Relatório 2º projecto ASA 2019/2020

Grupo: al112

Aluno(s): Duarte Bento (92456) e Susana Monteiro (92560)

Descrição do Problema e da Solução

O problema consiste em, dado um conjunto de cidadãos e supermercados, obter o máximo de clientes que conseguem chegar a um supermercado. No entanto no âmbito do distanciamento físico é necessário impedir que dois clientes passem no mesmo ponto. Em particular, como as ruas têm um arranjo quadriculado, o problema resume-se em determinar o número máximo de clientes que conseguem chegar a um supermercado, sem que dois clientes nunca se encontrem numa esquina.

Deste modo, optou-se por solucionar o problema recorrendo a um grafo dirigido, em que as arestas correspondem às ruas e avenidas de Manhattan e os vértices às esquinas que estas formam. A solução proposta passa por calcular o fluxo máximo neste grafo. Para tal foram adicionados dois vértices artificiais: a fonte - à qual foram ligados todos os cidadãos - e o vértice alvo - ao qual foram ligados todos os supermercados.

Para permitir que saia apenas um cidadão de casa em cada esquina, entre apenas um cidadão em cada supermercado e, no geral, haja apenas um cidadão a percorrer cada rua/avenida, todas as arestas do grafo têm capacidade unitária (e inicialmente fluxo nulo). Decidiu resolver-se o problema (calcular o fluxo máximo) implementando o método de Ford Fulkerson recorrendo a uma DFS (Depth First Search) para encontrar caminhos de aumento (caminho entre a fonte e o alvo), aumentando o fluxo total e atualizando as arestas do caminho até não haver mais caminhos de aumento. No final, o valor devolvido corresponde ao fluxo máximo e simboliza o número de cidadãos que conseguem deslocar-se a um supermercado.

Análise Teórica

Seguindo o anúncio, e para facilitar, consideramos M o número de avenidas, N o número de ruas, S o número de supermercados e C o número de cidadãos.

- Criação do grafo: ***O(MN + S + C)***
  - Criação de duas matrizes que guardam respetivamente os V_{in} e V_{out} e inicialização de todos os vértices, logo ***Θ(MN)*** *(1)*
  - Ler e processar os supermercados e os cidadãos, logo ***O(S + C)***
- Para cada vértice, adicionar as arestas correspondentes, logo ***O(MN)*** *(2)*
- Aplicar algoritmo de Ford Fulkerson, recorrendo a uma DFS: ***O(MN*min(S,C))***
  - Número máximo de caminhos de aumento: ***O(min(S,C))***
  - Tamanho máximo de um caminho de aumento: ***O(E) = O(MN)***
- Apresentação do resultado final (valor do fluxo maximo). ***Θ(1)***

Complexidade global da solução:

$$O(MN + S + C) + O(MN) + O(MN*\min(S,C)) + \Theta(1) = \mathbf{O(MN*min(S,C))}$$

Relatório 2º projecto ASA 2019/2020

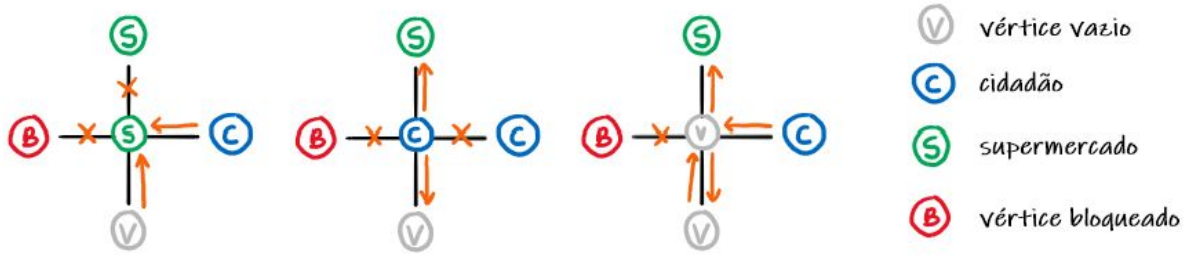
Grupo: al112

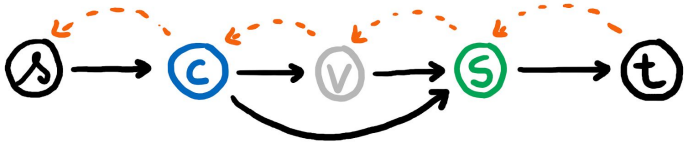
Aluno(s): Duarte Bento (92456) e Susana Monteiro (92560)

---

(1) Para impedir que passe apenas um cidadão por cada esquina, esta foi dividida em dois vértices (V_{in} e V_{out}) ligados por uma aresta de capacidade unitária.

