Grafos
Equipo 5

CENTRO DE ENSEÑANZA
TÉCNICA INDUSTRIAL

Plantel COLOMOS, Desarrollo de software, Estructura de datos

Integrantes:
22300891 Emmanuel Buenrostro Briseño, Emmanuel
22300951 Franco Abrajan Olmos, Franco
22300895 Emilio Mateo Rico García, Emilio

- **Investigación**

Algunas definiciones:
Definición de *aristas y vértices*:

- *Vértices* (o nodos) representan puntos individuales en el grafo. En programación, los vértices pueden ser ciudades, personas, o cualquier entidad que se desee conectar.
- *Aristas* son las conexiones entre pares de vértices, representando relaciones o caminos. Un grafo puede tener múltiples aristas entre dos vértices (en un grafo multigrafo).

Definición de *recorridos*: Un recorrido en un grafo es una secuencia de vértices donde cada vértice está conectado al siguiente por una arista. Los recorridos pueden ser abiertos (sin regresar al punto de inicio) o cerrados (donde el último vértice coincide con el primero).
Definición de *ciclos*: Un ciclo es un recorrido cerrado en el cual ningún vértice es repetido, excepto el de inicio y final, que coinciden. En programación, los ciclos permiten modelar situaciones que regresan a un punto inicial después de pasar por ciertos estados o nodos.
Recorridos y ciclos eulerianos:

- Un *recorrido euleriano* es un camino que pasa por todas las aristas del grafo exactamente una vez, sin importar si pasa varias veces por algún vértice.
- Un *ciclo euleriano* es un recorrido euleriano cerrado, es decir, que regresa al vértice inicial. Un grafo tiene un ciclo euleriano si todos sus vértices tienen un número par de aristas conectadas.

Recorridos y ciclos hamiltonianos:

- Un *recorrido hamiltoniano* es un camino que pasa por todos los vértices del grafo una vez, sin repetir ninguno.
- Un *ciclo hamiltoniano* es un recorrido hamiltoniano cerrado, donde el último vértice se conecta de vuelta al primero. Este concepto es más complejo de resolver que el euleriano y tiene aplicaciones en problemas de optimización, como el "problema del vendedor viajero".

En un grafo el *grado* de un vértice v es la cantidad de aristas incidentes en v, o también se puede ver como la cantidad de vértices adyacentes que tiene v.

Se cumple que la suma de los grados de todos los vértices es dos veces la cantidad de aristas (porque cada arista influyen dos veces en total).

Entonces existe un ciclo euleriano si y sólo si todos los vértices del grafo tienen grado par. Esto porque cada arista que "entra" al nodo tiene que "salir" por alguna otra arista, entonces tiene que haber una cantidad par de aristas en cada vértice lo que implica que el grado sea par.

En algunos grafos se usan *Aristas con peso* que es a cada arista asociar un valor numérico que puede representar distinta información, por ejemplo alguna distancia, un costo, etc.

Entonces esto nos agrega nuevos conceptos en los caminos, que son los caminos (o ciclos) con valor mínimo o máximo, y su valor se obtiene de sumar los valores de las aristas por las que pasa, aunque lo más común es que para estos se suele usar que no puedes pasar dos veces por la misma arista o dos veces por el mismo vértice.

Una *componente conexa* es un conjunto de nodos tales que mediante las aristas existentes puedes llegar de cualquier nodo a otro. Un grafo se dice conexo si esta propiedad se cumple para todos los nodos del grafo.

Recorrido de grafos

DFS (Búsqueda en profundidad):
En este algoritmo, se empieza por un nodo inicial y se va a recorrer el grafo en su profundidad, para esto lo que se hace es cada vez que el nodo en el que estamos se escoge alguno de sus nodos adyacentes que nunca hayamos visitado y vamos a ese nodo y en caso de que no haya ningún nodo adyacente que no hayamos visitado se regresa al nodo que se visitó antes de ese, este proceso se repite hasta que regresemos al nodo inicial y este no tenga más nodos que nunca hayamos visitado (en este punto ya no va a haber un nodo “anterior” al cual regresarnos).

