DEPTH FIRST SEARCH

- Recorrido (en profundidad) de un grafo G finito
- Inicialización : $\forall u \in G$ $u.color := WHITE$; $u.\Pi := NIL$
- Objetivo : ir recorriendo todos los nodos de G marcándolos con dos "horas" de visita $u.d$, $u.f$ que recuerdan la hora de la primera llegada a u y de la culminación de la exploración de la zona accesible desde u "sin volver atrás".
- Variación del coloreado : $u.color = WHILE \Leftrightarrow u.d$ indefinido ;
  $u.color = GRAY \Leftrightarrow (u.d$ definido $\wedge\ u.f$ indefinido) ;
  $u.color = BLACK \Leftrightarrow (u.f$ definido $\wedge\ u.d$ definido)
- Formulación del objetivo :
  $V_u = \{ v \in G \mid v$ es alcanzable desde u a través de un camino que sólo atraviesa nodos que eran blancos en el instante $u.d \}$
  Al concluir la ejecución del algoritmo DFS los campos $u.\Pi$ definen punteros al padre de un bosque formado por árboles que cubren G de forma que cada subárbol con raíz en u T_u contiene exactamente los nodos de V_u. La confección de T_u comienza en el instante $u.d$ y termina en el instante $u.f$, por lo que los nodos de V_u serán exactamente aquéllos que cumplan $u.d \leq v.d < v.f \leq u.f$.
- Propiedades del algoritmo DFS
  - Terminación garantizada : Los dos bucles de DFS y DFS-VISIT tienen tamaño limitado. La recursión termina pues cada llamada pone a un nodo a gris.
  - El bucle de DFS garantiza una llamada a DFS-VISIT con cada nodo de G. La misma pone a gris al nodo al comienzo y finalmente lo pone a negro, lo cual supone que al final $u.d$ y $u.f$ están todos definidos.
  - Cada llamada recursiva a DFS-VISIT se hace con un nodo u que es el final de la rama del bosque construido que forman todos los nodos que en ese momento están en gris. Además esos nodos se corresponden con las llamadas recursivas anidadas que DFS-VISIT que tenemos vivas en cada momento.
    Dem : En efecto, la rama y los nodos grises crecen al llamarse

a DFS-VISIT recursivamente (v.π := u genera el nuevo arco de la rama); la rama se vacía al terminar la correspondiente llamada principal desde DFS, momento en el cual no quedarán nodos grises hasta que en su caso el bucle de DFS encuentre un nodo blanco pendiente con el que efectuar una nueva llamada principal a DFS-VISIT.

- Cada llamada a DFS-VISIT comienza pintando a gris un nodo blanco u e inicializando u.d con la "hora" actual. Desde ese momento y hasta que termine esa activación de DFS-VISIT construimos bajo u un subárbol T_u que contiene exactamente los nodos del vigente conjunto V_u.

Lema: Si desde la llamada a DFS-VISIT correspondiente a u hacemos una llamada recursiva con un nodo u', tendremos $|V_{u'}| < |V_u|$

Dem: Inmediato ya que $u' \in G.Adj[u]$, por lo que $V_{u'} \subseteq V_u - \{u\}$.

Dem (del resultado): Por inducción sobre $|V_u|$

- $|V_u| = 1$. El bucle de DFS-VISIT no encontrará ningún vértice adyacente v blanco, por lo que la llamada a DFS-VISIT terminará tras haber construido el árbol T_u que tiene simplemente a u como raíz, lo que se corresponde con el hecho de que $|V_u| = u$.
- $|V_u| = k+1$. Cada sucesiva llamada recursiva lanzada por el bucle corresponde a un nodo u'_i que en su momento cumplirá $|V_{u'_i}| < |V_u|$. Por h.i. construirá bajo u'_i, que ha quedado conectado como hijo de u al hacer $u'_i.\pi := u$, el subárbol $T_{u'_i}$ que contiene los nodos de $V_{u'_i}$. Tras haber construido una serie $T_{u'_2}, \dots, T_{u'_\ell}$ de tales subárboles el resto de los nodos en $V_u - \{u\}$ seguirán estando blancos y accesibles desde u con un camino formado por nodos tales, pues si en ellos hubiera habido algún nodo de los anteriores conjuntos $V_{u'_i}$ $1 \le i \le \ell$ forzosamente habríamos incluido al nodo en cuestión en el primero de tales $V_{u'_i}$, con lo que el nodo ya no seguiría siendo blanco.

Esto nos asegura que tras terminar la serie de llamadas la colección completa $T_{u'_1}, \ldots, T_{u'_m}$ verificará que en total contendrá a los elementos de $V_u - \{u\}$, con lo que el árbol T_u contendrá en efecto a los elementos de V_u.

Corolario: Si un nodo v verifica $u.d \leq v.d < u.f$, o equivalentemente $u.d \lneq v.f \leq u.f$, o equivalentemente $u.d \leq v.d < v.f \leq u.f$, tendremos $v \in T_u$, y recíprocamente. Como consecuencia, si u y v son nodos independientes tendremos por fuerza $u.f < v.d$ o por contra $v.f < u.d$, o equivalentemente los intervalos $[u.d, u.f]$ y $[v.d, v.f]$ serán disjuntos.

Dem: Mientras u es gris estamos construyendo T_u y una vez que terminamos y definimos $u.f$ ya no podemos alterar T_u.

- Para grafos no dirigidos, para cada arco (u,v) de G tenemos que $v \in T_u$ o bien $u \in T_v$.

Dem: Sin pérdida de generalidad supongamos que $u.d < v.d$. En tal caso $v \in V_u$, y por tanto $v \in T_u$.

Ordenación topológica

- Si G es dirigido y acíclico, para cada arco (u,v) de G tendremos que $v.f < u.f$.

Dem: Si $u.d < v.d$ tendremos $v \in T_u$ y por tanto $v.f < u.f$. Si $v.d < u.d$ no puede ser que $u.d < v.f$, pues en tal caso tendríamos $u \in T_v$, lo que implica que u es accesible desde v en G, lo que es imposible, pues en tal caso el arco (u,v) formaría un ciclo en G. Por tanto $v.f < u.d$, con lo que $v.f < u.f$.

Corolario: La ordenación descendente atendiendo a $x.f$ de los nodos de un grafo dirigido acíclico G tras aplicar DFS produce una ordenación topológica del mismo.

- Si G es dirigido y contiene un ciclo para algún de los arcos (u,v) que lo forman tendremos $u \in T_v$ tras aplicar DFS

Dem: Si v es el nodo del ciclo con $v.d$ mínimo, para el correspondiente arco (u,v) de G tendremos $u \in V_v$, con lo que $u \in T_v$.