Algoritmos Metaheurísticos Aplicados ao Problema de Minimização de Pilhas Abertas

Júnior Rhis Lima

Universidade Federal de Ouro Preto

juniorrhis1@gmail.com

1 de agosto de 2016

Problemas de Corte

- Consistem no corte de unidades maiores de matéria-prima para produzir unidades menores (ou peças);

- Relevantes nas indústrias que realizam processamento de
  - metal;
  - madeira;
  - vidro;
  - papel.

Problema de Minimização de Pilhas Abertas

Características do Problema

- A disposição de peças dentro de unidades maiores para realização do corte define um padrão de corte;
- Todas as peças de um mesmo padrão de corte devem ser cortadas antes que um padrão de corte diferente seja processado;
- A cada estágio um determinado padrão de corte diferente é processado;
- Durante a produção de uma peça, todas as suas cópias são armazenadas temporariamente em uma pilha mantida ao redor da máquina que as produziu;
- Quando a primeira peça de um dado tipo for produzida ela abre uma pilha que permanece aberta até que a última peça do mesmo tipo seja produzida.

Problema de Minimização de Pilhas Abertas

Descrição do Problema

O **Problema de Minimização de Pilhas Abertas** (ou MOSP, de *Minimization of Open Stacks Problem*), remonta à um ambiente de produção em que peças com demandas específicas são produzidas por uma única máquina de corte e tem como objetivo atingir a melhor utilização do espaço físico disponível agilizando a linha de produção.

Motivação

- Cada vez mais as indústrias confiam questões operacionais a métodos computacionais;
- O MOSP é um problema NP-Difícil.

Objetivo

Elaborar métodos metaheurísticos consistentes e robustos que possam ser utilizadas no contexto do problema abordado e permitam a obtenção rápida de soluções próximas da solução ótima.

Problema de Minimização de Pilhas Abertas

Instância

	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

- Matriz binária;
- Padrões representados nas colunas;
- Peças representados nas linhas;
- $I_{ij} = 1$ se o padrão i contém a peça j.

Objetivo

Encontrar uma permutação de padrões de modo a minimizar a quantidade de pilhas abertas ao redor da máquina de corte.

Problema de Minimização de Pilhas Abertas

Dois possíveis sequenciamentos

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

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

Algumas propriedades

- Estágios de produção – Ordem de processamento dos padrões;
- 1s consecutivos – Descontinuidade na produção;
- Gargalo – Estágio (coluna) com maior quantidade de pilhas abertas;
- Quantidade de pilhas abertas é definida pelo gargalo.

Problema de Minimização de Pilhas Abertas

Definição Matemática

Seja a matriz Q^π representando os estágios de produção obtidos a partir da permutação π dos padrões, seus elementos q_{ij}^π são definidos como:

$$q_{ij}^\pi = \begin{cases} 1, & \text{se } \exists x, \exists y \mid \pi[x] \leq j \leq \pi[y] \text{ e } m_{ix} = m_{iy} = 1 \\ 0, & \text{caso contrário} \end{cases} \quad (1)$$

ou seja, $q_{ij}^\pi = 1$ entre o primeiro e o último 1s da linha.

A quantidade máxima de pilhas abertas simultaneamente será:

$$Z_{MOSP}^\pi(Q^\pi) = \max_{j \in P} \sum_{i=1}^{|C|} q_{ij}^\pi \quad (2)$$

ou seja, será dada pelo tamanho do gargalo.

Problema de Minimização de Pilhas Abertas

Função Objetivo

$$\min_{\pi \in \Pi} \; Z^{\pi}_{MOSP}(M) \tag{3}$$

Encontrar a permutação com a menor quantidade de pilhas abertas simultaneamente, ou seja, encontrar a permutação que possuir o menor gargalo.

Problema de Minimização de Pilhas Abertas

Etapas para Obtenção da Solução

O processo para geração de uma solução para o problema consiste nas seguintes etapas:

- Pré-processamento por dominância entre padrões;
- Representação computacional;
- Geração da lista ϕ de peças (utilizando uma heurística de busca em grafos);
- Geração da lista π de padrões (Solução Inicial);
- Aplicação dos métodos propostos neste trabalho para refinamento (melhoria) da solução.

Etapas para Obtenção da Solução

Pré-Processamento por Dominância Entre Padrões

Consiste na eliminação de redundâncias em uma instância MOSP.

Exemplo

	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**

	p_1	p_2, p_6	p_3	p_4	p_5
1	1	0	0	1	1
2	1	1	1	0	0
3	0	0	1	1	0
4	1	1	1	0	1
5	0	1	0	0	1
6	0	1	0	0	0

O problema é resolvido considerando apenas o padrão p_2. O padrão p_6 é sequenciado imediatamente após o padrão p_2 na solução final sem perda de otimalidade.

