Structuri de date – Curs 11

Conf. univ. dr. Cristian CIUREA
Departamentul de Informatica si Cibernetica Economica
Academia de Studii Economice din Bucuresti
cristian.ciurea@ie.ase.ro

Agenda

- Grafuri – definire și utilizare
- Complexitatea McCabbe
- Clasificarea grafurilor
- Metode de reprezentare
- Metode de traversare

Grafuri

- Graful, asemenea arborelui, este o structură de date în care relația dintre nodul părinte și nodul fiu este una ierarhică, dar care este mai putin restrictivă, în sensul că un nod are mai mulți succesori, dar și mai mulți predecesori.

Grafuri

- Graful este definit ca o colecție de date reunite în două mulțimi:
  - multimea $N = \{ N_1, N_2, \ldots, N_n \mid n$ – numarul de noduri ale grafului $\}$, ce conține toate nodurile grafului;
  - multimea $A = \{ (N_i, N_j) = A_{ij} \mid N_i, N_j \subset N$ si $i,j = 1,n$ cu $i \neq j\}$ care conține arcele dintre două noduri vecine.

Grafuri

- Graful este o pereche ordonată de mulțimi, notată **G=(N, A)**, unde:
  - **N** este o mulțime finită și nevidă de elemente numite **noduri** sau **vârfuri**;
  - **A** este o mulțime de perechi (ordonate sau neordonate) de elemente din N, numite **muchii** (dacă sunt perechi neordonate) sau **arce** (dacă sunt perechi ordonate).

Grafuri

- Utilizare structură de tip graf:
  - informaţii care au multiple legături între ele;
  - parcurgerea completă a elementelor structurii;
  - localizarea unui element din structură;
  - oferirea de soluţii optime de minimizare a costurilor.

Grafuri

- Graful este larg utilizat în domeniile:
  - ciberneticii;
  - matematicii;
  - cercetărilor operaţionale, în vederea optimizării diferitelor activităţi economice;
  - chimiei, pentru descrierea structurii cristalelor;
  - reţelelor de transport de toate tipurile, pentru optimizarea traseelor;
  - circuitelor electrice, pentru simularea funcţionării corecte;
  - inteligenţei artificiale;
  - analizei aplicaţiilor software.

Grafuri

- Complexitatea în sens McCabbe presupune asocierea fiecărui program a unui graf, în care nodurile corespund instrucţiunilor, iar arcele marchează sensul trecerii execuţiei de la o instrucţiune la alta;
- Complexitatea în sens McCabbe:

  $CC = n_{arc} - n_{nod} + 2$

  unde:

  n_{arc} – numărul de arce;

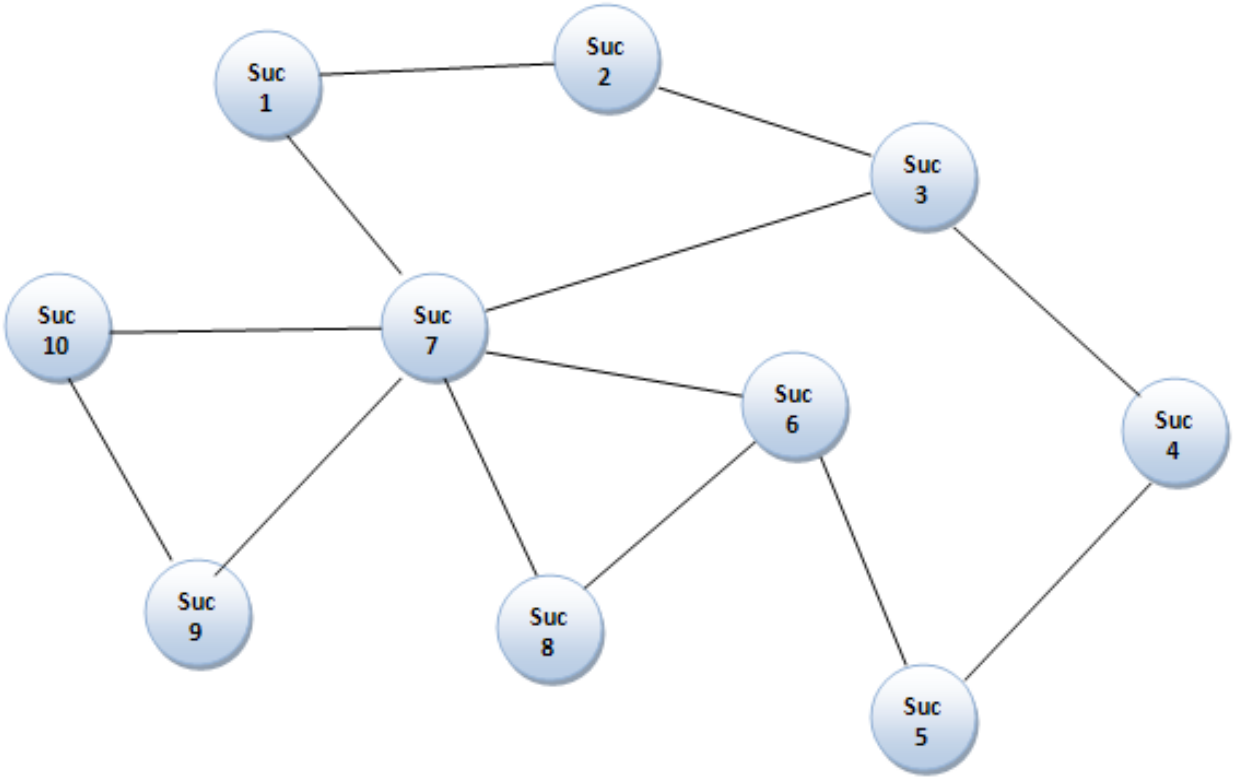
  n_{nod} – numărul de noduri.

Grafuri

- Sistemul colaborativ *bancă* are asociat graful în care nodurile reprezintă sucursalele băncii, iar arcele sunt legăturile dintre sucursale, reprezentate prin clienţii care au conturi deschise la mai multe agenţii.

Grafuri

- Graful asociat sistemului colaborativ *bancă*:
  - nodurile reprezentate de oraşele din ţară în care banca deţine sucursale;
  - arcele leagă sucursalele ce deţin clienţi comuni.



Grafuri

Structura de tip graf:

- relaţie ierarhică între nodul părinte şi nodul fiu;
- relație mai puţin restrictivă: un nod are mai mulţi succesori, dar şi mai mulţi predecesori;
- colecţie de date formată din două mulţimi:
  - mulţimea nodurilor grafului;
  - mulţimea arcelor dintre două noduri vecine.

