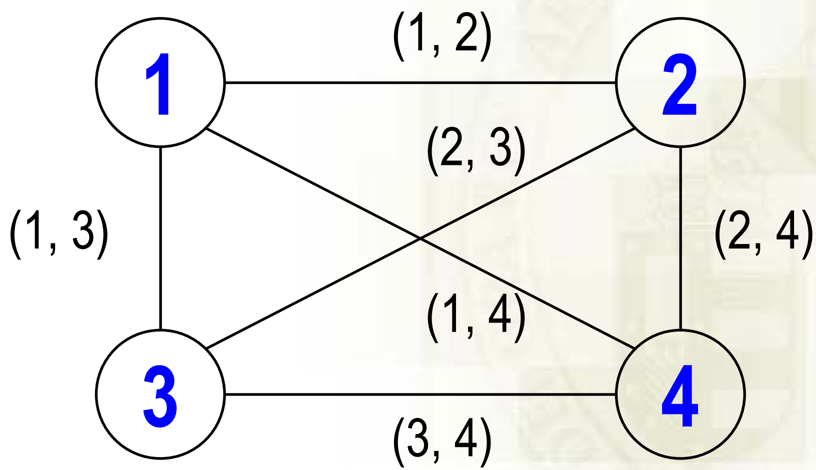
Tema 15: GRAFOS

Ejemplos de grafos

$G_1 = (V_1, A_1)$

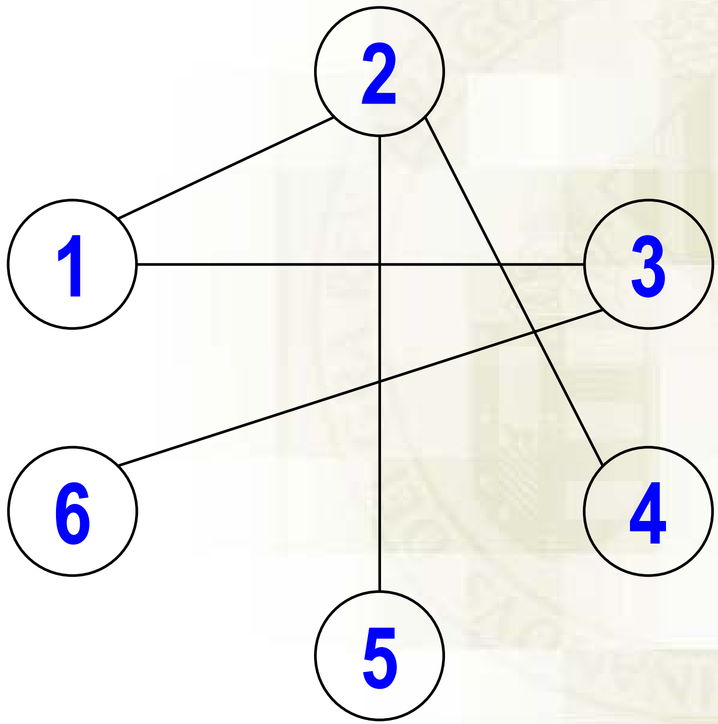
$V_1 = \{1,2,3,4\}$ $A_1 = \{ (1, 2), (1, 3), (1, 4), (2, 3), (2, 4), (3, 4) \}$



Ejemplos de grafos

$G_2 = (V_2, A_2)$

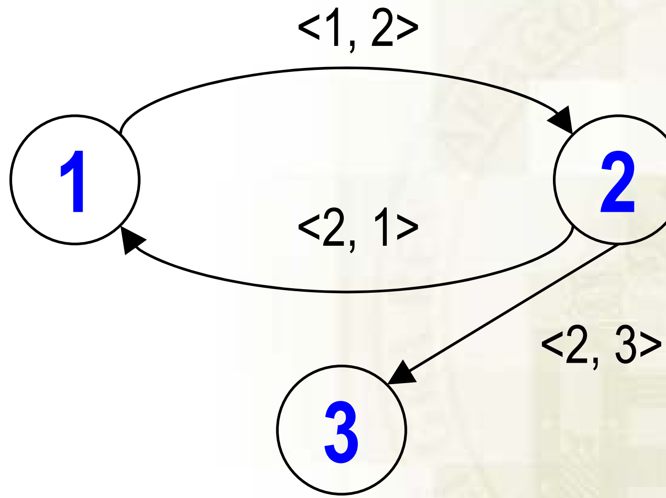
$V_2 = \{1,2,3,4,5,6\}$ $A_2 = \{ (1, 2), (1, 3), (2, 4), (2, 5), (3, 6) \}$



Ejemplos de grafos

$G_3 = (V_3, A_3)$

$V_3 = \{1,2,3\}$ $A_3 = \{ <1, 2>, <2, 1>, <2, 3> \}$



Terminología

- **Orden de un grafo**: Número de nodos (vértices) del grafo.
- **Grado de un nodo**: Número de ejes (arcos) que inciden sobre el nodo.
- **Grafo simétrico**: Grafo dirigido tal que si existe la relación $\langle u, v \rangle$ entonces existe $\langle v, u \rangle$, con $u,v \in V$.
- **Grafo no simétrico**: Grafo que no cumple la propiedad anterior.
- **Grafo reflexivo**: Grafo que cumple que para todo nodo $u \in V$ existe la relación $(u, u) \in A$.
- **Grafo transitivo**: Grafo que cumple que si existen las relaciones (u, v) y $(v, z) \in A$ entonces $(u,z) \in A$.
- **Grafo completo**: Grafo que contiene todos los posibles pares de relaciones, es decir, para cualquier par de nodos $u,v \in V$, $u \in V$, existe $(u,v) \in A$.

