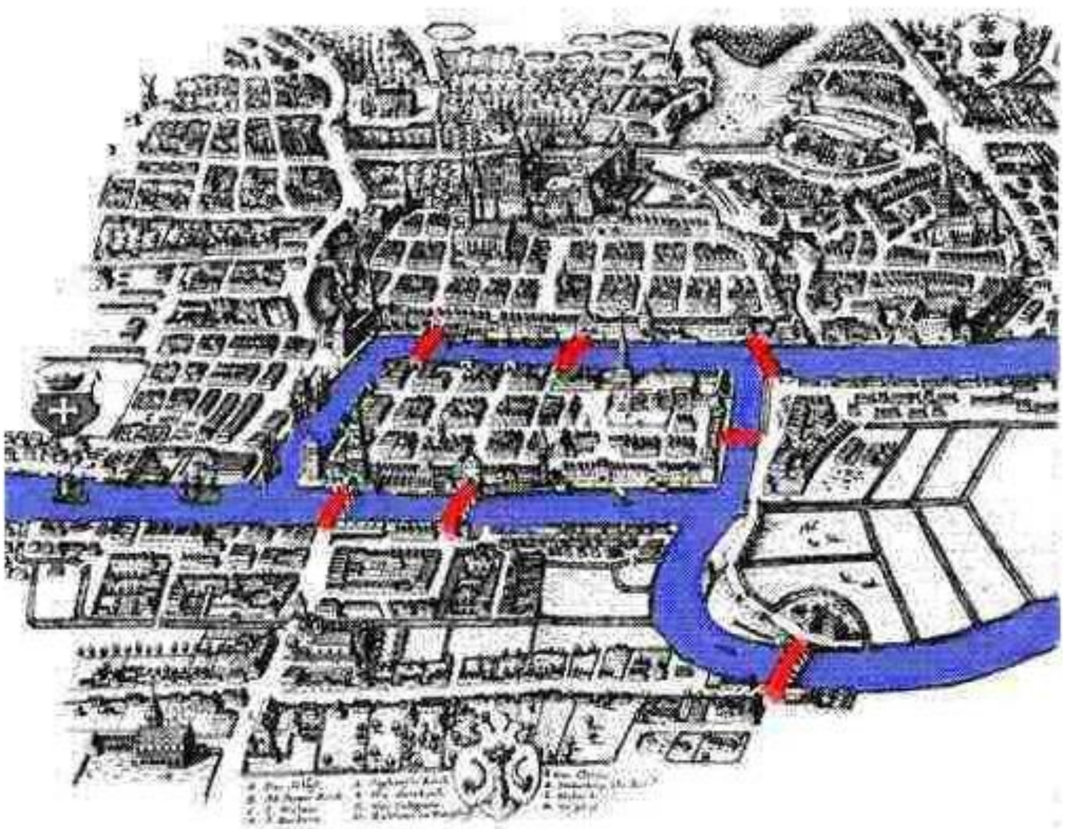
Grafuri euleriene

Istoric. Aplicații

– din cursul 1

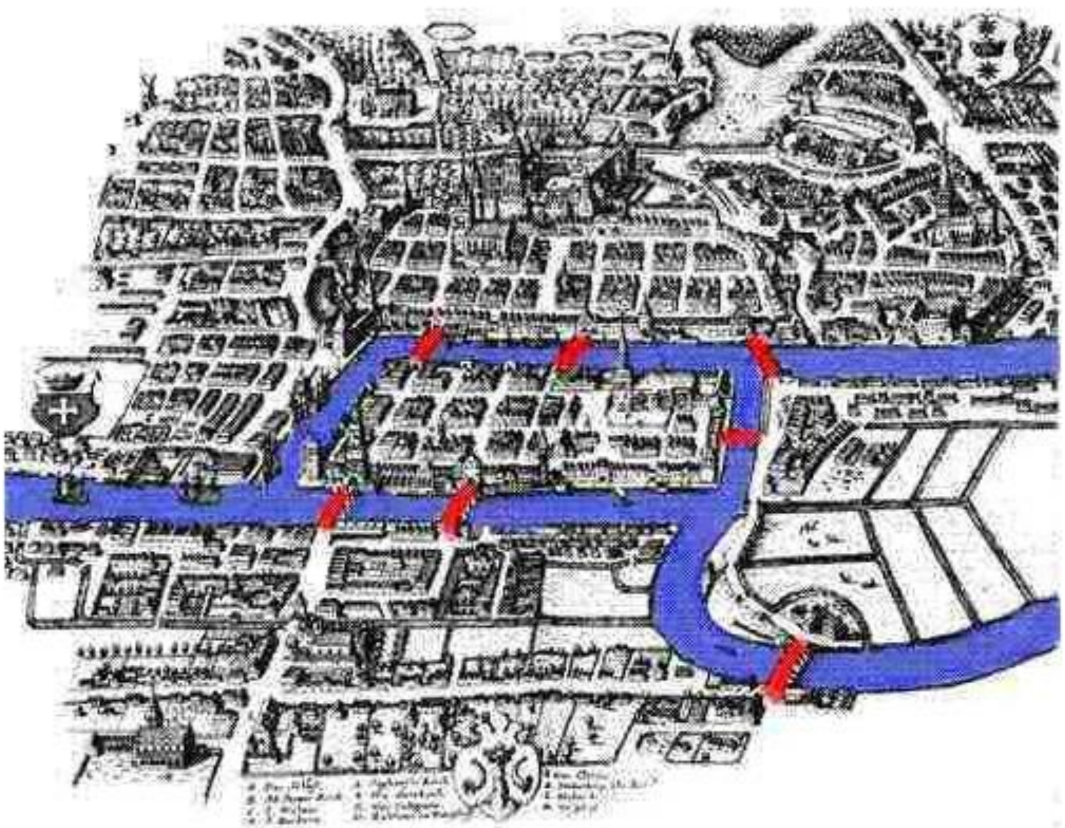
Problema celor 7 poduri din Königsberg

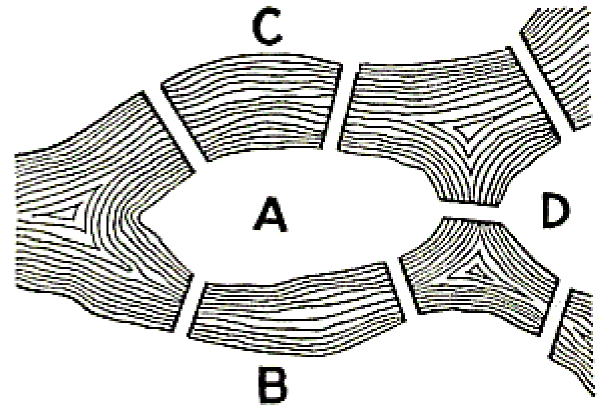


Este posibil ca un om să facă o plimbare în care să treacă pe toate cele 7 poduri o singură dată?

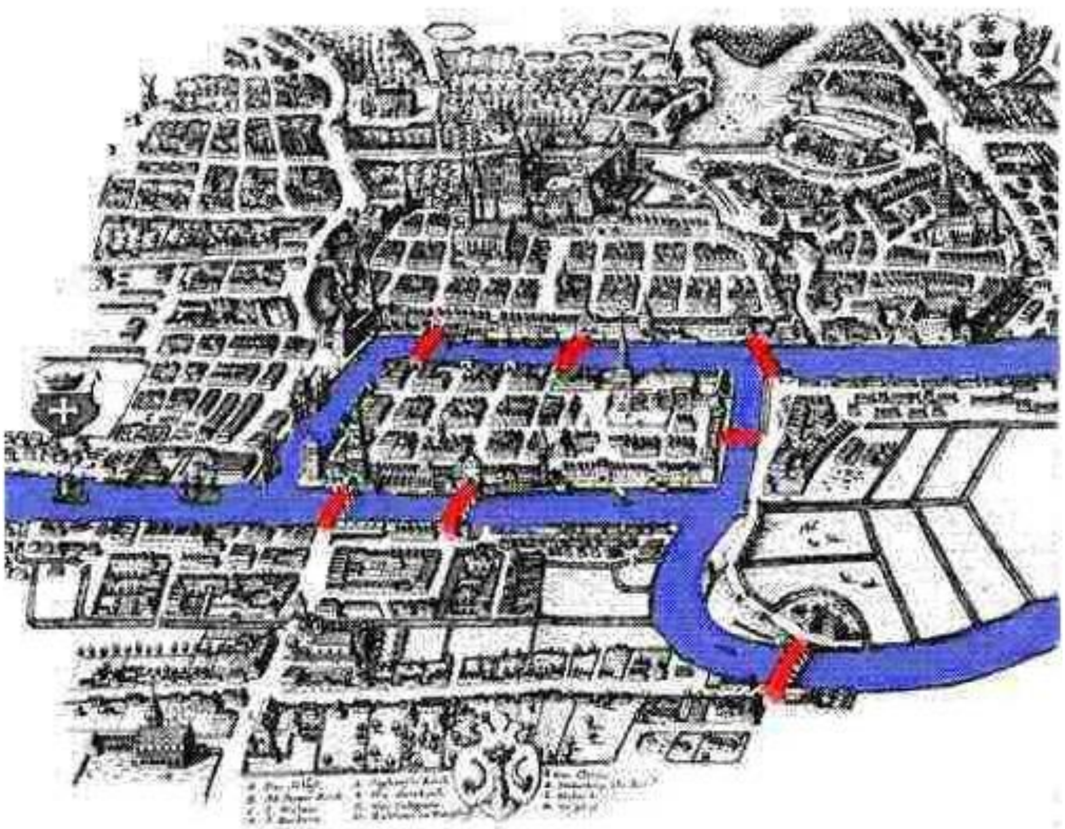
http://think-like-a-git.net/sections/graph-theory/seven-bridges-of-konigsberg.html

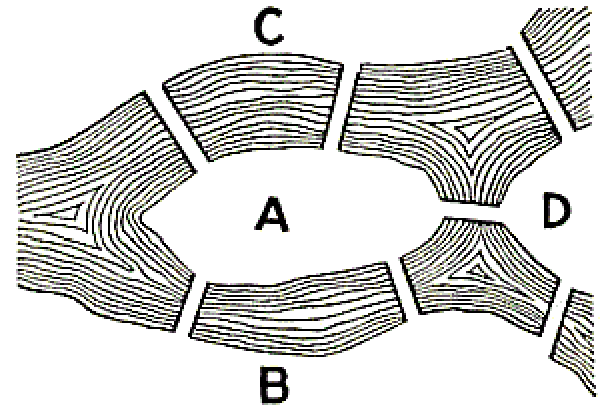
Problema celor 7 poduri din Königsberg



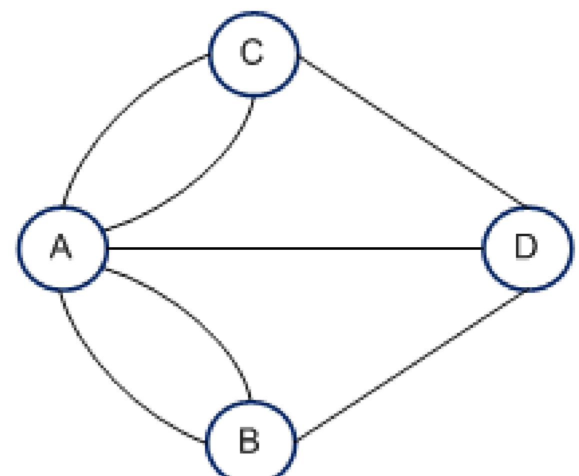


Problema celor 7 poduri din Königsberg

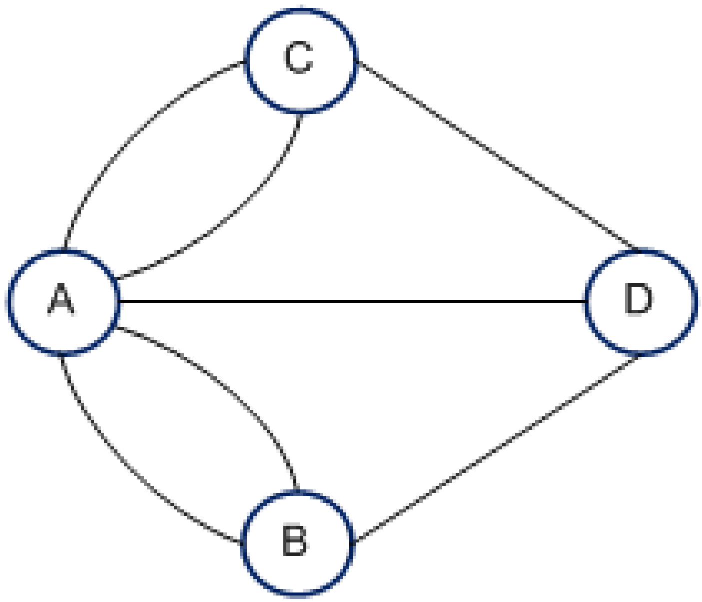




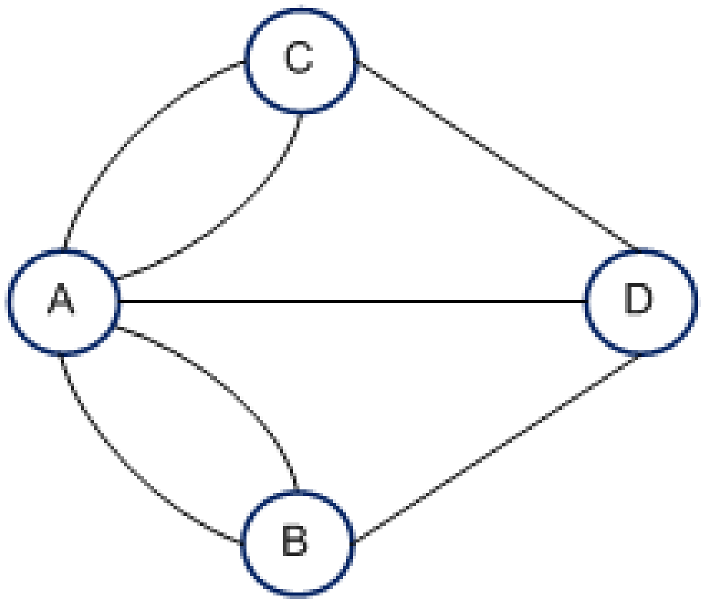
Modelare:

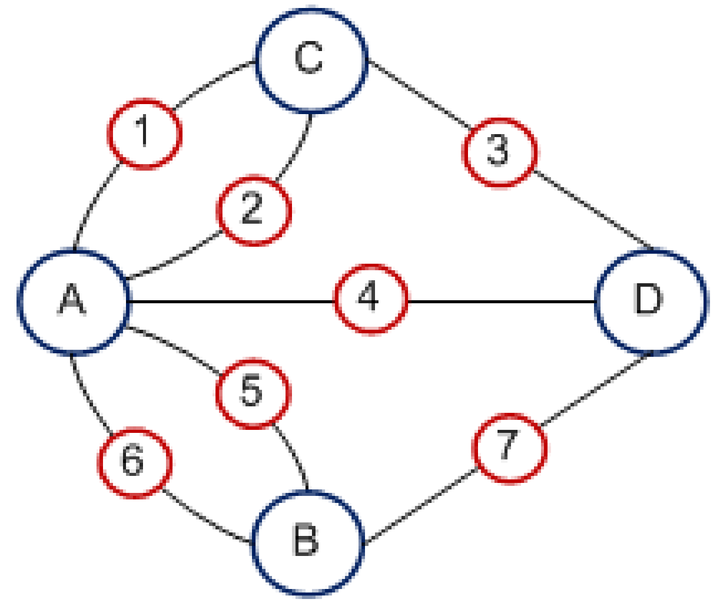


Problema celor 7 poduri din Königsberg



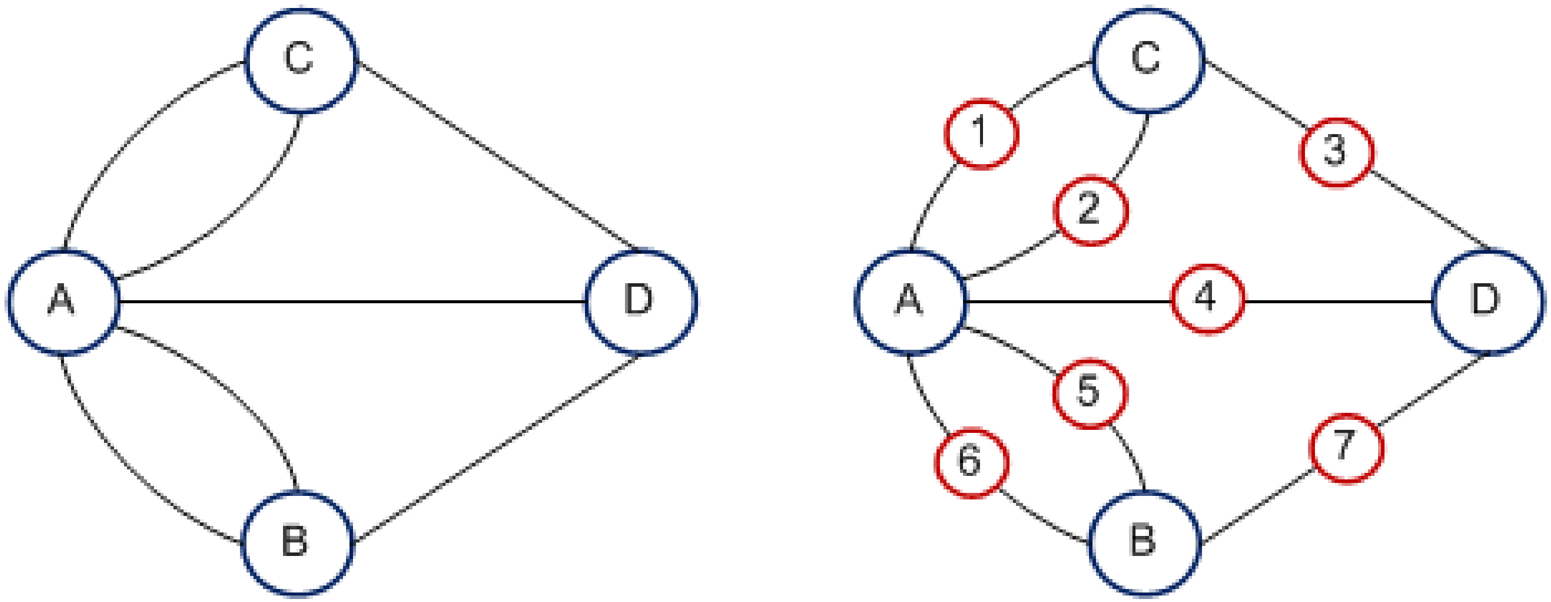
Problema celor 7 poduri din Königsberg





graf simplu

Problema celor 7 poduri din Königsberg



1736 – Leonhard Euler

Solutio problematis ad geometriam situs pertinentis

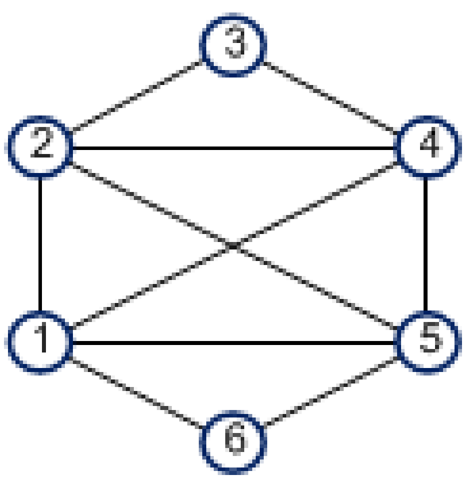
- **Ciclu eulerian** – traseu închis care trece o singură dată prin toate muchiile
- **Graf eulerian**

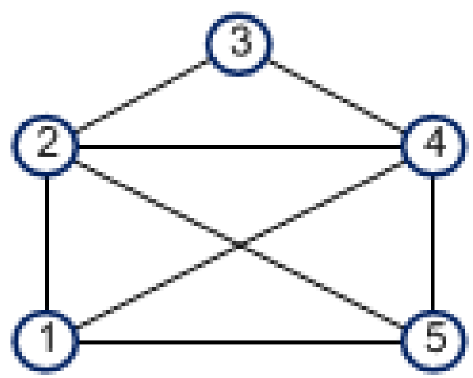
Problema celor 7 poduri din Königsberg

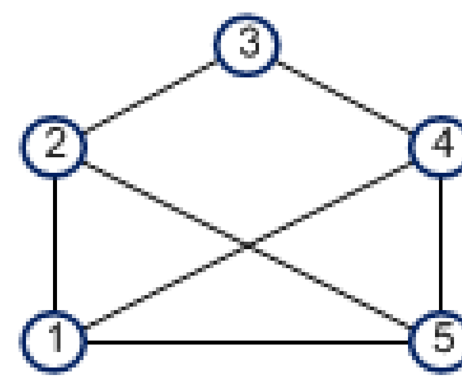
- **Interpretare**

Se poate desena diagrama printr-o curbă continuă închisă fără a ridica creionul de pe hârtie şi fără a desena o linie de două ori (în plus: să terminăm desenul în punctul în care l-am început)?

  - Tăierea unui material



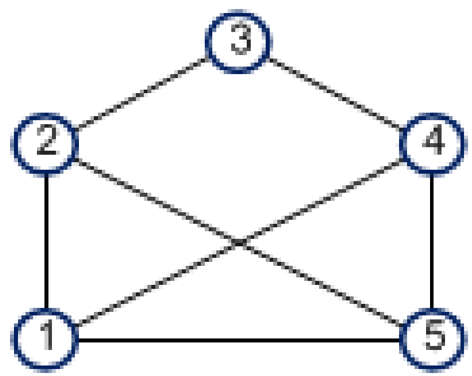




Problema celor 7 poduri din Königsberg

- **Interpretare**

  De câte ori (minim) trebuie să ridicăm creionul de pe hârtie pentru a desena diagrama?



Grafuri euleriene

Fie G graf neorientat

- **Ciclu eulerian** al lui G = ciclu C în G cu

$$E(C) = E(G)$$

- G e**ulerian** = conține un **ciclu** eulerian

- **Lanț eulerian** al lui G = lanț simplu P în G cu

$$E(P) = E(G)$$

Grafuri euleriene

Observație

- **Fie** $P=[v_1, \dots, v_k]$ un lanț (nu neapărat elementar)
  - Dacă $v_1 \neq v_k$, atunci vârfurile interne din P au gradul în P par, iar extremitățile au gradul în P impar
  - Dacă $v_1 = v_k$, atunci toate vârfurile din P au gradul în P par

Grafuri euleriene

Lemă

Fie $G=(V,E)$ un graf neorientat, **conex**, cu **toate vârfurile de grad par** și $E\neq\emptyset$.

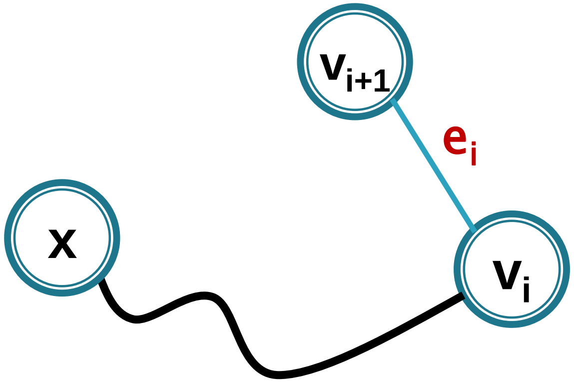
Atunci pentru orice $x \in V$ există **un ciclu** C în G cu $\mathbf{x \in V(C)}$

(**ciclu care conține x, nu neapărat eulerian, nici neapărat elementar**)

Grafuri euleriene

Demonstrație – Algoritm de determinare a unui ciclu care conține x:

- $i = 1$, $v_1 = x$
- $E(C) = \emptyset$
- Repetă



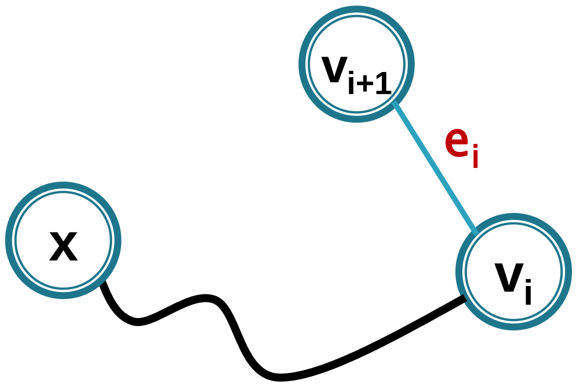
  până când $v_i = x$

Grafuri euleriene

Demonstrație – Algoritm de determinare a unui ciclu care conține x:

- $i = 1$, $v_1 = x$
- $E(C) = \varnothing$
- Repetă
  - **selectează $e_i = v_i v_{i+1} \in E(G) - E(C)$**
  - **$E(C) = E(C) \cup \{e_i\}$**
  - **$i = i + 1$**

  până când $v_i = x$

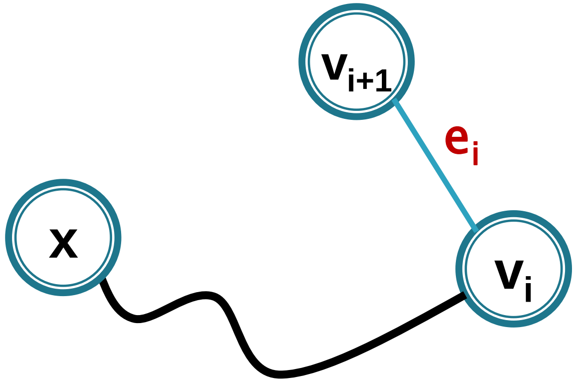


Grafuri euleriene

Demonstrație – Algoritm de determinare a unui ciclu care conține x:

- $i = 1$, $v_1 = x$
- $E(C) = \varnothing$
- Repetă
  - **selectează $e_i = v_i v_{i+1} \in E(G) - E(C)$**
  - **$E(C) = E(C) \cup \{e_i\}$**
  - **$i = i + 1$**

  până când $v_i = x$



Algoritmul este corect deoarece:

Grafuri euleriene

Demonstrație – Algoritm de determinare a unui ciclu care conține x:

- $i = 1$, $v_1 = x$
- $E(C) = \varnothing$
- Repetă
  - **selectează $e_i = v_i v_{i+1} \in E(G) - E(C)$**
  - **$E(C) = E(C) \cup \{e_i\}$**
  - **$i = i + 1$**

  până când $v_i = x$

Dacă $v_i \neq x$, atunci $d_C(v_i)$ este impar (cf. obs. Anterioare).

Din ipoteză, $d_G(v_i)$ este par deci $d_{G-E(C)}(v_i) > 0$

$\Rightarrow$ muchia e_i există

Grafuri euleriene

Demonstrație – Algoritm de determinare a unui ciclu care conține x:

- $i = 1$, $v_1 = x$
- $E(C) = \varnothing$
- Repetă
  - selectează $e_i = v_i v_{i+1} \in E(G) - E(C)$
  - $E(C) = E(C) \cup \{e_i\}$
  - $i = i + 1$

până când $v_i = x$

Dacă $v_i \neq x$, atunci $d_C(v_i)$ este impar (cf. obs. Anterioare).