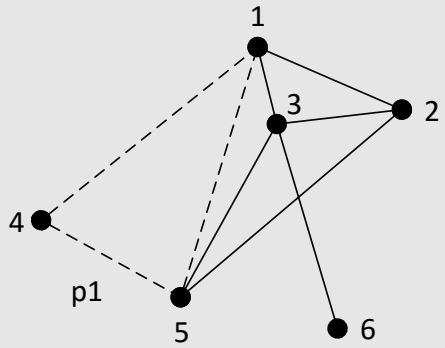
Etapas para Obtenção da Solução

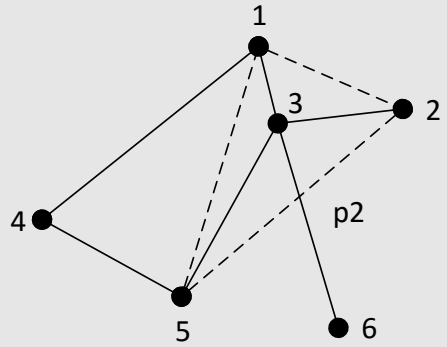
Representação Computacional

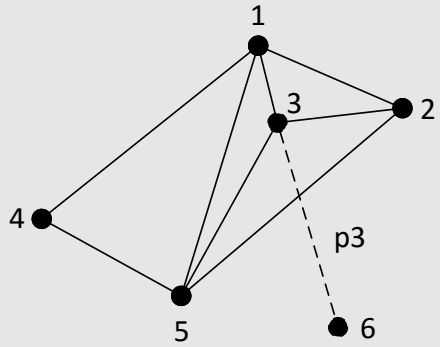
- Modela o MOSP por meio de grafos, denominados grafos MOSP;
- Os vértices representam as peças;
- Existe uma aresta entre dois vértices se e somente se as peças correspondentes a estes vértices forem cortadas juntas em pelo menos um mesmo padrão de corte;
- A ligação entre as peças presentes em um mesmo padrão de corte forma uma clique.

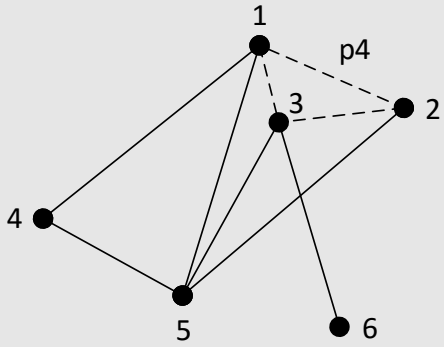
Formação de um grafo MOSP a partir da união de cliques

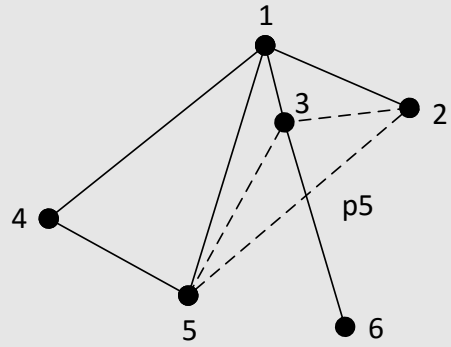
	p_1	p_2	p_3	p_4	p_5
1	1	1	0	1	0
2	0	1	0	1	1
3	0	0	1	1	1
4	1	0	0	0	0
5	1	1	0	0	1
6	0	0	1	0	0







