Modalităţi de reprezentare a grafurilor

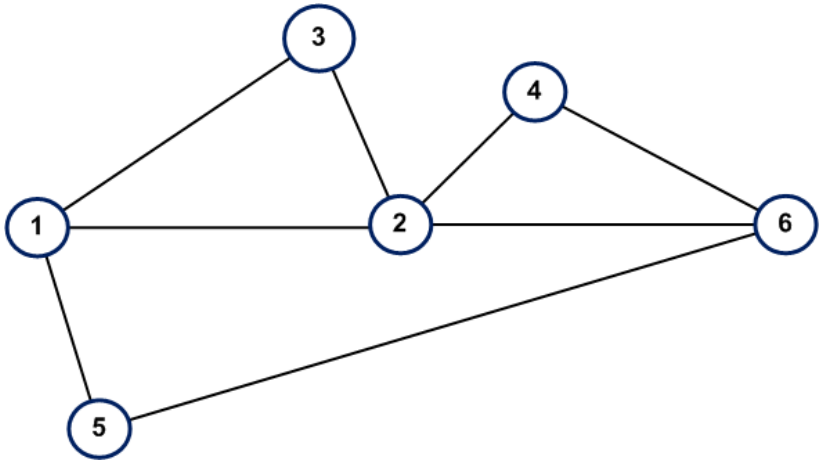
Reprezentarea grafurilor

- Matrice de adiacenţă
- Liste de adiacenţă
- Listă de muchii/arce

Reprezentarea grafurilor

Matrice de adiacenţă

	1	2	3	4	5	6
1	0	1	1	0	1	0
2	1	0	1	1	0	1
3	1	1	0	0	0	0
4	0	1	0	0	0	1
5	1	0	0	0	0	1
6	0	1	0	1	1	0



- este simetrică dacă graful este neorientat

Reprezentarea grafurilor

Lista de adiacenţă

6 noduri:

