Notiţe

Notiţe

BF - Breadth First (Parcurgerea pe nivele)

Definiția 3.5.2. *Fie $G = (V, E)$ un graf și $x \in V$ un nod arbitrar fixat.* ***Parcurgerea în lățime*** *(**BF**,"breadth first",* ***parcurgerea pe nivele****) a grafului G pornind din nodul x, numit și* ***rădăcină*** *a acestei parcurgeri, constă în:*

- *se vizitează nodul x, considerat nod de nivelul zero;*
- *se vizitează apoi succesorii direcți nevizitați ai acestuia (diferiți de x), considerați noduri de nivelul 1;*
- *se vizitează apoi, pe rând, succesorii direcți nevizitați ai acestora, considerați noduri de nivelul 2;*
  ș.a.m.d.;
- *parcurgerea se încheie când niciun nod de pe un nivel nu mai are succesori direcți nevizitați.*

Observația 3.5.6. Analog parcurgerii DF, considerând câte o muchie sau un arc de la fiecare nod curent v al parcurgerii BF la fiecare din nodurile nevizitate (de pe următorul nivel) pentru care v este predecesorul direct, se obține un arbore, numit **arbore BF**.

Exemplul 3.5.2. Pentru graful din Exemplul 3.1.2, parcurgerea în lățime pornind din nodul 2 este

$$BF(2) : 2, 3, 4, 5, 1$$

(considerând din nou ordinea dintre succesorii direcți ai fiecărui nod ca fiind ordinea crescătoare). Arborele BF corespunzător acestei parcurgeri este reprezentat în Figura 3.5.3.

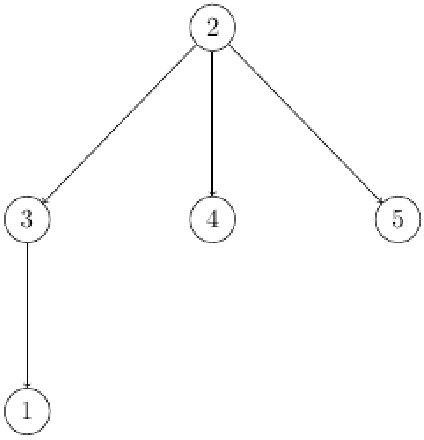

Figura 3.5.3:

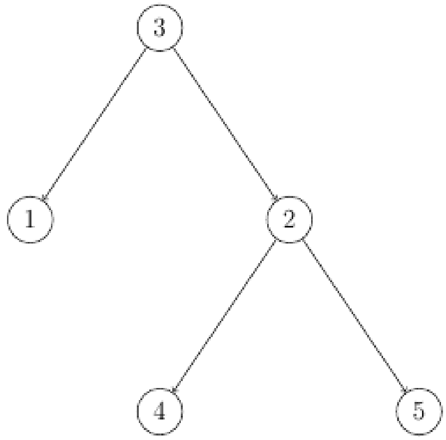

Figura 3.5.4:

Pentru același graf, parcurgerea BF pornind din nodul 3 este

$$BF(3) : 3, 1, 2, 4, 5,$$

iar arborele BF corespunzător este reprezentat în Figura 3.5.4.

DF - Depth First (Parcurgerea în adâncime)

3.5 Parcurgerea grafurilor

Prin parcurgerea unui graf se înțelege o metodă sistematică de vizitare succesivă a nodurilor sale (în vederea prelucrării informațiilor atașate în structura de date modelată prin graful dat).

