Heurísticas e Metaheuristicas

Busca Tabu

Prof. Guilherme de Castro Pena
guilherme.pena@ufsj.edu.br
Sala: DCOMP 3.11

Departamento de Ciência da Computação
Universidade Federal de São João del-Rei

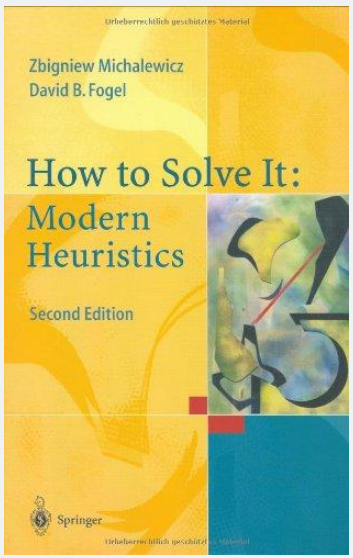
Material adaptado do Prof. André (UFV) e Prof. Marcone (UFOP)

Agenda

Referência

Livro

- Esse conteúdo está baseado no livro:
- MICHLEWICZ, Zbigniew; FOGEL, David B. How to solve it: modern heuristics. 2nd. ed. Berlin: Springer c2010 554 p. ISBN 9783642061349 (Capítulo 5, seção 5.2).



Introdução

Visão geral:

- A **Busca Tabu** (*Tabu Search*) é uma *meta-heurística* poderosa que também possui a característica de **escapar de ótimos locais**.
- Sua ideia principal é fazer uso de uma *memória* que força a exploração de novas áreas do espaço de busca.
- Ou seja, memorizar soluções examinadas recentemente, que se tornam **tabu** (proibidas) ao tomar decisões na seleção da próxima solução.
- Outro detalhe é que diferentemente do *simulated annealing* padrão, a busca tabu é determinística por padrão.

Introdução

Visão geral:

- De forma breve vamos ver um comparativo entre os procedimentos:

Algorithm 1: Simulated Annealing

```
1 s ← s_0 (Solucao inicial)
2 while Nao condicao de termino do
3 |   s ← melhora?(s, T)
4 |   atualiza(T)
5 end
6 return s
```

Algorithm 2: Busca Tabu

```
1 s ← s_0 (Solucao inicial)
2 while Nao condicao de termino do
3 |   s ← melhora?(s, H)
4 |   atualiza(H)
5 end
6 return s
```

- A busca tabu é quase idêntica ao SA com respeito à estrutura do algoritmo.
- A função ***melhora?(x, H)*** retorna uma solução aceita s' a partir da vizinhança de s, que não precisa ser melhor que s, mas a aceitação é baseada na **história da busca H**.
- Todo o resto é similar (pelo menos, de uma perspectiva mais abrangente).

Introdução

Visão geral:

- No entanto seguem diferenças importantes entre eles:
  1. **SA é estocástico** enquanto a **Busca Tabu (BT) é determinística**.
  2. O SA pode escolher solução pior *a qualquer momento*. Enquanto a BT só escolhe solução pior *quando está preso num ótimo local*.
  3. Em relação aos parâmetros, ambos tem iterações, o SA contém *temperatura* e *taxa de resfriamento*. Enquanto a BT tem o que chamamos de **memória** e **critério de aspiração**.

Introdução

Visão geral:

- No entanto seguem diferenças importantes entre eles:
  1. **SA é estocástico** enquanto a **Busca Tabu (BT) é determinística**.
  2. O SA pode escolher solução pior *a qualquer momento*. Enquanto a BT só escolhe solução pior *quando está preso num ótimo local*.
  3. Em relação aos parâmetros, ambos tem iterações, o SA contém *temperatura* e *taxa de resfriamento*. Enquanto a BT tem o que chamamos de **memória** e **critério de aspiração**.

Introdução

Visão geral:

- No entanto seguem diferenças importantes entre eles:
  1. **SA é estocástico** enquanto a **Busca Tabu (BT) é determinística**.
  2. O SA pode escolher solução pior *a qualquer momento*. Enquanto a BT só escolhe solução pior *quando está preso num ótimo local*.
  3. Em relação aos parâmetros, ambos tem iterações, o SA contém *temperatura* e *taxa de resfriamento*. Enquanto a BT tem o que chamamos de **memória** e **critério de aspiração**.

Introdução

Visão geral:

- No entanto seguem diferenças importantes entre eles:
  1. **SA é estocástico** enquanto a **Busca Tabu (BT) é determinística**.
  2. O SA pode escolher solução pior *a qualquer momento*. Enquanto a BT só escolhe solução pior *quando está preso num ótimo local*.
  3. Em relação aos parâmetros, ambos tem iterações, o SA contém *temperatura* e *taxa de resfriamento*. Enquanto a BT tem o que chamamos de **memória** e **critério de aspiração**.

Introdução

Considerações:

- Proposta por Glover e Hansen (1986).
- Metaheurística de busca local que se apoia em estruturas de memória.
- Resolução eficiente de vários problemas combinatórios, destacando-se:
  - Roteamento (Gendreau et al., 2006; Cordeau et al., 2002; Gendreau et al., 1999)
  - Sequenciamento (Allahverdi et al., 2008)
  - Programação de horários (Santos et al., 2005; Souza et al., 2004)

Introdução

Características Típicas:

- **Busca Tabu padrão (básica):**
  - *Geração da solução inicial;*
  - *Critério de escolha de vizinho;*
  - *Definição das regras de proibição: memória de curto prazo;*
  - *Critério de aspiração;*
- Busca Tabu mais avançada inclui:
  - Definição de uma memória de longo prazo: intensificação x diversificação.
  - Reconexão por Caminhos.
  - Aplicação da oscilação estratégica.

Introdução

Características Típicas:

- **Busca Tabu padrão (básica):**
  - *Geração da solução inicial;*
  - *Critério de escolha de vizinho;*
  - *Definição das regras de proibição: memória de curto prazo;*
  - *Critério de aspiração;*
- Busca Tabu mais avançada inclui:
  - Definição de uma memória de longo prazo: intensificação x diversificação.
  - Reconexão por Caminhos.
  - Aplicação da oscilação estratégica.

Agenda

Fundamentação

Exemplo:

- Vamos utilizar como exemplo o Problema da Mochila para fundamentarmos a Busca Tabu:

Problema da Mochila:

- Há um conjunto de n itens e uma mochila de capacidade C.
- A cada item está associado um peso w_i (weight) e um valor de retorno p_i (price).
- Objetivo: alocar os itens à mochila de forma que se tenha o maior valor de retorno, respeitando-se a capacidade da mochila.

Fundamentação

Exemplo:

- Podemos fazer uma formulação para o problema em Programação Linear Inteira (PLI) da seguinte forma:

Formulação em PLI do Problema da Mochila

$$\max \quad \sum_{i=1}^{n} p_i x_i$$

$$\sum_{i=1}^{n} w_i x_i \leq C$$

$$x_i \in \{0,1\} \quad \text{para} \quad i = 1, \ldots, n$$

onde:

- $x_i = 1$ se o item i é incluído na mochila ou (0) caso contrário.

Fundamentação

Exemplo:

- Podemos definir também uma estratégia de exploração do espaço de busca usando uma constante de penalidade (ρ):
  - Soluções factíveis
  - Soluções infactíveis

Função objetivo com penalidade:

$$f(s) = \sum_{i=1}^{n} p_i x_i - \rho \times max\{0, \sum_{i=1}^{n} w_i x_i - C\}$$

onde:

- ρ = penalidade por excesso de carga.