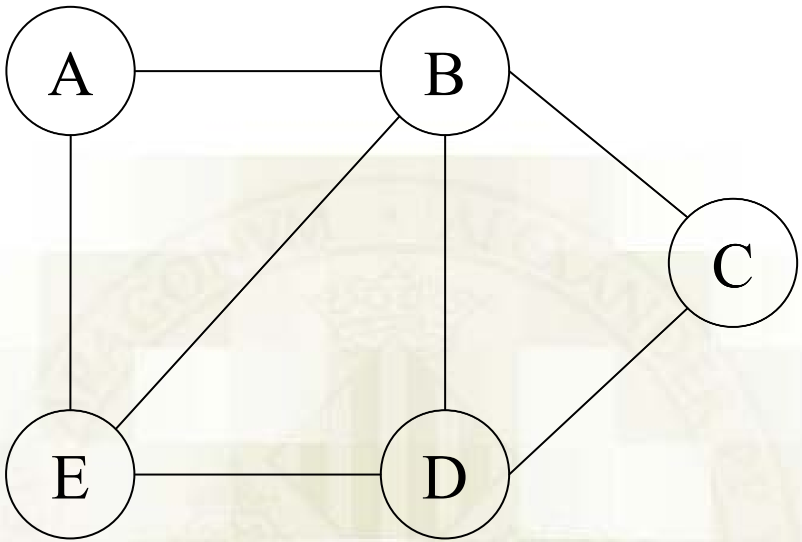
Terminología (2)

- **Camino**: Un camino en el grafo G es una sucesión de vértices y arcos: $v_0, a_1, v_1, a_2, v_2, ..., a_k, v_k$; tal que los extremos del arco a_i son los vértices v_{i-1} y v_i.
- **Longitud de un camino**: Es el número de arcos que componen el camino.
- **Camino cerrado (circuito)**: Camino en el que coinciden los vértices extremos ($v_0 = v_k$).
- **Camino simple**: Camino donde sus vértices son distintos dos a dos, salvo a lo sumo los extremos v_0 y v_k.
- **Camino elemental**: Camino donde sus arcos son distintos dos a dos.
- **Camino euleriano**: Camino simple que contiene todos los arcos del grafo.
- **Grafo euleriano**: Es un grafo que tiene un camino euleriano cerrado.

Terminología (3)

- **Grafo conexo**: Grafo no dirigido tal que para cualquier par de nodos existe al menos un camino que los une.
- **Grafo fuertemente conexo**: Grafo dirigido tal que para cualquier par de nodos existe un camino que los une.
- **Punto de articulación**: Nodo que si desaparece provoca que se cree un grafo no conexo.

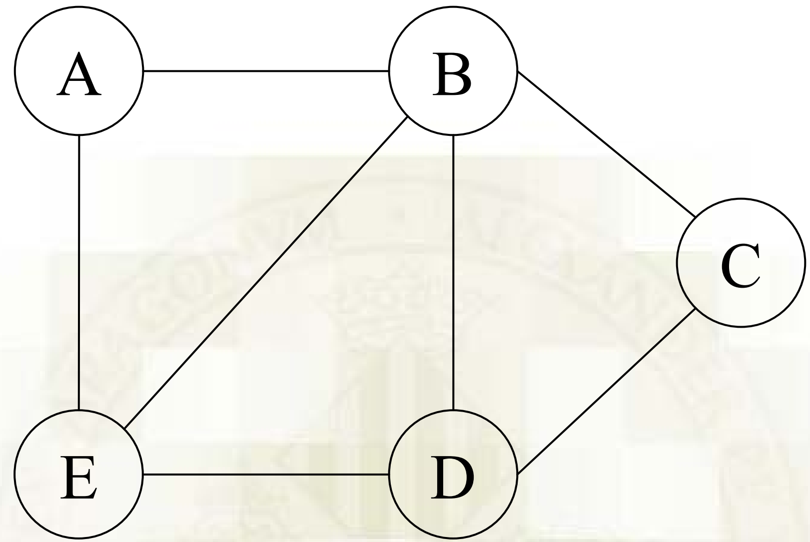
Representación de grafos (1)



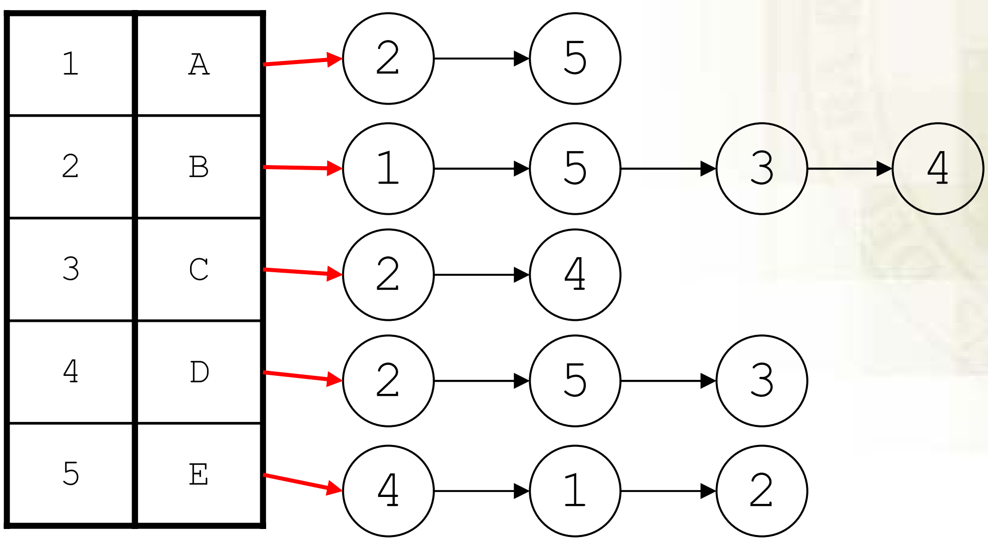
Matriz de Adyacencia

	A	B	C	D	E
A	0	1	0	0	1
B	1	0	1	1	1
C	0	1	0	1	0
D	0	1	1	0	1
E	1	1	0	1	0

Representación de grafos (2)

