Janeiro 2022

RELATÓRIO ESINF

Grupo 42

Manuela Leite - 1200720
Francisco Queiroz - 1201239
Pedro Rocha - 1201382
Rita Ariana Sobral - 1201386

Estruturas de Informação

Professor: José Marinho

isep Instituto Superior de Engenharia do Porto

US401

Sabemos que a centralidade de um vértice de um grafo pode ser definida através de vários critérios, mas para o nosso contexto de trabalho foi nos dito que a centralidade de um porto é definida pelo número de caminhos mais curtos que passam por ele.

Para obter o número de caminhos mais curtos que passam por cada porto, foi utilizado um loop para percorrer todos os vértices da freight network e, para cada vértice, foi chamado o método shortestPaths, pois com este método obtemos todos os caminhos mais curtos desse vértice para todos os restantes vértices do grafo, obtendo tb as respetivas distâncias.

Como o grafo que representa a nossa freight network é orientado o caminho mais curto, por exemplo de Lisboa para França pode ser diferente do caminho mais curto de França para Lisboa, e é por este motivo que realizamos para todos os vértices o raciocínio anterior.

Para ter os portos mais críticas, foi necessário organizar os arrays por ordem crescente. Para isso foi implementado o algoritmo Bubble Sort. No caso de dois portos terem o mesmo número de caminhos mais curtos que passam por ela, o porto que tiver menos distância de todos os caminhos mais curtos é considerado mais crítico que o outro com que estava a ser feita a comparação, este foi o processo de seleção que o grupo achou mais correto dado o contexto do negócio.

Os 5 portos mais críticos, criando a freight network com n=1, são Dunkirk, Rostock, Tallinn, Vlissingen e Zeebrugge.

Usando a medida do tempo (complexidade temporal), concluímos que o algoritmo tem complexidade (V^3E)), sendo V o número de vértices e E o número de arestas.

US402

Nesta US é nos pedido para, sempre que possível retornar os 4 caminhos mais curtos entre dois locais cumprindo alguns requisitos.

O primeiro tipo de caminho é o caminho terrestre. Para obter este caminho primeiro faço clone do grafo da nossa aplicação para depois conseguir remover os locais que são portos, exceto o local de partida e chegada caso algum deles o seja. Depois apenas invoco o método shortestPath.

O segundo tipo de caminho é o caminho marítimo. Para obter este caminho primeiro faço clone do grafo da nossa aplicação para depois conseguir remover os locais que são capitais, e caso o local de partida ou o de chegada sejam capitais já não é possível obter este caminho. Depois apenas invoco o método shortestPath.

O terceiro tipo de caminho é o caminho terrestre. Para obter este caminho apenas invoco o método shortestPath.

Por fim, o último tipo de caminho é o caminho que tem que conter n sítios específicos. Para obter este caminho vou vendo qual é de todos os locais que tenho ainda para percorrer obrigatoriamente o com uma distância mais curta ao local onde me encontro no caminho e adiciono ao circuito. Depois de ter percorrido todos os locais obrigatórios obtenho o caminho mais próximo desse local até ao local de destino.

Usando a medida do tempo (complexidade temporal), concluímos que o algoritmo tem complexidade (V^3E)), sendo V o número de vértices e E o número de arestas.

US402

Pedindo o caminho mais curto entre Lisboa e Roma, tendo o grafo sido criar com n=1, e sendo obrigatório passar em Amsterdam o ficheiro de saída obtido é o seguinte:

--- Caminho ---
Capital Lisbon
Capital Madrid
Capital Paris
Capital Rome
--- Caminho ---
Não foi possível determinar este caminho.
--- Caminho ---
Capital Lisbon
Porto Setubal
Porto Leixoes
Porto Brest
Porto Dunkirk
Capital Paris
Capital Rome
--- Caminho ---
Capital Lisbon
Porto Setubal
Porto Leixoes
Porto Brest
Porto Dunkirk
Porto Zeebrugge
Porto Vlissingen
Capital Amsterdam
Capital Brussels
Capital Paris
Capital Rome

Não é possível determinar o segundo caminho para este caso, pois este só pode conter portos.

US403

O método mostEfficientCircuit consiste num ciclo que só termina quando depois de iniciado o circuito voltarmos ao cértice inicial, ou seja, ao lugar inicial.

Depois consoante o local atual em que estamos a analisar, obtemos os vértices adjacentes a este e vamos obter qual o lugar com distância inferior para ser o próximo local do circuito.

Caso não exista nenhum local para ir a seguir ao local atual, recuamos um local no circuito e analisamos os outros vértices adjacentes a este. E assim sucessivamente até chegarmos ao local inicial.

Para este caso de uso existiam alguma abordagens possíveis sendo que a que achamos que seria a mais eficiente e geraria um circuito mais eficiente foi o raciocinio dos mínimos sucessivos, em que definimos o local de partida e vamos selecionando a mais próxima, sendo que não podemos repetir nenhum local exceto o último que tem que ser igual ao local de partida.

Usando a medida do tempo (complexidade temporal), concluímos que o algoritmo tem complexidade polinomial de grau 3 (O (n^3)).

US403

Pedindo o circuito mais eficiente para Lisboa, tendo o grafo sido criar com n=4 o ficheiro de saída obtido é o seguinte:

Capital Lisbon
Porto Setubal
Porto Leixoes
Porto Brest
Porto Dublin
Capital Dublin
Capital London
Porto London
Porto Dunkirk
Porto Zeebrugge
Porto Vlissingen
Porto Antwerp
Capital Brussels
Capital Amsterdam
Capital Berlin
Porto Rostock
Porto Copenhagen
Porto Aarhus
Porto Lysekil
Porto Bergen
Capital Oslo
Capital Stockholm
Porto Stockholm
Porto Tallinn
Capital Tallinn
Capital Riga
Capital Vilnius
Capital Minsk
Capital Kyiv
Capital Chisinau
Capital Bucharest
Porto Galatz
Porto Constantza
Porto Varna
Porto Bourgas
Porto Ambarli
Porto Mersin
Porto Larnaca
Porto Piraeus
Capital Athens
Capital Skopje
Capital Pristina
Capital Podgorica
Capital Tirana
Porto Vlore
Porto Split
Porto Rijeka
Porto Piran
Porto Venice
Capital Rome
Capital Ljubljana
Capital Zagreb
Capital Sarajevo
Capital Belgrade
Capital Vaduz
Capital Bern
Capital Paris
Capital Monaco
Porto Monaco
Porto Genoa
Porto Barcelona
Porto Valencia
Capital Madrid
Capital Lisbon

Por um erro que não conseguimos perceber nem corrigir ao correr o teste em separado este é o output obtido mas correndo os testes todos não dá o correto.