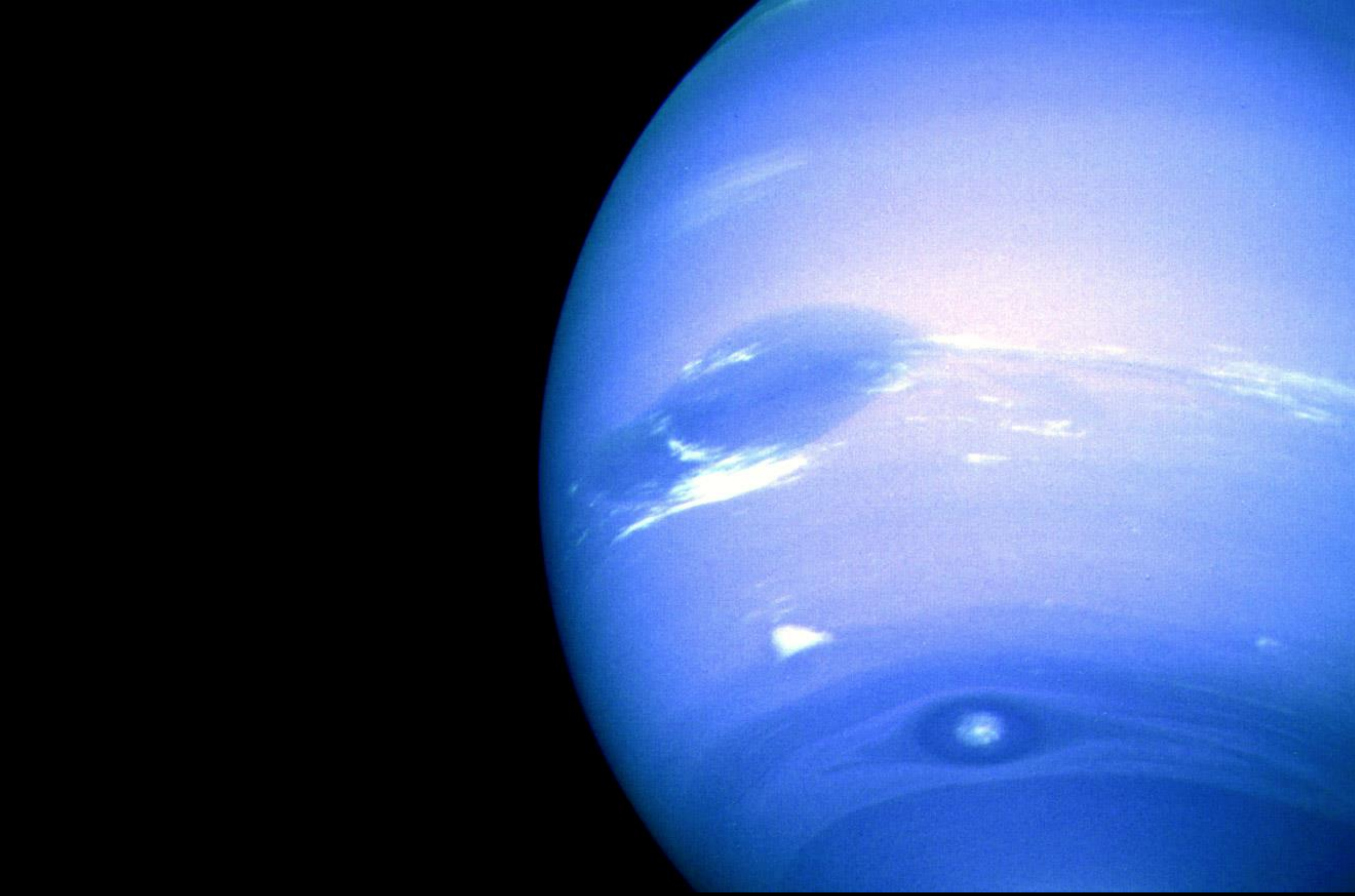
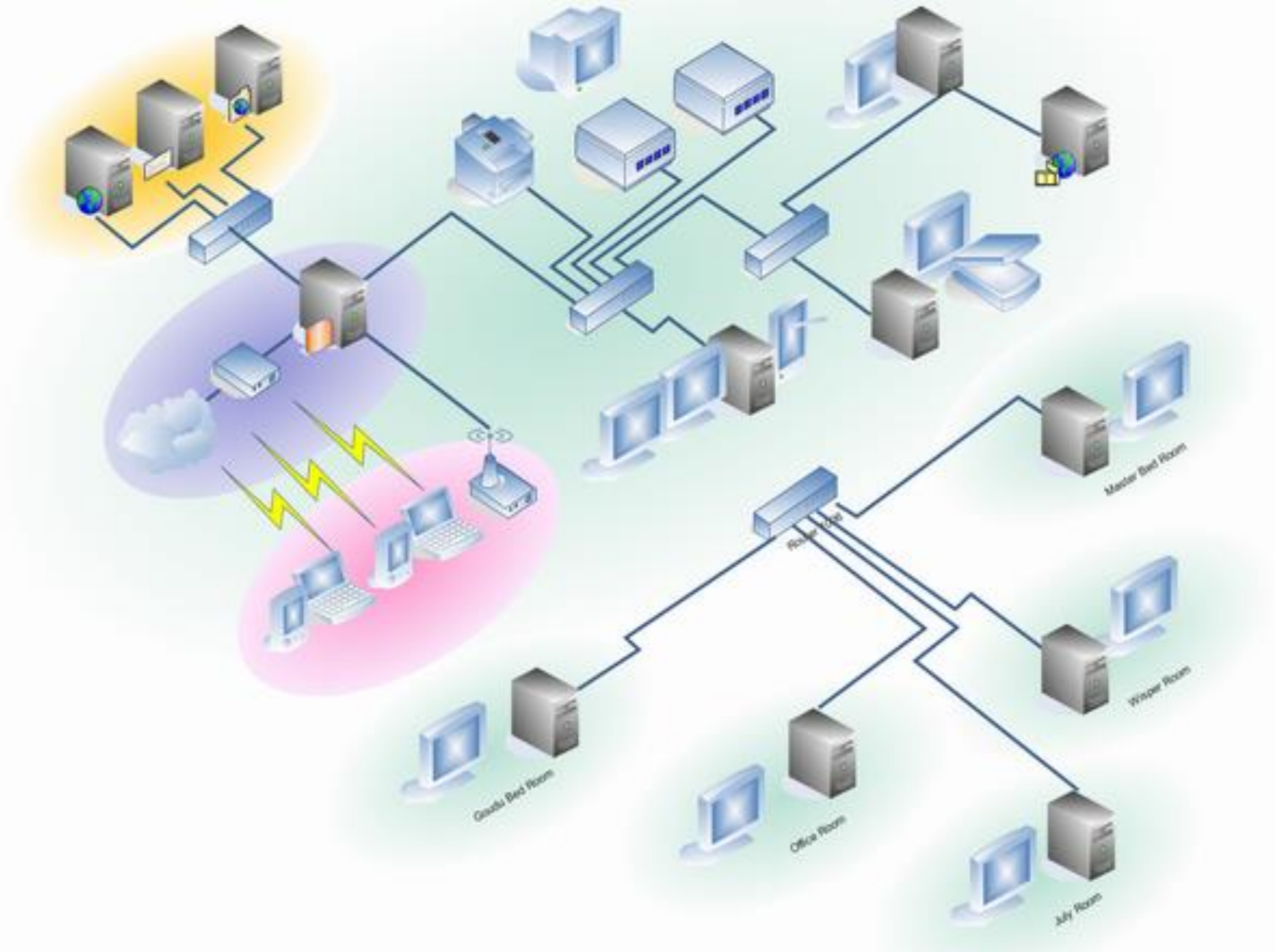
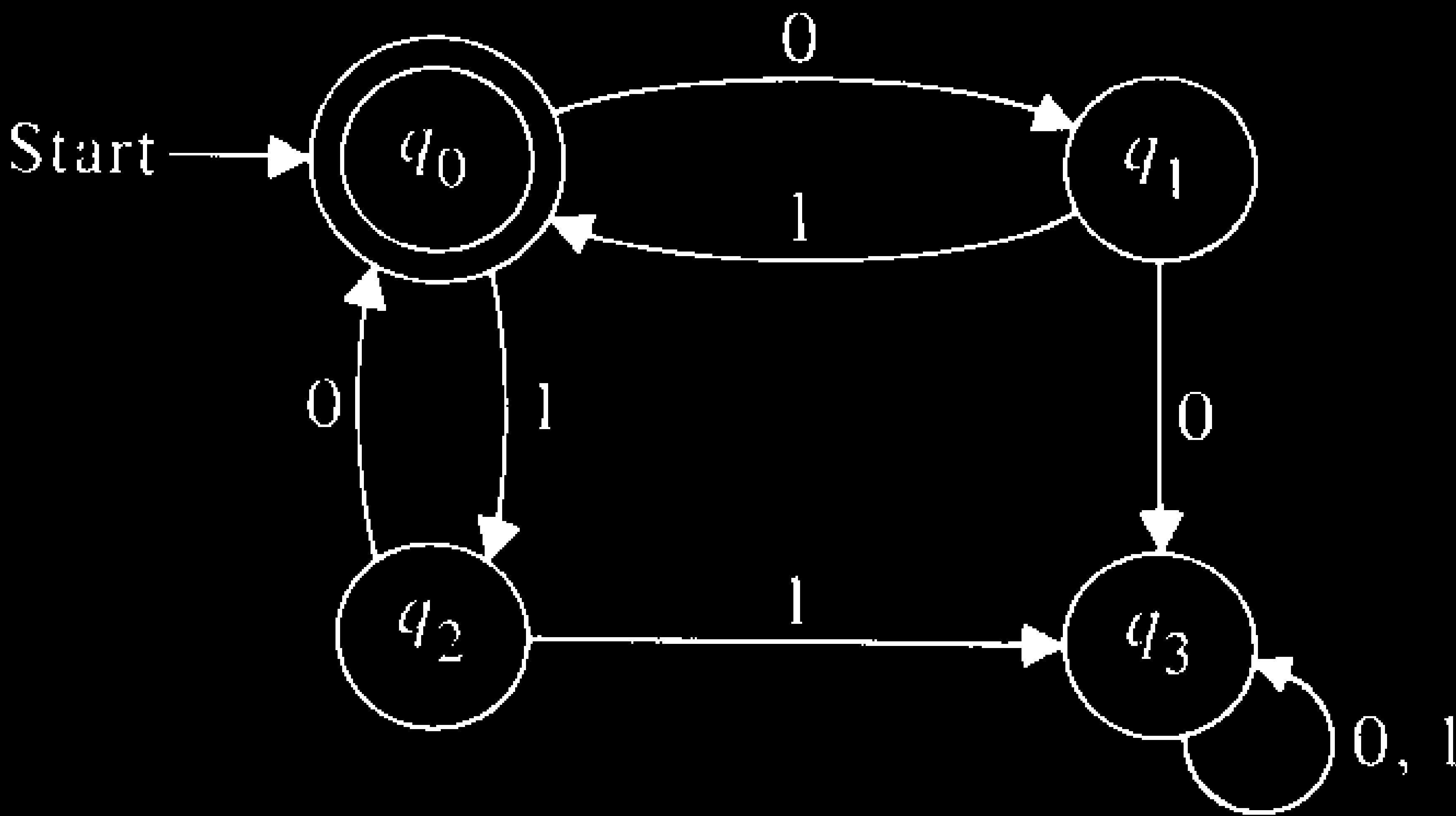