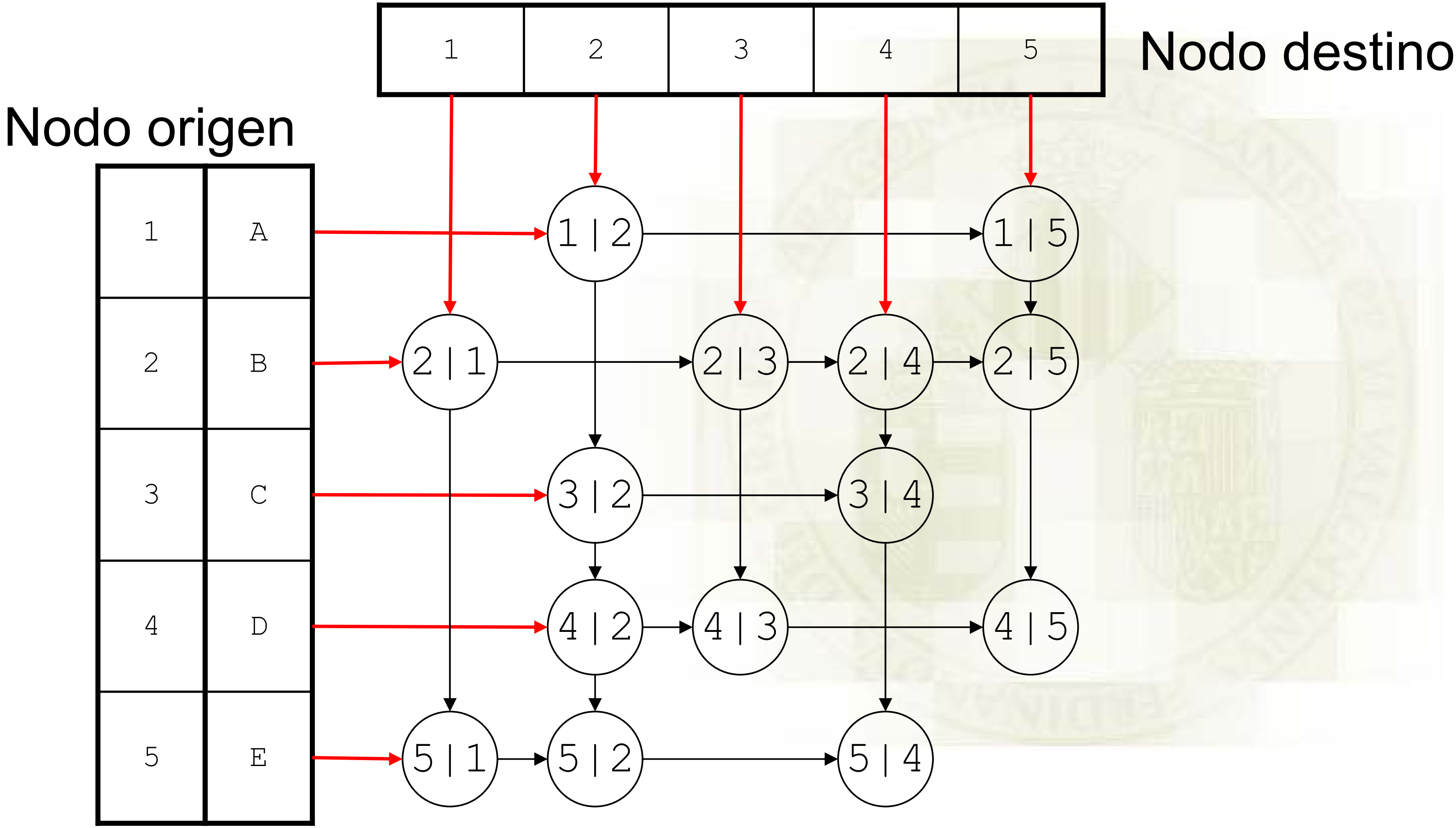


Lista de Adyacencia



Representación de grafos (3)

Matrices Dispersas



Exploración (recorrido) de grafos

- Es conveniente evitar los ciclos:
  - ✓ Marcar los nodos que se visitan en la exploración para no repetir los mismos caminos.

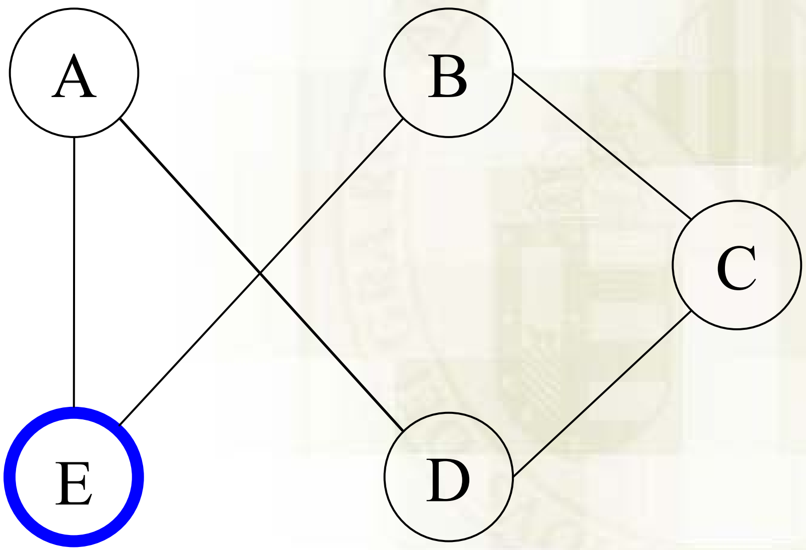
ident	info	visitado
1	A	t/f
2	B	t/f
3	C	t/f
4	D	t/f
5	E	t/f

- Como no hay un nodo cabeza (primero o raíz), es preciso fijar un origen para el recorrido.

Exploración en anchura (BFS)

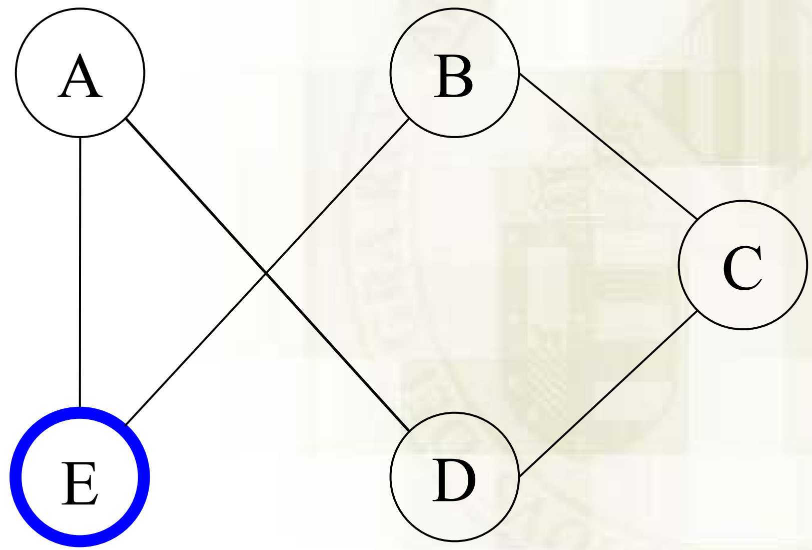
- Breadth First Search (BFS): A partir del nodo origen, recorrer por niveles de distancia a ese nodo.

