Teoria dos Grafos – COS 242

2020/2

Trabalho de Disciplina – Parte 1

1 Objetivos

O objetivo deste trabalho de disciplina é projetar e desenvolver uma biblioteca para manipular grafos. A biblioteca deverá ser capaz de representar grafos assim como implementar um conjunto de algoritmos em grafos. Para fins deste trabalho, a biblioteca pode ser um conjunto de funções (em C, por exemplo) ou uma classe em uma linguagem orientada a objetos (em Java, por exemplo). Você deve projetar e desenvolver sua biblioteca de forma que ela possa ser facilmente utilizada em outros programas.

2 Logística

O trabalho da disciplina está divido em duas partes, sendo esta a primeira. Cada parte corresponde a uma ou mais funcionalidades que deverão ser projetadas, desenvolvidas e incorporadas à biblioteca. O trabalho deve ser realizado em dupla que deve permanecer a mesma por todo o trabalho. Cada parte terá um prazo de entrega e será avaliada de maneira independente. Para entregar uma parte do trabalho, você irá preparar um relatório informando suas decisões de projeto e implementação das funcionalidades. Além disso, seu relatório deve responder as perguntas relacionadas aos estudos de caso. Este relatório deve ter no **máximo 4 páginas** e deve conter a URL para o código fonte da biblioteca e do programa utilizado para realizar os estudos de caso (ex. github).

3 Descrição – Parte 1

Segue abaixo as funcionalidades que precisam ser oferecidas pela biblioteca nesta parte do trabalho. Sua biblioteca irá trabalhar apenas com grafos não-direcionados.

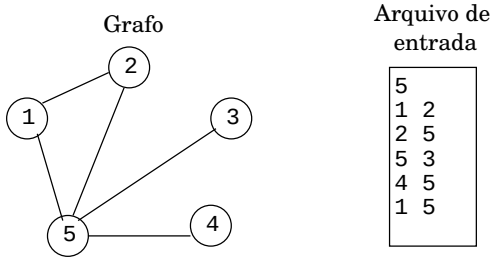

Figura 1: Exemplo de grafo e formato de arquivo de entrada.

1. **Entrada**. Sua biblioteca deve ser capaz de ler um grafo de um arquivo texto. O formato do grafo no arquivo será o seguinte. A primeira linha informa o número de vértices do grafo. Cada linha subsequente informa as arestas. Um exemplo de um grafo e seu respectivo arquivo texto é dado na figura 1.

2. **Saída**. Sua biblioteca deve ser capaz de gerar um arquivo texto com as seguintes informações sobre o grafo: número de vértices, número de arestas, grau mínimo, grau máximo, grau médio, e mediana de grau. Além disso, imprimir informações sobre as componentes conexas (ver abaixo).

3. **Representação de grafos**. Sua biblioteca deve ser capaz de representar grafos utilizando tanto uma matriz de adjacência, quanto uma lista ou vetor de adjacência. O usuário da biblioteca (programa que irá usá-la) poderá escolher a representação a ser utilizada.

4. **Busca em grafos: largura e profundidade**. Sua biblioteca deve ser capaz de percorrer o grafo utilizando busca em largura e busca em profundidade. O vértice inicial será dado pelo usuário da biblioteca. A respectiva árvore de busca deve ser gerada assim como o nível de cada vértice na árvore (nível da raiz é zero). Estas informações devem ser impressas em um arquivo. Para descrever a árvore gerada, basta informar o pai de cada vértice e seu nível no arquivo de saída.

5. **Distâncias e diâmetro**. Sua biblioteca deve ser capaz de determinar a distância entre dois vértices do grafo (utilizando como primitiva a BFS) assim como calcular o diâmetro do grafo. Lembrando que o diâmetro é a maior distância entre qualquer par de vértices do grafo (ou seja, o comprimento do maior caminho mínimo do grafo).

6. **Componentes conexos**. Sua biblioteca deve ser capaz descobrir as componentes conexas de um grafo. O número de componentes conexos, assim como o tamanho (em vértices) de cada componente e a lista de vértices pertencentes à componente. Os componentes devem estar listados em ordem decrescente de tamanho (listar primeiro o componente com o maior número de vértices, etc).

4 Estudos de Caso

Considerando cada um dos grafos indicados no website da disciplina, responda às perguntas abaixo:

1. Compare o desempenho em termos de quantidade de memória utilizada pelas duas representações do grafo. Ou seja, determine a quantidade de memória (em MB) utilizada pelo seu programa quando você representa o grafo utilizando uma matriz de adjacência e lista de adjacência. Dica: pause a execução do programa depois de carregar o grafo e verifique a memória sendo utilizada pelo processo.

2. Compare o desempenho em termos de tempo de execução das duas representações do grafo. Execute **1000** buscas em largura em cada um dos casos (utilize diferentes vértices como ponto de partida da busca), e obtenha o tempo médio de uma busca. Dica: obtenha o tempo do relógio da máquina no seu código antes de iniciar e depois de terminar as 1000 buscas.

3. Repita o item anterior para busca em profundidade (utilize os mesmos 1000 vértices iniciais). Faça uma tabela comparando os tempos de busca nos diferentes grafos.

4. Determine o pai dos vértices 10, 20, 30 na árvore geradora induzida pela BFS e pela DFS quando iniciamos a busca nos vértices 1, 2, 3.

5. Determine a distância entre os seguintes pares de vértices (10,20), (10,30), (20,30).

6. Obtenha as componentes conexas do grafo. Quantas componentes conexas tem o grafo? Qual é o tamanho da maior e da menor componente conexa?

7. Determine o diâmetro do grafo. Determine também o tempo que sua biblioteca levou para fazer este cálculo.

Você deve preparar tabelas com os resultados obtidos onde as colunas representam as características e as linhas representam os diferentes grafos analisados. Inclua esta tabela em seu relatório.

Importante: Todas as medidas de tempo devem ignorar o tempo gasto lendo o grafo do disco e o tempo gasto escrevendo o resultado no disco ou monitor. Ou seja, contabilize apenas o tempo de execução do algoritmo. Consulte o relógio da máquina antes do algoritmo iniciar e depois dele terminar, e use a diferença para obter o tempo decorrido.