

Algorísmica Avançada

Algorismes sobre grafs II

Sergio Escalera

A series of horizontal lines of varying lengths and colors (teal, light blue, and white) extending from the right side of the slide.

Algorismes sobre grafs

- **Pesos a les arestes**
- Exemple amb distàncies

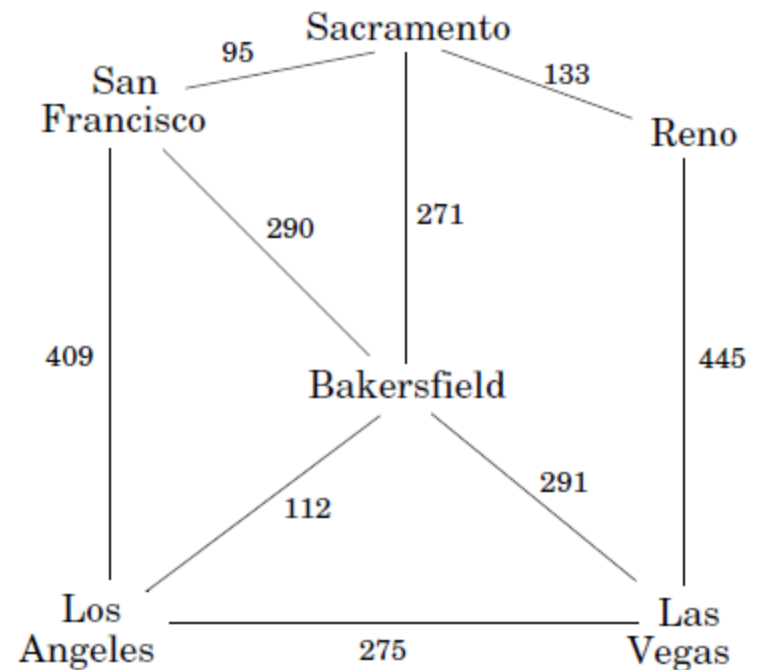
$e \in E$ Aresta

l_e Longitud

$e = (u, v)$ Notació d'aresta I

$l(u, v)$ Notació d'aresta II

l_{uv} Notació d'aresta III

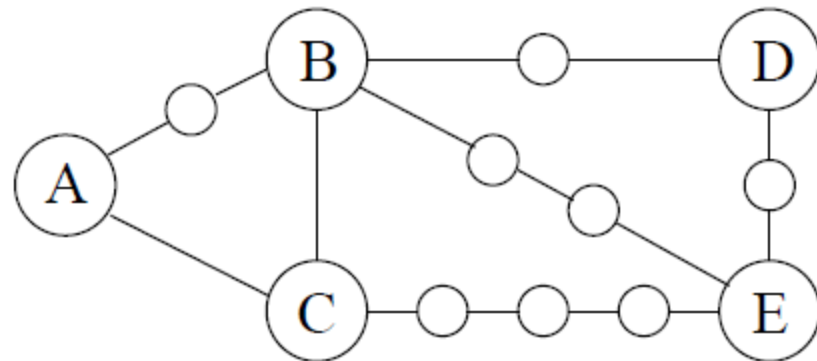
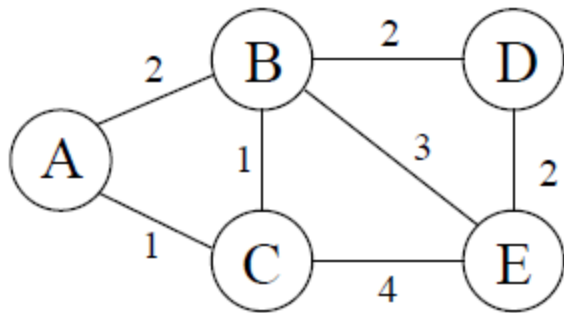


Algorismes sobre grafs

- De moment suposem que tots els pesos són positius ≥ 0
- BFS troba camins mínims on les arestes tenen un cost unitari.
- Cóm ho fem general per a qualsevol graf $G=(V,E)$ amb l_e enters positius?
 - **Algorisme de Dijkstra**