1. Fijar nodo origen del recorrido (arbitrario?)



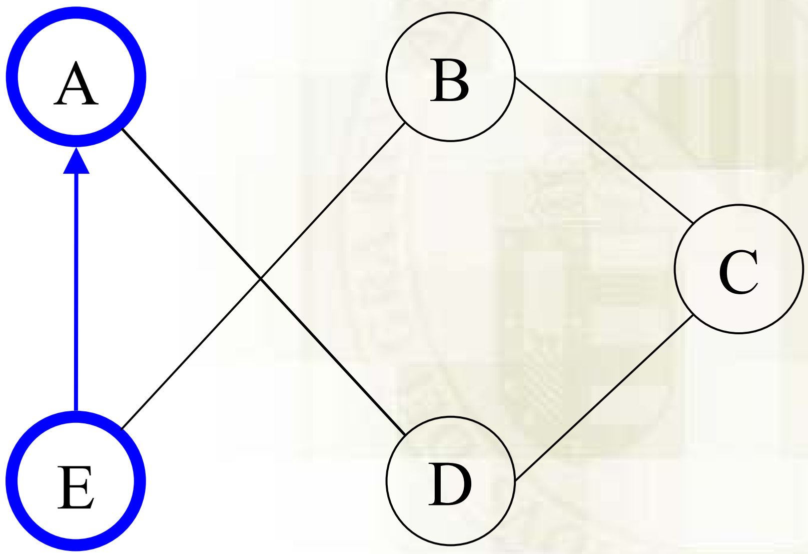
Exploración en anchura (BFS)

2. Acceder a todos los nodos que están a distancia 1, es decir, directamente relacionados con el origen (existe un arco que los une).



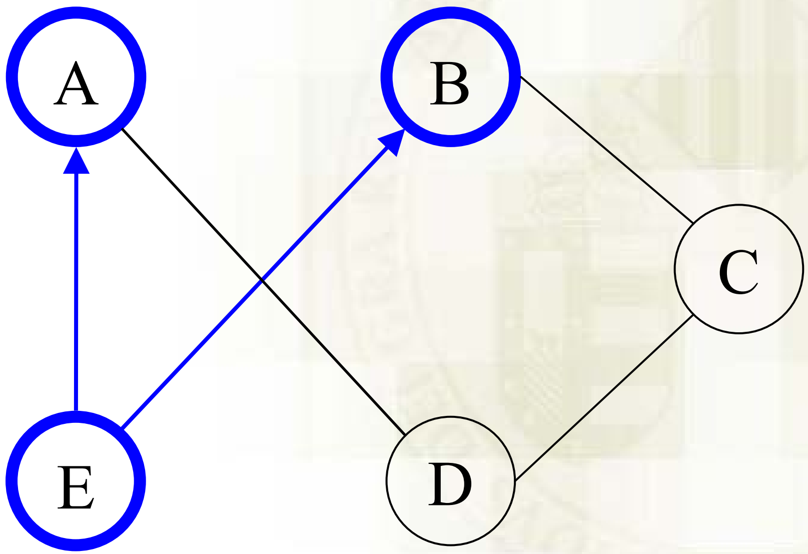
Exploración en anchura (BFS)

2. Acceder a todos los nodos que están a distancia 1, es decir, directamente relacionados con el origen (existe un arco que los une).



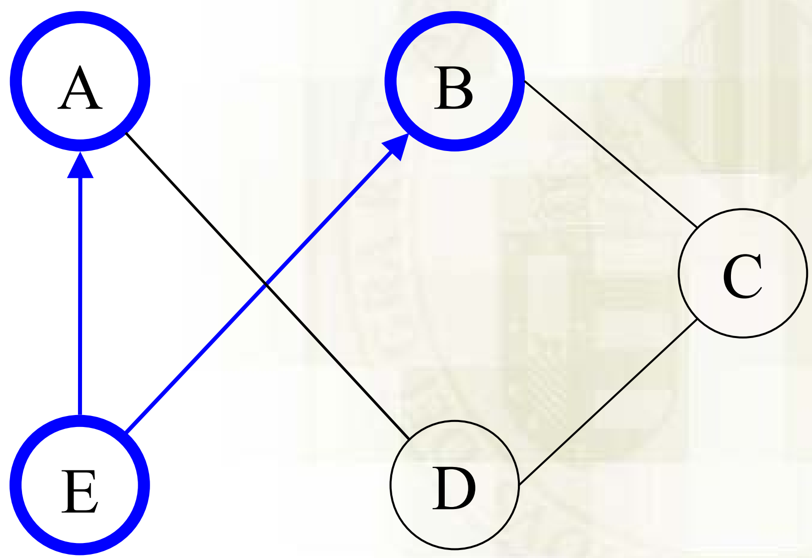
Exploración en anchura (BFS)

2. Acceder a todos los nodos que están a distancia 1, es decir, directamente relacionados con el origen (existe un arco que los une).



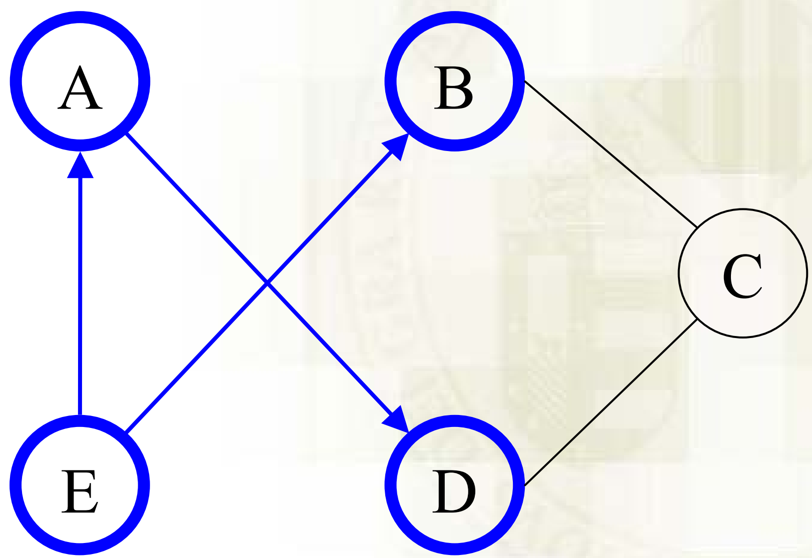
Exploración en anchura (BFS)

3. Acceder a todos los nodos que están a distancia 2, es decir, directamente relacionados con los que está a distancia 1.