1: 2, 3, 5

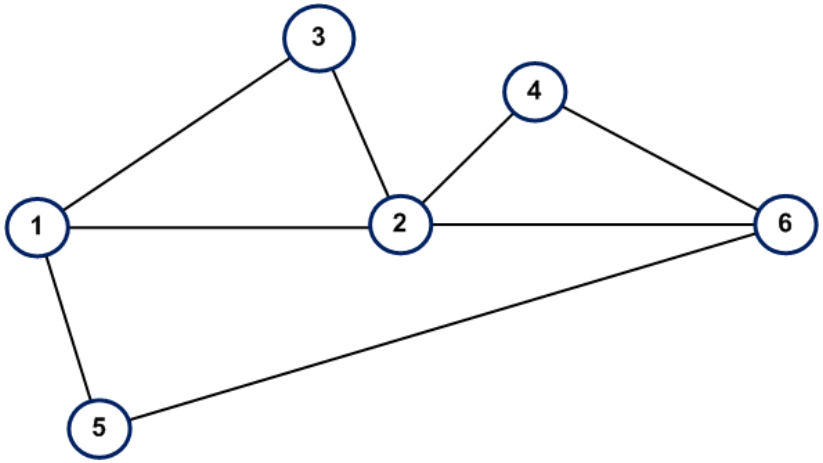
2: 1, 3, 4, 6

3: 1, 2

4: 2, 6

5: 1, 6

6: 2, 4, 5



Reprezentarea grafurilor

Lista de muchii

1 2

1 3

2 3

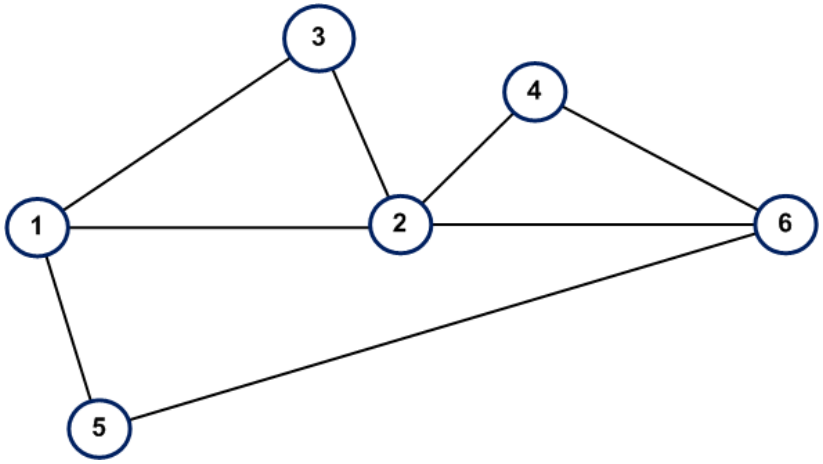
2 4

4 6

2 6

1 5

5 6



Reprezentarea grafurilor

- De ce avem mai multe tipuri de reprezentări?
- Care este "mai bună" ?

Reprezentarea grafurilor

- **Memorie consumată pentru graf cu n vârfuri și m muchii**
  - **Matrice de adiacență:** $O(n^2)$
  - **Lista de vecini**: $O(n+m)$
  - **Lista de muchii**: $O(m)$
- $m \leq n(n-1)/2$
  - Pentru un graf dens (aproape complet) matricea de adiacență nu mai necesită este "rea"

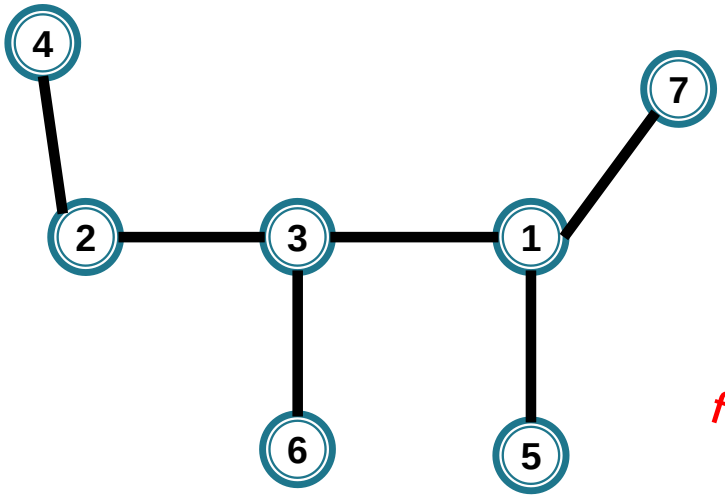
Reprezentarea grafurilor

- **Importantă pentru complexitatea timp a algoritmilor**
- **Alegem reprezentarea în funcție de operațiile din algoritm:**
  - parcurgerea vecinilor unui vârf
  - testarea adiacenței între două vârfuri
  - eliminare de muchii etc

Arbori cu rădăcină

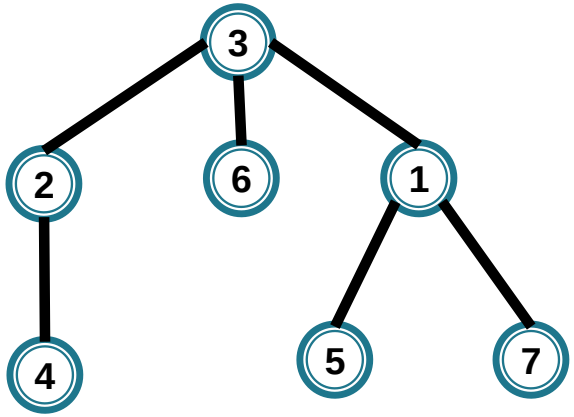
Arbore

arbore

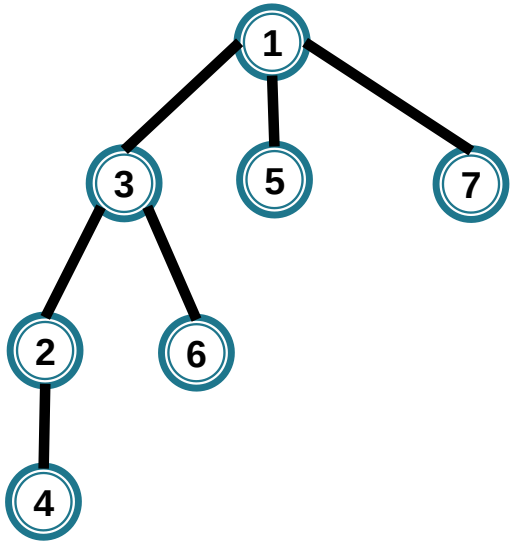


fixăm o rădăcină

arbori cu rădăcină



fixăm o altă rădăcină



Arbore cu rădăcină

- **Noțiuni**
  - După fixarea unei rădăcini, arborele se aşează pe niveluri
  - Nivelul unui nod v:

    `niv[v]` = distanţa de la rădăcină la nodul v
  - În arborele cu rădăcină există muchii doar între niveluri consecutive

Arbore cu rădăcină

- **Tată:** x este tată al lui y dacă există muchie de la x la y şi x se află în arbore pe un nivel cu 1 mai mic decât y

- **Fiu:** y este fiu al lui x $\Leftrightarrow$ x este tată al lui y

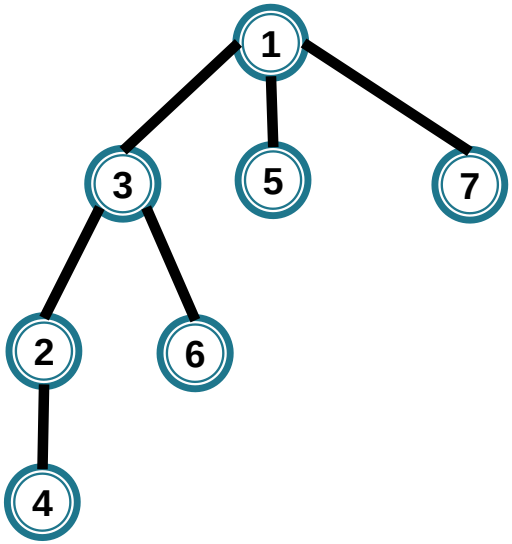
- **Ascendent (strămoş):** x este ascendent a lui y dacă x aparţine unicului lanţ elementar de la y la rădăcină (echivalent, dacă există un lanţ de la y la x care trece prin noduri situate pe niveluri din ce în ce mai mici)

- **Descendent (urmaş):** y este descendent al lui x $\Leftrightarrow$ x este ascendent a lui y

- **Frunză:** nod fără fii

Arbore cu rădăcină

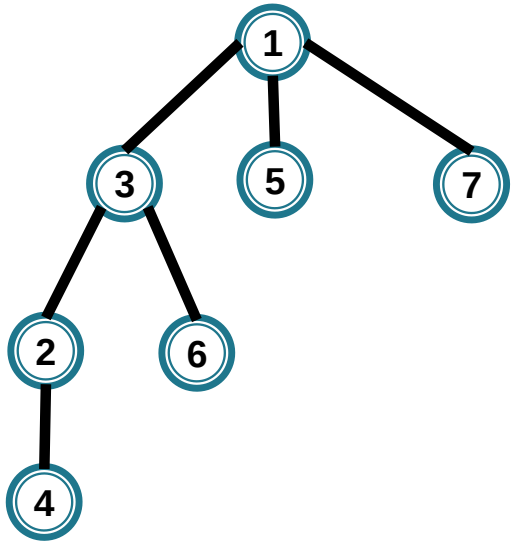
- **Fiu:** fii lui 3 sunt 2 şi 6
- **Tată:** 1 este tatăl lui 7
- **Ascendent:** ascendenţii lui 6 sunt 3 şi 1
- **Descendent:** descendenţii lui 3 sunt 2, 6 şi 4
- **Frunză:** frunzele arborelui sunt 4, 6, 5 şi 7



Modalităţi de reprezentare a arborilor cu rădăcină

Reprezentarea arborilor cu rădăcină

- Vector tata
- Lista de fii



Vectorul tata

Folosind vectorul tata putem determina **lanțuri de la orice vârf x la rădăcină**, **urcând** în arbore de la x la rădăcină

```
lant(x)
  cat timp x!=0 executa
     scrie x
     x = tata[x]
```

```
lantr(x)
  daca x!=0 atunci
        lantr(tata[x])
        scrie x
```

