Colorări în grafuri

Colorări ale grafurilor

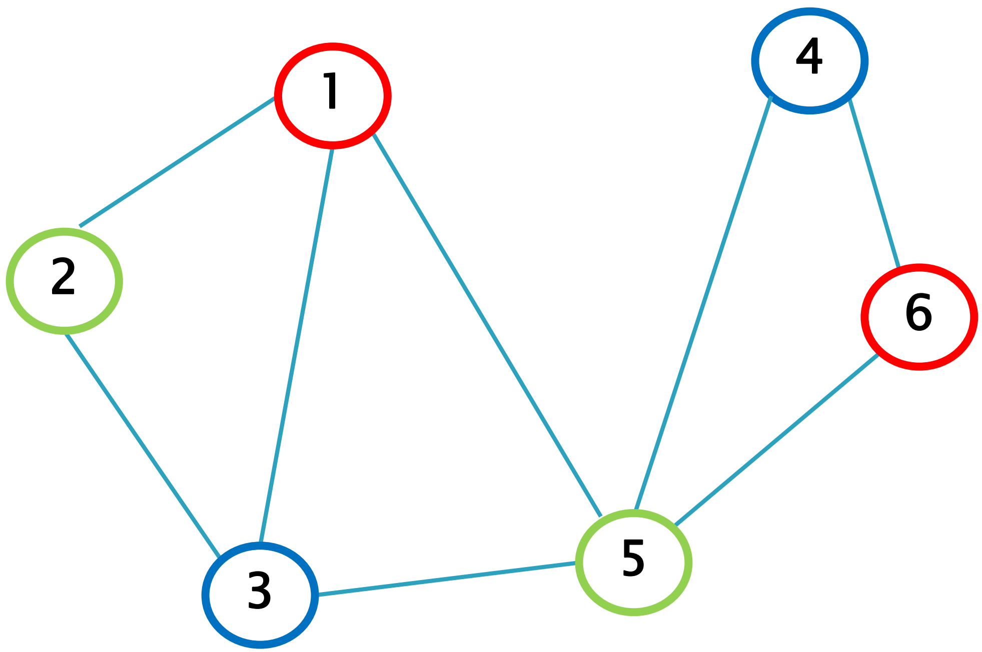
- $G = (V, E)$ graf neorientat
  - $c : V \rightarrow \{1, 2, \ldots, p\}$ s.n **p-colorare** a lui G
  - $c : V \rightarrow \{1, 2, \ldots, p\}$ cu $c(x) \neq c(y)$ $\forall xy \in E$ s.n **p-colorare proprie** a lui G
  - G s.n **p-colorabil** dacă admite o p-colorare proprie

Colorări ale grafurilor

- $G = (V, E)$ graf neorientat
  - Valoarea p minimă pentru care G este p-colorabil se numește **numărul cromatic** al lui G (notată $\chi(G)$)
  - Dacă G nu este conex

$$\chi(G) = \max\{\chi(H) \mid H \text{ componentă conexă în } G\}$$

Colorări ale grafurilor

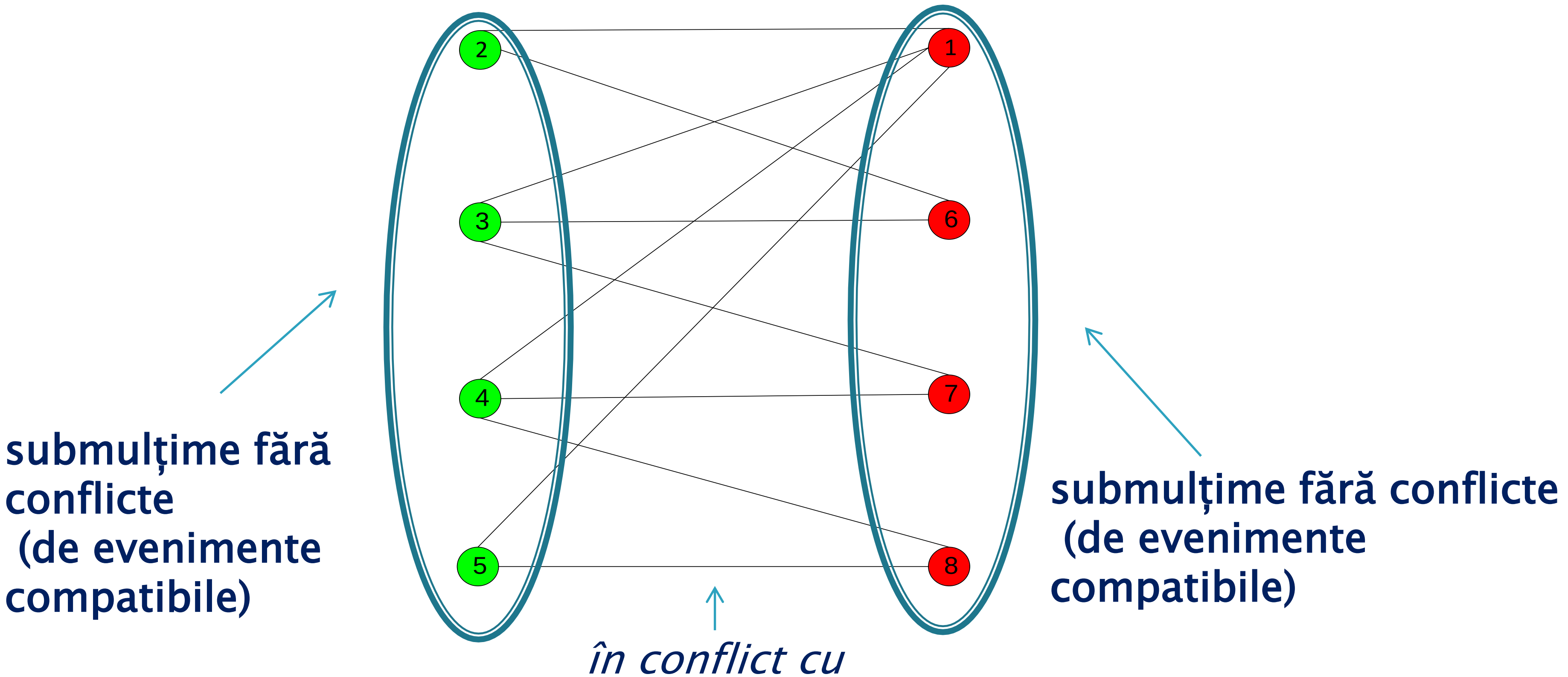


3-colorabil, dar nu și 2-colorabil (!)

=> are numărul cromatic 3

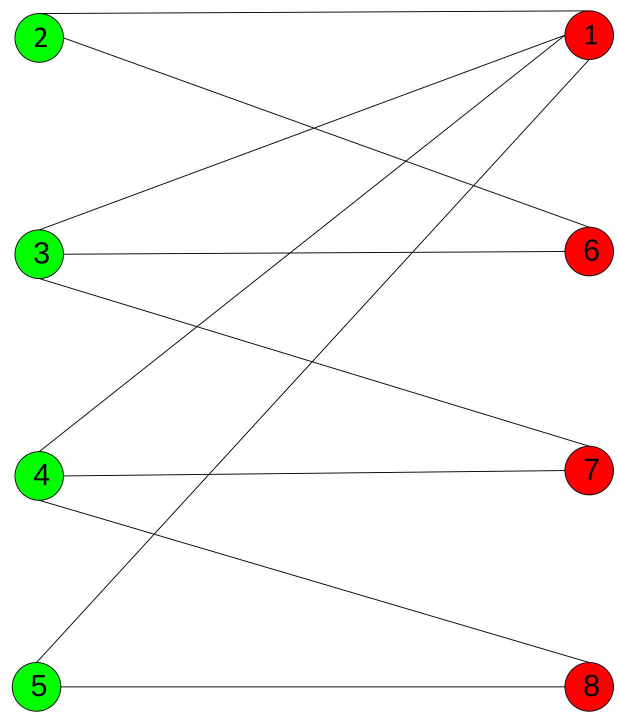
Aplicații p -colorări

Graf de conflicte (exemplu substanțe care interacționează, activități incompatibile, relații în rețele sociale)



- Cuplaje, rețele…

Aplicații p -colorări



Profesori *predau la* **Cursuri**

Candidați *depun CV la* **Joburi**

Aplicații p -colorări

De câte săli este nevoie minim pentru programarea într-o zi a n conferințe cu intervale de desfășurare date?

Conf. 1: interval (1,4)

Conf. 2: interval (2,3)

Conf. 3: interval (2,5)

Conf. 4: interval (6,8)

Conf. 5: interval (3,8)

Conf. 6: interval (6,7)

Aplicații p -colorări

De câte săli este nevoie minim pentru programarea într-o zi a n conferințe cu intervale de desfășurare date?

Conf. 1: interval (1,4)

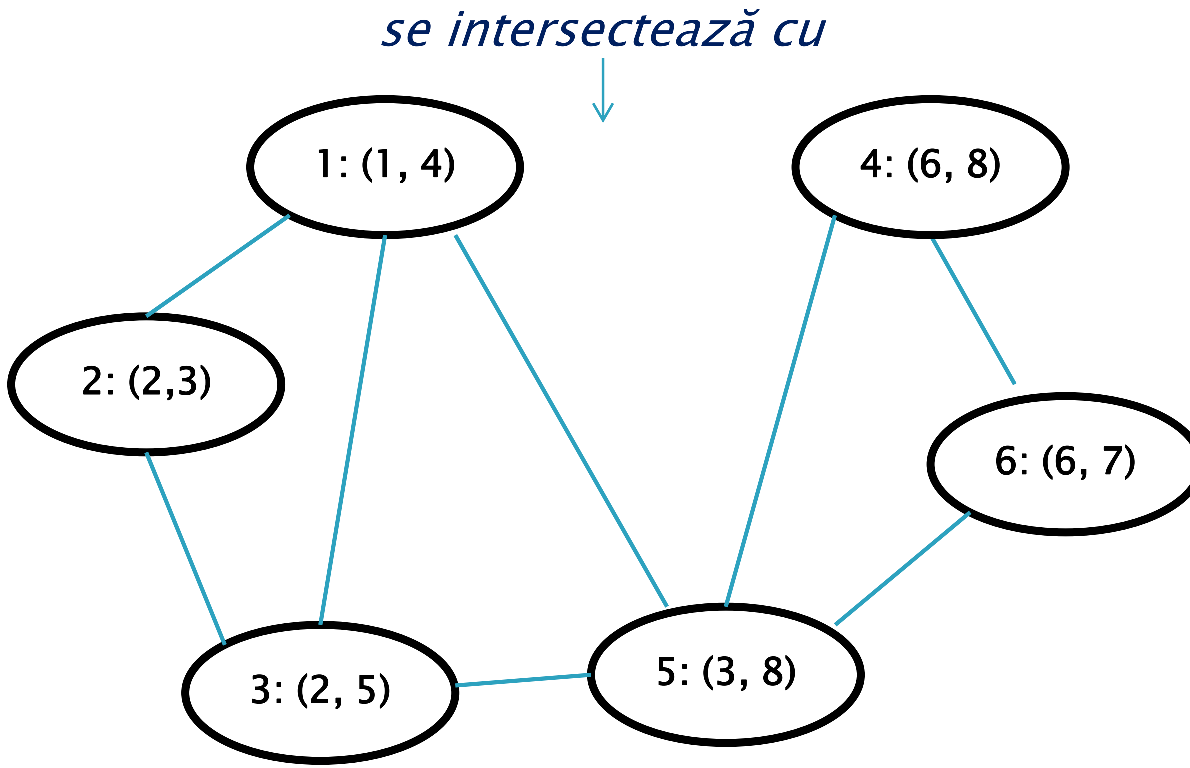
Conf. 2: interval (2,3)

Conf. 3: interval (2,5)

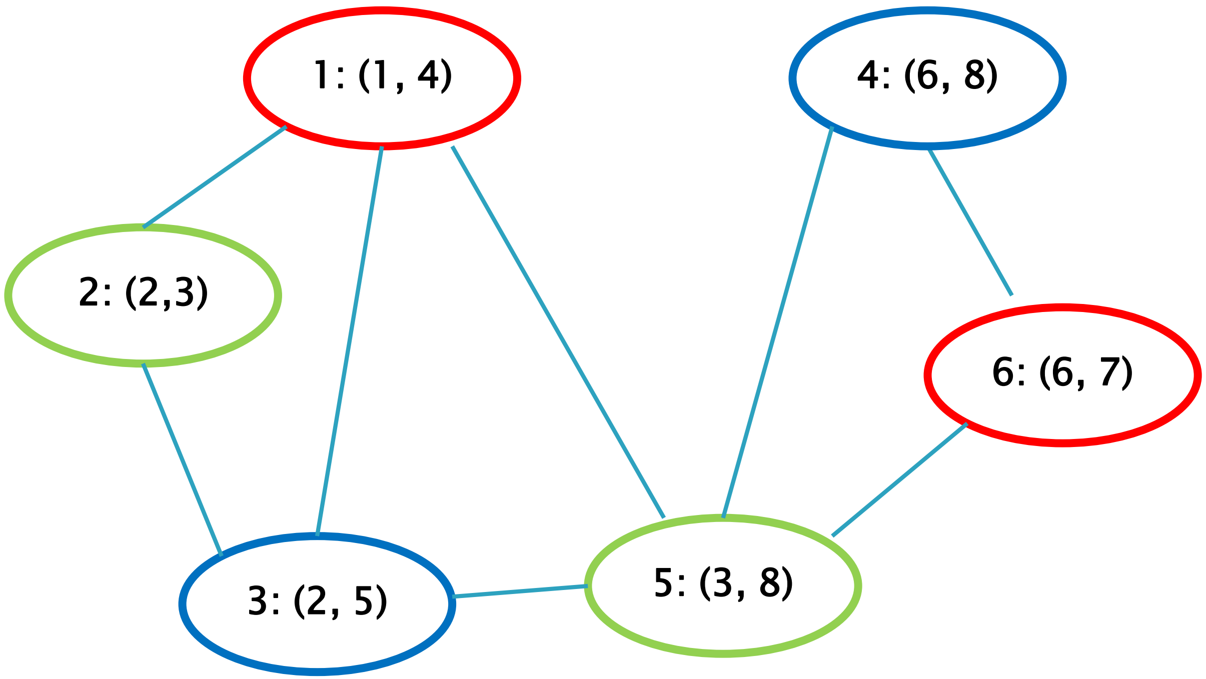
Conf. 4: interval (6,8)

Conf. 5: interval (3,8)

Conf. 6: interval (6,7)



Graful intersecției intervalelor este 3-colorabil:



Sunt necesare minim 3 săli (corespunzătoare celor 3 culori):

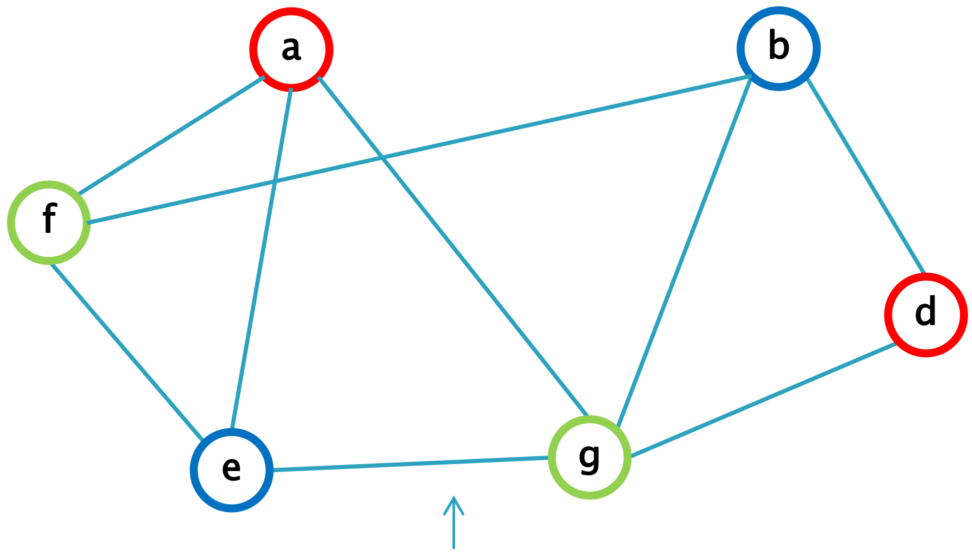
Sala 1: (1,4), (6,7)

Sala 2: (2,3), (3,8)

Sala 3: (2,5), (6,8)

Aplicații p -colorări

Alocare de regiștrii (Register allocation problem)



pot fi simultan active
(nu pot fi memorate în același registru)

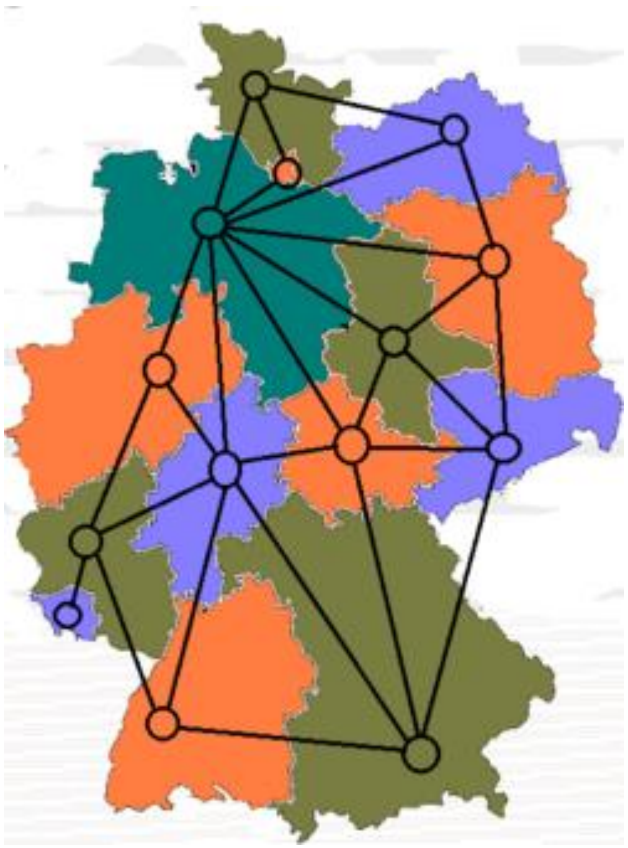
- Numărul de culori = numărul de regiștri
- Vârfuri de aceeași culoare = pot fi memorate în același registru

Aplicații p -colorări

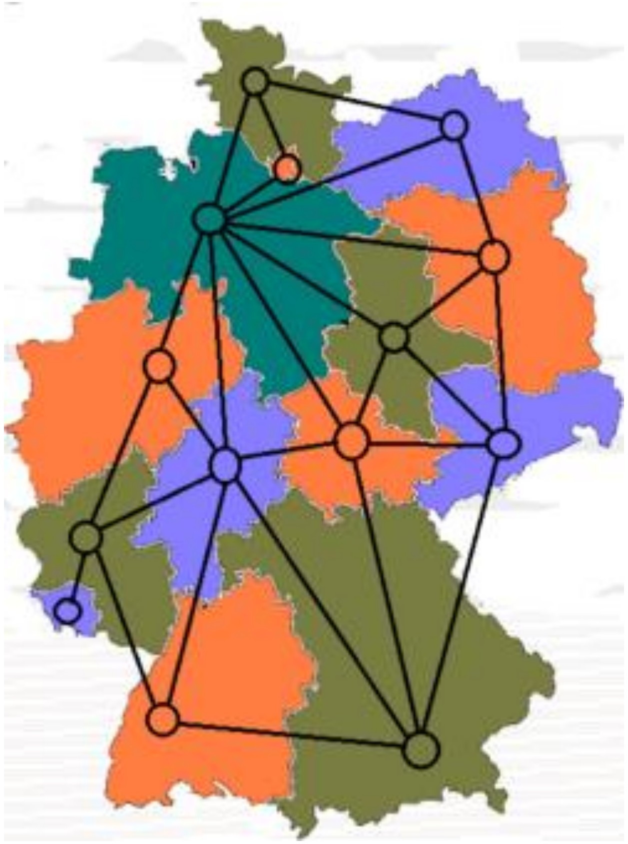
- **Problema colorării hărților**

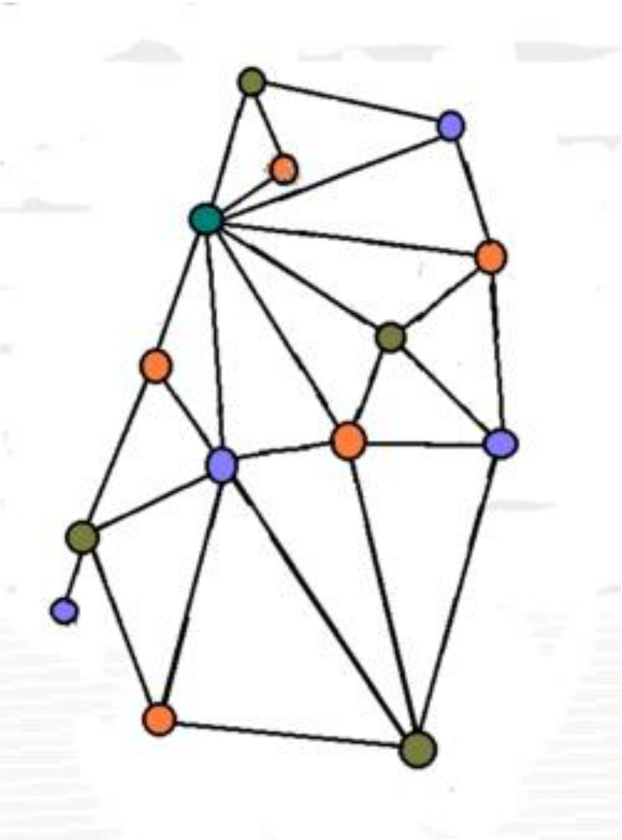
Se poate colora o hartă cu 4 culori astfel încât orice două țări, care au frontieră comună și care **nu se reduce la un punct**, să aibă culori diferite?

http://en.citizendium.org/wiki/Four_color_theorem



http://en.citizendium.org/wiki/Four_color_theorem





http://en.citizendium.org/wiki/Four_color_theorem

Colorări ale grafurilor

Computațional: Dat p, **este G p-colorabil**?

Care este p minim cu proprietatea că G este p-colorabil? = Care este numărul cromatic al lui G?

- Test graf 2-**colorabil** / graf bipartit – algoritm polinomial
- Test graf 3**-colorabil** – problemă NP-completă

Algoritmi polinomiali pentru colorarea cu 5 culori a unui graf planar

Colorări ale grafurilor

Subiecte tratate:

- Grafuri bipartite
- Colorări în grafuri planare
- Algoritmi de colorare de tip greedy (*neoptimali*)

Grafuri bipartite

Graf bipartit

Observații

- $G = (V, E)$ **bipartit** $\Leftrightarrow$

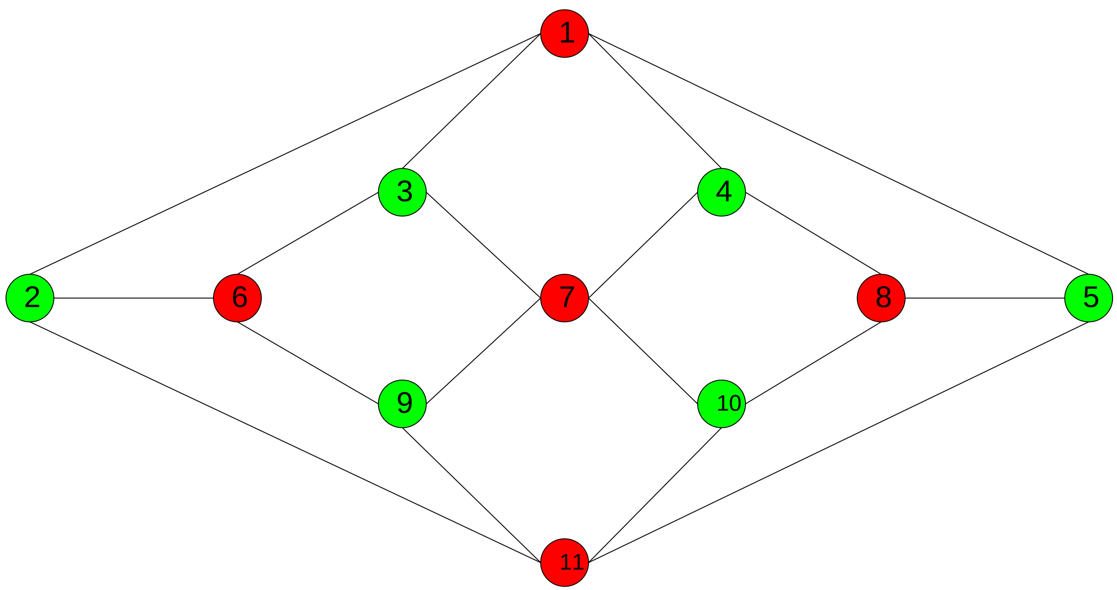
  există o 2-colorare proprie a vârfurilor (**bicolorare**):

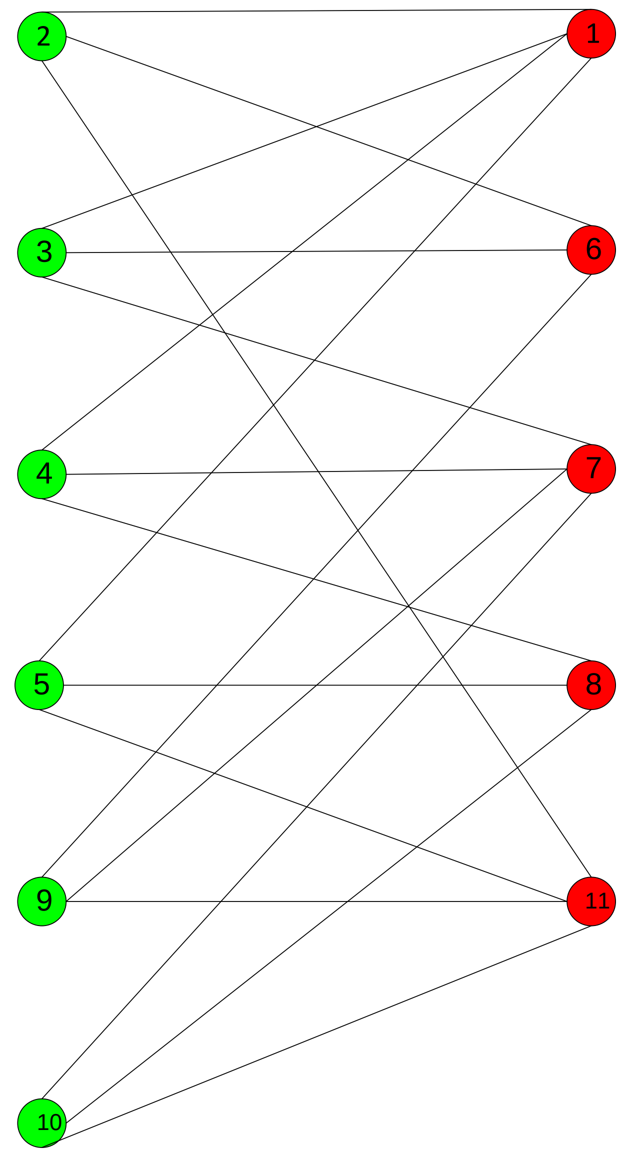
  $$c : V \rightarrow \{1, 2\}$$

  (i.e. astfel încât pentru orice muchie $e=xy\in E$ avem $c(x) \neq c(y)$)

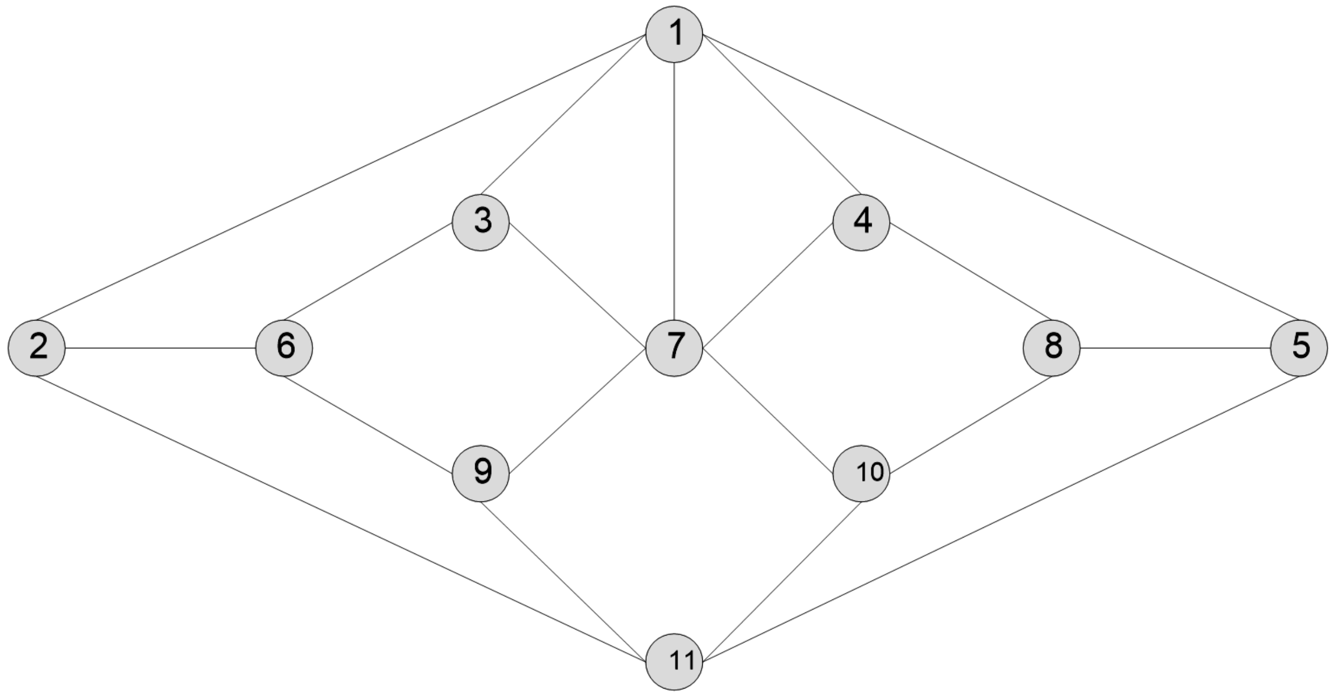
- $G = (V, E)$ bipartit $\Rightarrow \chi(G) \leq 2$

Graf bipartit





Graf bipartit



nu este bipartit

Graf bipartit

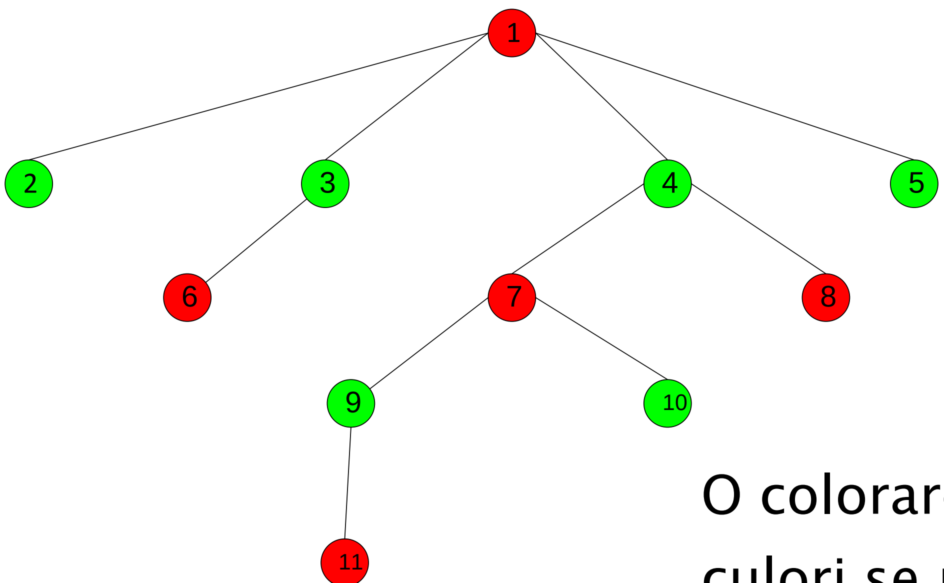
- **Propoziție**

  Un arbore este graf bipartit

Graf bipartit

Propoziție

Un arbore este graf bipartit



O colorare proprie a lui T cu cel mult 2 culori se poate obține astfel:

- fixăm o rădacină
- colorăm vârfurile de pe niveluri pare cu 1 și pe cele de pe niveluri impare cu 2.

