CC184 - Complejidad Algorítmica

Tema:	Algoritmo de Kosaraju
Formato:	Esquema de Aprendizaje
Elaborado por:	Robert Zubieta
Fuente:	Propia

Kosaraju Algorithm

I. Alcance

Componentes Fuertemente Conexos (SCC: Strongly Connected Components)

- Un grafo dirigido es fuertemente conexo si existe un camino entre todos los pares de vértices. Cada vértice puede llegar al otro vértice a través del camino dirigido.

Para todo (u,v): Existe un **camino de u a v** Y un **camino de v a u**

- Un componente fuertemente conectado (SCC) de un grafo dirigido es un subgrafo máximo fuertemente conectado.

Los componenetes SCC se pueden encontrar usando el **Algoritmo de Kosaraju**.

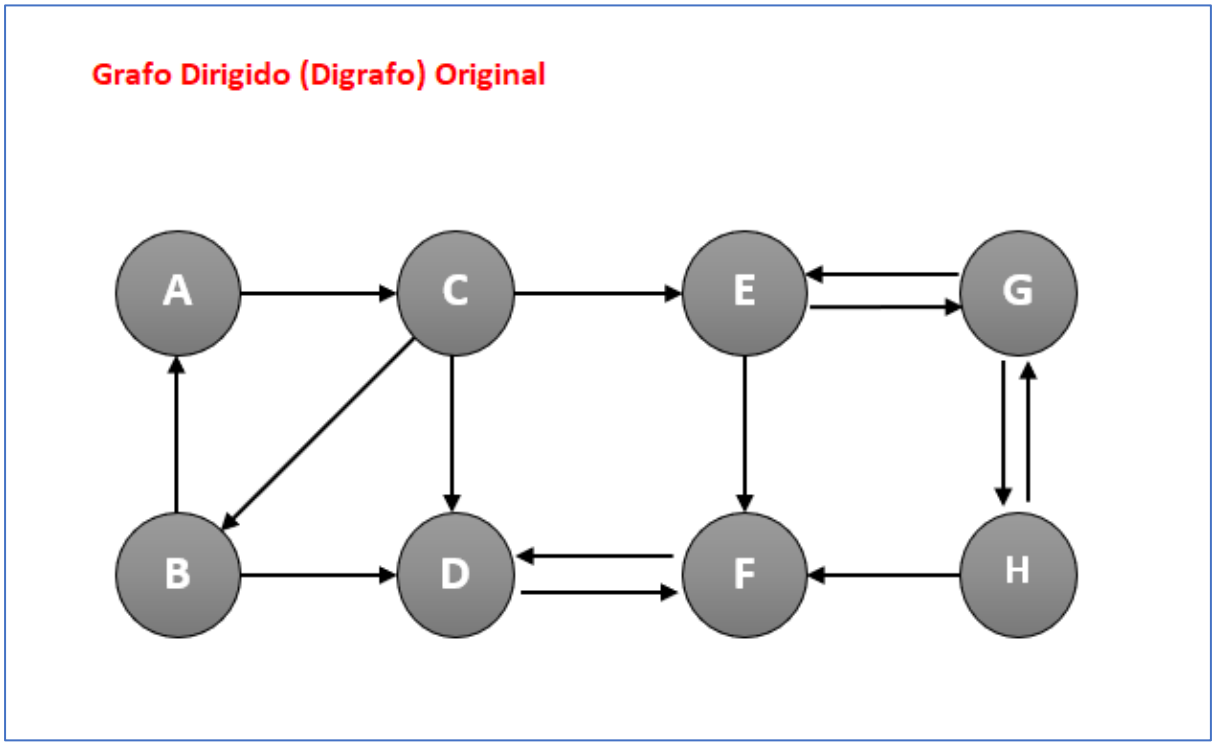
Algoritmo de Kosaraju

1. Realizar una búsqueda primero en profundidad (DFS) en todo el grafo.
2. Invertir el grafo original => Grafo Transpuesto: cambiar el sentido de las aristas.
3. Realizar otra búsqueda primero en profundidad (DFS) pero ahora sobre el grafo invertido

