# Fundamentos de Teoría de Grafos

# Definición

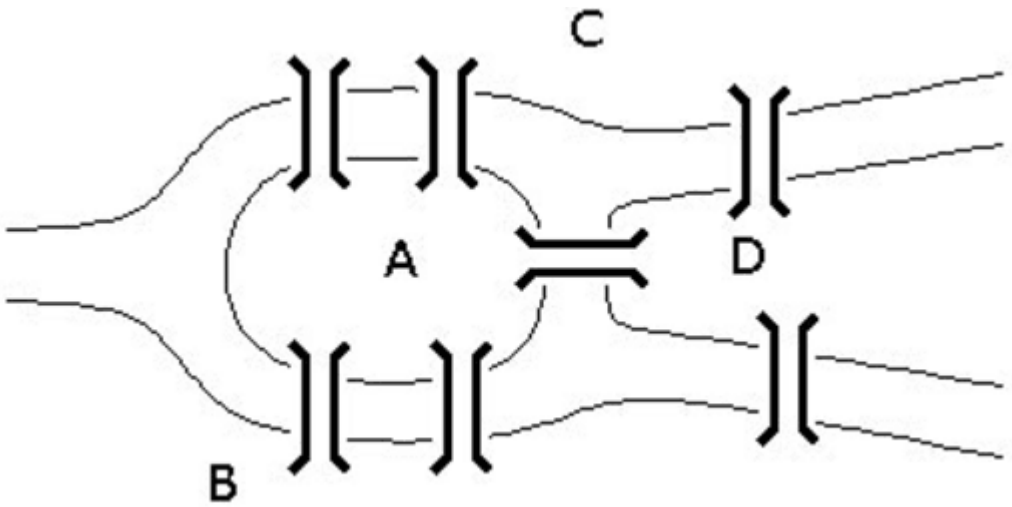
- Un GRAFO es un CONJUNTO de objetos llamados VÉRTICES o NODOS, que están unidos por ARISTAS O ARCOS.

- Su estudio se llama TEORÍA DE GRAFOS.
  - Es una rama de las ciencias de la computación y de las matemáticas.

- Un grafo se representa G = (V, E) donde V es un conjunto NO VACÍO de vértices y E es un conjunto de aristas.

# Historia de la Teoría de Grafos

- La primera vez que se habló de un grafo fue en 1736 gracias a Leonhard Euler, donde él trataba de resolver el problema de los puentes de Königsberg.

