UNIUNEA EUROPEANĂ

GUVERNUL ROMÂNIEI

Instrumente Structurale
2007-2013

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

Proiectarea Algoritmilor

9. Parcurgere în adâncime (DFS)

Bibliografie

- Giumale – Introducere în Analiza Algoritmilor cap 5 si 5.1
- Cormen – Introducere în Algoritmi cap 22, 22.1, 22.2, 22.3 si 22.4
- http://ist.marshall.edu/ist362/pics/OSPF.gif
- http://ashitani.jp/gv/
- http://en.wikipedia.org/wiki/PageRank

Algoritmi de parcurgere – Notații (1)

- G = (V,E);
- V – mulțimea de noduri;
- E – mulțimea de muchii / arce;
- (u,v) – arcul / muchia u,v;
- u..v – drum de la u la v; dacă există mai multe variante notăm u..x..v, u..y..v;
- R(u) - reachable(u) = mulțimea nodurilor ce pot fi atinse pe căi ce pleacă din u;

Algoritmi de parcurgere – Notații (2)

- succs(u) – mulțimea succesorilor lui u (graf orientat) sau mulțimea nodurilor adiacente lui u (graf neorientat);
- c(u) – culoarea nodului – specifică starea nodului la un anumit moment al parcurgerii:
  - Alb – nedescoperit;
  - Gri – descoperit, în curs de prelucrare;
  - Negru – descoperit si terminat (cu semnificații diferite pentru BFS si DFS).
- p(u) (π(u)) – "părintele lui u" – identificator al nodului din care s-a ajuns în nodul u prima oară.

Parcurgere în adâncime (DFS)

- Nu mai avem nod de start, nodurile fiind parcurse în ordine.

- d(u) = momentul descoperirii nodului (se trece prima oară prin u și e totodată și momentul începerii explorării zonei din graf ce poate fi atinsă din u).
- f(u) = timpul de finalizare al nodului (momentul în care prelucrarea nodului u a luat sfârșit)
  - Tot subarborele de adâncime dominat de u a fost explorat.
  - Alternativ: tot subgraful accesibil din u a fost descoperit și finalizat deja.

DFS – Structura de date

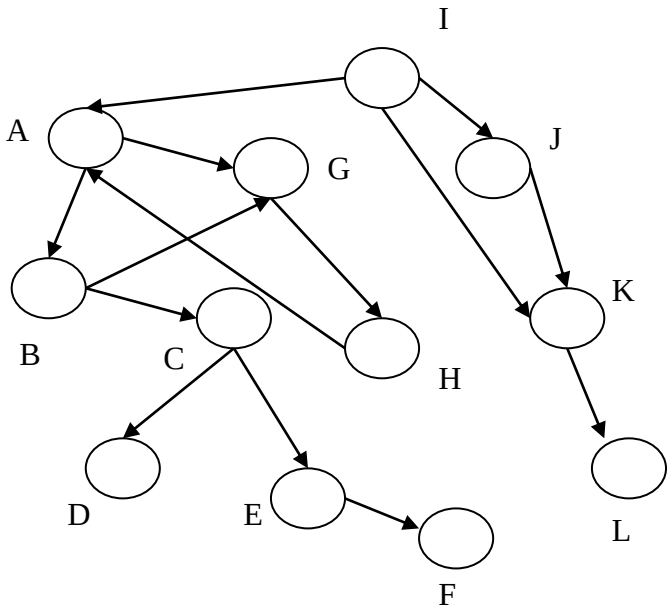
- Folosește o stiva (LIFO) pentru a reține nodurile ce trebuie prelucrate
  - În implementările uzuale, stiva este rareori folosită explicit;
  - Se apelează la recursivitate pentru a simula stiva.
- Folosește o variabilă globală *timp* pe baza căreia se calculează timpii de descoperire și de finalizare ai fiecărui nod.
- Pentru fiecare nod se reține:
  - Părintele – $\pi(u)$ ($p(u)$);
  - Timpul de descoperire – $d(u)$;
  - Timpul de finalizare – $f(u)$;
  - Culoarea nodului.