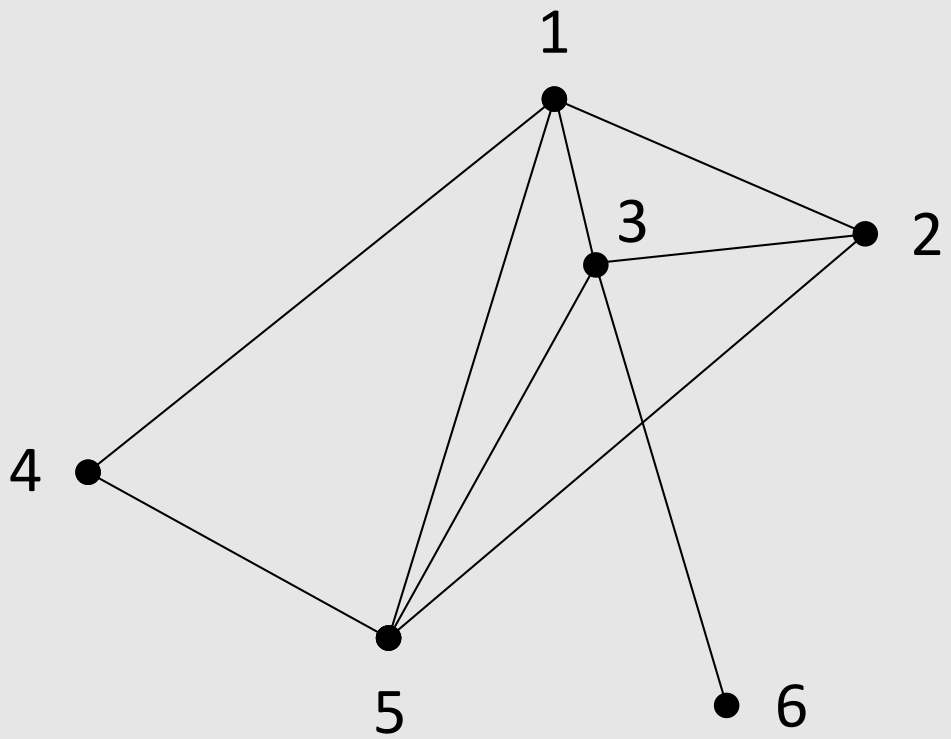




Etapas para Obtenção da Solução

Geração da lista de peças

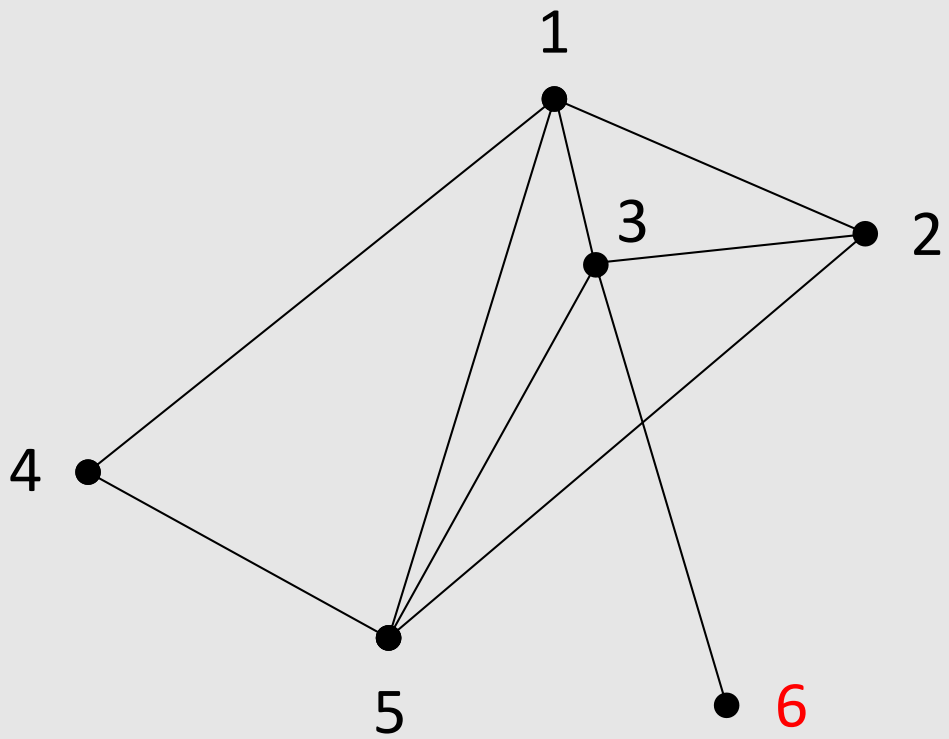
Execução de uma busca em largura no grafo MOSP a partir do vértice de menor grau.



