

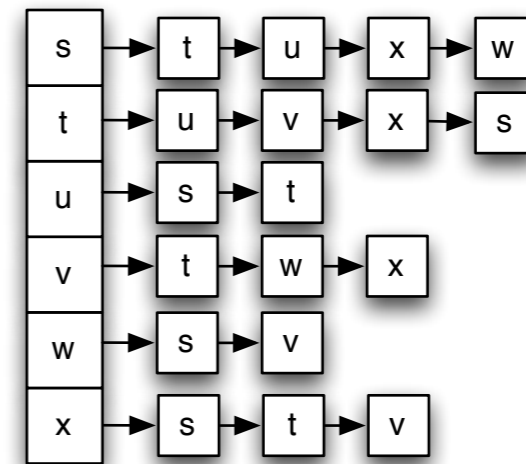
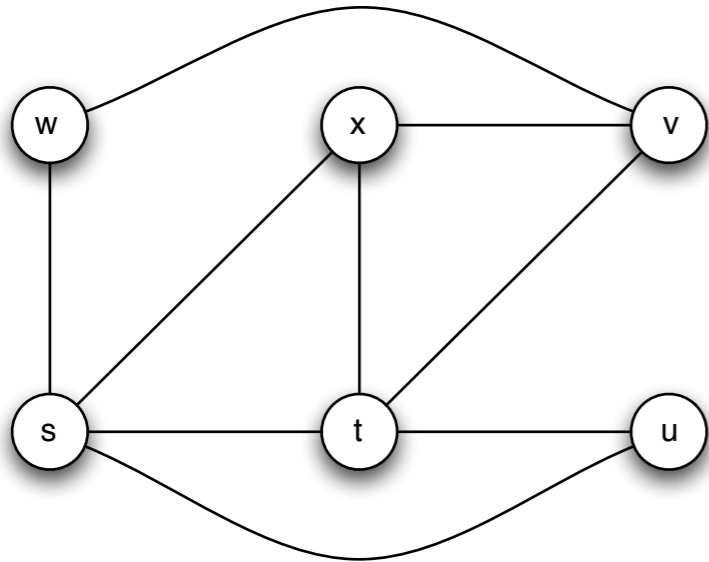
DFS

comp-420

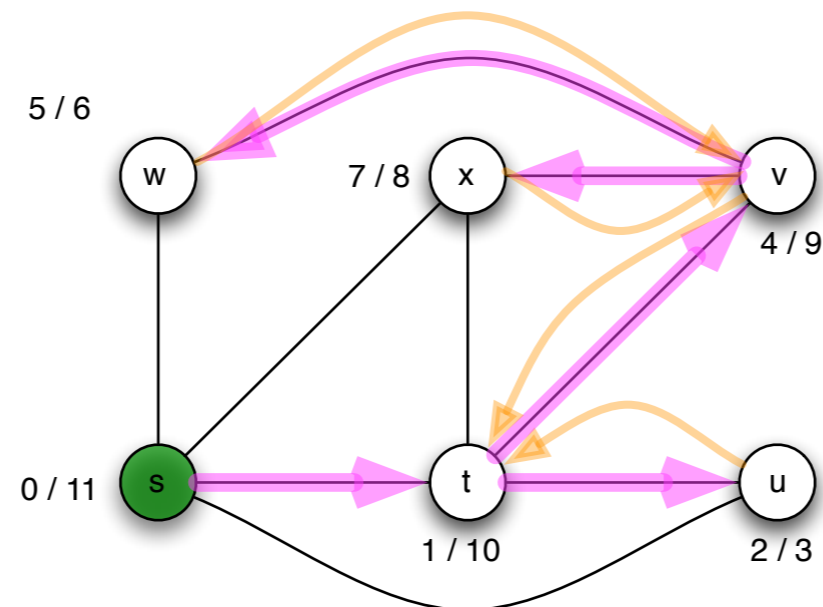
Depth-First Search

- Buscar lo más profundo de una gráfica mientras sea posible.
- Las aristas son exploradas a partir del vértice más recientemente descubierto que tenga aristas inexploradas.
- Cuando todas las aristas de v han sido exploradas, la búsqueda regresa (backtrack) hasta llegar a un vértice con aristas sin explorar.
- Este proceso se repite hasta que todos los vértices han sido descubiertos.

