São José dos Campos - SP
LV SBPO
LV Simpósio Brasileiro de Pesquisa Operacional
São José dos Campos, SP - 6 a 9 de novembro de 2023

Busca Local Iterada Aplicada ao Sequenciamento de Tarefas em Máquinas Flexíveis Paralelas Não Idênticas

Pedro Lucas Damasceno
Departamento de Computação, Universidade Federal de Ouro Preto
Campus Morro do Cruzeiro, Ouro Preto, Minas Gerais, 35400-000, Brasil
pedro.damasceno@aluno.ufop.edu.br

Marco Antonio Moreira de Carvalho
Departamento de Computação, Universidade Federal de Ouro Preto
Campus Morro do Cruzeiro, Ouro Preto, Minas Gerais, 35400-000, Brasil
mamc@ufop.edu.br

RESUMO

O sequenciamento de tarefas em máquinas flexíveis paralelas não idênticas é um problema que consiste em designar um conjunto de tarefas às máquinas flexíveis não idênticas de um sistema. Tais máquinas comportam ferramentas que são utilizadas para realizar diferentes tarefas, como parafusar ou lixar, por exemplo. Em contextos reais, comumente é impossível comportar todas as ferramentas do sistema simultaneamente, o que requer a interrupção das máquinas para a realização de trocas. Tais trocas devem ser minimizadas dadas as implicações financeiras envolvidas. Este problema *NP*-Difícil é abordado utilizando a metaheurística busca local iterada e são exibidos resultados inéditos para um conjunto de 640 instâncias da literatura. O método proposto foi comparado com o atual estado da arte e demonstrou ser competitivo, obtendo soluções até 16,6% melhores através de experimentos limitados a apenas uma hora de execução por instância.

PALAVRAS CHAVE. Sequenciamento, Máquinas Flexíveis, Busca Local Iterada.

Área Principal: POI, ADG&GP, MH

ABSTRACT

The job sequencing and tool switching problem with non-identical parallel machines consists of assigning a set of tasks to non-identical flexible machines. These machines are equipped with tools that are utilized for various tasks, such as screwing or sanding. In practical scenarios, it is often infeasible to load all the available tools on all machines simultaneously, resulting in machine interruptions for tool changes. Minimizing these tool changes is of utmost importance due to the significant financial implications associated with them. This *NP*-hard problem is addressed using the metaheuristic iterated local search. This study presents novel and unprecedented results for a comprehensive set of 640 instances sourced from the literature. Comparative evaluations against the current state of the art demonstrate the competitiveness of the proposed method, showcasing improvements of up to 16.6% in solution quality under the constraint of execution time per instance limited one-hour.

KEYWORDS. Sequencing. Flexible Machines. Iterated Local Search.

Main Area: OPI, AD&GP, MH

1. Introdução

As companhias de manufatura estão enfrentando desafios devido às recentes transformações do mercado consumidor e o interesse crescente por produtos customizados [Zawadzki e Żywicki, 2016]. Segundo o modelo fabril tradicional, uma nova linha de produção se faz necessária para produzir cada diferente produto; todavia, além dos entraves logísticos e financeiros envolvidos, a volatilidade com que os interesses do mercado se alteram [Panfilova et al., 2018] também inviabiliza essa forma de produção nos dias atuais. Dessa forma, para se adaptar à nova realidade socioeconômica e se manterem competitivas no mercado, as fábricas têm adotado o modelo de manufatura flexível, fato que deu origem aos conceitos de "indústria 4.0" e "*smart factories*" [Shrouf et al., 2014].

As principais características das *smart factories* são a alta flexibilidade, a automação da produção, a coleta e análise de dados em tempo real e a utilização de sistemas de máquinas flexíveis de manufatura (*Flexible Manufacturing System*, FMS) [Yadav e Jayswal, 2018]. Um FMS é caracterizado pela união de máquinas flexíveis através de uma linha de produção automatizada. Cada máquina é equipada com um *magazine* de capacidade limitada, que deve comportar ferramentas o suficiente para a realização de qualquer tarefa individualmente. Uma tarefa pode ser descrita como o processo de fabricação de um produto, para a qual várias ferramentas, como parafusadeiras, lixadeiras e outras, devem estar prontamente acopladas à máquina durante a execução da tarefa. Comumente é impossível comportar simultaneamente todas as ferramentas do sistema, o que faz necessário a interrupção das máquinas para a realização das trocas necessárias a fim de dar sequência ao processo de produção. Na maioria dos sistemas reais de manufatura, a troca de ferramentas e/ou componentes é o processo que consome mais tempo [Van Hop, 2005] e deve ser evitada sempre que possível, dadas as implicações financeiras decorrentes de uma linha de produção ociosa.

A fim de obter a melhor ordenação de um conjunto de tarefas, de modo a minimizar as trocas de ferramentas necessárias, foi definido o *minimization of tool switches problem* (MTSP) [Tang e Denardo, 1988], atualmente referido na literatura como *job sequencing and tool switching problem* (SSP). Nele estão contidos dois problemas: o sequenciamento das tarefas e a a determinação do plano de trocas de ferramentas. O sequenciamento é um problema *NP*-Difícil [Crama et al., 1994], ou seja, não se conhece algoritmo capaz de solucioná-lo de forma determinística em tempo polinomial. Já a determinação do plano de trocas pode ser realizada em tempo determinístico polinomial através da política de manutenção das ferramentas que serão utilizadas mais cedo, denominada *keep tool needed soonest* (KTNS) [Tang e Denardo, 1988], de complexidade $O(mn)$, onde m corresponde à quantidade de ferramentas necessárias para concluir as n tarefas atribuídas à máquina, ou através do algoritmo *greedy pipe construction algorithm* (GPCA) [Cherniavskii e Goldengorin, 2022], de complexidade $O(cn)$, em que c corresponde à capacidade do *magazine*.

Este estudo aborda a variação do SSP em que há um conjunto de máquinas flexíveis não-idênticas, conhecido como problema de sequenciamento de tarefas em máquinas flexíveis paralelas não idênticas (*job sequencing and tool switching problem with non-identical parallel machines*, SSP-NPM), definido primeiramente por Calmels [2022]. Formalmente, dado um FMS contendo um conjunto de ferramentas $F = \{1, ..., f\}$ no sistema; um conjunto de máquinas flexíveis não-idênticas paralelas $M = \{1, ..., m\}$, com capacidades para comportar até c_m ($m \in M$) ferramentas simultaneamente e um conjunto de tarefas $T = \{1, ..., t\}$ a serem processadas, o SSP-NPM consiste em determinar a designação e sequenciamento das tarefas em cada máquina. Além das capacidades dos *magazines*, as máquinas também diferem em relação ao tempo de processamento de cada tarefa p_{mt} ($m \in M$, $t \in T$) e ao tempo necessário para realização de uma troca de ferramentas sw_m ($m \in M$).

Em seu trabalho, Calmels [2019] considerou como objetivos a minimização das trocas de ferramentas necessárias durante a produção, o *makespan* (maior tempo decorrido, dentre todas as máquinas, desde o início da operação até o término da última tarefa processada) e o *flowtime* (soma dos tempos de conclusão de todas as tarefas), separadamente. Embora Calmels [2019] afirme que não há restrições de eligibilidade, foi constatada a existência dessa restrição em algumas instâncias propostas em seu trabalho. Diante disso, o problema foi tratado considerando essa característica e foram atribuídas às máquinas do FMS apenas tarefas cujo tamanho do subconjunto de ferramentas necessárias não exceda as capacidades dos *magazines*.

Para exemplificar o SSP-NPM, considere a instância "ins1_m=2_j=10_t=10_var=1", que descreve um FMS com $M = \{1, 2\}$, $F = \{1, ...,10\}$, $T = \{1, ...,10\}$, $C_m = \{5, 7\}$ e $SW_m = \{2, 4\}$. Os requerimentos de ferramentas por tarefa podem ser representadas por uma matriz binária $R^m = \{r^m_{ft}\}$, com $m \in M$, $f \in F$ e $t \in T$, na qual as linhas representam as ferramentas disponíveis e as colunas representam as tarefas designadas à máquina m, em ordem de processamento. Um elemento $r^m_{ft} = 1$ indica que a ferramenta f está carregada na máquina m durante o processamento da tarefa t, e $r^m_{ft} = 0$ caso contrário. A matriz exemplificada na Tabela 1 representa os requisitos de ferramentas por tarefas, e a Tabela 2 exibe os tempos de processamento de cada tarefa na instância de exemplo. Realizando o sequenciamento das tarefas de forma aleatória, obtemos a solução ilustrada na Tabela 3.

Tabela 1: Ferramentas requeridas por tarefa da instância de exemplo.

	Tarefas									
Ferramentas	1	2	3	4	5	6	7	8	9	10
1	0	0	1	0	1	0	0	1	0	1
2	1	1	0	0	1	0	1	1	0	0
3	0	0	1	1	0	1	1	1	1	0
4	1	0	0	1	0	0	0	0	1	1
5	0	1	1	0	0	1	1	1	1	0
6	0	0	0	0	0	0	0	0	1	1
7	0	1	1	0	1	1	0	1	0	0
8	1	0	0	0	0	0	0	0	0	1
9	0	0	1	1	0	1	1	0	0	1

Tabela 2: Tempos de processamento relativo às máquinas da instância de exemplo.

	Tarefas									
Máquinas	1	2	3	4	5	6	7	8	9	10
1	2	7	2	5	10	8	10	8	8	9
2	8	10	2	1	2	7	5	3	10	8

O número de trocas de ferramentas no FMS é calculado pela Equação (1). A equação soma as inversões de 0 para 1 nas matrizes R^i, que representam as trocas de ferramentas necessárias para a conclusão do processamento do subconjunto A_i de tarefas atribuídas à máquina i, tal que $i \in M$ e $A_i \subseteq T$. Nesta variação do SSP não se considera as inserções no *magazine* inicial como trocas. Na atribuição aleatória da Tabela 3, são realizadas 11 trocas no total; em uma solução ótima para esse objetivo, são realizadas apenas 3.

Tabela 3: Atribuição aleatória de tarefas para a instância de exemplo.

Máquina 1						**Máquina 2**					
Ferramentas	10	7	9	3	8	**Ferramentas**	1	4	6	2	5
1	1	0	0	1	1	1	0	0	0	0	1
2	0	1	0	0	1	2	1	0	0	1	1
3	0	1	1	1	1	3	0	1	1	0	0
4	1	0	1	0	0	4	1	1	0	0	0
5	0	1	1	1	1	5	0	0	1	1	0
6	1	0	1	0	0	6	0	0	0	0	0
7	0	0	0	1	1	7	0	0	1	1	1
8	1	0	0	0	0	8	1	0	0	0	0
9	1	1	0	1	0	9	0	1	1	0	0
10	0	0	1	0	0	10	0	0	1	1	0

Embora tenha sido abordado neste estudo, o objetivo de minimização de trocas de ferramentas, em um ambiente com máquinas que possuem tamanhos de *magazine* diferentes, pode não ser coerente com contextos reais. Isso se deve ao fato de que as tarefas tendem a se concentrar na máquina com o maior *magazine*, enquanto as demais serão subutilizadas, executando apenas um pequeno subconjunto de tarefas que não exigem nenhuma troca de ferramenta, uma vez que as inserções iniciais de ferramentas não são levadas em consideração. Logo, é importante revisar a relevância e aplicação desse objetivo em estudos futuros sobre o SSP-NPM.

$$\text{TS} = \sum_{i \in M} \sum_{j=2}^{j=|A_i|} \sum_{k \in F} r_{kj}^{i}(1 - r_{kj-1}^{i}) \tag{1}$$

O *makespan* é definido pela Equação (2), que soma o tempo de processamento das tarefas e o tempo gasto para efetuar as trocas de ferramentas individualmente para cada máquina do FMS. O valor do *makespan* corresponde ao maior tempo de término de processamento de uma máquina do FMS para a conclusão do seu subconjunto de tarefas designadas. Na atribuição aleatória da Tabela 3, o valor do *makespan* é de 55 unidades de tempo, enquanto em um solução ótima seria de apenas 29.

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

Por fim, o *flowtime* é calculado pela soma das variáveis k_i, que marcam os tempos de completude de cada uma das as tarefas processadas no FMS, como descrito na Equação (3). O tempo de completude de uma tarefa é calculado somando todo o tempo gasto pelas tarefas e trocas de ferramentas anteriores até a conclusão da tarefa em questão. Para a atribuição aleatória da Tabela 3, o *flowtime* total é de 274 unidades de tempo, enquanto a solução ótima para esse objetivo possui valor de 129.

$$\text{TFT} = \sum_{i \in T} k_i \tag{3}$$

Este estudo descreve o desenvolvimento e a aplicação da metaheurística busca local iterada (*iterated local search*, ILS) [Lourenço et al., 2003], combinada com as buscas locais estruturadas no formato de descida em vizinhança variável (*variable neighborhood descent*, VND) [Hansen et al., 2019], para a resolução do SSP-NPM. Os resultados obtidos são comparados ao método considerado o atual estado da arte[Cura, 2023], utilizando o mesmo conjunto de instâncias proposto por Calmels [2022] e sob as mesmas limitações de tempo de execução.

Este artigo está organizado como a seguir. Na Seção 2 é realizada uma revisão da restrita literatura do SSP-NPM, um problema de concepção recente. Na Seção 3, a heurística proposta é descrita em detalhes. Os experimentos computacionais e os resultados obtidos são descritos e apresentados na Seção 4. Por fim, na Seção 5 estão as conclusões e considerações finais acerca deste estudo e de trabalhos futuros.

2. Revisão da literatura

A literatura sobre o SSP conta com diversos trabalhos e uma ampla gama de variações. O SSP uniforme para uma única máquina, com tamanhos uniformes de ferramentas e tempos de configuração iguais e independentes de sequência, foi introduzido por Tang e Denardo [1988], que desenvolveram a política *keep tool needed soonest* para solucionar o subproblema de determinação do plano de trocas de ferramentas em tempo polinomial. Em seguida, Crama et al. [1994] provaram que o SSP se trata de um problema *NP*-difícil, o que levou maior foco ao desenvolvimento de métodos aproximados para a resolução do problema.

