**Descrição do Problema e da Solução**

Problema: Uma vez que os cidadãos não se podem cruzar, visto que há possibilidade de ficarem infetados com tal aproximação, cada rua, cada avenida e cada supermercado apenas pode ter um cidadão a deslocar-se de cada vez. Assim, neste problema foi nos pedido para desenvolver um algoritmo que devolvia o número máximo de cidadãos que podiam deslocar-se até supermercados diferentes sem se cruzarem uns com os outros.

Solução: Uma vez que se trata de um problema de fluxo máximo, modelámos o mapa da cidade como uma rede de fluxo construída da seguinte forma:

* Para cada cruzamento, criámos um node na rede de fluxo. Uma vez que cada cruzamento tem capacidade igual a 1, cada node é constituido por dois vertices, um de entrada e outro de saída, que são ligados por um arco de capacidade igual a 1;
* Adicionámos também dois vértices auxiliares *s* e *t* que representam a source e a sink da rede, respetivamente;
* O vértice *s* tem um arco para cada cidadão com capacidade igual a 1, visto que é o número máximo de pessoas que podem sair de casa (cidadãos representam múltiplas fontes);
* Cada supermercado tem um arco para o vértice *t* com capacidade igual a 1, visto que é o número máximo de cidadãos em cada supermercado (supermercados representam múltiplos destinos);
* Cada node tem o seu vértice de saída a apontar para o vértice de entrada dos outros nodes, que correspondem aos cruzamentos que lhe são adjacentes;

Assim, com o problema modelado, aplicámos um algoritmo de cálculo de fluxo máximo para obtermos o valor pretendido. Escolhemos o algoritmo Edmonds-Karp, porque a capacidade e o fluxo entre cada arco apenas variam entre 0 e 1. Esta escolha também teve em conta o facto do Edmonds-Karp utilizar uma BFS, o que permite calcular os caminhos mais curtos em primeiro lugar.

**Análise Teórica**

1. Leitura dos dados de entrada: na função que lê o input temos 2 *for’s:*

-Um para ler os supermercados e outro para ler os cidadãos → O(C) + O(S) = =O(V);

-Função que cria as ligações entre nodes → O(V);

1. Aplicação do algoritmo de Edmonds-Karp:

-Aplicação de BFS:

*For* para inicializar todos os vertices como não visitados, declarar o vértice predecessor e o arco que liga o predecessor ao vértice em questão a NULL→ O(V);

Análise teórica da complexidade total e das várias etapas da solução proposta:

Exemplo:

* Leitura dos dados de entrada: simples leitura do input, com ciclo(s) a depender de linearmente/quadraticamente/… de V/E/V+E/… Logo, Θ(V)
* Processamento do grafo para fazer alguma coisa. Logo, O(??)
* Aplicação do algoritmo X para fazer algo. Logo, O(?X?X)
* Transformação dos dados com uma dada finalidade. O(?Y?Y?)
* Apresentação dos dados. O(???)

Complexidade global da solução: O(!??!)

**Avaliação Experimental dos Resultados**

Descrição do tipo experiências feitas e gráfico demonstrativo da avaliação de tempos associados.

Gerar vários grafos de tamanho incremental e cálculo dos tempos para cada instância. Gerar o gráfico do tempo em função do tamanho do grafo de entrada como exemplificado abaixo.



Concluir se o gráfico gerado está concordante com a análise teórica prevista.