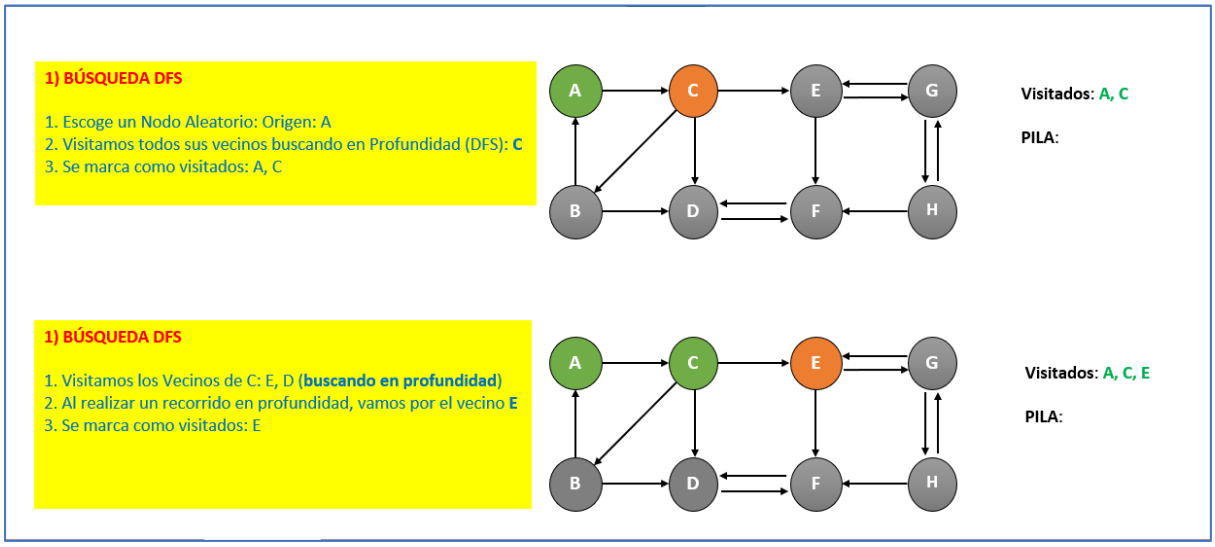
Aplicaciones:

Ejemplo: Facebook. Cada acción de una persona se almacena como un Nodo de un Grafo. Todas las personas que tienen intereses comunes puedan ser manejados como Componentes Fuertemente Conexos.

II. Grafo (Ejemplo usado en el desarrollo)

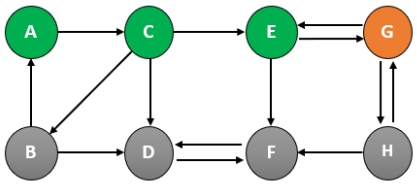


III. Paso 1: DFS a Grafo Original



1) BÚSQUEDA DFS

1. Visitamos los Vecinos de E: F, G
2. Al realizar un recorrido en profundidad, vamos por el vecino **G**
3. Se marca como visitados: G

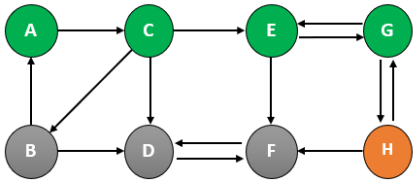


Visitados: A, C, E, G

PILA:

1) BÚSQUEDA DFS

1. Visitamos los Vecinos de G: E, H
2. Al realizar un recorrido en profundidad, vamos por el vecino **H**. E no se considera porque ya está visitado
3. Se marca como visitados: H

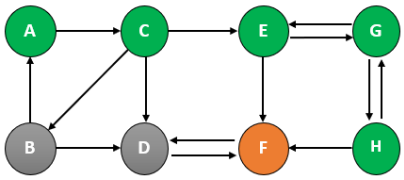


Visitados: A, C, E, G, H

PILA:

1) BÚSQUEDA DFS

1. Visitamos los Vecinos de H: F, G
2. Al realizar un recorrido en profundidad, vamos por el vecino **F**. G no se considera porque ya está visitado
3. Se marca como visitados: F