$\phi = \emptyset$

Etapas para Obtenção da Solução

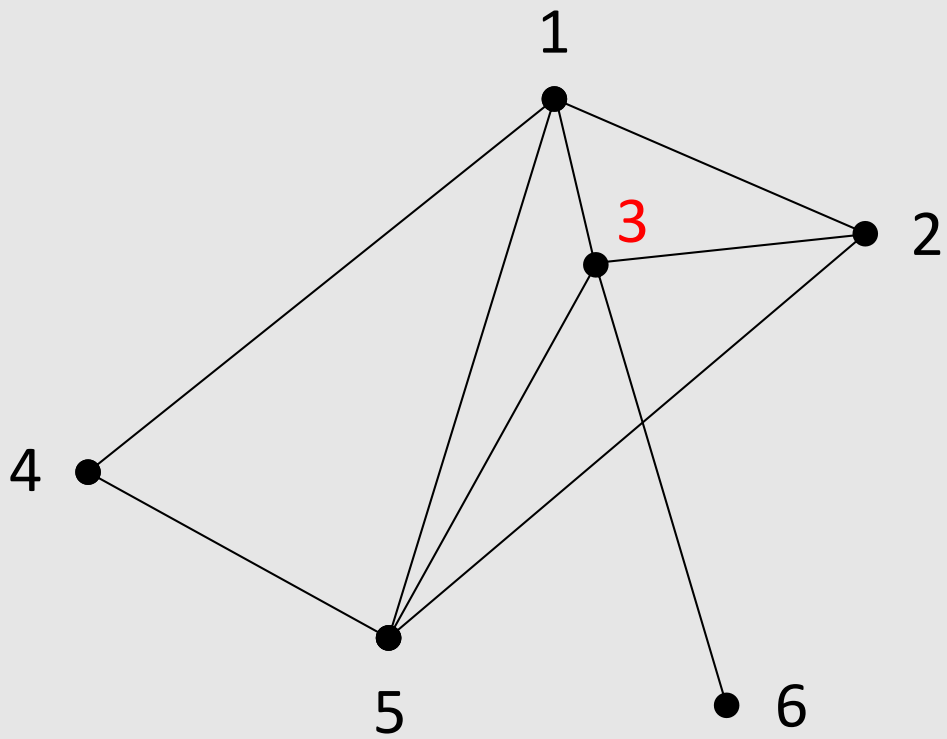
Geração da lista de peças



$\phi = [6]$

Etapas para Obtenção da Solução

Geração da lista de peças

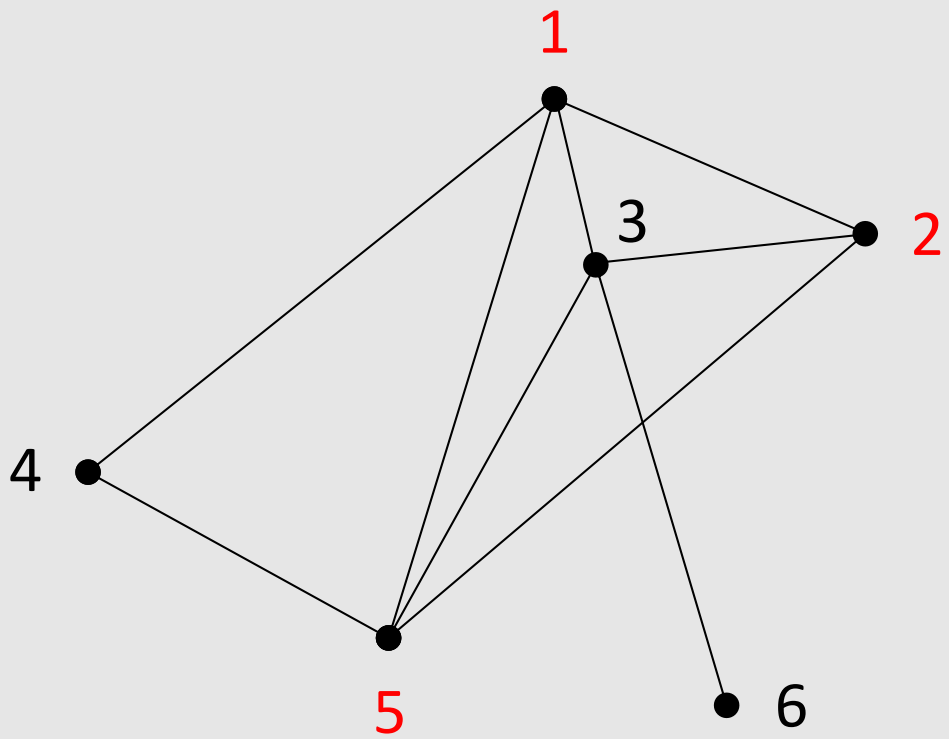


$\phi = [6, 3]$

Vértice 3 adicionado

Etapas para Obtenção da Solução

Geração da lista de peças

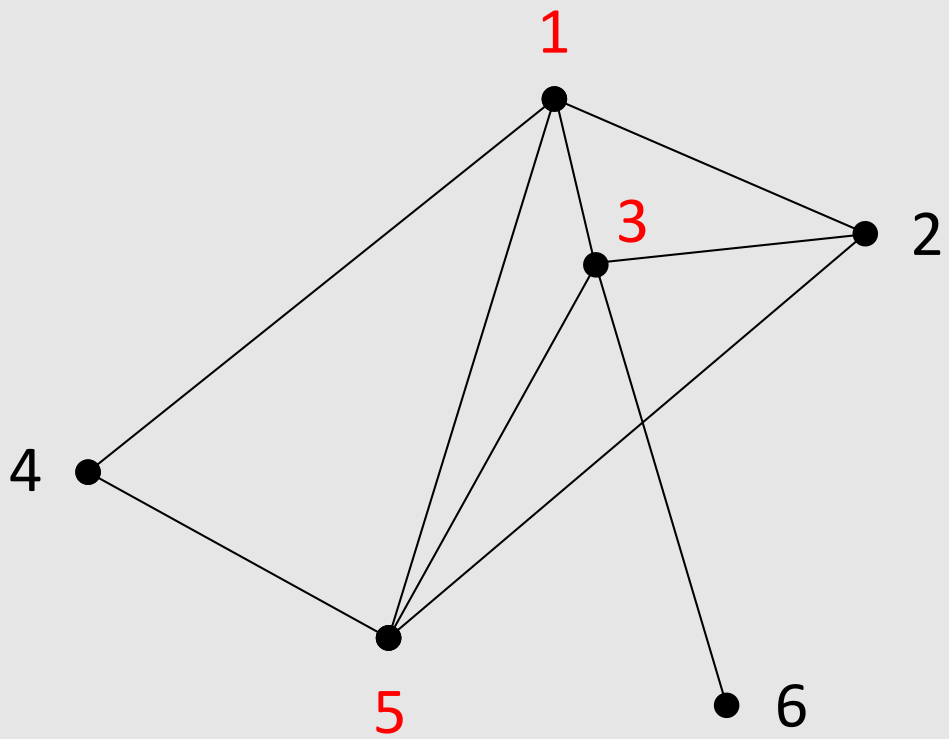


$\phi = [6, 3, 2, 1, 5]$

Vértices 2, 1 e 5 adicionados

Etapas para Obtenção da Solução

Geração da lista de peças

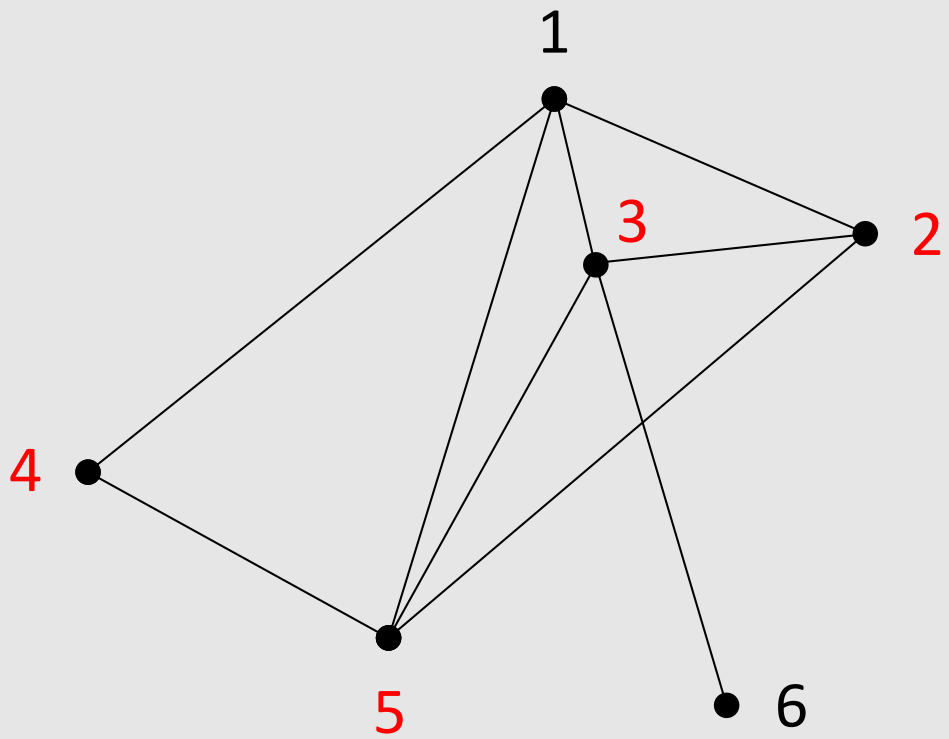


$\phi = [6, 3, 2, 1, 5]$

Nenhum vértice adicionado

Etapas para Obtenção da Solução

