05 - Ordenação topológica

Marco A L Barbosa malbarbo.pro.br

1. Exercício 22.4-1 do CLRS3 ou CLRS2.
2. Exercício 22.4-2 do CLRS3 ou CLRS2. (Dica: programação dinâmica)
3. Exercício 22.4-3 do CLRS3 ou CLRS2.
4. Exercício 22.4-4 do CLRS3 ou CLRS2.
5. Exercício 22.4-5 do CLRS3 ou CLRS2.

Referências

- [CLRS2] - Thomas H. Cormen et al. Introduction to Algorithms. 2nd edition. Capítulo 22.4.
- [CLRS3] - Thomas H. Cormen et al. Introduction to Algorithms. 3rd edition. Capítulo 22.4.

Exercício 5 (22.4-5)

Para cada vértice nós calculamos o grau de entrada no início do algoritmo. Nós mantemos uma fila com os vértices com grau de entrada 0. Enquanto houver vértices na fila, retiramos um vértice u e adicionamos ao final de uma lista L, em seguida diminuímos de 1 o grau de entrada dos vértices adjacentes a u para simular a remoção do vértice u do grafo. Se um vértice ficar com grau de entrada igual a zero, ele é adicionado na fila. No final, se não existirem ciclos no grafo de entrada, a lista L conterá os vértices em ordem topológica. Se existirem ciclos no grafo de entrada, alguns vértices nunca vão ficar com grau de entrada zero e portanto nunca vão entrar na fila. Desta forma o algoritmo irá devolver a lista L incompleta. O pseudo código a seguir implementa esta ideia. Fazendo uma análise agregada podemos concluir que o tempo de execução do algoritmo é $O(V + E)$.

TOPOLOGICAL-SORT2(G)

```
1  for u ∈ G.V
2      u.grau-entrada = 0
3  for u ∈ G.V
4      for v ∈ G.Adj[u]
5          v.grau-entrada = v.grau-entrada + 1
6  Q = ∅
7  L = ∅
8  for u ∈ G.V
9      if u.grau-entrada == 0
10         ENQUEUE(Q, u)
11 while Q ≠ ∅
12     u = DEQUEUE(Q)
13     Adicione u no final da lista L
14     for v ∈ G.Adj[u]
15         v.grau-entrada = v.grau-entrada − 1
16         if v.grau-entrada == 0
17             ENQUEUE(Q, v)
18 return L
```