MODULUL 2 - GRAFURI. ARBORI RĂDĂCINĂ

1 Grafuri - noţiuni de bază

1.1 Definiţii şi notaţii

Definiţie - graf orientat: Se numeşte *graf orientat* o pereche ordonată $G = (N, A)$, în care N reprezintă o mulţime finită de elemente numite noduri sau vârfuri, iar $A = \{a | a = (x, y), x, y \in N\}$ o mulţime de de perechi ordonate de elemente din N care se numesc arce. Numărul de noduri $n = |N|$ se numeşte *ordinul grafului*. Un arc $a = (x, x)$ se numeşte buclă.

Un exemplu de graf orientat este prezentat în figura 1.

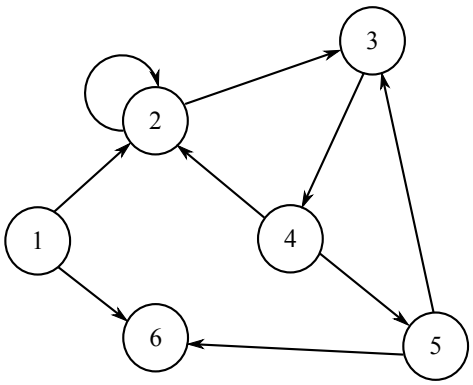


Figure 1: Exemplu de graf orientat

Definiţie - extremităţi: Se consideră un graf orientat $G = (N, A)$ şi $a = (x, y) \in A$ o muchie. Atunci

- x, y se numesc *extremităţi* ale arcului a
- x se numeşte *extremitate iniţială* a arcului şi predecesor al nodului y
- y se numeşte *extremitate finală* a arcului a şi succesor al nodului x
- x şi y se numesc *noduri adiacente*.

Notaţii:

$V^+(x) = \{y|(x,y) \in A\}$ = mulţimea succesorilor lui x

$V^-(x) = \{y|(y,x) \in A\}$ = mulţimea predecesorilor lui y

$V(x) = V^+(x) \cup V^-(x)$ = mulţimea vecinilor lui x

Exemplu: În graful din fig. 1 avem $V^+(4) = \{2,5\}, V^-(4) = \{3\}$

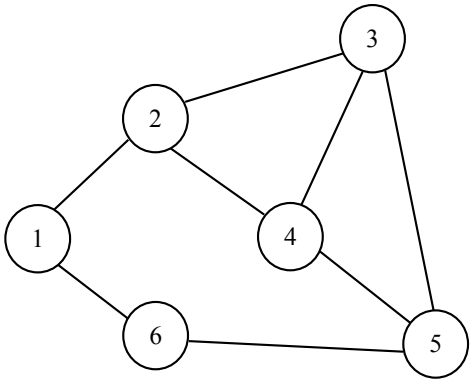


Figure 2: Exemplu de graf neorientat

Definiţie - graf neorientat: Se numeţe *graf neorientat* o pereche ordonată $G = (N, A)$ formată dintr-o mulţime de noduri N şi o mulţime de perechi neordonate $A = \{a|a = [x,y] = [y,x], x,y \in N\}$.

În cazul unei muchii $[x,y]$ dintr-un graf neorientat noţiunile de predecesor şi succesor îşi pierd semnificaţia. Se păstrează semnificaţia noţiunilor de noduri adiacente şi extremităţi. Mulţimea vecinilor lui x în cazul unui graf neorientat este:

$$V(x) = \{y|[x,y] \in A\}.$$

În figura 2 este prezentat un exemplu de graf neorientat.

Definiţie - lanţ: Fie graful orientat $G = (N, A)$. Se numeşte *lanţ* de la nodul x_1 la nodul x_{k+1} o secvenţă $L = (x_1, a_1, x_2, a_2, \ldots, a_k, x_{k+1}), x_i \in N$ pentru $i = \overline{1, k+1}$, $a_i \in A$, pentru $i = \overline{1,k}$ şi $a_i = (x_i, x_{i+1})$ sau $a_i = (x_{i+1}, x_i)$. x_1 şi x_{k+1} se numesc extremităţile lanţului.

