UFMG/ICEx/DCC — Algoritmos e Estrutura de Dados 3

Trabalho Prático 1

Bacharelado — 1º Semestre de 2012

Este trabalho prático tem por objetivo modelar um problema real com o uso de estruturas de grafos, elaborar diferentes heurísticas para solucionar variações de um mesmo problema e, por fim, apresentar uma análise comparativa através de diferentes métricas.

1 Definição do Problema

Este trabalho aborda o problema de alocação de alunos em universidades através da prova do ENEM (Exame Nacional do Ensino Médio). Várias universidades brasileiras utilizam a nota do ENEM como critério de seleção de candidatos aos cursos que ofertam. Assim, alunos de todo o país se inscrevem para a prova do ENEM, e apresentam uma lista limitada e ordenada com a preferência das universidades que gostariam de ingressar. Dessa maneira, tenta-se aprovar os alunos nos vestibulares de forma que eles sejam alocados às universidades de suas preferências considerando um número limitado de vagas disponibilizado para cada uma delas. A lista de preferências das universidades, por sua vez, é representada pelo *ranking* de todos os alunos por ordem decrescente de nota.

Basicamente, esse problema pode ser visto como uma variação do problema clássico do Casamento Estável (*Stable Marriage Problem*, SMP) que consiste em encontrar um *matching* estável (*stable matching*) entre dois conjuntos distintos de elementos. Contudo, o SMP assume que todas as listas de preferências são completas - contêm todos os elementos do conjunto oposto. Neste trabalho, assumimos que as listas são **incompletas** - de tamanho até k, problema denominado Casamento Estável de Listas Incompletas (*Stable Marriage with Incomplete Lists Problem*, SMIP).

Neste trabalho, o aluno deve elaborar **DUAS** heurísticas distintas a fim de favorecer dois aspectos para uma dada instância do problema relativo ao ENEM. Além disso, o aluno deve apresentar uma análise comparativa entre as heurísticas para avaliar a qualidade das aprovações sob diferentes perspectivas, de forma a mensurar vantagens e desvantagens de cada abordagem.

Observe que as listas de preferências **NÃO DEVEM** ser ignoradas, isto é, nenhum aluno pode ser alocado a uma universidade cujo *id* não esteja presente na sua respectiva lista de preferências. O cenário descrito acima permite **casamentos instáveis**, isto é, quando um dado aluno a_1 prefere uma dada universidade u_1 em relação a uma outra universidade u_2 a qual a_1 está alocado, e u_1 também prefere a_1 em relação a algum outro aluno que u_1 aprovou.

As duas heurísticas **DEVEM** priorizar o completo preenchimento das vagas das universidades consideradas, mesmo que alunos sejam alocados em universidades de menor preferência. Além disso, cada heurística deve favorecer individualmente cada um dos aspectos listados abaixo:

- Maximização do número de vagas ocupadas nas universidades consideradas e:
  - Heurística 1: Maximização da média da satisfação* dos alunos aprovados.
  - Heurística 2: Maximização da média da satisfação* das universidades consideradas.

*__Satisfação__: para um dado *aluno*, é uma métrica que expressa a posição ocupada na lista de preferências pela universidade onde ele foi aprovado. Já para uma dada *universidade*, é uma métrica que expressa a média das notas dos alunos aprovados.

A primeira abordagem objetiva maximizar a satisfação dos alunos a fim de favorecer suas prioridades. Dessa forma, a heurística criada deve priorizar a alocação de um dado aluno à uma universidade de alta preferência.

Por sua vez, a segunda abordagem favorece a satisfação da universidade, isto é, prioriza a alocação de alunos com notas altas. Nessa heurística, o objetivo é obter uma alta nota de corte, representada pela menor nota dos alunos aprovados em uma dada universidade.

