

Drumuri minime de sursă unică în grafuri aciclice DAG (fără circuite)

Drumuri minime de sursă unică în grafuri aciclice

► Ipoteze:

- Graful nu conține circuite
- Arcele pot avea și cost negativ

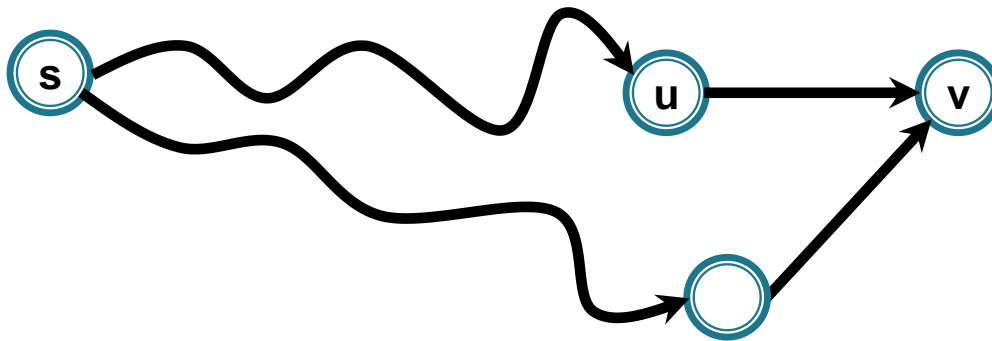
Drumuri minime de sursă unică în grafuri aciclice

► Amintim:

Când considerăm un vârf v , pentru a calcula $d(s,v)$ ar fi util să știm deja $\delta(s,u)$ pentru orice u cu $uv \in E$

- atunci putem calcula distanțele după relația

$$\delta(s,v) = \min\{\delta(s,u) + w(u,v) \mid uv \in E\}$$



Drumuri minime de sursă unică în grafuri aciclice

► Amintim:

Când considerăm un vârf v , pentru a calcula $d(s,v)$ ar fi util să știm deja $d(s,u)$ pentru orice u cu $uv \in E \Rightarrow$

- Ar fi utilă o ordonare a vârfurilor astfel încât dacă $uv \in E$, atunci u se află înaintea lui v



O astfel de ordonare există dacă graful nu conține circuite = sortarea topologică

Drumuri minime de sursă unică în grafuri
aciclice DAG
(fără circuite)

Pseudocod

Drumuri minime de sursă unică în grafuri aciclice

- ▶ Considerăm vârfurile în ordinea dată de sortarea topologică
 - Pentru fiecare vârf u relaxăm arcele uv către vecinii săi (pentru a găsi drumuri noi către aceștia)

Drumuri minime de sursă unică în grafuri aciclice

s - vârful de start

//initializam distante - ca la Dijkstra

pentru fiecare $u \in V$ executa

$d[u] = \infty$; tata[u]=0

$d[s] = 0$

//determinăm o sortare topologică a vârfurilor

SortTop = sortare_topologica(G)

pentru fiecare $u \in \text{SortTop}$

pentru fiecare $uv \in E$ executa

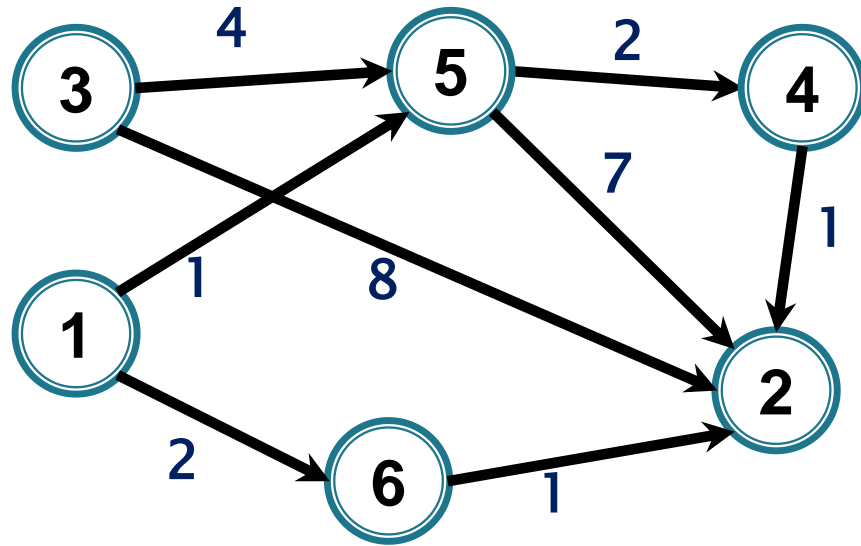
daca $d[u] + w(u, v) < d[v]$ atunci //relaxam uv

$d[v] = d[u] + w(u, v)$

tata[v] = u

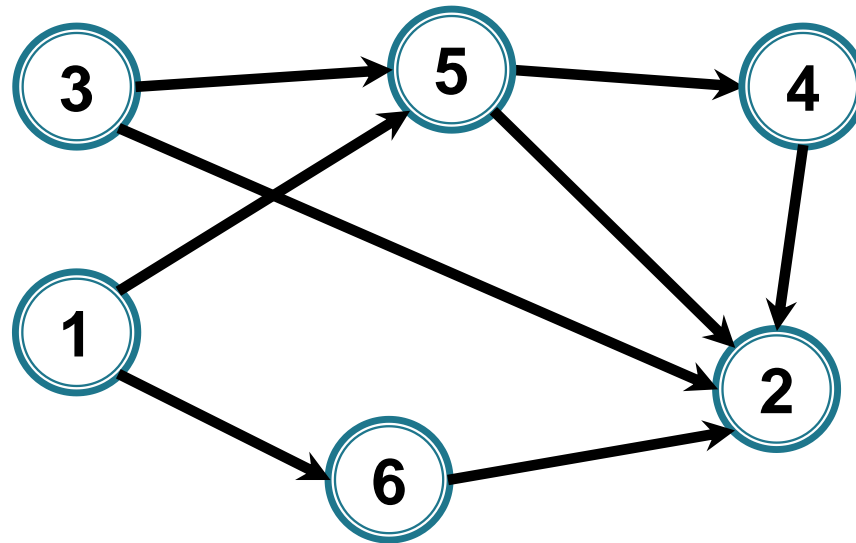
scrie d, tata

Exemplu



Drumuri minime de sursă unică în grafuri aciclice

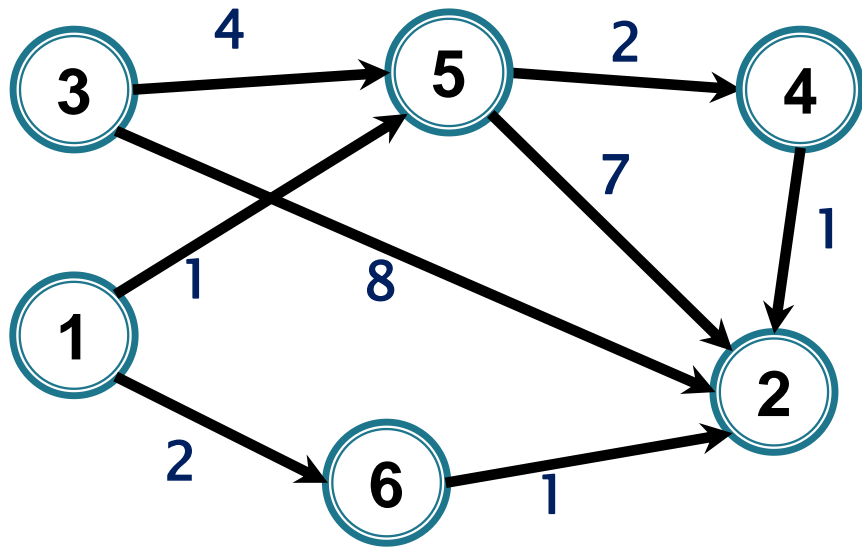
- ▶ Etapa 1 – determinăm o ordonare topologică a vârfurilor



Sortare topologică: **1 3 6 5 4 2**

Drumuri minime de sursă unică în grafuri aciclice

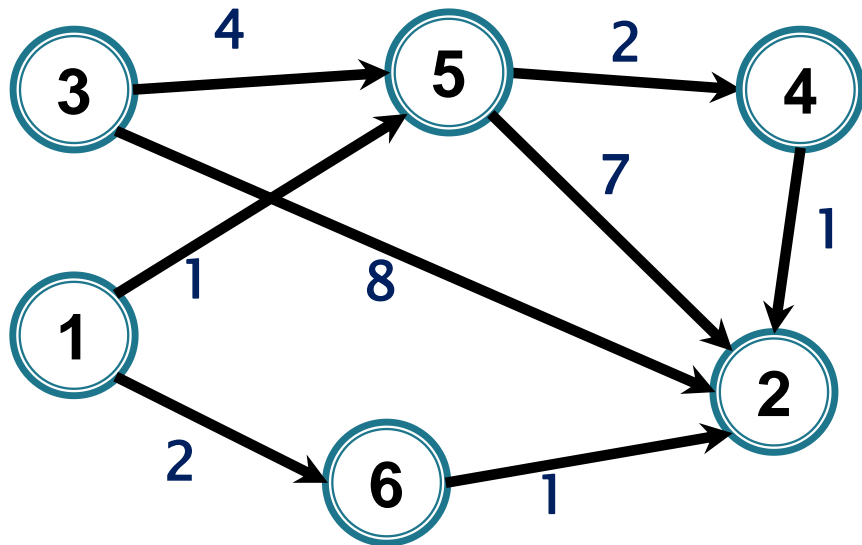
- ▶ Etapa 2 – parcurgem vârfurile în ordinea dată de sortarea topologică și relaxăm pentru fiecare vârf arcele care ies din acesta



Sortare topologică

1, 3, 6, 5, 4, 2

--	--	--	--	--	--



Sortare topologică

1, 3, 6, 5, 4, 2

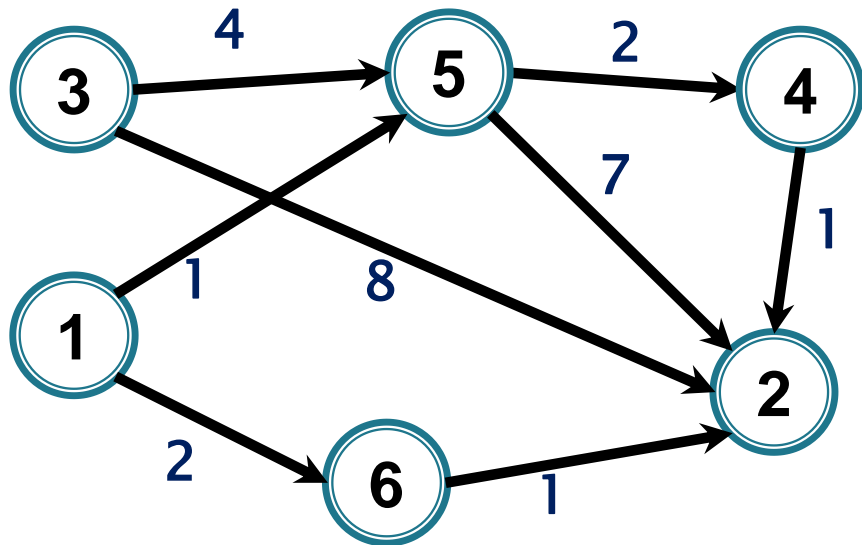
s=3 - vârf de start

Ordine de calcul distanțe:

1, 3, 6, 5, 4, 2

d/tata

	¹	²	³	⁴	⁵	⁶
	[$\infty/0$,	$\infty/0$,	$0/0$,	$\infty/0$,	$\infty/0$,	$\infty/0$]
u = 1:	[$\infty/0$,	$\infty/0$,	$0/0$,	$\infty/0$,	$\infty/0$,	$\infty/0$]
u = 3:	[$\infty/0$,	8/3 ,	$0/0$,	$\infty/0$,	4/3 ,	$\infty/0$]
u = 6:	[$\infty/0$,	8/3 ,	$0/0$,	$\infty/0$,	4/3 ,	$\infty/0$]
u = 5:	[$\infty/0$,	8/3 ,	$0/0$,	6/5 ,	4/3 ,	$\infty/0$]
u = 4:	[$\infty/0$,	7/4 ,	$0/0$,	6/5 ,	4/3 ,	$\infty/0$]
u = 2:	[$\infty/0$,	7/4 ,	$0/0$,	6/5 ,	4/3 ,	$\infty/0$]



Sortare topologică

1, 3, 6, 5, 4, 2

s=3 - vârf de start

Ordine de calcul distanțe:

1, 3, 6, 5, 4, 2

d/tata

1

2

3

4

5

6

Soluție

[$\infty/0$, 7/4, 0/0, 6/5, 4/3, $\infty/0$]

Un drum minim de la 3 la 2?

Drumuri minime de sursă unică în grafuri aciclice

► Observație

- Este suficient să considerăm în ordonarea topologică doar vârfurile accesibile din s
- În exemplu – fără 1 și 6

Complexitate



Drumuri minime de sursă unică în grafuri aciclice

s - vârful de start

//initializam distante - ca la Dijkstra

pentru fiecare $u \in V$ executa

$d[u] = \infty$; tata[u]=0

$d[s] = 0$

//determinăm o sortare topologică a vârfurilor

SortTop = sortare_topologica(G)

pentru fiecare $u \in \text{SortTop}$

pentru fiecare $uv \in E$ executa

daca $d[u] + w(u, v) < d[v]$ atunci //relaxam uv

$d[v] = d[u] + w(u, v)$

tata[v] = u

scrie d, tata

Drumuri minime de sursă unică în grafuri aciclice

Complexitate

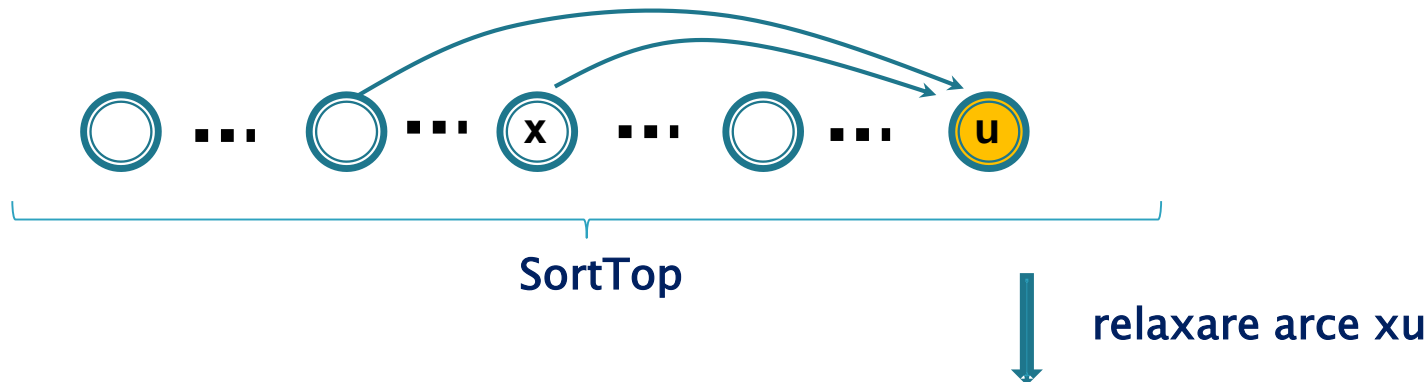
- ▶ Inițializare $\rightarrow O(n)$
 - ▶ Sortare topologică $\rightarrow O(m+n)$
 - ▶ m * relaxare uv $\rightarrow O(m)$
-
- $O(m + n)$

Corectitudine

Drumuri minime de sursă unică în grafuri aciclice

- ▶ Algoritmul funcționează corect și dacă există arce cu cost negativ – Inducție după numărul de iterații

Când algoritmul ajunge la vârful u avem



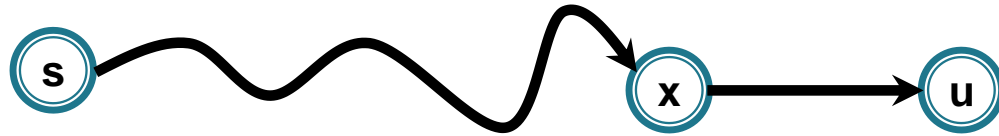
$$\begin{aligned} d[u] &= \min\{ d[x] + w(x,u) \mid xu \in E, \text{ x înaintea lui u în SortTop } \} \\ &= \min\{ d[x] + w(x,u) \mid xu \in E \} = \min\{ \delta(s, x) + w(x,u) \mid xu \in E \} \\ &= \delta(s, u) \end{aligned}$$

↑
deja corect calculate (ipoteza inducție)

Drumuri minime de sursă unică în grafuri aciclice

Varianta 2 de demonstrație – similar Dijkstra

Fie P s-u drum minim și x predecesorul lui u pe acest drum.