Definiția 3.5.1. *Fie $G = (V, E)$ un graf și $x \in V$ un nod arbitrar fixat.* ***Parcurgerea în adâncime*** *(**DF**, "**d**epth **f**irst") a grafului G pornind din nodul x, numit și* ***rădăcină*** *a acestei parcurgeri, constă în:*

- *se vizitează nodul x, acesta devine nod curent;*
- *dacă nodul curent v_i are succesori direcți (adică noduri v_j pentru care există muchie sau arc de la v_i la v_j) nevizitați, atunci se vizitează primul astfel de nod v_j; nodul v_j devine nod curent și se continuă procedeul de parcurgere pornind din acest nod;*
- *dacă nodul curent v_j nu mai are succesori direcți nevizitați, atunci se revine la nodul predecesor direct v_i (cel din care a fost vizitat); nodul v_i redevine nod curent și se continuă procedeul de parcurgere pornind din acest nod;*
- *dacă nodul curent nu mai are nici succesori direcți nevizitați, nici predecesor direct (deci este chiar radăcina x), atunci parcurgerea se încheie.*

Observația 3.5.1. Pentru parcurgerea DF, considerând câte o muchie sau un arc de la fiecare nod curent v_i la primul său succesor direct nevizitat v_j (care va deveni următorul nod curent) se obține un arbore, numit **arbore DF**.

Exemplul 3.5.1. Pentru graful din Exemplul 3.1.2, parcurgerea în adâncime pornind din nodul 2 este

$$DF(2) : 2, 3, 1, 4, 5$$

(considerând că ordinea dintre succesorii direcți ai fiecărui nod este ordinea crescătoare). Arborele DF corespunzător acestei parcurgeri este reprezentat în Figura 3.5.1.

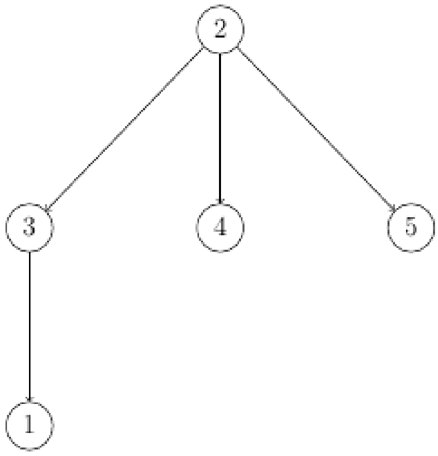

Figura 3.5.1:

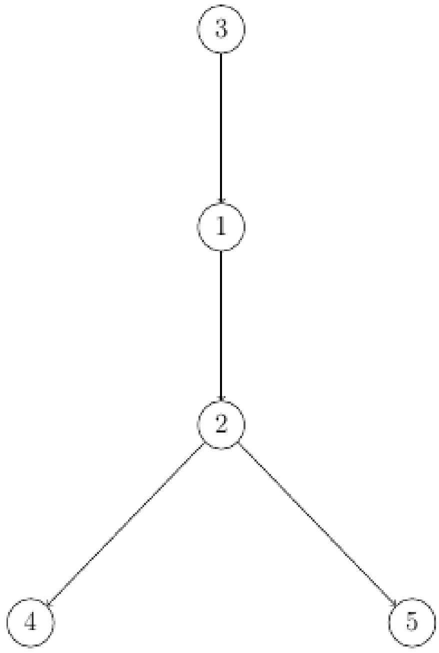

Figura 3.5.2:

Pentru același graf, parcurgerea DF pornind din nodul 3 este

$$DF(3) : 3, 1, 2, 4, 5,$$

iar arborele DF corespunzător este reprezentat în Figura 3.5.2.

Algoritmul Dijkstra

7.2 Algoritmul Dijkstra

Vom expune un algoritm pentru determinarea distanțelor minime și a drumurilor minime de la un nod fixat, numit și nod sursă, la toate nodurile grafului ponderat dat.

