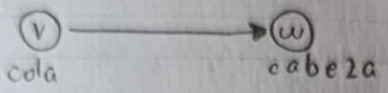
GRAFOS DIRIGIDOS

Un grafo dirigido consiste en un conjunto de vértices y arcos.
El Futuro se moldea
↳ nodos o puntos → arcos dirigidos o líneas dirigidas

arco: par ordenado de vértices.



w es adyacente a v

los vértices representan objetos y los arcos relaciones entre objetos.

Un grafo dirigido puede usarse para representar el flujo de control en un programa de computador.

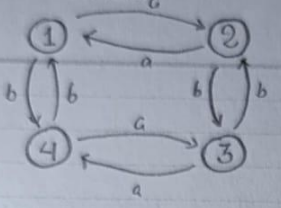
camino - secuencia de vértices
longitud de camino - número de arcos en el camino
camino→simple - todos sus vértices (excepto el primero y el último) son distintos.
camino→ciclo - camino simple de longitud mínima 1 y empieza y termina en el mismo vértice.

grafo dirigido etiquetado - c/arco, vértice o ambos pueden tener una etiqueta de cualquier tipo de dato.

Un vértice puede tener a la vez un nombre y una etiqueta.

6.2 Representaciones de grafos dirigidos

matriz de adyacencia



	1	2	3	4
1		a		b
2	a		b	
3		b		a
4	b		a	

Los espacios vacios representan la ausencia de un arco

La desventaja de la matriz es que requiere un espacio $\Omega(n^2)$ si el grafo dirigido tiene menos de n^2 arcos.

Iberla o examinarla lleva un tiempo $O(n^2)$

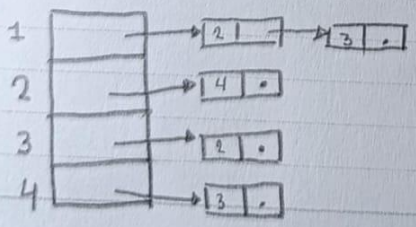
Lista de adyacencia

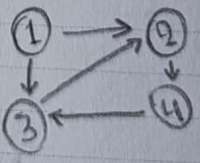
La lista de adyacencia para un vértice i es una lista en algún orden de todos los vértices adyacentes a i.

Requiere un espacio proporcional a la suma del número de vértices más el número de arcos (" " " $< n^2$)

puede llevar un tiempo $O(n)$ determinar si existe un arco del vértice i al vértice j.

ej. donde se usan listas enlazadas sencillas:





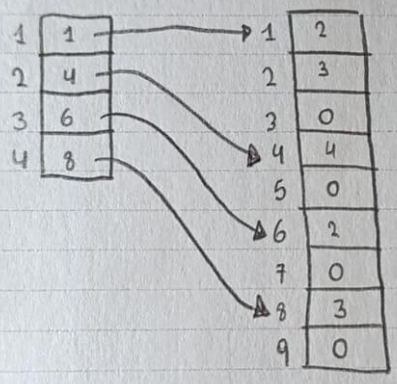
Si se espera que el grafo permanezca fijo, en las listas de adyacencia, sería preferible que CABEZA[i] fuera un cursor a un arreglo ADY, cuando ADY se encuentra un cero, es el fin de la lista de vértices adyacentes a i

TDA GRAFO DIRIGIDO

Operaciones:
* lectura de la etiqueta de un vértice o un arco, la inserción o supresión de vértices y arcos, y el recorrido de arcos desde la cola hasta la cabeza.

for cada vértice w adyacente a v do
{ acción con w }

Segunda representación con lista de adyacencia



índice - posición en la lista de adyacencia de v.
Si se usa una matriz de adyacencia, un índice es un entero que representa un vértice adyacente

3 Op:
1) Primero (v) - devuelve el índice del primer vértice adyacente a v. Se devuelve nulo (Λ) si no hay.
2) Siguiente (v, i) - devuelve i+1 de los vértices adyacentes a v. Se devuelve Λ si i es el último adyacente de v.
3) Vertice (v, i) - devuelve el vértice cuyo índice i está entre los vértices adyacentes a v.

Ver codigo pag 6 PDF

6.3 PROBLEMA DE LOS CAMINOS MÁS CORTOS CON UN SOLO ORIGEN

Un vertice se especifica como origen. El problema es determinar el costo del camino más corto del origen a todos los demás vertices de V.
longitud de un camino es la suma de los costos de los arcos del camino.

algoritmo de Dijkstra - opera a partir de un conjuntos de vértices cuya distancia más corta desde el origen ya es conocida.
especial - camino más corto entre el origen y v.

