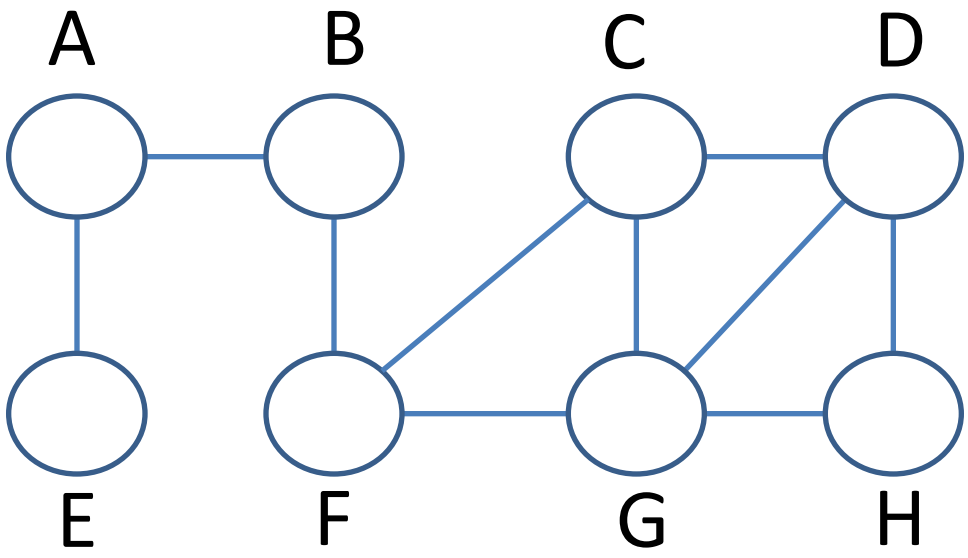
BREADTH-FIRST SEARCH (G, s)

```
1   for each vertex u ∈ V [G]-s
2           do color[u] ← WHITE
3              distance[u] ← ∞
4              predecessor[u] ← NIL
5   color[s] ← GRAY
6   distance[s] ← 0
7   predecessor[s] ← NIL
8   Q ← ∅
9   ENQUEUE (Q, s)
10  while Q ≠ ∅
11          do u ← DEQUEUE (Q)
12             for each v ∈ Adj[u]
13                 do if color [v] = WHITE
14                        then color [v] ← GRAY
15                             distance[v] ← distance[u] + 1
16                             predecessor[v] ← u
17                             ENQUEUE (Q, v)
18             color[u] ← BLACK
```