Algoritmul 7.2.1 (**Dijkstra**). Fie (G, c) un graf ponderat, $G = (V, E)$, $V = \{v_1, v_2, \ldots, v_n\}$, $c : E \to \mathbb{R}_+$. Fie $C = (c_{ij})_{i,j=\overline{1,n}}$ matricea distanțelor directe asociată grafului (G, c) și fie $v_s \in V$ un nod arbitrar fixat, numit **nod sursă**. Distanțele minime de la nodul v_s la nodurile grafului sunt calculate și memorate într-un vector $t = (t_1, \ldots, t_n)$ astfel:

- La pasul 1 se selectează nodul sursă v_s și se ia $t_s = 0$;
- La pasul k, $2 \leq k \leq n$, se cunosc nodurile $v_{i_1}, \ldots, v_{i_{k-1}}$ selectate la pașii anteriori și distanțele corespondente $t_{i_1}, \ldots, t_{i_{k-1}}$.

  a) Dacă nu mai există nicio muchie sau arc de la un nod selectat $v_j \in \{v_{i_1}, \ldots, v_{i_{k-1}}\}$ la un nod neselectat $v_i \in V \setminus \{v_{i_1}, \ldots, v_{i_{k-1}}\}$, atunci se ia $t_i = \infty$ pentru orice nod neselectat $v_i \in V \setminus \{v_{i_1}, \ldots, v_{i_{k-1}}\}$ și algoritmul se încheie.

  b) În caz contrar se selectează un nod $v_{i_k} \in V \setminus \{v_{i_1}, \ldots, v_{i_{k-1}}\}$ cu proprietatea că există un nod selectat $v_{j_k} \in \{v_{i_1}, \ldots, v_{i_{k-1}}\}$ astfel încât

  $$t_{j_k} + c_{j_k i_k} = \min\{t_j + c_{ji} | v_j \in \{v_{i_1}, \ldots, v_{i_{k-1}}\}, v_i \in V \setminus \{v_{i_1}, \ldots, v_{i_{k-1}}\}\}. \tag{7.2.1}$$

  Se ia

  $$t_{i_k} = t_{j_k} + c_{j_k i_k} \tag{7.2.2}$$

  și se trece la pasul $k + 1$.

Observația 7.2.1. Evident, algoritmul execută cel mult n pași.

Exemplul 7.2.2. Pentru graful ponderat din Exemplul 5.2.1, luând ca nod sursă nodul $s = 1$, aplicarea Algoritmului Dijkstra este evidențiată în următorul tabel:

Pas	Nodul selectat x	$TATA[x]$	Distanța minimă $t[x]$
1	1	0	0
2	2	1	30
3	5	1	70
4	3	2	80
5	8	5	80
6	6	3	100
7	7	1	110
8	9	5	110
9	4	3	180
10	10	9	240

Arborele drumurilor minime, memorat în vectorul $TATA$, este reprezentat în Figura 7.2.1.

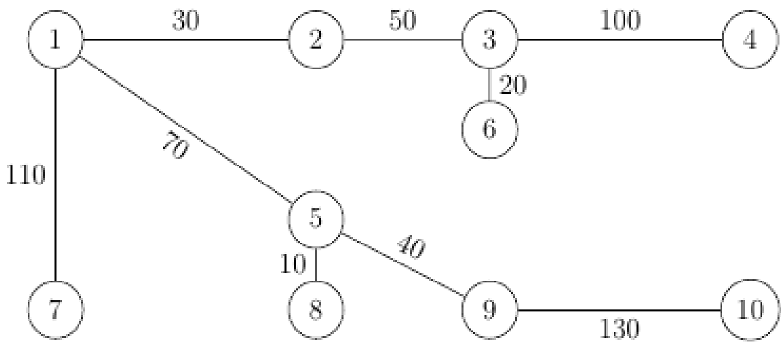

Figura 7.2.1:

Deci drumurile minime determinate de algoritm sunt:

- de la 1 la 1: $[1]$;
- de la 1 la 2: $[1, 2]$;
- de la 1 la 3: $[1, 2, 3]$;
- de la 1 la 4: $[1, 2, 3, 4]$;
- de la 1 la 5: $[1, 5]$;
- de la 1 la 6: $[1, 2, 3, 6]$;
- de la 1 la 7: $[1, 7]$;
- de la 1 la 8: $[1, 5, 8]$;
- de la 1 la 9: $[1, 5, 9]$;
- de la 1 la 10: $[1, 5, 9, 10]$.