Fundamentação

Exemplo:

- Assim, definimos os componentes finais do nosso problema:

Componentes:

Representação de uma solução:

- Vetor $s = (x_1, x_2, \ldots, x_n)$, em que $x_i \in \{0, 1\}$

Solução inicial:

- Gulosa: alocar os itens mais valiosos por unidade de peso
- Aleatória

Movimento m: inverter o valor de um bit

- Valor 0 muda para 1
- Valor 1 muda para 0

Vizinhança: $N(s) = \{s' : s' \leftarrow s \oplus m\}$
Constante de Penalidade: $\rho = \sum p_i$

Fundamentação

Exemplo:

Seja mochila de capacidade $C = 32$:

Item i	1	2	3	4	5	6	7	8
Peso w_i	4	15	7	9	8	10	9	11
Valor p_i	2	2	3	4	6	5	8	7

- $s_0 = (1, 0, 0, 1, 0, 1, 1, 0)$
- $\rho = \sum p_j = 37$
- $f(s_0) = 19$

Fundamentação

Exemplo:

Item i	1	2	3	4	5	6	7	8
Peso w_i	4	15	7	9	8	10	9	11
Valor p_i	2	2	3	4	6	5	8	7

$s^0 = (10010110)$

$f(s^0) = 19$

# viz.	$s' \in N(s^0)$	peso(s')	$f(s')$
1	(0 0 0 1 0 1 1 0)	28	**17**
2	(1 1 0 1 0 1 1 0)	47	-534
3	(1 0 1 1 0 1 1 0)	39	-237
4	(1 0 0 0 1 1 1 0)	23	15
5	(1 0 0 1 1 1 1 0)	40	-271
6	(1 0 0 1 0 0 1 0)	22	14
7	(1 0 0 1 0 1 0 0)	23	11
8	(1 0 0 1 0 1 1 1)	43	-381

- Esta solução é um ótimo local, pois não há uma solução vizinha com melhor função de avaliação.
- **Ideia:** Mover para o melhor vizinho, ainda que de piora.

Fundamentação

Exemplo:

Item i	1	2	3	4	5	6	7	8
Peso w_i	4	15	7	9	8	10	9	11
Valor p_i	2	2	3	4	6	5	8	7

$s^0 = (10010110)$

$f(s^0) = 19$

# viz.	$s' \in N(s^0)$	peso(s')	$f(s')$
1	(0 0 0 1 0 1 1 0)	28	**17**
2	(1 1 0 1 0 1 1 0)	47	-534
3	(1 0 1 1 0 1 1 0)	39	-237
4	(1 0 0 0 1 1 1 0)	23	15
5	(1 0 0 1 1 1 1 0)	40	-271
6	(1 0 0 1 0 0 1 0)	22	14
7	(1 0 0 1 0 1 0 0)	23	11
8	(1 0 0 1 0 1 1 1)	43	-381

- Esta solução é um ótimo local, pois não há uma solução vizinha com melhor função de avaliação.
- **Ideia:** Mover para o melhor vizinho, ainda que de piora.

Fundamentação

Exemplo:

Item i	1	2	3	4	5	6	7	8
Peso w_i	4	15	7	9	8	10	9	11
Valor p_i	2	2	3	4	6	5	8	7

$s^1 = (00010110)$

$f(s^1) = 17$

# viz.	$s' \in N(s^0)$	peso(s')	$f(s')$
1	(1 0 0 1 0 1 1 0)	32	**19**
2	(0 1 0 1 0 1 1 0)	43	-388
3	(0 0 1 1 0 1 1 0)	35	-91
4	(0 0 0 0 0 1 1 0)	19	13
5	(0 0 0 1 1 1 1 0)	36	-125
6	(0 0 0 1 0 0 1 0)	18	12
7	(0 0 0 1 0 1 0 0)	19	9
8	(0 0 0 1 0 1 1 1)	39	-235

- Observe que o vizinho #1, isto é, s^0, tem a melhor avaliação na vizinhança de s^1.
- Mas nesse caso, mover para o melhor vizinho, retorna-se a uma solução já gerada anteriormente!
- **Ideia:** Criar uma Lista T das soluções já geradas (**Lista Tabu**).

Fundamentação

Exemplo:

Item i	1	2	3	4	5	6	7	8
Peso w_i	4	15	7	9	8	10	9	11
Valor p_i	2	2	3	4	6	5	8	7

$$s^1 = (00010110)$$

$$f(s^1) = 17$$

# viz.	$s' \in N(s^0)$	peso(s')	$f(s')$
1	(1 0 0 1 0 1 1 0)	32	**19**
2	(0 1 0 1 0 1 1 0)	43	-388
3	(0 0 1 1 0 1 1 0)	35	-91
4	(0 0 0 0 0 1 1 0)	19	13
5	(0 0 0 1 1 1 1 0)	36	-125
6	(0 0 0 1 0 0 1 0)	18	12
7	(0 0 0 1 0 1 0 0)	19	9
8	(0 0 0 1 0 1 1 1)	39	-235

- Observe que o vizinho #1, isto é, s^0, tem a melhor avaliação na vizinhança de s^1.
- Mas nesse caso, mover para o melhor vizinho, retorna-se a uma solução já gerada anteriormente!
- **Ideia:** Criar uma Lista T das soluções já geradas (**Lista Tabu**).

Fundamentação

Exemplo: Lista Tabu

- É inviável armazenar uma lista de **todas** as soluções geradas.
- Então para nossa *Lista Tabu*, ela pode possuir um tamanho $|T|$ e podemos manter um **histórico dos passos feitos**!

Lista Tabu para o Problema da Mochila

- Atributo: posição de um item
- Regra de ativação (proibição): impedir a inversão do valor do bit da posição modificada.
- Representação da Lista Tabu: T = <posição modificada>
- Seja $|T| = 2$

Fundamentação

Exemplo: Lista Tabu

- É inviável armazenar uma lista de **todas** as soluções geradas.
- Então para nossa *Lista Tabu*, ela pode possuir um tamanho $|T|$ e podemos manter um **histórico dos passos feitos**!