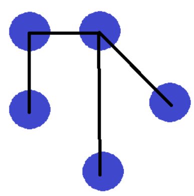
Este algoritmo tiene una complejidad de O(V+E) si el grafo está en una lista de adyacencia o en O(V^2) si está en una matriz de adyacencia.

BFS (Búsqueda en Anchura):
En este algoritmo, se empieza por un nodo inicial, después vamos a visitar los nodos adyacentes al nodo inicial, están a distancia 1 del nodo inicial, después los adyacentes a esos nodos, que están a distancia 2 del nodo inicial, y así consecutivamente, solo que cuidando que no visitamos dos veces un nodo.

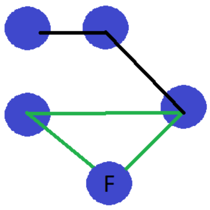
Estas dos son formas de obtener que nodos están en la misma componente conexa que el nodo inicial, ya que esos nodos son los que se visitan.

Tipos de grafos:

- Árboles
Los árboles son grafos que son conexos y que además no tienen ningún ciclo, a que nos referimos con estos dos términos, comenzando por que sean “conexos”, un grafo es conexo cuando todas sus aristas están conformadas por un solo camino. Otra manera de entenderlo sería que un grafo es conexo si puedes dibujar todas sus conexiones sin separar el lápiz de la hoja. Por otro lado, se dice que un grafo tiene “ciclos” cuando tiene por lo menos un nodo con un camino que vuelve a él mismo sin repetir aristas.

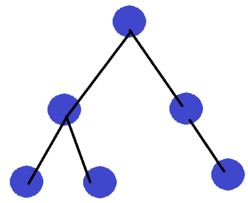


Un ejemplo de conexo es el siguiente, ya que todas sus aristas se conectan por un solo camino.

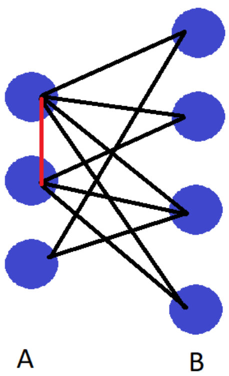


Y por otro lado un ejemplo de un ciclo dentro de un grafo sería el siguiente, ya que podemos notar cómo, podemos hacer un recorrido donde empezamos en el nodo F y terminamos en el mismo nodo F, sin necesidad de pasar por la misma arista más de una vez. Las aristas que conforman este ciclo están marcadas con verde.

Cuando estas dos condiciones se cumplen obtenemos como resultado un grafo árbol, que es muy comúnmente usado para estructuras de datos

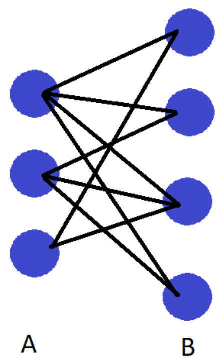


- Bipartitos
Los grafos bipartitos son grafos que surgen cuando tenemos 2 grupos de nodos que no están unidos entre sí pero si están unidos con otro grupo.



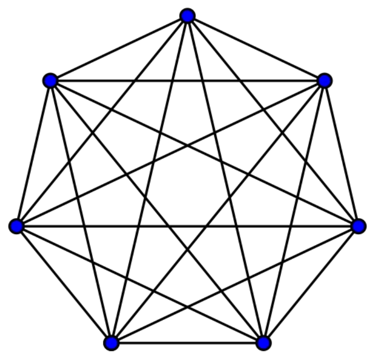
En el ejemplo de la imagen derecha tenemos dos grupos de nodos llamados A y B respectivamente, podemos notar como todos los nodos del grupo A están unidos mediante una arista a algún nodo del grupo B y viceversa, sin embargo, no hay nodos del grupo A que estén conectados a otro nodo del grupo A ni hay nodos del grupo B que estén conectados a otro nodo del grupo B

Si tuviésemos un caso donde algún nodo de A estuviera conectado a algún otro nodo de A, o algún nodo de B estuviera conectado a algún otro nodo de B, entonces el grafo dejaría de ser bipartito (imagen izquierda)



