



1

## Descrierea cursului

- Informații legate de curs

2

## Aplicații ale teoriei grafurilor

3

## Vocabularul teoriei grafurilor

- Definiția grafului

- Mulțimi stabile

- Cuplaje

- Colorări

- Izomorfism de grafuri

4

## Exerciții pentru seminarul 2 (săptămâna 6-10 octombrie)

C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms  
\* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph  
Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru -

### Pagina cursului:

<https://edu.info.uaic.ro/algorithmica-grafuri>

C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms  
\* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph

### Obiective:

Cursurile acoperă noțiunile de bază în Teoria algoritmică a grafurilor. Cunoștințele acumulate vor fi aplicate în proiectarea unor algoritmi eficienți pentru rezolvarea problemelor de optimizare combinatorială.

Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru -

**Cursuri:** E. F. Olariu (semianii A și E), C. Frășinaru (semianul B).

- Notele de curs vor fi postate ca fișiere .pdf înaintea fiecărui curs.
- Aceste fișiere vor conține și exercițiile pentru seminarii.

Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru  
- Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \*

C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms  
\* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph

**Seminarii: E. F. Olariu, C. Frăsinaru, P. Diac, S. Amihăesei și A. Ioniță.**

- Cursurile și seminariile vor fi organizate **în săli**.
- Fiecare seminar va conține un scurt test asupra cunoștințelor din cursurile și seminariile anterioare.
- În timpul seminariilor vor fi discutate soluții la exercițiile propuse (care se găsesc în fișierele care conțin cursurile).
- Fiecare seminar este dedicat unui număr de trei-patru exerciții (postate în avans în notele de curs) cu scopul de a aprofunda conceptele introduse la curs.
- Studenții sunt încurajați să propună soluții originale problemelor discutate.

Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru  
- Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \*

### Punctajele:

- **Seminar:** teste (max.  $2 \times 10 = 20$ p), activitatea la seminar<sup>a</sup> (max. 22p<sup>b</sup>) - max. 42 puncte.
- **Teme pentru acasă:** trei seturi de exerciții, 16p fiecare - max. 48 puncte. (O temă poate fi rezolvată de o echipă de cel mult patru studenți, fiecare fiind responsabil/lider pentru cel puțin una dintre problemele aferente.)
- **Examen final scris (obligatoriu):** șase exerciții/probleme (fiecare a câte 10p) +10p din oficiu - max. 70 puncte.

Pentru a susține teza scrisă în sesiune sunt necesare cel puțin 30 de puncte din cele maximum 90p din activitatea de seminar și temele pentru acasă.

Din maximum 160p limita de promovare este de 80 puncte.

---

<sup>a</sup>Două răspunsuri corecte și la obiect = 1 punct.

<sup>b</sup>Cu aproximație.



C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms  
\* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph

- **Seminar.** Testele se prezintă sub forma unor întrebări la care trebuie dat un răspuns nu prea lung, dar justificat.
- **Teme pentru acasă.** Soluțiile pot fi elaborate în echipe de până la patru studenți (fiecare fiind responsabil/lider pentru cel puțin una dintre problemele aferente temei). Soluțiile la aceste teme vor fi scrise în **LaTeX** sau Word (Write) și trimise electronic (atât sursa cât și .pdf-ul). Bonus 2 puncte pentru fiecare temă scrisă în **LaTeX**. **Aceste soluții vor fi verificate cu un instrument anti-plagiat iar studenții care copie vor fi penalizați (pentru fiecare problemă copiată punctajul va fi de  $-2$  puncte).**
- **Examen final scris (obligatoriu).** Studenții nu au voie să consulte materiale în timpul examenului.

Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru  
- Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \*

## Cuprinsul cursului:

- Vocabularul teoriei grafurilor.
- Probleme de drum: parcurgerea sistematică a grafurilor, drumuri de cost minim, conexiune.
- Arbori parțiali de cost minim: union-find, complexitate amortizată.
- Teoria cuplajelor.
- Fluxuri în rețele.
- Reduceri polinomiale între probleme de decizie pe grafuri.
- Abordări ale problemelor NP-dificile.
- Grafuri planare.
- Tree-decomposition.