- Dacă $a_i = (x_i, x_{i+1})$, atunci a_i se numeşte *arc direct*.
- Dacă $a_i = (x_{i+1}, x_i)$, atunci a_i ne numeşte *arc invers*.
- Dacă $x_1 = x_{k+1}$, atunci L se numeşte *ciclu*.

Putem reprezentat lanţul L în mod simplificat $L = (a_1, a_2, \ldots, a_k)$.

Exemplu: În graful orientat din figura 1 $L = \{(1,2),(2,3),(5,3),(4,5),\ (3,4)\}$ este un lanţ în care arcele $\{(1,2),(2,3)\}$ sunt arce directe, iar $\{(5,3),\ (4,5),(3,4)\}$ sunt arce inverse.

Definiţie - drum Fie $G = (N, A)$ un graf orientat. Se numeşte *drum* în G un lanţ $L = (a_1, a_2, \ldots, a_k)$ în care toate arcele sunt arce directe. Un ciclu care este drum se numeşte *circuit*.

Exemplu: În graful din figura 1 $L_1 = \{(1,2),(2,3),(3,4),(4,2)\}$ este drum, iar $L_2 = \{(2,3),(3,4),(4,2)\}$ este circuit.

Observaţie: noţiunea de drum se defineşte şi în cazul grafurilor neorientate.

Definiţie - graf ponderat: Se numeşte *graf orientat ponderat* sau *reţea orientată* un graf orientat $G = (N, A)$ pentru care se defineşte în plus o funcţie $P : A \to \mathbb{R}$ prin care fiecărei muchii (x, y) i se asociază o pondere/un cost $p(x, y)$.

1.2 Reprezentarea grafurilor

Există două moduri de reprezentare a unui graf orientat $G = (N, A)$ cu $N = \{x_1, x_2, \ldots, x_n\}$ şi A mulţimea muchiilor:

(1) Printr-o matrice de adiacenţă $M = (m_{ij})_{i,j\in 1,n}$ corespunzătoare grafului care se defineşte prin:

$$m_{ij} = \begin{cases} 1, & (x_i, x_j) \in A \\ 0, & \text{altfel} \end{cases}$$

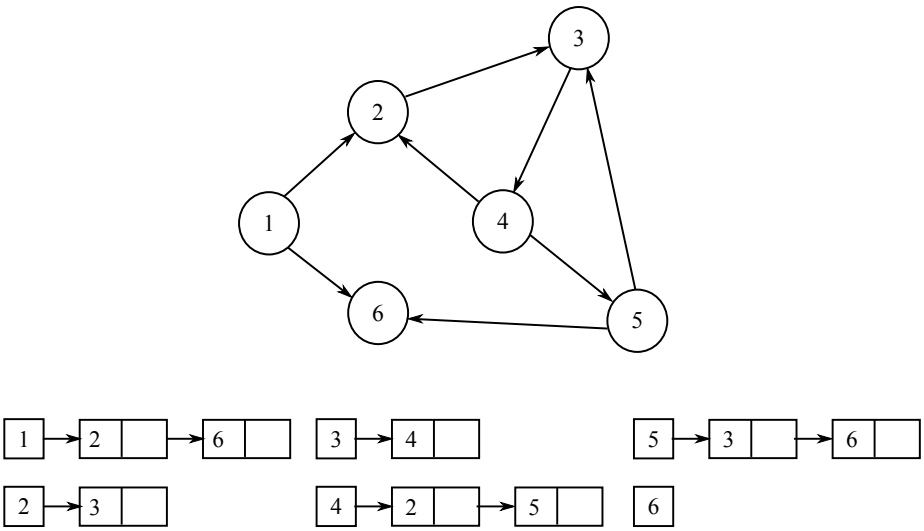
(2) Prin liste de adiacenţă. Pentru fiecare nod $x_i \in N$ se construieşte o listă de adiacenţă (reprezentată printr-o listă înlănţuită) care conţine toate nodurile din $V^+(x_i)$.

Exemplu: Se consideră graful orientat din figură.