Graf bipartit

- **Teorema König – Caracterizarea grafurilor bipartite**

Fie $G = (V, E)$ un graf cu $n \geq 2$ vârfuri.

Avem

$$G \text{ este bipartit} \Leftrightarrow \text{toate ciclurile elementare din } G \text{ sunt pare}$$

Graf bipartit

- **Teorema König – Caracterizarea grafurilor bipartite**

Demonstrație $\Rightarrow$ Evident, deoarece un ciclu impar nu poate fi colorat propriu cu două culori.

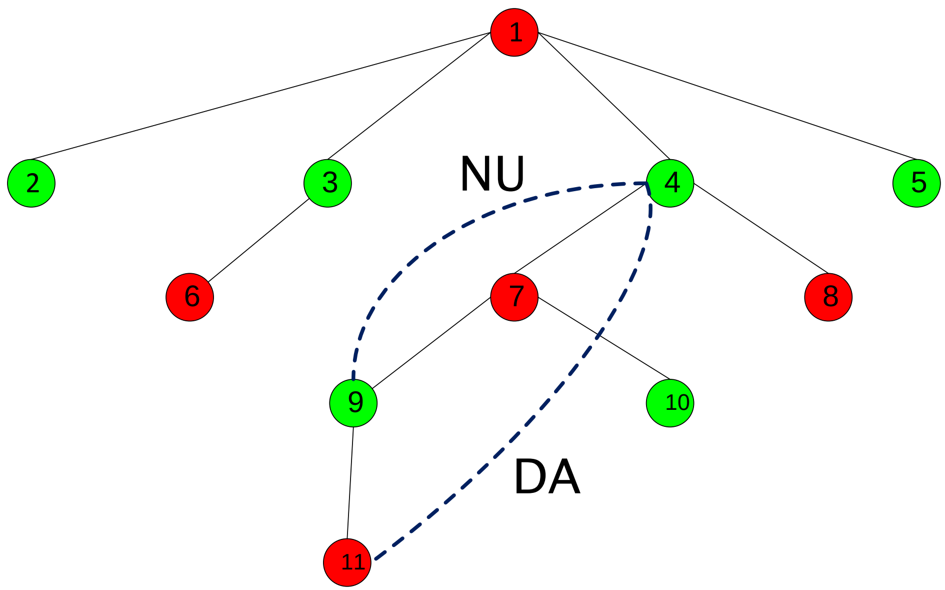
Graf bipartit

- **Teorema König – Caracterizarea grafurilor bipartite**

Demonstrație – $\Leftarrow$ Presupunem G conex.

Colorăm un arbore parțial T al său (de exp arborele DFS de rădăcină 1) ca în propoziția precedentă (alternativ pe niveluri).

Orice altă muchie uv din graf are extremitățile colorate diferit deoarece



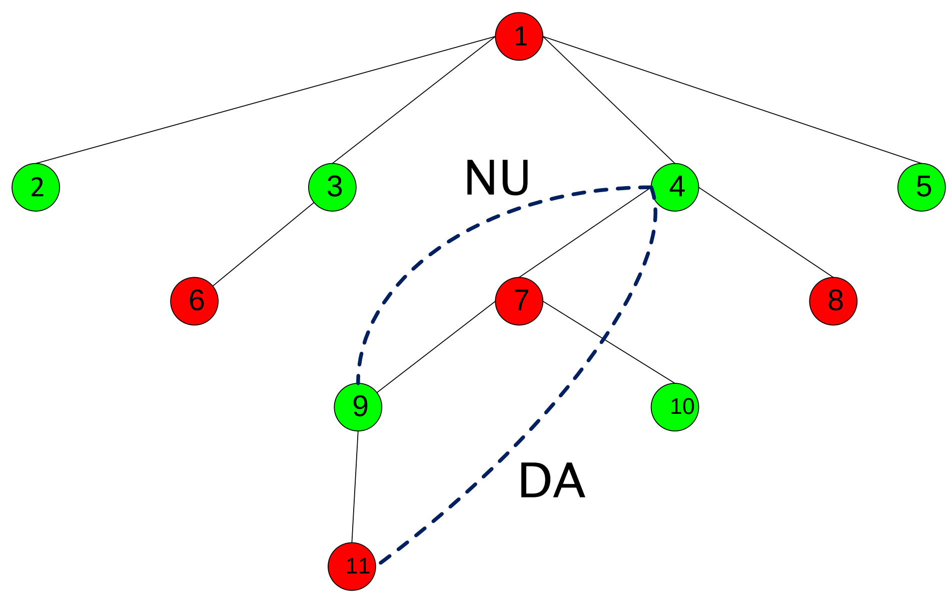
DR Popescu – Combinatorică și Teoria grafurilor (Teorema 4.18)

Graf bipartit

- **Teorema König – Caracterizarea grafurilor bipartite**

Demonstrație – ⇐ Presupunem G conex.

Colorăm un arbore parțial T al său (de exp arborele DFS de rădăcină 1) ca în propoziția precedentă (alternativ pe niveluri). Orice altă muchie uv din graf are extremitățile colorate diferit deoarece formează un ciclu elementar cu lanțul de la u la v din arbore și acest ciclul are lungime pară, deci u și v se află pe niveluri de paritate diferită în T



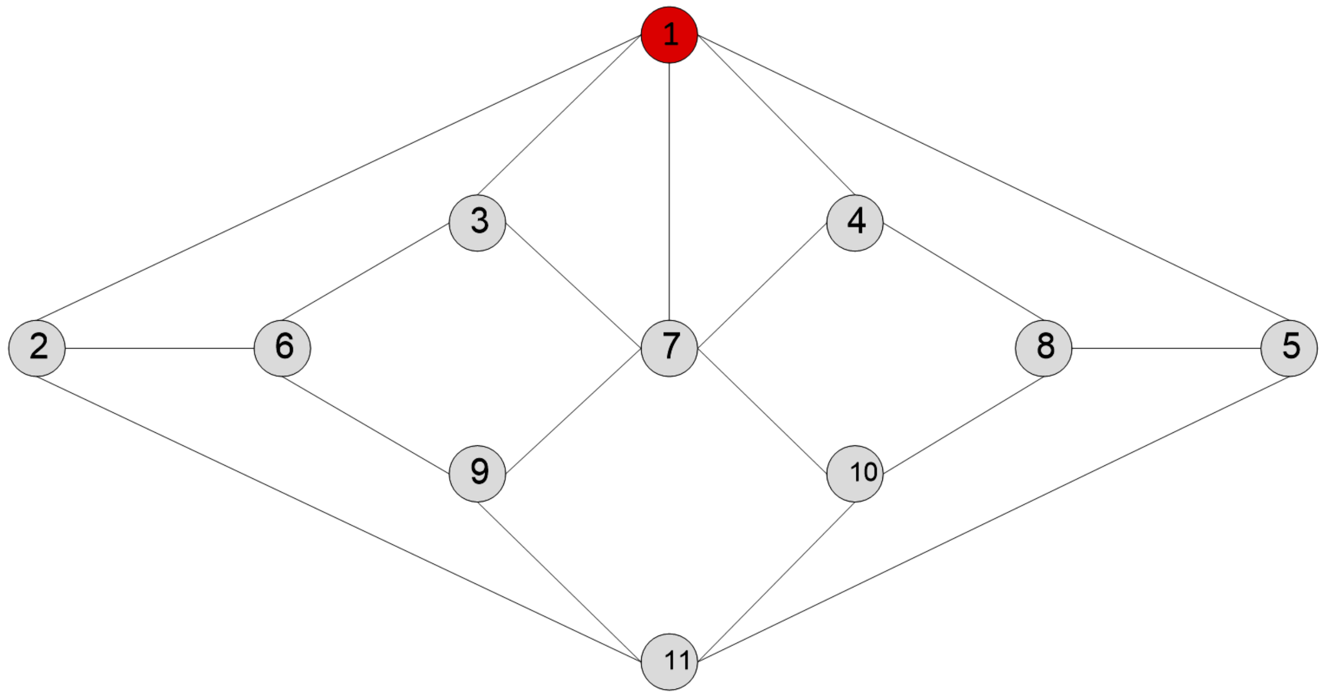
DR Popescu – Combinatorică și Teoria grafurilor (Teorema 4.18)

Graf bipartit

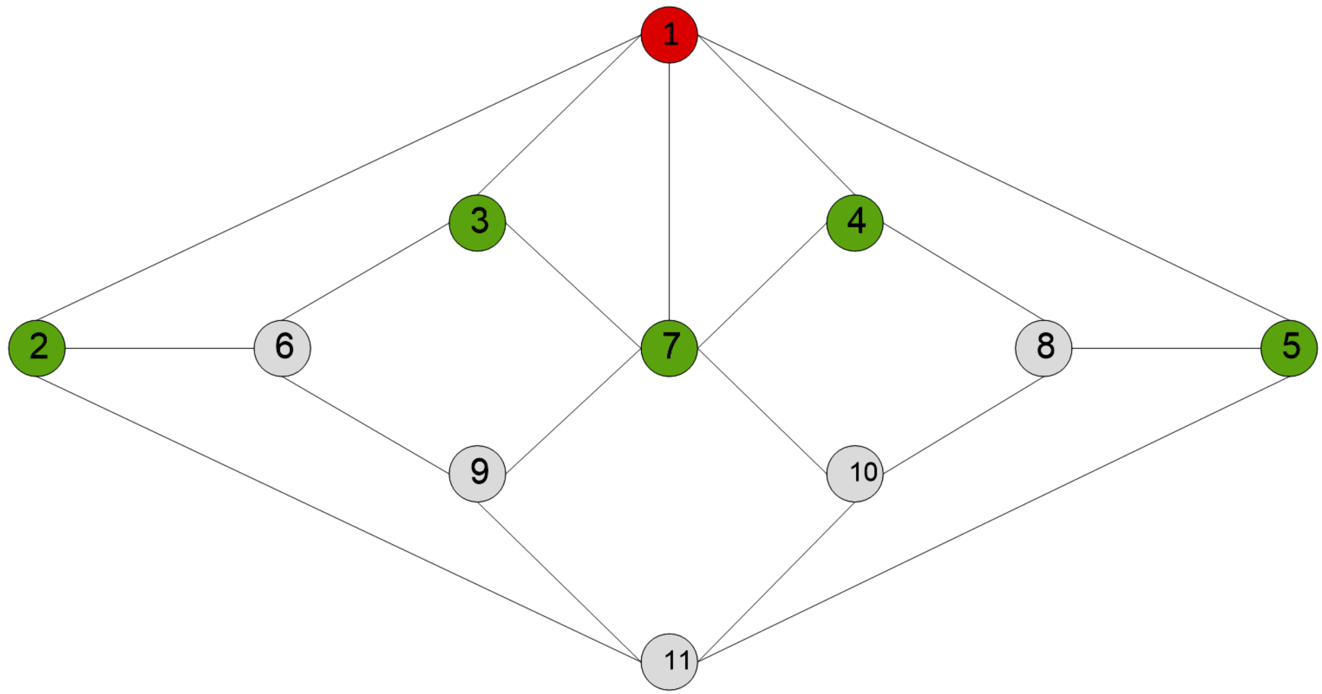
- **Teorema König ⇒ Agoritm pentru a testa dacă un graf conex este bipartit**
  - Colorăm cu (cel mult) 2 culori un arbore parțial al său printr-o **parcurgere** (**colorăm orice vecin j nevizitat al vârfului curent i cu culoarea diferită de cea a lui i**)
  - Testăm dacă celelalte muchii - de la i la **vecini j deja vizitați** (colorați) au extremitățile i și j colorate diferit

Dacă graful nu este conex, testăm fiecare componentă conexă

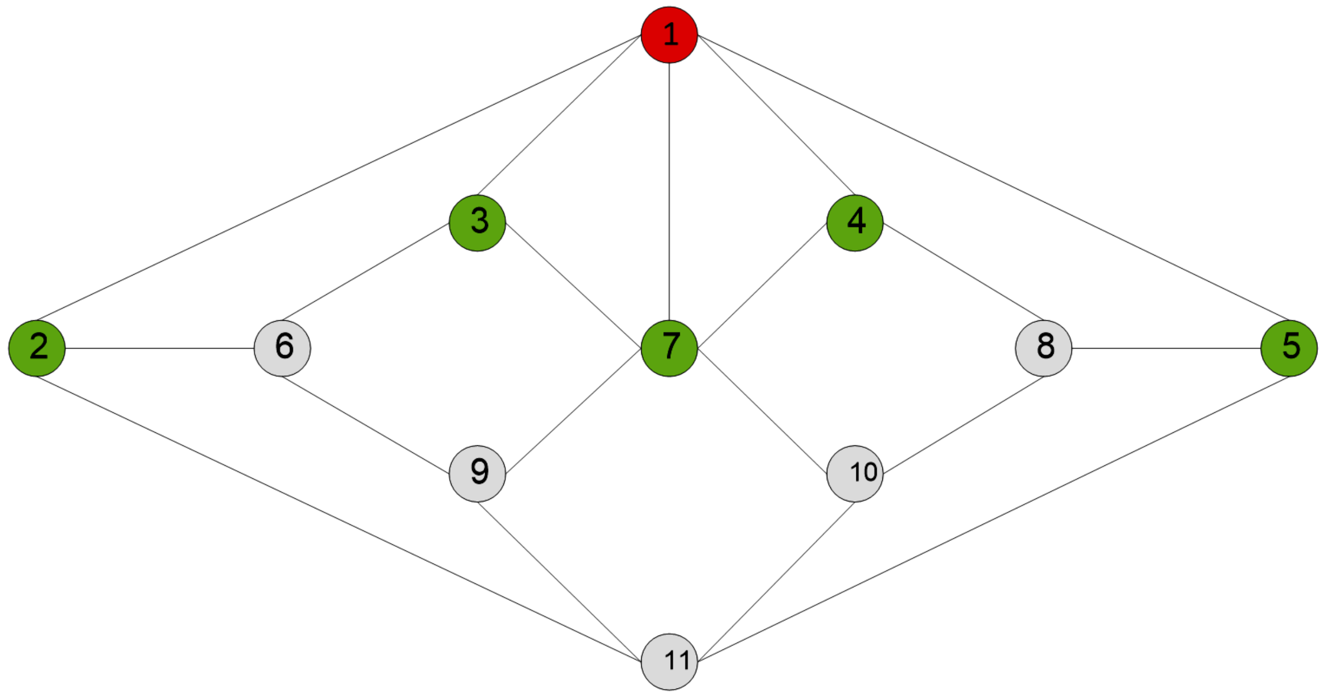
Exemplu test bipartit BF



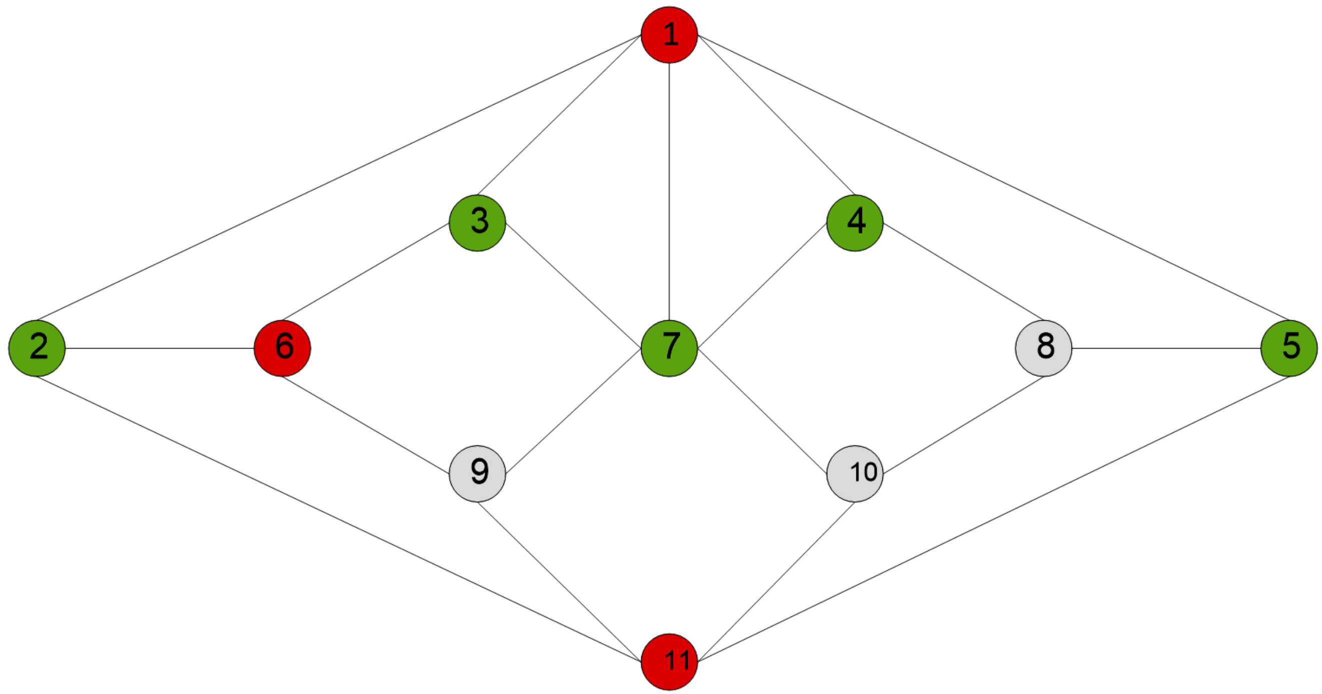
Exemplu test bipartit BF



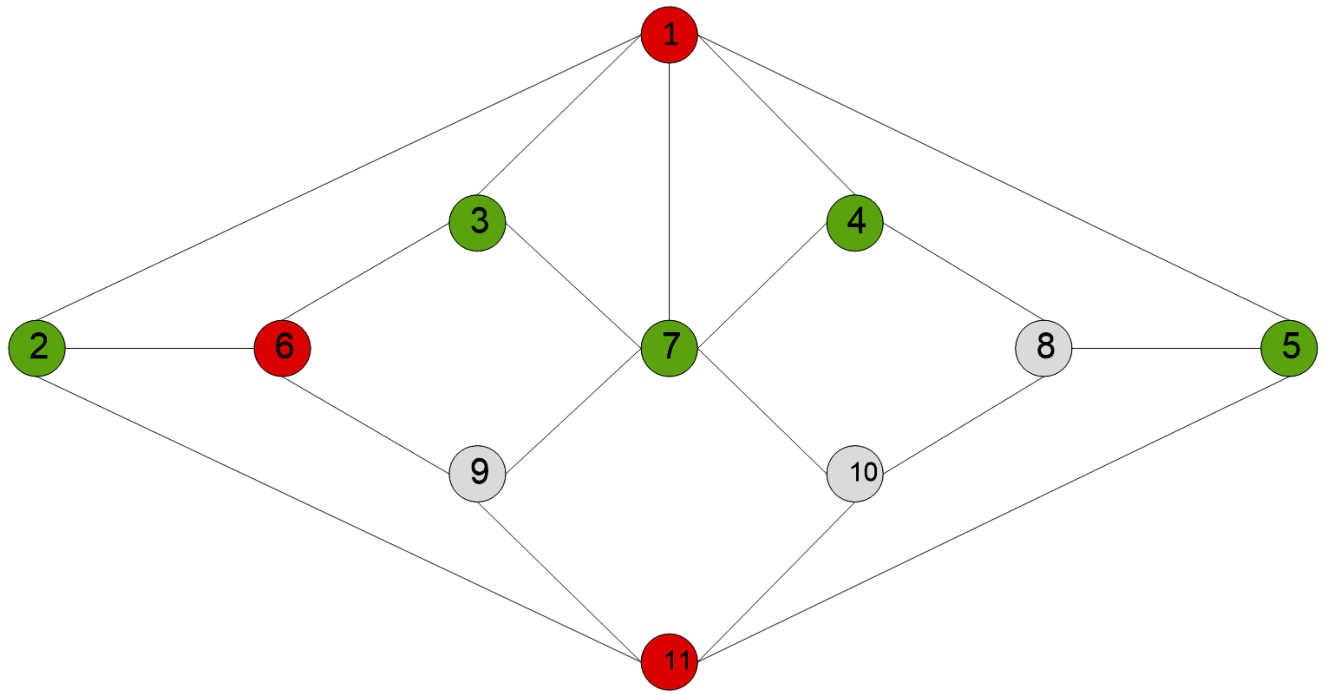
Exemplu test bipartit BF



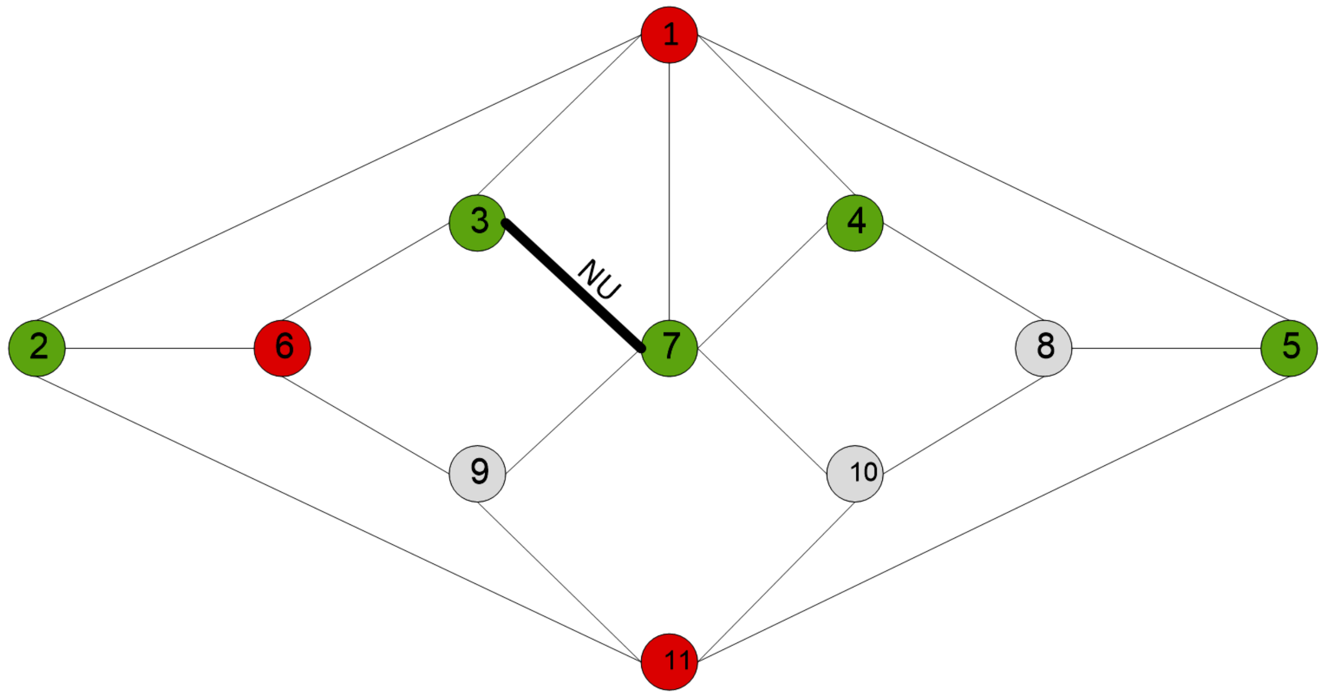
Exemplu test bipartit BF



Exemplu test bipartit BF

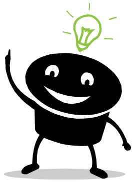


Exemplu test bipartit BF



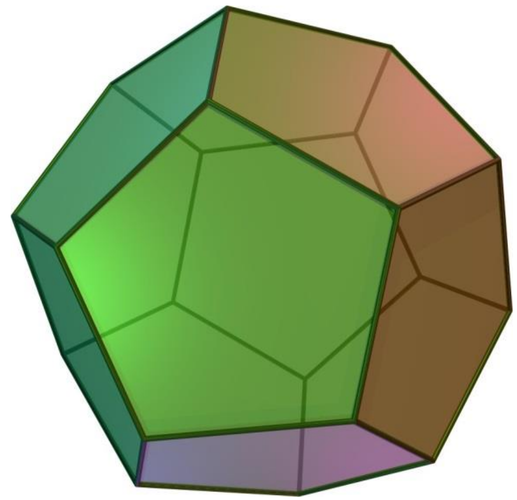
Grafuri planare

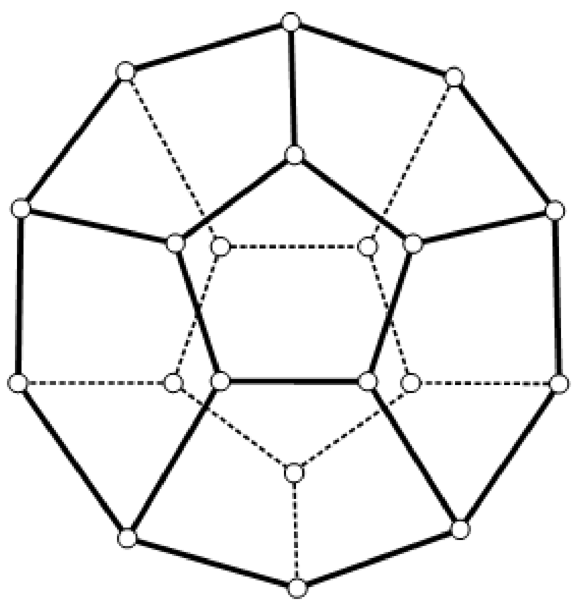
Graf planar

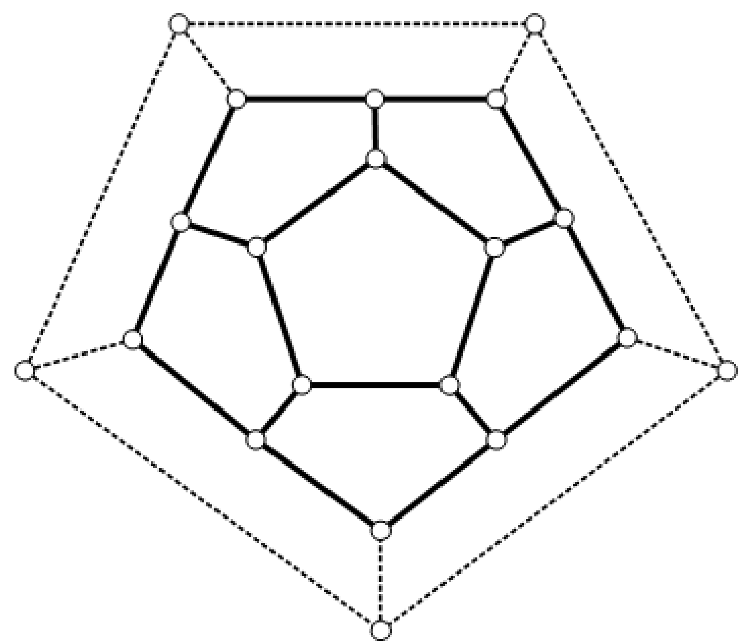


- Amintim din primul curs

Dodecaedrul







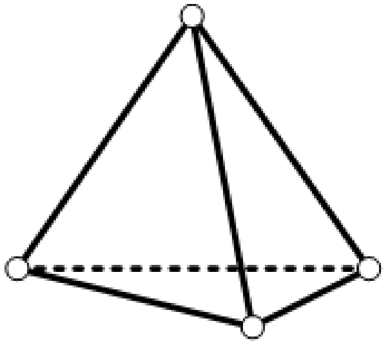
Corpuri platonice

- **Poliedru** – **corp** mărginit de suprafeţe plane
- **Poliedru convex** – segmentul care uneşte două puncte oarecare din el conţine numai puncte din interior
- **Poliedru regulat convex** – feţele sunt poligoane regulate **congruente**

