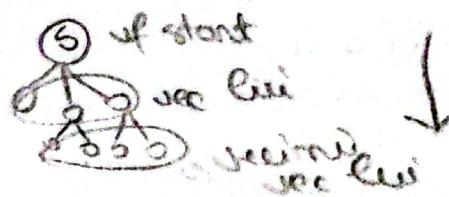


## Subiecte examen

- BT
  - 5
    - o bfs + aplicatii
    - o dfs + aplicatii
    - o algoritmi partiiali de cost minim
    - o dist + drumuri minime im grafuri

↳ TESTE 5: flux, telecură minima, lung de creștere

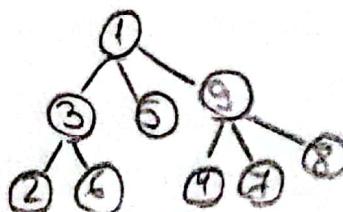
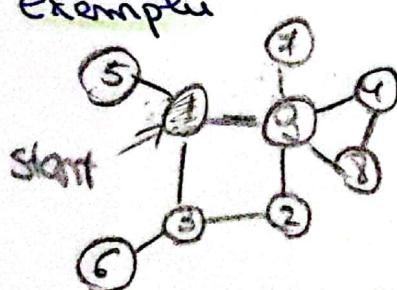
- ↳ grafuri plane
- ↳ dist intre vîrteze
- ↳ colorare



### - BFS -

- o gestionarea vf  $\Rightarrow$  coada
- o Idea alg
- adaugam nodul de start (niv 0) și-l marcam ca vizitat
- cît timpq!  $\leftarrow$  ! execuția
- extrage elem din coadă, adaugă vecinii acestuia și îl  
    marcheză ca vizitat

### Exemplu



$$2: + 25 \frac{5}{8} \frac{9}{2} 6478$$

nu mai avem vec  
 $\Rightarrow$  coada re-elib

- ↳ memm arbore  $\rightarrow$  vector tata
- ↳ pt ex nostru tata = {0, 3, 1, 9, 1, 3, 9, 9, 1}
- ↳ tata [j]: vf i din care este dreapta j
- ↳ dist [j] = lung drumeului  $s \rightarrow j$
- ↳ nivelul lui j în arbore

$$\text{dist}[j] = \text{dist}[\text{tata}[j]] + 1$$

↳ pt a parcurge vf grafului: pentru  $x$  marcat vf  
daca  $x$  nu e viz.  
 $BFS(x)$

### Complexitate:

imf  $O(m)$  + pt fiec vf parcurg

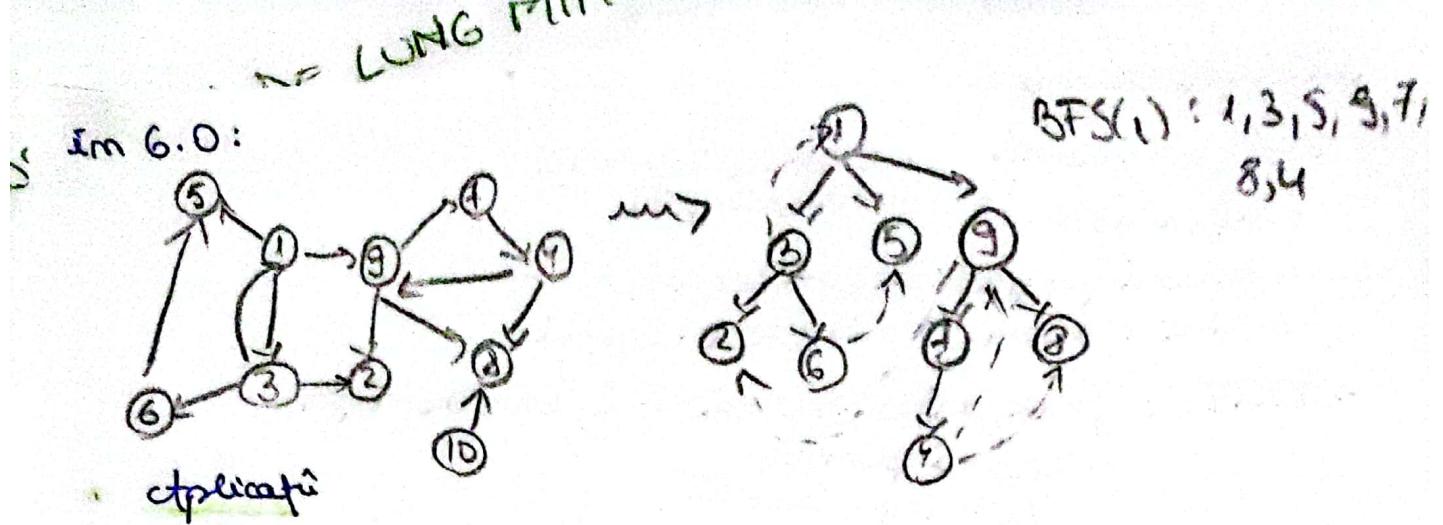
$$\hookrightarrow m^2 \cdot O(n) = O(m^2)$$

$$\hookrightarrow m^2 \cdot d(i) = O(m^2 \cdot m) = O(m^3)$$

im  $G, N \rightarrow$  muchile dim graf  $\rightarrow$  care nu sunt în arbore  $\hookrightarrow$  muchii circuite

im  $\star$ , 2-9 și 4-8 nu apar

①



Dacă comp conexe

$\Rightarrow$  arborele de noduri comp conexe în care spune  $\hookrightarrow$   
 nel BFS din modulile mewiz  $\Rightarrow$  pădure BFS  
 (toate comp conexe)

nr CC = 0;

pentru  $x \in V$  există

dacă  $viz[x] = 0$  atunci

BFS( $x$ )

$\hookrightarrow$  pădure parțială / arb parțial:

$\hookrightarrow$  total  $viz[x]$ ,  $viz[y]$ , total  $viz[z] \neq 0$

$\hookrightarrow$  transm unui mesaj în rețea

$\hookrightarrow$  Norman Enders

$\hookrightarrow$  Pasul 2 pe care nu reușește să stea

$\hookrightarrow$  Pasul minim în labirint la una din ieșiri

BFS matrice  $\Rightarrow$  alg deoarece

Determin de dist și lărguri ( $x \rightarrow y$ )

BFS( $x$ ) + afis drumul lui folos vechi de tafi

BFS( $x$ )

dacă  $viz[v] = -1$  atunci II oprire BFS

lant ( $v$ )

altfel

afis urmăriți

$\hookrightarrow$  ieșire din lab cu nr min de pași matn

$\hookrightarrow$  graf cu vf coresp celelelor și muchii coresp labirint

celulelor vecine libere

$\hookrightarrow$  parcursul BFS dim. cel un pasă găsește o ieșire

$\rightarrow$  faza( $v$ ): cunoaște dist. din graf de la  $v$  la cel mai

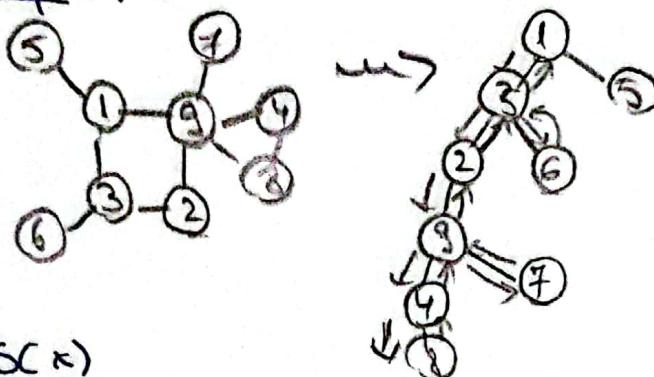
(2)

## - DFS -

- la un pas coboram in graf catre primul vecin de la v
- cel mai puternic vecin, ne intorcem pe drumul pe care am venit pana gasim un vf
- trece la primul vf de la v si reluam



Exemplu: G.N



DFS(x)

|| explorare x  
viz[x] = 1       $\rightarrow$  nivel "alb"