(1) Matricea de adiacenţă:

$$\begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

(2) Liste de adiacenţă:



Observaţii:

- În cazul unui graf neorientat matricea de adiacenţă este simetrică.
- În cazul unui graf cu multe noduri şi relativ puţine muchii, reprezentarea prin liste de adiacenţă este mai compactă şi de obicei preferată.
- În anumiţi algoritmi, în care este utilă determinarea rapidă a existenţei unei muchii între două noduri, se preferă reprezentarea sub formă de matrice de adiacenţă (ex. Alg Floyd-Warshall).

2 Arbori oarecare

2.1 Definiţii şi notaţii

Definiţie - graf conex: Un graf $G = (N, A)$ se numeşte *conex*, dacă pentru oricare $x, y \in N$, $x \neq y$, există un lanţ care are ca extremităţi pe x şi pe y.

Definiţie - arbore: Un graf $G = (N, A)$ se numeşte *arbore*, dacă este conex şi nu conţine cicluri. Un graf $G = (N, A)$ se numeşte *pădure* dacă nu conţine cicluri. Un exemplu de arbore este prezentat în fig. 3.

Definiţie - arborescenţă / arbore rădăcină: Un graf orientat $G = (N, A)$ se numeşte *arborescenţă* sau *arbore rădăcină* dacă este conex, fără cicluri, iar unul dintre vârfuri r a fost selectat drept *rădăcină*. În plus, oricare ar fi un alt nod $x \in N$, există un unic drum de la r la x.

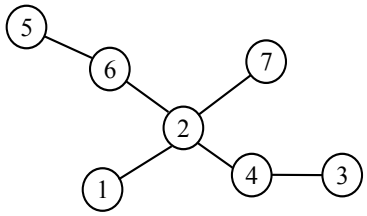


Figure 3: Arbore

Observaţie: De obicei, într-un arbore rădăcină sensul arcelor este dinspre rădăcină către celelalte noduri (*out-tree*). Aşa vor fi consideraţi în continuare arborii în acest curs.

Există şi arbori rădăcină în care orientarea arcelor este inversă, adică de la noduri către rădăcină (*in-tree*). În acest al doilea caz există un drum unic de la orice nod x către rădăcina r.

Proprietăţi:

- Un arbore cu n vârfuri are exact $n-1$ arce.
- Oricare două vârfuri dintr-un arbore sunt conectate prin exact un lanţ.
- Eliminarea orcărui arc are ca rezultat un graf neconex.
- Adăugarea unui nou arc între vârfuri existente produce un ciclu.

Definiţie - ascendent / descendent: Se consideră un arbore de rădăcină r şi x un nod oarecare al arborelui. Atunci:

- Un nod y pentru care (y, x) este o muchie din arbore se numeşte *ascendent direct* sau *părinte* al lui x. Într-un arbore fiecare nod diferit de rădăcină are un singur părinte iar rădăcina nu are nici un părinte.
- Un nod y pentru care (x, y) este o muchie din arbore se numeşte *descendent direct* sau *fiu* al lui x. Un nod oarecare poate avea 0 sau mai mulţi fii.
- Doi descendenţi direcţi ai aceluiaşi nod se numesc *fraţi*.
- Un nod care nu are nici un fiu se numeşte *frunză*.
- Un nod care nu este frunză se numeşte *nod intern* al arborelui.
- Un nod y pentru care există un drum de la x la y se numeşte *descendent* al lui x.

Reprezentarea pe niveluri:

- Pe nivelul 0 se reprezintă nodul rădăcină r.
- Pentru fiecare nod x aflat pe un nivel k, descendenţii săi direcţi se află pe nivelul $k+1$.

În figura 4 este reprezentat un arbore rădăcină pe niveluri.

