ILS Aplicada à Minimização do Uso de Estoque Intermediário em Sistemas Industriais

Douglas Matuzalem Pontes Belo Lança
Departamento de Ciência da Computação, Universidade Federal de Ouro Preto
Campus Morro do Cruzeiro, Ouro Preto, Minas Gerais, 35400-000, Brasil.
douglasvandersar@gmail.com

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

RESUMO

Relacionado aos problemas de sequenciamento de tarefas, o Problema de Minimização de Espalhamento de Ordens (ou *Minimization of Open Orders*, MORP) ocorre em contextos industriais e trata do sequenciamento da fabricação de produtos em uma linha de produção com o objetivo de minimizar a utilização de estoque intermediário e o tempo de entrega dos produtos acabados. Este problema, classificado como NP-Difícil, possui ampla aplicação prática e correlação com outros problemas relevantes na literatura. Neste artigo é reportada a aplicação da metaheurística Busca Local Iterada (*Iterated Local Search*, ILS) para solução do MORP. Esta é a primeira aplicação reportada do método ILS à solução deste problema. Os experimentos computacionais envolveram 865 instâncias de quatro diferentes conjuntos da literatura. Os resultados reportados mostram que o método proposto foi capaz de estabelecer novos melhores resultados para algumas destas instâncias, quando comparado aos métodos considerados o estado da arte em relação ao MORP.

PALAVRAS CHAVE. Escalonamento. Espalhamento de Ordens, Sequenciamento de Tarefas.

IND, MH.

ABSTRACT

Related to task sequencing problems, the Minimization of Order Spread (MORP) arises in industrial contexts and address the sequencing of product manufacturing in a production line with the objective of minimizing the use of intermediate stocking and the speed of finished products delivery. This problem, classified as NP-Hard, has wide practical application and is related to other relevant problems in the literature. In this paper we report the application of the Iterated Local Search (ILS) metaheuristic for solving the MORP. This is the first reported application of the ILS method to solving this problem. The computational experiments involved 865 instances of four different sets from the literature. The reported results show that the proposed method was able to establish new best results for some of these instances when compared to the state-of-the-art methods related to the MORP.

KEYWORDS. Scheduling. Order Spread. Job Sequencing.

IND, MH.

1. Introdução

Em contextos industriais, é comum o cenário de fabricação de diferentes tipos de produtos, os quais possuem demandas específicas, de acordo as ordens de compra efetuadas por clientes. É necessário atender estas ordens de compra e despachar os produtos acabados tão rápido quanto possível, de maneira a agilizar o tempo de atendimento aos clientes. Outro efeito atingido por esta agilidade é a redução do uso do estoque intermediário, que consiste no armazenamento temporário dos produtos fabricados, porém, ainda não preparados para serem despachados. Especificamente no contexto considerado neste trabalho, os produtos de uma mesma ordem de compra só podem ser despachados em conjuntos completos. A utilização de estoque intermediário implica em utilização de mão de obra e maquinário adicionais para manipulação dos produtos e área física disponível para tanto. Além disto, os produtos podem ser danificados devido a acidentes ou intempéries.

Considera-se que a fabricação dos produtos é realizada em *estágios*, sendo que em cada um dos estágios um tipo de produto é fabricado. O intervalo entre o início e o fim da fabricação dos produtos de uma mesma ordem de compra é chamado de *espalhamento da ordem*, e é medido pelo número de estágios necessários para concluir a fabricação de todos os produtos de uma ordem de compra específica de um cliente, incluindo aqueles em que há interrupção da fabricação dos produtos desta ordem de compra.

O *Problema de Minimização de Espalhamento de Ordens* (ou *Minimization of Order Spread*, MORP) é um problema de sequenciamento de produção proposto originalmente por Madsen [1988] no contexto de operações de corte na indústria de vidro, no qual a estocagem intermediária das peças é custosa. Este é um problema $\mathcal{NP}$-Difícil [Garey e Johnson, 1979] que consiste em, dadas a relação de produtos a serem fabricados e as ordens de compra por estes produtos, determinar a sequência de fabricação dos produtos tal que o espalhamento (médio ou o máximo) das ordens de compra, seja o minimizado. Quando o espalhamento das ordens de compra é minimizada, o tempo em que os clientes aguardam pelo atendimento de seus pedidos – e consequentemente a utilização do estoque intermediário – é minimizado. Há também a tendência a minimizar a heterogeneidade das características físicas dos produtos, quando a mesma matéria prima bruta é compartilhada entre os diferentes produtos.

Formalmente, o MORP considera um conjunto J de produtos distintos para atender a demanda de um conjunto I de ordens de compra, formadas por tais produtos. Uma instância para este problema pode ser representada como uma matriz binária $P = \{p_{ij}\}$ que relaciona ordens de compra e produtos. Caso o produto $j \in J$ esteja presente na ordem de compra $i \in I$, tem-se $p_{ij} = 1$. Caso contrário, $p_{ij} = 0$. A Tabela 1 apresenta um exemplo de instância para o MORP. As linhas, numeradas de i_1 a i_6, representam as ordens de compras dos clientes. As colunas, numeradas de j_1 a j_6, representam os produtos.

