Metaheurísticas

Introdução à Otimização

Gustavo Peixoto Silva

Métodos de Refinamento

Métodos de Refinamento = Busca Local

- Método da Descida/Subida = Min/Max
  - Aplicação ao PCV
  - Aplicação ao Problema da Mochila
- Método de Descida com Primeira Melhora
- Método de Descida Randômica
- Método de Descida em Vizinhança Variável- VND

Heurísticas de refinamento

- Baseadas na noção de Vizinhança
- Cada vizinhança de uma solução está associada a um tipo de Movimento (modificação na sol.)
- Para um movimento m_i temos a Vizinhança $N_i(s)$ da solução s.
- Portanto, $\forall\ s' \in N_i(s)$, temos que $s' = m_i(s)$

Noção de Vizinhança (Neighborhood)

N1. Meus vizinhos são todas as casas que estão no meu quarteirão.

N2. Meus vizinhos são todas as casas que eu posso acessar sem passar na frente de outras casas. Obs. Eu posso pular o muro!

N3. Meus vizinhos são todas as casas que estão num raio de 50 m da minha casa.

N4. Meus vizinhos são ...

Noção de Movimento

N1 = Meus vizinhos são todas as casas que estão no meu quarteirão.

Portanto, se o Zé (s´) for vizinho da minha casa (s), segundo esta estrutura, então ao fazer o movimento m_1 de percorrer o meu quarteirão, eu encontrarei a casa do Zé.

Matematicamente temos

- Se $s' \in N_1(s)$, temos que $s' = m_1(s)$

Noção de Movimento

N2 = Meus vizinhos são todas as casas que eu posso acessar sem passar na frente de outras casas. Obs. Eu posso pular o muro! Portanto são as casas “coladas” na minha além das casas que estiverem bem na frete da minha casa.

Portanto, se o João (s´) for vizinho da minha casa (s), segundo esta estrutura, o movimento m_2 de acessar as casas coladas à minha ou que estão na frente da minha permitirá que eu chegue à casa do João.

Matematicamente temos

- Se $s' \in N_2(s)$, temos que $s' = m_2(s)$

Heurísticas de refinamento

- Seja N uma função que associa a cada solução $s \in S$, sua vizinhança N(s) $\subseteq$ S
- A vizinhança N depende do problema tratado
- Cada solução $s' \in$ N(s) é chamada vizinho de s
- Denomina-se movimento a uma modificação m que transforma uma solução s em outra s' que esteja em sua vizinhança:

$$s' \leftarrow s \oplus m, \text{ ou seja, } s' = m(s)$$

Heurísticas de refinamento

- Parte de uma solução inicial qualquer
- Caminha de um vizinho para outro vizinho (de acordo com o movimento adotado) tentando encontrar uma solução melhor do que a solução anterior, dita solução corrente.

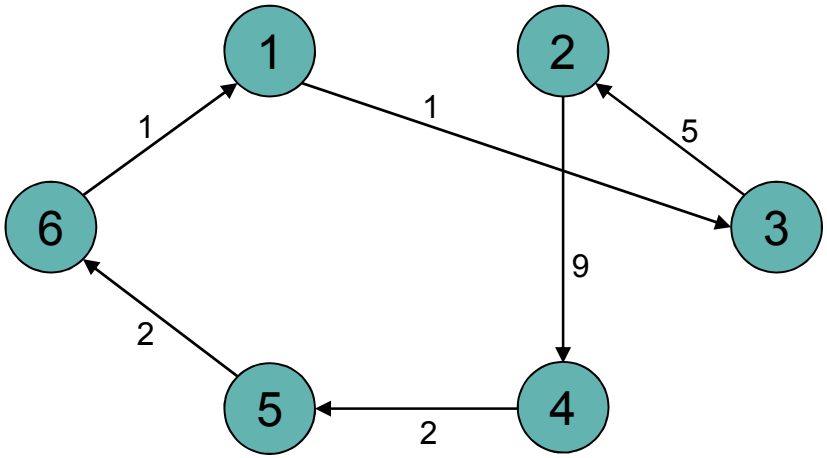
Método da descida/subida (*Descent/Uphill Method*)

- Parte de uma solução inicial qualquer
- A cada passo analisa todos os possíveis vizinhos
- Move para o vizinho que apresenta a maior melhoria no valor atual da função de avaliação (objetivo)
- O método é interrompido quando um ótimo local é encontrado, ou seja, nenhum vizinho for melhor do que a solução corrente (segundo aquele movimento).

- *Vizinhança do quarteirão*
- *Vizinhança das casas mais próximas*

PCV – Solução final da Inserção mais barata

Cid.	1	2	3	4	5	6
1	0	2	1	4	9	1
2	2	0	5	9	7	2
3	1	5	0	3	8	6
4	4	9	3	0	2	6
5	9	7	8	2	0	2
6	1	2	6	6	2	0



- **Distância Total =** 16 + 4 = 20

s = (6 1 3 2 4 5)

PCV – Solução final

Cid.	1	2	3	4	5	6
1	0	2	1	4	9	1
2	2	0	5	9	7	2
3	1	5	0	3	8	6
4	4	9	3	0	2	6
5	9	7	8	2	0	2
6	1	2	6	6	2	0

- **Distância Total = 20**

s = (6 1 3 2 4 5)

movimento: troca a posição de 2 cidades contíguas na sol. corrente

s1 = (1 6 3 2 4 5), D= 1+6+5+9+2+9 = 32

s2 = (6 3 1 2 4 5), D= 6+1+2+9+2+2 = 22

s3 = (6 1 2 3 4 5), D= 1+2+5+3+2+2 = 15

s4 = (6 1 3 4 2 5), D= 1+1+3+9+7+2 = 23

s5 = (6 1 3 2 5 4), D= 1+1+5+7+2+6 = 22

s6 = (5 1 3 2 4 6), D= 9+1+5+9+6+2 = 32

PCV – Solução final

Cid.	1	2	3	4	5	6
1	0	2	1	4	9	1
2	2	0	5	9	7	2
3	1	5	0	3	8	6
4	4	9	3	0	2	6
5	9	7	8	2	0	2
6	1	2	6	6	2	0

- **Distância Total = 15**

s = (6 1 2 3 4 5)

movimento: troca a posição de 2 cidades contíguas

s1 = (1 6 2 3 4 5), D= 1+2+5+3+2+9 = 22

s2 = (6 2 1 3 4 5), D= 2+2+1+3+2+2 = 12

s3 = (6 1 3 2 4 5), D= 1+1+5+9+2+2 = 20

s4 = (6 1 2 4 3 5), D= 1+2+9+3+8+2 = 25

s5 = (6 1 2 3 5 4), D= 1+2+5+8+2+6 = 24

s6 = (5 1 2 3 4 6), D= 9+2+5+3+6+2 = 27

PCV – Solução final

Cid.	1	2	3	4	5	6
1	0	2	1	4	9	1
2	2	0	5	9	7	2
3	1	5	0	3	8	6
4	4	9	3	0	2	6
5	9	7	8	2	0	2
6	1	2	6	6	2	0