A primeira variação a considerar máquinas em paralelo foi elaborada por Khan et al. [2000], que propuseram uma heurística gulosa para a resolução do SSP com duas máquinas idênticas em paralelo. Posteriormente, Beezão et al. [2017] elaboraram dois modelos de programação linear inteira e uma metaheurística de busca adaptativa para a resolução do SSP com máquinas paralelas idênticas e tempos de processamento especificados. Ademais, em 2020, Soares e Carvalho [2020] demonstraram a aplicação do algoritmo genético de chaves aleatórias viciadas para a variação do SSP que considera máquinas paralelas idênticas com restrições de ferramentas.

Uma revisão geral da literatura acerca do SSP e suas variações foi elaborada por Calmels [2019], que abrangeu desde a primeira definição do problema até as variações datadas a 2018. Em Calmels [2022], a variação inédita do SSP abordada neste trabalho foi definida, a qual considera máquinas flexíveis paralelas não idênticas. No referido artigo, foi estabelecida uma formulação matemática para o problema e um método de busca local iterada foi implementado para propor soluções a um conjunto de 640 instâncias, também propostos no mesmo estudo.

Em Cura [2023], um método baseado na hibridização entre buscas locais e algoritmo genético foi elaborado, resultando em soluções significativamente melhores, com uma melhoria média de 13,34% em relação aos conjuntos de instâncias propostos por Calmels [2022]. No trabalho em questão, o autor utilizou, além do KTNS, a política RRT (*randomly removed tool*) para obter uma resolução mais rápida. embora não necessariamente ótima, do plano de trocas de ferramentas. Por fim, foram expostos os resultados obtidos através de diversos experimentos, também limitados a 1 hora de execução por instância, que combinaram as diferentes políticas e operadores de buscas locais.

Como evidenciado, toda a literatura sobre o SSP-NPM é muito recente, desde a definição proposta por Calmels [2022] à publicação do estado da arte, da autoria de Cura [2023]. Este trabalho se situa em sequência, e propõe a aplicação de buscas locais diferentes dos trabalhos anteriores relacionados ao SSP-NPM e uma nova determinação da metaheurística ILS. O método desenvolvido se prova capaz de produzir resultados competitivos em comparação ao atual estado da arte.

3. Métodos propostos

Este trabalho combina a metaheurística ILS – uma técnica que consiste em gerar uma solução inicial e realizar buscas locais e perturbações a fim de abrangir diferentes vizinhanças até que se atinja o critério de parada especificado – ao VND, que explora a vizinhança da solução atual para encontrar ótimos locais. Ademais, buscas locais diferentes dos trabalhos anteriores relacionados ao SSP-NPM foram implementadas e calibradas como descrito na Seção 4.1. A implementação do método e os resultados obtidos estão disponíveis *online* [1].

3.1. Heurística construtiva

A qualidade da solução inicial é de suma importância para a eficiência da ILS. Ao partir de uma solução ruim, mais iterações e, consequentemente, mais tempo são necessários para que ocorra a convergência da solução em direção a ótimos locais de menor custo. Diante disso, foi observado que o critério de maior relevância para a minimização de todos os objetivos tratados neste trabalho é a redução das trocas de ferramentas. Isso porque as diferenças dos tempos de processamento das tarefas é significativamente menos relevante em relação ao tempo consumido para a realização das trocas.

A heurística proposta para a construção de soluções iniciais baseia-se na minimização das inversões de 0 para 1 nas matrizes R^m. Para isso, cada máquina do FMS recebe uma tarefa aleatória. Em seguida, seleciona-se a máquina com a menor razão de trocas de ferramentas por tarefas atribuídas em todo o FMS. A essa máquina é atribuída a tarefa mais similar à última do seu sequenciamento. A similaridade é calculada somando-se todas as ferramentas necessárias e desnecessárias em comum entre a última tarefa atribuída à máquina e todas as outras tarefas ainda não atribuídas. A Tabela 4 exemplifica uma solução gerada para a instância exemplificada na Tabela 1 a partir da heurística construtiva, cujos valores distam de apenas 1, 3 e 38 unidades para as soluções ótimas em relação aos objetivos TS, FMAX e TFT, respectivamente.

Tabela 4: Solução inicial construída a partir da instância exemplificada na Tabela 1.

Máquina 1				
Ferramentas	0	9	3	8
1	0	1	0	0
2	1	0	0	0
3	0	0	1	1
4	1	1	1	1
5	0	0	0	1
6	0	1	0	1
7	0	0	0	0
8	1	1	0	0
9	0	1	1	0
10	0	0	0	1

Máquina 2						
Ferramentas	6	7	4	1	5	2
1	0	1	1	0	0	1
2	1	1	1	1	0	0
3	1	1	0	0	1	1
4	0	0	0	0	0	0
5	1	1	0	1	1	1
6	0	0	0	0	0	0
7	0	1	1	1	1	1
8	0	0	0	0	0	0
9	1	0	0	0	1	1
10	0	0	0	1	1	0

3.2. Buscas locais

As buscas locais são responsáveis por explorar a vizinhança da solução sob análise, realizando movimentos que a alteram de forma pontual. É importante que sejam métodos complementares, de forma que cada busca altere a solução de formas diferentes a fim de evitar a reanálise e a

[1] https://github.com/pedroldm/ic_ssp_npm

convergência prematura para um ótimo local não global. Neste estudo foram consideradas buscas locais de amplo uso no âmbito do SSP, e algumas são inéditas em relação a trabalhos anteriores sobre o SSP-NPM.

