# Métodos de Descida Rápida e Descida Em Vizinhança Variável Aplicados à Resolução do Problema de Minimização de Pilhas Abertas

**Júnior Rhis Lima**
Departamento de Ciência da Computação, Universidade Federal de Ouro Preto
Campus Morro do Cruzeiro, Ouro Preto, Minas Gerais, 35400-000, Brasil.
juniorrhis1@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 corte de estoque, o Problema de Minimização de Pilhas Abertas é um problema de sequenciamento de padrões de corte cujo objetivo é minimizar a utilização de estoque intermediário, bem como a manipulação desnecessária dos produtos fabricados. Neste artigo é reportada a aplicação de dois métodos heurísticos a este problema NP-Difícil de ampla aplicação prática: Descida em Vizinhança Variável (*Variable Neighborhood Descent*) e Descida Mais Rápida (*Steepest Descent*). Salvo melhor juízo, esta é a primeira aplicação reportada do método Descida em Vizinhança Variável à solução deste problema. Os experimentos computacionais envolveram 595 instâncias de cinco diferentes conjuntos da literatura. Os resultados reportados demonstram que não houve dominância entre os métodos propostos e que os mesmos foram capazes de obter boa parte das soluções ótimas para as instâncias consideradas, obtendo uma pequena distância percentual das mesmas nos demais casos, demonstrando robustez.

**PALAVRAS CHAVE. Sequenciamento de Padrões. Heurísticas. Problema de Minimização de Pilhas Abertas.**

**Tópicos: IND, MH, OC.**

**ABSTRACT**

Related to cutting stock problems, the Minimization Open Stacks Problem is a cutting pattern sequencing problem, whose objective is to minimize the use of intermediate stock, as well as the unnecessary handling of manufactured products. This paper reports the application of two heuristics to this NP-Hard problem of wide practical application: Variable Neighborhood Descent and Steepest Descent. To the best of our knowledge, this is the first reported application of Variable Neighborhood Descent to the solution of this problem. Computational experiments involved 595 instances from five different set of the literature. The reported results demonstrate that there was no dominance between the proposed methods and that they were able to obtain a good part of the optimal solutions for the instances considered, abtaining a small gap in the other cases, demonstrating robustness.

**KEYWORDS. Pattern Sequencing. Heuristics. Minimization Open Stacks Problem.**

**Topics: IND, MH, OC.**

## 1. Introdução

Em contextos industriais, problemas de corte de estoque são problemas de otimização combinatória que consistem em cortar unidades maiores de matéria prima para obter unidades menores (ou peças) satisfazendo-se algum objetivo, como por exemplo, minimizar as sobras das unidades maiores. A disposição espacial das peças dentro de unidades maiores para realização do corte define um padrão de corte. Após a definição destes padrões de corte, passa-se à etapa de processamento dos mesmos. A sequência na qual os diferentes padrões são cortados pode afetar diferentes objetivos, tais como a utilização de estoque intermediário, o ritmo de atendimento a diferentes ordens de compra e a homogeneidade das características físicas das peças produzidas. Torna-se desejável então determinar o sequenciamento dos padrões a serem processados de acordo com tais critérios, definindo assim os problemas de sequenciamento de padrões.

Depois de produzida, cada peça é mantida em uma pilha que armazena peças de um mesmo tipo ao redor da máquina que as produziu. Quando a última peça de um tipo for produzida, a pilha referente a este tipo específico de peça pode ser removida da área de produção para outro local. Considera-se que há uma limitação física para o armazenamento de pilhas no ambiente de produção, de modo que não é possível haver pilhas abertas para todos os tipos de peça simultaneamente. Desse modo, deseja-se minimizar o número de pilhas abertas simultaneamente por meio do sequenciamento dos padrões, caracterizando assim o Problema de Minimização de Pilhas Abertas (ou MOSP, de *Minimization of Open Stacks Problem*).

O MOSP, como descrito por [Yuen, 1995] remonta à um ambiente de produção em que uma única máquina de corte processa um conjunto $P$ de diferentes padrões para produzir um conjunto $C$ de peças com demandas específicas. A produção é dividida em estágios, e é considerado que em cada estágio um lote de um determinado padrão de corte diferente é processado, ou seja, todas as cópias de um padrão de corte específico devem ser cortadas antes que um padrão de corte diferente seja processado. Durante a produção de um determinado tipo de peça, todas as suas cópias são armazenadas temporariamente em uma pilha mantida ao redor da máquina que as produziu; cada pilha armazena somente um único tipo específico de peça. Quando a primeira peça de um dado tipo tiver de ser produzida, a respectiva pilha é *aberta* e assim permanece até que a última peça do mesmo tipo seja produzida, quando então a pilha é *fechada*, podendo ser removida do local de produção. É necessário determinar uma sequência $\pi$ de processamento dos elementos de $P$ tal que o número máximo de pilhas simultaneamente abertas é minimizado.

A Tabela 1 apresenta um exemplo de instância MOSP e duas possíveis soluções. Na Tabela 1($a$), representando uma possível instância para o MOSP, o conjunto de peças, enumeradas de 1 a 6, está representado na horizontal, e o conjunto dos padrões de corte, enumerados de $p_1$ a $p_6$, está representado na vertical. O valor 1, na linha $i$ e coluna $j$ indica que o padrão $p_j$ contém a peça $i$ – o valor 0 indica o contrário. Por exemplo, o padrão $p_1$ é composto pelas peças 1, 2 e 4, e o padrão $p_2$ é composto pelas peças 2, 4, 5 e 6. Nas Tabelas 1($b$) e 1($c$), uma pilha é associada a cada peça, e também é representada na horizontal. Na vertical, são representados os estágios de produção, nos quais um lote de determinado padrão de corte é processado. As pilhas abertas são contabilizadas a cada estágio, na vertical. Caso uma peça cuja pilha correspondente estiver aberta não for produzida em um determinado estágio, esta pilha será contabilizada como aberta se ainda houver peças desse tipo a serem processadas.

