# Relatório 2º projecto ASA 2024/2025

**Grupo:** AL046

**Aluno(s):** Guilherme Lopes (ist1109913) e Pedro Vicente (ist1109852)

---

### Descrição da Solução

Explicação da solução:

- Descrição do grafo construído (podem ou não incluir o pseudo-código da construção do grafo);
- Descrição do(s) aplicado(s) sobre o grafo construído. Não é necessário incluir o pseudo-código de algoritmos standard (por exemplo, BFS) usados como parte da vossa solução.

Solução:

Construir um grafo inicial, "Graph", que é um vetor de sets. Terá tantos sets como o número de linhas do metro. Cada set terá as estações que pertencem a essa linha. Ao criar este grafo, temos o cuidado de verificar se alguma das estações está isolada das outras e se alguma linha contém todas as estações, pois nestes casos devemos imprimir -1 e 0, respetivamente, e terminar o programa.

De seguida, vamos construir um grafo reduzido, que será um vetor de nodes, "Nodes". Terá tantos nodes quanto o número de linhas de metro. Os nodes possuem atributos para criar o grafo: um inteiro que corresponde à linha que representam; um set de inteiros, que correspondem às linhas a que estão conectados diretamente; um booleano, que indica se o node já foi criado no grafo. Possuem também atributos para serem usados numa BFS: um ponteiro para um node pai; um inteiro para distância; um inteiro para representar uma cor (branco, cinzento ou preto). Vamos ver como construir o grafo reduzido.

Um loop for começa a iterar com "i" pelo número de linhas de metro. Se não existir Nodes[i], criamo-lo. Agora vamos iterar sobre as estações de G[i], uma "station" de cada vez. De seguida, iteramos com "j" pelo número de linhas de metro. Se não existir Nodes[j], criamo-lo. Verificamos se Graph[j] possui a "station". Se possuir, as linhas estão diretamente ligadas; inserimos j em Nodes[i].adjacents e i em Nodes[j].adjacents. Já temos o grafo reduzido, mas é necessário verificar se algum node tem o set de adjacentes com tamanho 0. Se sim, significa que há uma linha isolada das outras, e devemos imprimir -1 e terminar o programa.

Com o grafo reduzido montado, e inicializando dois inteiros, connectivity e metro_connectivity, a 0, aplicamos BFS a cada node. A BFS altera o valor de connectivity para distância máxima entre o node de "start" e os outros nodes, seguindo o caminho mais curto possível entre estes. A variável metro_connectivity guarda o maior valor de connectivity e será impressa no fim do programa. Fizemos uma pequena alteração na BFS, para que no final verifique se algum node não é alcançável pelo node de "start"; se tal acontecer, a connectivity passa a -1, é imprimida, e o programa termina.

### Análise Teórica da Solução Proposta

Indicar a complexidade de cada etapa da solução proposta, e a complexidade total. Por exemplo:

- Leitura dos dados de entrada: O(M) – tirando a primeira linha, todas as outras representam uma das M conexões.
- Construção do grafo: O($NL^2$) – o grafo reduzido consiste num loop for sobre as l linhas, outro loop for sobre as estações de cada linha, que é na pior das hipóteses N-1 (visto que, se uma linha tem N estações, imprimimos 0 e terminamos o programa), e um terceiro loop for, novamente sobre as L linhas. Posto isto, a complexidade de criar o grafo reduzido é $NL^2$.
- Aplicação do algoritmo indicado para cálculo do valor pedido: O($M^3$) – verificar se alguma linha tem todas as estações é O(1) porque ao ler o input tínhamos um booleano que verificava se G[linha].size() == n. Ver se há estações isoladas também é O(1) porque na leitura do input inserimos as estações num set chamado "not_isolated_stations" (bastou ver se o seu size < N). Ver se há linhas isoladas é O($L$), dado o loop for que vê se Nodes[linha].adjacents.size() == 0.
Nos casos em que a conectividade não é 0 nem -1, a complexidade é dada por um loop for que itera pelas L linhas de metro, e aplica BFS a cada uma. A BFS possui um ciclo while, que visita todos os L vértices do grafo reduzido. Possui um loop for que itera pelos vértices adjacentes de cada vértice, desde que ainda não tenham sido visitados; por outras palavras, itera por todos os arcos do grafo reduzido. O número de arcos do grafo reduzido é, no pior caso, o de um grafo

completo não dirigido, ou seja, $L(L-1)/2$. Assim, a análise agregada da BFS totaliza para O(V+E) = O($L + \frac{L^2 - L}{2}) = O\left(\frac{L^2+L}{2}\right) \leq O(L^2)$. Relembramos que a BFS é aplicada num ciclo for que percorre todos os L vértices do grafo reduzido, pelo que o ciclo totaliza para O($L^3$).

Após analisados todos os passos para aplicação do algoritmo, vemos que este é O($M + NL^2 + 1 + L + L^3$) = $O(M + L(NL + 1 + L^2)) = O(M + NL^2 + L^3)$). Lembrando que, no pior caso, o número de linhas L podem ser as M conexões, temos $O(M + M^2(N + M)) = O(M(1 + MN + M^2)) = O(NM^2 + M$^3). Será possível que o termo $NM^2$ domine o algoritmo? Isso significaria que o grafo original tinha mais estações do que conexões. Isto implicaria que uma estação estivesse isolada. Este caso não é processado devido a uma das condições de "early stopping" que já vimos anteriormente. Temos então $O(M^3)$ para a complexidade do algoritmo.

- Complexidade global da solução: O($M^3$) – Considerando os três parágrafos anteriores, temos $O(M + NL^2 + M^3)$. Pelo argumento do último parágrafo, $O(M^3)$.

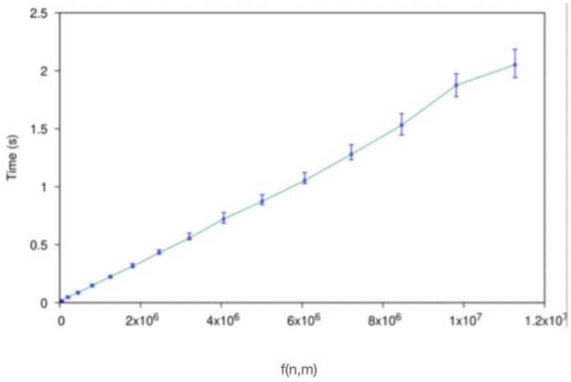
Cada bullet point deve incluir uma justificação sucinta para a complexidade indicada.

**Avaliação Experimental da Solução Proposta**

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

Gerar mais de 10 instâncias de tamanho incremental e incluir uma tabela com o tamanho das instâncias utilizadas e tempos respectivos.

Gerar o gráfico do tempo (eixo do YYs) em função da complexidade teórica prevista (eixo dos XX). Mais concretamente, colocar o eixo dos XX a variar com a quantidade prevista pela análise teórica; exemplo: se a análise teórica for O(f(n, m, l)), o tempo deve ser colocado em função de f(n, m, l).




Devemos observar uma relação linear entre a complexidade teórica prevista e os tempos registados, confirmando que a implementação está de acordo com a análise teórica.