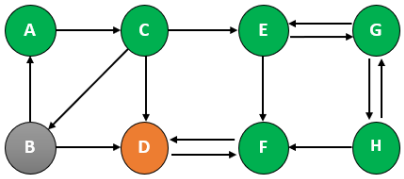


Visitados: A, C, E, G, H, F

PILA:

1) BÚSQUEDA DFS

1. Visitamos los Vecinos de F: D
2. Al realizar un recorrido en profundidad, vamos por el vecino **D**.
3. Se marca como visitados: D

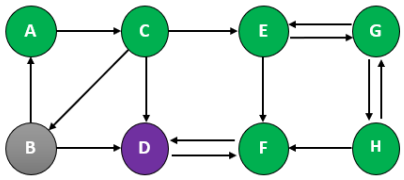


Visitados: A, C, E, G, H, F, D

PILA:

1) BÚSQUEDA DFS

1. Visitamos los Vecinos de D: F
2. Observamos que los vecinos de D (F) ya fueron visitados
3. Agregamos D a la **Pila** y **retrocedemos** (hacia F).

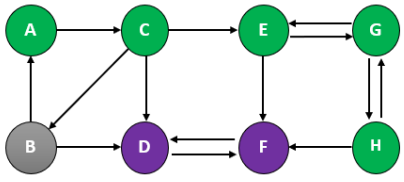


Visitados: A, C, E, G, H, F, D

PILA: D

1) BÚSQUEDA DFS

1. Visitamos los Vecinos de F: D
2. Observamos que los vecinos de F ya fueron visitados
3. Agregamos F a la Pila y retrocedemos (hacia H).

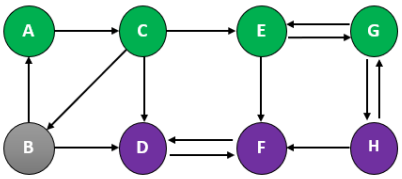


Visitados: A, C, E, G, H, F, D

PILA: D, F

1) BÚSQUEDA DFS

1. Visitamos los Vecinos de H: F,G
2. Observamos que los vecinos de H ya fueron visitados
3. Agregamos H a la Pila y retrocedemos (hacia G).

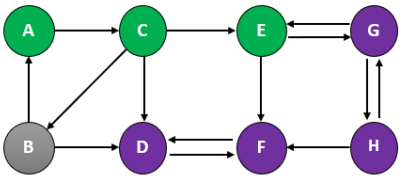


Visitados: A, C, E, G, H, F, D

PILA: D, F, H

1) BÚSQUEDA DFS

1. Visitamos los Vecinos de G: E, H
2. Observamos que los vecinos de G ya fueron visitados
3. Agregamos G a la Pila y retrocedemos (hacia E).

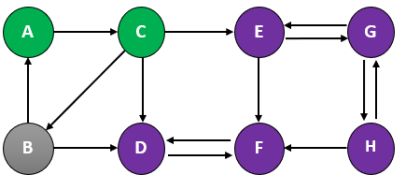


Visitados: A, C, E, G, H, F, D

PILA: D, F, H, G

1) BÚSQUEDA DFS

1. Visitamos los Vecinos de E: F, G
2. Observamos que los vecinos de E ya fueron visitados
3. Agregamos E a la Pila y retrocedemos (hacia C).

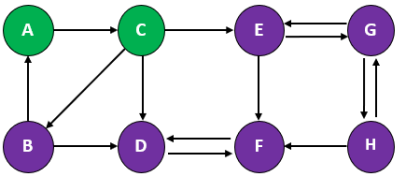


Visitados: A, C, E, G, H, F, D

PILA: D, F, H, G, E

1) BÚSQUEDA DFS

1. Visitamos los Vecinos de C: B, D
2. Observamos que solo B, no ha sido visitado
3. No se agrega C a la PIla
3. Recorremos en DFS a C => hallamos sus vecinos en profundidad: B
4. Se marca B como visitado
5. Los vecinos de B ya estan visitados => Agregamos B a la Pila
6. Retrocedemos (hacia C)

