INSTITUTO FEDERAL DE MINAS GERAIS

Bacharelado em Ciência da Computação

Disciplina: Matemática Discreta

Trabalho Prático 02

Professor: Diego Mello da Silva

Formiga-MG
17 de fevereiro de 2014

Sumário

1 Informações Gerais

Este documento descreve a especificação do Trabalho Prático 02 da disciplina Matemática Discreta, e deve ser seguido de forma a contemplar os itens considerados na avaliação por parte do professor da disciplina. O trabalho deve ser feito em grupo de até 2 alunos e tem o valor de 10 pontos.

Em relação ao prazo o trabalho prático deverá ser realizado em 3 semanas (21 dias) contados a partir da data da publicação do mesmo em sala de aula pelo professor. O trabalho deverá ser entregue até as 23:59 hs da data limite. Trabalhos entregues após este prazo serão desconsiderados (isto é, valerão zero).

O documento é organizado como segue. Na Seção 2 será apresentado o problema que este trabalho prático pretende resolver; na Seção 3 serão apresentados os requisitos de *software* que, se implementados, permitirão resolver o problema proposto; na Seção 4 serão apresentados os critérios de avaliação do trabalho e respectiva pontuação; na seção 5 serão abordadas algumas considerações importantes sobre o trabalho em equipe e código de conduta.

2 O problema

O presente documento especifica um *software* capaz de processar um arquivo de entrada que especifica um relação R sobre o produto cartesiano $A \times A$, onde A é um conjunto formado por números inteiros na faixa $1, \ldots, N$, com N informado no arquivo. O *software* deverá carregar o arquivo em memória, armazenando seu conteúdo em uma estrutura de dados do tipo matriz para representar os pares ordenados da relação.

O *software* deverá ainda processar a matriz e classificar a relação $R \subseteq A \times A$ em relação à (i) seu tipo (um-para-um, um-para-vários, vários-para-um, vários-para-vários) e (ii) suas propriedades (reflexiva e simétrica).

Uma relação $R \subseteq A \times A$ é do tipo 1-para-1 se os componentes x e y do par ordenado aparecem apenas uma vez em R; ela é do tipo 1-para-N se existe algum componente x do par ordenado aparece mais de uma vez em R; ela é N-para-1 se algum componente y aparece mais de uma vez em R; por fim, ela é N-para-N se existem componentes x e y do par ordenado que aparecem mais de uma vez em R. A Figura 1 apresenta um diagrama com os tipos de relações mencionadas.

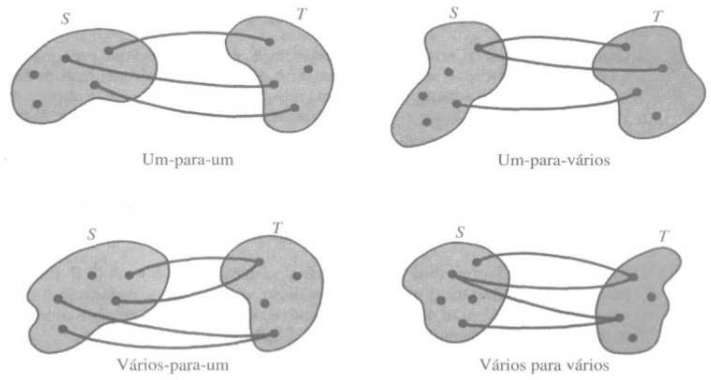

Figura 1: Tipos de Relações

Uma relação R em um conjunto A é chamada de **reflexiva** se $(x, x) \in R$ para todo elemento $x \in A$, ou seja, $\forall x \in A\big((x, x) \in R\big)$. Em outras palavras, R é reflexiva se contiver todos os pares do tipo (x, x), onde x é elemento do domínio A. Uma relação R em um conjunto A é chamada de **simétrica** se $(y, x) \in R$ sempre que $(x, y) \in R$, ou seja, $\forall x \forall y \ \big((x, y) \in R \rightarrow (y, x) \in R\big)$. Novamente, uma relação será simétrica desde que, para cada par ordenado (x, y) existente em R também exista o par ordenado (y, x) correspondente.