BREADTH-FIRST SEARCH

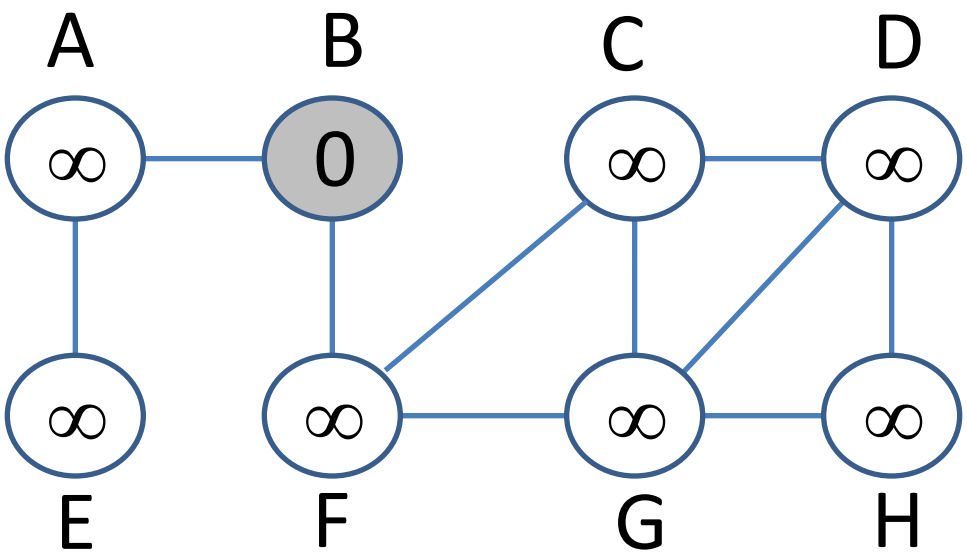
Source s = B



BREADTH-FIRST SEARCH

Source s = B

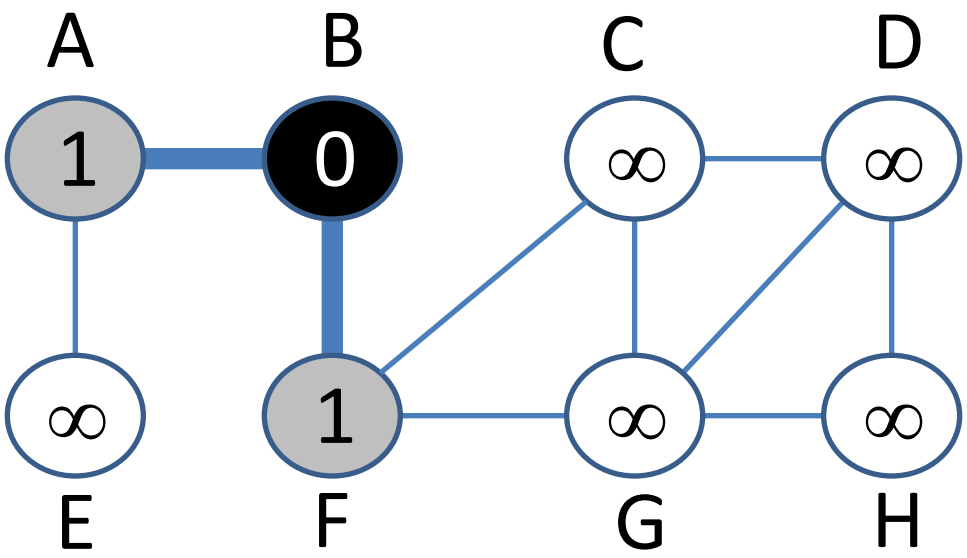
Breadth-first tree

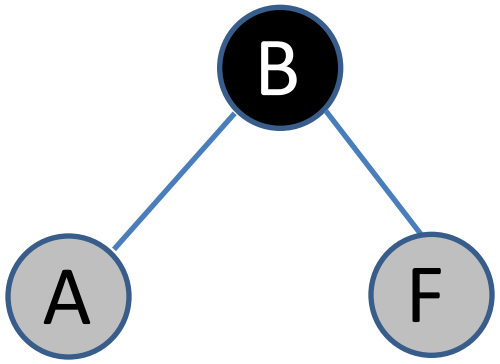


Q = {B_0}

BREADTH-FIRST SEARCH

$Q = \{B_0\}$ **expand B** **Breadth-first tree**

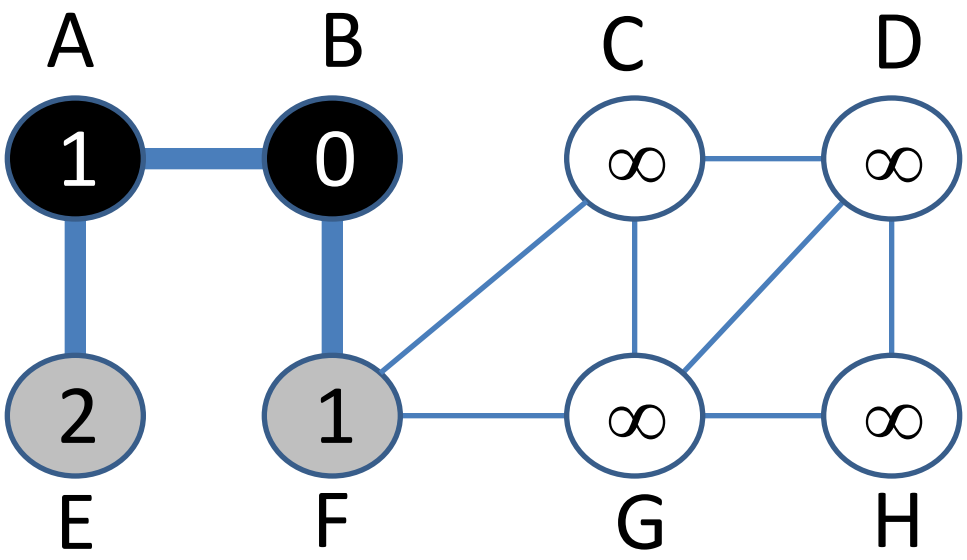


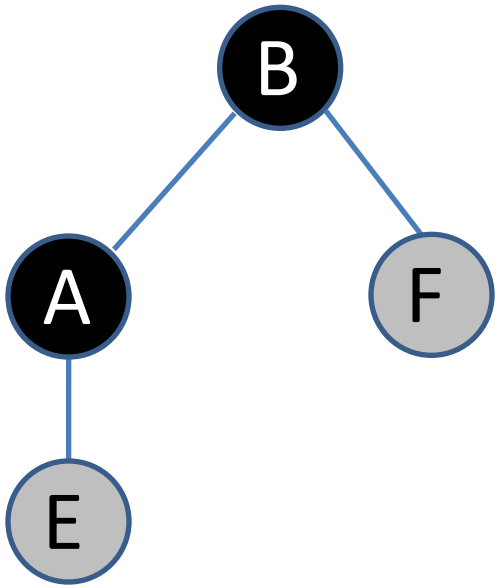


$Q = \{A_1, F_1\}$

BREADTH-FIRST SEARCH

$Q = \{A_1, F_1\}$ **expand A** **Breadth-first tree**

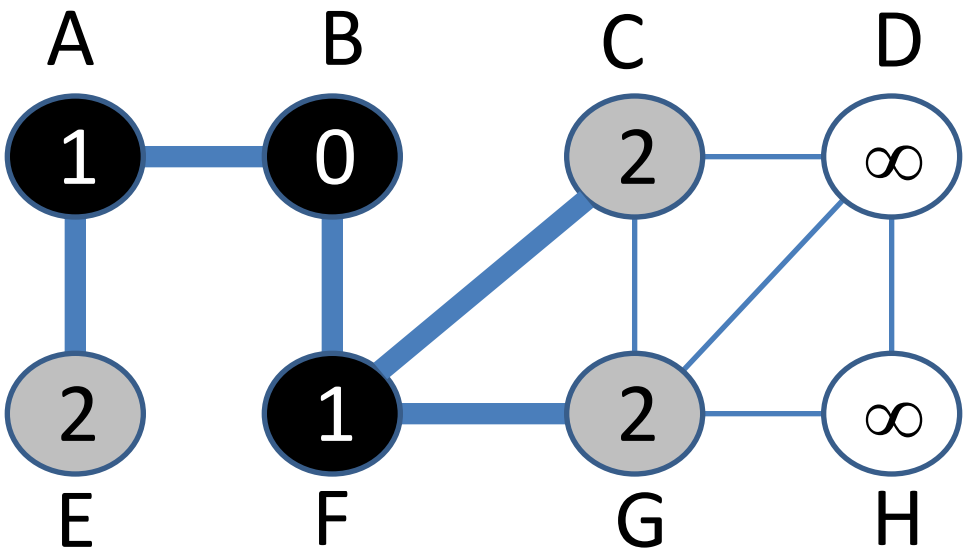


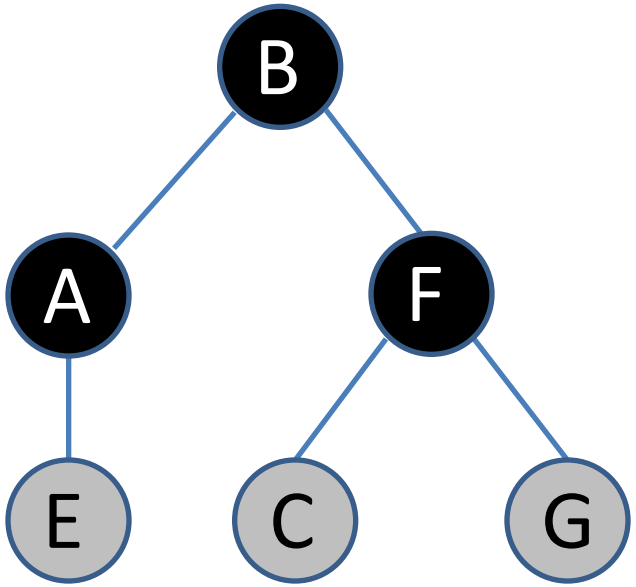


$Q = \{F_1, E_2\}$

BREADTH-FIRST SEARCH

$Q = \{F_1, E_2\}$ **expand F** **Breadth-first tree**