No primeiro sequenciamento, Tabela 1($b$), temos $\pi = [p_2, p_1, p_3, p_6, p_4, p_5]$, o que indica que no primeiro estágio serão processadas as cópias do padrão $p_2$, no segundo estágio serão processadas as cópias do padrão $p_1$ e assim sucessivamente até que no último estágio sejam processadas as cópias do padrão $p_5$. Neste sequenciamento, o número máximo de pilhas abertas é 6: no terceiro estágio as pilhas referentes à todas as peças estarão abertas simultaneamente. O segundo sequenciamento, na Tabela 1($c$), apresenta uma solução com um sequenciamento de padrões diferente, $\pi = [p_3, p_4, p_5, p_1, p_2, p_6]$, e com número máximo de pilhas abertas igual a 4.

O MOSP é um problema de grande importância prática na indústria, sendo relevante nas

| | $p_1$ | $p_2$ | $p_3$ | $p_4$ | $p_5$ | $p_6$ |
|---|---|---|---|---|---|---|
| 1 | 1 | 0 | 0 | 1 | 1 | 0 |
| 2 | 1 | 1 | 1 | 0 | 0 | 0 |
| 3 | 0 | 0 | 1 | 1 | 0 | 0 |
| 4 | 1 | 1 | 1 | 0 | 1 | 0 |
| 5 | 0 | 1 | 0 | 0 | 1 | 1 |
| 6 | 0 | 1 | 0 | 0 | 0 | 1 |

$(a)$

| | $p_2$ | $p_1$ | $p_3$ | $p_6$ | $p_4$ | $p_5$ |
|---|---|---|---|---|---|---|
| 1 | 0 | 1 | 0 | 0 | 1 | 1 |
| 2 | 1 | 1 | 1 | 0 | 0 | 0 |
| 3 | 0 | 0 | 1 | 0 | 1 | 0 |
| 4 | 1 | 1 | 1 | 0 | 0 | 1 |
| 5 | 1 | 0 | 0 | 1 | 0 | 1 |
| 6 | 1 | 0 | 0 | 1 | 0 | 0 |

$(b)$

| | $p_3$ | $p_4$ | $p_5$ | $p_1$ | $p_2$ | $p_6$ |
|---|---|---|---|---|---|---|
| 1 | 0 | 1 | 1 | 1 | 0 | 0 |
| 2 | 1 | 0 | 0 | 1 | 1 | 0 |
| 3 | 1 | 1 | 0 | 0 | 0 | 0 |
| 4 | 1 | 0 | 1 | 1 | 1 | 0 |
| 5 | 0 | 0 | 1 | 0 | 1 | 1 |
| 6 | 0 | 0 | 0 | 0 | 1 | 1 |

$(c)$

Tabela 1: Exemplo de instância e duas possíveis sequências para processamento de padrões.

indústrias que realizam o processamento de matérias-primas tais como metal, madeira, vidro e papel para a fabricação de bens de consumo. Trata-se de um problema NP-Difícil [Linhares e Yanasse, 2002], que possui formulação equivalente a diferentes problemas, conforme mostra o trabalho de [Möhring, 1990]: Problema de Corte Modificado (*Modified Cutwidth*), Leiaute de Matrizes de Portas (*Gate Matriz Layout*), Dobradura de Arranjos Lógicos Programáveis (*Programmable Logic Array Folding*, ou *PLA Folding*), *Interval Thickness, Node Search Game, Edge Search Game, Narrowness, Split Bandwidth, Graph Pathwidth, Edge Separation e Vertex Separation.*

Diferentes heurísticas construtivas gulosas foram propostas em [Yuen, 1991] e posteriormente aprimoradas em [Yuen, 1995]. Uma outra heurística gulosa também foi proposta por [Faggioli e Bentivoglio, 1998], porém, com uma fase de refinamento adicional, realizada pelo método Busca Tabu. Posteriormente, [Fink e Voß, 1999] compararam a aplicação de Busca Tabu em diferentes versões, *Simulated Annealing* e diversas heurísticas construtivas aplicadas ao MOSP e problemas correlatos. A *Heurística de Nó de Custo Mínimo*, baseada em busca em grafos e proposta por [Becceneri et al., 2004] é considerada a de melhor desempenho atualmente. Outra heurística baseada em busca em grafos, mais rápida e mais robusta que a anterior – porém, menos precisa – é apresentada em [Carvalho e Soma, 2014].

Diferentes formulações *Branch and Bound* foram apresentadas para o MOSP [Yuen e Richardson, 1995; Yanasse, 1997b; Faggioli e Bentivoglio, 1998; Yanasse e Limeira, 2004], porém, sem aplicação prática devido a restrições de tempo de execução. Outras importantes contribuições incluem a identificação de casos especiais solucionáveis em tempo determinístico polinomial [Yanasse, 1996], a modelagem utilizando grafos [Yanasse, 1997a] e diferentes operações de pré-processamento [Yanasse e Senne, 2010].

Devido a sua complexidade e equivalência com vários outros problemas, além da aplicação em manufatura, em 2005 o MOSP foi escolhido como tema do *First Constraint Modelling Challenge* [Smith e Gent, 2005]. O modelo vencedor do desafio foi o de programação dinâmica proposto por [de la Banda e Stuckey, 2007]. Posteriormente, [Chu e Stuckey, 2009] aprimoraram este modelo pela inclusão de uma nova estratégia de busca e diferentes procedimentos de poda de soluções parciais. Atualmente, este método representa o estado da arte relacionado a métodos exatos aplicados ao MOSP.