- Completos
Los grafos completos son grafos en los cuales cada nodo tiene una arista que se une a cada uno de los otros nodos, es decir, todos los nodos están conectados entre sí.
Estos grafos se representan con la letra “K”, que viene de la palabra komplett en alemán, que significa completo

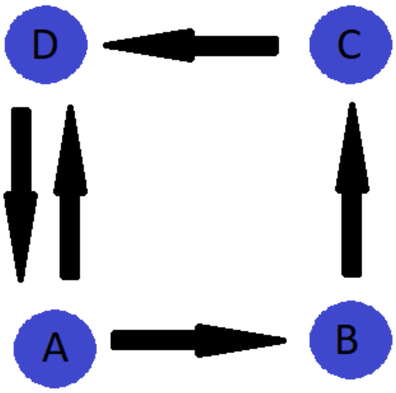
El grafo que se puede ver a la derecha por ejemplo sería un grafo “K7”, debido a que es un grafo completo que consta de 7 nodos que están unidos todos entre sí como podemos observar



- Dirigidos
Los grafos dirigidos son un subconjunto de grafos que utilizan flechas en lugar de aristas, a estas flechas se les llama “arcos”. La diferencia entre estos grafos y los “normales” por así decirlo que usan aristas es que, mientras que en los grafos con aristas podemos movernos

de un nodo a otro con libertad, con los arcos es distinto, ya que estos nos marcan cual es la dirección en la que sí nos podemos mover.

En el ejemplo que se muestra, tenemos un grafo dirigido conformado por los nodos A, B, C y D, como podemos ver, todos los nodos tienen un arco hacia su siguiente nodo, si nosotros quisiéramos movernos de A a B, no habría problema, debido a que el arco apunta de A a B, sin embargo si quisiéramos hacerlo al revés, de B a A, tendríamos que irnos primero a C, luego a D y luego ahora si a A, debido a que no hay un arco que nos apunte de B a A directamente. La cosa cambia con los nodos A y D, donde podemos ver si hay un arco de A a D y también hay un arco de D a A, por lo que podemos movernos entre A y D sin problemas como si fuese un grafo normal



Algo importante a aclarar es que en los grafos dirigidos, los nodos tienen 2 tipos de grados diferentes, uno para los arcos que salen de ellos y otro para los arcos que llegan de ellos.

$$\sum_{v \in G} graent(v) = \sum_{v \in G} grasal(v) = m$$

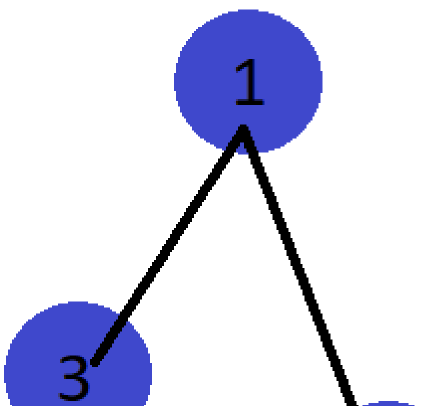
Esta es una propiedad de los grafos dirigidos, donde m es el total de arcos del grafo, graent(v) es el grado para los arcos que llegan al nodo "v" y grasal(v) es el grado para los arcos que salen del nodo "v"

Matriz de adyacencia y Lista de adyacencia:

Existen varias formas de armar nuestros propios grafos, dos de estas son las listas de adyacencia y la matriz de adyacencia.

- Matriz de adyacencia
Se crea una matriz cuadrada, donde tanto las filas como las columnas representan los vértices del grafo a crear. Dentro de esta matriz, se pone 0 en las casillas que no representan ninguna conexión y se pone el número de conexiones en las que sí representan una conexión.

