Trabalho Prático

MC548— Análise de Algoritmos II

Prof. Cid C. de Souza
Igor Ribeiro de Assis (PED)

1º Semestre de 2011

1 Parte 1: Modelagem PLI

Para cada um dos problemas abaixo escreva um programa usando a linguagem *MathProg* [2] que o resolva e modele o mesmo problema em uma planilha do OpenOffice seguindo o modelo distribuído na página da disciplina.

Um exemplo de entrada para cada um dos problemas é dado na página da disciplina. Seu programa deve suportar a leitura dos dados de entrada seguindo o formato das instâncias de exemplo.

1.1 Problemas

1. [gt54] Seja $G = (V, A)$ um grafo orientado, s, t vértices distintos de G e $C = \{(a_1, b_1), \ldots, (a_n, b_n)\}$ pares de vértices de G. Encontre o caminho mínimo de s a t que contenha no máximo um vértice de cada par de C.

2. [ss2] Seja $T = \{1, \ldots, n\}$ um conjunto de tarefas e S um conjunto de pares (i, j) para $1 \leq i, j \leq n$ determinando que a tarefa i deve ser executada antes da tarefa j. Para cada tarefa $i \in T$ seja t_i o tempo de execução e d_i o prazo de término da tarefa i. Determine uma plano de execução das tarefas em um único processador, de forma que a ordem de precedência das tarefas seja respeitada e o número de tarefas que terminem fora do prazo é minimizado.

3. [gt10] Dado um grafo simples $G = (V, E)$. Encontre um emparelhamento maximal mínimo de G.

4. [mn27] Dado um grafo simples $G = (V, E)$. Determine o menor k tal que G tenha uma $k-$coloração.

5. [nd32] Seja $G = (V, A)$ um grafo orientado, s e t dois vértices distintos de G, c_{ij} a capacidade e p_{ij} o custo de utilização do arco $(i, j) \in A$, respectivamente. Seja R a demanda em t. Encontre um fluxo viável f de s para t que minimize o custo total de utilização da rede. Note que o custo de utilização de um arco só deve ser computado se o fluxo que for passar nele for maior que zero.

2 Parte 2: Desenvolvimento de Heurísticas

O objetivo desta parte é desenvolver uma heurística para um problema NP-difícil. Você é livre para implementar qualquer heurística ou meta-heurística vista em sala ou não.

2.1 Descrição do Problema

Problemas de escalonamento de satélites para observação da Terra são bastante interessantes para a aplicação de técnicas de otimização e pesquisa operacional.

No problema da Seleção de Segmentos de Faixas (SSSP) estamos interessados em um caso especial do problema de observação da Terra por satélites em que são dados dois conjuntos de satélites S_h e S_v com o mesmo tamanho, cujos satélites em S_h se movem numa semi-órbita de nordeste a sudoeste e os satélites de S_v de sudeste a noroeste. Além disso é dada a localização de um objeto na Terra que queremos fotografar. Em geral são necessárias várias fotos de diversos satélites para observar um objeto

A área observada por um satélite é dada por uma região de largura fixa pré-determinada e uma altura dependente do comprimento da órbita, essa região de observação é chamada de *faixa*. Durante a órbita um satélite pode fotografar partes de sua faixa e armazenar as imagens em uma memória temporária que é descarregada ao final da órbita. A memória ocupada pelas imagens é proporcional a área fotografada, isto é, o produto da largura pela altura da imagem (dada pela distância percorrida na direção da órbita do momento que a captura inicia até o final).

A área total de observação dos satélites é dada pela intersecção das faixas dos satélites de S_h com os de S_v. Essas intersecções são chamadas de *segmentos de faixa*. As partes do objeto a ser fotografado que estão contidas em um segmento são chamadas de *shards*. Note que cada shard pertence a exatamente um segmento e que um segmento pode ou não conter um shard. A figura 1 ilustra cada uma dessas definições.

Para cada shard são atribuídos dois valores, o custo de armazenagem e o ganho de observação r. O custo de armazenagem no entanto depende de qual dos dois satélites fez tirou a foto, pois embora a largura da faixa seja fixa a altura é variável pois o satélite só precisa fotografar a região do shard. A figura 2 exemplifica a diferença de custo de fotografar um dado shard por cada um dos satélites cujas faixas o contém.