Pentru y ∈ E

dacă  $viz[y] = 0$  atunci

$total[y] = x$

$d[y] = d[x] + 1$  || nivel

DFS(y)

|| am terminat și finalizat = negru

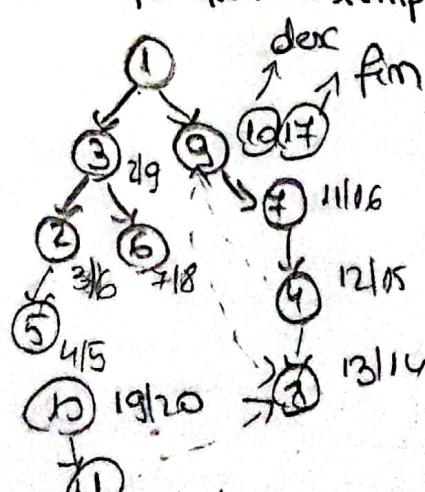
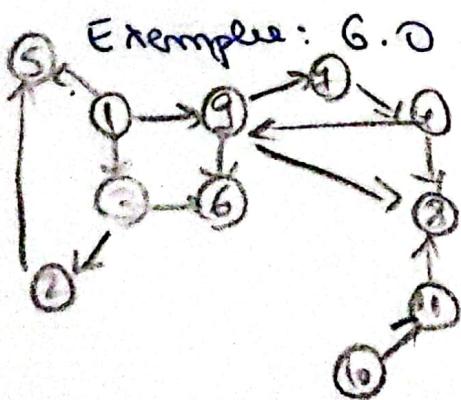
↳ determinăm eferm din graf < unele comp tone conexe

menim în parc DFS care face vf + menim în parc a fost deschis finalizat

↳ cind se incepe explozii x  $\Rightarrow$  timp  $t = 1$   
return ->

timp  $t = 1$

final(x) = timp



oferim explorare

vf de pe drumurile de la vf start  $\rightarrow$  eferm  
vf din stiva

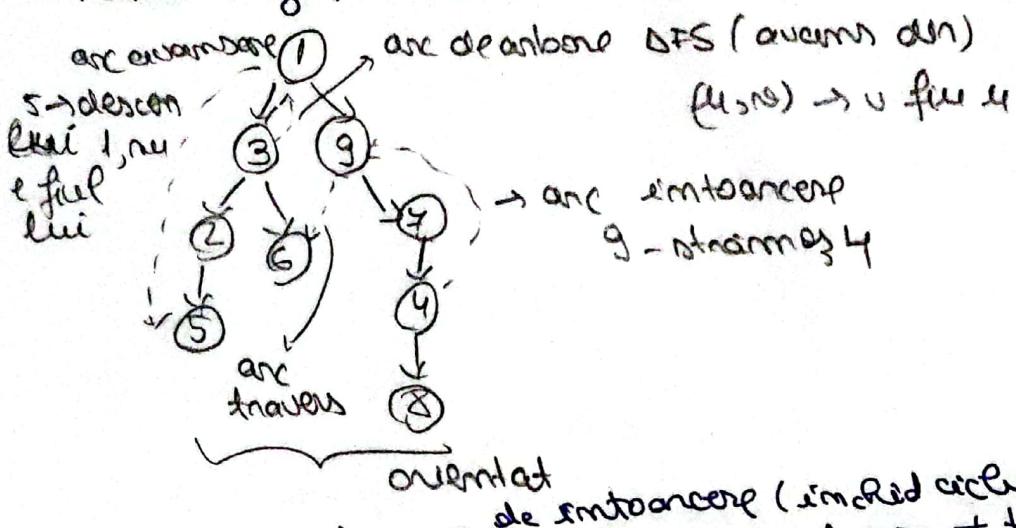
③

- o  $v \in \text{desc}(u) \rightarrow v$  este descendent în timp ce  $u$  este un cunoscător de apel
- o  $\text{desc}(u) \subset \text{desc}(v)$
- o  $\text{internal} \{ \text{desc}(u), \text{fin}(u) \} \cap \{ \text{desc}(v), \text{fin}(v) \}$  sunt alți  $\rightarrow$   
 v și u nu  
 sunt desc  
 unul altul

- o  $v$  este descendental  $u$ :

$$\Rightarrow \{\text{desc}(v), \text{fin}(v)\} \subset \{\text{desc}(u), \text{fin}(u)\}$$

- o Pentru ex graf orientat  $\rightarrow$  tipuri multe



- o Detectare multe:

$\rightarrow$  de avansare din  $\rightarrow$  multe

$\rightarrow$  de avansare înainte:  $\text{desc}(u) \subset \text{desc}(v) \subset \text{fin}(u) \subset \text{fin}(v)$

$\rightarrow$  întoarcere:  $\text{desc}(u) \subset \text{desc}(v) \subset \text{fin}(v) \subset \text{fin}(u)$

$\rightarrow$  traversare:  $\text{desc}(v) \subset \text{fin}(v) \subset \text{desc}(u) \subset \text{fin}(u)$

- o Aplicații DFS

det arbore parțial

alg complete

det circ (multe de multe / once intotdeauna)

G.N.: dacă  $\text{tata}(x) = y$  atunci "x și y nu sunt în același arbore"

$v = x_i$

căt timp  $v_i = y$  este

adăugă  $v$  la circ

$v = \text{tata}(v)$

adăugă  $y, x$  la circ

G.O

dacă  $\text{fin}(y) = 0$  atunci

căt timp  $v_i = y$  este

adăugă  $v$  la circuit

$v = \text{tata}(v)$

adăugă  $y, x$  la circ

înveță circ

$\text{fin}(x) = 1$

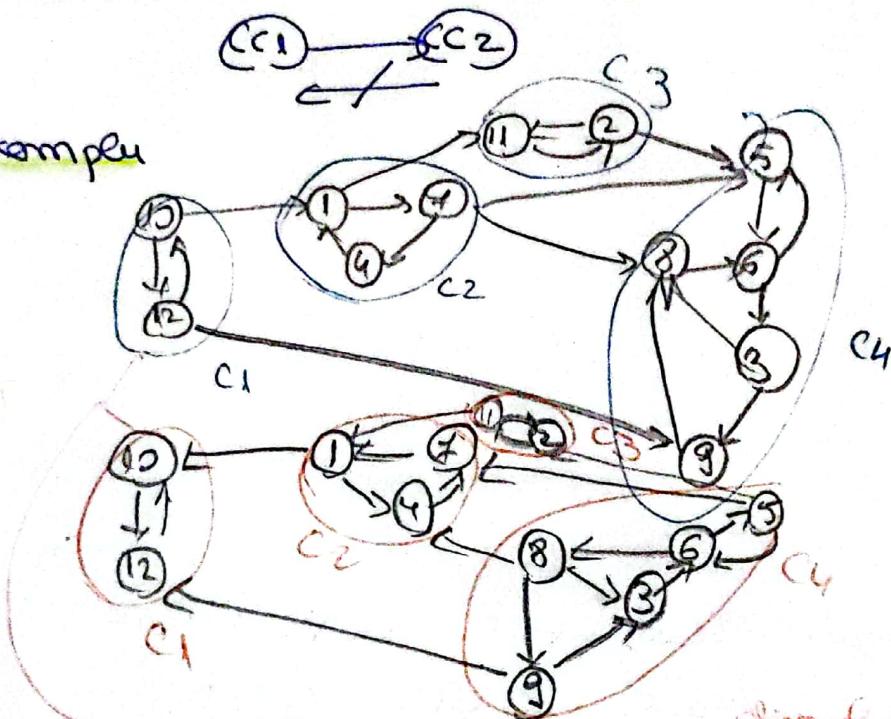


## - DETERMINARE CTC -

- o componele tone conexe în  $G_0$  = subgraf eindus al lui  $G$  tone conexe maxime  
dintre ele

- o algoritm: comp con ex. 1 = mult of acc dim 1 în  $G$  / mult im  $G^T$  (mult im  $G^T$  (mult im  $G$ )
- o comp tone conexe în  $G$  = comp tone conexe  $G^T$  DPS( $1, G^T$ )

