como um parâmetro de controle que afeta diretamente a probabilidade de aceitar movimentos para estados de maior energia: quanto maior a temperatura, mais permissiva é a aceitação desses movimentos. Entretanto, em temperaturas baixas, o algoritmo torna-se conservador, tendendo a aceitar apenas estados com melhor ou igual qualidade – o que pode resultar em estagnação em mínimos locais.

O método PT aborda esse problema simulando simultaneamente múltiplas réplicas do sistema, cada uma associada a uma temperatura distinta. Considera-se que existam κ réplicas $R = \{r_1, r_2, ..., r_\kappa\}$, com as temperaturas organizadas em ordem crescente, $T_1 < T_2 < \cdots < T_\kappa$. Réplicas em temperaturas mais elevadas realizam uma exploração mais ampla, sendo mais propensas a aceitar estados de energia maior. Já réplicas em temperaturas mais baixas realizam uma intensificação da busca, concentrando-se nas regiões de menor energia.

Cada réplica executa o algoritmo de Metropolis por um número fixo de etapas em sua temperatura. Periodicamente, o algoritmo propõe a troca de temperaturas entre pares de réplicas adjacentes, r_i e r_j. A troca é aceita com probabilidade

$$P(r_i \to r_j) = \min\left[1, \exp\left(\Delta\beta \cdot \Delta E\right)\right],$$

em que $\Delta\beta = \frac{1}{T_j} - \frac{1}{T_i}$ é a diferença entre os inversos das temperaturas e $\Delta E = E(r_j) - E(r_i)$ é a diferença de energia entre as soluções associadas às réplicas.

A troca de temperaturas induz uma trajetória dinâmica para cada réplica no espaço de temperaturas, permitindo que uma réplica escape de ótimos locais em temperaturas altas e, posteriormente, seja refinada em temperaturas baixas. Se adequado, uma réplica pode retornar a temperaturas mais altas posteriormente. Esse mecanismo automaticamente ajusta a relação de diversificação e intensificação, reduzindo assim o tempo de autocorrelação entre réplicas e melhorando substancialmente a cobertura do espaço de busca.

Apesar de sua eficácia na exploração de espaços de busca complexos, implementações tradicionais do PT são predominantemente sequenciais e apresentam limitações quanto à escalabilidade e desempenho computacional, especialmente em problemas de alta complexidade. Com o objetivo de mitigar tais limitações, Almeida et al. [2025b] propuseram uma abordagem inovadora que revisita o método PT sob a perspectiva da computação paralela.

Os autores desenvolveram uma API[1] (*Application Programming Interface*) dedicada, cuja arquitetura foi concebida para explorar o paralelismo intrínseco do método, viabilizando a execução simultânea de múltiplas cadeias de Markov em diferentes temperaturas. A API também contempla mecanismos otimizados de troca de réplicas, reduzindo redundâncias associadas à execução sequencial e promovendo ganhos expressivos em desempenho computacional. Além da eficiência, a proposta busca favorecer a modularidade, reusabilidade e facilidade de integração com novas aplicações, características essenciais para sua adoção por pesquisadores e profissionais da área.

A API foi avaliada experimentalmente em três problemas clássicos da literatura: o SSP, o escalonamento em máquinas paralelas idênticas com restrições de ferramentas, e o sequenciamento de tarefas com restrições de recursos em máquinas paralelas. Em todos os casos, os resultados demonstraram vantagens significativas da implementação paralela, tanto em termos de tempo de execução quanto na qualidade das soluções obtidas. Dessa forma, a utilização dessa API representa uma escolha fundamentada para a aplicação eficiente do PT na abordagem do SSP-NPM, sem necessidade de reescrita do paralelismo. As próximas seções apresentam os detalhes da implementação específica para adaptar a API para solução do SSP-NPM.

[1] Disponível em https://github.com/ALBA-CODES/PTAPI/ sob a licença *Creative Commons* BY-NC (CC BY-NC).

