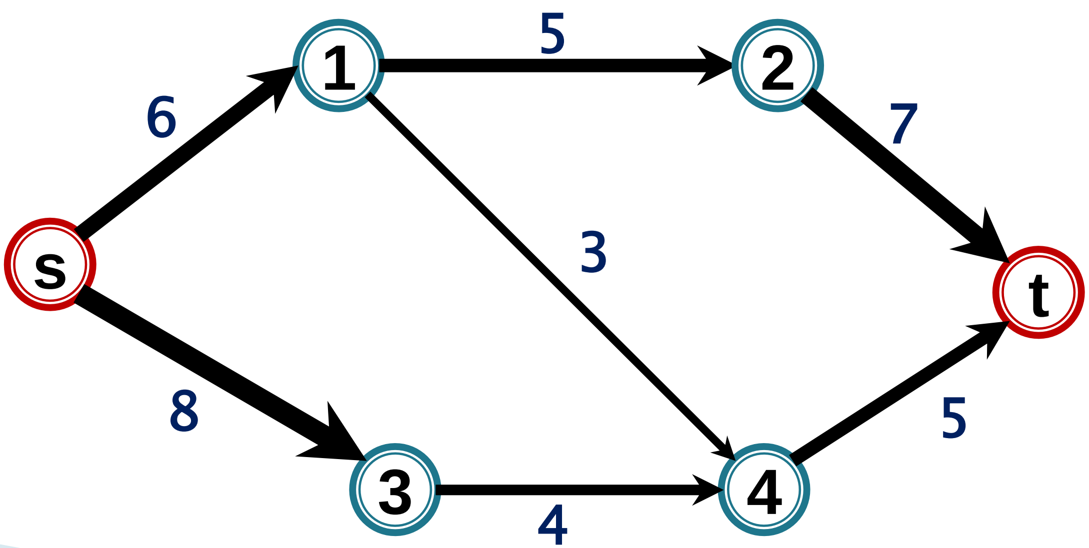
Aplicaţii la problema determinării unei tăieturi minime

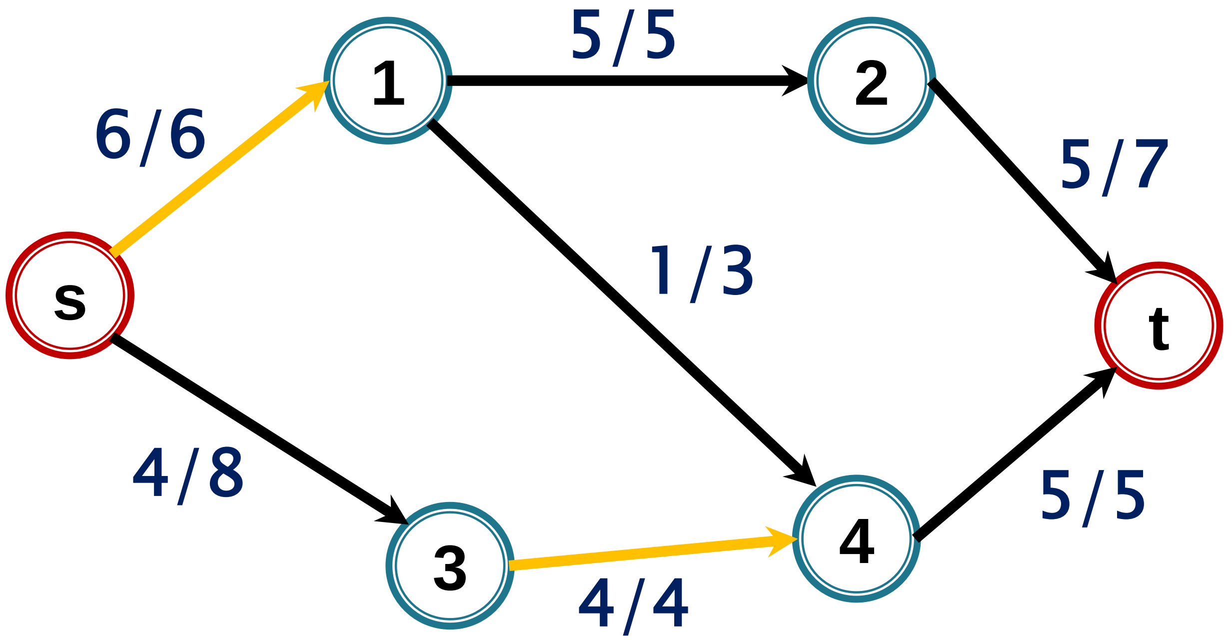
Tăietură minimă

- Determinarea unui flux maxim $\Rightarrow$ determinarea unei tăieturi minime
- **Aplicații**
  - Arce = poduri, capacitate = costul dărâmării podului.

Ce poduri trebuie dărâmate a.î. teritoriul sursă să nu mai fie conectat cu destinația și costul distrugerilor să fie minim?



Tăietură minimă



Planificare activități proiecte

Probleme de planificare

- **Se cunosc pentru un proiect cu n activități, numerotate 1,…, n:**
  - profitul p_i – care poate fi și <0

Probleme de planificare

- **Se cunosc pentru un proiect cu n activități, numerotate 1,…, n:**
  - profitul p_i – care poate fi și <0
  - perechi (i, j) = activitatea i **depinde de** activitatea j (activitatea nu poate fi efectuată decât dacă se efectuează j)

Probleme de planificare

- **Se cunosc pentru un proiect cu n activități, numerotate 1,…, n:**
  - profitul p_i – care poate fi și <0
  - perechi (i, j) = activitatea i **depinde de** activitatea j (activitatea nu poate fi efectuată decât dacă se efectuează j)

Se cere: **să se selecteze o mulțime de activități realizabile A de profit maxim**

O mulțime de activități este realizabilă dacă

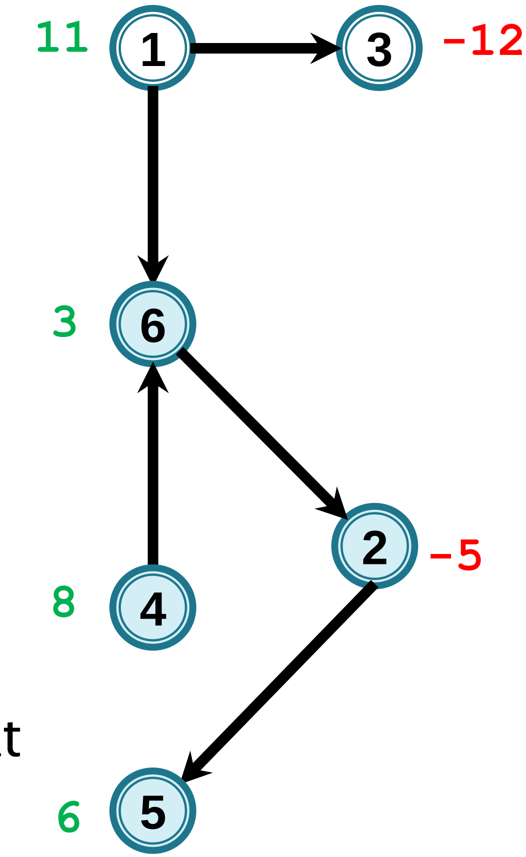
- **$i \in A$, i depinde de j $\Rightarrow$ $j \in A$**

Probleme de planificare

- **Notaţii pentru o mulţime A de activităţi**
  - $castig(A) = \sum_{i \in A, p_i \geq 0} p_i$
  - $pierdere(A) = \sum_{i \in A, p_i < 0} (-p_i)$
  - $profit(A) = castig(A) - pierdere(A)$
  - $C_{tot} = castig(\{1, \ldots, n\})$

Probleme de planificare

- **Exemplu:**
  - n = 6 activități
  - 1 – profit 11
  - 2 – profit –5
  - 3 – profit –12
  - 4 – profit 8
  - 5 – profit 6
  - 6 – profit 3
  - Dependențe indicate de graful alăturat

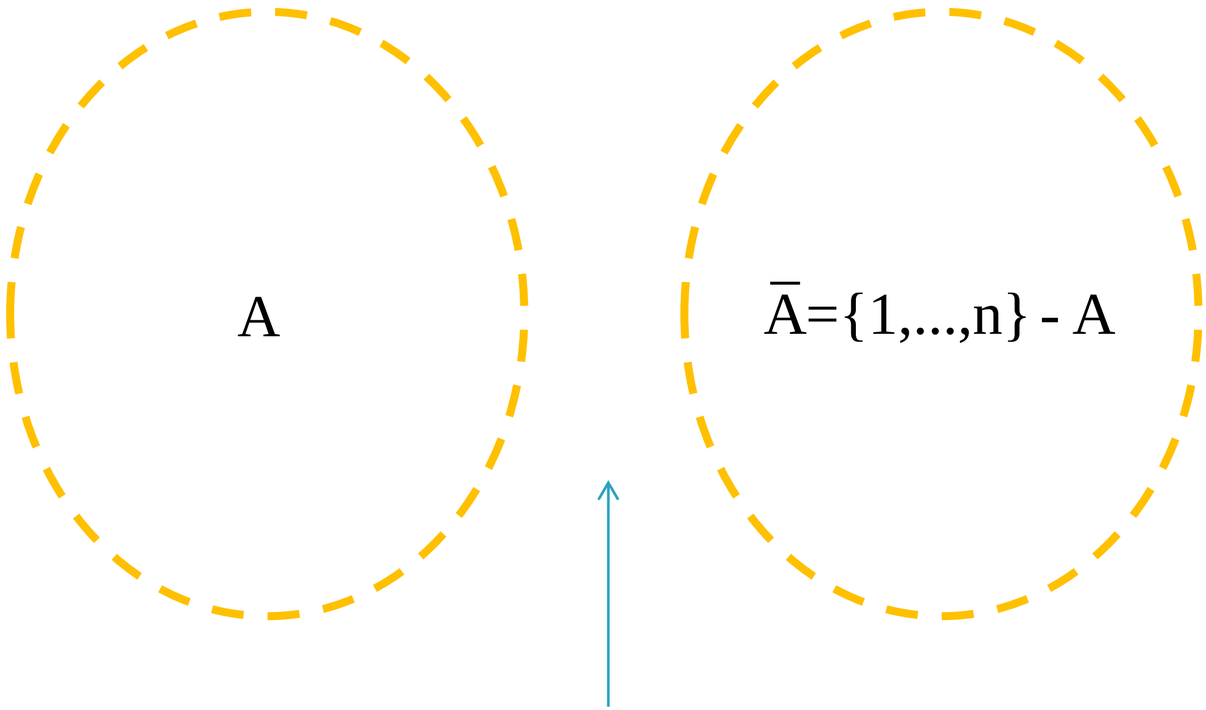


Soluție A = {2, 4, 5, 6}, profit(A) = 12

(castig(A) = 17, pierdere(A)=5)