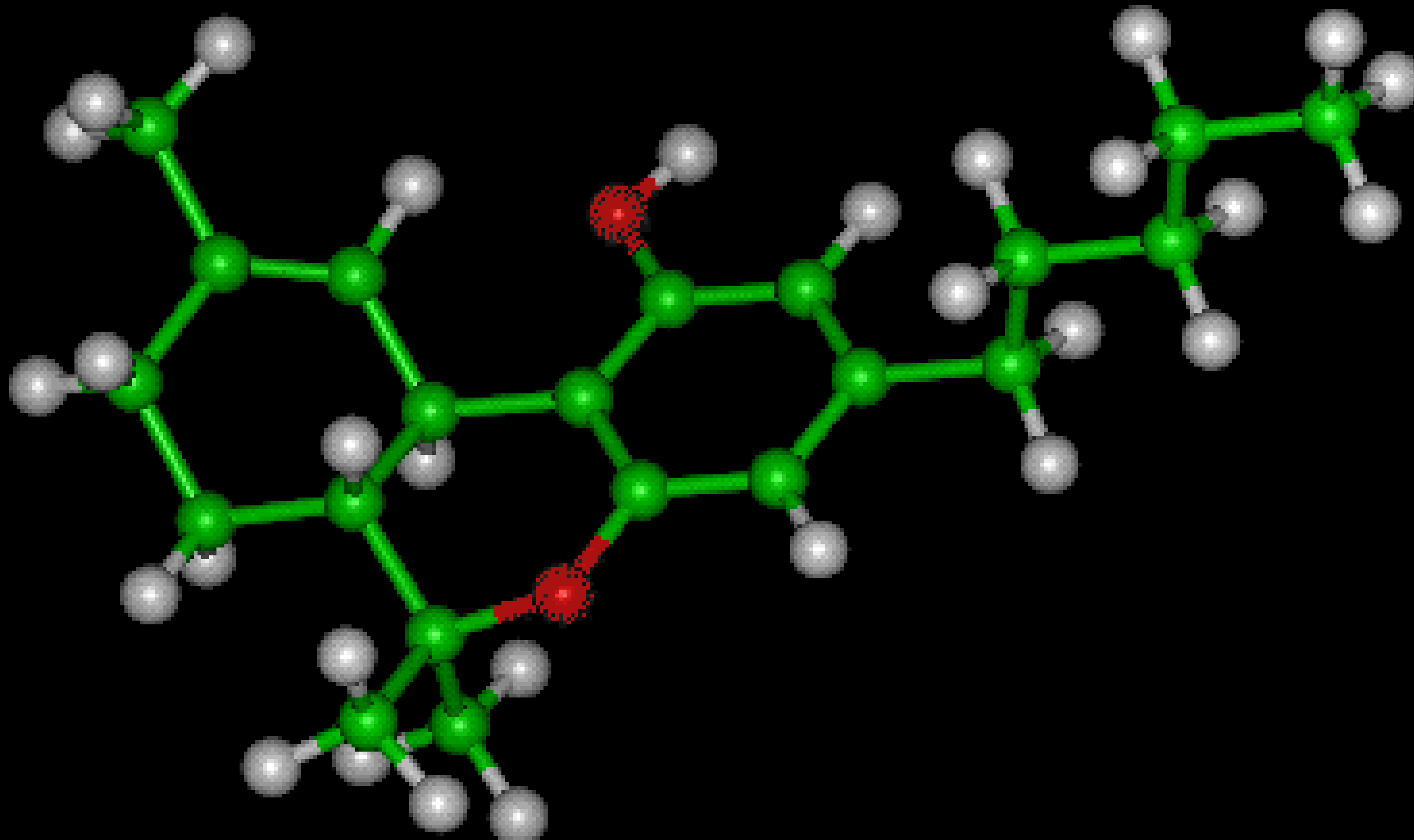
Grafos

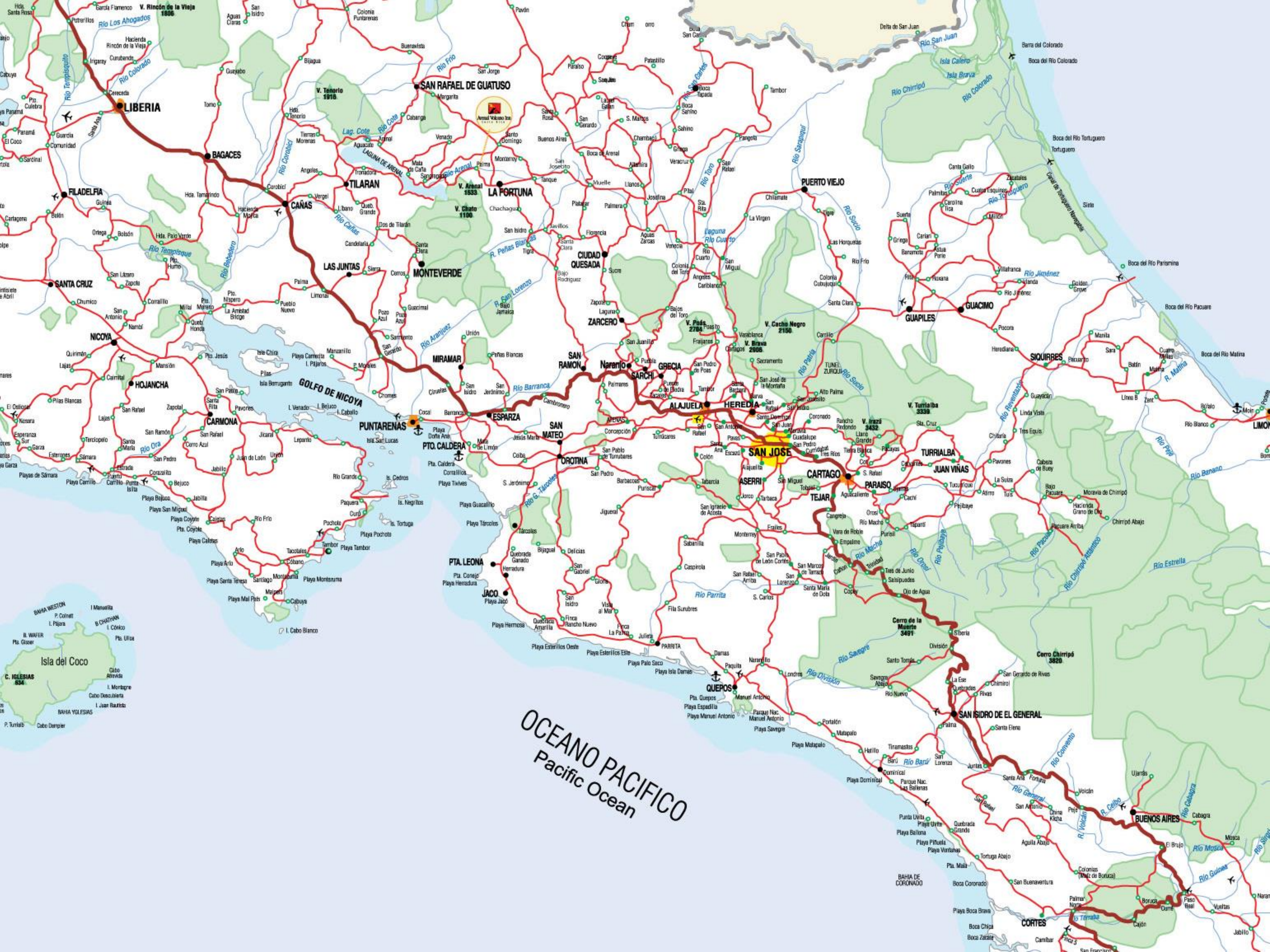
Mauricio Avilés


Master Bed Room
Wisper Room
Goudu Bed Room
Office Room
July Room



Start
q_0
q_1
q_2
q_3
0
1
0
1
0
1
0, 1





LIBERIA
BAGACES
FILADELFIA
CAÑAS
TILARAN
SAN RAFAEL DE GUATUSO
LA FORTUNA
V. Arenal 1633
V. Chato 1100
V. Tenorio 1916
V. Rincón de la Vieja 1806
MONTEVERDE
LAS JUNTAS
SANTA CRUZ
NICOYA
HOJANCHA
CARMONA
GOLFO DE NICOYA
PUNTARENAS
PTO. CALDERA
ESPARZA
MIRAMAR
SAN RAMON
SAN MATEO
OROTINA
CIUDAD QUESADA
ZARCERO
Naranjo
SARCHI
GRECIA
ALAJUELA
HEREDIA
SAN JOSE
ASERRI
CARTAGO
TEJAR
PARAISO
TURRIALBA
JUAN VIÑAS
PUERTO VIEJO
GUAPILES
GUACIMO
SIQUIRRES
LIMON
V. Poás 2704
V. Barva 2906
V. Cacho Negro 2150
V. Irazú 3432
V. Turrialba 3339
PTA. LEONA
JACO
PARRITA
QUEPOS
SAN ISIDRO DE EL GENERAL
BUENOS AIRES
CORTES
Cerro de la Muerte 3491
Cerro Chirripó 3820
Isla del Coco
C. IGLESIAS 634
OCEANO PACIFICO
Pacific Ocean
Río Tempisque
Río Frío
Río San Juan
Río Sarapiquí
Río Reventazón
Río Pacuare
Río Parrita
Río Savegre
Río General
Río Térraba
Lag. Cote
LAGUNA DE ARENAL

Formulados en términos de objetos y las conexiones entre ellos

Modelado de redes de comunicaciones

Representación de mapas; lugares y distancias

Encontrar rutas hacia una solución (IA)

Transiciones de un estado a otro

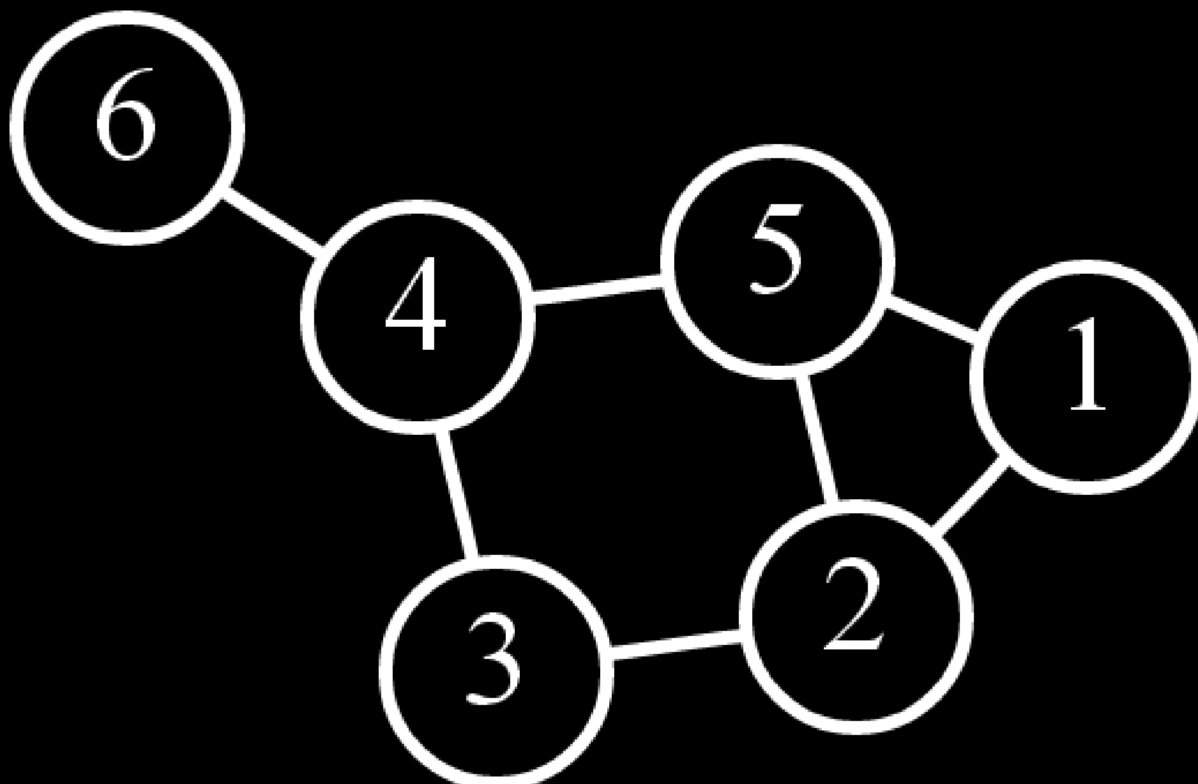
Representar organización de familias, empresas, etc.

Interconexiones (caminos)

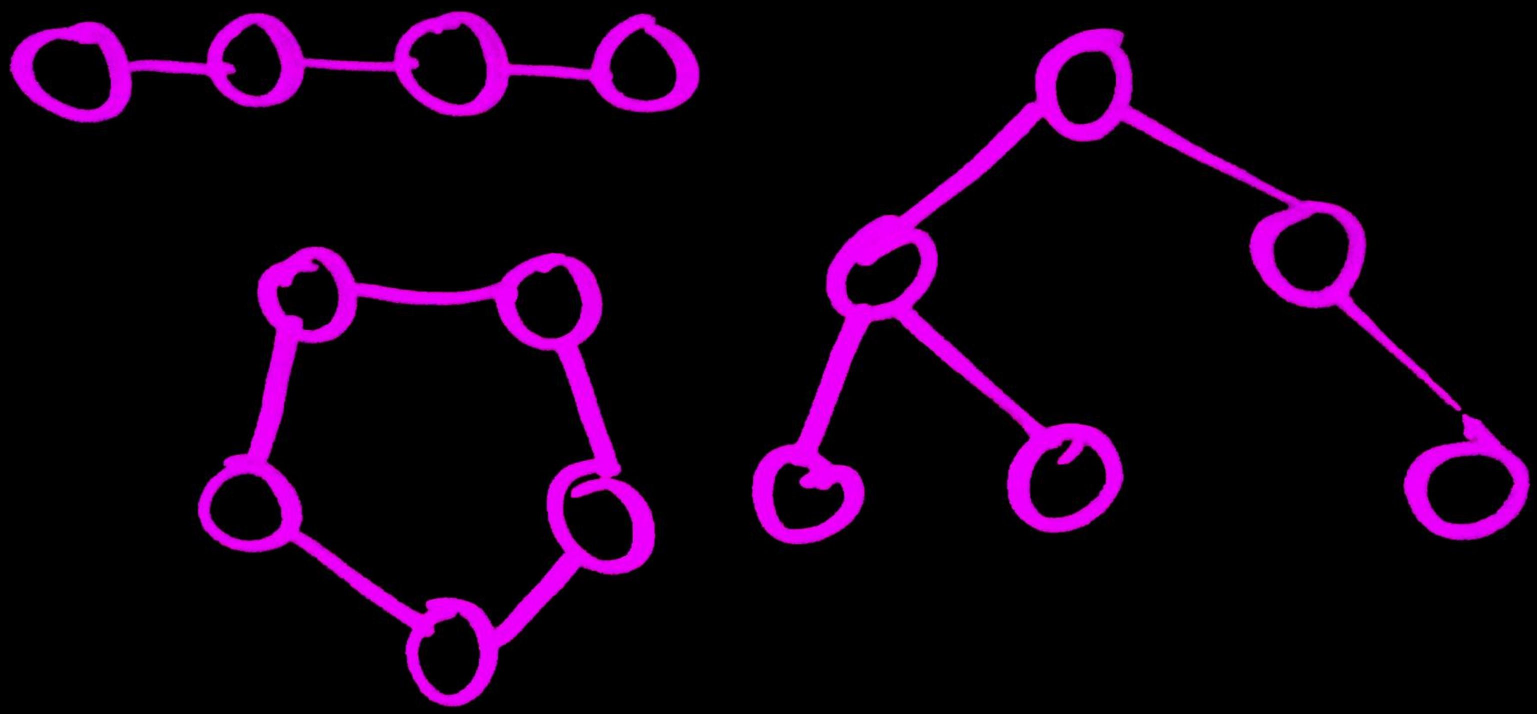
Objetos (pueblos)

Grafo

Objeto matemático que modela situaciones de este tipo


6
4
5
1
3
2

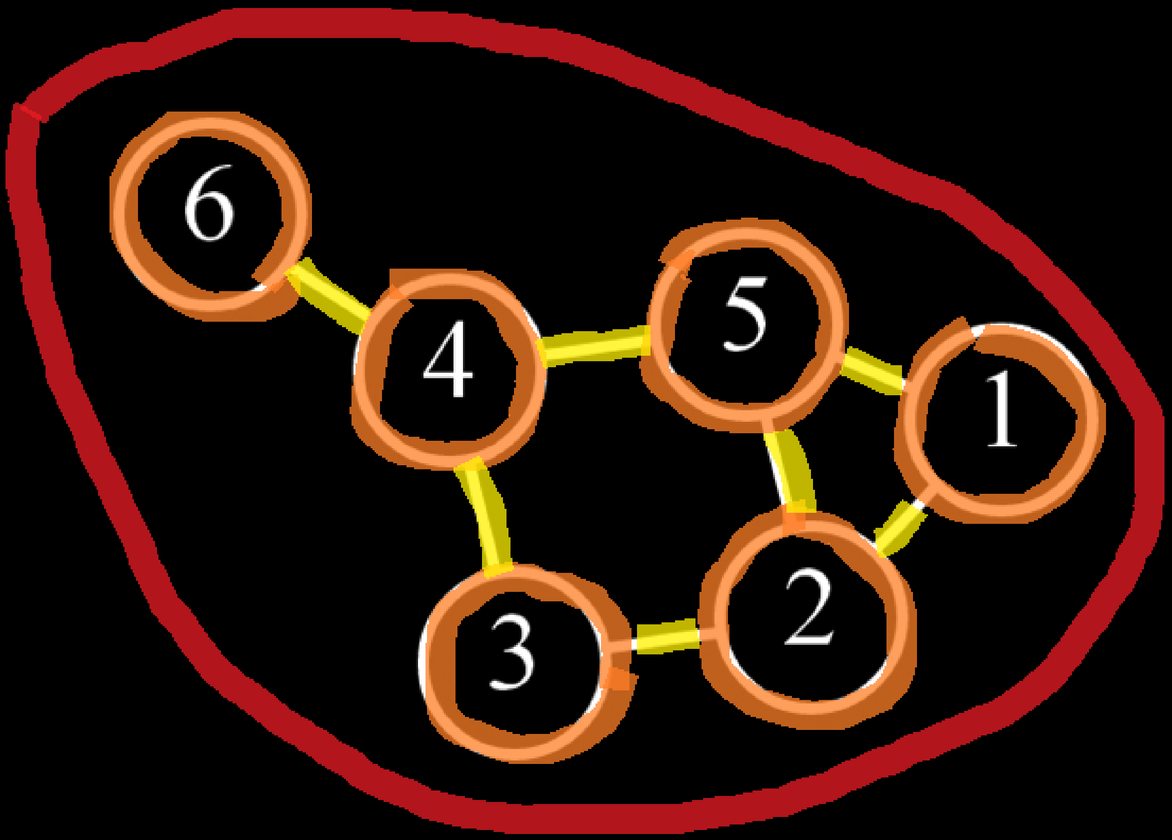
Las estructuras hasta ahora vistas son casos especiales de grafos



Grafo: colección de vértices/nodos y aristas/arcos

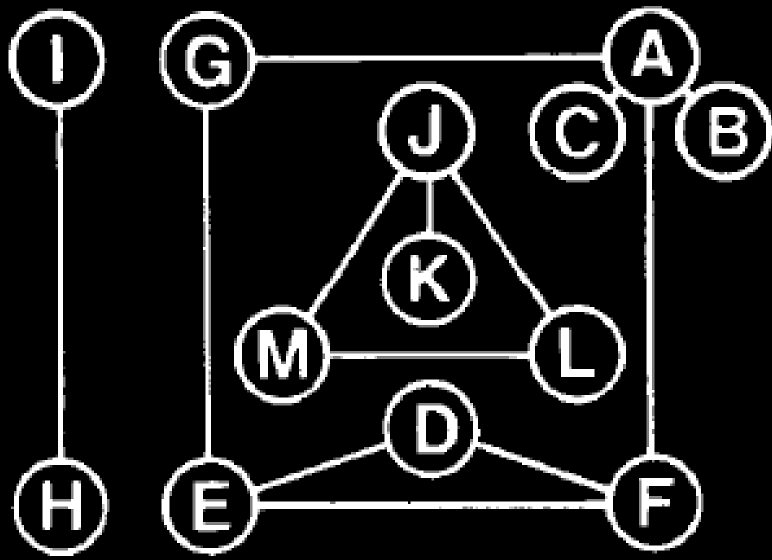
Nodos o vértices: objetos con nombre y más propiedades