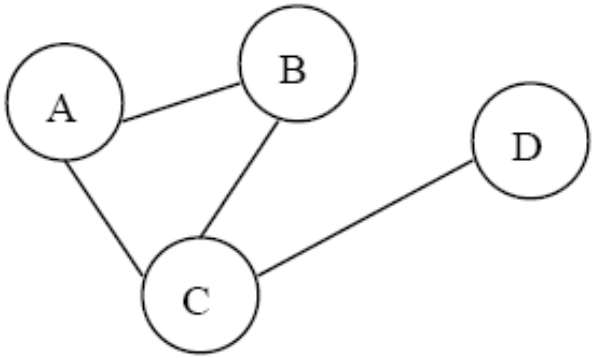
Grafuri

Criterii de clasificare a grafurilor:

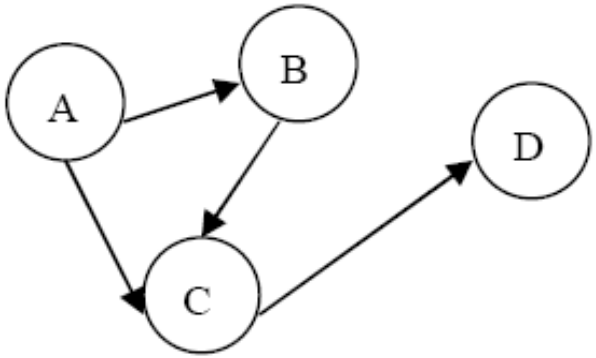
- **direcţia arcelor**:
  - **grafuri neorientate** (arce nedirecţionate);
  - **grafuri orientate** (există sens între două noduri);
- **greutatea arcelor**:
  - **grafuri cu greutate** (arce cu valoare numerică);
  - **grafuri fără greutate** (arcele nu au asociate valori numerice);
- **existenţa arcelor**:
  - **grafuri conectate** (nu există nici un nod izolat);
  - **grafuri neconectate** (există cel puţin un nod izolat).

Grafuri

- **direcția arcelor**:
  - în cazul în care arcele dintre nodurile grafului sunt nedirecționate, atunci graful este unul **neorientat**; când există sens între doua noduri Ni, Nj și arcul este direcționat, atunci graful este unul **orientat**;



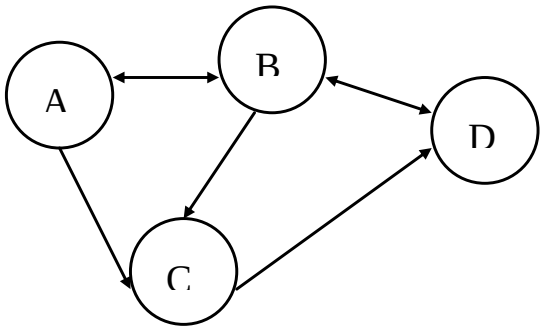
Graf neorientat



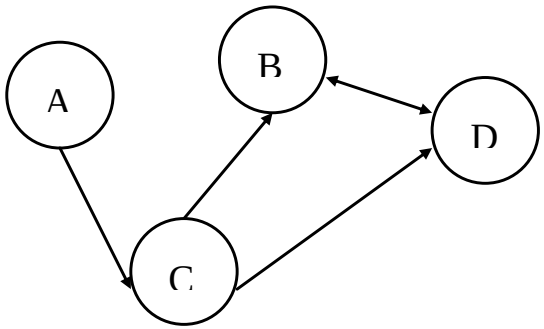
Graf orientat

Grafuri

- **direcția arcelor**:
  - graful orientat este **puternic conectat** dacă între oricare două noduri N_i şi N_j cu i, j = 1..n există drum (un drum este format din unul sau mai multe arce) orientat de la i la j, $N_i \rightarrow N_j$;
  - graful orientat este **slab conectat** dacă între oricare două noduri N_i şi N_j cu i, j = 1..n există drum orientat de la i la j, $N_i \rightarrow N_j$ **sau** de la j la i, $N_i \leftarrow N_j$ (doar unul dintre ele).



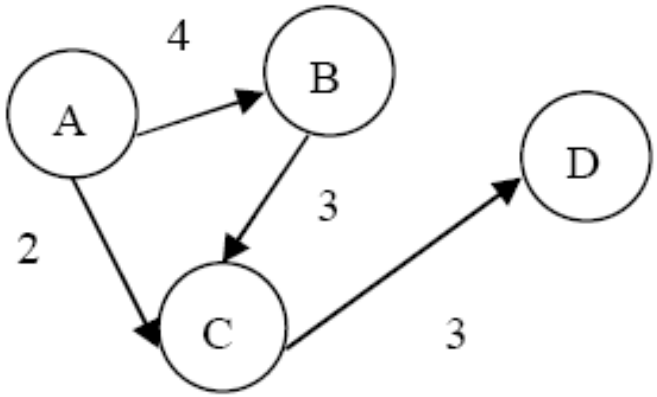
Graf orientat puternic conectat



Graf orientat slab conectat

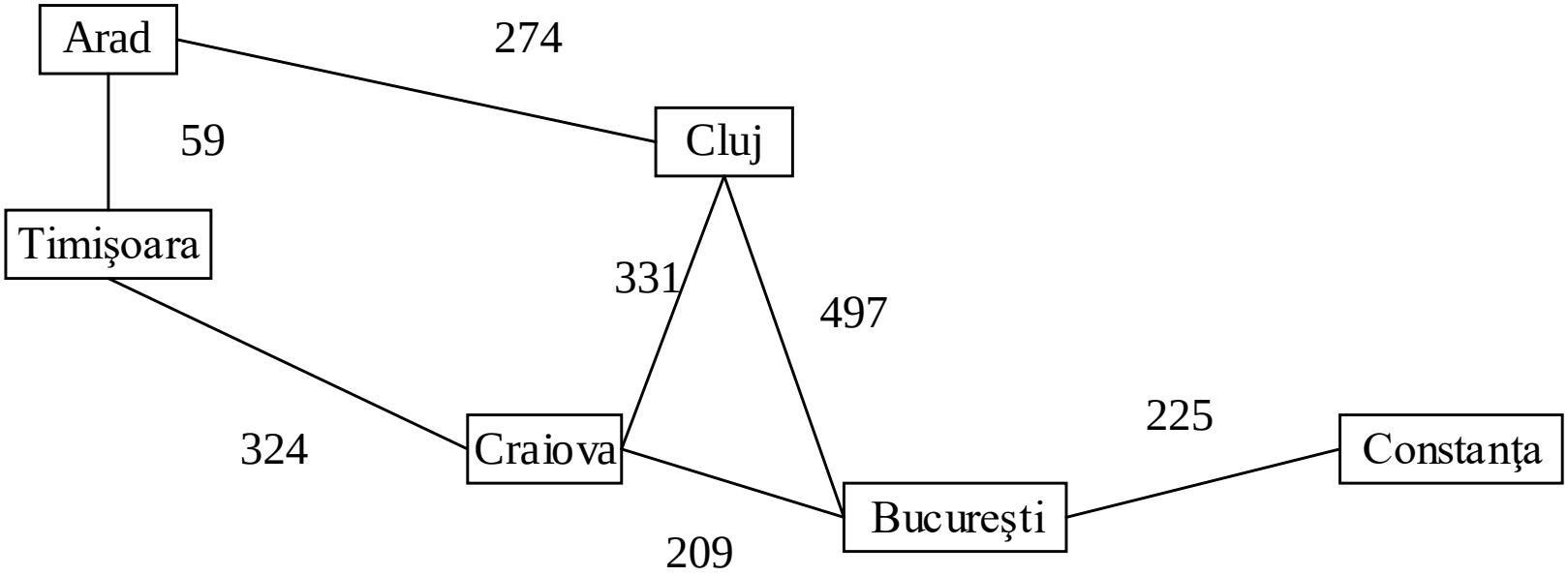
Grafuri

- **greutatea arcelor**:
  - daca oricare arc dintre două noduri ale grafului are asociată o valoare numerică (care reprezintă de cele mai multe ori distanța, durata de timp sau costul), atunci graful este **cu greutate**; în cazul în care arcele nu au asociate valori numerice, graful este unul **fără greutate.**