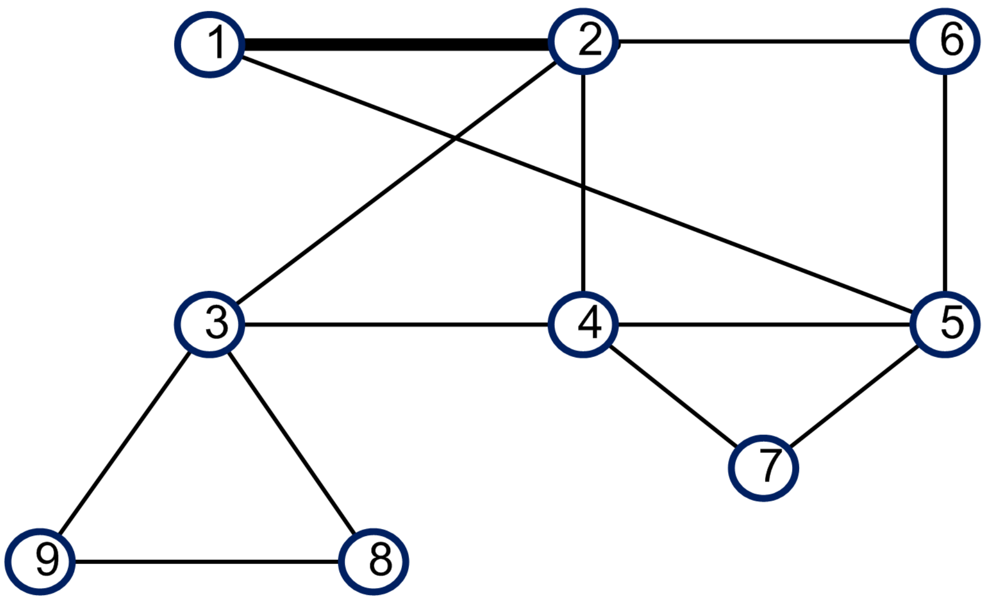
Din ipoteză, $d_G(v_i)$ este par deci $d_{G-E(C)}(v_i) > 0$

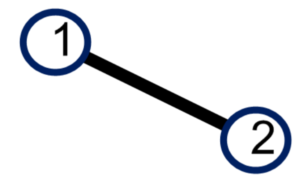
$\Rightarrow$ muchia e_i există

$|E(G)| < \infty$, deci algoritmul se termină (v_i ajunge egal cu x)