### Exemplu



→ prima def → 10 = numărul finitelor seturi în BFS

- Folos 2 pașeageri una în  $G$ , alta în  $G^T$  ⇒ alg lui Kosaraju (în fel de ordinea de finalizare a DFS-ului său în  $G$ )
- Folos o pașeagere ⇒ alg lui Tarjan

### Alg lui Kosaraju

constituim  $G^T$

pașeagere DFS graful  $G$

↪ într-o silvă introducem fiecăruia un cap și o finală

### Stack S

DFS( $G, i$ )

uit  $\{i\} = 1$   
pentru  $i, j \in E(G)$

dacă  $\{i\} = 0$  atunci

DFS( $j$ )

pentru  $x \in V$  exec  $(S, i)$  |  $i$  e finală

dacă  $\{x\} = 0$  atunci  $DFS(G, x)$

⑤

L părare DFS graful  $G^T \rightarrow$  în ordinea în care sunt extrase din  $S$

Cât timp  $S$  este menită

$$x = \text{pop}(S)$$

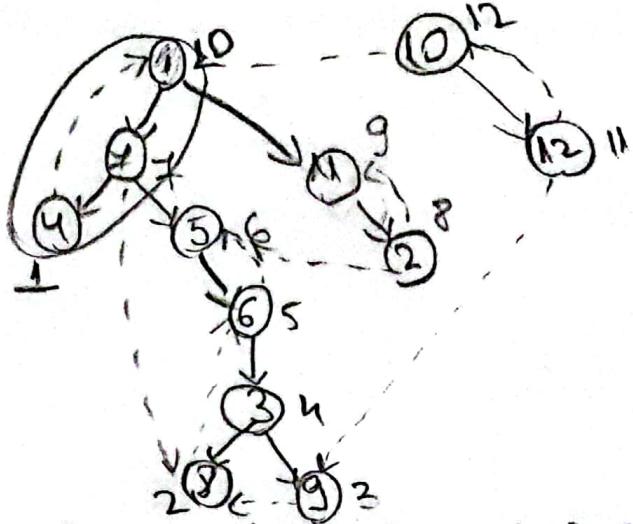
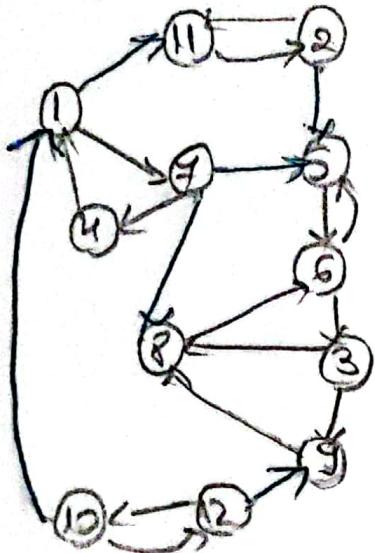
dacă  $x$  este vizitat

$$\text{DFS}(G^T, x)$$

eficiență similară cu cea în  $\text{DFS}(G^T, x)$

2 pasajele + constanță  $G^T \Rightarrow O(m+n)$

Exemplu:



Ordinea finală: 11, 2, 7, 5, 6, 3, 9, 8, 4

↳ parcurgerea  $\text{DFS}(10)$  pt  $G^T \rightarrow$  CTC : 10, 12

$\text{DFS}(1) \Rightarrow$  CTC : 1, 4, 7

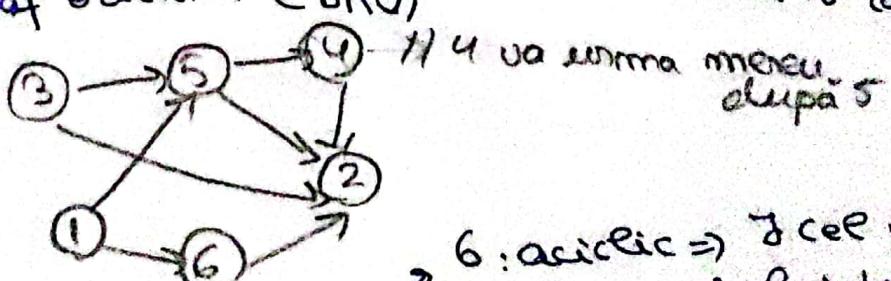
$\text{DFS}(5) \Rightarrow$  CTC : 5, 6, 8, 3, 9

### - SORTARE TOPOLOGICA -

$G = (V, E)$  graf orientat

Sortare topologică a lui  $G$  = ordonare a  $V$  a. i. dacă  $u \neq v \in V$  atunci  $u$  se află înaintea lui  $v$  ca ordineas

↳ nu e unică!  
↳ ! graf orientat (DAG)



6. aciclic  $\Rightarrow$  și cel puțin un  
văcă gradul internă  
 $d^-(v) = 0$

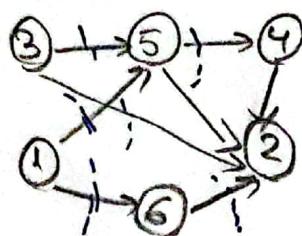
↳ primul văcă ales:

6

$\hookrightarrow$  cat timp  $|V(G)| > 0$  execută  
 alege  $v$  cu  $d^-(v) = 0$   
 adaugă  $v$  în ordonare  
 $G \leftarrow G - v$

- Implementare
  - permută cu toate vf cu grad  $\leq$  interm  $0$  și le adaugăm în coadă
  - repetăm extragând un vf din coadă
    - il elimin din graf (scadem cu internele adăugate în coadă vec cu  $g.i = 0$  vec)

exemplu



$C: 1 \underline{3} 6 5 4 2$

$\rightarrow$  Algoritm: bazat pe DFS  
 $\hookrightarrow$  dacă  $fim(v) = \text{max}$  la care a fost finalizat  
 $\quad\quad\quad$  vf și în parc DFS  
 $\hookrightarrow v \in E \Rightarrow fim(u) > fim(v)$   
 sortare ascendentă după  $fim$   
 $\hookrightarrow$  memorăm vf într-o stivă pe măsură  
 finalizării lor; ord scăderii din stivă  
 "sortare topologică"

Stack S

$DFS(i)$

$viz[i] = 1$

pentru  $i \in E$

dacă  $viz[j] = 0$  atunci

$DFS(j)$

push ( $S, i$ )  $\rightarrow i$  e finalizat

pentru  $i \in V$  execută

dacă  $viz[i] = 0$  at

$DFS(i)$

cât timp  $S$  este nevidabil exec

$n = pop(S)$

adaugă  $n$  în sortare

pop  
stack  
cont  
min

- GRAF BIPIRIT -  
 $\hookrightarrow$  graf neorientat  $G = (V, E)$  = bipartit ( $\Leftrightarrow$ ) și se poate să fie  
 în 2 mulțimi  
 $V_1 \cap V_2 = \emptyset$

$$V_1 = V_1 \cup V_2$$

$$V_1 \cap V_2 = \emptyset$$

$\wedge$  muchile  $e \in E$  are o extremitate în  $V_1$  și cealaltă în  $V_2$   
 $|e \cap V_1| = |e \cap V_2| = 1$

$\hookrightarrow G = (V, E)$  - bipartit ( $\Leftrightarrow$ )  
 și o 2-colorare proprie sau bicolorare  
 $c: V \rightarrow \{1, 2\}$

$\hookrightarrow G = (V, E)$  bip  $\Rightarrow \chi(G) \leq 2$

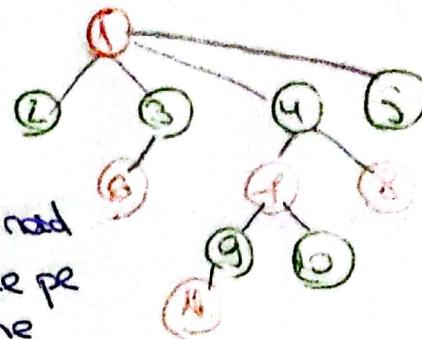
$\hookrightarrow$  un arbore - graf bip

o colorare proprie cu

cu cel mult 2 culori ne

poate obține:

$\hookrightarrow$  fixăm o rad  
 col ur de pe  
 mijloc



cu 1 și impune 2

teorema König:  $G$  este bipartit ( $\Leftrightarrow$  toate ciclurile elem  
 din  $G$  sunt pare

