

# Algorísmica Avançada

## Algorismes greedy

Sergio Escalera

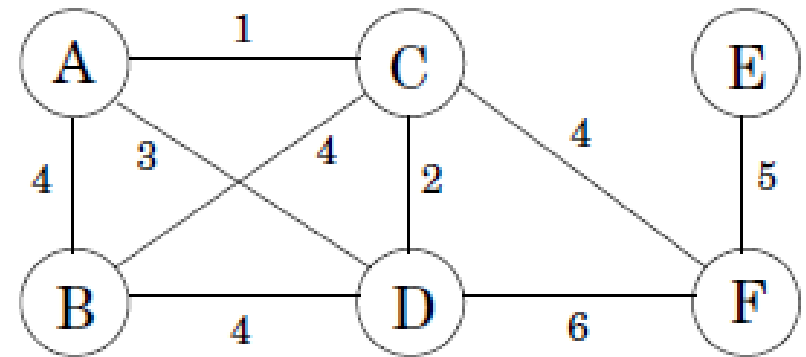
A series of horizontal lines of varying lengths and colors (teal, light blue, and white) extending from the left edge of the slide towards the right, positioned below the author's name.

# Algorismes greedy

- Podem guanyar als escacs pensant només en la següent jugada?
- I al scrabble? → algorithme greedy?
- **Algorismes greedy troben la millor “jugada” a cada pas**

# Algorismes greedy

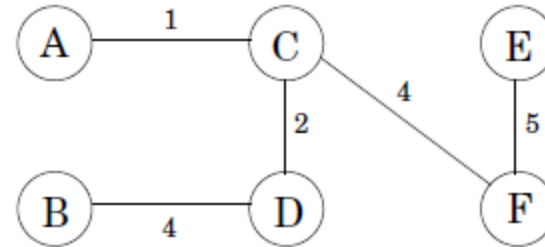
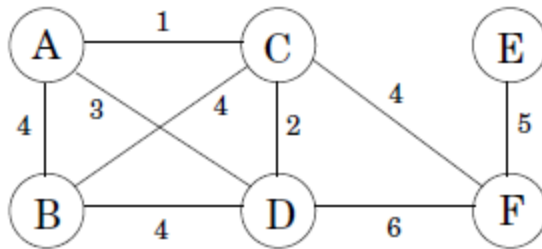
- Exemple



- Volem connectar els ordinadors (nodes) d'una xarxa. Les connexions són les arestes. Cadascuna té un cost. Volem el mínim cost.
  - → llavors no volem cicles
  - → volem un graf no dirigit acíclic connectat
    - → arbre !!!
  - → de mínim cost: **Minimum Spanning Tree (MST)**

# Algorismes greedy

- MST amb cost 16 (un dels possibles)

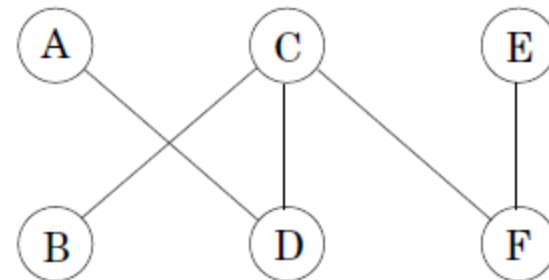
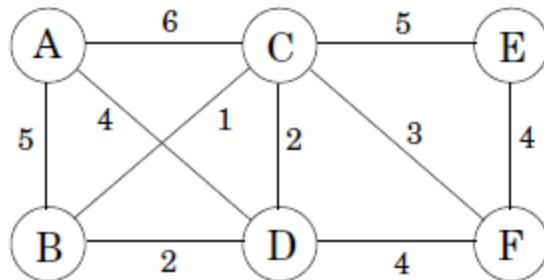


- Algorisme greedy: Kruskal
  - Començar amb arbre buit
  - Mentre no estiguin tots els nodes connectats:
    - Incloure aresta de cost mínim que no produeix un cicle

# Algorismes greedy

- Cost 14!

$B - C, C - D, B - D, C - F, D - F, E - F, A - D, A - B, C - E, A - C.$



- Aquest algorisme és **òptim!**

# Algorismes greedy

- **¿Per què? Propietat de tall “cut”:**
- Un tall és aquella aresta que si la traiem es genera una nova component connexa.
- El que fem amb Kruskal és anar connectant elements amb el tall de cost mínim.
- Com ho podem implementar eficientment?

# Algorismes greedy

procedure kruskal( $G, w$ )

Input: A connected undirected graph  $G = (V, E)$  with edge weights  $w_e$

Output: A minimum spanning tree defined by the edges  $X$

for all  $u \in V$ :

    makeset( $u$ )

$X = \{\}$

Sort the edges  $E$  by weight

for all edges  $\{u, v\} \in E$ , in increasing order of weight:

    if find( $u$ )  $\neq$  find( $v$ ):

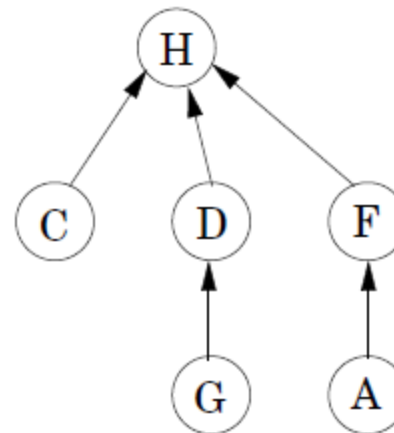
        add edge  $\{u, v\}$  to  $X$

        union( $u, v$ )

$ V $ makeset, $2 E $ find, $ V  - 1$ union
---

# Algorismes greedy

- Representació dels conjunts: arbres dirigits



procedure makeset( $x$ )

$\pi(x) = x$

$\text{rank}(x) = 0$

function find( $x$ )

while  $x \neq \pi(x)$ :  $x = \pi(x)$

return  $x$

$\pi$  punter

rank: altura dins de l'arbre



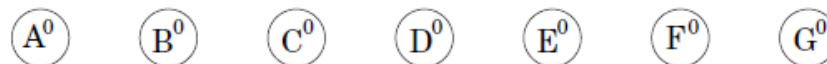
# Algorismes greedy

- **Makeset:** temps constant
- **Find:** segueix punters dels pares als roots, per tant el temps és proporcional a l'altura
- **Union:** com l'altura ens defineix la complexitat, posem el punter de l'arbre més curt apuntant al punter de l'arbre amb més altura

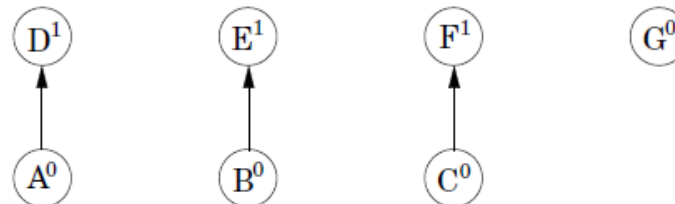
```
procedure union( $x, y$ )  
   $r_x = \text{find}(x)$   
   $r_y = \text{find}(y)$   
  if  $r_x = r_y$ : return  
  if  $\text{rank}(r_x) > \text{rank}(r_y)$ :  
     $\pi(r_y) = r_x$   
  else:  
     $\pi(r_x) = r_y$   
    if  $\text{rank}(r_x) = \text{rank}(r_y)$ :  $\text{rank}(r_y) = \text{rank}(r_y) + 1$ 
```

# Algorithms greedy

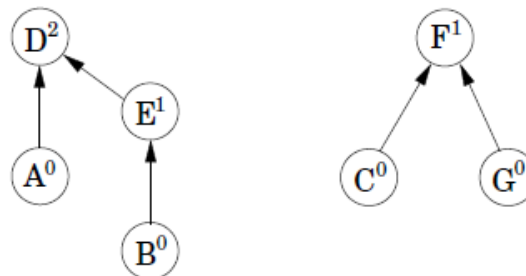
After  $\text{makeset}(A), \text{makeset}(B), \dots, \text{makeset}(G)$ :



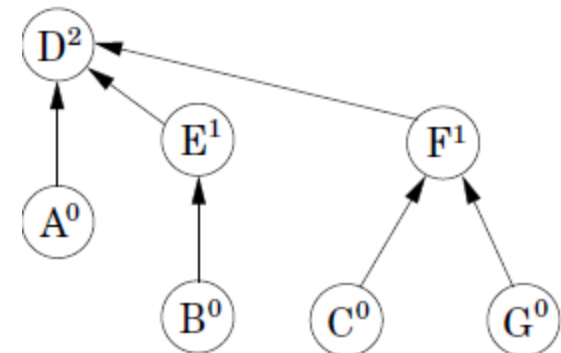
After  $\text{union}(A, D), \text{union}(B, E), \text{union}(C, F)$ :