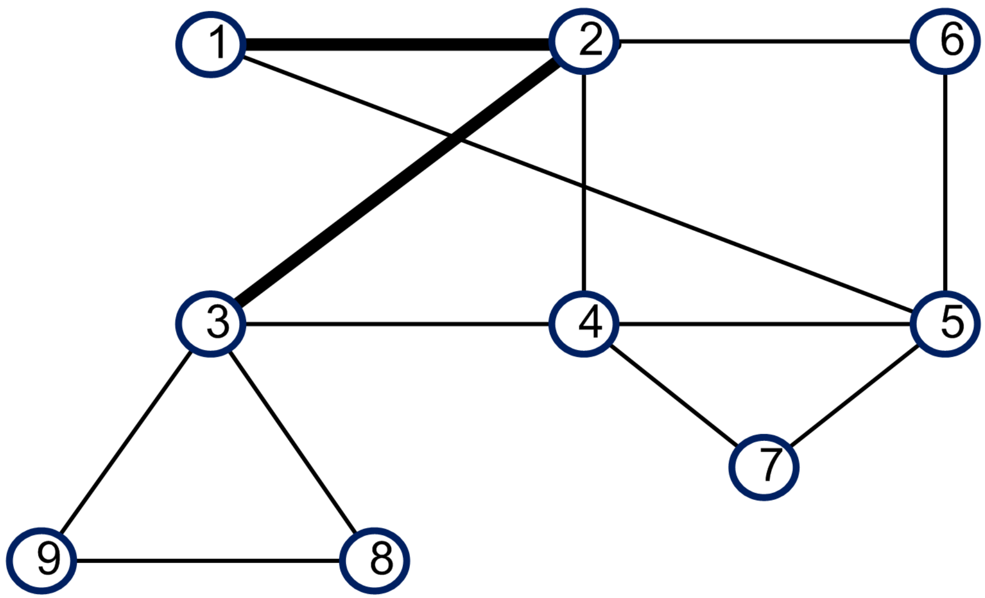
Grafuri euleriene

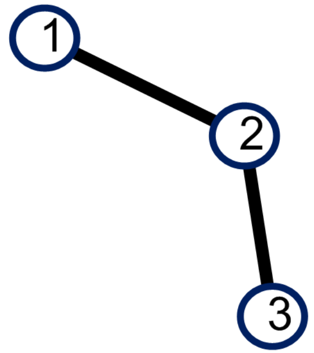
x=1



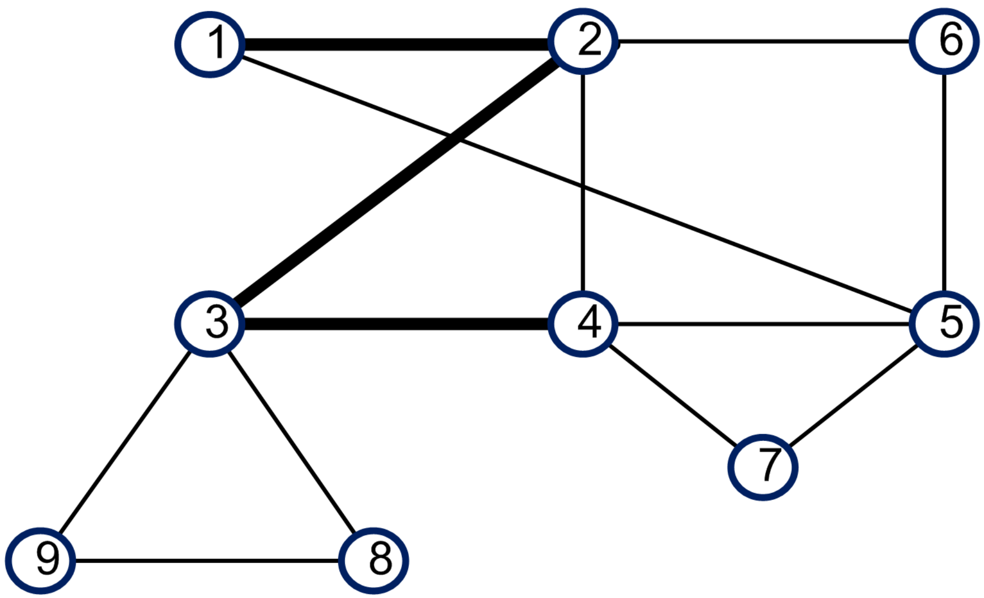


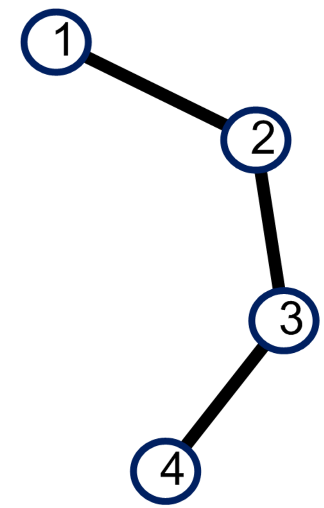
Grafuri euleriene



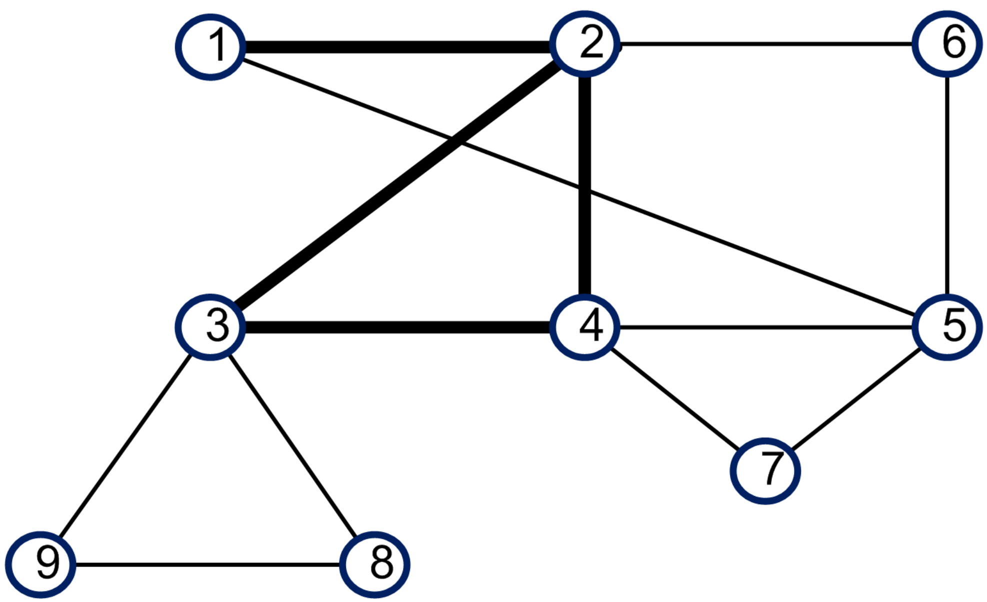


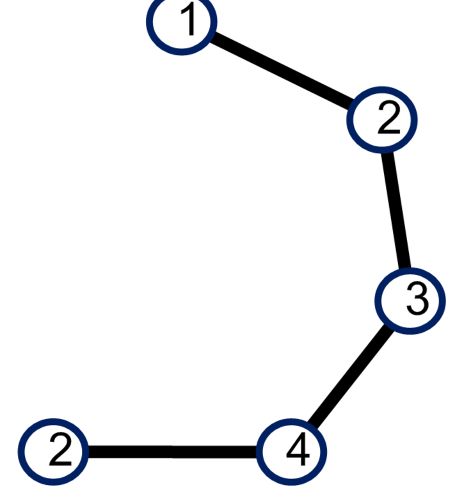
Grafuri euleriene



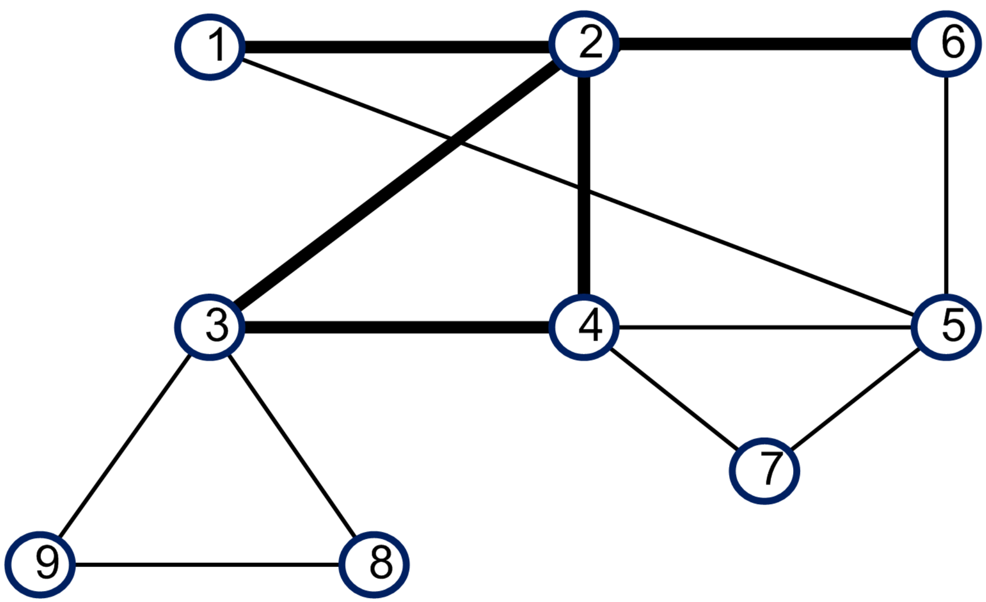


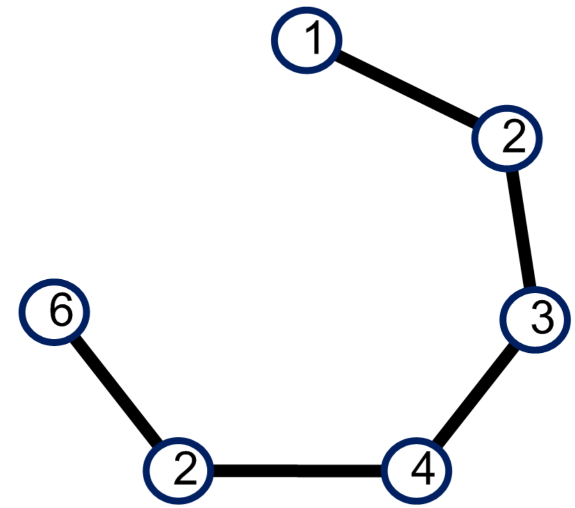
Grafuri euleriene



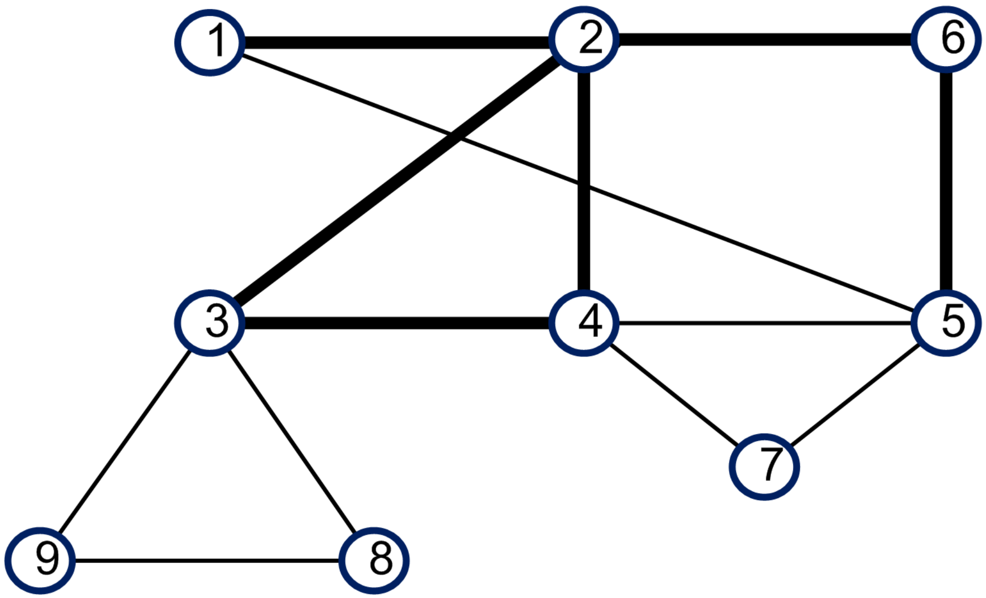


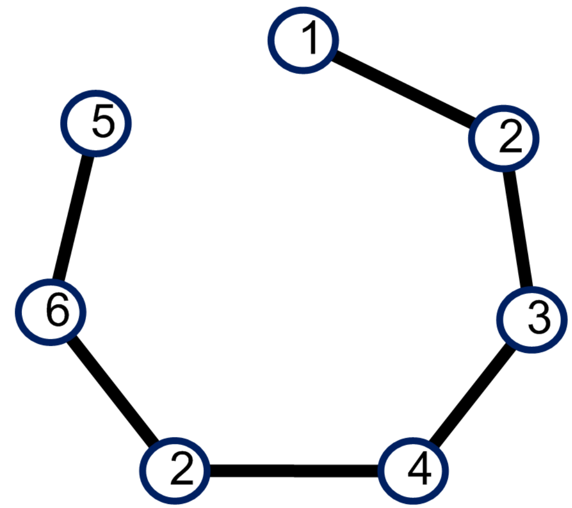
Grafuri euleriene



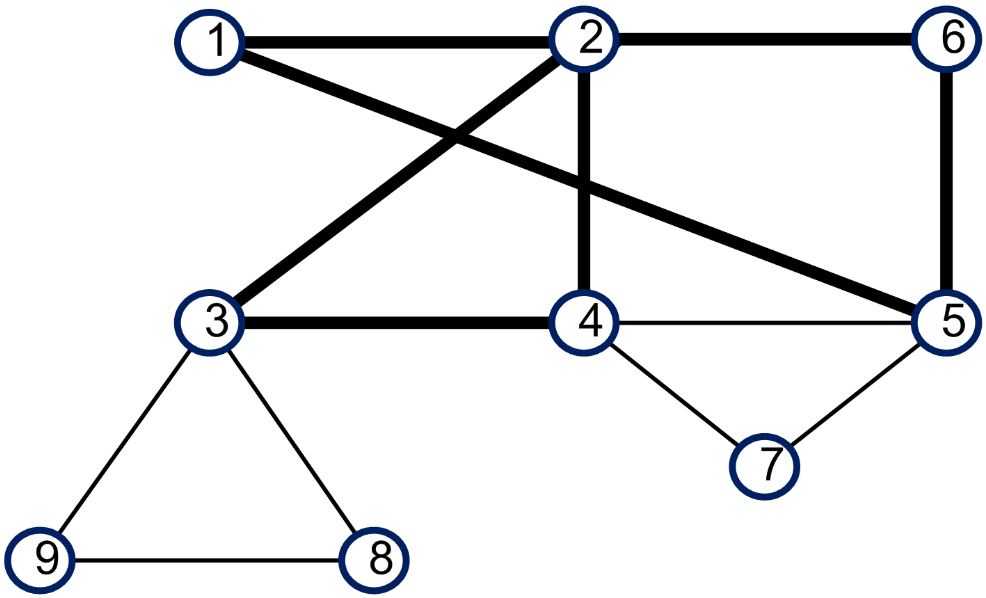


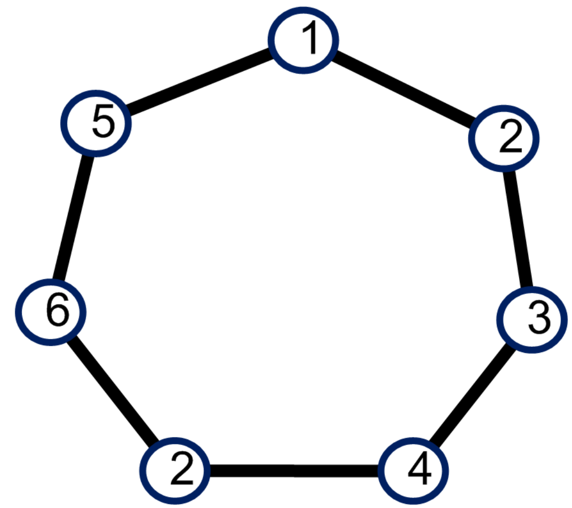
Grafuri euleriene





Grafuri euleriene





Grafuri euleriene

Teorema lui Euler

Fie G=(V, E) un (multi)**graf** neorientat, conex, cu $E \neq \emptyset$.

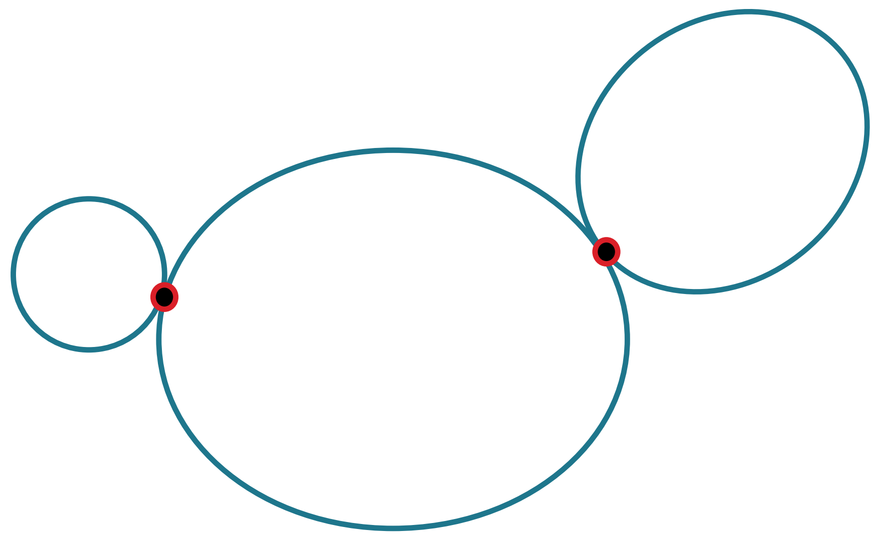
Atunci

G este eulerian $\Leftrightarrow$ **orice vârf din G are grad par**

Algoritmul lui Hierholzer

Determinarea unui ciclu eulerian într-un graf conex (sau un graf conex+ vârfuri izolate) **cu toate vârfurile de grad par**

- bazat pe ideea demonstrației Teoremei lui Euler – **fuziune de cicluri (succesiv)**



Algoritmul lui Hierholzer

- **Pasul 0** – verificare condiții (conex+vf. izolate, grade pare)

Algoritmul lui Hierholzer

- **Pasul 0** – verificare condiții (conex+vf. izolate, grade pare)
- **Pasul 1:**
  - **alege v ∈ V arbitrar**
  - **construiește C un ciclu în G care începe cu v (cu algoritmul din Lema)**

Algoritmul lui Hierholzer

- **Pasul 0** – verificare condiții (conex+vf. izolate, grade pare)
- **Pasul 1:**
  - **alege v ∈ V arbitrar**
  - **construiește C un ciclu în G care începe cu v (cu algoritmul din Lema)**
- **cât timp |E(C)| < |E(G)| execută**
  - **selectează v ∈ V(C) cu $d_{G-E(C)}(v) > 0$ (în care sunt incidente muchii care nu aparțin lui C)**

Algoritmul lui Hierholzer

- **Pasul 0** – verificare condiții (conex+vf. izolate, grade pare)
- **Pasul 1:**
  - **alege v ∈ V arbitrar**
  - **construiește C un ciclu în G care începe cu v (cu algoritmul din Lema)**
- **cât timp |E(C)| < |E(G)| execută**
  - **selectează v ∈ V(C) cu $d_{G-E(C)}(v) > 0$ (în care sunt incidente muchii care nu aparțin lui C)**
  - **construiește C′ un ciclu în G - E(C) care începe cu v**

Algoritmul lui Hierholzer

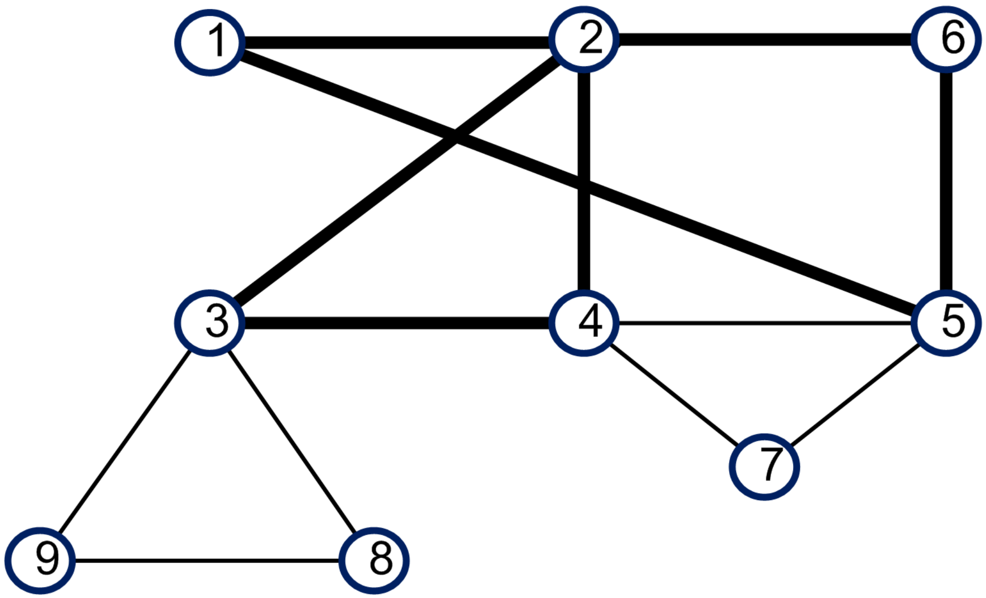
- **Pasul 0** – verificare condiții (conex+vf. izolate, grade pare)
- **Pasul 1:**
  - **alege v ∈ V arbitrar**
  - **construiește C un ciclu în G care începe cu v (cu algoritmul din Lema)**
- **cât timp |E(C)| < |E(G)| execută**
  - **selectează v ∈ V(C) cu $d_{G-E(C)}(v) > 0$ (în care sunt incidente muchii care nu aparțin lui C)**
  - **construiește C′ un ciclu în G - E(C) care începe cu v**
  - **C = ciclul obținut prin fuziunea ciclurilor C și C′ în v**

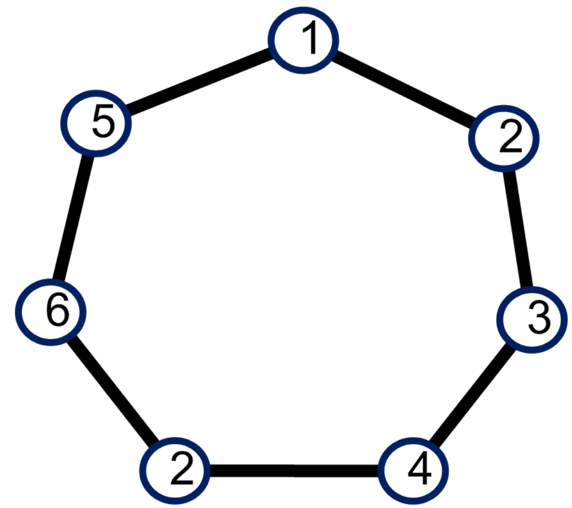
Algoritmul lui Hierholzer

- **Pasul 0** – verificare condiții (conex+vf. izolate, grade pare)
- **Pasul 1:**
  - `alege v ∈ V arbitrar`
  - `construiește C un ciclu în G care începe cu v (cu algoritmul din Lema)`
- **cât timp |E(C)| < |E(G)| execută**
  - `selectează v ∈ V(C) cu` $d_{G-E(C)}(v) > 0$ `(în care sunt incidente muchii care nu aparțin lui C)`
  - `construiește C′ un ciclu în G - E(C) care începe cu v`
  - `C = ciclul obținut prin fuziunea ciclurilor C și C′ în v`
- **scrie C**

Algoritmul lui Hierholzer

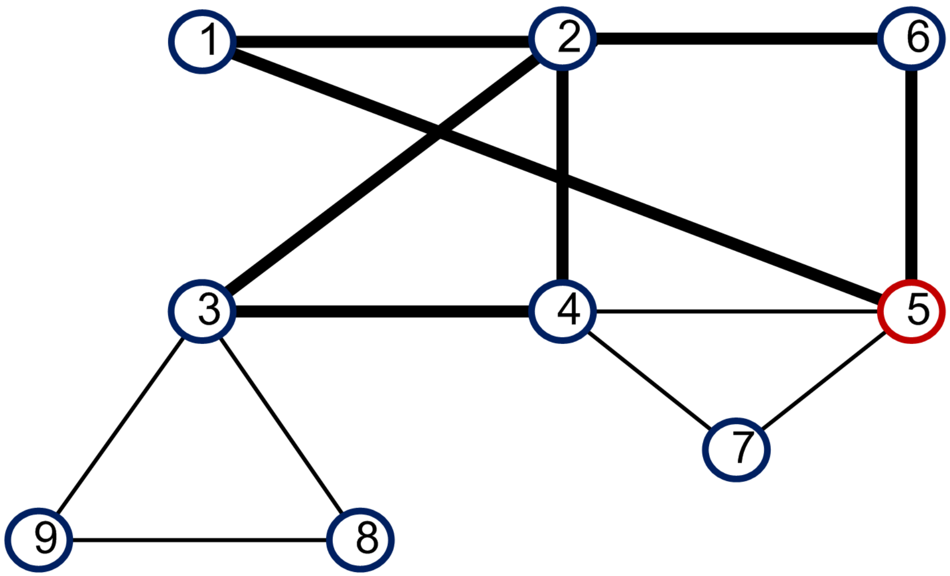
Pornim cu ciclul construit cu algoritmul din Lema 1



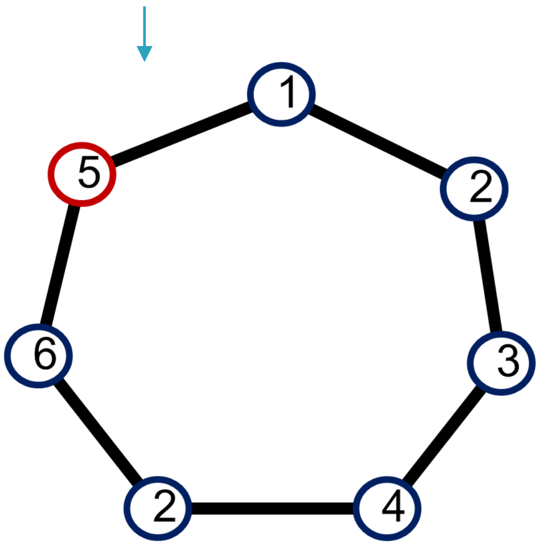


C= [1, 2, 3, 4, 2, 6, 5, 1]

Algoritmul lui Hierholzer



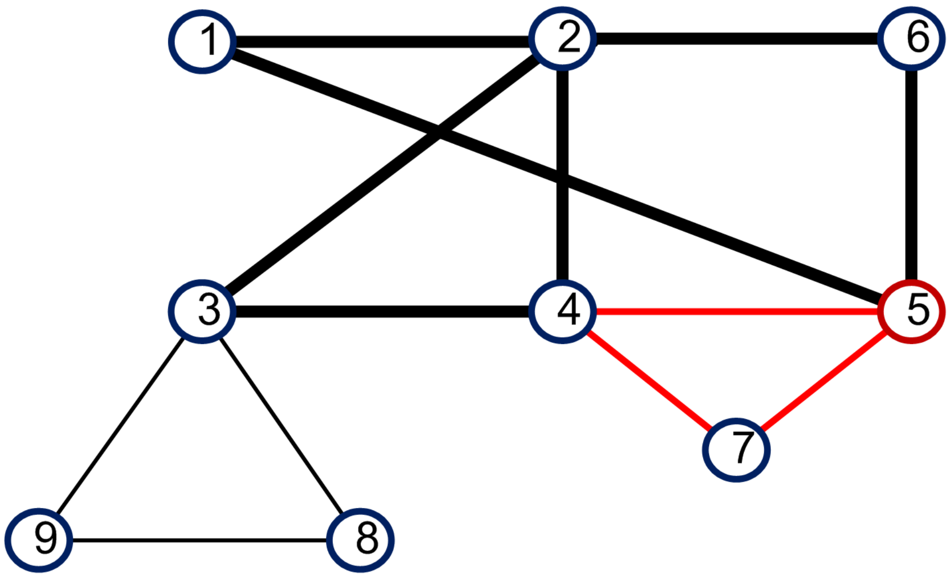
Alegem un vârf din C în care mai sunt incidente muchii,
de exemplu v = 5

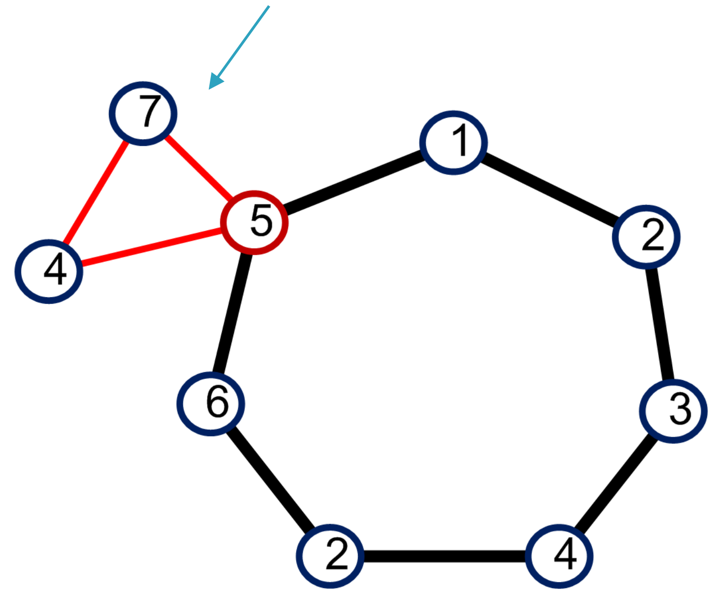


C= [1, 2, 3, 4, 2, 6, 5, 1]

Algoritmul lui Hierholzer

Construim un ciclu C' cu muchiile rămase care conține v = 5

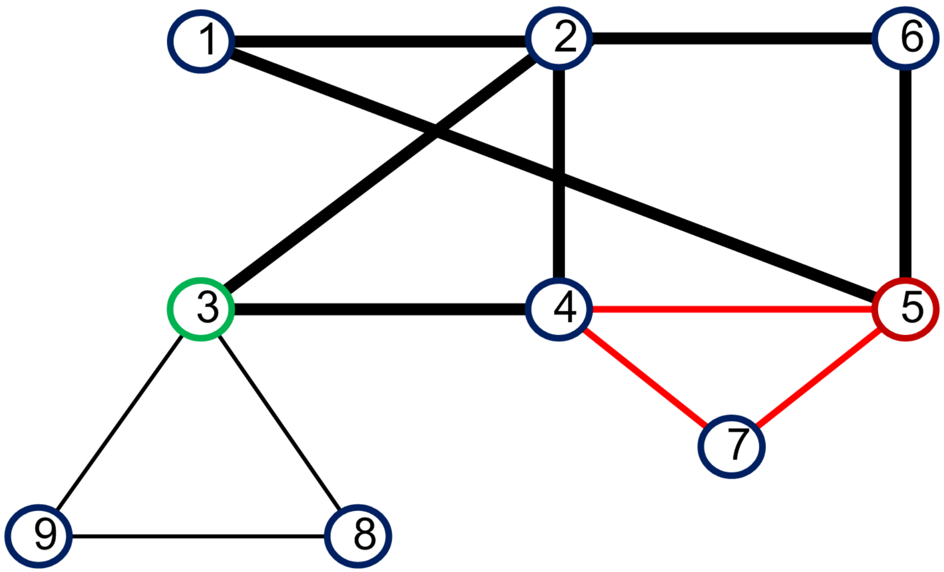




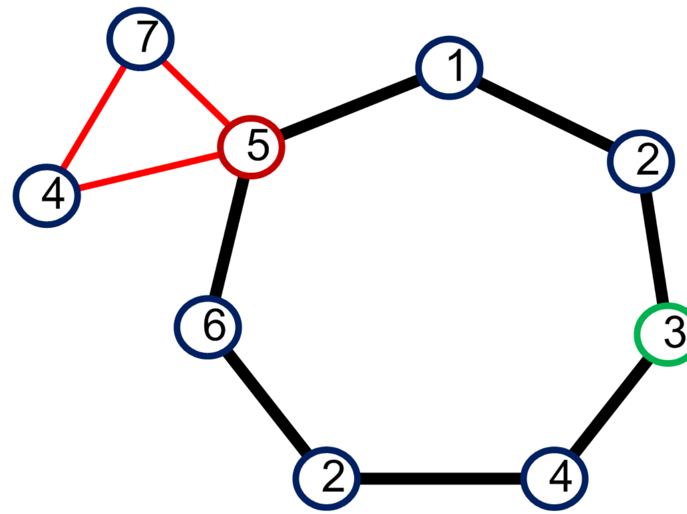
⇒ Un nou ciclu obținut prin fuziunea celor două cicluri

C= [1, 2, 3, 4, 2, 6, **5, 4, 7, 5**, 1]

Algoritmul lui Hierholzer



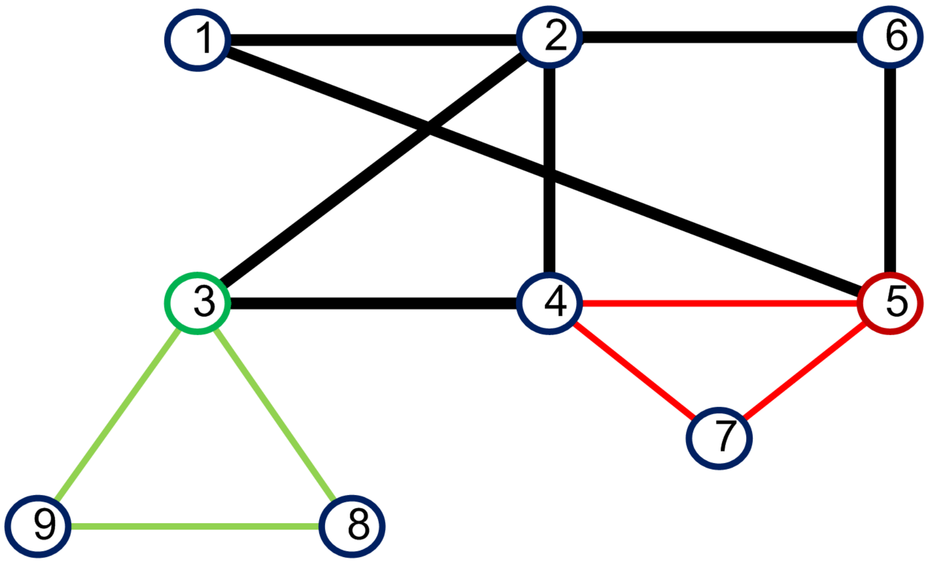
Alegem un vârf din C în care mai sunt incidente muchii,
de exemplu v = 3

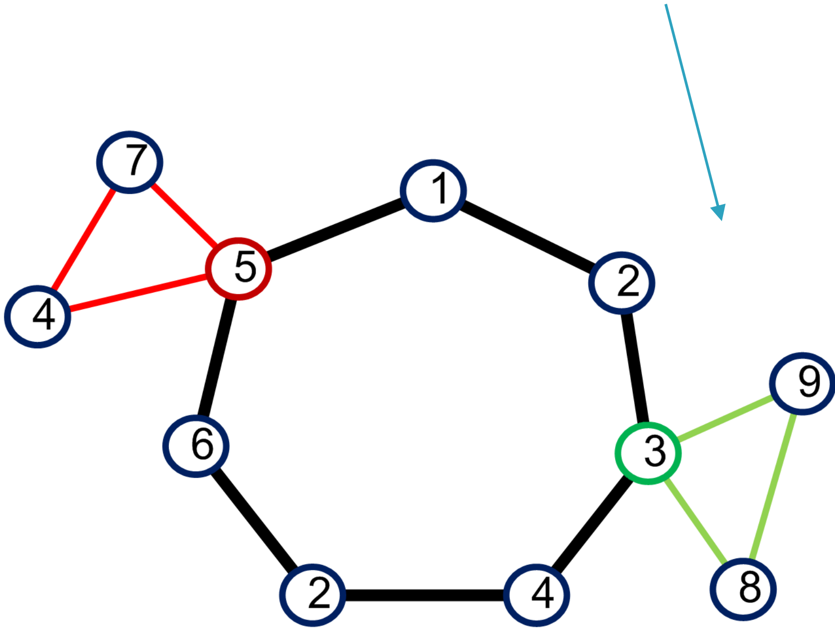


C= [1, 2, 3, 4, 2, 6, 5, 4, 7, 5, 1]

Algoritmul lui Hierholzer

Construim un ciclu C' cu muchiile rămase care conține v = 3

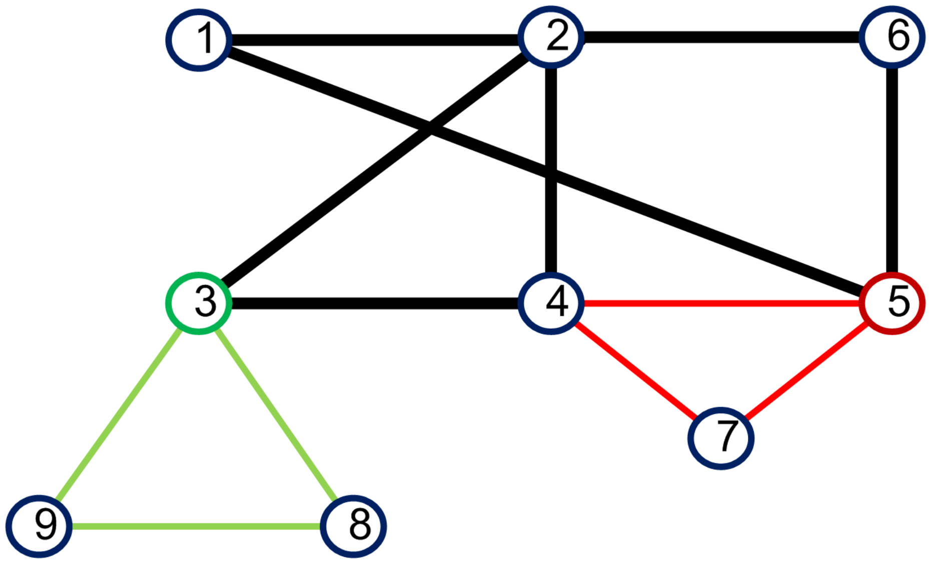


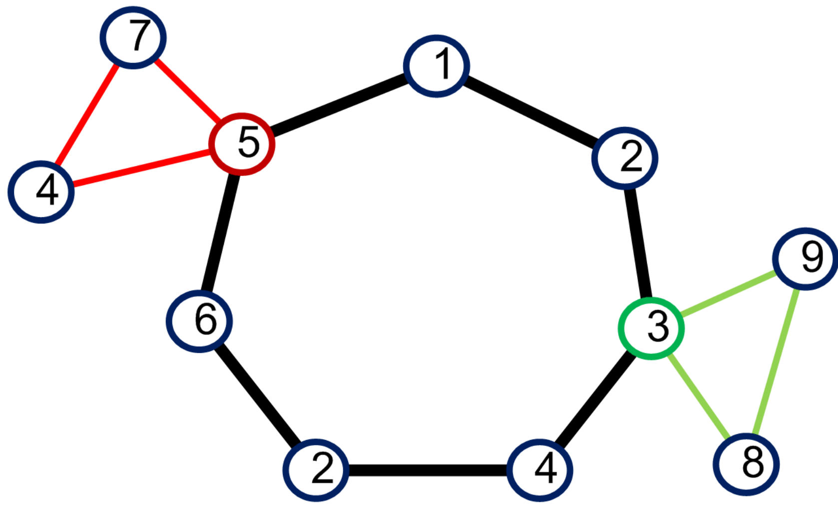


⇒ Un nou ciclu obținut prin fuziunea celor două cicluri

C= [1, 2, 3, 8, 9, 3, 4, 2, 6, 5, 4, 7, 5, 1]

Algoritmul lui Hierholzer





C= [1, 2, 3, 8, 9, 3, 4, 2, 6, 5, 4, 7, 5, 1]

Ciclul conține toate muchiile

$\Rightarrow$ este eulerian

Algoritmul lui Hierholzer

- **Complexitate – O(m)**

Algoritmul lui Hierholzer

- Posibile implementări
- Varianta recursivă

```
euler (nod v)
    cat timp d(v)> 0
          alege vw o muchie incidenta in v
          sterge muchia vw din G
          euler (w)
    C = C + v  //adaugam v la ciclul C
```

Inițial

```
C = Ø
euler(1)  //pornim construcția din varful 1
```

Algoritmul lui Hierholzer

- **Posibile implementări**
  - **Varianta** – Nerecursiv – Liste dublu înlănțuite/stive
  - Muchiile folosite – marcate (nu neapărat șterse) sau ștearsă cea de la finalul listei de adiacență

```
stiva St
push(1,St)
cat timp St este nevida executa
 v = top(St)
 daca grad(v) != 0 atunci
     alege vw o muchie incidenta in v
     sterge muchia vw din G
     push(w,St)
 altfel
     C = C + v
     pop(St)
```

de exp cea de la finalul listei de adiac a lui v

Lanțuri euleriene

Teorema lui Euler

Fie G=(V, E) un graf neorientat, conex, cu $E \neq \emptyset$.

Atunci

G are un lanț eulerian $\Leftrightarrow$ **G are cel mult două vârfuri de grad impar**

Grafuri orientate euleriene

Grafuri orientate euleriene

Observație

– **Fie** $P=[v_1, \ldots, v_k]$ dum

- Dacă $v_1 \neq v_k$, atunci vârfurile interne v din P au

$d_P^-(v) = d_P^+(v)$, iar pentru extremități:

$$d_P^-(v_1) = d_P^+(v_1) - 1, d_P^-(v_k) = d_P^+(v_k) + 1$$

- Dacă $v_1 = v_k$, atunci toate vârfurile v din P au gradul intern în P egal cu cel extern:

$$d_P^-(v) = d_P^+(v)$$

Grafuri orientate euleriene

Teorema lui Euler

Fie G=(V, E) un graf orientat, conex (= graful neorientat asociat este conex), cu $E \neq \emptyset$.

Atunci

G este eulerian $\Leftrightarrow$ $\forall$ $v \in V$ $d_G^-(v) = d_G^+(v)$

Drumuri euleriene

Teorema lui Euler

Fie $G=(V, E)$ un (multi)graf orientat, conex, cu $E\neq\emptyset$.

Atunci

G are un drum eulerian $\Leftrightarrow$

Drumuri euleriene

Teorema lui Euler

Fie G=(V, E) un (multi)graf orientat, conex, cu E≠∅.

Atunci

G are un drum eulerian ⇔

(∀ v∈V $d_G^-(v) = d_G^+(v)$) sau

(∃ x∈V cu $d_G^-(x) = d_G^+(x) - 1$,

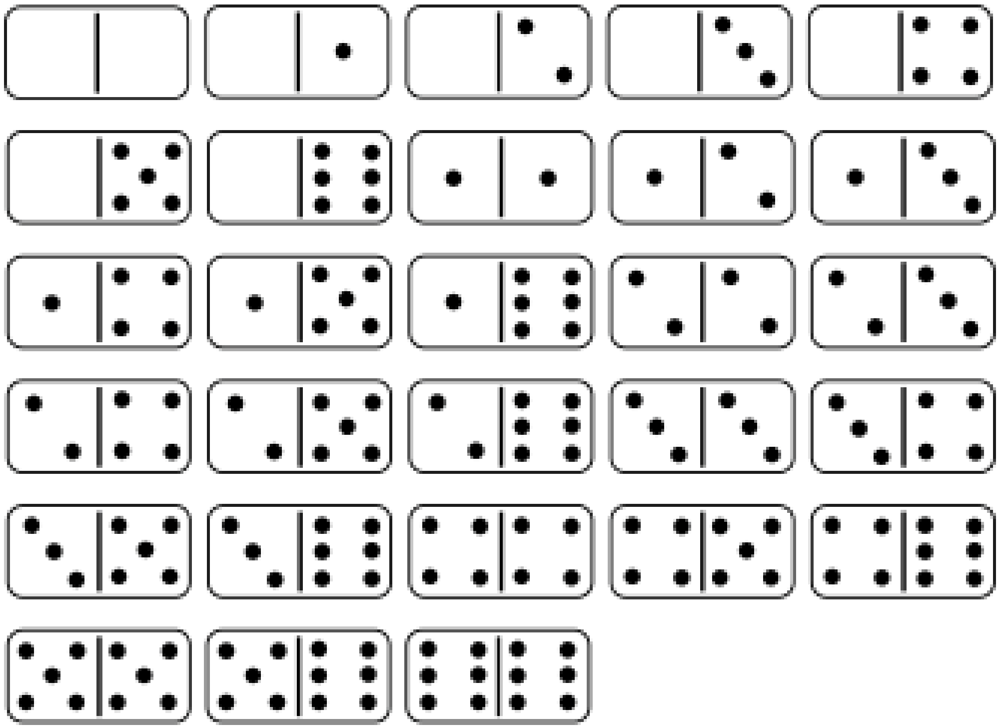
∃ y∈V , y ≠ x cu $d_G^-(y) = d_G^+(y) + 1$,

∀ v∈V−{x, y} $d_G^-(v) = d_G^+(v)$)

Grafuri euleriene

Problemă – joc domino

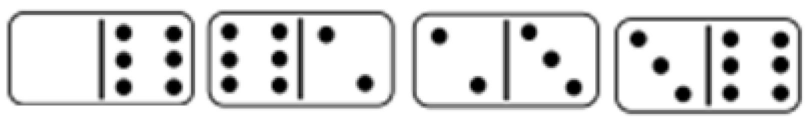
Piesă de domino – două fețe, numere 0..n, de obicei n=6



Grafuri euleriene

Problemă – joc domino

Șir de piese de domino – respectă regula de construcție: primul număr de pe piesa adăugată la șir = al doilea număr de pe ultima piesă din șir



Grafuri euleriene

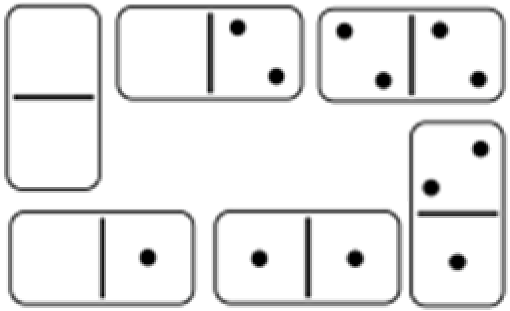
Problemă – joc domino

Se poate forma un șir de piese de domino care să conțină **toate piesele** + să se termine cu același număr cu care a început (un șir circular)?

Grafuri euleriene

Problemă – joc domino

Exemplu – daca folosim doar piese cu numere 0..2 putem forma un ciclu

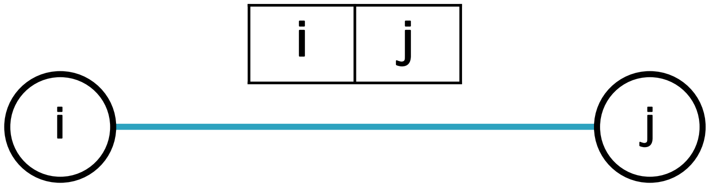


Grafuri euleriene

Problemă – joc domino

Graf asociat

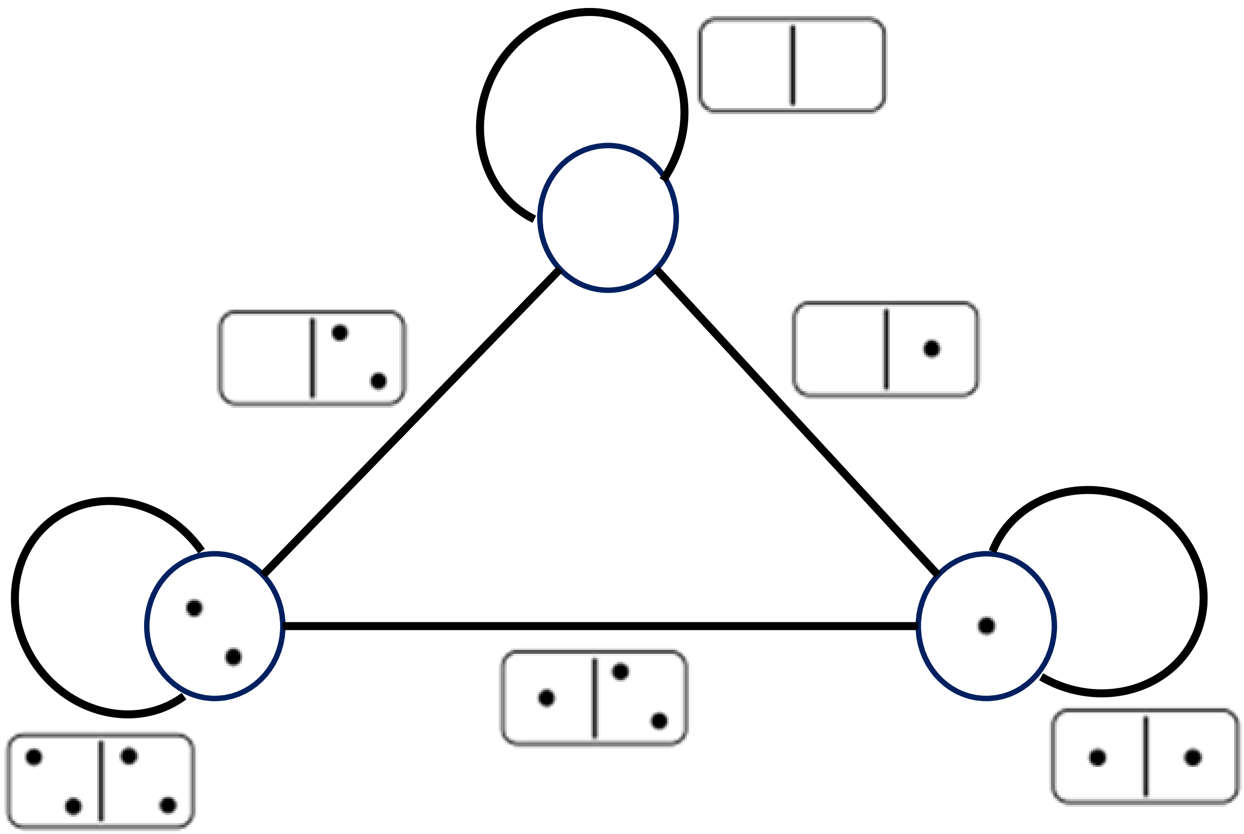
- vârfuri – numerele de pe piese
- muchii – perechi de numere (piesele)
- se pot lipi doar piese asociate muchiilor adiacente



Grafuri euleriene

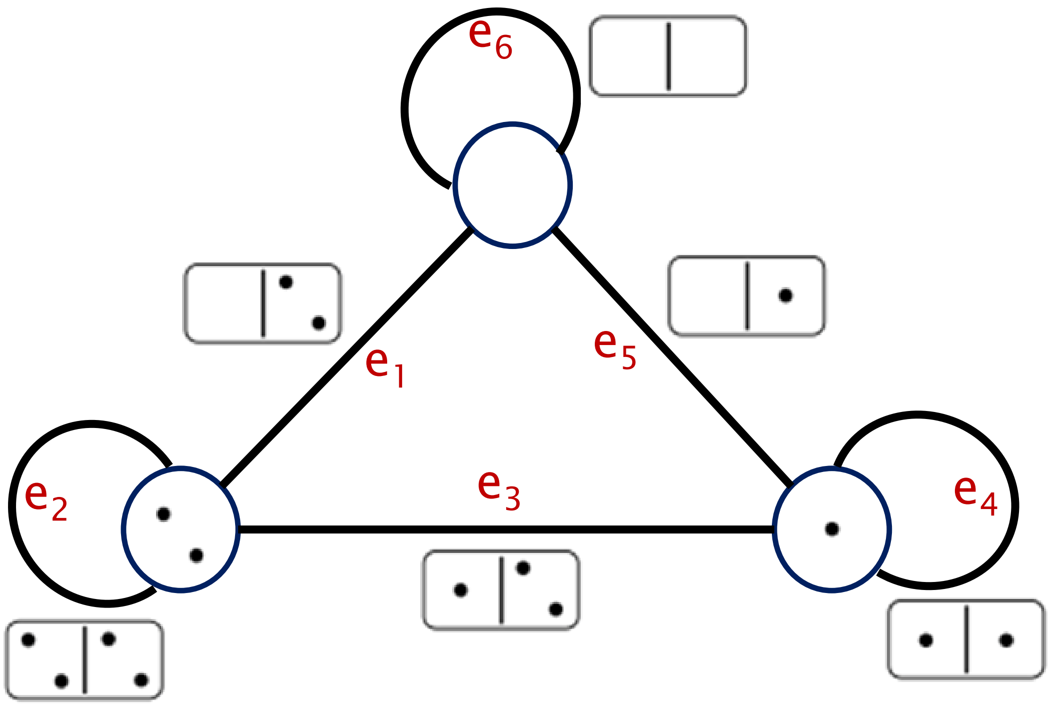
Problemă – joc domino

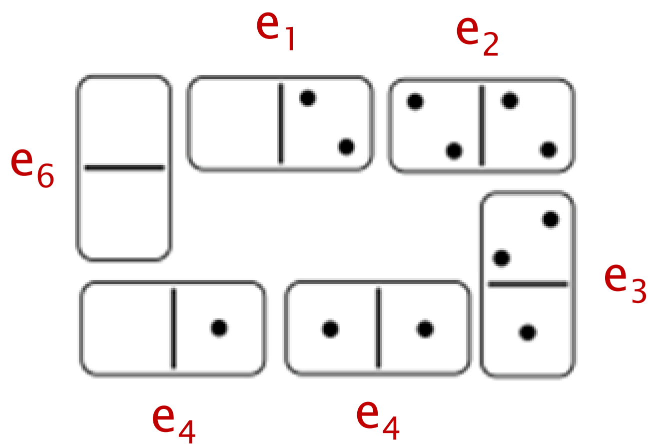
n=2



Grafuri euleriene

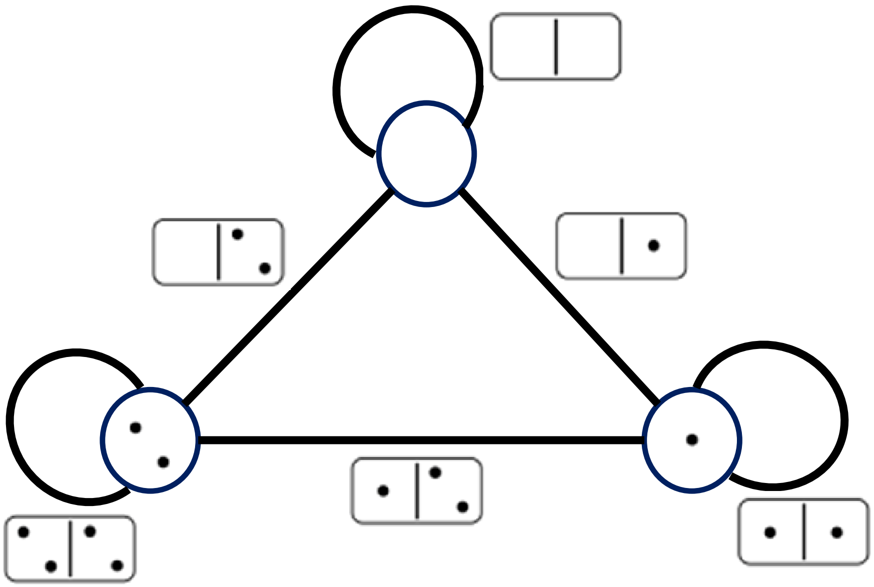
Există ciclu de piese ⇔ există ciclu eulerian în (multi)graf





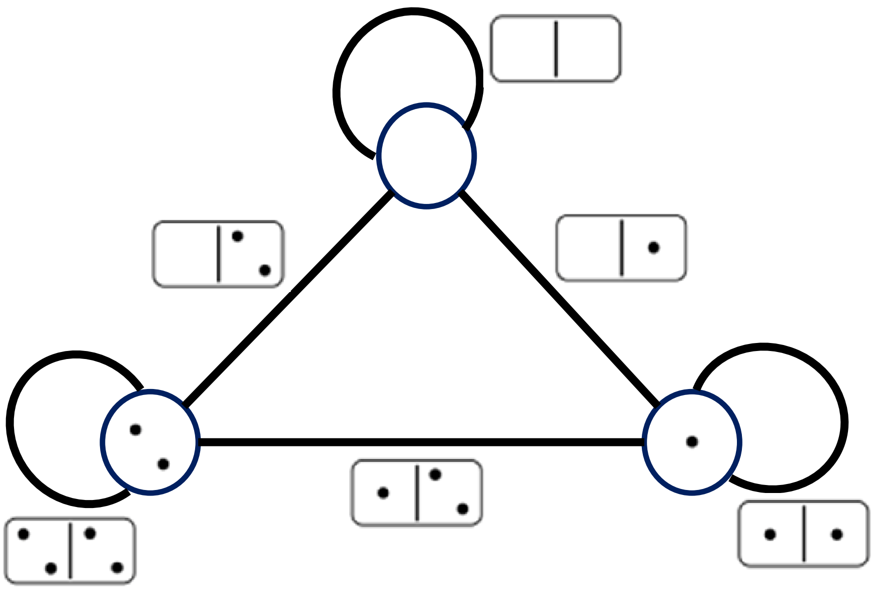
Grafuri euleriene

Pentru ce valori ale lui n există există ciclu eulerian în (multi)graful asociat?



Grafuri euleriene

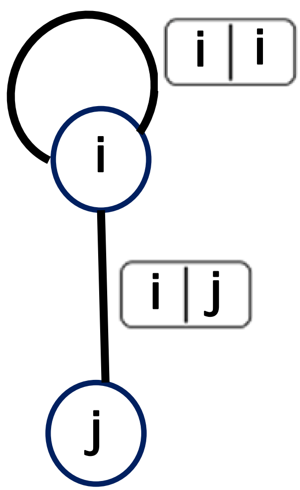
Pentru ce valori ale lui n există există ciclu eulerian în (multi)graful asociat?



d(i) = ? , pentru i = 0,…,n

Grafuri euleriene

Pentru ce valori ale lui n există există ciclu eulerian în (multi)graful asociat?



$d(i) = n+2$

(muchiile incidente în i sunt:

bucla etichetată (i,i) și

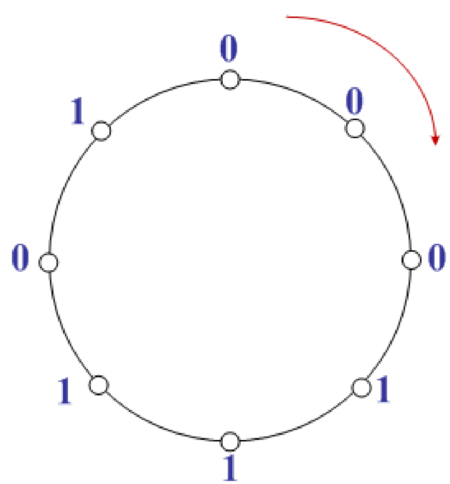
muchiile etichetate $\{i,j\}$ cu $j \neq i$, $j \in \{0,\ldots,n\}$)

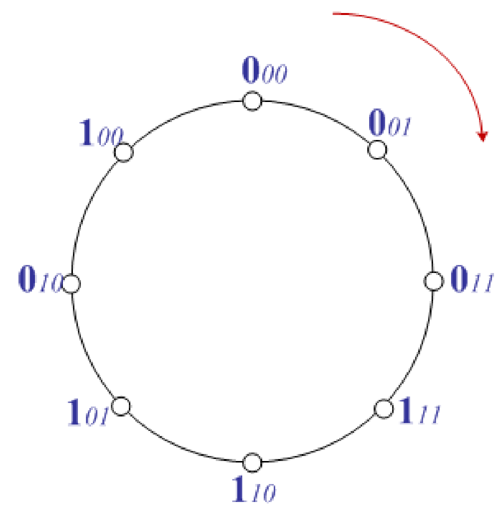
$\Rightarrow$ **trebuie ca n să fie par**

Grafuri euleriene

Problema lui POSTHUMUS

- $f\ (n)$ = numărul minim de cifre de 0 și 1 care se pot dispune circular a.î. între cele $f\ (n)$ secvențe de lungime n de cifre succesive apar toți cei 2^n vectori de lungime n peste {0,1} (citite în același sens).
- Evident $f\ (n) \geq 2^n$. **Are loc chiar egalitate?**





Grafuri euleriene

Universal string problem (de Bruijn, 1946)

- Determinați, dacă există, un șir care conține ca subsecvente toate șirurile binare de lungime k o singură dată

k = 3

000, 001, 010, 011, 100, 101, 110, 111

Grafuri euleriene

Universal string problem (de Bruijn, 1946)

- Determinați, dacă există, un șir care conține ca subsecvente toate șirurile binare de lungime k o singură dată

k = 3

000, 001, 010, 011, 100, 101, 110, 111

- Shortest superstring problem:

Date șiruri (de aceeași lungime / nu neapărat) determinați cel mai scurt șir care conține ca subsecvențe toate aceste șiruri.

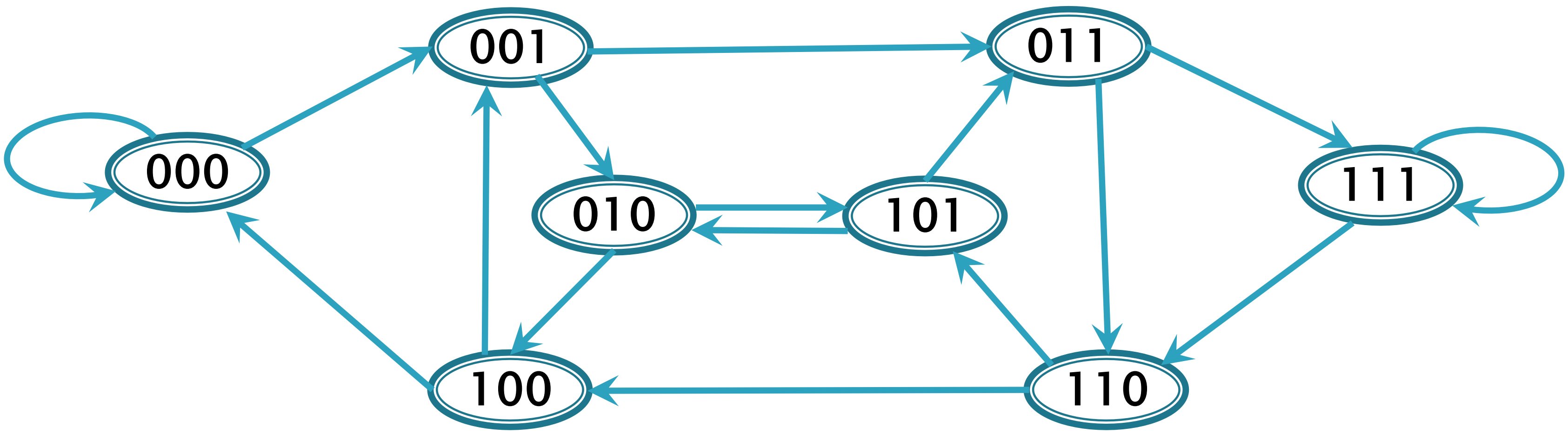
Grafuri euleriene

Universal string problem (de Bruijn, 1946)

- k = 3:   000, 001, 010, 011, 100, 101, 110, 111

Varianta 1: Modelare cu graf de suprapuneri:

  - vârf = k-șir
  - arc = se poate continua cu

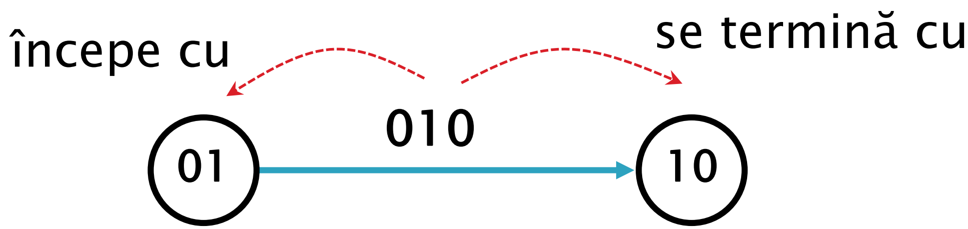


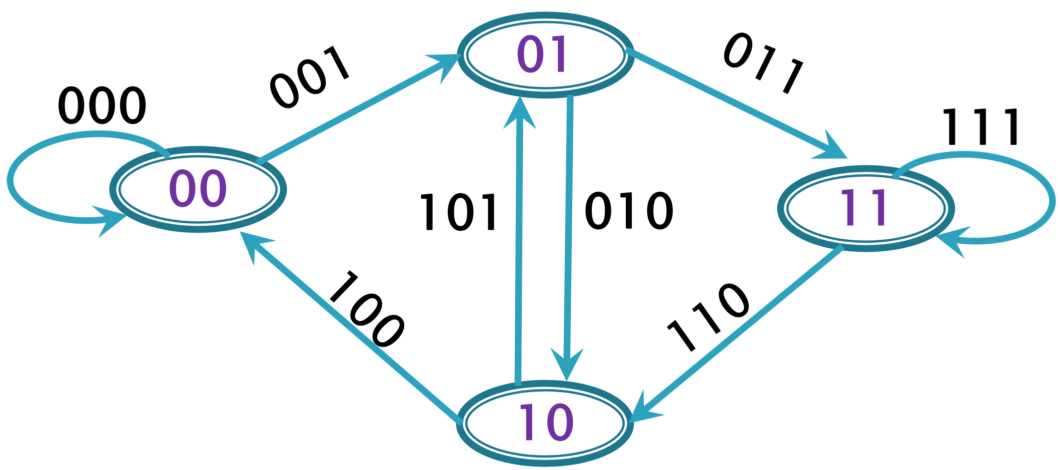
Pb se reduce la a determina dacă există drum hamiltonian în graf – dificil computațional

Grafuri euleriene

Varianta 2: Modelare de Bruijn a surpapunerilor – multigraf:

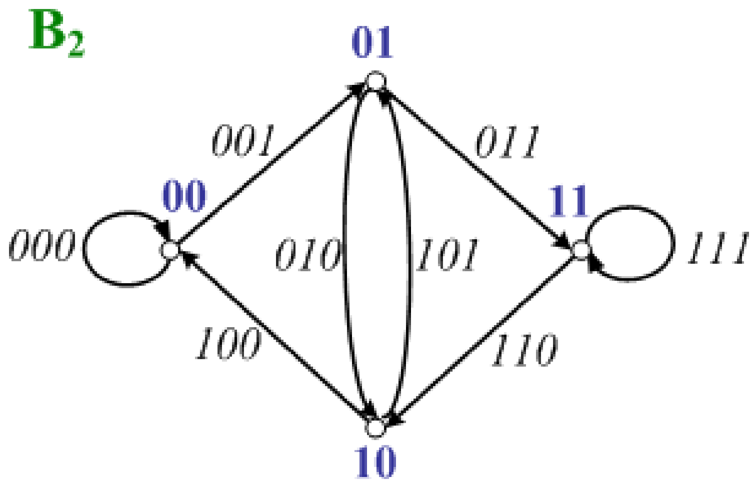
- Arcele corespund k șirurilor
- Vîrfurile corespund (k-1)-șirurilor și arată suprapunerile

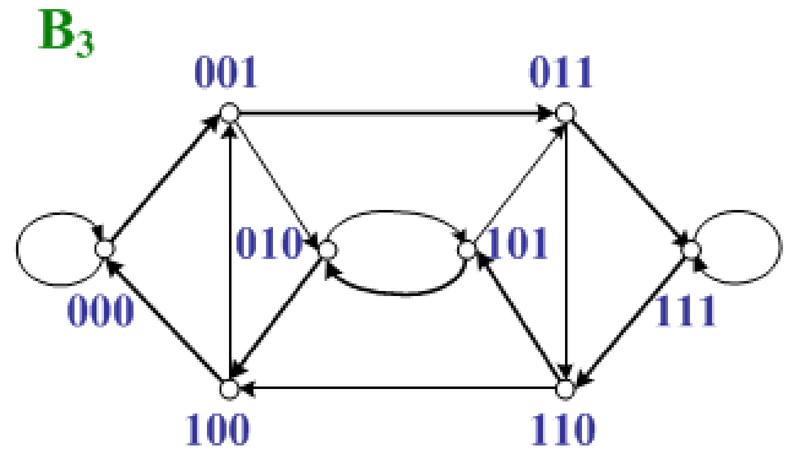


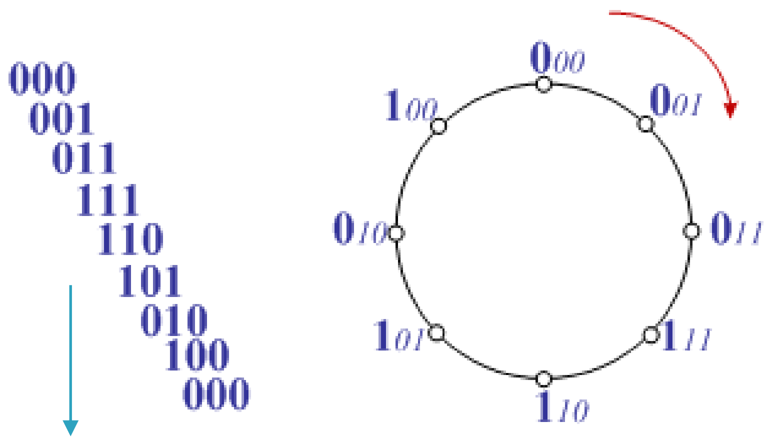


Pb se reduce la a determina dacă există drum/circuit eulerian în graf – polinomial

Grafuri de Bruijn







Soluția la problema lui POSTHUMUS pentru n=3 ⇔
etichetele arcelor unui circuit eulerian în graful B_2

Grafuri de Bruijn

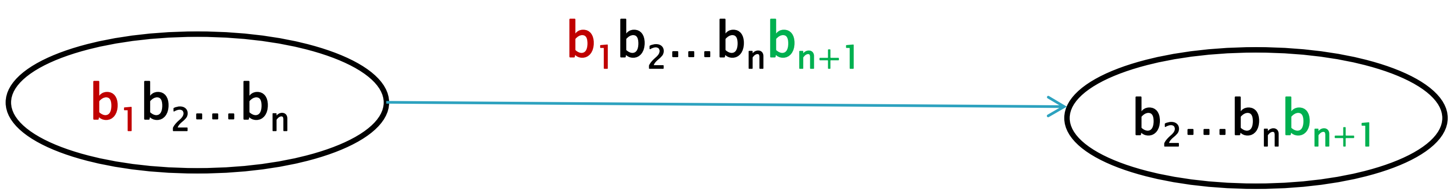
- Multigraf
- $V(B_n) = \{0,1\}^n$ (mai general $\{0,1,\ldots, p\}^n$)

  (sau cuvinte de lungime n peste un alfabet finit)

- $E(B_n)$ etichetate cu $\{0,1\}^{n+1}$ ($\{0,1,\ldots, p\}^{n+1}$)

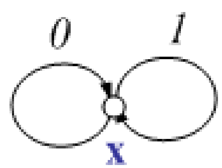
  $b_1 b_2 \ldots b_n b_{n+1}$ etichetează arcul de la

  $b_1 \mathbf{b_2 \ldots b_n}$ la $\mathbf{b_2 \ldots b_n} b_{n+1}$

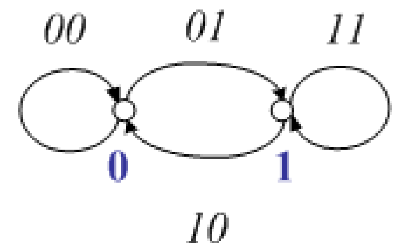


Grafuri de Bruijn

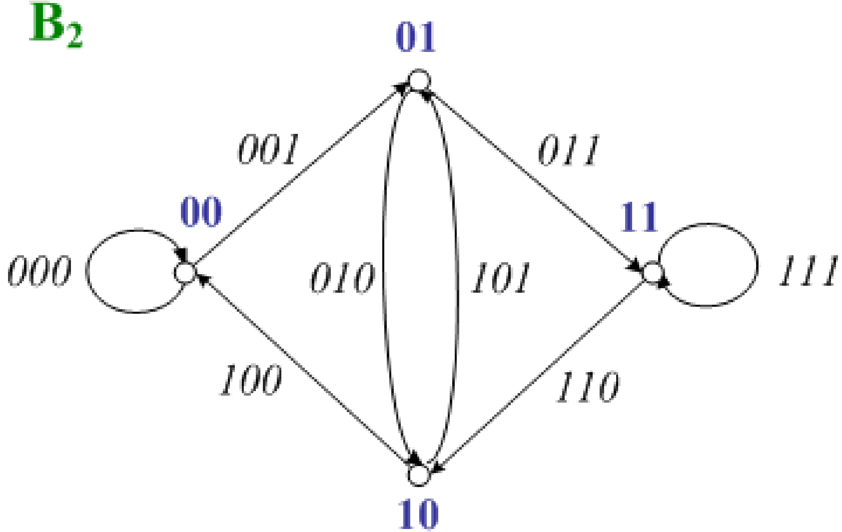
B_0



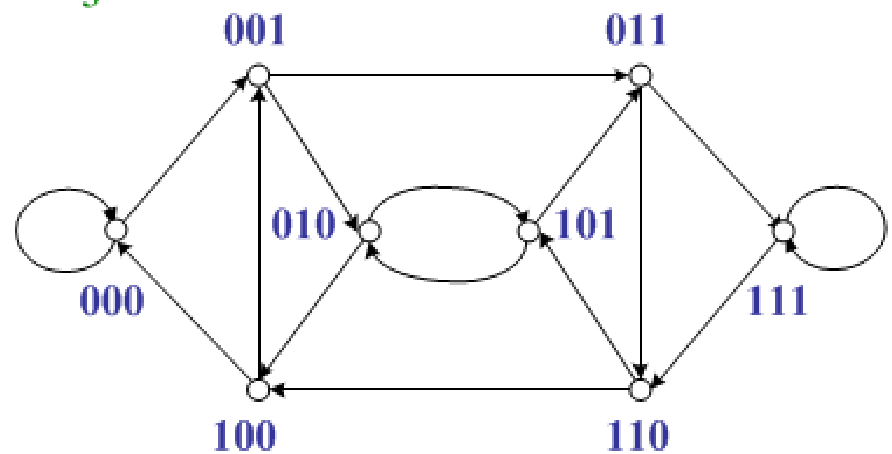
B_1



B_2



B_3

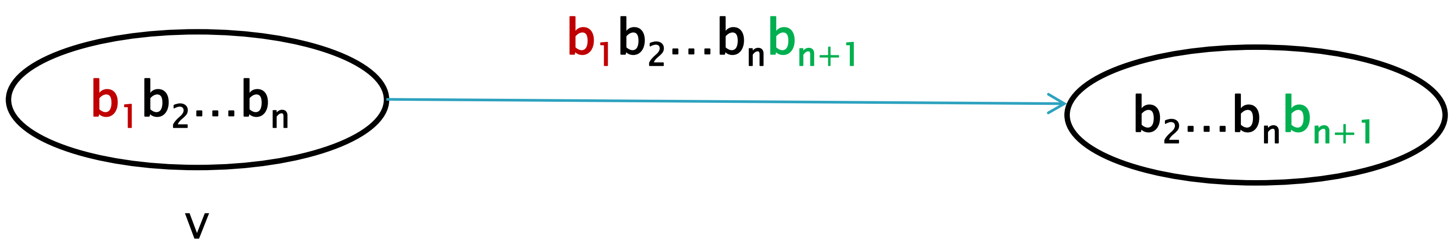


Grafuri de Bruijn

- B_n este eulerian?

  $d^+(v) = ?$

  $d^-(v) = ?$



orice b_{n+1} din alfabet

Grafuri de Bruijn

- B_n este eulerian

  $d^+(v) = |\{0,1\}| = 2$ (mai general $= |\{0,1,\ldots, p\}|$)

  $d^-(v) = d^+(v)$

$$\begin{tikzcd} b_1b_2\ldots b_n \arrow[r, "b_1b_2\ldots b_nb_{n+1}"] & b_2\ldots b_nb_{n+1} \end{tikzcd}$$

v

orice b_{n+1} din alfabet

Grafuri de Bruijn

- **Prima cifră din etichetele arcelor unui circuit eulerian în B_{n-1} – soluție pentru problema lui Posthumus $\Rightarrow$**

  $f(n) = 2^n$

- **Observație**

  Circuit eulerian in $B_{n-1} \leftrightarrow$ circuit hamiltonian in B_n

Descompuneri euleriene în lanțuri

- **k-descompunere euleriană în lanțuri** a unui graf G =

  o mulțime de k lanțuri simple, muchie-disjuncte

  $\Delta = \{P_1, P_2, \ldots, P_k\}$

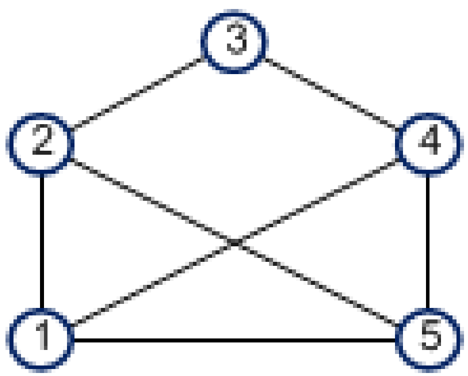
  ale căror muchii induc o k-partiție a lui E(G):

  $E(G) = E(P_1) \cup E(P_2) \cup \ldots \cup E(P_k)$

Descompuneri euleriene în lanțuri

- **Interpretare**

  De câte ori (minim) trebuie să ridicăm creionul de pe hârtie pentru a desena diagrama?



Descompuneri euleriene în lanțuri

Teoremă – Descompunere euleriană

Fie G=(V, E) un graf orientat, conex (= graful neorientat asociat este conex), cu **exact 2k vârfuri de grad impar** (k>0). Atunci există o k-descompunere euleriană a lui G și k este cel mai mic cu această proprietate.