Um conjunto de métricas deverá ser apresentado para **CADA** heurística, com o intuito de permitir uma análise rica do comportamento de diferentes abordagens frente uma mesma instância. As métricas são:

- Média e mediana do percentual do número de vagas ocupadas nas universidades.
- Media e mediana da satisfação dos alunos aprovados.
- Média e mediana da satisfação das universidades.
- Média e mediana das notas de corte das universidades.

A implementação do algoritmo **DEVE** utilizar listas encadeadas com o uso de ponteiros e alocação dinâmica de memória.

PONTO EXTRA: analisar o percentual médio de ocupação das vagas das universidades em função de um parâmetro p. O parâmetro p representa o tamanho **EXATO** da lista de preferências dos alunos. A geração das listas de preferências dos alunos nas instâncias deve seguir duas distribuições: Uniforme e Zipf (distribuição de cauda pesada discreta).

Para a distribuição Uniforme, considera-se que a probabilidade de cada universidade aparecer na lista de preferência dos alunos é dada por:

$$\frac{1}{u}, \text{ sendo } u \text{ o número de universidades consideradas.}$$

Já para a distribuição Zipf, a probabilidade $P(u_i)$ de cada universidade aparecer na lista de preferência dos alunos não é uniforme. Cada universidade u_i possui uma probabilidade dada por:

$$P(u_i) = (u_i \sum_{i=1}^{u} i^{-1})^{-1}$$

O aluno deve apresentar em sua análise um gráfico que represente a média da porcentagem de ocupação das vagas nas universidades em função do parâmetro p.

1.1 Entrada e Saída

O arquivo executável deve ser chamado de *tp1* e deve receber como parâmetro o arquivo de entrada *input.txt* e o arquivo de saída *output.txt*, conforme demonstrado a seguir:

```
./tp1 -i input.txt -o output.txt
```

O arquivo *input.txt* deve conter as seguintes informações, na ordem indicada abaixo:

1. número de instâncias do problema;
2. tamanhos: a do conjunto de alunos, u do conjunto de universidades, e k do tamanho máximo das listas de preferências dos alunos;
3. a linhas com: nota e lista de preferências de cada aluno;
4. u linhas com: número de vagas de cada universidade.

Cada aluno é representado por um identificador único (id), assim como cada universidade. Cada id é um valor inteiro entre 1 e a (para alunos), e entre 1 e u (para universidades). A lista de preferências dos alunos deve estar ordenada por preferência (da mais alta para a mais baixa).

Exemplo de entrada (arquivo *input.txt*):

1 // número de instâncias
4 2 2 // número a de alunos, número u de universidades e valor k para tamanho máximo de lista de preferências
90 1 2 // aluno de $id = 1$ tem nota 90 e prefere universidade de $id = 1$, seguido da universidade de $id = 2$
85 2 // aluno de $id = 2$ tem nota 85 e prefere universidade de $id = 2$
80 1 // aluno de $id = 3$ tem nota 80 e prefere universidade de $id = 1$
100 2 1 // aluno de $id = 4$ tem nota 100 e prefere universidade de $id = 2$, seguido da universidade de $id = 1$
1 // número de vagas da universidade de $id = 1$
2 // número de vagas da universidade de $id = 2$

A saída do programa deve ser impressa no arquivo *output.txt* e deve conter as seguintes informações, na ordem indicada abaixo por heurística:

1. alocação de alunos para cada universidade por ordem de nota;

2. conjunto de métricas (precisão de três casas decimais).