- **Distância Total = 12**

s = (6 2 1 3 4 5)

movimento: troca a posição de 2 cidades contíguas

s1 = (2 6 1 3 4 5), D= 2+1+1+3+2+7 = 16

s2 = (6 1 2 3 4 5), D= 1+2+5+3+2+2 = 15

s3 = (6 2 3 1 4 5), D= 2+5+1+4+2+2 = 16

s4 = (6 2 1 4 3 5), D= 2+2+4+3+8+2 = 21

s5 = (6 2 1 3 5 4), D= 2+2+1+8+2+6 = 21

s6 = (5 2 1 3 4 6), D= 7+2+1+3+6+2 = 21

Como não tem vizinho melhor, s = (6 2 1 3 4 5) é uma solução ótima local para a vizinhança definida. Isso não significa que s seja ótima para uma outra estrutura de vizinhança. Somente o Ótimo Global tem esta característica!!!
Ou seja, de ótima local para qualquer vizinhança.

Sequenciameto de Tarefas

Solução Inicial

Obtida pelo método gulosa *Earleast Due Date* – EDD
A tarefa com o menor tempo de entrega é realizada pela máquina com menor tempo total de processamento.

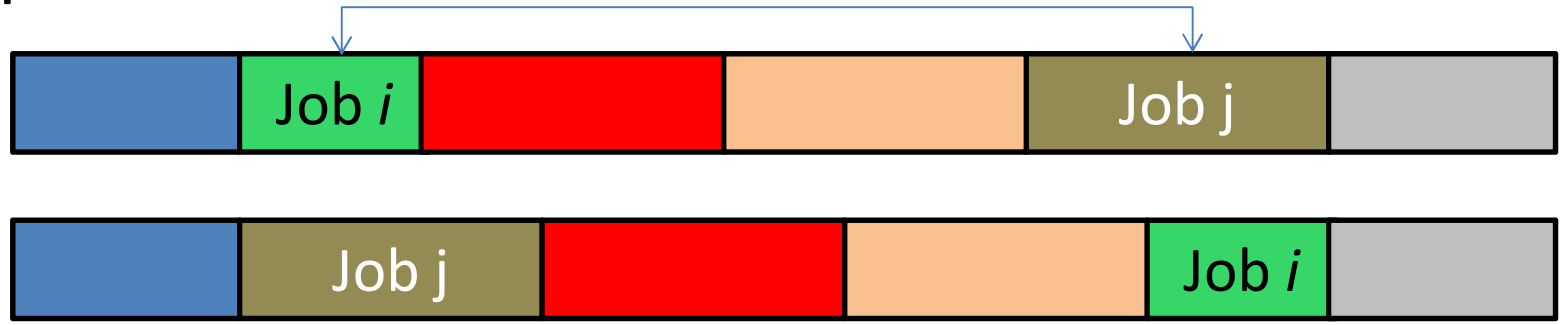
Busca Local para uma única Máquina

Se baseiam no seguintes movimentos:
Swap: $\alpha\ i\ \pi\ j\ \omega \rightarrow \alpha\ j\ \pi\ i\ \omega$, onde π pode ser um ou uma sequencia de tarefas

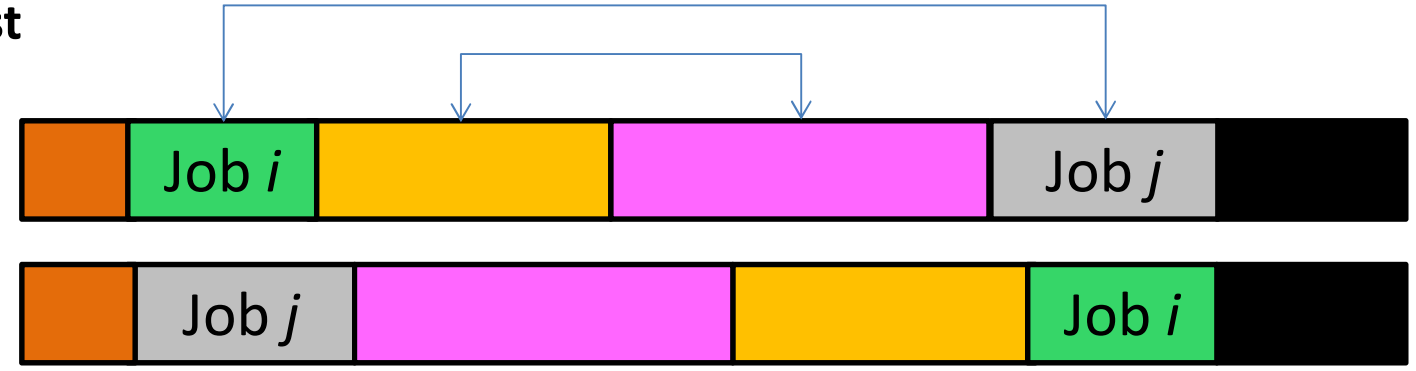
Twist: $\alpha\ i\ \pi\ j\ \omega \rightarrow \alpha\ j\ \pi'\ i\ \omega$, onde π' = π revertido

Busca Clássica em Sequenciamento de Máquinas

Swap



Twist



Método da descida
(*Descent Method*)

```
procedimento Descida(f(.), N(.), s);
1   V = {s' ∈ N(s) | f(s') < f(s)};
2   enquanto (|V| > 0) faça
3       Selecione s' ∈ V, onde s' = arg min{f(s') | s' ∈ V};
4       s ← s' ;
5       V = {s' ∈ N(s) | f(s') < f(s)};
6   fim-enquanto;
7   Retorne s;
fim Descida;
```

Método de Descida do Melhor Vizinho
(*Best Improvement Method*)

Método da Subida aplicado ao Problema da Mochila

Seja uma mochila de capacidade b = 23

Objeto (j)	1	2	3	4	5
Peso (w_j)	4	5	7	9	6
Benefício (p_j)	2	2	3	4	4

Representação de uma solução: $s = (s_1, s_2, ..., s_5)$, onde $s_j \in \{0,1\}$

Movimento *m* = troca no valor de um bit.

Ou seja, muda o *status* do objeto (vai/não vai na mochila)

Método da Subida aplicado ao Problema da Mochila

Função de avaliação:

$$f(s) = \sum_{j=1}^{n} p_j s_j - \alpha \times \max\{0, \sum_{j=1}^{n} w_j s_j - b\} \quad (31)$$

sendo α uma penalidade, por exemplo, $\alpha = \sum_{j=1}^{n} p_j = 15$.