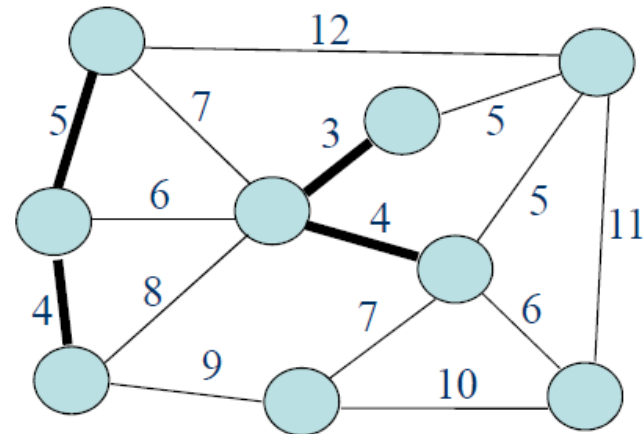
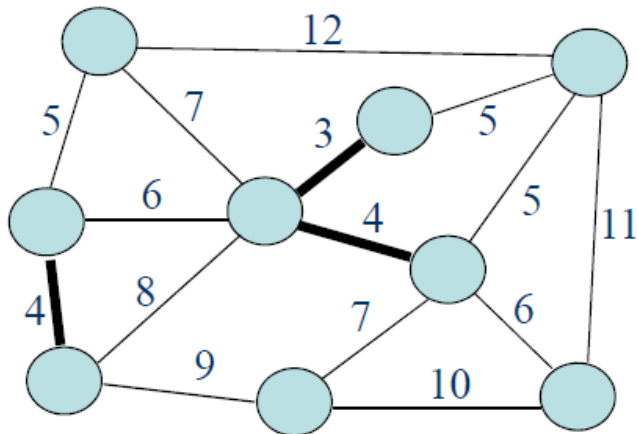
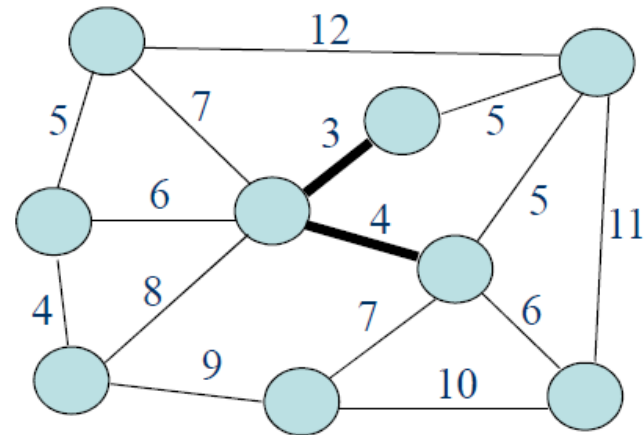
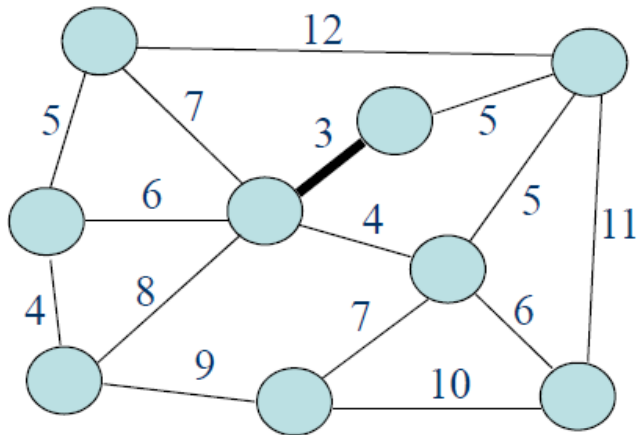
After  $\text{union}(C, G), \text{union}(E, A)$ :