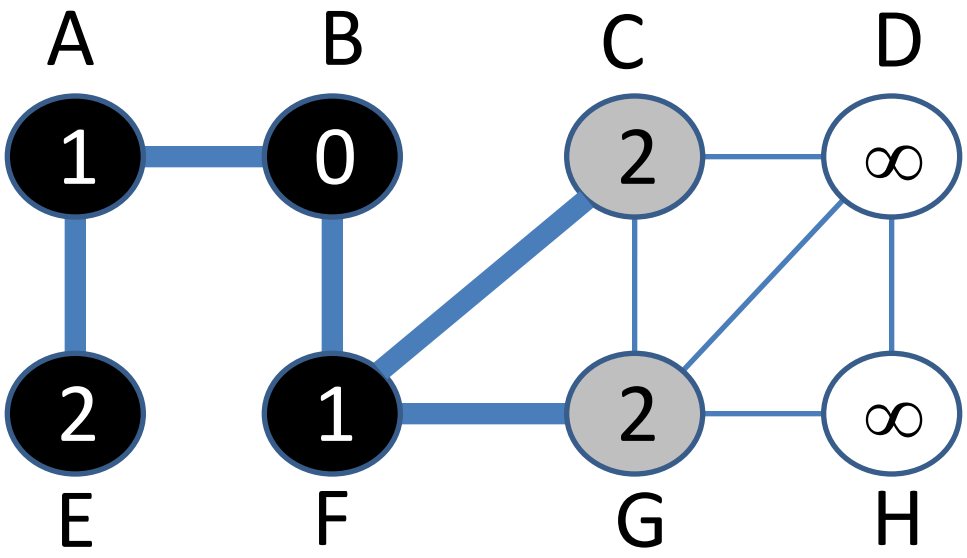




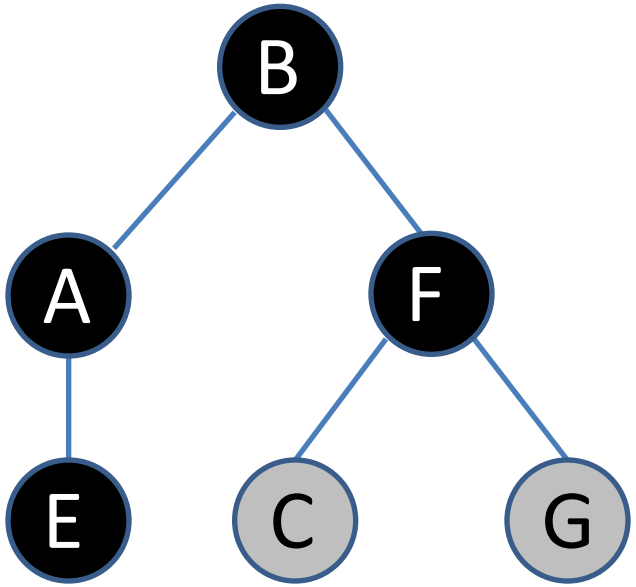
$Q = \{E_2, C_2, G_2\}$

BREADTH-FIRST SEARCH

$Q = \{E_2, C_2, G_2\}$ **expand E**



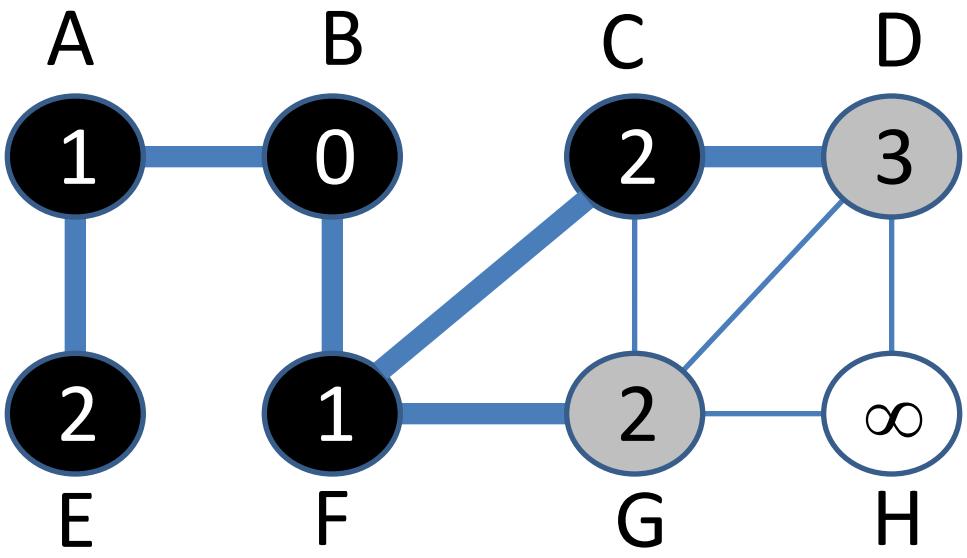
Breadth-first tree

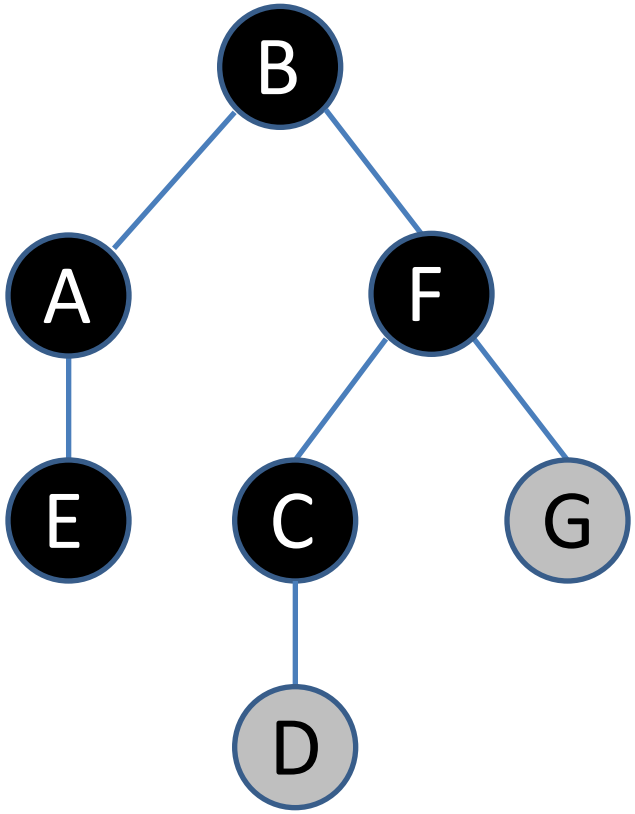


$Q = \{C_2, G_2\}$

BREADTH-FIRST SEARCH

$Q = \{C_2, G_2\}$ **expand C** **Breadth-first tree**

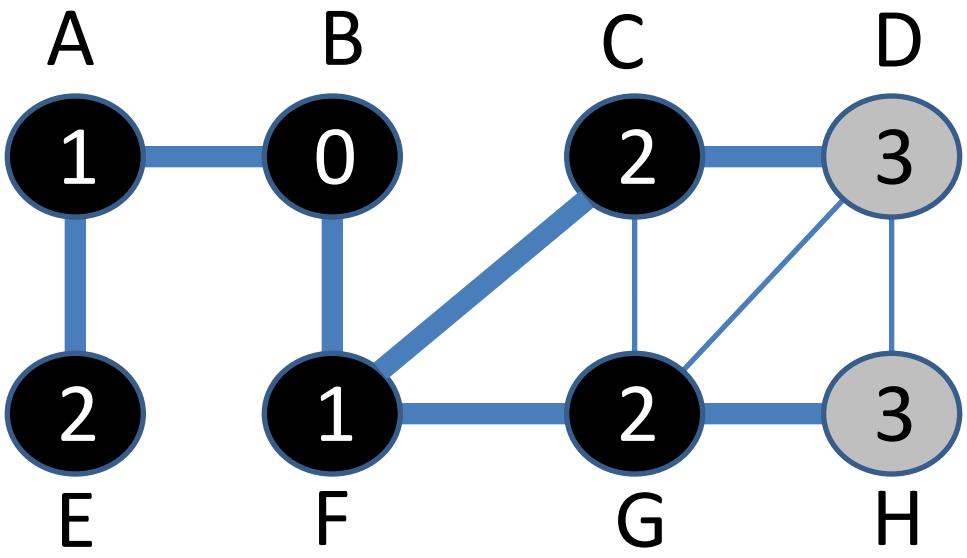


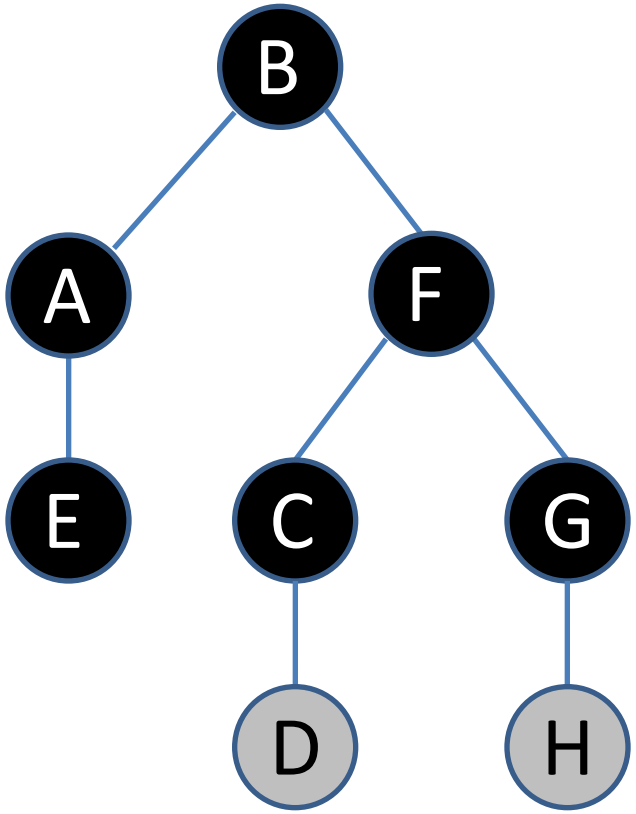


$Q = \{G_2, D_3\}$

BREADTH-FIRST SEARCH

$Q = \{G_2, D_3\}$ **expand G** **Breadth-first tree**

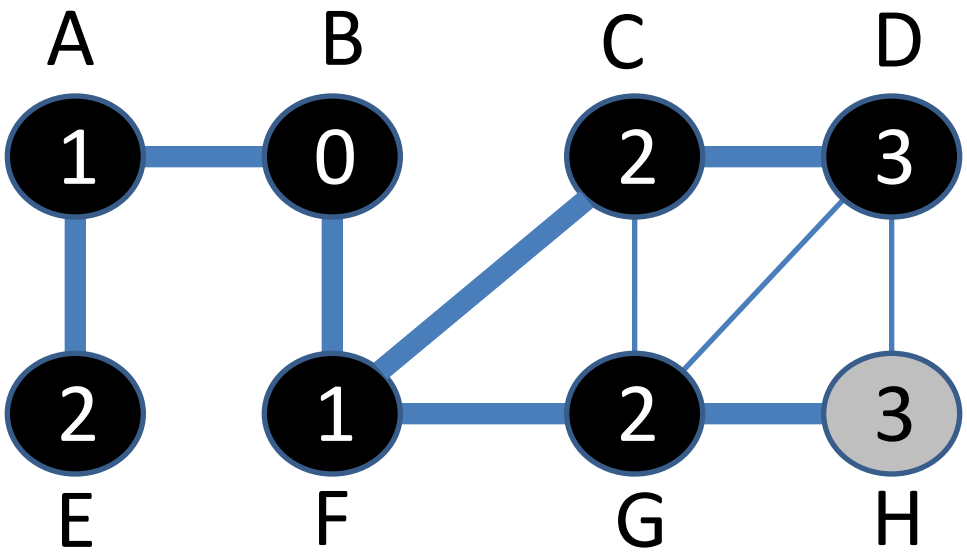


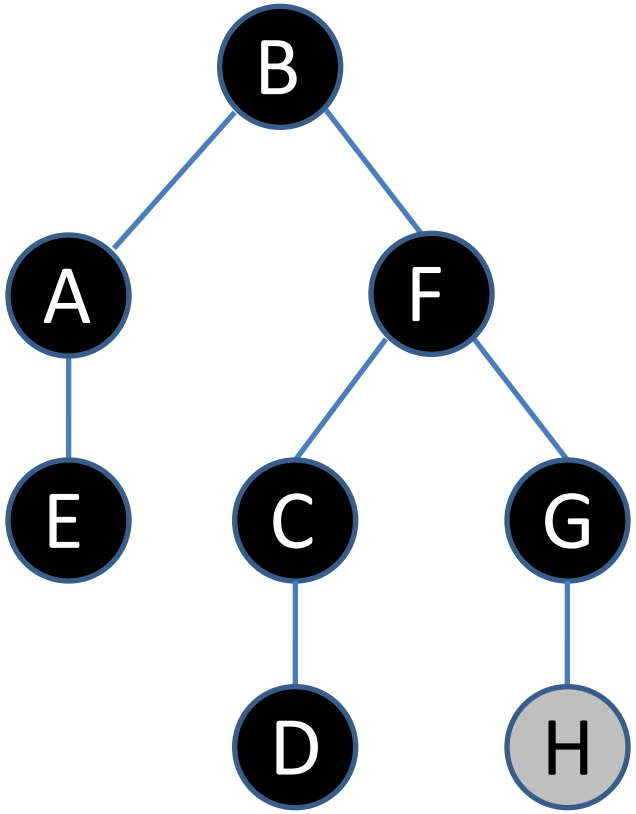


$Q = \{D_3, H_3\}$

BREADTH-FIRST SEARCH