Parcurgerea grafurilor

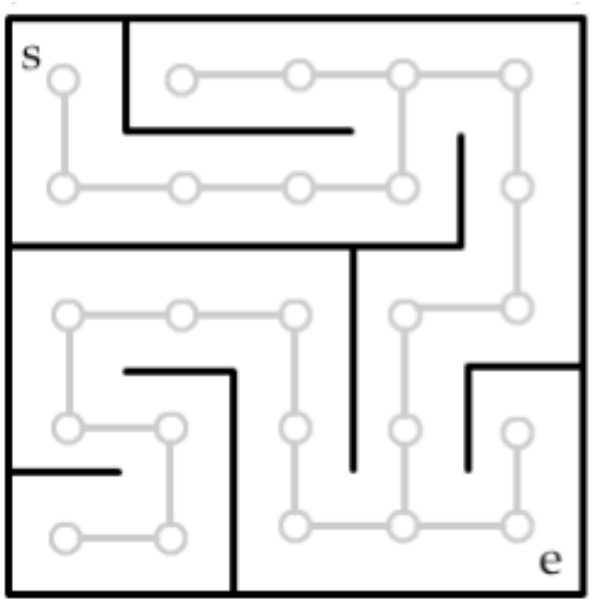
Parcurgerea grafurilor

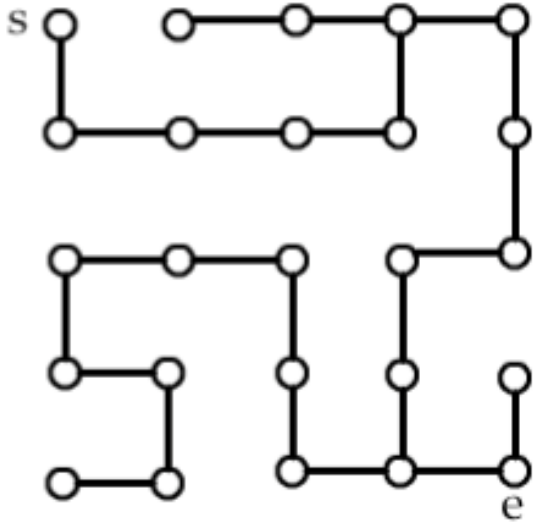
Parcurgere = o modalitate prin care, plecând de la un vârf de start s și mergând pe arce/muchii să ajungem la toate vârfurile accesibile din s