- ▶ x este înaintea lui u în SortTop \Rightarrow (ip. inducție)

$$d[x] = \delta(s; x) = w([s \underline{P} x])$$

- ▶ după relaxarea arcului xu avem:

$$\begin{aligned} d[u] &\leq d[x] + w(xu) = w([s \underline{P} x]) + w(xu) = \\ &= w([s \underline{P} u]) = \delta(s; u) \end{aligned}$$

Dar $\delta(s; u) \leq d[u]$ (estimare superioară) $\Rightarrow \delta(s; u) = d[u]$

Aplicație – Drumuri critice

Drumuri critice

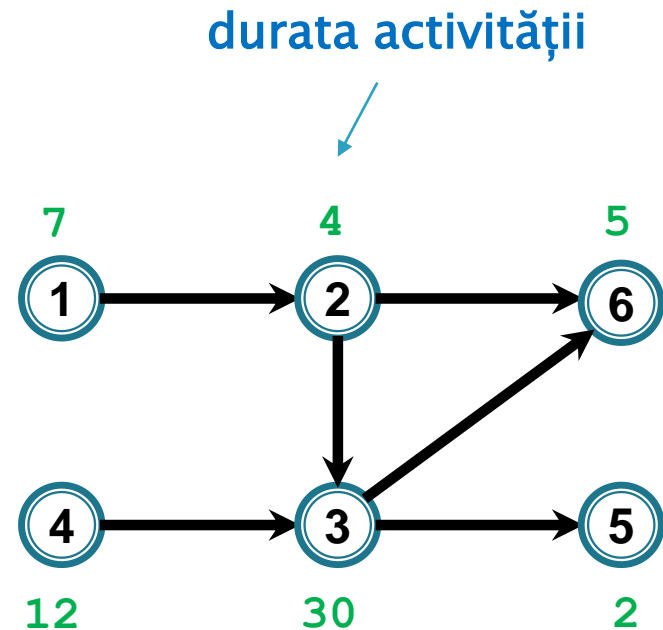
- ▶ Se cunosc pentru un proiect cu n activități, numerotate $1, \dots, n$:
 - durata fiecărei activități
 - perechi (i, j) = activitatea i trebuie să se încheie înainte să înceapă j
 - activitățile se pot desfășura și în paralel

Se cere: timpul minim de finalizare a proiectului (dacă începe la ora 0) + planificarea activităților

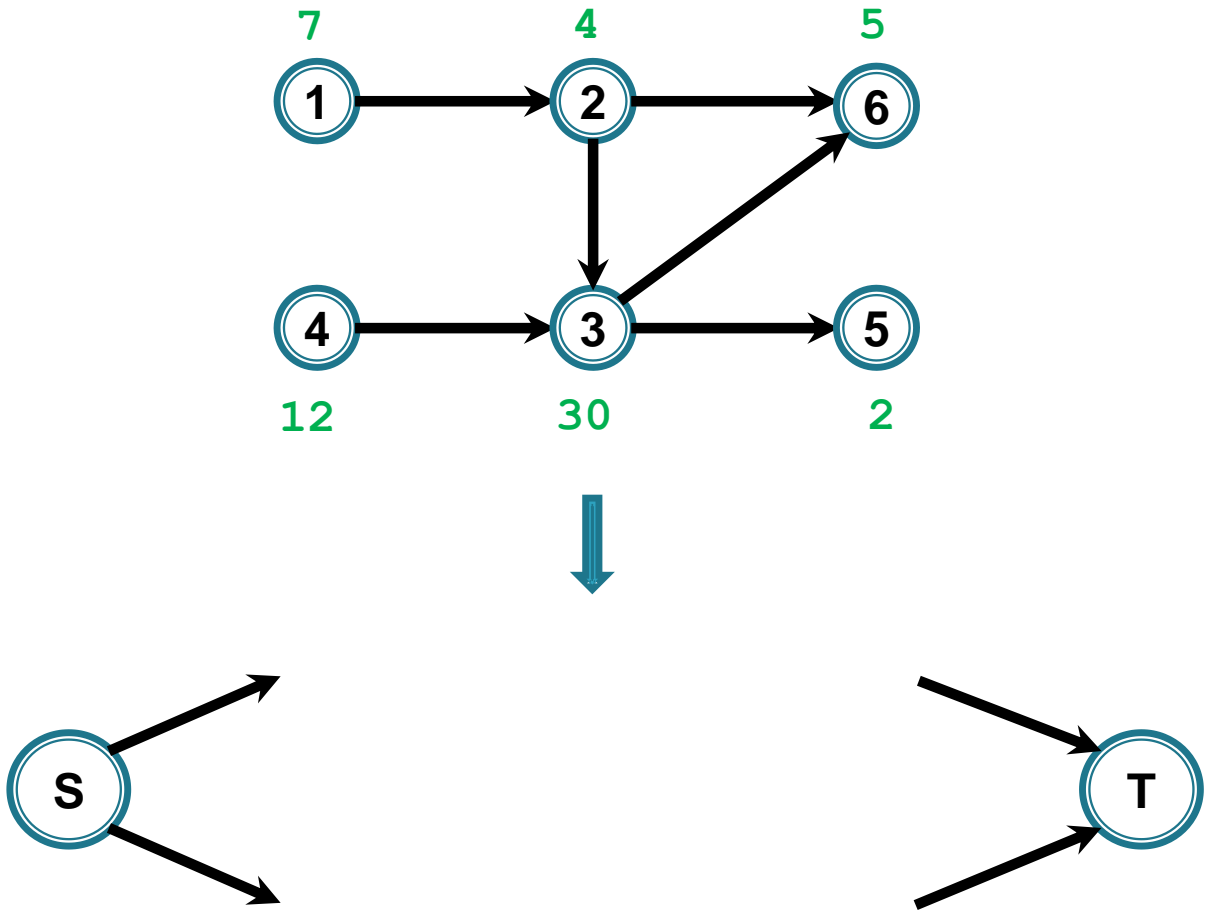
Drumuri critice

► $n = 6$

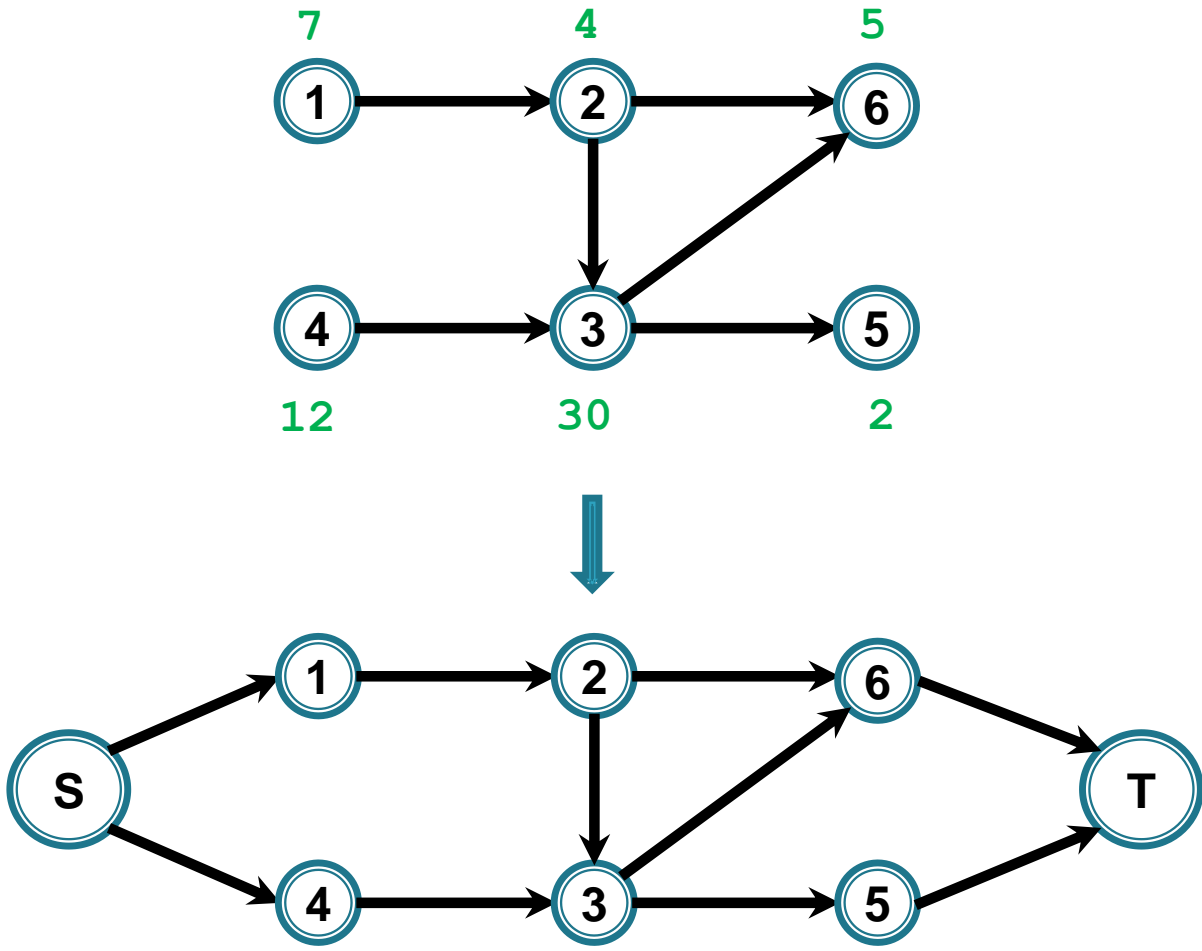
- Activitatea 1 – durata 7
- Activitatea 2 – durata 4
- Activitatea 3 – durata 30
- Activitatea 4 – durata 12
- Activitatea 5 – durata 2
- Activitatea 6 – durata 5
- (1, 2)
- (2, 3)
- (3, 6)
- (4, 3)
- (2, 6)
- (3, 5)