Fundamentos teóricos mais profundos sobre esta classificação devem ser pesquisados a parte deste documento, na bibliografia básica da disciplina ou em qualquer outra fonte de pesquisa confiável. É parte do trabalho e responsabilidade do grupo buscar informações sobre tais tipos e propriedades.

Para exemplificar, seja a relação $R \subseteq A \times A$ onde $A = \{1, 2, 3, 4\}$. Se a relação R contiver os pares ordenados:

$$R = \{(1, 1), (3, 2), (2, 2), (1, 3), (4, 4), (3, 3), (2, 3), (2, 1), (3, 1), (1, 2)\}$$

então o classificador deverá retornar como resposta que a relação R é:

- N-Para-N
- Reflexiva
- Simétrica

A próxima seção apresentará a especificação técnica do *software* classificador proposto neste trabalho, destacando os requisitos que ele deverá cumprir para completar o trabalho prático.

3 Especificação

Esta seção apresenta uma especificação técnica de elementos que devem ser implementados pelo grupo para atender ao *software* classificador especificado neste trabalho. A aplicação deverá ser escrita em linguagem `C` ou `Free Pascal`, usando os compiladores `gcc` para linguagem `C` ou `fpc` para a linguagem `Free Pascal`. A aplicação será do tipo console (linha de comando), sendo que toda a entrada e saída de dados ocorrerá mediante uso de teclado e monitor.

O código fonte da aplicação será compilado no ambiente operacional Linux, distribuição Ubuntu. Certifique-se que o seu código fonte é compatível com tal ambiente. Códigos que não compilarem no ambiente mencionado serão desconsiderados. Toda a implementação deverá ser entregue em um único arquivo de nome `classificador.c` se implementado em `C`, ou `classificador.pas` se implementado em `Free Pascal`. **O código fonte deverá possuir um cabeçalho em comentários contendo os nomes e matrículas dos integrantes do grupo.** O trabalho deverá ser enviado ao professor em um único arquivo compactado denominado `discreta-pratico-02.zip` contendo (i) o código fonte da aplicação classificadora e (ii) pelo menos 5 arquivos de entrada contendo relações numéricas a classificar.

As próximas sub-seções detalham cada requisito da aplicação.

3.1 Requisito: Entrada de Dados

Conforme mencionado, o aplicativo classificador será escrito como uma aplicação console, onde entrada de dados e saída de dados ocorrem via terminal e teclado. O aplicativo deverá possuir a seguinte sintaxe, após compilado:

```
user@machine$ ./classificador.bin <arquivo-entrada>
```

onde `<arquivo-entrada>` corresponde ao nome de um arquivo de entrada no formato ASCII informado pelo usuário no terminal que conterá os pares pertinentes à relação, no seguinte formato, sendo cada linha iniciada por um caracter de controle:

```
<n> <cardinalidade>
<r> <x_1> <y_1>
<r> <x_2> <y_2>
<r> <x_3> <y_3>
<r> <x_4> <y_4>
<r> <x_5> <y_5>
...
<r> <x_i> <y_i>
<f>
```

onde

- `n`: Caracter. Indica que nesta linha será informada a cardinalidade do conjunto domínio A.
- `cardinalidade`: Inteiro. Informa a dimensão do conjunto $A = \{1, \ldots, n\}$. A matriz binária que armazenará os pares da relação terá como dimensão este valor informado em arquivo.
- `r`: Caracter. Indica que nesta linha serão informados dois elementos x e y de um par ordenado $(x, y) \in R$
- `x_i`: Inteiro. Informa o 1^o elemento do i-ésimo par ordenado $(x_i, y_i) \in R$, com $i \leq n$
- `y_i`: Inteiro. Informa o 2^o elemento do i-ésimo par ordenado $(x_i, y_i) \in R$, com $i \leq n$
- `f`: Caracter. Informa que o arquivo de entrada chegou ao fim. Deve estar na última linha do arquivo de entrada.