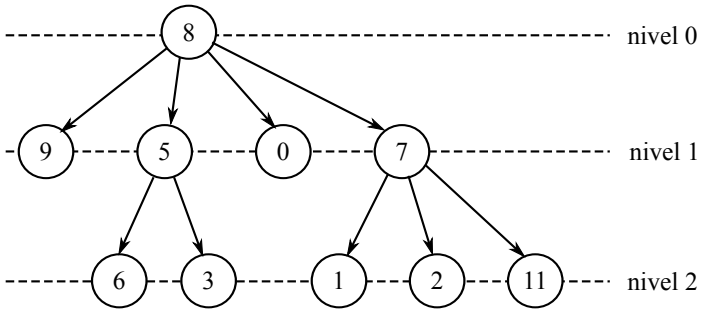


Figure 4: Arbore rădăcină reprezentat pe niveluri.

Definiţie - adâncimea / înălţimea unui arbore: Lungimea drumului de la rădăcina r la un nod x se numeşte adâncimea nodului x. Adâncimea rădăcinii este 0.

Lungimea celui mai lung drum de la x la o frunză se numeşte *înălţime* şi se notează prin $h(x)$.

Definiţie - subarbore: Se numeşte *subarbore de rădăcină* x al unui arbore, arborele format din nodul x împreună cu toţi descendenţii săi.

Arbori n-ari - definiţii

- Se numeşte **arbore n-ar** un arbore pentru care fiecare nod are cel mult n fii.
- Un arbore n-ar se numeşte *arbore plin*, dacă fiecare nod intern are exact n fii.
- Un arbore n-ar se numeşte *arbore perfect*, dacă dacă fiecare nod intern are exact n fii şi toate frunzele au aceeaşi adâncime.
- Un arbore n-ar se numeşte *complet*, dacă toate nodurile interne, cu excepţia eventual a celor de pe penultimul nivel, au exact n descendenţi, iar nodurile de pe ultimul nivel sunt aşezate cel mai la stânga posibil pe nivelul respectiv.

În figura 5 este reprezentat un arbore complet.

Definiţie - Arbore binar: Un arbore binar este un arbore în care fiecare nod are cel mult doi descendenţi direcţi. Atunci când fiecare nod are 0 sau 2 fii, arborele se numeşte *arbore binar strict*. În cazul unui arbore binar un nod are un *fiu stâng* şi un *fiu drept*.

2.2 Reprezentarea arborilor

Reprezentare secvenţială - se pretează în cazul arborilor completi.

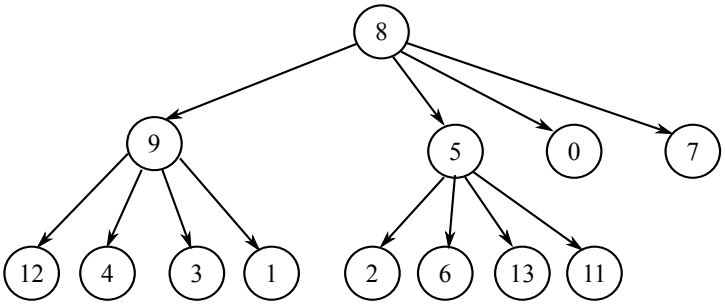


Figure 5: Arbore complet.

Un arbore n-ar complet de rădăcină r poate fi memorat într-un tablou liniar - *array* - în care pe poziţia 0 se află rădăcina, pe următoarele n poziţii se află fii rădăcinii şi în general pentru un nod oarecare aflat pe poziţia i fiul al k-lea, $1 \leq k \leq n$ se află pe poziţia $n*i+k$.

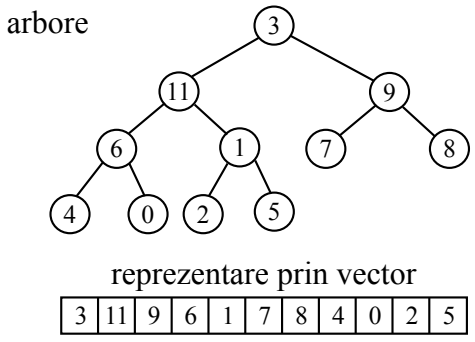


Figure 6: Arbore binar complet memorat într-un vector.