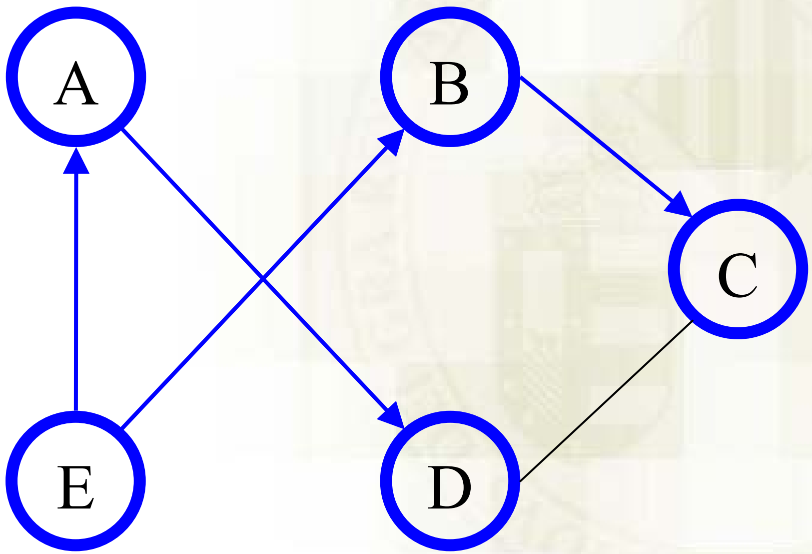
Exploración en anchura (BFS)

3. Acceder a todos los nodos que están a distancia 2, es decir, directamente relacionados con los que está a distancia 1.



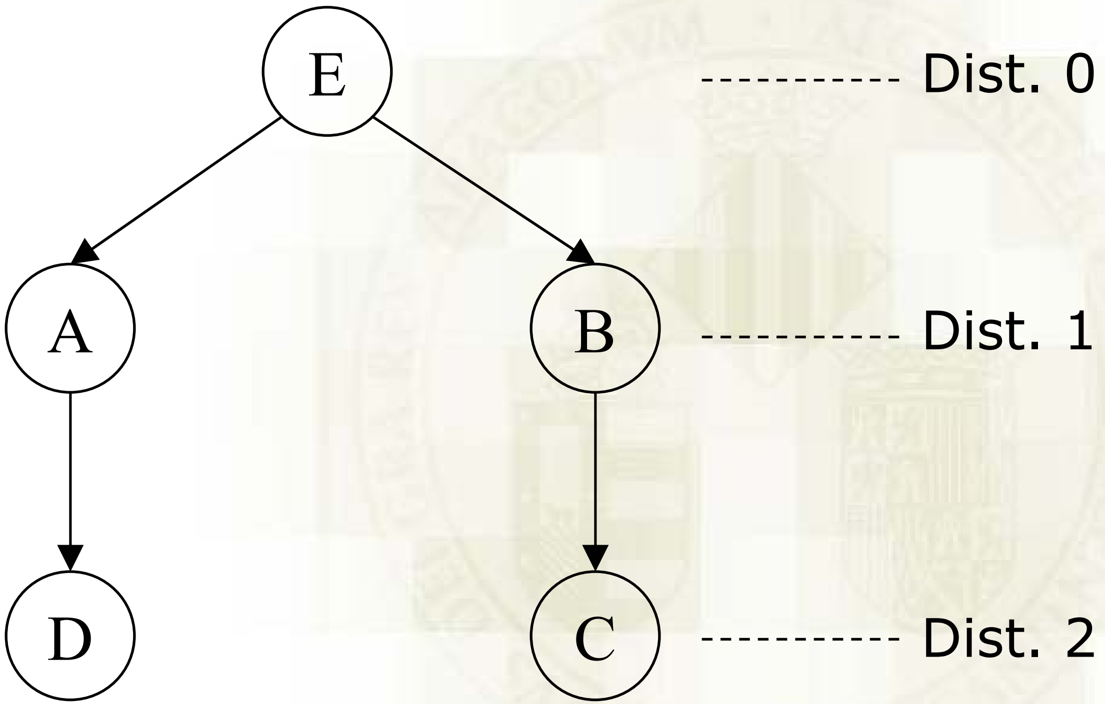
Exploración en anchura (BFS)

3. Acceder a todos los nodos que están a distancia 2, es decir, directamente relacionados con los que está a distancia 1.



Exploración en anchura (BFS)

- El resultado del recorrido es una árbol, que incluye los nodos visitados y los arcos utilizados para acceder a ellos.



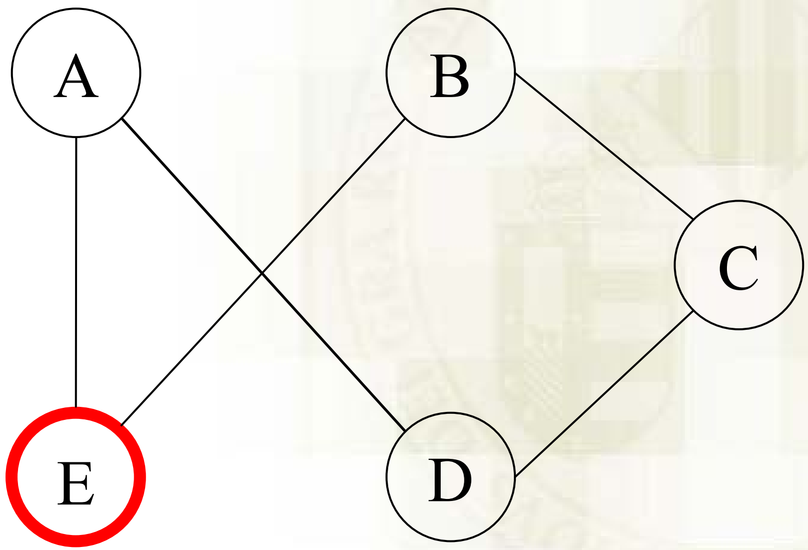
Algoritmo BFS

```
Entradas                    Variables
    G: Grafo                    Q: Cola de índices
    origen: indice              nod1, nod2: indice
---------------------------------------------------------------
Inicio
    Procesar (origen); //dist (origen) = 0
    Marcar como visitado (origen);
    Q.Encolar(origen);

    Mientras (no Q.ColaVacia()) hacer:
        nod1 = Q.Cima();
        Q.Desencolar();
        Para todo nod2 adyacente a nod1 hacer:
            Si (no visitado (nod2)) entonces
                Procesar (nod2); //dist (nod2) = dist (nod1) +1
                Marcar como visitado (nod2);
                Q.Encolar(nod2);
            fin_si
        fin_para
    fin_Mientras
Fin
```