Tabela 1: Matriz P, instância MORP.

	j_1	j_2	j_3	j_4	j_5	j_6
i_1	1	1	0	0	0	0
i_2	1	0	1	0	0	0
i_3	0	0	0	1	1	0
i_4	0	0	0	1	0	1
i_5	0	1	0	0	1	0
i_6	0	0	1	0	0	1

Uma solução para o MORP é uma permutação π das $|J|$ colunas da matriz P gerando uma matriz permutação $Q^\pi = \{q^\pi_{ij}\}$, cujas colunas seguem a ordem estabelecida em π e representam os estágios de fabricação dos produtos. O espalhamento de uma ordem é dado pela diferença entre o número dos estágios em que o último e o primeiro de seus produtos foram fabricados. Desta forma, uma ordem de compra que possui apenas um produto, não possui espalhamento. O espalhamento

médio é dado pela média de todos os espalhamento de ordens, ao passo que o espalhamento máximo é dado pelo maior espalhamento de uma ordem de compra.

A Tabela 2 representa uma possível solução π=[j_5, j_2, j_4, j_6, j_3, j_1] para a instância representada pela Tabela 1. Esta solução possui espalhamento médio de valor 1,67 e espalhamento máximo de valor 4, referente à ordem i_1. Nesta ordem compra, temos que o último e o primeiro dos produtos da ordem i_1 foram produzidos nos estágios 6 e 2 respectivamente, portanto, temos 6-2= 4.

Tabela 2: Matriz Q^{π} para π=[j_5, j_2, j_4, j_6, j_3, j_1].

	j_5	j_2	j_4	j_6	j_3	j_1
i_1	0	1	0	0	0	1
i_2	0	0	0	0	1	1
i_3	1	0	1	0	0	0
i_4	0	0	1	1	0	0
i_5	1	1	0	0	0	0
i_6	0	0	0	1	1	0

Neste trabalho, propõe-se a solução do MORP pela aplicação de uma heurística e pela aplicação da metaheurística Busca Local Iterada (*Iterated Local Search* – ILS) em uma fase posterior de aprimoramento. Esta é a primeira aplicação da ILS ao MORP reportada na literatura, e, de acordo com os experimentos computacionais reportados, este método foi capaz de aprimorar alguns dos melhores resultados da literatura.

O restante do trabalho está organizado da seguinte maneira:a revisão da restrita literatura sobre o MORP é apresentada na Seção 2; a Seção 3 detalha as implementações do método proposto composto por duas fases: geração de solução inicial e busca local iterada; o método é comparado com os melhores resultados da literatura utilizando diferentes conjuntos de instâncias da literatura na Seção 4; por fim, conclusões são realizadas a respeito do trabalho e futuras direções para o trabalho de pesquisa são apontadas na Seção 5.

2. Revisão da Literatura

A literatura sobre o MORP conta com poucos trabalhos. O problema foi proposto por Madsen [1988] ao estudar o processo de corte na indústria de vidro. Foi proposta uma estratégia de solução em três estágios. No primeiro, resolve-se o problema de corte de estoque (definição dos padrões de corte) sem levar em consideração o espalhamento de ordens. No segundo estágio, constrói-se uma matriz de distâncias entre produtos que reflete a frequência em que produtos são comprados simultaneamente. No terceiro estágio resolve-se o Problema do Caixeiro Viajante, considerando-se os produtos como vértices e a matriz de distâncias calculada no estágio anterior. A sequência dos vértices na rota estabelecida como solução do Problema do Caixeiro Viajante determina a ordem de fabricação dos produtos, visando minimizar o espalhamento de ordens. O resultado apresentado pelo autor mostrou a diminuição em média de 18% do espalhamento de ordens.

Em Foerster e Wäscher [1998], foi proposto o uso da metaheurística *Simulated Annealing* (SA) para solução do MORP. Os autores implementaram o procedimento de Madsen [1988], o procedimento 3-*opt* e o SA utilizando as mesmas instâncias usadas por Madsen [1988]. Nos experimentos computacionais realizados, o procedimento 3-*opt* superou o procedimento de Madsen [1988] e o SA superou o 3-*opt*. O SA apresentou redução considerável na média dos espalhamentos das ordens, porém, o procedimento de Madsen [1988] foi melhor em relação ao tempo de execução.

No ano seguinte, Fink e Voß [1999], empregaram diferentes heurísticas para solução do Problema de Minimização de Pilhas Abertas (um problema correlato) e do MORP. Neste trabalho, foram geradas diferentes soluções iniciais pelos métodos de Inserção Mais Barata (*Cheapest Insertion*) e Inserção Mais Barata do Pior Padrão (*Cheapest Insertion of the Worst Pattern*). Para a

otimização destas soluções iniciais foram utilizados os métodos 2-*opt*, uma nova implementação de SA e cinco variações estáticas e dinâmicas da busca tabu. Os métodos implementados foram comparados com métodos 3-*opt* e a implementação de SA proposta por Foerster e Wäscher [1998]. Os melhores resultados foram obtidos pela busca tabu reativa (ou TSRE5000, também considerada nos experimentos computacionais deste trabalho), melhorando a qualidade de solução inicial em até $49,5\%$. O SA proposto por Foerster e Wäscher [1998] atingiu resultados entre $4,9\%$ e $15,9\%$ distantes dos melhores valores encontrados. As instâncias utilizadas foram geradas aleatoriamente, e gentilmente fornecidas para testes neste trabalho.