DOI: 10.59254/sbpo-2025-212298
http://live-proceedings.pantheonsite.io/p/212298

3.1. Codificação e decodificação

As soluções são representadas como permutações codificadas em um conjunto de inteiros de 1 a n, em que cada inteiro corresponde ao índice de uma tarefa. Observa-se que as soluções são codificadas da mesma maneira que para um problema de máquina única.

Durante a decodificação de uma solução, as tarefas são atribuídas às máquinas seguindo a ordem da permutação da codificação. No intuito de distribuir as tarefas de maneira equilibrada entre as máquinas, estima-se de modo heurístico quantas tarefas serão atribuídas a cada uma delas. Entretanto, sempre são geradas soluções válidas: ao distribuir cada tarefa, considera-se somente as máquinas elegíveis para aquela tarefa em específico, o que pode alterar a divisão de tarefas entre máquinas estimada inicialmente.

3.2. Soluções iniciais

As soluções iniciais são geradas aleatoriamente. Entretanto, no intuito de reduzir potenciais violações da restrição de elegibilidade de máquinas, a estrutura da solução é induzida heuristicamente: o vetor que armazena a solução codificada é dividido em m seções lógicas, uma para cada máquina, cujo tamanho é estimado de acordo com a quantidade de tarefas passíveis de processamento por cada máquina. Então, tarefas são atribuídas aleatoriamente somente às seções do vetor que representam as máquinas elegíveis.

3.3. Função de avaliação

Dado um sequenciamento de tarefas, o número de trocas de ferramentas é determinado por uma implementação sequencial Almeida et al. [2025a] do recente do *greedy pipe construction algorithm* (GPCA) [Qiu et al., 2025]. Este método tem uma complexidade amortizada limitada por $O(Cn)$, que é menor que a complexidade $O(mn)$ do método alternativo KTNS.

3.4. Movimentos

Estruturas de vizinhança padrão para problemas de permutação foram consideradas. Os movimentos operam diretamente nas soluções codificadas das seguintes maneiras:

2-opt: dois índices no vetor de tarefas são selecionados aleatoriamente e as posições de todos os elementos entre esses índices são invertidas;

2-swap: dois índices no vetor de tarefas são escolhidos aleatoriamente e os elementos correspondentes são trocados.

inserção aleatória: dois índices no vetor de tarefas são escolhidos aleatoriamente e o elemento correspondente ao primeiro índice é inserido da posição indicada pelo segundo índice;

operador aleatório: uma vizinhança, dentre as três especificadas, é selecionada aleatoriamente com igual probabilidade.

Todos os movimentos são realizados observando-se a restrição de elegibilidade de máquinas. Caso um movimento potencialmente viole a restrição, então inicia-se novamente a aplicação do operador.

3.5. Critério de parada

O número de propostas de trocas de temperaturas foi utilizado como o principal critério de parada, por ser uma contagem combinatória facilmente mensurável. O algoritmo registra a melhor solução encontrada por qualquer uma das réplicas e a retorna quando o critério de parada é atendido.

DOI: 10.59254/sbpo-2025-212298
http://live-proceedings.pantheonsite.io/p/212298

4. Experimentos computacionais

Os experimentos foram conduzidos em um computador com dois processadores *Intel*® *Xeon*® CPU E5-2660 v2 de 2,20 GHz e 10 núcleos, cada um com 20 *threads* e 384 GB de RAM, executando o sistema operacional CentOS Linux 7.9.2009. A API do PT foi implementada utilizando a linguagem de programação C++ e compilada usando o compilador GCC 10.1.0 com as opções -O3 e -lpthread. A implementação paralela do PT utilizou 40 *threads* para executar os componentes descritos na Seção 3. O número de *threads* foi determinado considerando a escalabilidade do método e a qualidade dos resultados.

