TEMA 3: Algoritmos de Grafos

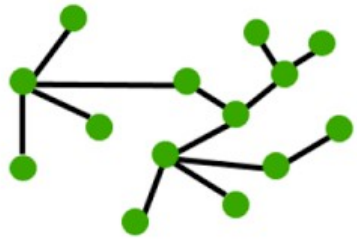
Objetivo: El alumno aplicará las formas de representar y operar los grafos y listas no lineales en la computadora

3.1 Generalidades

- En una estructura lineal (EDA I), cada elemento sólo puede ir enlazado al siguiente o al anterior.
- Las listas no lineales, se caracterizan porque no necesariamente existe una relación de sus elementos; un elemento se relaciona con cero, uno, o más elementos.
- A las estructuras de datos no lineales se les llama también estructuras de datos multi-enlazadas.

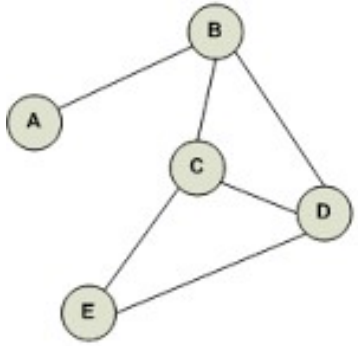
3.1 Generalidades

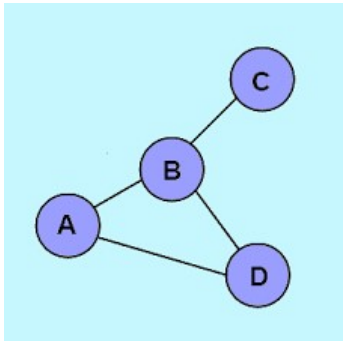
- Pueden representar elementos de tipo jerárquico o simplemente elementos del mismo tipo relacionados entre sí por alguna característica en particular.
- La representación de este tipo de estructuras en la computadora depende del tipo de relación entre los elementos.



3.1.1 Conceptos y definiciones de gráficas

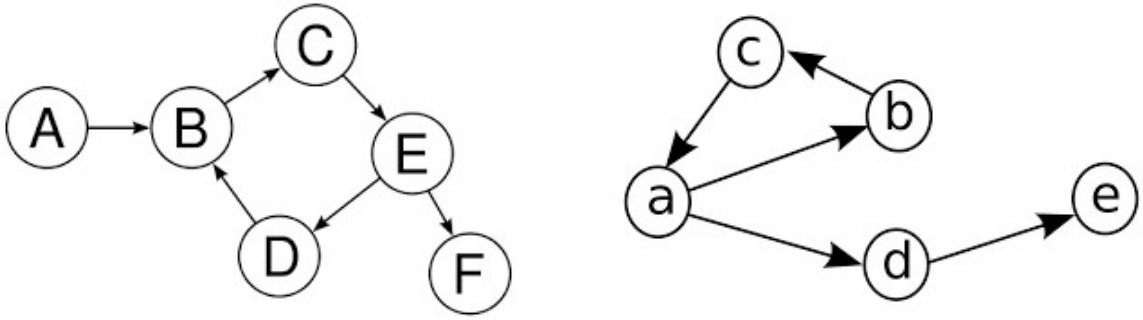
- Una gráfica (no dirigida) G consiste en un conjunto V de vértices y un conjunto E de aristas tal que cada arista e, que pertenece a E se asocia con un par no ordenado de vértices.





3.1.1 Conceptos y definiciones de gráficas

- Una gráfica dirigida G consiste en un conjunto V de vértices (o nodos) y un conjunto E de aristas tales que cada arista ***e*** que pertenece a E esta asociada con un par ordenado de vértices.