De Giovanni et al. [2013] propuseram dois algoritmos para resolver o Problema de Minimização de Custo de Conexões entre Portas, um problema equivalente ao MORP, cujo objetivo é minimizar o comprimento dos fios utilizados na produção de circuitos eletrônicos, no intuito de minimizar a área dos mesmos. Neste problema, o comprimento do fio necessário para conectar os componentes de um circuito equivale ao espalhamento de uma ordem no MORP. Foram propostos uma formulação de programação inteira e um algoritmo *branch and cut* para sua solução. Os experimentos computacionais reportados consideraram três diferentes conjuntos de instâncias reais e artificiais da literatura, e o *branch and cut* foi capaz de determinar soluções ótimas ou subótimas (devido a restrições quanto ao tempo de execução) para todas as instâncias analisadas. Estas mesmas instâncias são consideradas neste trabalho.

Mais recentemente, Kim et al. [2016] propuseram uma formulação matemática e uma heurística para resolver o MORP na produção de molduras para janelas, na qual os produtos são peças de quatro tipos que, quando montadas, formam uma moldura de janela e as ordens de compra são compostas por tais molduras de janelas. A heurística proposta modela o MORP como o Problema da Mochila e utiliza diferentes procedimentos de melhoria. Foram realizados testes com 20 instâncias reais, e a heurística obteve resultados eficientes, tendo sido adotada pela indústria onde originou-se o estudo.

3. Métodos Propostos

A proposta deste trabalho é a implementação de uma heurística em duas fases para solução do MORP. Na primeira, gera-se uma solução inicial por meio de uma heurística, a qual é aprimorada em uma segunda fase pela aplicação da metaheurística busca local iterada. A seguir, são apresentados os detalhes das implementações propostas, auxiliados por pseudo-códigos e exemplos de funcionamento.

3.1. Geração da Solução Inicial

A heurística proposta para geração de soluções iniciais consiste em construir uma solução viável através do método *Best Insertion* e posteriormente aplicar o clássico método de busca local *2-swap* e o recente método de agrupamento de 1-*blocks*, descritos a seguir.

3.1.1. *Best Insertion*

De maneira geral, o clássico método *Best Insertion* atua sobre uma solução viável para um problema e, iterativamente seleciona cada um dos elementos desta solução e o re-insere na melhor posição possível de acordo com a função objetivo, verificando todas as possibilidades de inserção. A seleção dos elementos pode ser feita de forma aleatória ou de acordo com critérios que dependem das características do problema. O método termina quando todos os elementos da solução tenham sido analisados.

O método *Best Insertion* aplicado ao MORP considera cada elemento sendo um produto, porém, ao invés de partir de uma solução completa, este método a construirá. Esta aplicação construtiva parte de uma solução π vazia e adiciona cada um dos produtos, iterativamente, na melhor posição possível, até que se obtenha uma solução π completa. Por exemplo, o método inicia com uma permutação parcial para π contendo apenas um produto, então, o segundo produto é inserido

à frente e após o primeiro, permanecendo a configuração de melhor valor da função objetivo. Para a inserção do terceiro produto, analisa-se todas as possibilidades de inserção e o mesmo é inserido na posição que configurar a melhor permutação parcial. Os demais produtos são inseridos de forma análoga ao terceiro até que uma permutação π completa seja construída.

A Figura 1 ilustra a construção de uma solução composta por três elementos e cuja função objetivo deve ser minimizada. A construção inicia com o elemento 0, então, o método verifica qual a melhor posição para a inserção do elemento 1, ou seja, antes ou depois do elemento 0 inserindo o elemento 1 na posição que melhor satisfaça a função objetivo. Neste caso é preciso minimizar o problema, portanto foi escolhida a inserção que resultou no valor 3 para a função objetivo. Para a inserção do elemento 2, o método verifica todas as possibilidades de inserção e o mantém na posição que consiste em melhor configuração da solução.

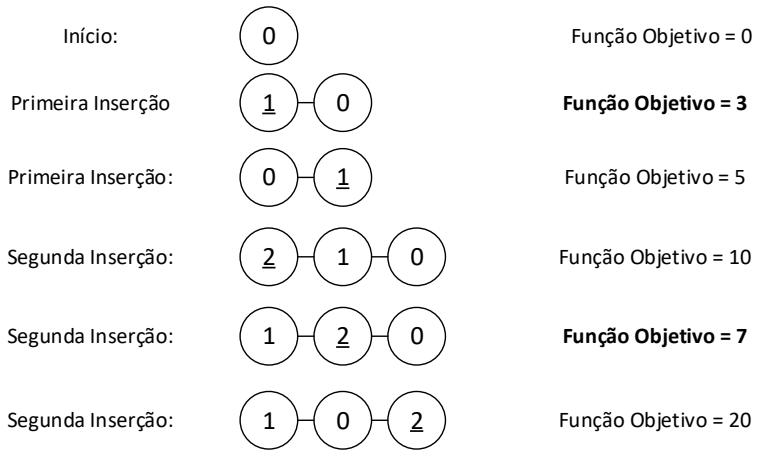

Figura 1: Exemplo da execução do método *Best Insertion*.