Método da Subida aplicado ao Problema da Mochila

Passo 0 : Seja uma solução inicial qualquer, por exemplo:

$s = (01010)^t$

$f(s) = 6$

Peso corrente da mochila = 14

Passo 1 : Devemos, agora, analisar todos os vizinhos de s e calcular a função de avaliação deles usando a função de avaliação (31).

Vizinhos de s	Peso dos vizinhos de s	Benefício dos vizinhos de s	$f(s')$
$(\mathbf{1}1010)^t$	18	8	8
$(0\mathbf{0}010)^t$	9	4	4
$(011\mathbf{1}0)^t$	21	9	9
$(0100\mathbf{0})^t$	5	2	2
$(0101\mathbf{1})^t$	20	10	10

Melhor vizinho: $s' = (01011)^t$

$f(s') = 10$

Como s' é melhor que s, pois $f(s') > f(s)$, então $s \leftarrow s'$, isto é, a nova solução corrente passa a ser:

$s = (01011)^t$

Método da Subida com primeiro de melhora aplicado ao Problema da Mochila

(1, 1, 0, 1, 0)	**F(s) = 8**		
(**0**, 1, 0, 1, 0)	F(s) = 6		
(1, **0**, 0 , 1, 0)	F(s) = 6		
(1, 1, **1**, 1 , 0)	F(s) = -19		
(1, 1, 0, 0, 0)	F(s) = 4		
(1, 1, 0, 1, 1)	F(s) = -3		

Portanto, a solução s = (1, 1, 0, 1, 0) com f(s) = 8 é um ótimo local para esta busca e partindo da solução anterior.

Método da Subida aplicado ao Problema da Mochila

Passo 2 : Determinemos, agora, o melhor vizinho de $s = (01011)^t$:

Vizinhos de s	Peso dos vizinhos de s	Benefício dos vizinhos de s	$f(s')$
$(\mathbf{1}1011)^t$	24	12	-3
$(0\mathbf{0}011)^t$	15	8	8
$(01\mathbf{1}11)^t$	27	13	-47
$(0100\mathbf{1})^t$	11	6	6
$(0101\mathbf{0})^t$	14	6	6

Melhor vizinho: $s' = (00011)^t$
$f(s') = 8$
Como $f(s')$ é pior que $f(s)$, pois $f(s') < f(s)$, então PARE. A solução anterior é um ótimo local, isto é, o método da subida retorna $s^* = (01011)^t$, com $f(s^*) = 10$ como solução final.

Método de Primeira Melhora (*First Improvement Method*)

- Variante do Método de Descida/Subida
- Evita a pesquisa exaustiva pelo melhor vizinho
- Interrompe a exploração da vizinhança quando um vizinho melhor é encontrado
- Desta forma, apenas no pior caso, toda a vizinhança será explorada
- A solução final é um ótimo local com respeito à vizinhança considerada

Método de Primeira Melhora para minimização

```
procedimento Primeira_Melhora(f(.), N(.), s);
1. Enquanto houver vizinho melhor do que s faça
2.     encontre um vizinho s' em N(s)
3.     tal que f(s') < f(s) (f(s') > f(s) para Max)
4.     s := s';
5. Fim-enquanto
Fim Primeira_Melhora;
```

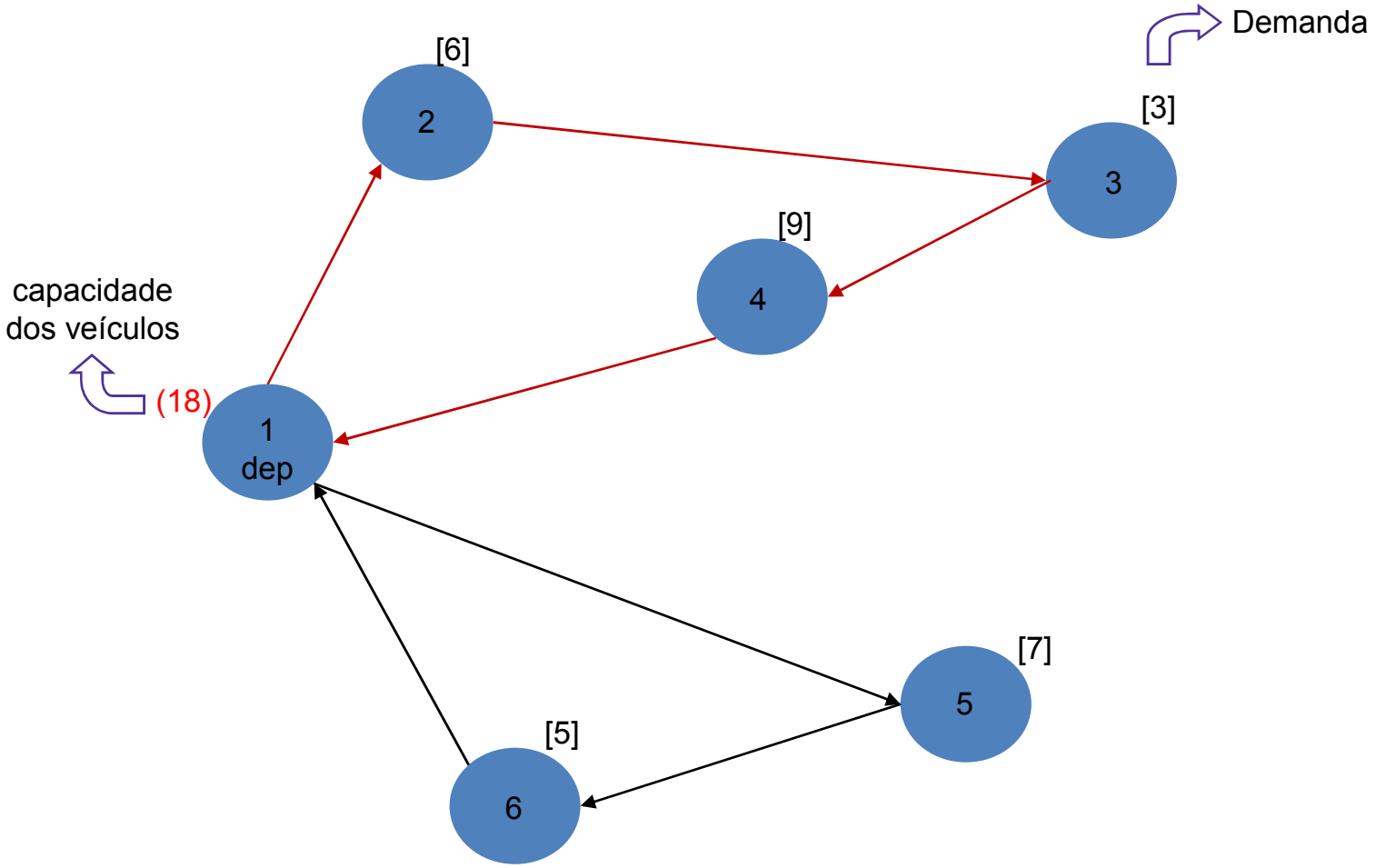
Método de Descida/Subida Randômica (*Random Descent/Uphill Method*)