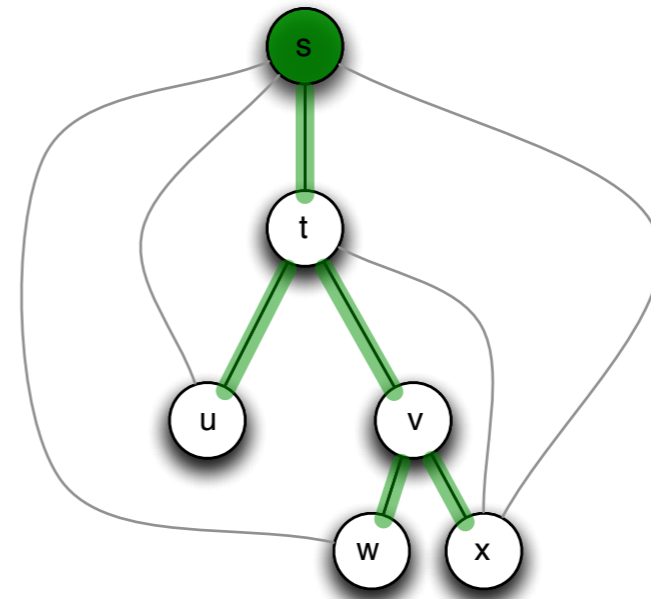
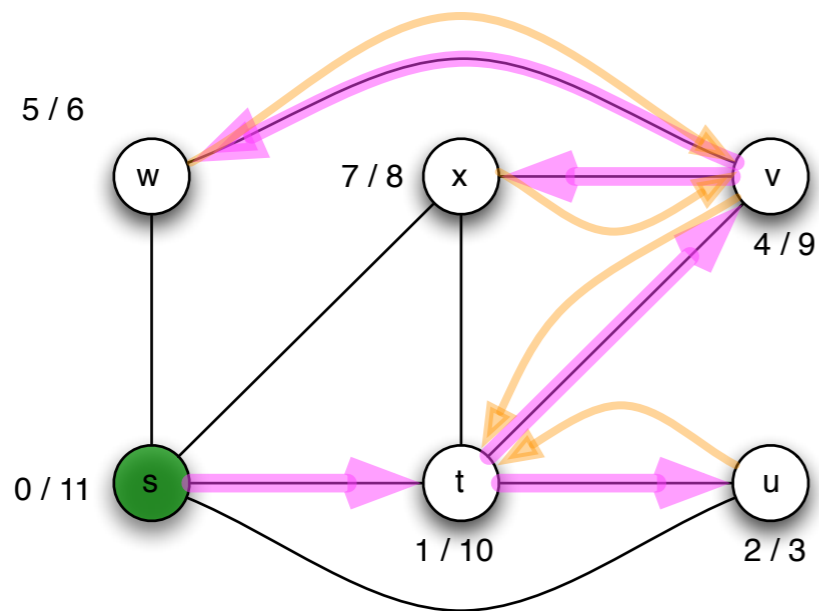
Depth-First Search



¿En qué orden visitamos los nodos?



Depth-First Tree



tree edges
back edges

Depth-First Search

- Supone una representación utilizando listas de adyacencia.
- El **color** de cada vértice $u \in V$ se almacena en la variable **color[u]**.
- El **predecesor de u** se almacena en la variable **$\pi[u]$** .
- La **subgráfica de predecesores** puede formar más de un árbol porque la búsqueda se puede repetir a partir de múltiples fuentes.
- La **subgráfica de predecesores** se define como:

$$G_{\pi} = (V, E_{\pi})$$

$$E_{\pi} = \{(\pi[v], v) : v \in V \text{ y } \pi[v] \neq \text{NIL}\}.$$

- La **subgráfica de predecesores** en DFS forma un **bosque** compuesto de depth-first trees.
- Las aristas en **E_{π}** se llaman tree edges.