Mais recentemente [Gonçalves et al., 2016] propuseram a aplicação de Algoritmos Genéticos de Chaves Aleatórias Viciadas ao MOSP. Os experimentos computacionais abrangentes reportados indicam que este método foi capaz de obter as soluções ótimas para uma grande variedade de instâncias da literatura, tornando-o o estado da arte relacionado a metaheurísticas aplicadas ao MOSP.

O restante do trabalho está organizado da seguinte maneira: A Seção 2 detalha as implementações dos métodos Descida em Vizinhança Variável e Descida Mais Rápida. Ambos métodos são comparados entre si e com os melhores resultados da literatura utilizando cinco diferentes conjuntos de instâncias da literatura na Seção 3. Por fim, conclusões são realizadas a respeito do

trabalho e futuras direções para o trabalho de pesquisa são apontadas na Seção 4.

## 2. Métodos Propostos

A proposta deste trabalho é a aplicação de implementações do método Descida em Vizinhança Variável e do método Descida Mais Rápida para resolução do MOSP. Salvo melhor juízo, esta é a primeira aplicação reportada de Descida em Vizinhança Variável ao MOSP. Ambas as implementações propostas são detalhadas nas seções a seguir.

### 2.1. Descida em Vizinhança Variável

[Hansen e Mladenović, 2001] propõem dois métodos metaheurísticos simples (e outras variações), baseadas em busca local em vizinhanças variáveis. A primeira delas é a *Descida em Vizinhança Variável* (ou *Variable Neighborhood Descent* – VND), um método que consiste em definir um conjunto $N_k$ de $k_{max}$ diferentes estruturas de vizinhança (definidas por movimentos específicos) e determinar o ótimo local em cada uma destas vizinhanças, iteradamente. A cada iteração, uma vizinhança $k$ específica é explorada pela aplicação sucessiva do movimento associado à solução $\pi$ e, a cada ótimo local $\pi$' encontrado, a solução atual $\pi$ é atualizada. Quando não há melhoria possível, a próxima vizinhança passa a ser explorada e o método se repete até que todas as $k_{max}$ vizinhanças sejam analisadas. O Algoritmo 1 apresenta um pseudocódigo genérico para o VND.

**Algoritmo 1:** Pseudocódigo do método de Descida em Vizinhança Variável.

```
1  Inicialização: Selecione o conjunto de estruturas de vizinhança N_k = 1, ..., k_max.
2  Determine uma solução inicial π.
3  k ← 1;
4  repita
5      Encontre um melhor vizinho π' de π (π' ∈ N_k(π));
6      se π' melhor que π então
7          π ← π';
8      fim
9      senão
10         k ← k + 1;
11     fim
12 até k = k_max;
```

A implementação do método VND proposta neste trabalho utiliza uma heurística baseada em busca em grafos para geração de uma solução inicial. A geração de uma solução inicial e as estruturas de vizinhança, bem como um método adicional de busca local são descritos nas seções a seguir.

#### 2.1.1. Solução Inicial

A solução inicial do problema é gerada por um método heurístico baseado no proposto por [Carvalho e Soma, 2014]. Utiliza-se também o procedimento de pré-processamento por dominância de padrões proposto por [Yanasse e Senne, 2010], no intuito de reduzir o tamanho dos problemas pela remoção de informações redundantes.

O problema é modelado por meio de grafos MOSP, conforme descrito por [Yanasse, 1997a]. Nestes grafos, os vértices representam as peças e há uma aresta entre dois vértices se e somente se as peças correspondentes a estes vértices forem cortadas juntas em pelo menos um mesmo padrão de corte. Desta forma, como as peças de um mesmo padrão de corte são adjacentes entre si, são formados cliques no grafo MOSP resultante.

Antes de determinar a sequência $\pi$ de padrões, determina-se antes uma sequência $\phi$ de peças, de maneira a refletir o relacionamento entre as mesmas e tentar identificar quais pilhas devem estar abertas simultaneamente. Para este fim, aplica-se a conhecida *Busca em Largura* (ou *Breadth-First Search* – BFS) no grafo MOSP. A BFS gera uma sequência de vértices, referentes à ordem

de exploração dos mesmos, utilizando-se como critério inicialmente escolher o vértice com menor grau dentre todos, e posteriormente, todos os demais em ordem crescente do grau. Caso o grafo seja desconexo, o processo se repete com o sequenciamento de um outro vértice com o menor grau dentre os demais vértices que ainda não tenham sido explorados.

O sequenciamento dos padrões é obtido por meio do método elaborado por [Becceneri, 1999]. Este método simula a abertura sucessiva das pilhas referentes às peças presentes em $\phi$ e verifica a composição dos padrões do problema, de modo que, caso a composição de um padrão seja o conjunto ou subconjunto das pilhas abertas em determinado instante, este padrão é imediatamente inserido ao final da solução $\pi$.

### 2.1.2. Estruturas de Vizinhança

A eficiência do VND reside na definição de estruturas de vizinhança que explorem diferentes regiões promissoras do espaço de soluções, em busca de soluções ótimas ou subótimas globais. Na implementação de VND proposta, o algoritmo *k-opt* [Lin e Kernighan, 1973] foi utilizado para gerar quatro diferentes estruturas de vizinhança. Trata-se de um método conhecido, principalmente em sua versão original 2-*opt*. Seu princípio consiste na realização de trocas de posição entre os elementos pertencentes a um problema, gerando soluções que diferem em exatamente *k* elementos da configuração inicial. Por exemplo, considere um problema em que são dispostos 4 elementos. A Figura 1 ilustra as transições decorrentes da aplicação da busca local *2-opt* entre todos os elementos dada uma configuração inicial $[1, 2, 3, 4]$. A cada instante, dois elementos trocam de posição – as setas indicam os elementos envolvidos na troca de posições. A aplicação do método produz uma estrutura de vizinhança de seis elementos, referentes as combinações dos elementos tomados 2 a 2.

