Grafo Direccionado Acíclico (DAG - Directed Acyclic Graph)

Pedro O. Pérez M., PhD.

Análisis y diseño de algoritmos avanzados
Tecnológico de Monterrey

pperezm@tec.mx

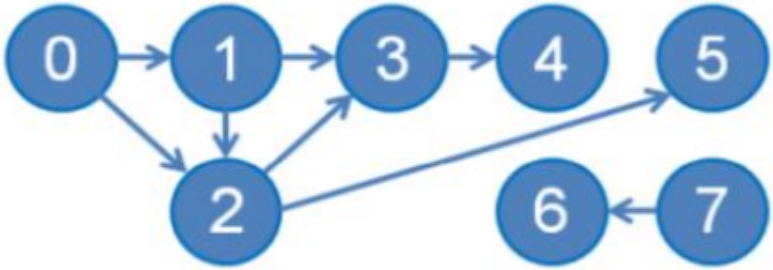
12-2021

Contenido

Un grafo direccionado $G = (V, E)$ acíclico es un grafo dirigido que no tiene ciclos; esto significa que para cada vértice v, no hay un camino directo que empiece y termine en v.

Topological Sort

Un "Topological Sort"de un Grafo Direccionado Acíclico (Directed Acyclic Graph, DAG) es un ordenamiento lineal de los vértices que aparecen en un DAG tal que si el vértice u aparece antes de v es porque existe un arco $(u \rightarrow v)$ en el DAG. Cada DAG tiene al menos, y posiblemente más, "topological sort".


0
1
3
4
5
2
6
7

Procedure 1 DFS2

Input: $u : Vertex, G : Graph, Reached : Set, TS : Stack$

```
Mark u as Explored and add to Reached
for each (u, v) incident to u do
  if v is not marked Explored then
    DFS2(v, G, Reached, TS)
  end if
end for
TS.push(u)
```

Procedure 2 TOPOLOGIGAL_SORT

Input: G : *Graph*
  Reached : *Set*
  TS : *Stack*
  for each *vertex* in G **do**
    if *vertex* is not marked *Explored* **then**
      $DFS2(G, v, Reached, TS)$
    end if
  end for
  while *TS* is not empty **do**
    print $TS.top()$
    $TS.pop()$
  end while

Procedure 3 TOPOLOGIGAL_SORT2

Input: G : *Graph*

Let d be an array of the same length as V; this will hold the shortest-path distances from s. Set $d[s] = 0$, all other d[u] = ínf.
Let p be an array of the same length as V, with all elements initialized to nil. Each $p[u]$ will hold the predecessor of u in the shortest path from s to u.
for each vertex u as ordered in V, starting from s **do**
  for each vertex v into u (i.e., there exists an edge from v to u) **do**
    Let w be the weight of the edge from v to u
    if d[u] > d[v] + w **then**
      d[u] $\leftarrow$ d[v] + w
      p[u] $\leftarrow$ v
    end if
  end for
end for