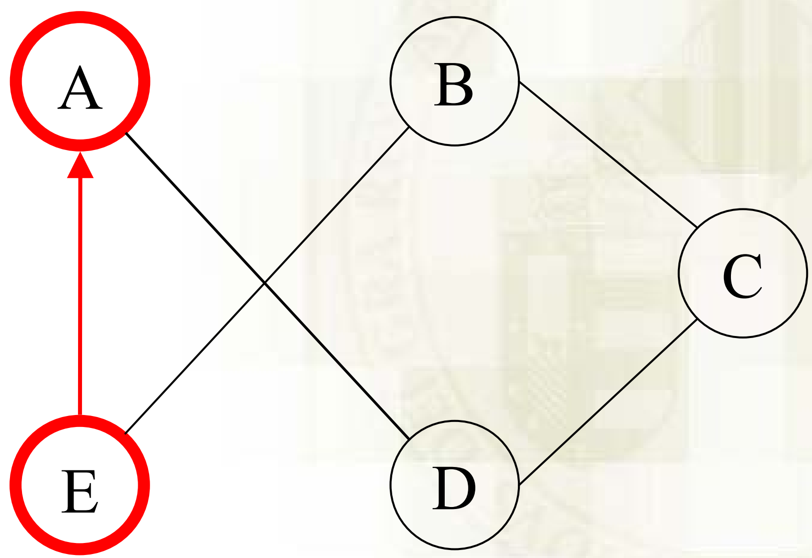
Exploración en profundidad (DFS)

- Depth First Search (DFS): A partir del nodo origen, avanzar a otro nodo no visitado y seguir el recorrido, de la misma manera, a partir de él.



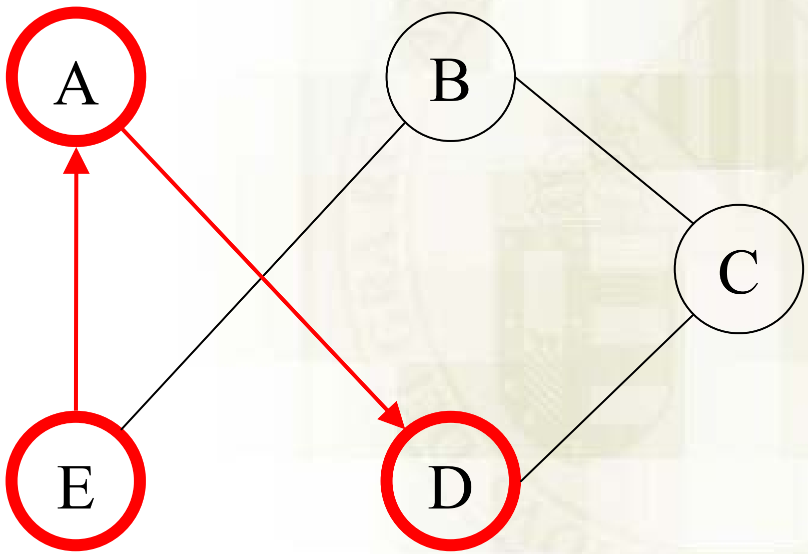
Exploración en profundidad (DFS)

- Depth First Search (DFS): A partir del nodo origen, avanzar a otro nodo no visitado y seguir el recorrido, de la misma manera, a partir de él.



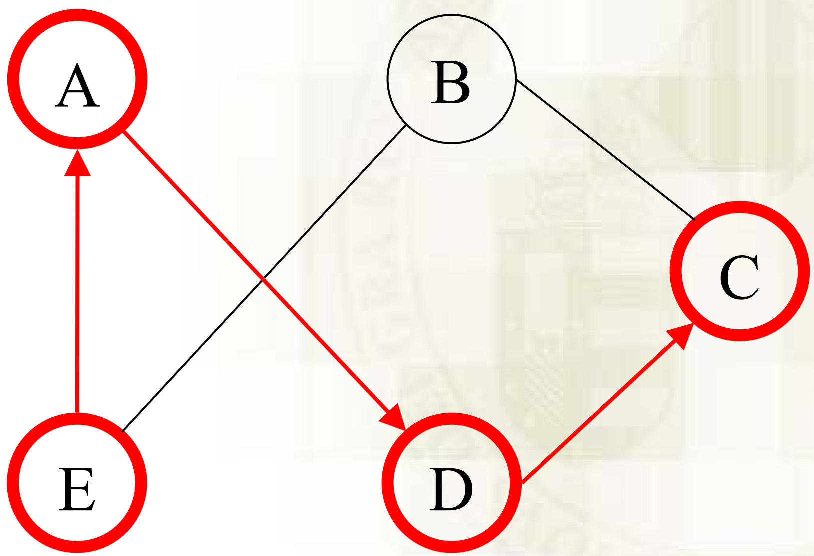
Exploración en profundidad (DFS)

- Depth First Search (DFS): A partir del nodo origen, avanzar a otro nodo no visitado y seguir el recorrido, de la misma manera, a partir de él.



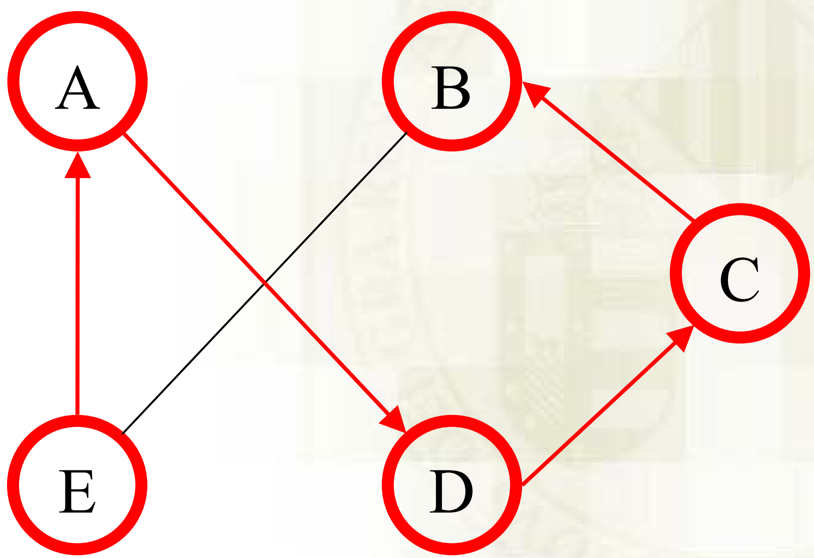
Exploración en profundidad (DFS)

- Depth First Search (DFS): A partir del nodo origen, avanzar a otro nodo no visitado y seguir el recorrido, de la misma manera, a partir de él.