Probleme de planificare

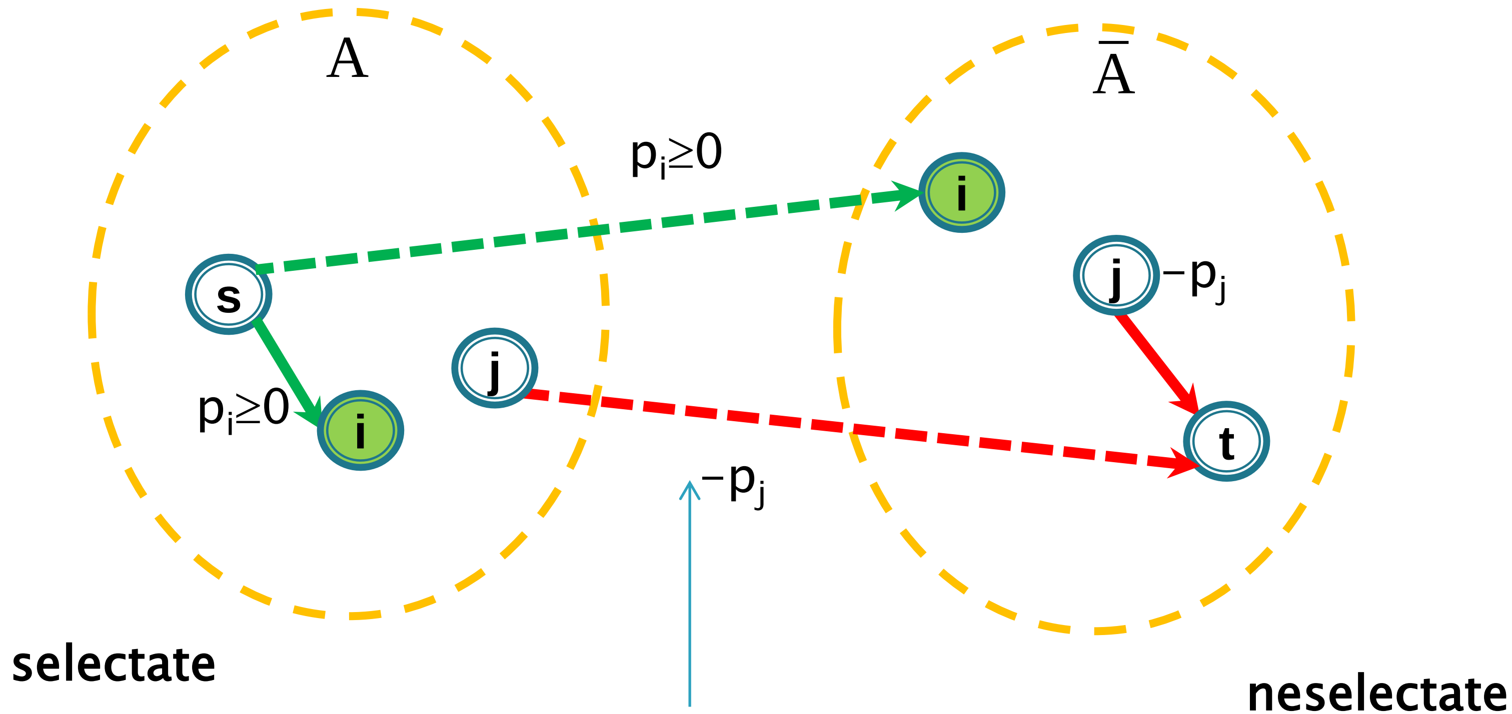
- **Trebuie să împărțim activitățile în două:**



pierderi în profit minime (față de C_{tot})

Probleme de planificare

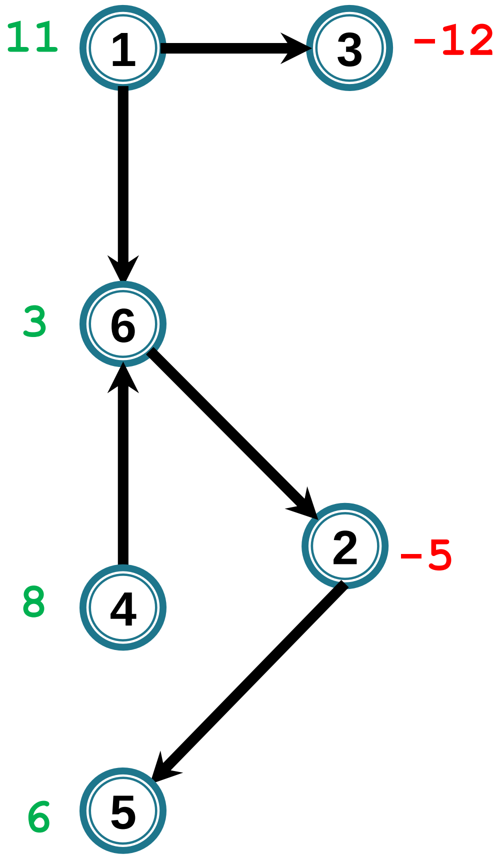
- Asociem problemei o rețea și reducem determinarea lui A la determinarea unei tăieturi minime în rețea



pierderi în profit minime (față de C_{tot})
⇔ capacitatea tăieturii trebuie să fie minimă

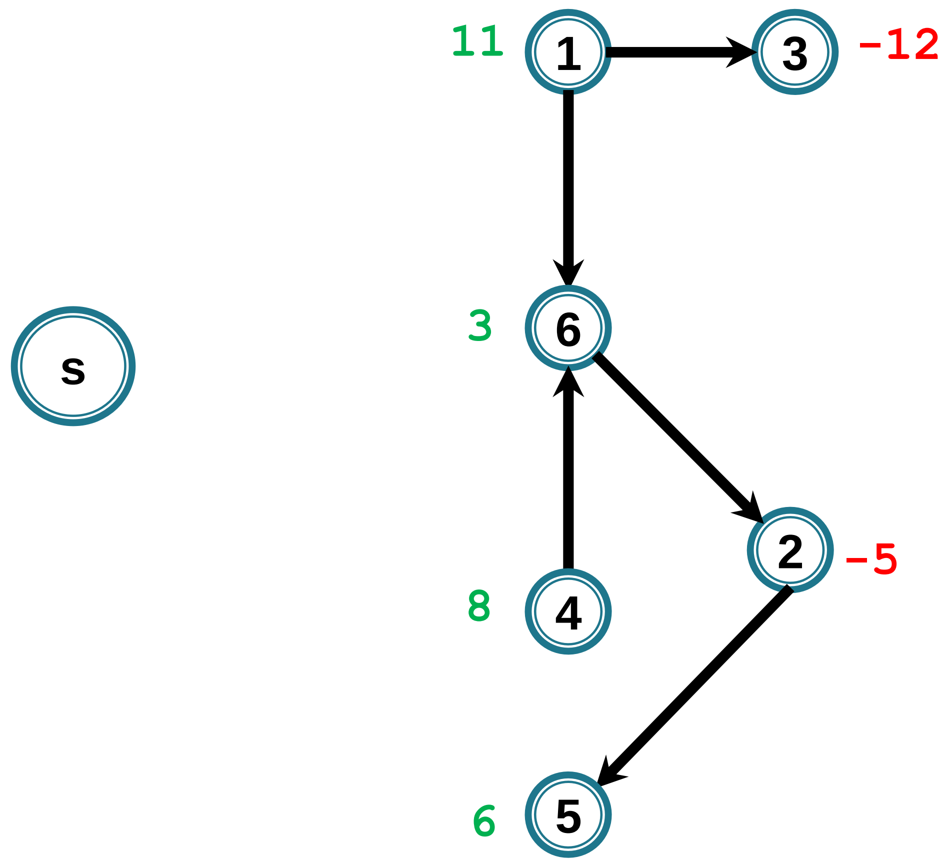
Probleme de planificare

- **Exemplu:**
  - n = 6 activități
  - 1 – profit 11
  - 2 – profit –5
  - 3 – profit –12
  - 4 – profit 8
  - 5 – profit 6
  - 6 – profit 3
  - Dependențe indicate de graful alăturat



