UNIUNEA EUROPEANĂ

GUVERNUL ROMÂNIEI

Instrumente Structurale
2007-2013

Platformă de e-learning și curriculă e-content pentru învățământul superior tehnic

Proiectarea Algoritmilor

18. Algoritmul Floyd-Warshall (Roy-Floyd)

Bibliografie

- [1] R. Sedgewick, K. Wayne - Algorithms and Data Structures Fall 2007 – Curs Princeton - http://www.cs.princeton.edu/~rs/AlgsDS07/

- [2] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms*,

Floyd-Warshall (Roy-Floyd)

- Algoritm prin care se calculează distanțele minime între oricare 2 noduri dintr-un graf (drumuri optime multipunct-multipunct).
- Exemplu clasic de programare dinamică.
- Idee: la pasul k se calculează cel mai bun cost între u și v folosind cel mai bun cost u..k si cel mai bun cost k..v calculat până în momentul respectiv.
- Se aplică pentru grafuri ce nu conțin cicluri de cost negativ.

Notații

- $G = (V,E)$; $V = \{1, 2, \ldots, n\}$;
- $w : V \times V \rightarrow \Re$; $w(i,i) = 0$; $w(i,j) = \infty$ dacă $(i,j) \notin E$;
- $d^k(i,j)$ = costul drumului i..j construit astfel încât drumul trece doar prin noduri din mulțimea $\{1, 2, .., k\}$;
- $\delta(i,j)$ = costul drumului optim i..j; $\delta(i,j) = \infty$ dacă $\nexists$ i..j;
- $\delta^k(i,j)$ = costul drumului optimi i..j ce trece doar prin noduri din mulțimea $\{1, 2, .., k\}$; $\delta^k(i,j) = \infty$ dacă $\nexists$ i..j;
- $p^k(i,j)$ = predecesorul lui j pe drumul i..j ce trece doar prin noduri din mulțimea $\{1, 2, .., k\}$.

Teorema Floyd - Warshall

- Teoremă: Fie formulele de mai jos pentru calculul valorii $d^k(i,j)$, $0 < k \le n$:
  - $d^0(i,j) = w(i,j)$;
  - $d^k(i,j) = \min\{d^{k-1}(i,j), d^{k-1}(i,k) + d^{k-1}(k,j)\}$, pentru $0 < k \le n$;

  Atunci $d^n(i,j) = \delta(i,j)$, pentru $\forall$ i, j $\in$ V

- Dem:
  - Prin inducție după k dem. că $d^k(i,j) = \delta^k(i,j)$. (next slide)
  - Pt. k = n, i..j trece prin $\forall$ v $\in$ V si avem $d^k(i,j) \le d^{k-1}(i,j)$, $\forall$ k = 1,n → $d^n(i,j) \le d^{k-1}(i,j)$, $\forall$ k = 1,n
  - Din $d^k(i,j) = \delta^k(i,j)$ → $d^n(i,j) = \delta^n(i,j) \le d^{k-1}(i,j) = \delta^{k-1}(i,j)$, $\forall$ k = 1,n → $d^n(i,j) = \delta^n(i,j) = \delta(i,j)$

Demonstrație teorema Floyd - Warshall

- K = 0: 0 noduri intermediare → i..j = (i,j), la fel ca inițializarea $d^0(i,j) = w(i,j)$;
- $0 < k \le n$: $d^{k-1}(i,j) = \delta^{k-1}(i,j) \rightarrow d^k(i,j) = \delta^k(i,j)$
  - a) k $\notin$ drumului optim i..j: drumul optim nu se modifică

    $(\delta^{k-1}(i,j) = \delta^k(i,j) \le \delta^{k-1}(i,k) + \delta^{k-1}(k,j))$
  - $d^k(i,j) = \min\{d^{k-1}(i,j), d^{k-1}(i,k) + d^{k-1}(k,j)\}$