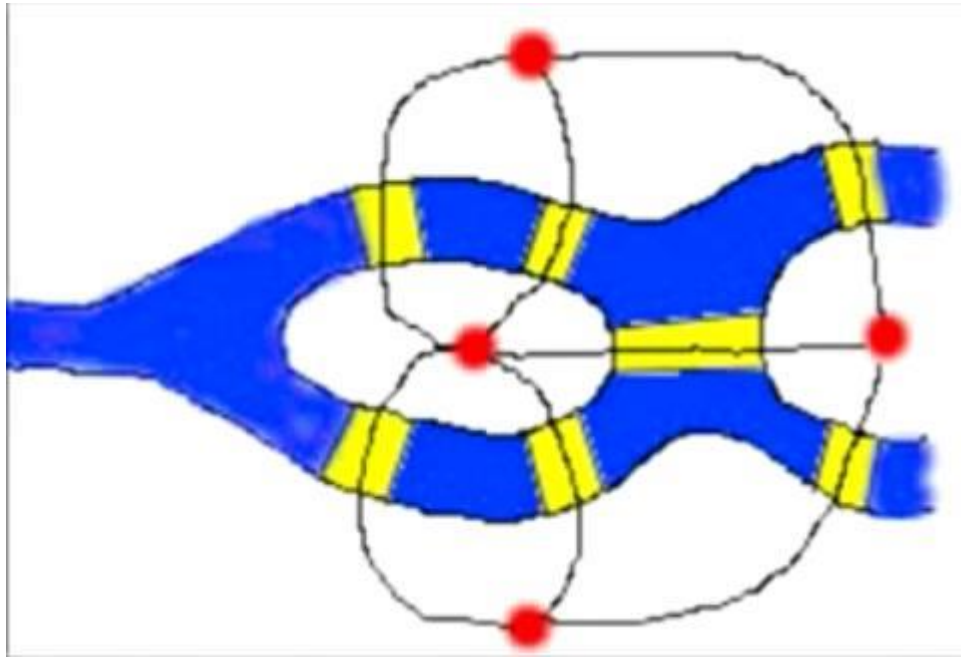



Königsberg es una ciudad hoy situada en Rusia, pero en su momento dentro del imperio prusiano.

Estaba atravesada por un río y en una área con un islote, tiene 7 Puentes.

El problema es: como pasar por todos los puentes una sola vez (sin hacer trampa usando algún mecanismo para pasar de territorio sin usar puentes)

# Historia de la Teoria de Grafos



Euler sustituyó cada “tierra” por un vértice,
y convirtió los puentes en “aristas”.

Se dio cuenta que lo importante no era la ruta (secuencia)
sino la cantidad de conexiones.

Y su respuesta fue: ES IMPOSIBLE.

Y no solo eso, realmente resolvió el problema de qué
características debía tener una estructura así para
poder recorrerla pasando por todos las rutas solo una vez.

Las reglas de Euler fueron:
- Si hay más de 2 regiones a las que conducen un número impar de puentes, es imposible.
- Si solo hay 2 regiones a las que llega un número impar de puentes, es posible comenzando de una de ellas.
- Si no hay regiones a las que conduzcan un número impar de puentes, se puede realizar desde cualquier punto

# Conceptos

- Un Grafo G es un par (V, E) donde V es un conjunto finito y E es una relación sobre elementos de V.

- V es el conjunto de vértices o nodos.

- E es el conjunto de Aristas o Rutas.

- Una arista es un conjunto {u, v} tal que u, v PERTENECEN a V y u != v
  Normalmente nos referiremos a un nodo como (u, v)

# Conceptos

- Se dice que un grafo es NO DIRIGIDO es aquel donde (v, u) = (u, v). Es decir, la relación funciona en ambos sentidos.

- En un grafo dirigido tener (v, u) NO implica que se pueda relacionar (u, v).

- Si existe (u, v) decimos que v es ADJACENTE a u.
  - Cuando el grafo es NO dirigido, la adyacencia es SIMÉTRICA.

# Conceptos

- Grado de un nodo:
  - En un grafo no dirigido, es el número de aristas relacionadas a él.
  - En un grafo dirigido:
    - Grado de Salida: cantidad de aristas que salen de él
    - Grado de Entrada: cantidad de aristas que llegan a él
    - Grado del Nodo: es el número de entradas MENOS el número de salidas.

# Conceptos

- Ruta o Camino (Path): dados 2 nodos u y v, una ruta es un conjunto no vacío de aristas que me llevan a u a v

- Longitud/Tamaño de una ruta: dados los nodos u y v, es la cantidad de aristas por las que tengo que pasar para llegar de u a v.
  - La longitud de u a u se considera 0.

- Se dice que un nodo v es ALCANZABLE desde u si existe una ruta que permita llegar de u a v

# Conceptos

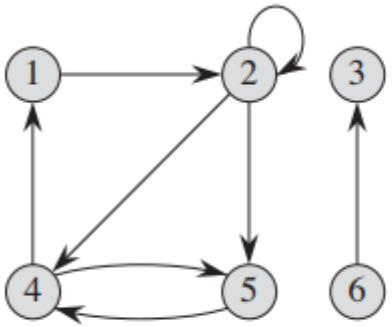
- Una Ruta se compone de TODOS los nodos que visito para llegar del nodo u al nodo v

- Una ruta es SIMPLE si todos los vértices en la ruta son diferentes

- Un grafo NO DIRECCIONADO se dice que es conectado si cualquier vértice es alcanzable desde cualquier otro.
  - Puede haber vértices aislados.

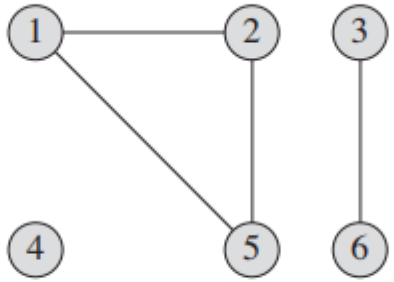
- Un grafo DIRIGIDO es FUERTEMENTE CONECTADO si para 2 vértices cualesquiera estos son mutuamente alcanzables

# Conceptos



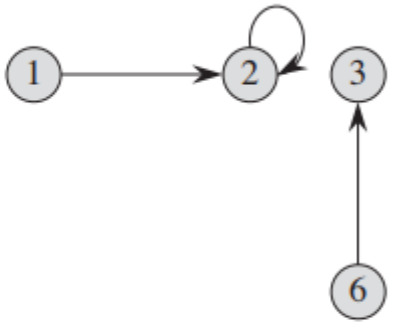
(a)

Grafo Dirigido



(b)

Grafo no Dirigido



(c)

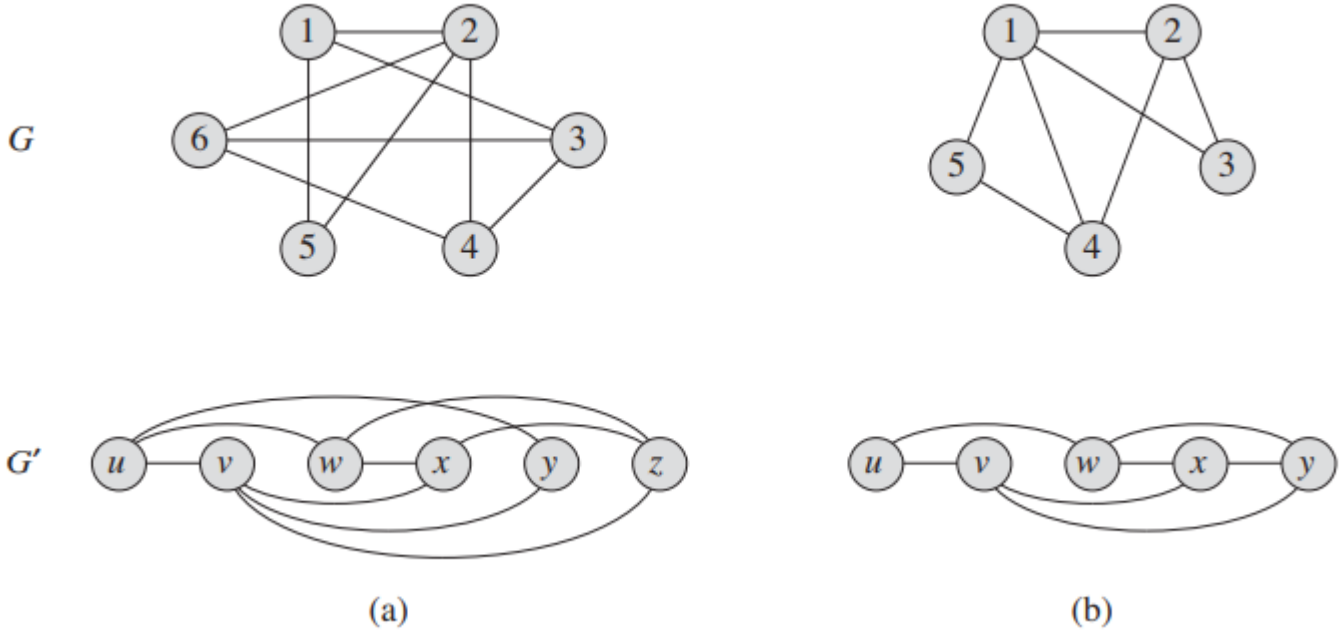
Sub grafo de A

# Conceptos

- Un Subgrafo es un conjuntro G(V’,E’) tal que existe otro grafo G(V,E) y se cumple que:
  - V’ es subconjunto propio de V
  - E’ es subconjunto propio de E