3.2.1. Busca local *job exchange*

Nesta busca local, um movimento consiste em trocar duas tarefas de posição. Para os objetivos de TS e TFT, essa busca abrange todos os pares de tarefas do FMS, o que a confere complexidade assintótica $\mathcal{O}(t^2)$. Para o objetivo FMAX, são consideradas apenas trocas de tarefas entre a máquina crítica – a responsável pelo *makespan* – e as outras máquinas. As trocas são realizadas apenas se não houver impedimento de eligibilidade. Considerando o exemplo da Tabela 3, um possível movimento consiste em trocar as tarefas 10 e 3 da máquina 1.

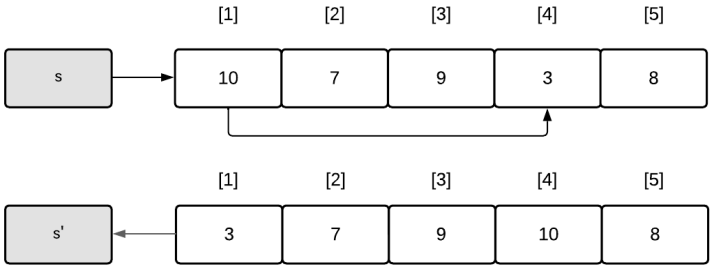

Figura 1: Exemplo de movimento realizado pela busca local *job exchange*.

3.2.2. Busca local *2-opt*

O *2-opt* é um algoritmo heurístico originalmente utilizado para melhorar soluções aproximadas para o problema do caixeiro-viajante (*traveling salesman problem*). Neste trabalho, o *2-opt* foi adaptado de forma a inverter intervalos de tarefas dos sequenciamentos das máquinas do FMS. Para os objetivos de TS e TFT, todos os possíveis pares de tarefas são examinados, resultando na complexidade assintótica $\mathcal{O}(t^2)$. Para o objetivo de FMAX, apenas pares da máquina crítica são considerados. Considerando o exemplo da Tabela 3, um possível movimento seria a inversão do intervalo compreendido entre as tarefas 10 e 3 da máquina 1.

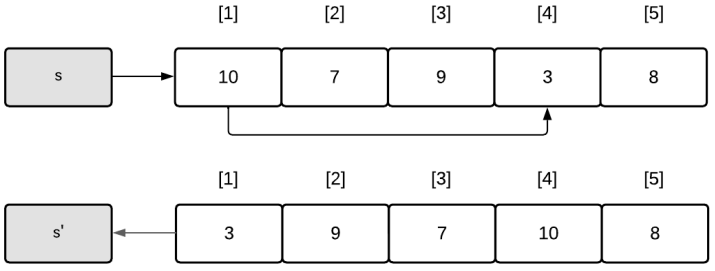

Figura 2: Exemplo de movimento realizado pela busca local *2-opt*.

3.2.3. Busca local *1-block*

O *1-block grouping* [Paiva e Carvalho, 2017] foi desenvolvido visando minimizar as inversões na matriz R^m. Primeiro, são identificados os blocos de um ou mais uns consecutivos em

uma linha. Em seguida, as colunas dos blocos são permutadas a fim de diminuir o número total de blocos. A complexidade assintótica do *1-block grouping* é $\mathcal{O}(ft^2)$ no pior caso, o que pode implicar na necessidade de limitar o número de soluções vizinhas a serem analisadas. Além disso, pode haver semelhança de blocos de uns entre as linhas, o que ocasiona na reanálise de soluções.

3.3. Descida em vizinhança variável

O refinamento das soluções a cada iteração da ILS foi implementado na estrutura do VND. Essa abordagem consiste na aplicação sequencial das buscas locais até que nenhuma delas apresente melhorias em relação à solução atualmente analisada. Caso alguma busca local encontre uma solução melhor, a função correspondente à busca retorna verdadeiro, a solução sob análise é atualizada e o VND é reiniciado a partir da primeira busca local. A ordem de execução das buscas locais foi disposta conforme exemplificado no Algoritmo 1, entre as linhas 2-11. Primeiramente, a busca local *job exchange* verifica todos os pares de trocas possíveis; em seguida, as buscas locais *2-opt* e *1-block* permutam o sequenciamento de tarefas das máquinas individualmente.