Exemple de aplicații

Drumuri în labirint

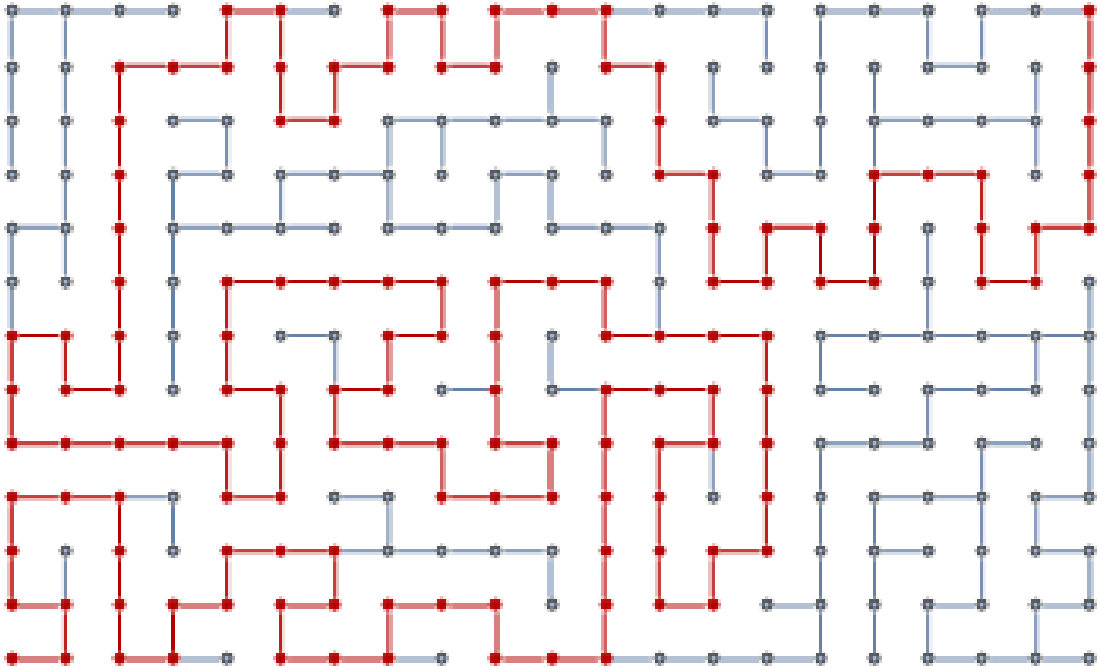
- Labirint – asociat un graf





http://www.cs.umd.edu/class/spring2019/cmsc132-020X-040X/Project8/proj8.html

Drumuri în labirint



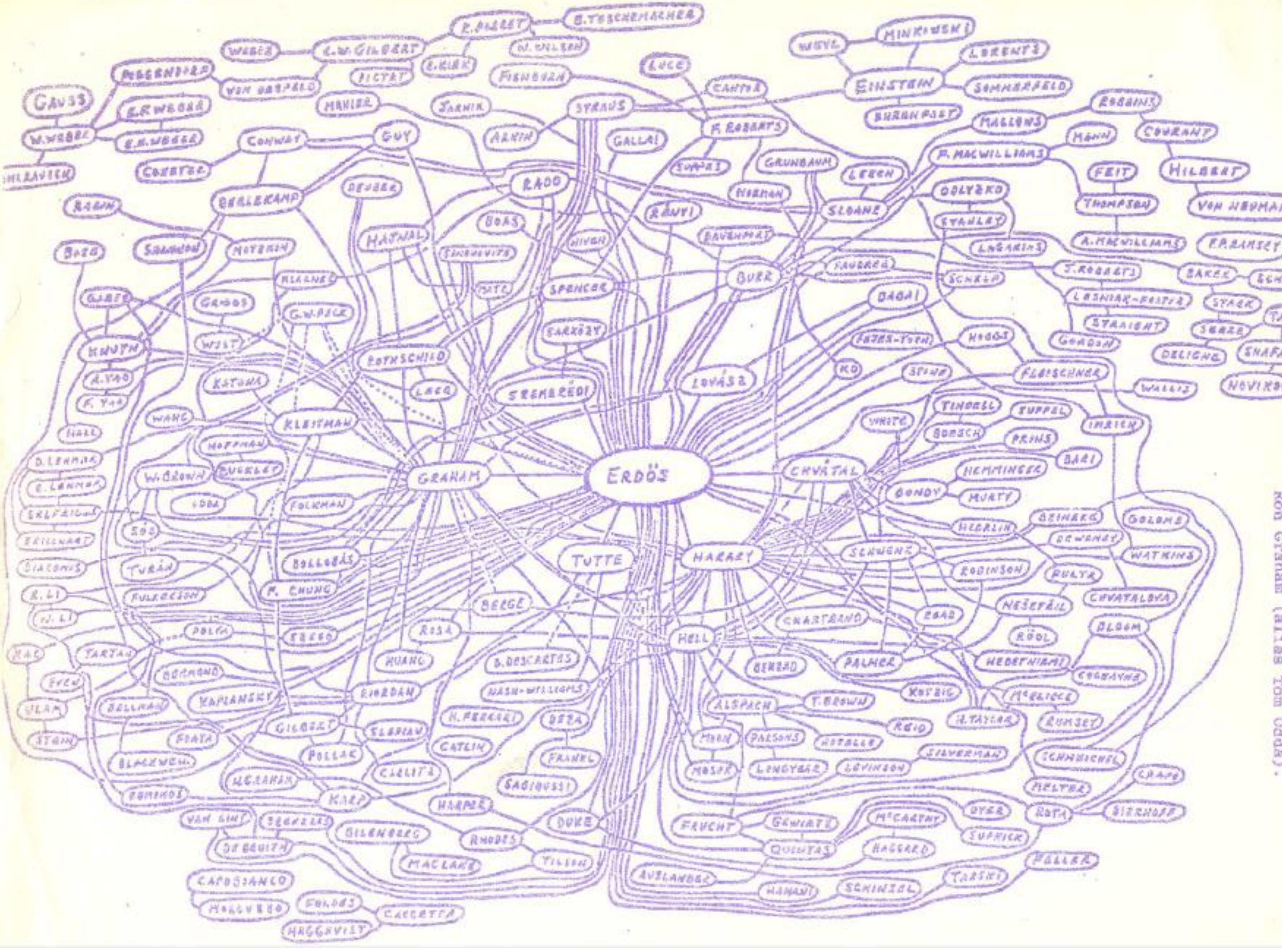
https://rosettacode.org/wiki/Maze_solving