## - ARBORE PARTIITI -

$\hookrightarrow$  graf conex, fără cicluri cu același costul oricărui

mijlociu

$\hookrightarrow$  grafuri ponderate:  $G = (V, E)$  ponderat  
 $= w: E \rightarrow \mathbb{R}$  funcție pondere

$$G = (V, E, w)$$

rezentare

matrice de costuri

$$w_{ij} = \begin{cases} 0 & \text{dcl } i=j \\ w(i,j) & \text{dcl } i \neq j \\ \infty & \text{dcl } i, j \notin E \end{cases}$$

lățime adiacență  
 confință nodurilor  
 pondere

1: 3/11

2: 1/15, 4/10

3: 4/18

4:

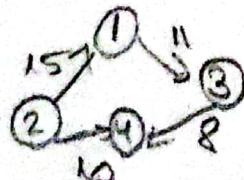
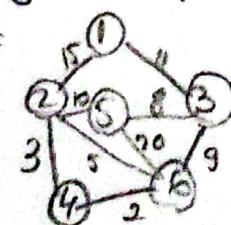
listă de muchii lățice

1 3 11

2 1 15

2 4 10

3 4 8



0	$\infty$	$\infty$	$\infty$
15	0	$\infty$	10
0	$\infty$	0	8
$\infty$	$\infty$	$\infty$	0

8

✓ APM al lui G = arbore parțial T minim al lui G cu  
 $w(T_{min}) = \min w(T) | T \text{ arb parf}, G$

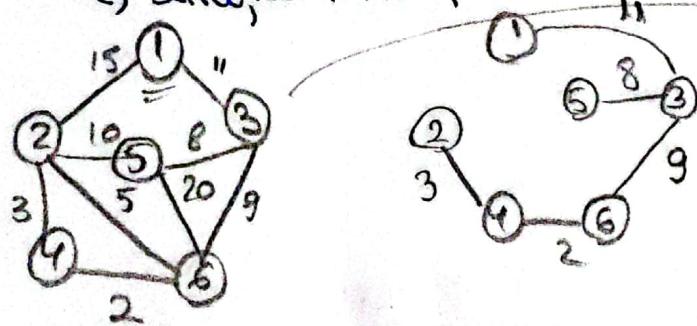
## ✓ Algoritme

### ✓ Kruskal

- initial  $T = (V; \emptyset)$
- pentru  $i \leftarrow 1, m-1$   
 ✓ alegem o muchie nu cu cost minim a.t u, v sunt în c.c definite  
 $(T + uv \text{ aciclic})$   
 $\hookrightarrow E(T) = E(T) \cup uv$

✓ la un pas este nel o muchie de cost minim dim(G) care nu formează cicluri cu muchile deja nel (unelk 2 c.c dim grafel deja construit)

✓ initial: muf izolată  $\Rightarrow m \text{ c.c}$



✓ pentru a nel o muchie de cost minim  $\Rightarrow$  ordonăm crescător muchiile după cost  $\hookrightarrow$  com o muchie în ac ordine  $\hookrightarrow$  listă muchii: extremitate costul

✓ urm primul-a parcursene dacă muchiile formează un lant

$\Rightarrow$  struct pt mult dinj  
 ✓ alegem costum  
 reprez pt fiecare

✓ operatii

✓ initializare(u)

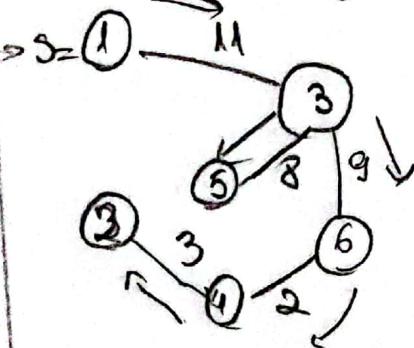
✓ reprez(u)

✓ reprez(u, v)  $\Leftrightarrow$  reprez(u)  $\neq$  reprez(v)

### ✓ Prim

- n-odifel de start
- initial  $T = (\{N\}; \emptyset)$
- pentru  $i \leftarrow 1, m-1$   
 ✓ alege o muchie nu cu cost minim a.t u  $\in V(T)$  și  $v \notin V(T)$   
 $\rightarrow V(T) = V(T) \cup \{v\}$   
 $\rightarrow E(T) = E(T) \cup uv$

✓ se permite de la un vf  
 ✓ la un pas este nel o muchie de cost minim de la un vf  
 deja adaugat la arbore la unul neadăugat



✓ pentru fiecare vf (mes)  
 memorăm doar o muchie de cost minim c.ye il unelk ce un vf dim arbore (selectat)

✓ ex pt vf 2 memorăm (2,5)

- variante -

•  $O(m^2) | O(m \log m)$

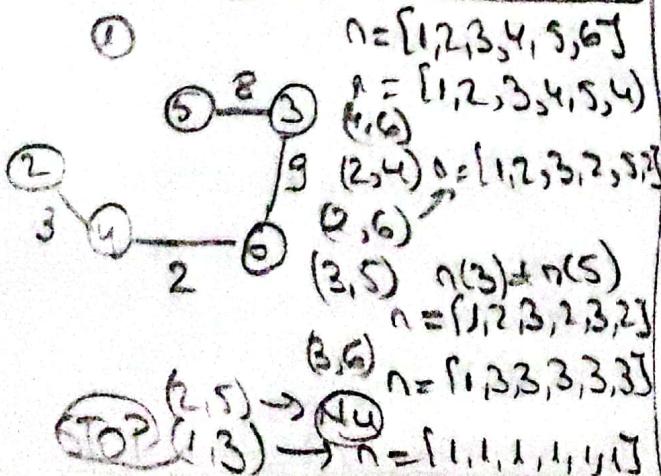
memorăm la fiecare pt muchie  
 pentru fiecare vf care cost este deja im arbore

⑨

Sortezä(E)  
 for ( $n = 1 \rightarrow m$ )  
     Initialisierung ( $v$ )  
     mrmehrsel = 0  
 for ( $u \in E$ )  
         if ( $\text{Reprez}(u) \neq \text{Reprez}(v)$ )
   
              $E(T) = E(T) \cup u \cup v$   
             Remerkte ( $u, v$ )  
             mrmehrsel + 1  
             if ( $mrmehrsel = n-1$ )
   
                 STOP;
   
 y
   
 • Sortare:  $O(m \log m) = O(m \log m)$   
 Treffzone  $\cdot 2m \rightarrow O(m)$   
 Reprez  $\cdot 2m \rightarrow O(m)$   
 Remerkte  $\cdot (m-1) \rightarrow O(m^2)$ 

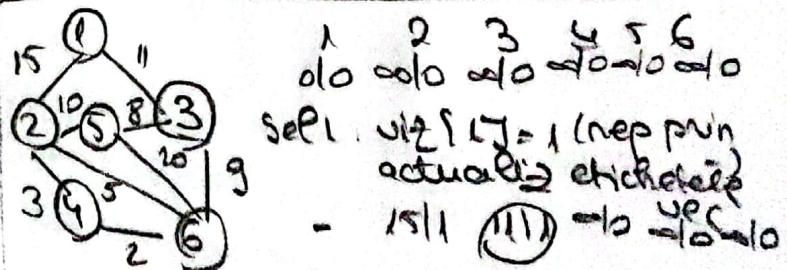

---

 $O(m \log m + m^2)$ 
  
 → (int  $u$ )  
 $n[u] = u; O(1)$ 
  
 → (int  $u$ )  
 return  $n[u]; O(1)$ 
  
 → (int  $u$ , int  $v$ )  
 $n_1 = \text{Reprez}(u); O(m)$ 
  
 $n_2 = \text{Reprez}(v); O(m)$ 
  
 for ( $u=1, m$ )  
     if ( $n[u] == n_2$ )
   