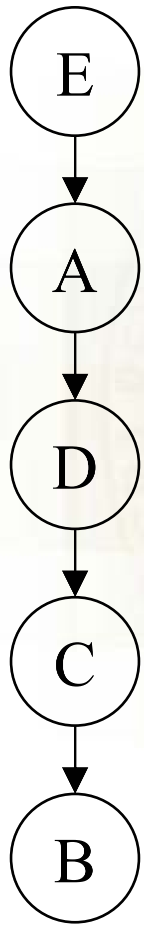
Exploración en profundidad (DFS)

- Depth First Search (DFS): A partir del nodo origen, avanzar a otro nodo no visitado y seguir el recorrido, de la misma manera, a partir de él.



Exploración en anchura (DFS)

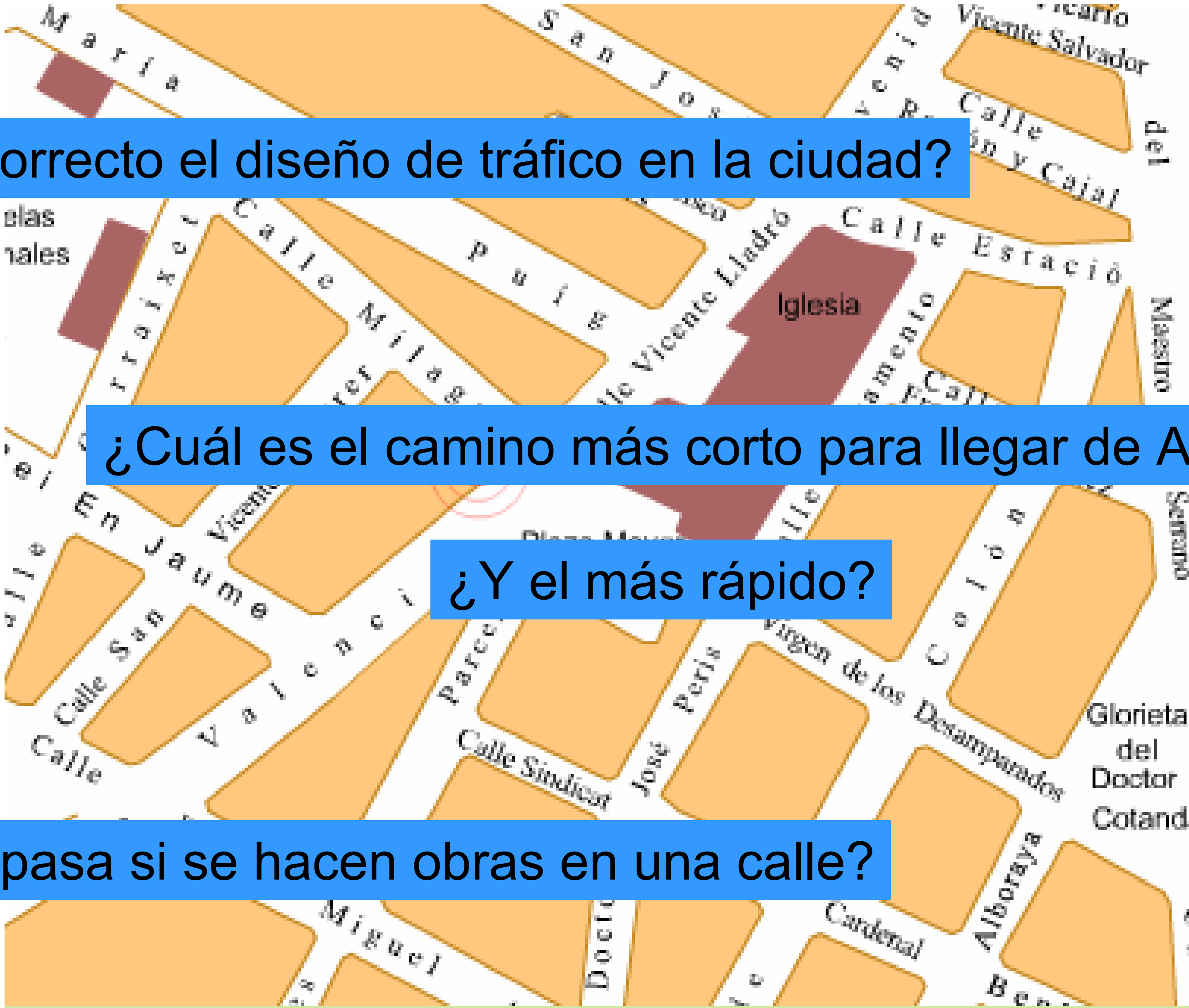
- El resultado del recorrido es una árbol, que incluye los nodos visitados y los arcos utilizados para acceder a ellos.



Algoritmo DFS

```
Entradas                      Variables
    G: Grafo                      nodo: indice
    origen: indice
-----------------------------------------------------
Inicio
    Procesar (origen);
    Marcar como visitado (origen);

    Para todo nodo adyacente a origen hacer:
        Si (no visitado (nodo)) entonces
            G.DFS ( nodo )
        fin_si
    fin_para
Fin
```