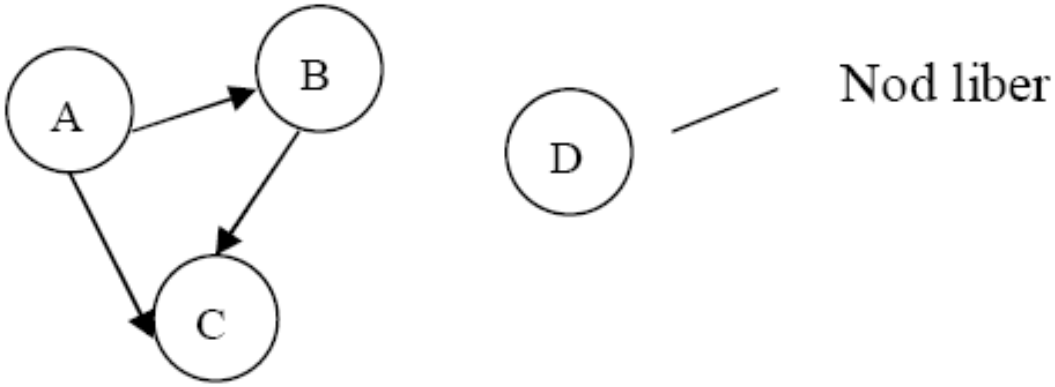
Grafuri

- N = {Arad, Bucureşti, Cluj, Craiova, Constanţa, Timişoara}
- A = {Arad – Timişoara = 59 km, Timişoara – Craiova = 324 km, Craiova – Bucureşti = 209 km, Bucureşti – Constanţa = 225 km, Arad – Cluj = 274 km, Cluj – Craiova = 331 km, Cluj – Bucureşti = 497 km}



Grafuri

- **existența arcelor**:
  - daca într-un graf nu există nici un nod izolat, altfel spus pentru oricare nod Ni cu i=1..n exista cel putin un nod Nj cu $1 \leq j \leq n$ si $i \neq j$ pentru care există arcul Aij asociat, atunci graful este **conectat**; un graf este **neconectat** dacă există cel puțin un nod izolat.



Grafuri

Metode de reprezentare a grafului:

- matrice de adiacenţă;
- liste de adiacenţă;
- vector de pointeri la liste simple sau dublu înlănţuite de noduri adiacente;
- listă simplu sau dublu înlănţuită de pointeri la liste simple sau dublu înlănţuite de noduri adiacente;
- vector de pointeri la liste simple sau dublu înlănţuite de arce.

Grafuri

- Reprezentarea prin matrice de adiacenţă – eficientă:
  - se cunoaşte numărul nodurilor;
  - se cunoaşte numărul mediu al arcelor – grad de umplere al matricei;
  - matrice pătratică;
  - reprezentare arce: valoarea 1 (graf fără greutate), greutate arc (graf cu greutate).

Grafuri

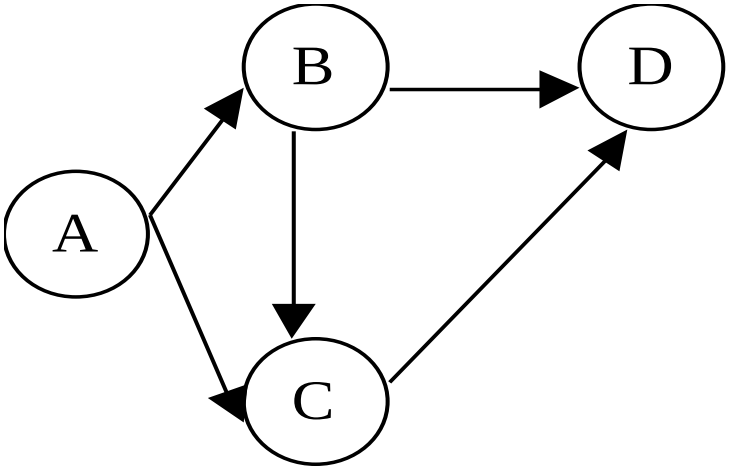
- Inițial matricea de adiacență are toate elementele egale cu valoarea 0.
- Pentru a reprezenta arcul *Aij* dintre nodurile *Ni* si *Nj* (orientat de la *Ni* la *Nj*) la intersecția liniei *i* cu coloana *j* se trece:
  - valoarea 1, în cazul grafului fără greutate, sau
  - greutatea arcului, pentru graful cu greutate.
- În cazul unui graf neorientat, matricea de adiacenţă asociată este simetrică.

Grafuri

- Pentru graful fără greutate:
  - M[i][j] = 1, dacă există arc între nodul N_i şi N_j
  - M[i][j] = 0, dacă nu există arc între nodul N_i şi N_j
- Pentru graful cu greutate:
  - M[i][j] = a_{ij}, dacă există arc între nodul N_i şi N_j, iar a_{ij} reprezintă greutatea arcului
  - M[i][j] = 0, dacă nu există arc între nodul N_i şi N_j

Grafuri

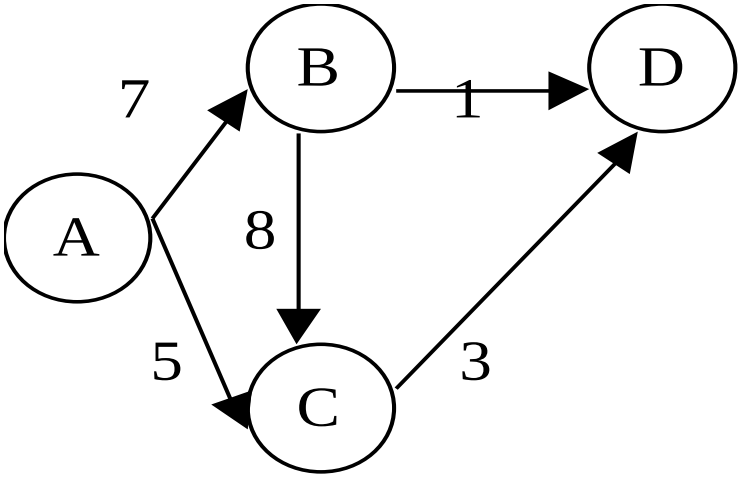
- Graf orientat fără greutate:



$$M[i][j] = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Grafuri

- Graf orientat cu greutate:



$$M[i][j] = \begin{bmatrix} 0 & 7 & 5 & 0 \\ 0 & 0 & 8 & 1 \\ 0 & 0 & 0 & 3 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Grafuri

- Reprezentarea prin liste de adiacenţă:
  - nu se cunoaşte numărul de noduri => nu se cunoaşte dimensiunea matricei de adiacenţă;
  - construirea dinamică a structurii de tip graf;
  - reţea de liste înlănţuite;
  - listă a arcelor ce este construită dinamic;
  - descrierea grafului cuprinde mulţimea de noduri şi mulţimea de arce, precizând orientarea arcului şi greutatea lui.

