Análise e Síntese de Algoritmos

Introdução:

O Sr. Caracol é responsável por assegurar uma rede de transporte de vários tipos de mercadorias. Fazendo com que o material produzido pelos seus fornecedores chegue até ao seu hipermercado, tendo ao seu dispor, vários meios de transportes com diferentes capacidades, os quais estão sujeitos a passar por estações de abastecimento ao longo do seu percurso.

Para auxiliar o Sr. Caracol neste problema, desenvolvemos uma aplicação que determina a capacidade da rede, ou seja, a quantidade máxima de material que a sua rede é capaz de transportar, bem como as várias ligações que precisam de ser aumentadas, as quais definem a capacidade da rede e se encontram mais próximas do hipermercado.

Descrição da solução:

Para a representação da rede usámos um grafo, em que cada vértice pode representar um fornecedor, uma entrada ou saída para uma estação de abastecimento, ou o hipermercado. Uma vez que as ligações entre eles têm direção, o grafo é dirigido. Contudo consideramos o seu grafo transposto, sendo que o corte mínimo garante incidir sobre as ligações que definem a capacidade do grafo mais perto da source, desta forma, irá incidir sobre as ligações que precisam de ser aumentadas mais perto do hipermercado.

Como existem vários fornecedores e precisamos de apenas um target para fazer a execução de qualquer algoritmo de fluxo, começamos por criar uma adjacência em todos os fornecedores para o target, aproveitando esta ligação para colocar os pesos dos materiais produzidos por estes. Para a representação das estações de abastecimento, criamos um vértice que representa a entrada na estação, o qual recebe as ligações de outros vértices e cuja única adjacência é o arco com capacidade igual àquela que consegue processar, que o liga ao vértice de saída, cujas adjacências são as ligações que têm origem na estação.

(F = #Fornecedores, S = #Estações)
Para guardar os vértices alocamos memória para uma lista de dimensão (2 + F + 2S), ficando os ids organizados da seguinte forma:

- Target = {0}
- Source = {1}
- Fornecedores = {2, F – 1}
- SaídasDasEstações = {F – 2, F + S – 1}
- EntradasDasEstações = {F + S – 2, F + 2S – 1}

Sara Machado | Nº 86923
Ricardo Grade | Nº 90774

Para encontrarmos a capacidade da rede e as ligações que precisam de ser aumentadas, recorremos ao Push-Relable-FIFO com as seguintes alterações:

- **Gap-Heuristics:** Quando um vértice necessita de aumentar a sua altura, se este for o único na sua altura original, arrasta para a altura da source todos os vértices, ele inclusive, que se encontrem entre a sua altura e a altura da source.
- **Stack das alturas limitada**: Todos os vértices que atingem uma altura igual à da source não sobem mais na stack. Desta maneira, ficamos com todos os vértices que fazem parte de um lado do corte na altura da source.

A capacidade da rede corresponde ao excesso do target. Para encontrar as ligações que precisam de ser aumentadas consideramos todas as adjacências de todos os vértices de um lado do corte. Aquelas que os liguem a vértices do outro lado do corte (expeto target), são consideradas ligações a aumentar. Não se consideram as adjacências que ligam ao target, pois essas representam a produção de cada fornecedor.

Para formatar o output da nossa aplicação, ao encontrarmos as ligações a aumentar como referido, guardamo-las numa lista, de seguida, para as ordenarmos de forma crescente, temos em conta primeiro os ids dos seus vértices de destino e depois dos seus vértices de origem, e o algoritmo a que recorremos é o qsort. Para selecionar as ligações que representam estações de abastecimento, verificamos se o id do seu vértice de origem é maior ou igual (F + S – 2), sendo o id da estação igual ao id do seu vértice de destino, caso contrário é uma ligação normal.

Análise Teórica:

(V = #Vértices, E = #Arcos)

Memoria:

- Vertexes: O(V)
- Edges: O(2E)
- $\sum_{i=0}^{V-1} \mathrm{Vertex[i].adjList}$: O(2E)
- stackList: O(V + 1)
- $\sum_{i=0}^{V} \mathrm{stackList[i]}$: O(V)
- queue: O(V)
- cutList: O(E)

Total: $O(V + E) = O(V^2)$

Sara Machado | Nº 86923
Ricardo Grade | Nº 90774

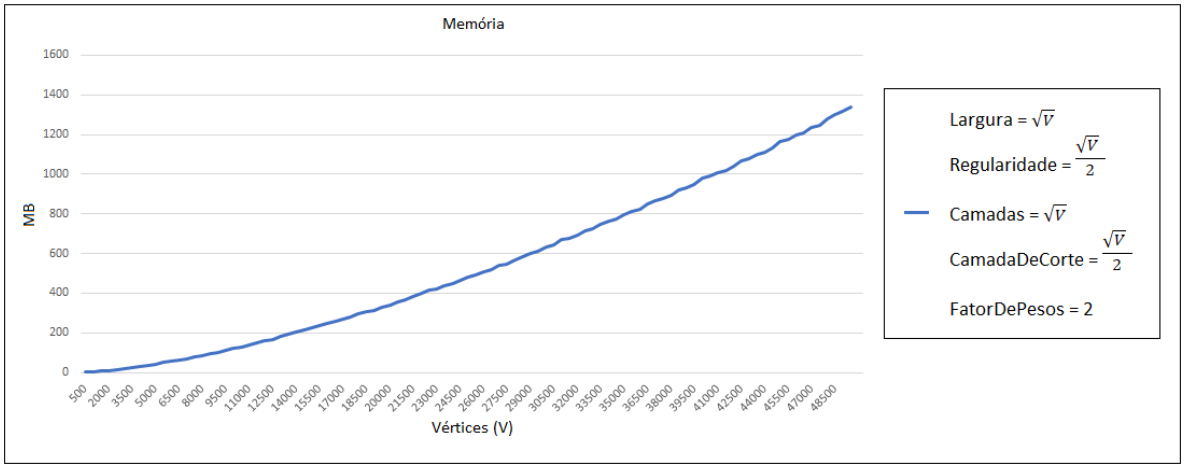
Tempo:

- Inicialização da vertexList: O(V)
- Inicialização da stackList: O(V + 1)
- Adicionar os vértices à stackList: O(V)
- Parse dos Fornecedores e Estações: O(V – S – 2)
- Parse das ligações: O(E)
- Push-Relable-FIFO com Gap-Heuristics: $O(V^3)$
  Como demonstrado em:
  https://linux.ime.usp.br/~marcosk/mac0499/files/monografia.pdf na secção 3.4.
- Procurar as ligações de aumento: O(E)
- Ordenar as ligações de aumento por ordem crescente: O(2(E – P)log(E – P))
  Uma vez que o qsort tem complexidade O(Nlog(N))
- Imprimir as restantes Estações de aumento: O(E – P)
- Imprimir as restantes ligações de aumento: O(E – P)

Total: $O(V^3 + E) = O(V^3)$

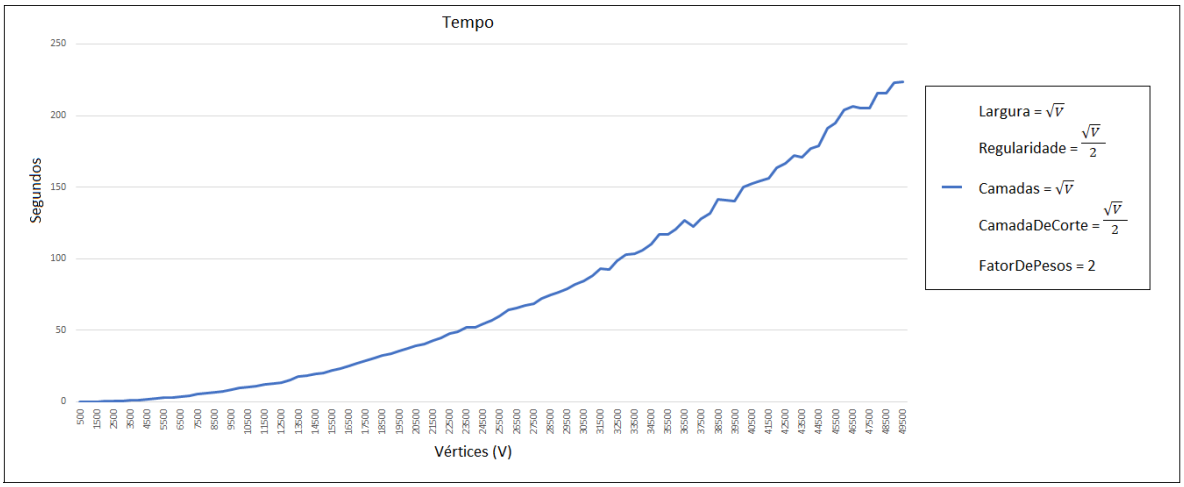
Análise Experimental:

Para realizar tanto os testes de memória como os testes de tempo, fizemos um script que executa repetidamente a nossa aplicação com o input gerado por:
./gerador $\sqrt{V}$ $\frac{\sqrt{V}}{2}$ $\sqrt{V}$ $\frac{\sqrt{V}}{2}$ 2. Fazendo V variar entre {500, 50000} incrementando-o em 500 unidades por iteração. Registando o seu tempo e memória utilizados.



Como podemos comprovar o gráfico de Memória da nossa aplicação apresenta uma curvatura semelhante a uma quadrática, desta forma podemos corroborar o facto de a memória utilizada pela nossa aplicação variar quadraticamente à medida que fazemos variar o número de vértices do grafo.

Sara Machado | Nº 86923
Ricardo Grade | Nº 90774



Como podemos comprovar o gráfico de Tempo da nossa aplicação é um pouco mais achatado que o gráfico de Memória, desta forma podemos corroborar o facto de o tempo utilizado pela nossa aplicação variar cubicamente à medida que fazemos variar o número de vértices do grafo.

Como podemos constatar, existem algumas imperfeições na curvatura do nosso gráfico, as quais se podem dever:

- Aos testes terem sido realizados numa máquina virtual.
- Ao estado de atividade no computador não ter sido regular, durante a execução do conjunto de testes.
- À duração prolongada dos múltiplos testes.
- Entre outras razões que escapam ao nosso controlo.

Sara Machado | Nº 86923
Ricardo Grade | Nº 90774