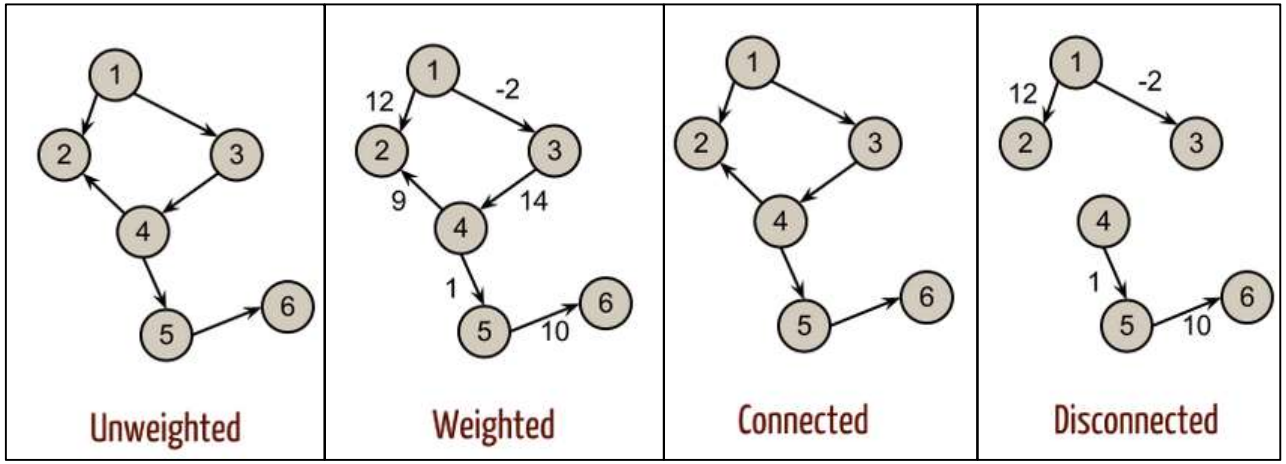
3.1.1 Conceptos y definiciones de gráficas

$$G = (V, E)$$

- A menos que se especifique lo contrario, se supone que los conjuntos E y V son finitos y que V no es vacío

3.1.1 Conceptos y definiciones de gráficas

- Algunos otros tipos de gráficas



3.1.1 Conceptos y definiciones de gráficas

Tarea: Describir en qué consisten los siguientes tipos de grafos (Dibujar un ejemplo):

- Grafo Completo
- Grafo Bipartito
- Grafo Planar(o Plano)
- Grafo regular

3.1.2 Representación de gráficas en la computadora

- Las tres formas más comunes de representar gráficas son:

  - Matriz de Adyacencia
  - Matriz de Incidencia.
  - Lista de adyacencia.

3.1.2 Representación de graficas en la computadora

- Matriz de adyacencia

Se trata de una matriz cuadrada de orden *n* por *n* donde a_{ij} es el número de aristas que unen los vértices v_i y v_j.

La matriz de adyacencia de una gráfica no dirigida es simétrica.

3.1.2 Representación de graficas en la computadora

- Matriz de adyacencia

Si existe un vértice aislado en la grafica, la respectiva fila de la columna está compuesta por ceros.

Si se trata de un grafo simple, entonces la matriz de adyacencia contiene solo ceros y unos y la diagonal principal está compuesta por ceros.

3.1.2 Representación de graficas en la computadora

- Matriz de adyacencia

La matriz de adyacencia de una gráfica dirigida es una matriz binaria cuyo numero de unos por fila representan el grado de salida del vértice y el número de unos que aparecen en una columna es el grado de entrada del vértice

3.1.2 Representación de graficas en la computadora

- Matriz de incidencia

Dada una grafica con *n* vértices y *m* aristas, la matriz de incidencia es una matriz de orden *n* x *m* donde a_{ij} es 1 si v_i es incidente con e_j y a_{ij} es 0 en caso contrario.