O Algoritmo 1 apresenta o pseudocódigo do método *Best Insertion*. O método recebe a lista de produtos J e atribui a π o primeiro produto (linha 2), escolhido de forma aleatória. Então, cada produto pertencente a J (laço das linhas 3-11) ainda não inserido na solução π (condição das linhas 4-10) tem sua inserção analisada em todas as posições possíveis da solução em construção (laço das linhas 5-8) e mantido na posição que melhor satisfaça a função objetivo (linha 9). Ao final, uma solução completa é retornada (linha 12).

Algoritmo 1: *Best Insertion*

```
1  Entrada: Lista de produtos J.
2  π ← j ∈ J|j selecionado aleatoriamente;
3  para cada produto j ∈ J faça
4      se j ∉ π então
5          para cada posição i em π faça
6              analise a inserção de j em π na posição i;
7              k ← melhor posição para j;
8          fim
9          insira j em π na posição k;
10     fim
11 fim
12 retorna π;
```

Após a construção de uma solução viável pelo método apresentado, a mesma é refinada

com a execução das buscas locais *2-swap* e agrupamento de 1-*blocks*. Estes dois métodos de busca local são apresentados na sequência.

3.1.2. Busca Local 2-*swap*

O conhecido movimento de 2-*swap* (ou 2-*trocas*), consiste em, dada uma solução para um problema permutacional, como o MORP, trocar dois elementos de posição. Este movimento é utilizado comumente como mecanismo de perturbação de solução e também como método de busca local. Neste último caso, o movimento é embutido em um método de descida rápida, ou seja, dada uma solução viável, o movimento 2-*swap* é aplicado sucessivamente e somente as trocas que gerem melhoria do valor da solução são concretizadas. A cada melhoria, o método é reiniciado e, quando todas as possíveis trocas forem analisadas, sem melhoria possível, o método termina.

A Figura 2 exemplifica a execução de um movimento 2-*swap*. A solução contém os elementos de 1 a 5 na ordem [3, 2, 4, 1 e 5]. Os elementos 2 e 5, sublinhados, foram escolhidos aleatoriamente, portanto, são trocados de posição, resultando na solução [3, 5, 4, 1, 2].

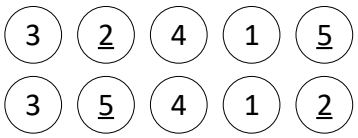

Figura 2: Exemplo da execução do método 2-*swap*.

O Algoritmo 2 apresenta o pseudocódigo da busca local 2-*swap*. O algoritmo recebe a solução π, então, sorteia duas posições aleatórias do vetor π (linhas 4 e 5) e troca os elementos correspondentes entre si (linha 6). O algoritmo verifica se a nova configuração é melhor que anterior (linha 7). Caso positivo, a nova configuração é mantida (linha 8); caso contrário, a troca é desfeita (linha 11). O algoritmo executa essas operações enquanto a solução for aprimorada (laço de repetição da linha 2 à linha 13) e então retorna o melhor resultado encontrado (linha 14).

Algoritmo 2: *2-swap*

```
1  Entrada: solução π.
2  enquanto houver melhoria da solução faça
3  |  π' ← π
4  |  i ← posição escolhida aleatoriamente;
5  |  j ← posição escolhida aleatoriamente;
6  |  π[i] troca de posição com π[j];
7  |  se π melhor que π' então
8  |  |  π' ← π;
9  |  fim
10 |  senão
11 |  |  π[j] troca de posição com π[i];
12 |  fim
13 fim
14 retorna π';
```

3.1.3. Busca Local por Agrupamento de 1-*Blocks*

Recentemente, a busca local por Agrupamento de 1-*Blocks* foi proposta por Paiva e Carvalho [2016], aplicada ao Problema de Minimização de Trocas de Ferramentas, um problema correlato ao MORP, embora não equivalente. Um 1-*block* é definido como uma sequência contígua de elementos não nulos em uma mesma linha de uma matriz binária. Dois ou mais 1-*blocks* são separados

em uma mesma linha por *0-blocks* – analogamente, uma sequência contígua de elementos nulos. No contexto do MORP, um 1-*block* representa uma sequência contígua de fabricação de produtos de uma mesma ordem de compra e um 0-*block* representa uma descontinuidade na produção dos elementos de uma mesma ordem de compra.

O agrupamento de 1-*blocks* é uma busca local que tem como objetivo minimizar o número de 1-*blocks* em cada linha de uma matriz binária, o que no contexto do MORP equivale a minimizar o espalhamento de cada uma das ordens de compra em Q^π. Esta busca local, apresentada no Algoritmo 3 é definida da seguinte forma: cada linha da matriz binária (linhas 1 a 11) é analisada em busca de dois ou mais 1-*blocks* (linhas 2 e 3) e então, movimenta-se todas as colunas do primeiro 1-*block*, uma a uma, para antes (representado pela função *antes*) ou depois (representado pela função *depois*) das colunas do segundo 1-*block* (linhas 4 a 8). Cada coluna só é movimentada caso não haja piora no valor da solução; a não movimentação de colunas é representada pela função *nenhum*. Aplicando-se sucessivamente estas operações, busca-se agrupar os 1-*blocks* de cada linha.