Grafuri

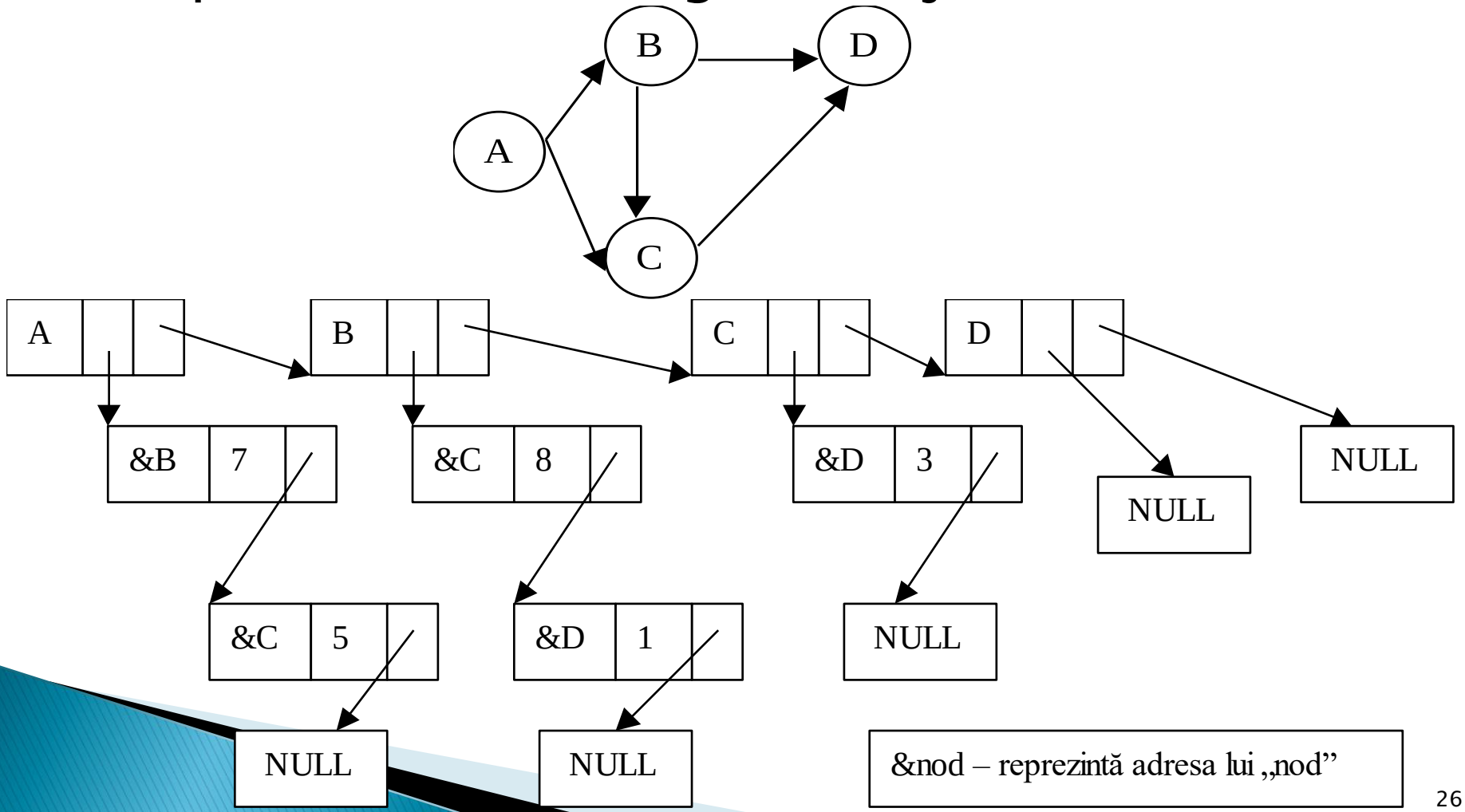
- Reprezentarea prin liste de adiacenţă:

```
struct nodgraf
{
      int info;                  //informaţia nodului;
      struct nodgraf *next;  //referinţă către următorul nod;
      struct arc *capat;        //capătul listei de arce;
}

struct arc
{
      struct nodgraf *destinatie; //adresa nodului catre care exista arc
      struct arc *next_arc; //referinta catre elementul urmator din lista de arce
      int greutate;    //greutatea arcului
}
```

Grafuri

- Reprezentarea unui graf cu ajutorul listelor:



Grafuri

Traversarea unui graf:

- nu este unică;
- oricare nod al grafului este un posibil punct de start al traversării;
- se pune problema trecerii o singura dată prin fiecare nod;
- evitarea revenirii într-un nod vizitat: asocierea unei etichete.

Grafuri

Metode de traversare:

- ***traversarea în adâncime***: Depth – First Traversal;
- ***traversarea în lăţime***: Breadth – First Traversal.

Grafuri

Traversarea în adâncime a unui graf:

- algoritm tip backtracking;
- algoritm similar cu traversarea în preordine a unui arbore;
- utilizare structuri de ajutor: vector, listă, stivă;
- nodurile sunt explorate începând cu ultimul nod selectat.

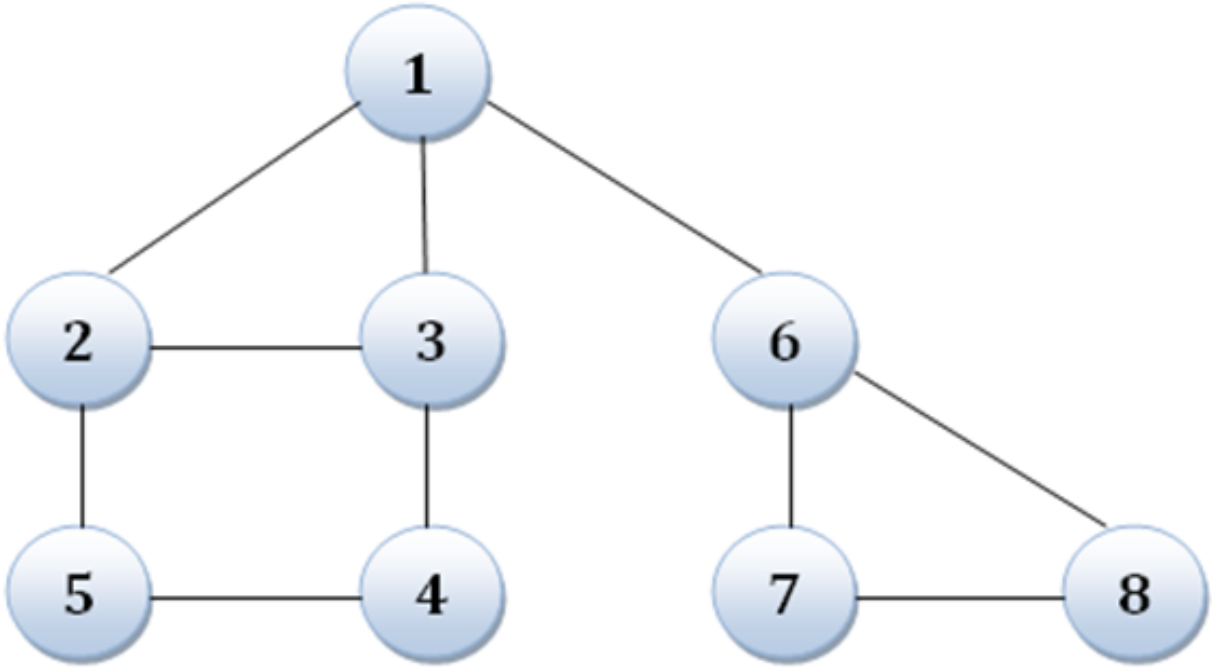
Grafuri

Traversarea în adâncime:

- se alege un nod oarecare; dacă există un nod adiacent care nu a fost vizitat, atunci el va fi noul punct de plecare;
- când toate nodurile adiacente au fost marcate, se încheie consultarea începută în nodul inițial;
- dacă au rămas noduri nevizitate, se alege unul și se reia algoritmul până când toate nodurile sunt marcate.