Geração da lista de peças

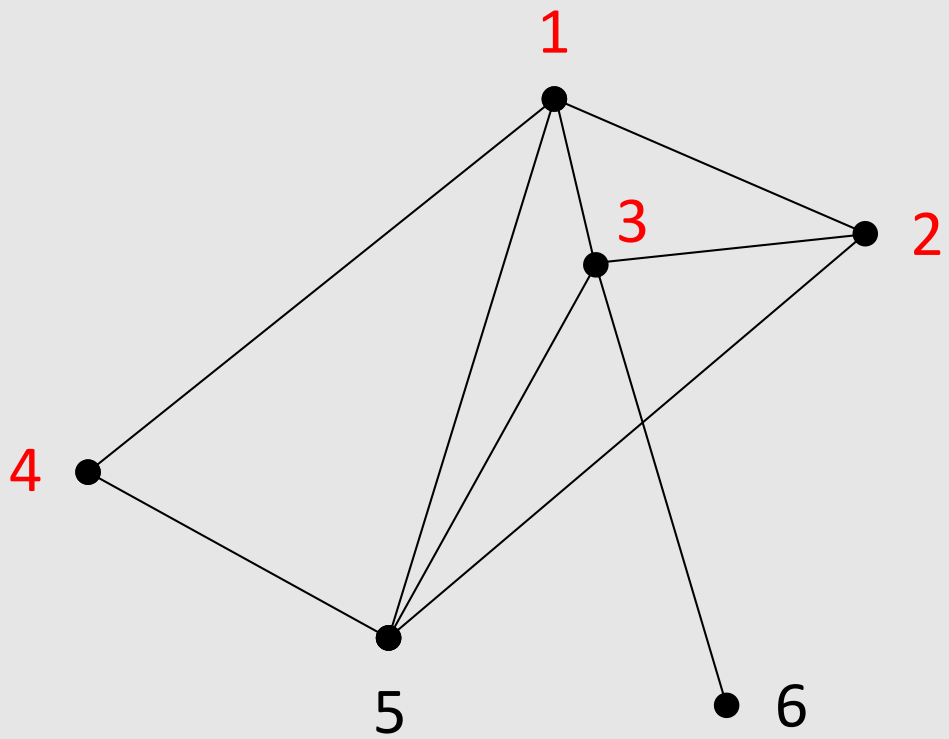


$\phi = [6, 3, 2, 1, 5, 4]$

Vértice 4 adicionado

Etapas para Obtenção da Solução

Geração da lista de peças

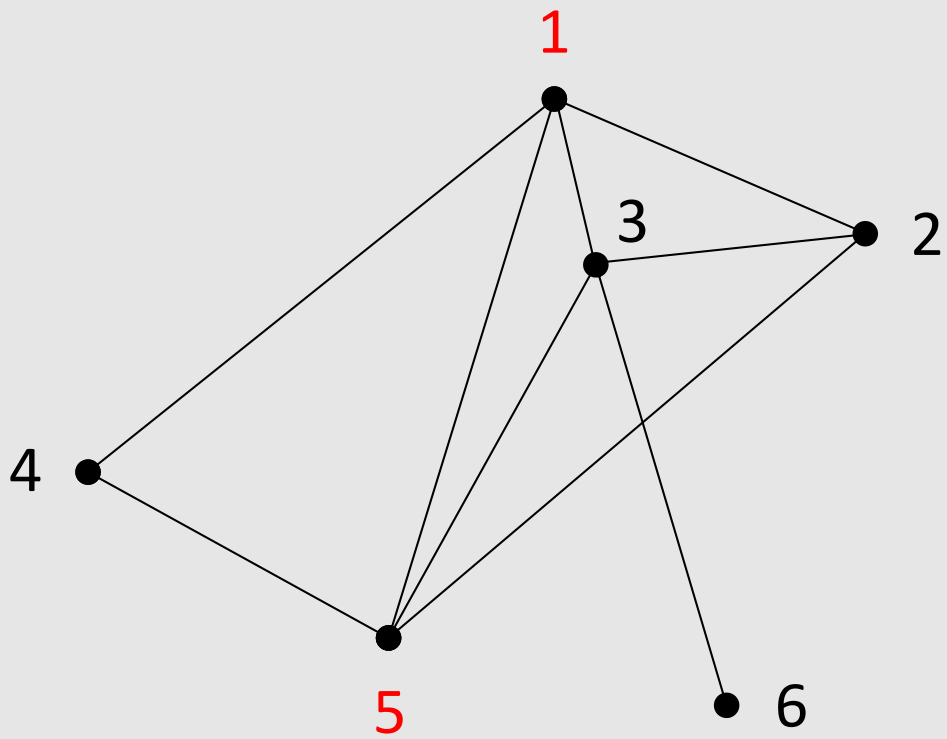


$\phi = [6, 3, 2, 1, 5, 4]$

Nenhum vértice adicionado

Etapas para Obtenção da Solução

Geração da lista de peças

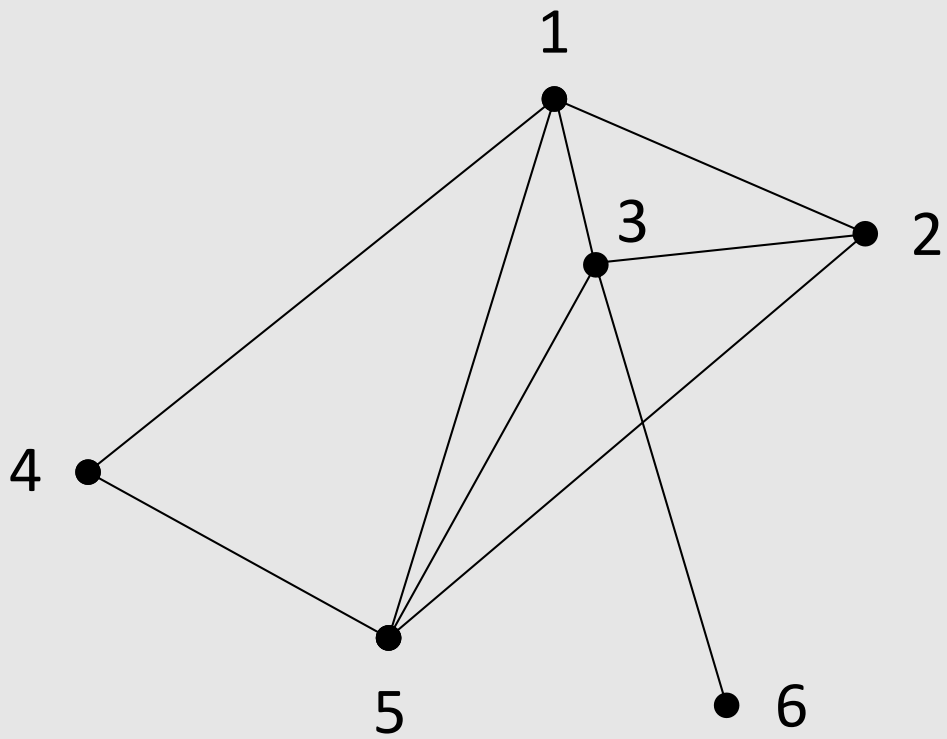


$\phi = [6, 3, 2, 1, 5, 4]$

Nenhum vértice adicionado

Etapas para Obtenção da Solução

Geração da lista de peças



$$\phi = [6, 3, 2, 1, 5, 4]$$

Etapas para Obtenção da Solução

Geração da lista de padrões

Simula a abertura das pilhas e sequencia os padrões cuja composição de peças forem um subconjunto das pilhas abertas até o momento.

	p_1	p_2	p_3	p_4	p_5
1	1	1	0	1	0
2	0	1	0	1	1
3	0	0	1	1	1
4	1	0	0	0	0
5	1	1	0	0	1
6	0	0	1	0	0

$\phi = [6, 3, 2, 1, 5, 4]$
$\pi = \emptyset$

Etapas para Obtenção da Solução

Geração da lista de padrões

	p_1	p_2	p_3	p_4	p_5
1	1	1	0	1	0
2	0	1	0	1	1
3	0	0	1	1	1
4	1	0	0	0	0
5	1	1	0	0	1
6	0	0	1	0	0

$$\phi = [6, 3, 2, 1, 5, 4]$$
$$\pi = \emptyset$$

Etapas para Obtenção da Solução

Geração da lista de padrões

	p_1	p_2	p_3	p_4	p_5
1	1	1	0	1	0
2	0	1	0	1	1
3	0	0	1	1	1
4	1	0	0	0	0
5	1	1	0	0	1
6	0	0	1	0	0

