Universidade Federal de Santa Catarina
Departamento tecnológico

Relatório Atividade 1 de Grafos

Aluno: Pedro Augusto Costa

20 de Abril
2025

Conteúdo

1 Resumo

Para esse trabalho, foi utilizado Python para implementar os algoritmos, além das anotações da disciplina providenciados pelo professor vigente, Rafael de Santiago.

2 Questão 1: Representação

A primeira questão, a qual é a base de todas as questões de todos os trabalhos, uma estrutura para criar e representar grafos não dirigidos e ponderados. Foi utilizado uma estrutura de classe e pelo método de lista de listas de adjacências, e sua implementação foi feita por meio de dicionários aninhados, utilizando operações em cima da lista gerada para os métodos solicitados.

3 Questão 2: Busca em Largura

Nessa questão, foi implementado um algoritmo de busca em largura a partir de um vértice inicial (s) do grafo, utilizando o algoritmo disponibilizado na apostila da matéria. Foram criadas três listas: Cv, que guardas os vértices visitados durante a busca, Dv, que guarda o número de arestas de distância entre (s) e um vértice (v) e Av, os antecessores de cada vértice visitados na busca. Uma fila (Q) também foi usada, que coloca e remove vértices conforme eles vão sendo visitados, para manter o controle de quais vértices já foram visitados.

4 Questão 3: Ciclo Euleriano

A primeira etapa, que consiste em informar ao usuário se um ciclo euleriano existe ou não, é feita verificando o grau de todos os vértices, se o grafo possuir pelo menos um vértice com grau ímpar, então ele não possui um ciclo euleriano.

Com essa informação em mente, a segunda parte consiste em, utilizando duas pilhas (chamadas de "S"e "T"), sendo "S"para armazenar os vértices visitados a partir de uma origem, e "T", que armazena a ordem correta para o ciclo, além de uma lista, utilizada para armazenar todas as arestas representadas por uma tupla que contenha os dois vértices pertencentes a essa aresta. Essa lista, é atualizada repetidamente por meio de um laço de repetição até que esteja completa, e "S"vazia. Por fim, o último passo é outro laço de repetição que se repete o número de vezes igual ao número de vizinhos que o vértice de origem tem, colocando cada vizinho na lista caso a aresta desse vizinho não foi visitada, o vizinho entra na pilha "S"e se torna o vértice atual, caso uma aresta já tenha sido visitada, o vértice é removido da pilha "S", e o processo se repete até completar as todas as arestas, e a pilha "T", agora com o caminho correto do ciclo, é exibida como resposta.

5 Questão 4: Algoritmo de Bellman-Ford ou Dijkstra

O algoritmo de Bellman-Ford foi escolhido, visto que Dijkstra não consegue lidar com ciclos negativos. Duas estruturas de lista foram usadas, "D"que representa o peso das arestas do vértice de origem "s"para qualquer outro vértice, e "A"que marca os antecessores de cada vértice no caminho mínimo encontrado. Aplicando ∞ para cada vértice do grafo presente em "D", o algoritmo então "relaxa", verificando cada vértice vizinho, e conferindo se a distância até o respectivo vizinho é maior que a distância até o respectivo vértice somado ao peso da aresta que leva até o respectivo vizinho. Esse algoritmo também verifica a existência de ciclos negativos, verificando se, após notar a possibilidade de relaxamentos consecutivos entre vizinhos, ele entende como um ciclo negativo.

6 Questão 5: Algoritmo de Floyd-Warshall

A primeira etapa consiste em criar uma Matriz quadrada, com linhas e colunas sendo constituídas de cada um dos vértices do grafo, e cada par linha, coluna possuindo como valor a distância entre os vértices da origem (linha) e do destino (coluna). Os valores desconhecidos (caso a distância entre 2 vértices seja de pelo menos composta de duas arestas) o valor de ∞ é colocado até que o valor mínimo real seja descoberto. Após isso, uma repetição será feita, igual ao número de vértices, onde uma segunda matriz será criada, mantendo todos os valores originais, com exceção dos casos onde o valor mínimo de distância entre o par de vértices for menos do que o valor representado na matriz anterior, essa nova matriz então se torna o modelo que vai ser reutilizado na próxima iteração, garantindo que a cada repetição, a matriz esteja mais próxima de exibir a distância mínima para qualquer par de vértices, por fim, é exibido ao usuário, o caminho de valor mínimo encontrado do vértice de origem "s"para cada outro vértice, e a distância desse caminho.