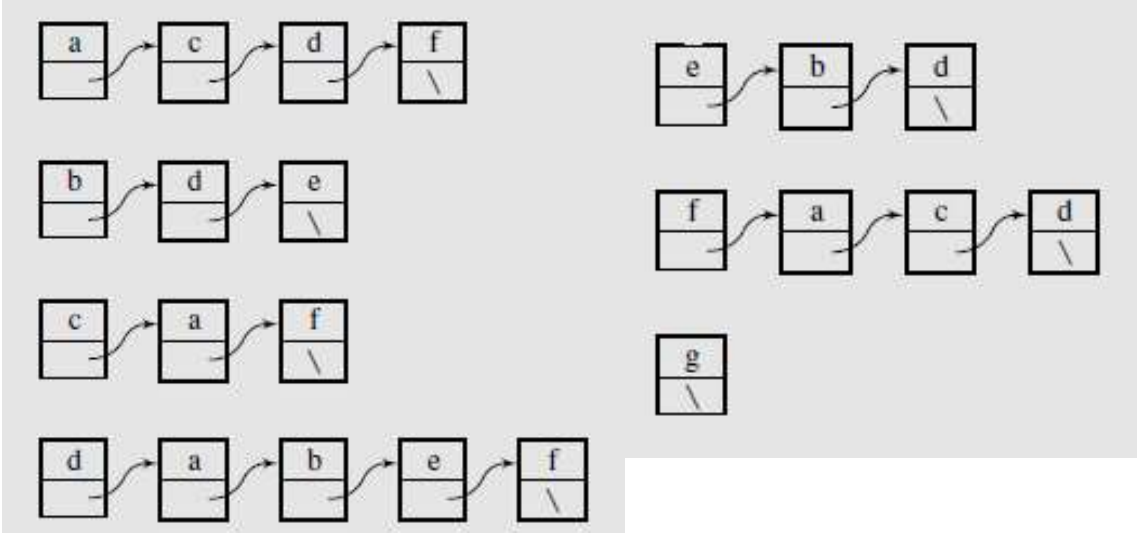
3.1.2 Representación de graficas en la computadora

- Lista de adyacencia

Se trata de una lista ligada donde se almacena la referencia a una lista de los vértices adyacentes de *i*

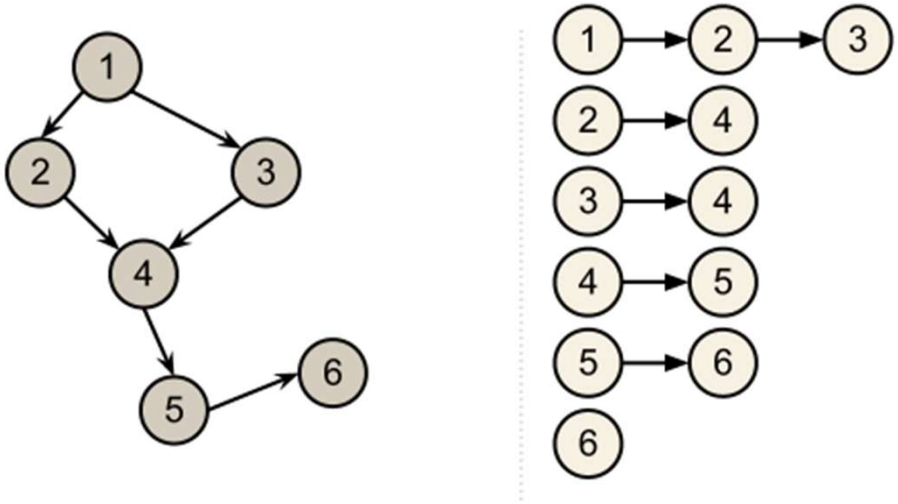
3.1.2 Representación de graficas en la computadora

- Lista de adyacencia



3.1.2 Representación de graficas en la computadora

- Lista de adyacencia



Recorridos sobre grafos

- El propósito de recorrer un grafo es visitar todos los nodos que le pertenecen
- Existen diversas maneras para recorrer los vértices de un Grafo.
- Dichas maneras pueden variar de acuerdo a las características del mismo (Dirigido, no- dirigido, ponderado, etc).
- Normalmente los recorridos no son únicos debido a que los grafos no tienen un orden específico.

3.2 Búsqueda por Expansión

- **BFS (Breadth First Search)**
- BFS es un algoritmo de desplazamiento en el que debe comenzar a visitar a partir de un nodo seleccionado (nodo de origen o de inicio) y recorrer el grafo en forma de capas o de niveles.