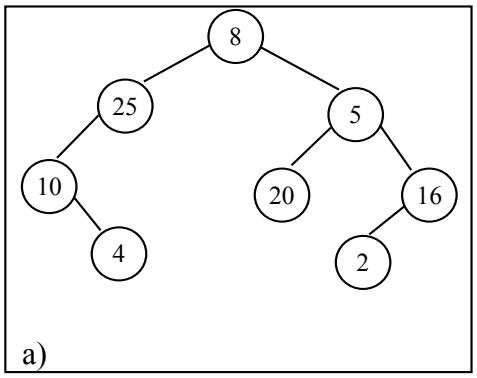
Reprezentare mixtă (pentru arbori binari)

Reprezentarea mixtă utilizează 3 vectori. Primul - INFO - conţine informaţiile din noduri, al doilea - *left* - are pe poziţia i indicele fiului stâng al nodului i - în vectorul INFO -, iar vectorul al treilea - *right* - are pe poziţia i indicele fiului drept al nodului i.

Pentru exemplul din fig. 7 a) reprezentarea mixtă este dată prin vectorii din tabelul din figura 7 b).

Reprezentare cu ajutorul unei structuri de noduri

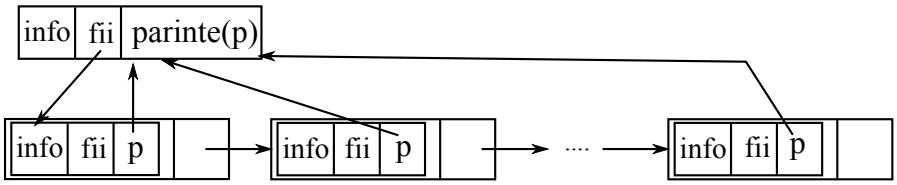
- **Pentru un arbore oarecare** (nu binar): pentru fiecare nod se utilizează o structură în care se reţine informaţia, legătura către o listă / vector de descendenţi direcţi şi (eventual) legătura către nodul părinte.
- Pentru un arbore binar: fiecare nod este reprezentat printr-o structură în care se reţine informaţia, legătura către fiul stâng, legătura către fiul drept şi legătura către părinte.



b)

indice	info	left	right
1	8	2	3
2	25	4	*
3	5	5	6
4	10	*	7
5	20	*	*
6	16	8	*
7	4	*	*
8	2	*	*

Figure 7: a) Arbore binar. b) Tabela de alocare mixtă.



- Pornind de la reprezentarea unui arbore binar, se poate utiliza pentru arbori oarecare o structură asemănătoare, în care se păstrează informaţia, o legătură - fiu - către cel mai din stânga fiu şi o legătură - frate - către primul frate din dreapta al nodului, precum şi o legătură către părinte. Acest lucru este reprezentat grafic în figura 8.

2.3 Parcurgerea arborilor

Există două moduri generale de parcurgere a arborilor oarecare:

- Parcurgerea în lăţime:
  - presupune parcurgerea pe rând a fiecărui nivel de la stânga spre dreapta.
  - se poate realiza practic utilizând o coadă C:
- Parcurgerea în adâncime (preordine):
  - În cazul parcurgerii în adâncime, fiii unui nod sunt vizitaţi tot de la stânga spre dreapta, dar trecerea de la nodul curent la fratele din dreapta se realizează numai dupa vizitarea tuturor descendenţilor nodului curent, deci a întregului subarbore al nodului respectiv.
  - Se poate realiza utilizând o stivă S:

În cazul arborelui din figura 8, parcurgerea în lăţime cu algoritmul de mai sus va avea ca rezultat afişarea cheilor în ordinea: 8, 9, 5, 0, 7, 6, 3, 1, 2, 11. Parcurgerea în preordine cu algoritmul de mai sus va avea ca rezultat afişarea cheilor în ordiea: 8, 9, 5, 6, 3, 0, 7, 1, 2, 11.