Lista Tabu para o Problema da Mochila

- Atributo: posição de um item
- Regra de ativação (proibição): impedir a inversão do valor do bit da posição modificada.
- Representação da Lista Tabu: T = <posição modificada>
- Seja $|T| = 2$

Fundamentação

Exemplo:

Item i	1	2	3	4	5	6	7	8
Peso w_i	4	15	7	9	8	10	9	11
Valor p_i	2	2	3	4	6	5	8	7

$s^0 = (10010110)$

$f(s^0) = 19$

# viz.	$s' \in N(s^0)$	peso(s')	$f(s')$
1	(0 0 0 1 0 1 1 0)	28	**17**
2	(1 1 0 1 0 1 1 0)	47	-534
3	(1 0 1 1 0 1 1 0)	39	-237
4	(1 0 0 0 1 1 1 0)	23	15
5	(1 0 0 1 1 1 1 0)	40	-271
6	(1 0 0 1 0 0 1 0)	22	14
7	(1 0 0 1 0 1 0 0)	23	11
8	(1 0 0 1 0 1 1 1)	43	-381

- O melhor vizinho de s^0 é o vizinho #1, que teve o bit da posição #1 alterado.
- O atributo #1 é, então, adicionado à *lista tabu*, passando a estar ***tabu-ativo***.
- $T = \{< 1 >\}$

Fundamentação

Exemplo:

Item i	1	2	3	4	5	6	7	8
Peso w_i	4	15	7	9	8	10	9	11
Valor p_i	2	2	3	4	6	5	8	7

$s^0 = (10010110)$

$f(s^0) = 19$

# viz.	$s' \in N(s^0)$	peso(s')	$f(s')$
1	(0 0 0 1 0 1 1 0)	28	**17**
2	(1 1 0 1 0 1 1 0)	47	-534
3	(1 0 1 1 0 1 1 0)	39	-237
4	(1 0 0 0 1 1 1 0)	23	15
5	(1 0 0 1 1 1 1 0)	40	-271
6	(1 0 0 1 0 0 1 0)	22	14
7	(1 0 0 1 0 1 0 0)	23	11
8	(1 0 0 1 0 1 1 1)	43	-381

- O melhor vizinho de s^0 é o vizinho #1, que teve o bit da posição #1 alterado.
- O atributo #1 é, então, adicionado à *lista tabu*, passando a estar ***tabu-ativo***.
- $T = \{< 1 >\}$

Fundamentação

Exemplo:

Item i	1	2	3	4	5	6	7	8
Peso w_i	4	15	7	9	8	10	9	11
Valor p_i	2	2	3	4	6	5	8	7

$s^1 = (00010110)$

$f(s^1) = 17$

# viz.	$s' \in N(s^1)$	peso(s')	$f(s')$
Tabu: 1	(1 0 0 1 0 1 1 0)	32	**19**
2	(0 1 0 1 0 1 1 0)	43	-388
3	(0 0 1 1 0 1 1 0)	35	-91
4	(0 0 0 0 0 1 1 0)	19	13
5	(0 0 0 1 1 1 1 0)	36	-125
6	(0 0 0 1 0 0 1 0)	18	12
7	(0 0 0 1 0 1 0 0)	19	9
8	(0 0 0 1 0 1 1 1)	39	-235

- Mover para o melhor vizinho da solução s^0, isto é, para s^1.
- Nesta nova solução, seu vizinho #1 é tabu porque ele tem um atributo **tabuativo** (a posição #1 está na lista tabu).
- $T = \{< 1 >\}$

Fundamentação

Exemplo:

Item i	1	2	3	4	5	6	7	8
Peso w_i	4	15	7	9	8	10	9	11
Valor p_i	2	2	3	4	6	5	8	7

$s^1 = (00010110)$

$f(s^1) = 17$

# viz.	$s' \in N(s^1)$	peso(s')	$f(s')$
Tabu: 1	(1 0 0 1 0 1 1 0)	32	**19**
2	(0 1 0 1 0 1 1 0)	43	-388
3	(0 0 1 1 0 1 1 0)	35	-91
4	(0 0 0 0 0 1 1 0)	19	13
5	(0 0 0 1 1 1 1 0)	36	-125
6	(0 0 0 1 0 0 1 0)	18	12
7	(0 0 0 1 0 1 0 0)	19	9
8	(0 0 0 1 0 1 1 1)	39	-235

- Mover para o melhor vizinho da solução s^0, isto é, para s^1.
- Nesta nova solução, seu vizinho #1 é tabu porque ele tem um atributo **tabuativo** (a posição #1 está na lista tabu).
- $T = \{< 1 >\}$

Fundamentação

Exemplo:

Item i	1	2	3	4	5	6	7	8
Peso w_i	4	15	7	9	8	10	9	11
Valor p_i	2	2	3	4	6	5	8	7

$s^1 = (00010110)$

$f(s^1) = 17$

# viz.	$s' \in N(s^1)$	peso(s')	$f(s')$
Tabu: 1	(1 0 0 1 0 1 1 0)	32	**19**
2	(0 1 0 1 0 1 1 0)	43	-388
3	(0 0 1 1 0 1 1 0)	35	-91
4	(0 0 0 0 0 1 1 0)	19	13
5	(0 0 0 1 1 1 1 0)	36	-125
6	(0 0 0 1 0 0 1 0)	18	12
7	(0 0 0 1 0 1 0 0)	19	9
8	(0 0 0 1 0 1 1 1)	39	-235

- Mover para o melhor vizinho da solução s^0, isto é, para s^1.
- Nesta nova solução, seu vizinho #1 é tabu porque ele tem um atributo **tabu-ativo** (a posição #1 está na lista tabu).
- $T = \{< 1 >\}$

Fundamentação

Exemplo:

Item i	1	2	3	4	5	6	7	8
Peso w_i	4	15	7	9	8	10	9	11
Valor p_i	2	2	3	4	6	5	8	7

$s^1 = (00010110)$

$f(s^1) = 17$

# viz.	$s' \in N(s^1)$	peso(s')	$f(s')$
Tabu: 1	(1 0 0 1 0 1 1 0)	32	19
2	(0 1 0 1 0 1 1 0)	43	-388
3	(0 0 1 1 0 1 1 0)	35	-91
4	(0 0 0 0 0 1 1 0)	19	**13**
5	(0 0 0 1 1 1 1 0)	36	-125
6	(0 0 0 1 0 0 1 0)	18	12
7	(0 0 0 1 0 1 0 0)	19	9
8	(0 0 0 1 0 1 1 1)	39	-235

- O melhor vizinho não-tabu da solução atual s^1 é o vizinho #4, no qual a posição #4 foi modificada, então esse atributo é adicionado à lista tabu.
- $T = T \cup < 4 >$
- $T = \{< 1 >, < 4 >\}$

Fundamentação

Exemplo:

Item i	1	2	3	4	5	6	7	8
Peso w_i	4	15	7	9	8	10	9	11
Valor p_i	2	2	3	4	6	5	8	7

$s^1 = (00010110)$

$f(s^1) = 17$

# viz.	$s' \in N(s^1)$	peso(s')	$f(s')$
Tabu: 1	(1 0 0 1 0 1 1 0)	32	19
2	(0 1 0 1 0 1 1 0)	43	-388
3	(0 0 1 1 0 1 1 0)	35	-91
4	(0 0 0 0 0 1 1 0)	19	**13**
5	(0 0 0 1 1 1 1 0)	36	-125
6	(0 0 0 1 0 0 1 0)	18	12
7	(0 0 0 1 0 1 0 0)	19	9
8	(0 0 0 1 0 1 1 1)	39	-235

- O melhor vizinho não-tabu da solução atual s^1 é o vizinho #4, no qual a posição #4 foi modificada, então esse atributo é adicionado à lista tabu.
- $T = T \cup < 4 >$
- $T = \{< 1 >, < 4 >\}$

Fundamentação

Exemplo:

Item i	1	2	3	4	5	6	7	8
Peso w_i	4	15	7	9	8	10	9	11
Valor p_i	2	2	3	4	6	5	8	7

$s^2 = (00000110)$

$f(s^2) = 13$

# viz.	$s' \in N(s^2)$	peso(s')	$f(s')$
Tabu: 1	(1 0 0 0 0 1 1 0)	23	15
2	(0 1 0 0 0 1 1 0)	34	-59
3	(0 0 1 0 0 1 1 0)	26	16
Tabu: 4	(0 0 0 1 0 1 1 0)	28	17
5	(0 0 0 0 1 1 1 0)	27	19
6	(0 0 0 0 0 0 1 0)	9	8
7	(0 0 0 0 0 1 0 0)	10	5
8	(0 0 0 0 0 1 1 1)	30	**20**

