

PUC-Rio
Departamento de Informática
Profs. Marcus Vinicius S. Poggi de Aragão
Período: 2006.1
20 de junho de 2006
Horário: 2as-feiras e 4as-feiras de 19 às 21 horas

PROJETO E ANÁLISE DE ALGORITMOS (INF 1309)

2º Trabalho Prático (T2)

- O objetivo do 2º Trabalho é a implementação de um algoritmo de *branch-and-bound* para um problema NP-difícil. Isto é, um problema de otimização cuja versão de decisão é um problema NP-completo.

Este trabalho deve ser feito em grupos de no MÁXIMO 3(TRÊS). Caso contrário não será considerado.

- O problema sobre o qual deve-se aplicar o algoritmo de *branch-and-bound* segue:
 - Problema da Árvore Geradora com Restrições de Capacidade (CAP-MST): Dado um grafo $G = (V \cup \{r\}, E)$, custos $c(i, j) \geq 0$ associados às arestas (i, j) em E , demandas $q(v)$ associadas aos vértices v em V , e uma capacidade Q . Deseja-se encontrar uma árvore geradora de G , cujas sub-árvores enraizadas no vértice r possuam a soma das demandas dos seus vértices inferiores ou iguais à Q , cujo custo total de suas arestas seja **mínimo**. As instâncias estão no link CAP-MST da página de Análise de Algoritmos 06.1.

A apresentação do seu algoritmo deve conter uma descrição de cada um dos elementos em um *branch-and-bound*, são eles:

- Critério de particionamento do espaço de soluções. Conforme apresentado em aula o critério a ser utilizado é o de particionar em dois conjuntos: um onde a árvore geradora contém obrigatoriamente uma dada aresta; e outro onde esta aresta não está na árvore.
- Um critério para percorrer os subconjuntos do espaço de soluções: Como definido em aula será utilizado o critério de busca em profundidade.
- Uma relaxação do problema (uma função que gera sempre um valor inferior ao da solução ótima para o subconjunto de soluções considerado). Neste item, mostre como a sua relaxação é modificada quando se considera apenas um subconjunto do espaço de soluções (ou seja, quando um elemento tem seu valor fixado, no caso uma aresta passa a estar obrigatoriamente na árvore geradora ou fora dela). Como definido em aula, será utilizada como relaxação a árvore geradora mínima que não considera a restrição de capacidade.
- Este item **NÃO É OBRIGATÓRIO**. Um método para obter uma “boa” solução viável (que seria a melhor solução conhecida antes de iniciar o *branch-and-bound*). Esta solução pode ser obtida por um método guloso, por exemplo.
- Critério de seleção do particionamento. Na aula foi proposto ordenar as arestas pelos seus respectivos custos. Outras ordens podem ser tentadas.

Apresente sempre a sua melhor solução (lista das arestas com seus respectivos valores) e o seu valor total. Caso o tempo de CPU ultrapasse 1 hora, faça com que seu algoritmo termine imprima a melhor solução encontrada até então.

Deverá ser apresentado um relatório sobre as experiências computacionais comentando os resultados obtidos. Este relatório deverá conter:

- Uma tabela com o valor da melhor solução obtida, indicando se é ótima (provado pelo seu algoritmo) ou não, o tempo de cpu total utilizado na resolução, o valor do limite inferior obtido no nó raiz, e o valor da solução ótima (ou melhor conhecida) que serão fornecidos. A tabela deverá ter uma linha para cada uma das 39 instâncias contidas no arquivo no link;
- Uma análise dos resultados com relação à complexidade assintótica do algoritmo implementado. Mostrar o crescimento do tempo é exponencial em função do tamanho da instância;
- Uma análise separada das diferentes etapas do algoritmo.

Os códigos (comentados) devem ser entregues eletronicamente apenas. Um roteiro para o documento a ser entregue segue:

- Descrever os algoritmos informalmente.
- Demonstrar o entendimento do algoritmo explicando, em detalhe, o resultado que o algoritmo deve obter e justificá-lo.
- Explicar a fundamentação do algoritmo e justificar a sua corretude. Apresentar e explicar a complexidade teórica esperada para cada algoritmo.
- Documente o arquivo contendo o código fonte de modo que cada passo do algoritmo esteja devidamente identificado e deixe claro como este passo é executado.

A corretude código será testada sobre o conjunto de instâncias que na página do curso. O trabalho entregue deve conter:

- Um documento contendo o roteiro de desenvolvimento dos algoritmos (e dos códigos), os itens pedidos acima, comentários e análises sobre a implementação e os testes realizados (papel).
- Um e-mail contendo um arquivo .zip com os códigos fonte e os executáveis correspondentes (mudar a extensão de .zip para .zxx) deve ser enviado para **marcus.poggi@gmail.com** com o ASSUNTO (ou SUBJECT) AA061T2.