$Q = \{D_3, H_3\}$ **expand D** **Breadth-first tree**

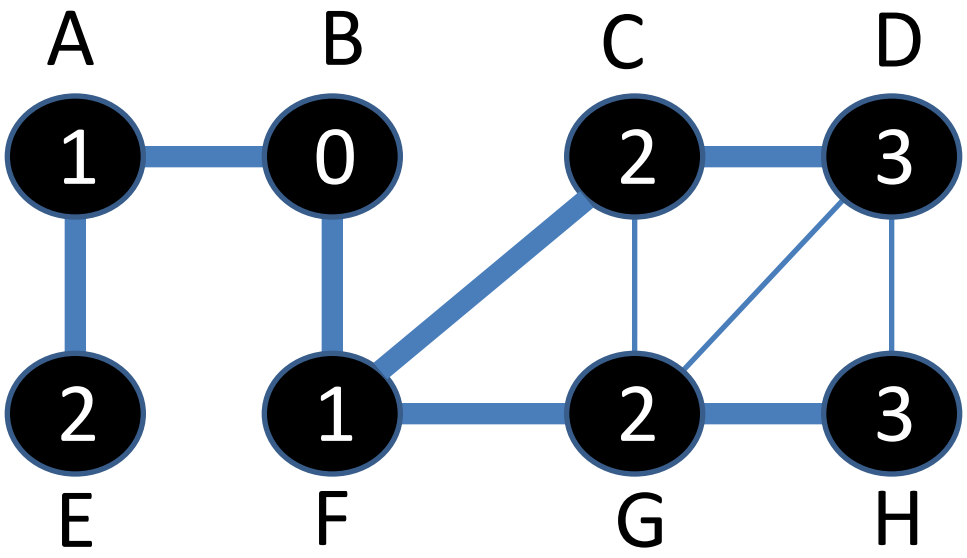


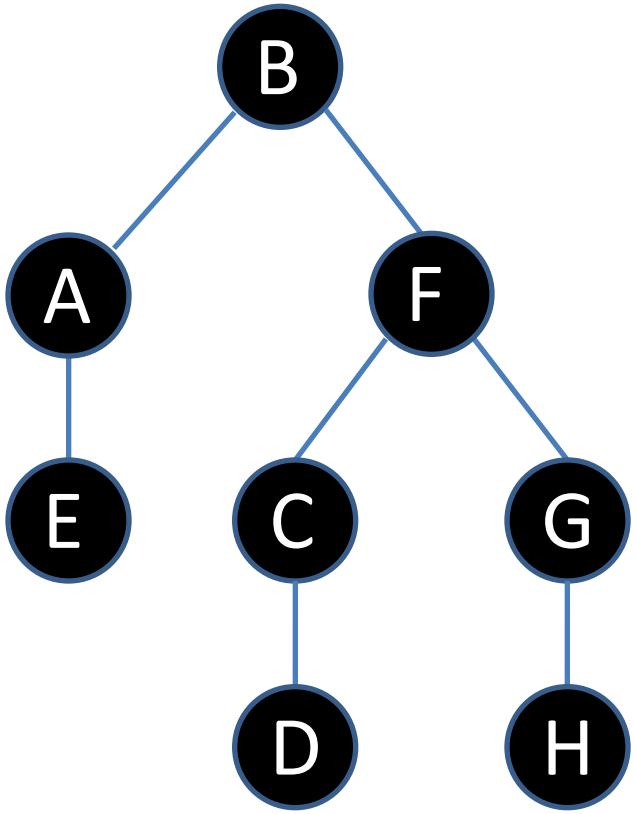


$Q = \{H_3\}$

BREADTH-FIRST SEARCH

$Q = \{H_3\}$ **expand H** **Breadth-first tree**





Q = {}

Demostraciones sobre BFS

1. Caminos más cortos (*shortest paths*)

2. Exactitud BFS (*correctness*)

3. Árbol de búsqueda en anchura (*BF tree*)

Demostraciones sobre BFS

1. **Caminos más cortos (*shortest paths*)**
2. Exactitud BFS (*correctness*)
3. Árbol de búsqueda en anchura (*BF tree*)

Shortest paths

$\delta(s,v)$=: número mínimo de aristas, si v se alcanza desde s (no ponderado)

$\delta(s,v) = \infty$, si v no se alcanza desde s