O objetivo do problema é maximizar o ganho de observação, dado pela soma do ganho de cada shard fotografado, respeitando a capacidade de armazenamento de cada satélite e cada shard é fotografado no máximo uma vez.

2.2 Tarefa a ser executada (parte 2)

Seu objetivo é desenvolver uma heurística para o SSSP. Você deve implementar sua heurística em linguagem *C* seguindo o padrão ISO C99 [3] ou em *C++* seguindo o padrão ISO C++ 2003 [4]. Seu programa não deve utilizar nenhuma biblioteca que não esteja definida nesses

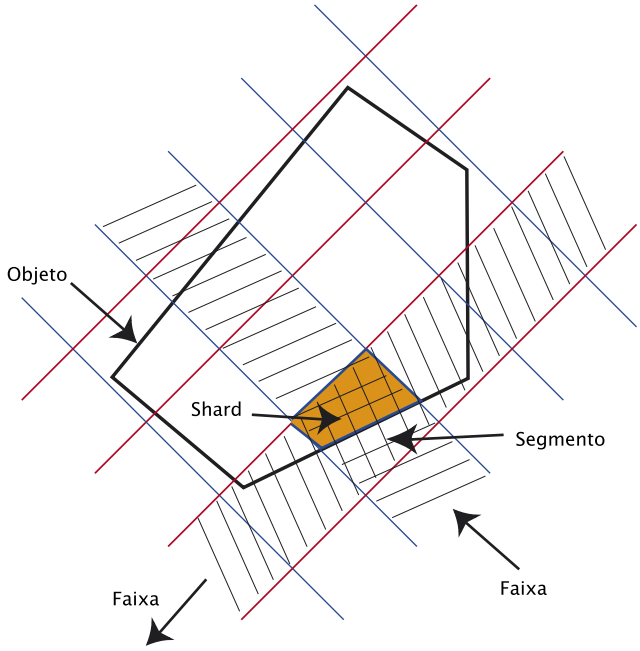


Figura 1: Faixas de dois satélites em direções contrárias, o segmento da intersecção das faixas e a shard correspondente.

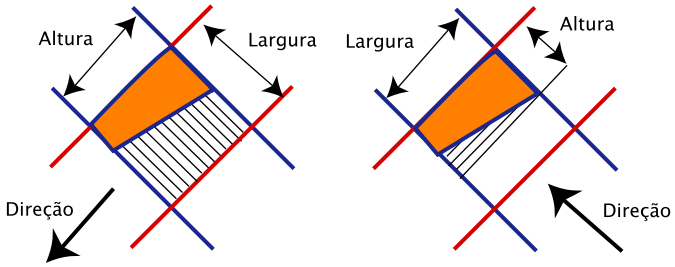


Figura 2: Custo de observação de um shard por cada um dos satélites que o contém.

padrões ou código que não seja inteiramente desenvolvido por você ou tenha sido distribuído na página da disciplina [1].

2.3 Entrada e Saída (parte 2)

A entrada é dada por um arquivo texto no seguinte formato. A primeira linha contém um inteiro m que representa o número de satélites em cada direção. Por simplificação, nas instâncias de teste vamos supor que os satélites movimentam-se nas direções horizontal e vertical. Os satélites de cada direção são rotulados com um inteiro distinto $0 \leq i < m$. As duas próximas linhas contém m pares de inteiros "i b_i" em que i é o rótulo do satélite e b_i a sua capacidade de memória. Os m pares da segunda linha correspondem aos satélites em S_h e os m da terceira aqueles em S_v.

A linha a seguir contém um inteiro k representando o número de shards que podem ser fotografados e cujo ganho é não-nulo. As próximas k linhas contém uma quintupla de inteiros "i j r_{ij} c^h_{ij} c^v_{ij}" em que r_{ij}, c^h_{ij}, c^v_{ij}, correspondem respectivamente, ao ganho de fotografar o shard contido na intersecção do satélite $i \in S_h$ com o satélite $j \in S_v$, o custo de armazenagem do shard pelo satélite i e o custo de armazenagem pelo satélite j.

A linha seguinte tem um inteiro n que representa o número de vértices do polígono de entrada que descreve o objeto a ser fotografado. As próximas n linhas contém pares "x y" com as coordenadas dos vértices do polígono em sentido horário.

Observe que o polígono de entrada não é necessário para resolver o problema, e o mesmo é informado na entrada com o intuito de auxiliar a construção de ferramentas de visualização e análise da entrada/solução, caso seja de interesse do grupo tal ferramenta.