Os métodos considerados para comparação tiveram seus códigos-fonte disponibilizados pelos autores e foram compilados e executados no mesmo ambiente do PT, utilizando os recursos computacionais especificados nos estudos originais ou no código-fonte. A implementação proposta neste estudo está disponível sob a licença *Creative Commons* BY-NC (CC BY-NC) em https://github.com/ALBA-CODES/NPM.

Conforme fornecido por Calmels [2022], o único conjunto de instâncias *benchmark* disponível para SSP-NPM é composto por 640 instâncias separadas em dois conjuntos, SSP-NPM-I e SSP-NPM-II, contendo 160 e 480 instâncias, respectivamente. A Tabela 3 apresenta as características de cada conjunto. Os títulos das colunas representam o número de instâncias ($\#$), tarefas (n), máquinas (m), ferramentas (f) e capacidade do magazine (c_i) de cada máquina i.

Tabela 3: Instâncias consideradas [Calmels, 2022].

conjunto	$\#$	n	m	f	c_i
SSP-NPM-I	80	{10, 15}	2	{10, 15}	[5, 7]
	80	{15, 20}	3	{15, 20}	[7, 10, 13]
SSP-NPM-II	320	{40, 60}	4	{60, 120}	[25, 30, 35, 40]
	160	{80, 120}	6	{120}	[45, 50, 55, 60, 65, 70]

4.1. Experimento preliminar

Para definir os valores dos parâmetros do PT e decidir entre diferentes componentes, foram conduzidos experimentos preliminares utilizando o método de configuração *offline* automática do *irace* [López-Ibáñez et al., 2016]. O conjunto de instâncias de treinamento representa 10% do total de instâncias consideradas em cada experimento, escolhidas aleatoriamente. Os parâmetros do PT ajustados neste experimento são elencados a seguir. A Tabela 4 apresenta os valores considerados e os selecionados (em negrito) pelo *irace*.

Temperaturas inicial e final: Definem os limites do intervalo de temperaturas a ser distribuído entre as réplicas;

Estratégia de distribuição inicial das temperaturas: Define como as temperaturas do intervalo são distribuídas entre as réplicas;

Quantidade de tentativas de troca entre réplicas: Controla o número de vezes em que trocas de temperaturas são propostas para réplicas em temperaturas adjacentes;

Método para atualização automática das temperaturas durante a execução: Determina de que maneira as temperaturas das réplicas podem ser ajustadas dinamicamente;

Taxa de ajuste das temperaturas: Define o intervalo de ajuste das temperaturas quando um método de atualização automática é utilizado;

DOI: 10.59254/sbpo-2025-212298
http://live-proceedings.pantheonsite.io/p/212298

Comprimento das cadeias de Markov: Define quantas soluções vizinhas serão geradas em cada execução do algoritmo de Metropolis.

Tabela 4: Parâmetros utilizados no *irace*.

Parâmetro	Valores
Temperatura inicial	{0,01; **0,1**; 0,2; 0,3}
Temperatura final	{1; 3; **5**}
Número de réplicas	{20; 30; **40**}
Comprimento da cadeia de Markov em cada temperatura	{200; 300; 400; **500**; 600}
Quantidade de propostas de trocas entre as temperaturas	{500; 1000; **2000**; 5000}
Distribuição inicial das temperaturas[a]	{**1**; 2; 3; 4}
Tipo de movimento[b]	{1; 2; 3; **4**}
Tipo do ajuste dinâmico das temperaturas[c]	{0; **1**; 2; 3}
Taxa de atualização das temperaturas	{400; 500; 666; **1000**}

[a] 1 - linear, 2 - linear-inverso, 3 - exponencial, 4 - progressão geométrica.
[b] 1 - 2-opt, 2 - 2-*swap*, 3 - inserção aleatória; seleção aleatória (1 ou 2) com igual probabilidade.
[c] 0 - desativado, 1 - 23%, 2 - iguala as taxas de aceitação, 3 - *feedback-optimized*.