 $n[u] = n_1; O(1)$



sau  
 $\hookrightarrow$  fiec rețea și asociem  
 o  $d[u]$  = cost minim al unei  
 muchii de la  $u$  la  $v$   
 și de la  $v$  în arbore  
 o  $tata[u] = v$  dlem arbore pentru  
 care se realizează  
 min  
 $(u, tata[u]) =$  muchia de  
 cost min  
 de la  $u$  la  
 un v din  
 arbore  
 o  $d[u] = w\{u, tata[u]\}$

de un pas  
 o ne alege un vf cu eticheta d  
 minimă care nu este încă în  
 arbore și se adaugă la arbore  
 muchia  $(tata[u], u)$   
 o actualizare etichete valori  
 dacă  $w(u, v) < d[v]$  at  
 $d[v] = w\{u, v\}$   
 $tata[v] = u$



Sep 3 - 1511 - 10 813 913  
 Sep 5 - 1015 - 10 - 913  
 Sep 6 - 516 - 216 - -  
 Sep 4 - 314 - - - -  
 Sep 2 - - - - -  
 date neg - (10)

o altă reprezentare

- o var2 : struct puncte mult  
dintre Union / Find
- ↳ comp com arbori
- ↳ reprez comp = radacina
- ↳ numere : radicei arbori  
devine fiul  
rad celeilalte
- o initializare :  
 $\text{total}[u] = \text{f}(u) = 0$

o reprez:  
 $\text{while} (\text{total}[u] = 0)$   
 $u = \text{total}[u]$   
 $\text{return } u; \boxed{O(\log m)}$

o numere:  
 $\boxed{O(m \log n)}$   
 $\text{int } nu, nv;$   
 $nu = \text{repraz}(u);$   
 $nv = \text{repraz}(v);$   
 $\text{if } (\text{f}(nu) > \text{f}(nv))$   
 $\quad \text{total}[nv] = nu;$   
 $\text{else}$   
 $\quad \text{if } (\text{f}(nu) ==$   
 $\quad \quad \quad \text{f}(nv))$   
 $\quad \quad \quad \text{f}(nv) = \text{f}(nv) + 1$

→ completează total:  
 $\boxed{O(m \log n)}$

o - ref de start  
 inserție într-o v  
 puncte  $u \in V$  exec  
 $d[u] = \infty, \text{total}[u] = 0$   
 $d[u] = 0$   
 cof simp  $Q \neq \emptyset$  exec  
 extrage  $u \in Q$   $u \in Q$  cu  
 puncte fieri  $v \in V$   $w(u, v) < d[u]$   
 dacă  $v \in Q$  și  $w(u, v) < d[v]$

atunci  
 $d[v] = w(u, v)$   
 $\text{total}[v] = u$   
 $\text{done } (u, \text{total}[u]) u \neq \top$   
 ↳ varianta optimă:  
 mem vfdm Q într-un min heap  
 $\rightarrow$  extrage vfh cu echipă  
 d minimă din Q pt fier  
 $v$  adiacent cu  $u$  exec  
 dacă  $v \in Q$  și  $w(u, v) <$   
 $d[v]$

atunci  
 $d[v] = w(u, v)$   
 $\text{total}[v] = u$

înțeleg  $Q : O(m)$   
 $m \cdot$  extragere vfdm  $O(m \log n)$   
 actualiz echipă vecini:  
 $\boxed{O(m \log n)}$

$O(m \log n)$

Corespond:

$A \leftarrow \emptyset$  (multimea muchilor nel în arborele comun)

puncte  $s = 1, m \rightarrow$  exec  
 alege o muchie e.g. A uhey's apcm

$A = A \cup \{s\}$

return  $T = (V, A)$

$S \subseteq V$  muchie din A are ambele extremități în S / V-S

$s = uv \rightarrow$  muchie cost minimum cu extrem în

$S$  și extă în  $V-S$

$A \cup \{s\} \subseteq$  apcm

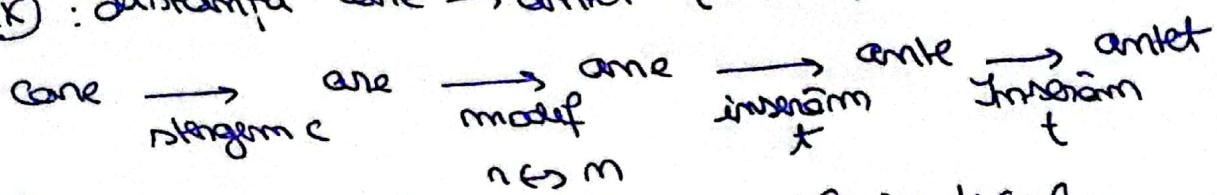
⑪

## - CLUSTERING

- gruparea unor obiecte în k clase care să mai bine se păstreze
- $\downarrow$  distanța de editare - număr minim de op (inserări, ștergeri) de către unul sau mai multe caractere pentru a transform prima secuime în a două

- distanța de editare denumită edit

ex: distanța care  $\rightarrow$  anotat (este u)



- k-clustering a lui S  $\rightarrow$  particionare a lui S în k sub集 menite (numărul claseelor)

$$b = (c_1, \dots, c_k)$$

- Gradul de separare  $b$

= dist min dintre 2 obiecte din clase dif

= dist min dintre 2 clase ale lui b

$$\text{sep}(b) = \min_{i,j} d(c_i, c_j) \mid i, j \in \{1, \dots, k\}$$

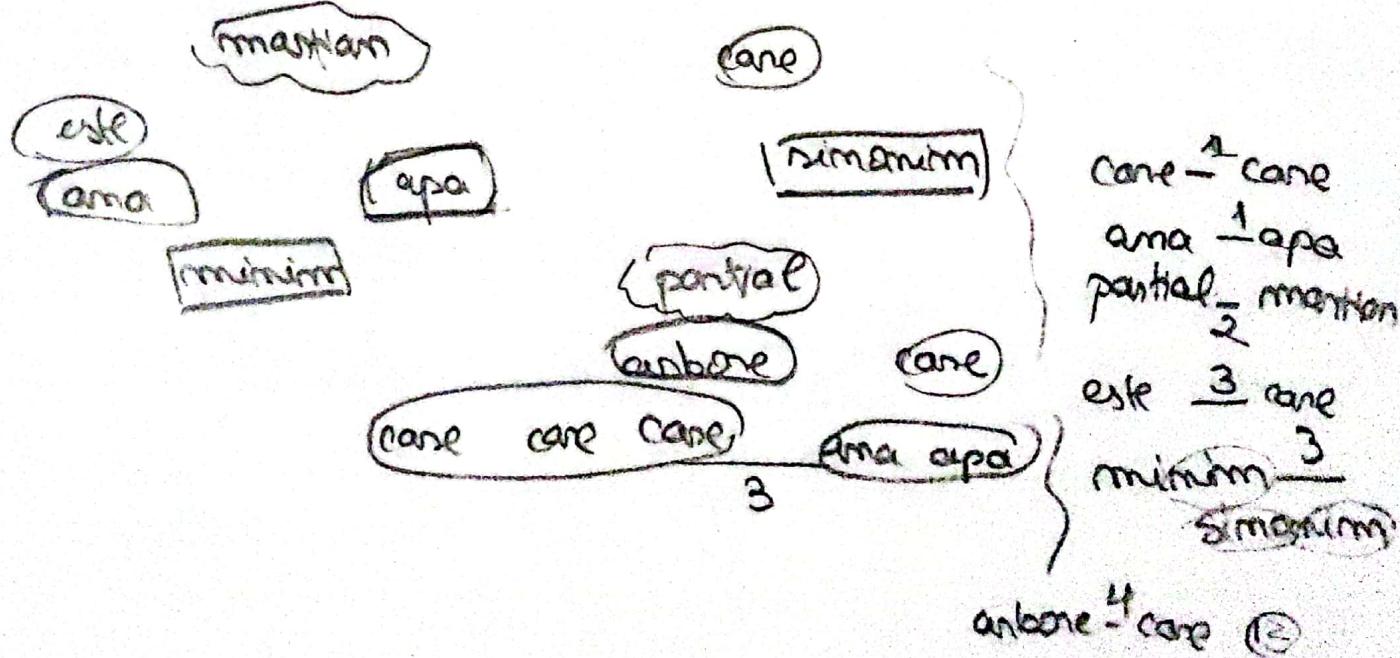
$$= \min_{i,j} h_d(c_i, c_j) \mid i \in \{1, \dots, k\}, j \in \{1, \dots, k\}$$

- initial  $\rightarrow$  fiecare clăsă

$\hookrightarrow$  la un pas determin cele mai diferențiale (inapropiate) 2 obiecte aparținând clasei diferențiale (cu dist min între ele)

și unim clasele lor

$\hookrightarrow$  pînă obținem k clase  $\Rightarrow$  m-k pași



- o afiz cele k clase obt
  - ↳ modelare graf ponderat  $\rightarrow$  alg binaricel
  - ↳ modelare graf neorientat  $\rightarrow$  alg binaricel
- o alg este echiv cu  $h^*$  - det aprem  $T$  al grafului complet  $G$
- o alg este echiv cu  $h^*$  - consideră mult.  $\{e_{m-k+1}, \dots, e_{n-1}\}$ 
  - form cu  $k-1$  muchii cu cele mai mari ponderi
  - pădurea  $T' = T - h_{\{e_{m-k+1}, \dots, e_{n-1}\}}$

→ Cum calc dist de vizitare?