(2) As arestas foram calculadas conforme o vértice: aos cidadãos adicionamos apenas arcos para fora e para os supermercados apenas arcos para dentro. Observou-se que arcos a ligar cidadão-cidadão ou supermercado-supermercado eram desnecessários visto que apenas formavam caminhos semelhantes mas mais longos. No caso de haver mais do que um cidadão ou supermercado por esquina considerou-se apenas um deles e no caso de um cidadão e um supermercado na mesma esquina cortou-se essa esquina e incrementou-se o número total de clientes a atingir um supermercado (esse cliente consegue ir ao supermercado da mesma esquina).

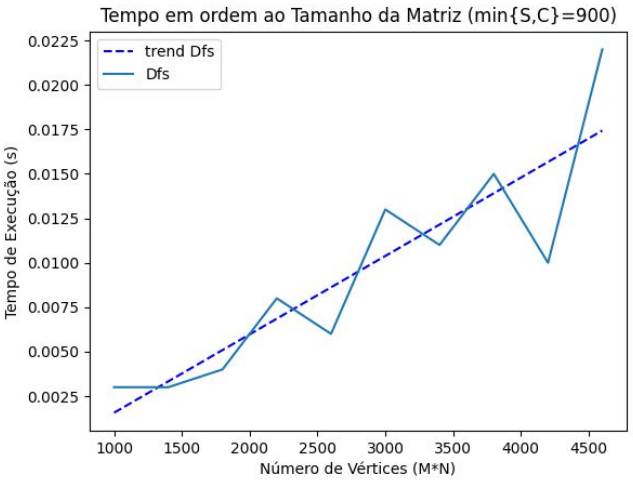




No exemplo do lado esquerdo, as arestas a preto fazem parte do nosso grafo, enquanto as que estão a laranja são aquelas que consideramos desnecessárias

Avaliação Experimental dos Resultados

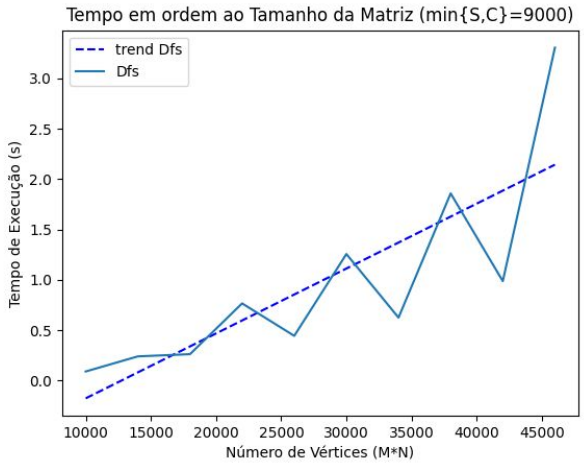
Recorrendo a um script, começou-se por gerar inputs aleatórios de modo a reunir um conjunto de valores que pudesse ser estudado. Decidiu manter-se constante o mínimo entre o número de cidadãos (C) e de supermercados (S) e variou-se o número de vértices da matriz (MN), esperando obter-se um gráfico linear. Foram criados 3 gráficos, para diferentes valores de min{S,C} (respetivamente 900, 9000 e 90000) e diferentes ordens de grandeza do tamanho da matriz (M*N).

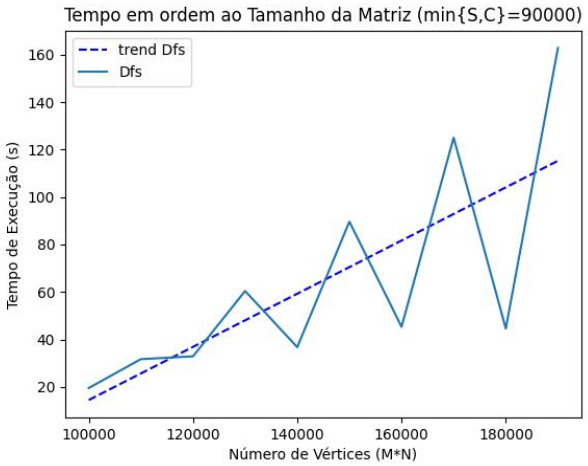


Relatório 2º projecto ASA 2019/2020

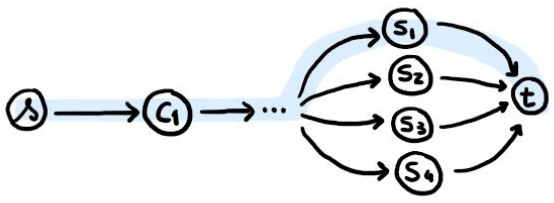
Grupo: al112

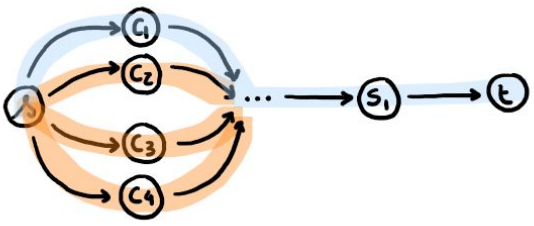
Aluno(s): Duarte Bento (92456) e Susana Monteiro (92560)