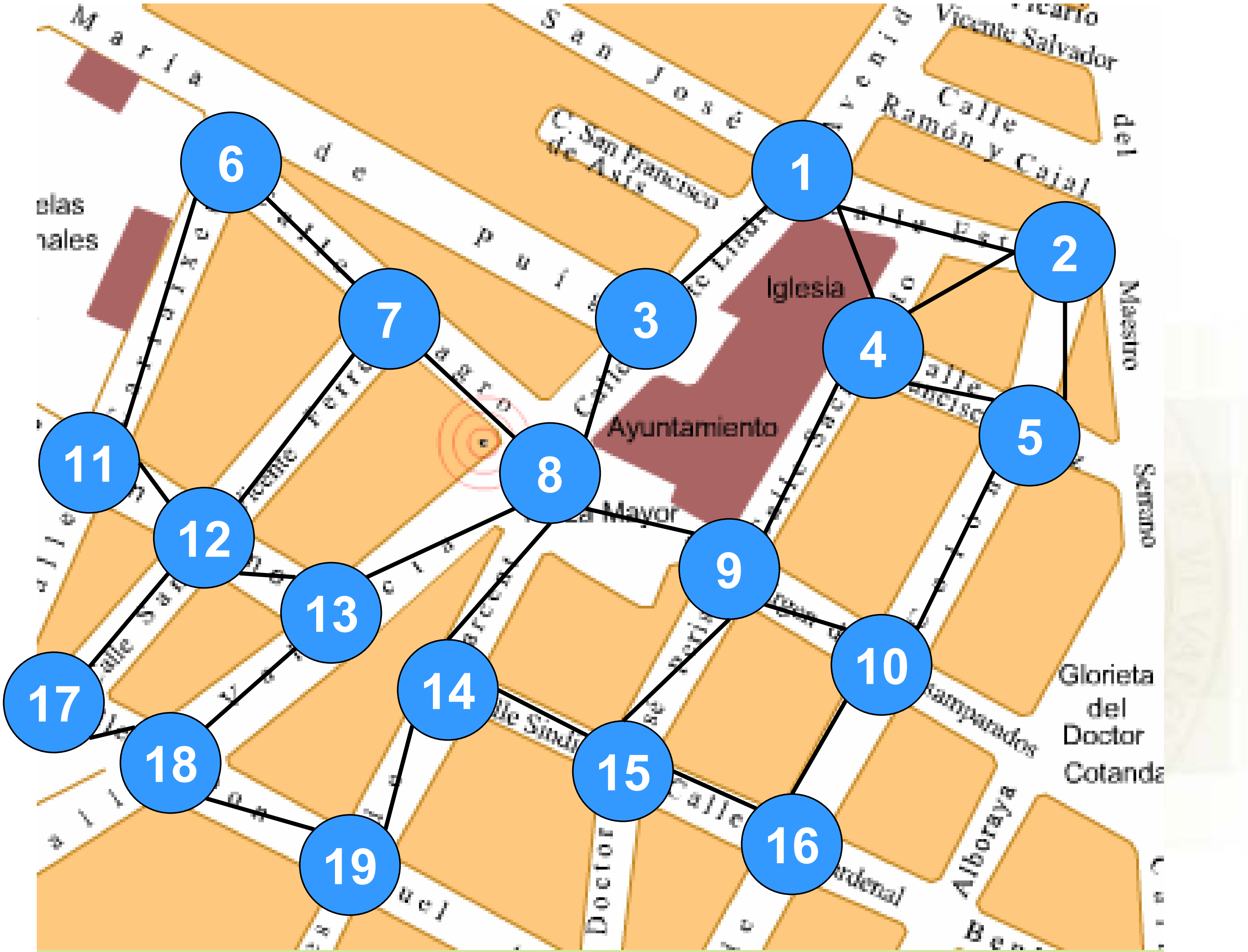


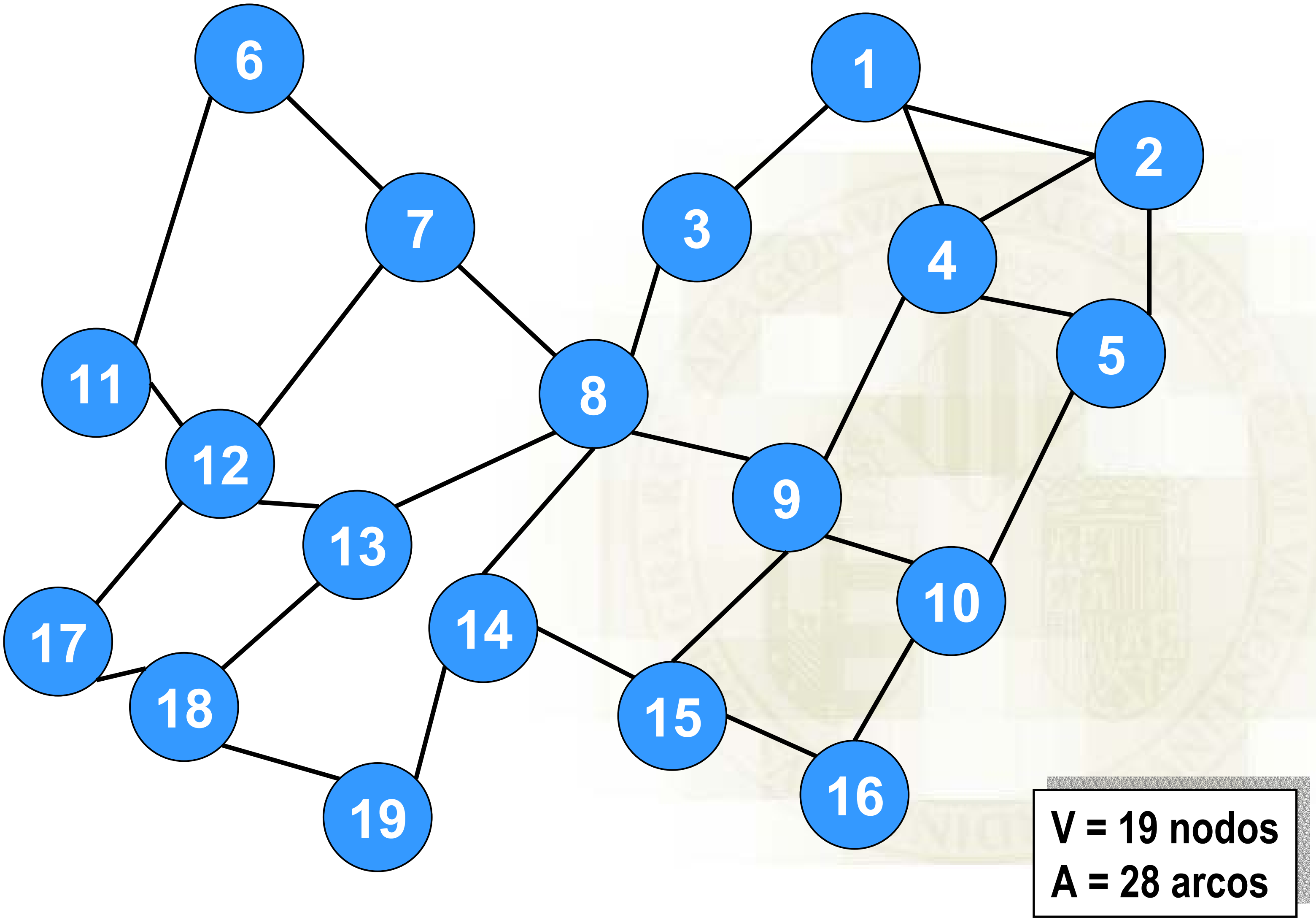
¿Es correcto el diseño de tráfico en la ciudad?

¿Cuál es el camino más corto para llegar de A a B?

¿Y el más rápido?

¿Qué pasa si se hacen obras en una calle?


María de Puig
San José
C. San Francisco de Asís
Vicente Salvador
Calle Ramón y Cajal
Iglesia
Ayuntamiento
Maestro Serrano
Glorieta del Doctor Cotanda
Alboraya
Calle Cardenal
Doctor
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19


6
7
1
2
3
4
5
11
8
12
9
13
10
17
14
18
15
16
19
V = 19 nodos
A = 28 arcos