- Dos gráficos son isomórficos si se pueden mapear directamente los vértices de los 2 grafos y sus caminos y la tabla de adyacencia es igual.

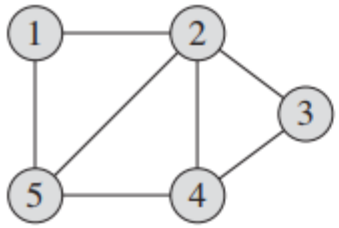
# Grafos isomórficos

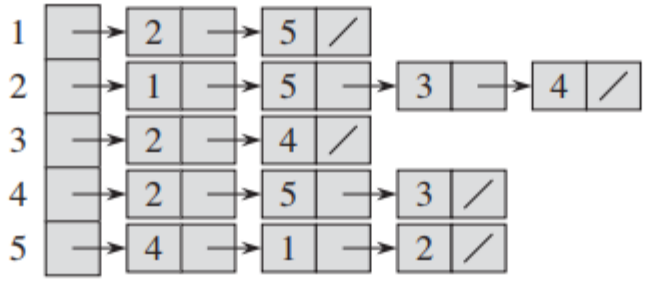



SI SON ISOMÓRFICOS

NO SON ISOMÓRFICOS

# Representación



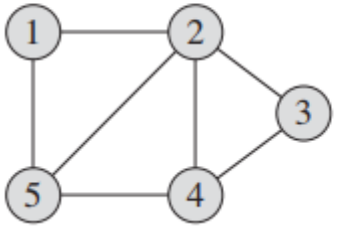

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | 0 | 1 | 0 | 0 | 1 |
| 2 | 1 | 0 | 1 | 1 | 1 |
| 3 | 0 | 1 | 0 | 1 | 0 |
| 4 | 0 | 1 | 1 | 0 | 1 |
| 5 | 1 | 1 | 0 | 1 | 0 |

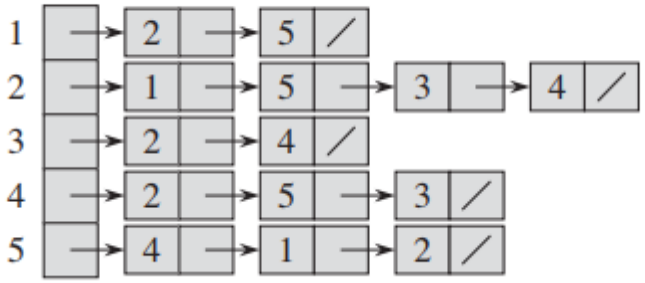
En este caso tenemos un grafo NO direccionado, representado como un arreglo de listas o como una matriz.

En el caso de la matriz, se podría sustituir el 1 por la distancia entre los nodos (o cualquier valor que sea representativo de la semántica de la arista

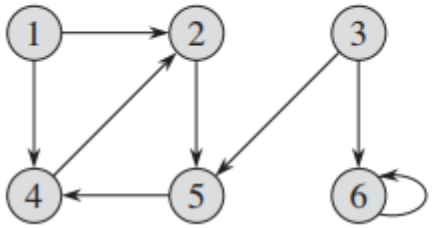
La matriz se le llama MATRIZ DE ADYACENCIA