Algoritmo 1 Descida de Vizinhança Variável

1: $k \leftarrow 1$
2: **enquanto** $k \neq 4$ **faça**
3: **se** $k = 1$ **então**
4: **se** $jobExchangeLocalSearch(s')$ **então** $k \leftarrow 1$ **senão** $k \leftarrow k + 1$;
5: **fim se**
6: **se** $k = 2$ **então**
7: **se** $twoOptLocalSearch(s')$ **então** $k \leftarrow 1$ **senão** $k \leftarrow k + 1$;
8: **fim se**
9: **se** $k = 3$ **então**
10: **se** $oneBlockLocalSearch(s')$ **então** $k \leftarrow 1$ **senão** $k \leftarrow k + 1$;
11: **fim se**
12: **fim enquanto**

3.4. Busca local iterada

A busca local iterada é uma metaheurística que incorpora estratégias de busca local e perturbação sistemática de soluções, visando obter melhores resultados em problemas de otimização. Seu funcionamento baseia-se em duas fases principais: a fase de busca local e a fase de perturbação. Durante a fase de busca local, uma solução inicial é gerada e refinada por meio de movimentos que exploram a vizinhança da solução atual. Esses movimentos buscam aprimorar a solução, porém podem se deparar com ótimos locais que restringem a melhoria. Nesse ponto, a fase de perturbação introduz alterações mais drásticas na solução, com o intuito de escapar desses ótimos locais e explorar novas regiões do espaço de busca. A partir da solução perturbada, a busca local é novamente aplicada, iniciando um ciclo iterativo.

A implementação proposta da da busca local iterada está representada no Algoritmo 2. Na primeira e segunda linha, uma solução inicial é gerada através da heurística construtiva detalhada na Seção 3.1 e refinada pelo VND, detalhado na Seção 3.3. O laço das linhas 5-12 repetem o processo de perturbar a solução e realizar as buscas locais na nova vizinhança. Conforme a condição das linhas 8-9, a solução é aceita se seu custo for menor ou igual à da melhor solução encontrada até então. A aceitação de soluções mesmo custo foi adotada para permitir que o método escape de ótimos locais em instâncias pequenas.

Algoritmo 2 Busca Local Iterada

```
1: função ILS
2:     s ← heuristicaConstrutiva();
3:     s ← VND(s);
4:     iteracoes ← 0
5:     enquanto iteracoes < maxIteracoes faça
6:         s' ← perturbarSolucao(s);
7:         s' ← VND(s*);
8:         se s' ⩽ s então
9:             s ← s';
10:        fim se
11:        iteracoes ← iteracoes + 1;
12:    fim enquanto
13: fim função
```

4. Experimentos computacionais

O ambiente computacional utilizado para a realização dos experimentos consiste em um computador com processador Intel(R) Core(TM) i7-10700 CPU @ 2.90GHz, 16GB de memória RAM DDR4 3200MHz, e sob o sistema operacional Ubuntu 22.04.1 LTS. O método proposto foi implementado em C++ e compilado utilizando o GCC versão 11.3.0, com as opções de otimização -O3 e march=native. Assim como feito por Cura [2023], o tempo de execução máximo para uma instância foi de 3.600 segundos. Devido aos componentes de aleatoriedade, o método foi executado de forma independente 10 vezes para cada instância.

As instâncias propostas por Calmels [2022] estão disponíveis *online* [2]. Elas estão divididas em dois subconjuntos: SSP-NPM-I e SSP-NPM-II. O primeiro subconjunto consiste em grupos de 20 instâncias que envolvem no máximo 3 máquinas, 20 ferramentas e 20 tarefas. Já o segundo subconjunto é composto por grupos de 80 instâncias que possuem até 6 máquinas, 120 ferramentas e 120 tarefas. Além disso, o segundo subconjunto é subdividido em instâncias densas e esparsas – referente à quantidade de ferramentas necessárias para as tarefas – e em instâncias de alto e baixo custo de trocas de ferramentas.

4.1. Experimentos preliminares

A disposição das buscas locais, o método e grau de perturbação utilizado na ILS e o número máximo de iterações foram definidos conforme testes estatísticos realizados pelo pacote iRace [López-Ibáñez et al., 2016]. Todas as possibilidades de ordenação das buscas foram avaliadas, inclusive a utilização ou não de cada método, e a considerada melhor está ilustrada no Algoritmo 1. A busca local *1-block* foi limitada a 25% das ferramentas, selecionadas de forma aleatória. Isso se mostrou necessário devido à alta complexidade assintótica do método, como detalhado na Seção 3.2.3

A perturbação foi projetada para redistribuir aleatoriamente uma quantidade específica de tarefas entre as máquinas. Essa quantidade é ajustada com base no número de tarefas do sistema, seguindo a proporção $\alpha \times |T|$. Nos testes, os possíveis valores de α foram 0.03, 0.05 ou 0.07, sendo 0.05 o considerado melhor. Ademais, as iterações da ILS foram limitadas a 1.000, que se mostraram suficientes para a convergência em instâncias pequenas ou médias. Em instâncias maiores, o critério de parada foi, majoritariamente, por tempo de execução.