- **Colorarea grafurilor:** colorarea hărților (a fețelor unui graf planar), planificarea cursurilor/seminariilor (problema orarului), planificarea unei sesiuni, alocarea frecvențelor radio mobile, alocarea regiștrilor de memorie.

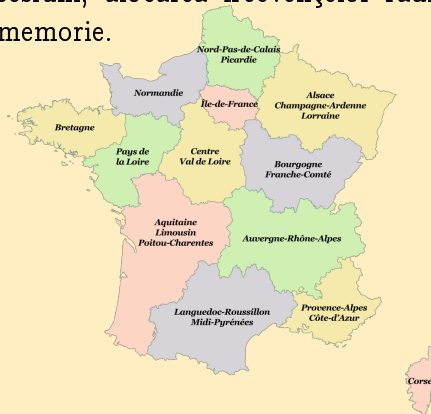


Figure: Regiunile Franței începând cu 2016

- **Cuplaje:** probleme de asignare, în studii de chimie computațională și de chimie matematică asupra materialelor organice.

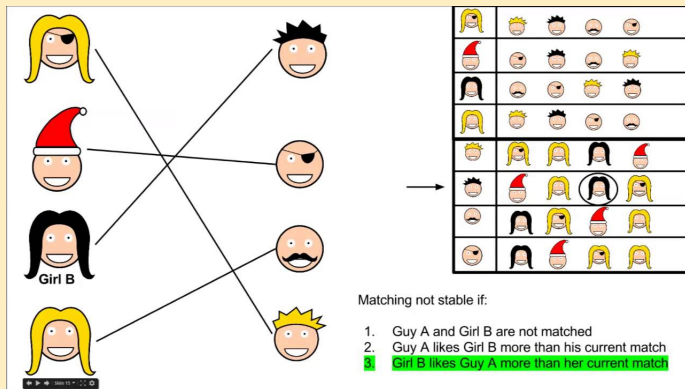


Figure: Gale Shapley algorithm



## Aplicații în informatică

- În managementul bazelor de date, **bazele de date de tip graf** folosesc structurile de date tip graf pentru stocare și interogare.
- **Graph rewriting systems** folosite în **verificarea sistemelor software**.
- **Quantum computation**.
- **Modelarea documentelor web drept grafuri** și clusterizarea acestora.
- Aproximarea și compresia datelor.
- Modelarea rețelelor de senzori cu grafuri (folosind de exemplu diagrame Voronoi).
- **Graph Neural Networks (IA)**.
- Și multe altele, cum ar fi **analiza rețelelor sociale** pentru a determina caracteristici cum ar fi:

## Aplicații ale teoriei grafurilor

C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms  
\* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph  
Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru -

- nivelul de **conexiune**, densitatea;
- influența utilizatorilor asupra rețelei (**centralitatea**, **potențialul rețelei sociale**);
- nivelul de **segmentare**: măsura clusterizării;
- robustețea sau stabilitatea structurală a rețelei.

Analiza rețelelor este folosită pentru

- data mining și agregare, marketing;
- analiza comportamentului și predicție în rețea;
- colectarea de informații și analiza securității (supravegherea terorismului și a crimei organizate) etc.

Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru  
- Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \*

# Aplicații ale teoriei grafurilor

G. Croitoru - Graph Algorithms \* G. Croitoru - Graph Algorithms \* G. Croitoru - Graph Algorithms

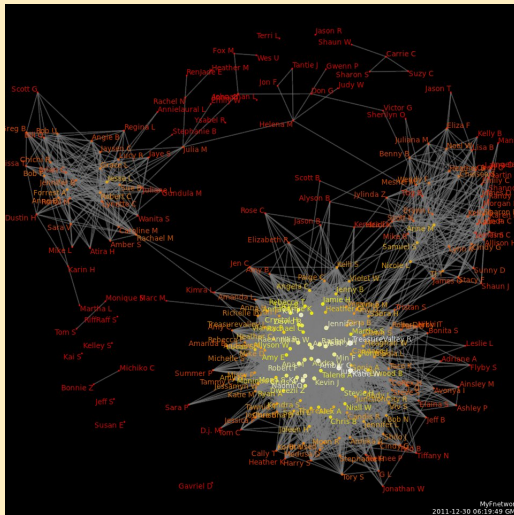


Figure: O rețea cu câțiva utilizatori Facebook

## Definiția grafului

C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms  
\* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph  
Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru -  
Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru

## Notații

Pentru o mulțime finită  $X$ :

- $|X| = \text{card}(X) \in \mathbb{N}$  este **cardinalul** lui  $X$ ;
- Dacă  $|X| = k$ , atunci  $X$  este o  **$k$ -mulțime**;
- $2^X = \mathcal{P}(X)$  este **mulțimea părților** lui  $X$ :  $2^X = \{Y : Y \subseteq X\}$ ,  
 $|2^X| = 2^{|X|}$ ;
- $\binom{X}{k} = \mathcal{P}_k(X) = \{Y : Y \subseteq X, |Y| = k\}$ ,  $\left| \binom{X}{k} \right| = \binom{|X|}{k}$ .

C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms  
\* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph  
Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru -  
Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru  
- Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \*



## Definiția grafului

C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms  
\* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph

### Definiția 2

Dacă  $G = (V, E)$  și  $e = uv = vu = \{u, v\} \in E$  este o muchie a lui  $G$ , spunem că

- $e$  **conectează** (sau **leagă**) nodurile  $u$  și  $v$ ;
- nodurile  $u$  și  $v$  sunt **adiacente** sau  $u$  și  $v$  sunt **vecine**;
- $e$  este **incidentă** cu  $u$  și  $v$ ;
- $u$  și  $v$  sunt **capetele** (**extremitățile**) lui  $e$ .

Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru

**Vecinătatea** nodului  $u$  este  $N_G(u) = \{v \in V(G) : uv \in E(G)\}$ .

C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms  
\* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph

Două muchii  $e$  și  $f$  sunt **adiacente** dacă au un capăt în comun:  $|e \cap f| = 1$ .

Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru  
- Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \*

## Definiția grafului

C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms  
\* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph

Fie  $G = (V, E)$  un graf și  $v \in V$ .

- **gradul** unui nod  $v$ :  $d_G(v) = |N_G(v)|$ .
- $v$  este un **nod izolat** dacă  $d_G(v) = 0$  și **pendant** (sau **frunză**) dacă  $d_G(v) = 1$ .
- o proprietate utilă:

$$\sum_{v \in V} d_G(v) = 2|E|.$$

Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru -

Un graf poate fi reprezentat în plan ca o figură constând dintr-o mulțime de puncte (forme geometrice mici: puncte, cercuri, pătrate etc) corespunzând vârfurilor sale și curbe care conectează vârfurile corespunzătoare muchiilor din graf.

- Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \*





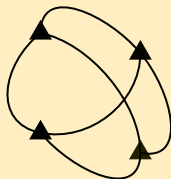
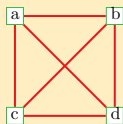
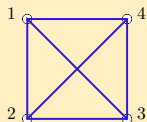
## Definiția grafului

C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms  
\* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph  
Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru -

Putem adăuga etichete (nume, numere etc) și culori nodurilor și muchiilor obținând reprezentări vizuale mai bune.

Mai jos sunt trei reprezentări vizuale ale aceluiași graf:

$$G = (\{1, 2, 3, 4\}, \{12, 13, 14, 23, 24, 34\})$$



Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru -  
Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru  
- Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \*

## Mulțimi stabile

C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms  
\* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph  
Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru -

### Definiția 3

Mulțime stabilă (sau mulțime independentă de noduri) în  $G = (V, E)$ :  
 $S \subseteq V$  astfel încât  $\binom{S}{2} \cap E = \emptyset$ .

- Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C.

Cu alte cuvinte  $S \subseteq V$  este o mulțime stabilă a lui  $G$  dacă nu există nicio muchie între nodurile sale.

Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru -  
Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru -

### Notăție

Cardinalul maxim al unei mulțimi stabile a lui  $G$  este numărul de stabilitate (sau numărul de independență) al lui  $G$  și se notează cu  $\alpha(G)$ .

Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru -  
Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru -  
Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \*



## Mulțimi stabile

C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms  
\* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph  
Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru -  
Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru -

### Mulțimi stabile: Problemă de optimizare

P1 Input:  $G$  graf.

Output:  $\alpha(G)$  și o mulțime stabilă  $S$  cu  $|S| = \alpha(G)$ .

- Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C.

### Mulțimi stabile: Problemă de decizie

SM Instanță:  $G$  graf,  $k \in \mathbb{N}$ .

Întrebare: Există o mulțime stabilă  $S$  în  $G$ , astfel încât  $|S| \geq k$ ?

### NP-completă (Karp, 1972).

Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru -  
Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru  
- Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \*





C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms  
\* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph  
Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru -

Cuplaj maxim: **Problemă de optimizare**

P2 Input:  $G$  graf.

Output:  $\nu(G)$  și un cuplaj  $M$  cu  $|M| = \nu(G)$ .

Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru -  
Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C.

**Edmonds (1965) a arătat că  $P2 \in P$ .**

\* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph

## Notă

Problemele  $P1$  și  $P2$  sunt similare: în amândouă se cere să se determine un membru de cardinal maxim al unei familii de mulțimi relativ la un graf dat. Ce face ca ele să fie diferite?

Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru -  
Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru  
- Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \*

## Definiția 5

Pentru  $p \in \mathbb{N}^*$ , o  **$p$ -colorare** a grafului  $G = (V, E)$  este o funcție  $c : V \rightarrow \{1, \dots, p\}$  astfel încât  $c(u) \neq c(v)$  pentru fiecare  $uv \in E$ .

Merită notat că mulțimea tuturor nodurilor cu aceeași culoare este o mulțime stabilă (se mai numește **clasă de colorare**). Deoarece noduri adiacente au culori diferite, o  $p$ -colorare corespunde unei partiții a lui  $V$  cu cel mult  $p$  mulțimi stabile (sau **clase de colorare**).

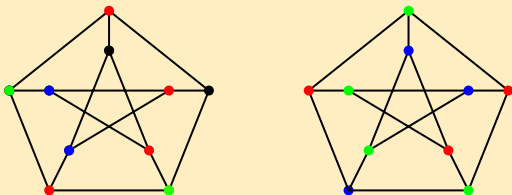
## Notăție

**Numărul cromatic** al grafului  $G$  este cea mai mică valoare a lui  $p$  astfel încât  $G$  are o  $p$ -colorare. Acest parametru se notează cu  $\chi(G)$ . ( $\chi(G) \leq |G|$  - de ce?)

C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms  
\* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph  
Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru -  
Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru  
- Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C.

## Exemplu

Pentru următorul graf avem marcate două colorări ( $\chi(G) = 3!$ )



Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \*  
C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms  
\* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph  
Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru -  
Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru  
- Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \*



## Definiția 6

Pentru  $p \in \mathbb{N}^*$ , a  **$p$ -muchie colorare** a grafului  $G = (V, E)$  este o funcție  $c : E \rightarrow \{1, \dots, p\}$  astfel încât  $c(e) \neq c(f)$  pentru orice  $e, f \in E$  cu  $|e \cap f| = 1$ .

Se poate observa că, dată o  $p$ -muchie colorare, o mulțime de muchii cu aceeași culoare este un cuplaj. Astfel, o  $p$ -muchie colorare corespunde unei partiții a lui  $E$  cu cel mult  $p$  cuplaje.

## Notăție

**Indexul cromatic** al grafului  $G$ : cea mai mică valoare a lui  $p$  astfel încât  $G$  are o  $p$ -muchie colorare. Acest parametru este notat cu  $\chi'(G)$ .  
( $\chi'(G) \leq |E(G)|$  - de ce?)





## Definiția 7

Două grafuri  $G_1 = (V_1, E_1)$  și  $G_2 = (V_2, E_2)$  sunt **izomorfe** dacă există o bijecție  $\varphi : V_1 \rightarrow V_2$  astfel încât pentru orice două noduri  $u_1, v_1 \in V_1$ ,  $u_1$  și  $v_1$  sunt adiacente în  $G_1$  (i. e.,  $u_1 v_1 \in E_1$ ) dacă și numai dacă  $\varphi(u_1)$  și  $\varphi(v_1)$  sunt adiacente în  $G_2$  (i. e.,  $\varphi(u_1)\varphi(v_1) \in E_2$ ).

Cu alte cuvinte, două grafuri sunt izomorfe dacă există o bijecție între mulțimile lor de noduri care induce o bijecție între mulțimile lor de muchii.