Para exemplificar, considere a relação

$$R = \{(1,1), (3,2), (2,2), (1,3), (4,4), (3,3), (2,3), (2,1), (3,1), (1,2)\}$$

já descrita neste documento. Se desejarmos classificar esta relação usando o aplicativo classificador proposto neste trabalho então deveríamos criar um arquivo de entrada no formato ASCII com o seguinte conteúdo:

```
n 4
r 1 1
r 3 2
r 2 2
r 1 3
r 4 4
r 3 3
r 2 3
r 2 1
r 3 1
r 1 2
f
```

Suponha que o arquivo foi salvo com o nome `relacao01.txt`. Para classificá-lo deveríamos usar a seguinte linha de comando no aplicativo classificador compilado:

```
user@machine$ ./classificador.bin relacao01.txt
```

Observe que o nome do arquivo é informado como argumento de linha de comando, e não mediante digitação pelo usuário. A leitura de argumentos por linha de comando deverá ser pesquisada pelo grupo, mas sugere-se buscar pelas funções do `Free Pascal` denominadas `ParamStr()` e `ParamCount()`. A leitura de caracteres e inteiros de arquivos texto (ASCII) também devem ser pesquisadas a parte pelo grupo para conclusão deste requisito. Sugere-se procurar compreender o funcionamento das funções `Assign()`, `Reset()`, `Read()` e `Close()`.

O grupo deverá pensar em situações problemas que devem ser tratadas na entrada de dados. Caso alguma delas ocorra a aplicação deverá ser abortada, exibindo uma mensagem de erro para o usuário no terminal. A entrada de dados deverá ser implementada em uma função auxiliar que percorrerá o arquivo lendo seu conteúdo.

3.2 Requisito: Carregamento da Matriz em Memória

Neste requisito pede-se que o aplicativo guarde em memória os pares contidos na relação R descrita no arquivo de entrada em memória.

Para tal sugere-se usar uma estrutura de dados do tipo matricial, contendo valores booleanos que representam se dois elementos quaisquer da matriz estão ou não relacionados segundo R. A interpretação e gravação dos dados deverá seguir a convenção de que as linhas quando percorridas referem-se aos elementos x do par ordenado $(x, y) \in R$, e que as colunas quando percorridas referem-se aos elementos y do mesmo par ordenado. Os elementos das linhas e colunas devem ser rotulados de forma sequencial, partindo de 1 até n. Para exemplificar, a relação

$$R = \{(1,1), (3,2), (2,2), (1,3), (4,4), (3,3), (2,3), (2,1), (3,1), (1,2)\}$$

seria representada matricialmente por

$$R = \begin{bmatrix} 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

3.3 Requisito: Classificador 1-Para-N

Neste requisito deve-se implementar uma função computacional que informa se a relação R carregada em memória é do tipo 1-Para-N. Será usada para gerar o resultado no requisito de saída de dados. Observe as características da matriz de exemplo descrita na Sub-seção 3.2 para projetar este algoritmo classificador.

3.4 Requisito: Classificador N-Para-1

Neste requisito deve-se implementar uma função computacional que informa se a relação R carregada em memória é do tipo N-Para-1. Será usada para gerar o resultado no requisito de saída de dados. Observe as características da matriz de exemplo descrita na Sub-seção 3.2 para projetar este algoritmo classificador.

3.5 Requisito: Classificador N-Para-N

Neste requisito deve-se implementar uma função computacional que informa se a relação R carregada em memória é do tipo N-Para-N. Será usada para gerar o resultado no requisito de saída de dados. Observe as características da matriz de exemplo descrita na Sub-seção 3.2 para projetar este algoritmo classificador.

3.6 Requisito: Classificador 1-Para-1

Neste requisito deve-se implementar uma função computacional que informa se a relação R carregada em memória é do tipo 1-Para-1. Será usada para gerar o resultado no requisito de saída de dados. Observe as características da matriz de exemplo descrita na Sub-seção 3.2 para projetar este algoritmo classificador.

3.7 Requisito: Classificador Prop. Reflexiva

Neste requisito deve-se implementar uma função computacional que informa se a relação R carregada em memória é uma relação reflexiva. Será usada para gerar o resultado no requisito de saída de dados. Observe as características da matriz de exemplo descrita na Sub-seção 3.2 para projetar este algoritmo classificador.

3.8 Requisito: Classificador Prop. Simétrica

Neste requisito deve-se implementar uma função computacional que informa se a relação R carregada em memória é uma relação simétrica. Será usada para gerar o resultado no requisito de saída de dados. Observe as características da matriz de exemplo descrita na Sub-seção 3.2 para projetar este algoritmo classificador.