Exemplo de saída genérico (arquivo *output.txt*):

```
#Heuristica1:
1: 1 // universidade de id = 1 aprovou o aluno de id = 1
2: 4 2 // universidade de id = 2 aprovou o aluno de id = 4, seguido do aluno de id = 2
100.000% 100.000% // média e mediana do percentual de vagas preenchidas
1.000 1.000 // média e mediana da satisfação dos alunos aprovados
91.250 91.250 // média e mediana da satisfação das universidades
87.500 87.500 // média e mediana das notas de cortes das universidades
#Heuristica2:
1: 1
2: 4 2
100.000% 100.000%
1.000 1.000
91.250 91.250
87.500 87.500
```

Os comentários nos exemplos de entrada e saída foram apresentados apenas por propósitos didáticos, logo eles **NÃO DEVEM** ser incluídos nos arquivos para submissão. Além disso, entrada e saída padrão devem seguir rigorosamente o formato descrito, inclusive no caso de múltiplas instâncias. Sendo assim, em ambos os arquivos, dados de diferentes instâncias devem ser apresentados na mesma ordem e em sequência (na linha seguinte do último dado da instância anterior). Instâncias distintas para o SMIP devem ser geradas pelo próprio aluno para testar e avaliar seu algoritmo.

2 O que deve ser entregue:

2.1 Documentação: deve abranger pelo menos os seguintes pontos

- Introdução do problema apresentado.
- Modelagem e solução do problema: explique as heurísticas utilizadas para resolver o SMIP, tais como suas respectivas complexidades de tempo e espaço.
- Principais decisões de implementação.
- Análise comparativa entre as heurísticas.
- A documentação não pode exceder 10 páginas.

2.2 Código:

- O código fonte do trabalho deve ser submetido para compilação e execução em ambiente Linux, tendo como padrão os computadores dos laboratórios de graduação do DCC;
- Deve ser **obrigatoriamente** escrito na linguagem C (trabalhos implementados em outras linguagens como C++/Java/Python e outras **não** serão aceitos);
- As estruturas de dados devem ser alocadas dinamicamente e o código deve ser modularizado (ou seja, dividido em múltiplos arquivos fonte e fazendo uso de arquivos cabeçalho - .h);
- O utilitário Make deve ser utilizado para compilar o programa **(o arquivo de makefile deve ser submetido juntamente com o código fonte)**;
- A saída deve ser impressa seguindo estritamente o formato da especificação, caso contrário o resultado será considerado errado;
- O arquivo executável deve ser chamado de **tp1**. Não serão aceitos outros nomes de executáveis além do mencionado;
- Faça seu código de forma legível.

2.3 Entrega:

- Data de entrega: 09/04/2012
- Submissão: a documentação e o código do trabalho devem ser submetidos ao *minha.ufmg*. Para isso, compacte os dois (formato *tp1_NomeSobrenome.zip*) e faça a submissão. Teste seu arquivo compactado antes de enviá-lo.
- Apenas a documentação deve ser entregue impressa na secretaria do DCC. Não coloque nos escaninhos dos professores, entregue para a secretária para que sua documentação seja colocada no envelope de AEDS3. A documentação impressa pode ser entregue no dia útil seguinte à submissão digital. **Trabalhos que não tiverem a documentação entregue na secretaria, dentro do prazo de entrega, receberão nota 0.**
- Será postada uma planilha no Moodle sobre a entrevista do trabalho, leia-a e siga as orientações para o agendamento da sua entrevista.
- Será adotado média harmônica entre a pontuação obtida na execução e na documentação do TP, o que implica em valor zero caso alguma das partes não seja apresentada.
- A política para desconto por atraso de entrega do trabalho prático considera a fórmula:

  $$2^{d-1}/0.32\%$$

  onde d é o atraso em dias úteis. Note que após 5 dias úteis, o trabalho não pode ser mais entregue.

Referências

Projeto de Algoritmos com implementação em Pascal e C. Nívio Ziviani - *http://www.dcc.ufmg.br/algoritmos/*.

Introduction to Algorithms. T. Cormen, C. Leiserson, R. Rivest, and C. Stein. The MIT Press, 3rd edition, 2009.

Introdução à Estatística. TRIOLA, Mário F. LTC. 10a. edição, 2008.

Stable Marriage Problem - *http://en.wikipedia.org/wiki/Stable_marriage_problem*.

Site interativo do SMP - *http://mathsite.math.berkeley.edu/smp/smp.html*.