## Notație

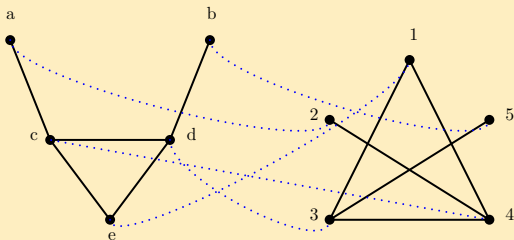
$$G_1 \cong G_2.$$

# Izomorfism de grafuri

C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms  
\* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph  
Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru -  
Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru  
Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru

## Exemplu

Grafurile de mai jos sunt izomorfe.



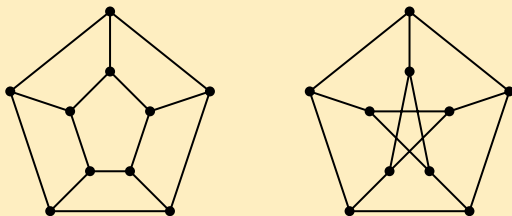
C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms  
\* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph  
Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru -  
Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru  
- Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \*



C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms  
\* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph  
Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru -  
Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru

## Exemplu

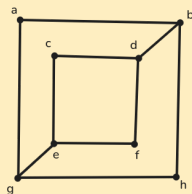
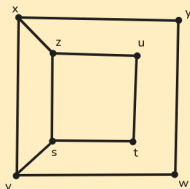
Următoarele două grafuri au aceleași ordine, dimensiuni și secvențe ale gradelor, dar nu sunt izomorfe (de ce?).



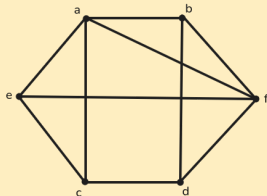
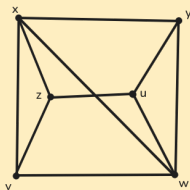
\* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph  
Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru -  
Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru  
- Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \*

## Exerciții pentru seminarul 2

Exercițiul 1. Grafurile de mai jos sunt izomorfe?



Exercițiul 2. Grafurile de mai jos sunt izomorfe?

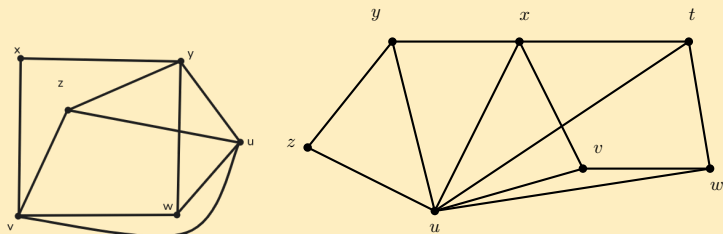




## Exerciții pentru seminarul 2

C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms

**Exercițiul 4.** Determinați numărul de stabilitate pentru grafurile de mai jos:



**Exercițiul 5.** Fie  $P(n) =$  "În orice graf cu cel puțin  $n$  noduri există trei noduri distincte care sunt două câte două adiacente sau două câte două neadiacente." Arătați că 6 este cea mai mică valoare a lui  $n \in \mathbb{N}$ ,  $n \geq 3$  pentru care  $P(n)$  este adevărată.

- Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \*



## Exerciții pentru seminarul 2

C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms  
\* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph  
Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph

**Exercițiul 7.** Doi studenți, **L(azy)** și **T(hinky)**, trebuie să găsească un drum particular între două noduri fixate într-un graf rar  $G$ :  $|E(G)| = O(|V(G)|)$ . **L** consideră că, deoarece graful este rar, numărul de drumuri dintre cele două noduri trebuie să fie mic, și o soluție lazy este să genereze (cu backtracking) toate aceste drumuri și apoi să rețină drumul dorit. **T** nu este de acord și dă următorul exemplu: fie  $H_n = K_2 \times P_n$  ( $n \geq 1$ ); adăugăm la  $H_n$  două noduri noi  $x$  și  $y$  fiecare unite prin câte două muchii cu câte una dintre cele două perechi de noduri adiacente de grad 2 din  $H_n$ .