Probleme de accesibilitate și distanțe

- Reteaua de colaborare cu Erdos
  - Muchii – a colaborat (publicat impreuna) cu
  - Noduri – matematicieni

- “Distanta” fata de Erdos? – Numărul Erdos

- Generalizare – distanța între 2 autori într-o rețea de colaborare

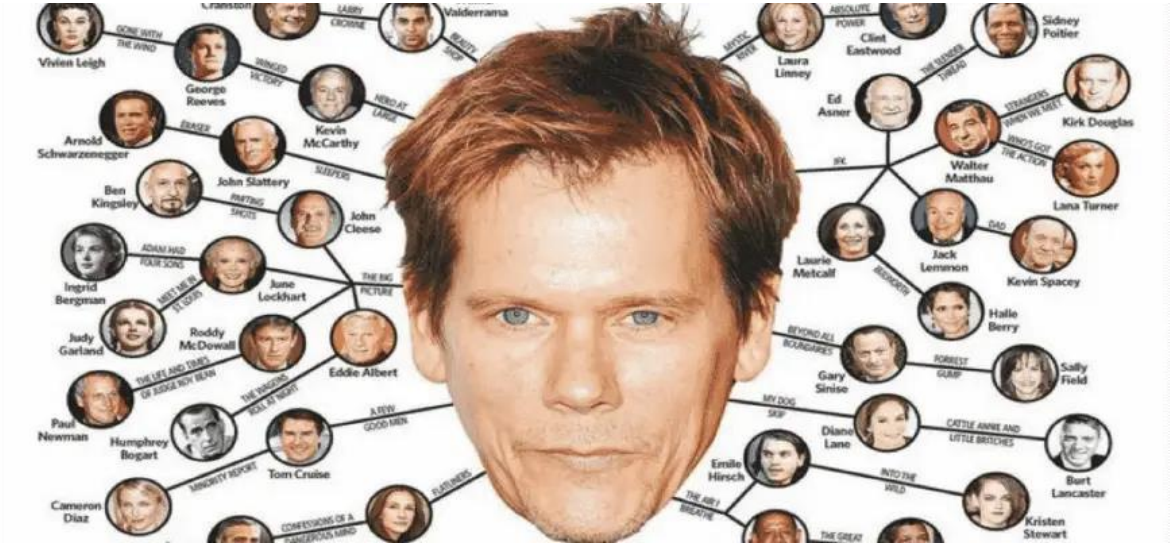


Ron Graham (alias Tom Odda).