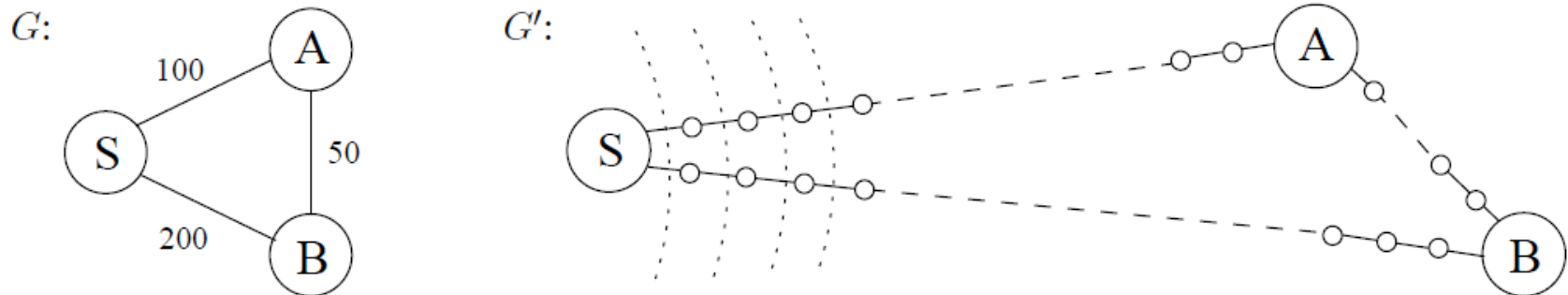
Algorismes sobre grafs

- **Algorisme de Dijkstra**
 - Una versió per fer ús de BFS
 - Dividir les longituds en valors unaris incloent vèrtexs extra



Algorismes sobre grafs

- Un problema evident



- Pensem millor en posar una “alarma” a cada node i l’actualitzem a mida que arribem fent ús de DFS. Els valors de les alarmes podrien ser els costos de les arestes!!!

Algorismes sobre grafs

- Set an alarm clock for node s at time 0.
- Repeat until there are no more alarms:
Say the next alarm goes off at time T , for node u . Then:
 - The distance from s to u is T .
 - For each neighbor v of u in G :
 - * If there is no alarm yet for v , set one for time $T + l(u, v)$.
 - * If v 's alarm is set for later than $T + l(u, v)$, then reset it to this earlier time.

Ara ens queda implementar el sistema d'alarmes

Algorismes sobre grafs

- **Algorisme de Dijkstra**
 - **Cua amb prioritats** → generalment **heap**

Manté un conjunt d'elements (nodes) amb les valors numèrics associats com a claus (temps de l'alarma) i suporta les següents operacions:

Inserció: inclou un nou element al conjunt.

Decrementar-clau: Decrementa el valor de la clau d'un element particular. La cua amb prioritats normalment no canvia el valor de les claus, el que fa és notificar a la cua que el valor d'una certa clau ha estat decrementat.

Eliminar-min: Retorna l'element amb la menor clau i l'elimina del conjunt.

Fer-cua: Construeix una cua amb prioritats amb els elements donats i els seus valors de clau associats.

Algorismes sobre grafs

- **Inserció:** inclou un nou element al conjunt.
 - **Decrementar-clau:** Decrementa el valor de la clau d'un element particular. La cua amb prioritats normalment no canvia el valor de les claus, el que fa és notificar a la cua que el valor d'una certa clau ha estat decrementat.
 - **Eliminar-min:** Retorna l'element amb la menor clau i l'elimina del conjunt.
 - **Fer-cua:** Construeix una cua amb prioritats amb els elements donats i els seus valors de clau associats.
-
- Inserir i decrementar clau ens permet fixar les alarmes, mentre que eliminar-min ens diu quina és la pròxima alarma a tenir en compte.

Algoritmos sobre grafos

procedure `dijkstra`(G, l, s)

Input: Graph $G = (V, E)$, directed or undirected;
positive edge lengths $\{l_e : e \in E\}$; vertex $s \in V$

Output: For all vertices u reachable from s , `dist`(u) is set
to the distance from s to u .

for all $u \in V$:

`dist`(u) = ∞

`prev`(u) = nil

`dist`(s) = 0

$H = \text{makequeue}(V)$ (using `dist`-values as keys)

while H is not empty:

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

 for all edges $(u, v) \in E$:

 if `dist`(v) > `dist`(u) + $l(u, v)$:

`dist`(v) = `dist`(u) + $l(u, v)$

`prev`(v) = u

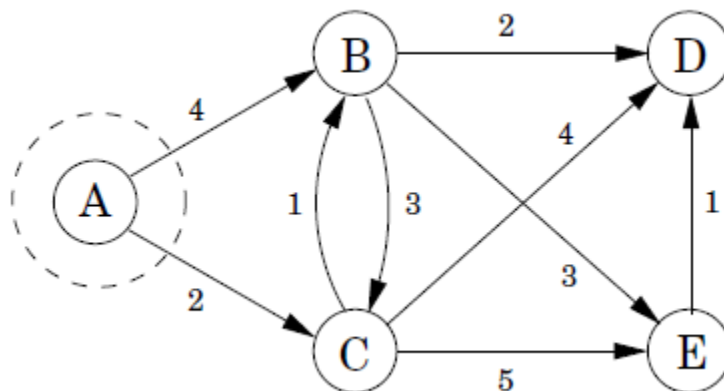
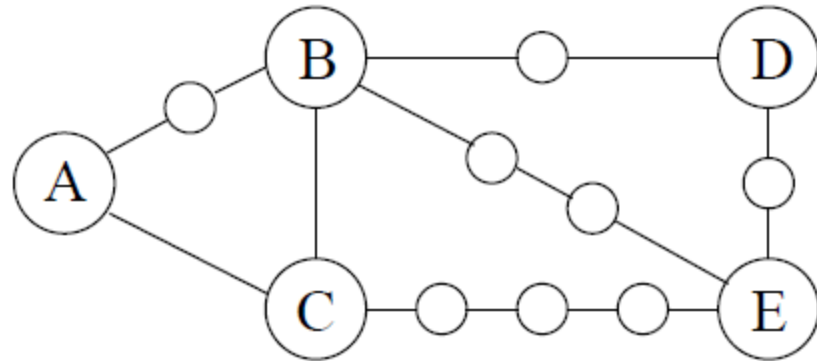
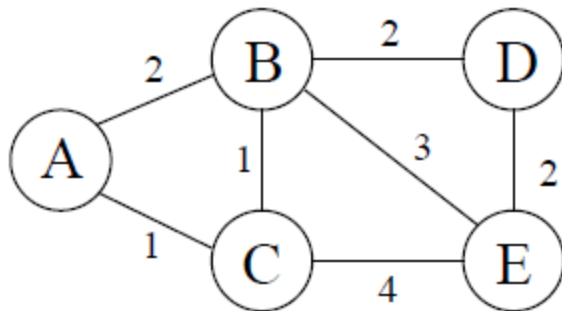
`decreasekey`(H, v)

Algorismes sobre grafs

- **dist(u)** es refereix als valors actuals d'alarma del node u. Un valor infinit significa que encara no hem posat valor a l'alarma.
- L'array **prev**: guarda informació del node immediat abans del node actual u dins la ruta més curta entre s i u.
- Si retornem fent ús dels valor d'aquests punters podem reconstruir els camins més curts de forma senzilla.
- Podem veure que la diferència principal entre l'algorisme Dijkstra I BFS és que el primer usa una cua amb prioritats en lloc d'una cua regular, de forma que prioritza nodes en funció dels costos de les arestes.

Algorismes sobre grafos

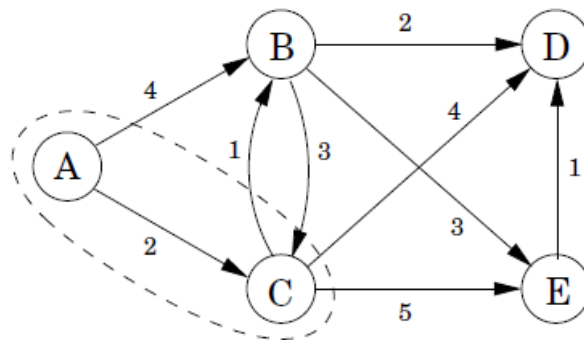
- Algorisme de Dijkstra: exemple graf dirigit