- Variante do Método de Descida/Subida
- Evita a pesquisa exaustiva pelo melhor vizinho
- Consiste em escolher um vizinho qualquer e o aceitar somente se ele for melhor do que a melhor solução conhecida até o momento
- Se o vizinho escolhido não for melhor, a solução corrente (melhor) permanece inalterada e outro vizinho é gerado
- O procedimento é interrompido após um certo número fixo de iterações sem melhora no valor da função objetivo
- A solução final não é necessariamente um ótimo local
- Por outro lado, é possível controlar melhor o tempo de processamento

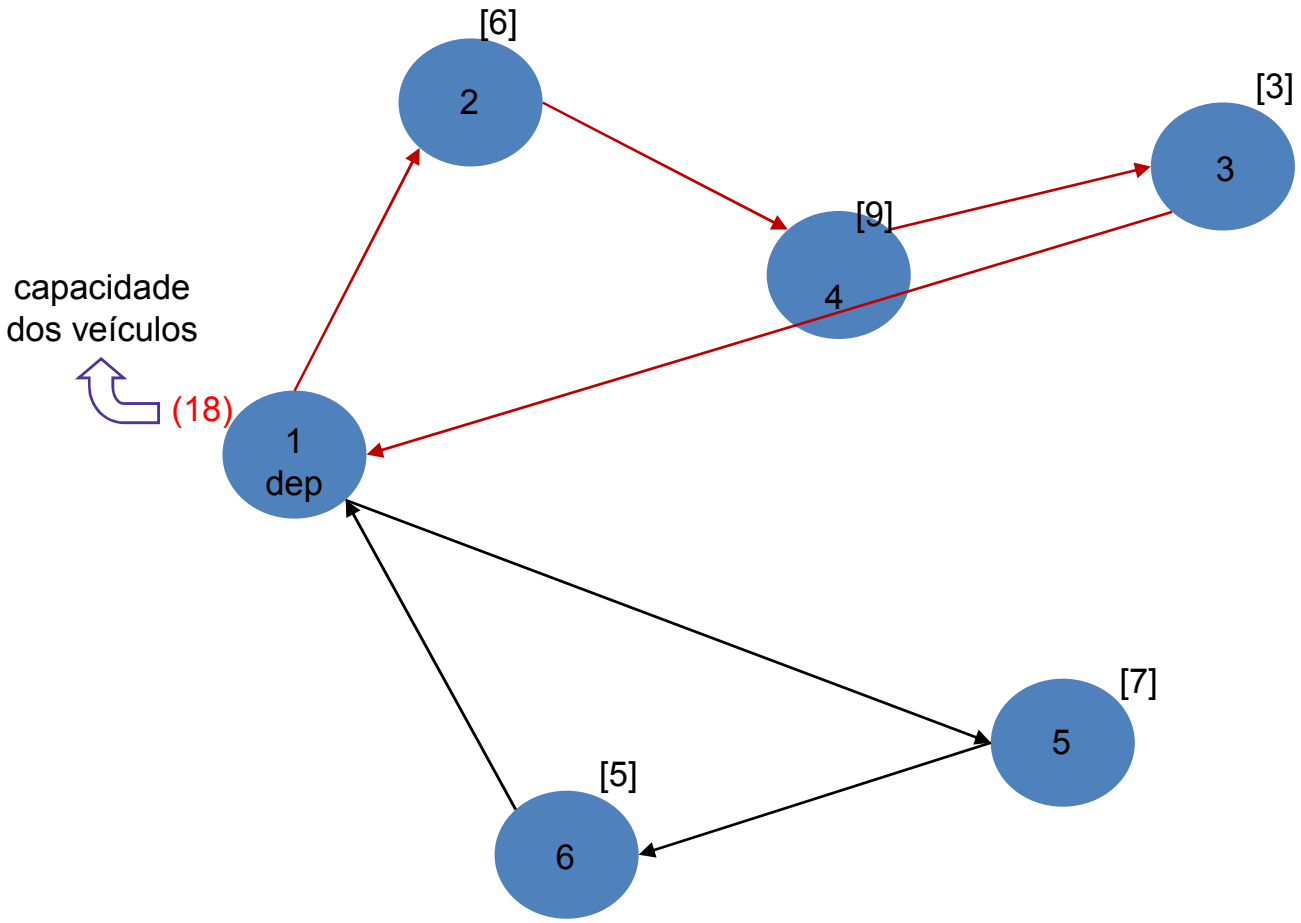
Método de Descida Randômica
(*Random Descent Method*)

```
procedimento DescidaRandomica(f(.), N(.), IterMax, s);
1   Iter ← 0;      {Contador de iterações sem melhora }
2   enquanto (Iter < IterMax) faça
3       Iter ← Iter + 1;
4       Selecione aleatoriamente s' ∈ N(s);
5       se (f(s') < f(s)) então
6           Iter ← 0;
7           s ← s' ;
8       fim-se;
9   fim-enquanto;
10  Retorne s;
fim DescidaRandomica;
```