Depth-First Search

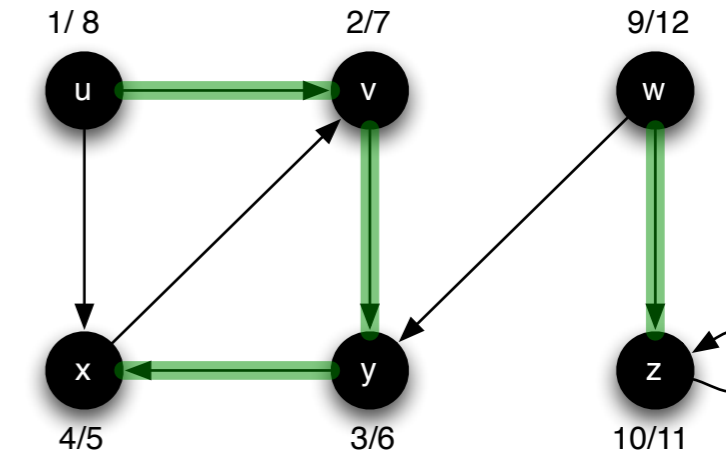
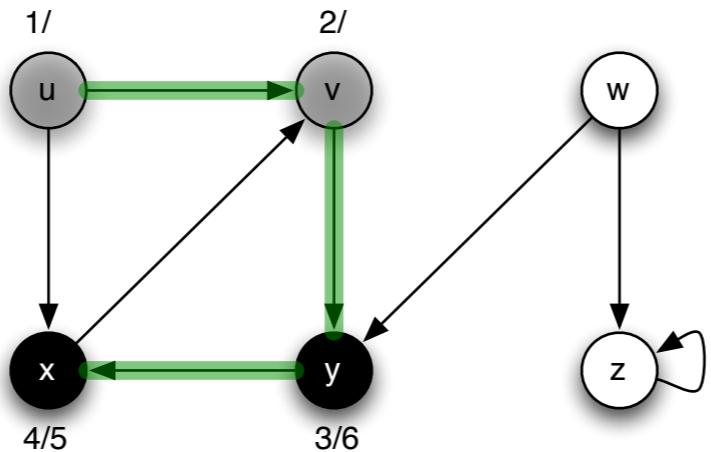
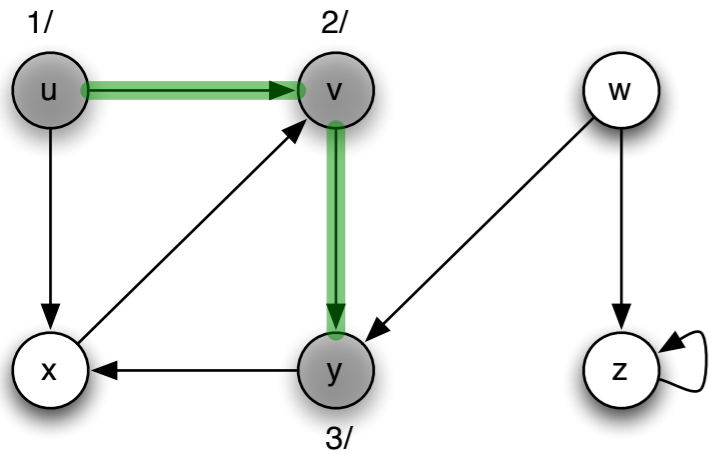
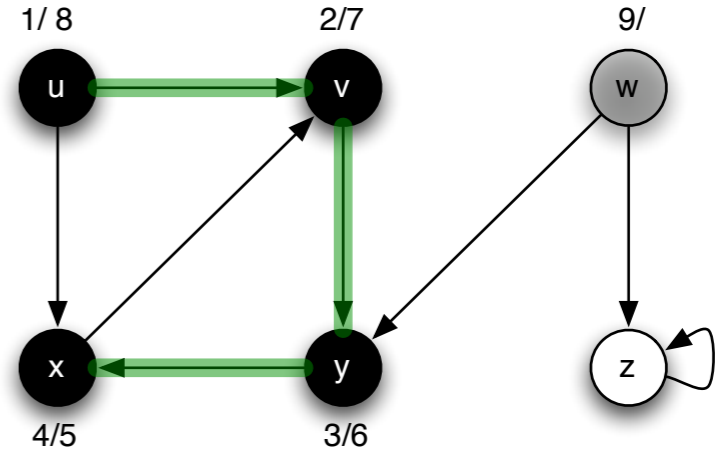
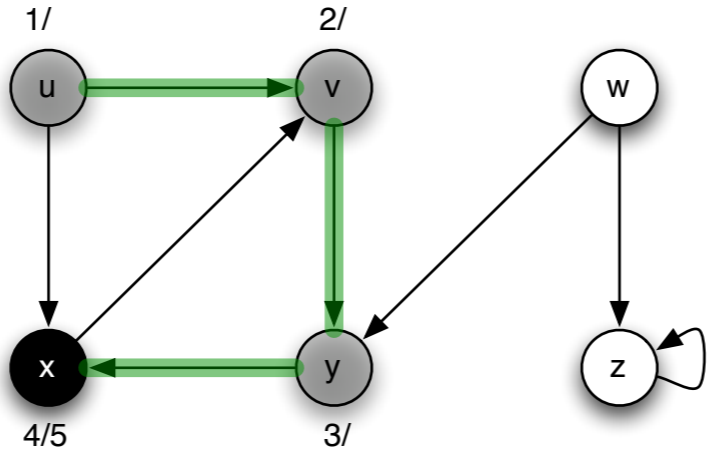
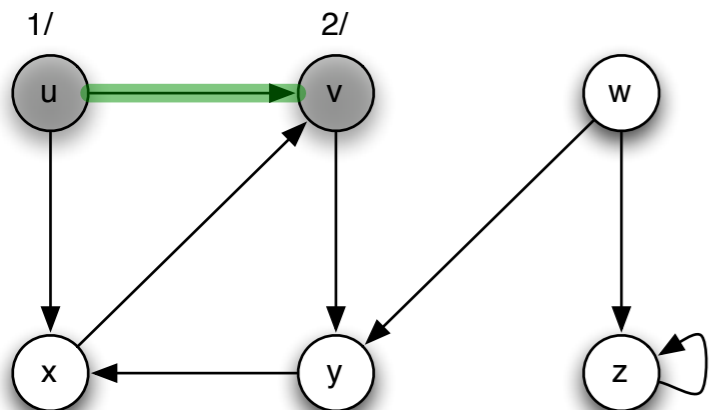
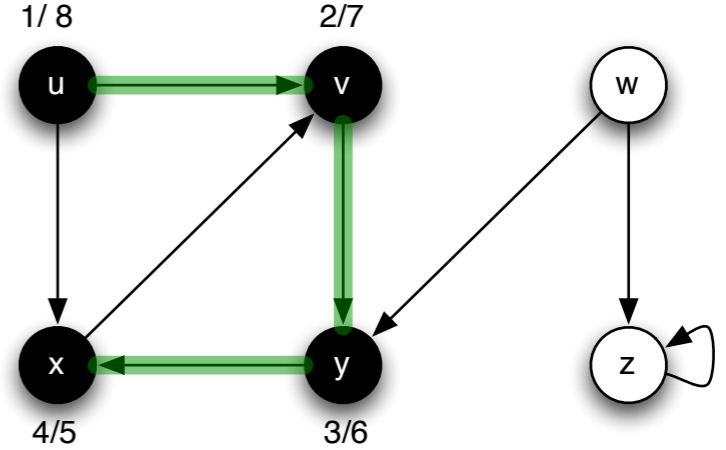
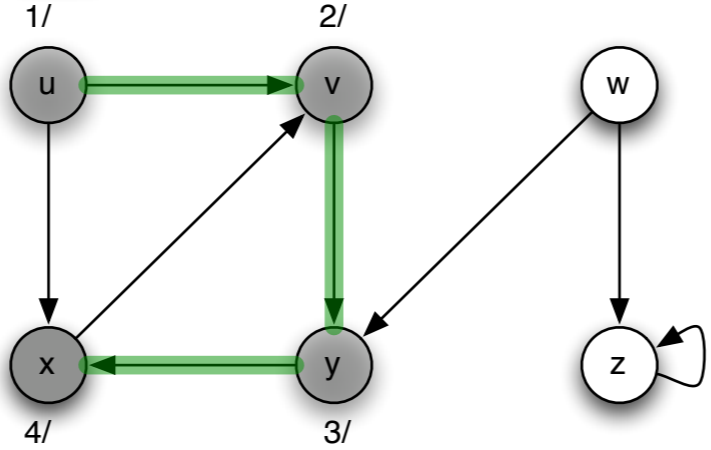