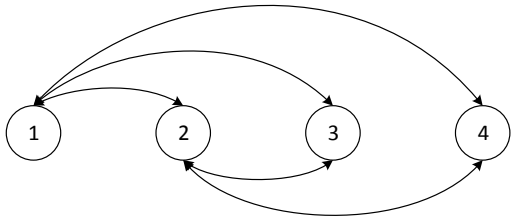


Figura 1: Exemplo de transições no método *k-opt*, no caso em que $k$=2.

Visando reduzir o espaço de busca e o tempo execução do VND, empregou-se uma estratégia de agrupamento de padrões consecutivos presentes em uma solução. Os grupos formados serão compostos por no máximo *n* padrões distintos. Por exemplo, considere uma solução inicial $\pi = [1, 3, 5, 4, 2]$. Para $n \leq 2$, os grupos seriam $[1, 3]$, $[5, 4]$ e $[2]$, ou seja, o *k-opt* passaria a ter 3 elementos e não 5, e os grupos seriam trocados entre si ao invés de cada um dos padrões. Em outras palavras, é realizado um *k-opt* sobre grupos de padrões, e não sobre padrões individuais. Após experimentos preliminares, optou-se por utilizar $n \leq 2$ e selecionar os pares para troca em ordem aleatória, embora todos os pares sejam testados.

Para gerar diferentes estruturas de vizinhança utilizando o algoritmo *k-opt*, variou-se o valor de $k$ entre 2 e 5, nesta ordem. À medida em que o valor de $k$ aumenta, maior é a exploração do espaço de busca. Este método foi escolhido para gerar as estruturas de vizinhança porque o método para geração da solução inicial fornece soluções de boa qualidade, das quais optou-se por não alterar fortemente a estrutura. Conforme reportado na Seção 3, bons resultados foram obtidos com este método.

### 2.1.3. Busca Local

Adicionalmente às estruturas de vizinhança induzidas pelo método *k-opt*, aplicou-se um método de busca local sobre as soluções vizinhas geradas, no intuito de aprimorá-las. O princípio deste método é identificar o gargalo do problema e reduzi-lo quanto ao número de pilhas abertas.

Em cada estrutura de vizinhança utilizada pelo VND, dada uma solução vizinha $\pi$', aplica-se o seguinte procedimento: a solução $\pi$' é avaliada quanto ao número de pilhas abertas e identifica-se o conjunto $G$ de pilhas relacionadas com o número máximo de pilhas abertas, ou seja, quais pilhas representam o *gargalo* do problema. Posteriormente, identifica-se quais padrões possuem em sua composição interseção com os elementos de $G$. Estes padrões são removidos da solução $\pi$' e inseridos novamente na melhor posição possível. A ordem de escolha dos padrões para reinserção é determinada pela similaridade da composição dos mesmos em relação aos elementos de $G$, de maneira decrescente.

Por exemplo, na Tabela 1($b$) o número máximo de pilhas abertas ocorre no terceiro estágio, quando o padrão $p_3$ é processado. O conjunto $G$ é formado pela peças 1, 2, 3, 4, 5 e 6, que representam o gargalo do problema, e, consequentemente, os padrões $p_2$, $p_1$, $p_5$, $p_4$ e $p_6$ seriam removidos e inseridos novamente na solução, nesta ordem.

### 2.2. Descida Mais Rápida

O segundo método proposto neste trabalho é do tipo *Descida Mais Rápida* (ou *Steepest Descent* – SD). Trata-se de um método de busca local baseado no conceito de melhora iterativa de uma solução inicial. Um método de busca local é aplicado a uma solução iterativamente, atualizando-a sempre que uma melhoria for atingida. Quando um ótimo local for atingido (ou seja, quando não houver melhoria possível), o método se encerra. Este método foi proposto como uma simplificação do VND descrito anteriormente, em que a vizinhança explorada é induzida somente pelo método 2-*opt*, em contraposição ao *k-opt* utilizado no VND proposto. Outra diferença se dá pela não utilização da estratégia de agrupamentos como proposta para o VND, tendo esta sido substituída por uma estratégia de *janela deslizante*.

Dada uma solução $\pi$', uma *janela* é uma subsequência de $n$ padrões consecutivos ($1 \leq n \leq |P|$). Inicialmente, esta subsequência consiste nos primeiros $n$ padrões em $\pi$' ($\pi$'[1..$n$]). Esta janela é então deslizada um estágio para a direita ($\pi$'[2..$n+1$]) e assim sucessivamente até que todos os padrões em $\pi$' tenham sido contemplados. Por exemplo, considerando a solução $\pi$'$= [1, 3, 5, 4, 2]$ e $n$=2, as janelas seriam [1, 3], [3, 5], [5, 4] e [4, 2]. O método 2-*opt* é aplicado sobre duas janelas diferentes, escolhidas aleatoriamente, como busca local. Após experimentos preliminares, optou-se por utilizar $n = 2$.

O Algoritmo 2 apresenta o pseudo-código para o método de descida rápida proposto, em que $N(\pi)$ denota a vizinhança induzida pelo método 2-*opt* sobre todas as janelas possíveis, em ordem aleatória.

**Algoritmo 2:** Pseudocódigo do método Descida Mais Rápida.

```
1 Inicialização: Determine uma solução inicial π.
2 melhoria ← falso;
3 repita
4 |   Encontre um vizinho π' de π (π' ∈ N(π));
5 |   se π' melhor que π então
6 |   |   π ← π';
7 |   |   melhoria ← verdadeiro;
8 |   fim
9 até melhoria = falso;
```

Embora esta estratégia de janela deslizante se assemelhe ao agrupamento de pares de padrões utilizado pelo VND, tratam-se de estratégias diferentes. A utilização de janelas deslizantes permite que um mesmo padrão figure em diferentes janelas, aumentando o espaço de busca. Ainda assim, isto permite que a estrutura da solução inicial seja modificada apenas pontualmente. Conforme descrito anteriormente, a busca em grafos utilizada produz boas soluções iniciais e optou-se

por não alterá-las significativamente.