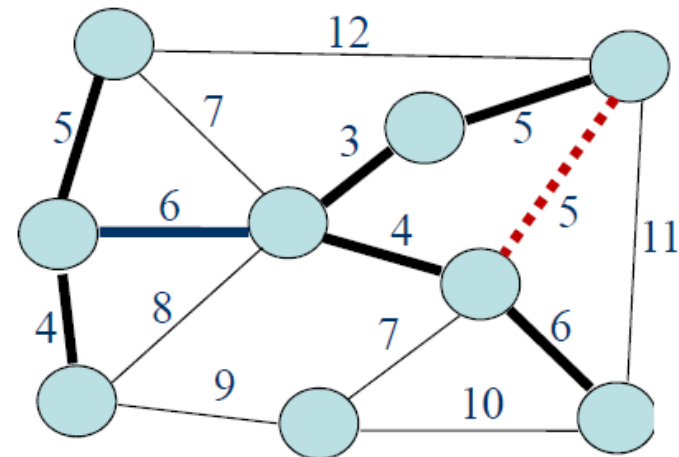
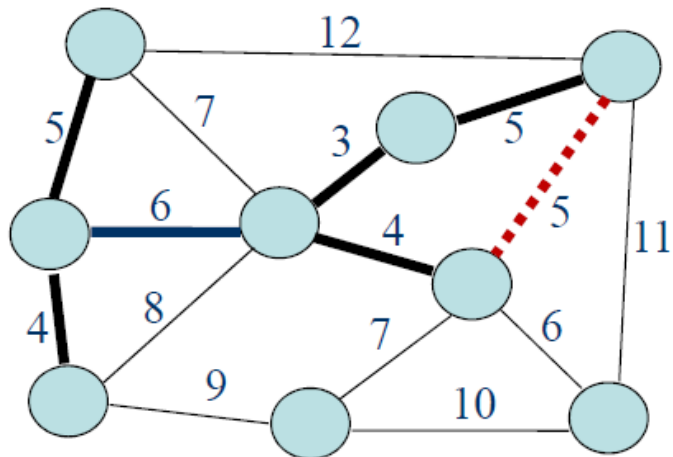
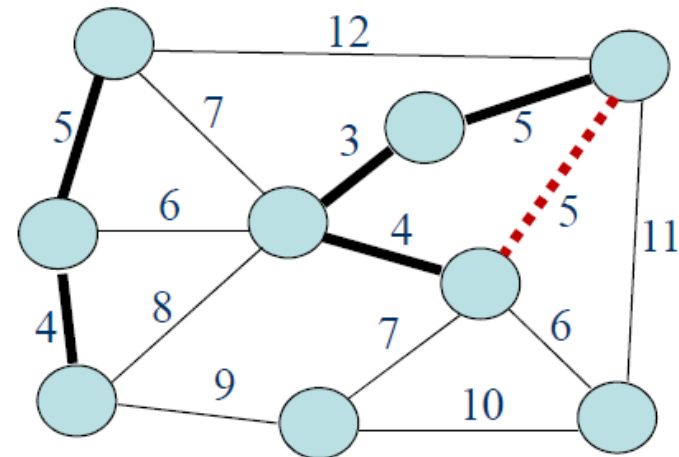
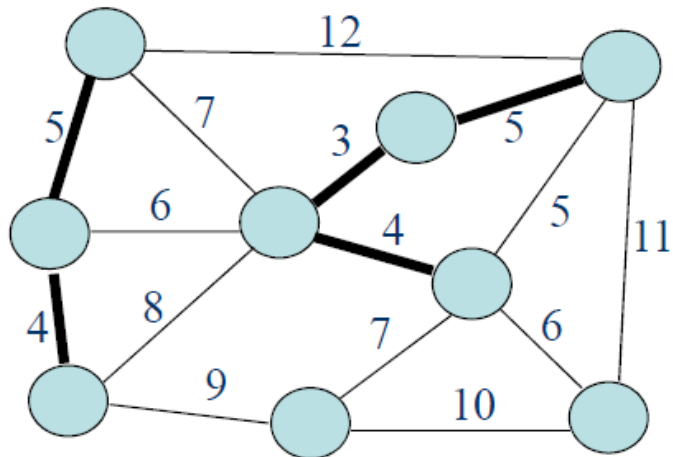
After  $\text{union}(B, G)$ :