Na figura 3 é dado um exemplo de entrada e o arquivo correspondente. Supõe-se neste exemplo que a largura de cada faixa é 4.

A saída deve ser dada em texto no seguinte formato. A primeira linha é um inteiro com o valor da função objetivo, a segunda linha é um inteiro s que representa o número de shards fotografados. As s linhas seguintes contém triplas "i j c" em que c é um dos caracteres `h` ou `v` e indica qual dos satélites fotografou o shard ij.

Na figura 4 é dada uma possível saída para a instância de entrada na figura 3 e o arquivo correspondente.

3 Forma de Entrega do Trabalho

Você deve entregar um arquivo chamado "raXXXXXX-raYYYYYY.tar.gz", que ao ser descompactado deve gerar um diretório chamado "raXXXXXX-raYYYYYY/" que deve conter: *(i)* um subdiretório "parte1/"; *(ii)* um subdiretório "parte2/"; e *(iii)* o arquivo `relatorio.pdf`. Dentro do subdiretório "parte1/", para cada exercício resolvido da parte 1 deve existir um arquivo chamado *NOME.pli* onde *NOME* é o identificador do exercício, dado entre "[" e "]" na primeira linha do enunciado.

Cada arquivo com extensão ".pli" no diretório "parte1/" ao ser executado com a linha de comando,

```
glpsol -m NOME.pli  -d NOME.dat
```

dever gerar um arquivo de saída chamado *NOME.out* em que a primeira linha deve conter apenas o valor da função objetivo e mais nada.

Para a parte 2, o subdiretório "parte2/" deve conter o código-fonte da sua heurística e um arquivo "Makefile" que ao ser executado pelo programa *make* irá gerar um executável chamado "heur" que deve suportar a seguinte forma de execução:

```
./heur -t  TIME -o  SAIDA ENTRADA
```

onde *TIME* é um inteiro que representa o número máximo de segundos que o programa será executado, *ENTRADA* e *SAIDA* são caminhos para arquivos que correspondem a uma instância de entrada e onde deve ser escrito o arquivo de saída, respectivamente.

Formato do relatório. O relatório deverá ser entregue em formato `pdf` (nenhum outro formato será aceito) no arquivo chamado `relatorio.pdf`, conforme já destacado anteriormente. O texto deverá ser composto das seguintes seções:

1. Seção 1: Identificação dos integrantes do grupo com nome e RA.

2. Seção 2: refere-se a parte 1 do trabalho. Para cada um dos cinco exercícios deverão ser apresentadas as seguintes informações: *(i)* o identificador do exercício que foi resolvido no formato `[XXX]` conforme o enunciado recebido pelo grupo; *(ii)* a definição das variáveis usadas no modelo; *(iii)* a descrição de cada uma (das famílias de) restrições do modelo, dando a sua fórmula acompanhada de uma explicação **sucinta** do seu significado; e *(iv)* a fórmula da função objetivo.

   Ao final dessa seção deverá ser dada uma tabela de resultados no seguinte formato. Cada linha estará associada a um exercício, identificado por `[XXX]` na primeira coluna, e as demais colunas corresponderão ao número da instância que foi resolvida (de 1 a 3). Em cada célula deverá ser apresentado o valor da função objetivo obtido para o par (exercício,instância) correspondente.

3. Seção 3: refere-se a parte 2 do trabalho. Aqui deve ser feita uma descrição de não mais de **4 páginas** dos passos principais da heurística desenvolvida pelo grupo para resolver o problema proposto. Devem ser enfatizados aspectos relevantes relativos à complexidade das rotinas implementadas, às estruturas de dados utilizadas e a tudo mais que diga respeito ao projeto e à implementação do algoritmo.

   Serão muito bem vindas análises da qualidade do resultados alcançados (por exemplo, comparando-os com limitantes duais que o grupo tenha identicado ou mostrando como os limitantes primais finais produzidos pela heurística são afetados pela inclusão/exclusão de algumas rotinas) e do tempo de computação gasto pela heurística desenvolvida pelo grupo (por exemplo, utilizando gráficos que mostrem a melhoria da qualidade da solução observada ao longo de uma execução típica do algoritmo).