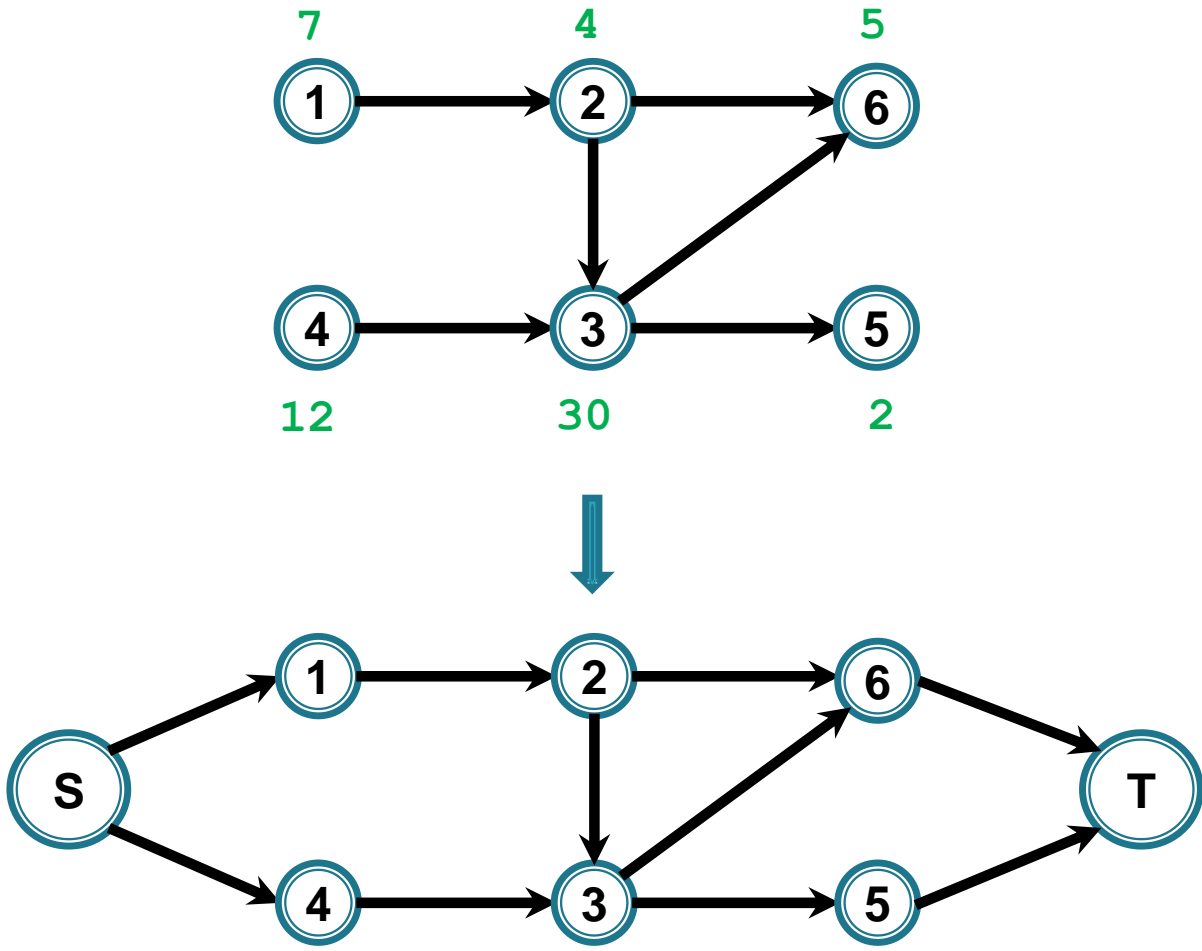
Drumuri critice



Drumuri critice

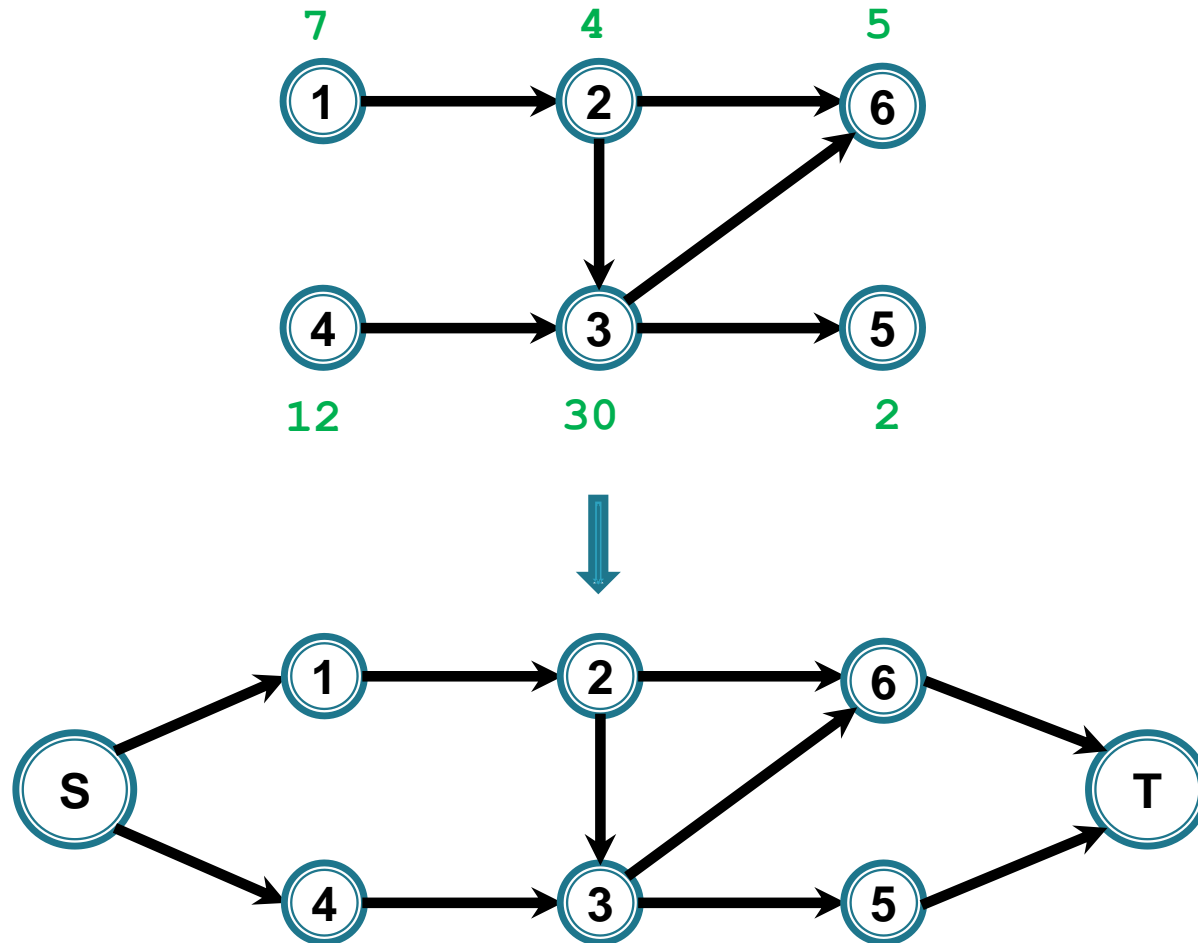


Drumuri critice



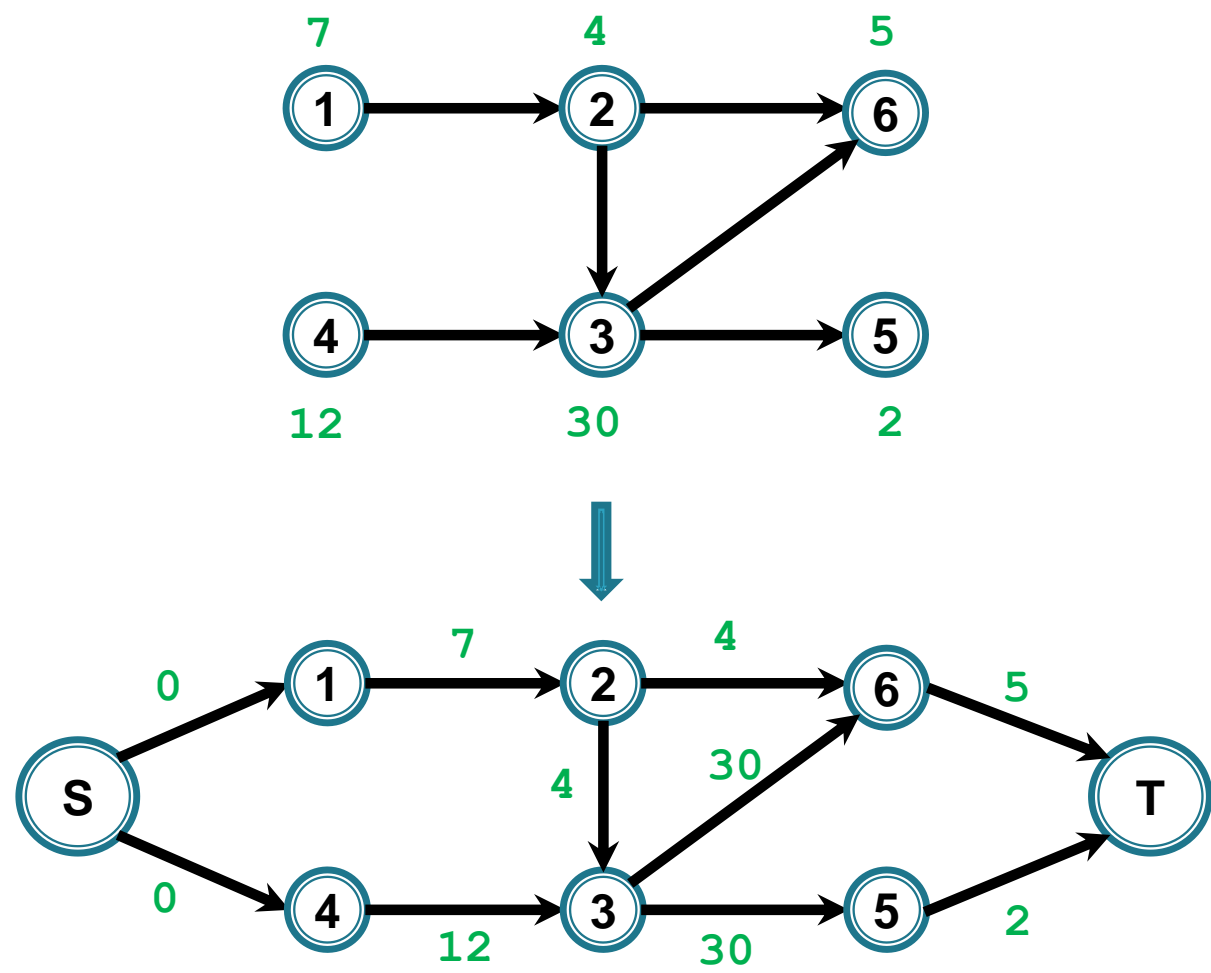
$w(i,j) = ?$

Drumuri critice



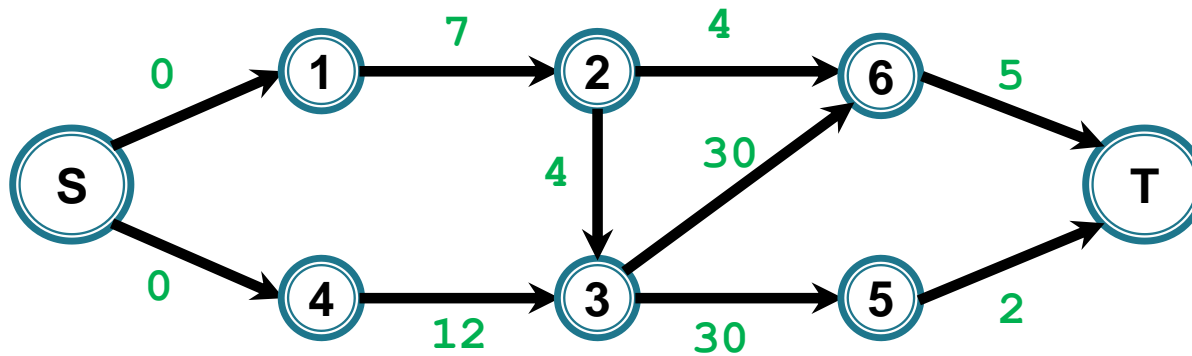
$w(i,j)$ = **durata activității i**
= întârzierea minimă între începutul activității i și începutul activității j
(mai general)

Drumuri critice



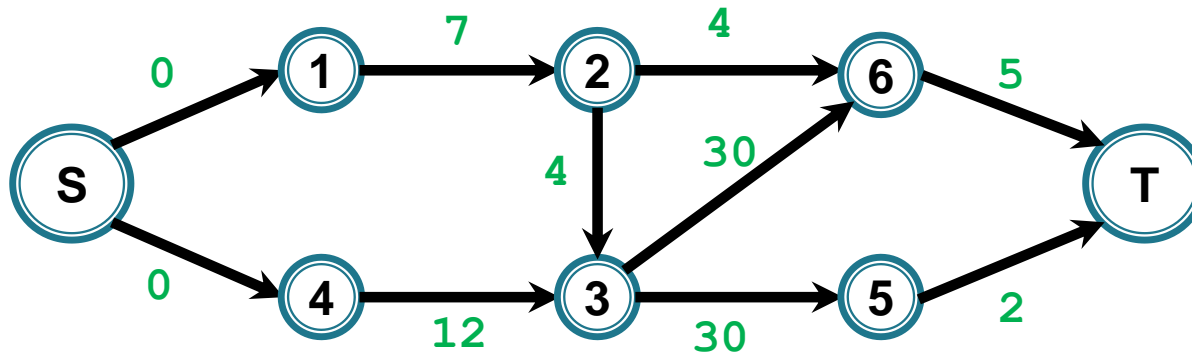
$w(i,j)$ = **durata activității i**
= întârzierea minimă între începutul activității i și începutul activității j
(mai general)

Drumuri critice



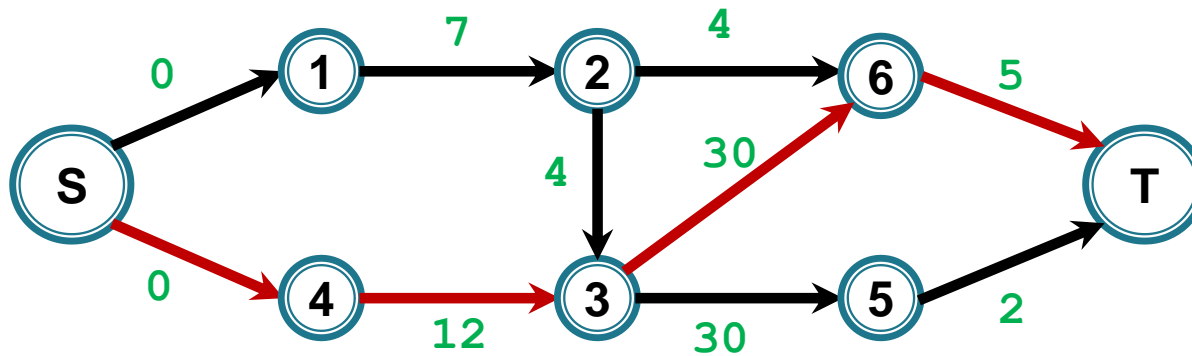
Timpul minim de finalizare a proiectului = ?

Drumuri critice

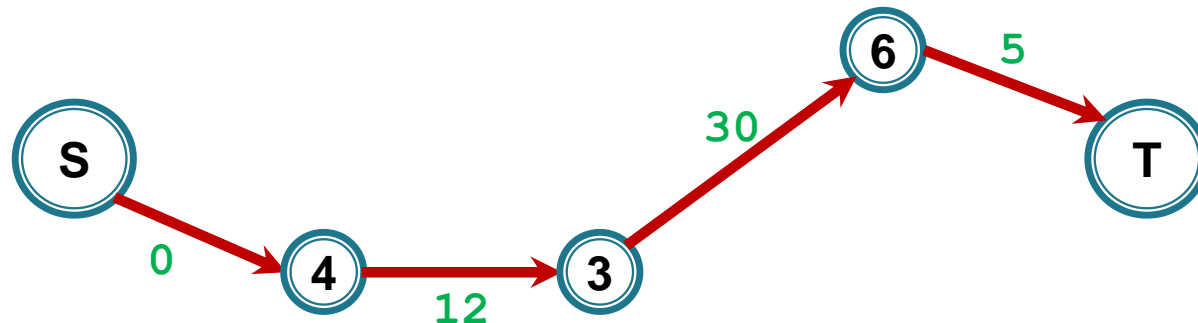


Timpul minim de finalizare a proiectului = **costul maxim al unui drum de la S la T**

Drumuri critice



Timpul minim de finalizare a proiectului = **costul maxim al unui drum de la S la T**



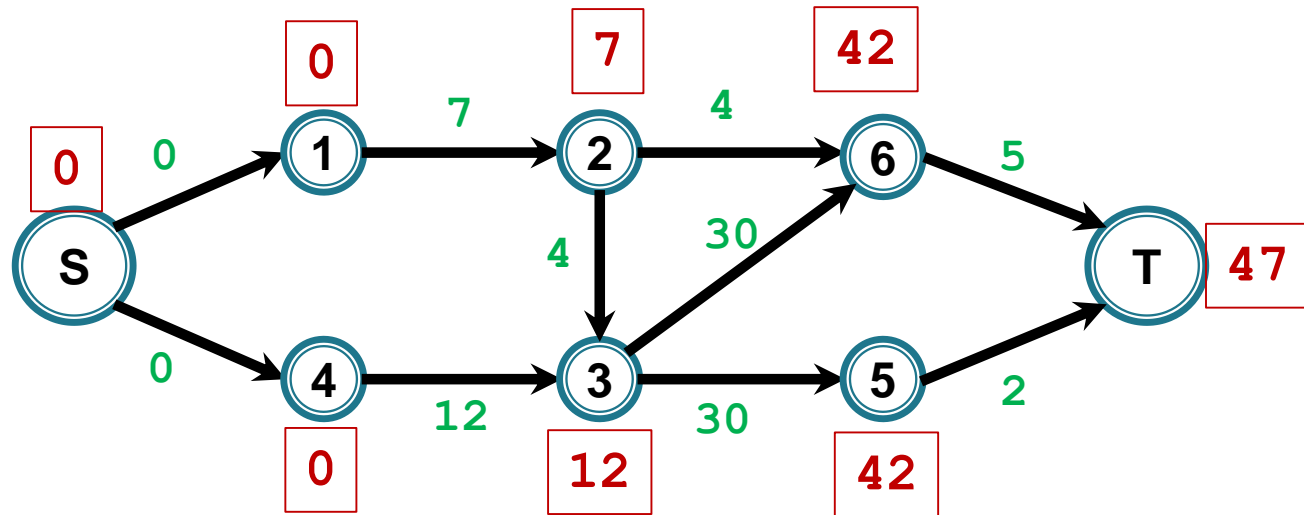
Drum CRITIC

Drumuri critice

- ▶ Durata minimă a proiectului = costul maxim al unui drum de la S la T
 - **Drum critic** = drum de cost maxim de la S la T
 - Orice întârziere în desfășurarea unei activități de pe acest drum duce la creșterea timpului de terminare al proiectului
 - PERT/CPM – Program Evaluation and Review Technique / Critical Path Method

Drumuri critice

- ▶ Durata minimă a proiectului = costul maxim al unui drum de la S la T
- ▶ Timpul minim de început al unei activități u = costul maxim al unui drum de la S la u



activitatea 1: intervalul de desfășurare (0,7)

activitatea 3: intervalul de desfășurare (12, 42)

Drumuri critice



Putem modifica algoritmul de determinare de drumuri minime în grafuri aciclice a.î. să determine drumuri maxime (de cost maxim) de la S la celelalte vârfuri

- Problema este echivalentă cu a determina drumuri minime din S în graful în care înlocuim fiecare pondere $w(e)$ cu $-w(e)$
- Modificăm astfel doar inițializarea distanțelor (cu $-\infty$ în loc de $+\infty$) și **inversăm condiția de la relaxarea** arcelor pentru a calcula maxim în loc de minim
- **Corectitudine** – rezultă din corectitudinea algoritmului pentru drumul minim