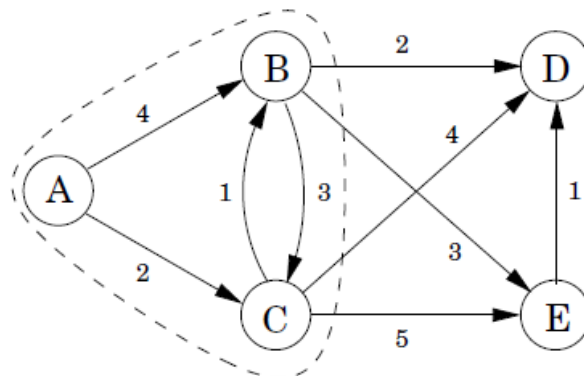
A: 0	D: ∞
B: 4	E: ∞
C: 2	

Algoritmes sobre grafos

- Algoritme de Dijkstra: exemple graf dirigit



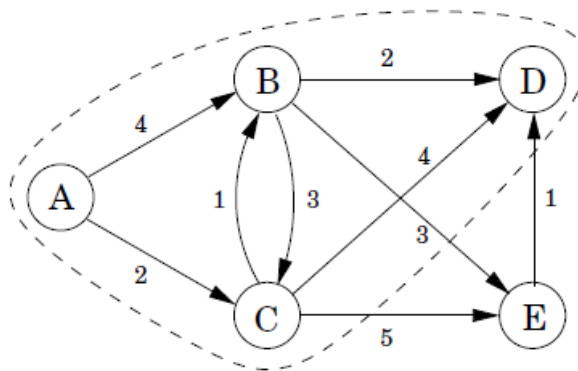
A: 0	D: 6
B: 3	E: 7
C: 2	



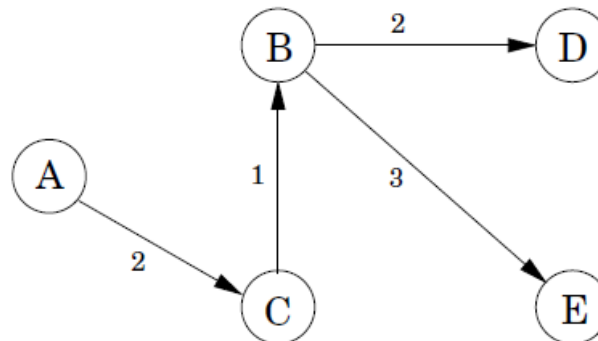
A: 0	D: 5
B: 3	E: 6
C: 2	

Algorismes sobre grafos

- Algorisme de Dijkstra: exemple graf dirigit

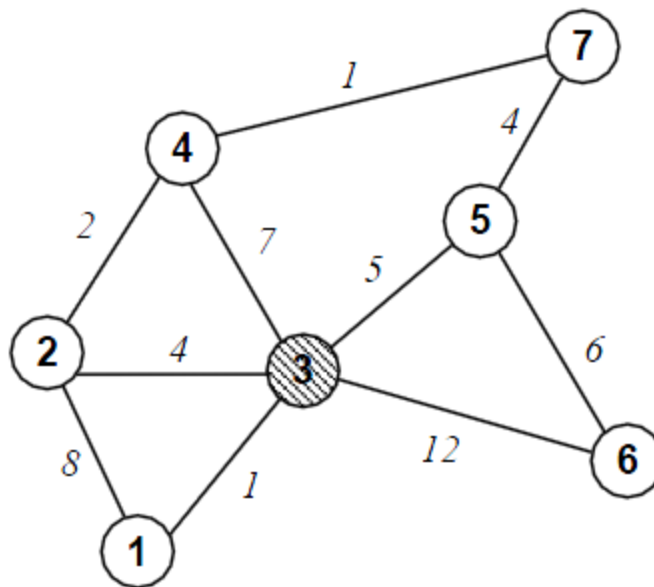


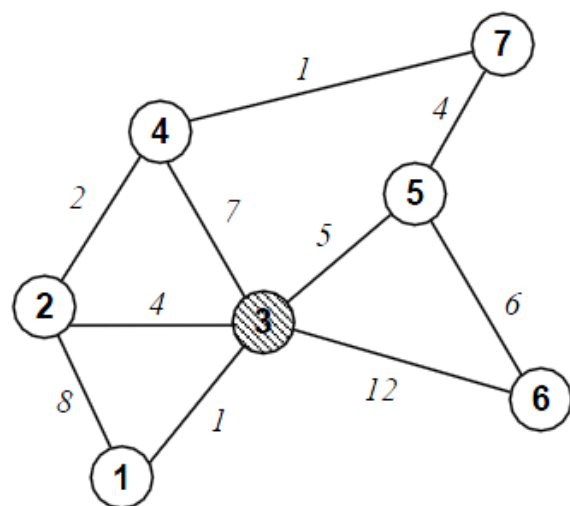
A: 0	D: 5
B: 3	E: 6
C: 2	



Algorismes sobre grafos

- Algorisme de Dijkstra: exemple graf no dirigit





Iteración	<i>u</i>	Vectores							S	
Inicial	---		1	2	3	4	5	6	7	[]
		D:	∞	∞	0	∞	∞	∞	∞	
		P:	-	-	-	-	-	-	-	
1	3	D:	1	4	0	7	5	12	∞	[3]
		P:	3	3	-	3	3	3	-	
2		D:								
		P:								
3		D:								
		P:								
4		D:								