Algoritmo 3: Agrupamento de 1-*Blocks*.

```
   Input: matriz Q^π
 1 para cada linha r ∈ Q^π aleatoriamente selecionada faça
 2     determine o primeiro 1-block i em r;
 3     enquanto existir um próximo 1-block j em r faça
 4         para cada coluna k em i faça
 5             movimentos ←{depois(k, j), antes(k, j), nenhum()};
 6             proximo_movimento ← movimento ∈ movimentos que resulta na melhor
                 solução;
 7             perform(proximo_movimento);
 8         fim
 9         i ← j;
10     fim
11 fim
12 retorna Q^π
```

Para ilustrar a aplicação desta busca local, a Tabela 3(a) apresenta a matriz Q^π para $\pi = [1, 2, 3, 4, 5]$. Inicialmente, a primeira linha é examinada, e os dois 1-*blocks* compostos pelas colunas [1, 2] e pela coluna [5] são identificados. As duas possibilidades de movimentação da coluna 1 geram as soluções $[2, 3, 4, 5, 1]$ e $[2, 3, 4, 1, 5]$, esta segunda com menor espalhamento e que passa a ser a nova solução. Em seguida, as possibilidades de movimentação da coluna 2 geram as soluções $[3, 4, 1, 5, 2]$ e $[3, 4, 1, 2, 5]$, ambas com o mesmo espalhamento. Como não há prejuízo ao valor da solução, o primeiro movimento é realizado, resultando na solução apresentada pela Tabela 3(b). A busca local analisa as demais linhas da mesma maneira, até que toda a matriz tenha sido considerada.

Tabela 3: Exemplo de aplicação do agrupamento de 1-*blocks*.

π	1	2	3	4	5
	1	1	0	0	1
	1	0	0	1	0
	0	1	1	1	0
	1	1	1	0	1
	0	0	1	1	0
	0	0	0	0	1

(a)

π	3	4	1	5	2
	0	0	1	1	1
	0	1	1	0	0
	1	1	0	0	1
	1	0	1	1	1
	1	1	0	0	0
	0	0	0	1	0

(b)

3.2. Busca Local Iterada

A Busca Local Iterada (ou *Iterated Local Search*, ILS) proposta por Lourenço et al. [2003], é uma meta-heurística que alterna iterativamente etapas de intensificação e diversificação da busca no espaço de soluções. Estas operações são realizadas por meio da aplicação sucessiva de métodos de busca local e métodos de perturbação a partir de uma solução inicial viável. Ao aplicarmos um método de busca local, existe a possibilidade do mesmo ficar preso em um ótimo local, de forma a reduzir a área de exploração no espaço de soluções. Para contornar esta dificuldade, a ILS realiza perturbações, ou seja, sutis modificações na estrutura da solução, com o objetivo de obter uma nova solução que esteja localizada em uma região diferente do espaço de soluções. Desta forma, aumenta-se a área de exploração do método e também as chances de encontrar uma solução ótima global.

A Figura 3 demonstra em seu lado esquerdo o espaço de busca de um problema de minimização hipotético, em que o ponto A representa a solução inicial. Ao aplicarmos uma busca local, a solução se encontra no ponto B, um mínimo local. Dado que somente movimentos de melhora são realizados pela busca local, não será possível obter outra solução. Após aplicar o método de busca, a ILS perturba a solução na tentativa de gerar uma nova solução que esteja localizada em outra região do espaço de soluções. O lado direito da Figura 3 demonstra a operação de perturbação. A linha pontilhada representa a perturbação da solução B, que resulta em uma nova solução C. Após a perturbação, a ILS executa novamente a busca local resultando em uma nova solução ótima local, representada pelo ponto D.

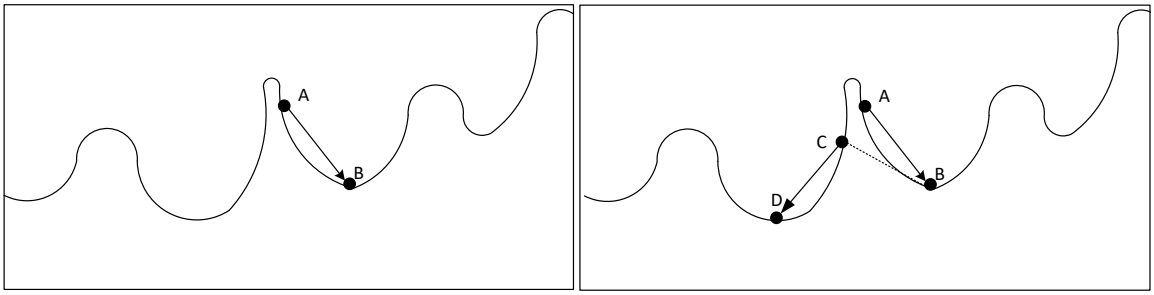

Figura 3: Ilustração da execução da ILS.