Drumuri maxime de sursă unică în grafuri aciclice

s - vârful de start

//initializam distante - ca la Dijkstra

pentru fiecare $u \in V$ executa

$d[u] = -\infty$; tata[u]=0

$d[s] = 0$

//determinăm o sortare topologică a vârfurilor

SortTop = sortare_topologica(G)

pentru fiecare $u \in \text{SortTop}$

pentru fiecare $uv \in E$ executa

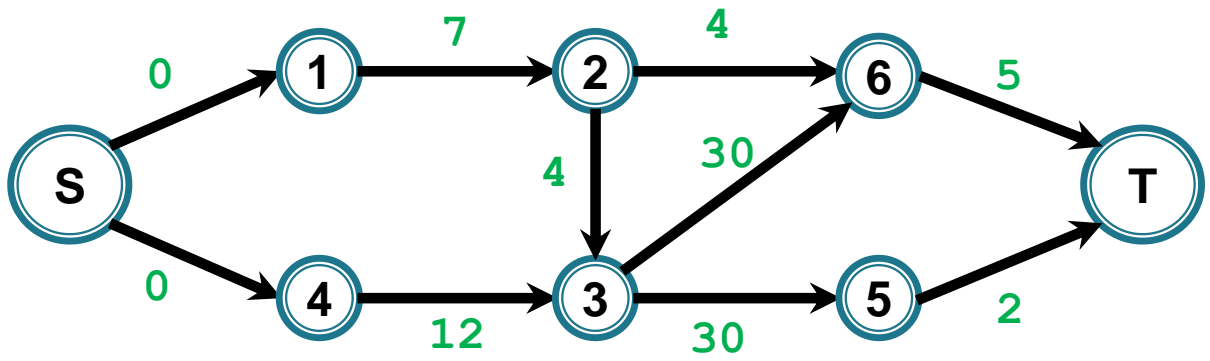
daca $d[u] + w(u, v) > d[v]$ atunci //relaxam uv

$d[v] = d[u] + w(u, v)$

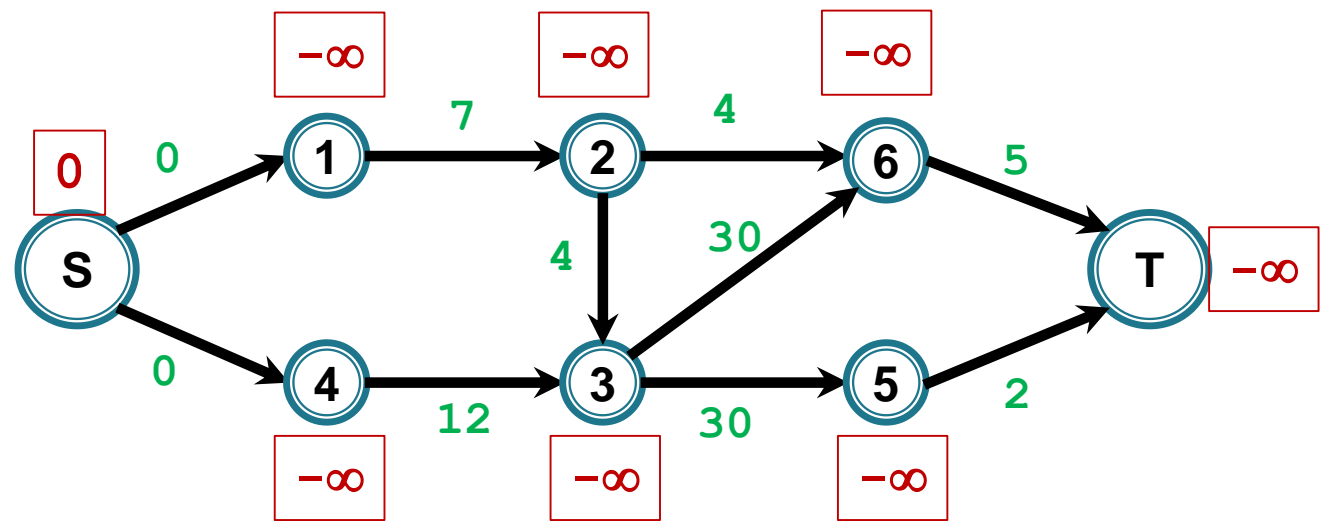
tata[v] = u

scrie d, tata

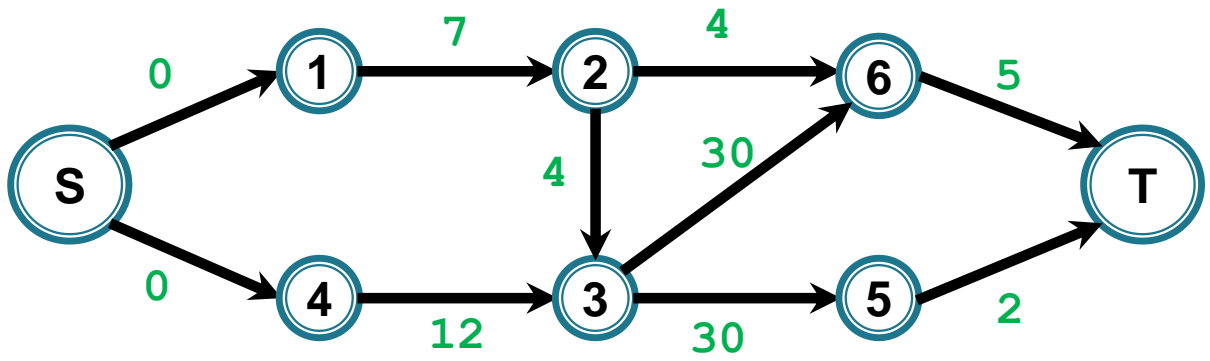
Drumuri critice



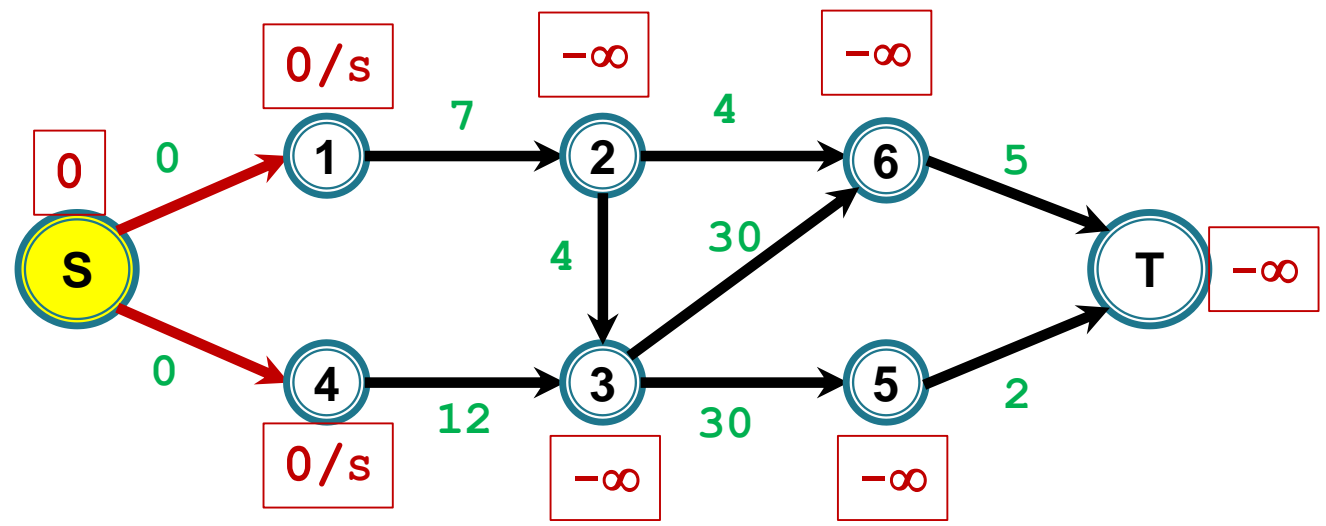
Ordine de calcul distanțe: **S**, 1, 4, 2, 3, 5, 6, T



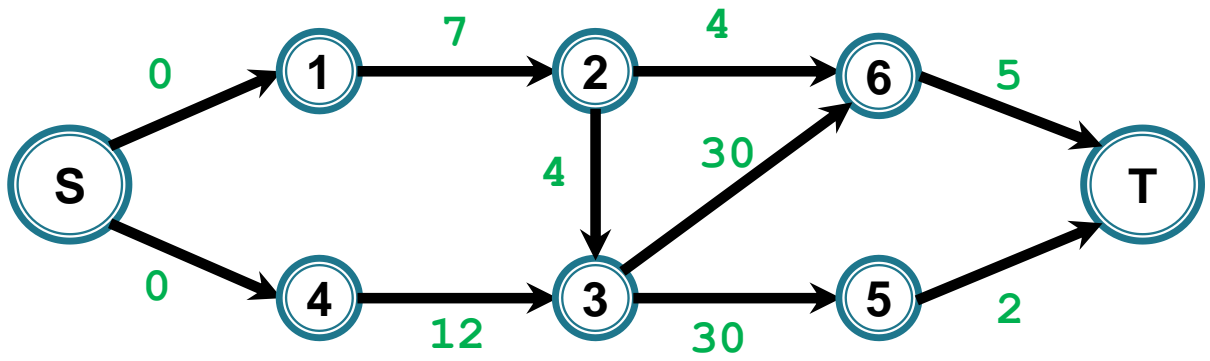
Drumuri critice



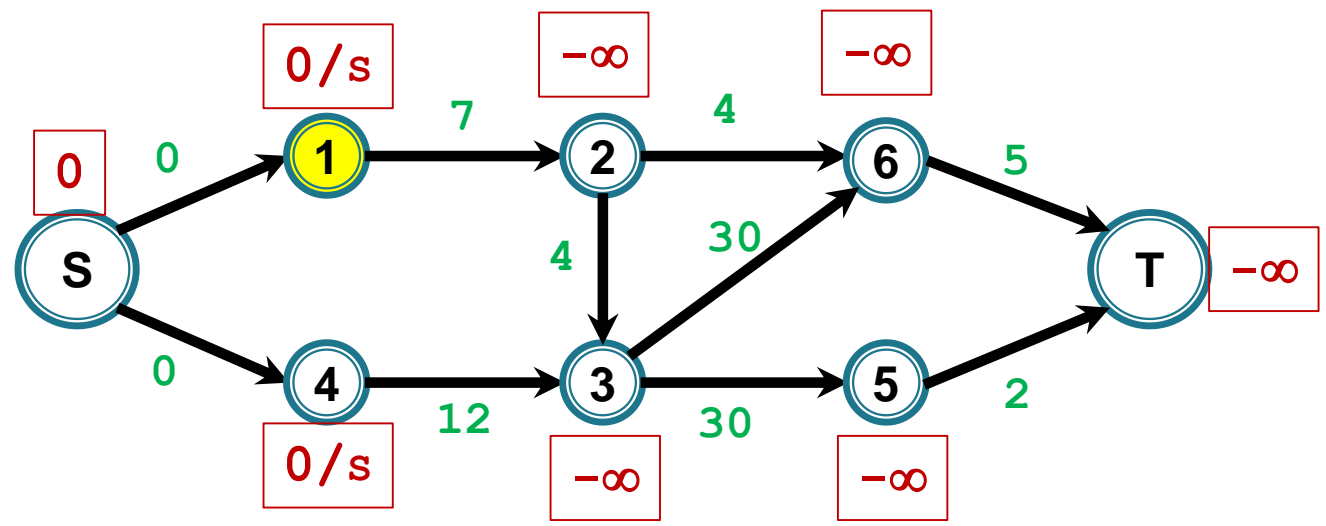
Ordine de calcul distanțe: **S**, 1, 4, 2, 3, 5, 6, T



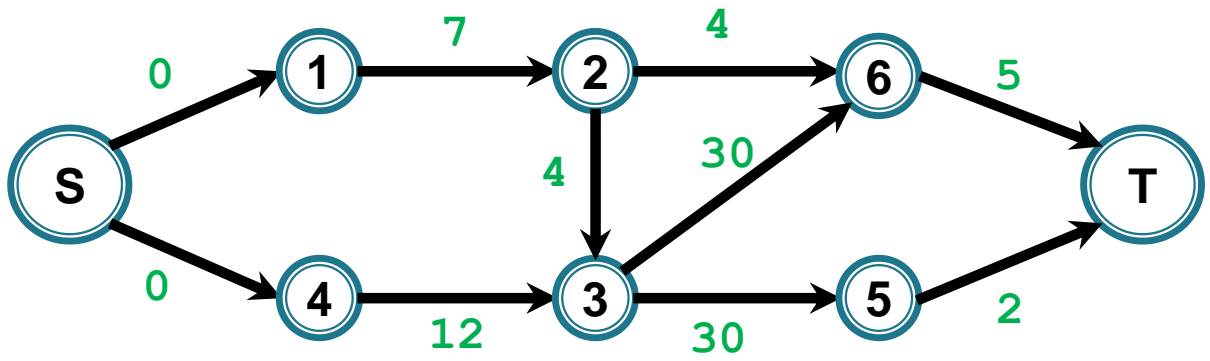
Drumuri critice



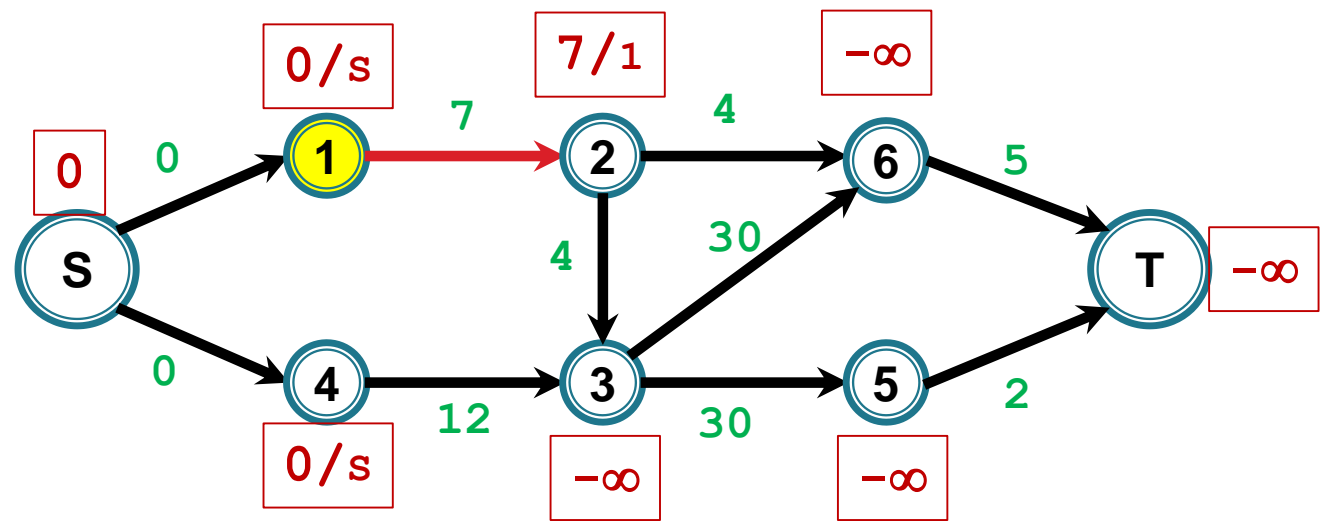
Ordine de calcul distanțe: S, 1, 4, 2, 3, 5, 6, T