		P:								
5		D:								
		P:								
6		D:								
		P:								
7		D:								
		P:								

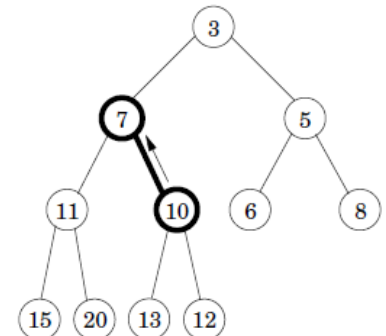
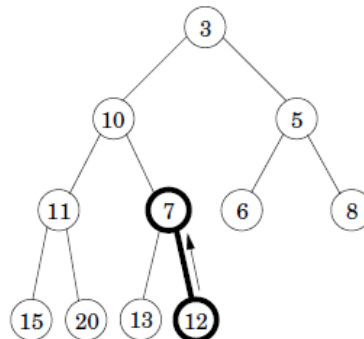
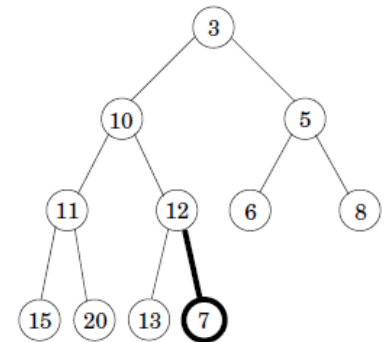
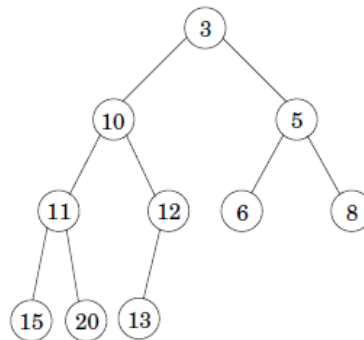
Algorismes sobre grafs

- Veiem la complexitat d'implementar les cues amb prioritat per fer l'algorisme de **Dijkstra**.

Implementation	deletemin	insert/ decreasekey	$ V \times \text{deletemin} + (V + E) \times \text{insert}$
Array	$O(V)$	$O(1)$	$O(V ^2)$
Binary heap	$O(\log V)$	$O(\log V)$	$O((V + E) \log V)$

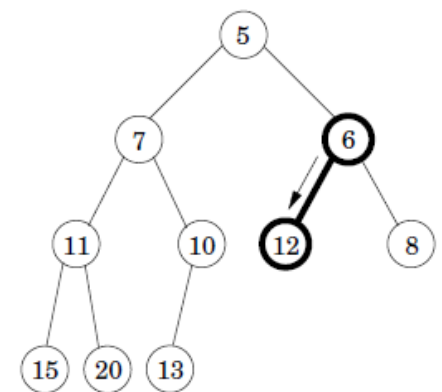
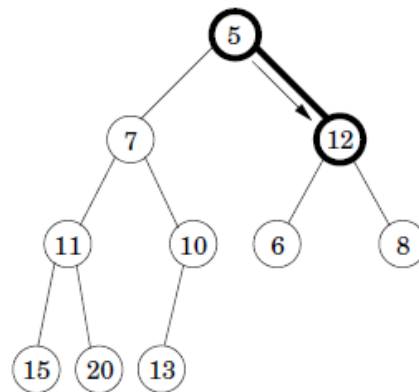
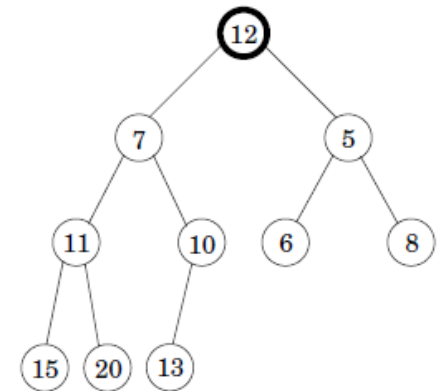
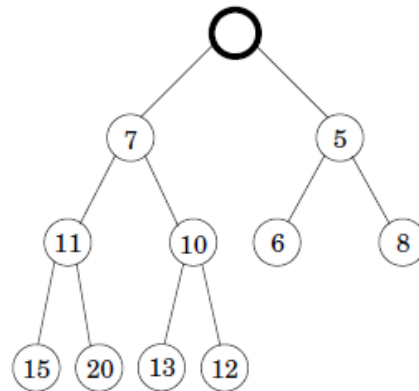
Algorismes sobre grafes