$$\phi = [6, 3, 2, 1, 5, 4]$$
$$\pi = [p_3]$$

Etapas para Obtenção da Solução

Geração da lista de padrões

	p_1	p_2	p_3	p_4	p_5
1	1	1	0	1	0
2	0	1	0	1	1
3	0	0	1	1	1
4	1	0	0	0	0
5	1	1	0	0	1
6	0	0	1	0	0

$$\phi = [6, 3, 2, 1, 5, 4]$$
$$\pi = [p_3]$$

Etapas para Obtenção da Solução

Geração da lista de padrões

	p_1	p_2	p_3	p_4	p_5
1	1	1	0	1	0
2	0	1	0	1	1
3	0	0	1	1	1
4	1	0	0	0	0
5	1	1	0	0	1
6	0	0	1	0	0

$$\phi = [6, 3, 2, 1, 5, 4]$$
$$\pi = [p_3, p_4]$$

Etapas para Obtenção da Solução

Geração da lista de padrões

	p_1	p_2	p_3	p_4	p_5
1	1	1	0	1	0
2	0	1	0	1	1
3	0	0	1	1	1
4	1	0	0	0	0
5	1	1	0	0	1
6	0	0	1	0	0

$$\phi = [6, 3, 2, 1, 5, 4]$$
$$\pi = [p_3, p_4, p_2, p_5]$$

Etapas para Obtenção da Solução

Geração da lista de padrões

	p_1	p_2	p_3	p_4	p_5
1	1	1	0	1	0
2	0	1	0	1	1
3	0	0	1	1	1
4	1	0	0	0	0
5	1	1	0	0	1
6	0	0	1	0	0

$$\phi = [6, 3, 2, 1, 5, 4]$$
$$\pi = [p_3, p_4, p_2, p_5, p_1]$$

Etapas para Obtenção da Solução

Aplicação dos Métodos Propostos

- Um Novo Método de Busca Local;
- Descida em Vizinhança Variável;
- Descida Mais Rápida.

Um Novo Método de Busca Local

Consiste na análise da similaridade da composição dos padrões com o gargalo da solução.

Composto de 3 passos:

1. Listar os padrões com alguma similaridade com o primeiro gargalo encontrado;
2. Ordenar os padrões em ordem decrescente de similaridade com o gargalo;
3. Remover o padrão da solução e reinseri-lo na primeira melhor posição possível.

Exemplo

Passo 1 - Listar os padrões com alguma similaridade com o primeiro gargalo encontrado.

	p_2	p_1	p_3	p_5	p_4
1	1	1	1	1	1
2	1	1	1	1	1
3	0	1	1	0	0
4	1	0	0	0	0
5	0	0	1	1	0

$$L = [p_2, p_1, p_5, p_4]$$

Exemplo

Passo 2 - Ordenar os padrões em ordem decrescente de similaridade com o primeiro gargalo encontrado.

	p_2	p_1	p_3	p_5	p_4
1	1	1	1	1	1
2	1	1	1	1	1
3	0	1	1	0	0
4	1	0	0	0	0
5	0	0	1	1	0
	2	2	-	2	2

$$L = [p_2, p_1, p_5, p_4]$$

Exemplo

Passo 3 - Remover o padrão da solução e reinseri-lo na primeira melhor posição possível.

	p_1	p_2	p_3	p_5	p_4
1	0	1	1	1	1
2	1	1	1	1	1
3	1	1	1	0	0
4	0	1	0	0	0
5	0	0	1	1	0

$L = [p_2, p_5, p_4, p_1]$

Abre 2 pilhas.

Referentes as peças 1 e 4

Exemplo

Passo 3 - Remover o padrão da solução e reinseri-lo na primeira melhor posição possível.

	p_1	p_3	p_2	p_5	p_4
1	0	0	1	1	1
2	1	1	1	1	1
3	1	1	0	0	0
4	0	0	1	0	0
5	0	1	1	1	0

$L = [p_2, p_5, p_4, p_1]$

Abre 2 pilhas.

Referentes as peças 1 e 4

Exemplo

Passo 3 - Remover o padrão da solução e reinseri-lo na primeira melhor posição possível.

	p_1	p_3	p_5	p_2	p_4
1	0	0	1	1	1
2	1	1	1	1	1
3	1	1	0	0	0
4	0	0	0	1	0
5	0	1	1	0	0

$L = [p_2, p_5, p_4, p_1]$

Abre 1 pilha.

Referente a peça 4

Melhor posição até o momento.

Exemplo

Passo 3 - Remover o padrão da solução e reinseri-lo na primeira melhor posição possível.

	p_1	p_3	p_5	p_4	p_2
1	0	0	1	1	1
2	1	1	1	1	1
3	1	1	0	0	0
4	0	0	0	0	1
5	0	1	1	0	0

$L = [p_2, p_5, p_4, p_1]$

Abre 1 pilha.

Referente a peça 4

Exemplo

Passo 3 - Remover o padrão da solução e reinseri-lo na primeira melhor posição possível.

	p_1	p_3	p_5	p_2	p_4
1	0	0	1	1	1
2	1	1	1	1	1
3	1	1	0	0	0
4	0	0	0	1	0
5	0	1	1	0	0

$L = [p_2, p_5, p_4, p_1]$

p_2 inserido no 4º estágio.

Redução de 4 para 3 pilhas abertas.

O algoritmo continua para os próximos padrões da lista. Caso haja alguma melhoria o processo de repete para nova solução encontrada.