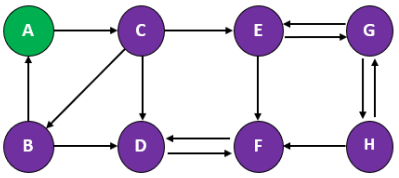


Visitados: A, C, E, G, H, F, D, B

PILA: D, F, H, G, E, B

1) BÚSQUEDA DFS

1. Visitamos los Vecinos de C: B, D
2. Observamos que los vecinos de C ya fueron visitados
3. Agregamos C a la Pila y retrocedemos (hacia A).

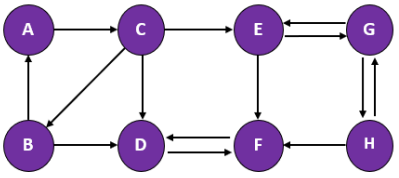


Visitados: A, C, E, G, H, F, D, B

PILA: D, F, H, G, E, B, C

1) BÚSQUEDA DFS

1. Visitamos los Vecinos de A: C
2. Observamos que los vecinos de A ya fueron visitados
3. Agregamos A a la Pila. Ya no hay donde retroceder.
4. Verificamos que hemos visitado todos los Nodos del Grafo
5. **Hallar el Grafo Transpuesto**

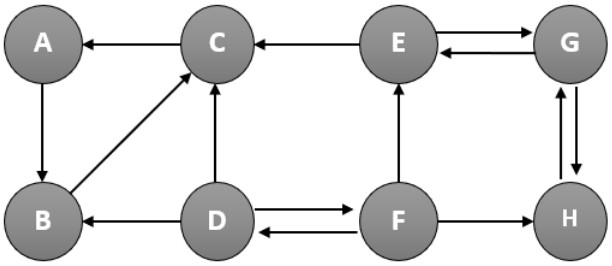


Visitados: A, C, E, G, H, F, D, B

PILA: D, F, H, G, E, B, C, A

IV. Paso 2: Invertir el Grafo Original: Grafo Transpuesto

Grafo Dirigido (Digrafo) Transpuesto Cambia el sentido de las Aristas Dirigidas



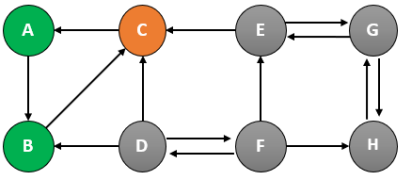
Visitados:

PILA: D, F, H, G, E, B, C, A

V. Paso 3: DFS a Grafo Transpuesto

3) BÚSQUEDA DFS

1. Obtenemos elemento de la Pila: A
2. Visitamos los Vecinos de A: B
3. Se marca B como visitado

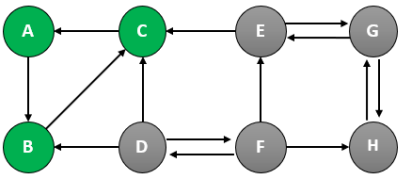


Visitados: A, B

PILA: D, F, H, G, E, B, C

3) BÚSQUEDA DFS

1. Visitamos los Vecinos de B: C
2. Se marca C como visitado
3. Agregamos C a la Pila y retrocedemos.

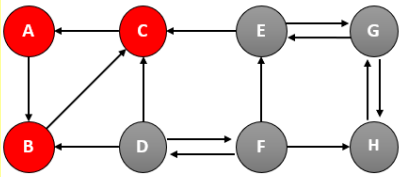


Visitados: A, B, C

PILA: D, F, H, G, E, B, C

3) BÚSQUEDA DFS

1. Visitamos los Vecinos de C: A
2. Observamos que los vecinos de C ya fueron visitados (A)
3. Observamos que hemos regresado al nodo A (sacado de la Pila).
4. Se concluye que este grupo **{A, B, C}** son **componentes fuertemente conexos ya que se pueden hacer dos DFS una en el grafo original y la otra en el grafo transpuesto.**