## 3. Experimentos Computacionais

Os experimentos computacionais foram realizados em um computador com processador *Intel i5 Quad Core* de 3.2 GHz com 16 GB RAM sob o sistema operacional Ubuntu 12.4.1. Os códigos dos métodos propostos foram escritos em C++, compilados com g++ 4.4.1 e a opção de otimização -O3. Comparam-se a quantidade de pilhas abertas obtidas pelos métodos propostos e o tempo de execução médio. Dada a utilização de componentes de aleatoriedade pelos algoritmos, foram realizadas 20 execuções independentes para cada uma das instâncias, considerando-se os resultados médios. Onde mencionado, o *gap* (ou distância percentual) é calculado como *100×(valor da solução obtida - valor ótimo) / valor ótimo*).

Cinco conjuntos de instâncias foram utilizados, divididos em diferentes subseções. Nas tabelas seguintes, a coluna *Instância* refere-se ao nome das instâncias, $|P|$ indica a quantidade de padrões, $|C|$ representa a quantidade de peças e $S^*$ indica o valor da solução ótima. Os valores de solução obtidos são apresentados na coluna *VND* e *SD*, respectivamente para os métodos Descida em Vizinhança Variável e Descida Mais Rápida. Os tempos médios de execução (expressos em segundos) são apresentados na coluna $T$ e o valor das soluções obtidas são apresentados na coluna $S$. Nos casos em que as soluções ótimas foram atingidas, os valores estão destacados em negrito.

### 3.1. Instâncias Reais MOSP

O primeiro conjunto, fornecido pelo *SCOOP Consortium* (disponível em `http://www.scoop-project.net`), contém 187 instâncias MOSP reais de duas empresas europeias. Entretanto, a maioria destas instâncias são muito pequenas (por exemplo, 2 linhas e colunas), possuindo soluções triviais. Deste conjunto, 24 instâncias com dimensões significativas, variando de 10 linhas e colunas a 49 linhas e 134 colunas, foram selecionadas para serem usadas nos experimentos. A Tabela 2 apresenta os valores médios para tempo de execução e acumulados para solução, que foram agrupados em dois subconjuntos de 12 instâncias cada.

Tabela 2: Instâncias SCOOP disponíveis em: `http://www.scoop-project.net`. O valor em negrito indica soluções ótimas.

| Grupo | $S^*$ | *VND* | | *SD* | |
|---|---|---|---|---|---|
| | | $T$ | $S$ | $T$ | $S$ |
| *Wood A* | 116 | 0,25 | 118 | 0,15 | 119 |
| *Wood B* | 70 | 0,08 | 73 | 0,01 | **70** |

Ambos os métodos obtiveram boas soluções para este conjunto de instâncias em muito baixo tempo de execução, porém, o método de Descida Mais Rápida apresentou melhores indicadores. Os *gaps* foram 2,69% e 1,61% respectivamente para o VND e o método de Descida Mais Rápida. Além disto, o primeiro método foi capaz de obter 79,17% das soluções ótimas, ao passo que o segundo obteve 87,5%, ou seja, apenas para três instâncias do conjunto *Wood A* a melhor solução não foi atingida pelo método de Descida Mais Rápida. O VND não foi capaz de obter as soluções ótimas para cinco instâncias, ambos conjuntos. Nos casos em que a solução ótima não foi obtida, uma pilha a mais foi aberta por instância pelos dois métodos.

### 3.2. Instâncias Reais VLSI

O segundo conjunto contém 25 instâncias reais para o problema de Leiaute de Matrizes de Portas (um problema de formulação equivalente ao MOSP, conforme mencionado na Seção 1), oriundas de empresas asiáticas, introduzidas por [Hu e Chen, 1990]. Os resultados são apresentados na Tabela 3.

Neste segundo conjunto de instâncias, o VND apresentou o melhor desempenho, obtendo 96% das soluções ótimas (apenas para uma instância o ótimo não foi obtido) e um *gap* de apenas

Tabela 3: Instâncias reais VLSI. Valores em negrito indicam soluções ótimas.

| Instância | $\|P\|$ | $\|C\|$ | $S^*$ | *VND* | | *SD* | |
|---|---|---|---|---|---|---|---|
| | | | | $T$ | $S$ | $T$ | $S$ |
| *v1* | 8 | 6 | 3 | 0,00 | 4 | 0,00 | 4 |
| *v4000* | 17 | 10 | 5 | 0,00 | **5** | 0,00 | **5** |
| *v4050* | 16 | 13 | 5 | 0,00 | **5** | 0,00 | **5** |
| *v4090* | 27 | 23 | 10 | 0,00 | **10** | 0,00 | **10** |
| *v4470* | 47 | 37 | 9 | 1,32 | **9** | 0,00 | 10 |
| *vc1* | 25 | 15 | 9 | 0,00 | **9** | 0,00 | **9** |
| *vw1* | 7 | 5 | 4 | 0,00 | **4** | 0,00 | **4** |
| *vw2* | 8 | 8 | 5 | 0,00 | **5** | 0,00 | **5** |
| *w1* | 21 | 18 | 4 | 0,00 | **4** | 0,00 | **4** |
| *w2* | 33 | 48 | 14 | 0,00 | **14** | 0,02 | **14** |
| *w3* | 70 | 84 | 18 | 20,87 | **18** | 2,40 | **18** |
| *w4* | 141 | 202 | 27 | 4070,19 | **27** | 35,5 | 28 |
| *wan* | 7 | 9 | 6 | 0,00 | **6** | 0,00 | **6** |
| *wli* | 10 | 11 | 4 | 0,00 | **4** | 0,00 | **4** |
| *wsn* | 25 | 17 | 8 | 0,00 | **8** | 0,00 | **8** |
| *x0* | 48 | 40 | 11 | 1,11 | **11** | 0,35 | **11** |
| *x1* | 10 | 5 | 5 | 0,00 | **5** | 0,00 | **5** |
| *x2* | 15 | 6 | 6 | 0,00 | **6** | 0,00 | **6** |
| *x3* | 21 | 7 | 7 | 0,00 | **7** | 0,01 | **7** |
| *x4* | 8 | 12 | 2 | 0,00 | **2** | 0,00 | **2** |
| *x5* | 16 | 24 | 2 | 0,00 | **2** | 0,00 | **2** |
| *x6* | 32 | 49 | 2 | 0,00 | **2** | 0,00 | **2** |
| *x7* | 8 | 7 | 4 | 0,00 | **4** | 0,00 | **4** |
| *x8* | 16 | 11 | 4 | 0,00 | **4** | 0,00 | **4** |
| *x9* | 32 | 19 | 4 | 0,06 | **4** | 0,03 | **4** |