3.2 Búsqueda por Expansión

- Se deben explorar los nodos vecinos (nodos que están conectados directamente al nodo de origen).
- A continuación, debe avanzar hacia los nodos vecinos del siguiente nivel.
- Como sugiere el nombre BFS, debe recorrer el gráfico a lo ancho.
- Se busca analizar al grafo por capas primero desplazandose horizontalmente (capa actual) y posteriormente avanzar a la siguiente capa

3.2 Búsqueda por Expansión

- Como sugiere el nombre BFS, debe recorrer el gráfico a lo ancho de la siguiente manera:
- Primero desplazarse horizontalmente y visitar todos los nodos de la capa actual y posteriormente avanzar a la siguiente capa

3.2 Búsqueda por Expansión

```
list BFS (Graph G, Node s)
      q = new Queue();
      list = nodes;
       q.enqueue(s)
      mark s as visited.
      while ( q is not empty)
            v  =  q.dequeue()
            for all neighbors w of v in Graph G
               if w is not visited
                    q.enqueue( w )
                    mark w as visited
                    nodes.add( w )
                else
                    ignore w
```

3.3 Búsqueda en Profundidad

- **DFS (Depth First Search)**
- DFS es un algoritmo de desplazamiento en el que debe comenzar a visitar a partir de un nodo seleccionado (nodo de origen o de inicio) y recorrer los vértices internamente.
- Una vez que hace el recorrido inicial, debe regresar en el camino y hacer un segundo recorrido con los nodos que no han sido visitados

3.3 Búsqueda en Profundidad

```
DFS-recursive(G, s):
   mark s as visited
   for all neighbours w of s in Graph G:
       if w is not visited:
           DFS-recursive(G, w)
```

Otros conceptos de grafos

- Árbol de Expansión Mínima
  - Algoritmo de Kruskal
  - Algoritmo de Primm
- Algoritmos de Ruta más corta
  - Algoritmo de Dijkstra
  - Algoritmo Bellman-Ford

Otros conceptos de grafos

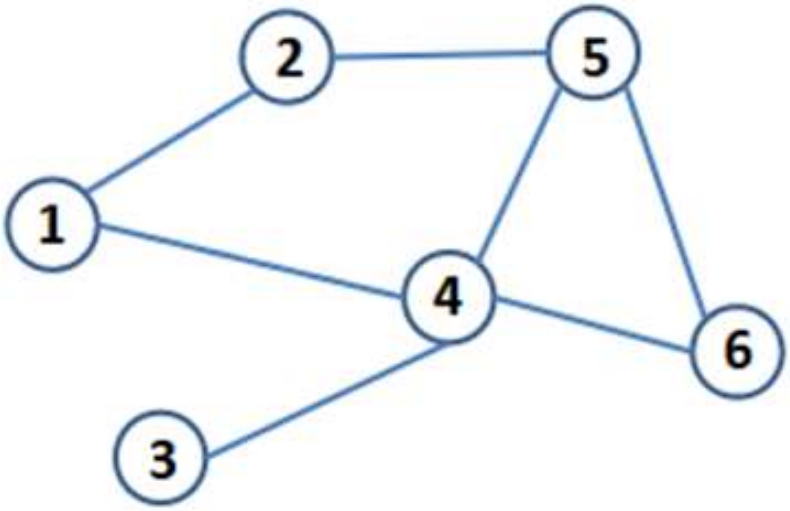
- Árbol de Expansión

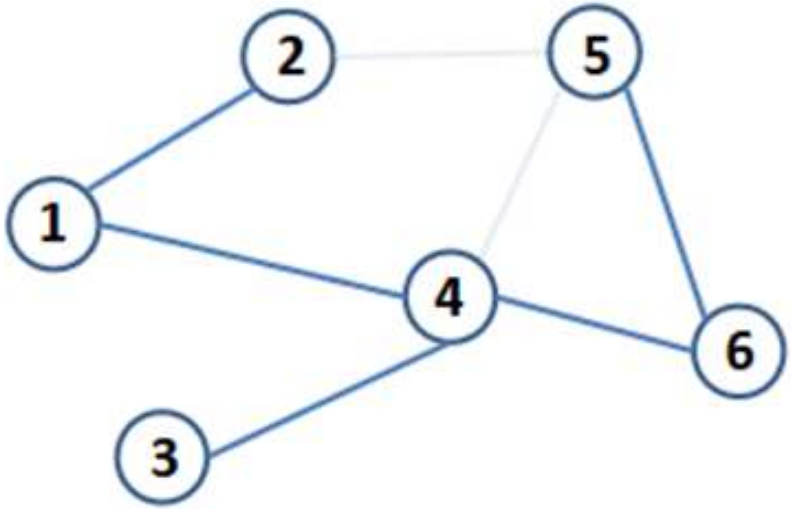
Dado un grafo no dirigido G. Un árbol de expansión es un árbol compuesto por todos los vértices y algunas (o todas) las aristas de G.