Neste trabalho, a ILS foi composta pelos métodos 2-*swap* e agrupamento de 1-*blocks*, descritos anteriormente, como mecanismos de busca local, além do método 2-*opt* como mecanismo de perturbação. Proposto por Croes [1958], método 2-*opt* é aplicado em problemas permutacionais e consiste em inverter as posições dos elementos dentro de um dado intervalo. Por exemplo, seja a solução π=[j_1, j_2, j_3, j_4, j_5, j_6] e o intervalo aberto (2,6). Após a aplicação do método 2-*opt* temos π=[j_1, j_2, j_5, j_4, j_3, j_6], dada a inversão dos elementos nas posições de 3 a 5. Cada solução é perturbada, independente do valor da solução resultante, em uma proporção α, definida *a priori*. A Seção 4.1 apresenta brevemente a definição deste e de outros parâmetros da ILS, como percentual do espaço de soluções explorado pela busca local e número máximo de iterações do método.

O Algoritmo 4 apresenta o pseudocódigo da ILS proposta aplicada ao MORP. O algoritmo recebe a solução inicial π, a proporção de aplicação da perturbação α, a proporção de aplicação da busca local β e o número máximo de iterações *iter_max*. A cada uma das *iter_max* iterações (laço das linhas 3 a 11), o método executa as operações de perturbação (linha 5) e busca local (linhas 6 e 7) nas proporções definidas por α e β, atualizando a solução π em caso de melhora (linhas 8, 9 e 10). Ao final, é retornada a solução π de melhor valor obtido (linha 12).

Algoritmo 4: ILS.

Entrada: π, α, β, *iter_max*.
aplique a busca local de agrupamento de 1-blocks;
enquanto *o número de iterações* < *iter_max* **faça**
    $\pi' \leftarrow \pi$;
    perturbe a solução π' em uma proporção β aplicando 2-*opt*;
    aplique a busca local *2-swap* em π' em uma proporção β;
    aplique a busca local de agrupamento de 1-*blocks* em π';
    se π' *melhor que* π **então**
        $\pi \leftarrow \pi'$;
    fim
fim
retorna π;

4. Experimentos Computacionais

Os experimentos computacionais foram divididos em duas etapas: experimentos preliminares e comparação de resultados. Na primeira etapa, foram conduzidos experimentos para o ajuste dos parâmetros utilizados pelo método proposto, ao passo que na segunda etapa os experimentos realizados compararam a versão final do método proposto com os melhores resultados da literatura. O ambiente computacional adotado para todos os experimentos consiste em um computador com processador *Intel i5 Quad Core* de 2.27 GHz com 4 GB RAM sob o sistema operacional Windows 10. O código do método proposto foi escrito em C++, compilado com g++ 4.4.1 e a opção de otimização -O3.

4.1. Ajuste de Parâmetros

A ILS aplicada ao MORP possui três principais parâmetros que influenciam diretamente no desempenho do algoritmo: porcentagem de busca local, porcentagem de perturbação e número máximo de iterações. A relação entre estes três parâmetros é preponderante para o tempo de execução e qualidade das soluções geradas. Para a escolha dos melhores valores para estes parâmetros foi utilizada a ferramenta *irace* [López-Ibáñez et al., 2016], um pacote que implementa uma série de procedimentos de configuração automática e, em particular, o procedimento de corrida iterada, que vem sendo usado com sucesso para configurar automaticamente metaheurísticas.

Para os parâmetros foram pré-selecionados alguns valores conforme apresentado na Tabela 4, em que a coluna *Parâmetros* indica qual parâmetro foi analisado e a coluna *Valores* apresenta o conjunto dos diferentes valores possíveis para cada parâmetro.

Tabela 4: Ajuste de parâmetros.

Parâmetros	Valores
Busca Local (%)	$10, 20, 30, 40, 50, 60, 70, 80, 90, 100$
Perturbação (%)	$5, 10, 15, 20, 25, 30, 35, 40, 45, 50$
Número de Iterações	$50, 100, 150, 200$

Dentre as opções sugeridas pela ferramenta, optou-se pelos valores 100% de busca local, perturbação de 20% da solução e 100 iterações como critério de parada da ILS. Como pode ser verificado na seção seguinte, estes valores resultaram em baixo tempo de execução e soluções de boa qualidade.

4.2. Comparação de Resultados

Devido à utilização de componentes de aleatoriedade pela ILS, o método foi executado independentemente dez vezes por instância, sendo que os melhores resultados obtidos e a média de

todos os resultados obtidos são utilizados nas análises realizadas a seguir. Apresenta-se os espalhamentos de ordens obtidos e o tempo de execução do método proposto e dos métodos que compõem o estado da arte. Todavia, ressalta-se que a comparação direta entre os tempos de execução não pode ser realizada, uma vez que os métodos foram executados em arquiteturas computacionais distintas.

As tabelas apresentadas a seguir são divididas em três seções. A primeira seção é referente à características das instâncias e incluem, quando disponível, o nome (*instância*), o número de ordens de compra (m), o número de produtos (n) e a densidade (v). A segunda seção é referente aos melhores resultados encontrados na literatura, incluindo o espalhamento médio (BKS) e o tempo médio de execução (T). A terceira seção é referente aos resultados obtidos pelo método proposto (*ILS-MORP*), indicando a solução média (S), a melhor solução obtida (S^*), a distância percentual entre melhor resultado obtido e o melhor resultado da literatura (gap), calculado como $100 \times (S^* - BKS)/BKS$), o desvio padrão em dez execuções independentes (σ) e o tempo médio de execução (T), em segundos.