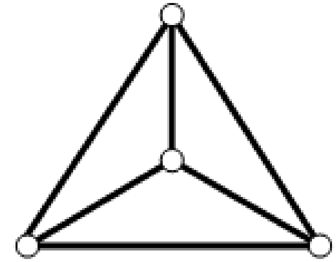
Corpuri platonice

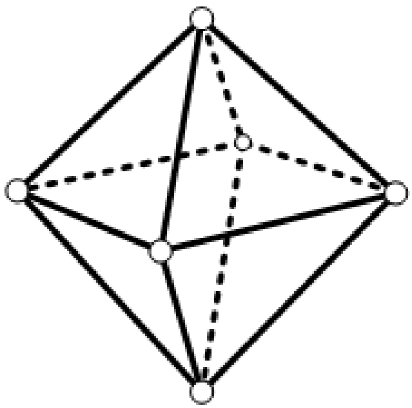
- **Poliedru** – **corp** mărginit de suprafeţe plane
- **Poliedru convex** – segmentul care uneşte două puncte oarecare din el conţine numai puncte din interior
- **Poliedru regulat convex** – feţele sunt poligoane regulate **congruente**

Corpuri platonice – grafuri planare

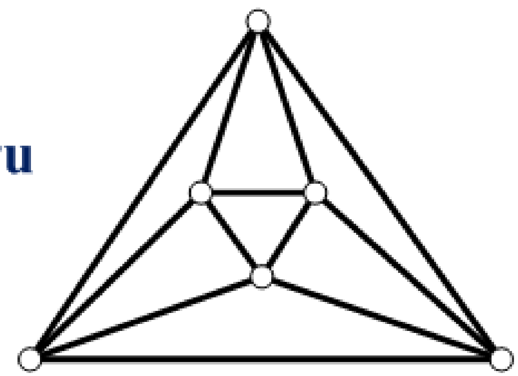


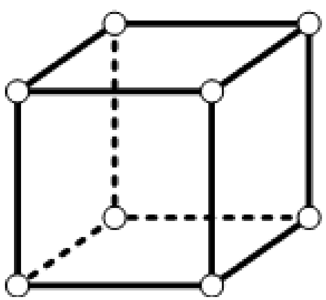
Tetraedru



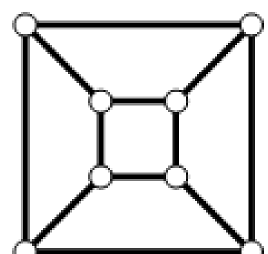


Octaedru

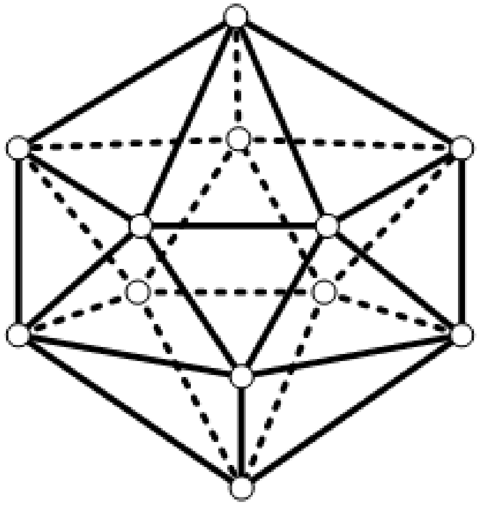




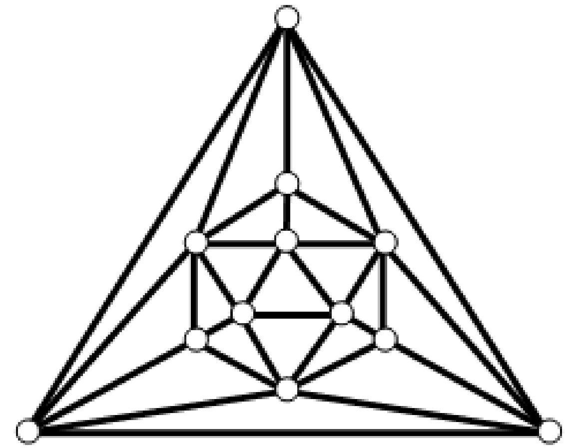
Cub

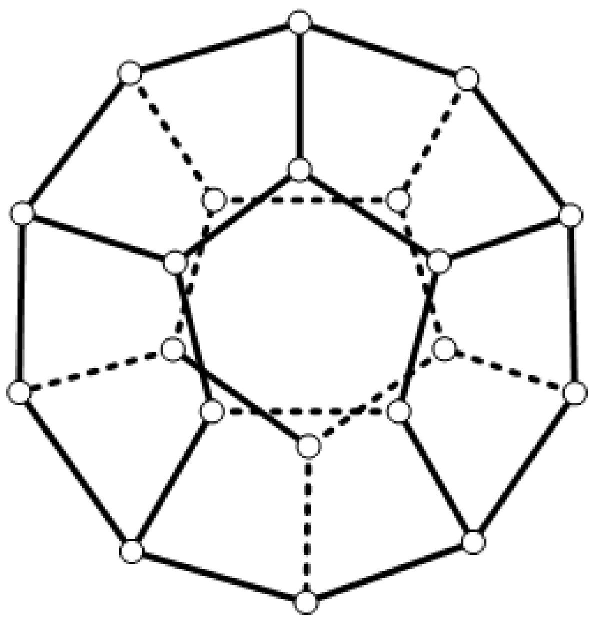


Corpuri platonice – grafuri planare

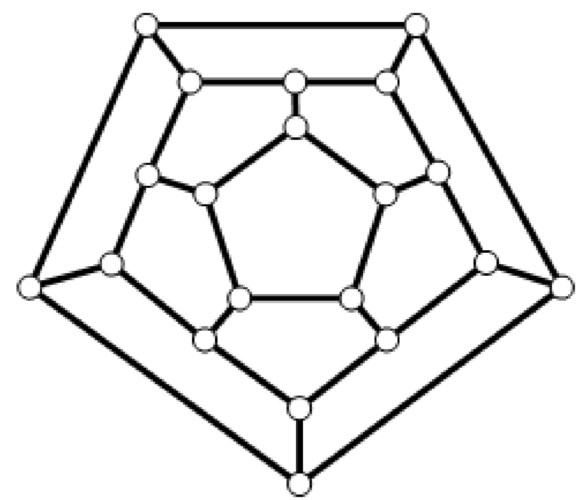


Icosaedru





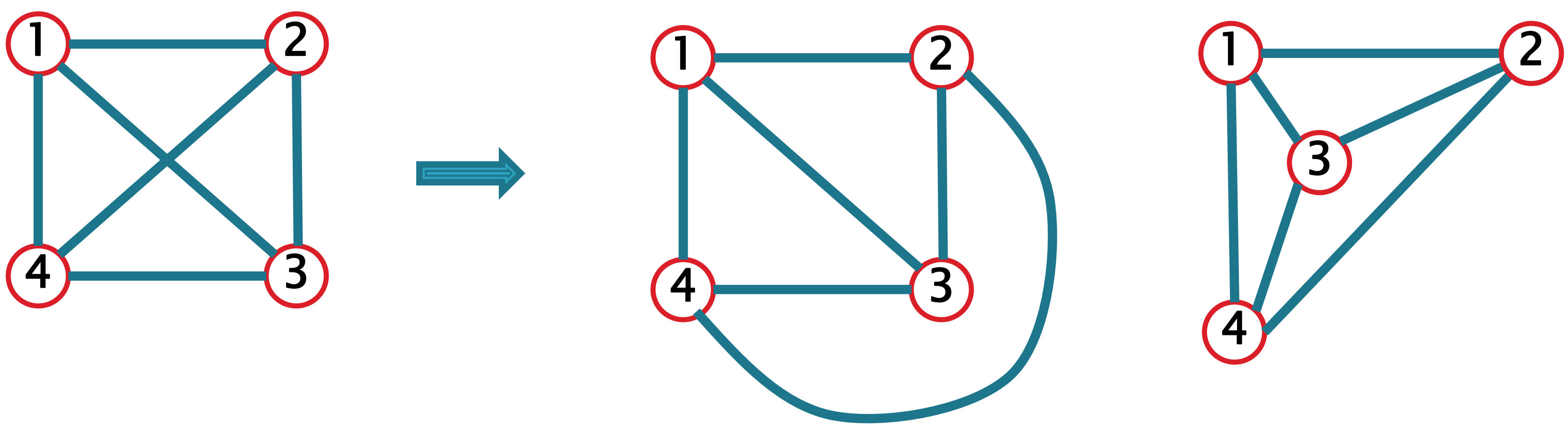
Dodecaedru



Graf planar

- G = (V, E) graf neorientat s.n. planar ⇔ admite o reprezentare în plan a.î. **muchiilor** le corespund segmente de curbe continue care **nu se intersectează în interior** unele pe altele

- O astfel de reprezentare s.n hartă a lui G



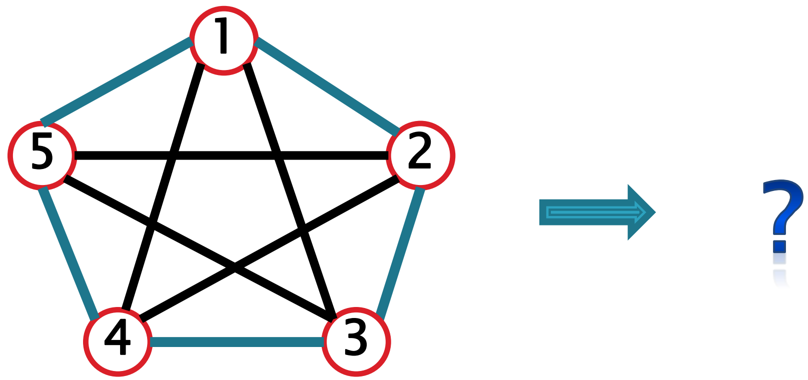
$G \sim K_4$

hărți

Graf planar

- G = (V, E) graf neorientat s.n. planar ⇔ admite o reprezentare în plan a.î. **muchiilor** le corespund segmente de curbe continue care **nu se intersectează în interior** unele pe altele

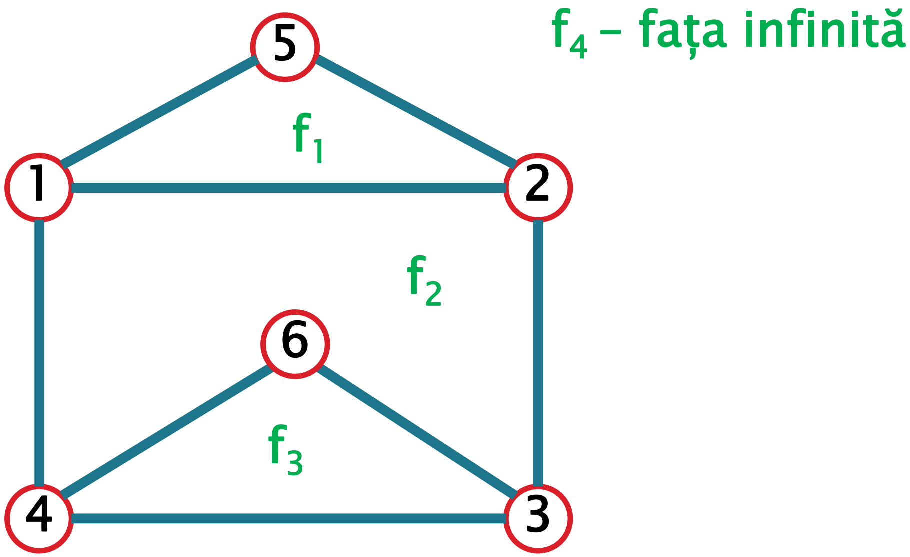
- O astfel de reprezentare s.n **hartă** a lui G



$G \sim K_5$

Graf planar

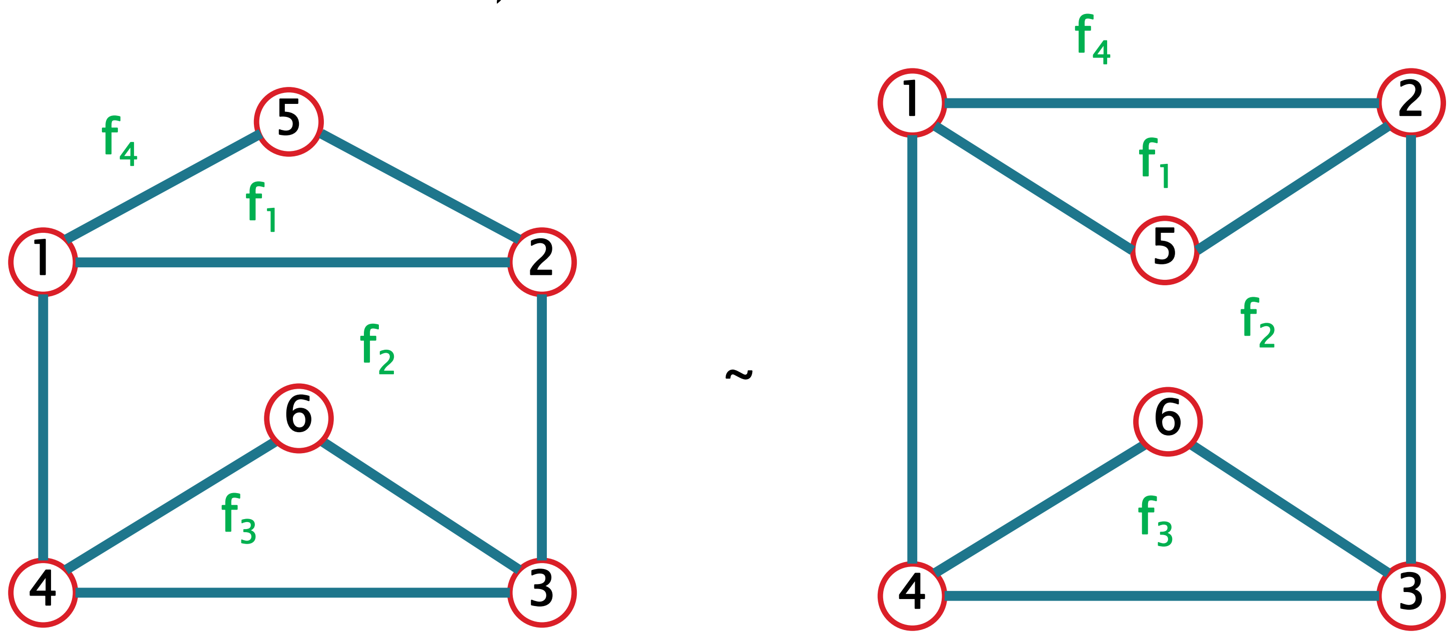
- Fie G = (V, E) graf planar, M o hartă a sa
- M induce o împărțire a planului într-o mulțime **F** de părți convexe numite **fețe**
- Una dintre acestea este **fața infinită (exterioară)**



Graf planar

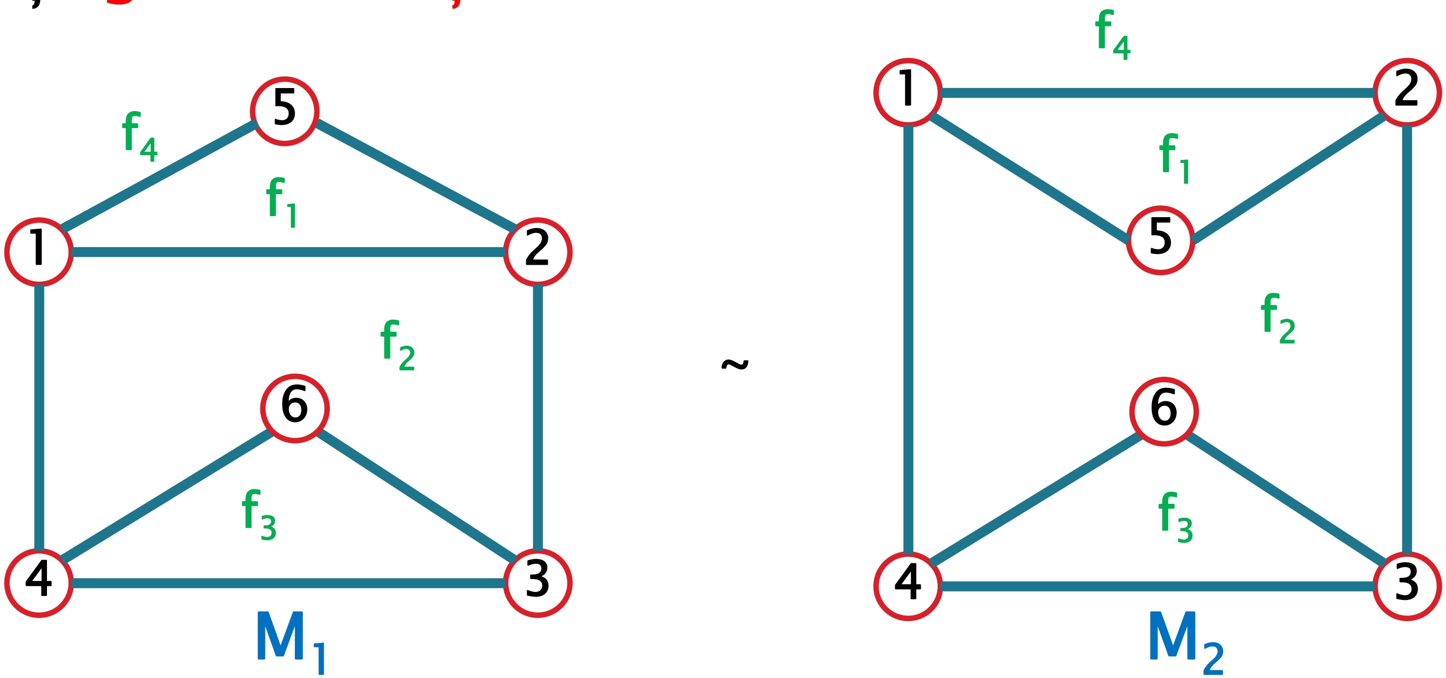
- M = **(V, E, F) hartă**

- Pentru o față f ∈ F definim
  - **d_M(f) = gradul feței f** = numărul muchiilor lanțului închis (**frontierei**) care delimitează f (*câte muchii sunt parcurse atunci când traversăm frontiera*)



Graf planar

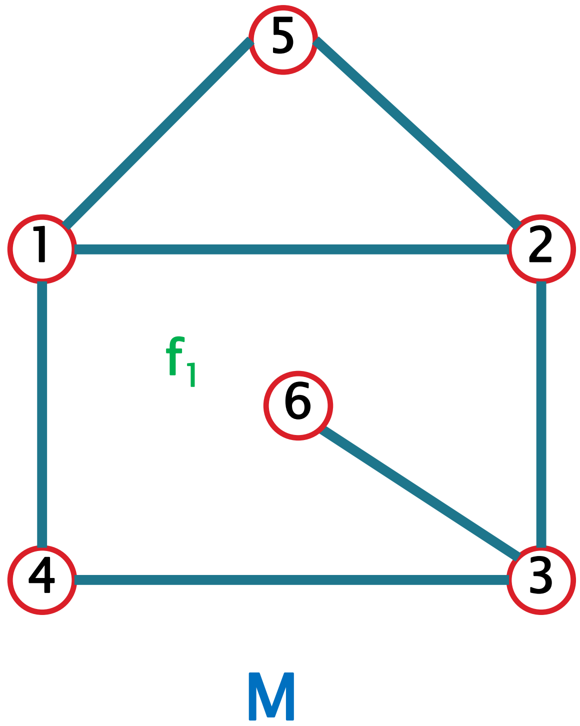
Observație: Hărți diferite ale aceluiași graf pot avea **secvența gradelor fețelor diferită**



$d_{M1}(f_1) = 3$
$d_{M1}(f_2) = 5$
$d_{M1}(f_3) = 3$
$d_{M1}(f_4) = 5$

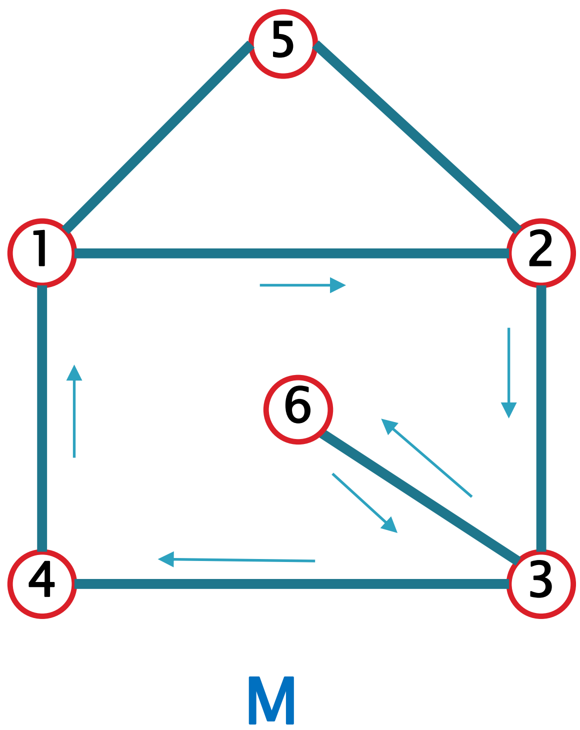
$d_{M2}(f_1) = 3$
$d_{M2}(f_2) = 6$
$d_{M2}(f_3) = 3$
$d_{M2}(f_4) = 4$

Graf planar



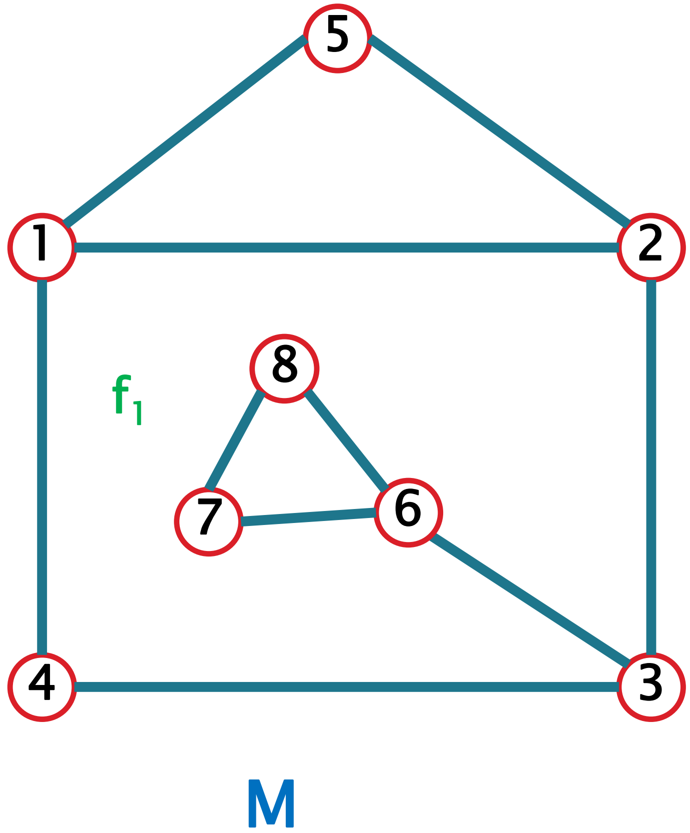
$d_M(f_1) = ?$

Graf planar



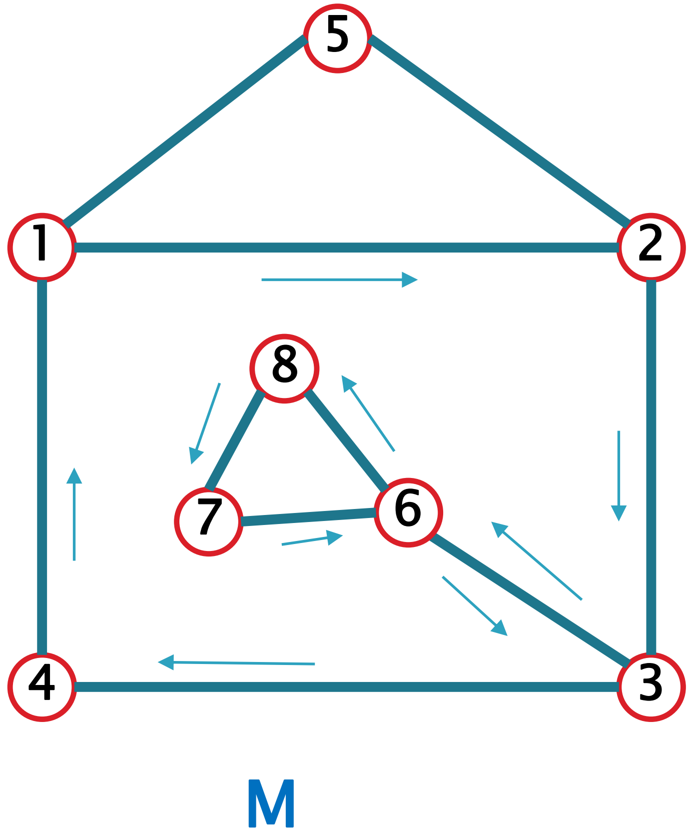
$d_M(f_1) = 6$

Graf planar



$d_M(f_1) = ?$

Graf planar



Graf planar

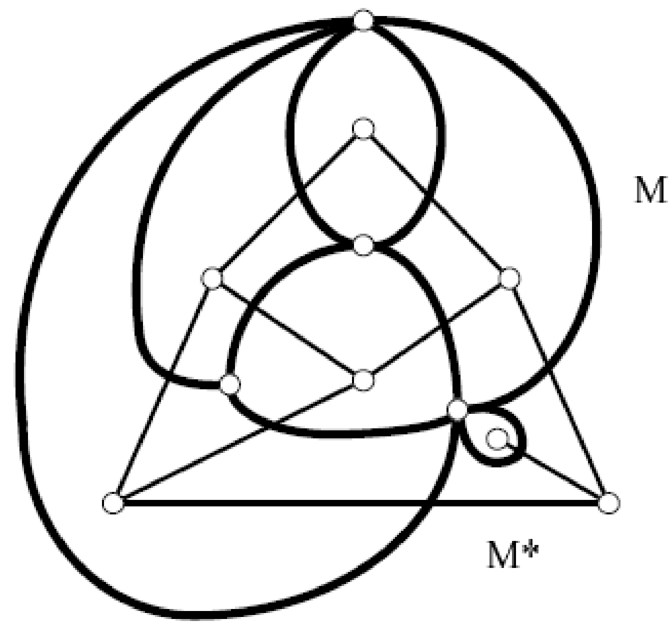
- M = (V, E, F) hartă
  - **Avem**

$$\sum_{f \in F} d_M(f) = 2|E|$$

(deoarece o muchie este incidentă cu două fețe)

Graf planar

- **Harta duală**
- M=(V,E,F) -> **harta duală** M*=(V*, E*, F*) se costruiește astfel:
  - V* : se consideră câte un punct f* în interiorul fiecărei fețe f din M
  - E*: pentru fiecare muchie *e* a lui M comună la două fețe *f′* și *f′′* se construiește un segment de curbă continuă e* cu capetele în vârfurile *f′** și *f′′** asociate fețelor *f′* și *f′′* astfel încât să intersecteze în interior muchia *e* și să nu mai intersecteze astfel nicio altă muchie a lui *M*



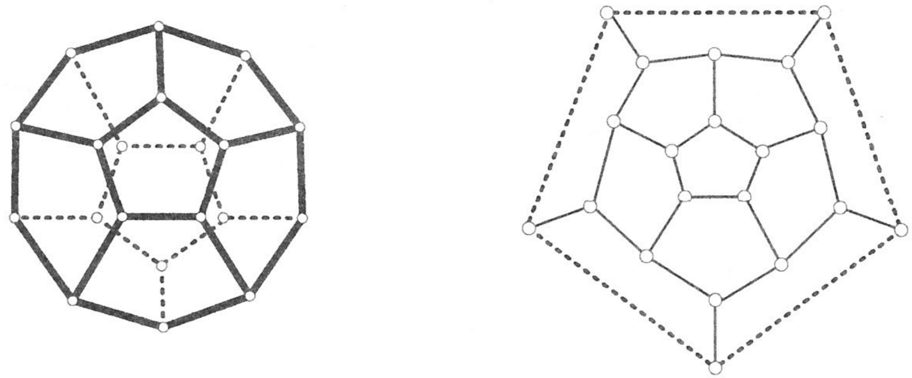
Graf planar

- **Harta duală**
- $M=(V,E,F)$ -> **harta duală** $M^*=(V^*, E^*, F^*)$
  - $(M^*)^*$ este izomorf cu M
  - $d_{M^*}(f^*) = d_M(f)$