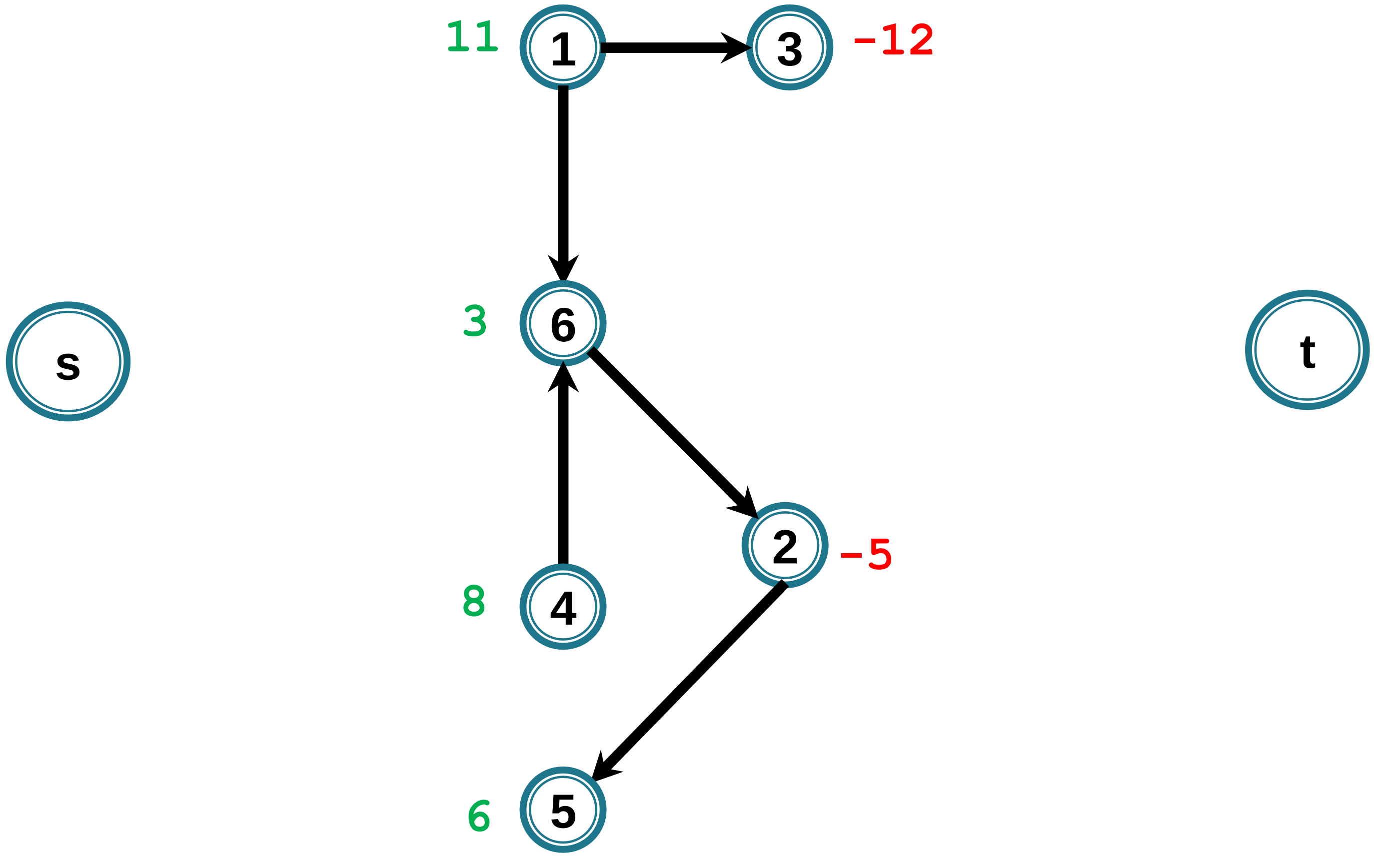
Probleme de planificare

- Asociem grafului o rețea de transport



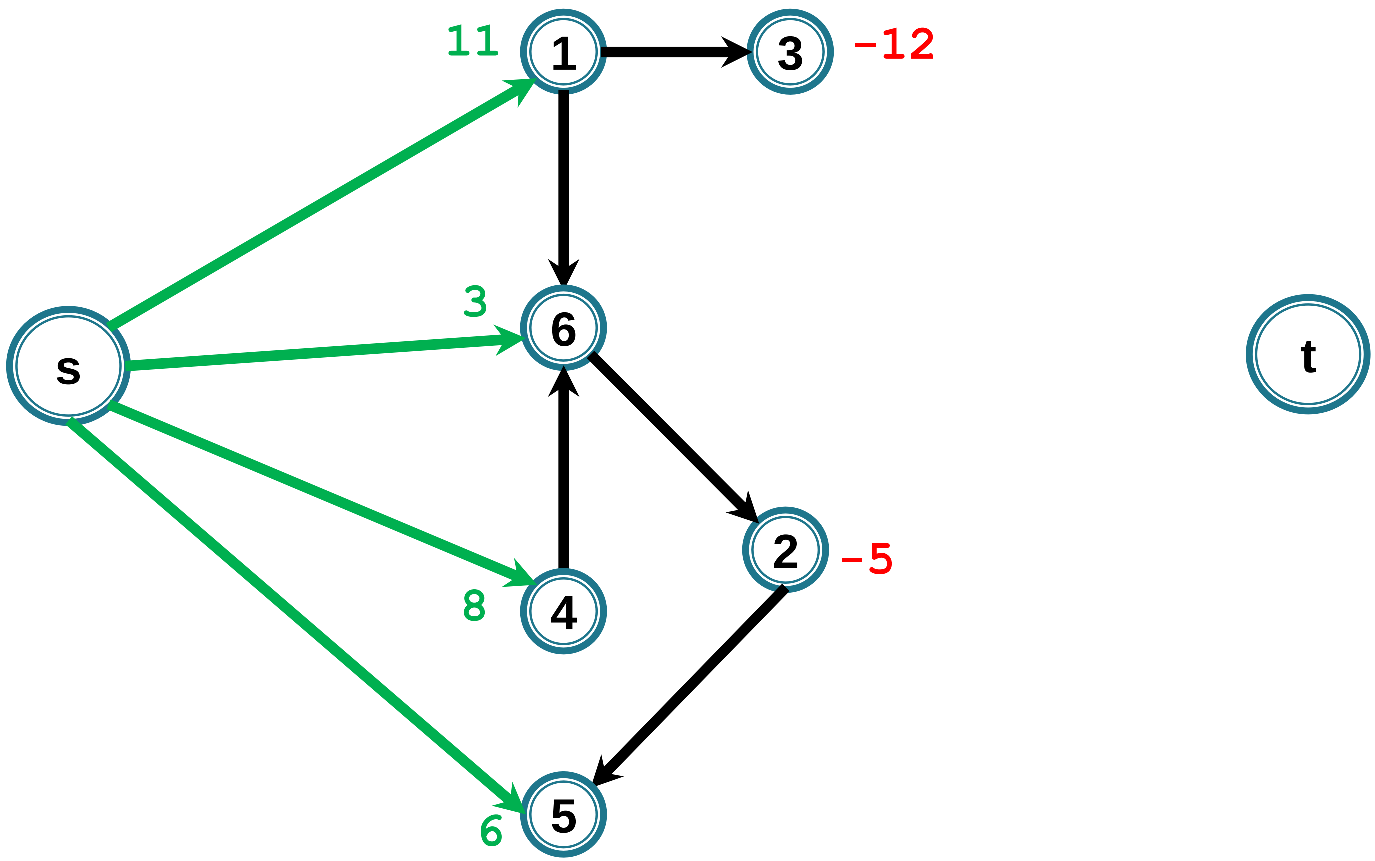
Probleme de planificare

- Adăugăm o sursă și o destinație



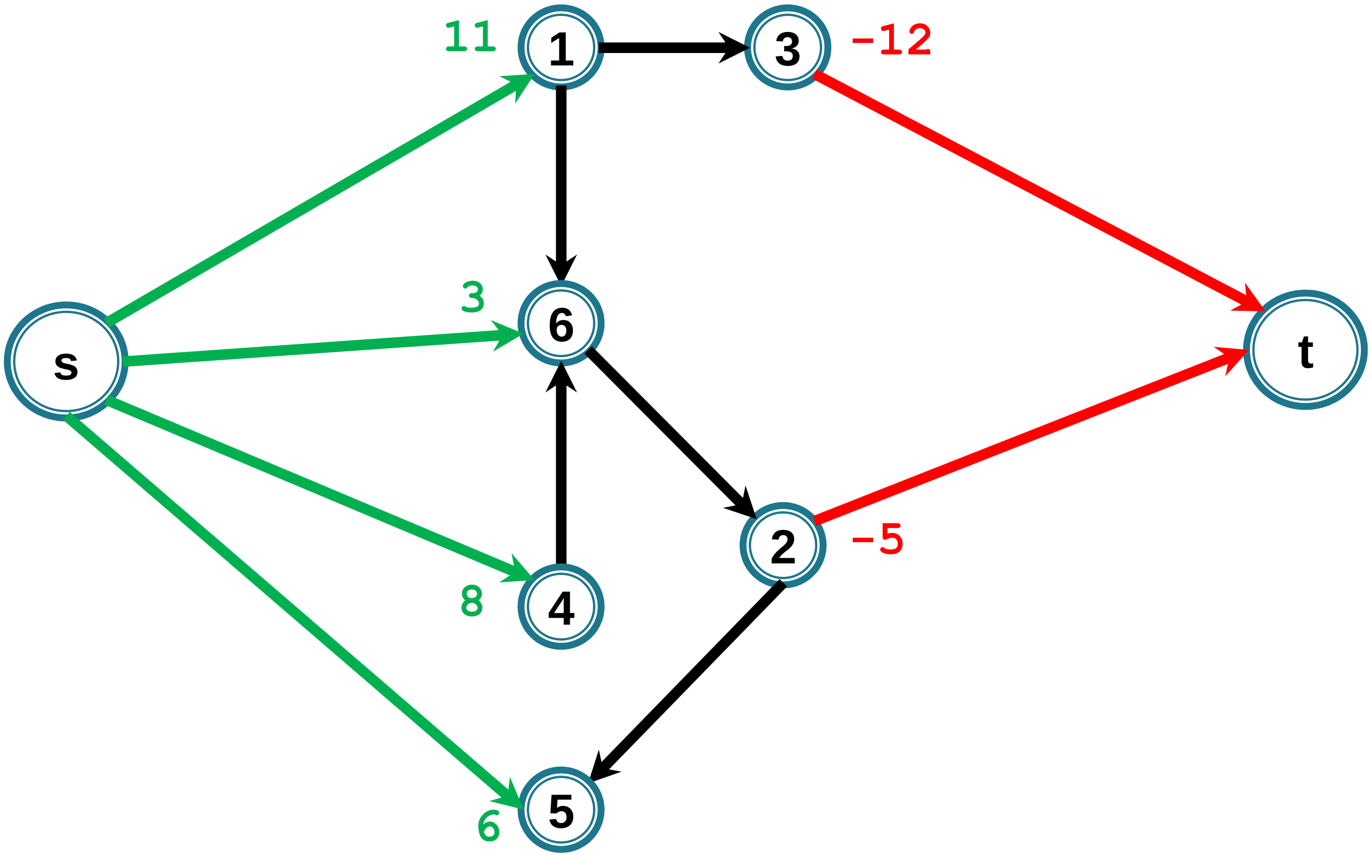
Probleme de planificare

- Unim s cu activități cu profit >=0



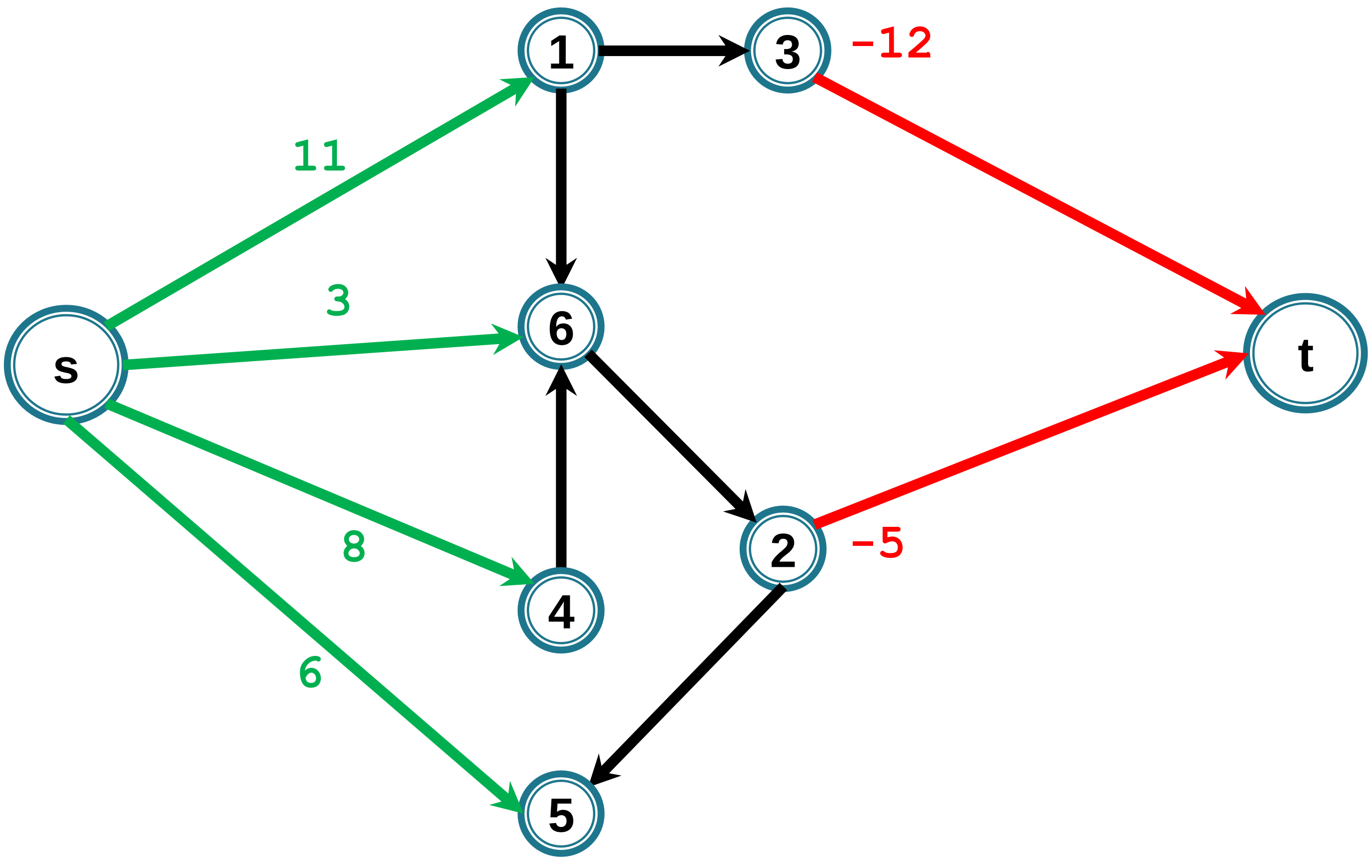
Probleme de planificare

- Unim activitățile cu profit <0 cu t



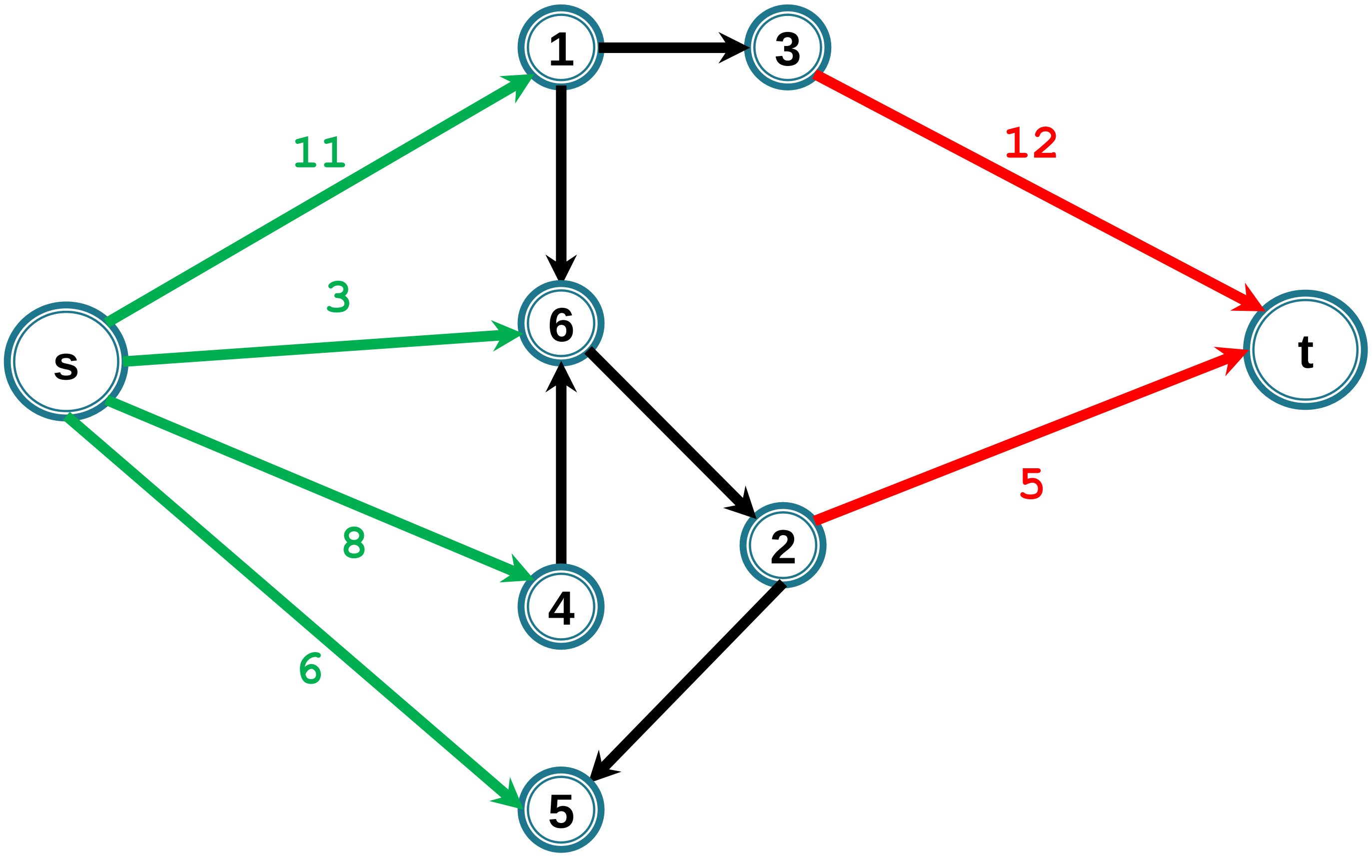
Probleme de planificare

- c(sx) = profitul lui x



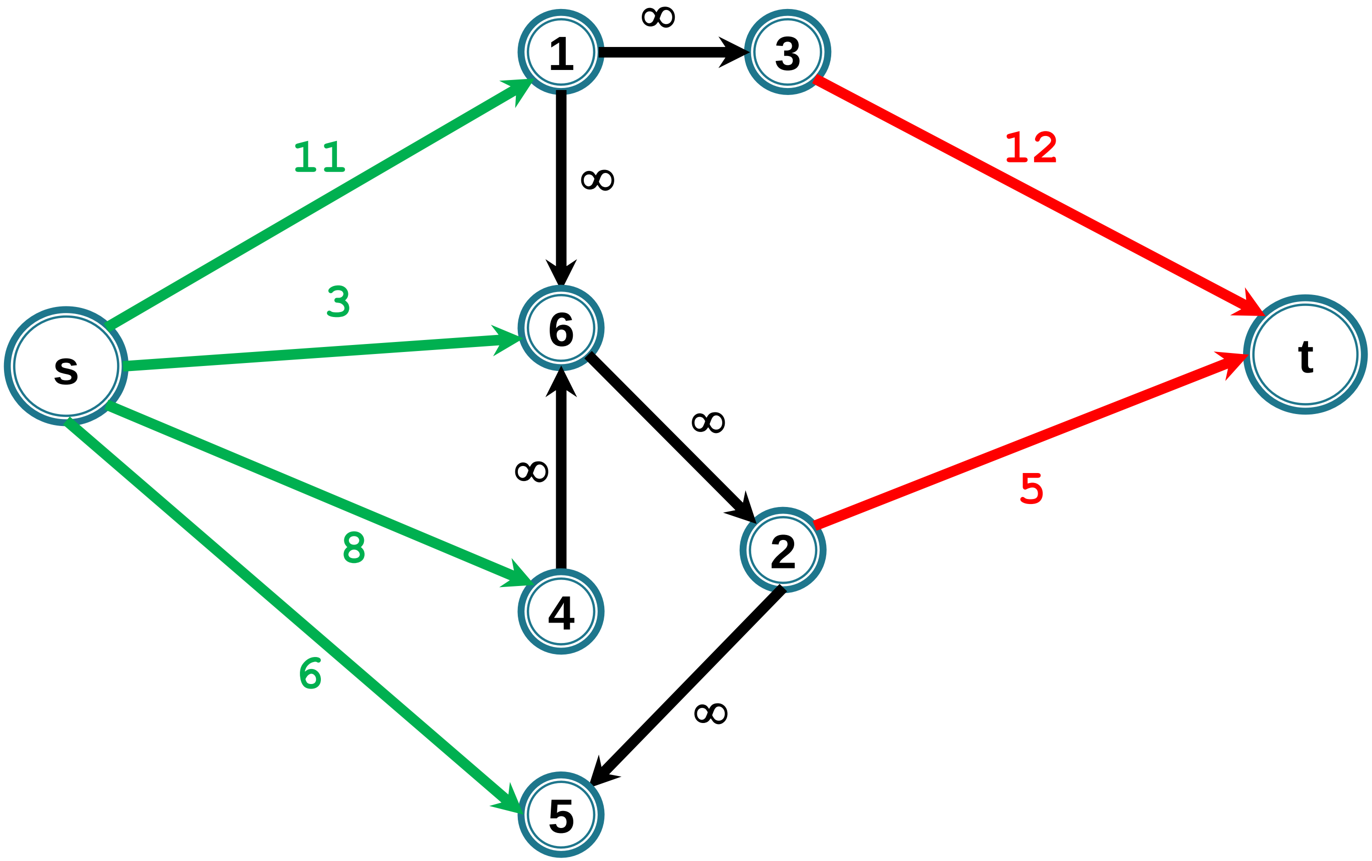
Probleme de planificare

- c(yt) = modulul profitului lui y

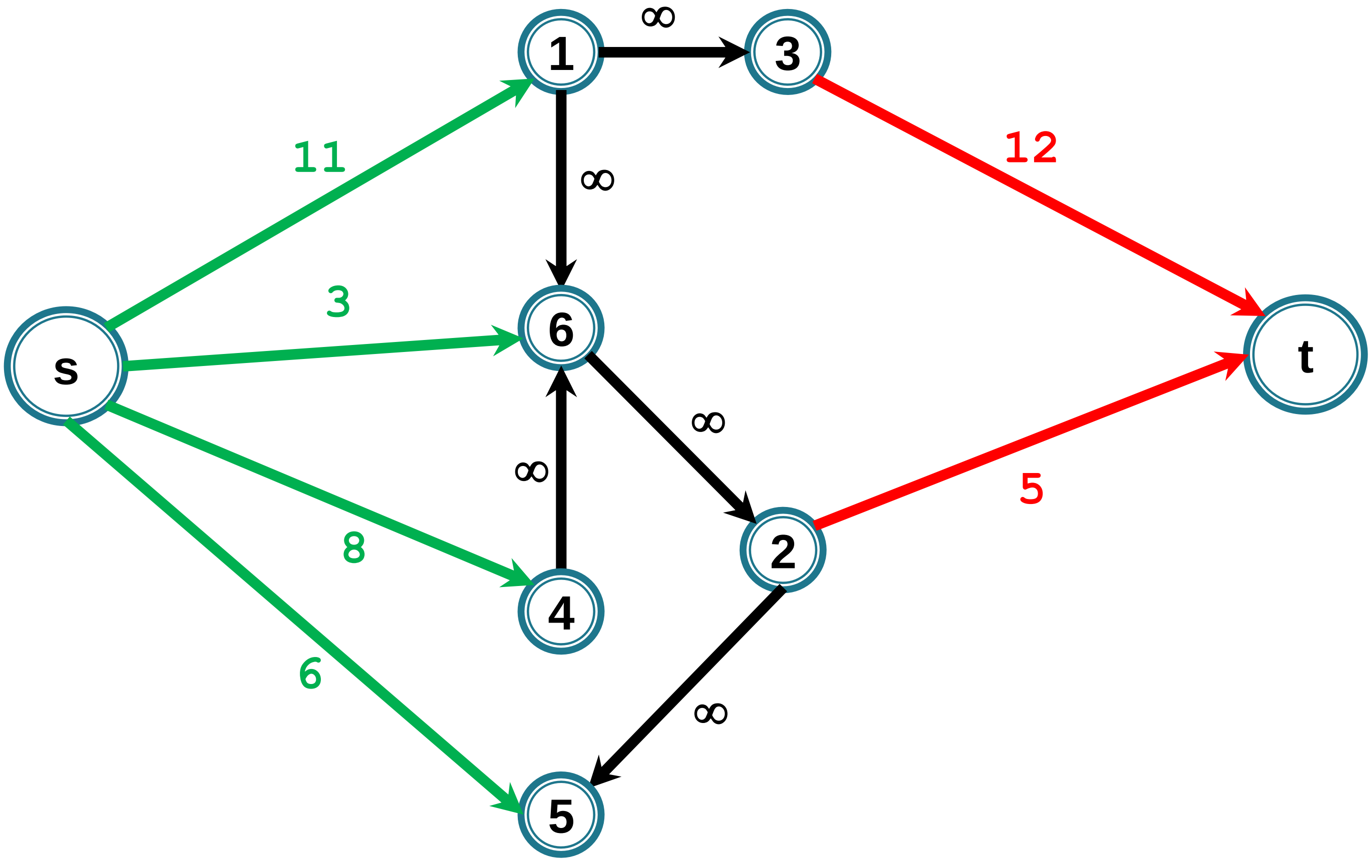


Probleme de planificare

- Restul capacităților = ∞ – pentru a ne asigura că A este realizabilă

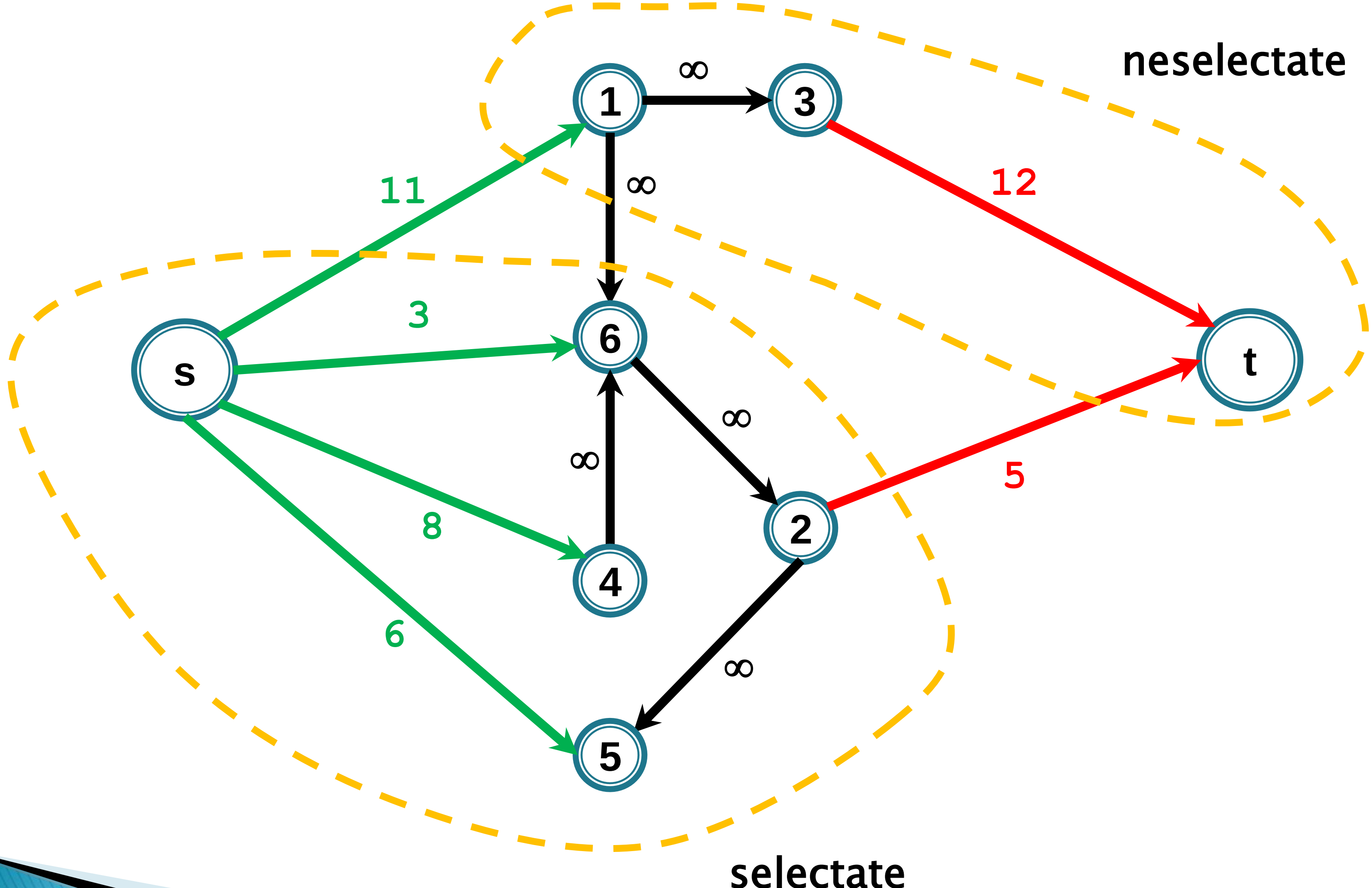


Probleme de planificare



Probleme de planificare