Retea de colaborări, citări

Numarul Kavin Bacon

- Noduri - actori
- Muchii – au jucat impreuna
- Six Degrees of Kevin Bacon

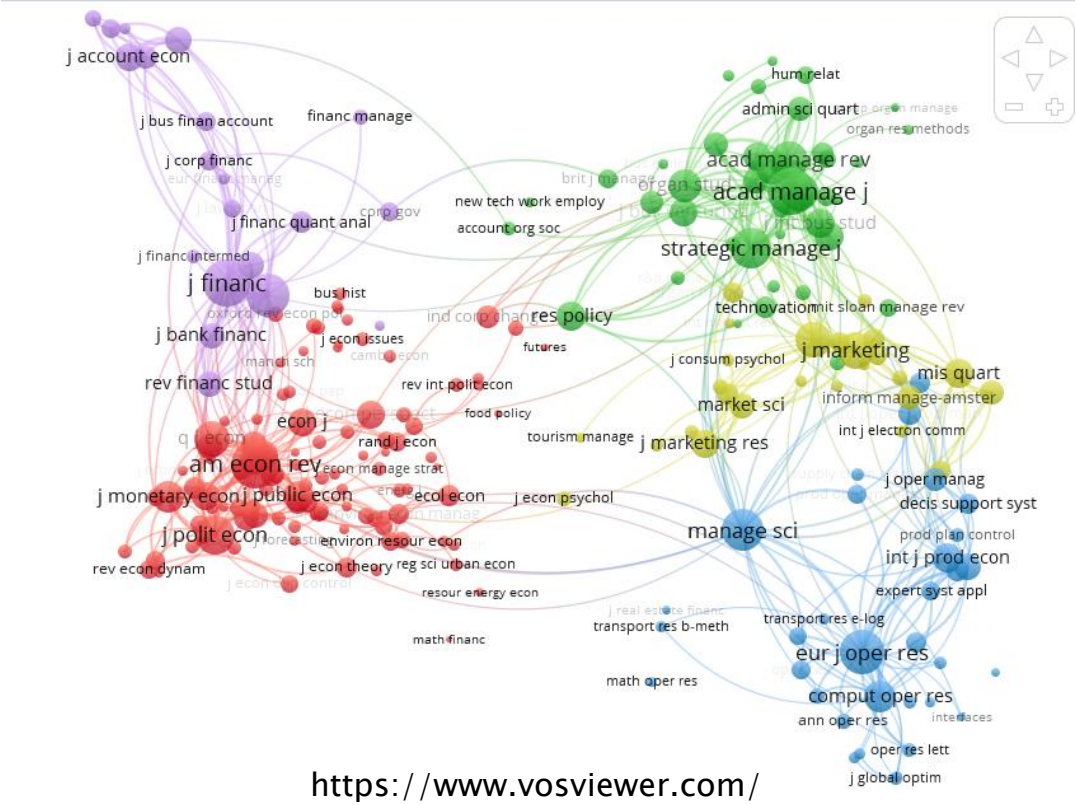


https://www.iceinstitute.org/blog/2019/4/1/six-degrees-of-kevin-bacon-education-edition

Retea de colaborări, citări

Probleme relevante

- Distanțe
- Componenta unui autor/jurnal/pagina web
- componenta dominantă (de dimensiune maximă, gigant): conexă (neorientat), tare conexă (orientat) etc

