Tratamento de problemas $\mathcal{NP}$-difíceis: Heurísticas Construtivas

Cid C. Souza
Eduardo C. Xavier

Instituto de Computação/Unicamp

18 de abril de 2011

Introdução

- Heurísticas são algoritmos que geram soluções viáveis para quais *não se pode dar garantias de qualidade*. Ou seja, não se sabe o quão *distante* a solução gerada estará de uma solução ótima (5% ?, 10% ?, 50% ?, 100% ?, ...).
- Tipos de heurísticas:
  - *construtivas*: normalmente adotam estratégias *gulosas* para construir as soluções. Tipicamente são aplicadas a problemas onde é fácil obter uma solução viável.
  - *de busca local*: partem de uma solução inicial e, através de transformações bem definidas, visitam outras soluções até atingir um *critério de parada* pré-definido.
  - *meta-heurística:* modelo genérico de uma heurística para problemas em geral.

Heurísticas Construtivas (TSP)

- Exemplo 1: TSP em um grafo não orientado completo.

```
Vizinho-Mais-Próximo(n, d)      (* d: matriz de distâncias *)
    Para i = 1 até n faça visitado[i] ← Falso ;
    visitado[1] ← Verdadeiro;
    ciclo ← {}, comp ← 0 e k ← 1;
    Para i = 1 até n − 1 faça
        j* ← argmin{d[k, j] : visitado[j] = Falso};
        visitado[j*] ← Verdadeiro ;
        ciclo ← ciclo ∪ {(k, j*)};      comp ← comp + d[k, j*];
        k ← j*;
    fim-para
    ciclo ← ciclo ∪ {(k, 1)};      comp ← comp + d[k, 1];
Retorne  comp.
```

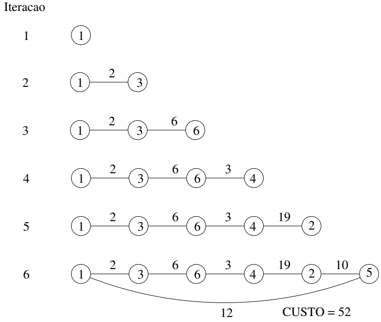
- Complexidade: $O(n^2)$

Heurísticas Construtivas (TSP)

Aplicação das heurísticas para o TSP:

$$d = \begin{bmatrix} - & 9 & 2 & 8 & 12 & 11 \\ 9 & - & 7 & 19 & 10 & 32 \\ 2 & 7 & - & 29 & 18 & 6 \\ 8 & 19 & 29 & - & 24 & 3 \\ 12 & 10 & 18 & 24 & - & 19 \\ 11 & 32 & 6 & 3 & 19 & - \end{bmatrix}$$

`Vizinho-Mais-Próximo`



Heurísticas Construtivas (TSP)

- Exemplo 2: heurística para o TSP $\equiv$ algoritmo de Kruskal para AGM.

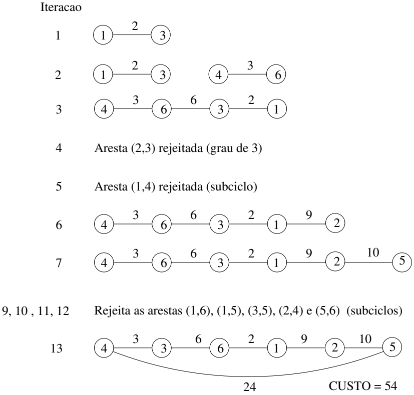
TSP-Guloso(n, d) (* d: matriz de distâncias *)
$\mathcal{L} \leftarrow$ lista das arestas ordenadas crecentemente pelo valor de d;
Para $i = 1$ **até** n **faça** grau$[i] \leftarrow 0$; componente$[i] = i$ **fim-para**
$k \leftarrow 0$; ciclo $\leftarrow \{\}$; comp $\leftarrow 0$;
Enquanto $k \neq n$ **faça**
$(u, v) \leftarrow$ Remove-primeiro$(\mathcal{L})$;
Se (grau$[u] \leq 1$ **e** grau$[v] \leq 1$ **e** componente$(u) \neq$ componente(v))
ou (grau$[u] =$ grau$[v] = 1$ **e** $k = n - 1$) **então**
ciclo $\leftarrow$ ciclo $\cup \{(u, v)\}$; comp $\leftarrow$ comp $+ d[u, v]$;
Unir-componentes(u, v);
grau$[u]++$; grau$[v]++$; $k++$;
fim-se
fim-enquanto
Retorne comp.

