Relatório 2º projeto ASA 2023/2024

Grupo: AL017

Aluno(s): Rodrigo Perestrelo (ist1106074) e Cristiano Pantea (ist1106324)

Descrição do Problema e da Solução

O problema consiste no estudo do pior caso de propagação de uma dada infeção em Portugal. Temos assim de determinar o maior número de saltos que uma dada doença pode fazer. No entanto, tendo em conta a densidade das cidades portuguesas, considerou-se um pressuposto simplificador: indivíduos que se conhecem mutuamente de forma direta ou indireta, ficam infetados instantaneamente.

A nossa solução consiste na travessia de dois grafos de forma iterativa, construídos a partir do input, sendo um deles o transposto do outro. Na primeira travessia, no grafo normal, obtemos através de uma DFS a ordem topológica inversa dos tempos de fim dos vértices. Na segunda travessia, agora no grafo transposto, percorremos o mesmo pela ordem obtida na 1ª travessia. Nesta travessia, começamos por inicializar um vetor de visitados e outro de saltos com tamanho nº vértices + 1. O primeiro inicializado com todos os valores a false e o segundo inicializado a -1. Garante-se que cada iteração do loop exterior corresponde a uma SCC, consequência da ordem pela qual percorremos o gráfico. Assim, a parte principal do algoritmo consiste em, para cada SCC, percorrer os seus vértices. Caso encontremos algum vizinho com valor de salto maior que o próprio valor de salto do vértice atual, guardamos esse valor e, no final da iteração, atribuímos esse valor + 1 a todos os vértices da SCC, caso contrário, colocamos todos os vértices desse SCC a 0. Logo, **o valor calculado para cada vértice é a maior distância de um caminho desde um vértice source até ao vértice.**

Análise Teórica

Complexidade de cada etapa da solução proposta, e complexidade total:

● Leitura dos dados de entrada: simples leitura do input para criação dos grafos, com um ciclo a depender linearmente de E (número de arestas). **Complexidade: O (E).**

● Primeira travessia iterativa no grafo normal. **Complexidade: O (V + E).**

● Aplicação do algoritmo indicado para cálculo do valor pedido, ou seja, segunda travessia iterativa. Pseudocódigo:

```
let V be an array obtained from the iterative DFS ordered by the reverse topological order
let jumps be an array initialized at -1 at all indexes
for each vertex in V not visited {
    SCC = getSCC(vertex);
    bool found = false;
    int value = 0;
    searchBiggerValue(SCC, found, value); // Iterates through the SCC and changes the
                                          // variables if finds a bigger neighbour jump value
    if(found) changeJumps(SCC, value + 1);  // Changes all the values belonging at the SCC given to the given value.
    else changeJumps(SCC, 0);
}
```

Relatório 2º projeto ASA 2023/2024

Grupo: AL017

Aluno(s): Rodrigo Perestrelo (ist1106074) e Cristiano Pantea (ist1106324)

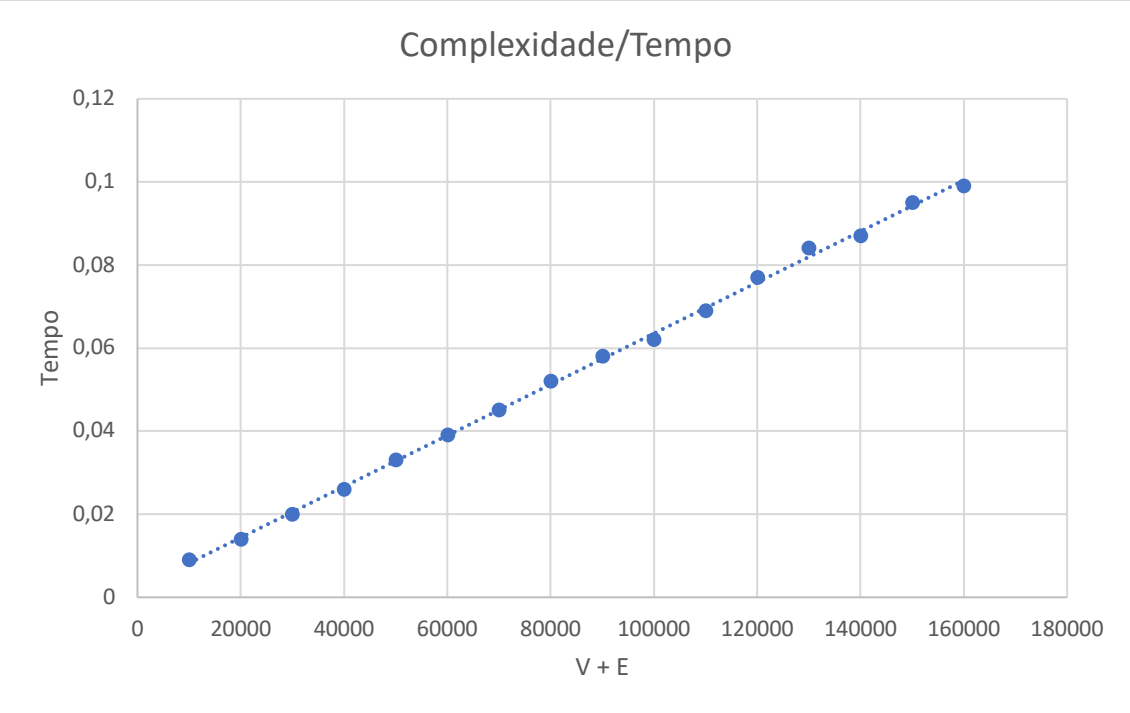
Como o nosso algoritmo itera por cada SCC uma primeira vez para verificar qual o maior valor de saltos a atribuir a cada vértice de cada SCC (percorre aqui as arestas e os vértices) e seguidamente itera novamente pela SCC toda apenas para mudar o valor de todos os vértices (percorre os vértices novamente) a complexidade da análise agregada acaba por ficar: **O (2V + E) → O (V + E).**

● Apresentação dos dados. **Complexidade: O (1).**

Complexidade global da solução: **O (V + E).**

Avaliação Experimental dos Resultados

Para demonstrar que a nossa análise teórica está de acordo com a nossa implementação do algoritmo, realizámos testes com diferente número de vértices e arestas. Estes deram origem ao seguinte gráfico:



Colocando V+E no eixo dos XX, vemos que temos uma relação linear com os tempos no eixo dos YY, confirmando que a nossa implementação está de acordo com a análise teórica prevista, ou seja, **O(V + E)**.