MC658 - Projeto e Análise de Algoritmos III

Prof. Flávio Keidi Miyazawa / PED: Gustavo
Laboratório 1 - 2o. Semestre de 2015

Descrição: Neste laboratório, você deve implementar um algoritmo, utilizando programação linear (não pode usar variáveis inteiras), para o seguinte problema:

> **Controle de Tráfego:** Suponha que o governo deseja monitorar as informações que passam pelas conexões existentes em uma certa rede de computadores bem particular. Esta rede de computadores será representada aqui através de um grafo não-orientado $G = (V, E)$, onde cada vértice em V representa um computador e cada aresta $\{u, v\}$ em E representa uma conexão direta entre o computador u e o computador v.
>
> Para fazer este monitoramento, o governo pode instalar aparelhos nos computadores, que chamaremos simplesmente de Aparelhos Controladores (AP), ou simplesmente por sua sigla AP. Ao instalar um AP em um computador $v \in V$, ele pode monitorar todas as conexões diretas existentes entre o computador v e outros computadores. Além disso, o único jeito para monitorar uma conexão direta $\{u, v\}$ é instalando um AP em u ou em v (basta em um deles).
>
> Uma solução possível é instalar um AP em cada computador em V e com isso todas as conexões diretas serão monitoradas. Porém, para cada computador $v \in V$, há um custo $custo(v) \geq 0$ para instalar um AP em v. Com isso, a solução de se instalar um AP em cada computador $v \in V$ pode ficar muito cara. Sabendo que o grafo G (que representa a rede) é bipartido, faça um programa que encontre um conjunto de computadores $S \subseteq V$ tal que todas as conexões diretas sejam monitoradas e o preço total para se instalar os computadores de S é mínimo (i.e., $\sum_{v \in S} custo(v)$ é mínimo).

Para este laboratório, você deverá resolver o problema acima, nas condições que foram dadas. Para isso, você deve modificar a rotina `monitoramento_em_grafo_bipartido` que se encontra dentro do arquivo `monitoramento_em_grafo_bipartido.cpp`.

A rotina `monitoramento_em_grafo_bipartido` tem o seguinte cabeçalho:

```
void monitoramento_em_grafo_bipartido(
                          ListGraph &g,
                          ListGraph::NodeMap<double>& custo,
                          ListGraph::NodeMap<int>& solucao)
```

onde `g` é o grafo que representa a rede de computadores, `custo` é um vetor indexado nos vértices (computadores) do grafo (rede) e dá o custo de instalação de um AP em cada vértice (computador). O vetor `solucao` é um vetor indexado nos vértices e deve ser alterado dentro da rotina para conter valores iguais a 0 ou 1. O vetor `solucao` identifica os vértices escolhidos para se instalar os AP's. Dado vértice (computador) $v \in V$, se `solucao`(v) é igual a 0 então não é instalado AP em v. Se `solucao`(v) é igual a 1 então é instalado AP em v (e naturalmente há um custo de `custo`(v) por esta instalação).

Para resolver este problema, você deve usar a resolução de programas lineares através do pacote GUROBI, junto com a biblioteca LEMON. O programa disponibilizado para este laboratório já funciona, mas a rotina `monitoramento_em_grafo_bipartido` simplesmente faz a atribuição `solucao`$(v) = 1$ para cada $v \in V$. Naturalmente esta é uma solução viável, mas não necessariamente a solução ótima. Seu objetivo será alterar a rotina resolvendo o problema através da resolução de programação linear.

Você só deve alterar a rotina `monitoramento_em_grafo_bipartido` e dentro dela, seu programa não deverá fazer nenhuma leitura pelo teclado nem impressão na tela. A solução será verificada pelo programa principal, através do parâmetro `solucao`.

Execução: Para compilar o código do arquivo `monitoramento_em_grafo_bipartido.cpp`, basta baixar o arquivo disponibilizado para o laboratório, descompactá-lo e dentro do diretório gerado, executar:

```
$ make
```

Para executar, faça:

```
$ ./monitoramento_em_grafo_bipartido.e <grafo_bipartido>
```

este comando irá apresentar o peso da solução encontrada (caso o parâmetro `solucao` represente de fato uma solução para o problema).

Existem alguns grafos de exemplo no mesmo diretório.

Submissão: Submeta somente o arquivo `monitoramento_em_grafo_bipartido.cpp`, com sua modificação, através do SuSy A submissão deve ser feita até as 15:50. O SuSy não testará sua solução, é necessário que você mesmo se assegure da corretude.

Exemplo de valores de soluções ótimas

Se seu programa estiver correto, o programa exibirá uma janela com o grafo e os vértices selecionados na solução em vermelho.

Alguns resultados de execução:

Grafo de Entrada	Valor da Solução Ótima
gr_bipartido_01	30.0
gr_bipartido_02	63.0
gr_bipartido_03	93.0
gr_bipartido_04	74.0
gr_bipartido_05	69.0

Table 1: Alguns grafos disponibilizados e valor de suas soluções ótimas

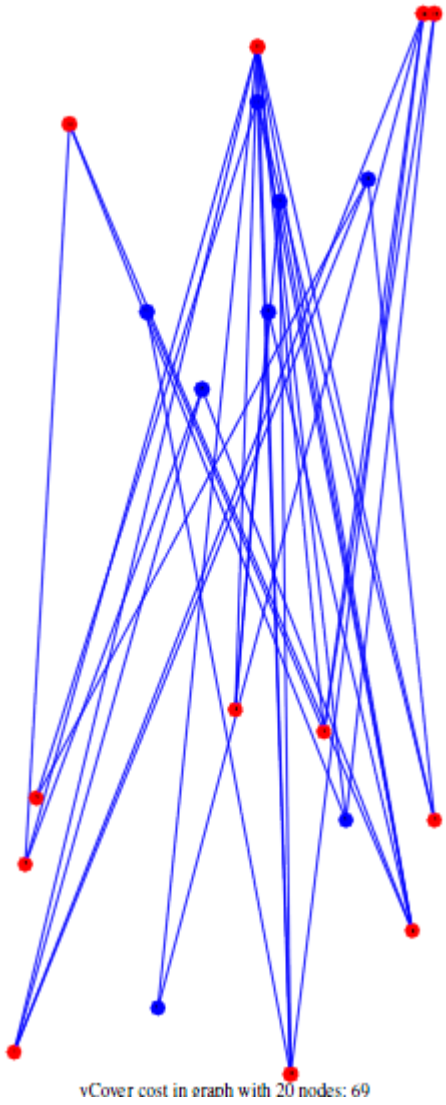

Figure 1: Grafo Resultante do arquivo de entrada gr-bipartido-5.in