[2] https://github.com/TerhiS/ILS_SSP-NPM

4.2. Comparação com o estado da arte

As Tabelas 5 e 6 apresentam as comparações entre as médias dos melhores resultados reportados pela ILS para cada conjunto de instâncias e os valores correspondentes ao estado da arte. O *gap(%)* foi calculado seguindo a fórmula $100 \times \frac{\text{ILS-AGH}}{AGH}$, sendo AGH os resultados do estado da arte [Cura, 2023] obtidos pela hibridização entre algoritmo genético e buscas locais. Nos conjuntos do SSP-NPM-I, contendo 20 instâncias cada, a ILS reportou resultados superiores ou equivalentes em relação a todos os objetivos e grupos de instâncias. Alguns dos resultados diferiram em poucas unidades, evidenciando a importância de levar casas decimais em consideração em pesquisas futuras, a fim de uma avaliação mais precisa dessas pequenas discrepâncias.

Tabela 5: Resultados obtidos para o conjunto de instâncias SSP-NPM-I.

M	T	F	$\mathbf{TS_{AGH}}$	$\mathbf{TS_{ILS}}$	$\mathbf{gap}(\%)$	$\mathbf{FMAX_{AGH}}$	$\mathbf{FMAX_{ILS}}$	$\mathbf{gap}(\%)$	$\mathbf{TFT_{AGH}}$	$\mathbf{TFT_{ILS}}$	$\mathbf{gap}(\%)$
2	10	10	3	3	0,00	29	29	0,00	130	**128**	-1,54
2	10	15	6	6	0,00	37	37	0,00	164	**163**	-0,61
2	15	10	4	4	0,00	43	**42**	-2,33	273	**268**	-1,83
2	15	15	10	**9**	-10,00	53	**52**	-1,89	341	**338**	-0,88
3	15	15	1	1	0,00	25	25	0,00	160	160	0,00
3	15	20	5	5	0,00	31	31	0,00	192	**191**	-0,52
3	20	15	2	2	0,00	33	**32**	-3,03	275	**273**	-0,73
3	20	20	8	**7**	-12,50	43	**42**	-2,33	356	**352**	-1,12

Para o subconjunto SSP-NPM-II, que contém 80 instâncias cada, a ILS demonstrou resultados significativamente superiores em todos os grupos de instâncias, apresentando uma média de redução de -29,96%, -13,41% e -12,21% nos objetivos de minimização de trocas de ferramentas, *makespan* e *flowtime*, respectivamente.

Tabela 6: Resultados obtidos para o conjunto de instâncias SSP-NPM-II.

M	T	F	$\mathbf{TS_{AGH}}$	$\mathbf{TS_{ILS}}$	$\mathbf{gap}(\%)$	$\mathbf{FMAX_{AGH}}$	$\mathbf{FMAX_{ILS}}$	$\mathbf{gap}(\%)$	$\mathbf{TFT_{AGH}}$	$\mathbf{TFT_{ILS}}$	$\mathbf{gap}(\%)$
4	40	60	127	**89**	-29,92	325	**276**	-15,08	4998	**4362**	-12,73
4	40	120	304	**253**	-16,78	622	**525**	-15,59	9672	**8625**	-10,83
4	60	60	213	**135**	-36,62	509	**425**	-16,50	11995	**9997**	-16,66
4	60	120	487	**388**	-20,33	942	**810**	-14,01	22620	**19978**	-11,68
6	80	120	691	**502**	-27,35	923	**802**	-13,11	29299	**25337**	-13,52
6	120	120	1137	**787**	-30,78	1468	**1377**	-6,20	68983	**63544**	-7,88

No entanto, a ILS apresentou um elevado desvio padrão em relação ao grupo de maiores instâncias (6 máquinas, 120 tarefas e 120 ferramentas). Essa variação ocorre devido à dificuldade do método em alcançar a convergência dentro do limite de tempo estabelecido para a execução, de 3.600 segundos. A Tabela 7 apresenta dados relativos à média geral (μ), ao desvio padrão (σ) e ao tempo de execução (T), expresso em segundos, obtidos através da ILS, referentes a cada conjunto de instâncias. É possível concluir que o método proposto foi capaz de produzir boas soluções com baixo desvio padrão para a maioria dos conjuntos de instâncias. Ademais, o tempo limite foi atingido apenas nos dois maiores grupos de instâncias, o que indica a capacidade do método de alcançar bons resultados em tempo razoável.

