***Resumen teoría Ayed***

***Arboles (Implementacion: Filmina 14)***

***Arboles Binarios ()***

***Arboles de Expresion ()***

***Arboles Generales ()***

***Cola de Implementacion ()***

***Heap (Implementacion: Filmina 14)***

***TIEMPOS (Implementacion: Filmina 14)***

***Grafos (Implementacion: Filmina 14)***

Terminología

* Grafo→ modelo para representar relaciones entre elementos de un conjunto.
* Grafo: (V,E), V es un conjunto de vértices o nodos, con una relación entre ellos; E es un conjunto de pares (u,v), u,v Є V , llamados aristas o arcos.
* Un grafo es un conjunto de nodos que mantienen relaciones entre ellos. Podemos definir a un grafo como un par de conjuntos finitos G=(V,A) donde V es el conjunto finito de vértices y A es el conjunto finito de Aristas. Por lo general, los vértices son nodos de procesamiento o estructuras que contienen algún tipo de información mientras que las aristas determinan las relaciones entre nodos. Las artistas también pueden tener algún tipo de información asociada (distancia, peso, etc.) en cuyo caso es un grafo pesado.
* En cualquiera de los grafos, las relaciones las determinan las aristas.
* Grafo dirigido: la relación sobre V no es simétrica. Arista ≡ par ordenado (flechas)
  + Si se utilizan flechas para conectar los nodos decimos que el grafo es dirigido (también llamado digrafo).
* Grafo no dirigido: la relación sobre V es simétrica. Arista ≡ par no ordenado (sin flechas)
  + Si la conexión entre los vértices no tiene dirección, el grafo es no dirigido.

-------------------------------------------------------------------------------------------------------------------

* V es **adyacente** a u si existe una arista (u,v) Є E.:
  + en un grafo no dirigido, (u,v) Є E incide en los nodos u, v.
  + en un grafo dirigido, (u,v) Є E incide en v, y parte de u.
* En grafos no dirigidos::
  + El grado de un nodo: número de arcos que inciden en él.
* En grafos dirigidos:
  + existen el grado de salida (**grado\_out**) y el grado de entrada (**grado\_in**).
    - el grado\_out es el número de arcos que parten de él y
    - el grado\_in es el número de arcos que inciden en él.
  + El grado del vértice será la suma de los grados de entrada y de salida.
* **Grado de un grafo**: máximo grado de sus vertices:

-------------------------------------------------------------------------------------------------------------------

* **Camino** desde u Є V a v Є V : secuencia v1 , v2 , ... , vk tal que u=v1 , v=v k , y (vi−1,vi ) Є E, para i = 2, ... ,k.
* Longitud de un camino: número de arcos del camino

-------------------------------------------------------------------------------------------------------------------

* **Camino simple:** camino en el que todos sus vértices, excepto, tal vez, el primero y el último, son distintos. P1 es un camino simple desde U a Z..

-------------------------------------------------------------------------------------------------------------------

* **Ciclo:** camino desde v1 , v2 , ... , vk tal que v1=vk
  + El ciclo es simple si el camino es simple.
  + **Bucle**: ciclo de longitud 1.
  + **Grafo acíclico**: grafo sin ciclos.

-------------------------------------------------------------------------------------------------------------------

* Dado un grafo G=(V, E), se dice que G´=(V´, E´) es un **subgrafo** de G, si V’⊆ V y E’ ⊆ E.
* Un **subgrafo inducido** por V’ ⊆ V : G’ = (V’,E’) tal que E’ = {(u,v) ∈ E | u,v ∈ V’}.

-------------------------------------------------------------------------------------------------------------------

* Un **grafo ponderado**, pesado o con costos: cada arco o arista tiene asociado un valor o etiqueta..

**Conectividad en grafos no dirigidos**

* Un grafo no dirigido es **conexo** si hay un camino entre cada par de vértices...

-------------------------------------------------------------------------------------------------------------------

* Un **bosque** es un grafo sin ciclo
* Un **árbol libre** es un bosque conexo.
* Un árbol es un árbol libre en el que un nodo se ha designado como raíz.