Grafuri

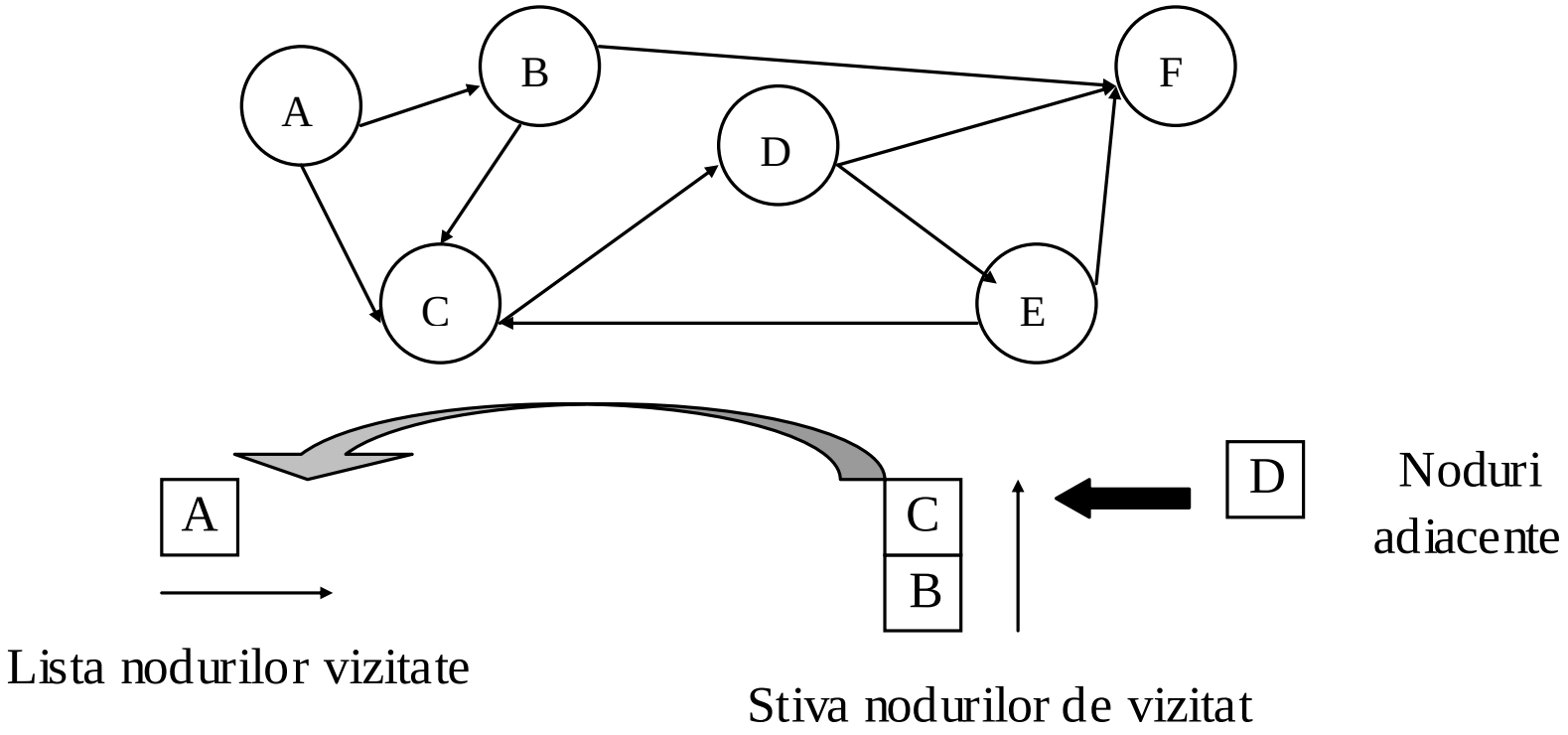
Traversarea în adâncime:



$1 \rightarrow 6 \rightarrow 8 \rightarrow 7 \rightarrow 3 \rightarrow 4 \rightarrow 5 \rightarrow 2$

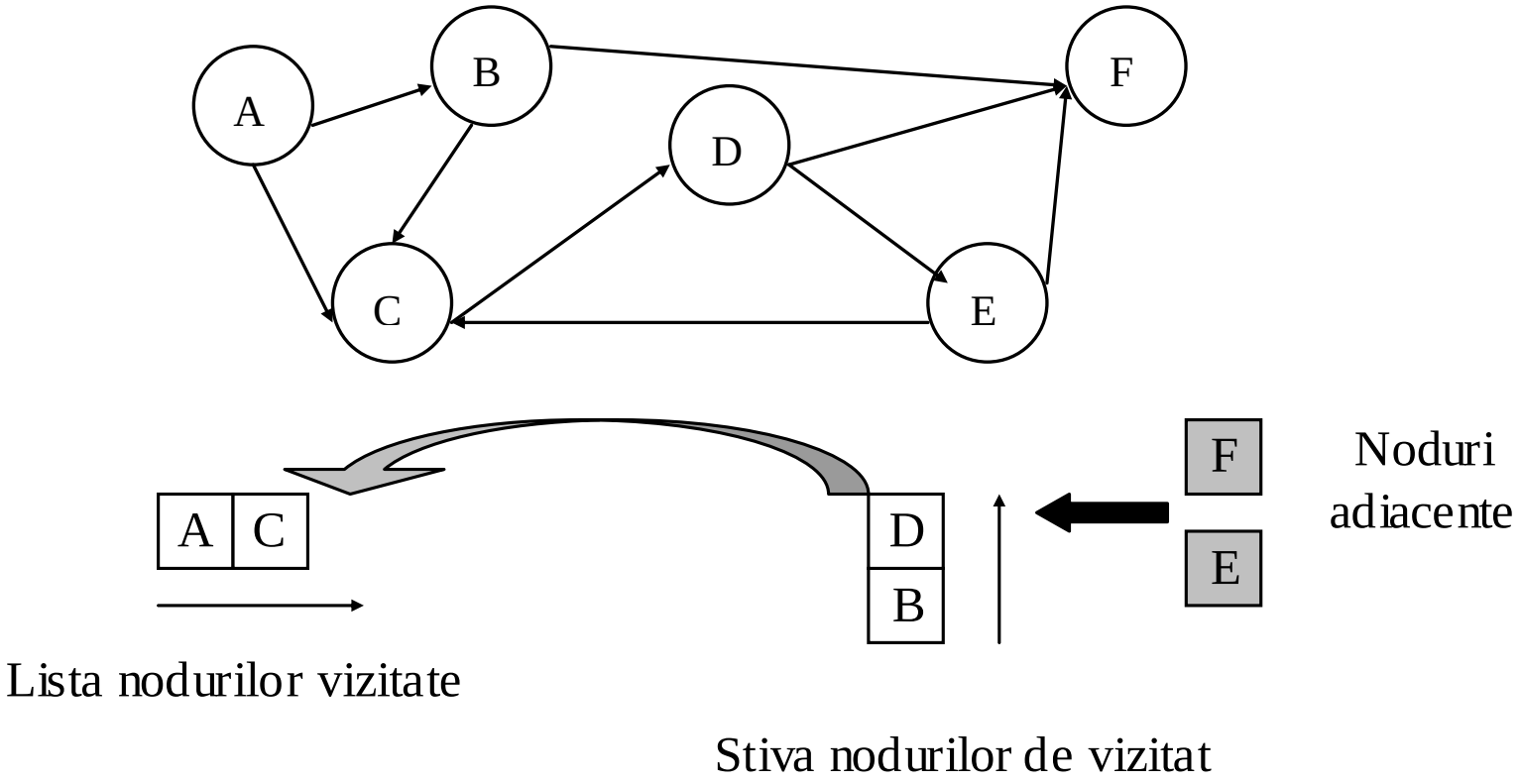
Grafuri

Traversarea în adâncime:



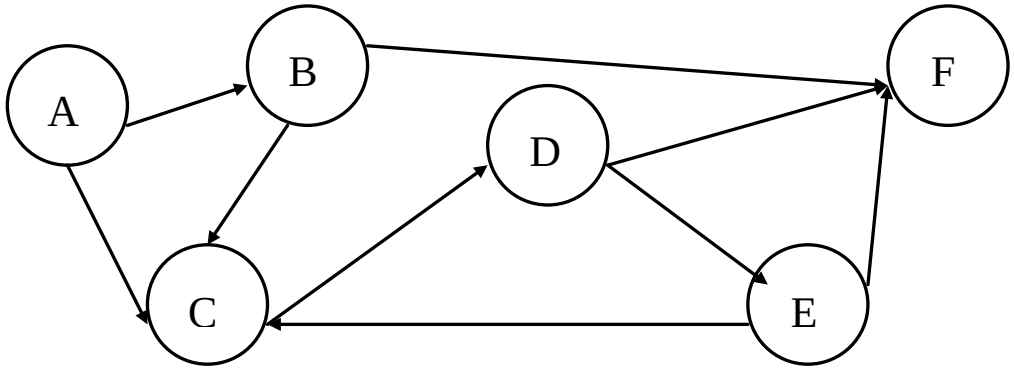
Grafuri

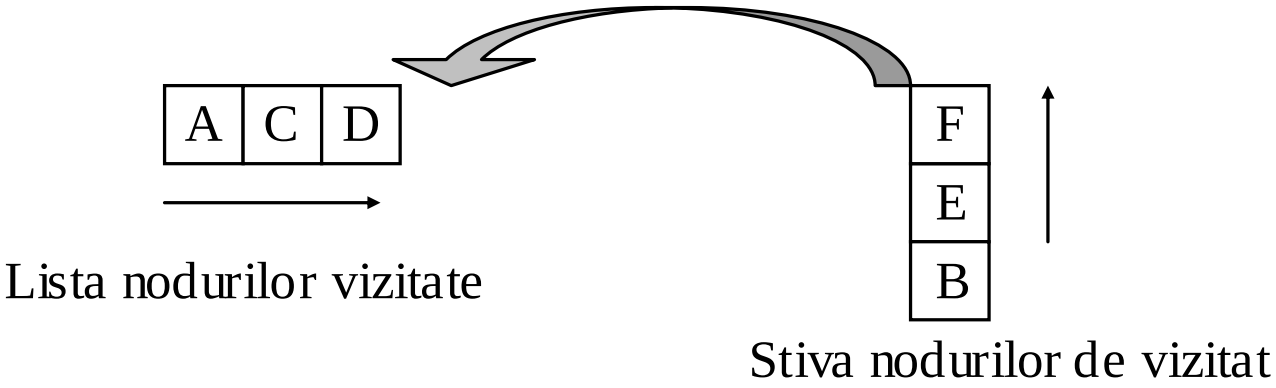
Traversarea în adâncime:



Grafuri

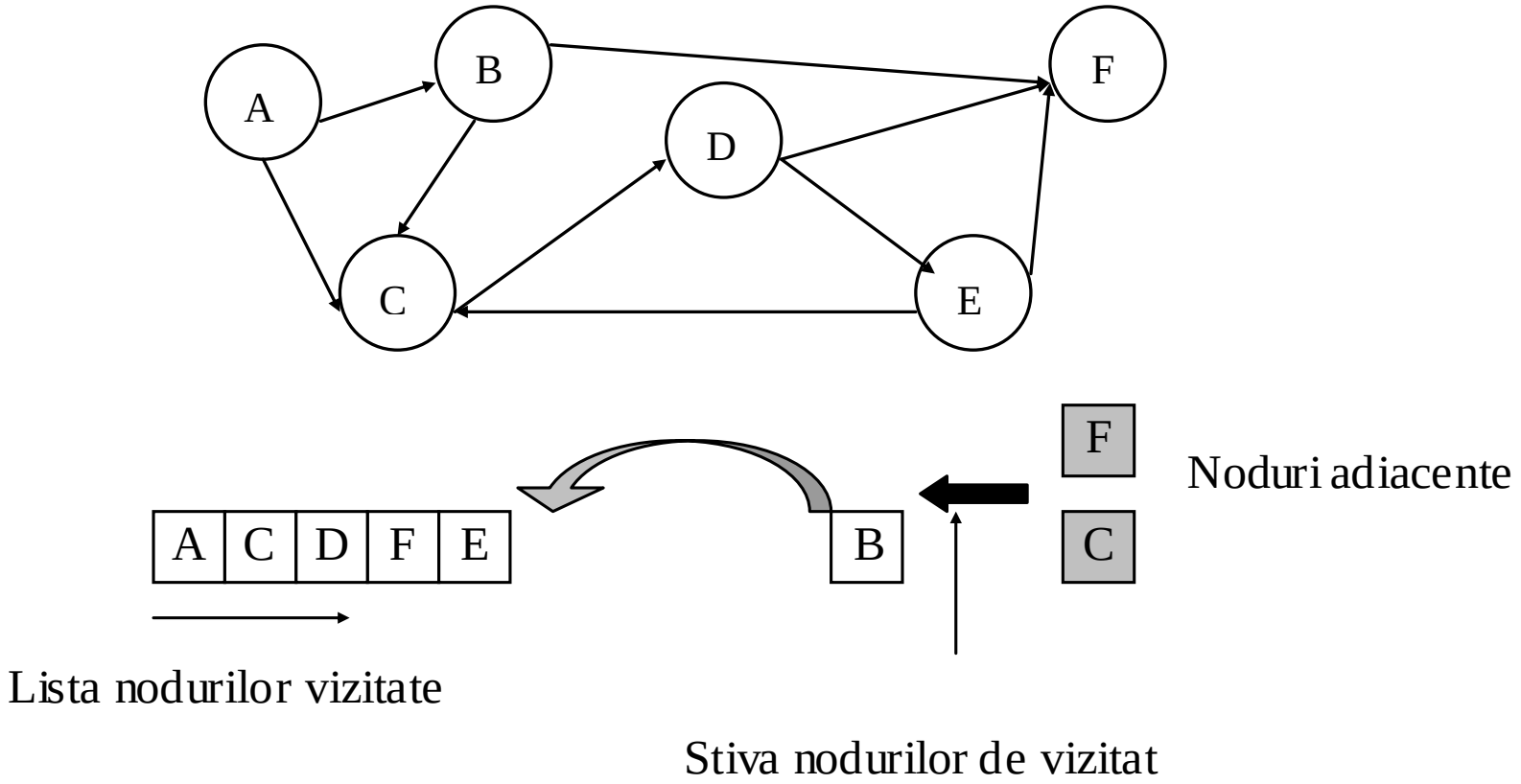
Traversarea în adâncime:





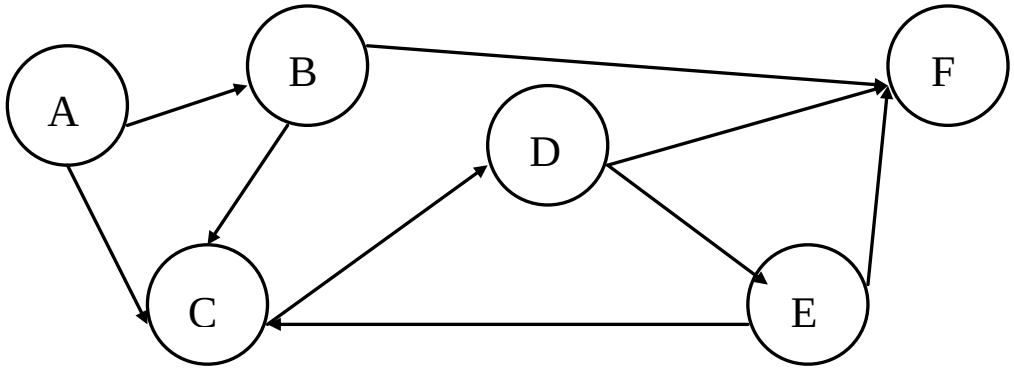
Grafuri

Traversarea în adâncime:



Grafuri

Traversarea în adâncime:



A	C	D	F	E	B

Grafuri

Traversarea în lăţime a unui graf:

- analog procesului de traversare în inordine a unui arbore;
- folosirea unei structuri de tip coadă pentru nodurile de verificat;
- algoritmul parcurge sistematic graful pentru a descoperi toate nodurile accesibile din nodul sursă;
- nodurile se pot afla în una dintre cele trei stări: nevizitat, vizitat sau procesat.