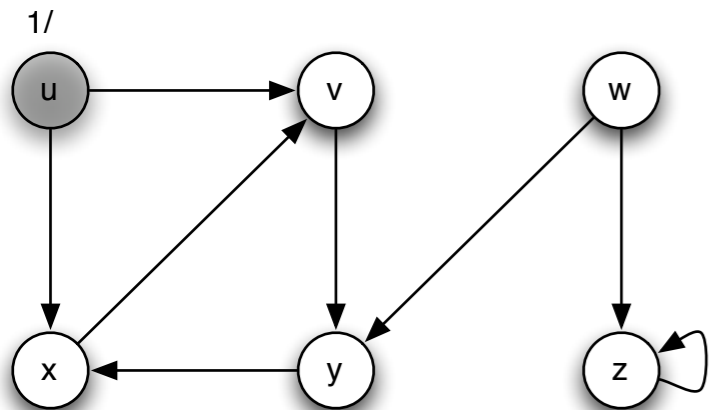
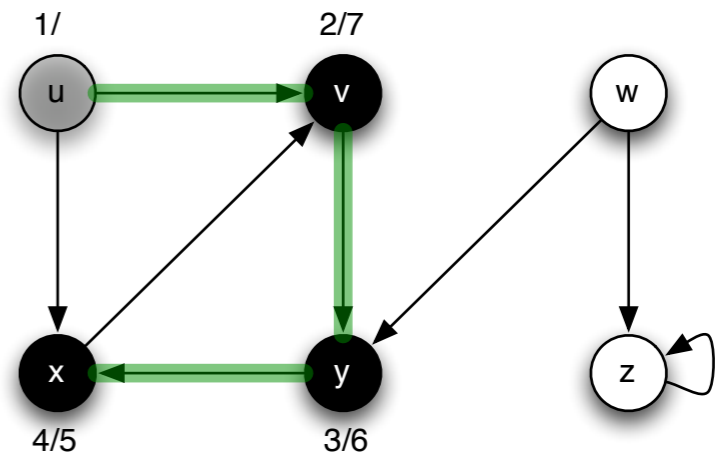
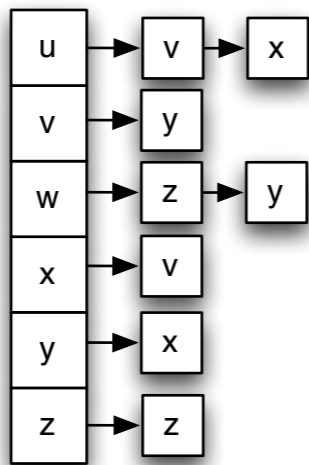
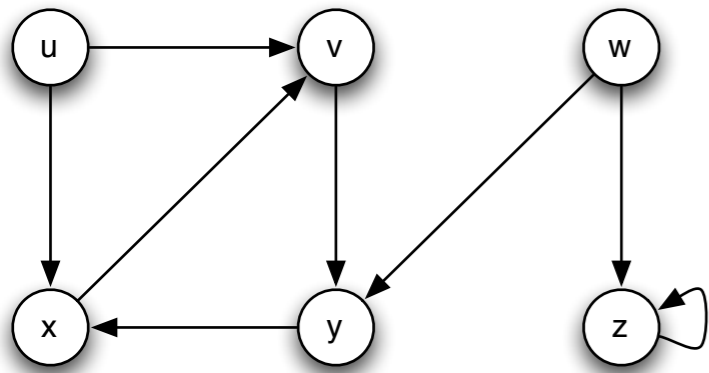
- DFS también mantiene un **sello de tiempo (timestamp)** en cada vértice.
- Cada **vértice v** tiene **dos timestamps**: $d[v]$ que almacena el momento en que v es descubierto (coloreado de gris), y $f[v]$ cuando se termina de examinar su lista de adyacencia (colorea de negro).
- $d[v]$ y $f[v]$ son números enteros entre 1 y $2|V|$, porque...
 - hay un solo evento de descubrimiento y un evento de terminación para cada uno de los $|V|$ vértices.
- Para cada vértice u ,
 - $d[u] < f[u]$.

