Relatório 2º projecto ASA 2021/2022

Grupo: tp063

Aluno(s): João Vaz (98946) e André Santos (99730)

Descrição do Problema e da Solução

Com o intuito de calcular os **ancestrais comuns mais próximos** de dois vértices num grafo dirigido, desenvolvemos um programa separado em duas etapas principais: verificar a validade do gráfico dirigido de acordo com as regras dadas no enunciado(**DFS**) e cálculo dos ancestrais comuns mais próximos(**LCA**).

O **LCA** foi dividido em 2 etapas, criação de um sub-grafo com os ancestrais comuns entre os dois vértices e de seguida eliminação dos ancestrais que não pertencem à geração mais próxima possível dos vértices pretendidos.

Análise Teórica

Leitura dos dados de entrada: simples leitura dos dois primeiros inteiros para aplicar o algoritmo **LCA** a esses dois vértices e de dois inteiros “n” e “m” para ver o número de vértices e arestas, respectivamente, e de seguida, leitura de “m” linhas cada uma com valores “x” e “y” a depender de linearmente de E. Logo, **O(E)**.

Aplicação do algoritmo **DFS** que cria um array de cores e inicializa todas a branco **(O(V))**, aplicação do algoritmo **DFS_Visit** a todos os vértices que estejam brancos **(O(V))**, chamada de **DFS_Visit** dentro de **DFS_Visit** que irá analisar todos os arcos do grafo, dando erro se revisitar um vértice que já tinha sido posteriormente visitado **(O(E))**. Complexidade **DFS** + **DFS_Visit** = **O(V + E)**.

LCA: Aplicamos o algoritmo **LCA_Visit** para os dois vértices pretendidos. Para o primeiro vértice, iteramos sobre todos os ancestrais, pintando de uma cor (GRAY). De seguida, ao iterar sobre os ancestrais do segundo vértice pintamo-los de uma cor diferente (RED). Se encontrarmos um vértice pintado de GRAY, significa que encontrámos um ancestral comum pintando-o de preto. **O(V + E)**.

Como apenas queremos os ancestrais comuns mais próximos, executamos a função **“filter”** para todos os vértices de preto. E iteramos pelos outros vértices deixando todos os ancestrais que não possuem adjacentes de preto, e pintando de uma cor diferente (ORANGE) os restantes. **O(V + E)**.

A complexidade do algoritmo é **O(V + E) + O(V + E) = O(V + E)**.

Relatório 2º projecto ASA 2021/2022

Grupo: tp063

Aluno(s): João Vaz (98946) e André Santos (99730)

DFS(Vertices)

- **For** u **in** N
    - **If** colour[u]==WHITE
        - DFS_Visit(Vertices, v)
    - **End if**
- **End for**

DFS_Visit(Vertices, u)

- colour[u] = GRAY
- **For** v **in** Childs(u)
    - **if** color[v] == WHITE
    - DFS-Visit(G, v)
    - **end if**
- **end for**

LCA(Vertices, v1, v2)

- LCA_Visit(Vertices, v1, GRAY)
- LCA_Visit(Vertices, v, RED)
- **For** v **in** Vertices
    - **If** colour[v] == BLACK or colour[v] == ORANGE
        - filter(Vertices, v)
    - **End if**
- **End for**
- **For** v **in** Vertices
    - **If** colour[v] == BLACK
        - print(v)
    - **End if**
- **End for**

LCA_Visit(Vertices, v, COLOUR)

- **If** colour[v] == WHITE
    - colour[v] = COLOUR
- **Else**
    - colour[v] = BLACK
- **If** number_of_parents[v] == 0 **do** nothing
- **Else If** number_of_parents[v] == 1 **do** LCA_Visit(Vertices, parent, COLOUR)
- **Else If** number_of_parents[v] == 2 **do**
    - LCA_Visit(Vertices, parent1, COLOUR)
    - LCA_Visit(Vertices, parent2, COLOUR)

Complexidade global da solução: O(V + E)

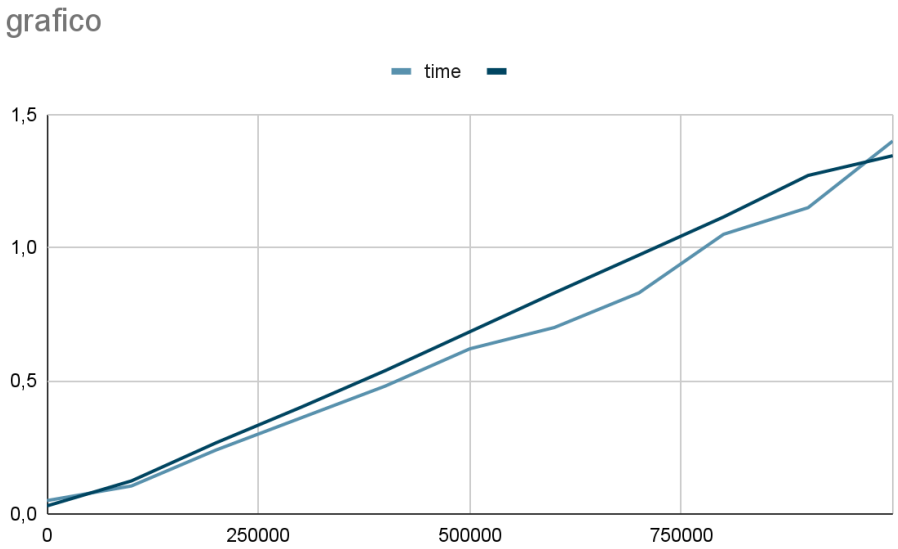
Relatório 2º projecto ASA 2021/2022

Grupo: tp063

Aluno(s): João Vaz (98946) e André Santos (99730)

Avaliação Experimental dos Resultados

Número de instâncias	Tempo(s)
0	0.03
100000	0.124
200000	0.267
300000	0.4
400000	0.538
500000	0.684
600000	0.83
700000	0.972
800000	1.115
900000	1.271
1000000	1.345



Conclui-se que o gráfico gerado está concordante com a análise teórica prevista, porém ligeiramente mais lenta.