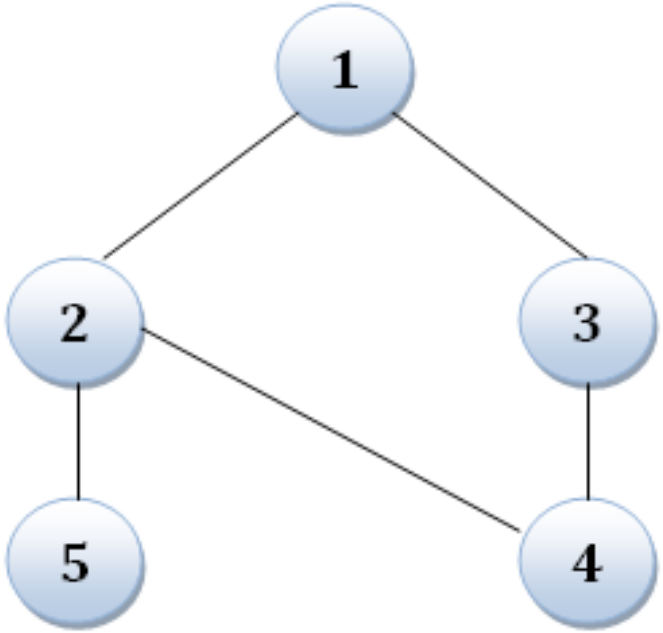
Grafuri

Traversarea în lăţime:

- se vizitează nodul inițial, apoi vecinii acestuia, apoi vecinii nevizitați ai acestora și așa mai departe.

Grafuri

Traversarea în lăţime:



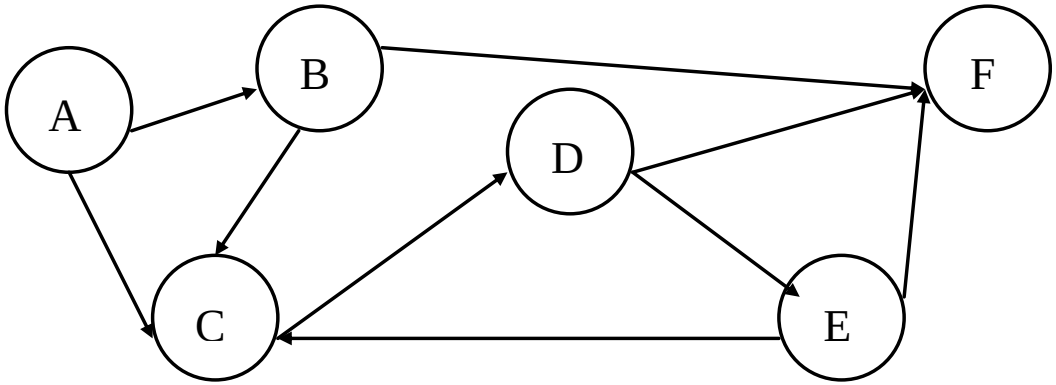
$1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5$

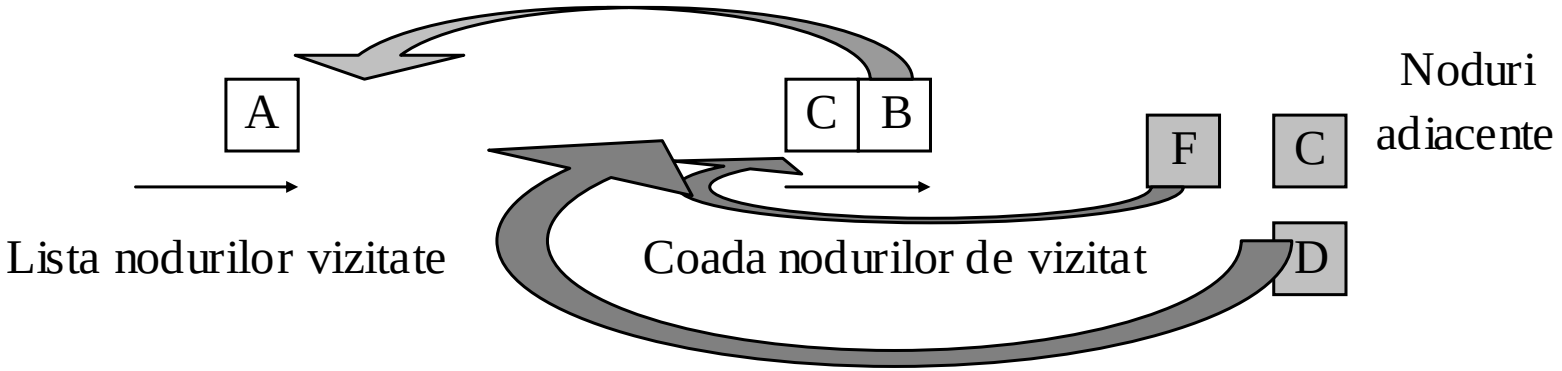
$5 \rightarrow 2 \rightarrow 1 \rightarrow 4 \rightarrow 3$