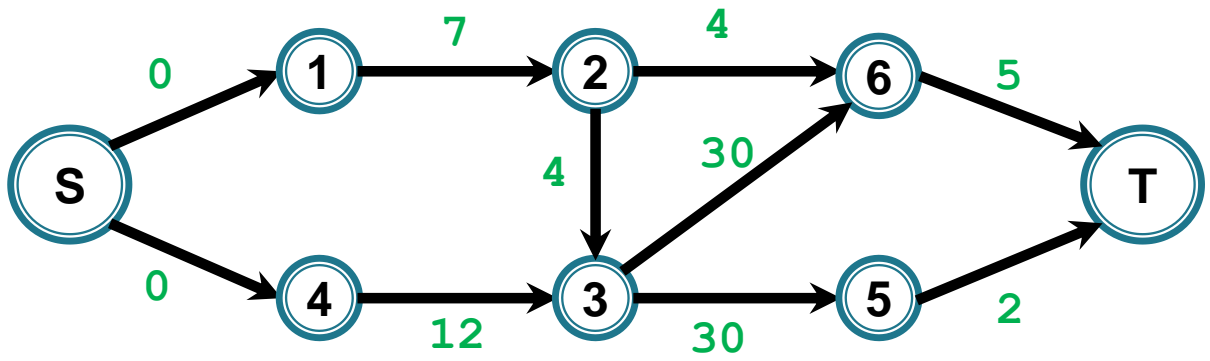
Drumuri critice



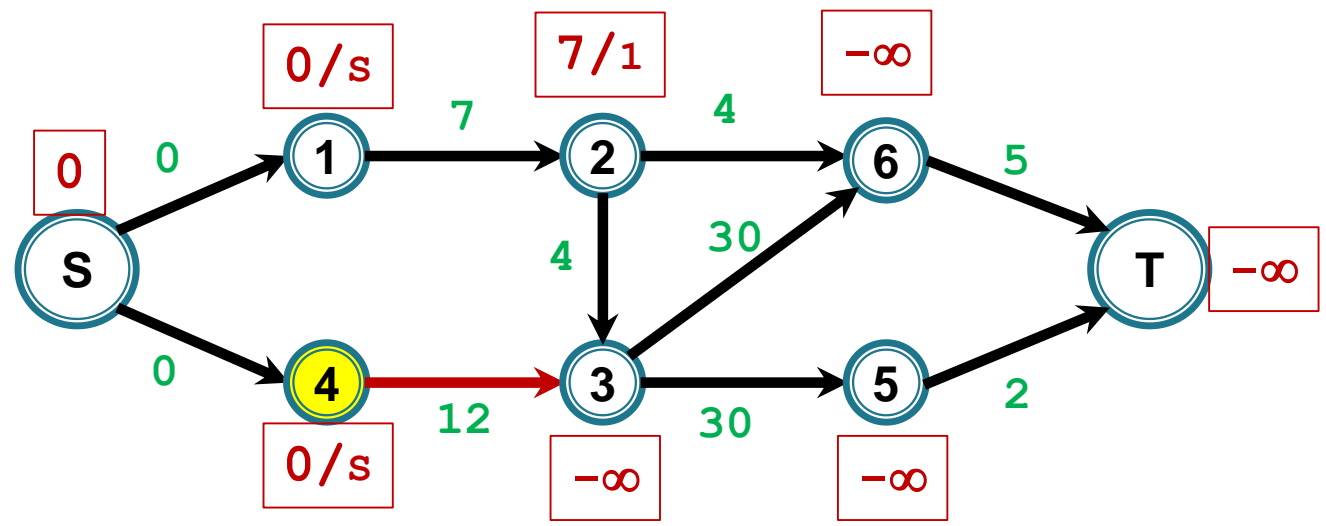
Ordine de calcul distanțe: S, 1, 4, 2, 3, 5, 6, T



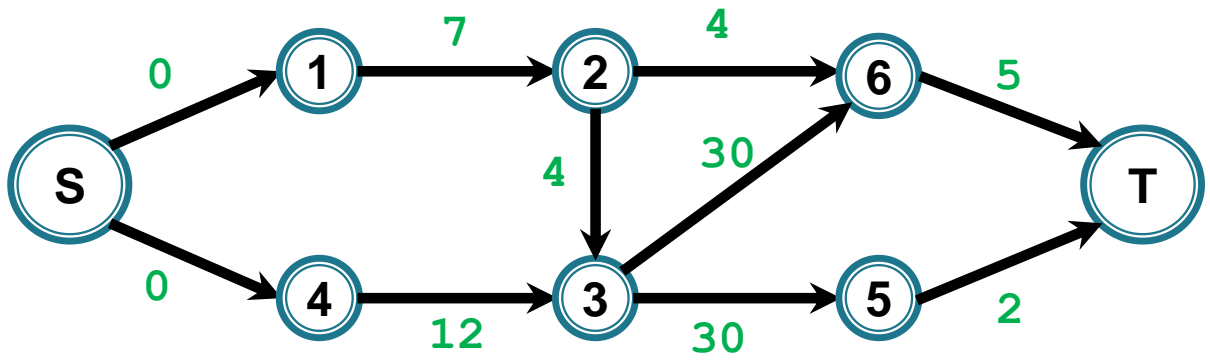
Drumuri critice



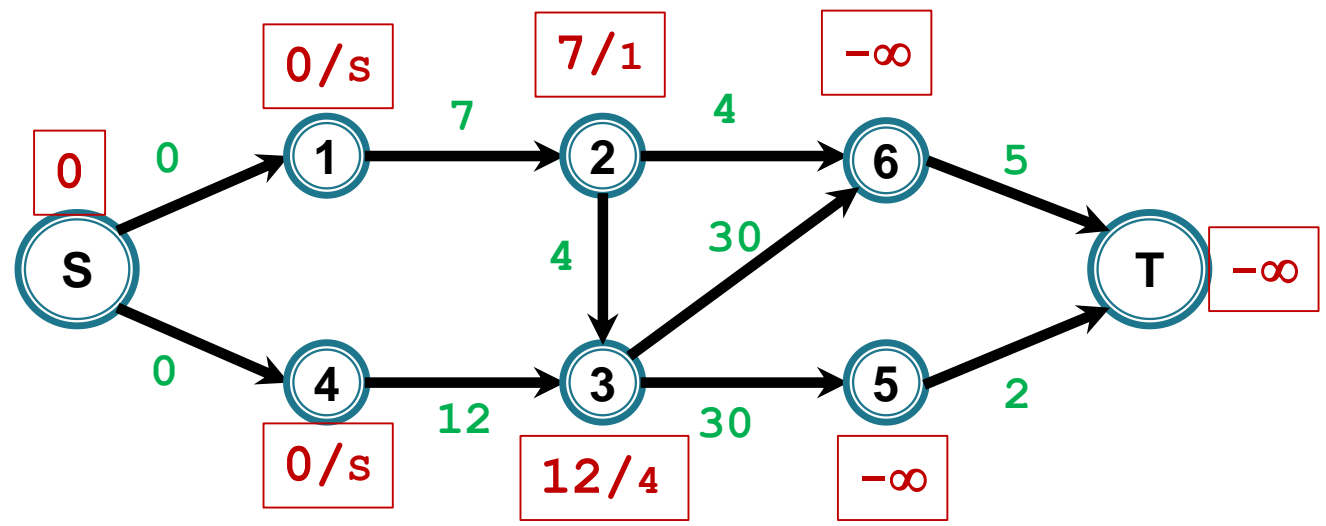
Ordine de calcul distanțe: S, 1, 4, 2, 3, 5, 6, T



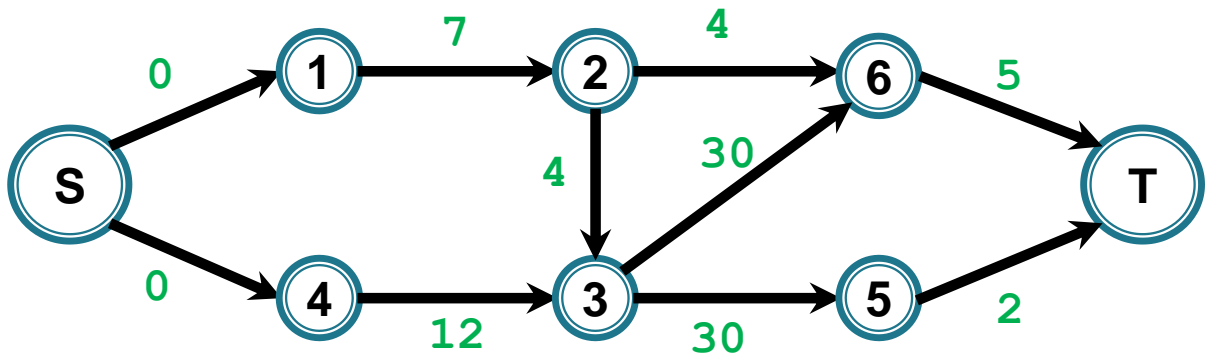
Drumuri critice



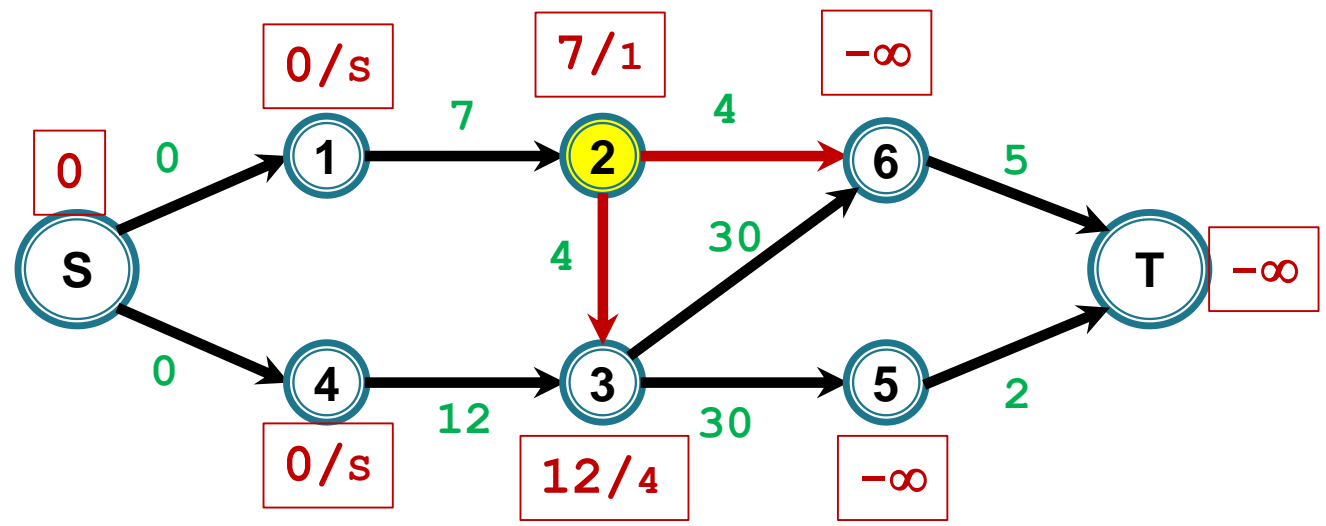
Ordine de calcul distanțe: S, 1, 4, 2, 3, 5, 6, T



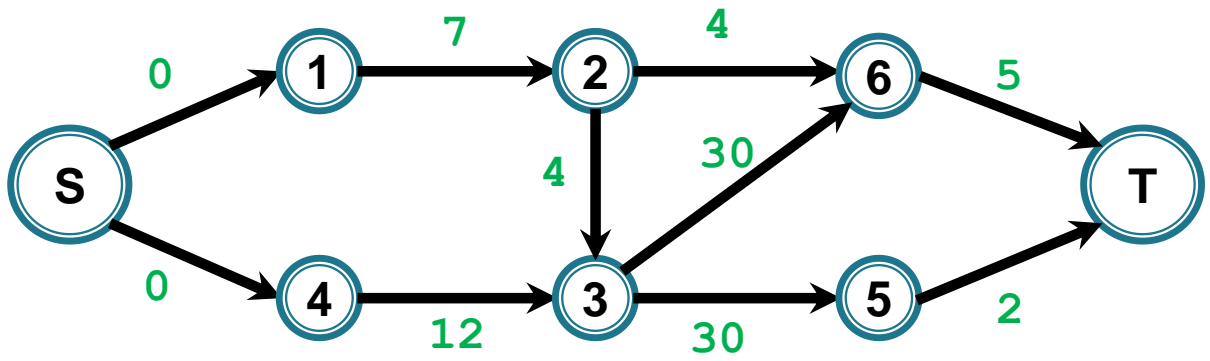
Drumuri critice



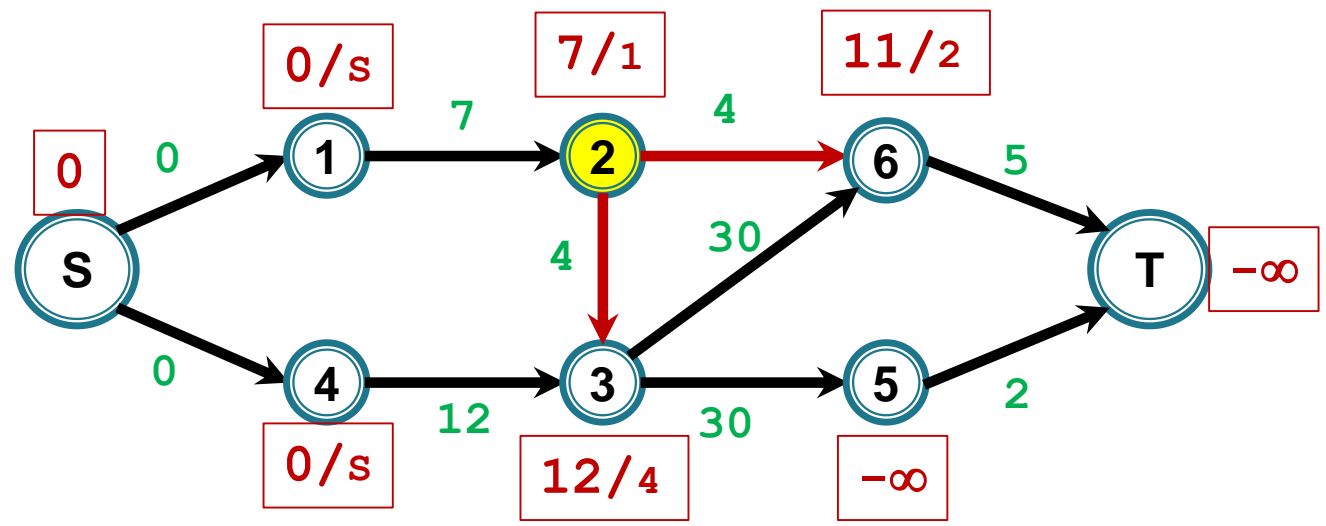
Ordine de calcul distanțe: S, 1, 4, 2, 3, 5, 6, T



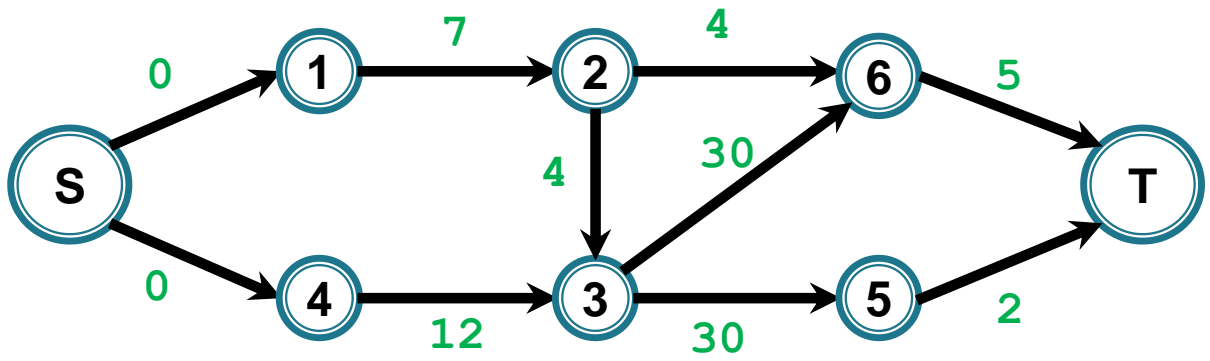
Drumuri critice



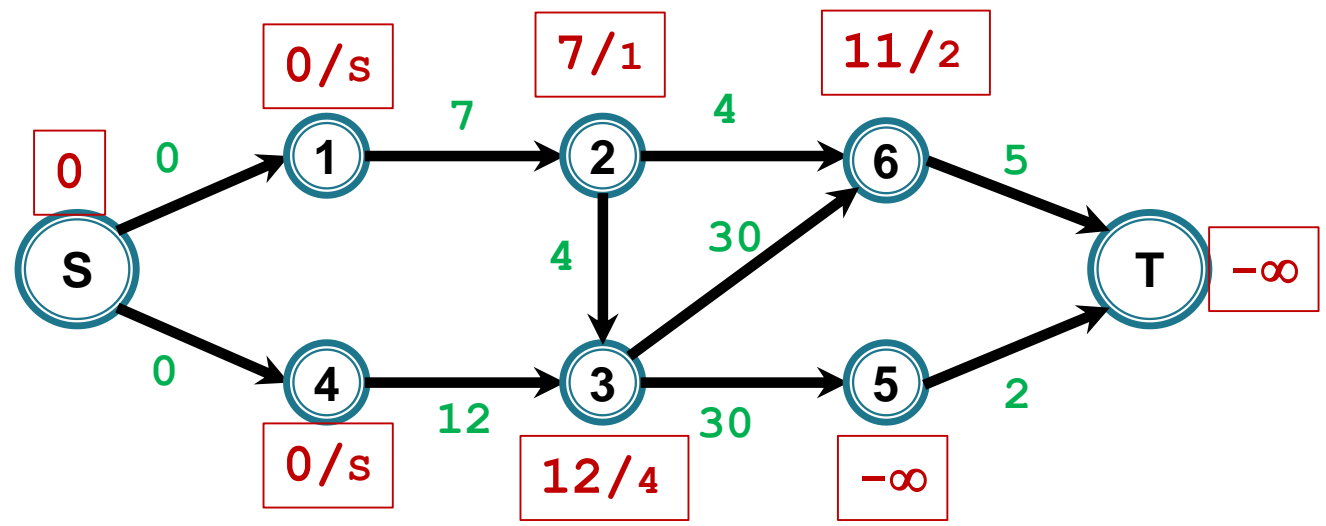
Ordine de calcul distanțe: S, 1, 4, 2, 3, 5, 6, T