- **Array:** inserció directa, eliminació lineal.
- **Binary heap:**



Algoritmos sobre grafos

- Binary heap



Algoritmos sobre grafos

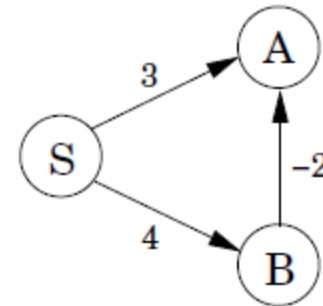
- Ejemplo práctico \rightarrow GEOCUT

Algorismes sobre grafs

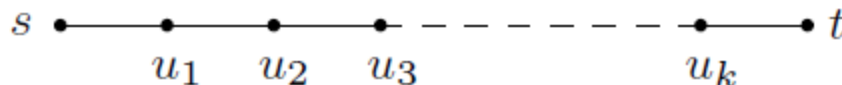
- **Dijkstra** amb pesos negatius

```

procedure update  $((u, v) \in E)$ 
 $\text{dist}(v) = \min\{\text{dist}(v), \text{dist}(u) + l(u, v)\}$ 
  
```



- Amb Dijkstra sempre arribem de s a t amb camí mínim independentment de l'ordre dels pesos de les arestes si aquests són positius. **No amb negatius!**



Algorismes sobre grafs

- Solució? Canviem l'algorisme perquè es calculin les distàncies simultàneament \rightarrow actualitzar totes les arestes $|V|-1$ vegades ($O(|V| \cdot |E|)$)

- **Bellman-Ford**

procedure shortest-paths(G, l, s)

Input: Directed graph $G = (V, E)$;

edge lengths $\{l_e : e \in E\}$ with no negative cycles;

vertex $s \in V$

Output: For all vertices u reachable from s , $\text{dist}(u)$ is set to the distance from s to u .

for all $u \in V$:

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

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

$\text{dist}(s) = 0$

repeat $|V| - 1$ times:

 for all $e \in E$:

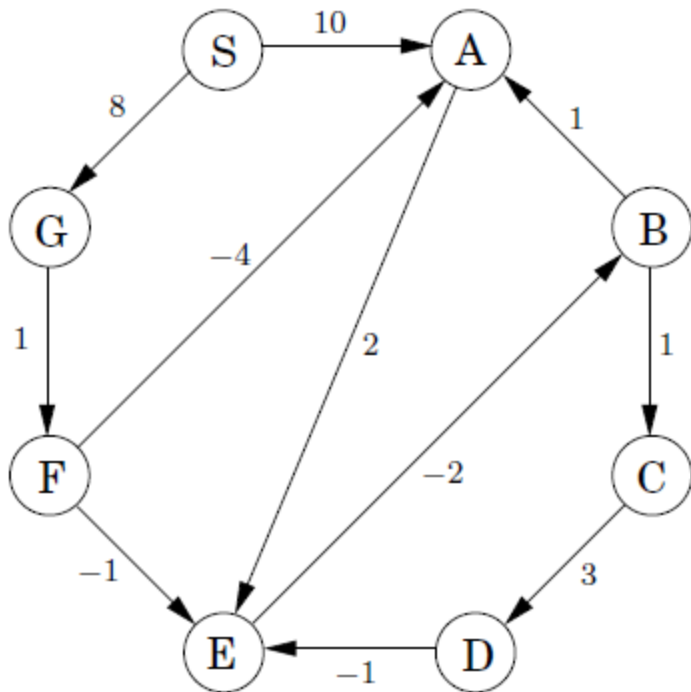
 update(e)

procedure update($(u, v) \in E$)