- Capacitate unei tăieturi finite = suma de module de profit



Probleme de planificare

1. **Orice s-t tăietură în rețea este de forma** $K_A = (A \cup \{s\}, \bar{A} \cup \{t\})$

2.

3.

Probleme de planificare

1. **Orice s–t tăietură în rețea este de forma** $K_A = (A \cup \{s\}, \bar{A} \cup \{t\})$

2. **Fie A o mulțime de activități.**

   Dacă $c(K_A) < \infty$, atunci A este o mulțime de activități realizabilă

   .

3.

Probleme de planificare

1. **Orice s–t tăietură în rețea este de forma** $K_A = (A \cup \{s\}, \bar{A} \cup \{t\})$

2. **Fie A o mulțime de activități.**

   Dacă $c(K_A) < \infty$, atunci A este o mulțime de activități realizabilă

   - dacă $i \in A$ şi i depinde de j, atunci ij are capacitate ∞, deci nu aparţine tăieturii; rezultă că j este tot în A

3.

Probleme de planificare

1. **Orice s-t tăietură în rețea este de forma** $K_A = (A \cup \{s\}, \bar{A} \cup \{t\})$

2. **Fie A o mulțime de activități.**

   Dacă $c(K_A) < \infty$, atunci A este o mulțime de activități realizabilă

   - dacă $i \in A$ şi i depinde de j, atunci ij are capacitate ∞, deci nu aparţine tăieturii; rezultă că j este tot în A

3. **Fie A o mulțime de activități.**

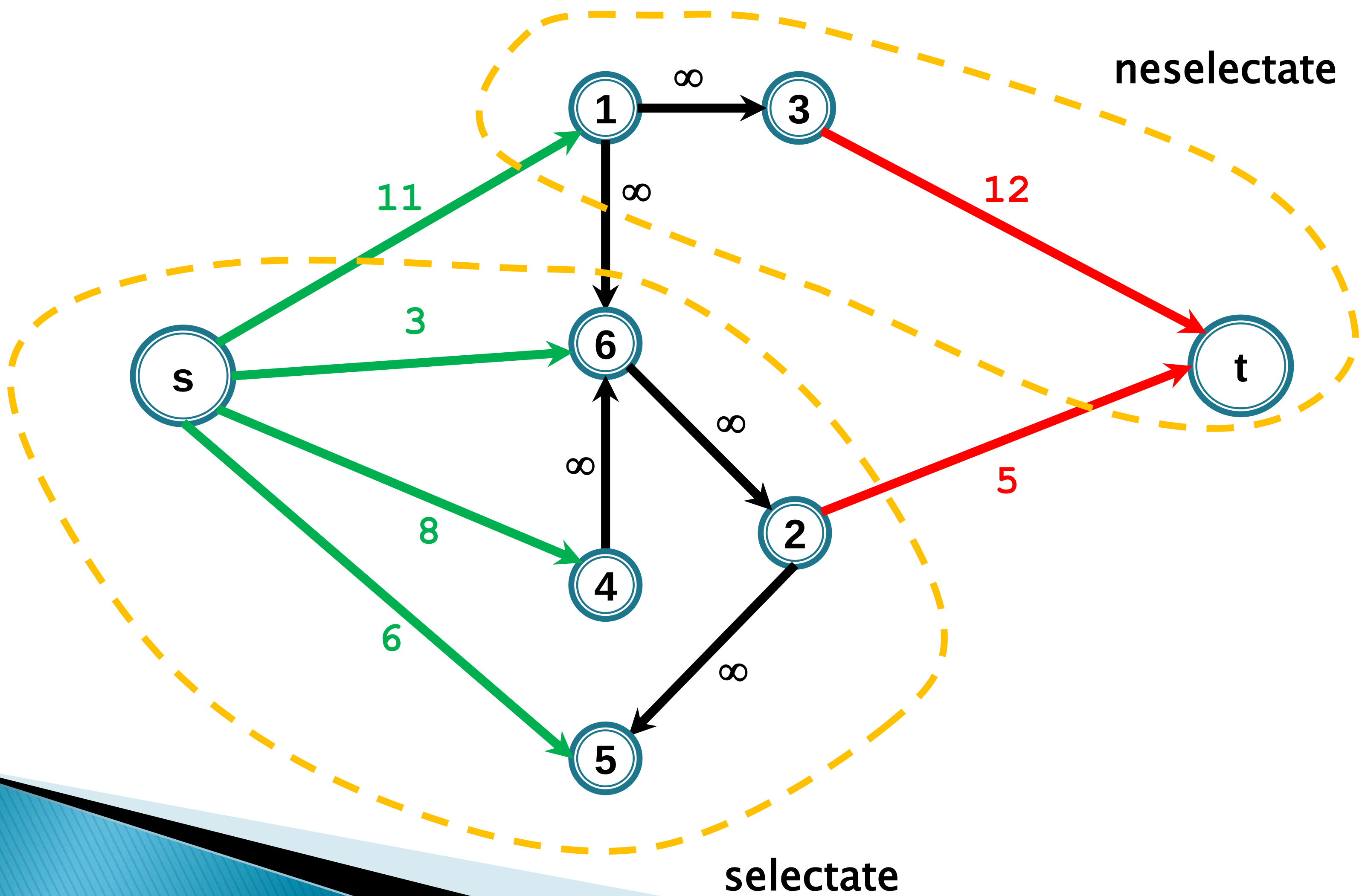
   A este realizabilă $\Leftrightarrow$

   s-t tăietura corespunzătoare K_A are capacitate finită

Probleme de planificare

Propoziţie. Dacă K_A este o s–t tăietură de capacitate finită avem

$$\texttt{profit}(A) = C_{tot} - c(K_A)$$



Probleme de planificare

Justificare: Avem

$$\texttt{profit(A)} = \texttt{castig(A)} - \texttt{pierdere(A)} =$$

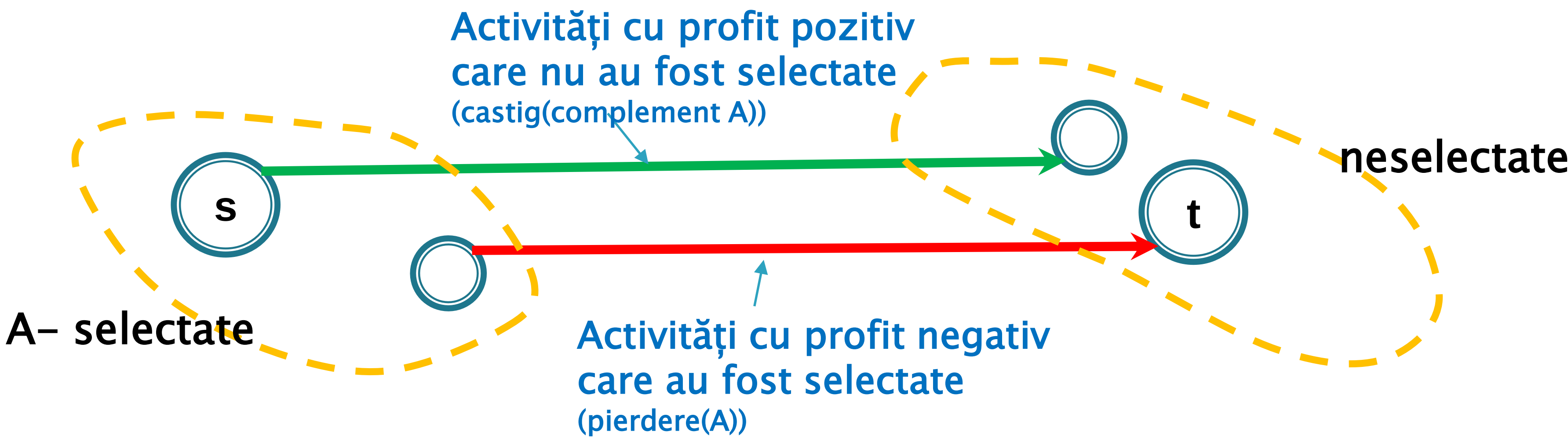
$$= C_{tot} - \texttt{castig}(\bar{A}) - \texttt{pierdere(A)}$$