0,56%. O método de Descida Mais Rápida obteve 88% das soluções ótimas e *gap* de 1,69%. Nota-se em ambos os métodos um crescimento no tempo de execução, especificamente em instâncias de maiores dimensões, como $w3$ e $w4$. A proposta de implementação do método de Descida Mais Rápida se deu justamente devido ao rápido aumento do tempo de execução do VND, que ultrapassa uma hora para a instância $w4$.

### 3.3. Instâncias MOSP [Faggioli e Bentivoglio, 1998]

O terceiro conjunto, proposto por [Faggioli e Bentivoglio, 1998] contém 300 instâncias MOSP artificiais. Os problemas foram agrupados em subconjuntos de 10 instâncias de acordo com o número de padrões e peças. A Tabela 4 apresenta os resultados.

Neste conjunto de instâncias com dimensões médias, novamente o método de Descida Mais Rápida apresenta leve superioridade sobre o VND, obtendo 88,67% das soluções ótimas e *gap* de 1,25%. O tempo de execução se manteve abaixo de 0,4 segundo. O VND obteve 82,00% das soluções ótimas e *gap* de 1,97%, e o tempo de execução se manteve abaixo de 1,5 segundo. Como esperado, os tempos de execução do VND crescem mais rapidamente do que os do método de Descida Mais Rápida.

Tabela 4: Instâncias [Faggioli e Bentivoglio, 1998]. Valores em negrito indicam soluções ótimas.

| $\|P\|$ | $\|C\|$ | $S^*$ | *VND* | | *SD* | |
|---|---|---|---|---|---|---|
| | | | $T$ | $S$ | $T$ | $S$ |
| 10 | 10 | 55 | 0,00 | 58 | 0,00 | **55** |
| 10 | 20 | 62 | 0,00 | 65 | 0,00 | **62** |
| 10 | 30 | 61 | 0,00 | 66 | 0,00 | 63 |
| 10 | 40 | 77 | 0,00 | 78 | 0,00 | 78 |
| 10 | 50 | 82 | 0,00 | **82** | 0,00 | **82** |
| 15 | 10 | 66 | 0,00 | **66** | 0,00 | **66** |
| 15 | 20 | 72 | 0,00 | 74 | 0,00 | 74 |
| 15 | 30 | 73 | 0,00 | 78 | 0,00 | 76 |
| 15 | 40 | 72 | 0,00 | 73 | 0,01 | **72** |
| 15 | 50 | 74 | 0,00 | **74** | 0,00 | 75 |
| 20 | 10 | 75 | 0,00 | **75** | 0,00 | **75** |
| 20 | 20 | 85 | 0,00 | **85** | 0,01 | **85** |
| 20 | 30 | 88 | 0,01 | 90 | 0,02 | 90 |
| 20 | 40 | 85 | 0,01 | 88 | 0,02 | 86 |
| 20 | 50 | 79 | 0,01 | 82 | 0,02 | 81 |
| 25 | 10 | 80 | 0,00 | **80** | 0,00 | **80** |
| 25 | 20 | 98 | 0,02 | **98** | 0,02 | **98** |
| 25 | 30 | 105 | 0,04 | 108 | 0,04 | 106 |
| 25 | 40 | 103 | 0,04 | 106 | 0,05 | 104 |
| 25 | 50 | 100 | 0,05 | 102 | 0,05 | 102 |
| 30 | 10 | 78 | 0,00 | **78** | 0,00 | **78** |
| 30 | 20 | 111 | 0,05 | 112 | 0,05 | 112 |
| 30 | 30 | 122 | 0,14 | **122** | 0,08 | 123 |
| 30 | 40 | 121 | 0,19 | **121** | 0,10 | **121** |
| 30 | 50 | 112 | 0,19 | 117 | 0,11 | 114 |
| 40 | 10 | 84 | 0,00 | **84** | 0,01 | **84** |
| 40 | 20 | 130 | 0,31 | **130** | 0,12 | 132 |
| 40 | 30 | 145 | 0,88 | 147 | 0,24 | 147 |
| 40 | 40 | 149 | 1,23 | 155 | 0,32 | 151 |
| 40 | 50 | 146 | 1,40 | 151 | 0,35 | 153 |

### 3.4. Instâncias do *First Constraint Modeling Challenge* [Smith e Gent, 2005]

O quarto conjunto contém 46 instâncias MOSP propostas para o *First Constraint Modeling Challenge* [Smith e Gent, 2005]. Este subconjunto foi selecionado levando em conta as dimensões das instâncias e a ausência de estruturas especiais que podem facilitar a solução, tal como analisado por [Yanasse e Senne, 2010]. Os resultados são apresentados na Tabela 5, que possui uma coluna adicional $n$ que identifica a quantidade de instâncias em cada grupo. Os resultados apresentados se referem aos valores acumulados para todas as instâncias do grupo.