    $d^k(i,j) = \min\{\delta^{k-1}(i,j), \delta^{k-1}(i,k) + \delta^{k-1}(k,j)\} = \delta^{k-1}(i,j) = \delta^k(i,j)$
  - b) k $\in$ drumului optim i..j: i..j se descompune în i..k și k..j optime $(\delta^{k-1}(i,k) = d^{k-1}(i,k)$ și $\delta^{k-1}(k,j)= d^{k-1}(k,j))$ și

    $\delta^k(i,j) = \delta^{k-1}(i,k) + \delta^{k-1}(k,j)$.
  - i..j optim → $\delta^k(i,j) \le \delta^{k-1}(i,j)$
  - $d^k(i,j) = \min\{d^{k-1}(i,j), d^{k-1}(i,k) + d^{k-1}(k,j)\}$
  - $d^k(i,j) = \min\{\delta^{k-1}(i,j), \delta^{k-1}(i,k) + \delta^{k-1}(k,j)\} = \delta^{k-1}(i,k) + \delta^{k-1}(k,j) = \delta^k(i,j)$

Algoritm Floyd-Warshall

Floyd-Warshall(G)

- **Pentru** i **de la** 1 **la** n
  - **Pentru** j **de la** 1 **la** n // inițializări
    - $d^0(i,j) = w(i,j)$
    - **Dacă** $(w(i,j) == \infty)$
      - $p^0(i,j) = null;$
    - **Altfel** $p^0(i,j) = i;$
- **Pentru** k **de la** 1 **la** n
  - **Pentru** i **de la** 1 **la** n
    - **Pentru** j **de la** 1 **la** n
      - **Dacă** $(d^{k-1}(i,j) > d^{k-1}(i,k) + d^{k-1}(k,j))$ // determinăm minimul
        - $d^k(i,j) = d^{k-1}(i,k) + d^{k-1}(k,j)$
        - $p^k(i,j) = p^{k-1}(k,j);$ // și actualizăm părintele
      - **Altfel**
        - $d^k(i,j) = d^{k-1}(i,j)$
        - $p^k(i,j) = p^{k-1}(i,j);$

Complexitate?

$O(V^3)$

Complexitate spațială?

$\mathbf{O(V^3)}$

Observație

- Putem folosi o singură matrice în loc de n?

- Problemă: în pasul k, pt k < i și k < j, d(i,k) și d(k,j) folosite la calculul d(i,j) sunt $d^k(k,j)$ și $d^k(i,k)$ în loc de $d^{k-1}(k,j)$ și $d^{k-1}(i,k)$. Dacă dem. că $d^k(k,j)= d^{k-1}(k,j)$ și $d^k(i,k)=d^{k-1}(i,k)$, atunci putem folosi o singură matrice.

- Dar:
  - $d^k(k,j) = d^{k-1}(k,k) + d^{k-1}(k,j) = d^{k-1}(k,j)$
  - $d^k(i,k) = d^{k-1}(i,k) + d^{k-1}(k,k) = d^{k-1}(i,k)$

- → Algoritm modificat pentru a folosi o singura matrice → complexitate spațială: $O(n^2)$.

Algoritm Floyd-Warshall

Floyd-Warshall2(G)

- **Pentru** i **de la** 1 **la** n
  - **Pentru** j **de la** 1 **la** n // inițializări
    - d(i,j) = w(i,j)
    - **Dacă** (w(i,j) == ∞)
      - p(i,j) = null;
    - **Altfel** p(i,j) = i;
- **Pentru** k **de la** 1 **la** n
  - **Pentru** i **de la** 1 **la** n
    - **Pentru** j **de la** 1 **la** n
      - **Dacă** (d(i,j) > d(i,k) + d(k,j)) // determinăm minimul
        - d(i,j) = d(i,k) + d(k,j)
        - p(i,j) = p(k,j); // și actualizăm părintele