L1: Para cualquier arista (u,v): $\delta(s,v) \leq \delta(s,u) + 1$

Dem:

Shortest paths

L1: Para cualquier arista (u,v): $\delta(s,v) \leq \delta(s,u) + 1$

Dem:

si $\exists$ trayectoria (s,u), como mucho se recorre una arista más

si no, $\delta(s,v) = \infty$, y también se cumple

Shortest paths

L2: Tras ejecutar BFS, para todo v: **distance[v]≥ δ(s,v)**

Dem (inducción):

Shortest paths

L2: Tras ejecutar BFS, para todo v: **distance[v]≥ δ(s,v)**

Dem (inducción):

base: situación inicial: **d[s]= 0 = δ(s,s) y d[v]= ∞ ≥ δ(s,v)**

hipótesis: **d[u]≥ δ(s,u)**

inducción: v descubierto desde u (v "blanco"), entonces:

$$d[v] = d[u]+1 \quad \text{(línea 15)}$$
$$\geq \delta(s,u) + 1 \quad \text{(hipótesis)}$$
$$\geq \delta(s,v) \quad \text{(L1)}$$

Y una vez ENQUEUED, d[v] no varía.

Shortest paths

L3: Q contiene (v_1, v_2…v_r). Entonces:

$d[v_r] \leq d[1]+1$, y **$d[v_i] \leq d[v_{i+1}]$,** para i=1, 2,… r-1

Dem (inducción sobre el número de operaciones en Q):

Shortest paths

L3(1/2): Q contiene (v1, v2…vr). Entonces:

$d[v_r] \leq d[v_1]+1$, y **$d[v_i] \leq d[v_{i+1}]$,** para i=1, 2,… r-1

Dem (inducción sobre el número de operaciones en Q):

base: es cierto para la situación inicial (sólo s en Q)

Inducción 1: (DEQUEUE v_1) $\Rightarrow$ el primer vértice es v_2

$d[v_1] \leq d[v_2]$ (hipótesis)

$d[v_r] \leq d[v_1]+1$

$\leq d[v_2]+1$

El resto de las desigualdades para los demás vértices no varían

Shortest paths

L3: Q contiene (v_1, v_2…v_r). Entonces:

$d[v_r] \leq d[v_1]+1$, y **$d[v_i] \leq d[v_{i+1}]$,** para i=1, 2,… r-1

Dem (inducción sobre el número de operaciones en Q):

base: es cierto para la situación inicial (sólo s en Q)

Inducción 2: (ENQUEUE v_{r+1})

Shortest paths

L3 (2/2): Q contiene (v1, v2…vr). Entonces:

$d[v_r] \leq d[v_1]+1$, y **$d[v_i] \leq d[v_{i+1}]$,** para i=1, 2,… r-1

Dem (inducción sobre el número de operaciones en Q):

base: es cierto para la situación inicial (sólo s en Q)

Inducción 2: (ENQUEUE $v=v_{r+1}$) $\Rightarrow$ se ha eliminado "u" de Q. Se está explorando el vértice v (adyacente a u). Hay un nuevo v_1.