DFS – Algoritm

- DFS(G)
  - V = noduri(G)
  - **Pentru fiecare** nod u (u ∈ V)
    - c(u) = alb; p(u) = null; // inițializare structură date
  - timp = 0; // reține distanța de la rădăcina arborelui DFS pană la nodul curent
  - **Pentru fiecare** nod u (u ∈ V)
    - **Dacă** c(u) este alb
      - **Atunci** explorare(u); // explorez nodul

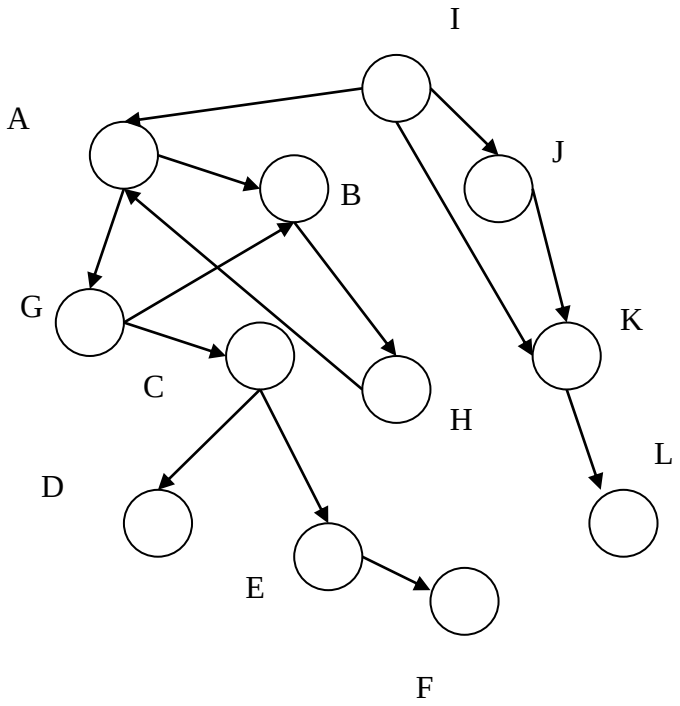
- explorare(u)
  - d(u) = ++ timp; // timpul de descoperire al nodului u
  - c(u) = gri; // nod in curs de explorare
  - **Pentru fiecare** nod v ∈ succs(u) // încerc sa prelucrez vecinii
    - **Dacă** c(v) este alb
      - **Atunci** {p(v) = u; explorare(v);} // dacă nu au fost prelucrați deja
  - c(u) = negru; // am terminat de explorat nodul u
  - f(u) = ++ timp; // timpul de finalizare al nodului u