Funciona pq lo que aparece como lo mejor se convierte en lo mejor de todo

El futuro se moldea

Comparación entre Flogd y Dijkstra

Si el numero de aristas es menor que n^2, la version de Dijkstra con lista de adyacencia toma un tiempo O(a log n)

Tiempo de ejecución del algoritmo de Dijkstra: ~~O(n²)~~
$O(a \log n)$ - lista de adyacencia + cola de prioridad (heap)
a = actualizaciones
↳ Matriz de adyacencia

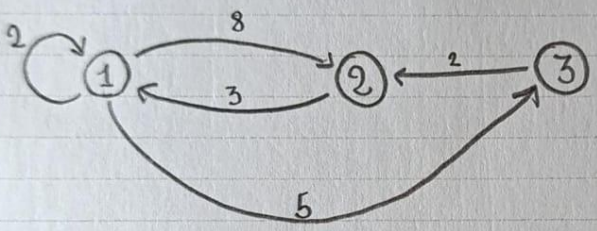
El Futuro se moldea

6.4 PROBLEMA DE LOS CAMINOS CORTOS ENTRE TODOS LOS PARES

Algoritmo de Floyd.

$$A_k[i,j] \min \begin{cases} A_{k-1}[i,j] \\ A_{k-1}[i,k] + A_{k-1}[k,j] \end{cases}$$

GRAFO DIRIGIDO PONDERADO



	1	2	3
1	0	8	5
2	3	0	∞
3	∞	2	0

$A_0[i,j]$

	1	2	3
1	0	8	5
2	3	0	8
3	∞	2	0

$A_1[i,j]$

	1	2	3
1	0	8	5
2	3	0	8
3	5	2	0

$A_2[i,j]$

	1	2	3
1	0	7	5
2	3	0	8
3	5	2	0

$A_3[i,j]$

Cerradura transitiva
algoritmo de Warshall.

Obj: determinar si existe un camino (directo o indirecto) entre todo par de vértices de un grafo dirigido

El Futuro se moldea

A[i][j] = True → arco directo $i \to j$

A[i][j] = False → Si no existe

$A[i][j] = A[i][j] \vee (A[i][k] \wedge A[k][j])$

$O(n^3)$

Sirve para grafos no ponderados

excentricidad de un vértice v: Máx distancia mínima desde cualquier otro vértice w hasta v.

Fórmula: $ex(v) = \max_{x \in V} \{\text{longitud del camino más corto de } w \text{ a } v\}$

Centro del grafo: vértice con la menor excentricidad (el más "cercano" al vértice más lejano)

6.5. Recorridos en grafos dirigidos

GERDAU
El futuro se moldea

Búsqueda en Profundidad (bpf)

Es un algoritmo recursivo que recorre sistemáticamente un grafo dirigido, explorando todos los vértices alcanzable desde un nodo inicial.

1) Inicialitación: Todos los vértices se marcan como no visitados.
2) Se recorren todas las adyacentes del actual si no fueron visitados
3) Si quedaron vértices sin visitar se repite el proceso desde otro vértice no explorado

O(a) (a=número de arcos)

Bosque abarcador en profundidad

bosque abarcador-

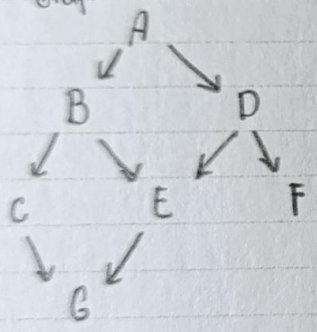
- compuesto por árboles bpf
- Es un bosque (conjunto de árboles) pq el grafo puede no ser conexo

El Futuro se moldea

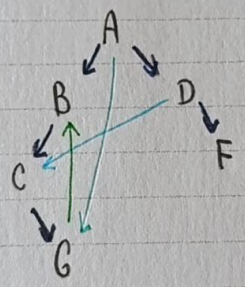
Tipos de Arcos

Tipo	Dirección	Ejemplo	Def
árbol	$u \to v$	$A \to B$	cuando v se visita x 1ra vez desde u
retroceso	$u \to v$	$C \to A$	v es un ancestro de u en el bosque
avance	$u \to v$	$A \to D$	v es un descendiente de u (pero no es arco de árbol)
cruzado	$u \to v$	$D \to C$	u y v no tienen relación ancestro-descendiente.

Grafo



Bosque Abarcador



Arco de árbol
retroceso
avance
cruzado