Debajo hay un ejemplo con un grafo y su respectiva matriz



$$\begin{array}{cccc} & 1 & 2 & 3 \\ 1 & 0 & 1 & 1 \end{array}$$

2 1 0 0
3 1 0 0

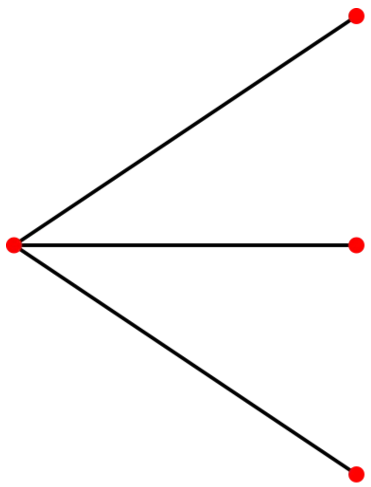
- Lista de adyacencia
Se crea una lista de listas, esta lista tiene dentro una lista pequeña que contiene las todas las conexiones que tiene cada nodo.

Utilizando el mismo grafo del ejemplo anterior podemos crear una lista que tenga en su primer elemento la lista [2, 3], que es la lista de todas las conexiones que tiene el nodo uno, luego el segundo elemento sería la lista [1], que es la lista de todas las conexiones que tiene el nodo dos, el último elemento de la lista sería la lista [1], que es la lista de todas las conexiones que tiene el nodo tres. Como resultado tendríamos la siguiente lista de adyacencia: [[2, 3], [1], [1]]

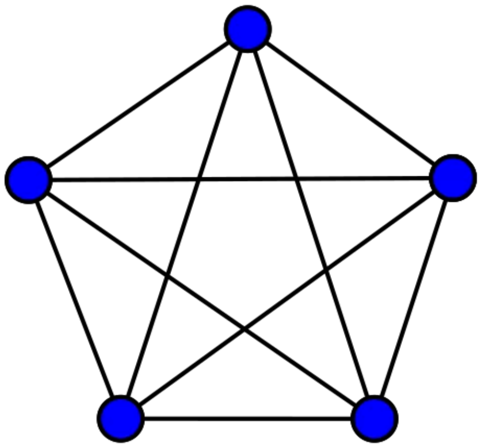
- Exposición (Esto tmb lo hago yo (Emilio), siempre no hagan paro aaaaaljkgnkbljtkl4rfklñ):

- Ejemplos:

1) Grafo bipartito



2) Grafo K5



Exposición:

Ciclos/Recorridos:

Vértice / Nodo: representan puntos individuales en el grafo.

Aristas: son las conexiones entre pares de vértices, representando relaciones o caminos.

Recorrido: Una secuencia de vértices donde cada vértice está conectado al siguiente por una arista.

Recorrido abierto: No se regresa al punto de inicio

Recorrido cerrado: Se regresa al punto de inicio

Ciclo: Recorrido cerrado en el cual ningún vértice es repetido, excepto el de inicio y final, que coinciden.

Recorrido euleriano: Camino que pasa por todas las aristas del grafo exactamente una vez, sin importar si pasa varias veces por algún vértice.

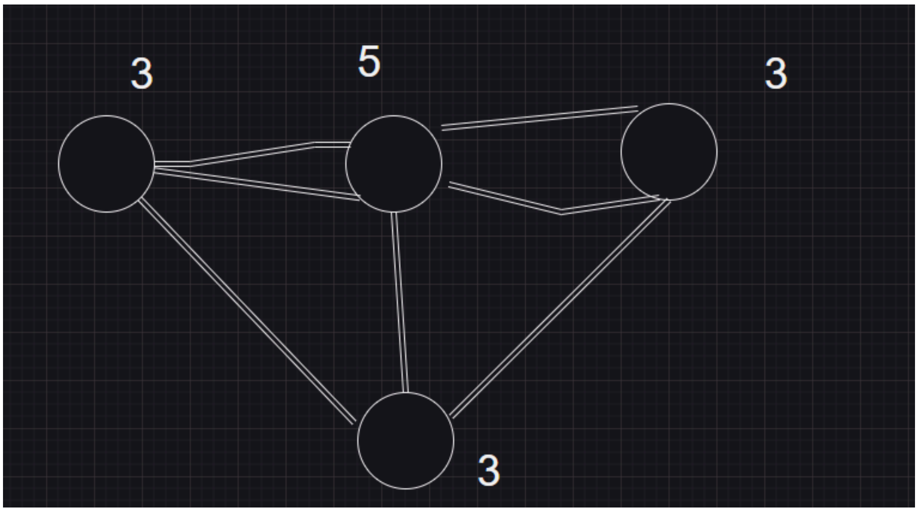
Ciclo euleriano: Recorrido euleriano cerrado.
Un grafo tiene un ciclo euleriano si todos sus vértices tienen un número par de aristas conectadas.