DFS(G)

```
1  for each vertex  $u \in V[G]$ 
2      do  $color[u] \leftarrow WHITE$ 
3           $\pi[u] \leftarrow NIL$ 
4   $time \leftarrow 0$ 
5  for each vertex  $u \in V[G]$ 
6      do if  $color[u] = WHITE$ 
7          then DFS-VISIT( $u$ )
```

DFS-VISIT(*u*)

```
1  color[u] ← GRAY // White vertex u has just been discovered.
2  time ← time + 1
3  d[u] ← time
4  for each v ∈ Adj[u] // Explore edge (u, v).
5      do if color[v] = WHITE
6          then  $\pi$ [v] ← u
7              DFS-VISIT(v)
8  color[u] ← BLACK // Blacken u; it is finished.
9  f[u] ← time ← time + 1
```

DFS: Tiempo de Cálculo

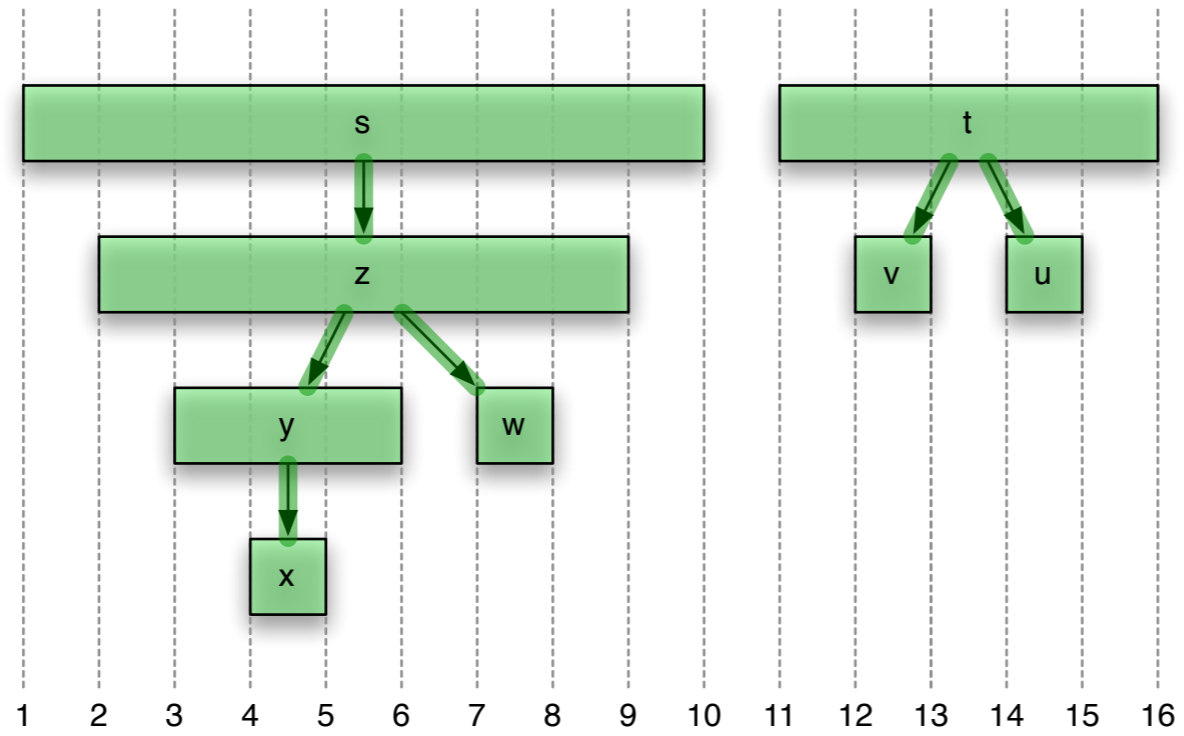
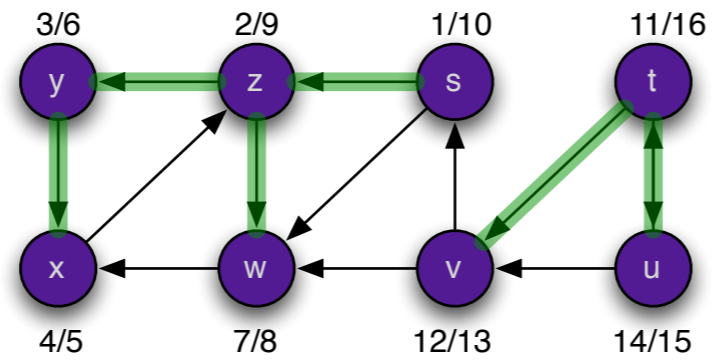
- ¿Cuál es el tiempo de cálculo de los ciclos de las líneas 1 a 3?
- ¿y de las líneas 5 a 7? (sin contar el tiempo que toma la llamada a `DFS-VISIT(v)`)
- $\Theta(V)$
- El procedimiento `DFS-VISIT(v)` es llamado exactamente
 - una vez para cada vértice $v \in V$ porque se llama sólo en los vértices blancos y la primera acción es pintarlos de gris.
- Durante la ejecución de `DFS-VISIT(v)`, el ciclo de las líneas 4 a 7 se ejecuta...
 - `Adj[v]` veces.

