Relatório Trabalho de ADA – Legionellosis

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| * Ano Letivo | * 2020/2021 | * Semestre | * 2 | * Cadeira | * ADA |
| * Alunos | * Gonçalo Martins Lourenço nº55780 | | | | |
| * Joana Soares Faria nº 55754 | | | | |

## Complexidade Temporal

Na análise da complexidade temporal consideramos as seguintes variáveis:

- número de pessoas doentes

- número de localizações existentes

- número de conexões entre localidades

- número de localizações perigosas

Em termos de complexidade temporal começamos por analisar a inicialização da lista ligada de sucessores (connections), que será executada l vezes e que em cada passo tem custo constante. Este ciclo tem então custo .

Para resolver o problema temos um método que será executado para cada pessoa doente, ou seja, será executado vezes. Nesse método é feita uma pesquisa em largura com limite de profundidade. A origem do grafo a considerar em cada passo é variável e a profundidade a explorar também é variável. Para doentes diferentes podemos ter de explorar vértices e arcos já explorados anteriormente.

Tendo todos esses aspetos em consideração, no pior dos casos podemos ter de explorar todos os arcos e todos os vértices, para todos os doentes, mas noutros casos, pode-se nem ter que explorar todos os vértices do grafo.

Para devolver o vetor ordenado e obedecer aos requisitos do enunciado temos uma complexidade de , sendo que será sempre menor ou igual a .

Assim se justifica uma complexidade de . Consideramos porque cada conexão apresentada é bidirecional.

# Complexidade Espacial

Na análise da complexidade espacial consideramos as seguintes variáveis:

- número de pessoas doentes

- número de localizações existentes

- número de conexões

- número de localizações perigosas

Fazendo a análise da complexidade do nosso código podemos aferir uma complexidade espacial de .

Para a execução do código contamos com uma variável que guarda o número de localizações , um vetor com todas as localizações , assim como uma lista ligada com as localizações perigosas . Contamos ainda com um vetor de listas ligadas, para guardar, para cada ponto, as suas conexões. Podemos por isso dizer que cada conexão é guardada duas vezes. Sendo assim a complexidade espacial desta parte será .

Assim sendo, e considerando que , a complexidade apresentada inicialmente pode ser simplificada para .

# Conclusões

Para a implementação do algoritmo foi escolhido um percurso em largura porque é necessário percorrer todos os vértices do grafo (localizações), sem repetições, a partir de uma dada origem e limitar a profundidade alcançada. Como o percurso em largura é feito por nível, mostrou-se a forma ideal de explorar todos os nós, nível a nível, e parando no último nível a considerar.

Para a estrutura que guarda as localizações perigosas encontradas foram consideradas várias opções. A primeira opção era apenas o vetor com o número total de doentes que passavam em cada localização e no final fazer uma verificação de quais as localizações que tinham um número de doentes igual ao número total de doentes. Isto obrigaria a percorrer todo o vetor das localizações e chegar a uma complexidade de Sendo de esperar que o número de localizações perigosas seja várias vezes inferior ao número total de localizações, esta solução não nos pareceu adequada.

A segunda opção era considerar uma estrutura de dados ordenada para guardar as localizações perigosas. Porém o custo de inserção e remoção seria de , pelo que excluímos esta opção.

Dado que a ordenação só é de realização necessária uma vez, optou-se por uma lista ligada para as localizações perigosas pois esta apresenta inserção e remoção constantes. O custo de ordenação desta estrutura é de pelo que se demonstrou a melhor solução.

Em estruturas de dados que não sabíamos que tamanho seria necessário escolhemos estruturas de dados dinâmicas a fim de diminuir a complexidade espacial e temporal do algoritmo.

Anexo – Código Main

Anexo – Código Class Legionellosis