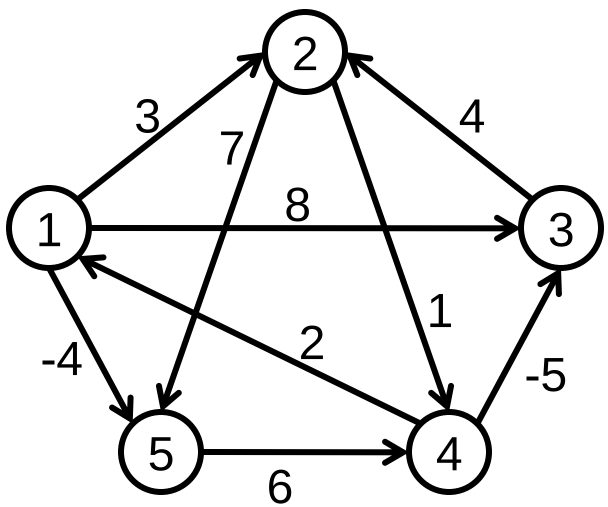
Complexitate?

$O(V^3)$

Complexitate spațială?

$O(V^2)$

Exemplu (I)



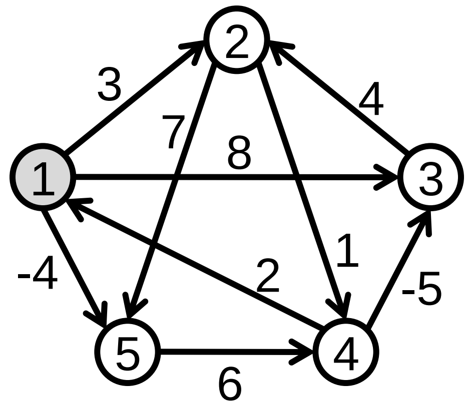
$$D = \begin{bmatrix} 0 & 3 & 8 & \infty & -4 \\ \infty & 0 & \infty & 1 & 7 \\ \infty & 4 & 0 & \infty & \infty \\ 2 & \infty & -5 & 0 & \infty \\ \infty & \infty & \infty & 6 & 0 \end{bmatrix}$$

$$p = \begin{bmatrix} nil & 1 & 1 & nil & 1 \\ nil & nil & nil & 2 & 2 \\ nil & 3 & nil & nil & nil \\ 4 & nil & 4 & nil & nil \\ nil & nil & nil & 5 & nil \end{bmatrix}$$

Exemplu (II)

$$D = \begin{bmatrix} 0 & 3 & 8 & \infty & -4 \\ \infty & 0 & \infty & 1 & 7 \\ \infty & 4 & 0 & \infty & \infty \\ 2 & \infty & -5 & 0 & \infty \\ \infty & \infty & \infty & 6 & 0 \end{bmatrix}$$

$$D = \begin{bmatrix} 0 & 3 & 8 & \infty & -4 \\ \infty & 0 & \infty & 1 & 7 \\ \infty & 4 & 0 & \infty & \infty \\ 2 & 5 & -5 & 0 & -2 \\ \infty & \infty & \infty & 6 & 0 \end{bmatrix}$$



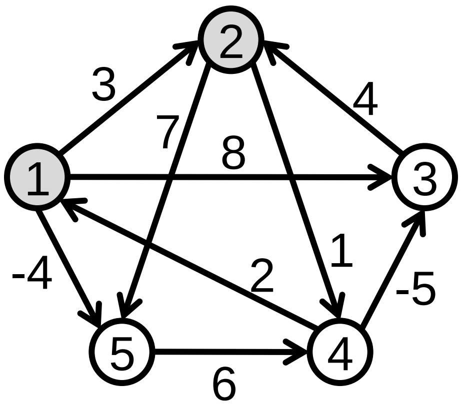
$$p = \begin{bmatrix} nil & 1 & 1 & nil & 1 \\ nil & nil & nil & 2 & 2 \\ nil & 3 & nil & nil & nil \\ 4 & nil & 4 & nil & nil \\ nil & nil & nil & 5 & nil \end{bmatrix}$$

$$p = \begin{bmatrix} nil & 1 & 1 & nil & 1 \\ nil & nil & nil & 2 & 2 \\ nil & 3 & nil & nil & nil \\ 4 & 1 & 4 & nil & 1 \\ nil & nil & nil & 5 & nil \end{bmatrix}$$