Grad Noduri

3.3 Grade, secvențe grafice

Definiția 3.3.1. *Fie $G = (V, E)$ un graf și $x \in V$ un nod arbitrar fixat.*

a) Dacă G este neorientat, atunci ***gradul*** *lui x, notat $d_G(x) = d(x)$, este definit prin*

$$d(x) = a(x) + 2 \cdot b(x),$$

unde $a(x)$ reprezintă numărul de muchii $e \in E$ ce nu sunt bucle și sunt incidente cu x, iar $b(x)$ este numărul de bucle $e \in E$ ce sunt incidente cu x.

b) Dacă G este orientat, atunci:

- ***gradul de ieșire (semigradul exterior)*** *al lui x, notat $d_G^+(x) = d^+(x)$, reprezintă numărul de arce $e \in E$ incidente cu x spre exterior;*
- ***gradul de intrare (semigradul interior)*** *al lui x, notat $d_G^-(x) = d^-(x)$, reprezintă numărul de arce $e \in E$ incidente cu x spre interior;*
- ***gradul*** *(**total** al) lui x, notat $d_G(x) = d(x)$, este*

$$d(x) = d^+(x) + d^-(x).$$

Exemplul 3.3.1. Pentru graful neorientat din Exemplul 3.1.1, gradele nodurilor sunt: $d(1) = 2$, $d(2) = 4$, $d(3) = 2$, $d(4) = 4$, $d(5) = 4$, $d(6) = 2$.

Pentru graful orientat din Exemplul 3.1.2, gradele nodurilor sunt:

$$\begin{aligned}
&d^+(1) = 1,\ \ d^-(1) = 1,\ \ d(1) = 2,\\
&d^+(2) = 3,\ \ d^-(2) = 2,\ \ d(2) = 5,\\
&d^+(3) = 2,\ \ d^-(3) = 1,\ \ d(3) = 3,\\
&d^+(4) = 0,\ \ d^-(4) = 2,\ \ d(4) = 2,\\
&d^+(5) = 1,\ \ d^-(5) = 1,\ \ d(5) = 2.
\end{aligned}$$

Graf Eulerian

6.1 Grafuri euleriene

Definiția 6.1.1. *a) Fie $G = (V, E)$ un graf. Un lanț simplu, ciclu, drum simplu sau circuit în graful G ce conține toate muchiile sau arcele lui G se numește* ***eulerian****.*

b) Un graf neorientat se numește ***eulerian*** *dacă are (cel puțin) un ciclu eulerian.*

c) Un graf orientat se numește ***eulerian*** *dacă are (cel puțin) un circuit eulerian.*

Exemplul 6.1.1. Graful neorientat reprezentat în Figura 6.1.1 este eulerian, un ciclu eulerian al său fiind $C = [1, 2, 3, 4, 10, 9, 3, 6, 2, 5, 8, 7, 5, 1]$.

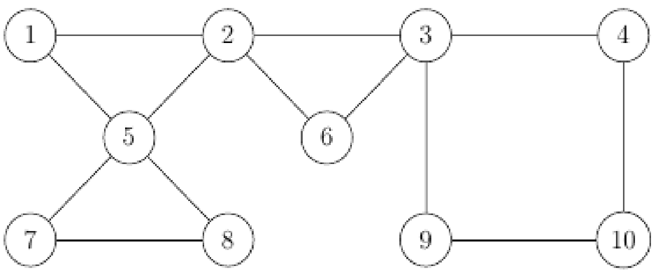

Figura 6.1.1:

Exemplul 6.1.2. Fie graful orientat reprezentat în Figura 6.1.2. Un drum eulerian în acest graf este $(1, 2, 3, 4, 6, 3, 5, 6, 1, 5, 4)$.