$\text{dist}(v) = \min\{\text{dist}(v), \text{dist}(u) + l(u, v)\}$

Algorismes sobre grafs

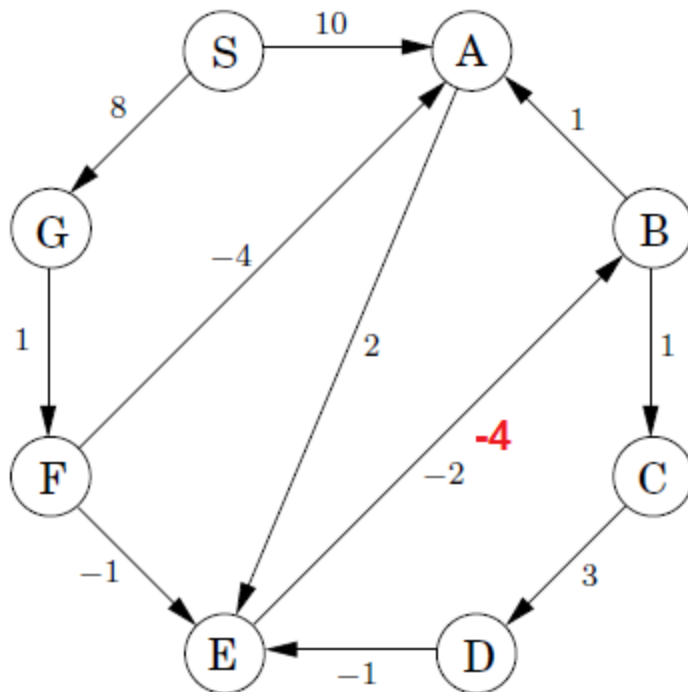
- Implementació: si en una iteració cap aresta e s'actualitza \rightarrow finalitzar



	Iteration							
Node	0	1	2	3	4	5	6	7
S	0	0	0					
A	∞	10	10					
B	∞	∞	∞					
C	∞	∞	∞					
D	∞	∞	∞					
E	∞	∞	12					
F	∞	∞	9					
G	∞	8	8					

Algorismes sobre grafs

- Cicles negatius



$A \rightarrow E \rightarrow B \rightarrow A$

Els podem trobar amb l'algorisme Bellman-Ford

El camí mínim té com a màxim longitud $|V|-1$

Podem detectar cicles si fem una iteració extra $|V|$

→ Hi ha cicle negatiu si a la iteració $|V|$ alguna aresta és actualitzada

Algorismes sobre grafs

- Cicles negatius
- Hi ha dos tipus de grafs que no tenen cicles negatius: sense pesos negatius i sense cicles
- El primer és directe. Per resoldre el camí mínim en acíclic grafs dirigits negatius:
 - Linealitzar usant DFS
 - Temps lineal!
- Si posem els negatius dels pesos podem trobar els camins de longitud màxima

Algoritmos sobre grafos

Linearizar usando DFS

procedure dag-shortest-paths(G, l, s)

Input: Dag $G = (V, E)$;

 edge lengths $\{l_e : e \in E\}$; vertex $s \in V$

Output: For all vertices u reachable from s , $\text{dist}(u)$ is set
 to the distance from s to u .

for all $u \in V$:

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

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

$\text{dist}(s) = 0$

Linearize G

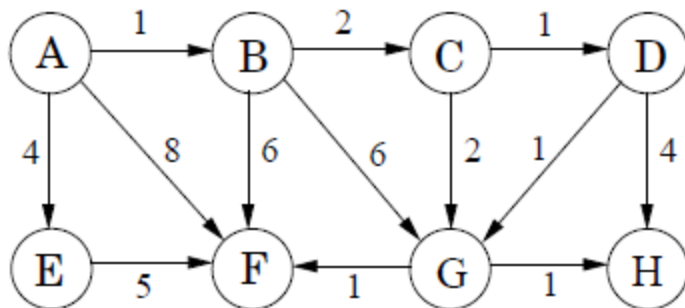
for each $u \in V$, in linearized order:

 for all edges $(u, v) \in E$:

 update(u, v)

Algorismes sobre grafs

- Exercicis: Dijkstra
- Començant a A: dibuixa la taula de distàncies immediates a tots els nodes a cada iteració.
- Mostra l'arbre de camins mínims



Algorismes sobre grafs

- Exercici: el mateix amb Bellman-Ford