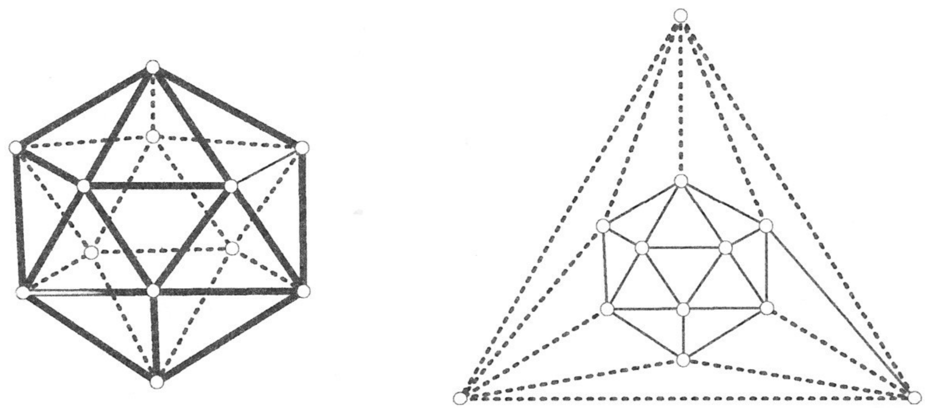
  (gradul vârfului corespunzător feței în dual = gradul feței în M)

Graf planar

Dodecaedru:

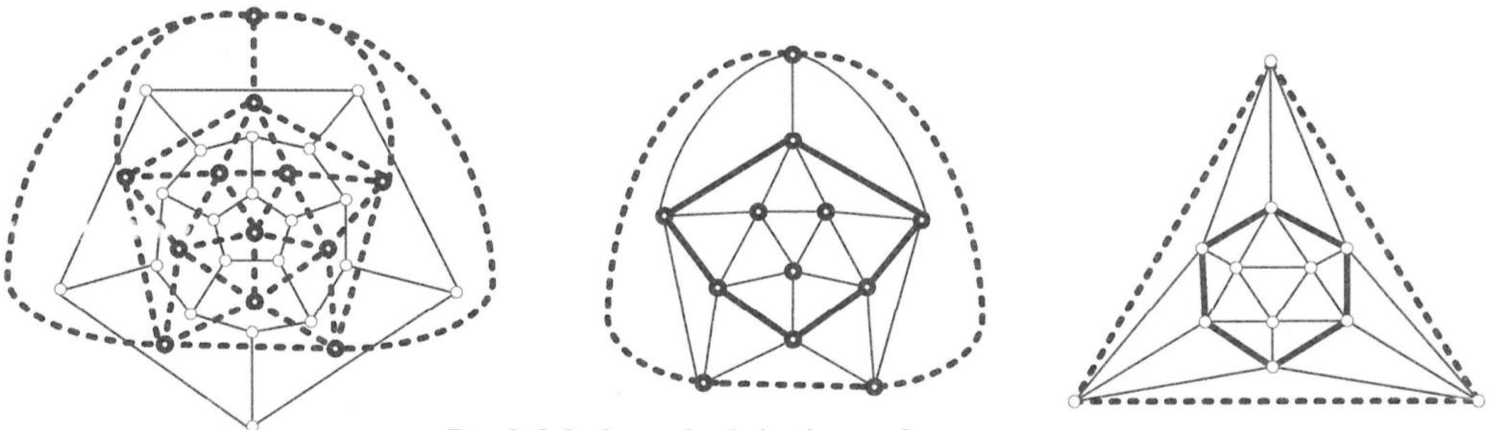


Icosaedru:

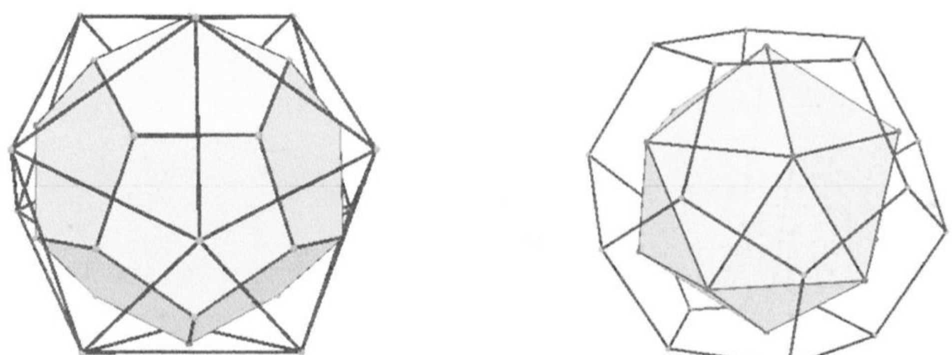


duale

Graf planar

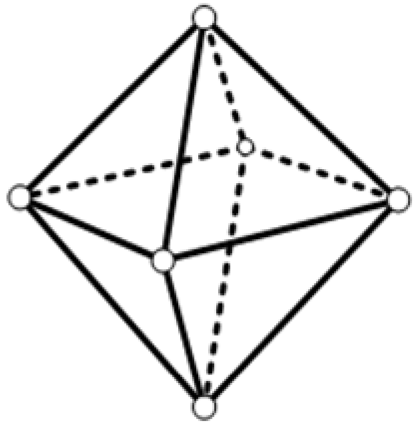


Dualul dodecaedrului = icosaedru

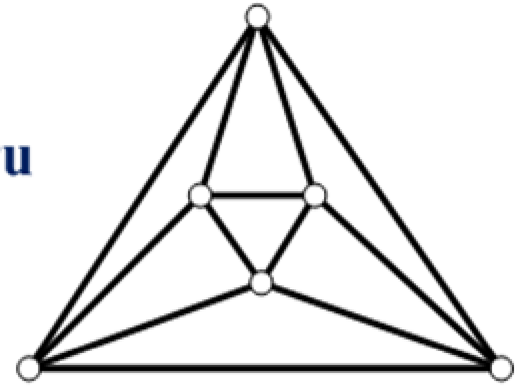


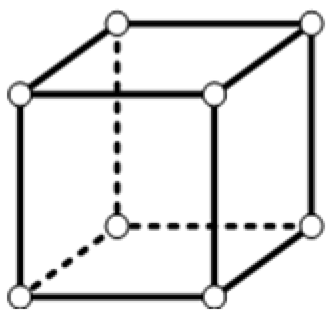
Dualitatea dodecaedru ⇔ icosaedru (http://apollonius.math.nthu.edu.tw/d1/dg-07-exe/943251/dynamic/duality.htm**)**

Graf planar

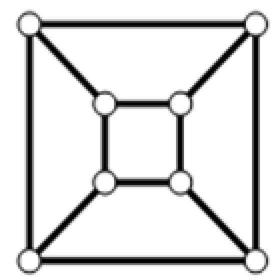


Octaedru



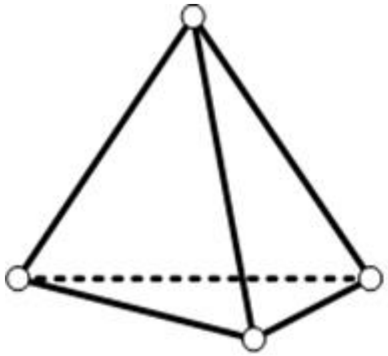


Cub

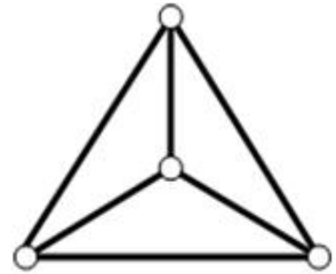


duale

Graf planar

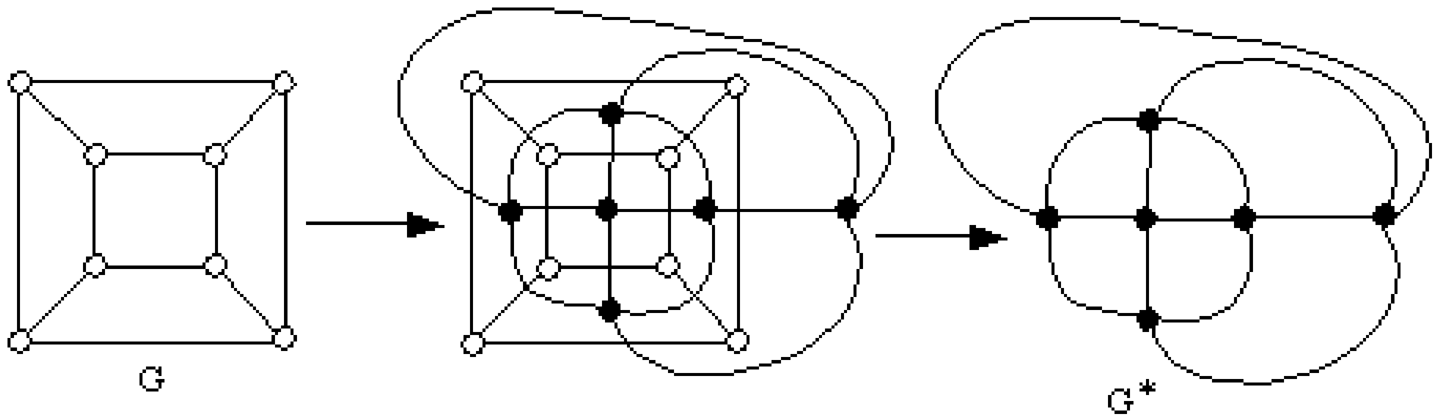


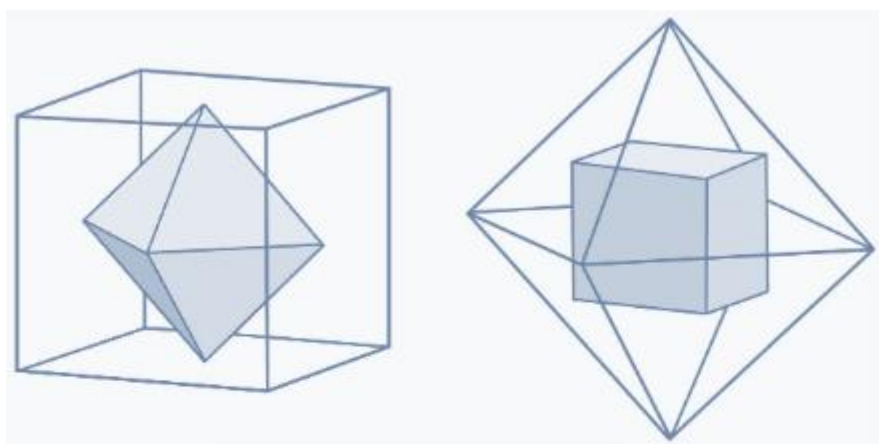
Tetraedru



Autodual (M izomorf cu M*)

Graf planar





http://origametry.net/combgeom/graphnotes.html

Graf planar

- **Teorema poliedrală a lui EULER**

  Fie $G=(V, E)$ un graf planar **conex** și $M = (V, E, F)$ o hartă a lui. Are loc relația

$$|V| - |E| + |F| = 2$$

Graf planar

- **Teorema poliedrală a lui EULER**

  Fie G=(V, E) un graf planar **conex** și M = (V, E, F) o hartă a lui. Are loc relația

$$|V| - |E| + |F| = 2$$

- **Consecință**

  Orice hartă M a lui G are $2 - |V| + |E|$ fețe

Graf planar

- **Proprietăți**

Fie $G=(V, E)$ un graf planar conex cu $n=|V|>2$ și $m=|E|$.

Atunci:

a) $m \leq 3n - 6$

b) $\exists\ x \in V$ cu $d(x) \leq 5$.

Graf planar

- **Proprietăți**

    Fie $G=(V, E)$ un graf planar conex cu $n=|V|>2$ și $m=|E|$.
Atunci:

    a) $\mathbf{m \leq 3n - 6}$

    b) $\mathbf{\exists\, x \in V}$ **cu** $\mathbf{d(x) \leq 5}$.

- **Consecință**

    K_5 nu este graf planar

Graf planar

- **Proprietăți (temă)**

Fie $G=(V, E)$ un graf planar conex **bipartit** cu $n=|V|>2$ și $m=|E|$. Atunci:

a) $m \leq 2n - 4$

b) $\exists\, x \in V$ cu $d(x) \leq 3$.

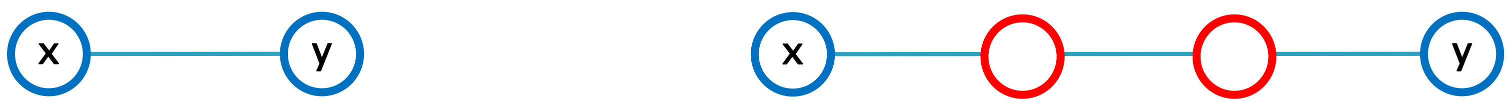
- **Consecință**

$K_{3,3}$ nu este graf planar

Graf planar

Teorema lui Kuratowski

subdiviziune a unei muchii = înlocuire a muchiei cu un lanț de la x la y cu vârfuri intermediare noi (se adaugă vârfuri noi "pe" muchia xy)



Graful H este o **subdiviziune a lui G** = se poate obține din G printr-o secvență finită de subdiviziuni de muchii

Graf planar

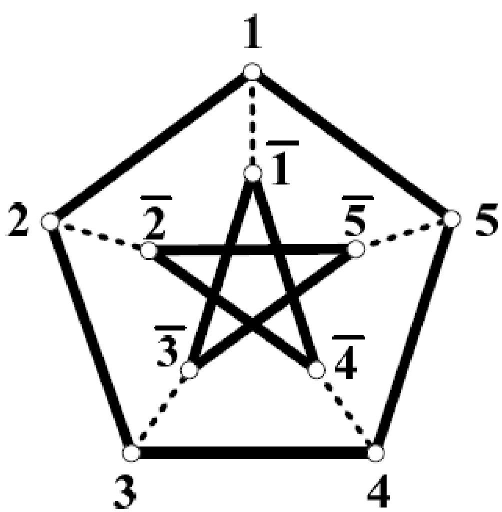
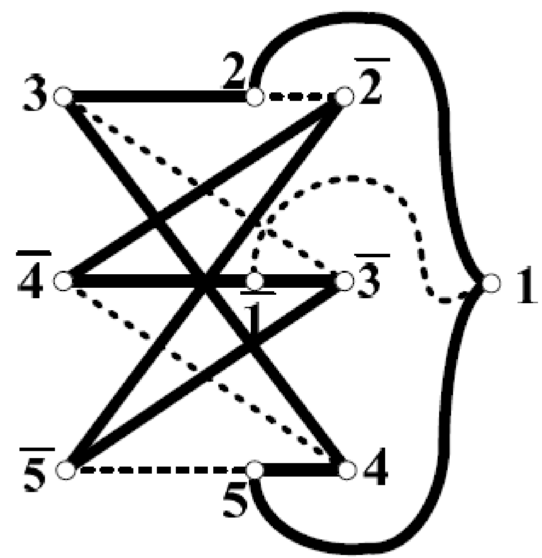
- **Teorema lui Kuratowski**

G este graf planar ⇔ nu conține subdiviziuni ale lui $K_{3,3}$ și ale lui K_5.

Graf planar

- **Teorema lui Kuratowski**

G este graf planar ⇔ nu conține subdiviziuni ale lui $K_{3,3}$ și ale lui K_5.

Graful lui Petersen

Graf planar

- **Teorema celor 6 culori**

  Orice graf planar conex este 6 -colorabil.

- **Demonstrație- Algoritm de colorare a unui graf planar cu 6 culori**

Graf planar

- ## Teorema celor 6 culori

  Orice graf planar conex este 6 -colorabil.

- ## Algoritm de colorare a unui graf planar cu 6 culori

```
colorare(G)
      daca |V(G)|≤ 6 atunci coloreaza varfurile cu
      culori distincte din {1,…,6}
```

Graf planar

- **Teorema celor 6 culori**

  Orice graf planar conex este 6 -colorabil.

- **Algoritm de colorare a unui graf planar cu 6 culori**

```
colorare(G)
      daca |V(G)|≤ 6 atunci coloreaza varfurile cu
      culori distincte din {1,…,6}
      altfel
            alege x cu d(x) ≤ 5
```

Graf planar

- **Teorema celor 6 culori**

  Orice graf planar conex este 6 -colorabil.

- **Algoritm de colorare a unui graf planar cu 6 culori**

```
colorare(G)
      daca |V(G)|≤ 6 atunci coloreaza varfurile cu
      culori distincte din {1,…,6}
      altfel
            alege x cu d(x) ≤ 5
            colorare(G - x)
```

Graf planar

- **Teorema celor 6 culori**

  Orice graf planar conex este 6 -colorabil.

- **Algoritm de colorare a unui graf planar cu 6 culori**

```
colorare(G)
      daca |V(G)|≤ 6 atunci coloreaza varfurile cu
      culori distincte din {1,…,6}
      altfel
            alege x cu d(x) ≤ 5
            colorare(G - x)
            colorează x cu o culoare din {1,…,6}
            diferită de culorile vecinilor deja
            colorați (!se poate, x are cel mult 5
            vecini din G - x)
```

Graf planar

- Algoritm de colorare a unui graf planar cu 6 culori

  Sugestie implementare nerecursivă

  1. Determinarea iterativă a ordinii $v_1,\ldots v_n$ în care sunt colorate vârfurile – de la ultimul la primul astfel:

     - v_n – un varf de grad ≤ 5 în G
     - v_{n-1} – un varf de grad ≤ 5 în G – v_n
     - v_{n-2} – un varf de grad ≤ 5 în G – $\{v_n , v_{n-1}\}$
     - …
     - v_i – un varf de grad ≤ 5 în G – $\{v_n , v_{n-1}, \ldots, v_{i+1}\}$
     - …
     - v_1 – un varf de grad ≤ 5 în G – $\{v_n , v_{n-1}, \ldots, v_2\}$

  2. Colorăm pe rând vârfurile $v_1,\ldots, v_n$ cu o culoare din $\{1,\ldots,6\}$ diferită de culorile vecinilor deja colorați

Determinarea iterativă a ordinii în care sunt colorate vârfurile la pasul 1 se poate face similar cu determinarea unei ordonări topologice:

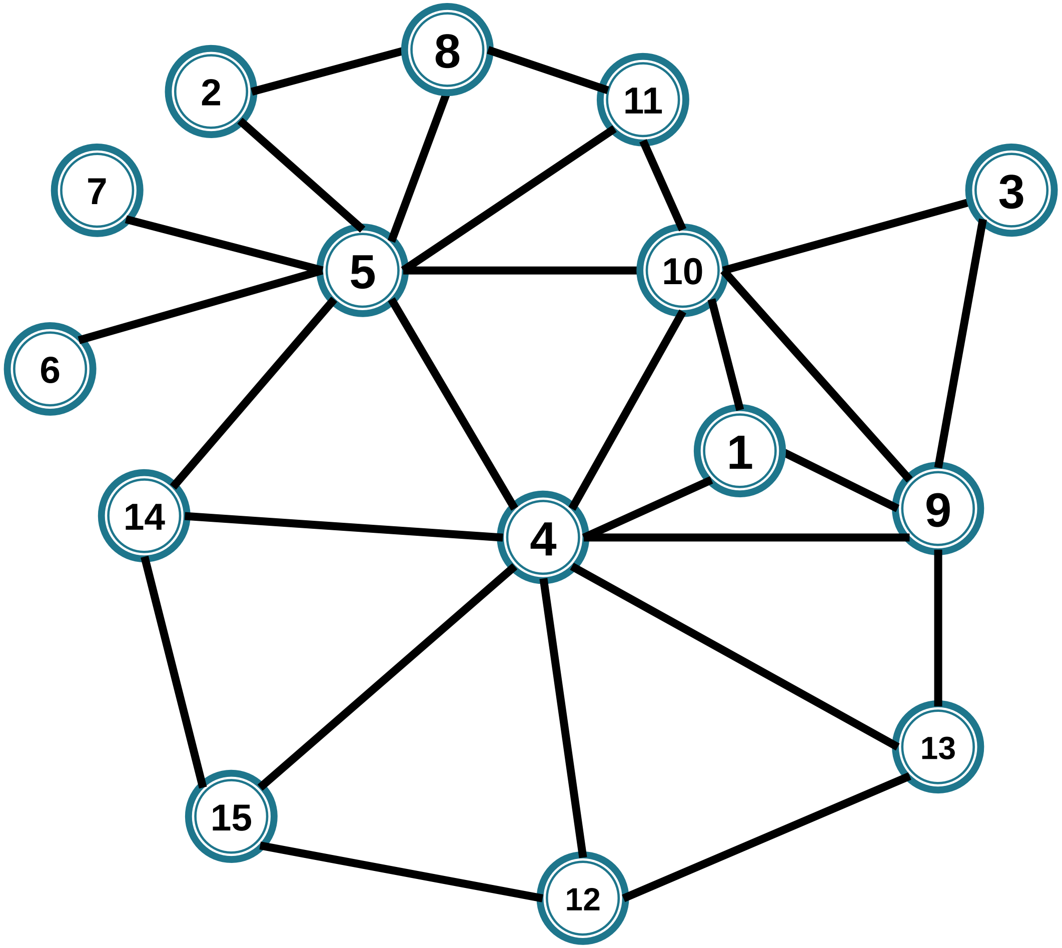
```
    coada C = ∅;
    adauga in C toate vârfurile v cu d[v] ≤ 5 și
marcheaza-le ca vizitate
```

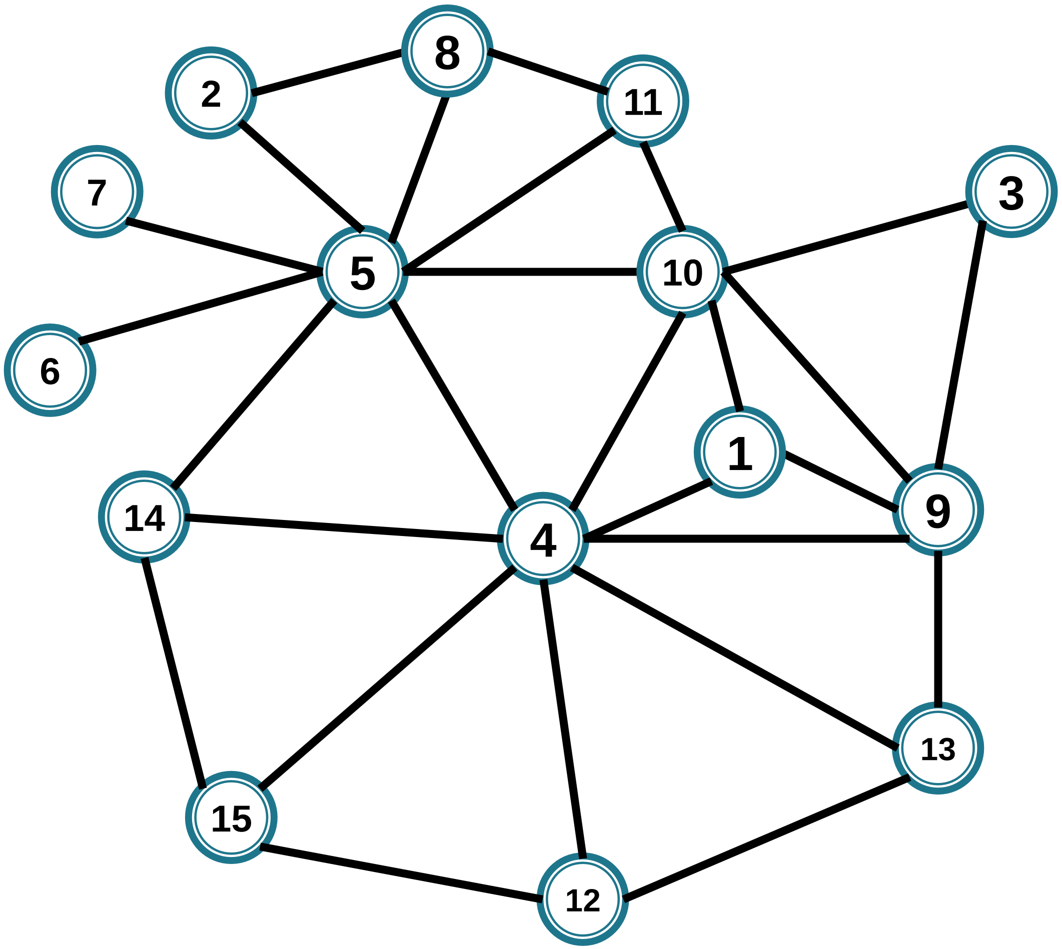
Determinarea iterativă a ordinii în care sunt colorate vârfurile la pasul 1 se poate face similar cu determinarea unei ordonări topologice:

```
    coada C = ∅;
    adauga in C toate vârfurile v cu d[v] ≤ 5 și
 marcheaza-le ca vizitate
    stiva S = ∅
    cat timp C ≠ ∅ executa
        i ←  extrage(C);
        adauga i in S
        pentru ij ∈ E executa
            d[j] = d[j] - 1
            daca d[j] ≤ 5 si j este nevizitat atunci
                adauga(j, C)
```

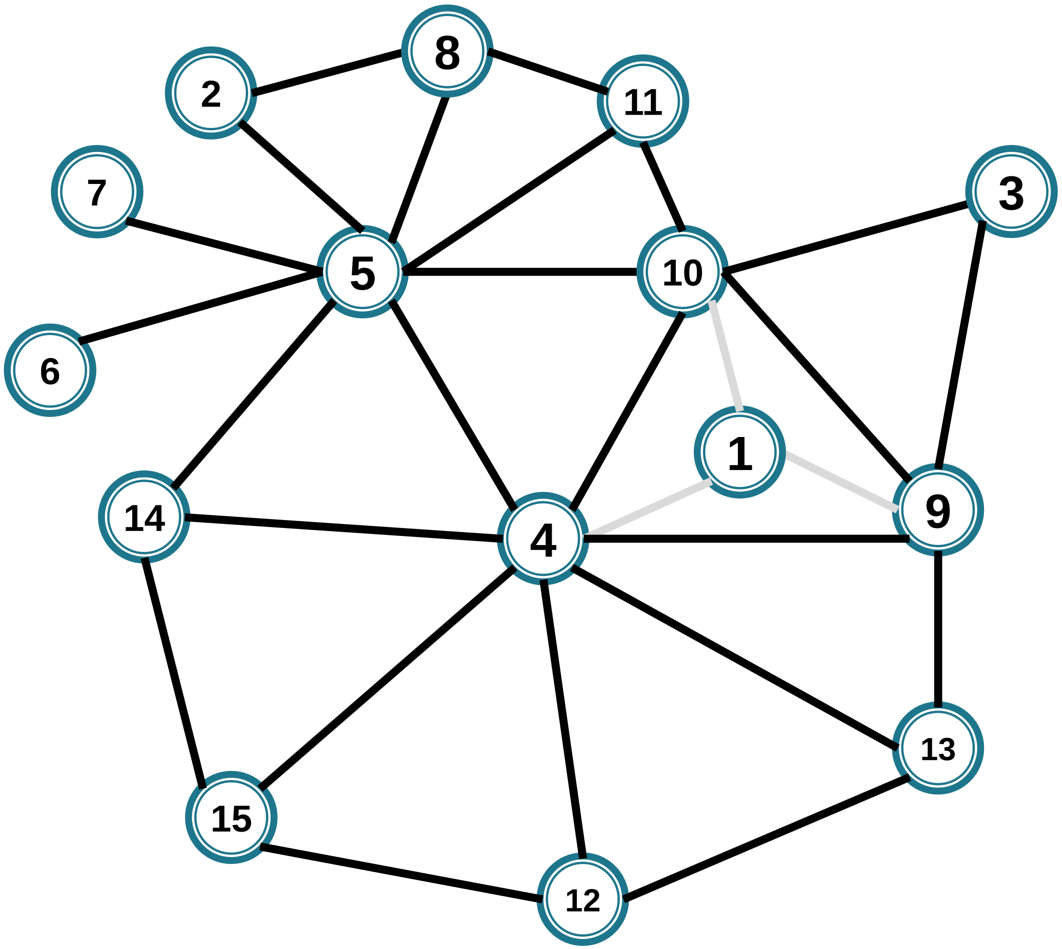
Determinarea iterativă a ordinii în care sunt colorate vârfurile la pasul 1 se poate face similar cu determinarea unei ordonări topologice:

```
    coada C = ∅;
    adauga in C toate vârfurile v cu d[v] ≤ 5 și
 marcheaza-le ca vizitate
    stiva S = ∅
    cat timp C ≠ ∅ executa
       i ←  extrage(C);
       adauga i in S
       pentru ij ∈ E executa
          d[j] = d[j] - 1
          daca d[j] ≤ 5 si j este nevizitat atunci
              adauga(j, C)
    cat timp S este nevida executa
        u = pop(S)
        adauga u in sortare
```