**Propiedades**

* Sea G un grafo no dirigido con n vértices y m arcos, entonces:
  + Σv Є G deg(v) = 2\*m
  + Siempre: m≤ (n\*(n-1))/2
  + Si G conexo: m≥n-1
  + Si G árbol: m=n-1
  + Si G bosque: m≤n-1

**Conectividad en grafos dirigidos**

* V es **alcanzable desde** u, si existe un camino de u a v:
* Un grafo dirigido se denomina **fuertemente conexo** si existe un camino desde cualquier vértice a cualquier otro vértice
* Si un grafo dirigido no es fuertemente conexo, pero el grafo subyacente (sin sentido en los arcos) es conexo, el grafo es **débilmente conexo.**

**Componentes conexas**

* En un grafo no dirigido, una **componente** conexa es un subgrafo conexo tal que no existe otra componente conexa que lo contenga.
* Es un **subgrafo conexo máximal**.
* Un grafo no dirigido es **no conexo** si está formado por varias componentes conexas.

**Componentes fuertemente conexas**

* En un grafo dirigido, una **componente fuertemente conexa**, es el máximo subgrafo fuertemente conexo. Un grafo dirigido es **no fuertemente conexo** si está formado por varias componentes fuertemente conexas.

**Representaciones**

* **Matriz de adyacencias:** el grafo se representa como una matriz de |V| x |V|, con valores enteros (o de otro tipo de dato).
* **Lista de adyacencias:** el grafo G=(V,A) se representa como un arreglo/lista de |V| de vértices.

**(Mirar filmina 1)**

Recorridos

* **En profundidad: DFS (Depth First Search).**
* **en amplitud: BFS (Breath First Search)**
* **Bosque de expansión DFS**

**--------------------------------------------------------------------------------------------------------------**

**DFS**

\*Generalización del recorrido pre orden de un árbol.

\*El recorrido del DFS depende del orden en que aparecen los vértices en las listas de adyacencia

Estrategia:

* + Partir de un vértice determinado v.
  + Cuando se visita un nuevo vértice, explorar cada camino que salga de él.
  + Hasta que no se haya finalizado de explorar uno de los caminos no se comienza con el siguiente.
  + Un camino deja de explorarse cuando se llega a un vértice ya visitado
  + Si existían vértices no alcanzables desde v el recorrido queda incompleto; entonces, se debe seleccionar algún vértice como nuevo vértice de partida, y repetir el proceso

Esquema recursivo: dado G = (V , E)

* + Marcar todos los vértices como no visitados.
  + Elegir vértice u como punto de partida
  + Marcar u como visitado.
  + ∀ v adyacente a u,(u,v) Є E, si v no ha sido visitado, repetir recursivamente (3) y (4) para v.
    - Finalizar cuando se hayan visitado todos los nodos alcanzables desde u.
    - Si desde u no fueran alcanzables todos los nodos del grafo: volver a (2), elegir un nuevo vértice de partida v no visitado, y repetir el proceso hasta que se hayan recorrido todos los vértices.

Tiempo de Ejecución

* + G (V, E) se representa mediante listas de adyacencia.
  + El método dfs(v) se aplica únicamente sobre vértices no visitados → sólo una vez sobre cada vértice.
  + dfs(v) depende del número de vértices adyacentes que tenga (longitud de la lista de adyacencia). → el tiempo de todas las llamadas a dfs(v): O(|E|)
  + añadir el tiempo asociado al bucle de main\_dfs(grafo): O (|V|)). ⇒ Tiempo del recorrido en profundidad es O(|V|+|E|).

---------------------------------------------------------------------------------------------------

**BFS**

\*Generalización del recorrido por niveles de un árbol.

Estrategia:

* + Partir de algún vértice u, visitar u y, después, visitar cada uno de los vértices adyacentes a u.
  + Repetir el proceso para cada nodo adyacente a u, siguiendo el orden en que fueron visitados

---------------------------------------------------------------------------------------------------

**Bosque de expansión del DFS**

* El recorrido **no es único**: depende del nodo inicial y del orden de visita de los adyacentes.
* El orden de visita de unos nodos a partir de otros puede ser visto como un árbol: **árbol de expansión (o abarcador) en profundidad asociado al grafo.**
* Si aparecen varios árboles: **bosque de expansión (o abarcador) en profundidad.**

**Clasificación de los arcos de un grafo dirigido en el bosque de expansión de un DFS**

* **Arcos tree (del árbol**): son los arcos en el bosque depth-first-search, arcos que conducen a vértices no visitados durante la búsqueda.
* **Arcos forward**: son los arcos u→v que no están en el bosque, donde v es descendiente, pero no es hijo en el árbol
* **Arcos backward**: son los arcos u→v, donde v es antecesor en el árbol. Un arco de un vértice a si mismo es considerado un arco back.
* **Arcos cross**: son todos los otros arcos u→v, donde v no es ni antecesor ni descendiente de u. Son arcos que pueden ir entre vértices del mismo árbol o entre vértices de diferentes árboles en el bosque depth-first-search