INE5413: Atividade 1

Vitor Della Torre dos Santos

Março, 2021

1 Introdução

A atividade 1 é a primeira atividade da disciplina INE5413, de Grafos. Seu intuito era realizar a implementação de uma biblioteca de grafos, a qual devia incluir uma classe representativa para estes (que seriam abstraídos a partir da leitura de arquivos `.net`), e de, pelo menos, 3 algoritmos, sendo o quarto opcional. Dentre os 3 selecionados estão:

1. O algoritmo da Busca em Largura (*Breadth First Search*);
2. O algoritmo de Dijkstra; e
3. O algoritmo de Floyd-Warshall

Os algoritmos enumerados serão descritos a partir da seção 3, visto que a seção 2 destina-se à descrição das funções implementadas na classe `Graph`.

2 API: a classe `Graph`

Como boa prática de desenvolvimento de software, fez-se uma classe *singleton* `GraphBuilder`. Esta era responsável por providenciar um grafo, quando requisitada. A classe `Graph` é sua classe privada e um objeto desta só é retornado quando passa-se um arquivo-texto com os moldes similares aos das instâncias de teste providenciadas..

Construiu-se um grafo fazendo uso, principalmente, de uma matriz de adjacência, mas, também, por meio de um dicionário. Com a dada matriz, cada vértice do grafo serviria de índice para sua indexação, enquanto o dicionário

seria utilizado somente para armazenar um dado rótulo que um vértice poderia receber. Ambas as estruturas de dados são efetivamente inicializadas após o *parsing* de um arquivo-texto.

3 Busca em Largura

Primeiro algoritmo implementado no trabalho, a Busca em Largura, juntamente com os demais algoritmos, foi separada numa outra classe: `Operator`, a qual também é um *singleton*.

Fez-se uso de listas, visto que seria mais fácil indexá-las a partir do identificador de cada um dos vértices disponibilizados pelo grafo, e criou-se 2 destas, a princípio: uma lista responsável por verificar quais vértices foram visitados e uma lista que é abstraída para uma fila.

Para acompanhá-las, criou-se, em conjunto, um dicionário, o qual indica em quais níveis estão os outros vértices a partir da perspectiva de um dado vértice passado como parâmetro da função. Ou seja, tomando-se, por exemplo, um vértice com identificador equivalente a 0, teria-se, "à distância 0", este próprio vértice, enquanto, "à distância 1", seus vizinhos, distantes do nodo inicial por uma única aresta.

O uso do dicionário também auxilia a reforçar o verdadeiro poder do algoritmo da Busca em Largura, que é o de achar um caminho entre dois vértices com o menor número de arestas.

4 Dijkstra

Soando como uma contra-parte da Busca em Largura, o Algoritmo de Dijkstra quer, também, achar o caminho mínimo entre dois vértices, mas realiza sua busca usando o peso entre as entidades como norteador principal, ao contrário da Busca em Largura, a qual busca pela menor quantidade de arestas sem que se leve em consideração o custo destas.

Para implementação, novamente utilizaram-se listas, as quais representam: os vértices já visitados, a distância dos vértices em relação ao nodo inicial (passado por parâmetro da função) e uma lista que, novamente, foi abstraída à uma fila.

Inicializa-se a lista de distâncias com todos os valores equivalentes ao limite superior do tipo numérico estipulado para a matriz de adjacência;

neste projeto, esta representa uma lista de listas de *floats* e, portanto, a lista representativa das distâncias é, também, uma lista de *floats*, a qual recebeu `Float.MAX_VALUE` na inicialização.

O algoritmo de Dijkstra consiste em garantir constante atualização à lista que mapeia as distâncias de um vértice aos outros. Desta forma, pode-se, por exemplo, encontrar um caminho que tenha o custo menor entre os pontos A e B ao invés de efetivamente percorrer a aresta que separa ambos, mesmo que seja necessário passar por quatro ou mais arestas, supondo-se que o custo entre A e B seja maior do que a soma dos custos das arestas visitadas.

5 Floyd-Warshall

Um algoritmo mais curto e de mais rápida implementação, o Algoritmo de Floyd-Warshall é sedutor quanto à sua simplicidade, mas peca em seu tempo de processamento, diametralmente oposto ao Algoritmo de Dijkstra.

Para Floyd-Warshall, criou-se somente uma matriz-cópia da matriz da adjacência, para se evitar o *overwrite* desta última. A matriz-cópia é a matriz representativa da distância entre todos os vértices do grafo. No caso, diferentemente da lista de distâncias em Dijkstra, aqui tem-se uma matriz, visto que o Algoritmo de Floyd-Warshall trabalha com a indexação excessiva dentro do escopo de seus três laços iterativos. Retorna-se a distância de cada vértice do grafo para um dado vértice `i`, computada dentro de um outro laço iterativo; ou seja, tem-se a distância que separa cada um dos vértices dos demais.

6 Conclusão

Por meio da elaboração do trabalho, foi possível adquirir base teórico-prática para futuras implementações de grafos e estruturas de dados mais complexas. O código-fonte pode ser encontrado no Moodle ou em

https://github.com/vdella/ine-5413