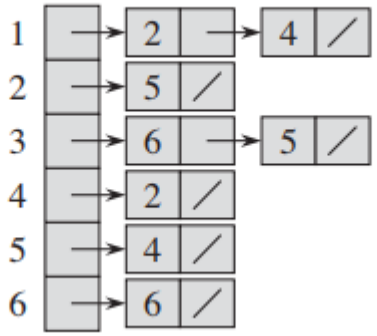
# Representación




| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | 0 | 1 | 0 | 0 | 1 |
| 2 | 1 | 0 | 1 | 1 | 1 |
| 3 | 0 | 1 | 0 | 1 | 0 |
| 4 | 0 | 1 | 1 | 0 | 1 |
| 5 | 1 | 1 | 0 | 1 | 0 |




| | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| 1 | 0 | 1 | 0 | 1 | 0 | 0 |
| 2 | 0 | 0 | 0 | 0 | 1 | 0 |
| 3 | 0 | 0 | 0 | 0 | 1 | 1 |
| 4 | 0 | 1 | 0 | 0 | 0 | 0 |
| 5 | 0 | 0 | 0 | 1 | 0 | 0 |
| 6 | 0 | 0 | 0 | 0 | 0 | 1 |

Comparemos un
grafo no dirigido contra uno
dirigido

# Formas de recorrer un grafo

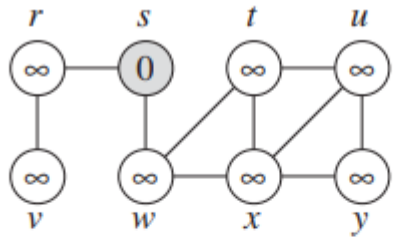
- Existen muchos problemas que se pueden resolver con grafos: pero en general existes 2 formas de recorrerlos:
  - Por amplitud: dado un nodo de inicio, iremos recorriéndolo por la longitud de las rutas al mismo tiempo
  - Por profundidad: se irá tan profundo como se pueda en cada ruta que se encuentre.

# Por amplitud

- Dado un grafo G y una “raíz” (nodo de inicio S)

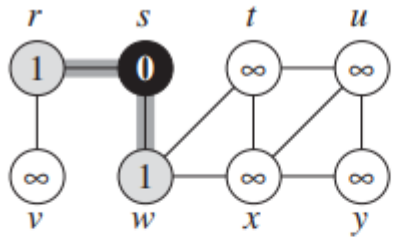
```
BFS(G, s)
 1  for each vertex u ∈ G.V − {s}
 2       u.color = WHITE
 3       u.d = ∞
 4       u.π = NIL
 5  s.color = GRAY
 6  s.d = 0
 7  s.π = NIL
 8  Q = ∅
 9  ENQUEUE(Q, s)
10  while Q ≠ ∅
11       u = DEQUEUE(Q)
12       for each v ∈ G.Adj[u]
13            if v.color == WHITE
14                 v.color = GRAY
15                 v.d = u.d + 1
16                 v.π = u
17                 ENQUEUE(Q, v)
18       u.color = BLACK
```

(a)



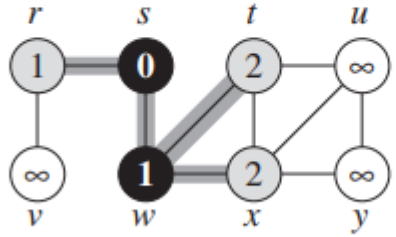

$Q$ | $s$ |
0

(b)



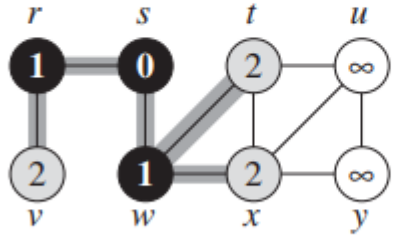

$Q$ | $w$ | $r$ |
1 1

(c)



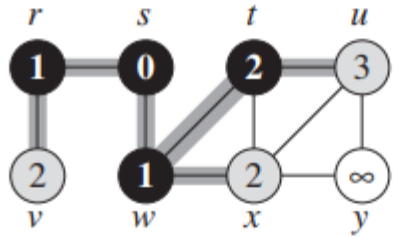

$Q$ | $r$ | $t$ | $x$ |
1 2 2

(d)