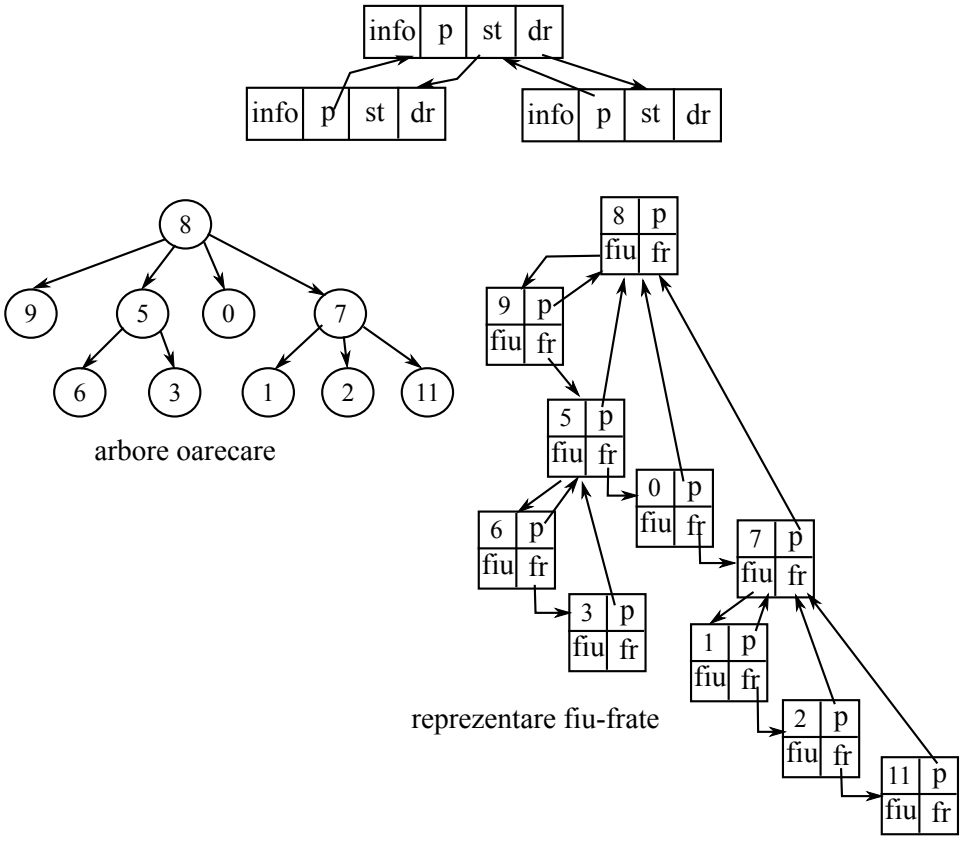


Figure 8: Arbore binar reprezentat printr-o structură de tip fiu-frate.

2.4 Parcurgerea arborilor binari

Parcurgerea unui arbore binar în adâncime presupune parcurgerea fiecărui nod împreună cu subarborele stâng şi subarborele drept. Subarborele stâng se ia în considerare înaintea subarborelui drept.

În funcţie de ordinea în care parcurg rădăcina şi subarborii, se disting trei tipuri de parcurgere:

- **Preordine (RSD)**: rădăcină, subarbore stâng, subarbore drept
- **Inordine (SRD)**: subarbore stâng, rădăcină, subarbore drept
- **Postordine (SDR)**: subarbore stâng, subarbore drept, rădăcină

Exemplu: Se consideră arborele binar din figura 9. Atunci:

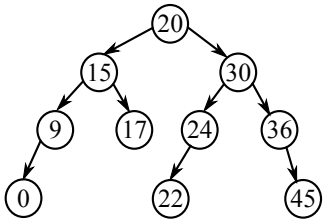


Figure 9: Arbore binar.

- RSD: 20, 15, 9, 0, 17, 30, 24, 22, 36, 45.
- SRD: 0, 9, 15, 17, 20, 22, 24, 30, 36, 45.
- SDR: 0, 9, 17, 15, 22, 24, 45, 36, 30, 20.

Exemple de aplicaţii ale arborilor:

- Sortare: Heap-sort
- Codificare: arborele Huffmann
- Compilatoare: arbori sintactici de derivare
- Procesare de imagine / grafică: arbori Quad, arbori PR
- Clasificare: arbori de decizie
- Dicţionare, căutare eficientă: arbori de căutare.