Probleme de planificare

Justificare: Avem

$$\texttt{profit(A)} = \texttt{castig(A)} - \texttt{pierdere(A)} =$$

$$= C_{tot} - \texttt{castig}(\bar{A}) - \texttt{pierdere(A)}$$



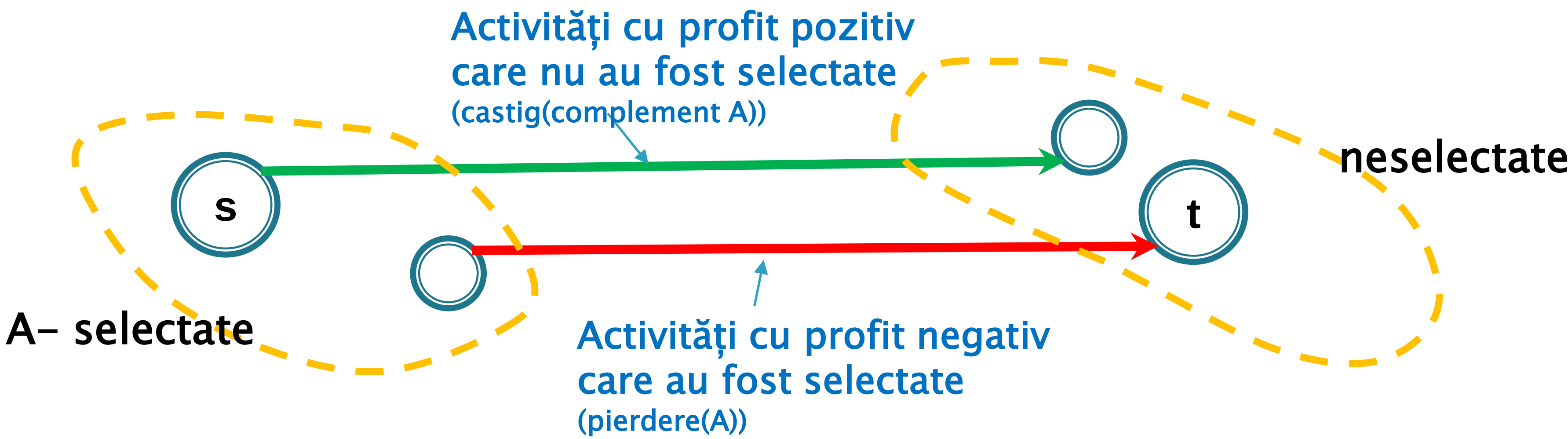
$$c(K_A) =$$

Probleme de planificare

Justificare: Avem

$$\texttt{profit(A) = castig(A) - pierdere(A) =}$$

$$\texttt{= } C_{tot} \texttt{ - castig(}\bar{A}\texttt{) - pierdere(A)}$$



$$c(K_A) = c^+(A) + c^+(s) = \sum_{i \in A, p_i < 0} (-p_i) + \sum_{i \in \bar{A}, p_i > 0} p_i =$$

$$= pierdere(A) + castig(\overline{A})$$

Probleme de planificare

Propoziţie. Dacă K_A este o s-t tăietură de capacitate finită avem

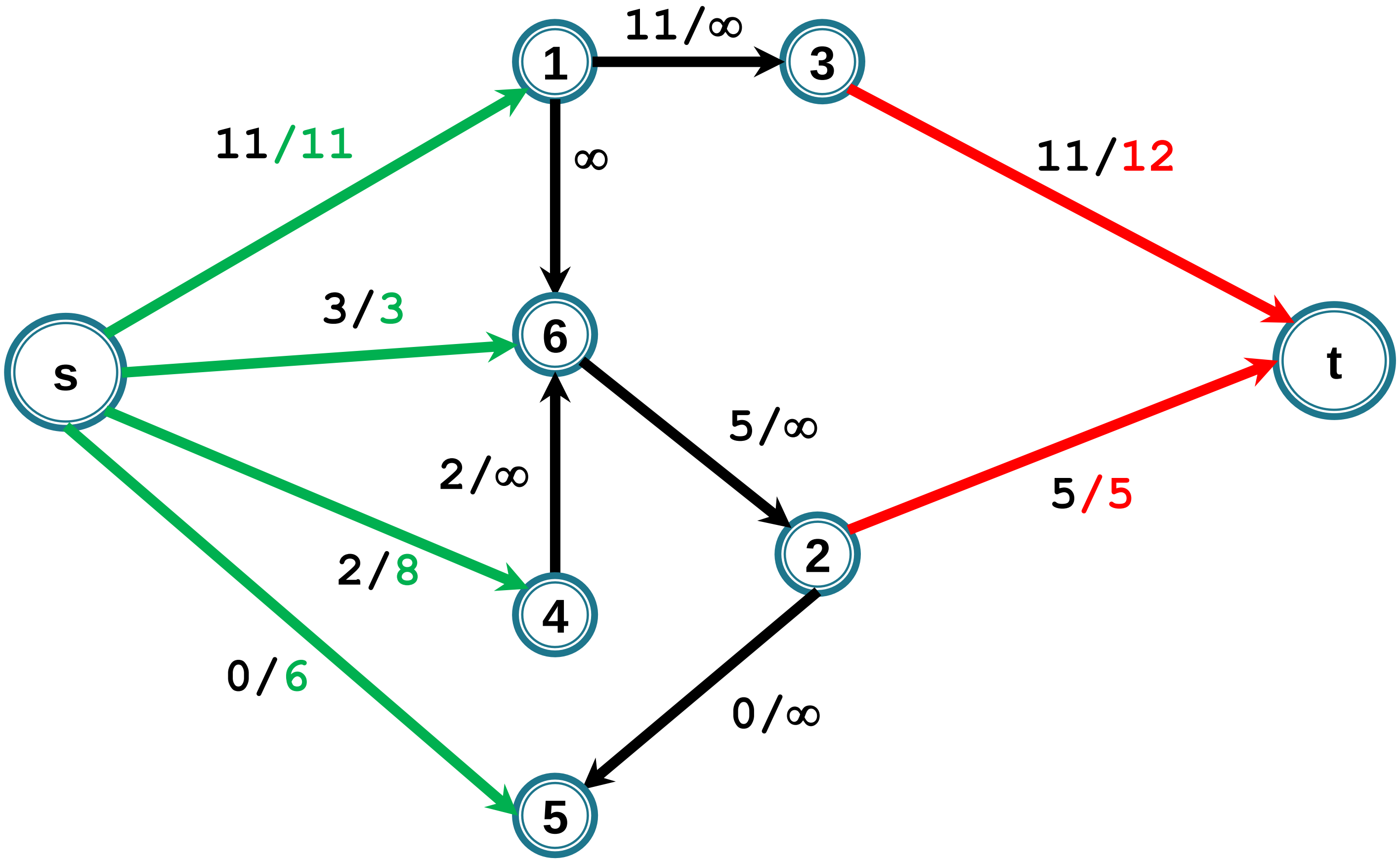
$$\texttt{profit(A)} = C_{tot} - c(K_A)$$

- **Corolar.**
  - Mulțimea de activități realizabile A este de **profit maxim** $\Leftrightarrow$ **tăietura** $K_A = (A \cup \{s\}, \bar{A} \cup \{t\})$ este de **capacitate minimă** în reţeaua asociată
- A determina o mulţime de activităţi de profit maxim se reduce astfel la a determina o tăietură minimă în reţeaua asociată

Cum o determinăm?