4.2. Comparação com o estado da arte

Após definir a configuração final do PT, uma série de experimentos computacionais foi conduzida para avaliar a qualidade das soluções reportada em comparação com os resultados publicados pelo método estado da arte [Soares e Carvalho, 2024], uma implementação híbrida paralela de BRKGA e VND.

As Tabelas 5 e 6 apresentam os valores médios considerando dez execuções independentes dos métodos comparados para cada instância dos conjuntos SSP-NPM-I e SSP-NPM-II, considerando o objetivo de minimização do *makespan*. São apresentados o número de máquinas (m), tarefas (n), ferramentas (f) e valores médios para melhor solução (S^*), solução média (S), tempo de execução médio (T), expresso em segundos, e a diferença percentual relativa (RPR(%)) entre estes indicadores, calculada como $100 \times \frac{\text{PT - BRKGA}}{\text{BRKGA}}$. Os valores em negrito indicam os melhores resultados para cada grupo de instâncias.

Tabela 5: Resultados para as instâncias SSP-NPM-I - minimização do *makespan*.

m	n	f	BRKGA			PT				
			S^*	S	$T(s)$	S^*	S	$T(s)$	$RPR_{S^*}(\%)$	$RPR_S(\%)$
2	10	10	29,25	29,25	1,75	29,25	29,25	29,28	0,00%	0,00%
2	10	15	36,60	36,60	2,48	36,60	36,60	30,20	0,00%	0,00%
2	15	10	43,20	43,20	7,27	43,20	43,20	35,52	0,00%	0,00%
2	15	15	52,05	52,05	10,56	52,05	52,05	39,55	0,00%	0,00%
3	15	15	25,20	25,20	4,91	25,20	25,20	52,87	0,00%	0,00%
3	15	20	31,25	31,25	6,23	31,25	31,25	43,44	0,00%	0,00%
3	20	15	32,30	32,30	14,44	32,30	32,30	51,62	0,00%	0,00%
3	20	20	41,80	41,82	19,14	41,80	41,80	57,79	0,00%	-0,05%

São conhecidas as soluções ótimas para 87 das 160 instâncias do conjunto SSP-NPM-I, reportadas originalmente por Calmels [2022]. O PT foi capaz de atingir todas as soluções ótimas, bem como igualar os melhores valores das soluções para as demais 73 instâncias, em tempo inferior a 60 segundos em média. Em ambos os casos, o desvio padrão na qualidade das soluções é zero, assim como no caso do BRKGA na grande maioria das instâncias.

DOI: 10.59254/sbpo-2025-212298
http://live-proceedings.pantheonsite.io/p/212298

Tabela 6: Resultados para as instâncias SSP-NPM-II - minimização do *makespan*.

m	n	f	BRKGA			PT					
			S^*	S	$T(s)$	S^*	S	$T(s)$	$RPR_{S^*}(\%)$	$RPR_S(\%)$	$RPR_T(\%)$
4	40	60	303,59	305,26	727,62	**303,06**	**303,41**	107,25	-0,17	-0,61	-85,26
4	40	120	608,41	609,86	2158,34	**607,88**	**608,14**	97,67	-0,09	-0,28	-95,47
4	60	60	445,59	448,24	3397,35	**443,10**	**444,71**	182,31	-0,56	-0,79	-94,63
4	60	120	894,58	898,17	3607,23	**891,49**	**892,86**	149,23	-0,35	-0,59	-95,86
6	80	120	861,55	866,41	3611,23	**841,50**	**846,20**	436,09	-2,33	-2,33	-87,92
6	120	120	1320,16	1330,26	3661,32	**1257,83**	**1262,94**	782,05	-4,72	-5,06	-78,64

No menor conjunto de instâncias, o método PT apresentou rápida convergência, muitas vezes a partir da própria solução inicial. Em média, foram necessárias apenas 2 propostas de troca de temperaturas para atingir a convergência, com valores variando entre 0 e 263 propostas. Em contraste, no maior conjunto de instâncias, o número médio de propostas de troca exigido para a convergência aumentou para 725, com um máximo observado de 1999 propostas.