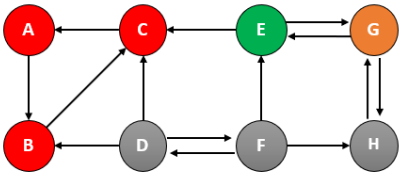


Visitados: A, B, C

PILA: D, F, H, G, E, B, C

3) BÚSQUEDA DFS

1. Obtenemos elemento de la Pila: C => Ya está visitada
2. Obtenemos elemento de la Pila: B => Ya está visitada
3. Obtenemos elemento de la Pila: E
4. Visitamos los Vecinos de E: C, G
5. Al realizar un recorrido en profundidad, vamos por el vecino G. C no se considera porque ya está visitado.
6. Se marca E como visitado

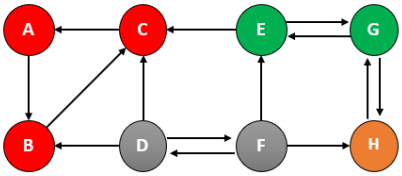


Visitados: A, B, C, E

PILA: D, F, H, G

3) BÚSQUEDA DFS

1. Visitamos los Vecinos de G: E, H
2. E ya visitado. Vamos por el vecino H
2. Se marca G como visitado

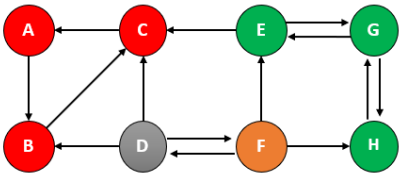


Visitados: A, B, C, E, G

PILA: D, F, H, G

3) BÚSQUEDA DFS

1. Visitamos los Vecinos de H: G
2. G ya visitado.
2. Se marca H como visitado

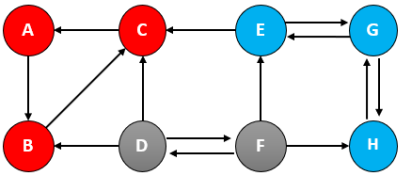


Visitados: A, B, C, E, G, H

PILA: D, F, H, G

3) BÚSQUEDA DFS

1. Observamos que los vecinos de H ya fueron visitados. No tiene nodos por visitar.
2. Se concluye que este grupo **{E, G, H}** son **componentes fuertemente conexos ya que se pueden hacer dos DFS una en el grafo original y la otra en el grafo transpuesto.**

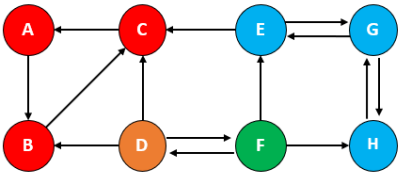


Visitados: A, B, C, E, G, H

PILA: D, F, H, G

3) BÚSQUEDA DFS

1. Obtenemos elemento de la Pila: G => Ya está visitada
2. Obtenemos elemento de la Pila: H => Ya está visitada
3. Obtenemos elemento de la Pila: F
4. Visitamos los Vecinos de F: D, E, H
5. E, H ya visitado. Vamos por el vecino D
6. Se marca F como visitado

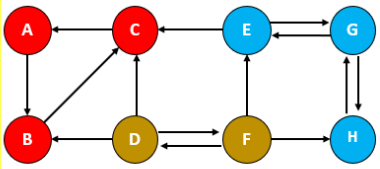


Visitados: A, B, C, E, G, H, F

PILA: D

3) BÚSQUEDA DFS

1. Observamos que los vecinos de D ya fueron visitados. No tiene nodos por visitar.
2. Se marca D como visitado
3. Se concluye que este grupo **{D, F}** son **componentes fuertemente conexos ya que se pueden hacer dos DFS una en el grafo original y la otra en el grafo transpuesto.**



Visitados: A, B, C, E, G, H, F, D

PILA: D

3) BÚSQUEDA DFS

1. Obtenemos elemento de la Pila: D => Ya está visitada
2. Se finaliza proceso porque todos los nodos ya fueron visitados.

Componentes Fuertemente Conexos:

{ A, B, C }
{ E, G, H }
{ D, F }

FIN