DFS: Tiempo de Cálculo

- Ya que $\sum_{v \in V} |Adj[v]| = \Theta(E)$, el **costo total** de ejecutar las líneas 4 a 7 de **DFS-VISIT(v)** es $\Theta(E)$.
- El costo total de DFS es por lo tanto:
 - $\Theta(V+E)$.

Propiedades de DFS

- La subgráfica de predecesores, G_π forma un bosque de árboles ya que la estructura de los árboles produce exactamente la estructura de llamadas recursivas de $\text{DFS-VISIT}(v)$.
- $u = \pi[v]$ si y solo si $\text{DFS-VISIT}(v)$ fue llamada durante una búsqueda en la lista de adyacencia de u .
- el vértice v es un descendiente de u en el bosque si y solo si v es descubierto durante el tiempo en que u es gris.
- Los tiempos de descubrimiento y terminación de un vértice tienen estructura de paréntesis:
 - si representamos el descubrimiento por “(u” y la terminación por “u)”, el historial del algoritmo estará propiamente anidado.



Teorema del Paréntesis

- En una búsqueda en profundidad (en una gráfica dirigida o no dirigida) $G=(V,E)$, para cualquier par de vértices u y v , exactamente una de las siguientes condiciones se cumple:
 - Los intervalos $[d[u],f[u]]$ y $[d[v],f[v]]$ están completamente disjuntos y ningún vértice u o v son descendientes entre sí en el bosque depth-first.
 - El intervalo $[d[u], f[u]]$ está enteramente contenido en el intervalo $[d[v], f[v]]$, y u es descendiente de v en un árbol depth-first.
 - El intervalo $[d[v], f[v]]$ está enteramente contenido en el intervalo $[d[u], f[u]]$, y v es descendiente de u en un árbol depth-first.

Teorema del Paréntesis

- Para probar este teorema pensar en el caso en que $d[u] < d[v]$ y en sus subcasos:
 - $d[v] < f[u]$: v ha sido descubierto mientras u era todavía gris. Esto implica que v es descendiente de u . Además como v fue descubierto más recientemente que u , sus nodos adyacentes serán explorados antes que el algoritmo regrese y termine de explorar u . En este caso el intervalo $[d[v], f[v]]$ estará contenido en $[d[u], f[u]]$.
 - $d[v] > f[u]$: los intervalos están disjuntos, ninguno era gris, ninguno es descendiente de otro.
 - el caso en que $d[v] < d[u]$ es simétrico.
- Corolario: un vértice v es un descendiente propio de otro vértice u en el bosque de profundidad para una gráfica (dirigida o no) si y solo si:

$$d[u] < d[v] < f[v] < f[u]$$