Aluno: Luiz Rodrigo Lacé Rodrigues
DRE: 118049873

Tema: Grafos sem tripla asteroidal

Definição: Três vértices distintos, dois a dois não adjacentes em um grafo formam uma tripla asteroidal quando para quaisquer dois deles existe um caminho que os liga e não passa pela vizinhança do terceiro. Um grafo que não admite tripla asteroidal é chamado grafo sem tripla asteroidal (STA).

Implementação
O nome do arquivo principal se chama AlgGrafos.java .Primeiramente temos a classe “LeitorDeArquivo” que fará a leitura do arquivo, direcionando o caminho para a pasta “myfiles” Essa classe possui o método “abreArquivo” que vai retornar linha a linha de do arquivo passado no parâmetro.

Depois temos a classe Grafo. Um dos seus principais atributos é o “verticesAjacentes”, que será um hashmap que a chave será o id do vértice e cada chave estará relacionada com uma lista de vértices adjacentes. A classe também conta com o número de vértices e numero de arestas (verticesCont e arestasCont).

Teremos o construtor do Grafo e os getters e setters de cada atributo. A classe também conta com métodos básicos da teoria dos grafos, como adicionar vértices e arestas. Teremos também o método “closedNeighborhood” que vai nos retornar a vizinhança fechada do vértice passado como parâmetro.

Em seguida teremos um método BFS modificado para que sirva ao nosso propósito de encontrar as triplas asteroidais. Nesse método teremos 2 parâmetros (BFS(s, d)), nesse método o que vamos fazer é exatamente o algoritmo básico de BFS, mas com uma alteração básica, antes de encontrar os vértices alcançáveis de s, vamos marcar toda a vizinhança de d, e depois vamos iniciar o algoritmo do BFS, dessa forma para todo vértice novo visitado, ele vai guardar numa lista “alcançavel”. Como marcamos a vizinhança fechada de “d”, antes do BFS começar, então vamos limitar os alcançáveis de s a nosso favor.

Em seguida teremos o método “findTriple”, é nele que vamos retornar ou não uma tripla asteroidal.
Teremos 3 funções “for” onde vamos iterar todas as permutações possíveis de 3 vértices.

Dentro do último for , teremos 3 “ifs” aninhados, cada if, vamos verificar se existe um caminho do vértice 1 para o 3 sem passar pela vizinhança de 2, 2 para o 3 sem passar pelos vizinhos de 1 e 2 para o 1 sem passar pelos vizinhos de 3.

Caso consigamos contemplar esses 3 ifs, então temos uma tripla. Adicionamos o id de cada vértice dessa permutação em uma lista e a retornamos. Caso nenhuma tripa seja encontrada, esse método retorna null.

Em seguida temos o método que vai nos dizer se o grafo passado no arquivo grafo.txt é asteroidal sem tripla ou não. Primeiro ele verifica se “this.findTriple()” vai retornar null, ou seja, nenhuma tripla foi encontrada no método findTriple, logo vai retornar uma mensagem dizendo que o grafo é asteroidal sem tripla. Caso contrário, vai dizer que o grafo não é asteroidal sem tripla e vai retornar a tripla asteroidal explicando que existe um caminho para qualquer dois vértices sem passar pelo vizinho do terceiro.

Em seguida temos o método “is_undirected” que vai nos retornar true ou falso caso o grafo seja não direcionado. E depois temos o método “criaVertice” para ler linha a linha e passar os dados para o HashMap “verticesAdjacentes” do grafo.

Na main, vamos começar lendo o arquivo com nome “grafo.txt” , deois vamos criar um grafo a partir das linhas do arquivo e então printamos o grafo com os ids e seus respectivos vértices adjacentes.

Depois disso vamos passar por um if para verificar se o grafo é não direcionado para que possa verificar se é asteroidal sem tripla. Caso seja direcionado, retornamos uma mensagem falando para inserir um grafo não direcionado.

Quando o grafo é não direcionado, ele pode retornar se é asteroidal sem tripla ou não, caso não seja, ele também retorna a tripla asteroidal que o grafo possui.

Bibliografia:

https://www.sciencedire
ct.com/science/article/pii/S157086670400019X#aep-section-id14
https://doc.sagemath.org/html/en/reference/graphs/sage/graphs/asteroidal_triples.html
https://networkx.org/documentation/networkx-2.4/reference/algorithms/generated/networkx.algorithms.asteroidal.find_asteroidal_triple.html
https://www.graphclasses.org/classes/gc_61.html
https://networkx.org/documentation/networkx-2.4/_modules/networkx/algorithms/asteroidal.html#find_asteroidal_triple
https://sciendo.com/pdf/10.7151/dmgt.1821
https://www.geeksforgeeks.org/connected-components-in-an-undirected-graph/
https://www.sciencedirect.com/science/article/pii/S157086670400019X
https://networkx.org/documentation/networkx-2.4/_modules/index.html
http://www.cs.toronto.edu/~stacho/public/asterorder2d.pdf