Al ser creado un árbol no existirán ciclos, además debe existir una ruta entre cada par de vértices.

Otros conceptos de grafos

- Ejemplo





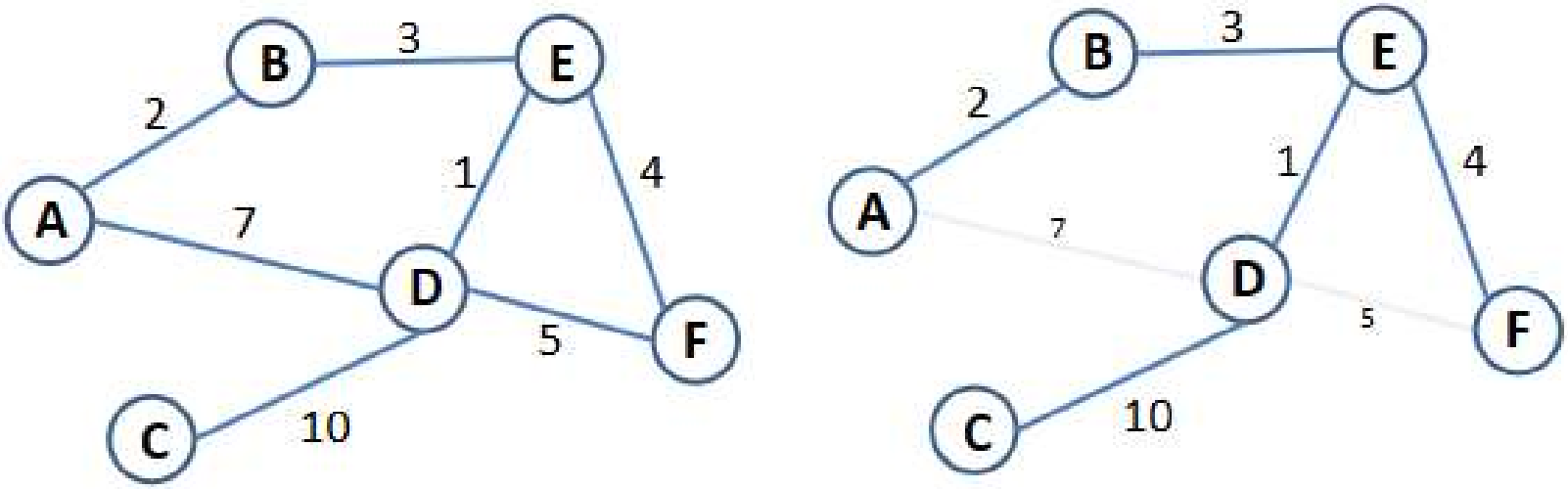
Otros conceptos de grafos

- Árbol de expansión Mínimo

En un grafo ponderado *G, el "costo" o "peso"* de un sub-grafo es la suma de los pesos de las aristas incluidas en ese sub-grafo.

Un *árbol de expansión mínimo* para un grafo ponderado es un árbol abarcante cuyo costo es el mínimo.

Otros conceptos de grafos



Otros conceptos de grados

- Algoritmo de Dijkstra
- El propósito de este algoritmo es resolver el problema del camino más corto, dado un nodo origen, hacia el resto de los vértices en un grafo ponderado.
- Creado por E. Dijkstra, en 1959

Otros conceptos de grafos

- La idea consiste en ir explorando todos los caminos más cortos que parten del vértice origen hacia todos los demás vértices.
- El procedimiento finaliza cuando se obtiene el camino más corto desde el vértice origen hasta el resto de los vértices que componen el grafo (cuando se han cubierto todos los nodos)