Arista o arco: conexión entre dos vértices

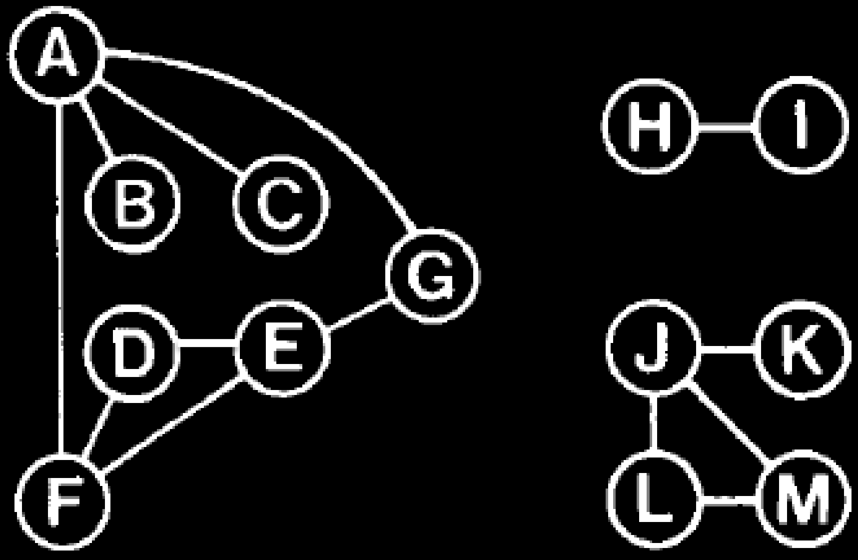


Lo que importa son los objetos y sus conexiones no tanto sus posiciones

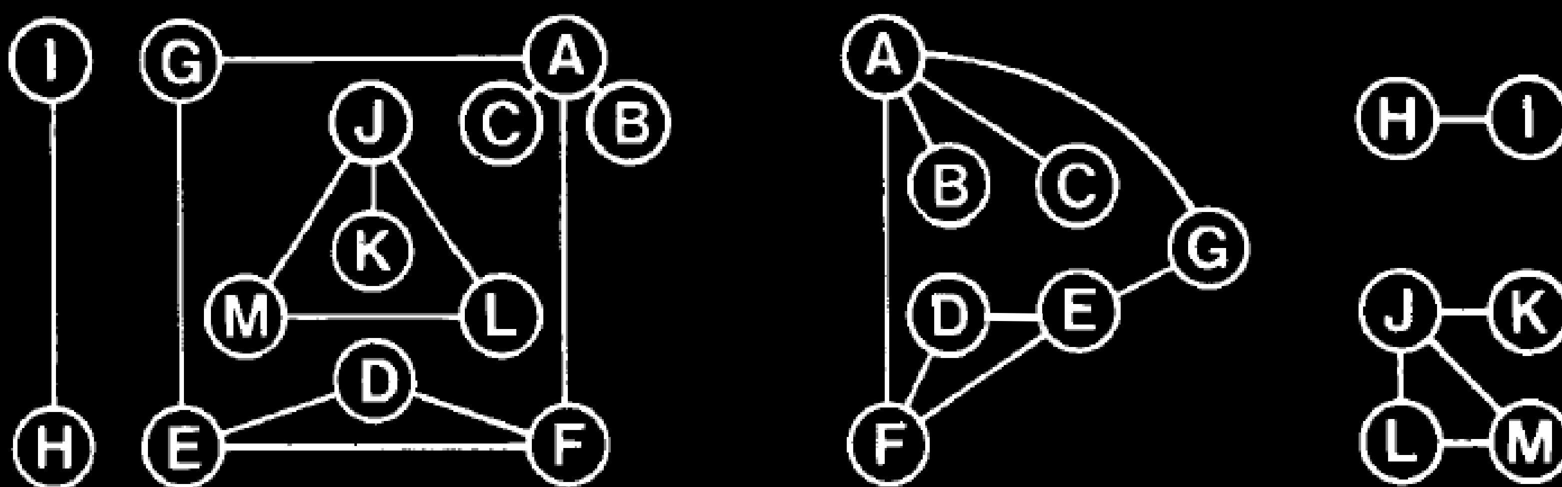
1



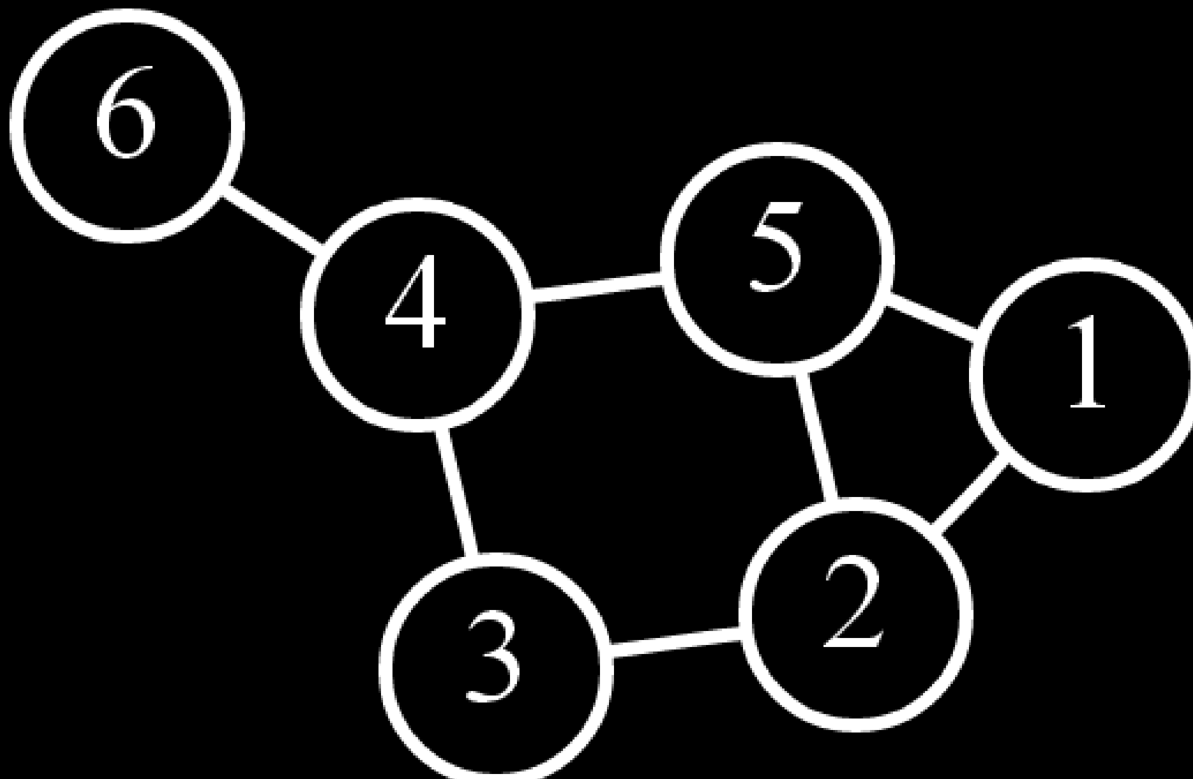
2



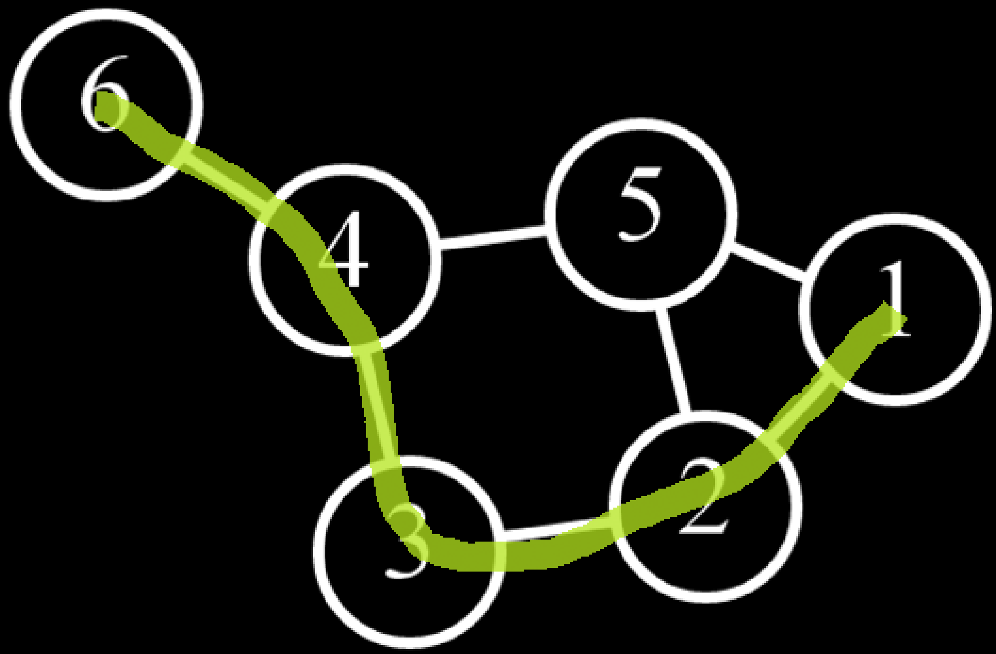
- Se representan:
  - Nodos: A, B, C, D, E, F, G, H, I, J, K, L, M
  - Aristas: AB, AC, AF, AG, DE, DF, EF, EG, HI, JK, JL, JM, LM



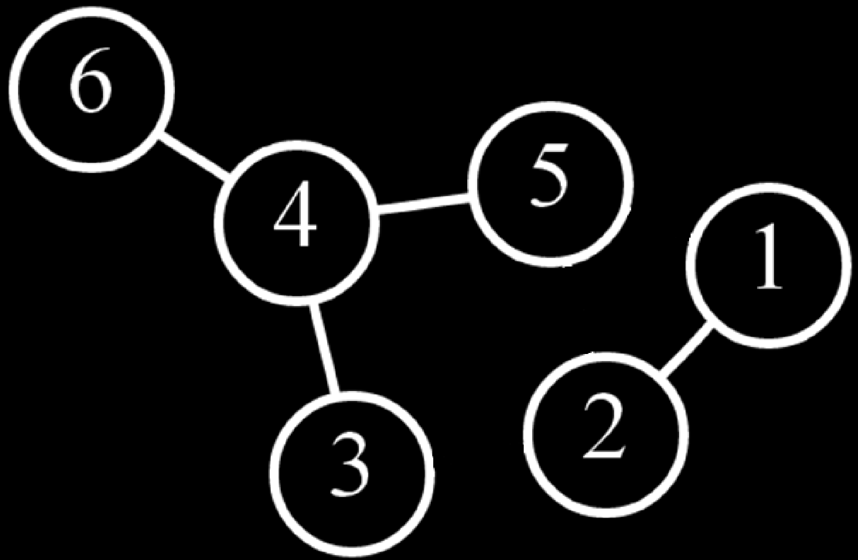
Dos vértices son adyacentes si existe una arista entre ellos dos (vecinos)

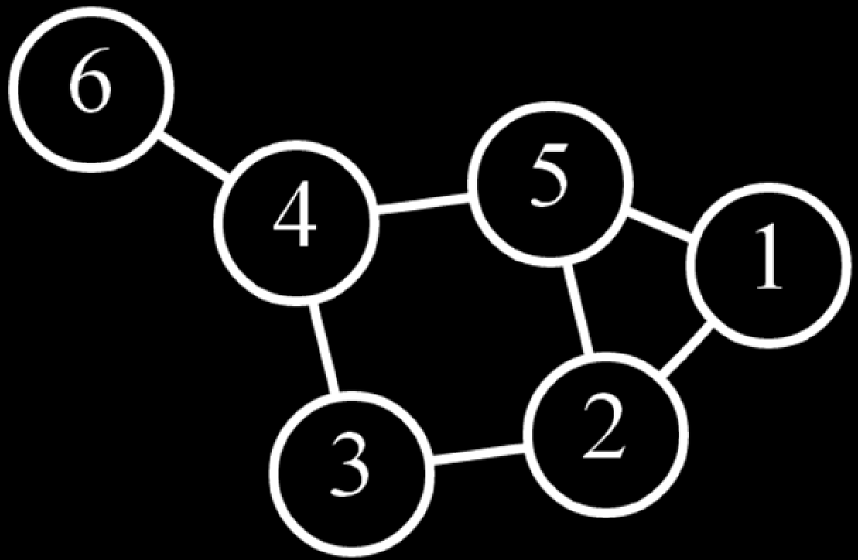


- Camino entre dos vértices
  - Lista de vértices conectados sucesivamente por aristas
  - Ejemplo: 6 4 3 2 1 es un camino de 6 a 1
  - El largo de un camino es la cantidad de aristas que contiene

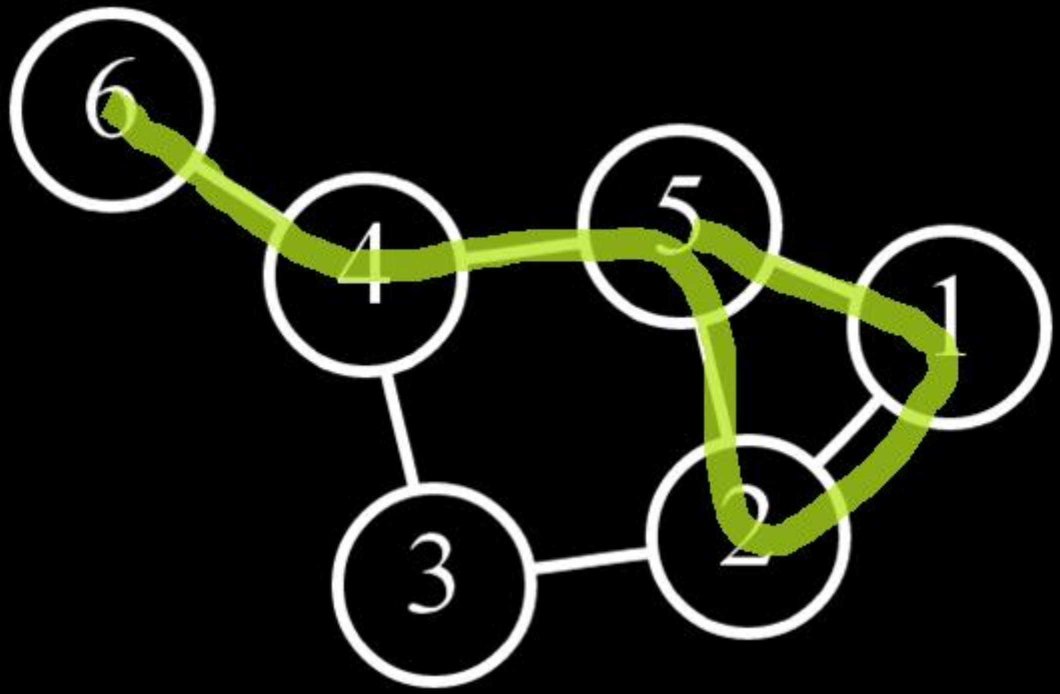


- Grafo conectado
  - Si existe un camino de cada nodo a cualquier otro nodo del grafo
  - Un grafo no conectado está formado por componentes conectados