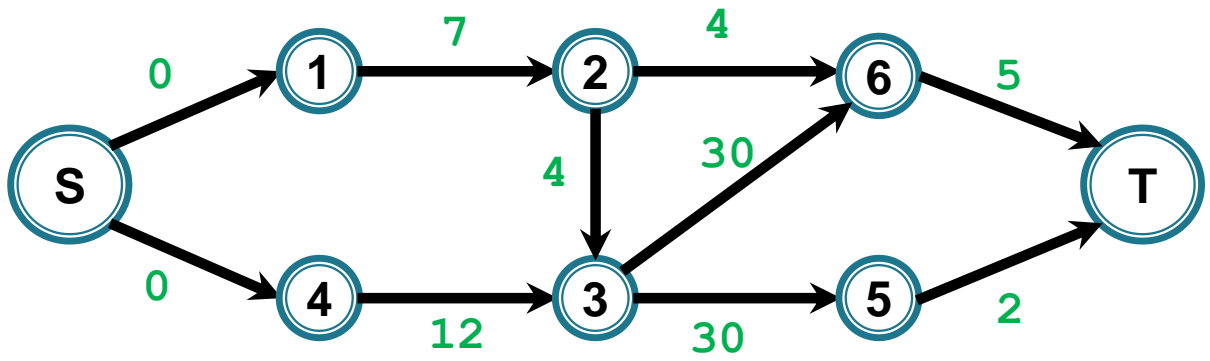
Drumuri critice



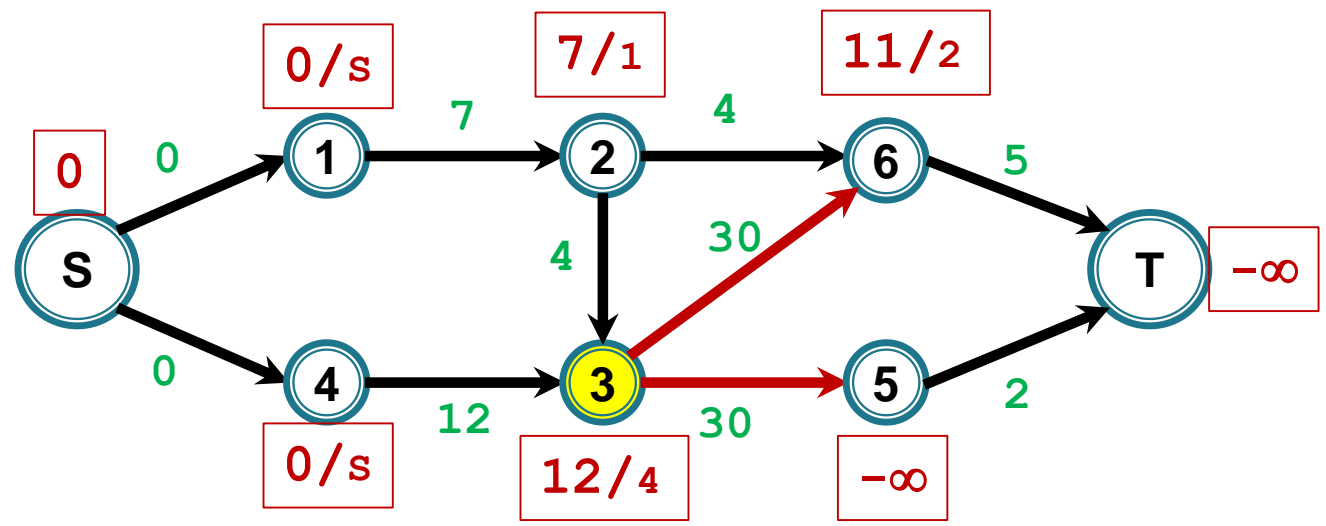
Ordine de calcul distanțe: S, 1, 4, 2, 3, 5, 6, T



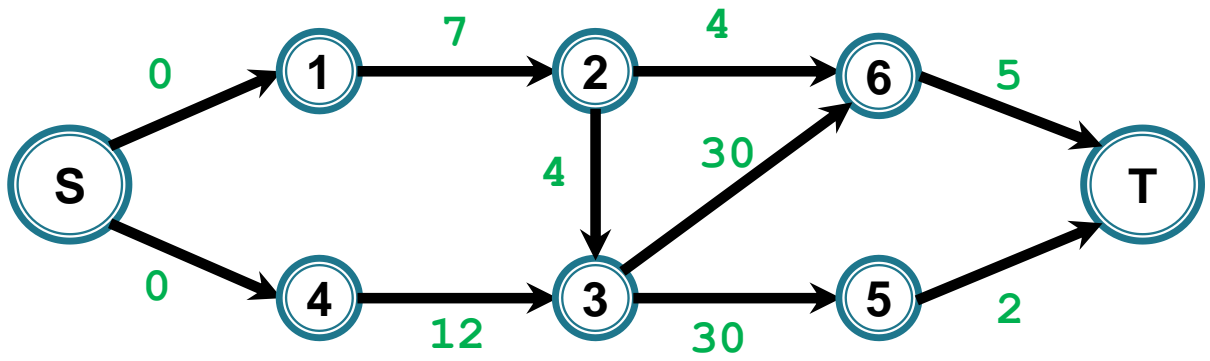
Drumuri critice



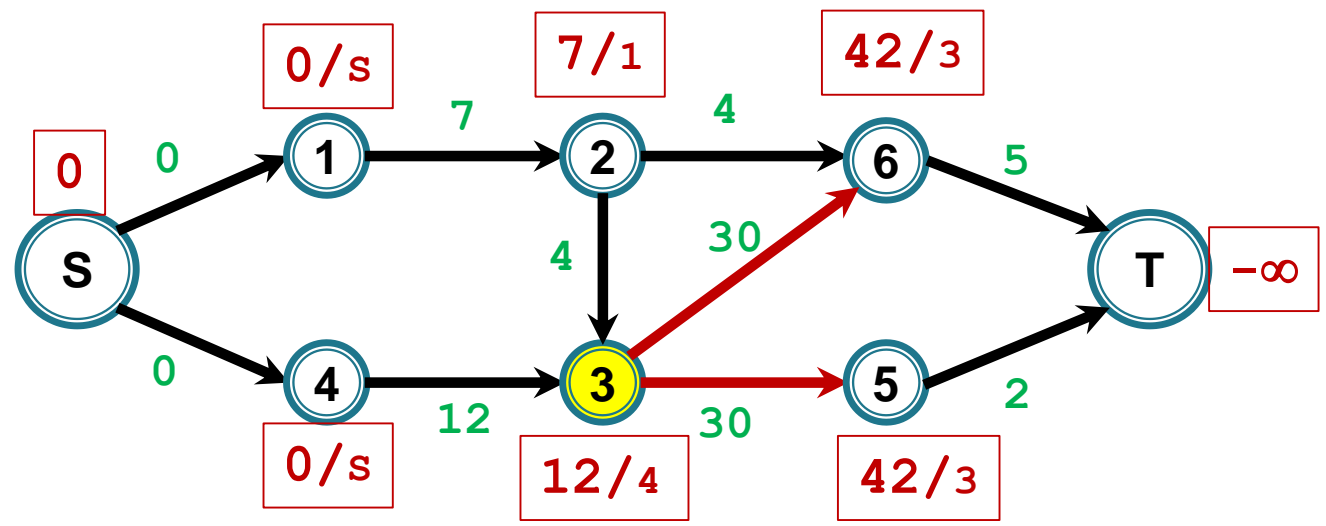
Ordine de calcul distanțe: S, 1, 4, 2, 3, 5, 6, T



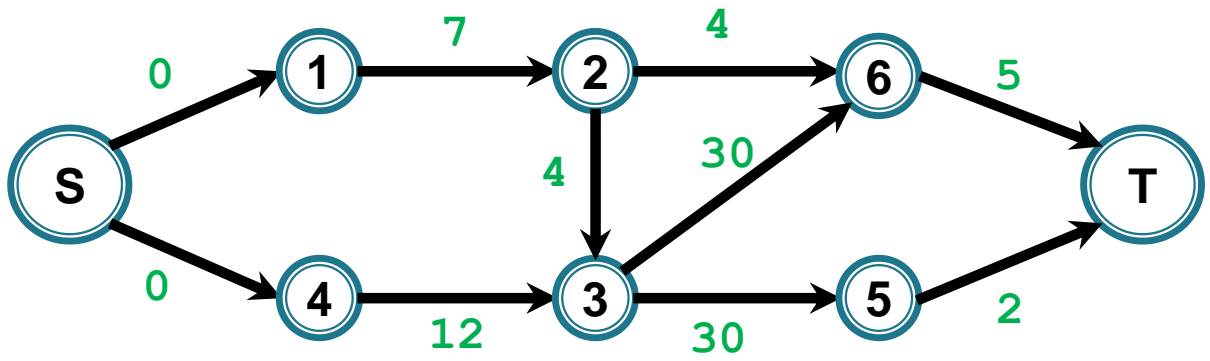
Drumuri critice



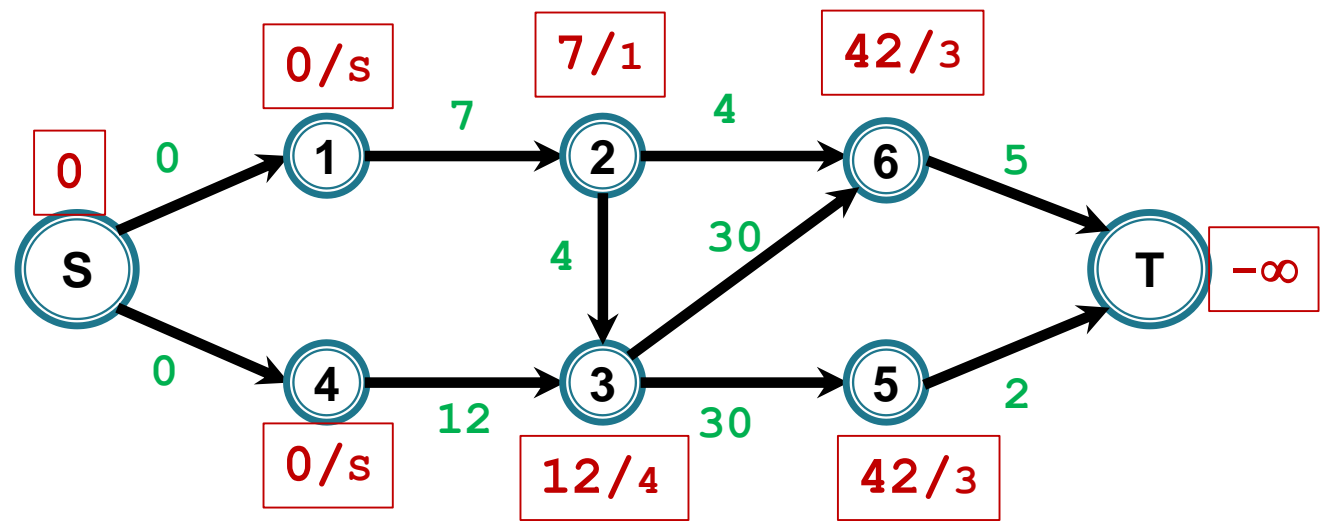
Ordine de calcul distanțe: S, 1, 4, 2, 3, 5, 6, T



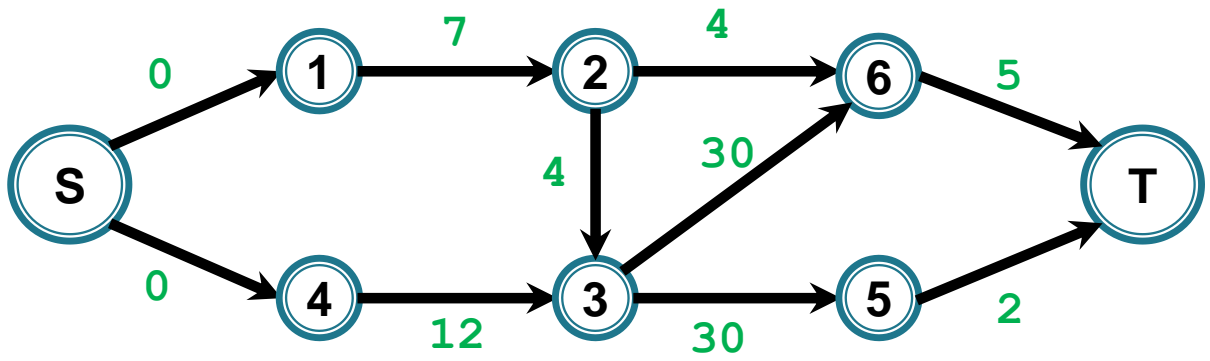
Drumuri critice



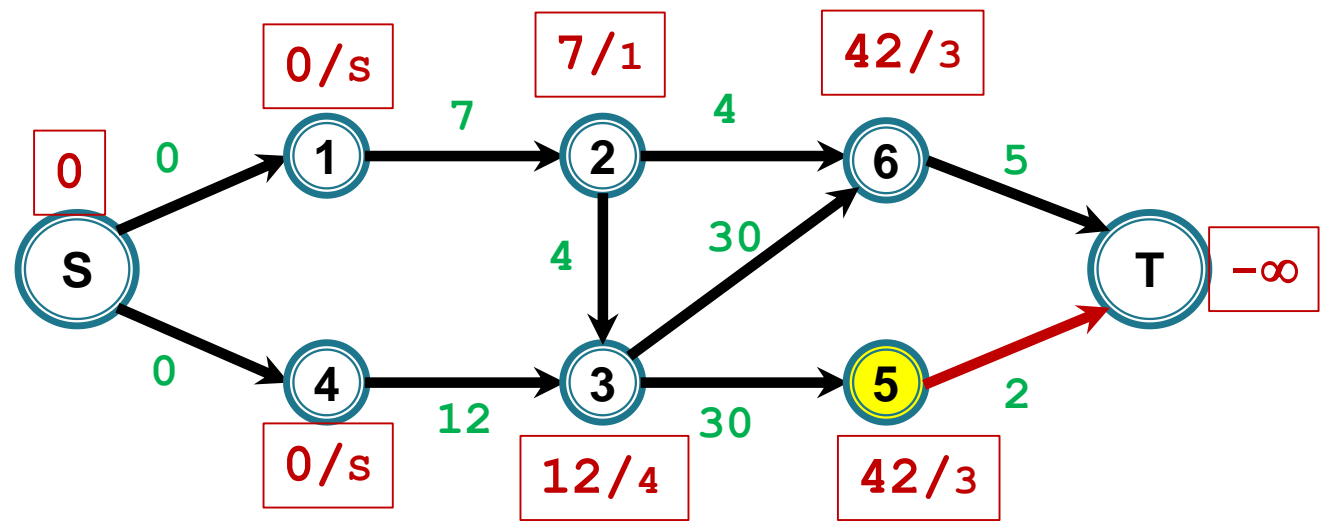
Ordine de calcul distanțe: S, 1, 4, 2, 3, 5, 6, T