Exemplu (III)

$$D = \begin{bmatrix} 0 & 3 & 8 & \infty & -4 \\ \infty & 0 & \infty & 1 & 7 \\ \infty & 4 & 0 & \infty & \infty \\ 2 & 5 & -5 & 0 & -2 \\ \infty & \infty & \infty & 6 & 0 \end{bmatrix}$$

$$D = \begin{bmatrix} 0 & 3 & 8 & 4 & -4 \\ \infty & 0 & \infty & 1 & 7 \\ \infty & 4 & 0 & 5 & 11 \\ 2 & 5 & -5 & 0 & -2 \\ \infty & \infty & \infty & 6 & 0 \end{bmatrix}$$



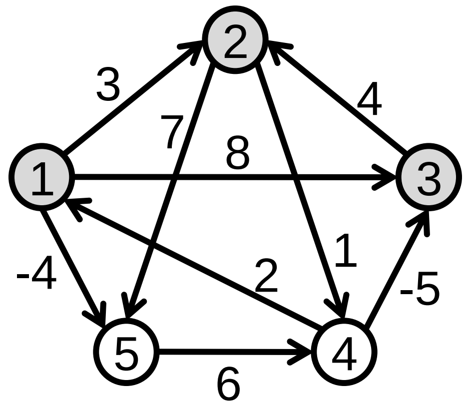
$$p = \begin{bmatrix} nil & 1 & 1 & nil & 1 \\ nil & nil & nil & 2 & 2 \\ nil & 3 & nil & nil & nil \\ 4 & 1 & 4 & nil & 1 \\ nil & nil & nil & 5 & nil \end{bmatrix}$$

$$p = \begin{bmatrix} nil & 1 & 1 & 2 & 1 \\ nil & nil & nil & 2 & 2 \\ nil & 3 & nil & 2 & 2 \\ 4 & 1 & 4 & nil & 1 \\ nil & nil & nil & 5 & nil \end{bmatrix}$$

Exemplu (IV)

$$D = \begin{bmatrix} 0 & 3 & 8 & 4 & -4 \\ \infty & 0 & \infty & 1 & 7 \\ \infty & 4 & 0 & 5 & 11 \\ 2 & 5 & -5 & 0 & -2 \\ \infty & \infty & \infty & 6 & 0 \end{bmatrix}$$

$$D = \begin{bmatrix} 0 & 3 & 8 & 4 & -4 \\ \infty & 0 & \infty & 1 & 7 \\ \infty & 4 & 0 & 5 & 11 \\ 2 & -1 & -5 & 0 & -2 \\ \infty & \infty & \infty & 6 & 0 \end{bmatrix}$$



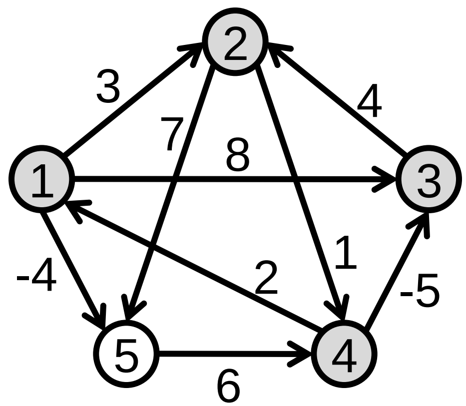
$$p = \begin{bmatrix} nil & 1 & 1 & 2 & 1 \\ nil & nil & nil & 2 & 2 \\ nil & 3 & nil & 2 & 2 \\ 4 & 1 & 4 & nil & 1 \\ nil & nil & nil & 5 & nil \end{bmatrix}$$

$$p = \begin{bmatrix} nil & 1 & 1 & 2 & 1 \\ nil & nil & nil & 2 & 2 \\ nil & 3 & nil & 2 & 2 \\ 4 & 3 & 4 & nil & 1 \\ nil & nil & nil & 5 & nil \end{bmatrix}$$

Exemplu (V)

$$D = \begin{bmatrix} 0 & 3 & 8 & 4 & -4 \\ \infty & 0 & \infty & 1 & 7 \\ \infty & 4 & 0 & 5 & 11 \\ 2 & -1 & -5 & 0 & -2 \\ \infty & \infty & \infty & 6 & 0 \end{bmatrix}$$

$$D = \begin{bmatrix} 0 & 3 & \color{red}{-1} & 4 & -4 \\ \color{red}{3} & 0 & \color{red}{-4} & 1 & \color{red}{-1} \\ \color{red}{7} & 4 & 0 & 5 & \color{red}{3} \\ 2 & -1 & -5 & 0 & -2 \\ \color{red}{8} & \color{red}{5} & \color{red}{1} & 6 & 0 \end{bmatrix}$$



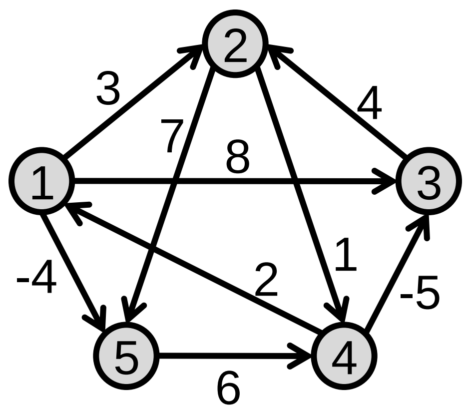
$$p = \begin{bmatrix} nil & 1 & 1 & 2 & 1 \\ nil & nil & nil & 2 & 2 \\ nil & 3 & nil & 2 & 2 \\ 4 & 3 & 4 & nil & 1 \\ nil & nil & nil & 5 & nil \end{bmatrix}$$

$$p = \begin{bmatrix} nil & 1 & \color{red}{4} & 2 & 1 \\ \color{red}{4} & nil & \color{red}{4} & 2 & \color{red}{1} \\ \color{red}{4} & 3 & nil & 2 & \color{red}{1} \\ 4 & 3 & 4 & nil & 1 \\ \color{red}{4} & \color{red}{3} & \color{red}{4} & 5 & nil \end{bmatrix}$$

Exemplu (VI)

$$D = \begin{bmatrix} 0 & 3 & -1 & 4 & -4 \\ 3 & 0 & -4 & 1 & -1 \\ 7 & 4 & 0 & 5 & 3 \\ 2 & -1 & -5 & 0 & -2 \\ 8 & 5 & 1 & 6 & 0 \end{bmatrix}$$

$$D = \begin{bmatrix} 0 & 1 & -3 & -2 & -4 \\ 3 & 0 & -4 & 1 & -1 \\ 7 & 4 & 0 & 5 & 3 \\ 2 & -1 & -5 & 0 & -2 \\ 8 & 5 & 1 & 6 & 0 \end{bmatrix}$$



$$p = \begin{bmatrix} nil & 1 & 4 & 2 & 1 \\ 4 & nil & 4 & 2 & 1 \\ 4 & 3 & nil & 2 & 1 \\ 4 & 3 & 4 & nil & 1 \\ 4 & 3 & 4 & 5 & nil \end{bmatrix}$$

$$p = \begin{bmatrix} nil & 3 & 4 & 5 & 1 \\ 4 & nil & 4 & 2 & 1 \\ 4 & 3 & nil & 2 & 1 \\ 4 & 3 & 4 & nil & 1 \\ 4 & 3 & 4 & 5 & nil \end{bmatrix}$$

Închiderea tranzitivă (I)

- Fie G = (V,E). Închiderea tranzitivă a lui E e un G* = (V,E*), unde

$$E^*(i,j)=\begin{cases} 1, \text{ dacă } \exists\ i..j \\ 0, \text{ dacă } \nexists\ i..j \end{cases}$$