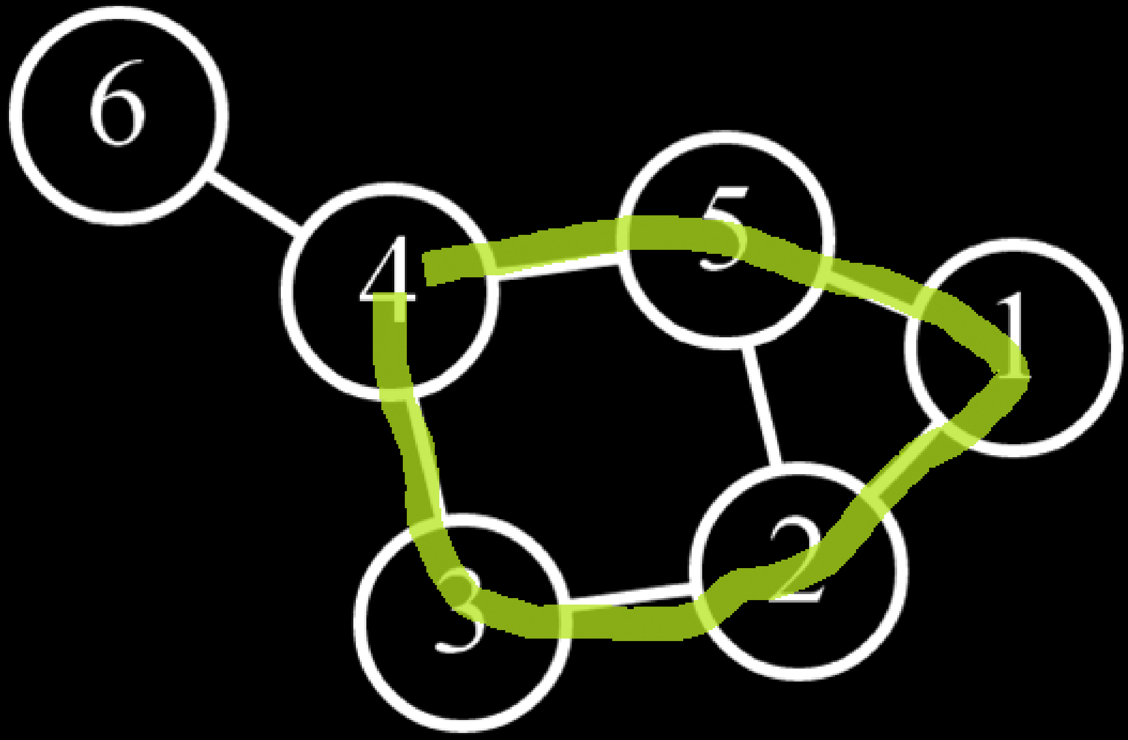




- Camino simple
  - Un camino donde no se repite un vértice
  - 6 4 5 2 1 5 no es un camino simple

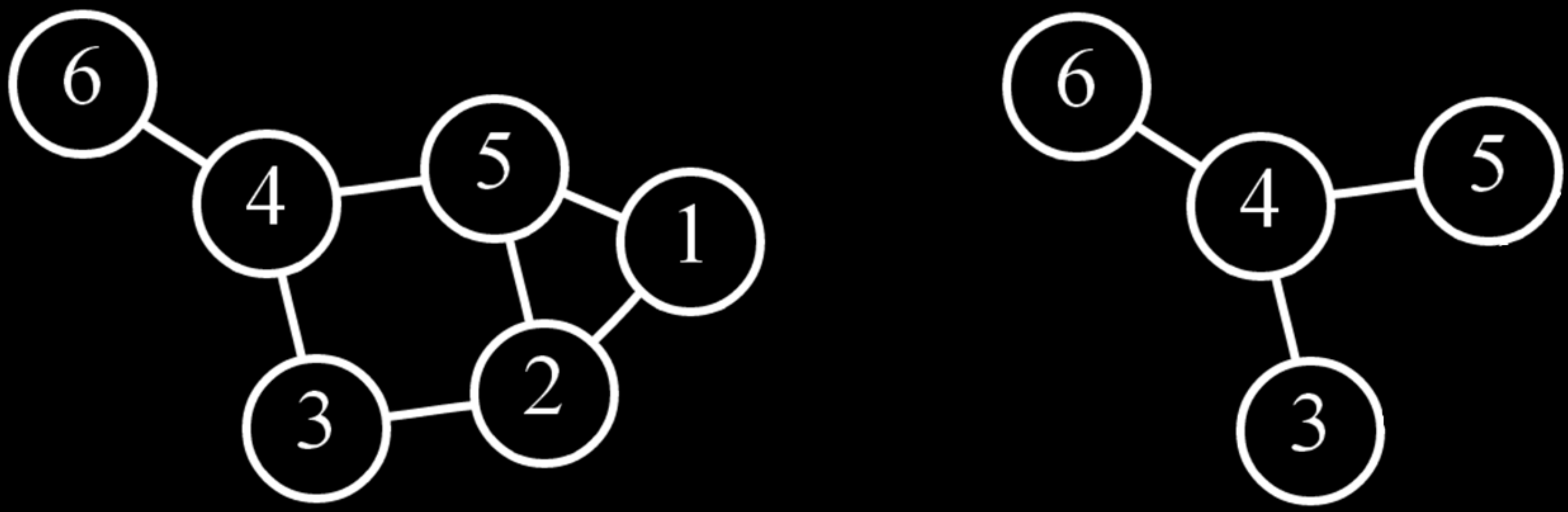


- Ciclo
  - Camino simple donde el primero y el último vértice son el mismo y no se repiten aristas

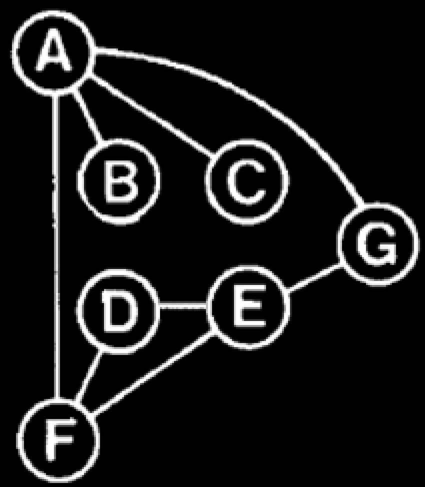


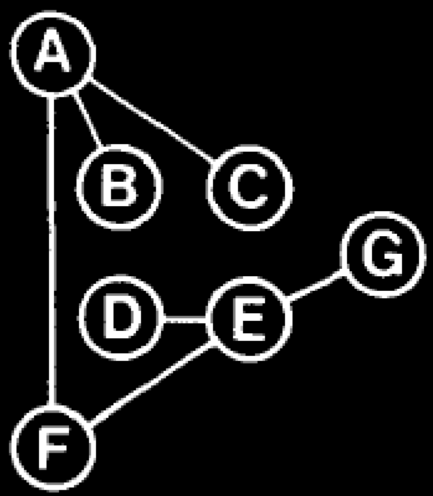
Un árbol es un grafo sin ciclos. Solo existe un camino entre dos nodos cualesquiera

- Subgrafo
  - Contiene un subconjunto de vértices y aristas de un grafo



- Árbol de expansión de un grafo
  - Es un subgrafo que contiene todos los vértices pero sólo las aristas necesarias para formar un árbol





- Grafo completo
  - Existe una arista entre todos los vértices del grafo
- Grafo no dirigido
  - Las aristas no tienen dirección. AB = BA
- Grafo dirigido
  - Las aristas tienen una dirección. AB != BA
- Grafo etiquetado
  - Las aristas tienen un valor asociado, texto o numérico
- Redes
  - Grafos dirigidos y etiquetados
- Entre más información contenga un grafo, más compleja es su representación

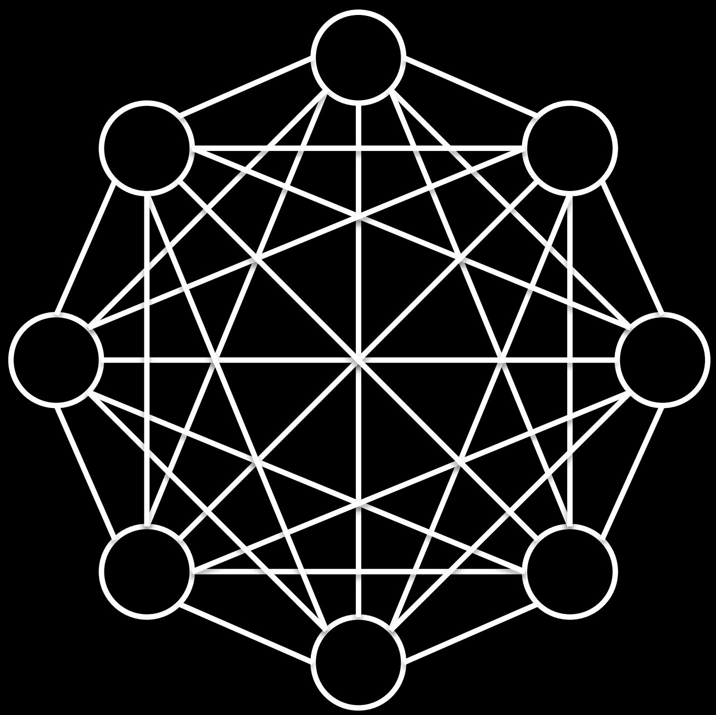
?

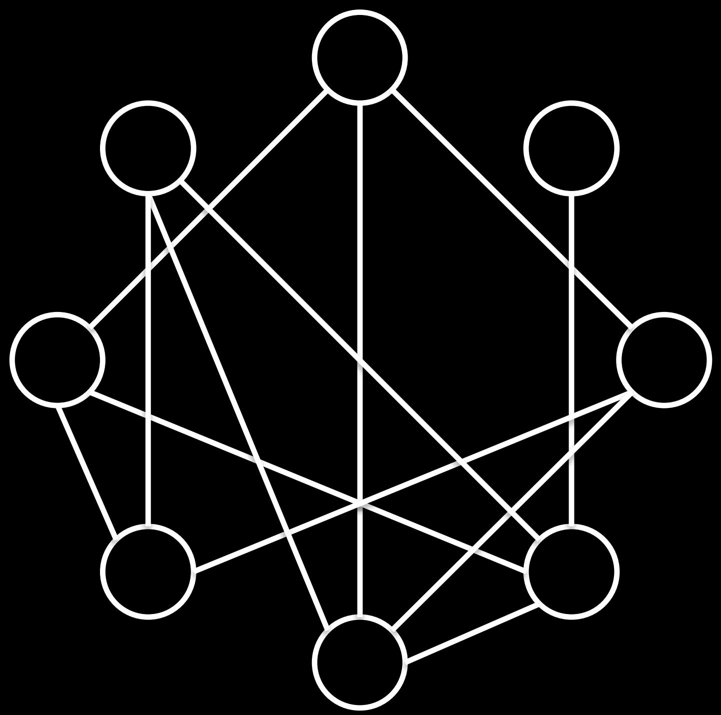
¿Cómo representar un grafo?

Representaciones de un grafo

Matriz de adyacencia → grafo denso

Listas de adyacencia → grafo ralo





Para cualquier representación

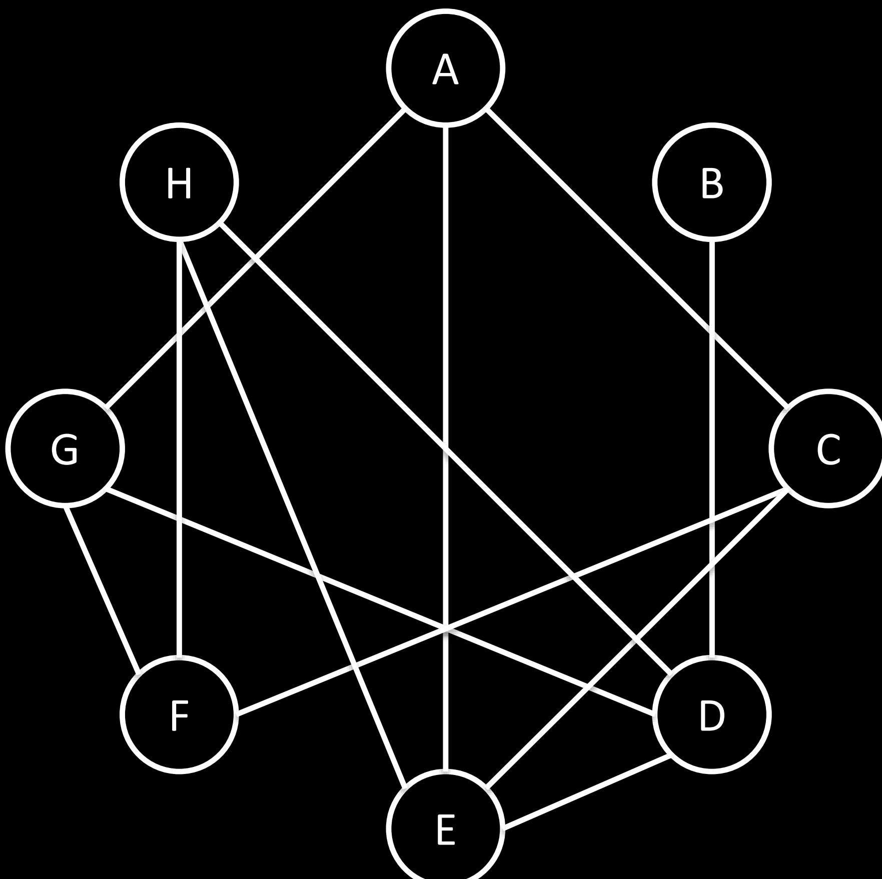
Mapear los nombres de los vértices a números para acceder su información con índices

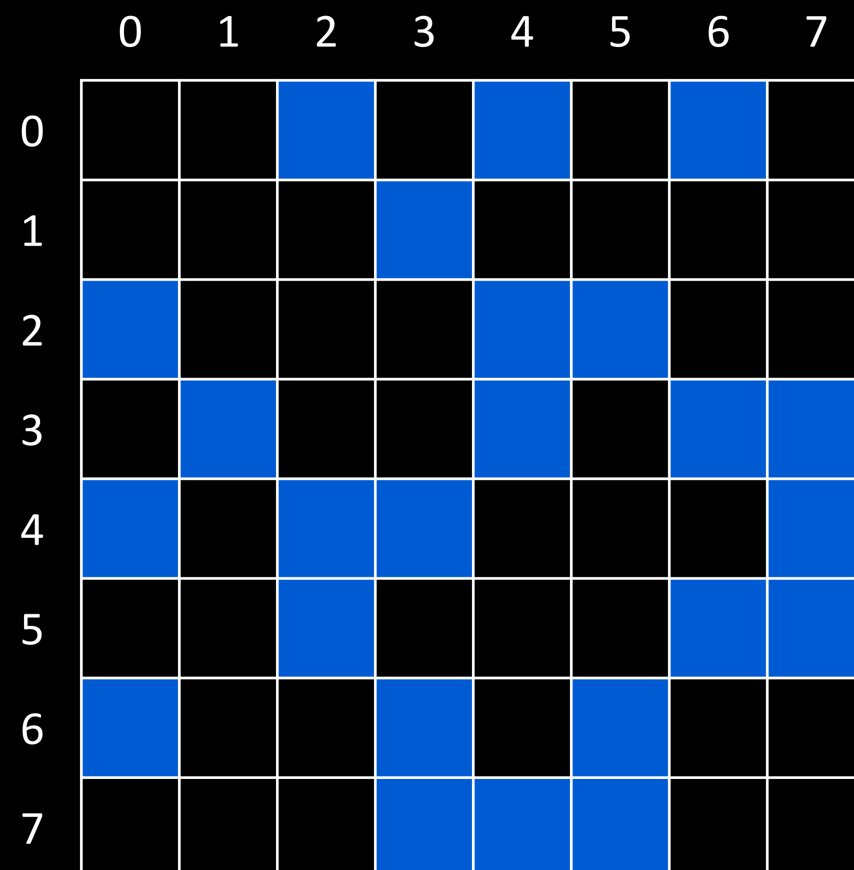
Se puede utilizar alguna estructura como diccionario o un árbol para esta relación