- o la fiecare repetiție a unui caracter

o PD

- o  $x_1 \dots x_m$  și corespond.  
 $y_1 \dots y_m$  și ultima operatie

$x_m$  zintă: transf

$x_1 \dots x_{m-1} \leftarrow x_m$

$y_1 \dots y_m \leftarrow$  stergere

$x_m$

$x_m$  modif în  $y_m$

$(x_1 \dots x_{m-1} \Rightarrow y_1 \dots y_{m-1}) +$

$y_m$  a fost  
vizitat:

modif  $x_m \leftarrow y_m$

$(x_1 \dots x_{m-1} \Rightarrow y_1 \dots y_{m-1}) +$  inserare,  $y_m$

$x_1 \dots x_i \leftarrow y_1 \dots y_j$

stergere,  
modif

Ref de recurență:

$$c[i][j] = \begin{cases} c[i-1][j-1], & \text{dacă } x_i = y_j \\ 1 + \min(c[i-1][j], c[i-1][j-1], \\ c[i-1][j-1]), & \text{stergere } x_i \\ & \text{inserare } y_j \end{cases}$$

### Exemplu

$\Delta_1$ = Parne	$\Delta_2$ = Urtăt	$i$	$j$	$c[i][j]$
0	0	0	0	0
0	1	1	1	0
1	1	1	2	1
0	2	2	2	1
1	2	2	3	2
0	3	3	3	2
1	4	4	4	3

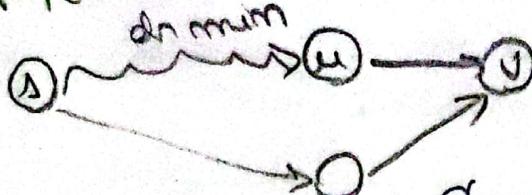
$$\text{URAT! } i=4 \quad \Delta_1[i]=e \quad j=4 \quad \Delta_2[j]=e \quad \rightarrow$$

$$c[i][j] = c[i-1][j-1] + 1 = 3$$

$m=3$

- 10 010 110 ...

## - DRUMURI DE LUNG MINIMĂ -



$$d(v) = \min_{u \in V} \{ d(u) + w(u,v) \mid u \in V \}$$

$\downarrow$   
 $u = \text{predecesor } v$

- Pe parcursul algoritmului  $d(v)$  - distanță dintre  $s - v$
- $\rightarrow$  state( $v$ ) total( $v$ )

pentru graf nepondent  $\Rightarrow$  BFS

→ Graf ponderat

Dijkstra (ponderi  $> 0$ )

$Q \subseteq V$  (putem încrepe) cu  
 $Q \subseteq D$

pentru fiecare  $v \in V$

$$d(v) = \infty, \text{total}(v) = 0$$

$$d(s) = 0$$

căt timp  $Q \neq \emptyset$

$v = \text{extrage}(Q)$  vârf cu et d min

pentru fiecărui  $u \in E$

$$\text{dacă } d(v) + w(v,u) < d(u)$$

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

$$\text{total}(u) = u$$

reporn(u, Q)

scrie d, totala

$$O(m \log m) / O(m^2)$$

Prim

dr min

$\rightarrow$  state( $v$ ) total( $v$ )



DAG (grafuri ponderat biciclic)

SortTop  $\leftarrow$  sortare topologică (G)

pentru fiecare  $v \in V$   
 $d(v) = \infty, \text{total}(v) = 0$

$$d(s) = 0$$

pentru fiecărui  $v \in \text{SortTop}$

pentru fiecărui  $u \in E$

$$\text{dacă } d(v) + w(v,u) < d(u)$$

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

$$\text{total}(u) = u$$

scrie d, totala

$$O(m \cdot m)$$

- Dreapta minimă din  $S \Rightarrow$  sub dreapta minimă  
 + anăpare cost minim  
 (minimum costului total)
  - Alg. Bellman-Ford  
 $d_{i+1}^k \leq d_i^k(p, u)$   
 costul minim al unei  $\Delta$ -x drepte cu cel mult  $k$   
 $d_i^k(p, u) = \min h d_{i-1}^k(p, v), \min h d_{i-1}^k(p, v) + w_{uv}$   
 la pasul  $k$ : costul minim al unei drepte de la sârba cel mult  $k$  pasuri

pentru fiecare  $\pi \in X$

$$df_{ij} = \infty \Rightarrow \text{total } f_{ij} = 0$$

$$d(\delta J) = 0$$

d)  $S = \emptyset$   
pentru  $k=1, m-1$  execută  
Piese si  $\forall i \in E$

then  $k=1, m-1$  exists  
for some  $m \in \mathbb{N}$

$$d\{u\} = d\{u\} + w(u, v)$$

$$f(a) \in U] = u$$

$O(nm)$

## - Conchitidine -

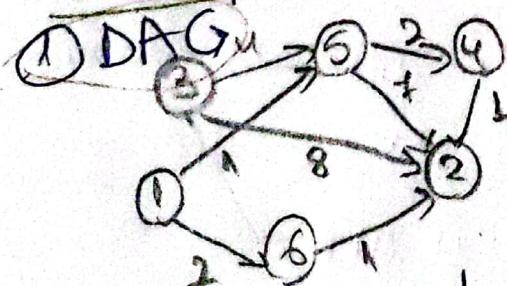
Constituente - la filcone pas al alg Dijustha IDA6

•  $d(\bar{u}) < \delta$  dim de la D la u în G de cost  
acum :  $d(\bar{u}) < \delta$  dim de la D la u în G de cost  
acesta reprezintă dim verităta

Sono topologiche:

1,36,5,4,2

$s = 3$  - of de start  
ordine de calcul sisteme



all tata

$$\begin{matrix} 1 & 2 \\ \infty/0 & \infty/0 \\ -1/0 & \infty/1 \end{matrix}$$

3 do 4 =10 5 =10 6 =10

$$ee = 1$$

010 8/3

$\mu = 3$

8/3 0/0 4/3 0/0 (15)

$$\begin{array}{l}
 u=6 : \{ \infty, 0, \quad 8/3 \quad 0/0 \quad \infty/0 \quad 4/3 \quad \infty(0) \\
 u=5 : \{ \infty/0 \quad 8/3 \quad 0/0 \quad 6/5 \quad 4/3 \quad \infty/0 \} \\
 u=4 : \{ \infty/0 \quad 7/4 \quad 0/0 \quad 0/5 \quad 4/3 \quad \infty/0 \} \\
 u=2 : \{ \infty/0 \quad 7/4 \quad 0/0 \quad 6/5 \quad 4/3 \quad \infty/0 \}
 \end{array}$$

## Solutie

## Aplicații

- I) Aplicări

Se cunosc pt un proiect cu  $n$  activități numerotate  $1, \dots, n$

  - durata fiec activitate
  - perechi  $(i, j)$   $\rightarrow$  activit  $i$  se poate înainta să înceapă j
  - activități imparale