$d[u] \leq d[v_1]$ (hipótesis)

$d[v_{r+1}] = d[v] = d[u] + 1 \leq d[v_1]+1$ (v es adyacente a u)

$d[v_r] \leq d[u] +1$ (hipótesis)

Así que: $d[v_r] \leq d[u] +1 = d[v] = d[v_{r+1}]$

El resto de las desigualdades para los demás vértices no varían

Demostraciones sobre BFS

1. Caminos más cortos (*shortest paths*)

2. **Exactitud BFS (*correctness*)**

3. Árbol de búsqueda en profundidad (*BF tree*)

Correctness of BFS

Teorema: BFS en G, desde s. Entonces:

- Durante la ejecución, BFS descubre todos los vértices alcanzables desde s
- Al finalizar, $d[v] = \delta(s,v)$ para todo v
- Para cada $v \neq s$, uno de los caminos más cortos en G para ir de s a v es uno de los caminos más cortos para ir de s al predecesor[v] más la arista (predecesor[v], v)

Correctness of BFS

Dem (contradicción 1/3):

1. Supongamos v, con el menor valor de $\delta(s,v)$ tal que $d[v] \neq \delta(s,v)$. Evidentemente, $v \neq s$
2. $d[v] \geq \delta(s,v)$ *(por L2)* $\Rightarrow d[v] > \delta(s,v)$
3. v es alcanzable desde s (si no, $\delta(s,v) = \infty \geq d[v]$, que contradice la desigualdad anterior)
4. Sea u el vértice inmediatamente anterior a v por uno de los caminos más cortos: $\delta(s,v) = \delta(s,u)+1$ (luego $\delta(s,u) < \delta(s,v)$)
5. Por otra parte: $d[u] = \delta(s,u)$, por la forma en que v ha sido elegido

<u>Entonces:</u>

$$d[v] > \delta(s,v) = \delta(s,u)+1 = d[u]+1 \Rightarrow \quad d[v] > d[u]+1 \quad (*)$$

Correctness of BFS

Dem. (contradicción 2/3):

Ahora, en el algoritmo, BFS DEQUEUE u.

- Si v fuera "blanco" $\Rightarrow$ d[v]=d[u]+1 (línea 15) $\Rightarrow$ **contradicción ***
- Si v fuera "negro" v ya no estaría en Q, y d[v] $\leq$ d[u] $\Rightarrow$ **contradicción ***
- Si v fuera "gris" $\Rightarrow$ v es gris antes de DEQUEUE u, al sacar otro vértice distinto w, para el que d[v]=d[w]+1

  Si w salió de Q antes que u $\Rightarrow$ d[w] $\leq$ d[u] $\Rightarrow$ d[v]$\leq$ d[u]+1 $\Rightarrow$ **contradicción ***

<u>Por lo que d[v]= δ(s,v) para todo v</u>

Correctness of BFS

Dem (contradicción 3/3):

Se deduce también que:

- BFS descubre todos los vértices alcanzables desde s. Si no fuera así, tendríamos para algún v ∞ = d[v] > δ(s,v), que contradice el resultado anterior.

- Como predecesor[v]=u, entonces d[v]=d[u]+1. Así que uno de los caminos más cortos para ir de s a v es uno de los caminos más cortos para ir de s al predecesor[v] más la arista (predecesor[v], v)

Demostraciones sobre BFS

1. Caminos más cortos (*shortest paths*)

2. Exactitud BFS (*correctness*)

3. **Árbol de búsqueda en profundidad (*BF tree*)**

Breadth-first tree

Def: Dados G=(V,E) y s, se define subgrafo predecesor de G: $G_p=(V_p, E_p)$, siendo:

$$V_p=\{v \in V : p[v] \neq NIL\} \cup \{s\}$$

$$E_p=\{(p(v)\ v): v \in V_p - \{s\}\ \}$$

G_p es un **breadth-first tree** si:

- V_p = todos los v alcanzables desde s
- Existe un único camino desde s a v en G_p, que es, además, el camino más corto (s,v) en G

Breadth-first tree

L: BFS construye una estructura de datos con **predecessor** de modo que el correspondiente subgrafo predecesor $G_p=(V_p, E_p)$ es un **breadth-first tree**

Dem:

Breadth-first tree

L: BFS construye una estructura de datos con **predecessor** de modo que el correspondiente subgrafo predecesor $G_p=(V_p, E_p)$ es un **breadth-first tree**

Dem:

p[v]=u (línea 16), sii $(u,v)\in E$ y $\delta(s,v) < \infty$ (esto es, si v puede alcanzarse desde s)

$\Rightarrow V_p$ contiene todos los v alcanzables desde s

El árbol G_p tiene un único camino para cada $v\in V_p \Rightarrow$todos los caminos son los más cortos (teorema anterior)