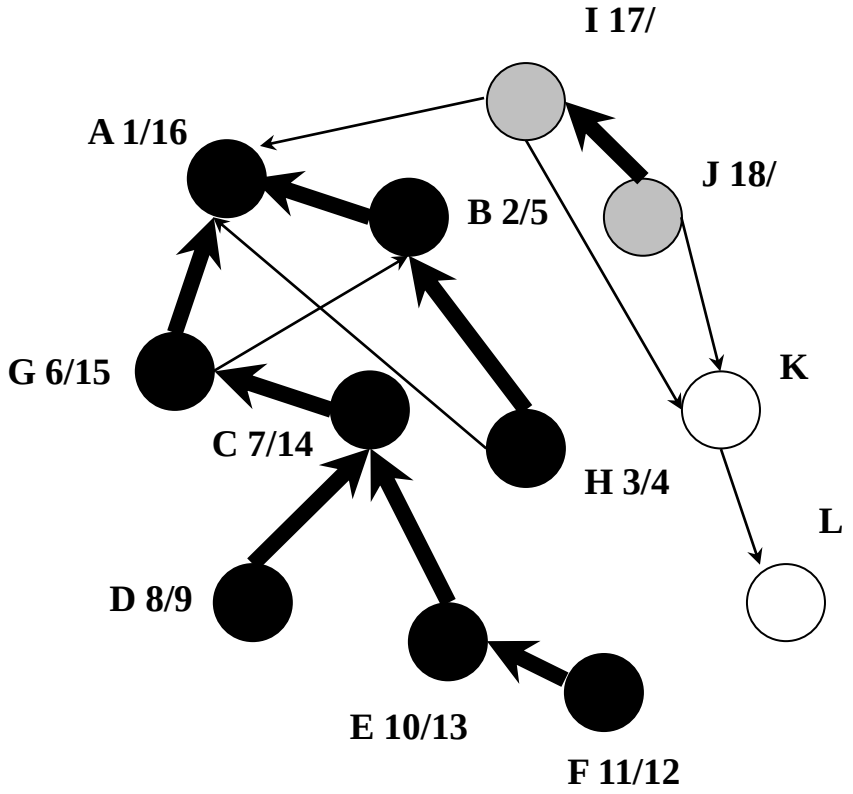
DFS – Exemplu



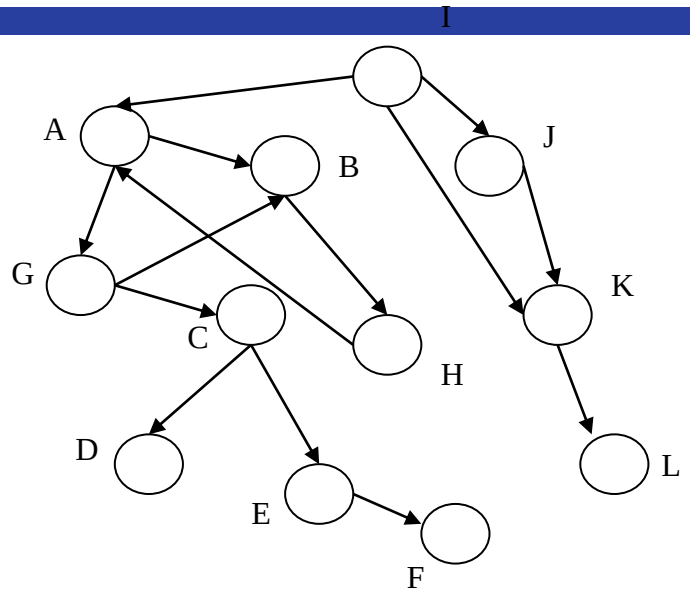
- DFS(G)
  - V = noduri(G)
  - **Pentru fiecare** nod u (u ∈ V)
    - c(u) = alb; p(u) = null; // inițializare structură date
  - timp = 0; // reține distanța de la rădăcina arborelui DFS pană la nodul curent
  - **Pentru fiecare** nod u (u ∈ V)
    - **Dacă** c(u) este alb
      - **Atunci** explorare(u); // explorez nodul
- explorare(u)
  - d(u) = ++ timp; // timpul de descoperire al nodului u
  - c(u) = gri; // nod in curs de explorare
  - **Pentru fiecare** nod v ∈ succs(u) // încerc sa prelucrez vecinii
    - **Dacă** c(v) este alb
      - **Atunci** {p(v) = u; explorare(v);} // dacă nu au fost prelucrați deja
  - c(u) = negru; // am terminat de explorat nodul u
  - f(u) = ++ timp; // timpul de finalizare al nodului u

