Relatório ASA

Mihail Brinza 83533 Ricardo Brancas 83557

23 de Março de 2017

1 Introdução

O objectivo deste projeto é desenvolver um sistema que ajude o utilizador a tarefa de ordenar fotografias por ordem cronológica. O utilizador dá como input ao sistema o número de fotografias e o número de pares das quais o utilizador se lembra da ordem relativa. O software indica ainda caso não seja possível encontrar uma solução coerente ou os casos em que existe mais que uma.

2 Descrição da Solução

Para resolver o problema de ordenar n fotografias em que o utilizador indica e pares dos quais é conhecida a ordem, considerámos um grafo $G = (V, E)$ em que $|V| = n, |E| = e$. Neste grafo, cada fotografia corresponde a um vértice e cada par de fotografias dadas pelo utilizador f_n, f_m corresponde a um arco de v_n para v_m. Assim, assumindo que o grafo é acíclico e que só possui uma ordenação topológica, essa ordenção corresponde à informação pretendida. Por outro lado, caso o grafo contenha ciclos então a informação introduzida pelo utilizador é incoerente, e caso exista mais do que uma ordenação topológica a informação é insuficiente para gerar uma ordem definitiva.

3 Análise Teórica

Para encontrar a ordem topológica utilizámos o algoritmo de Kahn (muito semelhante ao Algortimo da Eliminação de Vértices); este algoritmo divide-se em 3 passos:

1. Inicialização: $O(V)$
   Em primeiro lugar é necessário verificar qual(is) vértice(s) não possui(em) arco(s) de entrada, pois esse(s) será obrigatoriamente o primeiro da ordem topológica. Esse(s) vértice é colocado na *queue*.

2. Ciclo Principal: $O(V + E)$
   O ciclo principal é executado enquanto existirem vértices na *queue*. Caso em alguma iteração do ciclo exista mais que um vértice na *queue*, então existem pelo menos duas ordens topológicas e o problema é insuficiente. Em cada iteração remove-se o vértice m da *queue* e coloca-se na lista final, eliminando depois cada um dos arcos que saem de m. Caso algum dos vértices adjacentes fique sem arcos de entrada, esse vértice é adicionado à *queue*.

3. Verificação: $O(V)$
   No fim é necessário verificar se o grafo ainda tem arcos, pois nesse caso existe obrigatoriamente um ciclo; ou seja, o problema é incoerente.

O algortimo é assim um $O(V + (V + E) + V) = O(V + E)$

```
TOPOLOGICAL-SORT(G)
 1  L = {}
 2  Q = {}
 3  for each vertex u ∈ G.V
 4       if HAS-NO-INCOMING-EDGES(u)
 5            ENQUEUE(Q, u)

 6  while Q ≠ {}
 7       if Q.lenght ≠ 1
 8            error "Multiple topological orders"
 9       u = Q.head
10       DEQUEUE(Q)
11       PUSH-BACK(L,u)
12       for each edge e = (u, m) ∈ u.Adj
13            REMOVE-EDGE(G, e)
14            if HAS-NO-INCOMING-EDGES(m)
15                 ENQUEUE(Q, m)
16  if HAS-EDGES(G)
17       error "Graph has at least a cycle"
18  return L
```

4 Avaliação Experimental

Para testar experimentalmente o nosso algortimo, executámos o nosso programa para valores variando entre $|V| = 5$ e $|V| = 10\,000\,000$ com número de arcos $|E| <= |V| + 1$, considerando apenas problemas bem formados. Concluimos que o tempo de execução do algoritmo é linear ao número de fotografias (e consequentemente de vértices), tal como esperado.