Recorrido hamiltoniano: Un recorrido hamiltoniano es un camino que pasa por todos los vértices del grafo una vez, sin repetir ninguno.

Ciclo hamiltoniano:Recorrido hamiltoniano cerrado.
Este concepto es más complejo de resolver que el euleriano y tiene aplicaciones en problemas de optimización

Problema de los puentes

Consiste en determinar si es posible recorrer ciertos puentes de la ciudad de Königsberg sin cruzar el mismo puente más de una vez.
Podemos convertir el problema en un grafo.



Para pasar por todas las aristas y regresar al mismo vértice, cada vez que entramos a un vértice tenemos que tener una arista para “salir” de ese vértice. Entonces ocupamos que cada vértice tenga una cantidad par de aristas.
El número de aristas de cada vértice se llama grado, queremos que el grado de cada vértice sea par

Pero en el problema podemos notar que hay vértices con grado impar. Por eso no es posible

Distancia mas corta

El problema consiste en encontrar la distancia más corta entre dos nodos.

Podemos agregar valores a las aristas. Estas son llamadas aristas con peso
Estas suelen hacer referencia a algo, por ejemplo a una distancia, a un costo, etc.

Una componente conexa es un conjunto de nodos que se puede llegar de cualquier nodo en ese conjunto a cualquier otro de nodo de ese conjunto por medio de las aristas.
Un grafo es conexo si es una sola componente conexa

<u>Recorridos (DFS)</u>

Se recorre el árbol en profundidad.
Tienes un nodo inicial.
Vas recorriendo escogiendo un nodo adyacente que no haya sido visitado anteriormente.
Si no hay te regresas al nodo anterior y vuelves a escoger un nodo no visitado anteriormente.

<u>Recorridos (BFS)</u>

Se recorre el árbol en anchura.
Tienes un nodo inicial.

Visitas todos los de adyacentes al inicial.

Luego todos los adyacentes a esos que no hayas visitado.
Y así sucesivamente.

Al hacer cualquiera de estos dos recorridos obtenemos los nodos que están en la misma componente conexa que el nodo inicial que son los nodos que en algún punto visitamos

<u>Solución al problema de distancia mínima:</u>

Al hacer una BFS iniciando desde el nodo que queremos saber la distancia mínima, al ver en qué nivel visitamos el nodo del que queremos saber la distancia esa será su distancia mínima.

En caso de que queramos resolverlo en un grafo cuyas aristas tienen pesos la solución es un poco más compleja y utiliza algoritmos como Dijkstra

Tipos de Grafos:

Árbol: Un árbol es un grafo conexo que no tiene ciclos en ninguna parte.
Suelen ser utilizados para estructuras de datos

¿A qué nos referimos con "conexo" y "ciclos"?
Se dice que un grafo es conexo cuando todas sus aristas están conectadas por un mismo camino, otra manera de verlo podría ser que, cuando un grafo es convexo, puedes dibujar todos sus enlaces sin separar el lápiz de la hoja

Por otro lado, se dice que un grafo tiene ciclos cuando existe por lo menos un nodo del que se puede comenzar un recorrido que termina en ese mismo nodo sin necesidad de pasar por la misma arista más de una vez

Bipartito: Un grafo bipartito consiste en dos grupos de nodos que no están conectados entre sí pero si con otro grupo

¿Cuando es y cuando no es?
Si tenemos dos grupos de nodos por ejemplo A y B, que están conectados el uno con el otro, pero ningún de A está con otro de A ni otro de B está con otro de B, entonces es bipartito

Por otro lado, si en alguno de los grupos, ya sea A, B o ambos, hay un nodo unido a otro nodo de ese mismo grupo, entonces ese grafo ya no es bipartito.