- Poate fi determinată prin modificarea algoritmului Floyd-Warshall:
  - min $\Rightarrow$ operatorul boolean sau ($\vee$)
  - + $\Rightarrow$ operatorul boolean și ($\wedge$)

Închiderea tranzitivă (II)

Închidere_tranzitivă(G)

- **Pentru** i **de la** 1 **la** n
  - **Pentru** j **de la** 1 **la** n
    - E* (i,j) = (i,j) ∊ E ∨ i = j // inițializări
- **Pentru** k **de la** 1 **la** n
  - **Pentru** i **de la** 1 **la** n
    - **Pentru** j **de la** 1 **la** n
      - E* (i,j) = E* (i,j) ∨ (E* (i,k) ∧ E* (k,j))

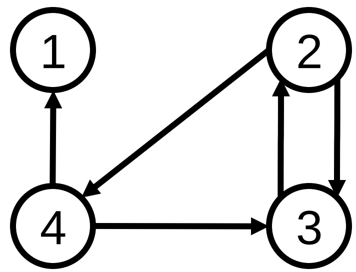
Complexitate? $O(V^3)$

Complexitate spațială? $O(V^2)$

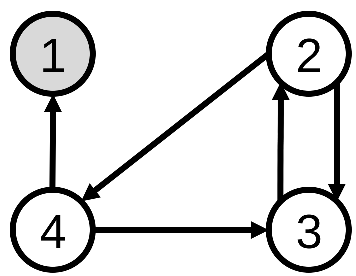
Exemplu (I)

Închidere_tranzitivă(G)

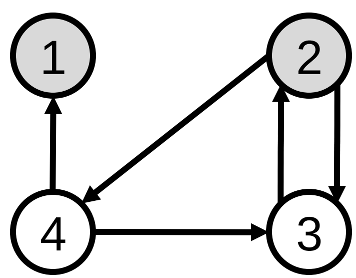
- **Pentru** i **de la** 1 **la** n
  - **Pentru** j **de la** 1 **la** n
    - E* (i,j) = (i,j) ∊ E ∨ i = j
      // inițializări
- **Pentru** k **de la** 1 **la** n
  - **Pentru** i **de la** 1 **la** n
    - **Pentru** j **de la** 1 **la** n
      - E* (i,j) = E* (i,j) ∨ (E* (i,k) ∧ E* (k,j))



$$T^{(0)} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \end{bmatrix}$$



$$T^{(1)} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \end{bmatrix}$$

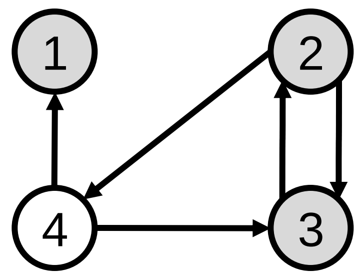


$$T^{(2)} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & \color{red}{1} \\ 1 & 0 & 1 & 1 \end{bmatrix}$$

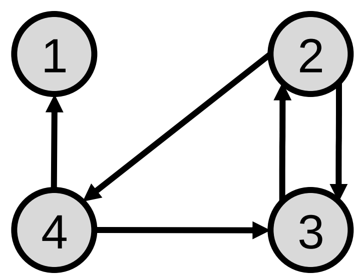
Exemplu (II)

Închidere_tranzitivă(G)

- **Pentru** i **de la** 1 **la** n
  - **Pentru** j **de la** 1 **la** n
    - E* (i,j) = (i,j) ∊ E ∨ i = j
      // inițializări
- **Pentru** k **de la** 1 **la** n
  - **Pentru** i **de la** 1 **la** n
    - **Pentru** j **de la** 1 **la** n
      - E* (i,j) = E* (i,j) ∨ (E* (i,k) ∧ E* (k,j))



$$T^{(3)} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$



$$T^{(4)} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$