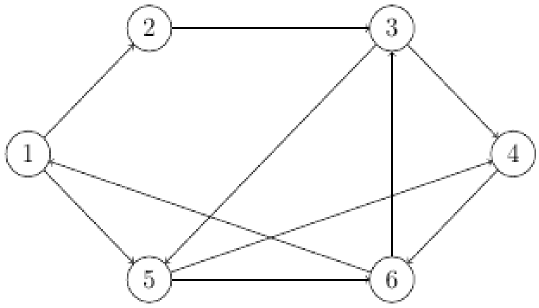

Figura 6.1.2:

Graf Hemiltonian

6.2 Grafuri hamiltoniene

Definiția 6.2.1. *a) Fie $G = (V, E)$ un graf. Un lanț elementar, ciclu elementar, drum elementar sau circuit elementar în graful G ce conține toate nodurile lui G se numește* ***hamiltonian****.*

b) Un graf neorientat se numește ***trasabil*** *dacă are un singur nod sau are (cel puțin) un lanț hamiltonian deschis.*

c) Un graf neorientat se numește ***hamiltonian*** *dacă are (cel puțin) un ciclu hamiltonian.*

d) Un graf orientat se numește ***trasabil*** *dacă are un singur nod sau are (cel puțin) un drum hamiltonian deschis.*

e) Un graf orientat se numește ***hamiltonian*** *dacă are (cel puțin) un circuit hamiltonian.*

Observația 6.2.1. Orice graf hamiltonian este trasabil, deoarece prin eliminarea unei muchii a unui ciclu hamiltonian rămâne un lanț hamiltonian deschis, iar prin eliminarea unui arc al unui circuit hamiltonian rămâne un drum hamiltonian deschis. Reciproca nu este adevărată. De exemplu, un graf format dintr-un drum deschis elementar este trasabil, dar nu este hamiltonian.

Exemplul 6.2.1. Graful orientat din Exemplul 6.1.2 este hamiltomian, un circuit hamiltonian fiind $C = (1, 2, 3, 5, 4, 6, 1)$. El este și trasabil, 6 drumuri hamiltoniene deschise obținându-se prin eliminarea a câte unuia din arcele circuitului C și anume $\mu_1 = C \setminus \{(1, 2)\} = (2, 3, 5, 4, 6, 1)$, $\mu_2 = C \setminus \{(2, 3)\} = (3, 5, 4, 6, 1, 2)$, $\mu_3 = C \setminus \{(3, 5)\} = (5, 4, 6, 1, 2, 3)$, $\mu_4 = C \setminus \{(5, 4)\} = (4, 6, 1, 2, 3, 5)$, $\mu_5 = C \setminus \{(4, 6)\} = (6, 1, 2, 3, 5, 4)$, $\mu_6 = C \setminus \{(6, 1)\} = (1, 2, 3, 5, 4, 6)$.

Graf Orientat

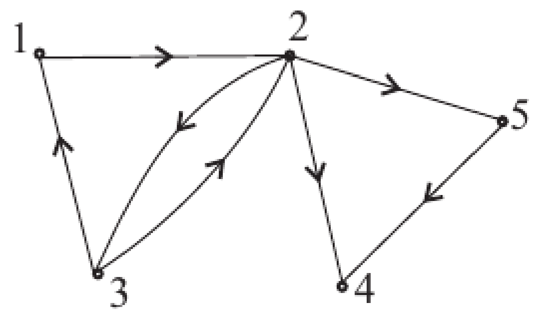

Figura 3.1.3:

Graf Neorientat

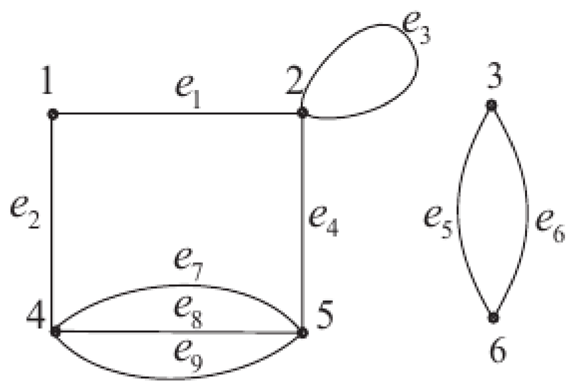

Figura 3.1.1:

Algoritmii lui Kruskal şi Prim

5.2 Algoritmii lui Kruskal și Prim

Algoritmul 5.2.1 (**Kruskal**). Fie (G, c) un graf ponderat conex cu $G = (V, E)$, $V = \{v_1, \dots, v_n\}$. Algoritmul are $n-1$ pași.

- La pasul i, $i = \overline{1, n-1}$, dintre muchiile neselectate la pașii anteriori se selectează o muchie $e_i \in E$ de cost minim cu proprietatea că nu formează cicluri cu muchiile $\{e_1, \dots, e_{i-1}\}$ selectate la pașii anteriori.

Algoritmul 5.2.2 (**Prim**). Fie (G, c) un graf ponderat conex cu $G = (V, E)$, $V = \{v_1, \dots, v_n\}$. Algoritmul are n pași.

- La pasul 0 se selectează un nod arbitrar $x_0 \in V$.

- La pasul i, $i = \overline{1, n-1}$, se selectează o muchie $e_i = [x_j, x_i] \in E$ de cost minim cu proprietatea că are ca extremități un nod $x_j \in V$ selectat la un pas anterior și celălalt nod $x_i \in V$ neselectat la pașii anteriori; se selectează și nodul x_i.

kruskal_exemplu

Arborele parțial de cost minim obținut este reprezentat în Figura 5.2.2. Costul acestui APM este de 510.

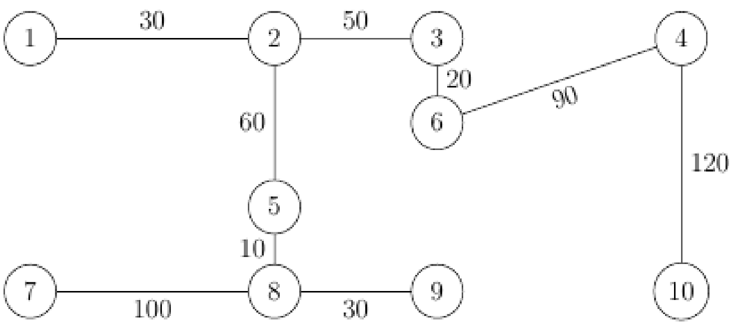

Figura 5.2.2:

Aplicarea Algoritmului Prim pentru același graf este evidențiată în următorul tabel:

Pas	Muchia selectată	Costul ei	Nodul selectat
0	-	-	1
1	$[1, 2]$	30	2
2	$[2, 3]$	50	3
3	$[3, 6]$	20	6
4	$[2, 5]$	60	5
5	$[5, 8]$	10	8
6	$[8, 9]$	30	9
7	$[6, 4]$	90	4
8	$[8, 7]$	100	7
9	$[4, 10]$	120	10

Arborele parțial de cost minim obținut este deci același cu cel obținut prin aplicarea Algoritmului Kruskal.