Completo: Un grafo completo es un grafo cuyos nodos están unidos todos entre sí

Si tenemos un grafo, donde cada nodo tiene una arista que conecta con cada uno de los otros nodos del grafo, entonces el grafo es completo.
Estos grafos se representan con una K que viene de la palabra komplett en alemán, que significa completo en alemán.

Dirigidos: Un grafo dirigido es un grafo que tiene arcos o “flechas” en lugar de aristas como conexiones entre los nodos

¿Cuál es la diferencia entre los arcos y las aristas?

En un grafo normal la unión se realiza con aristas, que se representa con simples líneas rectas. En estos grafos podemos movernos de un grafo para otro sin problemas, a diferencia de los grafos dirigidos.
En un grafo dirigido se utilizan arcos en lugar de aristas, que se representan con flechas, y estas marcan hacia donde puedes moverte y hacia donde no

Creación de grafos

Existen varias formas de crear la matriz que nosotros queramos, siendo dos de estas formas las matrices de adyacencia y las listas de adyacencia.

<u>Matriz de adyacencia</u>

Consiste en una matriz cuadrada que representa en sus filas y columnas a cada nodo del grafo y que representa todas las uniones que tiene un grafo

<u>Lista de adyacencia</u>

Consiste en una lista de listas, donde las listas dentro de la lista principal representan las uniones que tiene cada uno de los nodos del grafo

-Ejemplos:

Problema de los puentes: Este problema, conocido como el "Problema de los puentes de Königsberg", fue planteado por el matemático Leonhard Euler. Se trataba de determinar si era posible recorrer una serie de puentes de la ciudad de Königsberg sin cruzar el mismo puente más de una vez. Este problema es importante en la teoría de grafos porque llevó a la definición de conceptos como los grafos eulerianos, además de establecer las bases de la teoría de grafos moderna.

Utiliza que para que exista un ciclo euleriano todos los nodos del grafo tienen que tener grado par.

Pathfinding: Encontrar la distancia mas corta entre algunos nodos del grafo.

Para grafos sin pesos basta con una simple BFS. Pero para grafos con pesos Dijkstra es una manera de resolverlo (para aristas con peso positivo).

Esto suele usarse para encontrar el menor/mayor costo posible, o distancia entre dos posiciones, etc.

- Preguntas y respuestas:

Verdadero o falso:

1) Un grafo tiene que ser convexo y no debe tener ciclos para clasificar como árbol
R/ Verdadero

2) Un grafo “K9” sería un grafo bipartito de 9 nodos
R/ Falso

3) Todos los nodos de un grafo completo tienen el mismo grado
R/ Verdadero

4) Un recorrido euleriano pasa por todos los vértices exactamente una vez, sin importar si pasa varias veces por un vértice
R/ Falso

5) En un ciclo no se repiten vértices, con la excepción de al inicio y al final en caso de coincidir
R/ Verdadero

6) DFS es un algoritmo que permite encontrar la distancia mínima entre dos nodos en un grafo sin pesos?
R/Falso

7) Si los grados de todos los vértices de un grafo son pares, entonces el grafo tiene un ciclo euleriano?
R/Verdadero

- Bibliografía:

- BitBoss. (2022, April 23). APRENDE GRAFOS DESDE CERO: Grafos básicos, lista y matriz de adyacencia, definiciones y propiedades. [Video]. YouTube. https://www.youtube.com/watch?v=vnNFiNVy9KM
- Teoría de grafos. (2013). Universidad del Bio-Bio. http://repobib.ubiobio.cl/jspui/bitstream/123456789/1953/3/Alvarez_Nunez_Marcelino.pdf
- Villalobos, A. R. (n.d.). *Grafos - Software para el diseño y análisis de grafos - Alejandro Rodríguez Villalobos*. Creative Commons License (Cc) 2003-2010 Alejandro Rodríguez Villalobos. https://personales.upv.es/arodrigu/grafos/
-Halim, S., & Halim, F. (s. f.). Programación competitiva. OJBooks.

-Componentes conexas y bicoloración de grafos | Aprende Programación Competitiva. (s. f.). https://aprende.olimpiada-informatica.org/algoritmia-componentes-conexas-bicoloracion