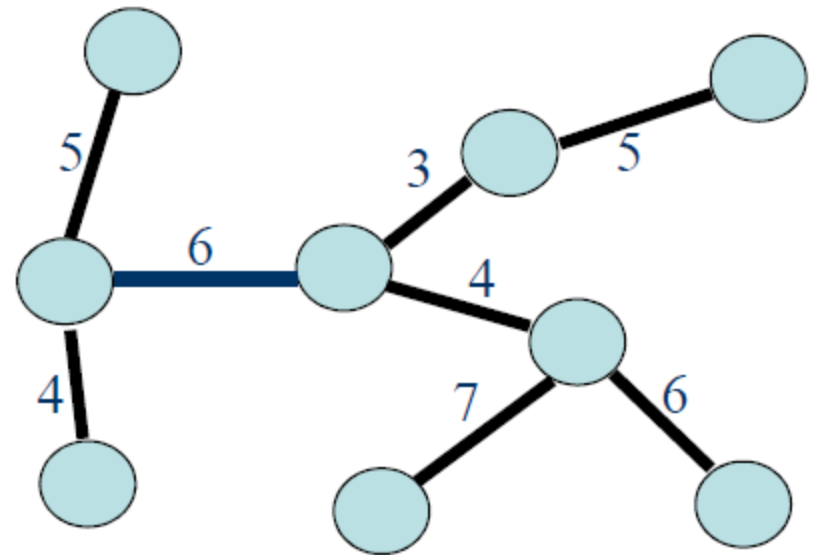
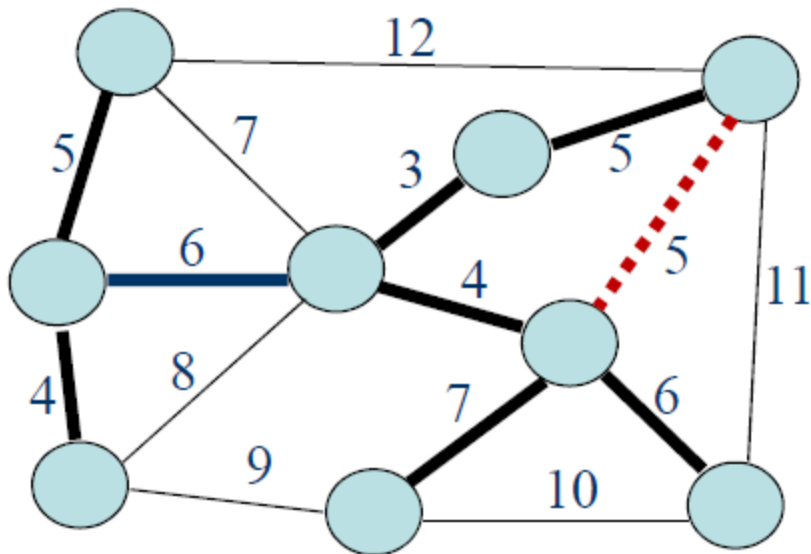
# Kruskal exemple



# Kruskal exemple



# Kruskal exemple



# Algorismes greedy

- Exemple: Algorisme de Prim
  - Alternativa a Kruskal
- La propietat de tall ens diu que qualsevol algorisme que segueix el següent procediment hauria de funcionar :

$X = \{ \}$  (edges picked so far)

repeat until  $|X| = |V| - 1$ :

    pick a set  $S \subset V$  for which  $X$  has no edges between  $S$  and  $V - S$

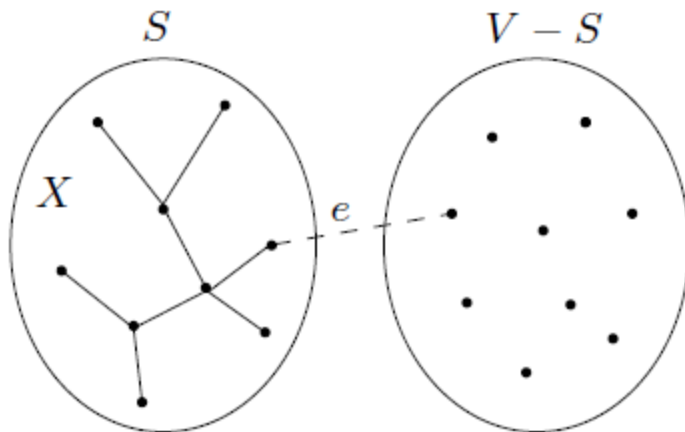
    let  $e \in E$  be the minimum-weight edge between  $S$  and  $V - S$

$X = X \cup \{e\}$

# Algorismes greedy

- Prim: similar a kruskal però per nodes

$$\text{cost}(v) = \min_{u \in S} w(u, v).$$



La complexitat és similar  
a l'algorisme de kruskal

# Algorismes greedy

procedure prim( $G, w$ )

Input: A connected undirected graph  $G = (V, E)$  with edge weights  $w_e$

Output: A minimum spanning tree defined by the array prev

for all  $u \in V$ :