Problema de Roteamento de Veículos

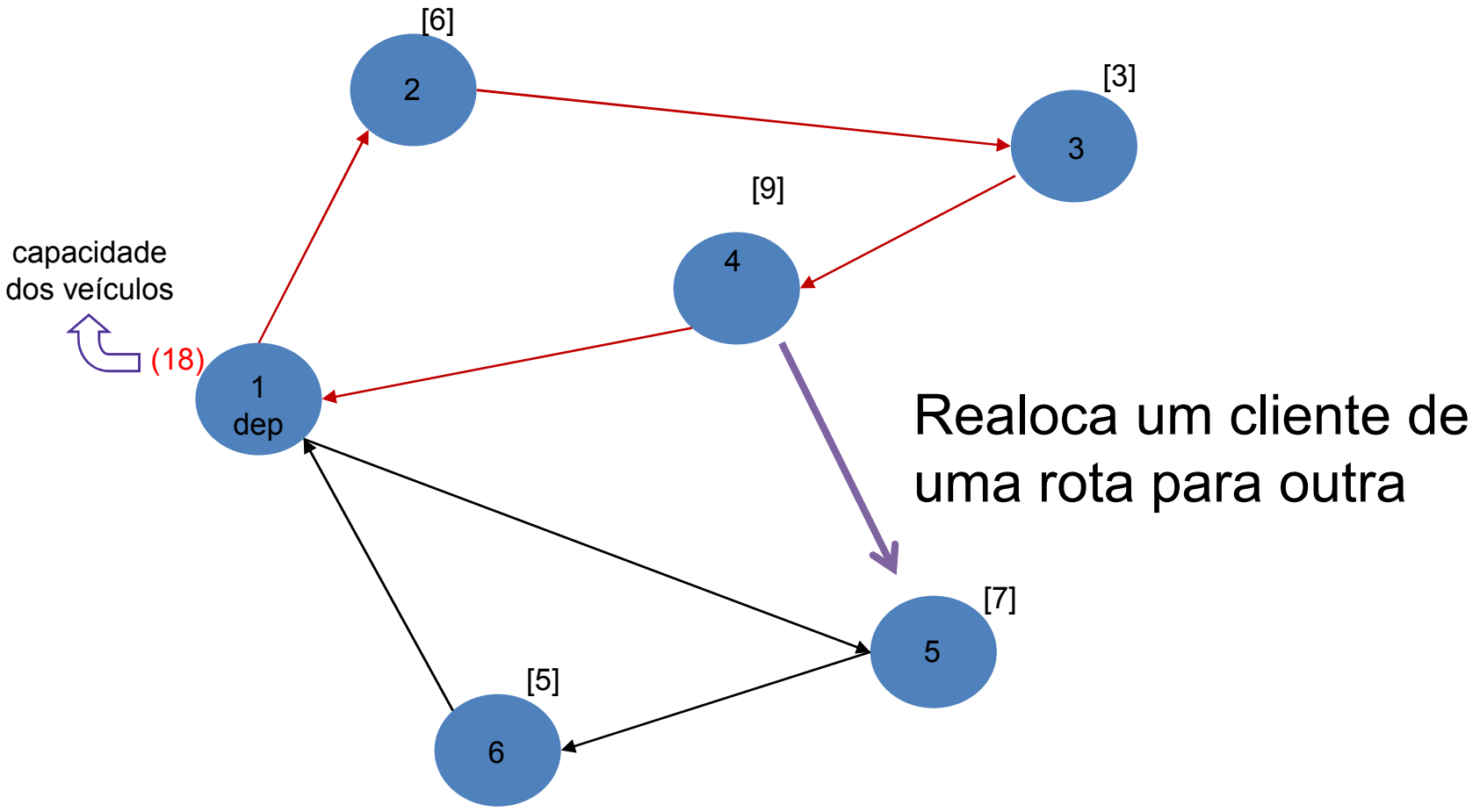


Vizinhança 1 - Movimento Intra-rota

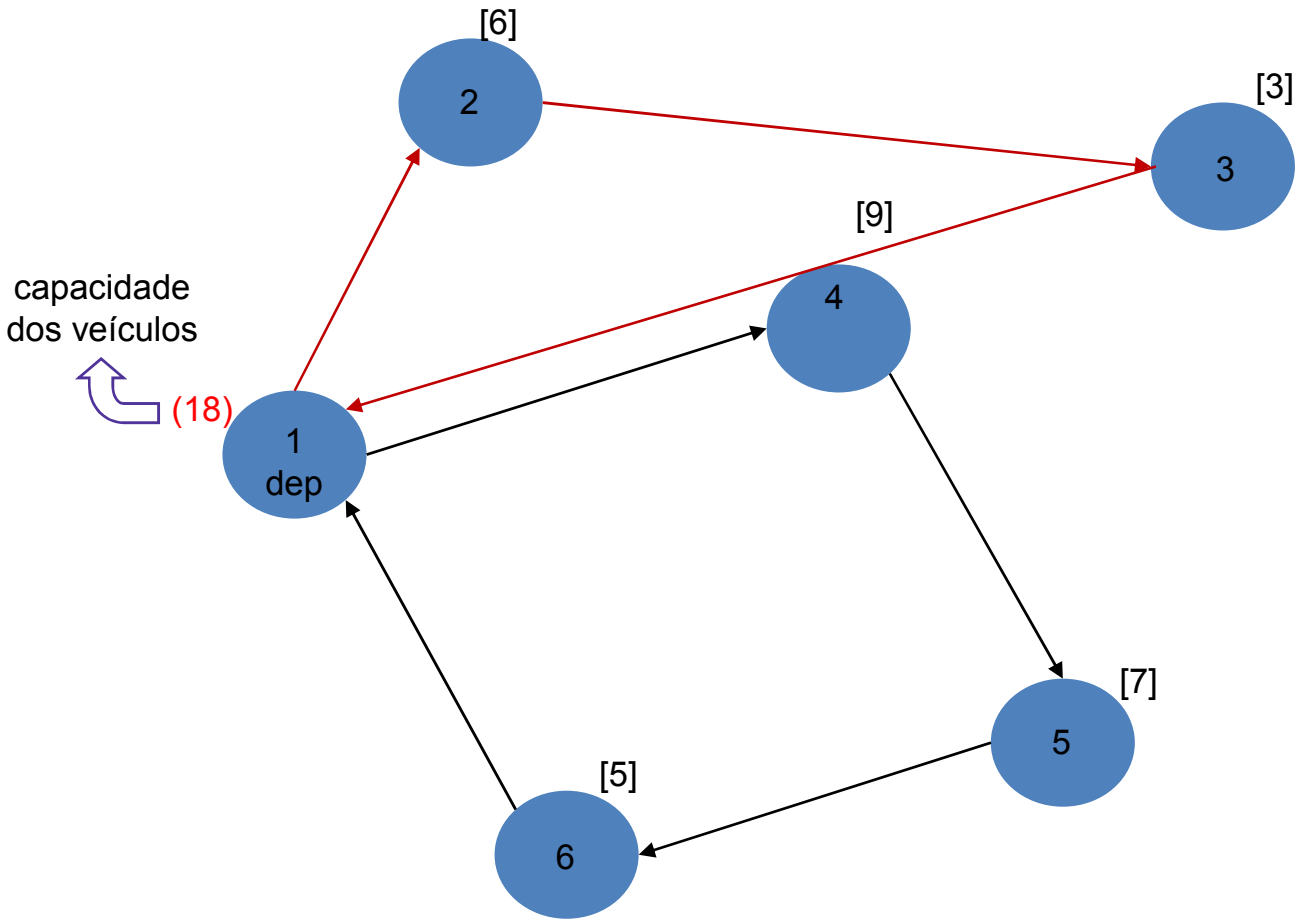


Troca a ordem de dois clientes dentro da rota