Descida em Vizinhança Variável

Definição

O método de **Descida em Vizinhança Variável** (ou *Variable Neighborhood Descent* – VND) utiliza o conceito de busca local para solucionar problemas de otimização combinatória.

Características

- Definição de N_k vizinhanças a serem exploradas, onde cada vizinhança é composta por soluções que diferem em exatamente k elementos da solução inicial;
- A cada iteração uma vizinhança é explorada;
- Atualiza a solução corrente caso encontre um melhor vizinho;
- Permanece na vizinhança enquanto houver melhoria.

Descida em Vizinhança Variável

Características do método proposto

- Agrupamento de padrões – Grupos distintos com n padrões cada. Ex.: Seja $\pi = [1, 3, 5, 4, 2, 6]$ e n=2, os grupos seriam [1, 3], [5, 4] e [2, 6];
- *k-opt* – Método que gera soluções que diferem em exatamente k elementos da configuração inicial;
- Trocas de grupos – *k-opt* aplicado a grupos para geração dos vizinhos de cada vizinhança;
- A cada iteração uma vizinhança é explorada;
- Exploração dos vizinhos de uma vizinhança de forma aleatória;
- Aplicação da busca local proposta para cada vizinho;
- Atualiza a solução corrente caso a solução da busca local seja melhor;
- Permanece na vizinhança enquanto houver melhoria.
- Repete todo o processo caso haja melhoria.

Descida Mais Rápida

Definição

O método de **Descida Mais Rápida** (ou *Steepest Descent* – SD) é um método de busca local baseado no conceito de melhoria iterativa de uma solução inicial.

Características

- Aplicado sobre uma única vizinhança;
- Atualiza a solução corrente caso encontre um melhor vizinho;
- Repete todo o processo caso haja melhoria.

Descida Mais Rápida

Características do método proposto

- Janela deslizante – Janelas de tamanho n padrões cada. Ex.: Seja $\pi = [1, 3, 5, 4, 2]$ e n=2, as janelas seriam [1, 3], [3, 5], [5, 4] e [4, 2];
- *2-opt* – soluções que diferem em 2 elementos da solução atual;
- Trocas de janelas – *2-opt* aplicado a janelas para geração dos vizinhos;
- Exploração dos vizinhos da vizinhança de forma aleatória;
- Aplicação da busca local proposta para cada vizinho;
- Atualiza a solução corrente caso a solução da busca local seja melhor;
- Repete todo o processo caso haja melhoria.

Experimentos Computacionais

Ambiente Computacional

- Processador Intel Core i7 3.6 GHz;
- 16 GB RAM;
- Ubuntu 14.04 LTS;
- Código escrito em C++, compilado com g++ 4.4.1 e a opção de otimização -O3.

Métodos:

- Descida em Vizinhança Variável (VND);
- Descida Mais Rápida (SD).

Experimentos Computacionais

Instâncias

1. *SCOOP Consortium* – 24 instâncias MOSP reais de duas empresas moveleiras europeias.
2. *VLSI* – 25 instâncias reais;
3. *Faggioli & Bentivoglio* (F & B) – 300 instâncias artificiais;
4. *Challenge* – 46 instâncias artificiais;
5. Instâncias MOSP – 200 instâncias MOSP de maiores dimensões geradas aleatoriamente por (Chu e Stuckey, 2009).

20 testes foram executados para cada conjunto de instâncias.

Resultados Médios

- OPT – Média das soluções ótimas;
- T – Tempo médio em segundos;
- S^* – Média das melhores soluções encontradas;
- σ – Desvio padrão das soluções obtidas (Componente aleatório).

Conjunto	OPT	VND			SD		
		T	S^*	σ	T	S^*	σ
SCOOP	7,75	0,16	7,96	0,26	0,05	7,88	0,24
VLSI	7,12	163,74	7,16	0,13	1,53	7,24	0,18
F & B	93,00	0,15	94,83	0,26	0,05	94,17	0,27
Challenge	21,76	427,35	21,96	0,18	7,40	21,91	0,19
MOSP	1028,88	255,68	1041,25	0,30	15,61	1045,00	0,58

gap

$$gap = 100 \times \frac{S^* - OPT}{OPT} \tag{4}$$

Distância das soluções obtidas em relação as soluções ótimas.

Conjunto	*VND*	*SD*
	gap	*gap*
SCOOP	2,69%	1,61%
VLSI	0,56%	1,69%
F & B	1,97%	1,25%
Challenge	0,90%	0,70%
MOSP	1,20%	1,57%

Conclusão

Este trabalho propõe dois métodos para solução do MOSP. O primeiro consiste na metaheurística Descida em Vizinhança Variável (VND) e o segundo consiste em um método de Descida Mais Rápida (SD). Primeira vez em que esses métodos são aplicados a resolução do MOSP.

Os experimentos computacionais demonstraram a eficiência dos métodos propostos, principalmente em relação a qualidade da solução obtida. O método de Descida Mais Rápida (SD) apresentou menores tempos mantendo a qualidade da solução, superando inclusive o VND em alguns conjuntos de instâncias. Os *gaps* e desvio padrão baixos demonstram a robustez dos métodos ao obterem soluções próximas aos valores ótimos.

Fim