Matrice de Incidenta

Definiția 3.2.2. *Fie $G = (V, E)$ un graf, unde $V = \{v_1, \ldots, v_n\}$ și $E = \{e_1, \ldots, e_m\}$, $E \neq \emptyset$.*

a) Dacă G este neorientat, atunci ***matricea de incidență*** *asociată grafului G este matricea $B = (b_{ij})_{\substack{i=\overline{1,n} \\ j=\overline{1,m}}}$ definită prin*

$$b_{ij} = \begin{cases} 0, & \text{dacă } e_j \text{ nu este incidentă cu } v_i, \\ 1, & \text{dacă } e_j \text{ nu este buclă și este incidentă cu } v_i, \\ 2, & \text{dacă } e_j \text{ este o buclă incidentă cu } v_i, \end{cases}$$

$\forall\, i \in \{1, \ldots, n\},\ \forall\, j \in \{1, \ldots, m\}$.

b) Dacă G este orientat și fără bucle, atunci ***matricea de incidență*** *asociată grafului G este matricea $B = (b_{ij})_{\substack{i=\overline{1,n} \\ j=\overline{1,m}}}$ definită prin*

$$b_{ij} = \begin{cases} 0, & \text{dacă } e_j \text{ nu este incidentă cu } v_i, \\ 1, & \text{dacă } e_j \text{ este incidentă cu } v_i \text{ spre exterior}, \\ -1, & \text{dacă } e_j \text{ este incidentă cu } v_i \text{ spre interior}, \end{cases}$$

$\forall\, i \in \{1, \ldots, n\},\ \forall\, j \in \{1, \ldots, m\}$.

Exemplul 3.2.2. Matricea de incidență a grafului neorientat din Exemplul 3.1.1 este

$$B = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 2 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \end{pmatrix},$$

iar matricea de incidență a grafului orientat din Exemplul 3.1.2 este

$$B = \begin{pmatrix} 1 & 0 & 0 & 0 & -1 & 0 & 0 \\ -1 & 1 & 1 & 1 & 0 & -1 & 0 \\ 0 & -1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & -1 & 0 & 0 & 1 \end{pmatrix}.$$

Matricea Drumurilor

3.6 Algoritmul Roy-Warshall

Definiția 3.6.1. *Fie $G = (V, E)$ un graf (neorientat sau orientat), unde $V = \{v_1, \ldots, v_n\}$.* ***Matricea drumurilor*** *asociată grafului G este matricea $D = (d_{ij})_{i,j=\overline{1,n}}$ definită prin*

$$d_{ij} = \begin{cases} 1, & \textit{dacă } \exists\, \mu = (v_i, \ldots, v_j) \textit{ drum cu } l(\mu) > 0, \\ 0, & \textit{în caz contrar,} \end{cases}$$

unde $l(\mu)$ reprezintă lungimea drumului μ.

Exemplul 3.6.1. Matricea drumurilor asociată grafului neorientat din Exemplul 3.1.1 este

$$D = \begin{pmatrix} 1 & 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 \end{pmatrix},$$

iar matricea drumurilor asociată grafului orientat din Exemplul 3.1.2 este

$$D = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix}.$$

Exemplul 3.6.2. Pentru graful din Exemplul 3.1.2, prin aplicarea Algoritmului Roy-Warshall obținem matricele:

$$D^{(0)} = A = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix} \text{ (matricea de adiacență);}$$

$$D^{(1)} = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix}; \quad D^{(2)} = \begin{pmatrix} 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

(deoarece, de exemplu, $d_{13}^{(2)} = d_{13}^{(1)} \vee d_{12}^{(1)} d_{23}^{(1)} = 0 \vee 1 \cdot 1 = 0 \vee 1 = 1$);

$$D^{(3)} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix}; \quad D^{(4)} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix};$$

$$D^{(5)} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix} = D \text{ (matricea drumurilor).}$$

Numar Ciclomatic

Teorema 4.1.1 (Teorema numărului ciclomatic). *Orice graf $G = (V, E)$ cu n noduri, m muchii și k componente conexe are numărul ciclomatic*

$$\boxed{\gamma(G) = m - n + k.}$$

Demonstrație. Demonstrăm egalitatea din enunț în două etape.