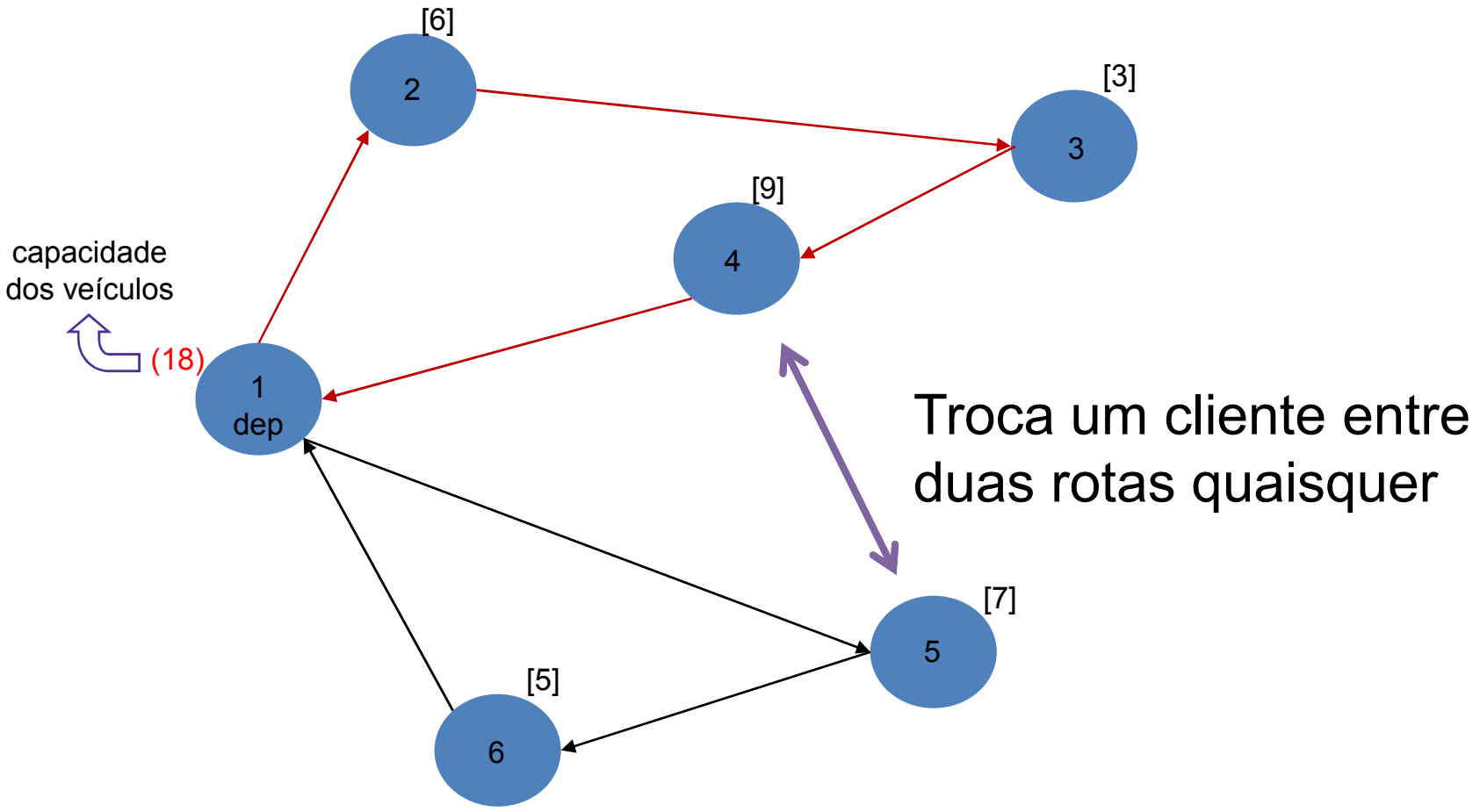
Vizinhança 2 – Movimento Inter-rotas
Realoca 1



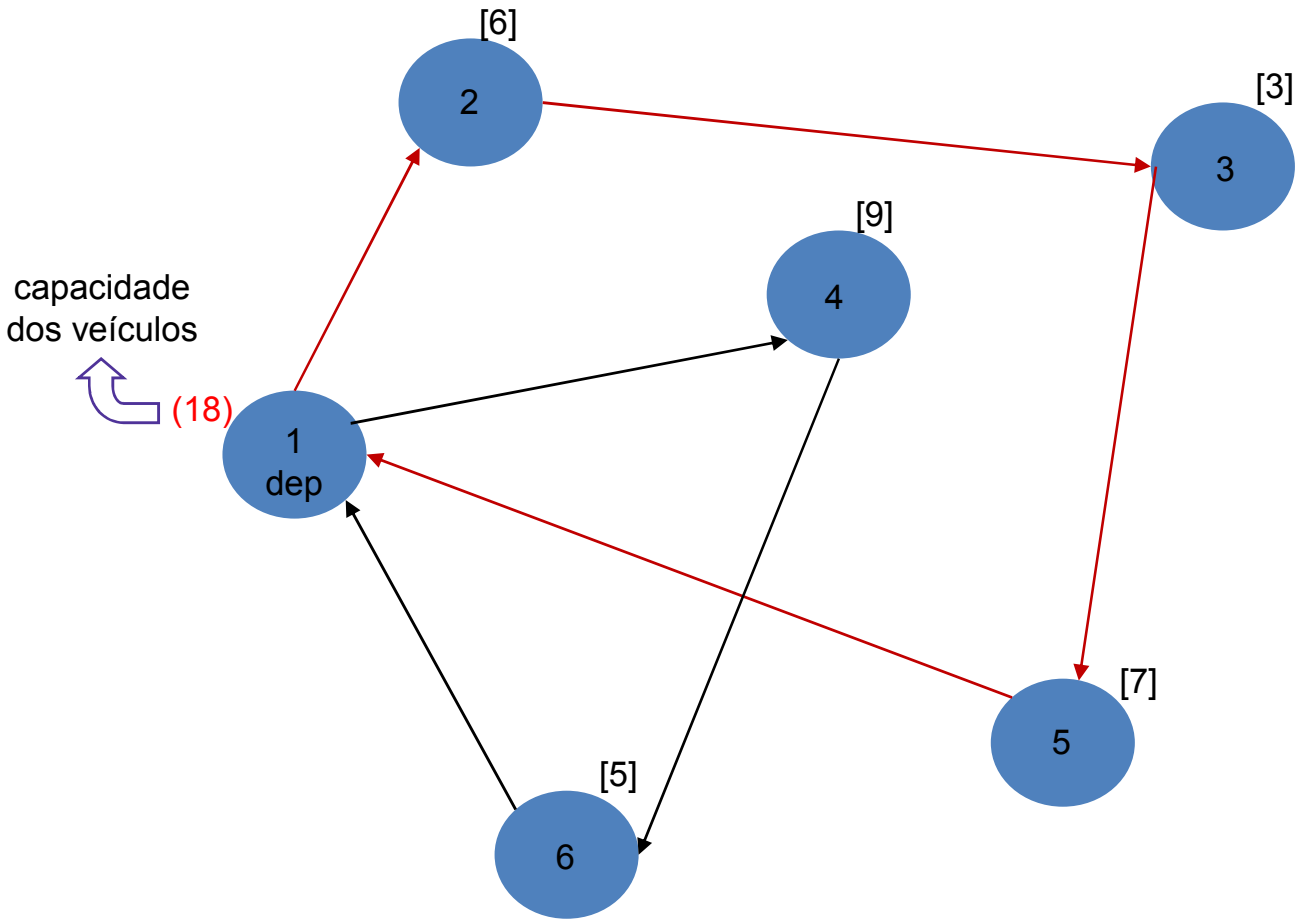
Vizinhança 2 – Movimento Inter-rotas Realoca 1