- Mover para o melhor vizinho não-tabu da solução s^1, isto é, para s^2.
- Os vizinhos #1 e #4 de s^2 são tabus porque eles contêm atributos *tabu-ativos* (posições 1 e 4).
- $T = \{< 1 >, < 4 >\}$

Fundamentação

Exemplo:

Item i	1	2	3	4	5	6	7	8
Peso w_i	4	15	7	9	8	10	9	11
Valor p_i	2	2	3	4	6	5	8	7

$s^2 = (00000110)$

$f(s^2) = 13$

# viz.	$s' \in N(s^2)$	peso(s')	$f(s')$
Tabu: 1	(1 0 0 0 0 1 1 0)	23	15
2	(0 1 0 0 0 1 1 0)	34	-59
3	(0 0 1 0 0 1 1 0)	26	16
Tabu: 4	(0 0 0 1 0 1 1 0)	28	17
5	(0 0 0 0 1 1 1 0)	27	19
6	(0 0 0 0 0 0 1 0)	9	8
7	(0 0 0 0 0 1 0 0)	10	5
8	(0 0 0 0 0 1 1 1)	30	**20**

- Mover para o melhor vizinho não-tabu da solução s^1, isto é, para s^2.
- Os vizinhos #1 e #4 de s^2 são tabus porque eles contêm atributos *tabu-ativos* (posições 1 e 4).
- $T = \{< 1 >, < 4 >\}$

Fundamentação

Exemplo:

Item i	1	2	3	4	5	6	7	8
Peso w_i	4	15	7	9	8	10	9	11
Valor p_i	2	2	3	4	6	5	8	7

$s^2 = (00000110)$

$f(s^2) = 13$

# viz.	$s' \in N(s^2)$	peso(s')	$f(s')$
Tabu: 1	(1 0 0 0 0 1 1 0)	23	15
2	(0 1 0 0 0 1 1 0)	34	-59
3	(0 0 1 0 0 1 1 0)	26	16
Tabu: 4	(0 0 0 1 0 1 1 0)	28	17
5	(0 0 0 0 1 1 1 0)	27	19
6	(0 0 0 0 0 0 1 0)	9	8
7	(0 0 0 0 0 1 0 0)	10	5
8	(0 0 0 0 0 1 1 1)	30	**20**

- Melhor vizinho não-tabu da solução atual é #8 (gerado pela alteração da posição #8). Esse atributo é adicionado à lista tabu.
- Como $|T| = 2$, então pela estratégia FIFO, o atributo *tabu-ativo* 1 deixa de ser ativo e entra o atributo 8 ao final da lista tabu.
- $T = T \setminus <1> \cup <8>$
- $T = \{<4>, <8>\}$

Fundamentação

Exemplo:

Item i	1	2	3	4	5	6	7	8
Peso w_i	4	15	7	9	8	10	9	11
Valor p_i	2	2	3	4	6	5	8	7

$s^2 = (00000110)$

$f(s^2) = 13$

# viz.	$s' \in N(s^2)$	peso(s')	$f(s')$
Tabu: 1	(1 0 0 0 0 1 1 0)	23	15
2	(0 1 0 0 0 1 1 0)	34	-59
3	(0 0 1 0 0 1 1 0)	26	16
Tabu: 4	(0 0 0 1 0 1 1 0)	28	17
5	(0 0 0 0 1 1 1 0)	27	19
6	(0 0 0 0 0 0 1 0)	9	8
7	(0 0 0 0 0 1 0 0)	10	5
8	(0 0 0 0 0 1 1 1)	30	**20**

- Melhor vizinho não-tabu da solução atual é #8 (gerado pela alteração da posição #8). Esse atributo é adicionado à lista tabu.
- Como $|T| = 2$, então pela estratégia FIFO, o atributo *tabu-ativo* 1 deixa de ser ativo e entra o atributo 8 ao final da lista tabu.
- $T = T \setminus <1> \cup <8>$
- $T = \{<4>, <8>\}$

Fundamentação

Exemplo:

Item i	1	2	3	4	5	6	7	8
Peso w_i	4	15	7	9	8	10	9	11
Valor p_i	2	2	3	4	6	5	8	7

$s^3 = (00000111)$

$f(s^3) = 20$

# viz.	$s' \in N(s^3)$	peso(s')	$f(s')$
1	(1 0 0 0 0 1 1 1)	34	-52
2	(0 1 0 0 0 1 1 1)	45	-459
3	(0 0 1 0 0 1 1 1)	37	-162
Tabu: 4	(0 0 0 1 0 1 1 1)	39	-235
5	(0 0 0 0 1 1 1 1)	38	-196
6	(0 0 0 0 0 0 1 1)	20	**15**
7	(0 0 0 0 0 1 0 1)	21	12
Tabu: 8	(0 0 0 0 0 1 1 0)	19	13

- Mover para o melhor vizinho não-tabu da solução s^2, i.e., para s^3.
- Os vizinhos #4 e #8 de s^3 são tabus.
- O vizinho #6 (não-tabu) é o melhor. O atributo 6 entra e sai o 4 da lista.
- $T = T \setminus < 4 > \cup < 6 >$
- $T = \{< 8 >, < 6 >\}$

Fundamentação

Exemplo:

Item i	1	2	3	4	5	6	7	8
Peso w_i	4	15	7	9	8	10	9	11
Valor p_i	2	2	3	4	6	5	8	7

$s^3 = (00000111)$

$f(s^3) = 20$

# viz.	$s' \in N(s^3)$	peso(s')	$f(s')$
1	(1 0 0 0 0 1 1 1)	34	-52
2	(0 1 0 0 0 1 1 1)	45	-459
3	(0 0 1 0 0 1 1 1)	37	-162
Tabu: 4	(0 0 0 1 0 1 1 1)	39	-235
5	(0 0 0 0 1 1 1 1)	38	-196
6	(0 0 0 0 0 0 1 1)	20	**15**
7	(0 0 0 0 0 1 0 1)	21	12
Tabu: 8	(0 0 0 0 0 1 1 0)	19	13

- Mover para o melhor vizinho não-tabu da solução s^2, i.e., para s^3.
- Os vizinhos #4 e #8 de s^3 são tabus.
- O vizinho #6 (não-tabu) é o melhor. O atributo 6 entra e sai o 4 da lista.
- $T = T \setminus <4> \cup <6>$
- $T = \{<8>, <6>\}$

Fundamentação

Exemplo:

Item i	1	2	3	4	5	6	7	8
Peso w_i	4	15	7	9	8	10	9	11
Valor p_i	2	2	3	4	6	5	8	7

$s^4 = (00000011)$

$f(s^4) = 15$

# viz.	$s' \in N(s^4)$	peso(s')	$f(s')$
1	(1 0 0 0 0 0 1 1)	24	17
2	(0 1 0 0 0 0 1 1)	35	-94
3	(0 0 1 0 0 0 1 1)	27	18
4	(0 0 0 1 0 0 1 1)	29	19
5	(0 0 0 0 1 0 1 1)	28	**21**
Tabu: 6	(0 0 0 0 0 1 1 1)	30	20
7	(0 0 0 0 0 1 0 1)	11	7
Tabu: 8	(0 0 0 0 0 0 1 0)	9	8