Verificou-se que os valores observados nos gráficos parecem estar intercalados, ainda que a sua tendência seja linear. Isto é explicado pela forma como introduzimos os valores de cada ponto: o min{S,C} estabelecido foi atribuído ao número de cidadãos (*low peaks*) e ao número de supermercados (*high peaks*), intercaladamente. Embora o fluxo máximo não dependa de qual o valor que se minimiza (S ou C), as várias DFS feitas ao longo do projeto são mais eficientes nos casos em que existem mais supermercados do que cidadãos, uma vez que têm de percorrer um menor número de vértices até encontrar um caminho de aumento. A adição de uma verificação totalFlow == min{S,C} pouparia uma última procura de caminhos de aumento, que consiste numa execução da DFS completa (visto que o objetivo é não conseguir encontrar um caminho entre a fonte e o alvo, sendo necessário explorar todas as possibilidades).



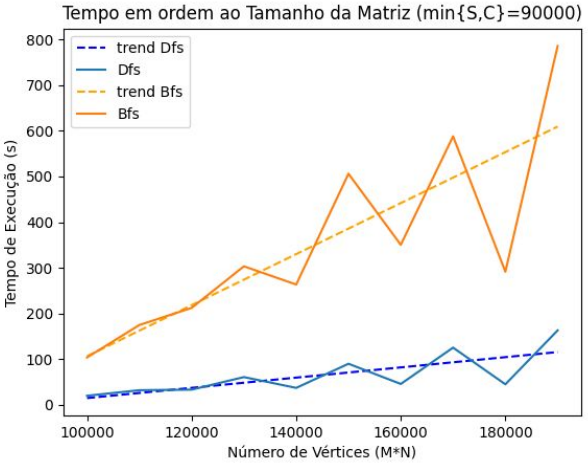


Relatório 2º projecto ASA 2019/2020

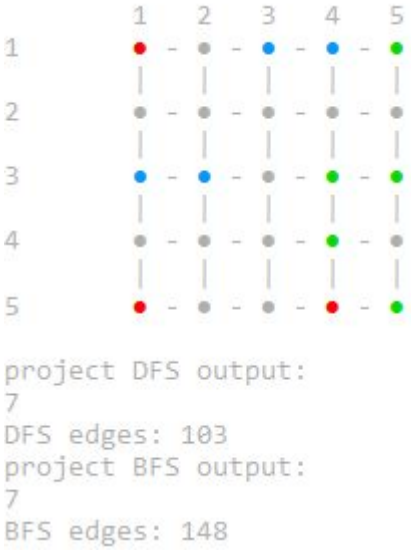
Grupo: al112

Aluno(s): Duarte Bento (92456) e Susana Monteiro (92560)

Além disso, durante o processo de decisão de qual o algoritmo a utilizar para resolver o problema proposto, tentou optar-se pela solução mais eficiente na procura dos caminhos de aumento. Foi implementada uma DFS, por parecer uma melhor opção. Mais tarde, por curiosidade, decidiu implementar-se a BFS para que fosse possível comparar o desempenho dos dois algoritmos. Após a construção de um gráfico com os valores de desempenho (em termos de tempo) obtidos, concluiu-se que embora ambas as opções sejam suficientes no âmbito do projeto (ambas aceites pelo Mooshak), a DFS tem um melhor desempenho. Para além do tempo de execução, foi determinado também o número de visitas a vértices (ainda não visitados), sendo que este número chega a ser uma ordem de grandeza superior na BFS, quando comparada com uma DFS.



Adicionalmente, de modo a confirmar visualmente o resultado do nosso programa, foi desenvolvido um visualizador de matrizes, tomando como "input" problemas produzidos pelo gerador disponibilizado pela cadeira.



Através da linha de tendência dos gráficos, que se provou semelhante à esperada, comprovou-se a proposição inicial de que a complexidade do algoritmo proposto depende linearmente do tamanho do problema (número de avenidas por número de ruas).