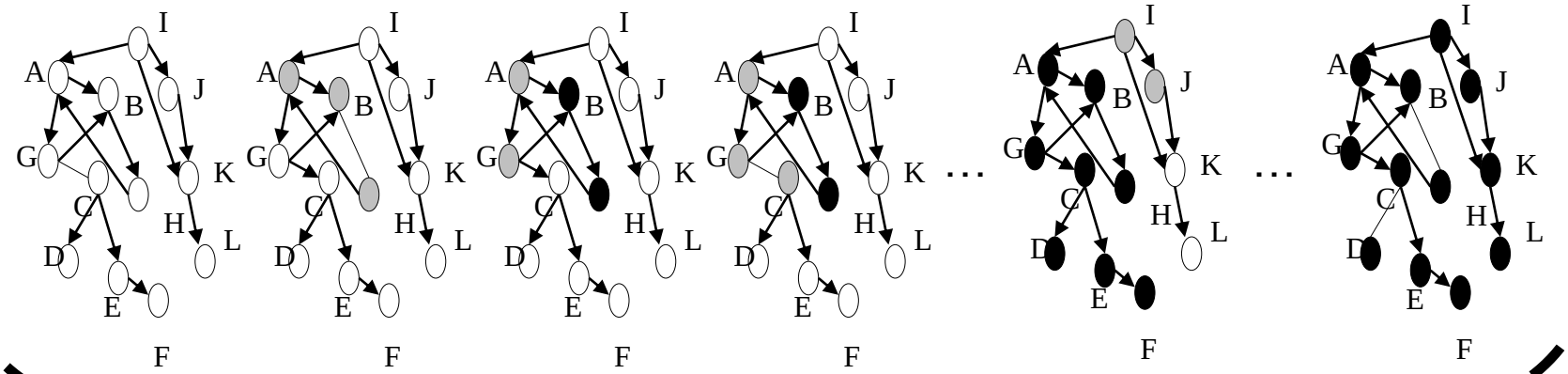
Calculul timpilor



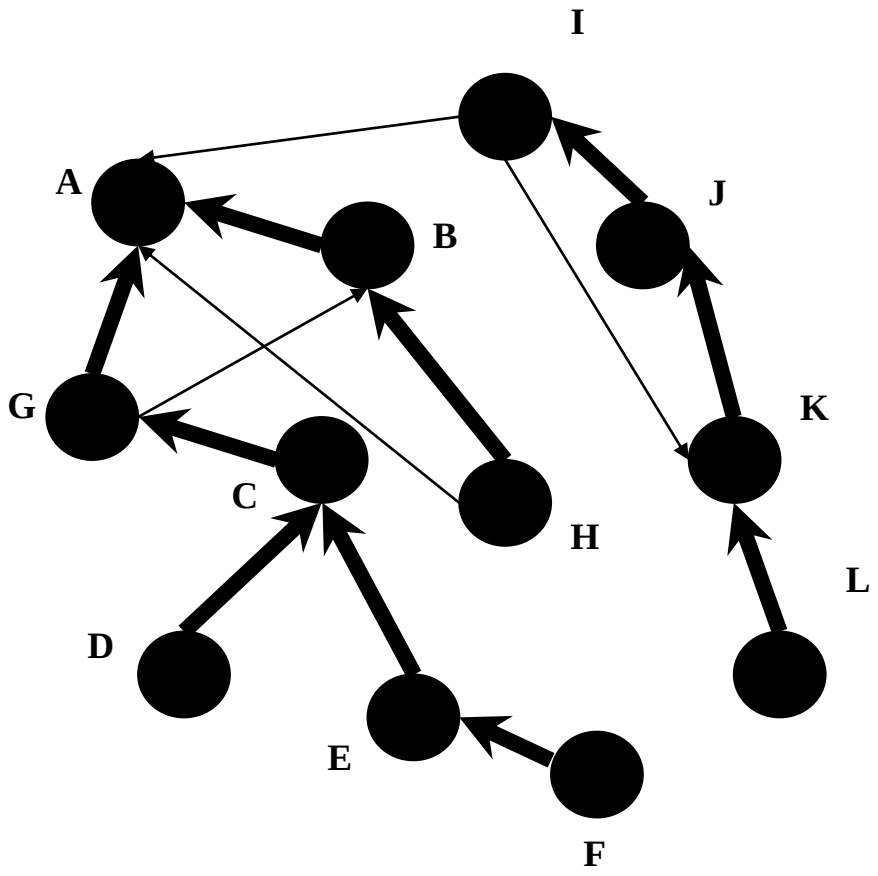


DFS – Evoluția explorării

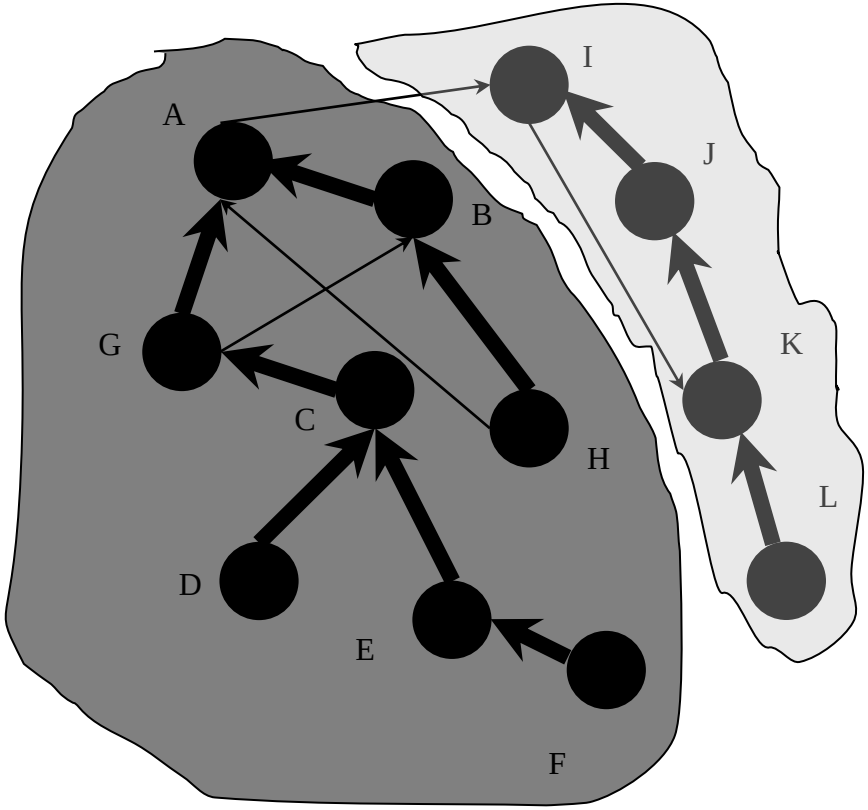




Arborele de parcurgere în adâncime



DFS – Zone de explorare



DFS – Proprietăți (I)

- I(u) = intervalul de prelucrare al nodului (d(u),f(u)).
- Lema 5.5. G = (V,E); u ∈ V; pentru fiecare v descoperit de DFS pornind din u este construită o cale v, p(v), p(p(v)),…, u.
  - Fie calea u = $v_0v_1…v_n$ = v. Dem prin inducție ca $\pi(v_i) = v_{i-1}$!
- Teorema 5.2. G = (V,E); DFS(G) sparge graful G într-o pădure de arbori Arb(G) = { Arb(u); p(u) = null } unde Arb(u) = (V(u),E(u));
  - V(u) = { v | d(u) < d(v) < f(u) } + {u};
  - E(u) = { (v, z) | v, z ∈ V(u) && p(z) = v }.