- Mover para o melhor vizinho não-tabu da solução s^3, i.e., para s^4.
- Os vizinhos #6 e #8 de s^4 são tabus.
- O vizinho #5 (não-tabu) é o melhor. O atributo 5 entra e sai o 8 da lista.
- $T = T \setminus < 8 > \cup < 5 >$
- $T = \{< 6 >, < 5 >\}$

Fundamentação

Exemplo:

Item i	1	2	3	4	5	6	7	8
Peso w_i	4	15	7	9	8	10	9	11
Valor p_i	2	2	3	4	6	5	8	7

$s^5 = (00001011)$

$f(s^5) = 21$

# viz.	$s' \in N(s^5)$	peso(s')	$f(s')$
1	(1 0 0 0 1 0 1 1)	32	**23**
2	(0 1 0 0 1 0 1 1)	43	-384
3	(0 0 1 0 1 0 1 1)	35	-87
4	(0 0 0 1 1 0 1 1)	37	-160
Tabu: 5	(0 0 0 0 0 0 1 1)	20	15
Tabu: 6	(0 0 0 0 1 1 1 1)	38	-196
7	(0 0 0 0 1 0 0 1)	19	13
8	(0 0 0 0 1 0 1 0)	17	14

- Mover para o melhor vizinho não-tabu da solução s^4, i.e., para s^5.
- Os vizinhos #5 e #6 de s^5 são tabus.
- O vizinho #1 (não-tabu) é o melhor. O atributo 1 entra e sai o 6 da lista.
- $T = T \setminus <6> \cup <1>$
- $T = \{<5>, <1>\}$

Fundamentação

Exemplo:

Item i	1	2	3	4	5	6	7	8
Peso w_i	4	15	7	9	8	10	9	11
Valor p_i	2	2	3	4	6	5	8	7

$s^6 = (10001011)$

$f(s^6) = 23$

# viz.	$s' \in N(s^6)$	peso(s')	$f(s')$
Tabu: 1	(0 0 0 0 1 0 1 1)	28	21
2	(1 1 0 0 1 0 1 1)	47	-530
3	(1 0 1 0 1 0 1 1)	39	-233
4	(1 0 0 1 1 0 1 1)	41	-306
Tabu: 5	(1 0 0 0 0 0 1 1)	24	17
6	(1 0 0 0 1 1 1 1)	42	-342
7	(1 0 0 0 1 0 0 1)	23	15
8	(1 0 0 0 1 0 1 0)	21	**16**

- Mover para o melhor vizinho não-tabu da solução s^5, i.e., para s^6.
- Os vizinhos #1 e #5 de s^5 são tabus.
- O vizinho #8 (não-tabu) é o melhor. O atributo 8 entra e sai o 5 da lista.
- $T = T \setminus < 5 > \cup < 8 >$
- $T = \{< 1 >, < 8 >\}$

Fundamentação

Exemplo:

Item i	1	2	3	4	5	6	7	8
Peso w_i	4	15	7	9	8	10	9	11
Valor p_i	2	2	3	4	6	5	8	7

$s^7 = (10001010)$

$f(s^7) = 16$

# viz.	$s' \in N(s^7)$	peso(s')	$f(s')$
Tabu: 1	(0 0 0 0 1 0 1 0)	17	14
2	(1 1 0 0 1 0 1 0)	36	-130
3	(1 0 1 0 1 0 1 0)	28	19
4	(1 0 0 1 1 0 1 0)	30	20
5	(1 0 0 0 0 0 1 0)	13	10
6	(1 0 0 0 1 1 1 0)	31	**21**
7	(1 0 0 0 1 0 0 0)	12	8
Tabu: 8	(1 0 0 0 1 0 1 1)	32	21

- Mover para o melhor vizinho não-tabu da solução s^6, i.e., para s^7.
- Os vizinhos #1 e #8 de s^6 são tabus.
- O vizinho #6 (não-tabu) é o melhor. O atributo 6 entra e sai o 1 da lista.
- $T = T \setminus <1> \cup <6>$
- $T = \{<8>, <6>\}$

Fundamentação

Exemplo:

Item i	1	2	3	4	5	6	7	8
Peso w_i	4	15	7	9	8	10	9	11
Valor p_i	2	2	3	4	6	5	8	7

$s^8 = (10001110)$

$f(s^8) = 21$

# viz.	$s' \in N(s^8)$	peso(s')	$f(s')$
1	(0 0 0 0 1 1 1 0)	27	**19**
2	(1 1 0 0 1 1 1 0)	46	-495
3	(1 0 1 0 1 1 1 0)	38	-198
4	(1 0 0 1 1 1 1 0)	40	-271
5	(1 0 0 0 0 1 1 0)	23	15
Tabu: 6	(1 0 0 0 1 0 1 0)	21	16
7	(1 0 0 0 1 1 0 0)	22	13
Tabu: 8	(1 0 0 0 1 1 1 1)	42	-342

- Mover para o melhor vizinho não-tabu da solução s^7, i.e., para s^8.
- Os vizinhos #6 e #8 de s^8 são tabus.
- O vizinho #1 (não-tabu) é o melhor. O atributo 1 entra e sai o 8 da lista.
- $T = T \setminus <8> \cup <1>$
- $T = \{<6>, <1>\}$

Fundamentação

Exemplo:

Item i	1	2	3	4	5	6	7	8
Peso w_i	4	15	7	9	8	10	9	11
Valor p_i	2	2	3	4	6	5	8	7

$s^9 = (00001110)$

$f(s^9) = 19$

# viz.	$s' \in N(s^9)$	peso(s')	$f(s')$
Tabu: 1	(1 0 0 0 1 1 1 0)	31	21
2	(0 1 0 0 1 1 1 0)	42	-349
3	(0 0 1 0 1 1 1 0)	34	-52
4	(0 0 0 1 1 1 1 0)	36	-125
5	(0 0 0 0 0 1 1 0)	19	**13**
Tabu: 6	(0 0 0 0 1 0 1 0)	17	14
7	(0 0 0 0 1 1 0 0)	18	11
8	(0 0 0 0 1 1 1 1)	38	-196

- Mover para o melhor vizinho não-tabu da solução s^8, i.e., para s^9.
- Os vizinhos #1 e #6 de s^9 são tabus.
- O vizinho #5 (não-tabu) é o melhor. O atributo 5 entra e sai o 6 da lista.
- $T = T \setminus < 6 > \cup < 5 >$
- $T = \{< 1 >, < 5 >\}$

Fundamentação

Exemplo:

Item i	1	2	3	4	5	6	7	8
Peso w_i	4	15	7	9	8	10	9	11
Valor p_i	2	2	3	4	6	5	8	7

$s^{10} = (00000110)$

$f(s^{10}) = 13$

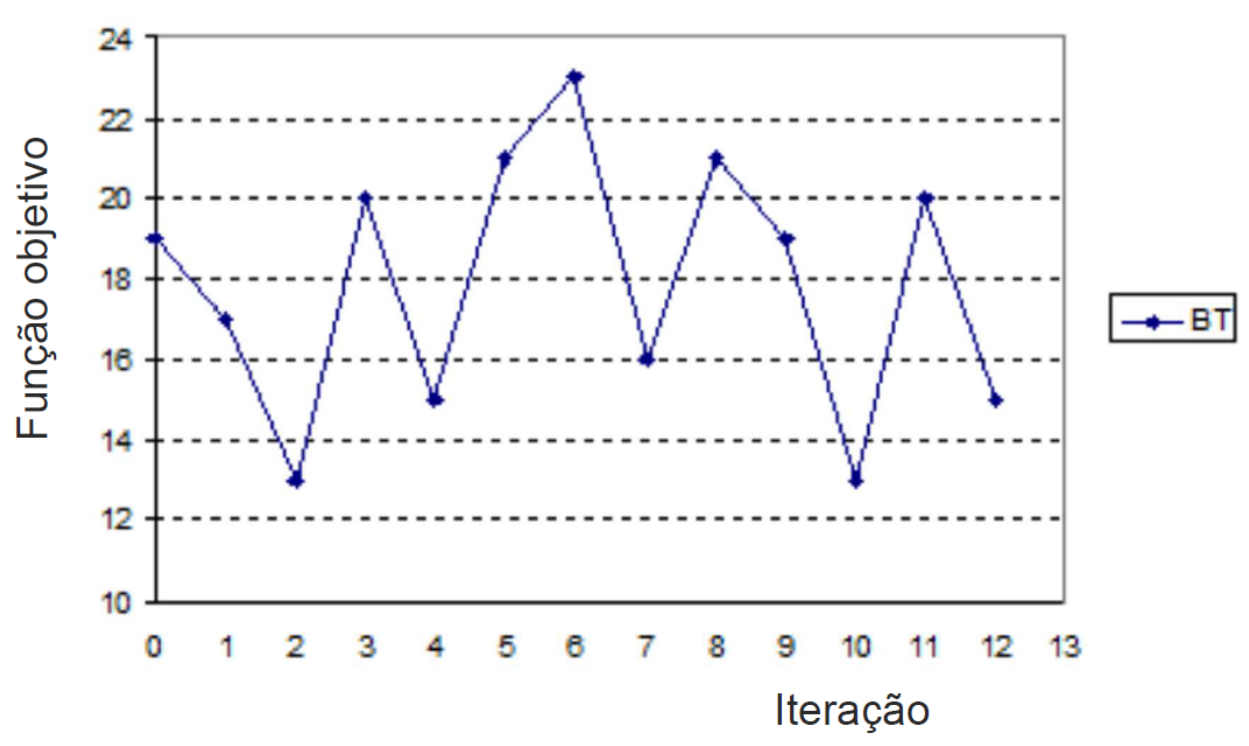
# viz.	$s' \in N(s^{10})$	peso(s')	$f(s')$
Tabu: 1	(1 0 0 0 0 1 1 0)	23	15
2	(0 1 0 0 0 1 1 0)	34	-59
3	(0 0 1 0 0 1 1 0)	26	16
4	(0 0 0 1 0 1 1 0)	28	17
Tabu: 5	(0 0 0 0 1 1 1 0)	27	19
6	(0 0 0 0 0 0 1 0)	9	8
7	(0 0 0 0 0 1 0 0)	10	5
8	(0 0 0 0 0 1 1 1)	30	**20**

- Mover para o melhor vizinho não-tabu da solução s^9, i.e., para s^10.
- Os vizinhos #1 e #5 de s^10 são tabus.
- O vizinho #8 (não-tabu) é o melhor. O atributo 8 entra e sai o 1 da lista.
- $T = T \setminus <1> \cup <8>$
- $T = \{<5>, <8>\}$

Fundamentação

Exemplo:

- Evolução do valor da função objetivo ao longo das iterações.



- Note que as iterações das soluções s^2 e s^{10} apesar de terem tabus diferentes, vão gerar as mesmas próximas soluções até os tabus.

Fundamentação

Exemplo:

- Na implementação da lista tabu, verificar se um movimento é ou não tabu pode ser dispendioso.
- No Problema da Mochila, $T = \{< j_1 >, < j_2 >, \ldots, < j_{|T|} >\}$
- Considerando o vetor de n posições e $|T|$ elementos na lista tabu:
  - Pior caso: $O(n.|T|)$ avaliações, por iteração.

Fundamentação

Exemplo - Lista Tabu com Prazo:

- Uma nova ideia consiste em substituir a lista T por um vetor de n posições, em que cada posição j armazene a iteração até a qual o atributo está **tabu-ativo**.
- T: vetor de prazo tabu (*tabu tenure*).
- Na primeira iteração do PM ($iter = 1$), considerando $DuracaoTabu = 2$, temos:
  - $T = \{< 1 >\}$ é substituida por $T = \{3, 0, 0, 0, 0, 0, 0, 0\}$
  - pois, $iter + DuracaoTabu = 1 + 2 = 3$
- **Significado:** O movimento de inverter o bit da primeira posição está tabu-ativo até a iteração 3. Após a terceira iteração, o movimento deixa de ser tabu.

Fundamentação

Exemplo - Lista Tabu com Prazo:

- Uma nova ideia consiste em substituir a lista T por um vetor de n posições, em que cada posição j armazene a iteração até a qual o atributo está **tabu-ativo**.
- T: vetor de prazo tabu (*tabu tenure*).
- Na primeira iteração do PM ($iter = 1$), considerando $DuracaoTabu = 2$, temos:
  - $T = \{< 1 >\}$ é substituida por $T = \{3, 0, 0, 0, 0, 0, 0, 0\}$
  - pois, $iter + DuracaoTabu = 1 + 2 = 3$
- **Significado:** O movimento de inverter o bit da primeira posição está tabu-ativo até a iteração 3. Após a terceira iteração, o movimento deixa de ser tabu.

Fundamentação

Exemplo - Lista Tabu com Prazo:

- Com um vetor T de prazo tabu, a verificação se um movimento é tabu ou não é bastante simples.
- Seja *iter* a iteração atual. Então o movimento de inverter o valor do bit da posição j é tabu se tivermos: $T(j) \geq iter$
- Ex.: Dado $T = \{3, 0, 8, 5, 9, 10, 7, 11\}$ e $iter = 10$, então são tabus os movimentos envolvendo as posições 6 e 8.
- Complexidade de verificação se um movimento está ou não na lista tabu é: $O(1)$. Logo, a complexidade de verificação de todos vizinhos é $O(n).O(1) = O(n)$.