Os métodos VND e Descida Mais Rápida apresentam valores semelhantes para *gap* (0,90% e 0,70%, respectivamente) e percentual de soluções ótimas obtidas (86,96% e 84,78%, respectivamente). O *gap* baixo indica a robustez dos métodos em relação a este conjunto, e, analisado em conjunto com a taxa de resultados ótimos obtidos, indica que o erro dos métodos em relação às soluções ótimas foi baixo. O tempo de execução do método de Descida Mais Rápida manteve-se abaixo de 75 segundos, aumentando consideravelmente para instâncias com 100 padrões e peças. Para estas mesmas instâncias, o tempo de execução do VND ultrapassou uma hora.

Tabela 5: Resultados para as instâncias selecionadas do *First Constraint Modeling Challenge* [Smith e Gent, 2005]. Valores em negrito indicam soluções ótimas.

| Instância | $\|P\|$ | $\|C\|$ | $n$ | $S^*$ | *VND* | | *SD* | |
|---|---|---|---|---|---|---|---|---|
| | | | | | $T$ | $S$ | $T$ | $S$ |
| *nwrsSmaller-a* | 20 | 10 | 2 | 7 | 0,00 | **7** | 0,00 | **7** |
| *nwrsSmaller-b* | 25 | 15 | 2 | 14 | 0,00 | **14** | 0,00 | **14** |
| *nrwsLarger-a* | 30 | 20 | 2 | 24 | 0,01 | **24** | 0,03 | **24** |
| *nrwsLarger-b* | 60 | 25 | 2 | 26 | 0,57 | **26** | 0,19 | **26** |
| *GP1* | 50 | 50 | 4 | 155 | 6,79 | 157 | 1,46 | 156 |
| *GP2* | 100 | 100 | 4 | 305 | 4444,24 | 311 | 75,20 | 308 |
| *Shaw* | 20 | 20 | 25 | 342 | 0,01 | **342** | 0,01 | **342** |
| *Miller* | 40 | 20 | 1 | 13 | 0,67 | **13** | 0,19 | **13** |
| *SP4-1* | 25 | 25 | 1 | 9 | 0,00 | **9** | 0,01 | **9** |
| *SP4-2* | 50 | 50 | 1 | 19 | 2,54 | 20 | 0,52 | 20 |
| *SP4-3* | 75 | 75 | 1 | 34 | 177,93 | **34** | 5,89 | 35 |
| *SP4-4* | 100 | 100 | 1 | 53 | 1671,34 | **53** | 26,31 | 54 |

### 3.5. Instâncias MOSP [Chu e Stuckey, 2009]

O quinto conjunto consiste em 200 instâncias MOSP de maiores dimensões, geradas aleatoriamente por [Chu e Stuckey, 2009]. A Tabela 6 apresenta os resultados. Novamente, os problemas foram agrupados, em subconjuntos de 25 instâncias de acordo com o número de padrões e peças. Devido às dimensões destas instâncias, algumas das maiores consideradas nestes experimentos, optou-se por executar o VND com parâmetro *K_MAX=4*.

Tabela 6: Instâncias *Random* [Chu e Stuckey, 2009]. O valor em negrito indica solução ótima.

| $\|P\|$ | $\|C\|$ | $S^*$ | *VND* | | *SD* | |
|---|---|---|---|---|---|---|
| | | | $T$ | $S$ | $T$ | $S$ |
| 100 | 100 | 1455 | 339,66 | 1480 | 26,60 | 1484 |
| 50 | 100 | 1027 | 5,69 | 1044 | 2,03 | 1036 |
| 125 | 125 | 1755 | 1563,22 | 1783 | 76,78 | 1799 |
| 30 | 30 | 490 | 0,07 | 491 | 0,08 | 492 |
| 40 | 40 | 637 | 0,60 | 639 | 0,35 | 641 |
| 100 | 50 | 961 | 82,99 | 966 | 11,06 | 976 |
| 50 | 50 | 785 | 2,82 | 792 | 1,08 | 794 |
| 75 | 75 | 1121 | 50,42 | 1135 | 6,89 | 1138 |

Neste último conjunto de instâncias, os métodos VND e Descida Mais Rápida apresentam os piores valores para o percentual de soluções ótimas obtidas: 58,50% e 53,50% respectivamente. Entretanto, os valores de *gap* continuam baixos: 1,20% e 1,57% respectivamente. Estes dados demonstram que ambos os métodos continuam com soluções próximas do ótimo, embora as soluções ótimas sejam obtidas somente em aproximadamente metade dos casos. Novamente, este é um indício que os métodos mantém o erro em relação às soluções ótimas sob controle, demonstrando robustez. Condeirando apenas as maiores instâncas (dimensões $100 \times 100$ e $125 \times 125$) tempo de execução do VND esteve entre 5,6 minutos e aproximadamente 26 minutos, ao passo que o tempo de execução do método de Descida Rápida esteve entre 26 segundos e aproximadamente 1,2 minuto.

## 4. Conclusões e Trabalhos Futuros

O Problema de Minimização de Pilhas Abertas (ou MOSP) é um problema NP-Difícil de sequenciamento de padrões de corte com ampla aplicação industrial que visa otimizar a utilização de estoque intermediário e manipulação de produtos fabricados dentro do ambiente de fabricação.

Este trabalho propõe dois métodos para solução do MOSP. O primeiro consiste na metaheurística Descida em Vizinhança Variável, cujas estruturas de vizinhança foram induzidas pelo método *k-opt* em grupos de padrões; o segundo consiste em um método de descida rápida, cuja busca local é realizada pelo método *2-opt* em conjunto com uma estratégia de janela deslizante. Ambos os métodos foram executados sobre uma solução inicial gerada por uma adaptação de uma heurística da literatura. Salvo melhor juízo, esta é a primeira aplicação reportada de VND para solução do MOSP.

