Relatório 2º Projeto de ASA

Grupo: tp047
Alunos: Inês Cadete (número: 102935), Luís Miguel Rebanda (número: 83925)

Descrição do Problema e da Solução

O desafio deste projeto consistia em determinar o maior número de saltos dados de propagação dados por um vírus, tendo informações sobre as conexões entre pessoas da população a ser estudada. Assim, o objetivo seria mapear os possíveis caminhos de transmissão, reduzindo ciclos em que todos os indivíduos são instantaneamente infetados, calculando o pior caso, ou seja, o caminho mais longo.

Portanto, representando o grafo de ligações sociais numa lista de adjacências, decidimos utilizar o algoritmo de Kosaraju-Sharir (que aplica a depth-first search, DFS, ao grafo e ao seu transposto), de forma a identificar os ciclos e posteriormente criar um DAG (directed acyclic graph). Por fim, para encontrar o caminho mais longo no grafo simplificado, usámos o algoritmo da DFS sobre o mesmo.

Análise Teórica

Solução recursiva: algoritmo Kosaraju-Sharir, incluindo DFS

- Leitura dos dados de entrada: leitura em ciclo do input, que depende do número de ligações socias (E). Logo, O(E).
- Processamento dos dados de entrada: preenchimento de uma lista de adjacências com os vértices(indivíduos) e as suas ligações. O tamanho da lista de adjacências depende do número de indivíduos e de ligações sociais indicadas no input (V e E). Logo, a ocupação em memória é O(V + E).
- Aplicação do algoritmo de Kosaraju-Sharir:
    - aplicar a DFS (recursivamente) ao grafo (para cada vértice), indo a todas as ligações existentes (E), ficando O(V*E); Esta primeira aplicação da DFS enche uma stack com os vértices percorridos (O(V)), para guardar a sua ordem
    - reverter o grafo (para obter um grafo transposto, dependendo do número de vértices e do número de ligações, que ficando em dois ciclos ‘for’ aninhados), e aplicar de novo uma DFS ao transposto. Ou seja, o algoritmo tem: O(V + E) + O(V + E) + O(V + E). Isto é 3*O(V + E), simplificando para O(V + E)
    - Nota: neste algoritmo foi preenchido um vetor de booleans para verificar se os vértices já tinham ou não sido visitados, fica O(V); Preenche-se também um vetor com os identificadores de cada SCC (strongly connected component), de

forma a retirar os ciclos, sendo também O(V)

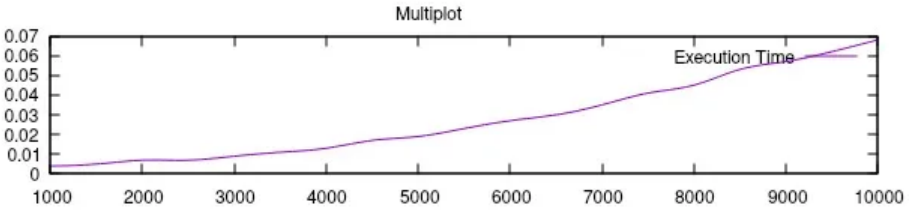
- Tendo identificado os SCC's, cria-se um DAG (directed acyclic graph). Finalmente, aplica-se novamente o algoritmo da DFS sobre o mesmo. Fica O(V + E)

----> Complexidade global da solução: somando as complexidades de todos os passos anteriores, a de maior peso é O(V + E), com os ciclos aninhados, por isso, essa é a complexidade total desta implementação

Avaliação Experimental dos Resultados

Foram geradas instâncias com valores incrementais de vértices e ligações (V e E).

Gerou o gráfico do tempo (eixo do YYs), em que o mesmo foi colocado em função de V + E(eixo dos XXs), ou seja, vértices mais as ligações.



Como se comprova no gráfico anterior, de acordo com a previsão teórica, verifica-se que há uma relação linear entre V+E e o tempo registados. Considerando que a análise teórica concluiu que a complexidade total do programa seria O(V + E).