Fundamentação

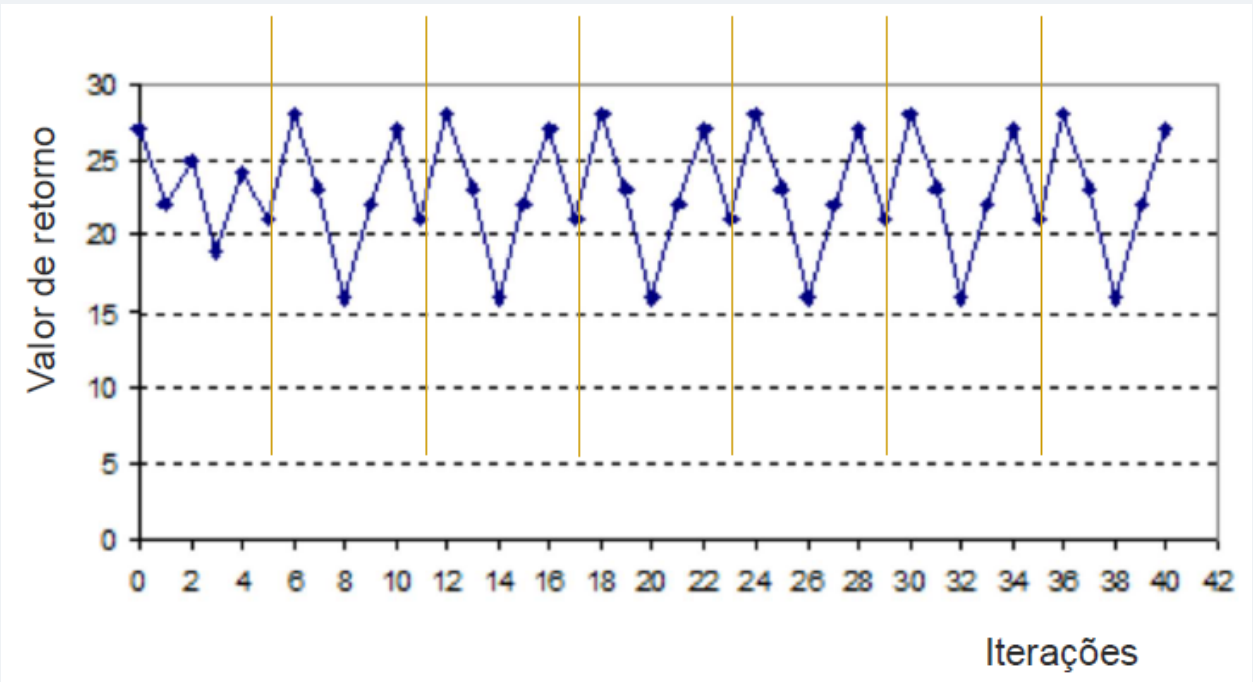
Exemplo - Lista Tabu com Prazo:

- A *DuracaoTabu* indica a quantidade máxima de iterações que cada movimento permanece tabu para impedir o retorno a uma solução já visitada.
- A *DuracaoTabu* depende:
  - Do tipo de atributo considerado: Regras de ativação mais restritivas têm duração menor.
  - Do tamanho da instância: Quanto maior o tamanho, maior a lista (crescimento não-linear).
  - Do estágio da busca: Pode-se usar uma duração dinâmica que muda durante a busca, evitando ciclagens.

Fundamentação

Exemplo - Lista Tabu com Prazo:

- Uma *DuracaoTabu* muito pequena pode causar ciclagem (repetições).



- Uma *DuracaoTabu* muito grande pode causar deterioração.

Fundamentação

Exemplo - Lista Tabu com Prazo:

- A duração tabu também depende do atributo e da regra de proibição.
- No problema da mochila, considerando que:
  - A cada iteração uma posição é tornada tabu-ativa;
  - Ao final de $n-1$ iterações há apenas um movimento possível.
  - Assim, o limitante superior para a duração tabu de um movimento é $n-1$.

Fundamentação

Exemplo - Lista Tabu com Prazo:

- Uma versão alternativa para a Lista Tabu com Prazo, seria a de considerar a *DuracaoTabu* para o atributo, e a cada nova iteração reduzindo-o uma unidade até chegar em 0.
- Assim, somente as casas que valem 0, seriam os vizinhos não-tabu.

Lista Tabu com Prazo (Decrescente)

- Na primeira iteração do PM ($iter = 1$), considerando $DuracaoTabu = 2$, temos:
  - $T = \{< 1 >\}$ é substituída por $T = \{2, 0, 0, 0, 0, 0, 0, 0\}$ pois, foi trocado o bit 1;
  - Na iteração seguinte, troca-se o bit 4, logo $T = \{1, 0, 0, 2, 0, 0, 0, 0\}$
  - Na iteração seguinte, troca-se o bit 8, logo $T = \{0, 0, 0, 1, 0, 0, 0, 2\}$
  - E assim por diante..
- **Significado:** O movimento de inverter o bit, deixa a posição tabu-ativa por 2 iterações. Duas iterações depois, o movimento deixa de ser tabu.

Fundamentação

Exemplo: Critérios de Aspiração

- Mas essa proibição tabu às vezes é restritiva demais!
- Suponha, que por exemplo, que uma solução tabu produza uma solução melhor do que a $s_b\,est$ (i.e., melhor do que a melhor já vista)
- Nesse caso podemos esquecer que era tabu, aplicando um critério de aspiração (ou função de aspiração).

Critério de Aspiração por objetivo regional

- Sobrepõe a Regra de ativação (proibição)
- Aceita o vizinho, mesmo sendo gerado por movimento tabu, quando se produz uma solução melhor que a melhor já vista.

Fundamentação

Exemplo: Critérios de Aspiração

- Existem outros critérios de aspiração, como por exemplo o *critério de aspiração por default*.
- Se todos os movimentos possíveis são tabus e não é possível aplicar outro critério de aspiração, então o movimento mais antigo **perde** sua condição tabu.

Critério de Aspiração por default

- Implementação baseada em tempo de permanência na lista, como o do Problema da Mochila.
  - Se $T(i) \geq$ iter, para todo i, em que iter representa a iteração atual,
    - então a inversão do bit da posição k tal que:

$$T(k) = \min\{T(i) \text{ para todo } i\}$$

pode ser realizada.

  - Em outras palavras, se todos os movimentos estão proibidos, então o movimento tabu há mais tempo na lista é realizado.

Agenda

Busca Tabu

Algorithm 3: Busca Tabu Básica

1 $s \leftarrow s_0$ (Solucao inicial)
2 $s_{best} \leftarrow s_0$
3 $iter \leftarrow 0$
4 $T \leftarrow \emptyset$ (Lista Tabu)
5 **while** *condicao parada não satisfeita* **do**
6 $\quad iter \leftarrow iter + 1$
7 $\quad$ Seja $s' \leftarrow s \oplus m$ o melhor elemento de $V \subseteq N(s)$ tal que o movimento m não seja tabu ($m \notin T$) **OU** s' atenda o critério de aspiração ($f(s') > f(s_{best})$)
8 $\quad$ Atualiza a lista tabu T
9 $\quad s \leftarrow s'$
10 $\quad$ **if** $f(s) > f(s_{best})$ **then**
11 $\quad\quad s_{best} \leftarrow s$
12 $\quad$ **end**
13 $\quad$ **if** *todos os movimentos estão proibidos* **then**
14 $\quad\quad$ Permitir o movimento tabu há mais tempo na lista
15 $\quad$ **end**
16 **end**
17 **return** s_{best}

Busca Tabu

Considerações

- As **condições de parada** para a Busca Tabu mais comuns podem ser:
  - Número máximo de iterações.
  - Número de iterações sem melhorias.
  - Tempo limite.
  - Dentre outras.
- **Critério de aspiração** e *Soluções Inviáveis*:
  - Note também que o critério de aspiração da busca tabu pode permitir a exploração de soluções inviáveis.
  - Essa característica de escapar de ótimos locais permite explorar regiões mais amplas do espaço de soluções.
  - No entanto, esse fato está relacionado com penalizar soluções inviáveis, direcionando a busca de volta para a viabilidade.
  - Assim, mesmo que a busca explore soluções inviáveis temporariamente, o objetivo final é encontrar uma solução viável e de alta qualidade.