DFS – Proprietăți (II)

- Teorema 5.3. Dacă DFS(G) generează 1 singur arbore => G este conex. (Reciproca este adevărată?)
- Teorema 5.4. Teorema parantezelor:
  - $\forall$ u, v atunci I(u) $\cap$ I(v) = $\varnothing$ sau I(u) $\subset$ I(v) sau I(v) $\subset$ I(u).
  - Dem prin considerarea tuturor combinațiilor posibile!
- Teorema 5.5. $\forall$ u, v $\in$ V, atunci v $\in$ V(u) $\Leftrightarrow$ I(v) $\subset$ I(u).
- Teorema 5.6. Teorema drumurilor albe:
  - G = (V,E); Arb(u); v este descendent al lui u in Arb(u) $\Leftrightarrow$ la momentul d(u) exista o cale numai cu noduri albe u..v.
  - Dem prin inducție!

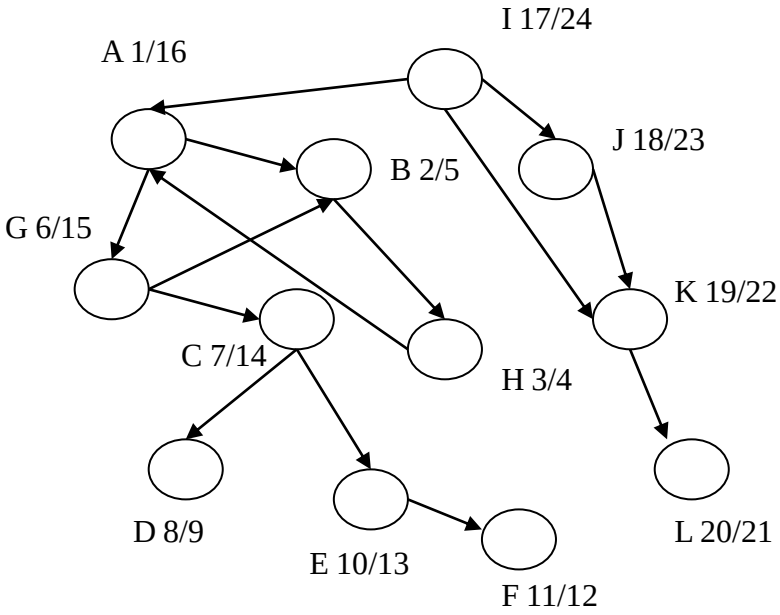
Clasificări ale arcelor grafului (I)

- Arc direct (de arbore)
  - Ce fel de noduri?
- Arc invers (de ciclu)
  - Ce fel de noduri?
- Arc înainte
  - Ce fel de noduri?
- Arc transversal
  - Ce fel de noduri?

Clasificări ale arcelor grafului (II)

- Arc direct (de arbore)
  - între nod gri si nod alb;
- Arc invers (de ciclu)
  - între nod gri si nod gri;
- Arc înainte
  - nod gri si nod negru si d(u) < d(v);
- Arc transversal
  - nod gri si nod negru si d(u) > d(v).

Clasificări ale arcelor grafului (III)



Arc direct (de arbore)
între nod gri si nod alb;
Arc invers (de ciclu)
între nod gri si nod gri;
Arc înainte
nod gri si nod negru si d(u) < d(v);
Arc transversal
nod gri si nod negru si d(u) > d(v).

Arc direct (de arbore):
AB, BH, AG, GC, CD, CE, EF, IJ, JK, KL
Arc invers (de ciclu):
HA
Arc înainte:
IK
Arc transversal:
GB, IA

DFS – Proprietăți (III)

- Teorema 5.7. Intr-un graf neorientat, DFS poate descoperi doar arce directe si inverse.
  - Dem prin considerarea cazurilor posibile!

- Teorema 5.8. G = graf orientat; G ciclic ⇔ in timpul execuției DFS găsim arce inverse.
  - Dem prin exploatarea proprietăților de ciclu si de arc invers!

DFS – Complexitate și Optimalitate

Complexitate:

O(n+m)

n = număr noduri

m = număr muchii

Optimalitate: NU

Parcurge tot graful? DA