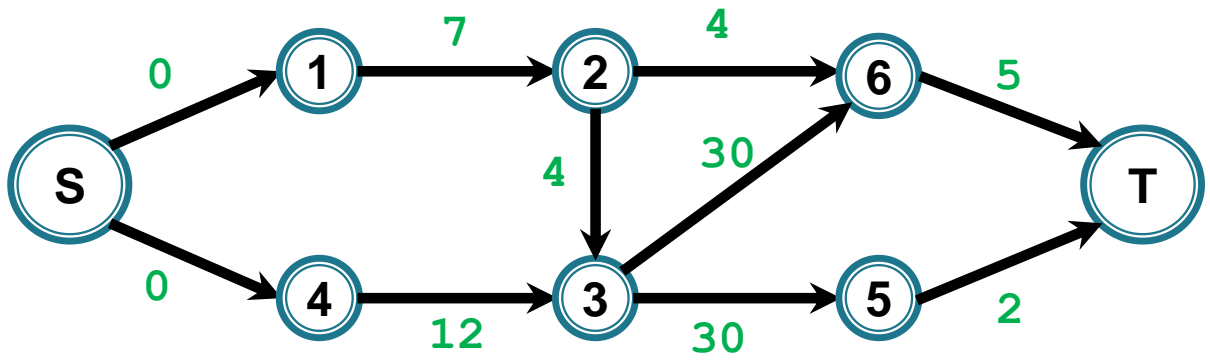
Drumuri critice



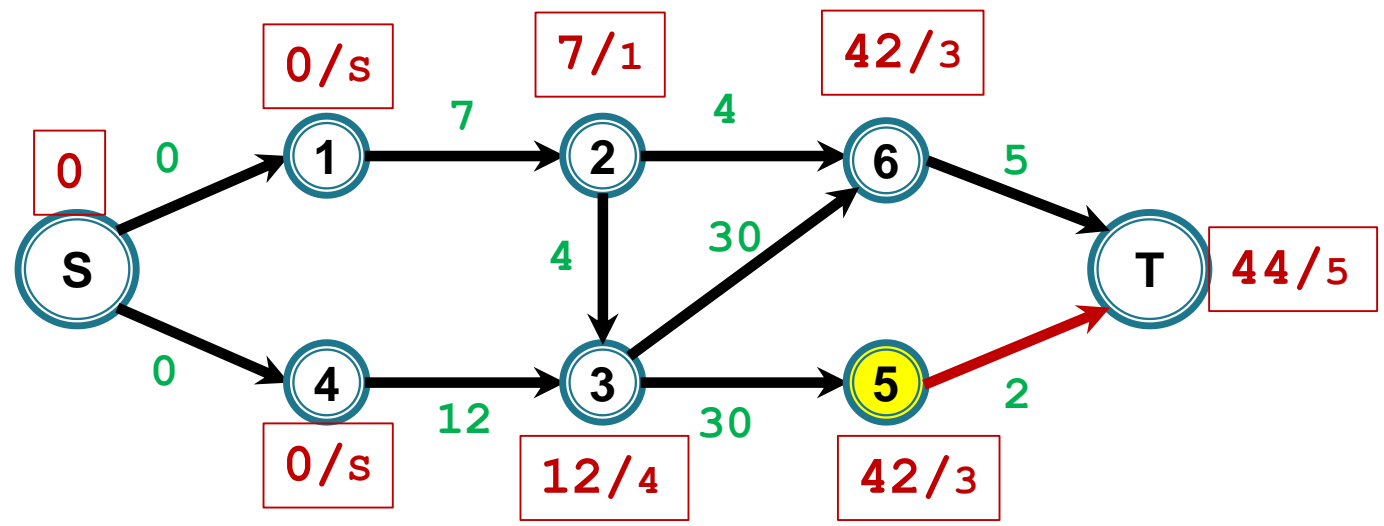
Ordine de calcul distanțe: S, 1, 4, 2, 3, 5, 6, T



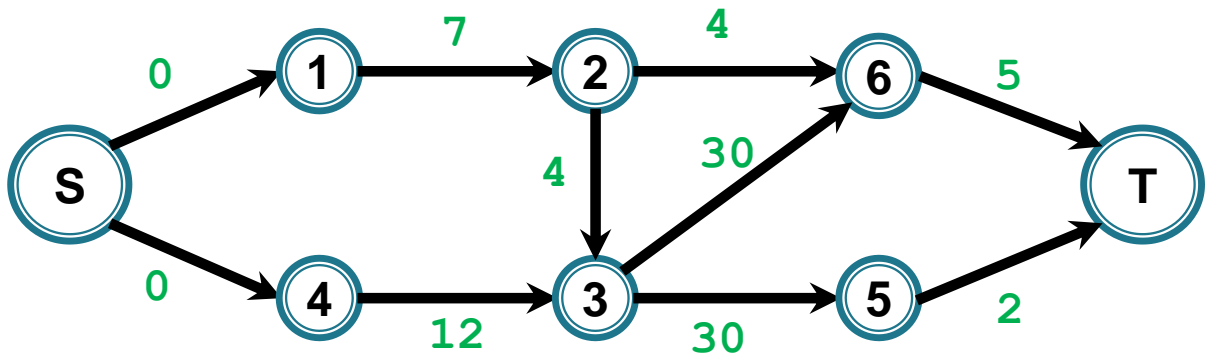
Drumuri critice



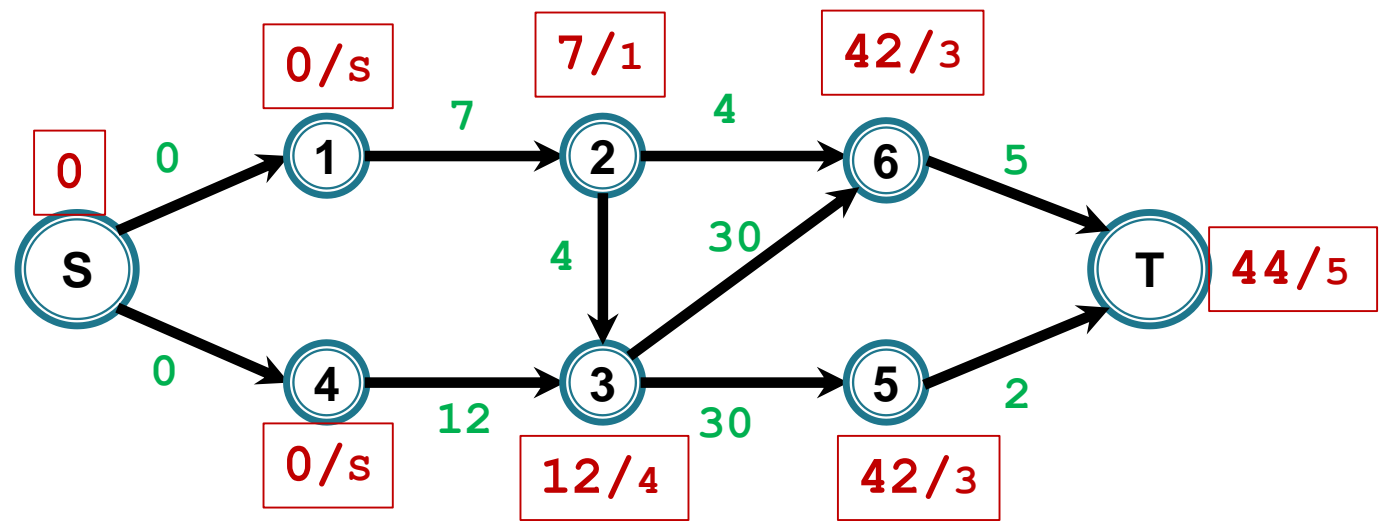
Ordine de calcul distanțe: S, 1, 4, 2, 3, 5, 6, T



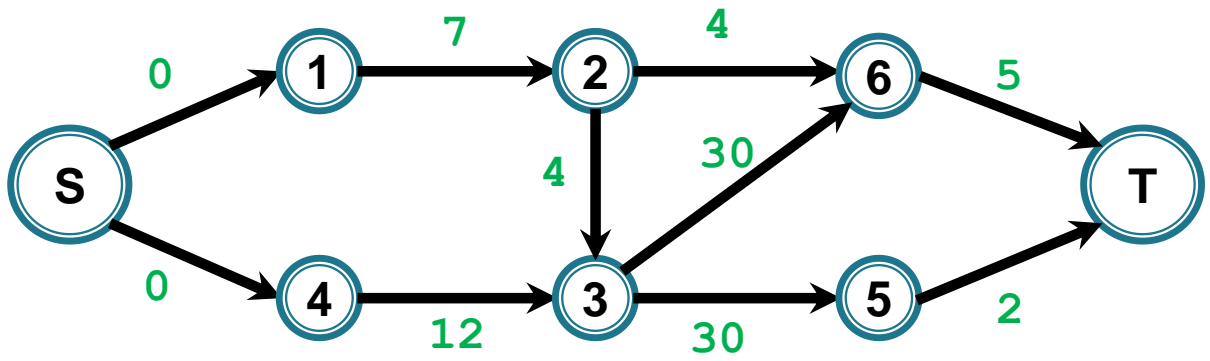
Drumuri critice



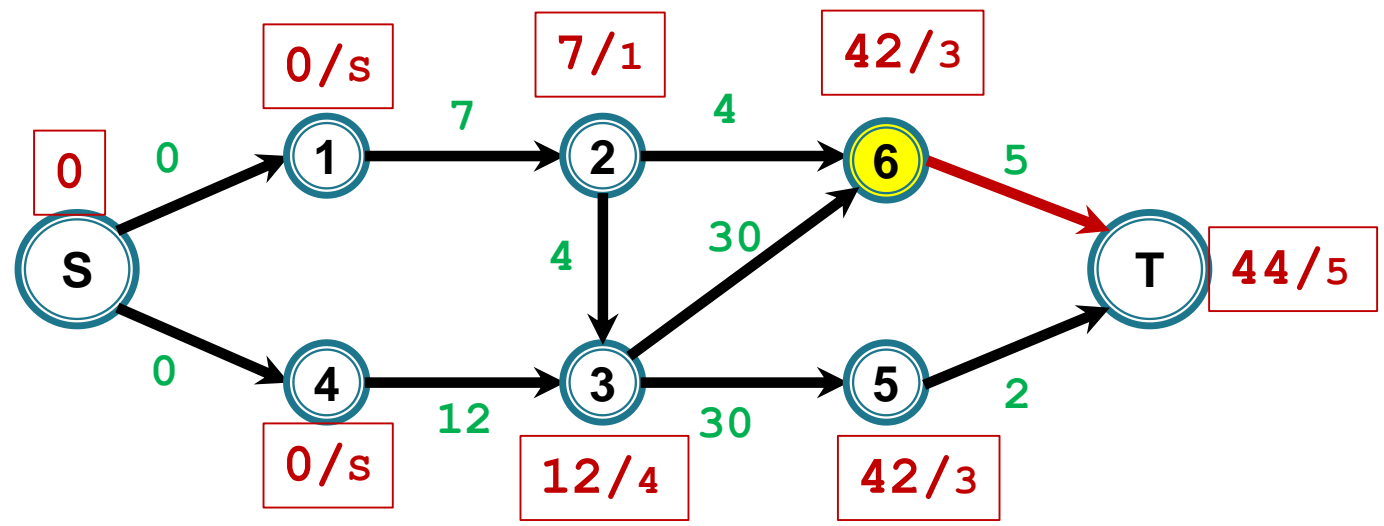
Ordine de calcul distanțe: S, 1, 4, 2, 3, 5, 6, T



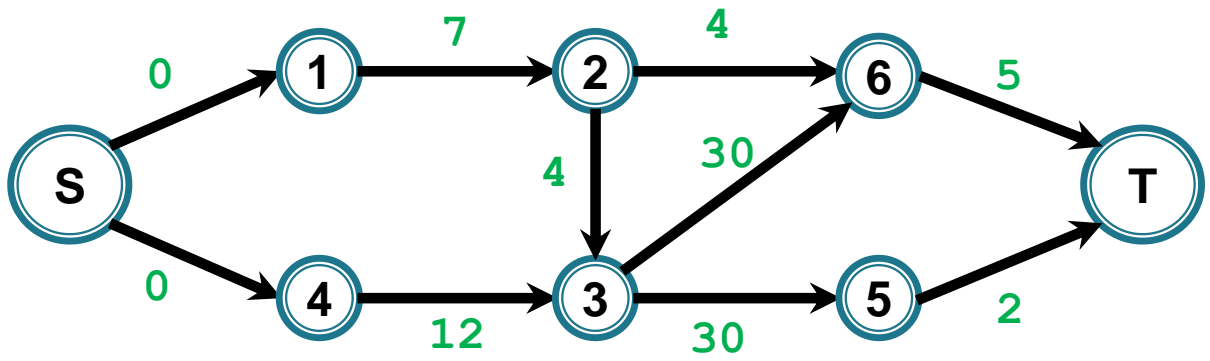
Drumuri critice



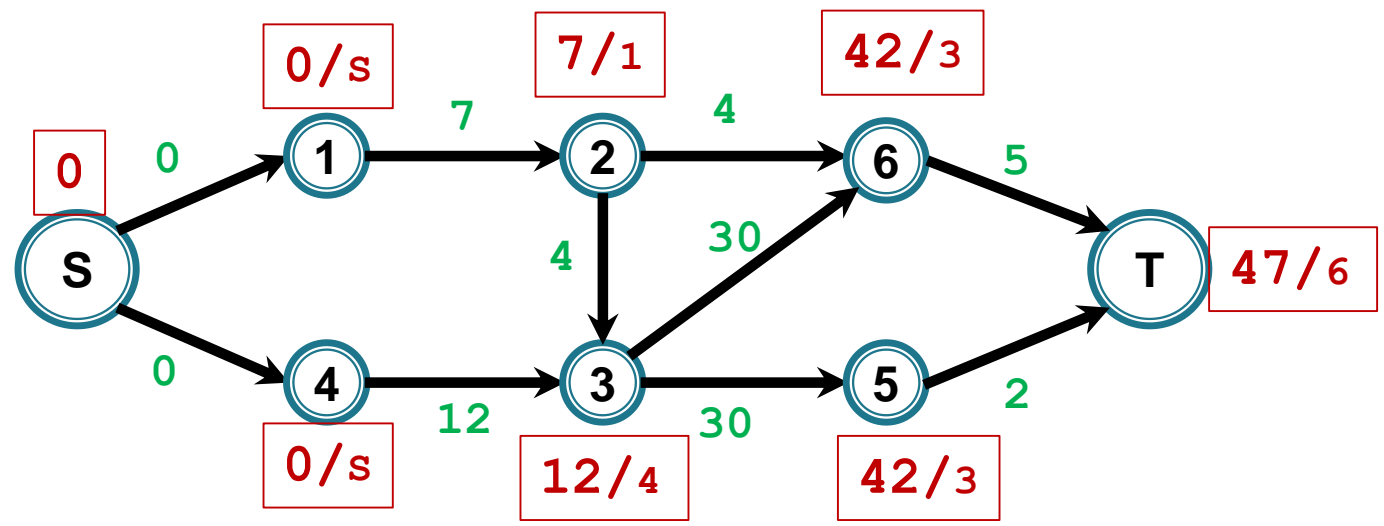
Ordine de calcul distanțe: S, 1, 4, 2, 3, 5, 6, T