5. Conclusões e trabalhos futuros

O *job sequencing and tool switching problem with non-identical parallel machines* é um problema *NP*-Difícil com ampla aplicação no contexto das indústrias de manufatura. Neste trabalho, foram propostos uma heurística construtiva e um método que combina a metaheurística busca local iterada e a heurística de busca local descida de vizinhança variável.

Tabela 7: Média geral, desvio padrão e tempo de execução reportados pela ILS.

M	T	F	**TS**	μ	σ	**T**	**FMAX**	μ	σ	**T**	**TFT**	μ	σ	**T**
2	10	10	3	3	0,04	0,6	29	29	0,02	0,6	128	130	0,01	0,6
2	10	15	6	7	0,04	0,6	37	37	0,02	0,7	163	164	0,00	0,7
2	15	10	4	4	0,03	2,2	42	43	0,02	2,0	268	272	0,01	2,3
2	15	15	9	9	0,03	2,4	52	53	0,01	2,3	338	340	0,01	2,7
3	15	15	1	1	0,07	2,5	25	26	0,01	1,6	160	160	0,00	2,6
3	15	20	5	5	0,13	2,9	31	32	0,01	2,0	191	192	0,00	3,0
3	20	15	2	2	0,18	6,3	32	33	0,01	3,1	273	273	0,00	4,9
3	20	20	7	7	0,13	8,0	42	42	0,01	4,6	352	353	0,01	8,7
4	40	60	89	91	0,02	409,0	276	288	0,03	61,0	4362	4478	0,02	244,1
4	40	120	253	258	0,02	453,5	525	549	0,03	68,6	8625	8819	0,02	269,2
4	60	60	135	139	0,03	2734,1	425	439	0,02	261,8	9997	10225	0,01	3092,5
4	60	120	388	398	0,02	2859,8	810	843	0,03	271,7	19978	20415	0,01	2966,8
6	80	120	502	534	0,04	3613,4	802	837	0,05	1548,4	25337	26073	0,03	3610,9
6	120	120	787	846	0,05	3648,8	1377	2255	0,29	3652,7	63544	92620	0,25	3723,9

Experimentos intensivos foram realizados considerando o único conjunto de 640 instâncias da literatura, divididos em 14 subconjuntos. Resultados melhores ou equivalentes foram obtidos para todos os subconjuntos; entretanto, o método apresentou alto desvio padrão em relação ao maior conjunto de instâncias, com 6 máquinas e 120 tarefas a serem executadas. Isso se deve à limitação do tempo de execução de apenas 1 hora, que dificulta a convergência do método em instâncias maiores.

As próximas etapas de pesquisa irão se concentrar na exploração de outras configurações de busca local e na melhor implementação do operador de inserção. Esse operador é responsável por alocar uma tarefa na posição mais adequada dentro das sequências de todas as máquinas. O objetivo é aumentar a consistência dos resultados obtidos para o maior subconjunto de instâncias, sem comprometer a qualidade dos resultados obtidos para os subconjuntos de instâncias pequenas e médias, que apresentaram desvio padrão próximo de zero. Além disso, é importante rever o limite de tempo definido para a execução e considerar a relevância do objetivo de trocas de ferramentas de forma independente, a fim de garantir uma contextualização mais adequada à realidade.

6. Agradecimentos

Esta pesquisa foi apoiada pelo Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq), pela Fundação de Apoio à Pesquisa do Estado de Minas Gerais (FAPEMIG) e pela Universidade Federal de Ouro Preto (UFOP).

Referências

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.

Cherniavskii, M. e Goldengorin, B. (2022). An almost linear time complexity algorithm for the tool loading problem.

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*, 6:33–54.

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. ISSN 0957-4174.

Hansen, P., Mladenović, N., Brimberg, J., e Pérez, J. A. M. (2019). *Variable neighborhood search*. Springer.

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., Cáceres, L. P., Birattari, M., e Stützle, T. (2016). The irace package: Iterated racing for automatic algorithm configuration. *Operations Research Perspectives*, 3:43–58.

Lourenço, H. R., Martin, O. C., e Stützle, T. (2003). *Iterated local search*. Springer.

Paiva, G. S. e Carvalho, M. A. M. (2017). Improved heuristic algorithms for the job sequencing and tool switching problem. *Computers & Operations Research*, 88:208–219. ISSN 0305-0548.

Panfilova, O., Okrepilov, V., e Kuzmina, S. (2018). Globalization impact on consumption and distribution in society. In *Matec web of conferences*, volume 170, p. 01032. EDP Sciences.

Shrouf, F., Ordieres, J., e Miragliotta, G. (2014). Smart factories in industry 4.0: A review of the concept and of energy management approached in production based on the internet of things paradigm. In *2014 IEEE international conference on industrial engineering and engineering management*, p. 697–701. IEEE.

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. ISSN 0377-2217.

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.

Zawadzki, P. e Żywicki, K. (2016). Smart product design and production control for effective mass customization in the industry 4.0 concept. *Management and production engineering review*.