Os experimentos computacionais envolveram aproximadamente 600 instâncias de cinco diferentes conjuntos da literatura, incluindo instâncias reais e artificiais. Os resultados obtidos foram comparados com as soluções ótimas. De acordo com os dados reportados, não houve método dominante, havendo alternância de melhor desempenho nos três primeiros conjuntos de instâncias e não havendo diferença significativa nos dois últimos. Entretanto, há de se destacar $(i)$ o baixo tempo de execução do método de descida rápida – máximo de 1,2 segundo – em contraposição ao tempo de execução do VND, que ultrapassa 65 minutos para algumas instâncias; $(ii)$ as consideráveis taxas de soluções ótimas obtidas nos quatro primeiros conjuntos de instâncias – mínimo de 79,17% e 84,78% respectivamente para o VND e o método de descida rápida; $(iii)$ os baixos valores de *gap* para ambos os métodos em todos os conjuntos de instâncias– máximo de 2,69% para o VND e 1,61% para o método de descida rápida – o que indica a robustez dos métodos ao obterem soluções próximas aos valores ótimos.

Os trabalhos futuros serão concentrados em aprimorar as estruturas de vizinhança do VND, experimentando diferentes métodos e também na implementação do métodos Busca em Vizinhança Variável, considerado um aprimoramento do VND. Adicionalmente, o método descida rápida pode ser inserido em métodos *branch and bound* da literatura, pois fornece um limite superior razoável para o problema em pouco tempo, além de poder ser embutido em uma heurística híbrida, ou *matheuristic*.

## 5. 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.

## Referências

Becceneri, J. C., Yanasse, H. H., e Soma, N. Y. (2004). A method for solving the minimization of the maximum number of open stacks problem within a cutting process. *Comput. Oper. Res.*, 31 (14):2315–2332. ISSN 0305-0548.

Becceneri, J. C. (1999). O problema de sequenciamento de padrões para a minimização do número máximo de pilhas abertas em ambientes de cortes industriais. *European Journal of Operational Research*, p. 145.

Carvalho, M. A. M. d. e Soma, N. Y. (2014). A breadth-first search applied to the minimization of open stacks. *Journal of the Operational Research Society*. (to appear).

Chu, G. e Stuckey, P. J. (2009). Minimizing the maximum number of open stacks by customer search. In *Proceedings...*, volume 5732 of *Lecture Notes in Computer Science*, p. 242–257, Berlin. INTERNATIONAL CONFERENCE ON PRINCIPLES AND PRACTICE OF CONSTRAINT PROGRAMMING, Springer.

de la Banda, M. G. e Stuckey, P. J. (2007). Dynamic programming to minimize the maximum number of open stacks. *INFORMS Journal on Computing*, 19(4):607–617.

Faggioli, E. e Bentivoglio, C. A. (1998). Heuristic and exact methods for the cutting sequencing problem. *European Journal of Operational Research*, 110(3):564–575.

Fink, A. e Voß, S. (1999). Applications of modern heuristic search methods to pattern sequencing problems. *Computers & Operations Research*, 26(1):17–34.

Gonçalves, J. F., Resende, M. G. C., e Costa, M. D. (2016). A biased random-key genetic algorithm for the minimization of open stacks problem. *International Transactions in Operational Research*, 23(1-2):25–46. ISSN 1475-3995. URL http://dx.doi.org/10.1111/itor.12109.

Hansen, P. e Mladenović, N. (2001). Variable neighborhood search: Principles and applications. *European Journal of Operational Research*, 130(3):449 – 467. ISSN 0377-2217.

Hu, Y. H. e Chen, S.-J. (1990). Gm plan: a gate matrix layout algorithm based on artificial intelligence planning techniques. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, 9(8):836–845. ISSN 0278-0070.

Lin, S. e Kernighan, B. W. (1973). An effective heuristic algorithm for the traveling-salesman problem. *Operations research*, 21(2):498–516.

Linhares, A. e Yanasse, H. H. (2002). Connections between cutting-pattern sequencing, vlsi design, and flexible machines. *Comput. Oper. Res.*, 29(12):1759–1772. ISSN 0305-0548.

Möhring, R. H. (1990). *Graph Problems Related to Gate Matrix Layout and PLA Folding*. Springer Vienna.

Smith, B. e Gent, I. (2005). Constraint modeling challenge. In Smith, B. e Gent, I., editors, *The fifth workshop on modelling and solving problems with constraints*, Edinburgh, Scotland. IJCAI. URL http://www.dcs.st-and.ac.uk/~ipg/challenge/ModelChallenge05.pdf.

Yanasse, H. (1996). Minimization of open orders-polynomial algorithms for some special cases. *Pesquisa Operacional*, 16(1):1–26.

Yanasse, H. e Limeira, M. (2004). Refinements on an enumeration scheme for solving a pattern sequencing problem. *International Transactions in Operational Research*, 11(3):277–292.

Yanasse, H. H. (1997a). On a pattern sequencing problem to minimize the maximum number of open stacks. *European Journal of Operational Research*, 100(3):454–463.

Yanasse, H. H. (1997b). A transformation for solving a pattern sequencing in the wood cut industry. *Pesquisa Operacional*, 17(1):57–70.

Yanasse, H. H. e Senne, E. L. F. (2010). The minimization of open stacks problem: A review of some properties and their use in pre-processing operations. *European Journal of Operational Research*, 203(3):559–567. URL http://ideas.repec.org/a/eee/ejores/v203y2010i3p559-567.html.

Yuen, B. J. (1991). Heuristics for sequencing cutting patterns. *European Journal of Operational Research*, 55(2):183–190.

Yuen, B. J. (1995). Improved heuristics for sequencing cutting patterns. *European Journal of Operational Research*, 87(1):57 – 64. ISSN 0377-2217.

Yuen, B. J. e Richardson, K. V. (1995). Establishing the optimality of sequencing heuristics for cutting stock problems. *European Journal of Operational Research*, 84(3):590–598.