- Complexidade: $O(n^2 \log n)$ (usar *compressão de caminhos* para *união de conjuntos disjuntos*).

Heurísticas Construtivas (TSP)

`TSP-GULOSO` Aplicação das heurísticas para o TSP:

$$d = \begin{bmatrix} - & 9 & 2 & 8 & 12 & 11 \\ 9 & - & 7 & 19 & 10 & 32 \\ 2 & 7 & - & 29 & 18 & 6 \\ 8 & 19 & 29 & - & 24 & 3 \\ 12 & 10 & 18 & 24 & - & 19 \\ 11 & 32 & 6 & 3 & 19 & - \end{bmatrix}$$



Heurísticas Construtivas (Mochila)

- Exemplo 2: Problema da Mochila.

```
Mochila-guloso(c, w, W)
    Ordenar itens segundo a razão c_i/w_i;
    (* assuma que c_1/w_1 ≥ c_2/w_2 ≥ ... ≥ c_n/w_n *)

    W̄ ← W;      S ← {};
    Para i = 1 até n faça
        Se w_i ≤ W̄ então
            W̄ ← W̄ − w_i;
            S ← S ∪ {i};
        fim-se
    fim-para
Retorne S.
```

- Complexidade: $O(n \log n)$.

Heurísticas Construtivas (Mochila)

- Aplicação da heurística `Mochila-guloso`.

$$
\begin{array}{ll}
\text{maximize} & 8x_1 + 16x_2 + 20x_3 + 12x_4 + 6x_5 + 10x_6 + 4x_7 \\
\text{Sujeito a} & 3x_1 + 7x_2 + 9x_3 + 6x_4 + 3x_5 + 5x_6 + 2x_7 \leq 17, \\
 & x \in \mathbb{B}^7.
\end{array}
$$

Observação: $\frac{8}{3} \geq \frac{16}{7} \geq \frac{20}{9} \geq \frac{12}{6} \geq \frac{6}{3} \geq \frac{10}{5} \geq \frac{4}{2}$

- Solução gulosa: $S = \{1, 2, 4\}$, custo $= 36$.
- Solução ótima: $S = \{1, 2, 6, 7\}$, custo $= 38$.

Heurísticas Construtivas

- Soluções gulosas podem ser *arbitrariamente ruins* !
- `Mochila-guloso` é arbitrariamente ruim.
- Instância: W, $c_1 = 2$, $w_1 = 1$ e, $c_2 = W$, $w_2 = W$.
  Observação: $\frac{c_1}{w_1} = 2 \geq \frac{c_2}{w_2} = 1$.
- Solução gulosa: $S = \{1\}$, custo $= 2$.
- Solução ótima: $S = \{2\}$, custo $= W$.
- $\lim_{W \to \infty} \frac{2}{W} = 0$.

Ou seja, aumentando o valor de W nesta instância, a solução gulosa pode se afastar tanto quanto eu quiser da solução ótima !

Heurísticas Construtivas

- `Vizinho-Mais-Próximo` para o TSP é arbitrariamente ruim.
- Instância: matriz simétrica de distâncias d onde, para $i < j$, tem-se:

$$d[i,j] = \begin{cases} n^2, & \text{se } i = n-1 \text{ e } j = n, \\ 1, & \text{se } j = i+1, \\ 2, & \text{caso contrário.} \end{cases}$$

- Solução gulosa: ciclo $= \{1, 2, \ldots, n-1, n, 1\}$ e comp $= n^2 + n$.
- Solução ótima: ciclo $= \{1, 2, \ldots, n-3, n, n-2, n-1, 1\}$ e comp $= n + 3$.
- $\lim_{n\to\infty} \frac{n+3}{n^2+n} = 0$.

Novamente, aumentando o valor de n nesta instância, a solução gulosa pode se afastar tanto quanto eu quiser da solução ótima !

Heurísticas Construtivas

- Heurísticas construtivas são simples e rápidas.
- Porém seus resultados podem ser muito ruins:
  - O espaço de soluções possíveis é muito grande
  - Uma única solução deste espaço de soluções é considerado.
- Heurísticas construtivas podem ser usadas para gerar soluções iniciais para outros métodos heurísticos.