Relatório 2º projeto ASA 2024/2025

Grupo: AL010
Alunos: Francisco Silva (110409) e Marta Braga (110034)

Descrição do Problema e da Solução

A solução desenvolvida começa por ler o input linha a linha, criando uma lista de adjacências entre as estações e as linhas de metro, que associa cada estação às linhas a que esta pertence.
Exemplo:

```
input: 3 3 2
       1 2 1
       2 3 1
       1 3 2
lista: [[1, 2], [1], [1, 2]
```

Com base nesta estrutura, verificamos se há alguma linha que conecte todas as estações (caso em que devolvemos 0). Caso contrário, é construído um segundo grafo em que cada vértice representa uma linha de metro e duas linhas estão conectadas por uma aresta se partilharem pelo menos uma estação. Este grafo de linhas abstrai a rede original e permite focar apenas nas mudanças de linha necessárias.

Para construir este grafo, percorremos as linhas adjacentes a cada estação. Para cada linha, adicionamos todas as outras linhas adjacentes à estação à lista de adjacências desta linha.

Construção do grafo das linhas

```
Ensure: Return 1 if any station has no connections, otherwise 0
  for station :=1 to stationsCount do
      for each currentLine in stationsGraph[station] do
          lineNode :=getNode(linesGraph, currentLine)
          for each connectedLine in stationsGraph[station] do
              if connectedLine ≠ currentLine then
                  neighborNode := getNode(linesGraph, connectedLine)
                  lineNode.addNeighbor(neighborNode)
              end if
          end for
      end for
  end for
  return 0
```

Uma vez tendo gerado o grafo das linhas, podemos aplicar uma BFS iterativa a cada linha onde, simultaneamente, verificamos se existem vértices isolados ou subconjuntos desconexos, caso em que devolvemos -1. No fim da BFS percorremos todos os nós novamente e registamos o maior índice de conectividade (mc) calculado. Fazemos isto para todas as linhas comparando e guardando sempre o maior mc entre o calculado na BFS atual e o máximo anterior. No fim é retornado o maior índice de conectividade encontrado.

Análise Teórica

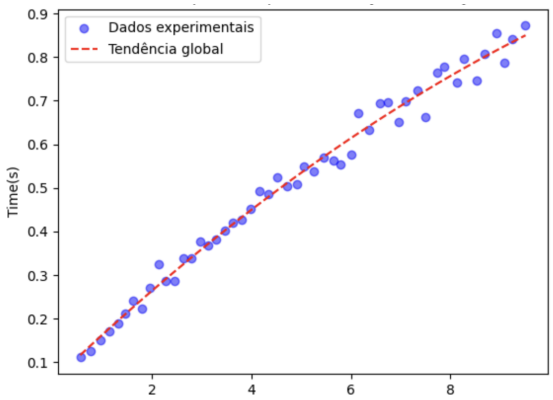
Leitura de input: O(m). Iteramos por todas as linhas do input uma única vez.
Construção do grafo das linhas: $O(n \times l^2)$. Para cada uma das n estações, vamos à sua lista de adjacências que terá no máximo l ligações. Para cada linha percorremos novamente as l linhas adjacentes a esta.
Cálculo do mc: $O(l^3)$. Fazemos l BFS's. Cada BFS tem custo l^2 uma vez que percorremos todas as l linhas e, para cada linha, percorremos todas as suas adjacências que serão, no pior caso, também l.
Complexidade temporal final: $O(n \times l^2)$. Corresponde à maior complexidade encontrada no código, uma vez que podemos assumir que existirão sempre mais estações do que linhas, logo $(n \times l^2) > l^3$.

Relatório 2º projeto ASA 2024/2025

Grupo: AL010
Alunos: Francisco Silva (110409) e Marta Braga (110034)

Avaliação Experimental dos Resultados

O gráfico abaixo ilustra a relação entre o tempo de execução do código em segundos e a função $f(n, m, l) = n \times l^2$, que corresponde à complexidade calculada.
Para gerar os gráficos, foram feitos 50 testes. Nesses testes, os valores de n utilizados variaram entre 5 000 e 30 000, os valores de m entre 110 000 e 600 000 e os valores de l entre 100 e 200.



No gráfico acima podemos observar que os dados experimentais variam linearmente com a função calculada. Assim sendo, podemos confirmar que a implementação está de acordo com a análise teórica feita.