Ao final dessa seção deverá ser exibida uma tabela que resumirá os resultados alcançados pela heurística. Nessa tabela cada linha irá se referir a uma das instâncias disponibilizadas para teste e terá três colunas. Na primeira deverá constar o identificador da instância de teste, na segunda o valor da função objetivo para a solução encontrada pela heurística e, por fim, na terceira coluna o tempo gasto para obter esta solução.

O texto deverá informar necessariamente o ambiente computacional onde foram realizados os experimentos, incluindo detalhes de *hardware* (CPU, memória RAM, clock, etc) e *software* (sistema operacional, linguagem de programação, compilador, etc).

OSERVAÇÕES IMPORTANTE:

- não colocar no relatório o enunciado dos problemas tratados. Isso vale tanto para parte 1 quanto para a parte 2. O relatório deve limitar-se a descrever o que foi feito pelo grupo e, quando solicitado, a analisar os resultados obtidos.
-

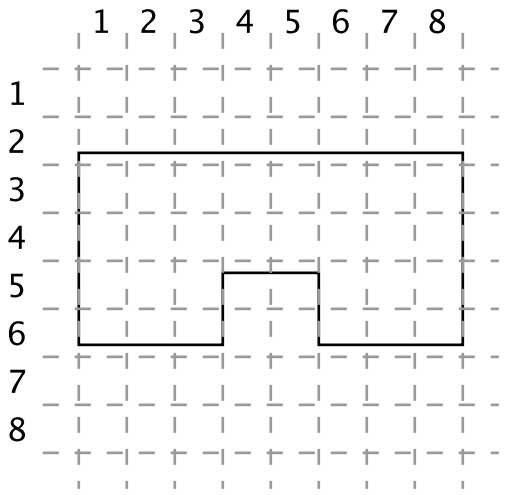
Referências

[1] Cid C. de Souza. MC548 – Projeto e Análise de Algoritmos II, Instituto de Computação, UNICAMP. www.ic.unicamp.br/∼cid/cursos/MC548/201101/, Primeiro Semestre de 2011.

[2] GNU. GNU linear programming kit, May 2011.

[3] ISO. ISO C standard 1999. Technical report, 1999. ISO/IEC 9899:1999.

[4] ISO. ISO C++ standard 2003. Technical report, 2003. ISO/IEC 14882:2003.




```
8
1 32 2 32 3 32 4 32 5 32 6 64 7 32 8 32
1 32 2 32 3 32 4 36 5 5 6 32 7 32 8 32
10
3 1 8 16 16
3 4 4 16 16
4 4 10 16 16
4 7 12 16 16
5 1 5 16 6
5 4 8 16 4
5 5 25 16 4
6 3 4 16 12
6 6 3 16 12
6 8 2 16 12
8
0 7
32 7
32 23
20 23
20 17
12 17
12 23
0 23
```

Figura 3: Exemplo de instância e o arquivo de entrada correspondente.

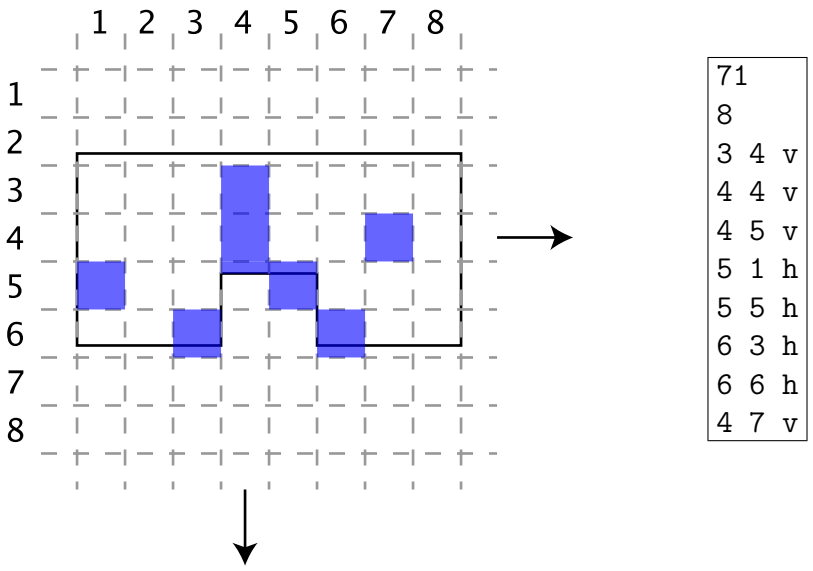


Figura 4: Exemplo do arquivo de saida com sua representação gráfica.