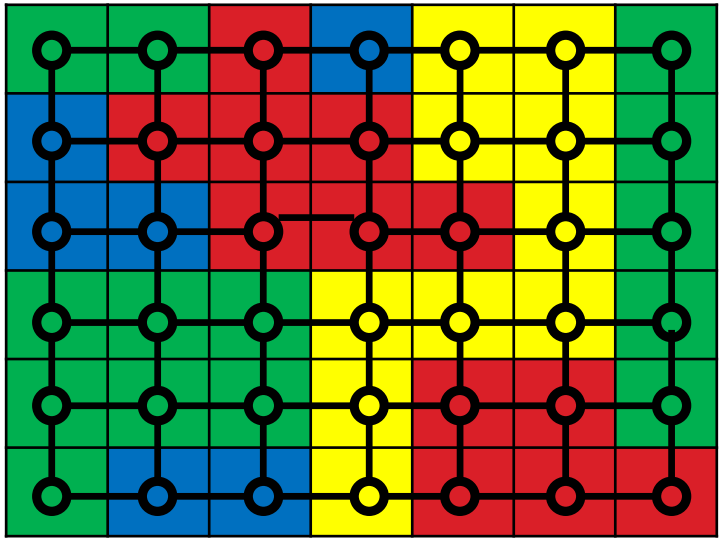


https://www.vosviewer.com/

Procesarea imaginilor

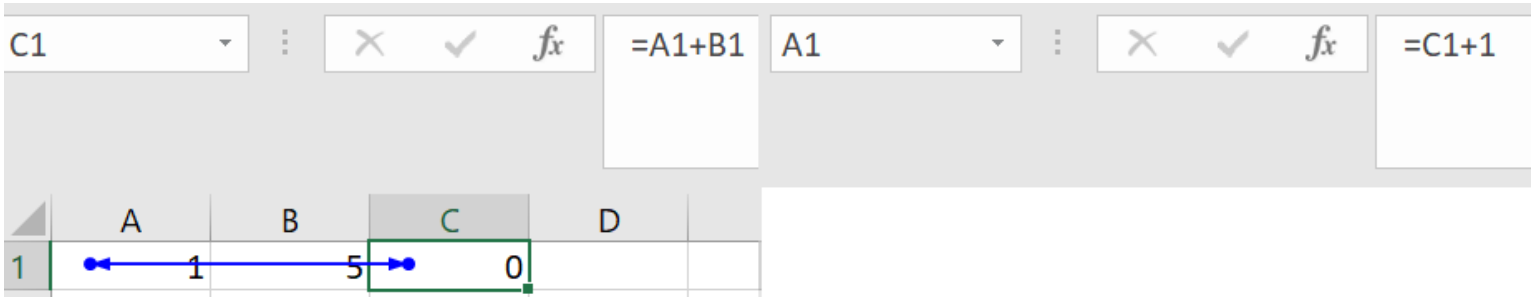
Algoritmi de umplere (fill)

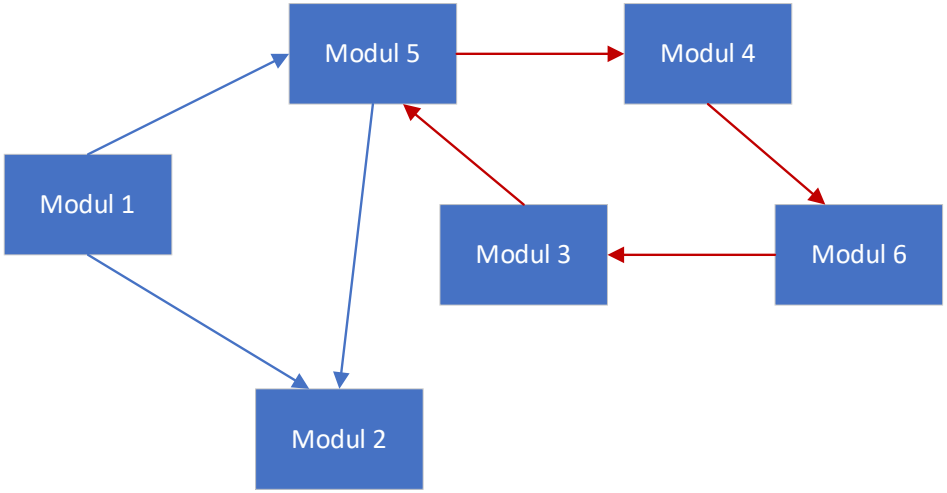
- Imagine – îi putem asocia graf grid
- Pornind de la un pixel de o anumita culoare c – determinăm componenta lui de culoare c (obiect)



Detectarea de dependențe circulare

- Existența de dependențe circulare în formule, module:





Tipuri de parcurgeri ale unui graf

Parcurgerea grafurilor

Dat un graf G și un vârf s, cum putem determina care sunt toate vârfurile accesibile din s?

- Un vârf v este **accesibil din s** dacă există un drum/lanț de la s la v în G.

Parcurgerea grafurilor

Idee: Pas cu pas

Dacă

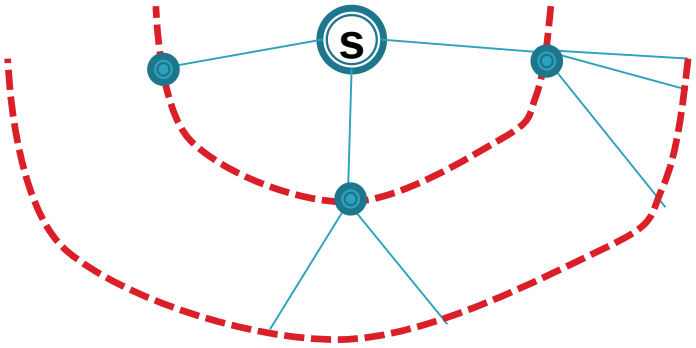
- u este **accesibil din s (vizitat)**
- $uv \in E(G)$

atunci v este accesibil din s.

Ce diferă? – ordinea în care considerăm vârfurile deja vizitate pentru a descoperi noi vârfuri accesibile

Parcurgerea grafurilor

- Parcurgerea în lățime (BF = breadth first)



- Parcurgerea în adâncime (DF = depth first)