8
2
11
7
3
5
10
6
1
14
4
9
13
15
12

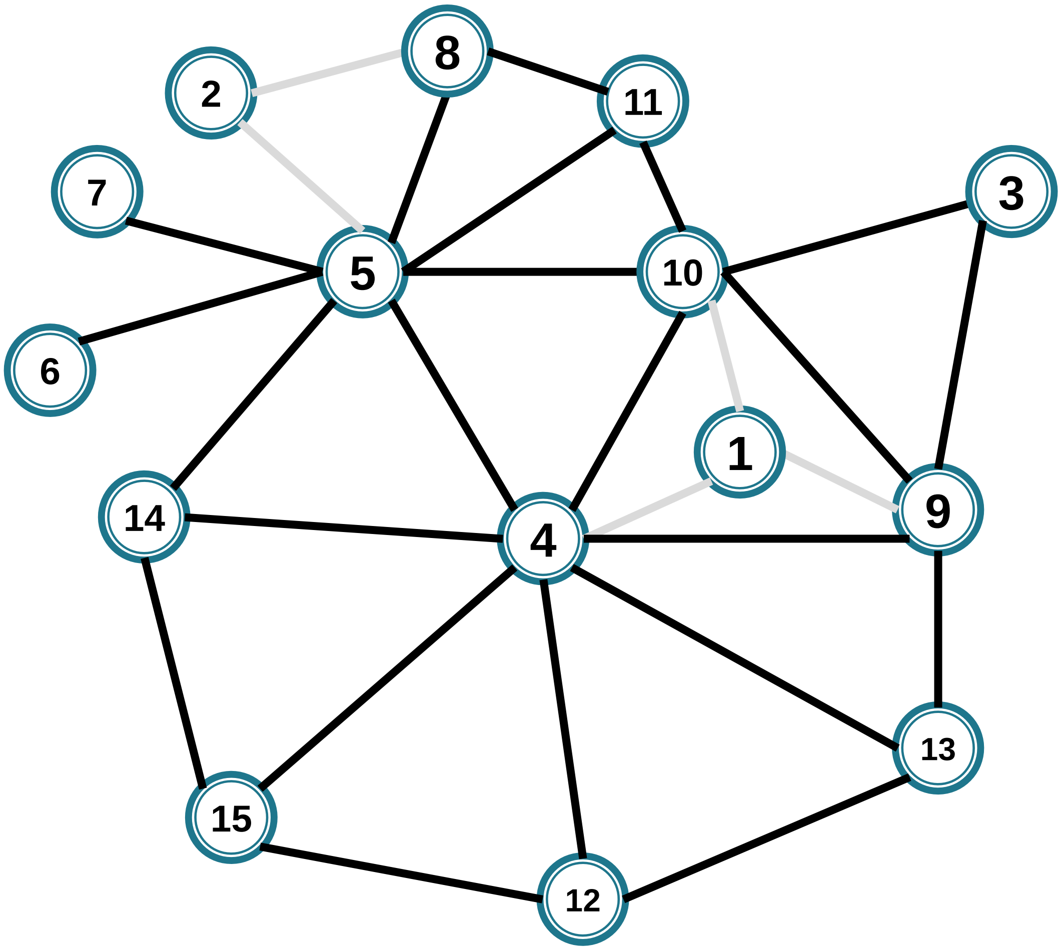


$$\underbrace{1,\ 2,\ 3,\ 6,\ 7,\ 8,\ 9,\ 11,\ 12,\ 13,\ 14,\ 15}_{\text{grad}<=5}$$

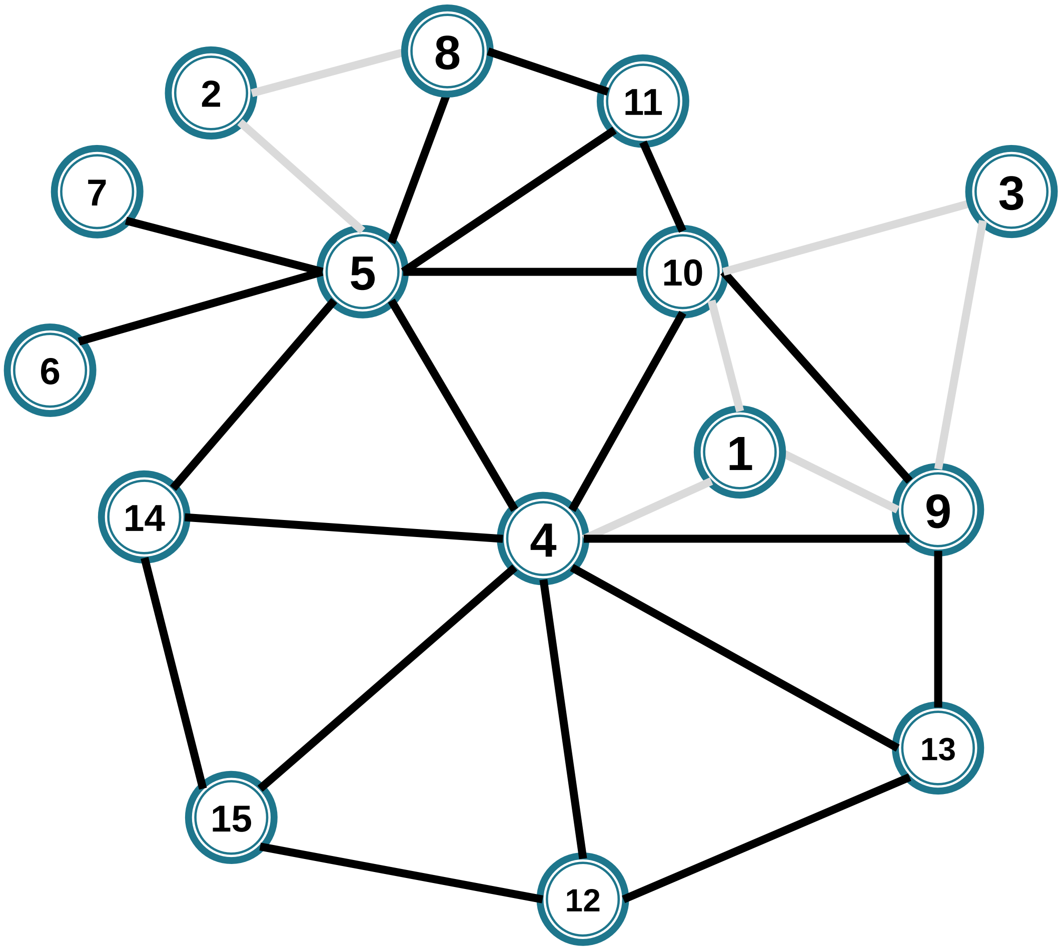


~~1~~, 2, 3, 6, 7, 8, 9, 11, 12, 13, 14, 15, 10

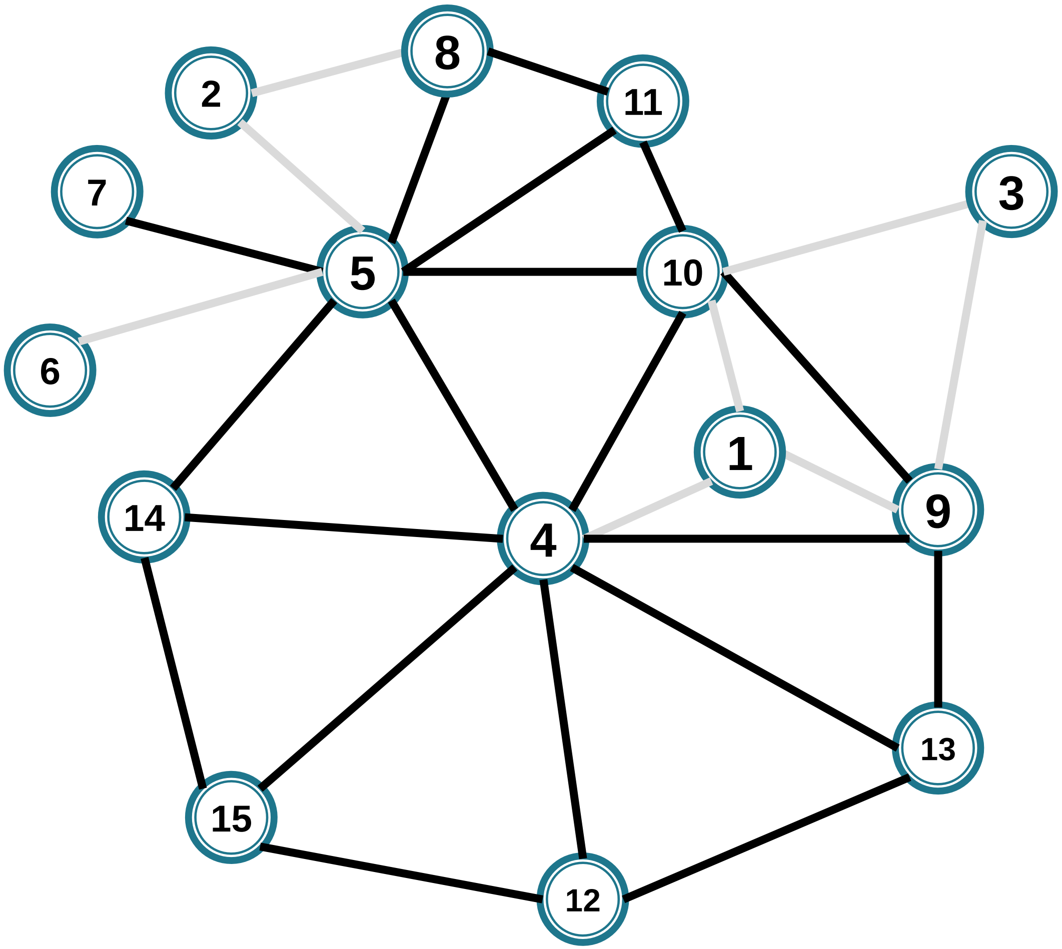
vecinul lui 1 care a devenit de grad<=5



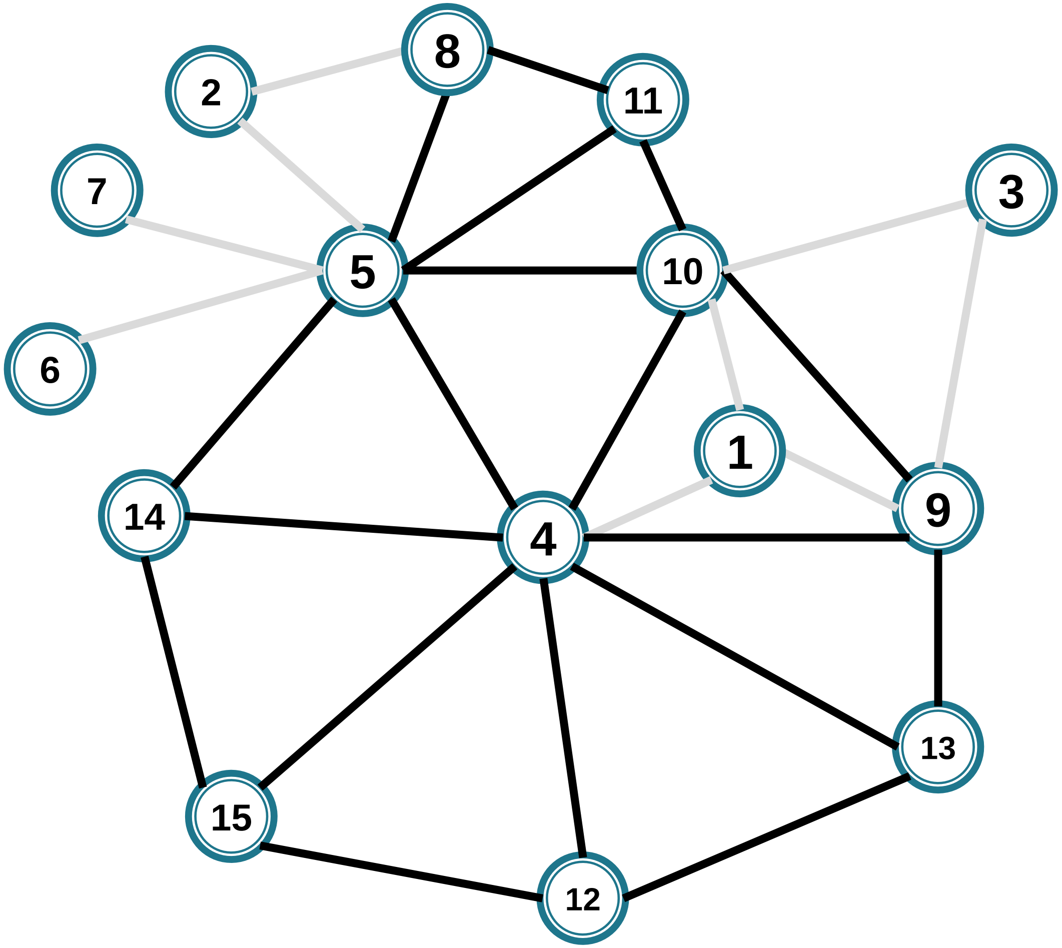
~~1~~, ~~2~~, 3, 6, 7, 8, 9, 11, 12, 13, 14, 15, 10



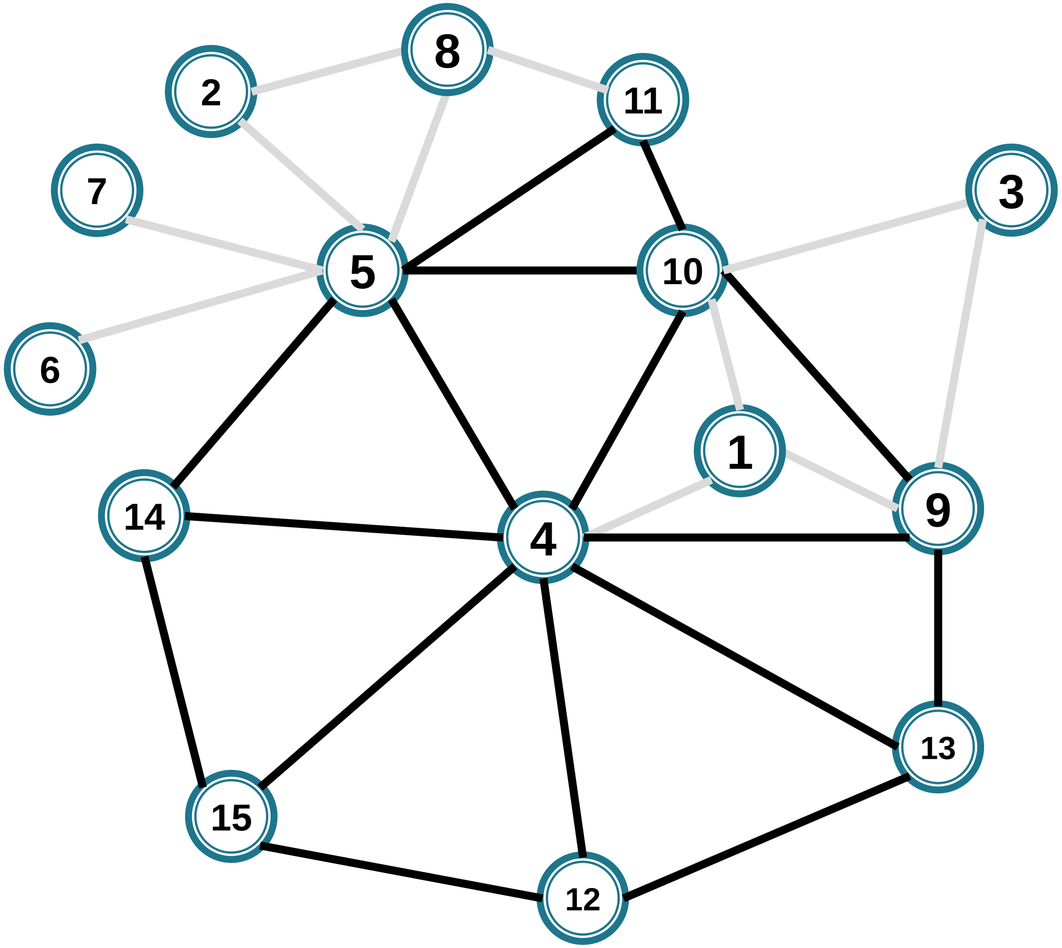
~~1~~, ~~2~~, ~~3~~, 6, 7, 8, 9, 11, 12, 13, 14, 15, 10



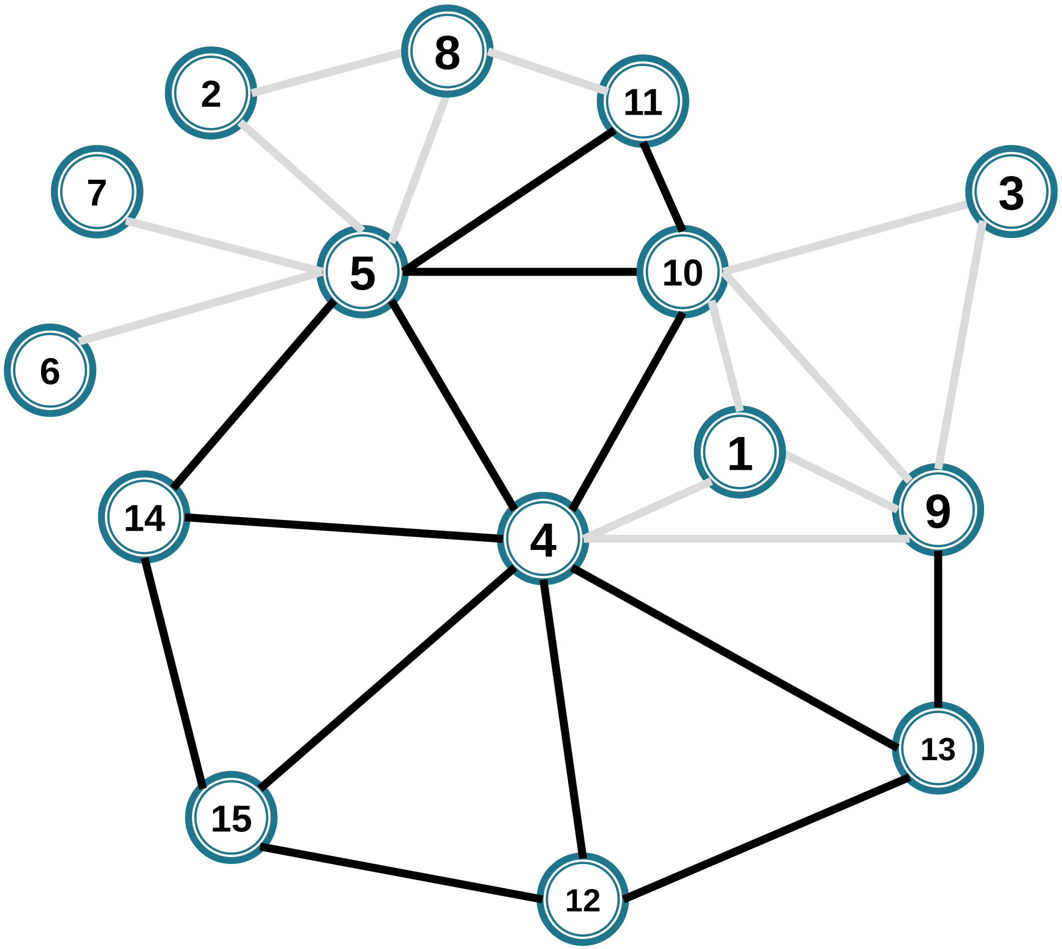
~~1~~, ~~2~~, ~~3~~, ~~6~~, 7, 8, 9, 11, 12, 13, 14, 15, 10



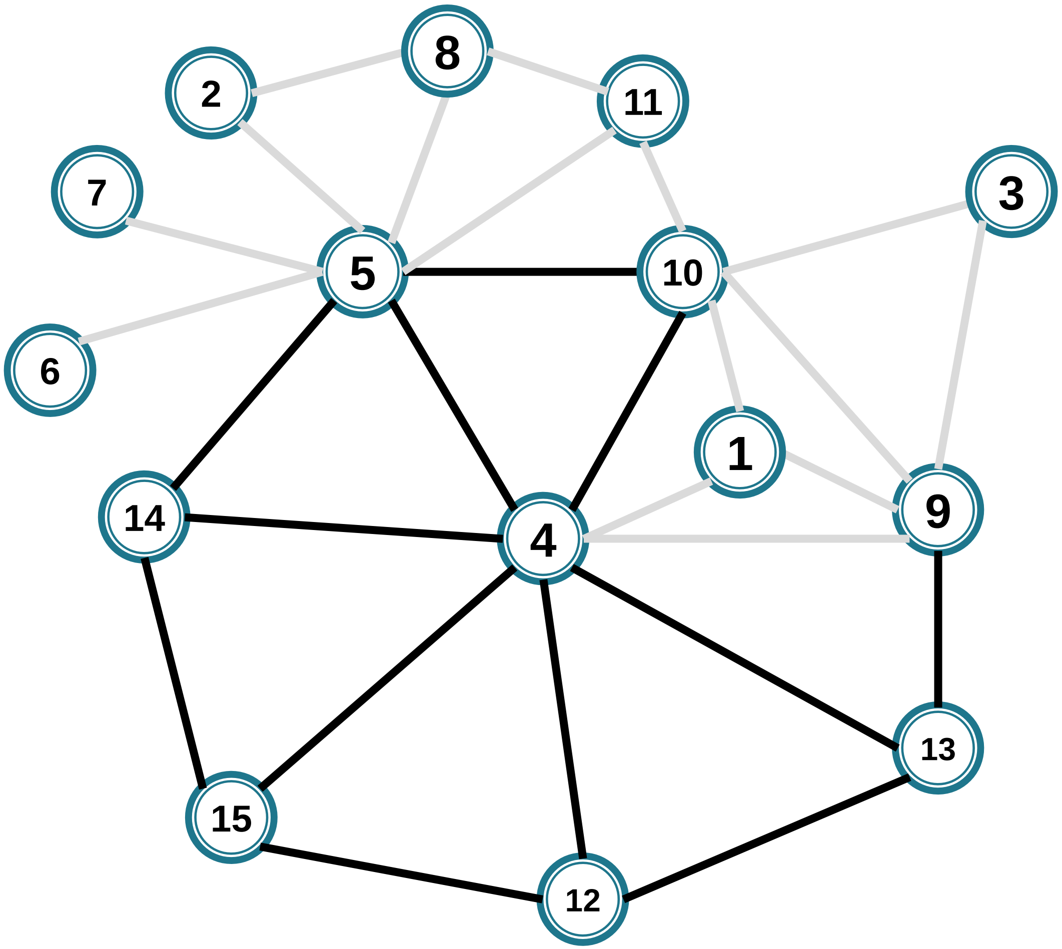
~~1~~, ~~2~~, ~~3~~, ~~6~~, ~~7~~, 8, 9, 11, 12, 13, 14, 15, 10, 5



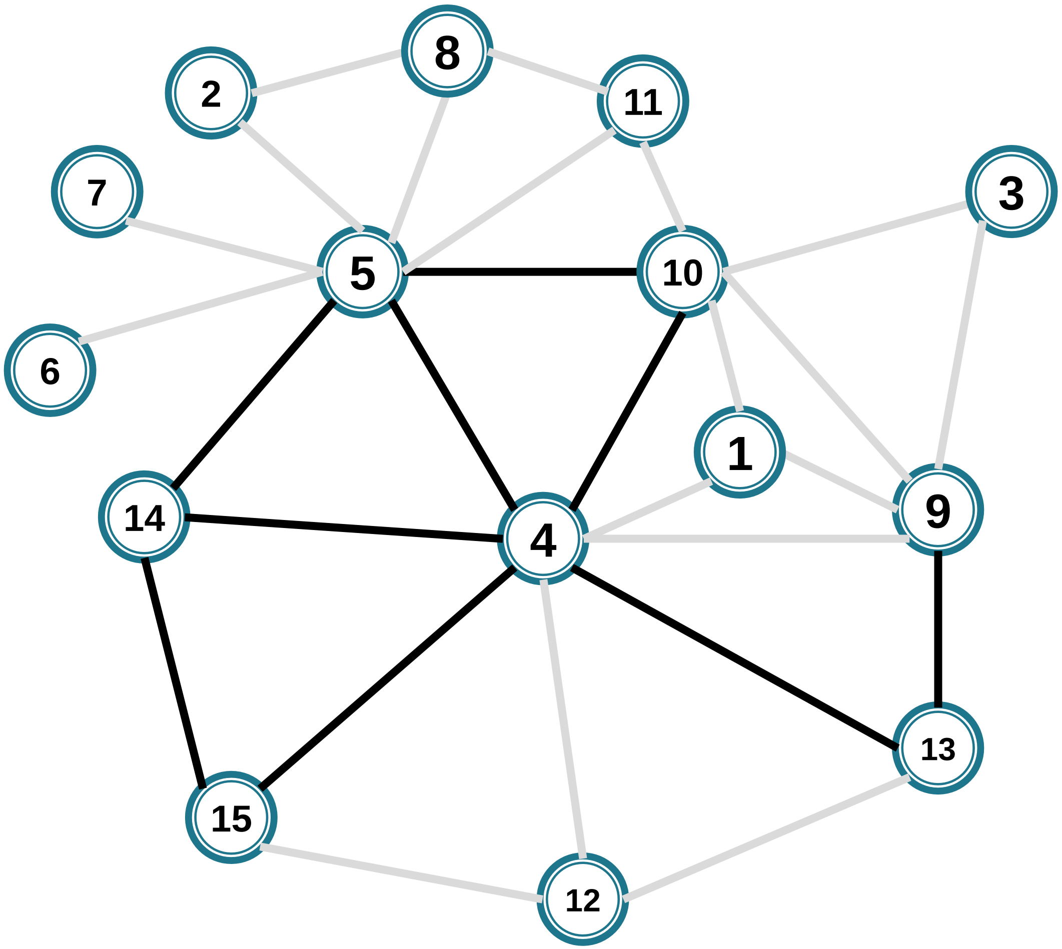
~~1~~, ~~2~~, ~~3~~, ~~6~~, ~~7~~, ~~8~~, 9, 11, 12, 13, 14, 15, 10, 5



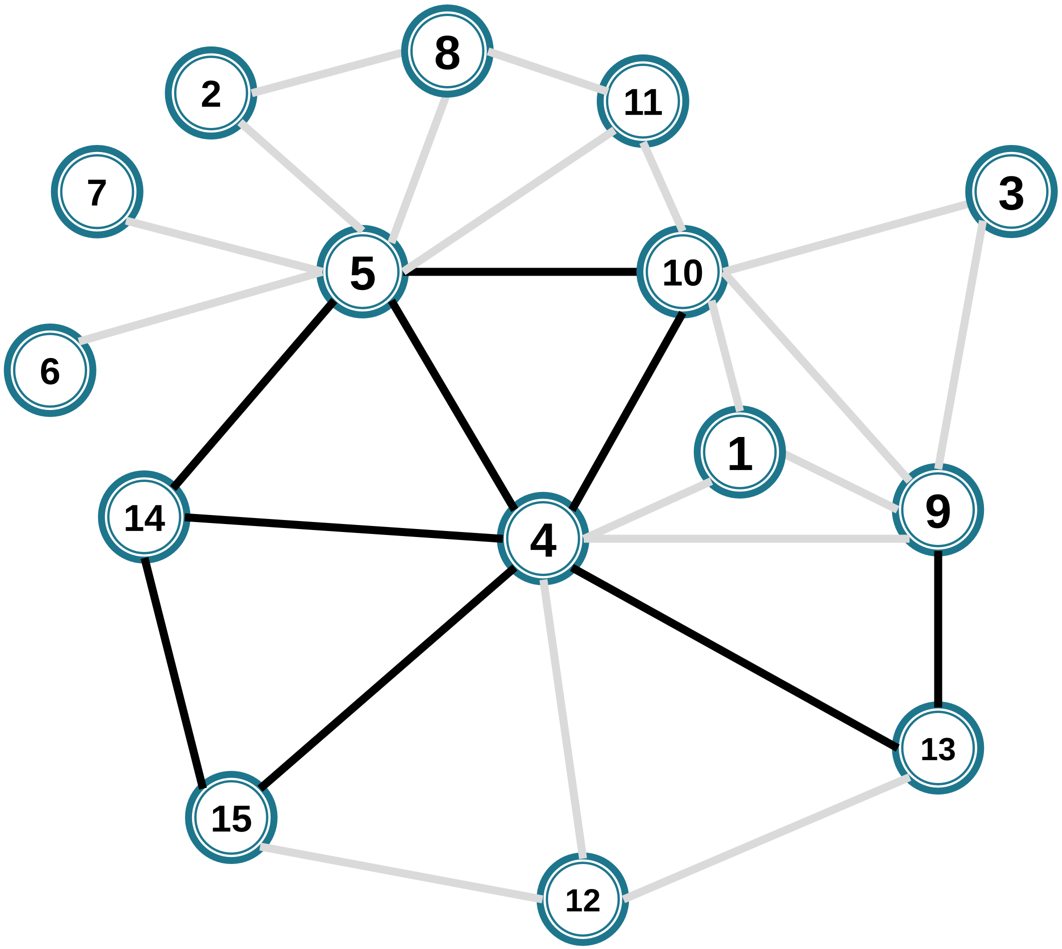
~~1~~, ~~2~~, ~~3~~, ~~6~~, ~~7~~, ~~8~~, ~~9~~, 11, 12, 13, 14, 15, 10, 5