Se cere timpul minim de finalizare a proiectului

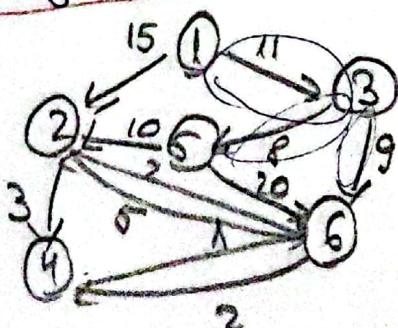
$w(i, j) \rightarrow$  durata activității i

$\rightarrow$  DRUM CRITIC  
(de cost maxim)

$\rightarrow$  Înloc  $w(e)$  cu  $-w(e)$

timpul min  $\sum$  finalizare = cost maxim drum

initializ dist



dltata

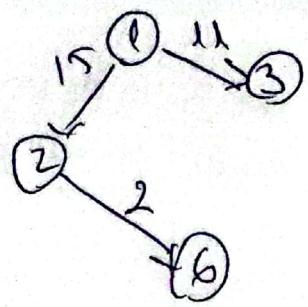
1	2	3	4	5	6
010	010	010	010	010	010
-	1511	(111)	010	10010	,

sel 3

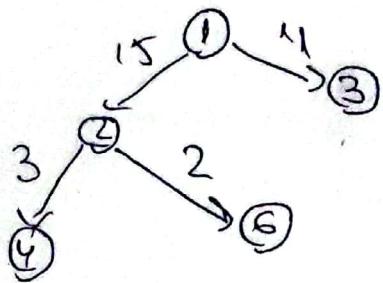
-  $ISI_{(III)} \approx 10 B_3 20/3$

sel 2

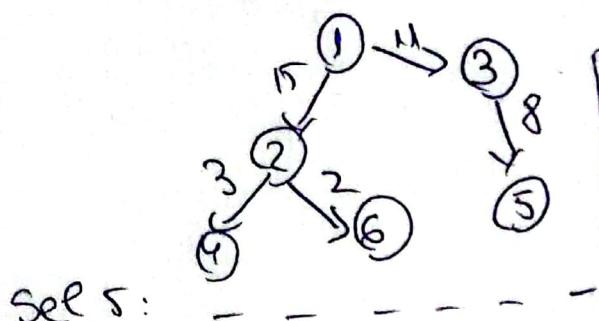
$$\begin{array}{r} 151 \xrightarrow{11} 3 \\ -2 \swarrow \\ - - - 1812 \end{array} \quad \boxed{16} \quad \begin{array}{l} 19/3 \\ 17/2 \end{array}$$



sel 6: - - - 18/2 19/3 -



sel 4: - - - - 19/3 -



sel 5: - - - - -

sel: {0|0, 15|1, 11|1, 18|2} 19|3  
17|2}

obs  
1) dc nf cur u are d[u] =  $\Rightarrow$  alg se poate opri  
2) ved tota mern arborele dintr  
radă de s

- înă o dată alg-

pentru fiecare  $u \in V$  execută

$$d[u] = \infty, \text{tata}[u] = \emptyset$$

$$d[S] = 0$$

căt timp  $Q \neq \emptyset$  execută

$u = \text{extaga cf cu eficacia } d \text{ min dim } Q$

pentru fiecare  $v \in E$  execută

dacă  $d[u] + w(u, v) < d[v]$  atunci

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

$$\text{tata}[v] = u$$

Scrie  $d_1, d_2, \dots$

obișn min de la  $s$  este un sf t dat folos  
totă

17

## O BELLMAN-FORD

La un pas relaxăm toate anele

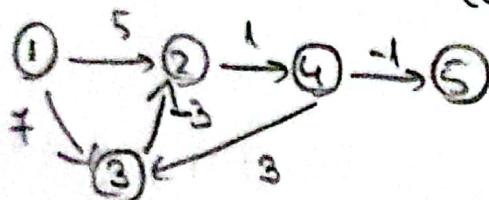
( $m-1$  etape : după  $k$  etape  $d_{ij} \leq \text{cost min drum}$   
acele multe  $i-j$  ane)

pentru fiec u v execută  
 $d_{uv} = \infty$ ;  $tad_{uv} = 0$

$$\alpha_{SSJ} = 0$$

pentru  $k=1, m-1$  execută

pentru fiecare u v E execută  
dacă  $d_{uj} + w(u, v) < d_{vj}$  atunci  
 $d_{vj} = d_{uj} + w(u, v)$   
 $tad_{vj} = u$



Relaxăm

12 → nu a fost actualizat

13 → putem ignoră

$$d: \begin{matrix} 1 & 2 & 3 & 4 & 5 \\ 0 & 5 & 7 & 6 & \infty \\ \curvearrowleft & 4 & 3 & & \\ & 4 & 5 & & \end{matrix} \quad \begin{matrix} 1 & 2 \\ 1 & 3 \end{matrix}$$

$$d: \begin{matrix} 1 & 2 & 3 & 4 & 5 \\ 0 & 5 & 7 & 6 & 5 \\ \curvearrowleft & 3 & 2 & & \end{matrix}$$

$$d: \begin{matrix} 1 & 2 & 3 & 4 & 5 \\ 0 & 4 & 7 & 6 & 5 \end{matrix}$$

Et 2: Relaxăm

$$\begin{matrix} 2 & 4 \\ 4 & 3 \end{matrix}$$

$$d: \begin{matrix} 1 & 2 & 3 & 4 & 5 \\ 0 & 4 & 7 & 5 & 5 \end{matrix}$$

$$45$$

$$d: \begin{matrix} 1 & 2 & 3 & 4 & 5 \\ 0 & 4 & 7 & 5 & 4 \end{matrix}$$

$$32$$

$$d: \begin{matrix} 1 & 2 & 3 & 4 & 5 \\ 0 & 4 & 7 & 5 & 4 \end{matrix}$$

nu se mai actualizează → STOP

La un pas putem relaxe anul din urm ale anelor  
etichetă s-a modificat anterior

(folosirea de rezultat / coadă cu următorii noduri la et

⑧

coadă cu următorii noduri la et

FLOYD-WARSHALL  $O(M^3)$   
Opb dn minimum între toate per de nf  
Graf orientat:  $G = (V, E, \omega)$   
ponderat

$\omega = (w_{ij})$  matr costuri graf  $G$

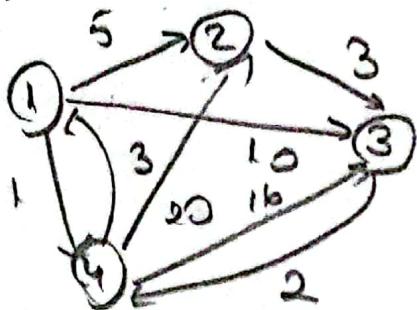
$$\Leftrightarrow w_{ij} = \begin{cases} 0 & i=j \\ w_{ij}, & \text{dacă } ij \in E \\ \infty & \text{de } ij \notin E \end{cases}$$

matr  
distanțelor  
 $D = (d_{ij})$

$$d_{ij} = \underline{\omega(i, j)}$$

cost minim al drumului de la  $i$  la  $j$   
care are  $k$  intermedii,  $1, 2, \dots, k-1$

Ex:



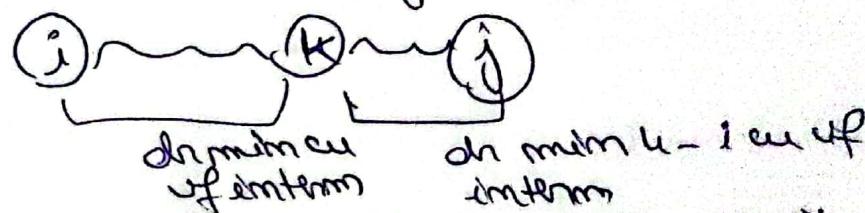
$$d^0_{4,3} = 16 \text{ arc dn}$$

$$d^1_{4,3} = 13 - \text{drum}\{4, 1, 3\}$$

$$d^2_{4,3} = 11 - \text{drum}\{4, 1, 2, 3\}$$

$$d^3_{4,3} = 8 - \text{drum}\{1, 2, 3\}$$

$\hookrightarrow$  calc  $D$ :  $w(P) = d^{k-1}_{ik} + d^{k-1}_{kj} = d^k_{ij}$



$\hookrightarrow P = \text{matr drumuri}$   $h_{1,2,\dots,k-1}$   $h_{1,2,\dots,k-1}$

$$d[i][j] = \omega(i, j) - \text{cost}(i, j)$$

$$P[i][j] = \begin{cases} i, & i, j \in E \\ 0, & \text{altfel} \end{cases}$$

$\hookrightarrow$  Implement:

Floyd ( $G, \omega$ )

for ( $i = 1 \rightarrow m$ )

    for ( $j = 1 \rightarrow m$ )

$$d[i][j] = \omega[i][j]$$

$$\text{else if } (d[i][j] = \infty) \quad P[i][j] = 0,$$

$$\text{else } P[i][j] = i$$

(9)

```

for(u=1→m)
  for(i=1→m)
    for(j=1→m)
      if( d[i][j] > d[i][u] + d[u][j] )
        d[i][j] = d[i][u] + d[u][j]
        p[i][j] = p[u][j]

```

effis dumm:

```

void dumm(int i, int j)
  if(i==j)

```

prog

(ex)

$$d = \begin{matrix} 0 & 5 & 10 & 1 \\ - & 0 & 3 & \infty \\ - & - & 0 & 2 \\ - & \infty & 0 & 2 \\ 3 & 20 & 16 & 0 \end{matrix}$$

$$P = \begin{matrix} 0 & 1 & 1 & 1 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 3 & 3 \\ 4 & 4 & 4 & 0 \end{matrix}$$

u=1

$$\begin{matrix} 0 & 5 & 10 & 1 \\ - & 0 & 3 & 5 \\ - & - & 0 & 2 \\ - & 3 & 8 & 13 & 0 \end{matrix}$$

$$P = \begin{matrix} 0 & 1 & 1 & 1 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 3 \\ 4 & 1 & 1 & 0 \end{matrix}$$

u=2

$$\begin{matrix} 0 & 5 & 8 & 1 \\ - & 0 & 3 & \infty \\ - & - & 0 & 2 \\ - & 3 & 8 & 11 & 0 \end{matrix}$$

$$P = \begin{matrix} 0 & 1 & 2 & 1 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 3 & 0 \\ 4 & 1 & 2 & 0 \end{matrix}$$

u=3

$$\begin{matrix} 0 & 5 & 8 & 1 \\ 0 & 0 & 3 & 5 \\ 0 & 0 & 0 & 2 \\ 3 & 8 & 11 & 0 \end{matrix}$$

$$P = \begin{matrix} 0 & 1 & 2 & 1 \\ 0 & 0 & 2 & 3 \\ 0 & 0 & 0 & 3 \\ 4 & 1 & 2 & 0 \end{matrix}$$

u=4

$$\begin{matrix} 0 & 5 & 8 & 1 \\ 2 & 0 & 3 & 5 \\ 5 & 10 & 0 & 2 \\ 3 & 8 & 11 & 0 \end{matrix}$$

$$P = \begin{matrix} 0 & 1 & 2 & 1 \\ 8 & 9 & 0 & 3 \\ 4 & 1 & 2 & 0 \end{matrix}$$

## - MUCILII CRITICE -

↳ muchie critică = muchie conformă ciclului

↳ găsesc ciclul → parcurg DFS

muchie avansare, anb DFS  
 ↳ muchie întoarcere: închid ciclul, nu pot fi critice  
 ↳ e critică de mereu în ciclul închis de o muchie de întoarcere  
 ↳  $\min_{i \in V} \text{nivel}[i]$  este critică  
 ↳ muchie de întoarcere ce extinde un j (deci și cealaltă extindere împreună arătă că ciclul)

memonarmă pt fiecare i:

$\text{miv-min}[i] = \text{miv min al unui vf care este extremul a unei muchii de întoarcere din i sau desigur}$

( $\rightarrow$  miv minimum la care se închide același cicl care conține vfi (prin urmare muchie întoarcere)

$\text{mivel}[i] = \text{miv lui } i \text{ în anb DFS}$

$\text{miv-min}[i] = \min_{j \in V} \text{mivel}[j], A, B$

$A = \min_{j \in V} \text{mivel}[j] \text{ sau - muchie întoarcere}$

$B = \min_{j \in V} \text{mivel}[j] \text{ și este al lui } i$

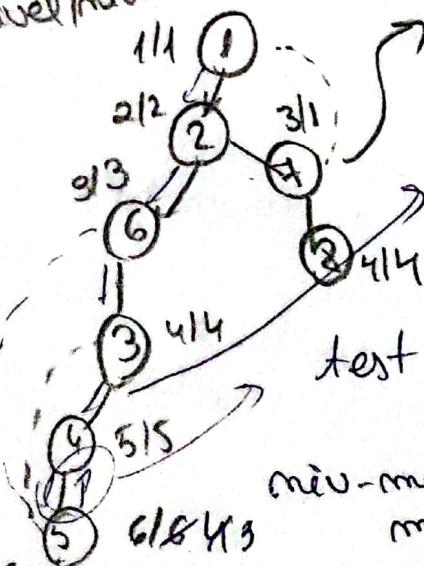
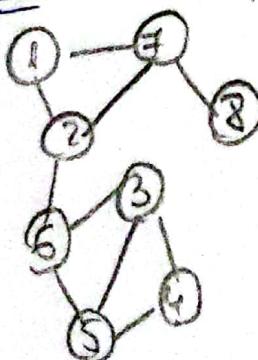
$\text{miv-min}[j] > \text{mivel}[j]$

$\Rightarrow B = \min_{j \in V} \text{miv-min}[j] \text{ și } fără } y$

$\text{mivel} / \text{miv-min}$

$\text{miv-min}[8] = 4 > \text{mivel}[7] = 3$   
 inclusiv  $i \neq 8$

EX



test m eritică

$\text{miv-min}[4] = 3 \leq$

$\text{mivel}[3] = 4 \text{ (NU)}$   
 ⇒ nu e critică

$\text{miv-min}[8] = 4 > \text{mivel}[7] = 3$

$\text{miv-min}[5] = 3 \leq$

$\text{mivel}[4] = 5 \text{ (NU)}$

$\text{miv-min}[7] =$

$\min_{j \in V} \text{miv-min}[j], \text{mivel}[j]$

$\text{miv-min}[4] = \min_{j \in V} \text{miv-min}[j], \text{miv-min}[5] = 4$   
 $= 3$

②1

$\min - \min[3] = 3$

la fie par  $\rightarrow$  test m orifici + actualiz  $\min - \min$

2-6:  $\min - \min[6] = 3 \rightarrow \text{nivel}[2] = 2$  (A) UPRA

## Implementare

void dfs(int i)

$\downarrow \text{nivel}[i] = 1;$

$\min - \min[i] = \text{nivel}[i];$

for (j vec al lui i)

$\downarrow \text{if } (\text{nivel}[j] = 0)$  // j este neschis

$\downarrow \text{nivel}[j] = \text{nivel}[i] + 1;$

$\text{df}(j);$

// actualiz  $\min - \min$

$\min - \min[i] = \min[\min - \min[i], \text{nivel}[j]], \text{nivel}[j]$

// test  $i \rightarrow$  m orifici

$\downarrow \text{if } (\min - \min[j] > \text{nivel}[i]) \rightarrow \text{scade } ij$

$\downarrow \text{else}$

$\downarrow \text{if } (\text{nivel}[j] \geq \text{nivel}[i-1])$

$\downarrow \min - \min[i] = \min[\min - \min[i], \text{nivel}[j]], \text{nivel}[j]$

$\downarrow \text{j este neschis}$

POARTA

Puncte critice  
un vf v este pt critic  $\Leftrightarrow$  d 2 vf x-yf Va.P  
v este unicuri lang x-y

## ARB DFS

$\downarrow \text{rad } \triangleright = \text{punct critic}$

$\Rightarrow$  one min 2 fir ambar DFS

$\downarrow$  alt vf i este critic

one cel putin un fir j

ce  $\min - \min[j] \geq \text{nivel}[i]$