4.2.1. Comparação com Fink e Voß [1999]

Este conjunto de 797 instâncias geradas aleatoriamente, são as mesmas utilizadas em Fink e Voß [1999], gentilmente cedidas pelo autor. O parâmetro v representa o nível de densidade das matrizes, de maneira que, quanto menor o valor de v, mais densa é a instância. Há 4 conjuntos de instâncias para $m \approx 50$ e $v = \{0,25, 0,5, 0,75, 1\}$ com 100 instâncias cada, e 4 conjuntos de instâncias para $m \approx 60$ com 100 instâncias para $v = \{0,25, 0,5\}$, 98 instâncias para $v = 0,75$ e 99 instâncias para $v = 1$. Para todas as instâncias, o número de colunas (n) é próximo ao número de linhas.

A Tabela 5 apresenta a comparação entre o método proposto e a busca tabu reativa (*TSRE500*) proposta por Fink e Voß [1999]. A busca tabu reativa possui os melhores resultados reportados na literatura para estas instâncias.

Tabela 5: Comparação com a busca tabu reativa de Fink e Voß [1999].

m	v	*TSRE5000*		*ILS-MORP*				
		BKS	T	S	S^*	gap	σ	T
50	0,25	14,04	142,6	14,90	14,31	1,92	0,49	0,48
50	0,50	12,43	79,5	15,03	14,56	17,13	0,30	0,44
50	0,75	4,87	58,6	3,04	2,71	-37,57	0,24	0,41
50	1,00	1,38	35,1	1,48	1,29	-6,52	0,19	0,70
60	0,25	16,41	159,5	18,18	17,44	6,27	0,54	0,97
60	0,50	14,61	142,6	16,84	16,35	11,90	0,35	0,80
60	0,75	5,33	98,2	4,18	3,79	-28,89	0,31	0,92
60	1,00	1,45	58,9	1,66	1,32	-8,96	0,29	0,83

De acordo com os valores apresentados, observa-se que a ILS proposta obteve bons resultados, alcançando gap médio de $-5,59\%$, o que indica que novos melhores resultados foram obtidos para algumas das instâncias. Com efeito, o método obteve melhores resultados significativos para instâncias esparsas (i.e., $v = 0,75$ e $v = 1,00$), igualando os melhores resultados em outros 330 casos, dentre 397. Para instâncias com densidade $v \in \{0,25, 0,50\}$, o algoritmo demonstrou boa convergência, encontrando o melhor resultado após executar, em média, 48,69% das iterações.

O aprimoramento da solução pela ILS em relação a solução inicial foi significativo para instâncias mais densas, obtendo aprimoramento máximo de 41,22%, suprindo a dificuldade do método de geração da solução inicial nestes casos. O tempo de execução se manteve abaixo de um segundo para qualquer instância, independente da densidade ou das dimensões. Adicionalmente, nota-se que o método proposto é consistente, vide o baixo desvio padrão do valor das soluções em execuções independentes.

4.2.2. Comparação com De Giovanni et al. [2013]

Conforme descrito anteriormente, De Giovanni et al. [2013] propuseram um algoritmo *Branch and Cut* (*B&C*) para solução de um problema equivalente ao MORP. Os experimentos computacionais reportados originalmente consideraram 11 instâncias reais advindas da indústria microeletrônica, 33 instâncias artificiais da literatura e 24 instâncias reais advindas de indústrias de produtos de madeira, totalizando 68 instâncias. A Tabela 6 apresenta a comparação entre os resultados obtidos pelo método proposto e o *B&C*. Ressalta-se que as linhas *Shaw*, *Wood A* e *Wood B* são referem a grupos com 25, 4 e 11 instâncias, respectivamente.

Tabela 6: Comparação com o *branch and cut* De Giovanni et al. [2013].

instância	*B&C*		*ILS-MORP*				
	BKS	T	S	S^*	gap	σ	T
wli	24	0,00	24,00	24	0,00	0,00	0,01
wsn	96	0,50	101,10	100	4,17	0,74	0,05
v4000	56	0,20	52,60	51	-8,93	0,97	0,02
v4050	38	0,20	38,80	38	0,00	0,42	0,01
v4090	109	0,80	116,50	113	3,67	1,96	0,07
v4470	237	4,20	278,00	265	11,81	17,85	0,21
x0	298	5,30	334,90	315	5,70	18,80	0,23
w1	39	0,30	44,50	41	5,13	2,76	0,02
w2	233	2,00	254,90	251	7,73	3,70	0,16
w3	668	21,60	815,10	735	10,03	54,24	1,59
w4	1683	29,10	2427,30	2214	31,55	152,41	23,27
Shaw	184,32	0,19	185,82	183,96	-0,20	1,2	0,05
GP5	9341	15,09	9340,80	9340	-0,01	1,03	10,85
GP6	7359	27,23	7360,50	7359	0,00	1,96	8,89
GP7	7366	23,11	7377,10	7367	0,01	21,98	8,14
GP8	5858	37,93	5858,00	5858	0,00	0,00	8,22
NWRS7	316	4,46	364,30	317	0,32	25,37	0,42
NWRS8	591	7,53	613,30	604	2,20	8,51	0,40
SP3	1622	26,28	1767,40	1690	4,19	43,52	1,33
SP4	3450	56,79	3584,40	3390	-1,74	114,67	3,52
A_FA+AA-_1	184	4,08	218,90	194	5,43	14,96	0,44
A_FA+AA-_2	58	0,13	57,90	51	-12,07	3,38	0,07
A_FA+AA-_6	91	0,19	94,60	91	0,00	1,51	0,09
A_FA+AA-_8	112	0,65	130,50	121	8,04	7,09	0,17
A_FA+AA-_11	87	0,4	94,90	89	2,30	3,25	0,20
A_FA+AA-_12	48	0,15	53,70	48	0,00	4,06	0,08
A_FA+AA-_13	304	1,2	327,40	315	3,62	11,03	0,55
A_FA+AA-_15	52	0,1	47,30	44	-15,38	1,70	0,05
B_REVAL_145	129	0,5	175,90	157	21,71	19,11	0,41
Wood A	35,25	0,07	37,25	35,25	0,00	1,1	0,03
Wood B	35,73	0,06	39,84	39,1	9,43	0,62	0,02