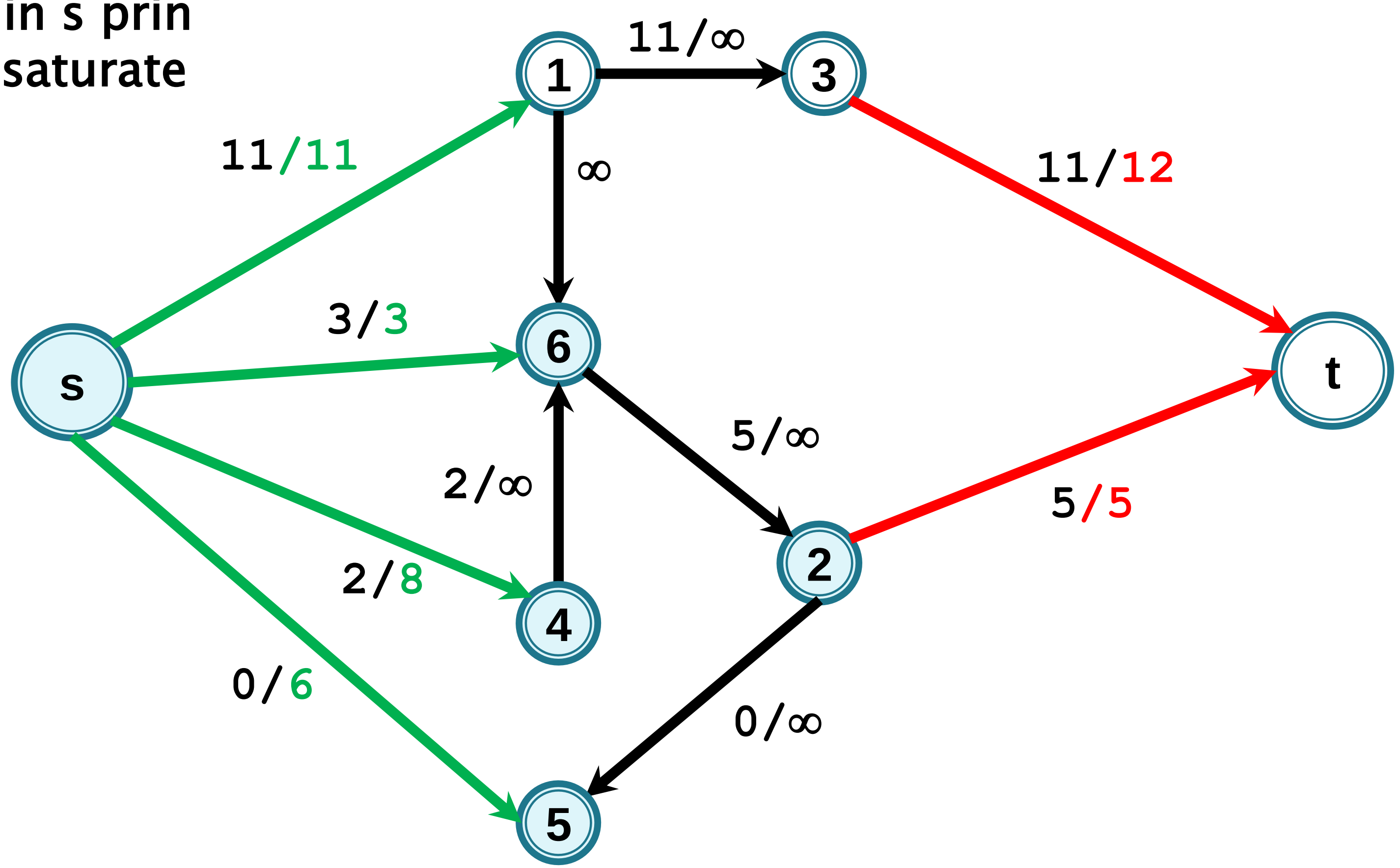
Probleme de planificare

Determinăm un flux maxim

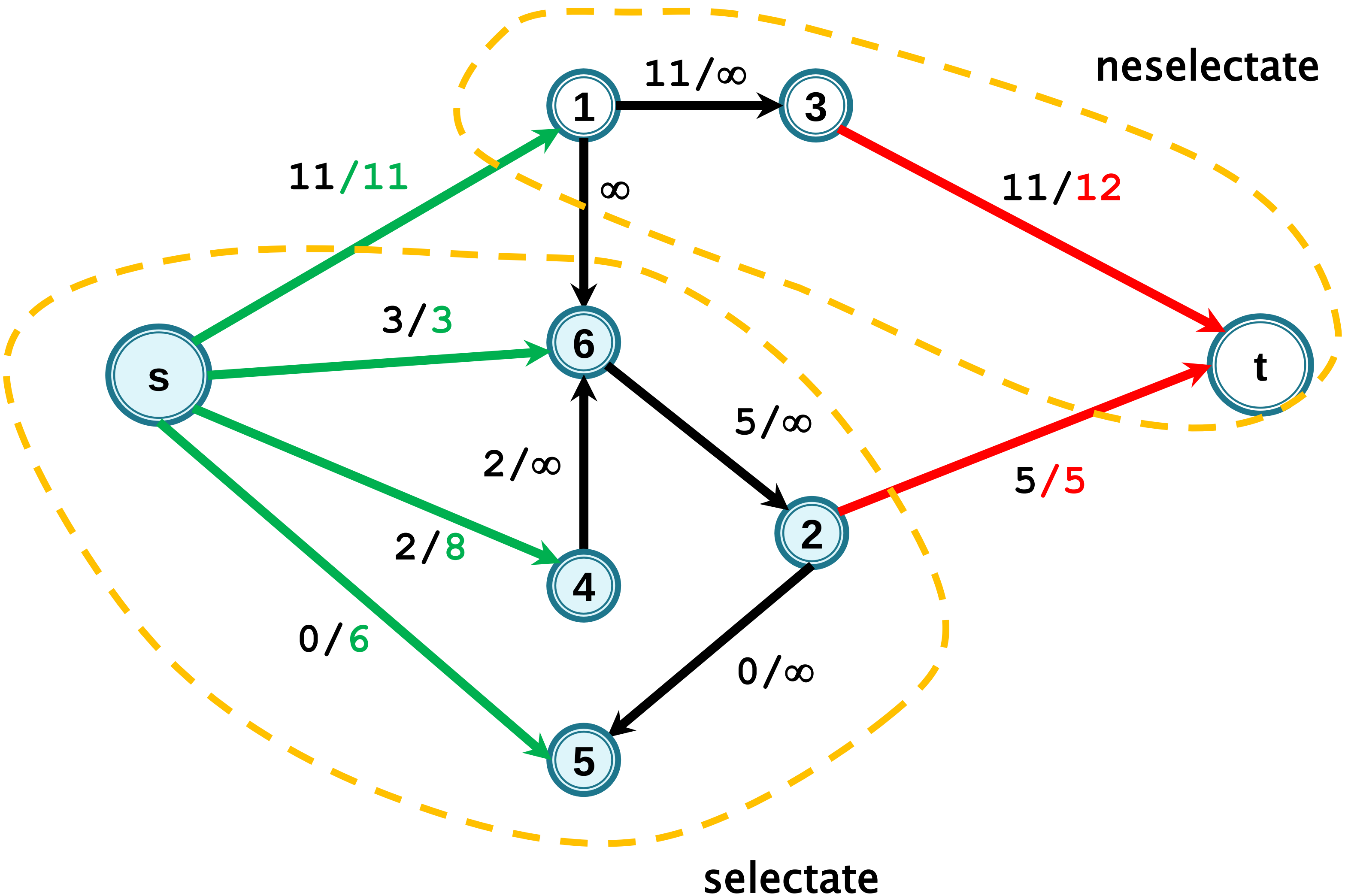


Probleme de planificare

Determinăm vârfurile accesibile din s prin lanțuri f-nesaturate



Probleme de planificare



Probleme de planificare

Algoritm de determinare a unei mulțimi de activități cu profit maxim

1. Construim N reţeaua de transport asociată

2. Determinăm K = (X,Y) tăietură minimă în N

Probleme de planificare

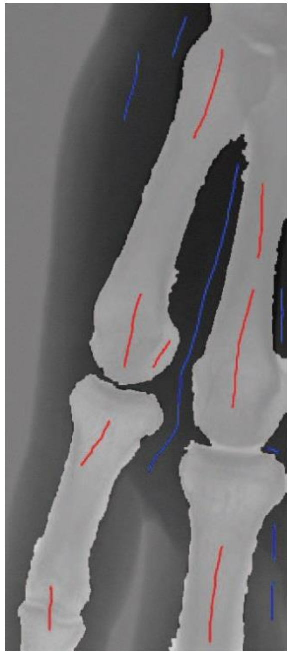
Algoritm de determinare a unei mulțimi de activități cu profit maxim

1. Construim N reţeaua de transport asociată
2. Determinăm K = (X,Y) tăietură minimă în N
3. A = X – {s}

Segmentarea imaginilor

Image segmentation

- medicină



Spatially Varying Color Distributions for Interactive Multi-Label Segmentation (C. Nieuwenhuis, D. Cremers), In IEEE Transactions on Pattern Analysis and Machine Intelligence, volume 35, 2013

Segmentarea imaginilor

Delimitare regiuni foreground/background – decidem pentru fiecare pixel dacă aparține fundalului (este în background) sau prim planului (este în foreground)

https://courses.engr.illinois.edu/cs473/sp2013/w/lec/19_add_notes.pdf

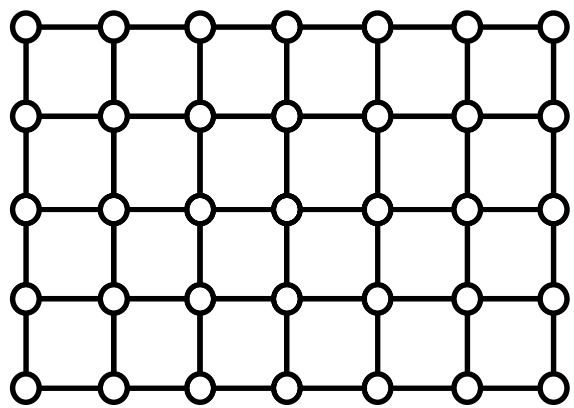
Segmentarea imaginilor

Delimitare regiuni foreground/background –

- **pixeli vecini** $\Rightarrow$ o imagine poate fi privită ca un graf grid de pixeli

  – vârfuri = pixeli

  – muchii = pixeli vecini



Segmentarea imaginilor

Delimitare regiuni foreground/background –

- **Date**
  - Pentru fiecare **pixel** i:

    f_i = probabilitatea ca i $\in$ foreground

    b_i = probabilitatea ca i $\in$ background

Segmentarea imaginilor

Delimitare regiuni foreground/background –

- **Date**
  - Pentru fiecare **pixel** i:

    f_i = probabilitatea ca $i \in$ foreground

    b_i = probabilitatea ca $i \in$ background
  - Pentru fiecare **pereche de pixeli vecini** (i, j):

    p_{ij} = **penalizarea de separare** = pentru plasarea lui i și j în regiuni diferite (unul în foreground și altul în background)

Segmentarea imaginilor

Delimitare regiuni foreground/background –

- **Se cere** o partiţionare a pixelilor în două mulţimi F şi B (corespunzătoare pixelilor din foreground/backgroud) care să maximizeze

$$q(F,B) = \sum_{i \in F} f_i + \sum_{i \in B} b_i - \sum_{\substack{ij \in E \\ |F \cap \{i,j\}| = 1}} p_{ij}$$

Segmentarea imaginilor

Delimitare regiuni foreground/background –

- **Avem**

$$q(F,B)=\sum_{i}(f_i+b_i)-\sum_{i\in F}b_i-\sum_{j\in B}f_j-\sum_{\substack{ij\in E\\ |F\cap\{i,j\}|=1}}p_{ij}$$

Segmentarea imaginilor

Delimitare regiuni foreground/background –

- **Avem**

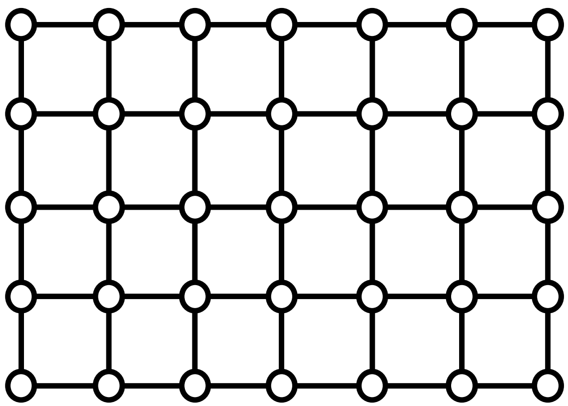
$$q(F,B) = \sum_{i}(f_i + b_i) \underbrace{- \sum_{i \in F} b_i - \sum_{j \in B} f_j - \sum_{\substack{ij \in E \\ |F \cap \{i,j\}|=1}} p_{ij}}$$

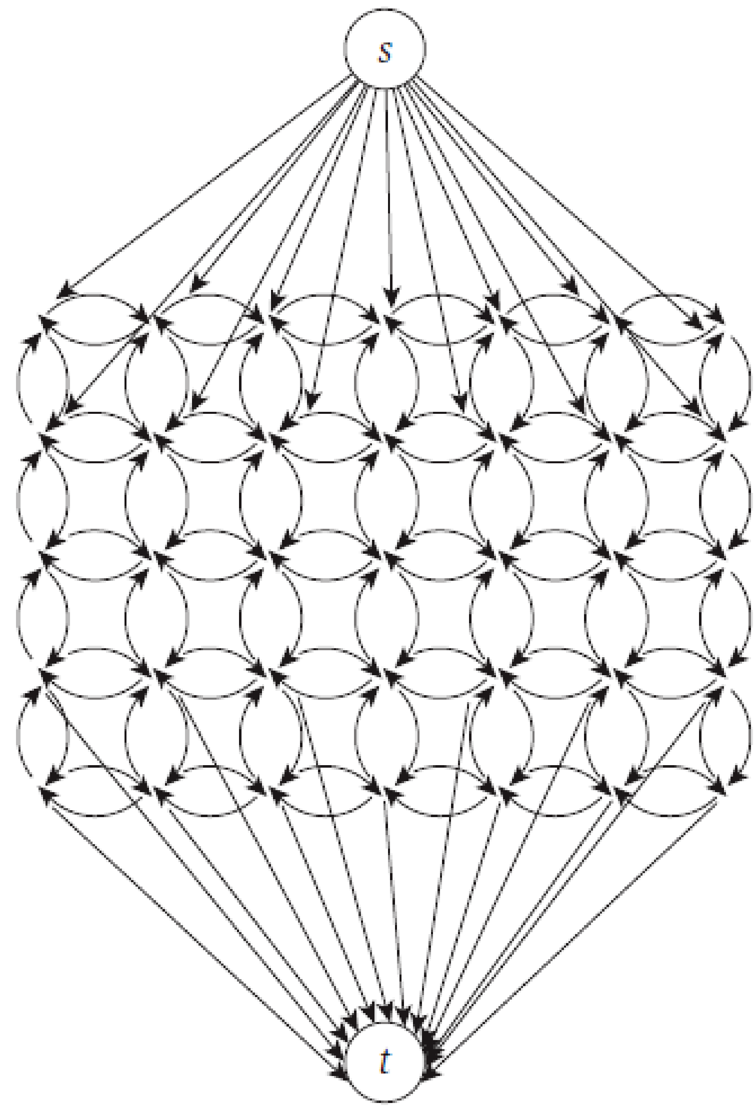
- **Problema se reduce la a minimiza**

$$r(F,B) = \sum_{i \in F} b_i + \sum_{j \in B} f_j + \sum_{\substack{ij \in E \\ |F \cap \{i,j\}|=1}} p_{ij}$$