$2 \rightarrow 1 \rightarrow 4 \rightarrow 5 \rightarrow 3$

Grafuri

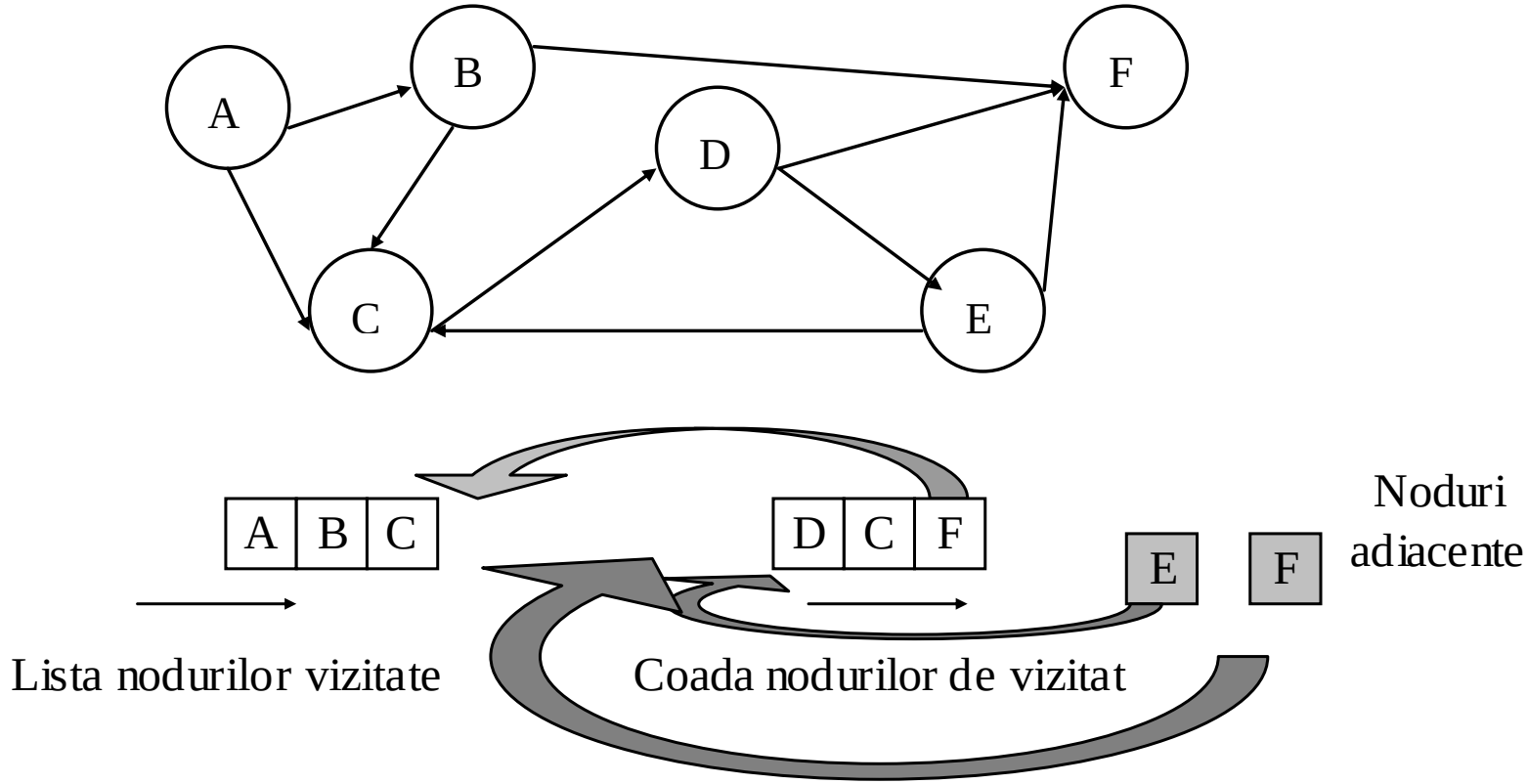
Traversarea în lăţime:





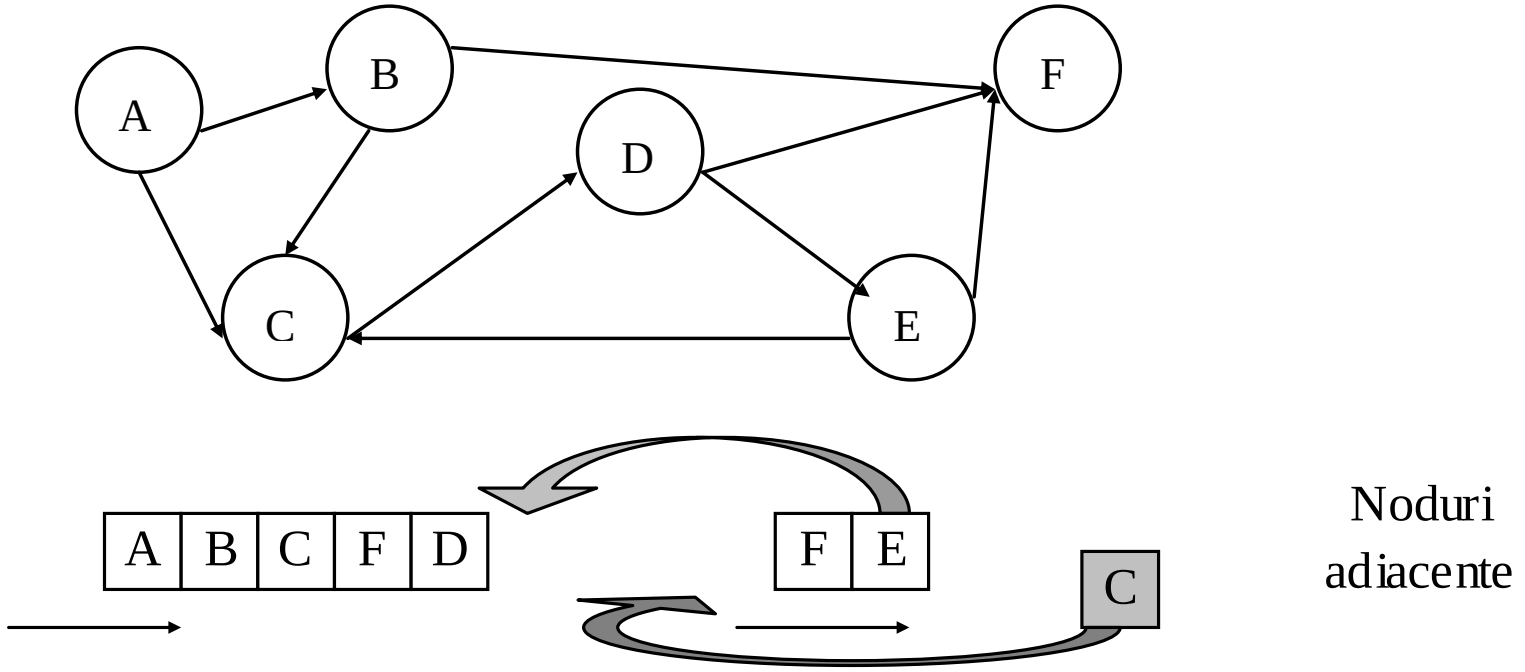
Grafuri

Traversarea în lăţime:



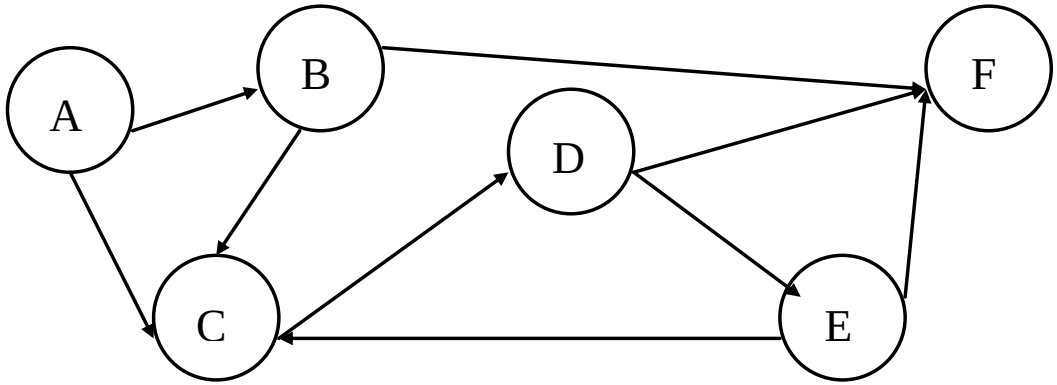
Grafuri

Traversarea în lăţime:



Grafuri

Traversarea în lăţime:



A	B	C	F	D	E

Bibliografie

- Ion Ivan, Marius Popa, Paul Pocatilu (coordonatori) – *Structuri de date*, Editura ASE, Bucureşti, 2008.
  - Cap. 16. Grafuri