Matriz de adyacencia

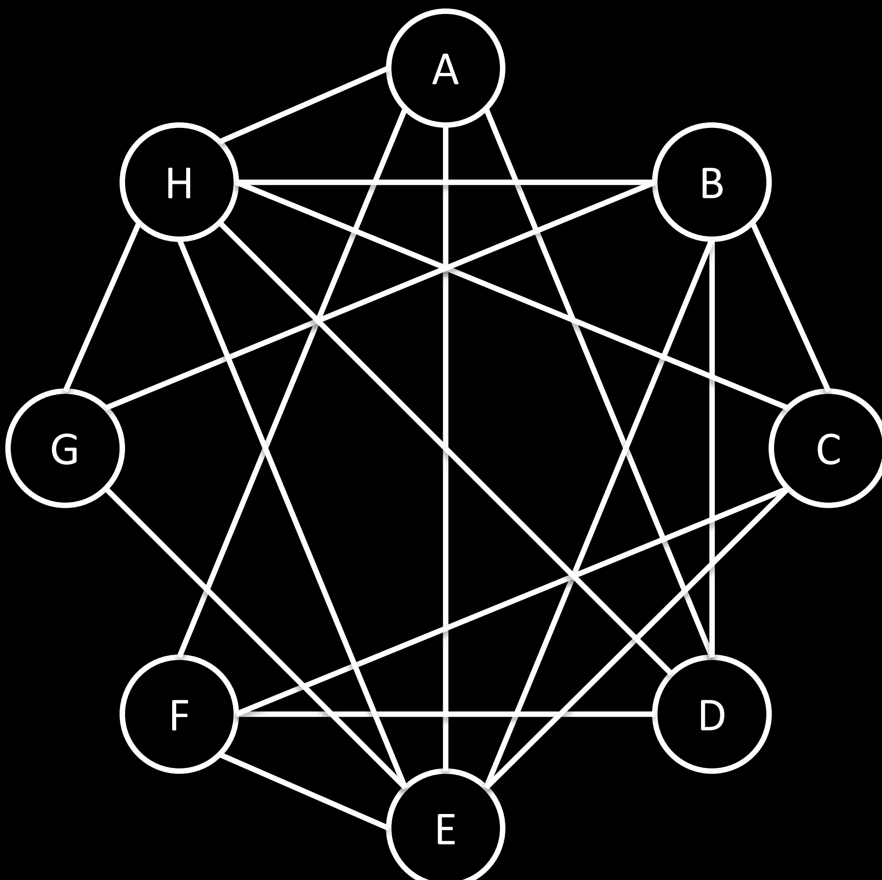
- Arreglo bidimensional de booleanos
- A[x][y] es true si existe una arista entre los vértices X y Y
- Cada arista se representa con dos celdas de la matriz
- Dependiendo del problema, se asume que existe una arista de un vértice hacia sí mismo

{ "A":0, "B":1, "C":2, "D":3, "E":4, "F":5, "G":6, "H":7 }





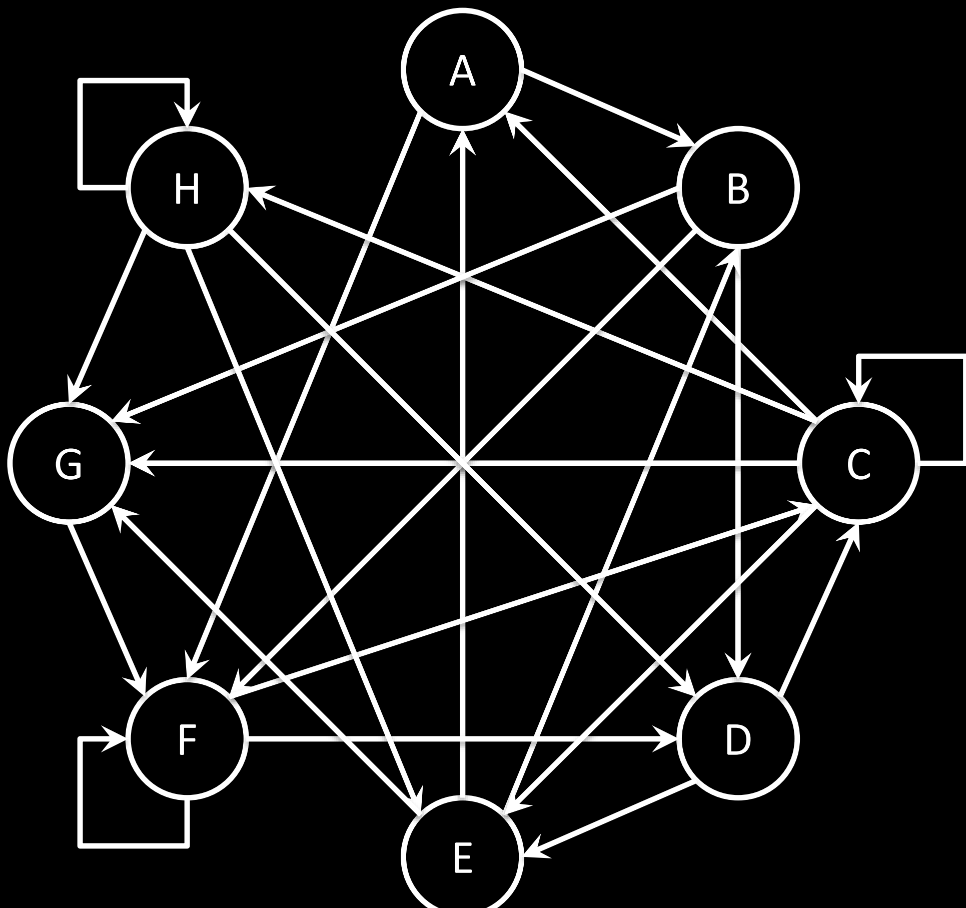
{ "A":0, "B":1, "C":2, "D":3, "E":4, "F":5, "G":6, "H":7 }



	0	1	2	3	4	5	6	7
0								
1								
2								
3								
4								
5								
6								
7								

- La matriz de adyacencia de un grafo no dirigido siempre es simétrica
- Si el grafo es denso, entonces la matriz es apropiada porque se ocupan la mayoría de las celdas
- Si el grafo es ralo, entonces la matriz va a tener muchos espacios vacíos
- Muchos vértices = matriz muy grande

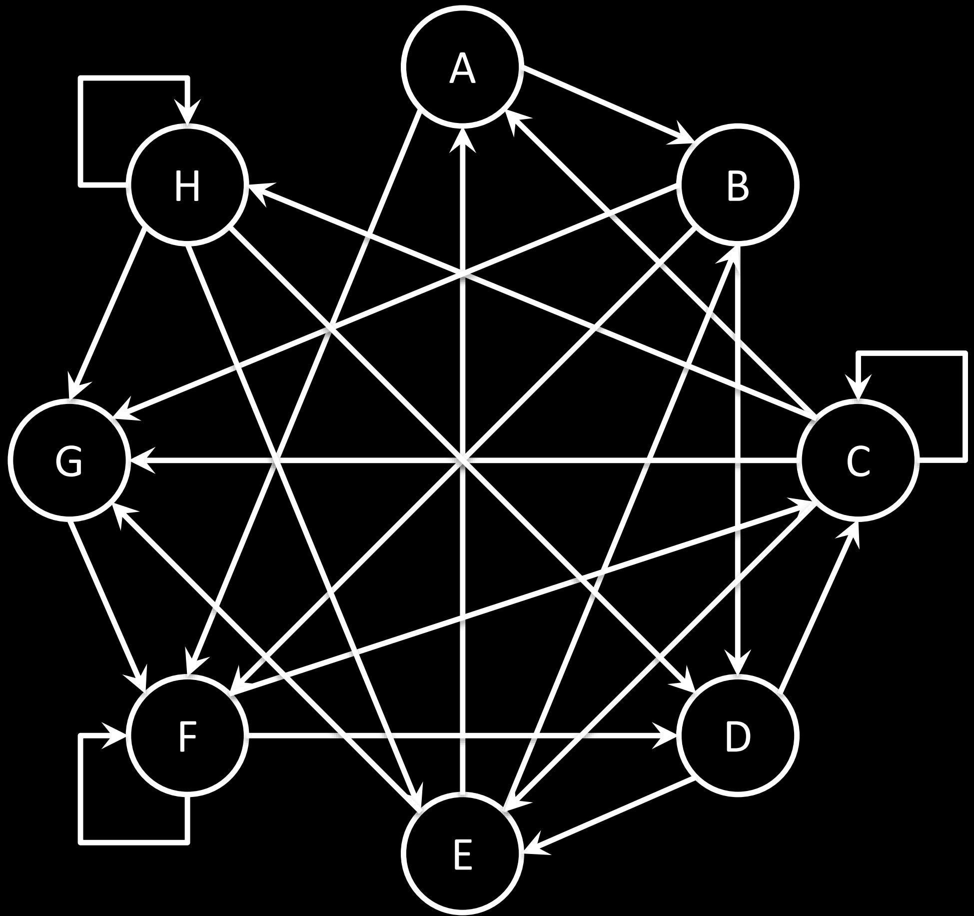
{ "A":0, "B":1, "C":2, "D":3, "E":4, "F":5, "G":6, "H":7 }



	0	1	2	3	4	5	6	7
0								
1								
2								
3								
4								
5								
6								
7								

Lista de adyacencia

- Cada vértice tiene una lista enlazada con las aristas que tiene
- El orden en que se inserten las aristas determina la forma de cada lista
- El borrado de un vértice es complicado pues se tienen que recorrer todas las listas


A
B
C
D
E
F
G
H

Grafos dirigidos y los grafos etiquetados se representan con las mismas estructuras

Grafo dirigido

Cada arista aparece sólo una vez en la matriz o en la lista de adyacencia

Un grafo no dirigido puede verse como uno dirigido con aristas en las dos direcciones

Grafo etiquetado

Matriz: Se guardan valores o pesos en vez de booleanos

Lista: Se incluye un campo para el peso en los nodos de las lisas de adyacencia

Algoritmos de grafos

- Recorridos
  - En profundidad
  - En anchura
- Árbol de expansión mínimo
  - Prim
  - Kruskal
- Ruta más corta entre dos nodos
  - Dijkstra
  - Floyd

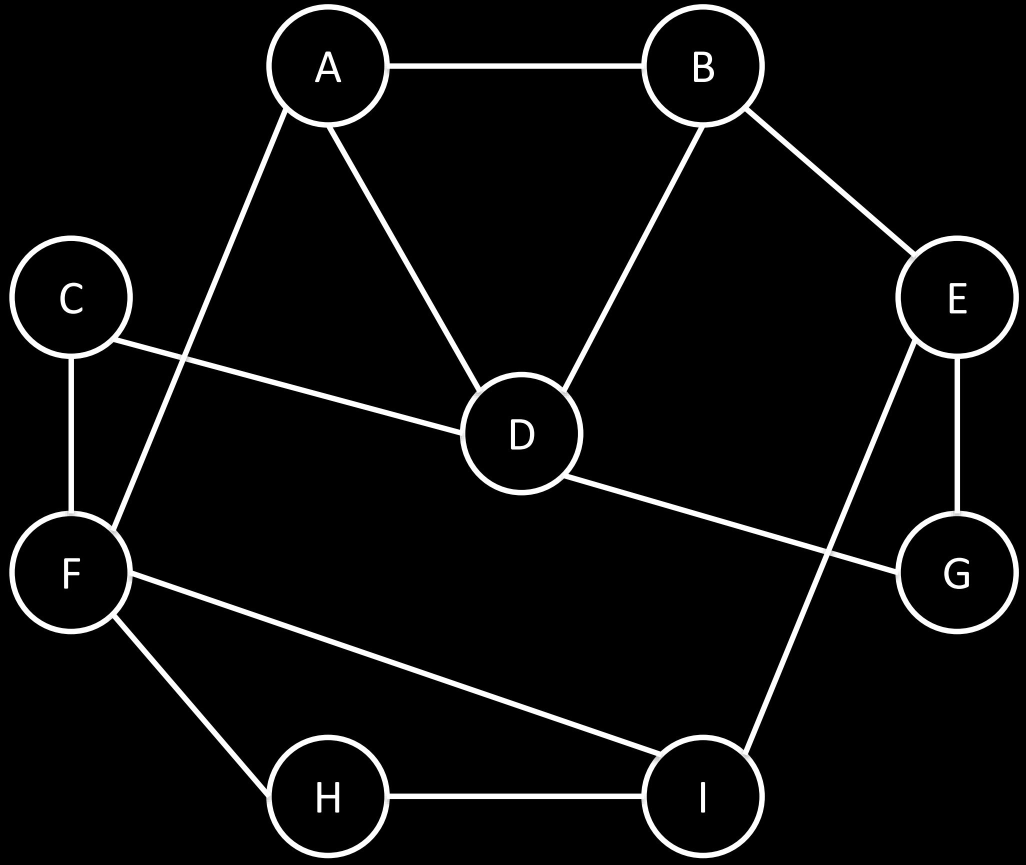
Recorridos

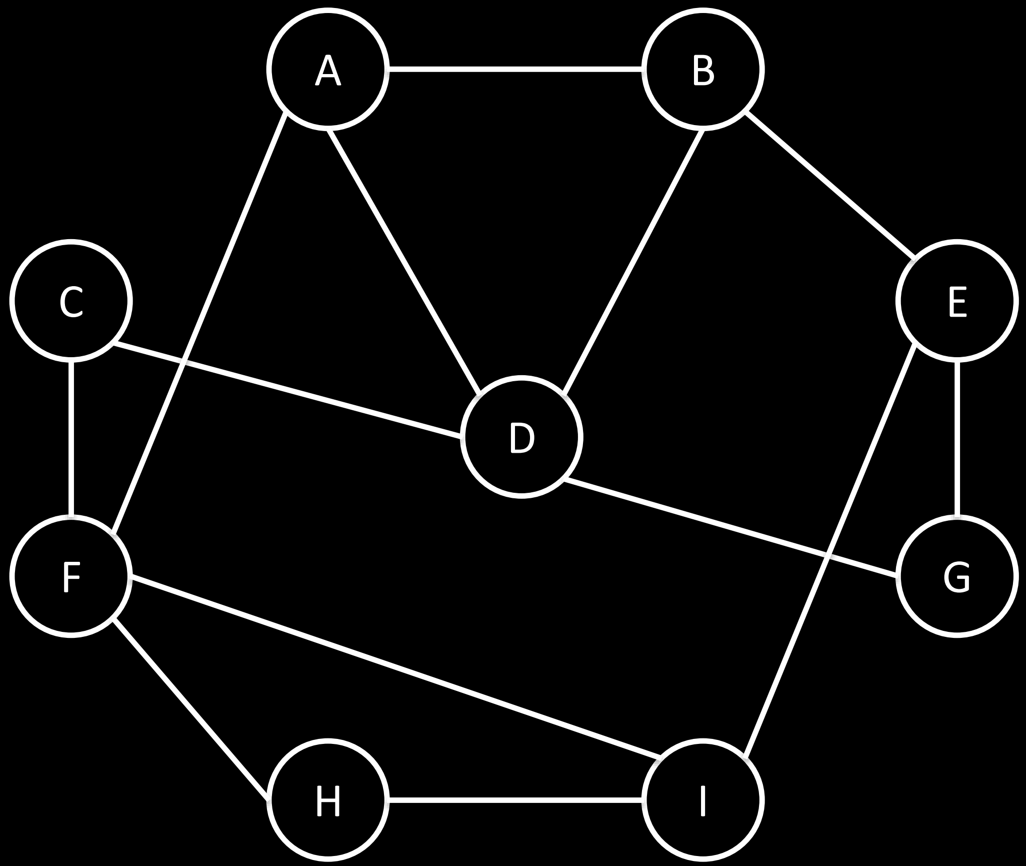
- Visitar todos los vértices de un grafo en un orden específico
- Inician en un vértice e intenta visitar todos desde ahí
- El grafo puede ser desconectado y puede que no se visiten todos los nodos
- Un grafo puede contener ciclos, el recorrido no puede quedarse enciclado
- Se utiliza una marca de visitado para determinar si ya se recorrió un vértice
  - Arreglo de bits para indicar si un nodo ha sido visitado

Búsqueda en profundidad
Depth-first search (DFS)