Segmentarea imaginilor

Asociem problemei o reţea N

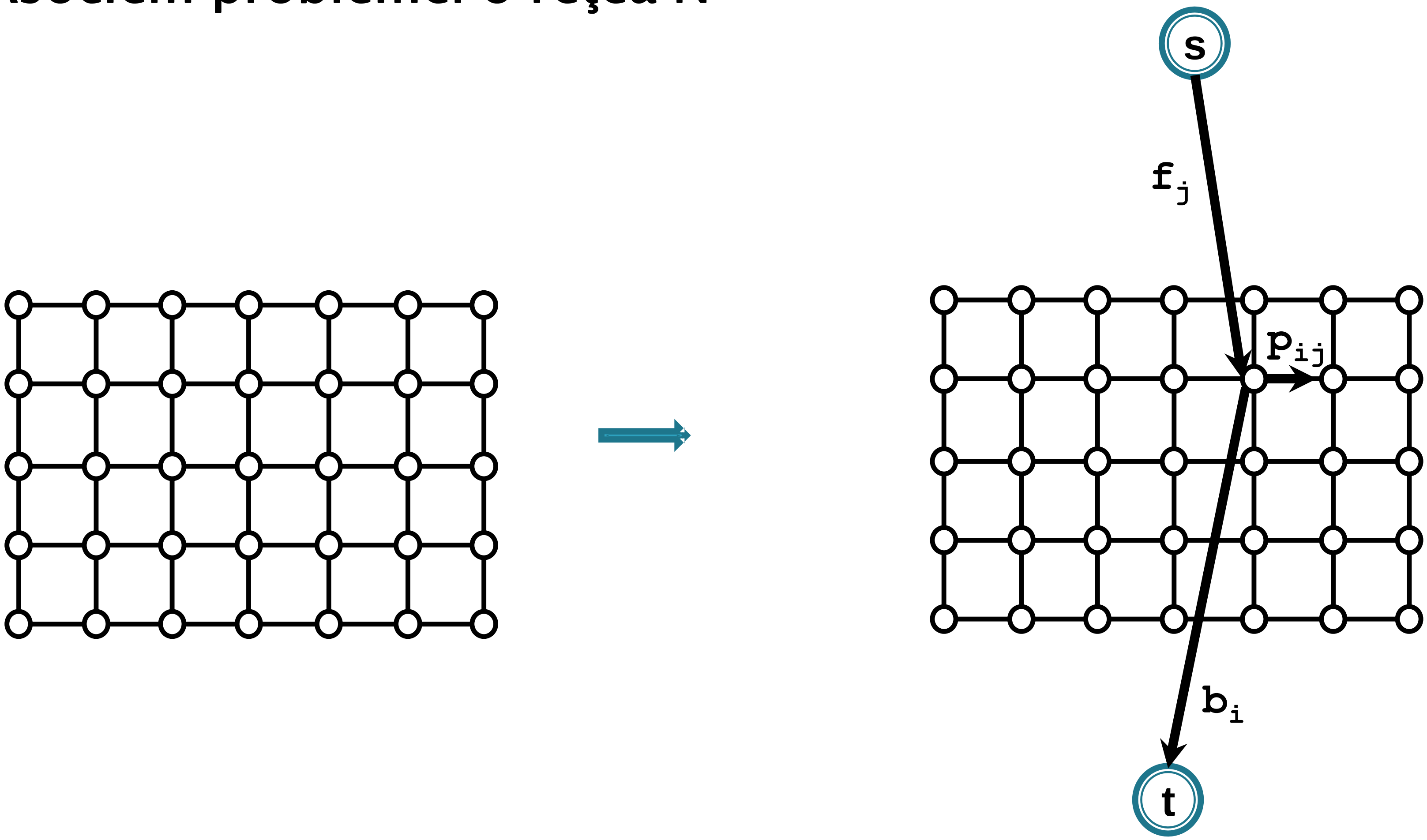




Jon Kleinberg, Éva Tardos, **Algorithm Design**, Addison-Wesley 2005

Segmentarea imaginilor

Asociem problemei o reţea N

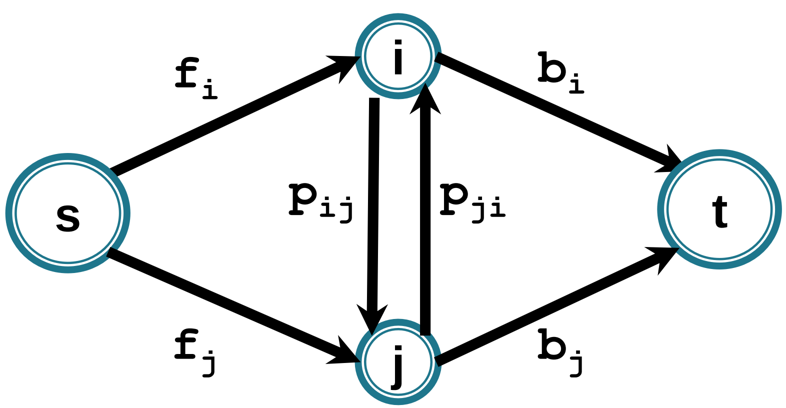


Segmentarea imaginilor

Delimitare regiuni foreground/background –

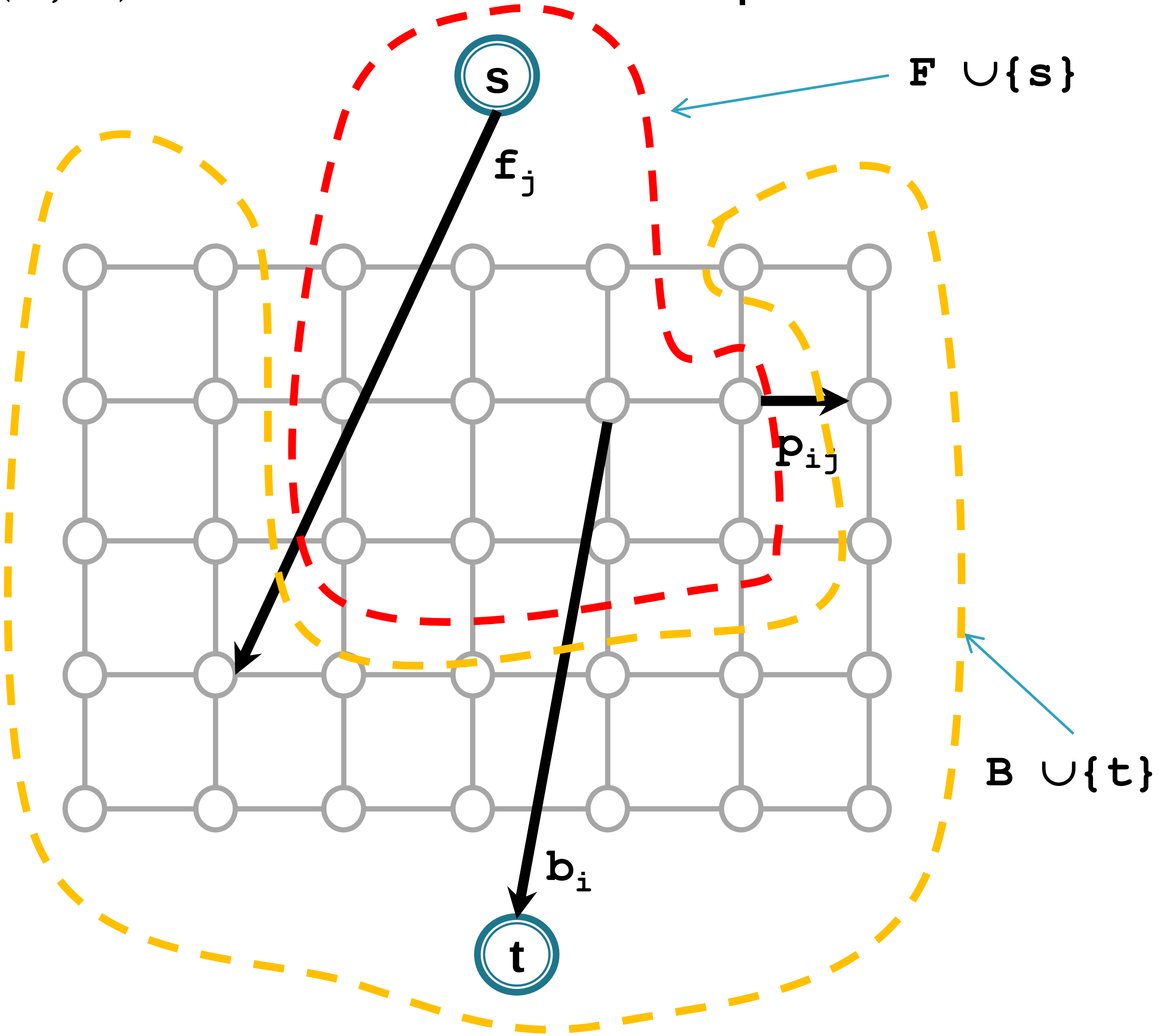
- Rețeaua N
  - noduri noi s, t
  - arce si, $c(si) = f_i$ pentru orice i
  - arce it, $c(it) = b_i$ pentru orice i
  - se înlocuiește muchie ij cu 2 arce

$$c(ij) = c(ji) = p_{ij}$$



Segmentarea imaginilor

Pentru K = (X, Y) tăietură în N avem capacitatea:



Segmentarea imaginilor

Pentru K = (X, Y) tăietură în N, F = X – {s}, B = Y – {t} avem

$$c(X,Y) = ?$$

Segmentarea imaginilor

Pentru K = (X, Y) tăietură în N, F = X – {s}, B = Y – {t} avem

$$c(X,Y) = \sum_{i \in F} c(it) + \sum_{j \in B} c(sj) + \sum_{\substack{ij \in E \\ |F \cap \{i,j\}|=1}} c(ij) =$$

Segmentarea imaginilor

Pentru K = (X, Y) tăietură în N, F = X – {s}, B = Y – {t} avem

$$c(X,Y) = \sum_{i \in F} c(it) + \sum_{j \in B} c(sj) + \sum_{\substack{ij \in E \\ |F \cap \{i,j\}|=1}} c(ij) =$$

$$= \sum_{i \in F} b_i + \sum_{j \in B} f_j + \sum_{\substack{ij \in E \\ |F \cap \{i,j\}|=1}} p_{ij} = r(F,B)$$

Segmentarea imaginilor

Concluzii

- Pentru K = (X, Y) tăietură în N cu F = X – {s}, B = Y – {t} avem

$$c(X,Y) = r(F,B)$$

Rezultă:

determinarea unei segmentări care maximizează măsura de performanţă q (adică a unei partiții (F,B) a mulțimii pixelilor cu r(F, B) minimă) $\Leftrightarrow$

determinarea unei tăieturi (X, Y) de capacitate minimă:

F = X – {s}, B = Y – {t}

https://docs.opencv.org/trunk/d8/d83/tutorial_py_grabcut.html