~~1~~, ~~2~~, ~~3~~, ~~6~~, ~~7~~, ~~8~~, ~~9~~, ~~11~~, 12, 13, 14, 15, 10, 5

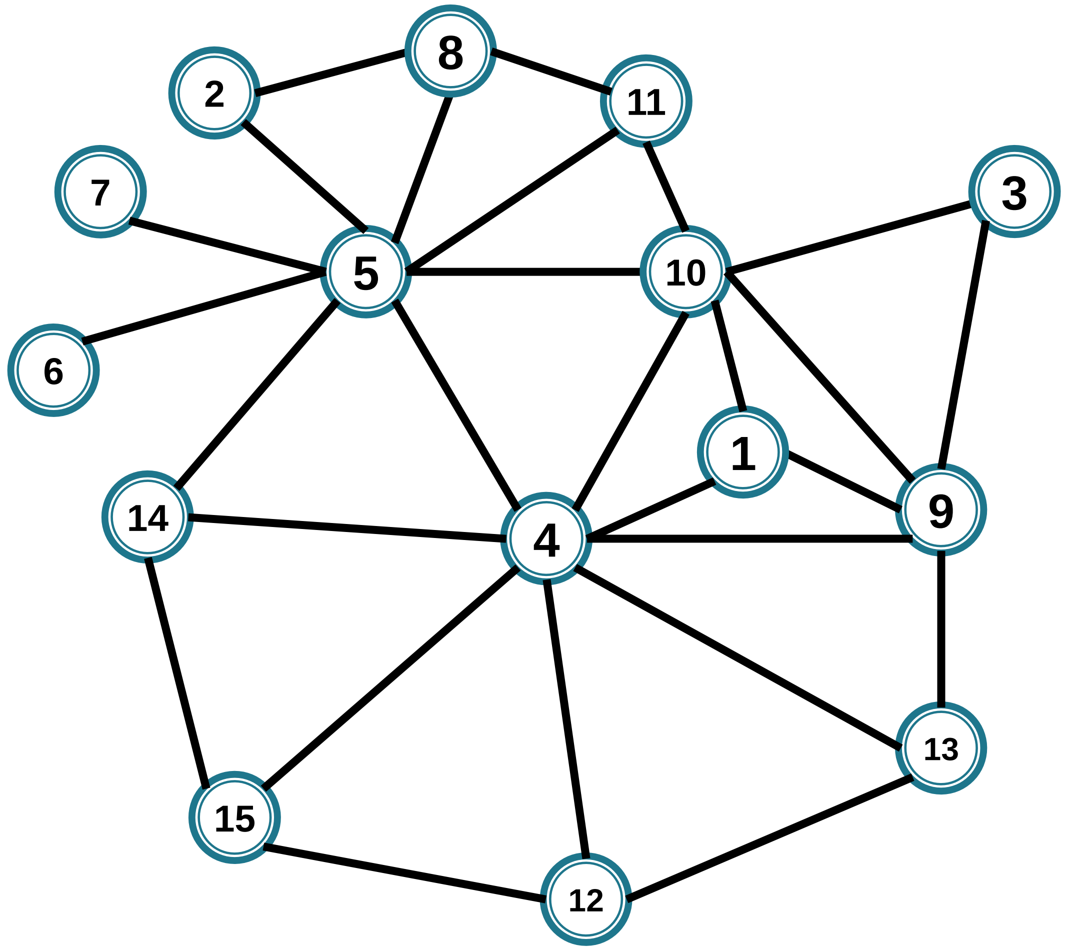


~~1~~, ~~2~~, ~~3~~, ~~6~~, ~~7~~, ~~8~~, 9, ~~11~~, ~~12~~, 13, 14, 15, 10, 5, 4



~~1~~, ~~2~~, ~~3~~, ~~6~~, ~~7~~, ~~8~~, 9, ~~11~~, ~~12~~, 13, 14, 15, 10, 5, 4

Ordinea în care se colorează vârfurile



Culoarea 1 –

Culoarea 2 –

Culoarea 3 –

Culoarea 4 –

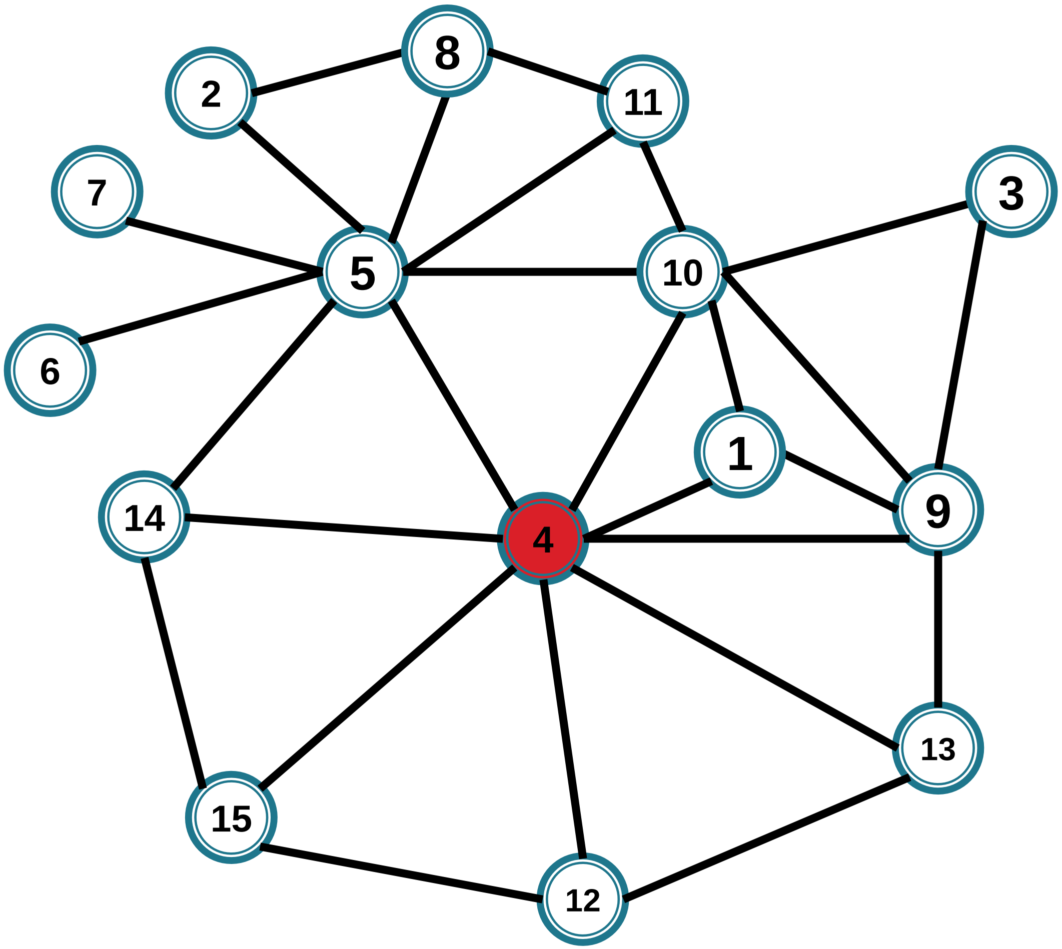
Culoarea 5 –

Culoarea 6 –

Ordinea în care se colorează vârfurile

4, 5, 10, 15, 14, 13, 12, 11, 9, 8, 7, 6, 3, 2, 1

– cu prima culoare disponibilă din cele 6 (nefolosită de un vecin)



Culoarea 1 –

Culoarea 2 –

Culoarea 3 –

Culoarea 4 –

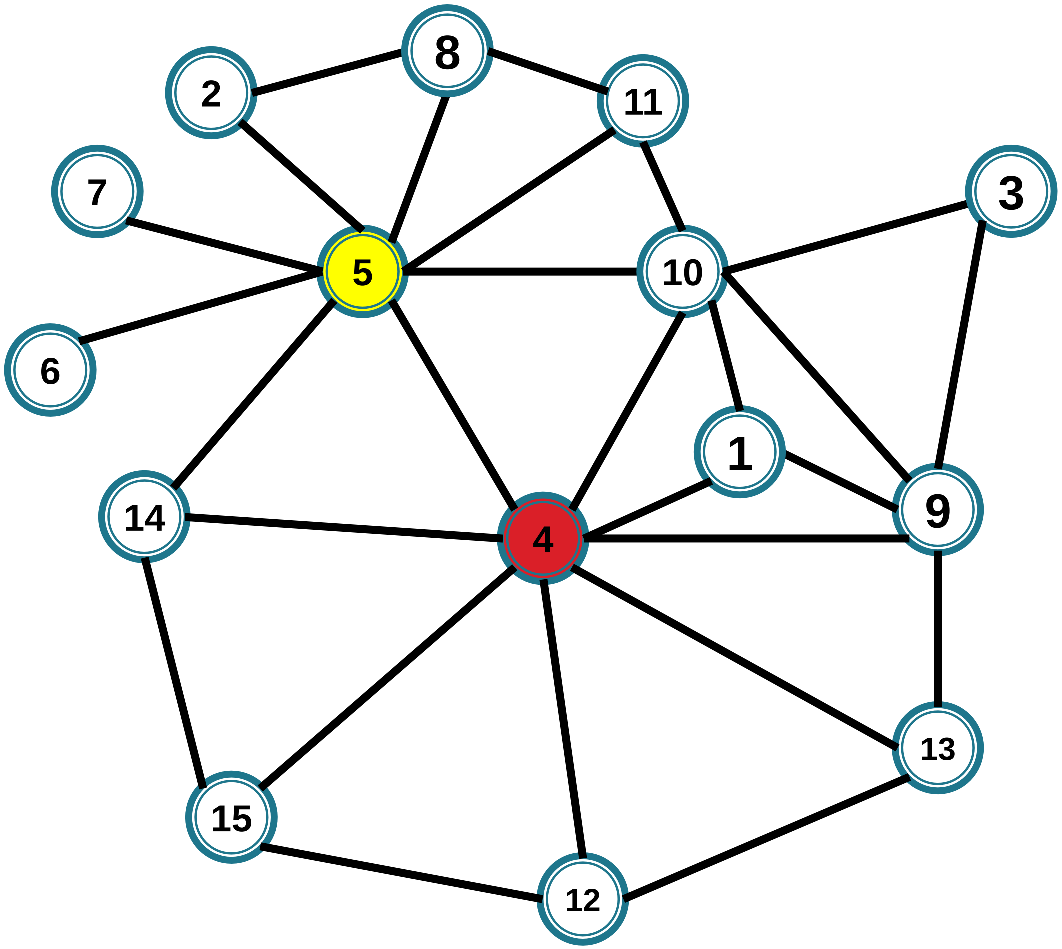
Culoarea 5 –

Culoarea 6 –

Ordinea în care se colorează vârfurile

4, 5, 10, 15, 14, 13, 12, 11, 9, 8, 7, 6, 3, 2, 1

– cu prima culoare disponibilă din cele 6 (nefolosită de un vecin)



Culoarea 1 –

Culoarea 2 –

Culoarea 3 –

Culoarea 4 –

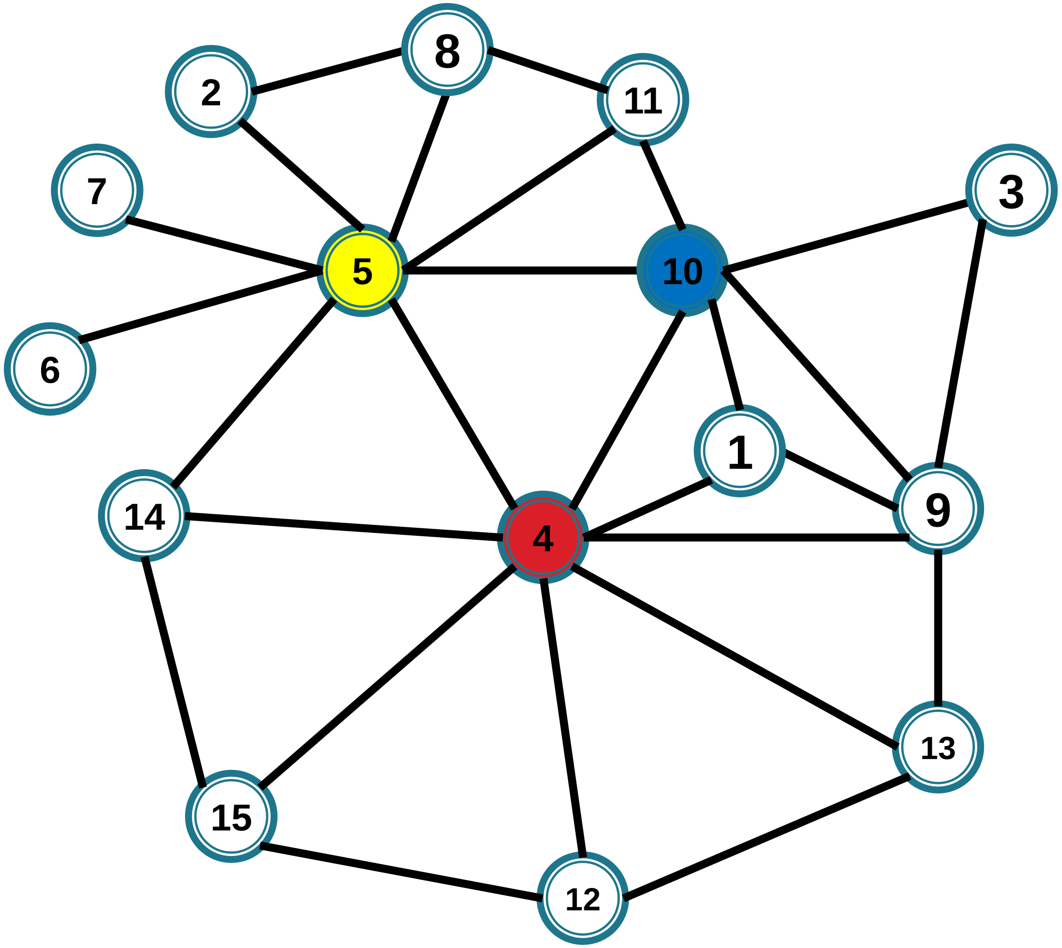
Culoarea 5 –

Culoarea 6 –

Ordinea în care se colorează vârfurile

4, 5, 10, 15, 14, 13, 12, 11, 9, 8, 7, 6, 3, 2, 1

– cu prima culoare disponibilă din cele 6 (nefolosită de un vecin)



Culoarea 1 –

Culoarea 2 –

Culoarea 3 –

Culoarea 4 –

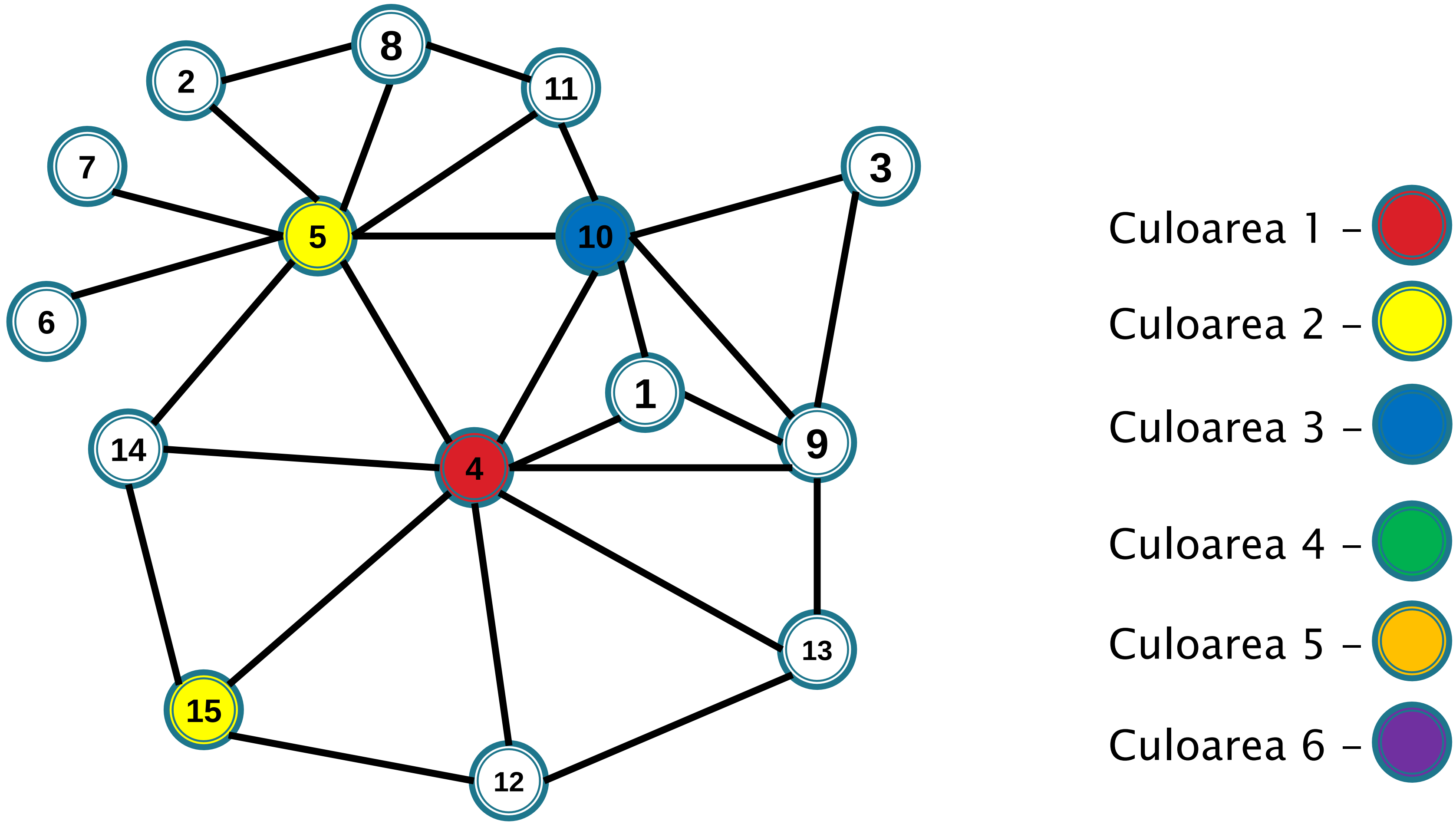
Culoarea 5 –

Culoarea 6 –

Ordinea în care se colorează vârfurile

4, 5, 10, 15, 14, 13, 12, 11, 9, 8, 7, 6, 3, 2, 1

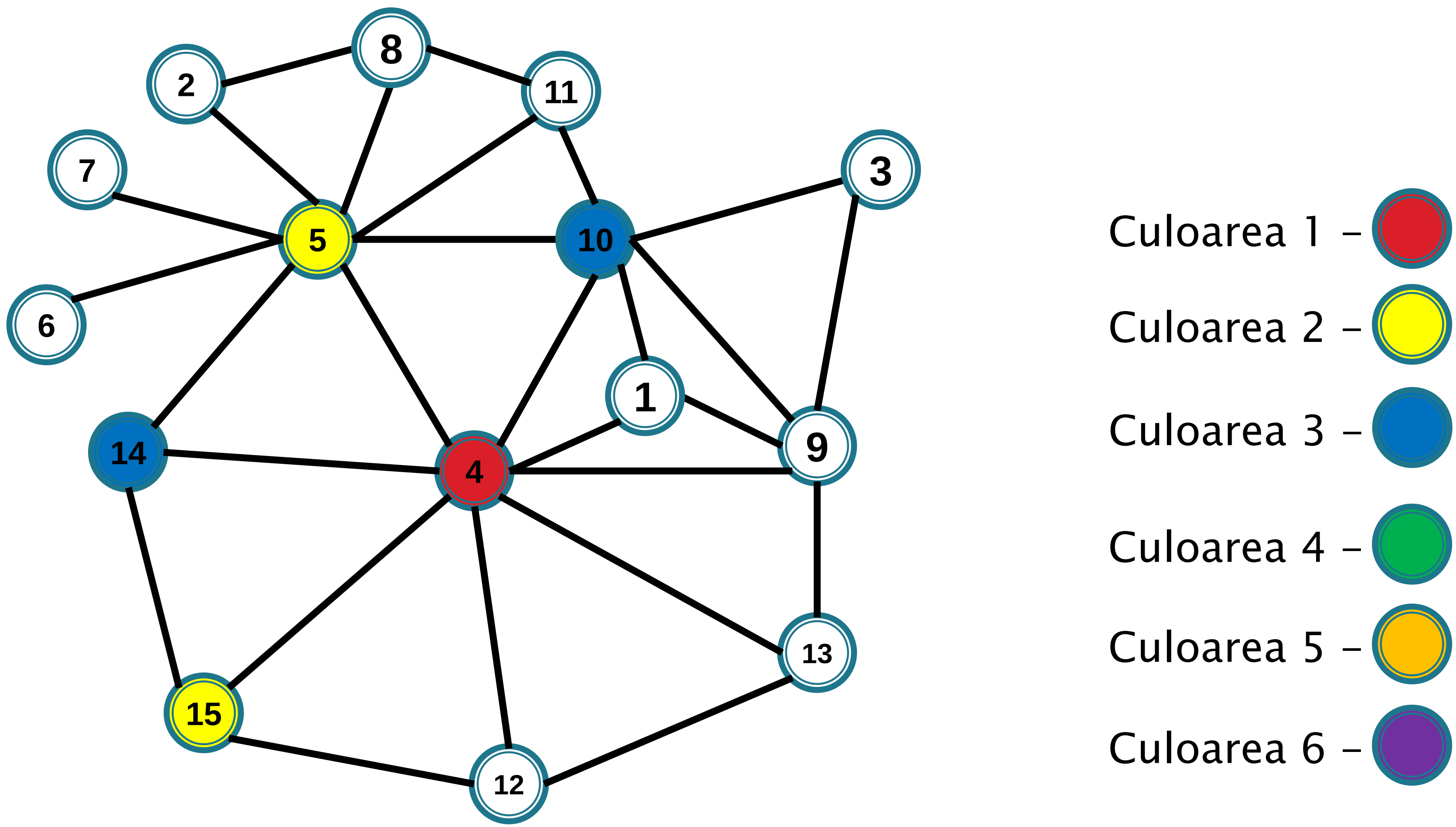
– cu prima culoare disponibilă din cele 6 (nefolosită de un vecin)



Ordinea în care se colorează vârfurile

4, 5, 10, 15, 14, 13, 12, 11, 9, 8, 7, 6, 3, 2, 1

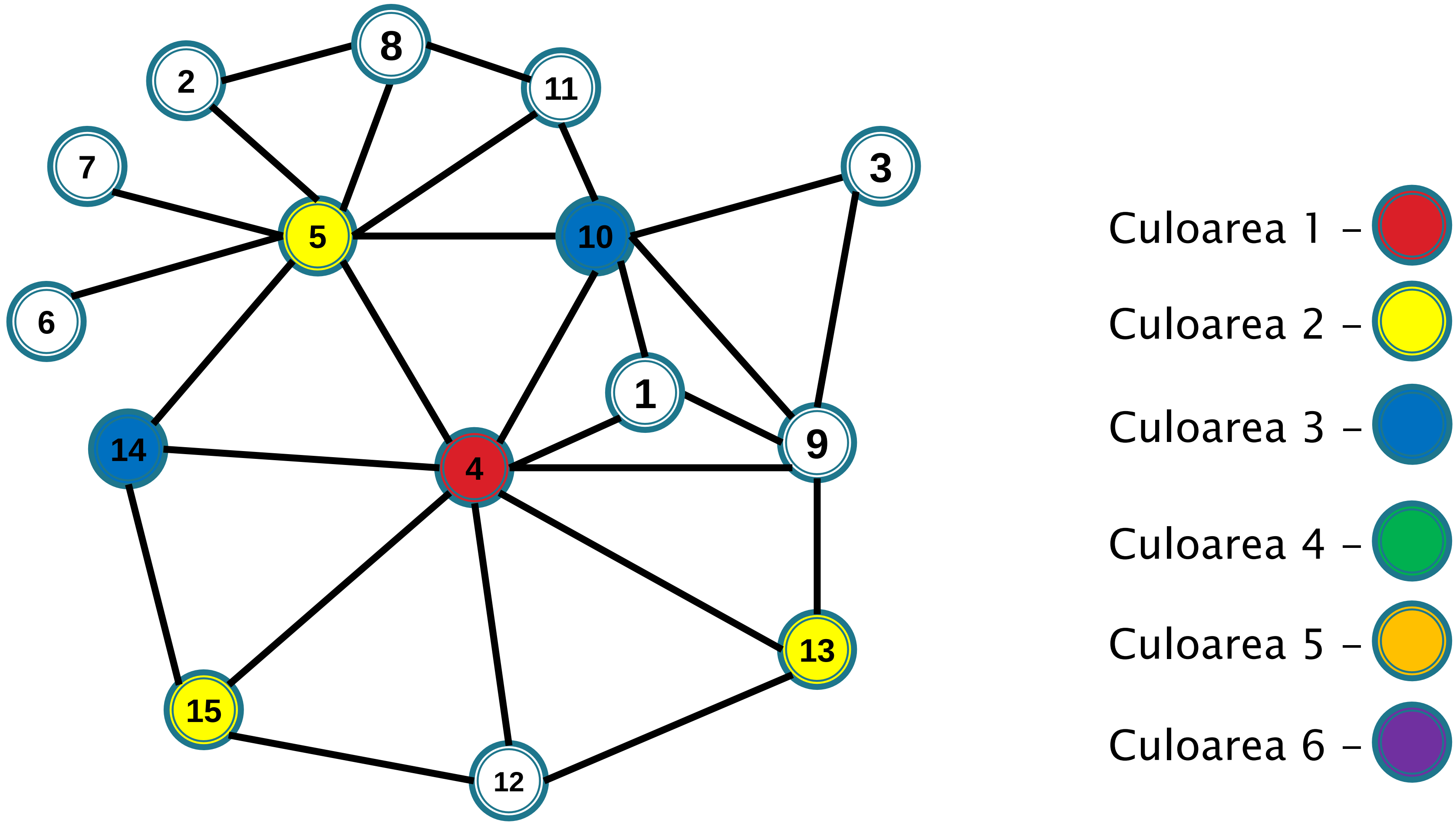
– cu prima culoare disponibilă din cele 6 (nefolosită de un vecin)



Ordinea în care se colorează vârfurile

4, 5, 10, 15, 14, 13, 12, 11, 9, 8, 7, 6, 3, 2, 1

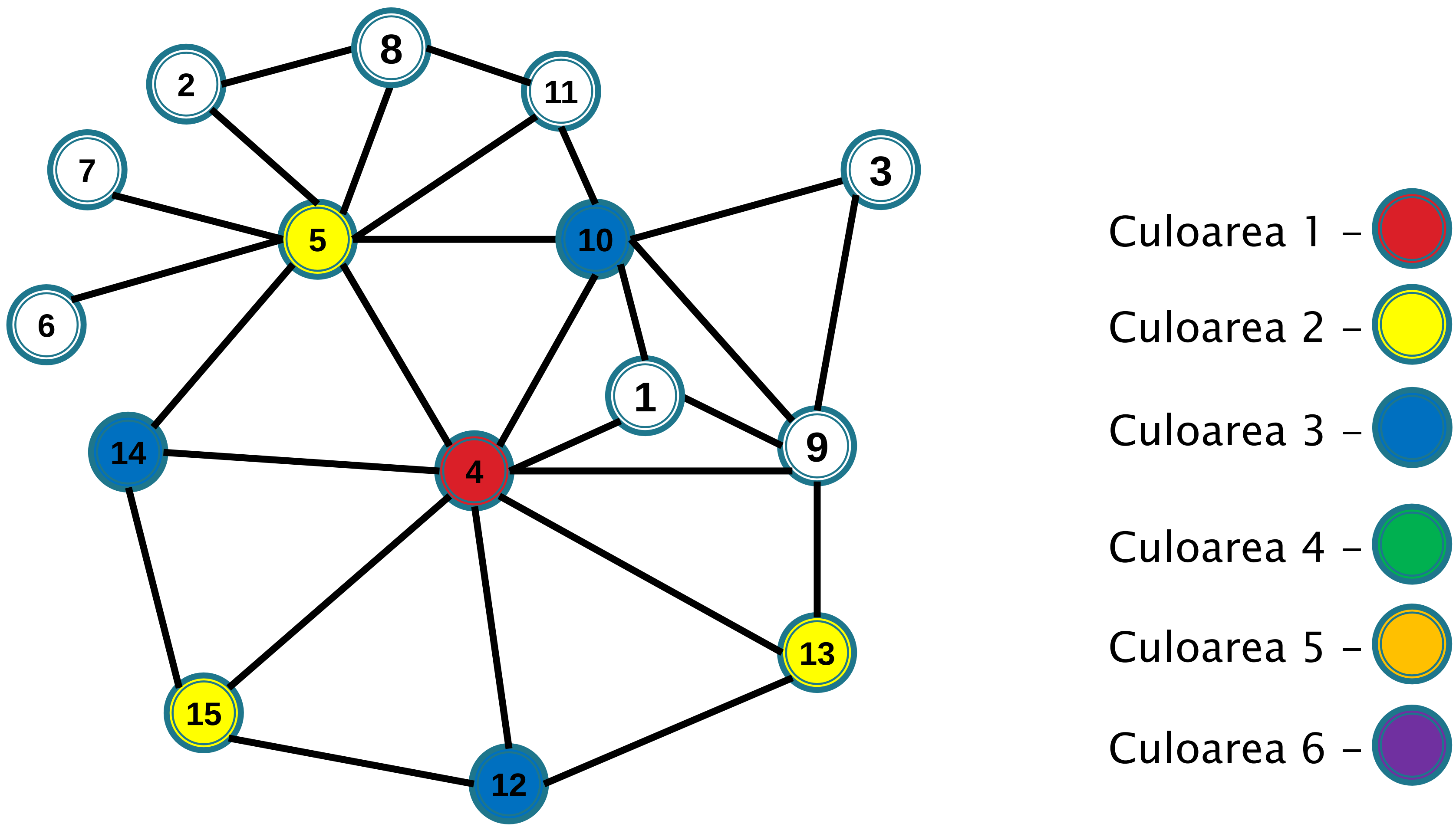
– cu prima culoare disponibilă din cele 6 (nefolosită de un vecin)