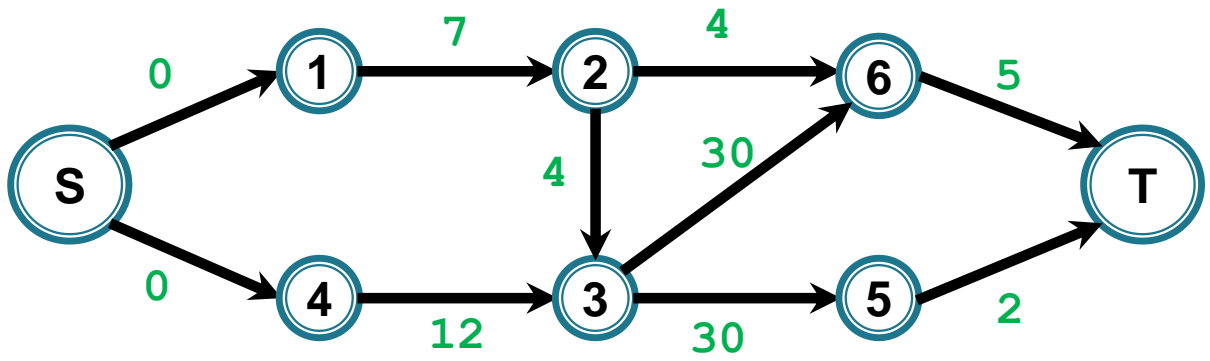
Drumuri critice



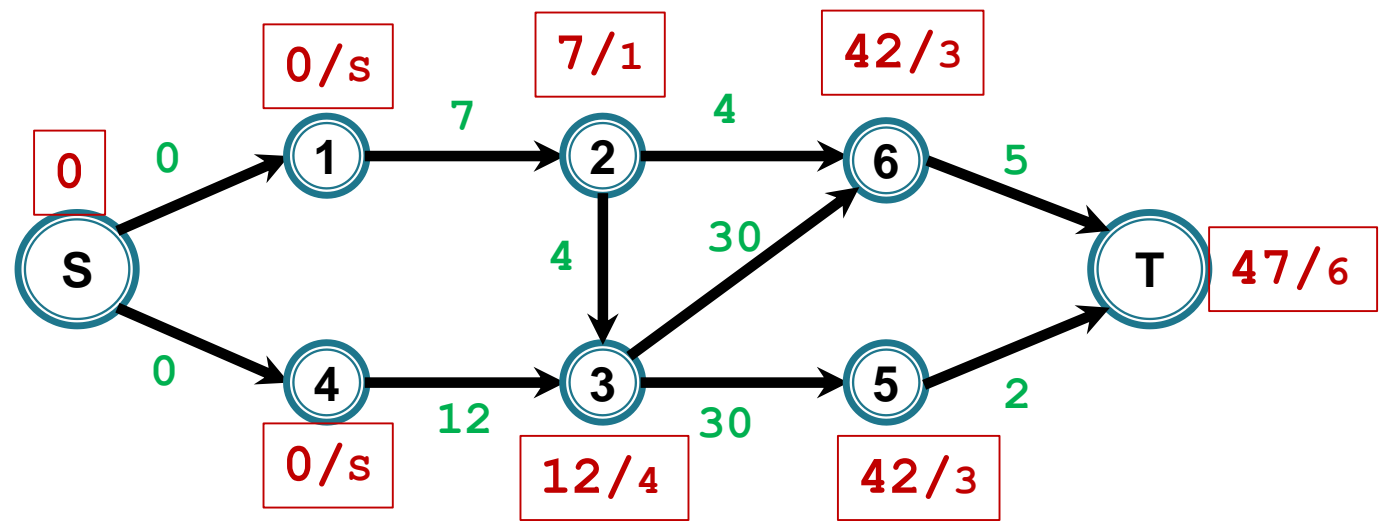
Ordine de calcul distanțe: S, 1, 4, 2, 3, 5, 6, T



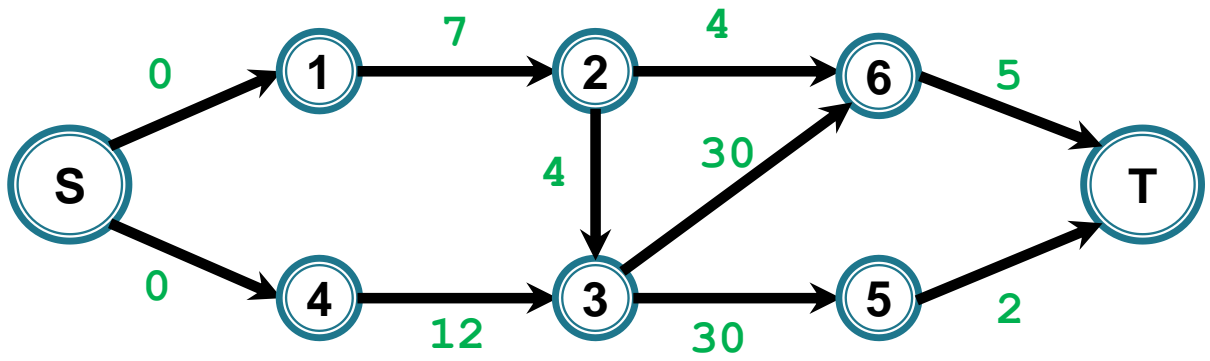
Drumuri critice



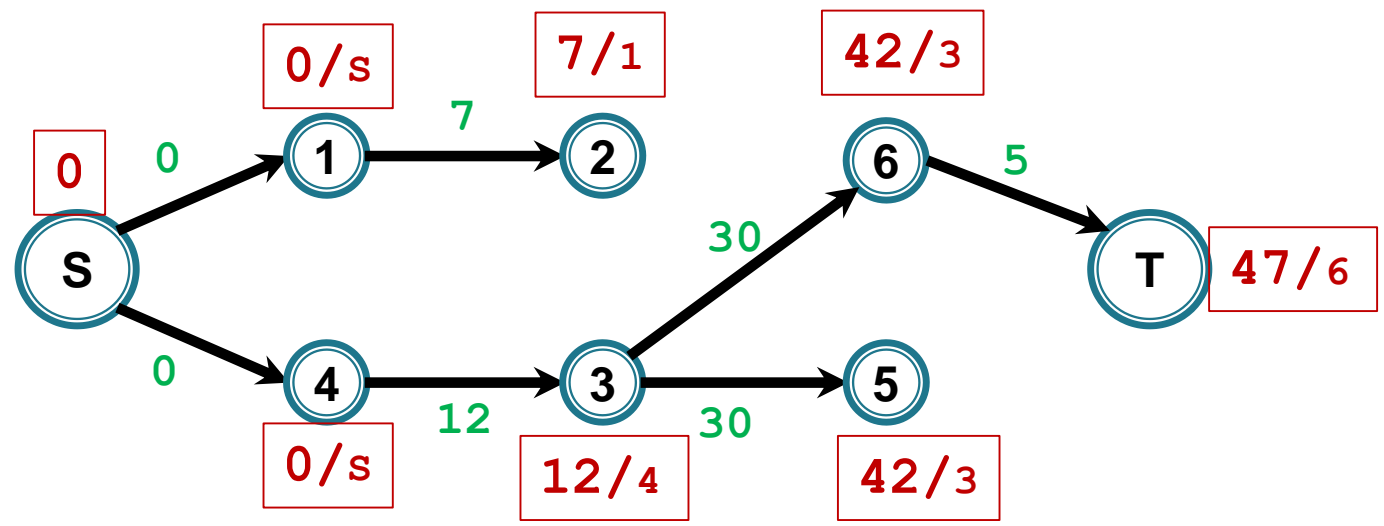
Ordine de calcul distanțe: S, 1, 4, 2, 3, 5, 6, T



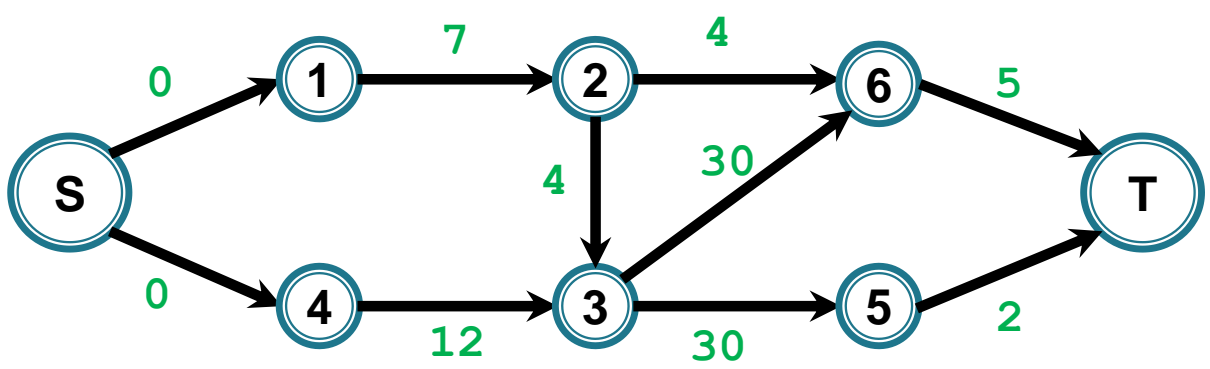
Drumuri critice



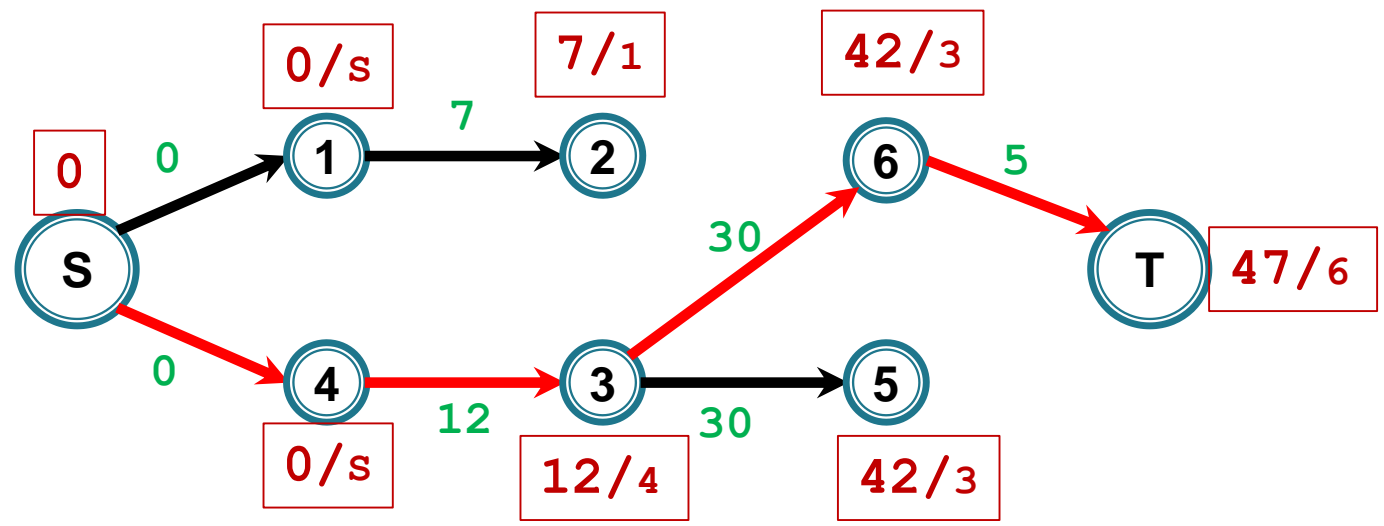
Ordine de calcul distanțe: S, 1, 4, 2, 3, 5, 6, T



Drumuri critice

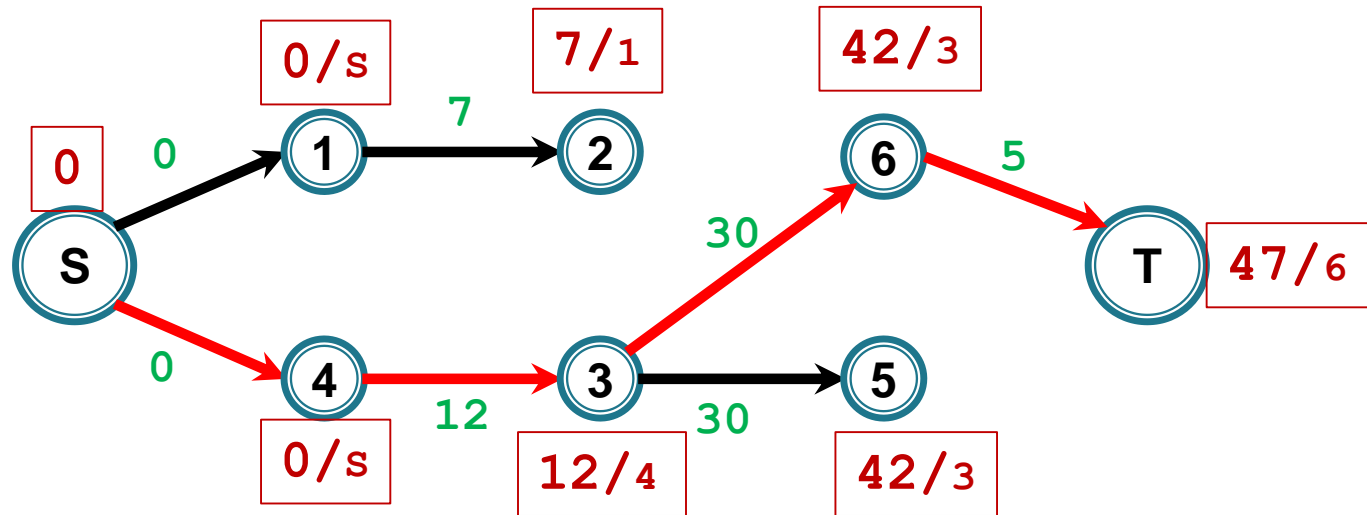


Ordine de calcul distanțe: S, 1, 4, 2, 3, 5, 6, T




Drum critic \Rightarrow succesiune de activități care determină durata proiectului

Drumuri critice



- Durata minimă a proiectului: 47
- Activități critice: 4 3 6
- Intervalele de desfășurare pentru fiecare activitate:
 - 1: (0, 7)
 - 2: (7, 8)
 - 3: (12, 42)
 - 4: (0, 12)
 - 5: (12, 42)
 - 6: (42, 47)

Drumuri maxime

 Putem modifica algoritmul lui Dijkstra de determinare de drumuri minime în grafuri (nu neapărat aciclice) a.î. să determine drumuri maxime (elementare) de la s la celelalte vârfuri?

Drumuri maxime

Putem modifica algoritmul lui Dijkstra de determinare de drumuri minime în grafuri (nu neapărat aciclice) a.î. să determine drumuri maxime (elementare) de la S la celelalte vârfuri

- Modificăm astfel doar inițializarea distanțelor (cu $-\infty$ în loc de $+\infty$) și inversăm condiția de la relaxarea arcelor pentru a calcula maxim în loc de minim
- **Corectitudine** - probabil similar cu Dijkstra?!!



Temă – Drumuri de capacitate maximă

► Problemă: Într-o rețea orientată de comunicație

- $w(e)$ = capacitatea legăturii e (exp: lățimea de bandă, diametrul unei conducte etc)
- Pentru un drum P
 $w(P) = \min \{w(e) \mid e \in E(P)\}$
= cantitatea de informație care se poate transmite de-a lungul drumui P
= capacitatea minimă a arcelor ce îl compun (pentru ca informația să poată trece prin toate arcele drumului)

Date două vârfuri s și t , să se determine un drum de capacitate maximă de la s la t – **Propuneți un algoritm bazat pe o idee similară cu cea din algoritmul lui Dijkstra. Justificați corectitudinea algoritmului propus.**