O método proposto se igualou ao método comparado em 10 instâncias, encontrou melhores soluções em 30 instâncias e obteve soluções inferiores em outras 28 instâncias, com gap médio de 3,18%. Com efeito, a solução inicial gerada pela heurística proposta encontrou melhores resultados para 19 das instâncias, portanto, antes da fase de aprimoramento pela aplicação da ILS.

Os tempos de execução se mantiveram baixos para este conjunto de instâncias, embora tenham aumentado para instâncias de maior dimensão, como *w4*. Da mesma maneira, o método proposto obteve maior desvio padrão para instâncias grandes, como *w4*, *SP3* e *SP4*, indicando maior dificuldade para resolvê-las de maneira consistente. De maneira geral, a ILS demonstrou boa convergência, encontrando o melhor resultado após executar 45,02% das iterações, em média. Além disso, a ILS aprimorou a solução inicial de forma significativa, com máximo de 34,67%.

5. Conclusões

Neste trabalho abordou-se o Problema de Minimização de Espalhamento de Ordens (*Minimization of Order Spread* – MORP), um problema de sequenciamento de tarefas de ampla aplicação industrial prática. Foi proposta uma implementação de uma heurística para geração de soluções iniciais e também uma implementação da metaheurística Busca Local Iterada (ou *Iterated Local Serach* – ILS).

Experimentos computacionais extensivos foram realizados considerando 865 instâncias da literatura, incluindo instâncias reais. Os resultados gerados foram comparados com dois métodos da literatura, reconhecidos por gerarem as melhores soluções para as instâncias consideradas. A metaheurística proposta foi capaz de gerar novos melhores resultados para parte das instâncias consideradas, além de igualar ou se aproximar dos melhores resultados conhecidos em outros casos, em baixo tempo de execução. Entretanto, o método demonstrou dificuldades na solução de instâncias densas e também de maiores dimensões. Destaca-se também a qualidade da solução inicial gerada pela heurística proposta, a qual contribuiu para a rápida convergência da ILS. Em alguns casos, a solução inicial igualou as melhores soluções conhecidas.

Os trabalhos futuros se concentrarão em estudar abordagens para melhoria dos resultados para instâncias mais densas e também para aumentar a consistência dos resultados obtidos para instâncias de maiores dimensões.

6. Agradecimentos

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

Referências

Croes, G. A. (1958). A method for solving traveling-salesman problems. *Oper. Res.*, 6(6):791–812.

De Giovanni, L., Massi, G., Pezzella, F., Pfetsch, M., Rinaldi, G., e Ventura, P. (2013). A heuristic and an exact method for the gate matrix connection cost minimization problem. *Int. Trans. in Oper. Res.*, 20(5):627–643.

Fink, A. e Voß, S. (1999). Applications of modern heuristic search methods to pattern sequencing problems. *Comp. & Oper. Res.*, 26(1):17 – 34. ISSN 0305-0548.

Foerster, H. e Wäscher, G. (1998). Euro best applied paper competition simulated annealing for order spread minimization in sequencing cutting patterns. *Eur. J. of Oper. Res.*, 110(2):272 – 281. ISSN 0377-2217.

Garey, M. R. e Johnson, D. S. (1979). *A Guide to the Theory of NP-Completeness*. WH Freemann, New York.

Kim, B.-I., Ki, Y., Son, D., Bae, B., e Park, J.-S. (2016). An algorithm for a cutting problem in window frame production. *Int. J. of Prod. Res.*, 54(14):4327–4339.

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

Lourenço, H. R., Martin, O. C., e Stützle, T. (2003). Iterated local search. In *Handbook of metaheuristics*, p. 320–353. Springer.

Madsen, O. B. G. (1988). An application of travelling-salesman routines to solve pattern-allocation problems in the glass industry. *J. of the Oper. Res. Soc.*, 39(3):249 – 256. ISSN 0377-2217.

Paiva, G. S. e Carvalho, M. A. M. (2016). Um método para planejamento da produção em sistemas de manufatura flexível. In *Anais do XLVIII Simpósio Brasileiro de Pesquisa Operacional*, p. 3717–3724.