O PT demonstrou forte capacidade de aprimoramento das soluções, gerando novas melhores soluções para 243 instâncias, o que representa 37,96% do total. Além disso, para outras 386 instâncias (60,31%), o método conseguiu alcançar os melhores valores anteriormente conhecidos. Apenas 11 instâncias (1,72%) apresentaram soluções com qualidade inferior, com uma piora máxima de 2,47%.

Em termos de qualidade da solução, a melhoria obtida pelo PT em relação ao melhor valor conhecido chegou a 14,19%. Isso evidencia a eficiência do método não apenas em igualar, mas também em superar soluções de referência disponíveis na literatura ou obtidas por outros algoritmos.

No que diz respeito ao tempo computacional, o PT também se mostrou bastante eficiente. A redução no tempo de execução em comparação com o BRKGA atingiu até 98,45%, sem que nenhuma execução ultrapassasse o limite de 21 minutos, mesmo nas maiores instâncias avaliadas. Esse resultado reforça a aplicabilidade do PT em contextos que exigem soluções de alta qualidade dentro de restrições computacionais estritas.

5. Conclusão

Este estudo apresentou uma nova abordagem para o problema de sequenciamento de tarefas e trocas de ferramentas em máquinas paralelas não idênticas (SSP-NPM), um problema $\mathcal{NP}$-difícil com aplicações relevantes em ambientes industriais modernos. Foi considerada a minimização do *makespan* e proposta a aplicação do método *parallel tempering* (PT). O método proposto foi avaliado com base no único conjunto *benchmark* disponível na literatura, totalizando 640 instâncias. Os resultados mostraram-se superiores ou equivalentes aos melhores resultados disponíveis, alcançando todas as soluções ótimas conhecidas e apresentando valores de *RPR* médios negativos para todos os grupos de instâncias. Foram geradas novas melhores soluções para 243 instâncias (ou 37,96%), e as melhores soluções conhecidas foram equiparadas para outras 386 instâncias (ou 60,31%). Adicionalmente, a redução do tempo computacional em relação ao método estado da arte foi de até 98,45%. Essas evidências reforçam a competitividade e a robustez do método proposto frente ao estado da arte.

A versão do PT projetada para minimização do *makespan* servirá de base para o projeto de uma versão do mesmo algoritmo para a minimização do *flowtime* total, um objetivo relacionado mas não equivalente, também considerado na literatura sobre o SSP-NPM. Espera-se que os avanços aqui apresentados contribuam significativamente para o desenvolvimento da literatura sobre o SSP-NPM e motivem novas pesquisas em variantes relacionadas, como, por exemplo, tempos de troca não uniformes em uma mesma máquina, o compartilhamento de ferramentas entre máquinas, o desgaste de ferramentas e a indexação de ferramentas nas máquinas flexíveis.

DOI: 10.59254/sbpo-2025-212298
http://live-proceedings.pantheonsite.io/p/212298

Referências

Almeida, A. L. B., de Castro Lima, J., e Carvalho, M. A. M. (2025a). On serial and parallel evaluation functions for job sequencing and tool switching problems. *Computers & Operations Research*, 177:106969. ISSN 0305-0548.

Almeida, A. L. B., de Castro Lima, J., e Carvalho, M. A. M. (2025b). Revisiting the parallel tempering algorithm: High-performance computing and applications in operations research. *Computers & Operations Research*, 178:107000. ISSN 0305-0548.

Beezão, A. C., Cordeau, J.-F., Laporte, G., e Yanasse, H. H. (2017). Scheduling identical parallel machines with tooling constraints. *European journal of operational research*, 257(3):834–844.

Calmels, D. (2019). The job sequencing and tool switching problem: state-of-the-art literature review, classification, and trends. *International Journal of Production Research*, 57(15-16):5005–5025.

