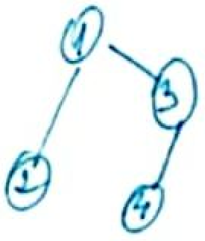
CURS 2 - ALGORITMI GRAFURI - 04.03.2024

$G = (V, E)$

$V = \{v_1, \dots, v_n\}$

$1, 2, \dots, n$



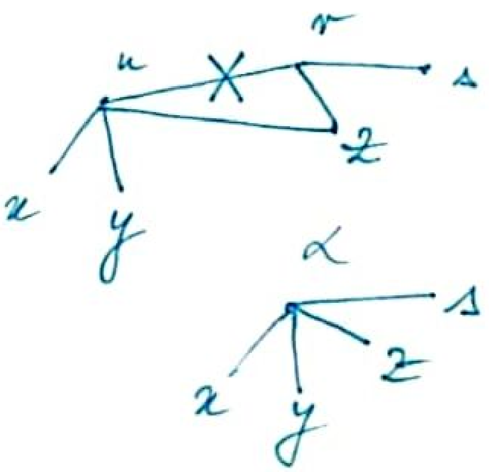
$$\begin{array}{c|c|c|c|c|c}
 & 1 & 2 & 3 & 4 & 5 \\
\hline
1 & 0 & 1 & 1 & 0 & 0 \\
\hline
2 & 1 & 0 & 0 & 0 & 0 \\
\hline
3 & 1 & 0 & 0 & 1 & 0 \\
\hline
4 & 0 & 0 & 1 & 0 & 0 \\
5 & 0 & 0 & 0 & 0 & 0
\end{array}$$

1 → 2,3
2 → 1
3 → 4, 1
4 → 3

$$\sum_{v \in V} d(v) = 2 \cdot m$$

op: $\text{grad\_max}(G) = \max_{v \in V} d(v)$

Test adiacență



Enumerate all edges $O(n^2)$

```
for i = 1..n
   for j = ~~1~~ i+1..n
      dacă M[i,j] = 1 atunci
         print (i,j)
```

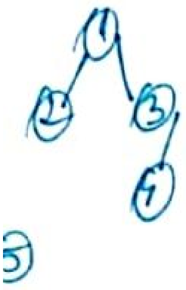
vecini(i)

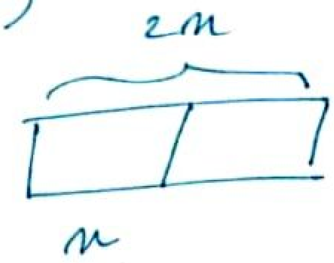
```
for j=1..n
   dacă M[i][j] = 1 atunci print j
```

vector<vector<int>> | Output degree of vertex : scan line of matrix in $O(n)$

$O(1)$ - store degree separately

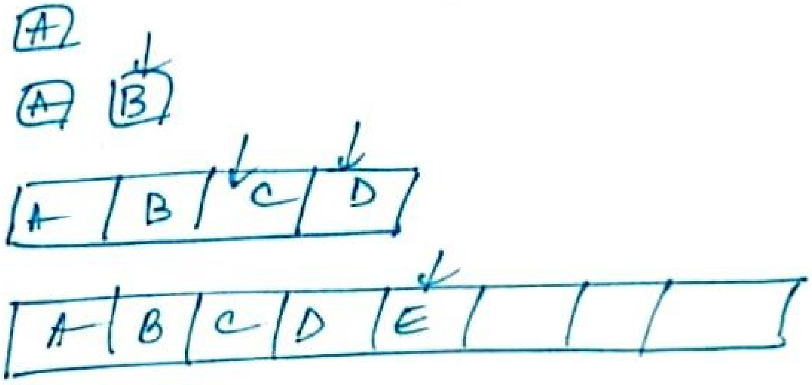
Additional / Removal of a vertex : Reallocate matrix in $O(n^2)$





Realocare:
dublăm
spațiul

$CA = O(1)$



Better: use doubling arrays

Removal of vertex: interschimbare linia i cu n

$M[i,j] \leftrightarrow M(n,j]$

$M[j,i] \leftrightarrow M[j,n]$

struct. date – pt. un id să știm valoarea

map (id, pos) = structură randomizată

↑ key ↑ val

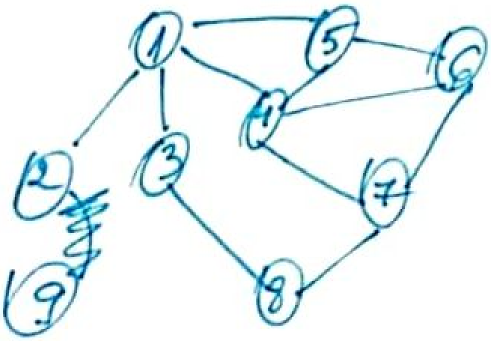
$O(1)$ CR

Nu putem sorta liste pt. că nu avem acces constant la elemente.

$$\sum_{v \in V} O(d(v) \log(d(v)))$$

$$= \sum_{v \in V} O(d(v) \log n$$

$$= O(m \log n)$$



1 : 2, 5, 4, 3

2 : 1 ~~9~~

3 : 8, 1

4 : 5, 7, 1, 6

5 : 1, 4, 6

6 : 5, 7, 4

7 : ~~6, 4, 8~~ 6, 8, 4

8 : 3, 7

9 :

Creez un graf în care vectorii sunt inițial vizi:

1 : 2, 3, 4, 5

2 : 1

3 : 1, 8

4 : 1, 5, 6, 7

5 : 1, 4, 6

6 : 4, 5, 7

7 : 4, 6, 8

8 : 3, 7

9 :

$O(n+m)$

Am parcurs listele de adiacență și am adăugat vârful curent la vectorii vecinilor

```
contracție (u, v)

  $N_u \leftarrow$ vecini (u)          $O(n)$
  $N_v \leftarrow$ vecini (v)          $O(n)$
  sterge (u)                  $O(n)$
  sterge (v)                  $O(n)$
  $\alpha \leftarrow$ insert Varf            $O(n)$  CA
  for $x \in N_u \cup N_v$
    insertMuchie $(\alpha, x)$          $O(1)$
```

Liste adiacență:

Enumerate all edges: $O(n+m)$

```
for i = 1..n
  for $j \in Adj(i)$   // vecinii lui i
    dacă $i<j$
      print (i, j)
```

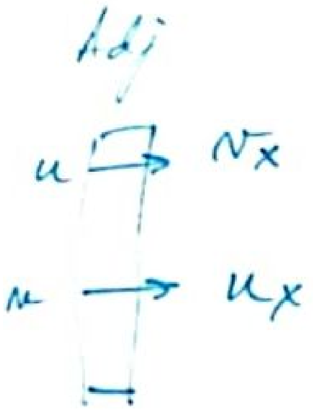
Dacă ne asigurăm că în lista de adiacență vârfurile izolate sunt la final $\Rightarrow O(m)$

Removal of an isolated squad: switch v with the last vertex, then remove last vertex in $O(1)$ amortized

$$d(u) + \sum_{v \in N(u)} d(v)$$

Test whether u, v are adjacent in $O(\log \min\{d(u), d(v)\})$ in static graphs
having vectors

Grafuri dinamice : hash table



Muchii : dict

key : muchii

valoare :

(u,v) → poz. u în lista lui v
poz. v în Adj(u)

când fac o inserare în liste, updatez dicționarul

dict
în care
cheile
sunt id-urile
valorile = listele

Putem avea un dict Adj

vârf v → dict ale cărui chei sunt vecinii
lui v

⇒ set

Examen : parcurgeri