Etapa 1) Presupunem că graful G este conex, deci $k = 1$. Fie $T = (V, F)$ un arbore parțial al lui G (există, conform Propoziției 4.1.7). Conform Corolarului 4.1.1, card $(F) = n - 1$. Fie

$$E \setminus F = \{e_1, \ldots, e_{m-n+1}\}.$$

Pentru orice $i \in \{1, \ldots, m - n + 1\}$, fie C_i mulțimea muchiilor ciclului elementar unic din $T + e_i$ (conform Propoziției 4.1.8). Atunci

$$\mathcal{B} = \{C_1, \ldots, C_{m-n+1}\}$$

este o bază a spațiului ciclurilor $\mathcal{C}(E)$ (numită **bază de cicluri** a grafului G). Rezultă că

$$\gamma(G) = \text{card}\,(\mathcal{B}) = m - n + 1 = m - n + k.$$

Etapa 2) Fie acum G un graf oarecare și $G_1, \ldots, G_k$ componentele sale conexe. Fie n_i și m_i numerele de noduri, respectiv muchii ale componentei G_i. Evident, $n = n_1 + \cdots + n_k$ și $m = m_1 + \cdots + m_k$.

Deoarece subspațiile ciclurilor componentelor $G_1, \ldots, G_k$ sunt liniar independente, avem

$$\gamma(G) = \gamma(G_1) + \cdots + \gamma(G_k).$$

Dar, conform etapei 1), $\gamma(G_i) = m_i - n_i + 1 \ \forall\, i \in \{1, \ldots, k\}$, și astfel prin adunare rezultă că $\gamma(G) = m - n + k$. □

Observația 4.1.1. Demonstrația teoremei anterioare este constructivă, indicând următorul **algoritm de determinare a unei baze de cicluri** pentru graful G:

- se determină componentele conexe $G_1, \ldots, G_k$;
- pentru fiecare componentă G_i, se determină un arbore parțial T (de exemplu arborele DF sau BF) și ciclurile elementare $C_1, \ldots, C_{m_i - n_i + 1}$ din grafurile $T + e_i$, pentru fiecare muchie e_i din G_i ce nu aparține lui T;
- ciclurile astfel determinate formează împreună o bază de cicluri pentru graful G.

Exemplul 4.1.1. Fie graful G reprezentat în Figura 4.1.1.

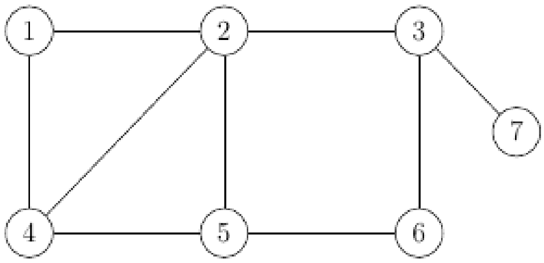

Figura 4.1.1:

Numărul său ciclomatic este $\gamma(G) = m - n + k = 9 - 7 + 1 = 3$.
Considerăm arborele parțial T reprezentat în Figura 4.1.2.

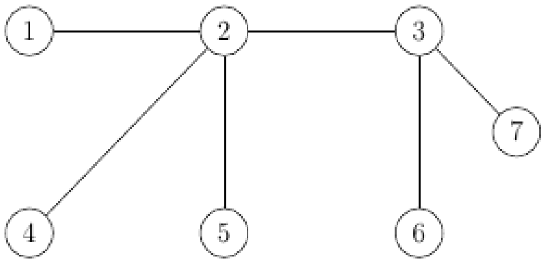

Figura 4.1.2:

Aplicând algoritmul din observația anterioară se obține baza de cicluri $\{C_1, C_2, C_3\}$ unde

$$C_1 = \{[1,4],[4,2],[2,1]\},\ C_2 = \{[4,5],[5,2],[2,4]\} \text{ și } C_3 = \{[5,6],[6,3],[3,2],[2,5]\}.$$