$\text{cost}(u) = \infty$

$\text{prev}(u) = \text{nil}$

Pick any initial node  $u_0$

$\text{cost}(u_0) = 0$

$H = \text{makequeue}(V)$  (priority queue, using cost-values as keys)

while  $H$  is not empty:

$v = \text{deletemin}(H)$

    for each  $\{v, z\} \in E$ :

        if  $\text{cost}(z) > w(v, z)$ :

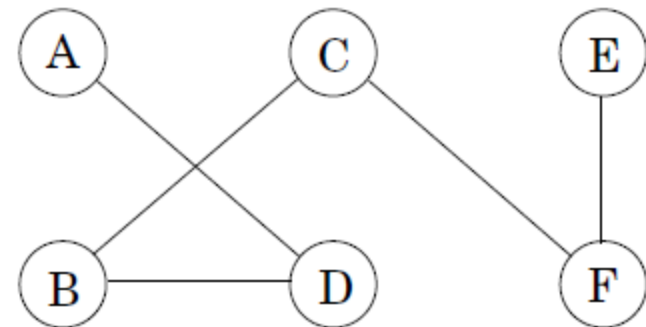
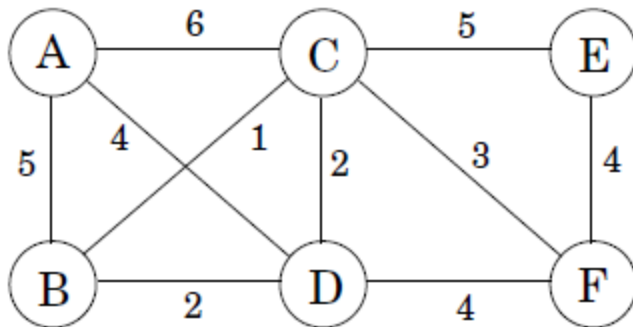
$\text{cost}(z) = w(v, z)$

$\text{prev}(z) = v$

$\text{decreasekey}(H, z)$



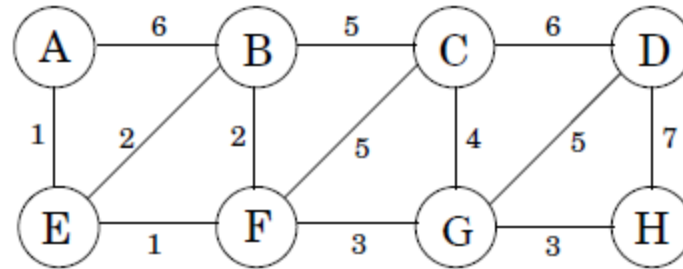
# Algorismes greedy



Set $S$	$A$	$B$	$C$	$D$	$E$	$F$
$\{\}$	0/nil	$\infty$ /nil	$\infty$ /nil	$\infty$ /nil	$\infty$ /nil	$\infty$ /nil
$A$		5/ $A$	6/ $A$	4/ $A$	$\infty$ /nil	$\infty$ /nil
$A, D$		2/ $D$	2/ $D$		$\infty$ /nil	4/ $D$
$A, D, B$			1/ $B$		$\infty$ /nil	4/ $D$
$A, D, B, C$					5/ $C$	3/ $C$
$A, D, B, C, F$					4/ $F$	

# Algorismes greedy

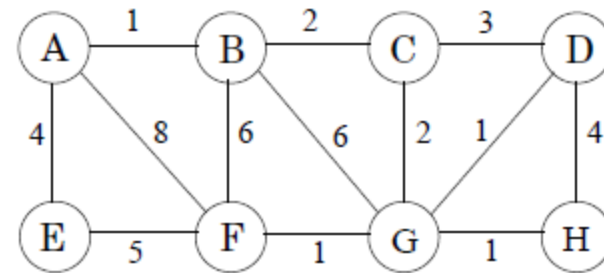
- Exercicis (1):



- A) Quin és el cost del MST?
- B) En quin ordre les arestes són incloses en el MST usant l'algorisme Kruskal?

# Algorismes greedy

- Exercicis (2):



- Aplica l'algorisme Prim (order alfabètic)
  - Escriu la taula de costos intermitjos
- Aplica l'algorisme Kruskal i mostra els diferents arbres intermitjos

# Algorismes greedy

- En altres casos, els algorismes greedy obtenen respostes aproximades
  - → factor d'aproximació
- No són òptimes, però no existeixen algorismes lineals que solucionen el problema
  - → Ho veurem a problemes NP