Reprezentarea unui graf

1) Matrice de adiacență:

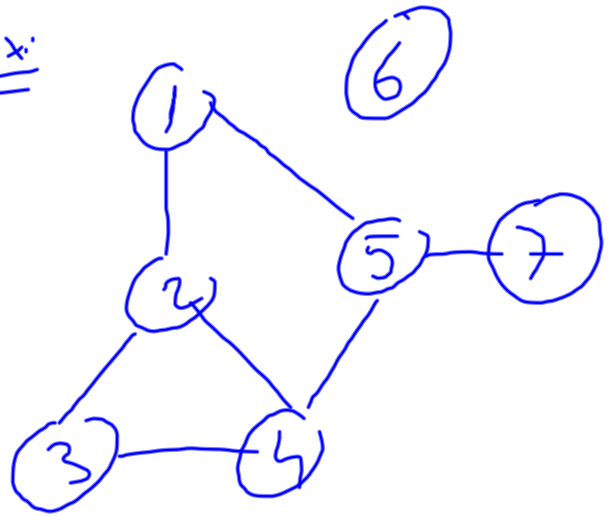
avantaj: ușor de lucrat

dezavantaj: consumă multă memorie.

$$a[i][j] = \begin{cases} 1 & \text{dacă nodurile } i,j \text{ sunt adiacente} \\ 0 & \text{în caz contrar.} \end{cases}$$

La grafurile neorientate, pe diag. principală avem 0 iar matricea este simetrică față de diag. principală.

Ex:



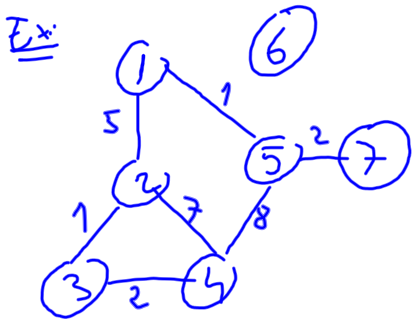
Matr. de adiacență

i \ j	1	2	3	4	5	6	7
1	0	1	0	0	1	0	0
2	1	0	1	1	0	0	0
3	0	1	0	1	0	0	0
4	0	1	1	0	1	0	0
5	1	0	0	1	0	0	1
6	0	0	0	0	0	0	0
7	0	0	0	0	1	0	0

2) Dacă avem un graf ponderat ↔ muchiile au asociate costuri - putem vorbi de <u>matricea costurilor</u>:

$$c[i][j] = \begin{cases} 0 & \text{dacă } i = j \\ \text{costul muchiei } (i,j) & \text{dacă } (i,j) \text{ sunt adiacente} \\ \infty & \text{în rest} \end{cases}$$

Ex:



matricea costurilor:

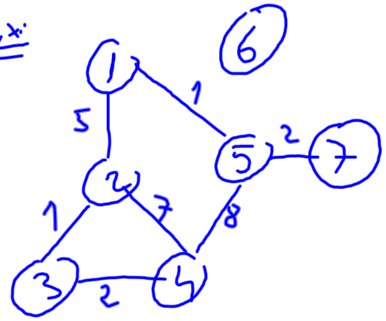
0	5	∞	∞	1	∞	∞
5	0	1	7	∞	∞	∞
∞	1	0	2	∞	∞	∞
∞	7	2	0	8	∞	∞
1	∞	∞	8	0	∞	2
∞	∞	∞	∞	∞	0	∞
∞	∞	∞	∞	2	∞	0

3) Vector de muchii: înșirăm toate muchiile sale.

Ex:

a) vector fără costuri:

Ex:



[1,2] [1,5] [2,3] [2,1] [3,4]

[4,5] [5,7]

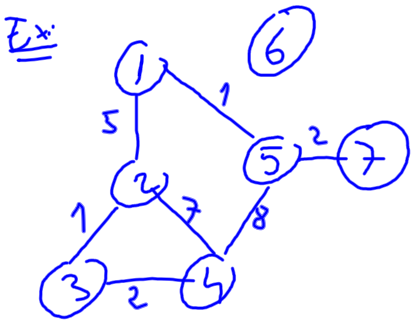
x y cost

b) vector cu costuri: [1,2,5] [1,5,1] [2,3,1] [2,4,7]

[3,4,2] [4,5,8] [5,7,2].

1) Liste de adiacență:

Ex:



Nod	Listă de vecini
1	2, 5
2	1, 3, 4
3	2, 4
4	2, 3, 5
5	1, 4, 7
6	$\emptyset$
7	5

Cu costuri

Nod	Listă vecini
1	(2,5) (5,1)
2	(1,5) (3,1) (4,7)
3	(2,1) (4,2)
4	(2,7) (3,2) (5,8)
5	(1,1) (4,8) (7,2)
6	$\emptyset$
7	(5,2)

APM -> arbore parțial de cost minim.

Dat fiind un graf conex cu costuri, dorim să-i determinăm un graf parțial care să fie arbore iar costul său este minim.

Algoritm (alg. lui Kruskal): alegem muchii în ordine crescătoare și alegem dacă să le păstrăm sau nu în graful parțial în care inițial toate nodurile sunt izolate. Algoritmul se încheie în momentul alegerii a n-1 muchii.

Ex.