MC538/MC548 - 2ª Lista de Exercícios

Anamaria Gomide e Jorge Stolfi

1º Semestre de 2003

1. Seja $G = (V, E)$ um grafo completo não orientado de 5 vértices com comprimento nas arestas dados pela matriz (simétrica) D abaixo. Nesta matriz, o elemento d_{ij} indica o comprimento da aresta (i, j) o qual representa a distância entre cidades cujos rótulos são i e j.

$$D = \begin{pmatrix} - & 8 & 12 & 14 & 10 \\ - & - & 7 & 4 & 16 \\ - & - & - & 18 & 15 \\ - & - & - & - & 9 \\ - & - & - & - & - \end{pmatrix}$$

   (a) Deseja-se encontrar boas soluções do *caixeiro viajante* (TSP), cuja instância é representada na matriz acima, usando a heurística de busca local baseada na vizinhança do tipo **2-troca** (**2-exchange** ou **2-opt**).
   Partindo da solução inicial formada pelo ciclo $C_0 = (1, 4, 3, 2, 5, 1)$ cujo comprimento é 65, aplique a heurística mencionada acima até que você atinja um mínimo local. Mostre por meio de desenhos todas as soluções (ciclos) vizinhas da solução corrente a cada iteração, sempre indicando claramente qual o custo destas soluções e qual foi aquela escolhida para tornar-se a solução corrente na iteração seguinte.

   (b) Deseja-se identificar pelo menos uma melhor solução (solução de valor ótimo) para a instância representada pela matriz D. Para tal, construa a árvore de enumeração para o problema. Sugestão: utilize o resultado da busca local acima para evitar fazer uma enumeração explícita, isto é, para tentar podar ramos da árvore. Questão: é possível usar o valor da aresta de menor comprimento para tentar aumentar o número de podas na árvore?

2. Suponha que você deseja resolver um problema de otimização discreta utilizando a técnica de *branch-and-bound* como abordagem de solução. Você sabe que uma forma de obter limitantes para os problemas correspondentes aos nós da árvore de busca é resolver uma relaxação do problema alvo em cada nó - por exemplo, uma relaxação linear para o problema. Entretanto, suponha também que: (a) você não tem conhecimento suficiente sobre métodos de solução de problemas de programação linear e que (b)

motivado pela questão 1 desta lista você escolheu o problema do *caixeiro viajante* como problema de otimização a ser resolvido.
Assim sendo, elabore uma proposta de *branch-and-bound* para o TSP. Isto é, proponha uma forma de: (a) ramificação para uma árvore de enumeração (*branching*) para o problema e (b) cálculo de limitantes inferiores (*bounds*) para os problemas relativos a cada nó da árvore (sugestão: encontre um problema - da classe P - cuja solução provê um limitante inferior para o TSP).

3. Avalie se as afirmativas abaixo são verdadeiras ou falsas. Justifique suas respostas:

   (a) Num passo de ramificação de um algoritmo de *branch-and-bound* é necessário que o particionamento do conjunto de soluções seja disjunto?

   (b) Considere o problema $z = \max\{cx : x \in S\}$ e sejam:

      i. $S = S_1 \cup \cdots \cup S_k$ uma decomposição de S em conjuntos menores
      ii. $z^k = \max\{cx : x \in S_k\}$ para $k = 1, \cdots, K$
      iii. $\overline{z}^k$ um limitante superior para z^k
      iv. $\underline{z}^k$ um limitante inferior para z^k

      Então $\overline{z} = \max_x \overline{z}^k$ é um limitante superior para z^k e $\underline{z} = \max_x \underline{z}^k$ é um limitante inferior para z_k?

4. Nas figuras abaixo são mostrados três exemplos de decomposição do problema S (maximização) nos problemas S_1 e S_2, bem como limitantes inferiores e superiores obtidos para cada problema.
Indique, para cada problema, quais podas (caso exista alguma) podem ser feitas em cada uma das árvores, explicitando os motivos da(s) poda(s). Indique também as transferêcias de limitantes inferiores e superiores de S_1 e S_2 para S (sugestão: redesenhe as árvores indicando os novos valores de limitantes).

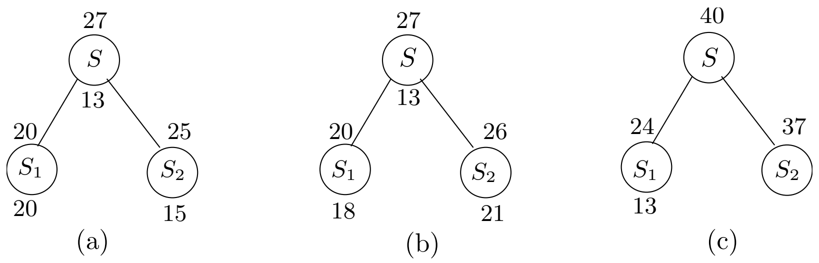

Figura 1: Exemplos de limitantes para decomposições de S em S_1 e S_2.

5. Considere a árvore de enumeração (problema de minimização) na figura abaixo:

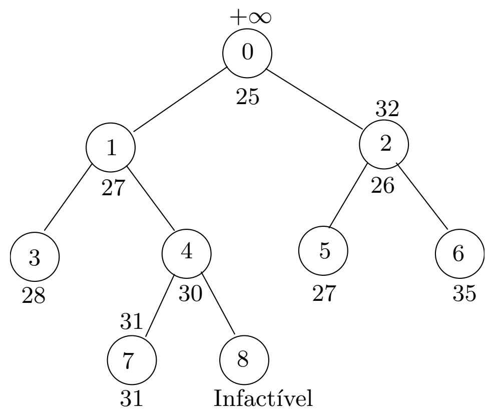

Figura 2: Árvore de enumeração (problema de minimização).

(a) Determine os melhores limitantes possíveis para o valor de uma solução ótima para o problema.

(b) Quais nós podem ser podados e quais ainda devem ser explorados?

6. Considere o seguinte algoritmo (heurística gulosa) para o problema de cobertura de vértices de cardinalidade mínima:
A cada passo do algoritmo, um vértice de maior grau (empates são quebrados arbitrariamente) é adicionado à cobertura e é removido do grafo junto com todas as arestas que nele incidem. O algoritmo termina quando não existem mais arestas no grafo[1].
Considere a afirmação: *a solução encontrada pelo algoritmo acima é uma cobertura de vértices cuja cardinalidade é, no máximo, o dobro da cardinalidade de uma cobertura de vértices de cardinalidade mínima.*
Se você concorda com a afirmação, use seus conhecimentos de algoritmos de aproximação e prove que ela está correta. Caso contrário, mostre um exemplo de um grafo e uma ordem de execução do algoritmo descrito que levam a uma cobertura de vértices com mais que o dobro do número de vértices de uma cobertura mínima para o grafo escolhido.

[1] Note que, como uma aresta só é removida após um vértice incidente a ela ser incluído na cobertura, o algoritmo encontra uma cobertura de vértices.