Considerações de Memória

Referência

- Seção 2.5 (Talbi, El-Ghazali; Metaheuristics: From Design to Implementation, Wiley Publishing, 2009.).

Tipos de memória na Busca Tabu

- Memória de curto prazo (tabu)
- Memória de médio prazo (intensificação)
- Memória de longo prazo (diversificação)

Exemplo TSP

Exemplo TSP

- Considere o movimento swap: trocar duas cidades de lugar na rota
- Para n = 8 cidades, são 28 vizinhos (pares de cidades a trocar)
- Após trocar cidades nas i e j como marcar o movimento tabu?
- Uma ideia seria de guardar em uma matriz de posições, a *DuracaoTabu* das ultimas trocas entre posições.

Lista Tabu com Prazo (Decrescente) - TSP

- Considerando $DuracaoTabu = 5$.
- A Lista Tabu (Matriz Tabu) mostra que nas ultimas 5 iterações foram trocadas as cidades das posições:
  (1, 4), (5, 8), (4, 5), (3, 7), (2, 6).
- A cada nova iteração, a matriz é atualizada, reduzindo 1 unidade das trocas já feitas.

	2	3	4	5	6	7	8
1	0	0	1	0	0	0	0
2		0	0	0	5	0	0
3			0	0	0	4	0
4				3	0	0	0
5					0	0	2
6						0	0
7							0

Exemplo TSP

Exemplo TSP

- Outras ideias de memórias também são apresentadas nos livros.

Memórias TSP

- **Memória de curto prazo**
  - Lista de arcos que não devem ser considerados.
- **Memória de médio e longo prazo**
  - Lista de arcos considerados nas últimas k melhores (piores) soluções.
  - Encorajar (desencorajar) sua seleção futura de acordo com a frequência que aparece em soluções elites e qualidade dessas soluções.

Aplicações

Considerações finais

- A **Busca Tabu** tem sido usada com sucesso em muitos problemas
- Seus **componentes**, no entanto, são específicos de cada problema:
  - Representação da lista tabu
  - Tamanho da lista tabu (*pode ser muito sensível a este parâmetro*)
  - Uso de memória de médio e longo prazo
  - ...
  - Assim, ela é mais difícil de se aplicar que Busca Local e *Simulated Annealing*.

Exercício

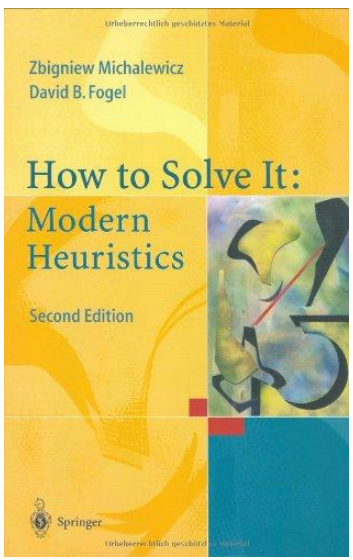
1. Faça uma *Busca Tabu* para o problema da mochila 0/1 baseado no Algoritmo 3 da página 43:
   (a) Gere uma **solução inicial gulosa**.
   (b) Usa a vizinhança ***flip***.
   (c) Baseado na aula, defina um critério de parada iterações sem melhora.
   (d) A **regra de ativação** como o ultimo movimento feito.
   (e) **Lista Tabu com Prazo (Decrescente)**.
   (f) **Critério de Aspiração por objetivo regional** na vizinhança e **Critério de Aspiração por default** caso proiba todos movimentos.

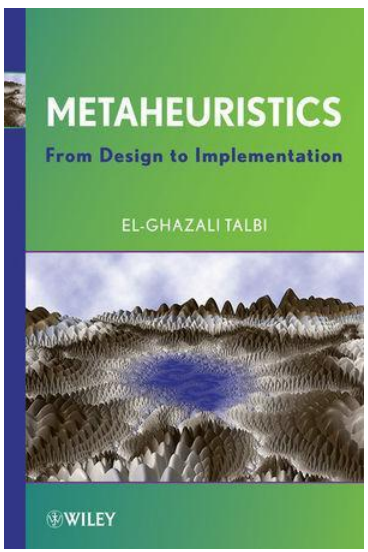
   Obs: Podemos usar a mesmas instâncias utilizadas na busca local anteriormente.

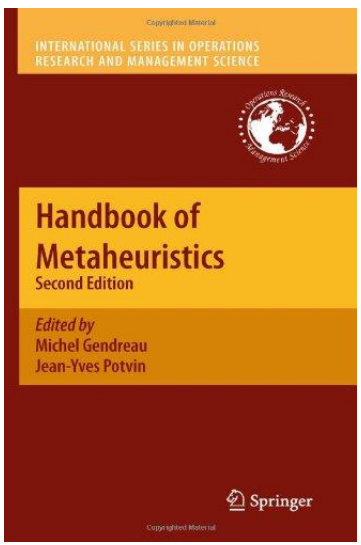
Bibliografias

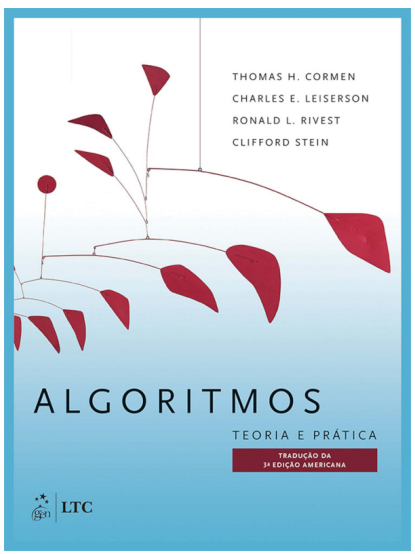
Bibliografia Básica

1. MICHLEWICZ, Zbigniew; FOGEL, David B. How to solve it: modern heuristics. 2nd. ed. Berlin: Springer c2010 554 p. ISBN 9783642061349.
2. Talbi, El-Ghazali; Metaheuristics: From Design to Implementation, Wiley Publishing, 2009.
3. GENDREAU, Michel. Handbook of metaheuristics. 2.ed. New York: Springer 2010 648 p. (International series in operations research & management science ; 146).
4. T. Cormen, C. Leiserson, R. Rivest, C. Stein, Introduction to Algorithms, The MIT Press, 3rd edition, 2009 (**Pergamum**).









Bibliografias

Bibliografia Complementar

1. GLOVER, Fred; KOCHENBERGER, Gary A. (ed.). Handbook of metaheuristics. Boston: Kluwer, 2003. 556 p. (International series in operations research & management science ; 57).
2. BLUM, Christian Et Al. Hybrid metaheuristics: an emerging approach to optimization. Berlin: Springer 2008 289 p. (Studies in Computational intelligence; 114).
3. DOERNER, Karl F. (ed.) Et Al. Metaheuristics: progress in complex systems optimization. New York: Springer 2007 408 p. (Operations research / computer science interfaces series).
4. GLOVER, Fred; LAGUNA, Manuel. Tabu search. Boston: Kluwer Academic, 1997. 382 p.
5. AARTS, Emile. Local search in combinatorial optimization. Princeton: Princeton University Press, 2003 512 p.
6. Gaspar-Cunha, A.; Takahashi, R.; Antunes, C.H.; Manual de Computação Evolutiva e Metaheurística; Belo Horizonte: Editora UFMG; Coimbra: Imprensa da Universidade de Coimbra; 2013.