

Estudo Dirigido - Parte 3 - Teoria dos Grafos (BCC204)

Marco Antonio M. Carvalho
Universidade Federal de Ouro Preto
Departamento de Computação

16 de agosto de 2021

Instruções

- Siga atentamente quanto ao formato da entrada e saída de seu programa, exemplificados no enunciado;
- Na plataforma run.codes, os programas serão submetidos a vários casos de testes, com características variadas;
- A avaliação do run.codes considera o tempo de execução e o percentual de respostas corretas;
- Considere que os dados serão fornecidos pela entrada padrão. Não utilize abertura de arquivos pelo seu programa. Se necessário, utilize o redirecionamento de entrada;
- Os códigos fonte serão submetidos a uma ferramenta de detecção de plágios em software;
- Códigos cuja autoria não seja do aluno, com alto nível de similaridade em relação a outros trabalhos, ou que não puder ser explicado, acarretará na perda da nota e frequência;
- Códigos ou funções prontas específicos de algoritmos para solução dos problemas elencados não são aceitos;
- Não serão considerados algoritmos parcialmente implementados.
- Serão realizadas entrevistas sobre os estudos dirigidos para complementar a avaliação;
- A avaliação do estudo dirigido consiste de: (I) avaliação do run.codes; (II) verificação de plágio; (III) verificação de aderência ao enunciado e à estrutura obrigatória dos códigos; e (IV) entrevista. As quatro etapas da avaliação são eliminatórias.

1 Algoritmo de Dijkstra

Dijkstra apresentou em 1959 um algoritmo de rotulação de vértices para encontrar os caminhos mais curtos de uma origem para todos os demais vértices em grafos com custos positivos nos arcos. Neste estudo dirigido é pedido ao aluno que implemente este algoritmo.

Especificação da Entrada

A primeira linha da entrada contém quatro inteiros n , m , b e i , indicando a quantidade de vértices, a quantidade de arestas/arcos, um valor binário indicando se o grafo é direcionado (valor 1) ou não (valor 0) e um índice do vértice (enumerados de 1 a n) a partir do qual será executado o algoritmo.

Em seguida haverá m linhas, cada uma contendo três inteiros, indicando o vértice de origem (enumerados de 1 a n), o vértice de destino e o peso das arestas/arcos.

Especificação da Saída

Após executar o algoritmo, imprima o vértice de destino, o comprimento dos caminhos mais curtos entre parênteses e a estrutura do caminho do vértice i para cada um dos vértices do grafo, usando uma linha para cada destino. Cada linha da saída deve ser terminada com '\n'.

Exemplo de Entrada

```
5 7 1 1
1 2 2
1 3 6
1 4 7
2 4 3
2 5 6
3 5 1
4 5 5
```

Exemplo de Saída

```
2 (2) : 1 2
3 (6) : 1 3
4 (5) : 1 2 4
5 (7) : 1 3 5
```

Estrutura do código

O código-fonte deve ser modularizado corretamente em três arquivos: `principal.c`, `principal.cpp`, `dijkstra.h`, `dijkstra.cpp` e `dijkstra.h`. O arquivo `principal.c` deve apenas invocar as funções e procedimentos definidos no arquivo `dijkstra.h`. A separação das operações em funções e procedimentos está a cargo do aluno, porém, não deve haver acúmulo de operações dentro uma mesma função/procedimento. A implementação deve se basear **obrigatoriamente** no código da BFS desenvolvido na parte anterior do estudo dirigido.

Diretivas de Compilação C

```
$ gcc dijkstra.c -c
$ gcc principal.c -c
$ gcc dijkstra.o principal.o -o programa
```

Diretivas de Compilação C++

```
$ g++ dijkstra.cpp -c  
$ g++ principal.cpp -c  
$ g++ dijkstra.o principal.o -o programa
```