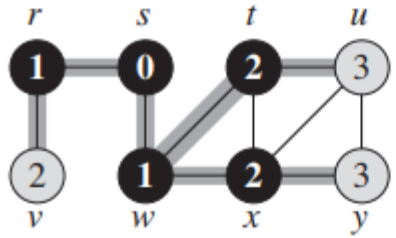

$Q$ | $t$ | $x$ | $v$ |
2 2 2

(e)



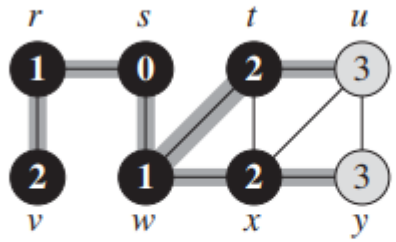

$Q$ | $x$ | $v$ | $u$ |
2 2 3

(f)



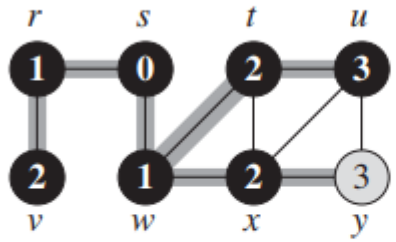

$Q$ | $v$ | $u$ | $y$ |
2 3 3

(g)



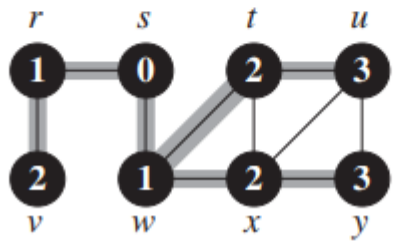

$Q$ | $u$ | $y$ |
3 3

(h)




$Q$ | $y$ |
3

(i)