Calmels, D. (2022). An iterated local search procedure for the job sequencing and tool switching problem with non-identical parallel machines. *European Journal of Operational Research*, 297(1): 66–85.

Crama, Y., Kolen, A. W. J., Oerlemans, A. G., e Spieksma, F. C. R. (1994). Minimizing the number of tool switches on a flexible machine. *International Journal of Flexible Manufacturing Systems*, 6(1):33–54.

Cura, T. (2023). Hybridizing local searching with genetic algorithms for the job sequencing and tool switching problem with non-identical parallel machines. *Expert Systems with Applications*, 223: 119908.

Gan, Z. L., Musa, S. N., e Yap, H. J. (2023). A review of the high-mix, low-volume manufacturing industry. *Applied Sciences*, 13(3):1687.

Geyer, C. J. (1991). Markov chain monte carlo maximum likelihood. In *Computing Science and Statistics, Proceedings of the 23rd Symposium on the Interface*, p. 156–163. Interface Foundation of North America.

Gökgür, B., Hnich, B., e Özpeynirci, S. (2018). Parallel machine scheduling with tool loading: a constraint programming approach. *International Journal of Production Research*, p. 1–17.

Hukushima, K. e Nemoto, K. (1996). Exchange monte carlo method and application to spin glass simulations. *Journal of the Physical Society of Japan*, 65(6):1604–1608.

Jones, G. L. e Qin, Q. (2022). Markov chain monte carlo in practice. *Annual Review of Statistics and Its Application*, 9:557–578.

Khan, B. K., Gupta, B., Gupta, D. S., e Kumar, K. (2000). A generalized procedure for minimizing tool changeovers of two parallel and identical cnc machining centres. *Production Planning & Control*, 11(1):62–72.

López-Ibáñez, M., Dubois-Lacoste, J., Pérez Cáceres, L., Birattari, M., e Stützle, T. (2016). The irace package: Iterated racing for automatic algorithm configuration. *Operations Research Perspectives*, 3:43–58.

DOI: 10.59254/sbpo-2025-212298
http://live-proceedings.pantheonsite.io/p/212298

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., e Teller, E. (1953). Equation of state calculations by fast computing machines. *The journal of chemical physics*, 21(6):1087–1092.

Özpeynirci, S., Gökgür, B., e Hnich, B. (2016). Parallel machine scheduling with tool loading. *Applied Mathematical Modelling*, 40(9-10):5660–5671.

Qiu, Y., Cherniavskii, M., Goldengorin, B., e Pardalos, P. M. (2025). A computational study of the tool replacement problem. *INFORMS Journal on Computing*.

Soares, L. C. R. e Carvalho, M. A. M. (2020). Biased random-key genetic algorithm for scheduling identical parallel machines with tooling constraints. *European Journal of Operational Research*, 285(3):955–964.

Soares, L. C. e Carvalho, M. A. (2024). Biased random-key genetic algorithm for the job sequencing and tool switching problem with non-identical parallel machines. *Computers & Operations Research*, 163:106509.

Stecke, K. E. (1983). Formulation and solution of nonlinear integer production planning problems for flexible manufacturing systems. *Management Science*, 29(3):273–288.

Swendsen, R. H. e Wang, J.-S. (1986). Replica monte carlo simulation of spin-glasses. *Physical review letters*, 57(21):2607.

Tang, C. S. e Denardo, E. V. (1988). Models arising from a flexible manufacturing machine, part i: minimization of the number of tool switches. *Operations research*, 36(5):767–777.

Van Hop, N. (2005). The tool-switching problem with magazine capacity and tool size constraints. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, 35(5): 617–628.

Yadav, A. e Jayswal, S. (2018). Modelling of flexible manufacturing system: a review. *International Journal of Production Research*, 56(7):2464–2487.

DOI: 10.59254/sbpo-2025-212298
http://live-proceedings.pantheonsite.io/p/212298