3.9 Requisito: Saída de Dados

Este requisito especifica o formato da saída de dados. Uma vez que o aplicativo tenha sido executado sobre um arquivo de entrada contendo uma relação válida haverá a classificação da relação quanto ao tipo e propriedades especificadas no escopo deste documento. Desta forma, neste requisito pede-se que a saída de dados seja dada no

terminal, em um formato similar ao apresentado a seguir, para o exemplo de arquivo denominado `relacao01.txt`:

```
user@machine$ ./classificador relacao01.txt

Classificador
=============

1) Pares da Relacao:
    (1,1)
    (3,2)
    (2,2)
    (1,3)
    (4,4)
    (3,3)
    (2,3)
    (2,1)
    (3,1)
    (1,2)

2) Classificacao quanto ao tipo:

    N-Para-1

3) Classificacao quanto a propriedades:

    Simetrica
    Reflexiva

Fim de Processamento.
```

A saída de dados poderá ser implementada na função principal do programa, usando as funções classificadoras auxiliares detalhadas nos requisitos anteriores.

3.10 Requisito: Documentação de Código

A documentação de código é importante em qualquer implementação computacional e será considerada neste trabalho. Pede-se que o arquivo que contem o código fonte possua ao menos (i) cabeçalho inicial, contendo nome do aplicativo, membros do grupo com nome e matrícula, instruções de compilação, ambiente de desenvolvimento, data e objetivo do arquivo; (ii) cabeçalho das funções auxiliares e procedimentos implementados no trabalho; (iii) comentários nos principais trechos de código de cada algoritmo, explicando resumidamente o que está sendo codificado a seguir.

3.11 Requisito: Corretude dos Resultados

O requisito mais importante do trabalho é aquele que lida com resultados corretos. Desta forma, é um requisito essencial do trabalho que além do classificador ser

implementado respeitando-se os requisitos anteriores é importante que ele retorne resultados de qualidade. Em virtude disso o grupo deverá testar exaustivamente o classificador, gerando arquivos de entrada para diferentes relações. Os arquivos de entrada usados no teste devem ser compactados junto com o código fonte e enviados para o professor da disciplina no e-mail `diego.silva@ifmg.edu.br` até a data limite do trabalho.

É importante que o grupo faça testes completos com o classificador. O professor adotará seu próprio conjunto de relações R durante a correção.

4 Barema de Correção

Conforme mencionado, o trabalho prático tem valor de 10 pontos. A correção seguirá o barema apresentado a seguir, que lista os requisitos do trabalho prático e suas respectivas pontuações.

Requisito	**Pontos**
Entrada de Dados	0.5 pto
Carregamento da Matriz em Memória	1.5 pto
Classificador 1-Para-N	1.0 pto
Classificador N-Para-1	1.0 pto
Classificador N-Para-N	0.5 pto
Classificador 1-Para-1	0.5 pto
Classificador Prop. Reflexiva	1.0 pto
Classificador Prop. Simétrica	1.0 pto
Saída de Dados	0.5 pto
Documentação do Código	0.5 pto
Corretude dos Resultados	2.0 ptos
TOTAL	**10.0 ptos**

5 Considerações Finais

O trabalho prático especificado neste documento procura desenvolver habilidades técnicas e interpessoais entre os membros da equipe de maneira saudável. Para tal, é fundamental que o trabalho seja feito de maneira ética e profissional. Desta forma algumas considerações finais devem ser feitas:

- Este é um trabalho de programação, portanto aqueles que possuem dificuldades devem estudar o assunto, procurando livros, tutoriais e outros materiais que complementem sua deficiência em programação por conta própria;

- Trabalhos plagiados ou feitos por terceiros valerão zero;

- Todos os membros da equipe deverão participar efetivamente da implementação do trabalho de forma a aumentar suas perícias em matemática e programação;

- Caso o professor julgue necessário poderá fazer uma arguição dos membros de um ou mais grupos;

- Trabalhos entregues fora do prazo serão desconsiderados;

- Dúvidas sobre o trabalho devem ser tiradas com no máximo 1 semana do prazo de entrega.