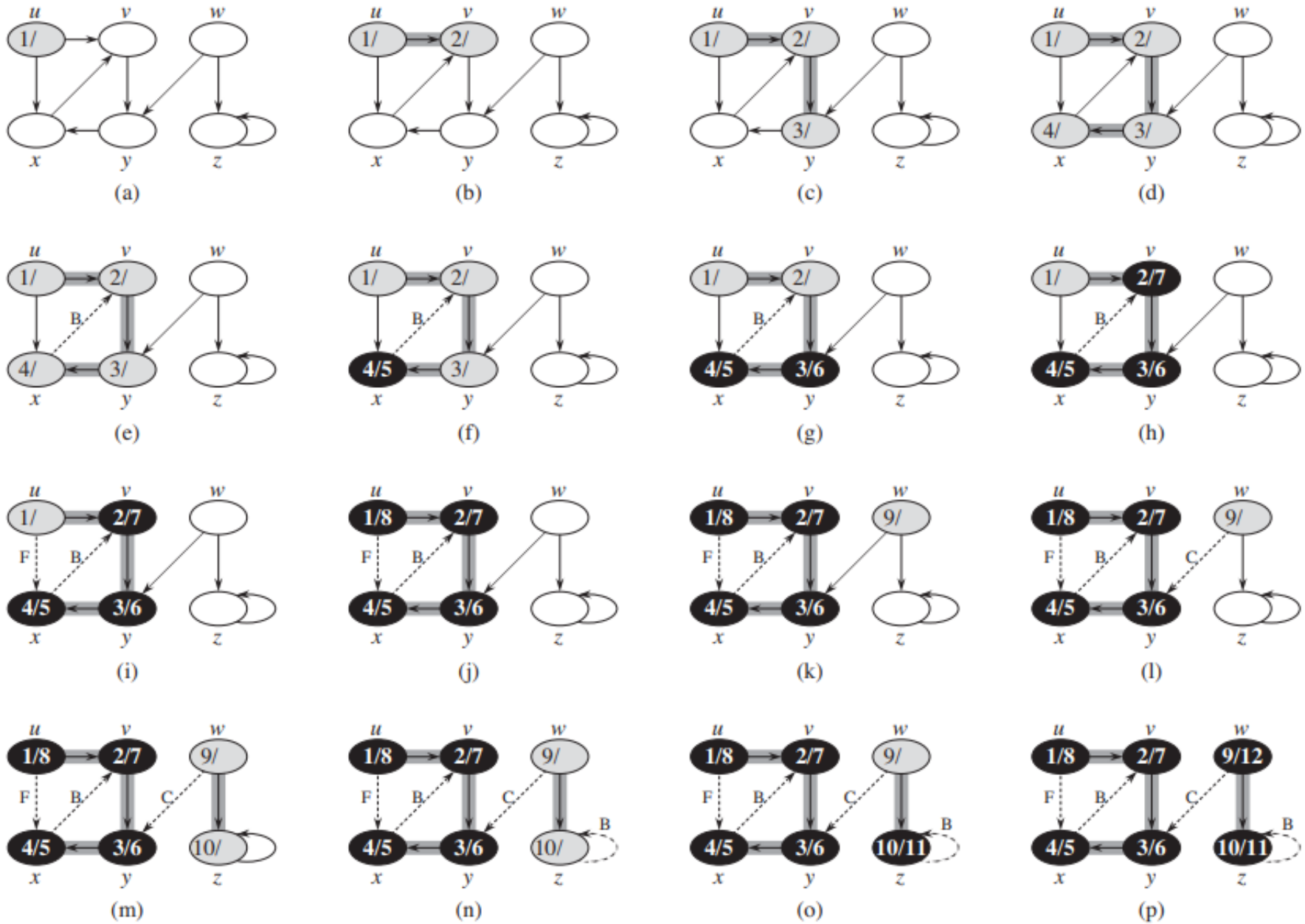

$Q$ $\emptyset$

# Por profundidad

- Dado un grafo G se visita asi

```
DFS(G)
1  for each vertex u ∈ G.V
2      u.color = WHITE
3      u.π = NIL
4  time = 0
5  for each vertex u ∈ G.V
6      if u.color == WHITE
7          DFS-VISIT(G, u)

DFS-VISIT(G, u)
 1  time = time + 1                  // white vertex u has just been discovered
 2  u.d = time
 3  u.color = GRAY
 4  for each v ∈ G.Adj[u]            // explore edge (u, v)
 5      if v.color == WHITE
 6          v.π = u
 7          DFS-VISIT(G, v)
 8  u.color = BLACK                  // blacken u; it is finished
 9  time = time + 1
10  u.f = time
```


u v w
1/
x y z
(a)
u v w
1/ 2/
x y z
(b)
u v w
1/ 2/
3/
x y z
(c)
u v w
1/ 2/
4/ 3/
x y z
(d)
u v w
1/ 2/
B
4/ 3/
x y z
(e)
u v w
1/ 2/
B
4/5 3/
x y z
(f)
u v w
1/ 2/
B
4/5 3/6
x y z
(g)
u v w
1/ 2/7
B
4/5 3/6
x y z
(h)
u v w
1/ 2/7
F B
4/5 3/6
x y z
(i)
u v w
1/8 2/7
F B
4/5 3/6
x y z
(j)
u v w
1/8 2/7 9/
F B
4/5 3/6
x y z
(k)
u v w
1/8 2/7 9/
F B C
4/5 3/6
x y z
(l)
u v w
1/8 2/7 9/
F B C
4/5 3/6 10/
x y z
(m)
u v w
1/8 2/7 9/
F B C
4/5 3/6 10/ B
x y z
(n)
u v w
1/8 2/7 9/
F B C
4/5 3/6 10/11 B
x y z
(o)
u v w
1/8 2/7 9/12
F B C
4/5 3/6 10/11 B
x y z
(p)