Ordinea în care se colorează vârfurile

4, 5, 10, 15, 14, 13, 12, 11, 9, 8, 7, 6, 3, 2, 1

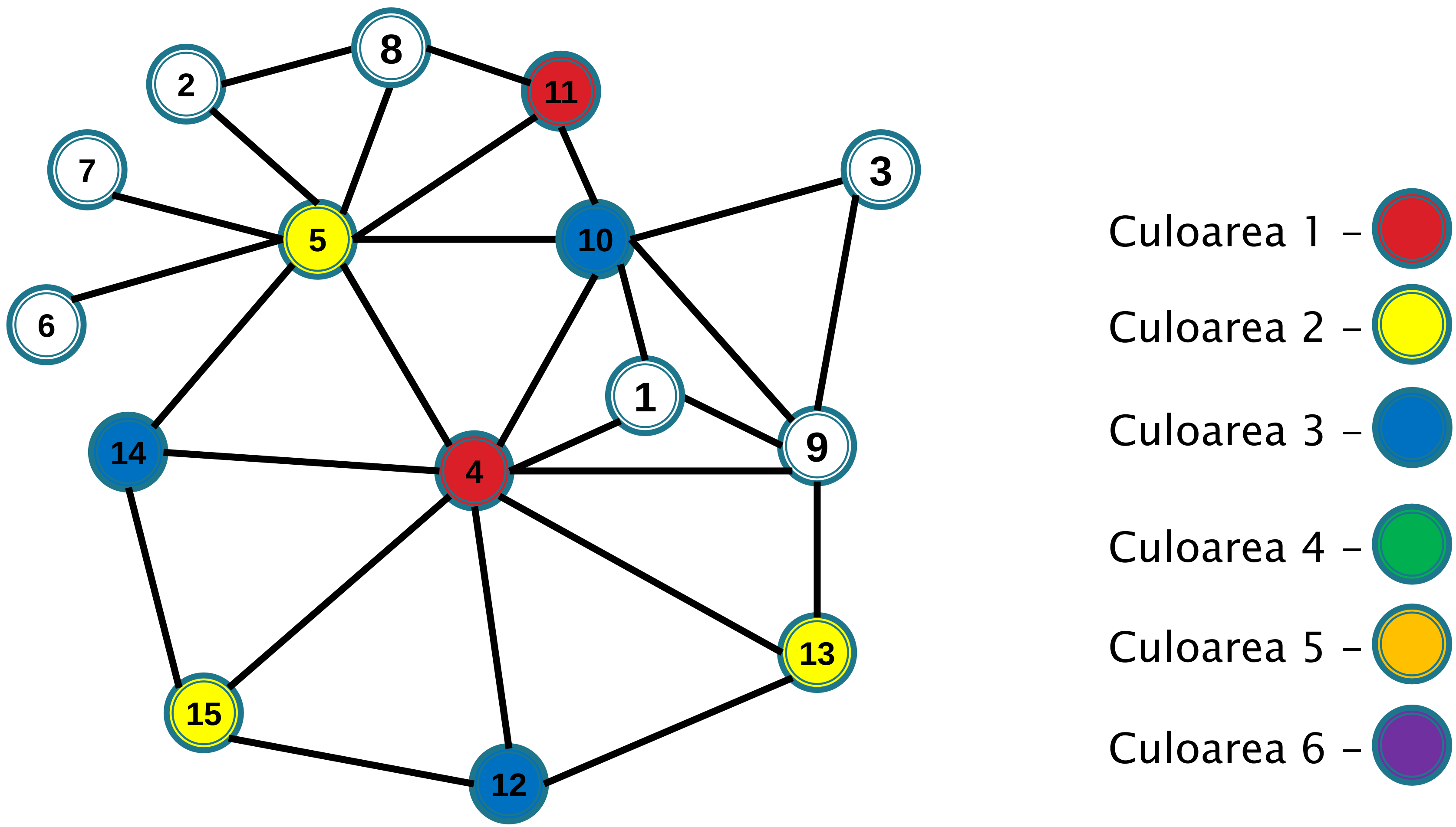
– cu prima culoare disponibilă din cele 6 (nefolosită de un vecin)



Ordinea în care se colorează vârfurile

4, 5, 10, 15, 14, 13, 12, 11, 9, 8, 7, 6, 3, 2, 1

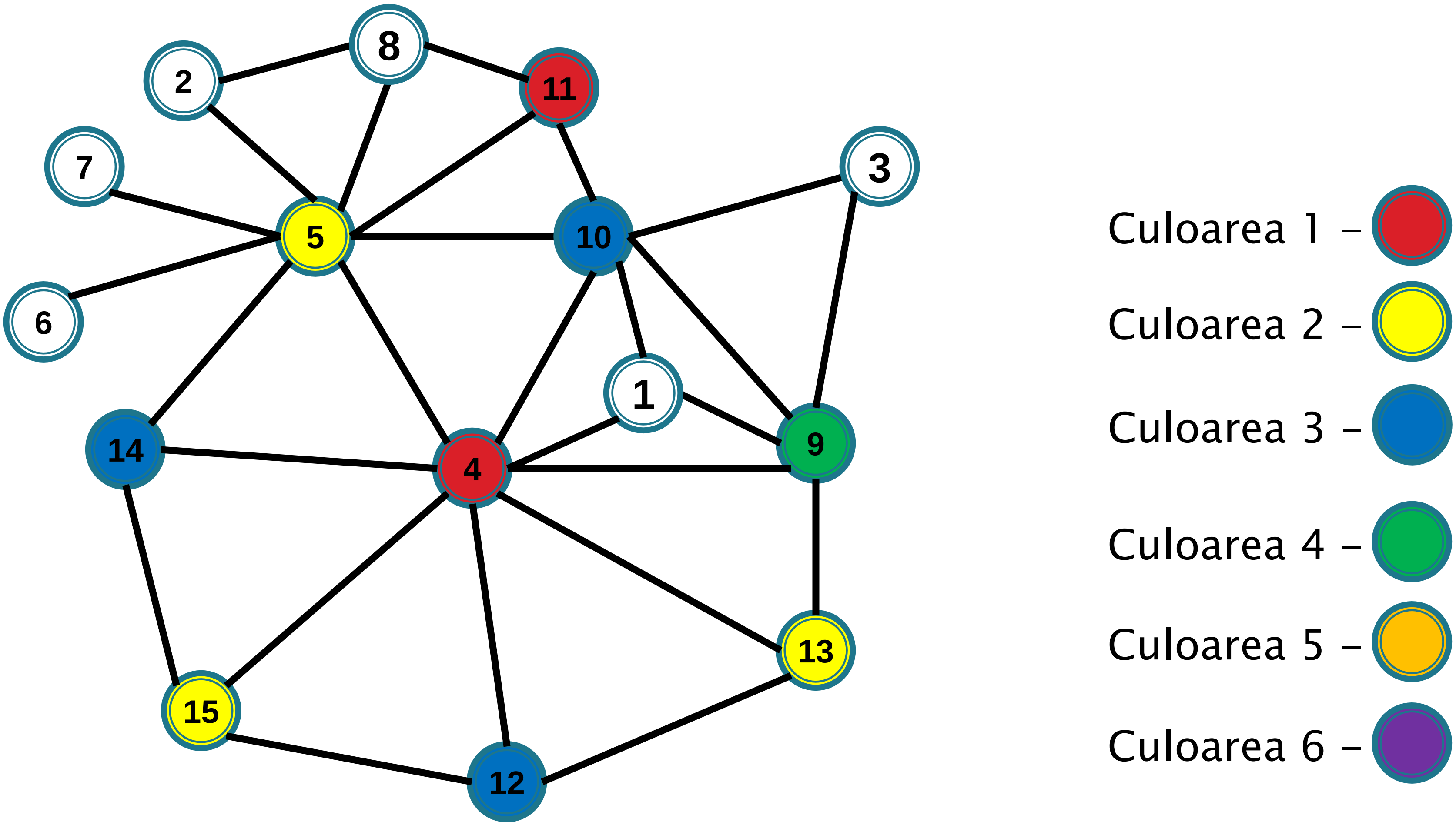
– cu prima culoare disponibilă din cele 6 (nefolosită de un vecin)



Ordinea în care se colorează vârfurile

4, 5, 10, 15, 14, 13, 12, 11, 9, 8, 7, 6, 3, 2, 1

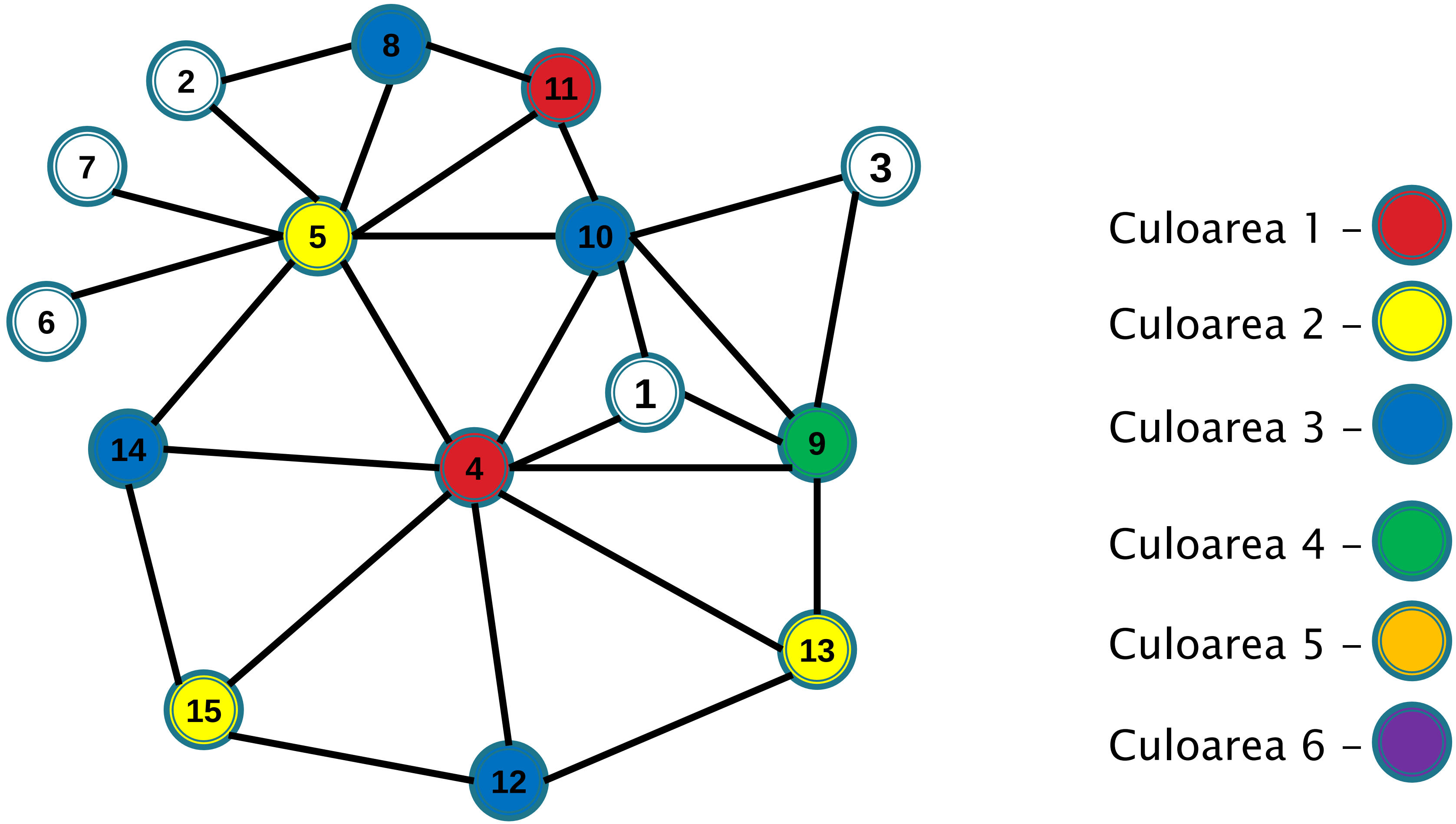
– cu prima culoare disponibilă din cele 6 (nefolosită de un vecin)



Ordinea în care se colorează vârfurile

4, 5, 10, 15, 14, 13, 12, 11, 9, 8, 7, 6, 3, 2, 1

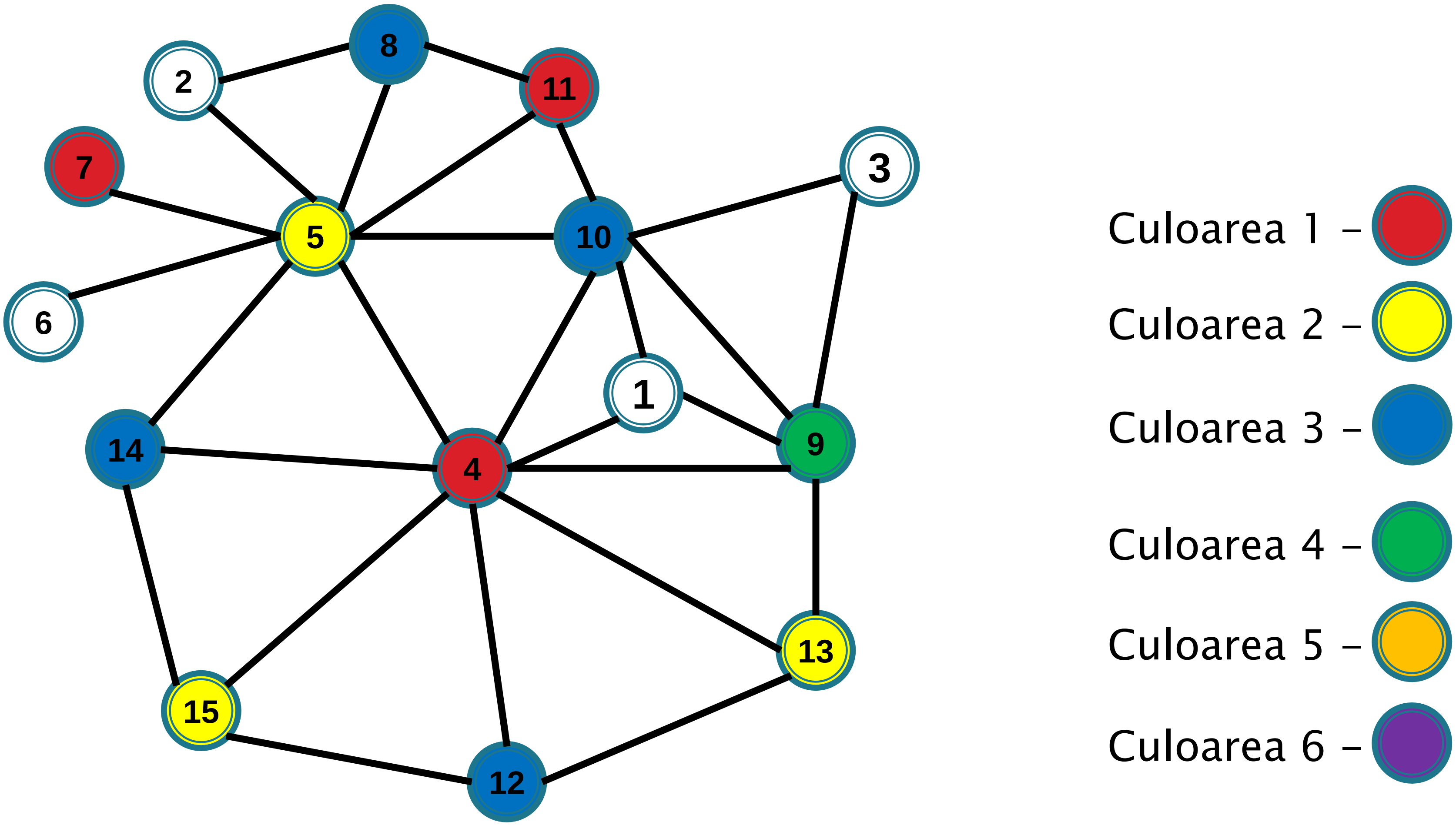
– cu prima culoare disponibilă din cele 6 (nefolosită de un vecin)



Ordinea în care se colorează vârfurile

4, 5, 10, 15, 14, 13, 12, 11, 9, 8, 7, 6, 3, 2, 1

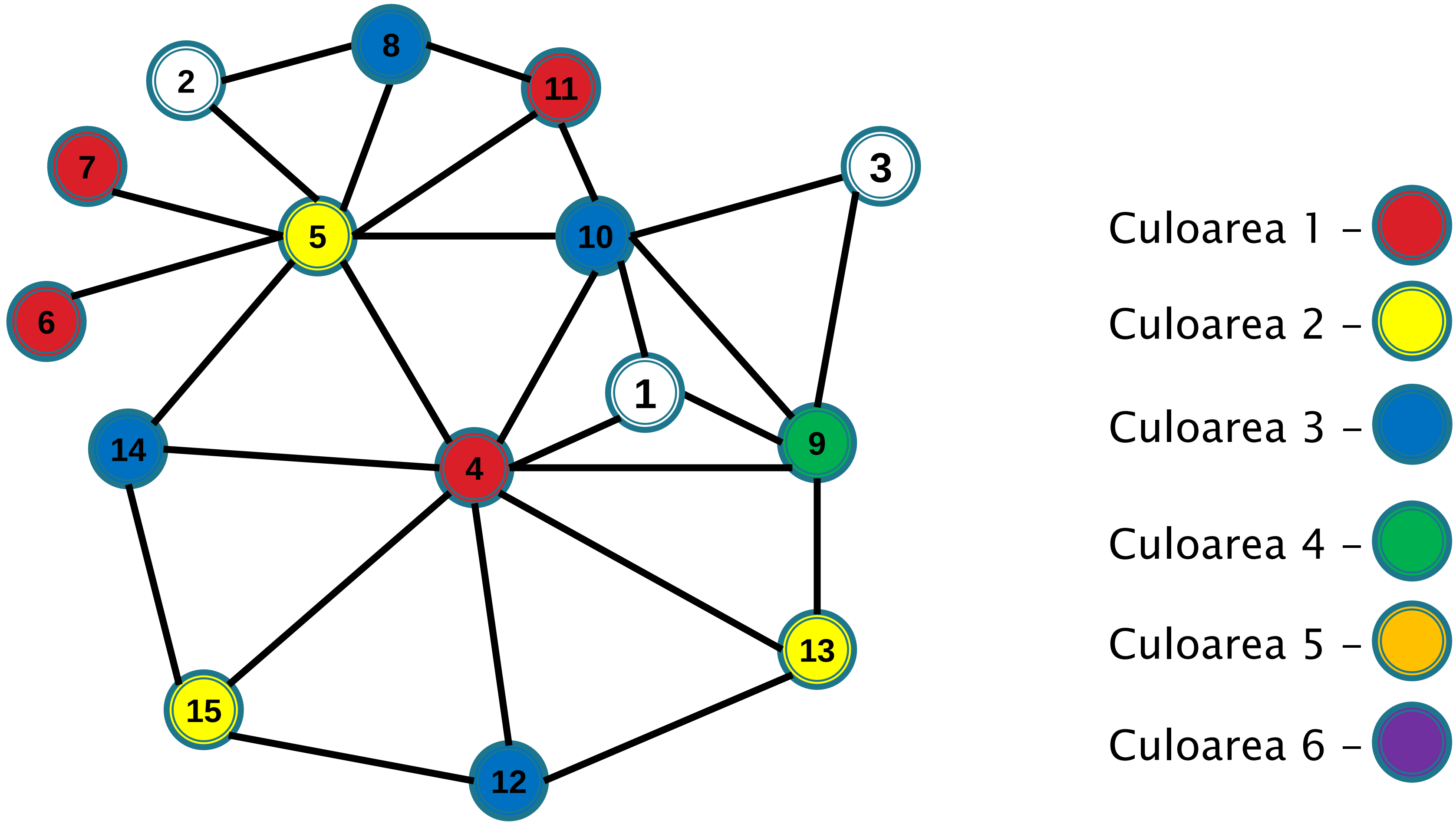
– cu prima culoare disponibilă din cele 6 (nefolosită de un vecin)



Ordinea în care se colorează vârfurile

4, 5, 10, 15, 14, 13, 12, 11, 9, 8, 7, 6, 3, 2, 1

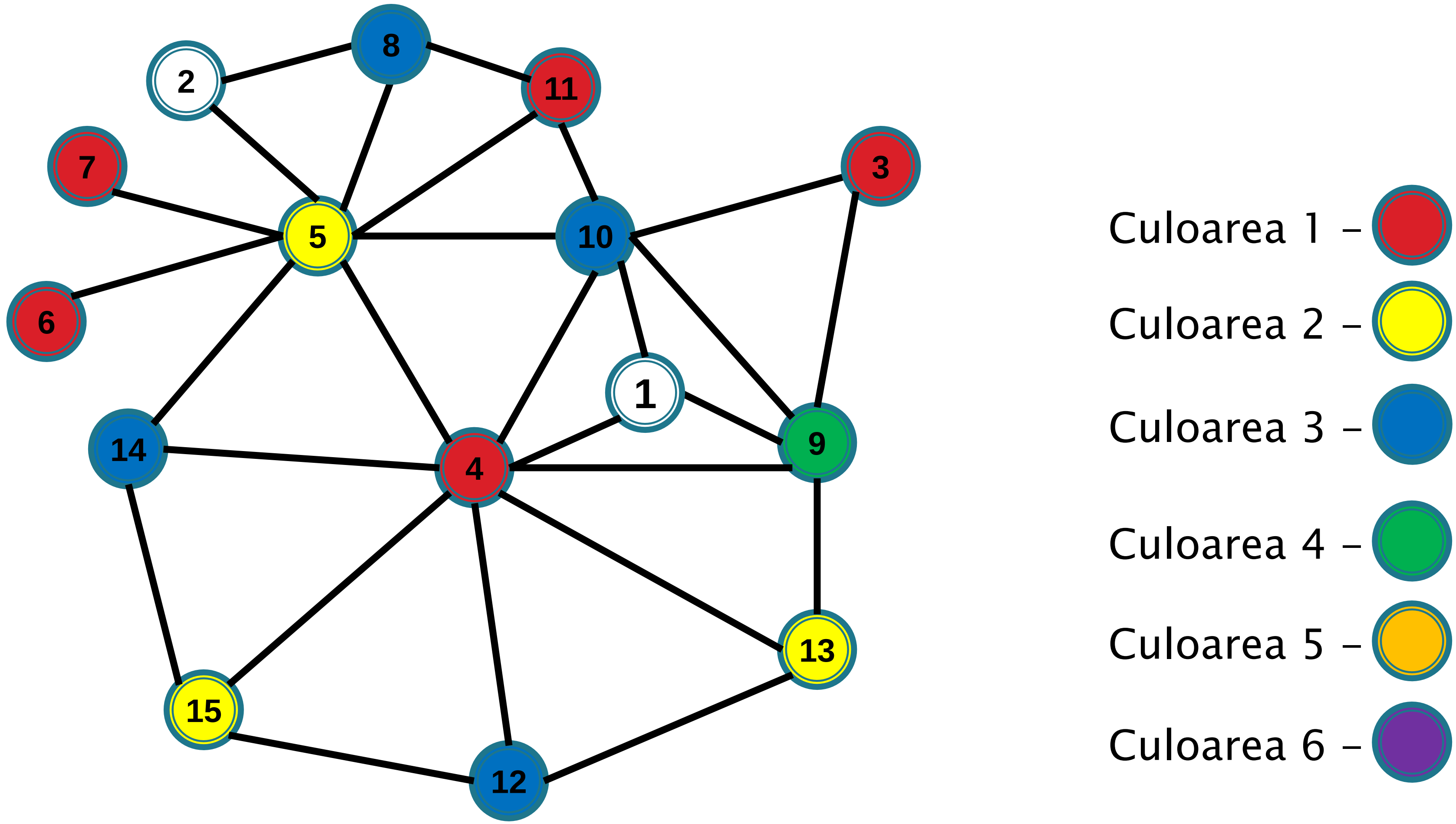
– cu prima culoare disponibilă din cele 6 (nefolosită de un vecin)



Ordinea în care se colorează vârfurile

4, 5, 10, 15, 14, 13, 12, 11, 9, 8, 7, 6, 3, 2, 1

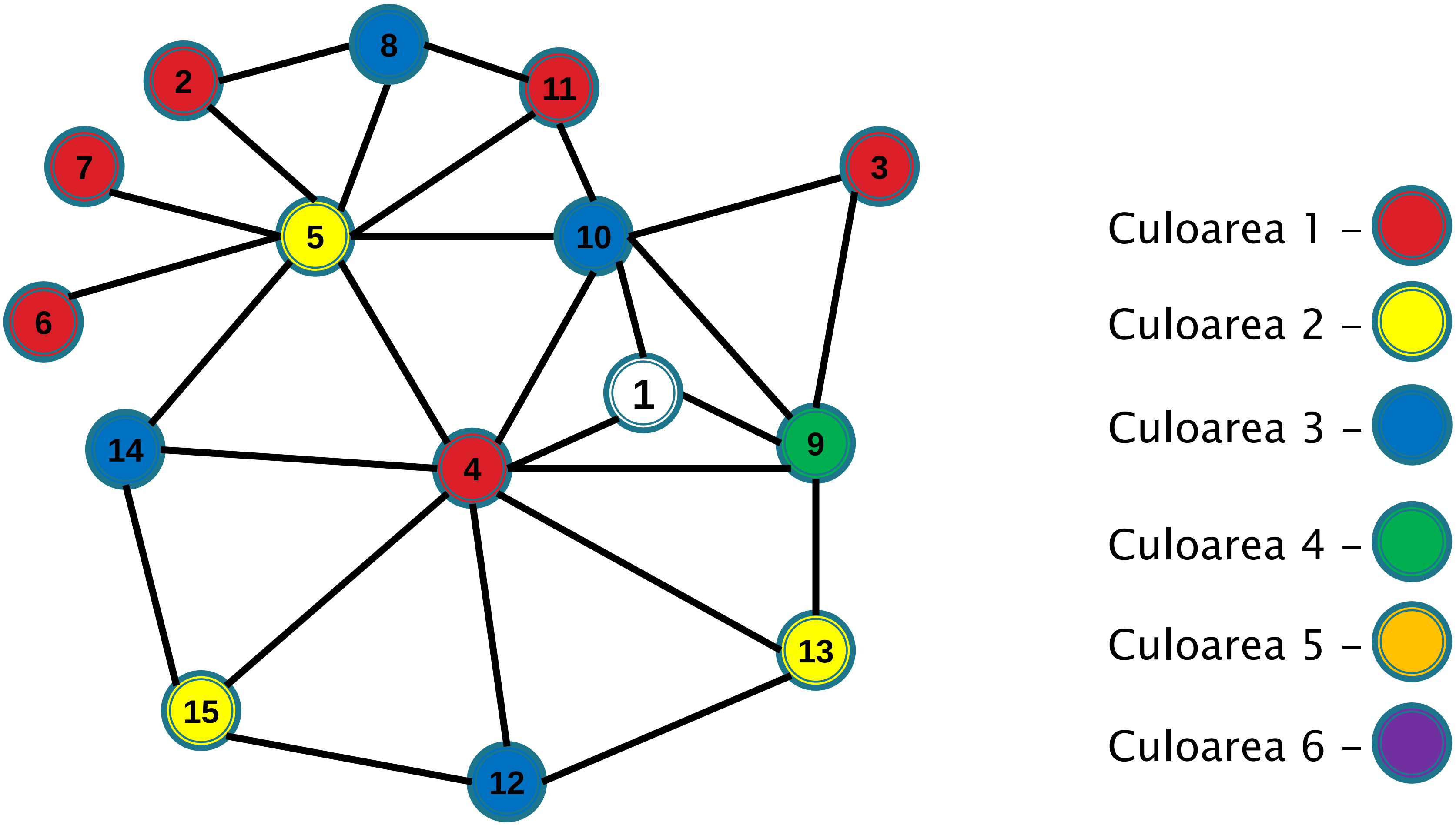
– cu prima culoare disponibilă din cele 6 (nefolosită de un vecin)



Ordinea în care se colorează vârfurile

4, 5, 10, 15, 14, 13, 12, 11, 9, 8, 7, 6, 3, 2, 1

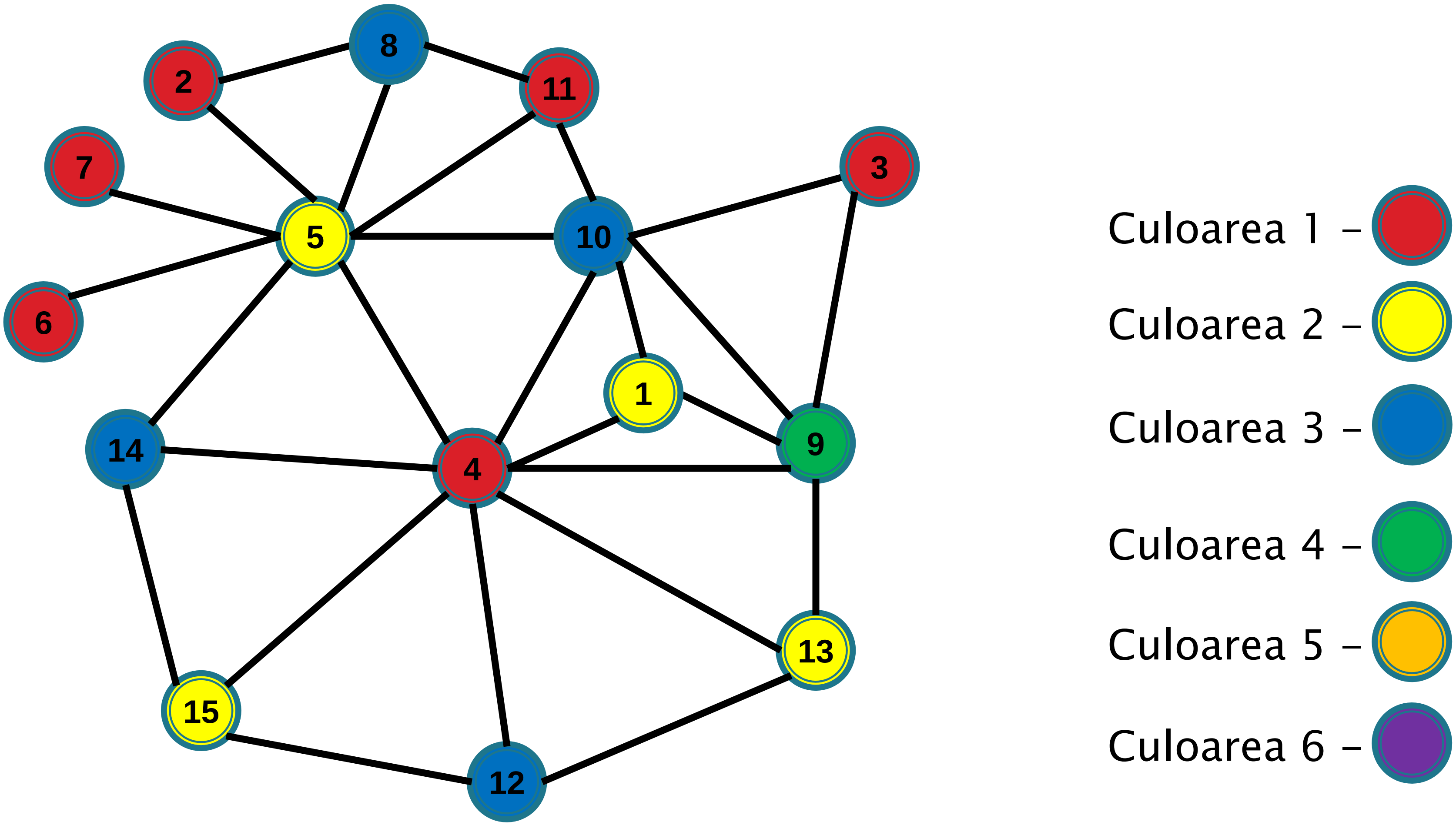
– cu prima culoare disponibilă din cele 6 (nefolosită de un vecin)



Ordinea în care se colorează vârfurile

4, 5, 10, 15, 14, 13, 12, 11, 9, 8, 7, 6, 3, 2, 1

– cu prima culoare disponibilă din cele 6 (nefolosită de un vecin)



Ordinea în care se colorează vârfurile

4, 5, 10, 15, 14, 13, 12, 11, 9, 8, 7, 6, 3, 2, 1

– cu prima culoare disponibilă din cele 6 (nefolosită de un vecin)

Colorări în grafuri oarecare

- **Versiunea iterativă a algoritmului de colorare a unui graf planar cu 6 culori se poate generaliza la următorul algoritm greedy de colorare a unui graf oarecare:**

1. **Stabilim o ordonare $v_1, \ldots v_n$ în care vor fi colorate vârfurile** (vom reveni cu exemple de strategii de ordonare, de exemplu Smallest Last – v_n= vârful cu grad minim care generalizează strategia de la colorarea cu 6 culori a grafurilor planare)

2. **Colorăm pe rând vârfurile $v_1, \ldots, v_n$ cu prima culoare diferită de culorile vecinilor deja colorați**

Colorări în grafuri oarecare

- **Versiunea iterativă a algoritmului de colorare a unui graf planar cu 6 culori se poate generaliza la următorul algoritm greedy de colorare a unui graf oarecare:**

  1. **Stabilim o ordonare $v_1,\dots v_n$ în care vor fi colorate vârfurile** (vom reveni cu exemple de strategii de ordonare, de exemplu Smallest Last – v_n= vârful cu grad minim care generalizează strategia de la colorarea cu 6 culori a grafurilor planare)

  2. **Colorăm pe rând vârfurile $v_1,\dots, v_n$ cu prima culoare diferită de culorile vecinilor deja colorați**

  Acest tip de algoritm (greedy) nu furnizează o colorare proprie cu număr minim de culori pentru orice clasă de grafuri, problema: dat un graf să se determine dacă este p-colorabil pentru un p dat fiind NP-completă

Graf planar

- **Teorema celor 5 culori**

  Orice graf planar conex este 5 -colorabil.