- La idea de este algoritmo es visitar recursivamente todos los vecinos no visitados de un nodo
- Cuando se visita un nodo se marca como visitado y se repite lo mismo con los vecinos no visitados
- Hay que hacer caso especial para vértices desconectados
  - Si al final de la ejecución del algoritmo quedan nodos sin visitar, repetir búsqueda desde uno de ellos
- También puede implementarse mediante una pila en lugar de la recusión

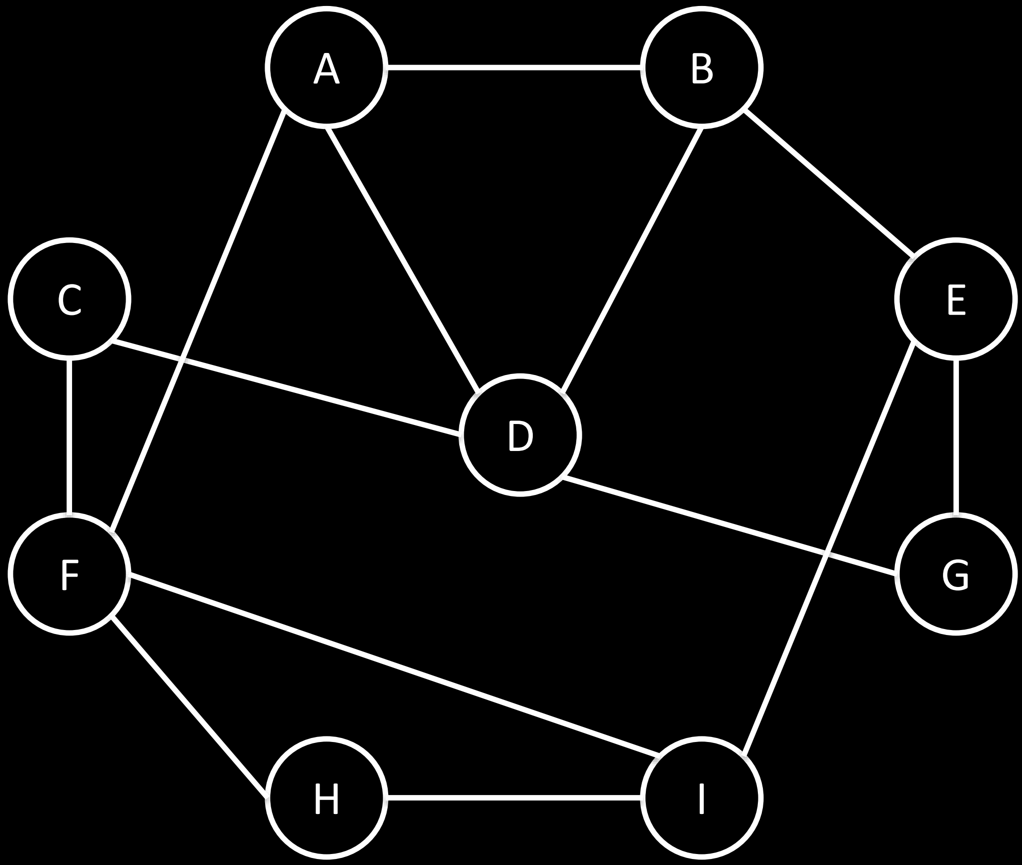
- La búsqueda en profundidad genera un árbol de expansión por profundidad (si el grafo es conectado)
- Puede ser utilizada para grafos dirigidos y no dirigidos

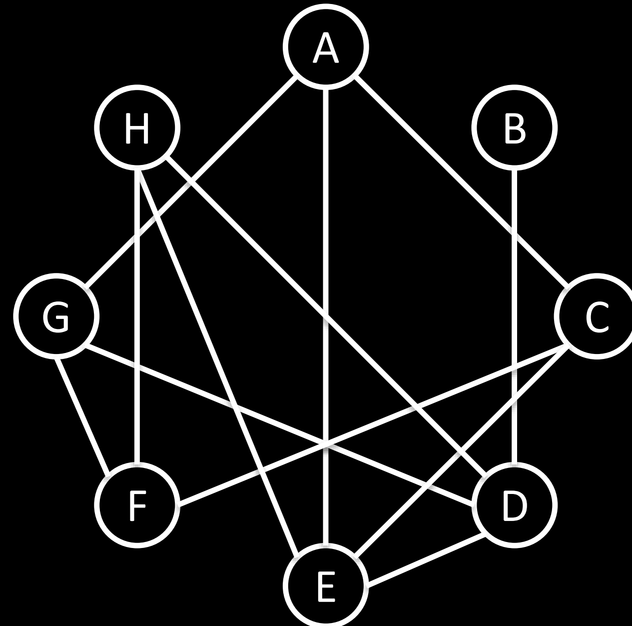


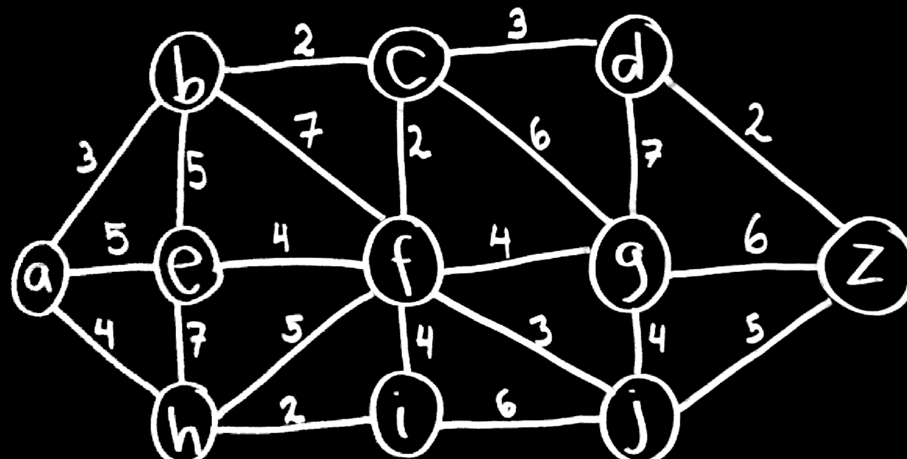

A
B
C
D
E
F
G
H
I

Búsqueda en anchura
Breadth-first search (BFS)

- Parecido al DFS, pero examina cada vecino antes de pasar a otro nodo
- Usa una cola en lugar de la pila de recursión
- Se visita un nodo, se marca como visitado
- Por cada vecino no visitado del nodo
  - Se visita
  - Se marca como visitado
  - Se agrega a la cola
- Se saca el primero de la cola y se repite el proceso hasta que la cola esté vacía


A
B
C
D
E
F
G
H
I


A
H
B
G
C
F
D
E

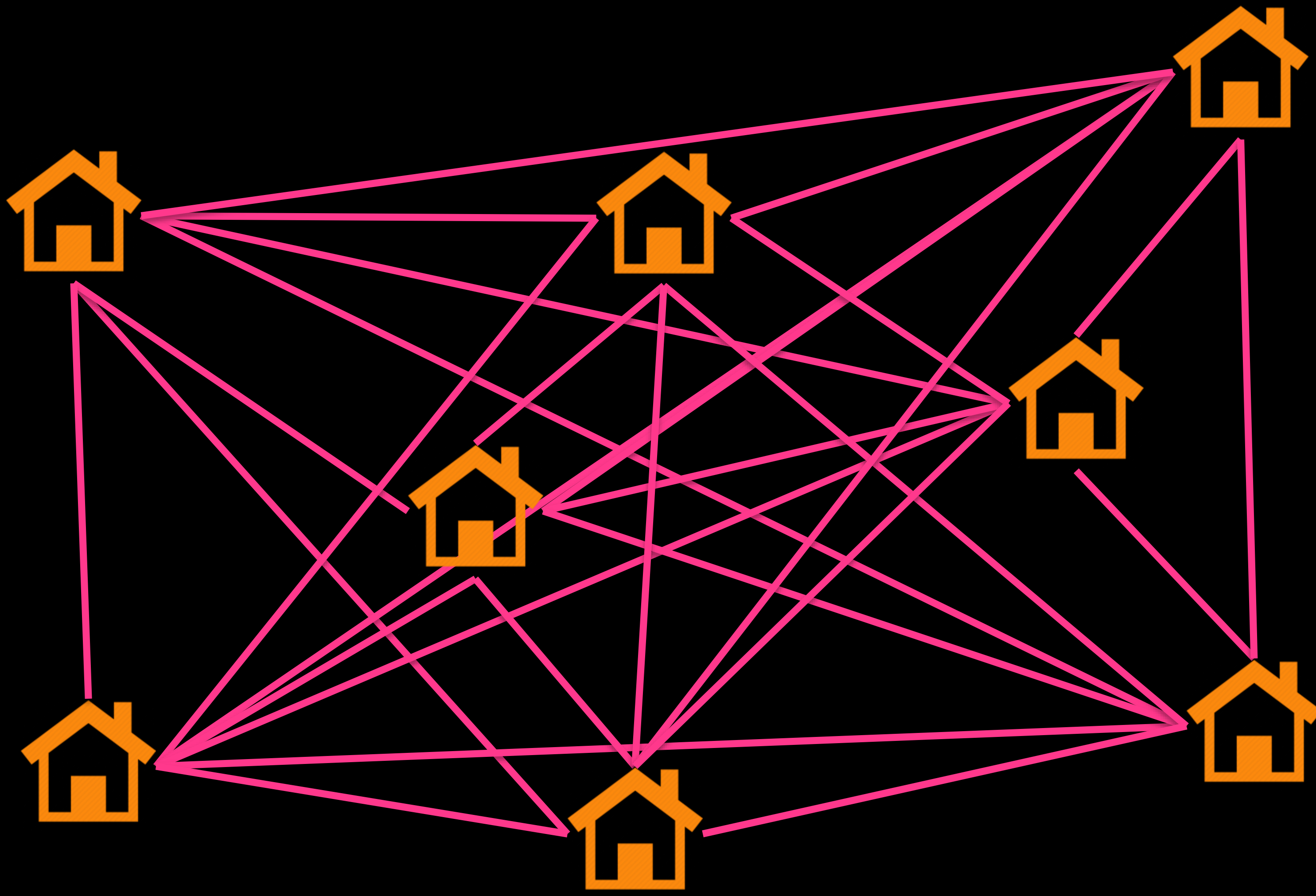

a
b
c
d
e
f
g
z
h
i
j
2
3
3
7
2
6
5
7
2
5
4
4
6
4
7
5
4
3
4
5
2
6

Grafos etiquetados

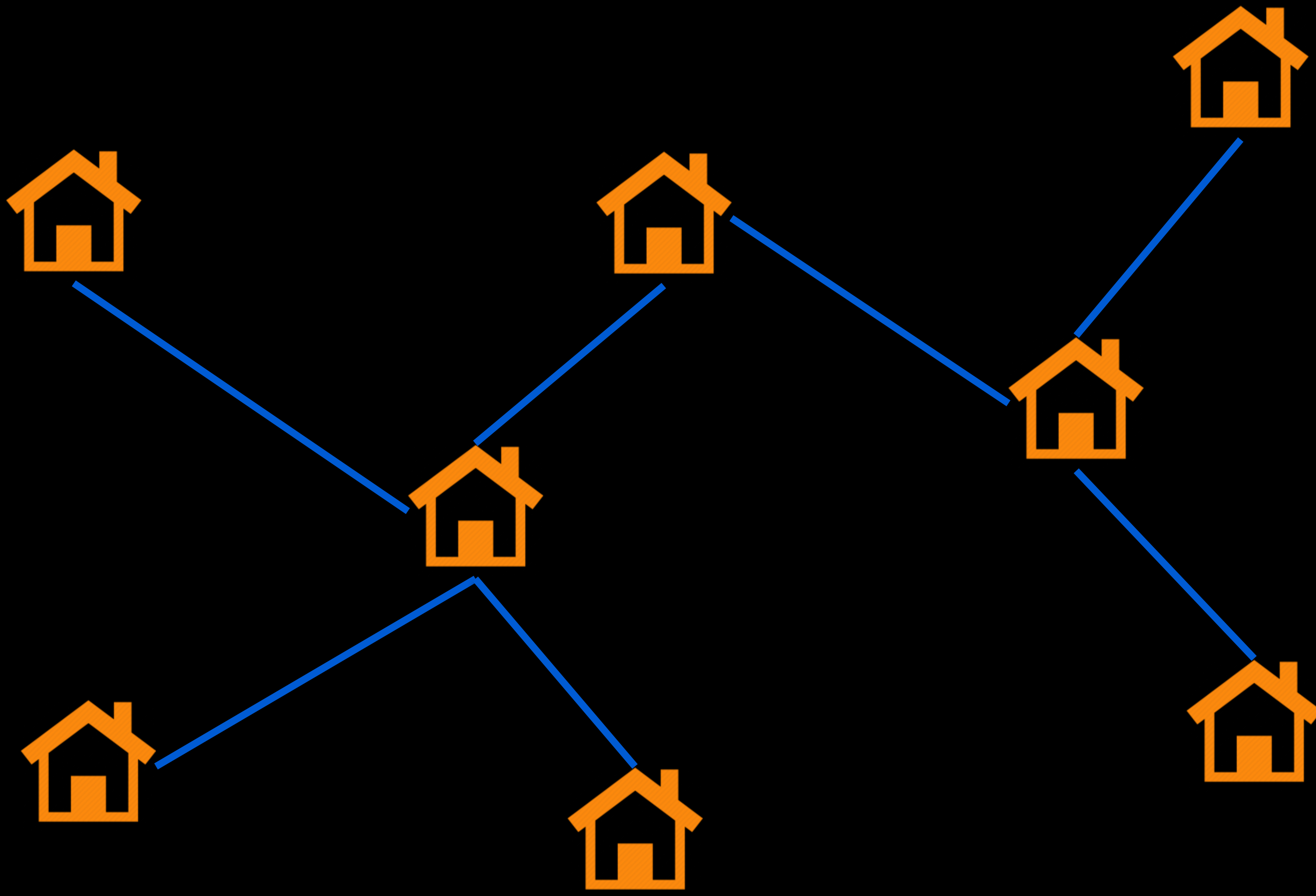
- Usualmente es necesario modelar problemas donde hay pesos o costos en las aristas
  - Mapa de rutas aéreas (distancias o costo)
- Surgen problemas:
  - La forma más barata de conectar todos los vértices
    - minimum spanning tree/árbol de expansión mínimo
  - Camino más barato entre dos vértices
    - shortest path/camino más corto

Árbol de expansión mínimo

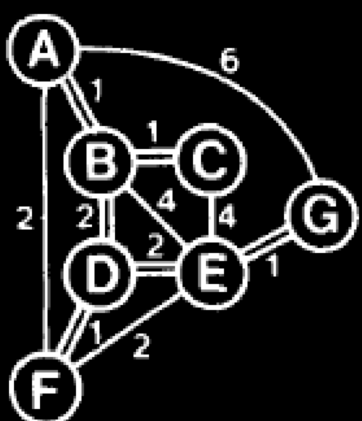
Árbol de expansión mínimo

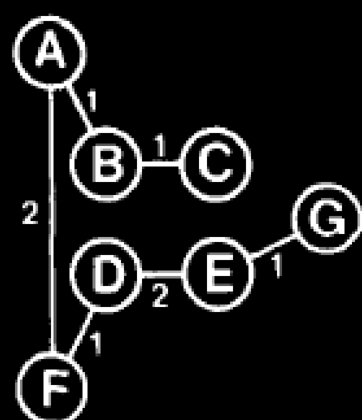


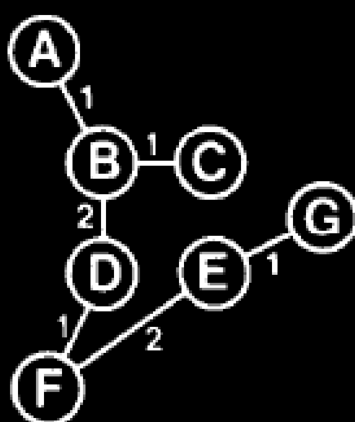
Árbol de expansión mínimo

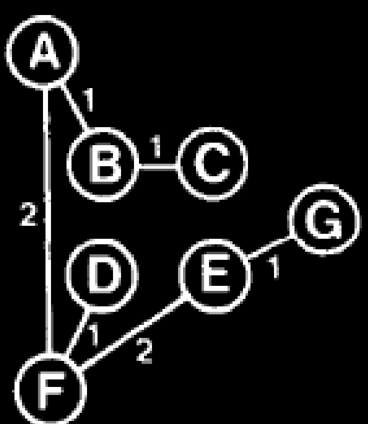


- Las aristas pueden representar tiempo, costos, etc. y no sólo distancias
- Subgrafo, árbol, contiene todos los vértices
- La suma del peso de todas sus aristas es la mínima
- Pueden haber varios árboles de expansión mínimos









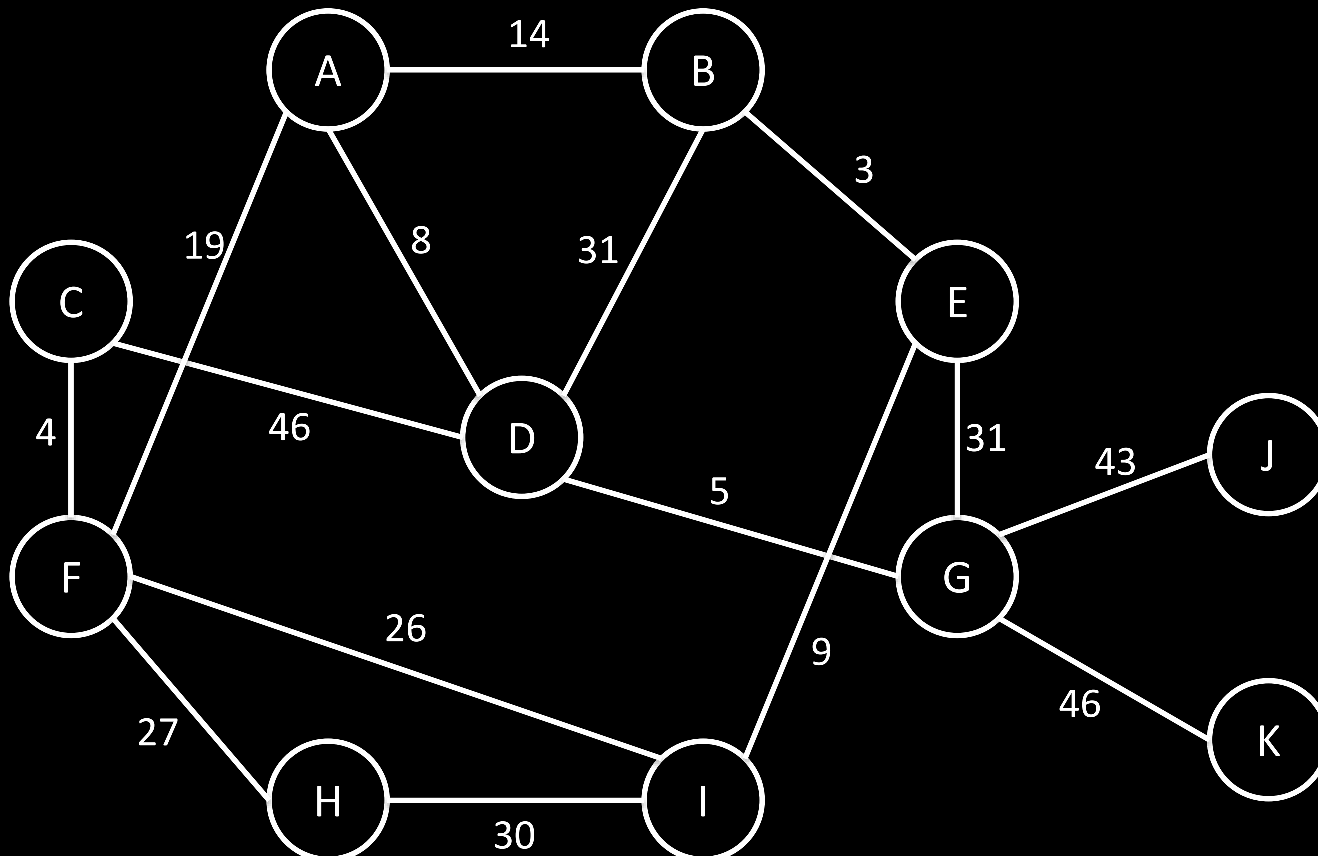
Algoritmo de Prim

Vojtěch Jarník (1897-1970 checo) en 1930
Robert Prim (1921- estadounidense) en 1957
Edsger Dijkstra (1930-2002 holandés) en 1959
También se conoce como algoritmo DJP

Requiere que el grafo esté conectado

Escoger un vértice cualquiera y agregarlo al árbol

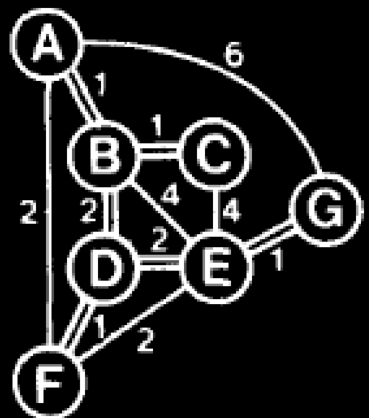
1. Ver las aristas que conectan el árbol con nodos desconectados y escoger la más corta
2. Agregar el nodo que conecta esa arista al árbol
3. Repetir 1 y 2 hasta que todos los vértices estén en el árbol

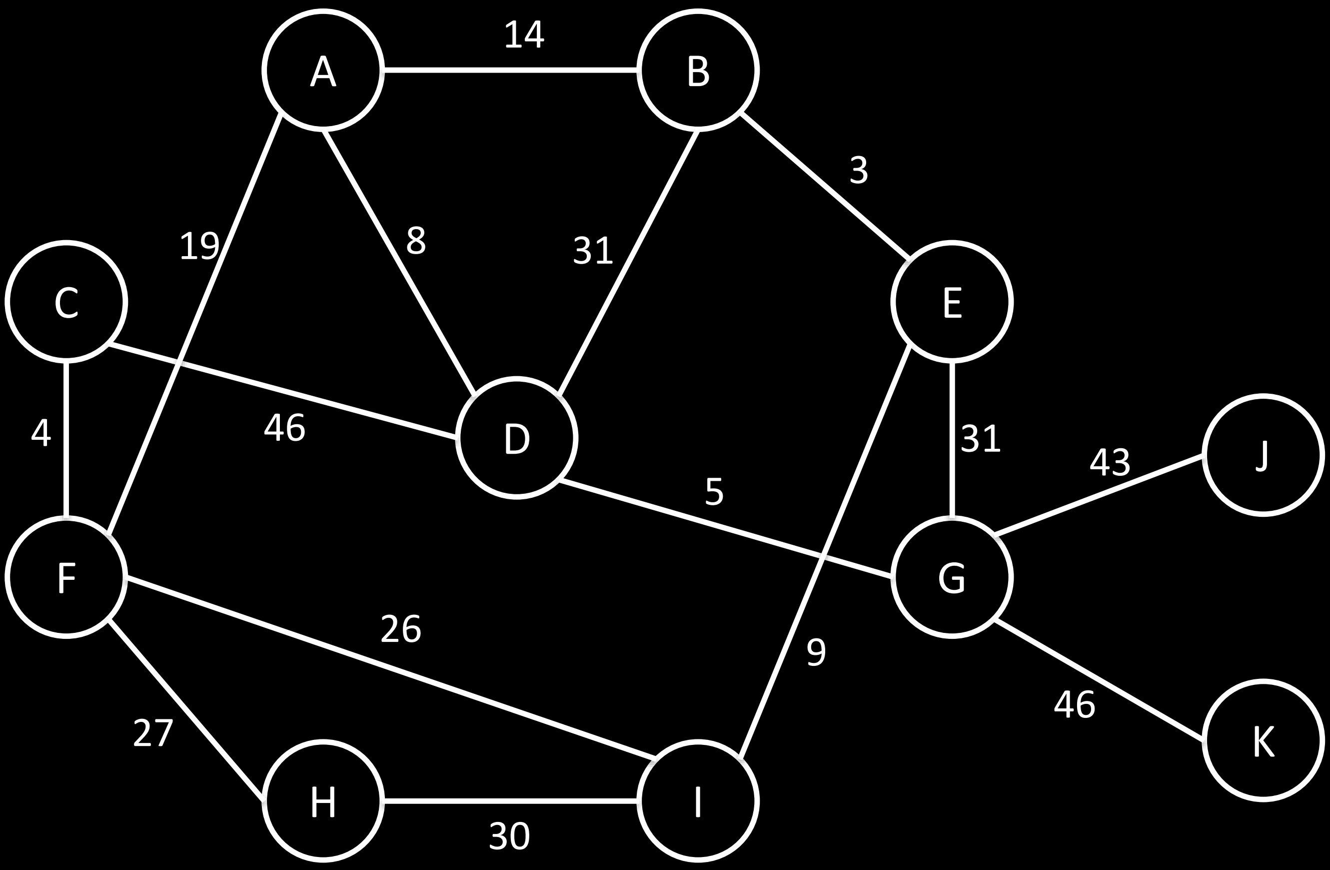

A
B
C
D
E
F
G
H
I
J
K
14
19
8
31
3
4
46
5
31
43
26
9
27
30
46

Algoritmo de Kruskal

Joseph Kruskal (1928-2010 estadounidense) en 1956

1. Ordenar las aristas de menor a mayor según su peso
2. Agregar al árbol la menor arista encontrada, si esta no produce un ciclo
3. Repetir el paso 2 hasta que todos los vértices estén conectados





A
B
C
D
E
F
G
H
I
J
K
14
3
19
8
31
4
46
5
31
43
26
9
46
27
30

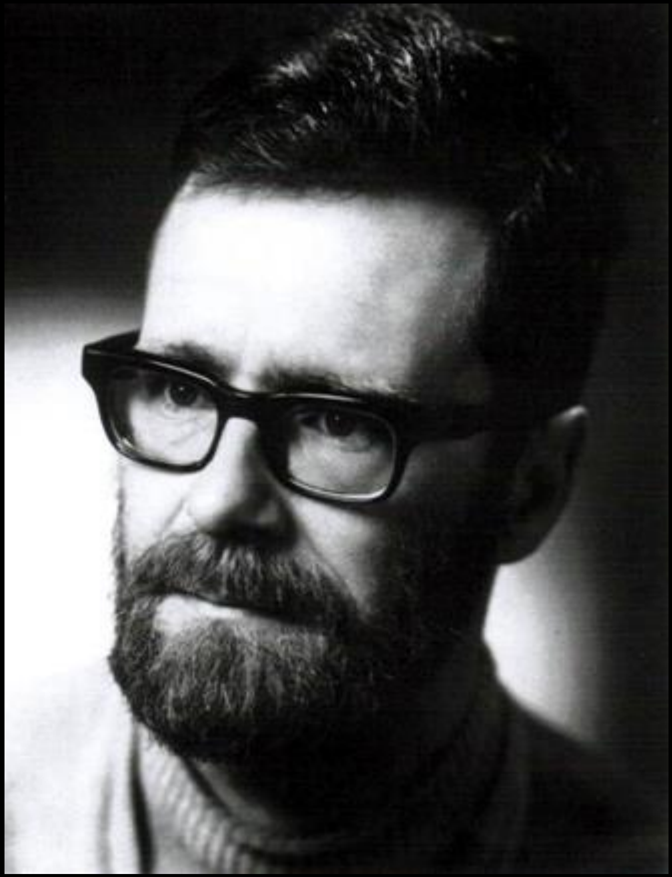
Algoritmo de Dijkstra

Edsger Dijkstra (1930-2002 holandés) en 1959

Sirve para encontrar el camino más corto entre un nodo y todos los demás

No funciona si hay aristas con valores negativos

Utiliza programación dinámica



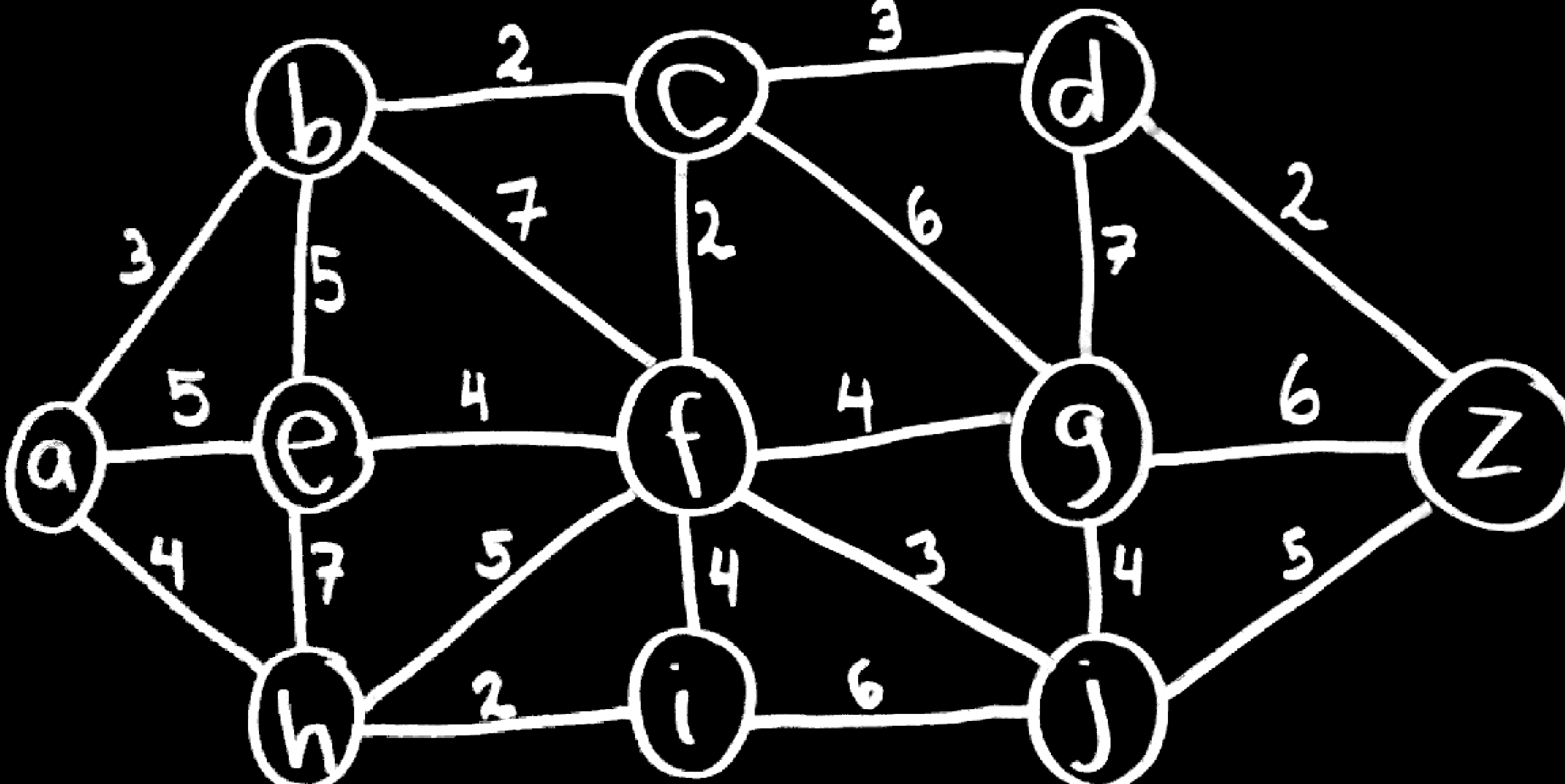
Inicialización

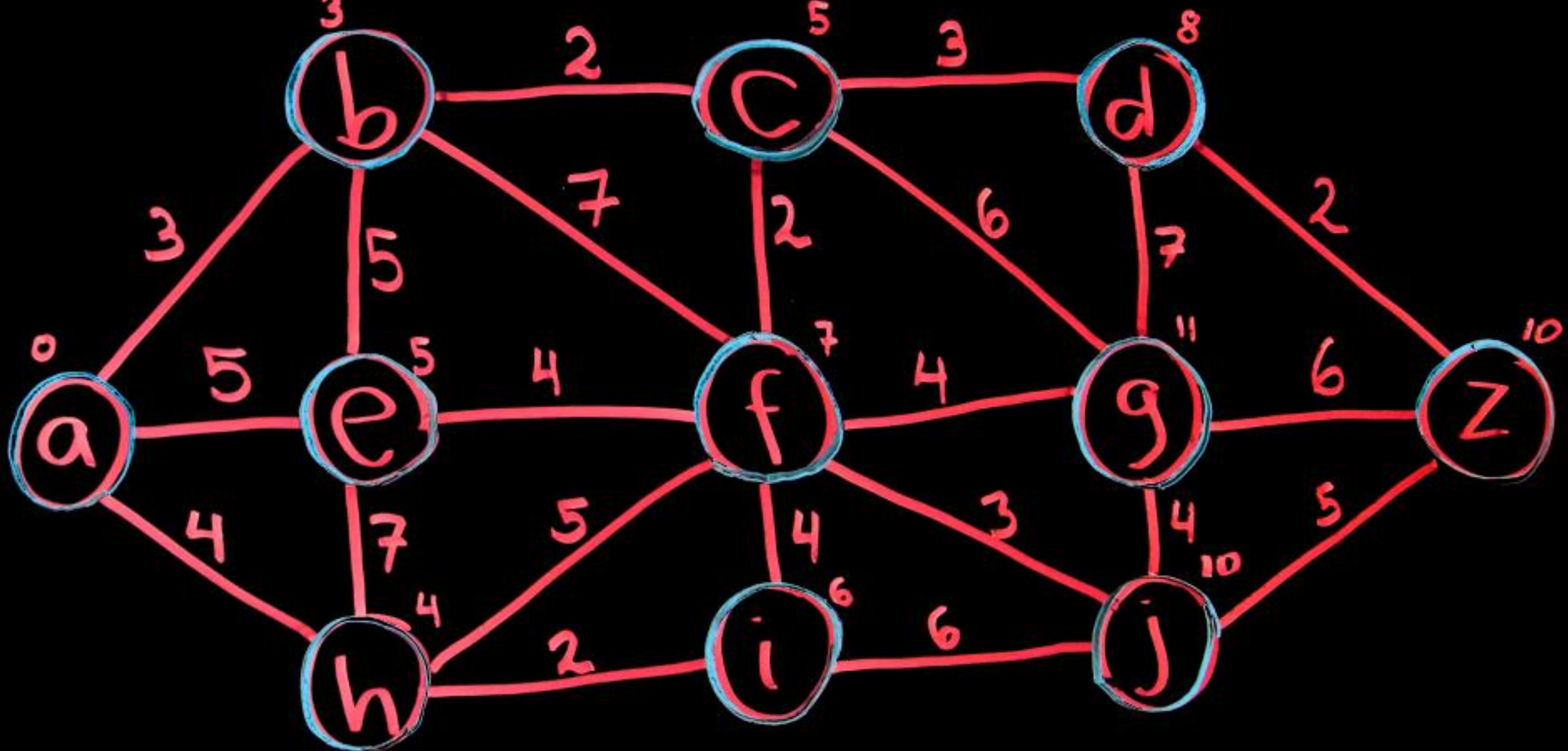
Se necesita:

1. Un nodo actual. Se escoge el origen para iniciar.
2. Arreglo de distancias para llegar a cada nodo. Se inicializa con valores infinitos, excepto el nodo inicial en 0.
3. Arreglo para indicar de cuál nodo proviene cada distancia (rutas). Se puede inicializar con cualquier valor, excepto el inicial que se inicializa con el mismo nodo inicial.
4. Arreglo de booleanos para distancias definitivas. Se inicializa todo en falso, excepto el inicial que se pone como definitivo.

Pasos

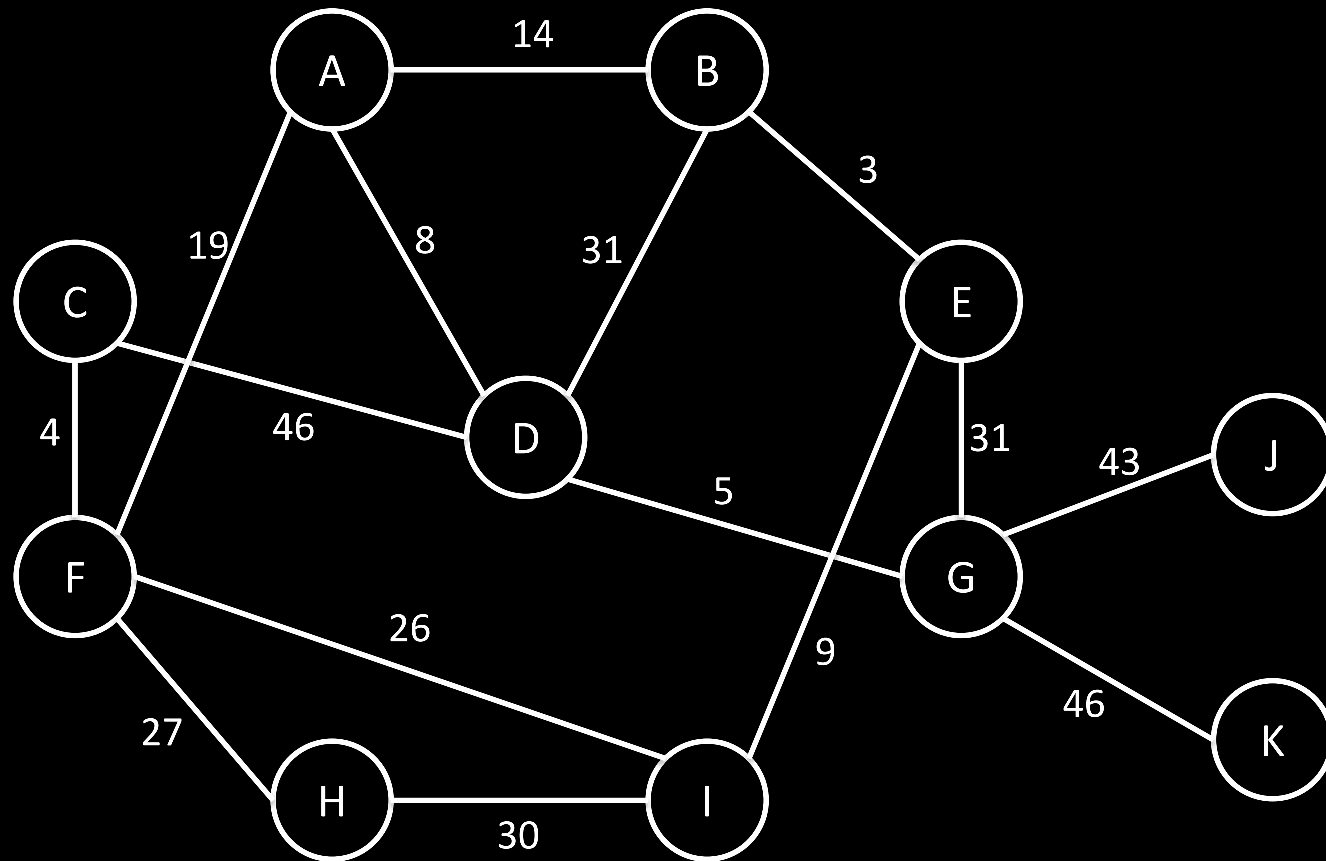
1. Considerar las aristas del nodo actual hacia vecinos que no estén marcados como definitivos
2. Sumar la distancia acumulada del nodo actual con la distancia hacia cada nodo vecino
3. Si la distancia calculada es menor que la registrada, actualizar arreglo de distancias y actualizar el nodo de donde proviene
4. De todos los nodos del grafo que no estén marcados como definitivos, escoger el que tenga la distancia acumulada menor y marcarlo como definitivo
5. Escoger dicho nodo como nuevo nodo actual
6. Repetir desde el paso 1 hasta que todos los nodos hayan sido marcados como definitivos.


3
2
b
c
d
7
2
6
2
7
3
5
5
4
4
6
a
e
f
g
z
4
7
5
4
3
4
5
h
2
i
6
j



a
b
c
d
e
f
g
h
i
j
z
0
3
5
8
5
7
11
10
4
6
10
3
5
2
5
7
2
3
6
7
2
4
4
6
5
7
5
4
3
4
5
2
6

	a	b	c	d	e	f	g	h	i	j	z
	$\boxed{a0}$	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞
a	$\boxed{a0}$	$\boxed{a3}$	∞	∞	a5	∞	∞	a4	∞	∞	∞
b	$\boxed{a0}$	$\boxed{a3}$	b5	∞	a5	b10	∞	$\boxed{a4}$	∞	∞	∞
h	$\boxed{a0}$	$\boxed{a3}$	$\boxed{b5}$	∞	a5	h9	∞	$\boxed{a4}$	h6	∞	∞
c	$\boxed{a0}$	$\boxed{a3}$	$\boxed{b5}$	c8	$\boxed{a5}$	c7	c11	$\boxed{a4}$	h6	∞	∞
e	$\boxed{a0}$	$\boxed{a3}$	$\boxed{b5}$	c8	$\boxed{a5}$	c7	c11	$\boxed{a4}$	$\boxed{h6}$	∞	∞
i	$\boxed{a0}$	$\boxed{a3}$	$\boxed{b5}$	c8	$\boxed{a5}$	$\boxed{c7}$	c11	$\boxed{a4}$	$\boxed{h6}$	i12	∞
f	$\boxed{a0}$	$\boxed{a3}$	$\boxed{b5}$	$\boxed{c8}$	$\boxed{a5}$	$\boxed{c7}$	c11	$\boxed{a4}$	$\boxed{h6}$	f10	∞
d	$\boxed{a0}$	$\boxed{a3}$	$\boxed{b5}$	$\boxed{c8}$	$\boxed{a5}$	$\boxed{c7}$	c11	$\boxed{a4}$	$\boxed{h6}$	$\boxed{f10}$	d10
j	$\boxed{a0}$	$\boxed{a3}$	$\boxed{b5}$	$\boxed{c8}$	$\boxed{a5}$	$\boxed{c7}$	c11	$\boxed{a4}$	$\boxed{h6}$	$\boxed{f10}$	$\boxed{d10}$
z	$\boxed{a0}$	$\boxed{a3}$	$\boxed{b5}$	$\boxed{c8}$	$\boxed{a5}$	$\boxed{c7}$	$\boxed{c11}$	$\boxed{a4}$	$\boxed{h6}$	$\boxed{f10}$	$\boxed{d10}$


A
B
C
D
E
F
G
H
I
J
K
14
3
19
8
31
4
46
5
31
43
26
9
27
46
30

Algoritmo de Floyd

Bernard Roy (1934- francés) en 1959

Stephen Warshall (1935-2006) en 1962

Robert Floyd (1936-2001) en 1962

También conocido como algoritmo WFI

Encuentra las rutas más cortas entre todos los pares de nodos en un grafo

Funciona con arcos negativos, pero no pueden haber ciclos negativos

Utiliza programación dinámica

Inicialización

- Dos matrices (n x n)
  - Matriz para distancias acumuladas (D)
  - Matriz para rutas (P)
- Deben eliminarse aristas del mismo nodo hacia el mismo nodo y aristas dobles (dejar la menor)
- La matriz D se inicializa con la matriz de adyacencia del grafo
- La matriz P se inicializa cada fila con el número de fila

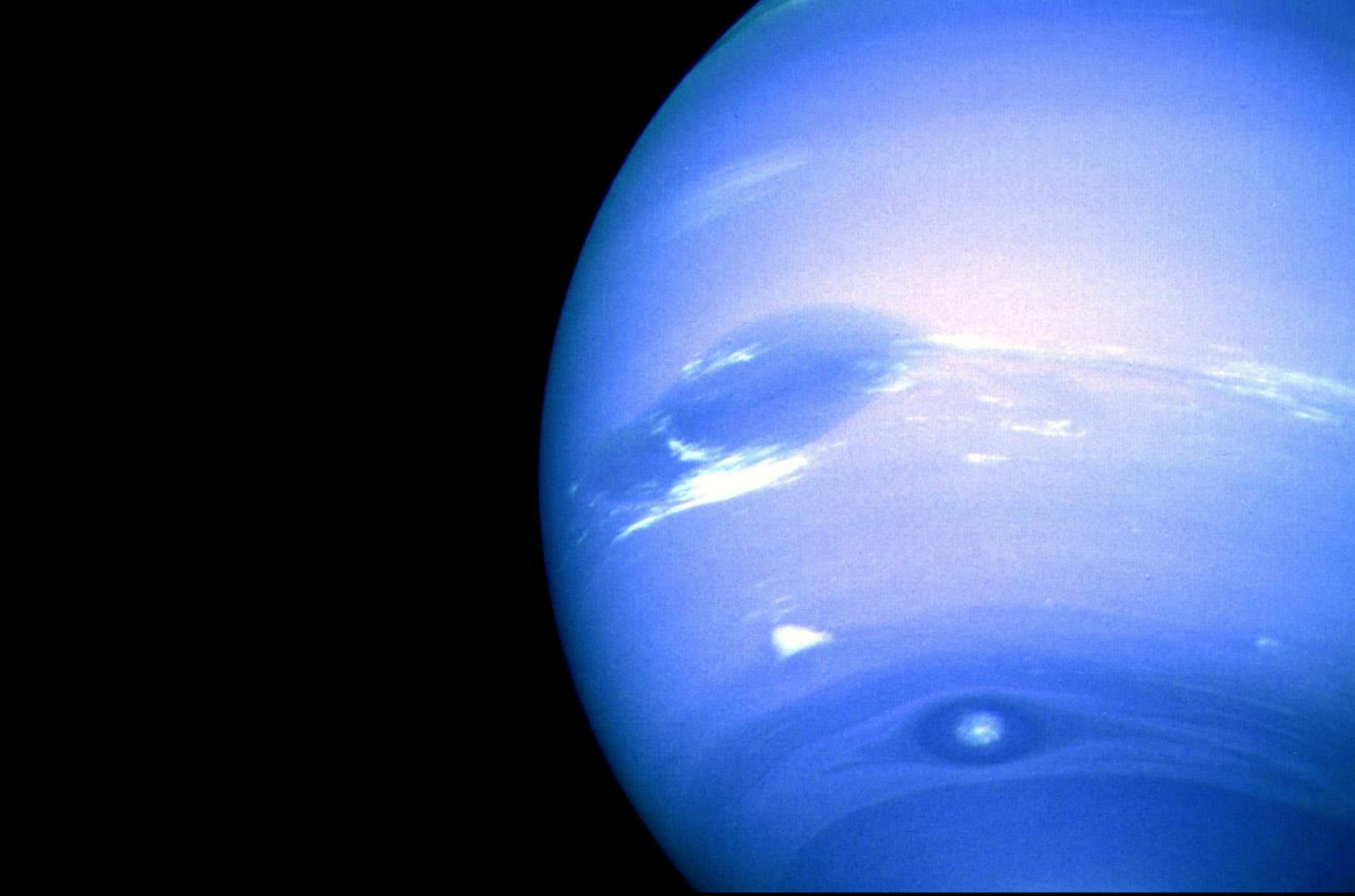
Pasos

Se va a iterar n veces sobre las dos matrices. En cada iteración se considera un nodo como paso intermedio entre dos nodos

1. La fila y columna correspondiente al número de iteración no se toma en cuenta
2. Para el resto de celdas se determina si la distancia actual es mayor que la suma de los pesos utilizando el nodo intermedio
3. Si es menor, se actualiza en la tabla D y en la tabla P se actualiza el nodo de donde proviene
4. Se repite hasta hacerlo con todos los nodos

Lecturas

- Goodrich - Cap. 13
- Shaffer - Cap. 11



Grafos

Mauricio Avilés