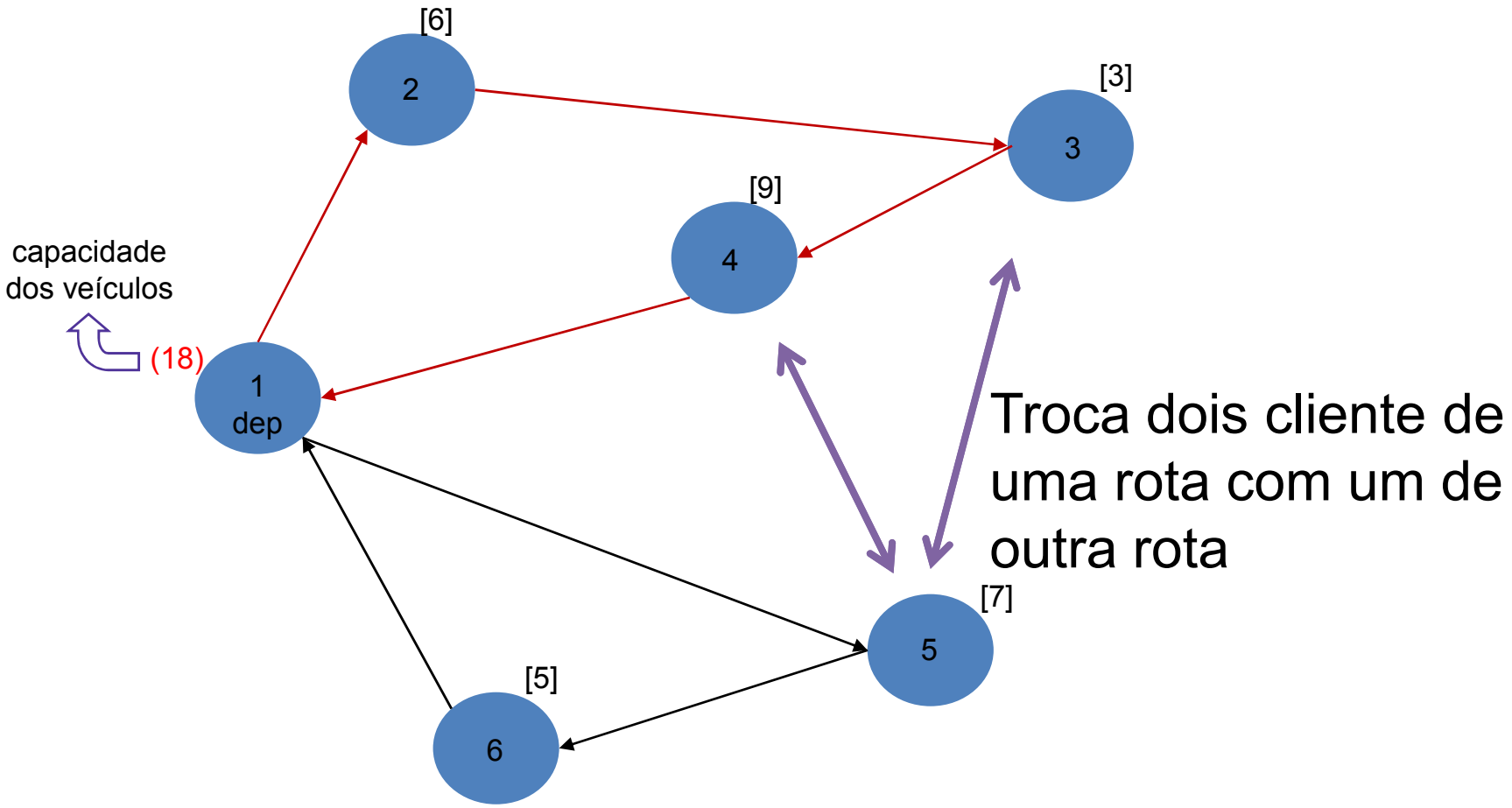
Vizinhança 3 – Movimento Inter-rotas
Troca 1-1



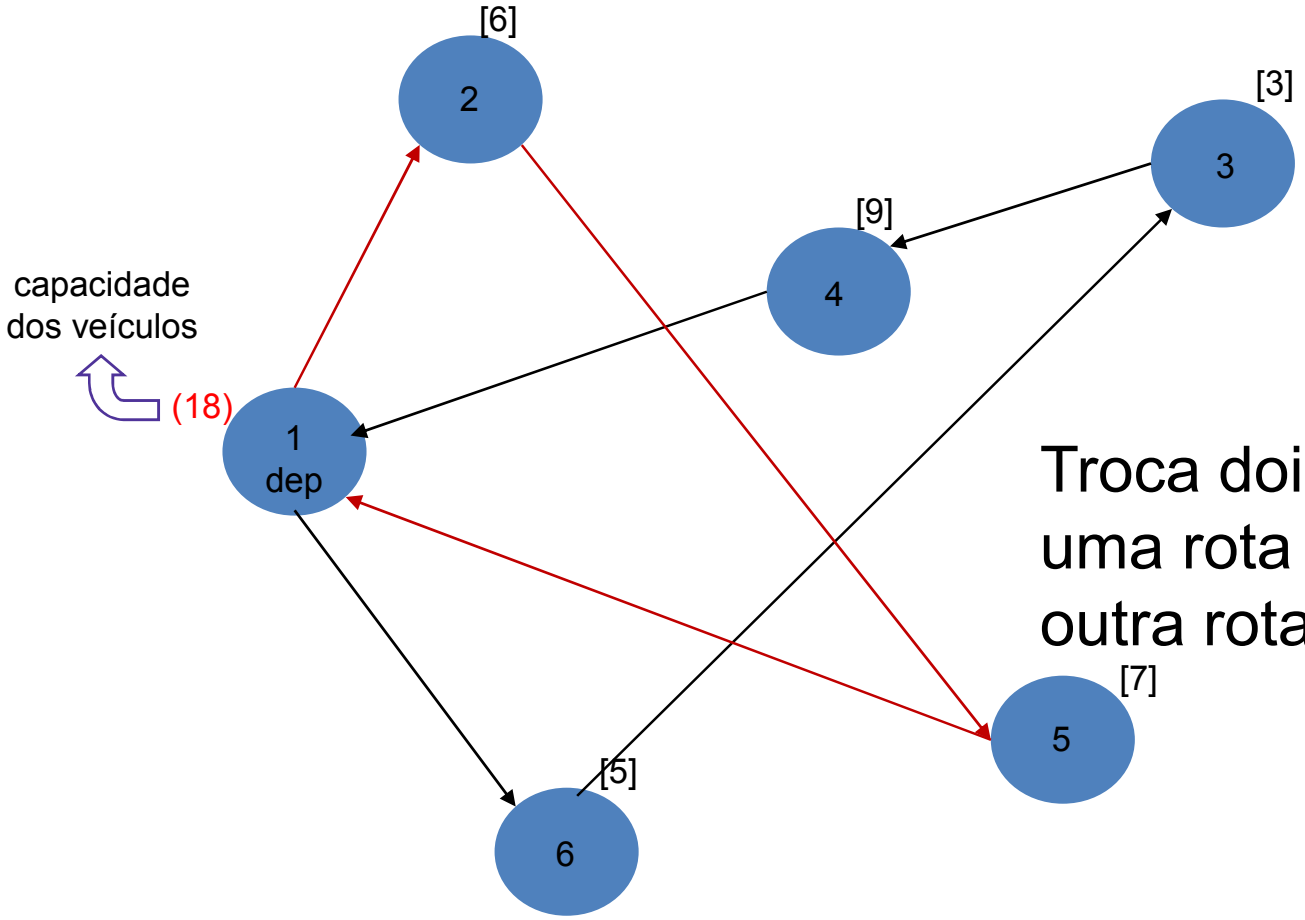
Vizinhança 3 – Movimento Inter-rotas
Troca 1-1