Graful obținut,  $G_n$ , este rar dar numărul de drumuri dintre  $x$  și  $y$  este foarte mare. Explicați lui **L** acest exemplu desenând  $G_n$ , arătând că este rar și determinând numărul de drumuri dintre  $x$  și  $y$ .

Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru  
- Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \*

## Exerciții pentru seminarul 2

C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms  
\* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph  
Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru -  
Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru -

**Exercițiul 8.** O sesiune de examene trebuie planificată folosind următoarele specificații: mulțimea examenelor este cunoscută; fiecare student trimite o listă a examenelor la care dorește să fie examinat; fiecare examen are loc cu toți studenții înscriși la examen (care este scris); fiecare student poate participa la cel mult un examen într-o aceeași zi.

Construiți un graf cu ajutorul căruia să răspundeți la următoarele întrebări (prin determinarea unor parametri corespunzători):

- Care este numărul maxim de examene care pot fi organizate într-o aceeași zi?
- Care este numărul minim de zile necesare organizării unei sesiuni?

Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru -  
Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru  
- Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \*

### Exercițiul 8'. (exercițiul 8 - continuare)

Un programator isteț și îndemânic se întreabă dacă cele două probleme NP-hard din exercițiul anterior nu ar putea fi rezolvate în timp polinomial deoarece graful construit pare să aparțină unei clase speciale de grafuri.

- (a) Arătați că pentru orice graf dat,  $G$ , există un input pentru problema de planificare anterioară astfel încât graful construit (ca mai sus) să fie tocmai  $G$ .

Programatorul sugerează următoarea “abordare greedy” pentru a răspunde la a doua întrebare din exercițiul 8: începând cu prima zi, se planifică zilnic un număr maxim de examene (din mulțimea examenelor neplanificate încă), până când toate examenele vor fi planificate.

- b) Arătați că această abordare greedy este greșită printr-un contraexemplu.

## Exerciții pentru seminarul 2

C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms  
\* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph

**Exercițiul 9.** Un exemplu de optimizare a unui compilator este **tehnica alocării regiștrilor** de memorie: cele mai folosite variabile sunt ținute în registrii cu acces rapid ai procesorului pentru a le avea la îndemână în momentul când compilatorul are nevoie de ele (pentru anumite operații CPU).

Construiți un graf cu ajutorul căruia să răspundeți la următoarele întrebări (prin determinarea unor parametri corespunzători):

- (a) Care este numărul maxim de variabile de care nu este nevoie în același timp?
- (b) Care este numărul minim de regiștri necesari acestor variabile?

Indicație: avem două tipuri de obiecte: **variabilele** (cu valorile lor) și **operațiile CPU** (sau operatorii) care folosesc una sau mai multe variabile.

- Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \*

**Exercițiul 9'. (exercițiul 9 - continuare)** Un student se întreabă dacă la întrebările de mai sus (care corespund unor probleme NP-hard) nu s-ar putea da un răspuns în timp polinomial deoarece graful construit pare să aparțină unei clase speciale de grafuri.

- (a) Arătați că pentru orice graf dat,  $G$ , există un input pentru problema de alocare a regiștrilor astfel încât graful construit (ca mai sus) să fie tocmai  $G$ .

Studentul sugerează următoarea “abordare greedy” pentru a răspunde la a doua întrebare din exercițiul 9: începând cu primul registru, se alocă fiecărui registru nefolosit un număr maxim de variabile (dintre cele nealocate încă), până când toate variabilele vor fi alocate.

- b) Arătați că această abordare greedy este greșită printr-un contraexemplu.

## Exerciții pentru seminarul 2

C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms  
\* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph  
Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru -  
Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru  
Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C.

**Exercițiul 10.**  $G$  este numit graf *autocomplementar* dacă  $G$  și complementul său,  $\overline{G}$ , sunt izomorfe ( $G \cong \overline{G}$ ).

- (a) Arătați că un graf *autocomplementar* este conex și că  $|G| \equiv 0$  sau  $1 \pmod{4}$ .
- (b) Determinați toate grafurile *autocomplementare* cu cel mult 7 noduri.
- (c) Arătați că pentru orice graf  $G$  există un graf *autocomplementar*  $H$  astfel încât  $G$  este subgraf indus al lui  $H$ .

C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms  
\* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph  
Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru -  
Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru  
- Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \*