FUNDAÇÃO EDSON QUEIROZ
UNIVERSIDADE DE FORTALEZA
Centro de Ciências Tecnológicas – CCT
Programa de Pós-Graduação em Informática Aplicada – PPGIA

Período: 2014.2 **Disciplina**: D279 – Otimização Combinatória e em Redes

Atividade computacional

Instruções:
1. A atividade deve ser realizada individualmente.
2. Implementações semelhantes estarão sujeitas a anulação imediata e definitiva.
3. A avaliação será progressiva através de acompanhamento pelo professor.

Parte I – Implementação de um modelo para *Fixed Charge Capacitated Network Design Problem*.

Formulação. Seja $G = (N, A)$ uma rede orientada, onde N é o conjunto de nós e A é o conjunto de arcos. Seja K o conjunto de demandas, cada uma delas caracterizada por uma origem s^k, um destino t^k, e uma quantidade d^k que deve ser transportada da origem para o destino. Seja f_{ij} o custo fixo de utilização do arco ij, c_{ij} o custo variável para transportar uma unidade de fluxo através do arco ij, e u_{ij} a capacidade do arco ij. Considere a variável x_{ij}^k que indica a quantidade de fluxo referente à demanda k no arco ij, e a variável binária y_{ij} que indica se o arco ij é utilizado ou não. O objetivo é minimizar os custos fixos e os custos variáveis.

$$\min \sum_{k\in K}\sum_{ij\in A} c_{ij}x_{ij}^k + \sum_{ij\in A} f_{ij}y_{ij}$$

$$\text{s.t.} \sum_{ij\in A} x_{ij}^k - \sum_{ji\in A} x_{ji}^k = \begin{cases} d^k, & \text{for } i = s^k \\ -d^k, & \text{for } i = t^k \\ 0, & \text{otherwise} \end{cases} \qquad \forall i \in N, \forall k \in K$$

$$\sum_{k\in K} x_{ij}^k \le u_{ij}y_{ij} \qquad \forall ij \in A$$

$$x_{ij}^k \ge 0 \qquad \forall ij \in A, \forall k \in K$$

$$y_{ij} \in \{0,1\} \qquad \forall ij \in A$$

Implementação. Implementar um modelo de programação matemática utilizando Java Concert para o problema. Uma instância deve ser lida de um arquivo de entrada (FCND.dat). Devem ser gerados 3 arquivos de saída: um arquivo com a saída padrão do CPLEX (FCND.log), um arquivo com a formulação matemática (FCND.lp), e um arquivo de solução (FCND.out) conforme especificado abaixo.

Entrada. A primeira linha contém um inteiro n, indicando a quantidade de nós (rotulados de 1 a n), e um inteiro m, indicando a quantidade de arcos. Cada uma das m linhas seguintes possui uma quintupla que caracteriza um arco: $(i, j, f_{ij}, c_{ij}, u_{ij})$. A linha seguinte contém um inteiro k indicando a quantidade de demandas. Cada uma das k linhas seguintes possui uma tripla que caracteriza uma demanda: (s^k, t^k, d^k).

Saída. O arquivo de solução deve apresentar o custo total, o tempo e os arcos efetivamente utilizados.

Exemplo de entrada:

```
 6              8
 1              2            100              1            100
 2              3             50              1             60
 2              4            100              1            100
 3              1            100              1            100
 4              6            100              1            100
 5              3            100              1            100
 5              4             50              1             60
 6              5            100              1            100
10
 1              3             10
 1              4             10
 2              1             10
 2              5             10
 3              6             10
 4              1             10
 4              5             10
 5              1             10
 5              6             10
 6              2             10
```

Exemplo de saída:

```
Objetive:              950,00
Lower bound:           950,00
Gap:                   0,0000
Status:               Optimal
Time:                    0,03
 1              2            100
 2              3             50
 2              4            100
 3              1            100
 4              6            100
 5              3            100
 6              5            100
```

Parte II – Implementação de uma relaxação lagrangeana para o problema.

Relaxação. Relaxar a restrição de capacidade de cada arco *ij* e associar um multiplicador lagrangeano μ_{ij}, conforme modelo abaixo.

$$\min_{y} \sum_{k\in K}\sum_{ij\in A} c_{ij}x_{ij}^{k} + \sum_{ij\in A} f_{ij}y_{ij} + \sum_{ij\in A}\mu_{ij}\left(\sum_{k\in K} x_{ij}^{k} - u_{ij}y_{ij}\right)$$

$$\text{s.t.} \quad \sum_{ij\in A} x_{ij}^{k} - \sum_{ji\in A} x_{ji}^{k} = \begin{cases} d^{k}, & \text{for } i = s^{k} \\ -d^{k}, & \text{for } i = t^{k} \\ 0, & \text{otherwise} \end{cases} \qquad \forall i \in N, \forall k \in K$$

$$x_{ij}^{k} \geq 0 \qquad \forall ij \in A, \forall k \in K$$

$$y_{ij} \in \{0,1\} \qquad \forall ij \in A$$

Implementação. Implementar o algoritmo do subgradiente conforme ilustrado abaixo. UB deve ser obtido pela resolução do modelo original onde todos os links são utilizados ($y_{ij} = 1, \forall ij \in A$). Em cada iteração, o subproblema lagrangeano deve ser resolvido, os limites inferiores e superiores devem ser adequadamente atualizados. Considere os parâmetros $K = 1000, \beta = 10, \varepsilon = 0.000001$. Note que a cada iteração apenas a função objetivo deve ser modificada de acordo com os multiplicadores lagrangeanos. Deve ser gerado um arquivo de saída FCND.lgr conforme especificado abaixo.

Algorithm 1 Subgradient algorithm

{Input}
An upper bound UB
{Initialization}
$\mu^0 = 0$
$\lambda_0 = 2$
{Subgradient iterations}
$k = 0$
while $k \leq K$ **do** {stopping criterion}
  $\gamma^k = Ax^k - b$ {gradient of $L(\mu^k)$}
  $\theta_k = \lambda_k(UB - L(\mu^k))/||\gamma^k||^2$ {step size} $\{||\gamma|| = (\sum_j \gamma_j^2)^{1/2}\}$
  $\mu^{k+1} = \max\{0, \mu^k + \theta_k\gamma^k\}$
  if $||\mu^{k+1} - \mu^k|| < \varepsilon$ **then**
    Stop
  end if
  if no progress in more than β iterations **then**
    $\lambda_{k+1} = \lambda_k/2$
  else
    $\lambda_{k+1} = \lambda_k$
  end if
  $k = k + 1$
end while

Saída. Para cada iteração do algoritmo, o arquivo de saída deve apresentar o limite superior, o limite inferior, o gap e o tempo decorrido. Ao final, deve ser também apresentado o limite inferior obtido pela relaxação linear do modelo original.