Vizinhança 4 – Movimento Inter-rotas
Troca 2-1



Vizinhança 4 – Movimento Inter-rotas
Troca 2-1



Troca dois cliente de uma rota com um de outra rota

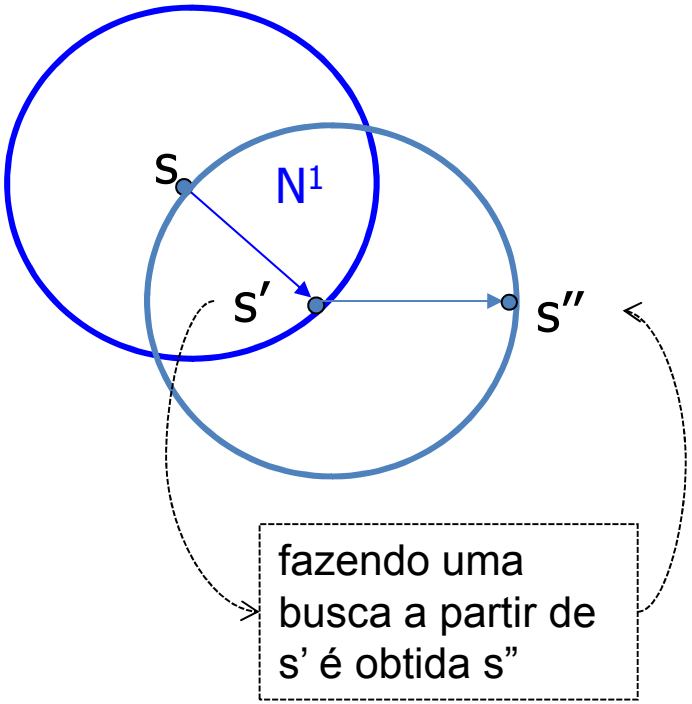
Vizinhança - Movimento

- Existem diferentes tipos de vizinhança em função dos diferentes movimentos que podem ser feitos para modificar uma solução
- Há movimentos que “modificam mais” a solução do que outros
- Ou seja, movimentos que analisam soluções “mais próximas” da solução corrente e outros que analisam soluções “mais distantes” da solução corrente

Método de Descida em Vizinhança Variável (*Variable Neighborhood Descent*)

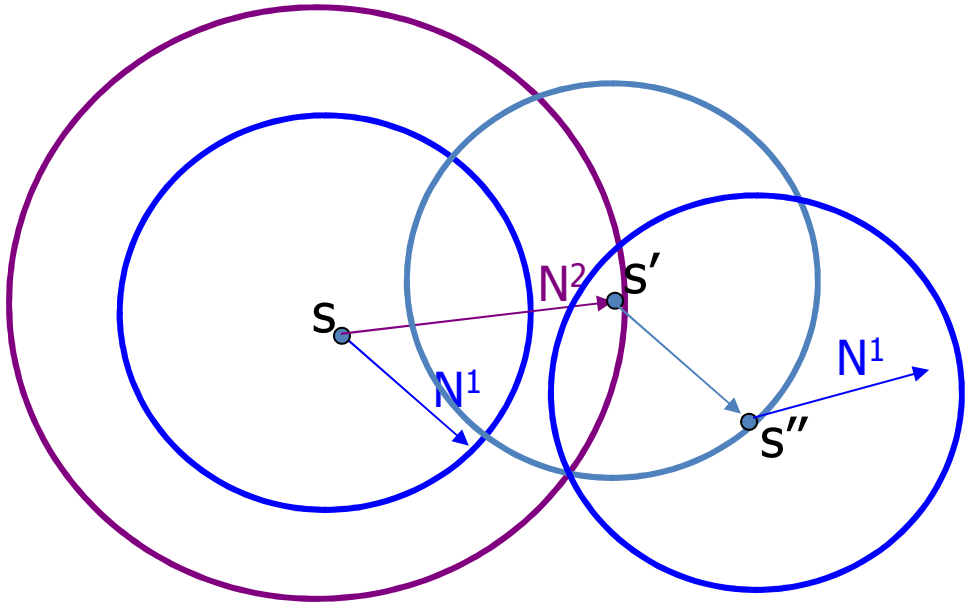
```
procedimento VND(f(.),N(.),r,s)
1   Seja r o número de estruturas diferentes de vizinhança;
2   k ← 1;                {Tipo de estrutura de vizinhança corrente}
3   enquanto (k ≤ r) faça
4       Encontre o melhor vizinho s' ∈ N^(k)(s);
5       se (f(s') < f(s))
6           então
7               s ← s';
8               k ← 1;
9           senão
10              k ← k + 1;
11      fim-se;
12  fim-enquanto;
13  Retorne s;
fim VND;
```

Variable Neighborhood Search



s'' será a nova solução ótima local se f(s'') < f(s)

Variable Neighborhood Search



Variable Neighborhood Search