Demonstrație -suplimentar, v. D.R. Popescu

Colorări în grafuri

Colorări ale grafurilor

Amintim:

Computațional: Dat p, **este G p-colorabil**?

Care este p minim cu proprietatea că G este p-colorabil?
= Care este numărul cromatic al lui G?

- Este G 2-**colorabil** / graf bipartit – algoritm polinomial
- Este G 3**-colorabil** – problemă NP-completă

P/NP

- Complexitatea în timp a algoritmilor joacă un rol esenţial.

- **Un algoritm este considerat "acceptabil" numai dacă timpul său de executare este polinomial**

Nu ştim algoritm polinomial – problemă grea?

P/NP

Nu ştim algoritm polinomial – problemă grea?

P = clasa problemelor pentru care există
algoritmi polinomiali (determinişti)

P/NP

Nu ştim algoritm polinomial – problemă grea?

NP

– există algoritm polinomial pentru a testa o soluţie candidat dacă este soluţie posibilă (verificator polinomial) / există algoritm polinomial nedeterminist

⇒ o problemă NP poate fi rezolvată în timp exponenţial (considerând toate soluţiile candidat)

P/NP

Nu ştim algoritm polinomial – problemă grea?

NP

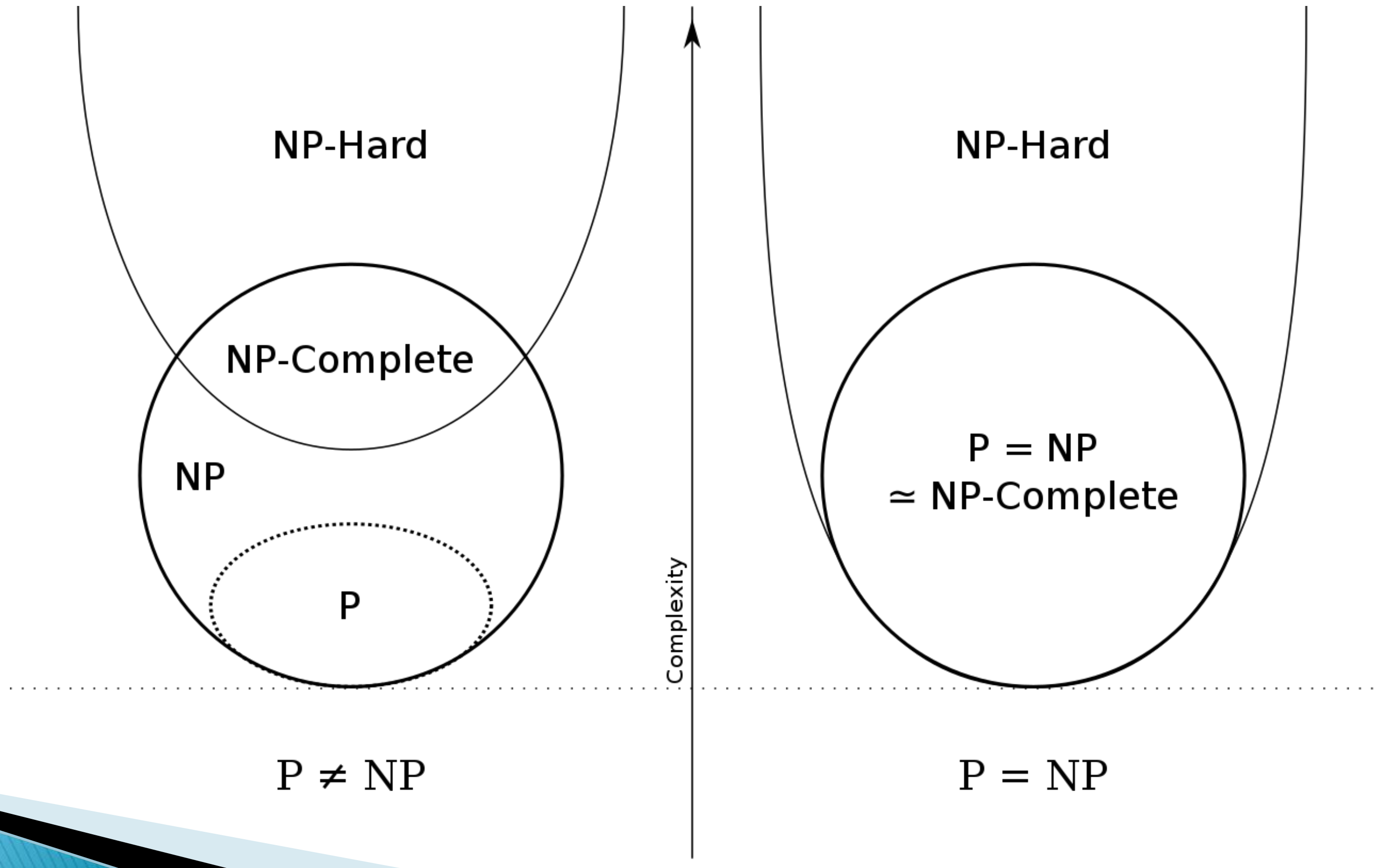
- **P ≠ NP ?**
- **Probleme NP-complete**
  - B ∈ **NP** a.î. ∀A ∈ NP, A ≤ p B (reducere în timp polinomial)
  - Dacă pentru un B se găsește algoritm polinomial, atunci P = NP
  - SAT (Cook-Levin)
- **Probleme NP-dificile (NP-hard)**
  - B a.î. ∀A ∈ NP, A ≤ p B.

P/NP

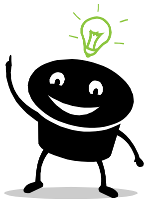
Nu ştim algoritm polinomial – problemă grea?

NP

- **P versus NP**



Metode de elaborare a algoritmilor

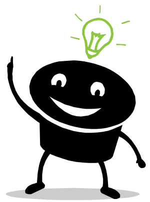


Nu ştim algoritm polinomial

- **Demonstrăm** NP - dificilă

Soluţii:

Metode de elaborare a algoritmilor



Nu ştim algoritm polinomial

- **Demonstrăm** NP – dificilă

Soluţii:

- algoritmi exponenţiali mai rapizi decât cei exhaustivi (brute force) de căutare în spaţiul soluţiilor: **Backtracking, Branch & Bound**

- **Compromis**: algoritmi mai rapizi care produc soluţii care nu sunt optime – algoritmi **euristici, aleatorii, genetici**…

Algoritmi de colorare de tip greedy (euristici)

Algoritm Greedy de coloare

- Fie $v_1, \ldots , v_n$ o ordonare a vârfurilor
- Pentru $i = 1,\ldots,n$
  - Colorează v_i cu cea mai mică culoare posibilă (care nu este culoare a unui vecin al său deja colorat)

Colorări în grafuri

Agoritm Greedy de coloare

Ordonări ale vârfurilor – strategii generale

- **SL Smallest Last** [Matula et al]: $v_1, \ldots, v_n$ astfel încât v_i este vârful de grad minim din $G-v_n-\ldots-v_{i+1}$

  – folosește cel mult 6 culori pentru grafuri planare

- **LF Largest first** [Welsh, Powell]: $v_1,\ldots,v_n$ în ordine descrescătoare după grad

Agoritm Greedy de coloare

- **Ordonări dinamice**:

- DSatur – se dă prioritate în ordonare vârfurilor care au un număr maxim de vecini deja colorați (și, in caz de egalitate, celor cu gradul cel mai mare)
  - optim pentru grafuri bipartite

Colorări în grafuri

Agoritm Greedy de coloare

- **Repetarea algoritmului pe ordonări diferite**:

repetă în timpul avut la dispoziție:

- generează aleator o ordonare a vârfurilor

- colorează G folosind algoritmul Greedy pentru această ordonare

- dacă colorarea obținută folosește un număr mai mic de culori decât cea mai bună găsită până acum, memorează această colorare ca fiind cea mai bună

Colorări în grafuri

Algoritm Greedy de coloare

- Complexitate?

Colorări în grafuri

Algoritm Greedy de coloare

- **Complexitate**?
  - $O(n+m)$ – determinarea primei culori disponibile pentru v – ordin $d(v)$

Colorări în grafuri

Algoritm Greedy de coloare

- Câte culori folosește maxim? (=> limită superioară pentru numărul cromatic)
- Cât de mare poate fi diferența între numărul de culori folosite de Algoritmul Greedy de colorare și numărul cromatic? Sunt clase de grafuri pentru care avem egalitate?

Colorări în grafuri

Algoritm Greedy de coloare

- Ordonarea contează, în funcție de ea numărul de culori poate diferi

Exemplul 1:

Colorări în grafuri

Algoritm Greedy de coloare

- Ordonarea contează, în funcție de ea numărul de culori poate diferi

Exemplul 1:

Ordinea 1, 2, 3, 4:

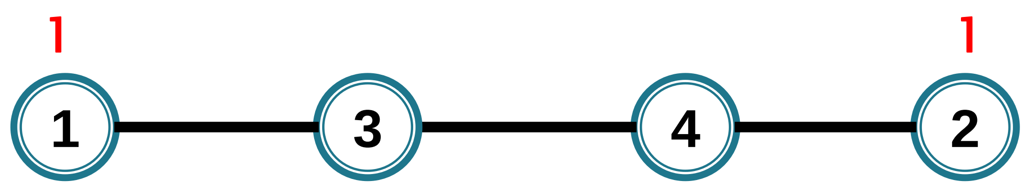
Colorări în grafuri

Algoritm Greedy de coloare

- Ordonarea contează, în funcție de ea numărul de culori poate diferi

Exemplul 1:

Ordinea 1, 2, 3, 4:



Colorări în grafuri

Algoritm Greedy de coloare

- Ordonarea contează, în funcție de ea numărul de culori poate diferi

Exemplul 1:

Ordinea 1, 2, 3, 4:

Colorări în grafuri

Algoritm Greedy de coloare

- Ordonarea contează, în funcție de ea numărul de culori poate diferi

Exemplul 1:

Ordinea 1, 2, 3, 4:

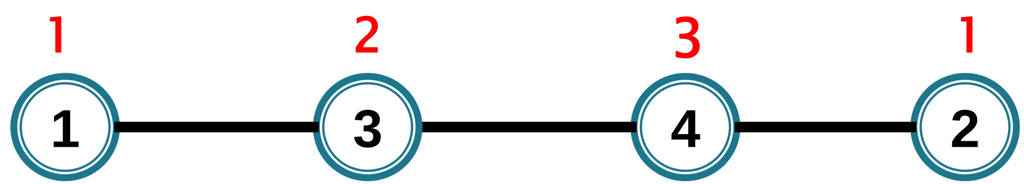
Colorări în grafuri

Algoritm Greedy de coloare

- Ordonarea contează, în funcție de ea numărul de culori poate diferi

Exemplul 1:

Ordinea 1, 2, 3, 4:

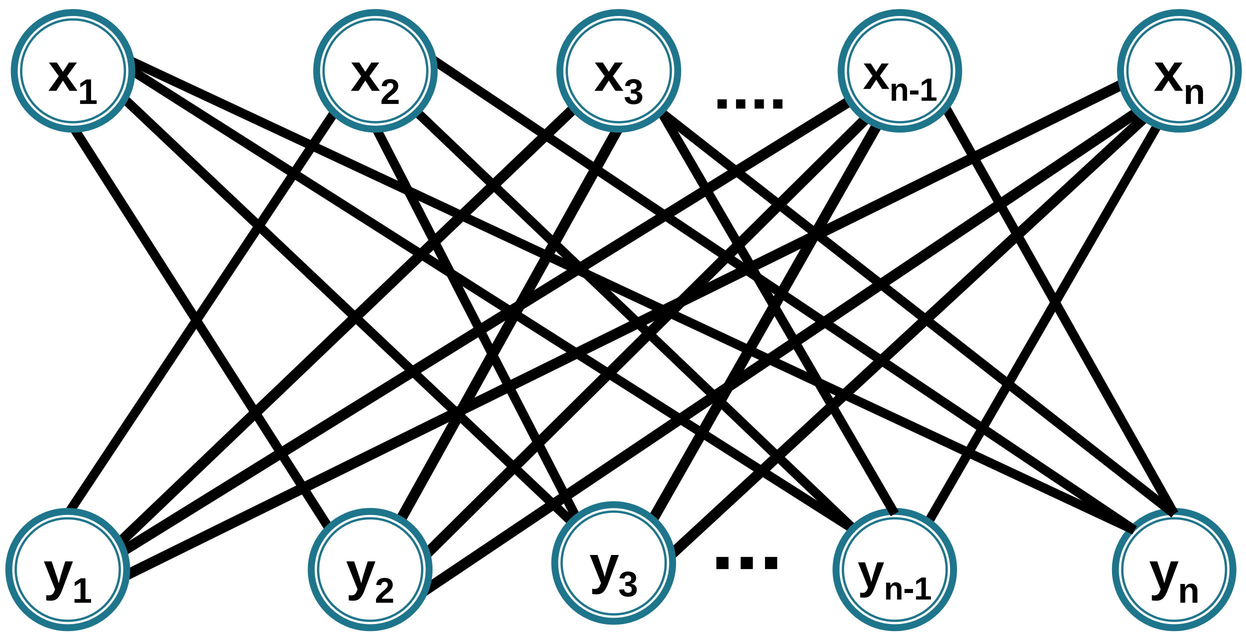


Ordinea 1, 3, 4, 2:

Colorări în grafuri

Algoritm Greedy de coloare

Exemplul 2: – Graful $G = K_{n,n}$ fără muchiile x_iy_i



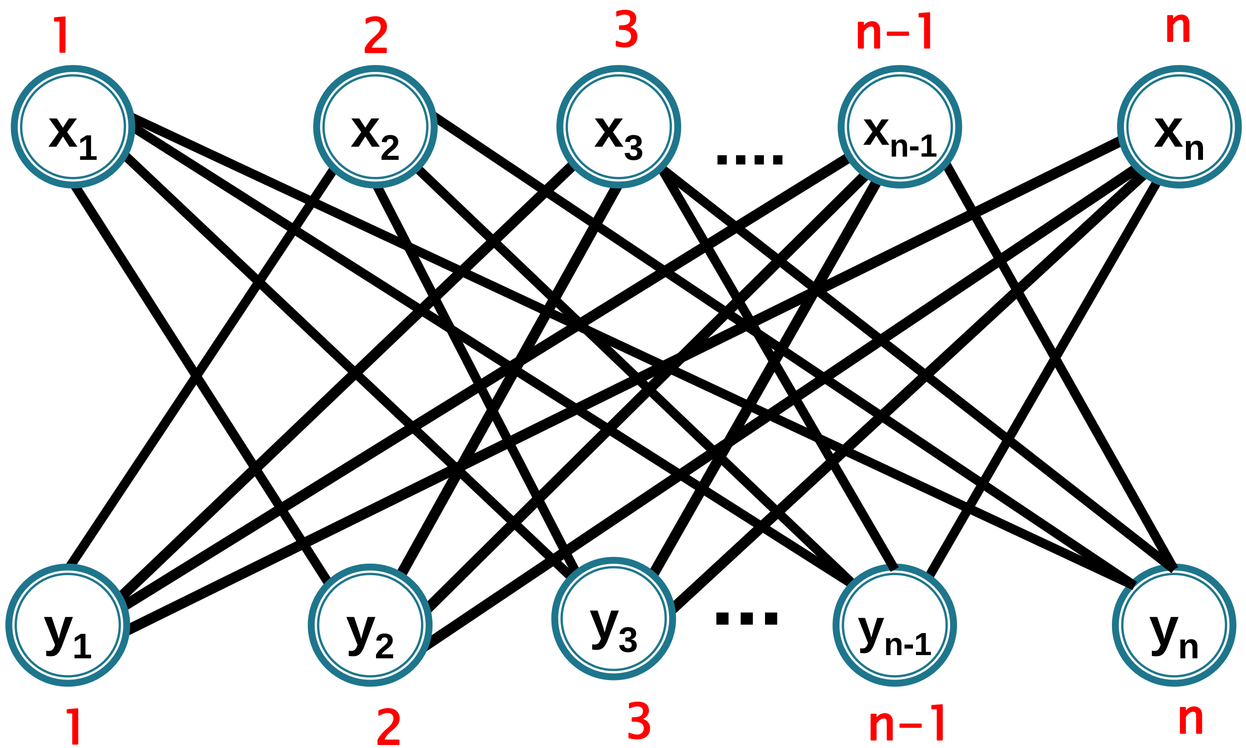
Ordinea $x_1, y_1, x_2, y_2, x_3, y_3, \ldots, x_n, y_n$: ? culori

$$\chi(G) = 2$$

Colorări în grafuri

Algoritm Greedy de coloare

Exemplul 2: – Graful G = $K_{n,n}$ fără muchiile $x_i y_i$



Ordinea $x_1, y_1, x_2, y_2, x_3, y_3, \ldots, x_n, y_n$: $n = \frac{|V(G)|}{2}$ **culori**

$\chi(G) = 2$

Algoritm Greedy de coloare

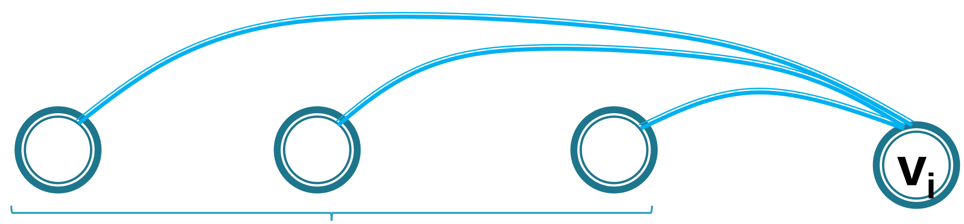
- $N_{greedy}(G,\{v_1,\ldots,v_n\})$ = numărul de culori folosit de Algoritmul Greedy de colorare pentru G considerând vârfurile în ordinea $v_1,\ldots,v_n$
- $\Delta(G)$ = gradul maxim al lui G

Colorări în grafuri

Algoritm Greedy de coloare

- $N_{greedy}(G,\{v_1,\dots,v_n\}) \leq \Delta(G) + 1$

=> Algoritmul Greedy folosește cel mult $\Delta(G) + 1$ culori



v_i are cel mult $\Delta(G)$ vecini deja colorați când se alege culoare lui

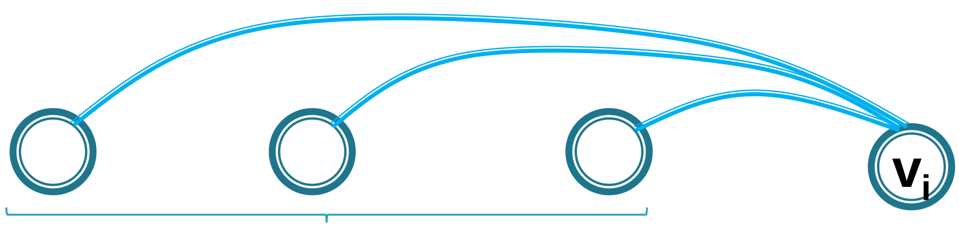
Colorări în grafuri

Algoritm Greedy de coloare

Mai precis:

- $N_{greedy}(G,\{v_1,\ldots,v_n\}) \leq \max\{\min(d(v_i)+1, i) \mid i=1,..,n\}$

$$\leq \Delta(G) + 1$$



v_i are $d(v_i)$ vecini și sunt cel mult i-1 vârfuri deja colorate =>
v_i are cel mult $\min\{d(v_i), i-1\} \leq \Delta(G)$ vecini deja colorați când se alege culoare lui

Limite pentru numărul cromatic

Proprietate

$$\chi(G) \leq \Delta(G) + 1$$

(Demonstrație: $\chi(G) \leq \mathbf{N_{greedy}(G,\{v_1,\ldots,v_n\})}$)

Colorări în grafuri

Limite pentru numărul cromatic

Proprietate

$$\chi(G) \leq \Delta(G) + 1$$

(Demonstrație: $\chi(G) \leq \mathbf{N_{greedy}(G,\{v_1,\ldots,v_n\})}$)

Observații:

1. $\chi(K_n) = n = \Delta(K_n) + 1$
2. $\chi(C_n) = 3 = \Delta(C_n) + 1$ pentru n impar
3. $\chi(S_n) = 2, \ \Delta(S_n) = n - 1$ (graful stea)

Colorări în grafuri

Limite pentru numărul cromatic

Teorema lui Brooks

$\chi(G) \le \Delta(G)$ pentru G care nu este graf complet sau ciclu impar

(Demonstrație- suplimentar, v. bibliografie)

Colorări în grafuri

Limite pentru numărul cromatic

Proprietate

$\chi(G) \geq \omega(G)$ =numărul clică = cardinalul maxim al unei clici (subgraf complet) din G

Colorări în grafuri

Limite pentru numărul cromatic

Proprietate

$\chi(G) \geq \omega(G)$ = numărul clică = cardinalul maxim al unei clici (subgraf complet) din G

Demonstrație – vârfurile dintr-o clică trebuie demonstrate diferit

Colorări în grafuri

Algoritm Greedy de coloare

- Am demonstrat la grafuri planare: există o ordonare a vârfurilor pentru care Algoritmul Greedy furnizează o colorare cu cel mult 6 culori (Teorema celor 6 culori)
- Exista ordonări + clase de grafuri pentru care Algoritmul Greedy de colorare este optim?

Colorări în grafuri

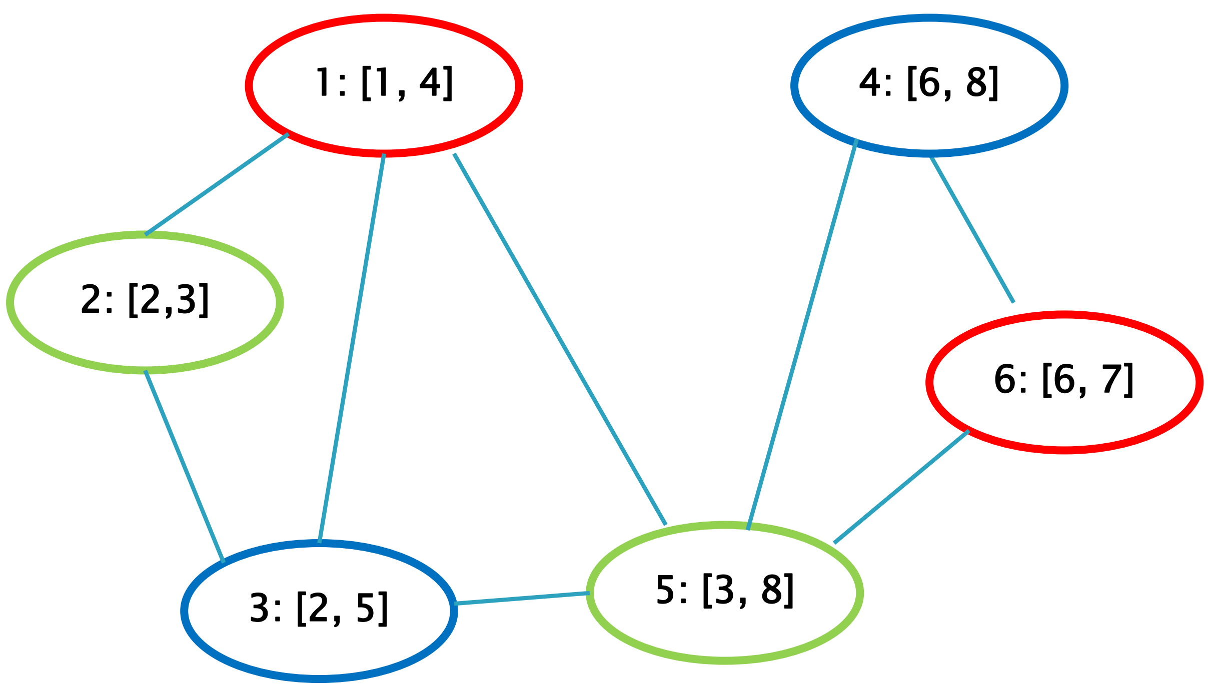
Algoritm Greedy de coloare

- Exista ordonări + clase de grafuri pentru care Algoritmul Greedy de colorare este optim?
  - Grafuri bipartite, ordonare dată de o parcurgere (sau orice ordonare în care orice vârf v_i este adiacent cu cel puțin un vârf v_j cu $j<i$)
  - Grafuri interval

Colorări în grafuri

Algoritm Greedy de coloare

- **Grafuri interval**
  - Fiecare vârf i – asociat unui interval $[a_i,b_i]$
  - Muchii – între intervale care se intersectează



Colorări în grafuri

Algoritm Greedy de colorare

- **Grafuri interval**

Proprietate: Fie G un graf interval si $v_1,\ldots,v_n$ o ordonare a vârfurilor sale după extremitatea inițială a intervalelor corespunzătoare.

Avem $\chi(G) = \mathbf{N_{greedy}(G,\{v_1,\ldots,v_n\})}$

(algoritmul greedy este optim pentru această ordonare a vârfurilor, furnizând o colorare cu număr minim de culori = numărul cromatic al lui G)

Colorări în grafuri

Algoritm Greedy de coloare

- **Grafuri interval**

Proprietate: Fie G un graf interval si $v_1,\ldots,v_n$ o ordonare a vârfurilor sale **după extremitatea inițială** a intervalelor corespunzătoare.

Avem $\chi(G) = \mathbf{N_{greedy}(G,\{v_1,\ldots,v_n\})}$

Demonstrație – Programarea Algoritmilor –

culoare = sala în care distribui intervalul

Algoritm Greedy de coloare

- **Grafuri interval**

Algoritm O(n log(n)) de determinare a numărului cromoatic $\chi(G)$ si a unei $\chi(G)$-colorări - vezi metoda Greedy, problema Partiționării intervalelor