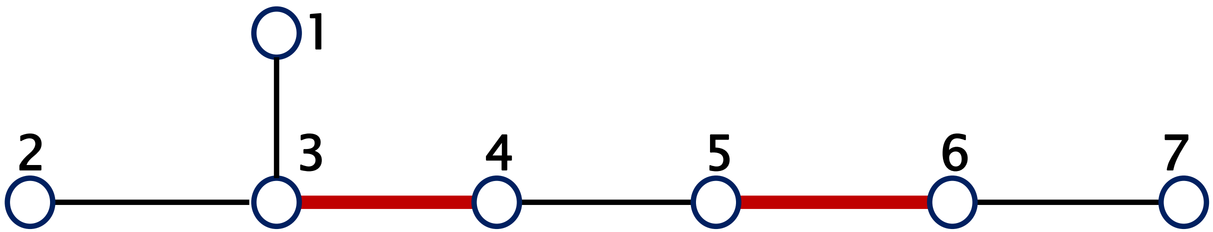
Cuplaje

Fie G = (V, E) un graf simplu şi **M** $\subseteq$ **E**.

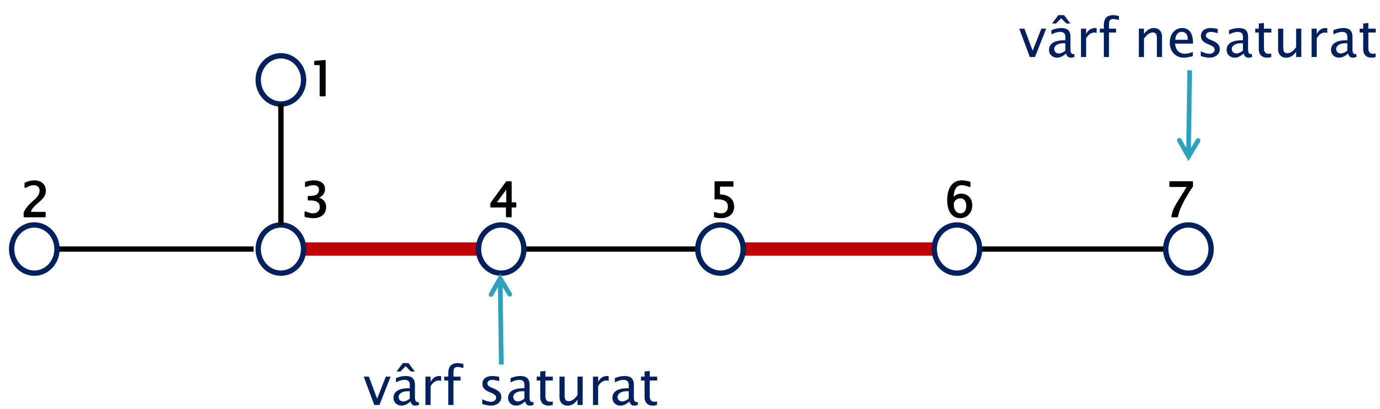
- M s.n **cuplaj** dacă orice două muchii din M sunt neadiacente



cuplaj M={ {3,4}, {5,6} }

Fie G = (V, E) un graf simplu şi **M** $\subseteq$ **E**.

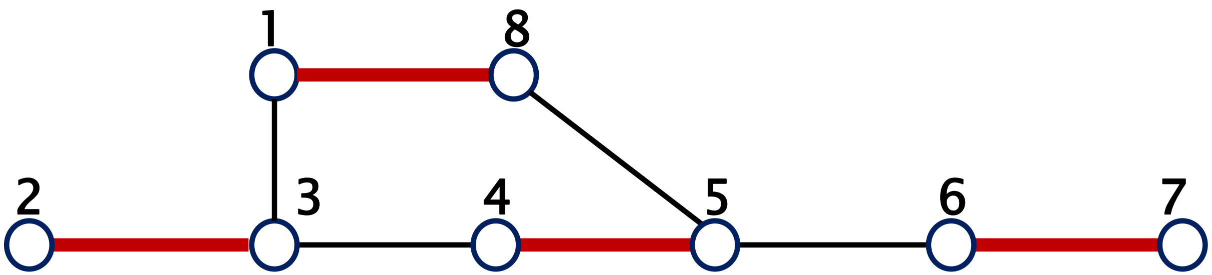
- M s.n **cuplaj** dacă orice două muchii din M sunt neadiacente
- V(M) = mulţimea vârfurilor **M-saturate**
- V(G) - V(M) = mulţimea vârfurilor **M-nesaturate**



- Un cuplaj M^* s.n **cuplaj de cardinal maxim** (**cuplaj maxim**):

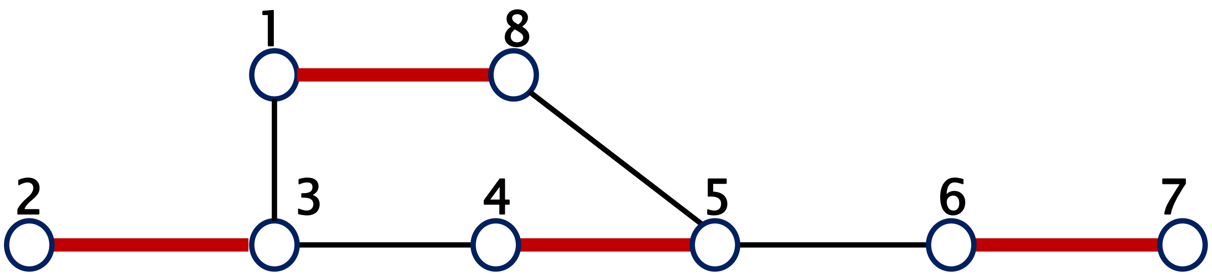
$$|M^*| \geq |M|, \forall M \subseteq E \text{ cuplaj}$$

- Un cuplaj M s.n **cuplaj perfect** dacă orice vârf este M-saturat



cuplaj perfect

- Un cuplaj M s.n **cuplaj perfect** dacă orice vârf este M-saturat

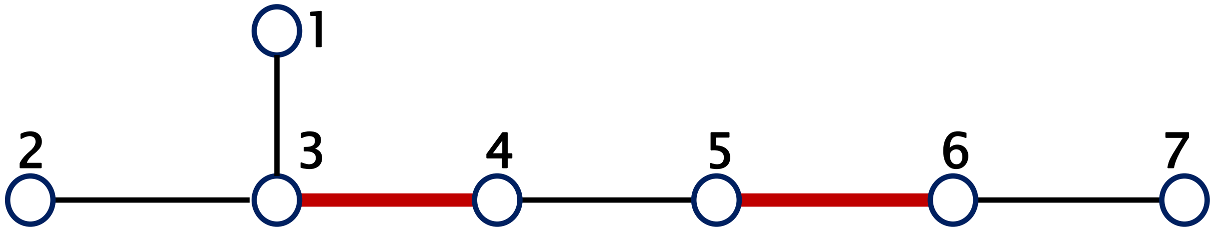


cuplaj perfect

- **Nu orice graf admite un cuplaj perfect**

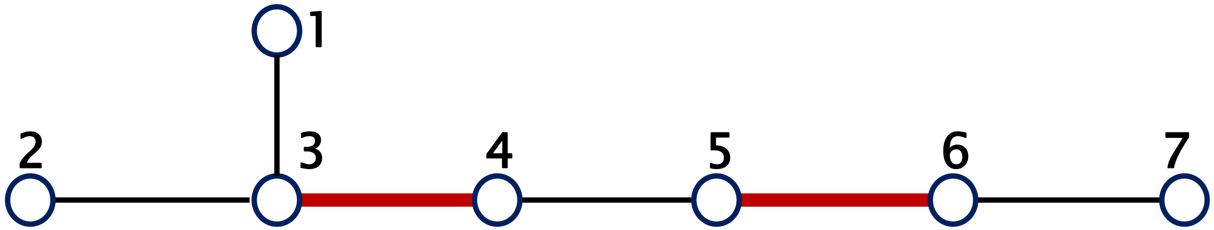
Fie $\mathbf{M} \subseteq \mathbf{E}$ cuplaj.

- Un lanţ elementar P s.n. **lanţ M-alternant** dacă muchiile sale aparţin alternativ lui M şi E – M



Fie **M** ⊆ **E** cuplaj.

- Un lanţ M-alternant P s.n. **lanţ M-alternant deschis** dacă extremităţile sale sunt M-nesaturate



P = [1, 3, 4, 5, 6, 7]

P = [2, 3, 4, 5, 6, 7]

- Fie P un lanţ M-alternant **deschis**

- **Operaţie de transfer** de-a lungul lanţului P = obţinerea unui nou cuplaj M' din M astfel:

$$\mathbf{M' = M \,\Delta\, E(P) = (M - E(P)) \cup (E(P) - M)}$$

- **Observaţie**

$$\mathbf{|M'| = |M| + 1}$$

Condiţii necesare şi suficiente ca un cuplaj să fie de cardinal maxim

Teorema lui BERGE

Fie $G=(V, E)$ un graf simplu cu $E \neq \emptyset$, şi $M \subseteq E$ un cuplaj. Avem echivalenţa:

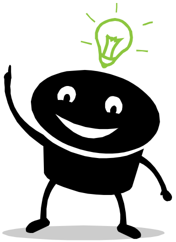
M este **cuplaj de cardinal maxim** în G $\Leftrightarrow$

nu există nici un lanţ M−alternant **deschis** în G

Idee algoritm de determinare a unui cuplaj maxim

- **Fie** M un cuplaj arbitrar în G (exp. $\emptyset$)
- Cât timp **există un lanţ M–alternant deschis P în G**
  - determină un astfel de lanţ P
  - $M = M \,\Delta\, E(P)$

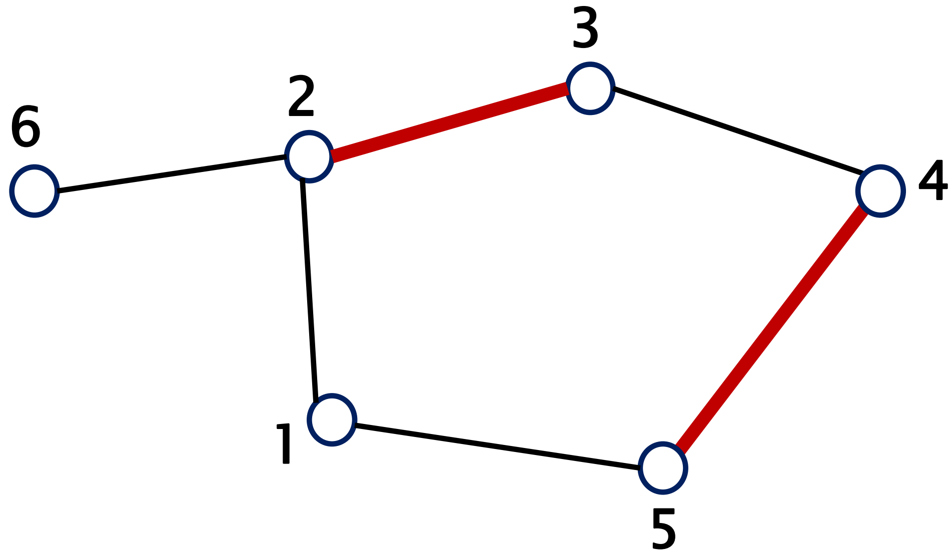
Cum determinăm un lanţ M-alternant deschis?

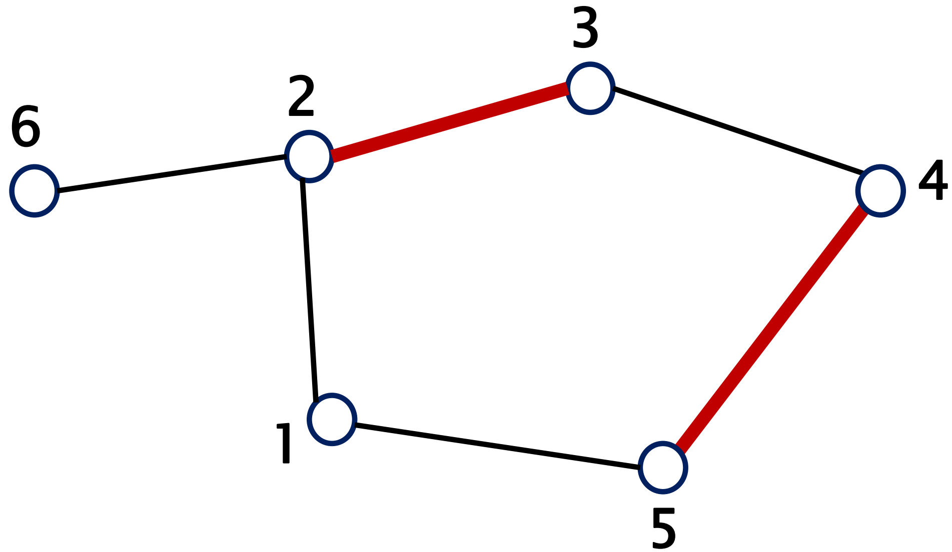


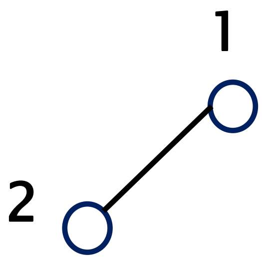
Prin parcurgerea grafului, vector tata…

Prin parcurgere nu determinăm toate lanţurile elementare dintr-un graf.

Dacă există un lanţ M-alternant deschis, va fi sigur el găsit printr-o parcurgere?


6
2
3
4
1
5





Prin parcurgere (BF sau DF) din 1 nu găsim lanţul M-alternant deschis, deoarece 2 este deja vizitat ca fiu al lui 1

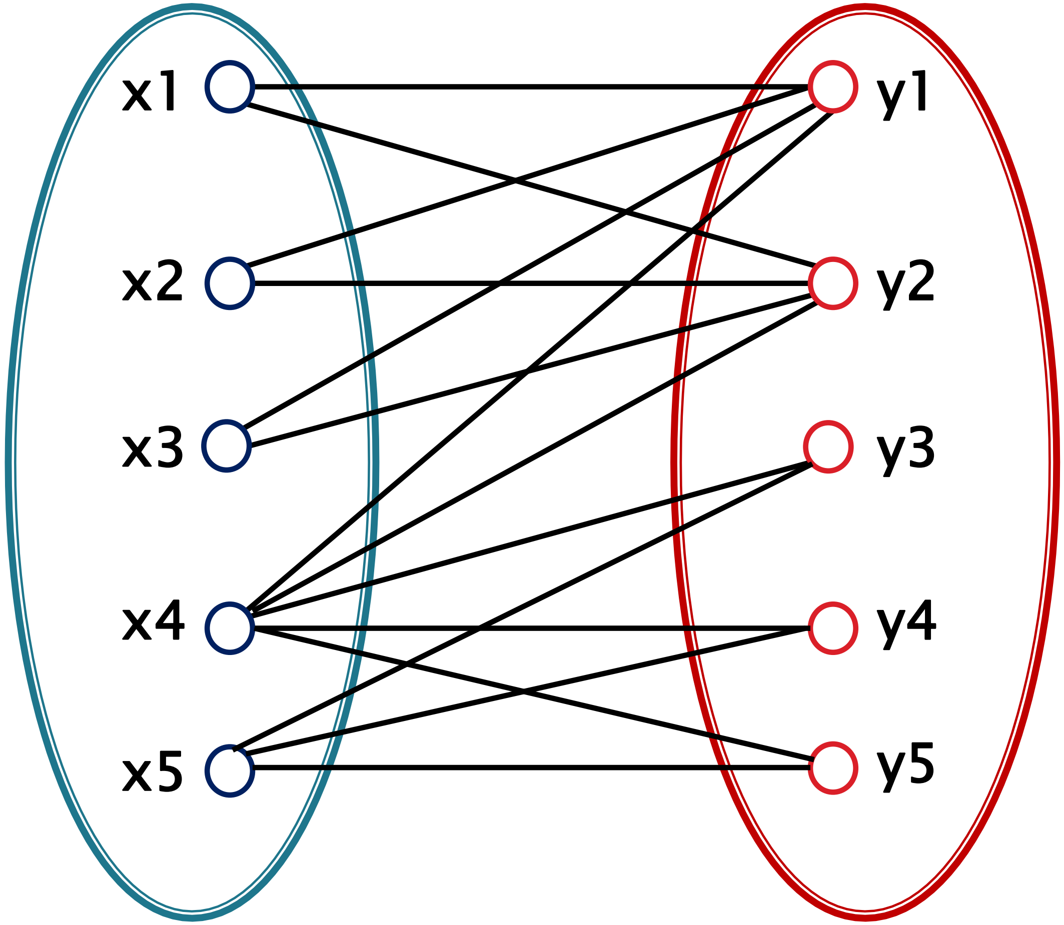
Prin parcurgere nu putem determina toate lanţurile elementare dintr-un graf.

Dacă există un lanţ M-alternant deschis, va fi sigur el găsit printr-o parcurgere?

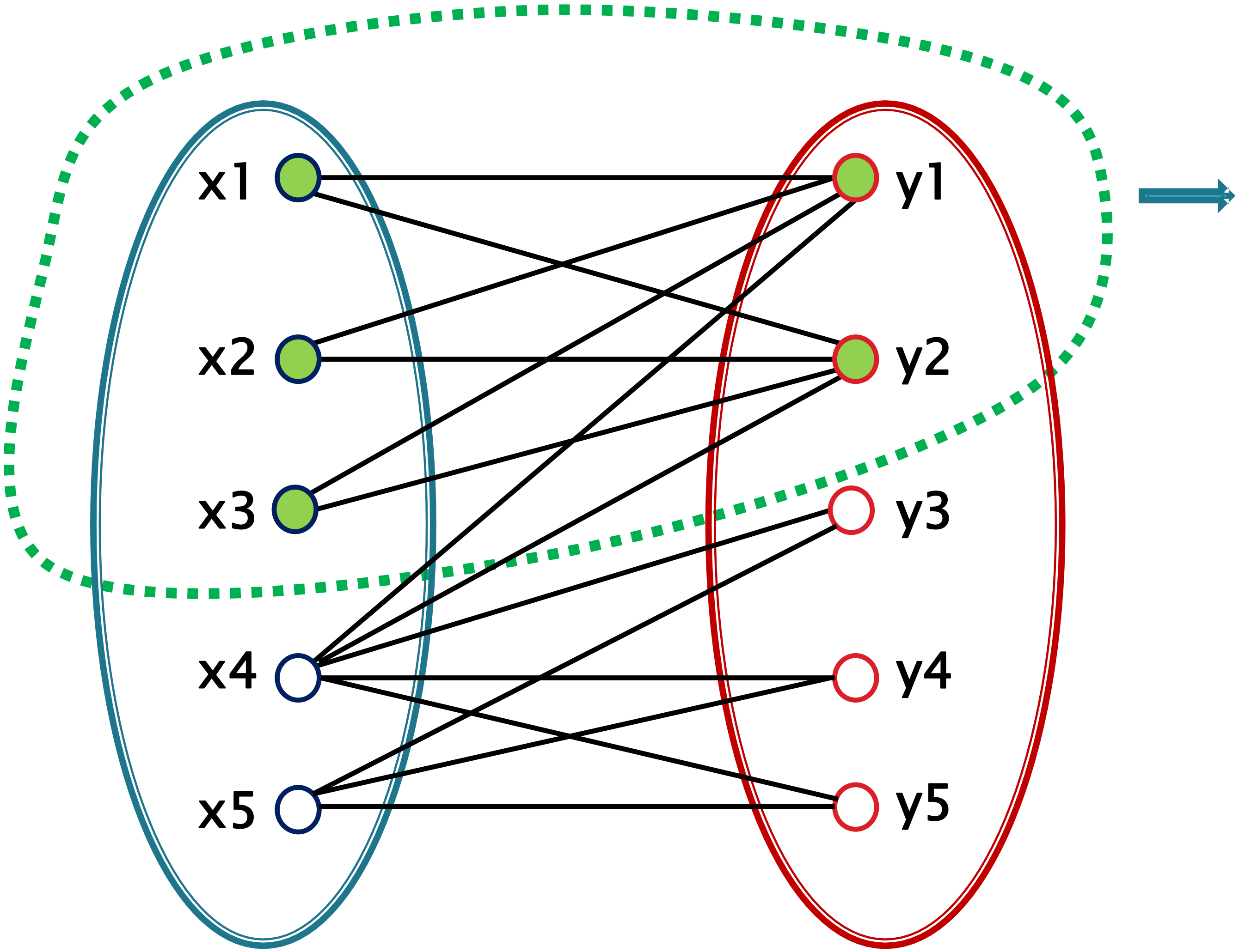
- Grafuri bipartite

Cuplaje în grafuri bipartite

Cuplaje perfecte în grafuri bipartite



Cuplaje perfecte în grafuri bipartite



nu există cuplaj perfect

Teorema lui HALL

Notăm

$$N_G(S) = \bigcup_{s \in S} N_G(s) = \{u \mid \exists s \in S \text{ cu } \{s,u\} \in E(G)\}$$

= mulţimea vecinilor vârfurilor din S.

Teorema lui HALL

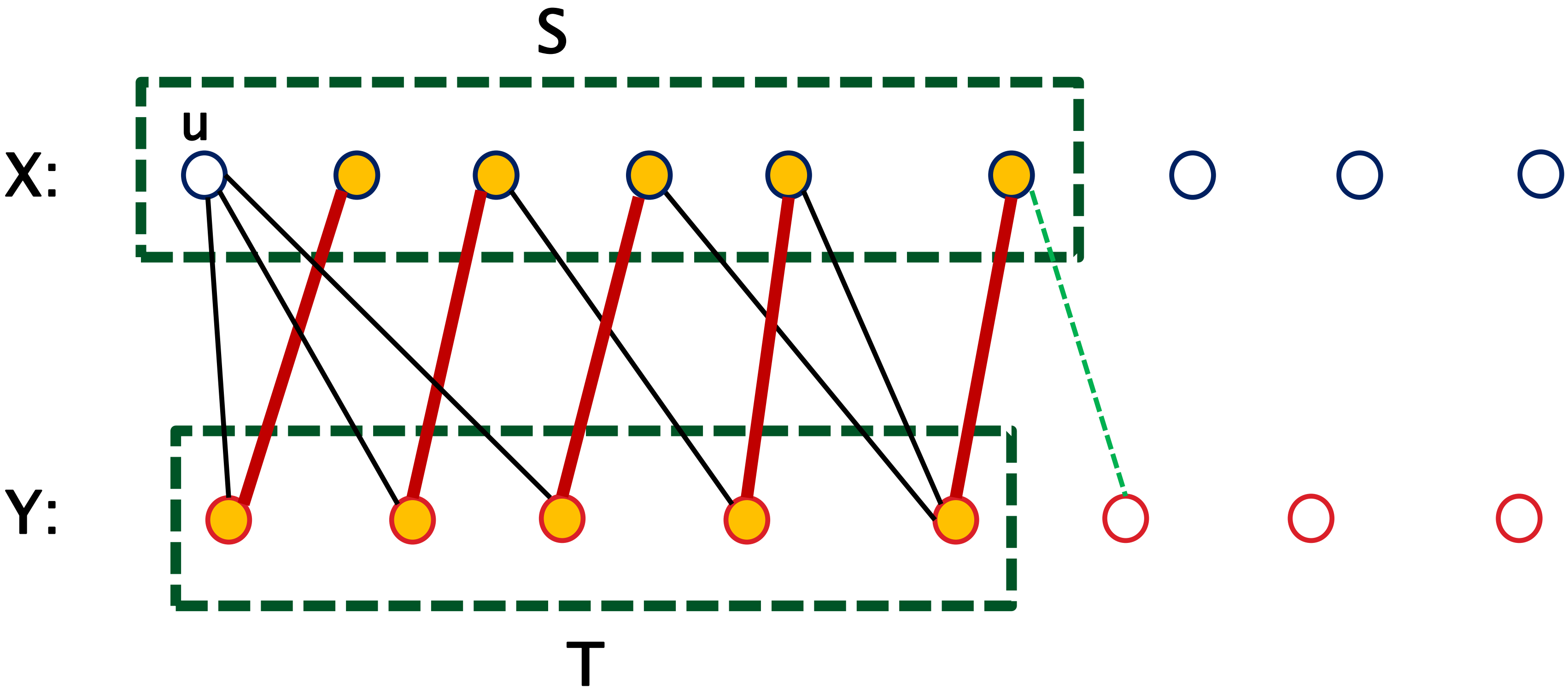
Fie $G=(V, E)$ un graf **bipartit** cu bipartiţia $V = \mathbf{X} \cup \mathbf{Y}$.

Atunci

G conţine un **cuplaj care saturează toate vârfurile din X** (un cuplaj **al lui X în Y**) $\Leftrightarrow$

$\forall \mathbf{S} \subseteq \mathbf{X}, |\mathbf{S}| \leq |\mathbf{N_G(S)}|$

Teorema lui HALL



M* cuplaj de cardinal maxim

Corolar 1 (Teorema căsătoriei)

Fie $G=(V, E)$ un graf bipartit p-regulat ($p>0$). Atunci G are un cuplaj perfect.

Corolar 2

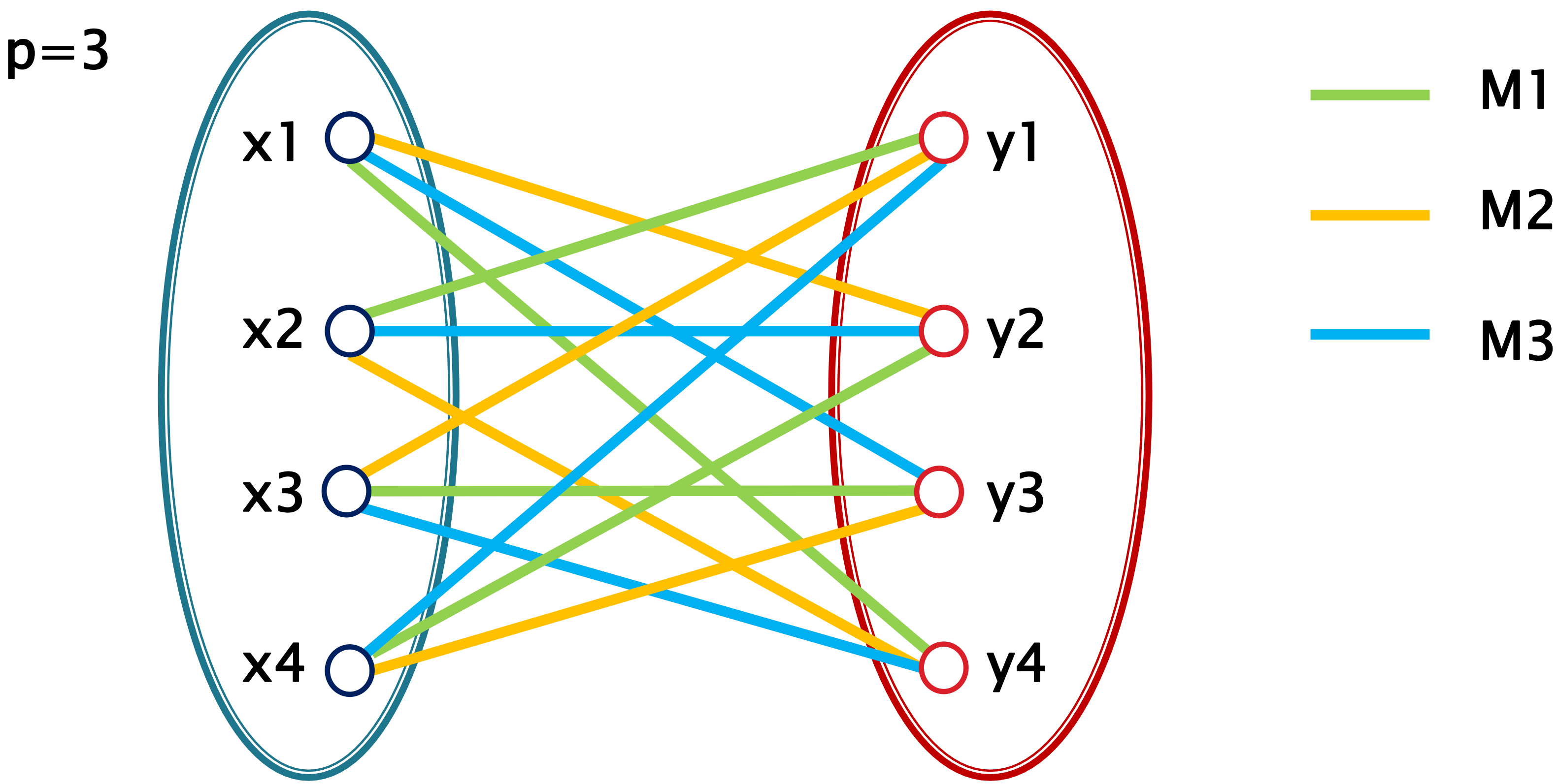
Fie $G=(V, E)$ un graf bipartit p-regulat ($p>0$). Atunci există M_1, M_2,..., M_p cuplaje perfecte disjuncte în G cu

$$\mathbf{E(P) = M_1 \cup M_2 \cup \ldots \cup M_p}$$

Corolar 2

Fie G=(V, E) un graf bipartit p-regulat (p>0). Atunci există M_1, M_2,..., M_p cuplaje perfecte disjuncte în G cu

$$\mathbf{E(P) = M_1 \cup M_2 \cup \ldots \cup M_p}$$



Corolar 2

Fie $G=(V, E)$ un graf bipartit p-regulat ($p>0$). Atunci există $M_1, M_2, \ldots, M_p$ cuplaje perfecte disjuncte în G cu

$$\mathbf{E(P) = M_1 \cup M_2 \cup \ldots \cup M_p}$$

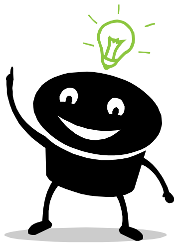
Interpretare: Muchiile unu graf bipartit p-regulat pot fi colorate cu p culori astfel încât orice două muchii adiacente au culori diferite

- p este numărul minim de culori necesar unei astfel de colorări a muchiilor

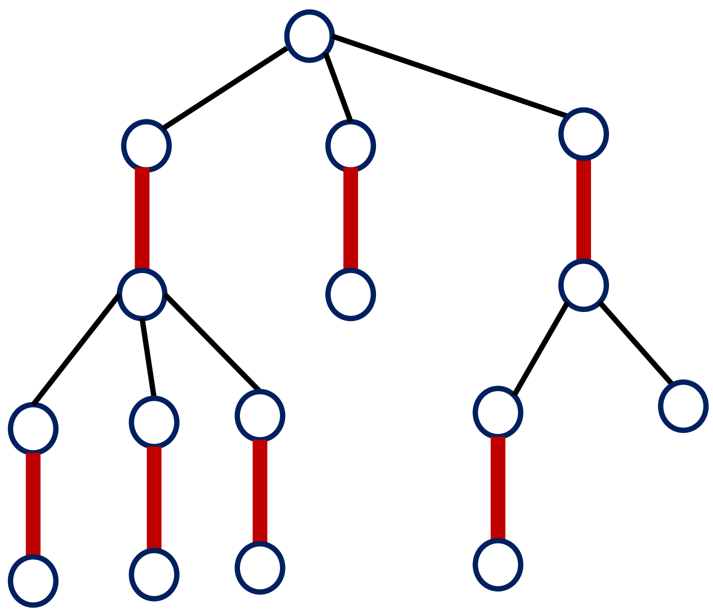
Algoritmul ungar

Amintim următorul algoritm generic de determinare a unui cuplaj maxim:

- **Fie** M un cuplaj arbitrar în G (exp. $\varnothing$)
- Cât timp **există un lanţ M–alternant deschis P în G**
  - determină un astfel de lanţ P
  - $M = M \,\Delta\, E(P)$

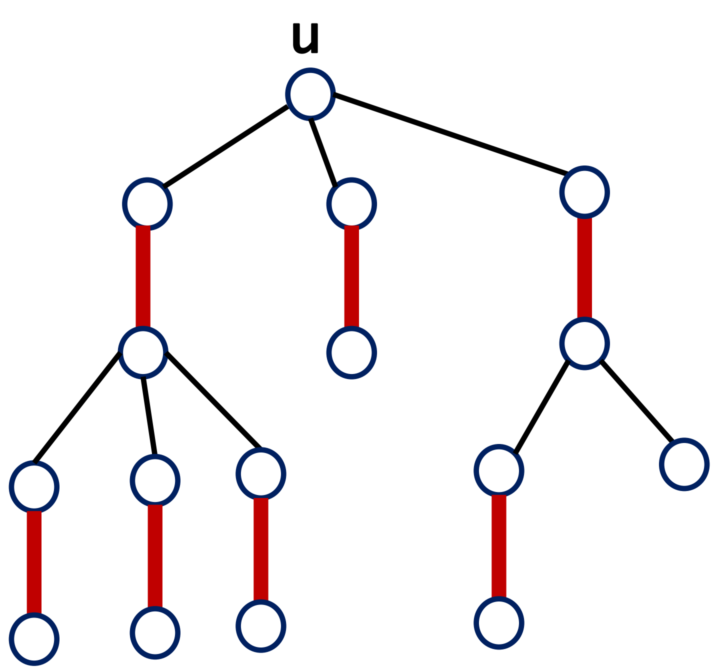


Pentru grafuri bipartite putem determina lanţuri M–alternante deschise **parcurgând “alternant”** graful dintr–un vârf nesaturat şi construind un **arbore M–alternant**



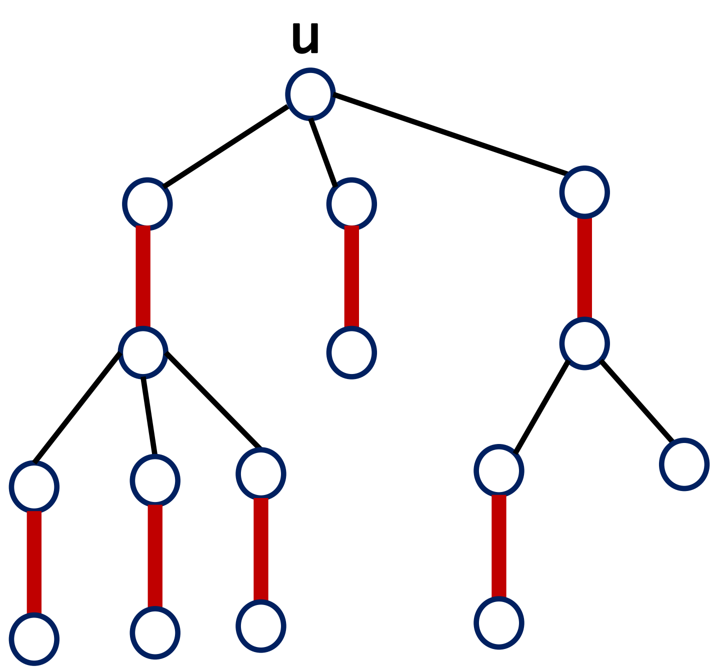
Fie G un graf bipartit, M un cuplaj în G şi u un vârf **M-nesaturat**

- Un **arbore M-alternant cu rădăcină u în G** este un subgraf al lui G cu proprietăţile:
  -
  -



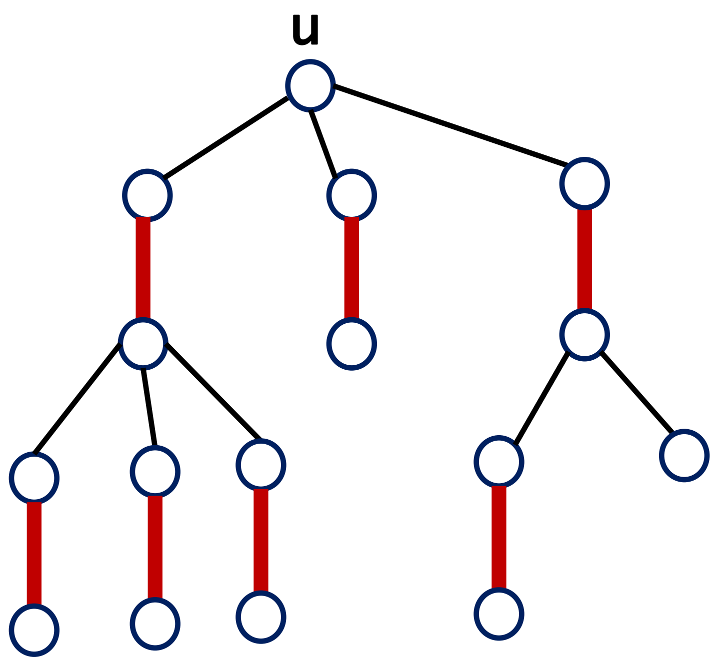
Fie G un graf bipartit, M un cuplaj în G şi u un vârf **M-nesaturat**

- Un **arbore M-alternant cu rădăcină u în G** este un subgraf al lui G cu proprietăţile:
  - este arbore cu rădăcina u şi
  -



Fie G un graf bipartit, M un cuplaj în G şi u un vârf **M-nesaturat**

- Un **arbore M-alternant cu rădăcină u în G** este un subgraf al lui G cu proprietăţile:
  - este arbore cu rădăcina u şi
  - pentru orice vârf v al arborelui, unicul u-v lanţ elementar din arbore este M-alternant



Fie G un graf bipartit, M un cuplaj în G şi u un vârf **M-nesaturat**

- Un **arbore M-alternant cu rădăcină u în G** este un subgraf al lui G cu proprietăţile:
  - este arbore cu rădăcina u şi
  - pentru orice vârf v al arborelui, unicul u-v lanţ elementar din arbore este M-alternant

- Un **arbore M-alternant** s.n. **maximal** dacă nu mai poate fi extins cu alte vârfuri

Algoritmul ungar

- **Intrare**: G – graf bipartit cu bipartiţia $V = X \cup Y$
- **Ieşire**: Un **cuplaj al lui X în Y**, dacă există

Algoritmul ungar

- **Schema**:
  1. Fie M un cuplaj arbitrar în G (de exemplu $\varnothing$)

Algoritmul ungar

- **Schema**:
  1. Fie M un cuplaj arbitrar în G (de exemplu $\emptyset$)
  2. Cât timp M nu saturează toate vârfurile din X execută

Algoritmul ungar

- **Schema**:
  1. Fie M un cuplaj arbitrar în G (de exemplu $\emptyset$)
  2. Cât timp M nu saturează toate vârfurile din X execută
     - Alege u un vârf **M-nesaturat** din X

Algoritmul ungar

- **Schema**:
  1. Fie M un cuplaj arbitrar în G (de exemplu $\emptyset$)
  2. Cât timp M nu saturează toate vârfurile din X execută
     - Alege u un vârf **M-nesaturat** din X
     - Construieşte un arbore M-alternant cu rădăcina în u până când

Algoritmul ungar

- **Schema**:
  1. Fie M un cuplaj arbitrar în G (de exemplu $\emptyset$)
  2. Cât timp M nu saturează toate vârfurile din X execută
     - Alege u un vârf **M-nesaturat** din X
     - Construieşte un arbore M-alternant cu rădăcina în u până când
       a. un vârf nesaturat y este adăugat la arbore;

Algoritmul ungar

- **Schema**:
  1. Fie M un cuplaj arbitrar în G (de exemplu $\emptyset$)
  2. Cât timp M nu saturează toate vârfurile din X execută
     - Alege u un vârf **M-nesaturat** din X
     - Construieşte un arbore M-alternant cu rădăcina în u până când
       a. un vârf nesaturat y este adăugat la arbore;
          fie P unicul u-y lanţ elementar din arbore
          $M \leftarrow M \,\Delta\, E(P)$

Algoritmul ungar

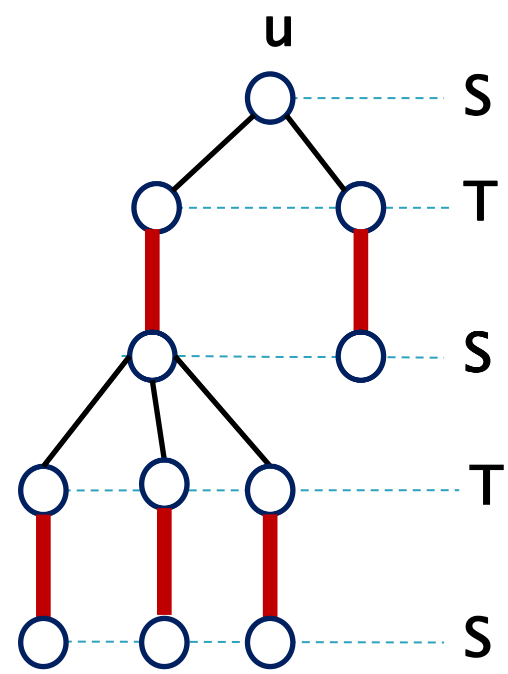
- **Schema**:
  1. Fie M un cuplaj arbitrar în G (de exemplu $\emptyset$)
  2. Cât timp M nu saturează toate vârfurile din X execută
     - Alege u un vârf **M-nesaturat** din X
     - Construieşte un arbore M-alternant cu rădăcina în u până când
       a. un vârf nesaturat y este adăugat la arbore;
          fie P unicul u-y lanţ elementar din arbore
          $M \leftarrow M \,\Delta\, E(P)$
       b. Arborele nu mai poate fi extins (este maximal).
          STOP, NU există cuplaj al lui X în Y

Algoritmul ungar

- **Detalii**: Construcţia unui arbore T M-alternant cu rădăcina în u ("**parcurgere alternantă**")

Algoritmul ungar

- **Detalii**: Construcţia unui arbore T M-alternant cu rădăcina în u ("**parcurgere alternantă**")

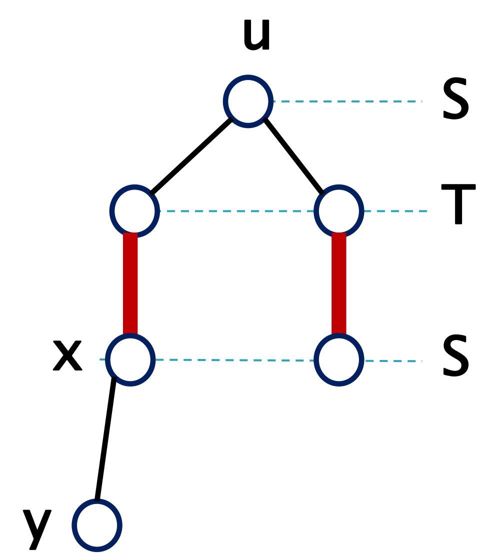


- Notăm cu
  - S = V(T) $\cap$ X
  - T = V(T) $\cap$ Y

Algoritmul ungar

- **Detalii**: Construcţia unui arbore T M-alternant cu rădăcina în u ("parcurgere alternantă")

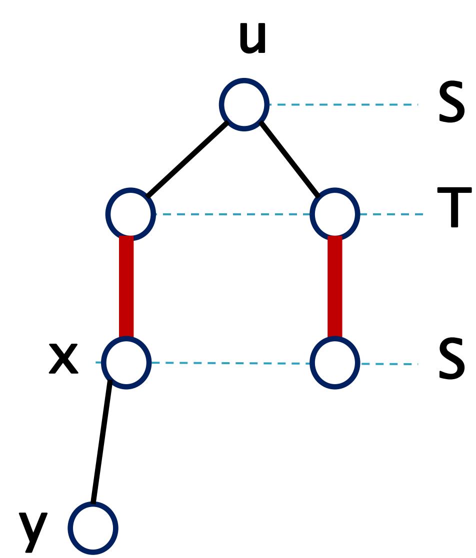
1. $V(T) = \{u\}$, $S = \{u\}$, $T=\emptyset$



Algoritmul ungar

- **Detalii**: Construcţia unui arbore T M-alternant cu rădăcina în u ("parcurgere alternantă")

1. $V(T) = \{u\}$, $S = \{u\}$, $T=\varnothing$
2. Cât timp $N(S) - T \neq \varnothing$ (mai există vârfuri accesibile) execută
   - alege o muchie xy cu $x \in S$ şi $y \notin T$ (x vizitat, y nevizitat)



Algoritmul ungar

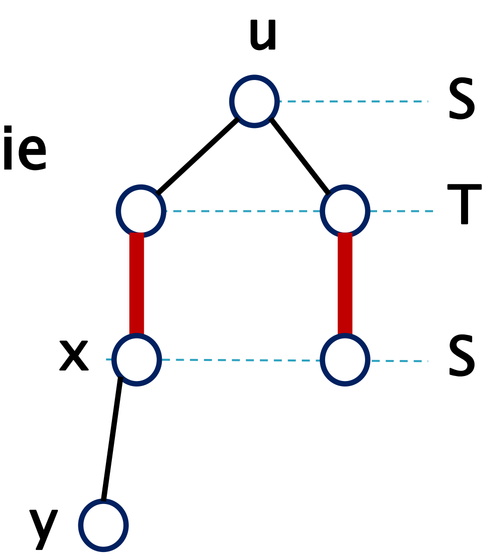
- **Detalii**: Construcţia unui arbore T M-alternant cu rădăcina în u ("parcurgere alternantă")

1. $V(T) = \{u\}$, $S = \{u\}$, $T = \emptyset$
2. Cât timp $N(S) - T \neq \emptyset$ (mai există vârfuri accesibile) execută
   - alege o muchie xy cu $x \in S$ şi $y \notin T$
   - dacă y este M-nesaturat **STOP construcţie**

     treci la pasul 2.a:

       fie P u-y lanţul elementar din T

       $M \leftarrow M \Delta E(P)$;



Algoritmul ungar

- **Detalii**: Construcţia unui arbore T M–alternant cu rădăcina în u ("parcurgere alternantă")

1. V(T) = {u}, S = {u}, T=∅
2. Cât timp N(S) – T ≠ ∅ (mai există vârfuri accesibile) execută
   - alege o muchie xy cu x∈S şi y∉T
   - dacă y este M–nesaturat **STOP construcţie**

     treci la pasul 2.a:

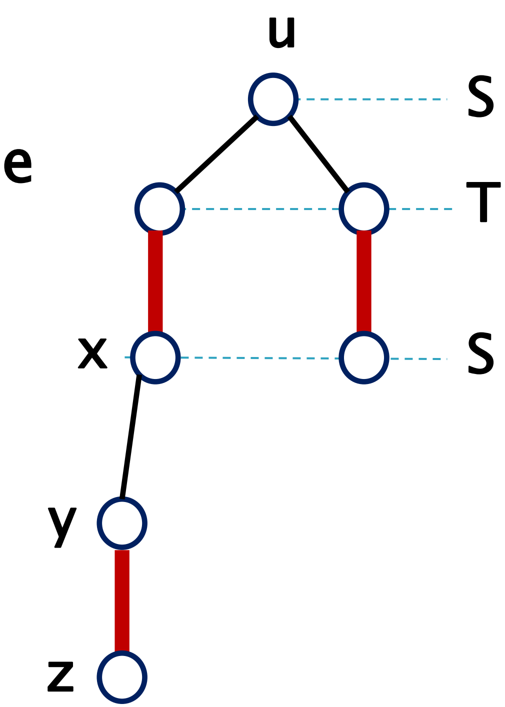
     fie P u–y lanţul elementar din T

     $M \leftarrow M \,\Delta\, E(P)$;
   - altfel

     fie z unicul vârf cu yz ∈ M

     $S \leftarrow S \cup \{z\}$, $T \leftarrow T \cup \{y\}$ (vizităm vârfurile y şi z)

     $E(T) \leftarrow E(T) \cup \{xy, yz\}$ (adăugăm şi pe y şi pe z în arbore)



Algoritmul ungar - sugestii de implementare

Construcţia arborelui M-alternant folosind BF din u

1. Iniţializări
2. Cât timp $C \neq \emptyset$ execută

    $x \leftarrow$ extrage(C)

    pentru y vecin al lui x execută

        daca viz[y]=0 atunci

            tata[y] $\leftarrow$ x; viz[y] $\leftarrow$1

Algoritmul ungar – sugestii de implementare

Construcţia arborelui M-alternant folosind BF din u

1. Iniţializări
2. Cât timp $C \neq \emptyset$ execută

    $x \leftarrow$ extrage(C)

    pentru y vecin al lui x execută

    daca viz[y]=0 atunci

    tata[y] $\leftarrow$ x; viz[y] $\leftarrow$1

    dacă y este M-nesaturat

    fie P u-y lanţul elementar din T (obţinut folosind tata)

    $M \leftarrow M \Delta E(P)$; **break**

    altfel

    fie z unicul vârf cu $yz \in M$

    viz[z] $\leftarrow$1; tata[z] $\leftarrow$ y;

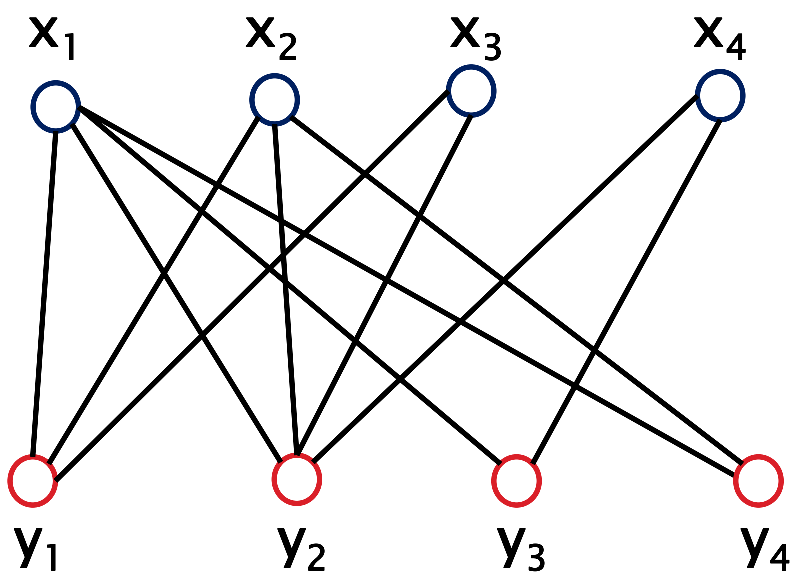
    adauga(z, C) (!!**nu si y**)

Algoritmul ungar – sugestii de implementare

Construcţia arborelui M-alternant se poate face şi folosind metoda de parcurgere DF

Algoritmul ungar
Exemplu

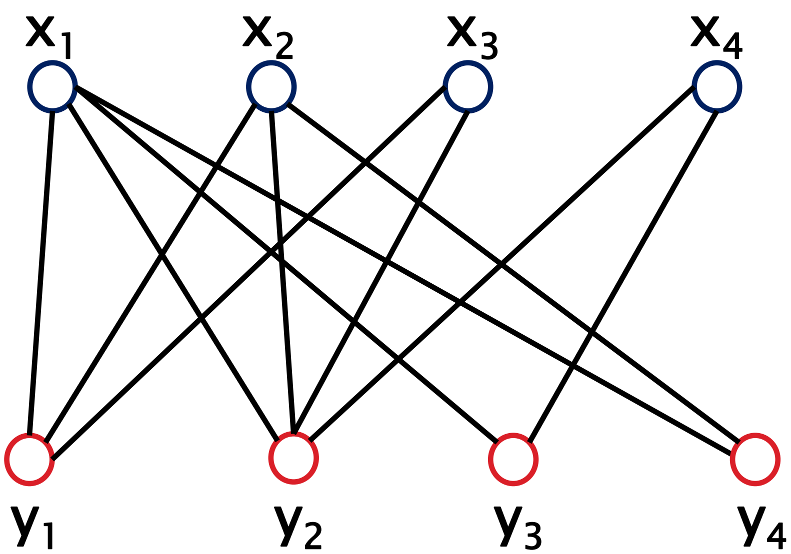
Iniţializări



$M = \emptyset$

Etapa 1

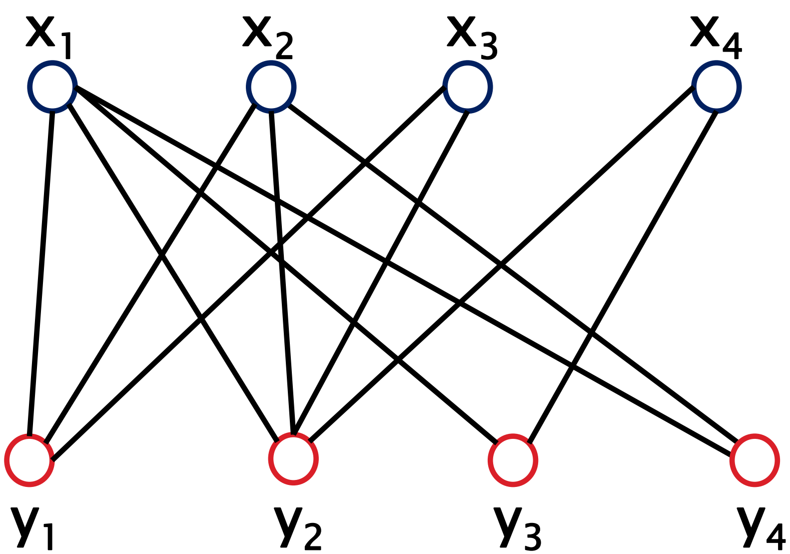
$M = \varnothing$



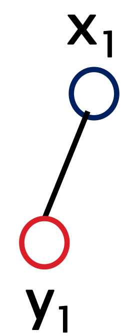
$u = x_1$

Etapa 1

$M = \varnothing$

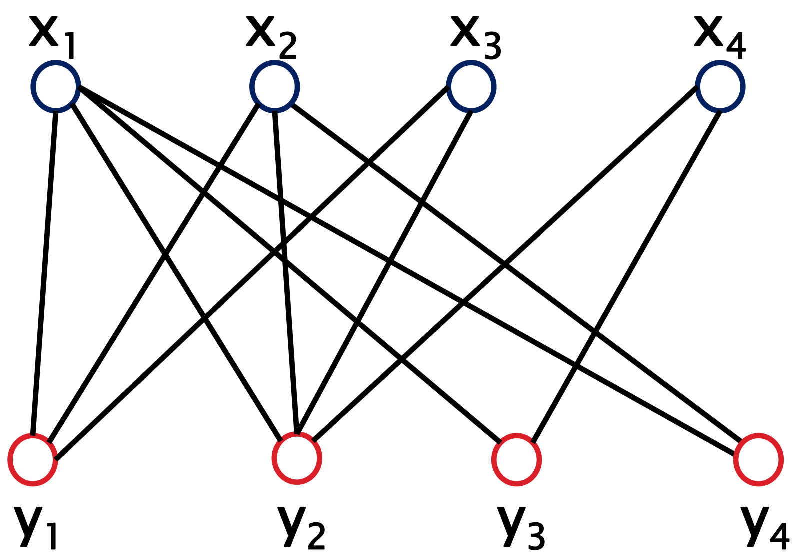


$u = x_1$

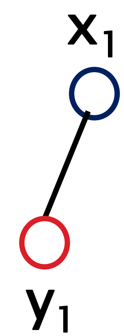


Etapa 1

$M = \varnothing$



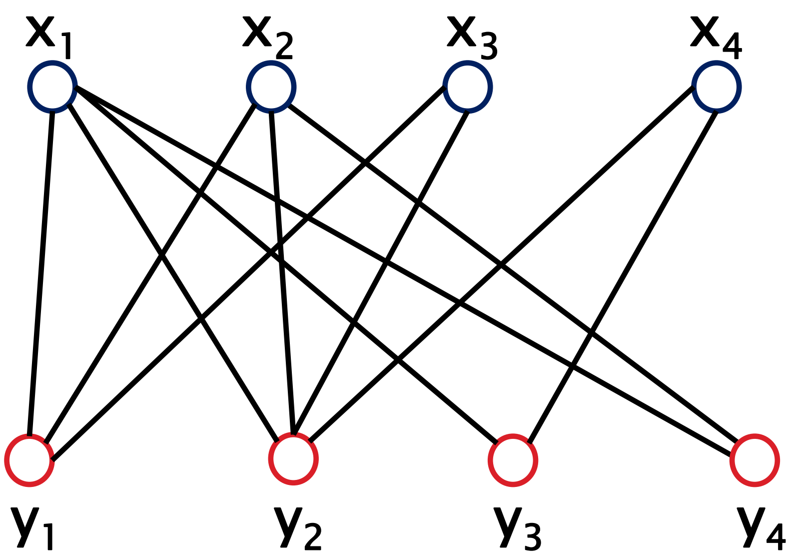
$u = x_1$



STOP

Etapa 1

$M = \varnothing$



$u = x_1$

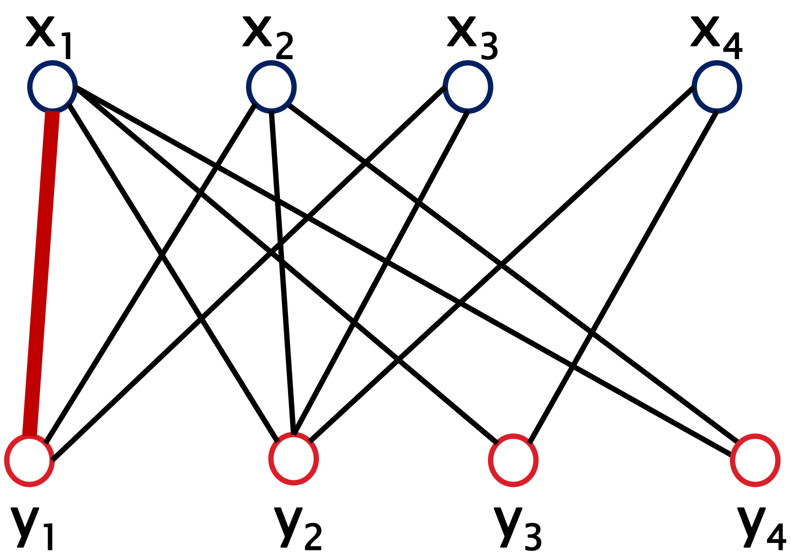
STOP

$P = [x_1, y_1]$

$M_1 = \{x_1\ y_1\}$

Etapa 2

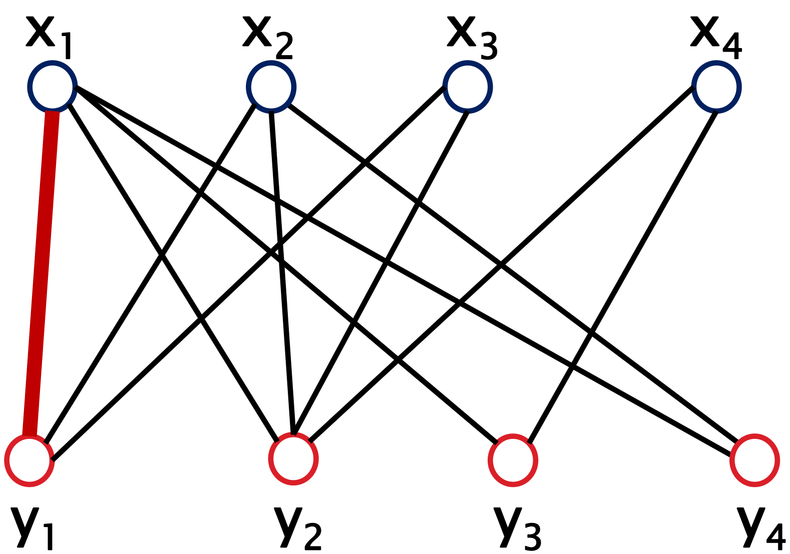
$M = \{x_1y_1\}$

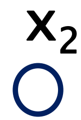


Etapa 2

$M = \{x_1 y_1\}$

$u = x_2$

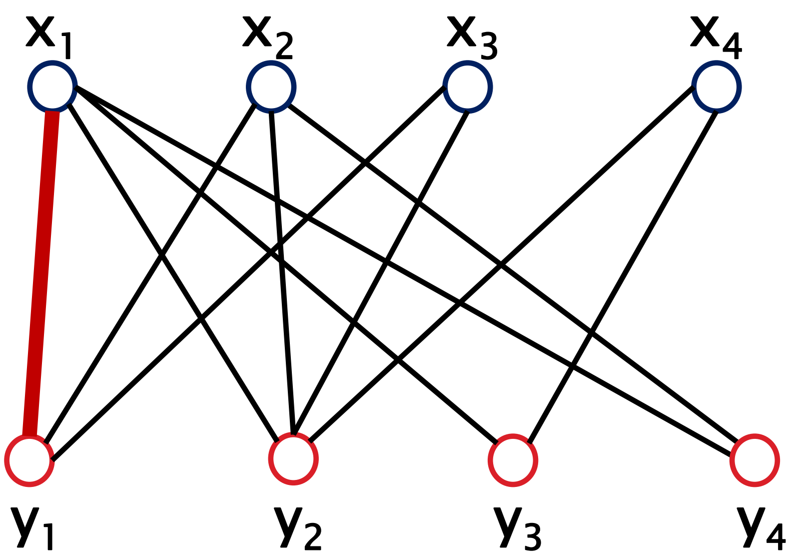


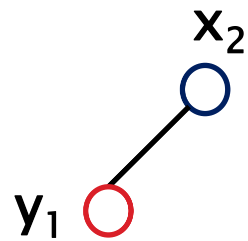


Etapa 2

$M = \{x_1 y_1\}$

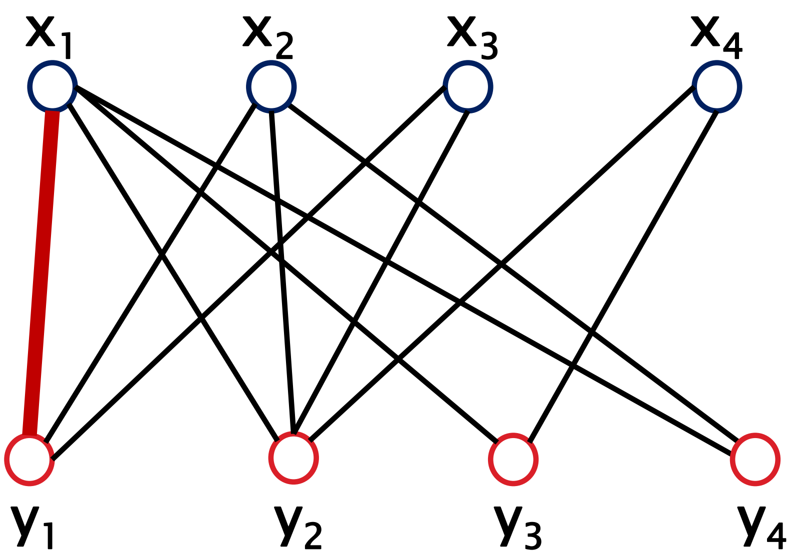
$u = x_2$

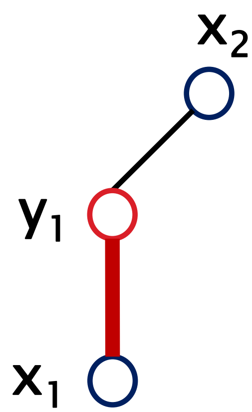




Etapa 2

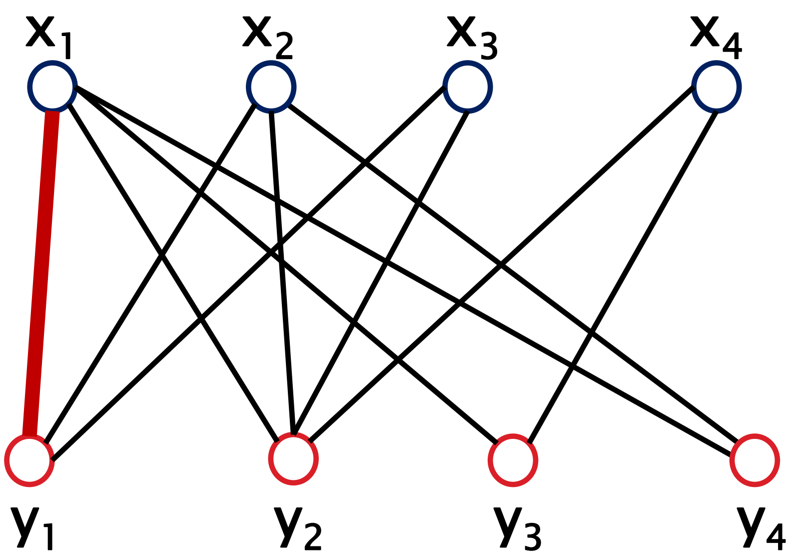
$M = \{x_1 y_1\}$ $u = x_2$



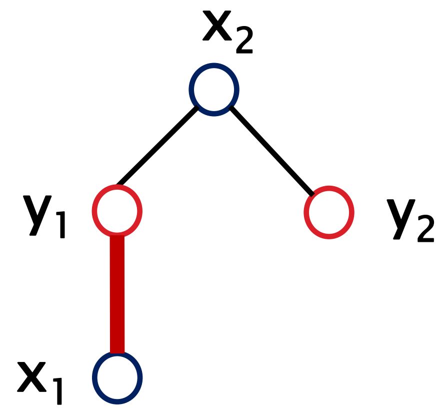


Etapa 2

$M = \{x_1y_1\}$



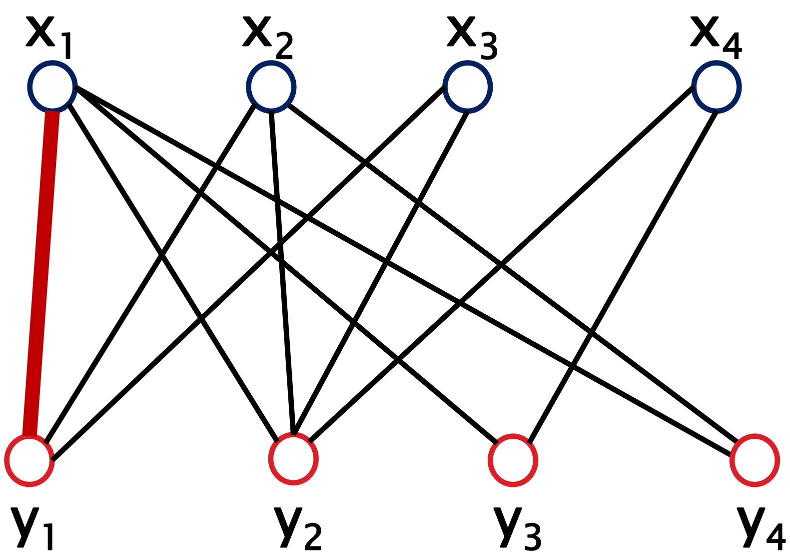
$u = x_2$

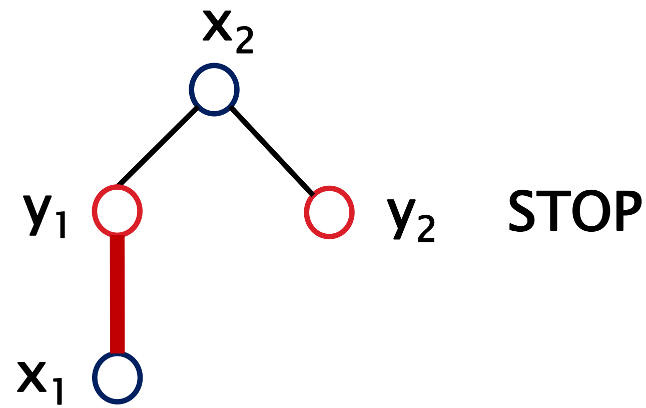


Etapa 2

$M = \{x_1y_1\}$

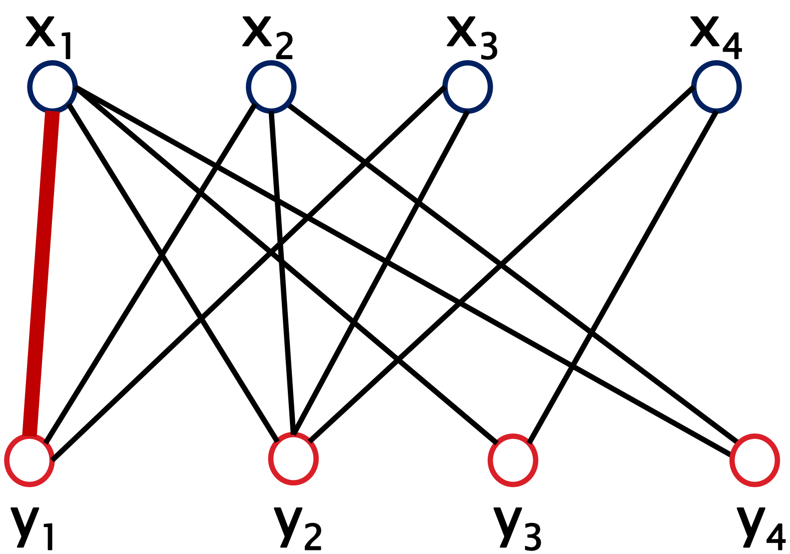
$u = x_2$



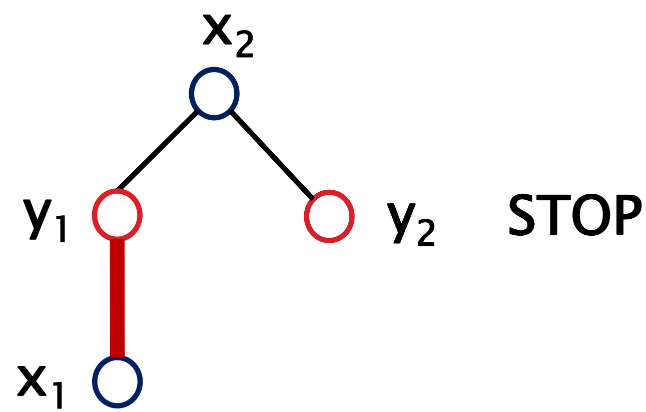


Etapa 2

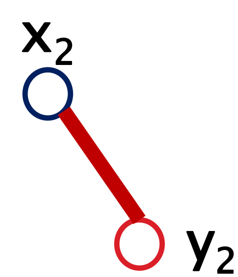
$M = \{x_1y_1\}$



$u = x_2$



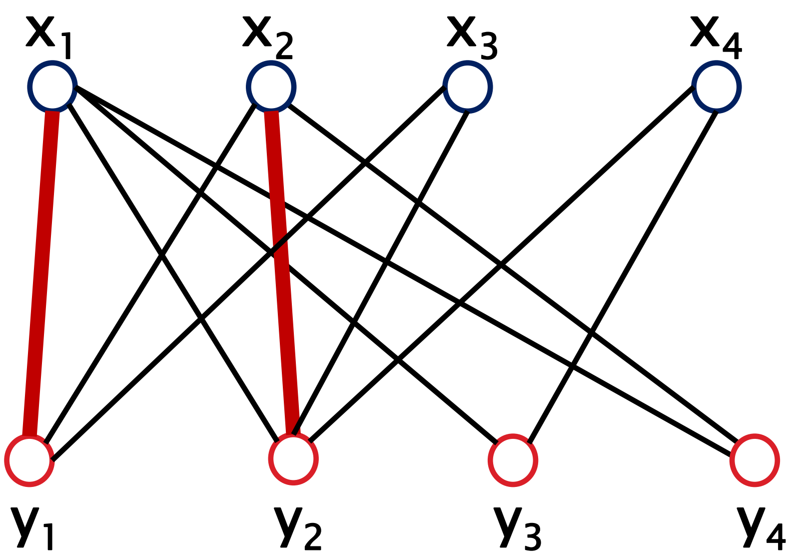
$P = [x_2, y_2]$



$M = \{x_1y_1, x_2y_2\}$

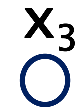
Etapa 3

$M = \{x_1y_1, x_2y_2\}$

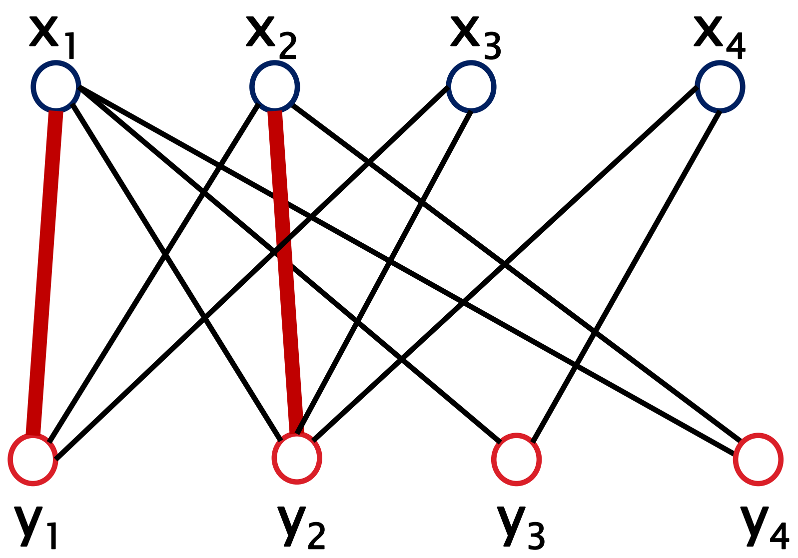


Etapa 3

$u = x_3$

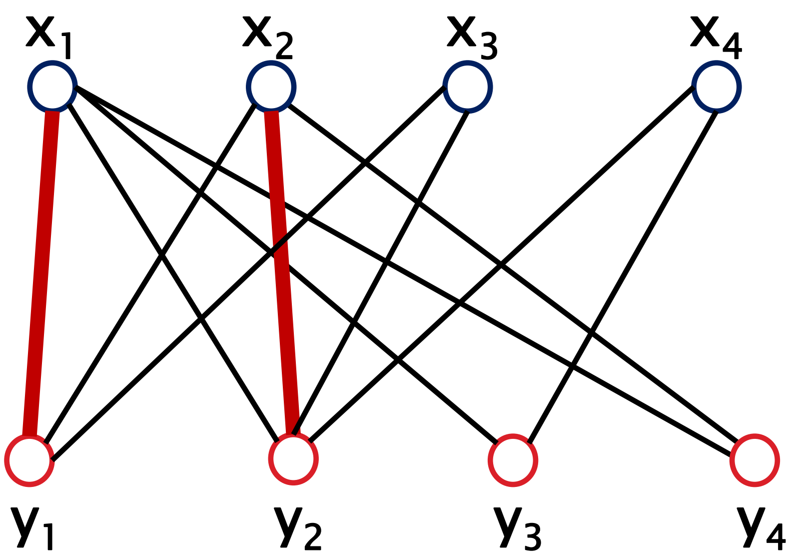


$M = \{x_1y_1,\ x_2y_2\}$

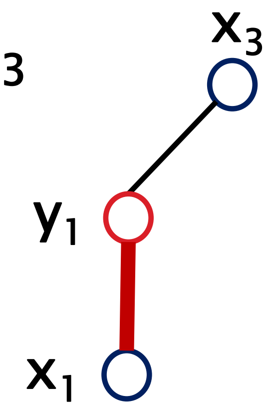


Etapa 3

$M = \{x_1y_1, x_2y_2\}$



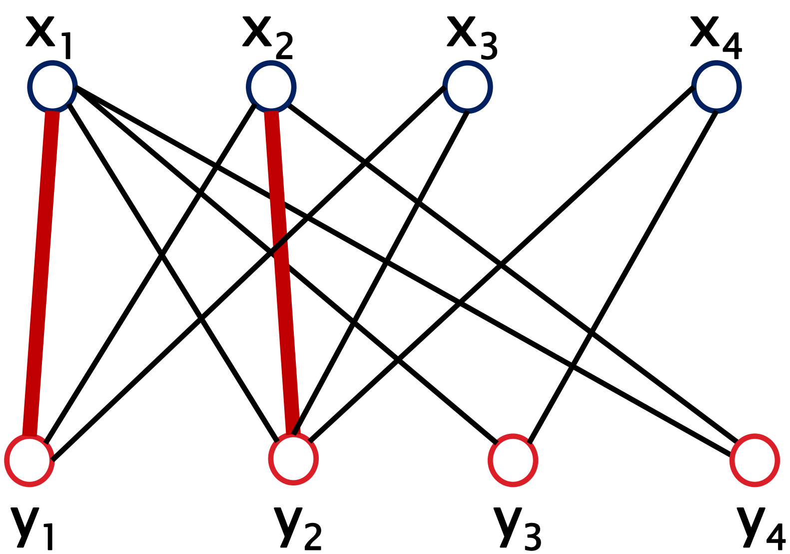
$u = x_3$

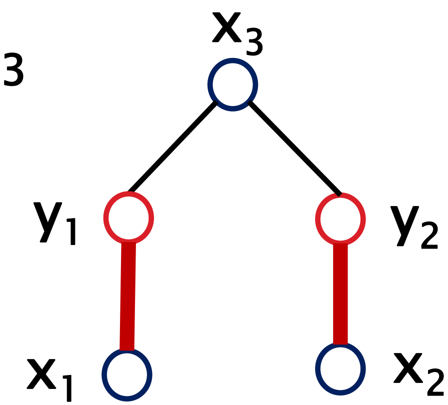


Etapa 3

$u = x_3$

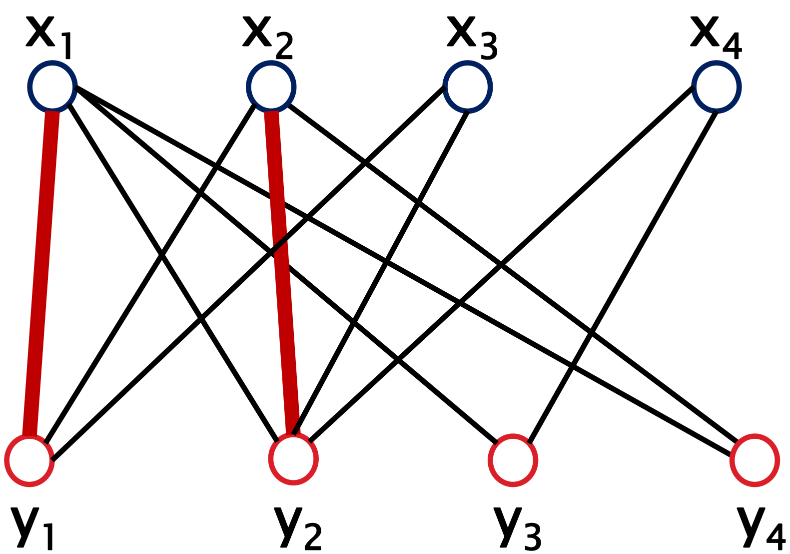
$M = \{x_1y_1,\ x_2y_2\}$



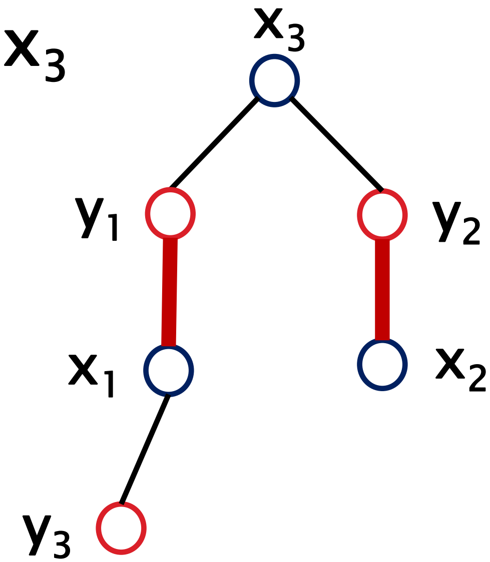


Etapa 3

$M = \{x_1y_1, x_2y_2\}$

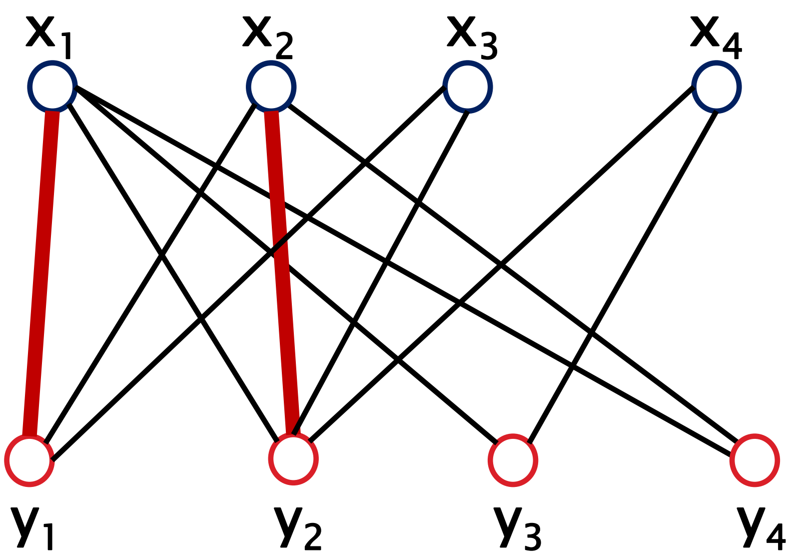


$u = x_3$

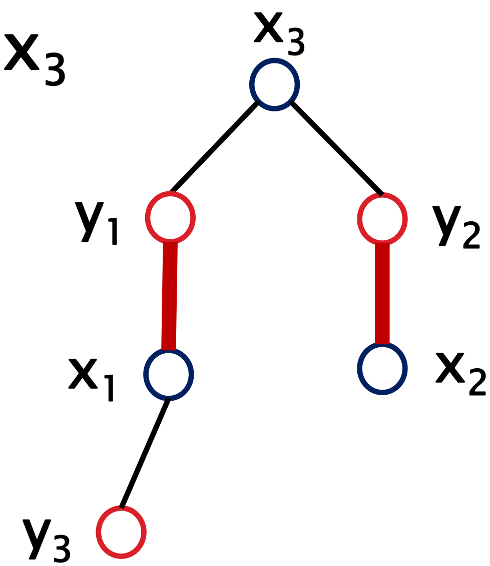


Etapa 3

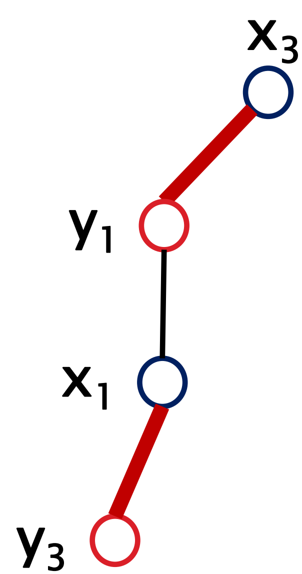
$M = \{x_1y_1, x_2y_2\}$



$u = x_3$



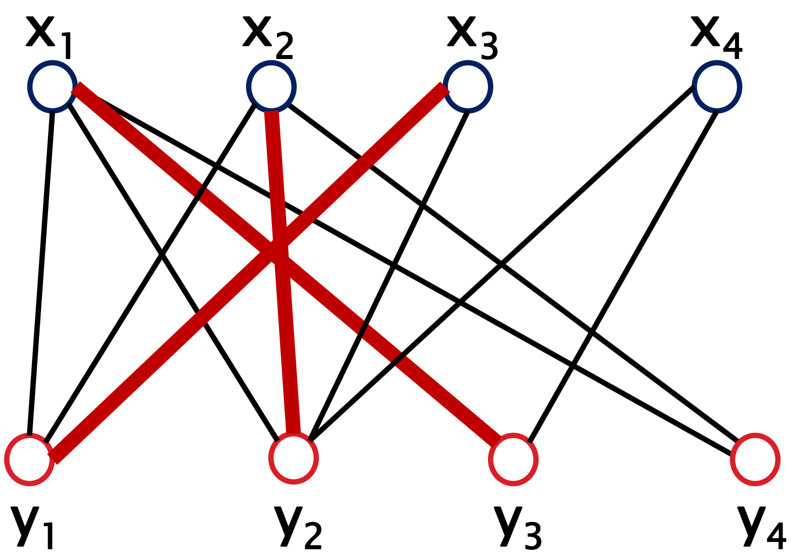
$P = [x_3, y_1, x_1, y_3]$



$M = \{x_3y_1, x_1y_3, x_2y_2\}$

Etapa 4

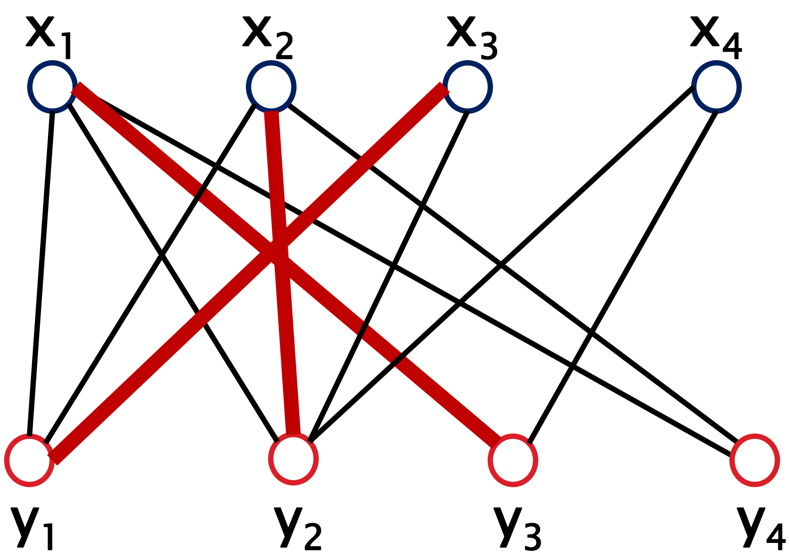
$M = \{x_3y_1, x_1y_3, x_2y_2\}$



Etapa 4

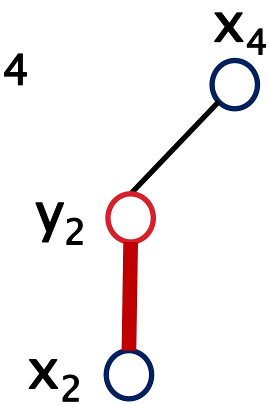
$u = x_4$

$M = \{x_3y_1, x_1y_3, x_2y_2\}$

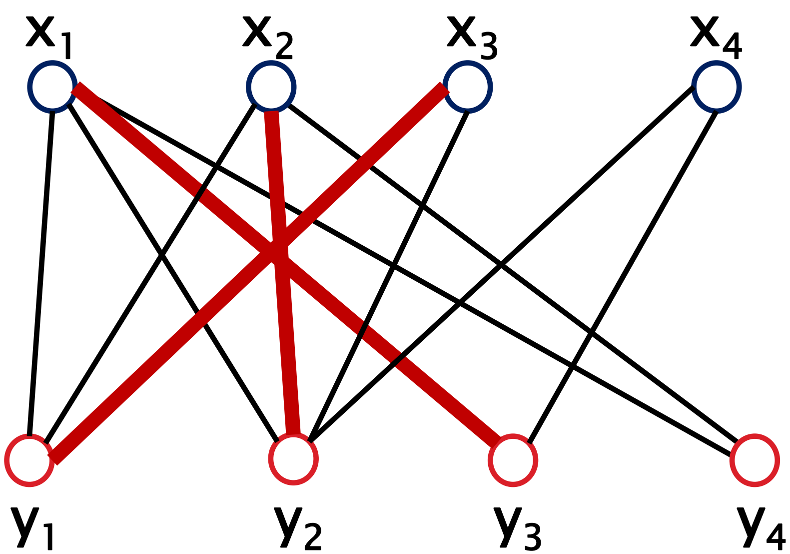


Etapa 4

$u = x_4$

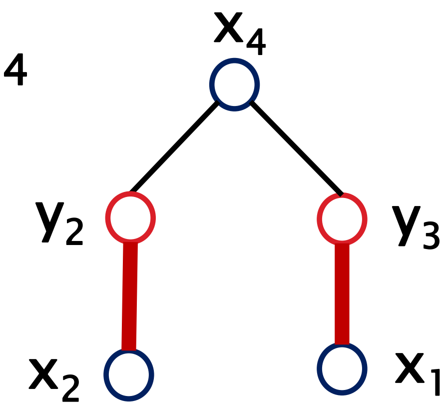


$M = \{x_3y_1,\ x_1y_3,\ x_2y_2\ \}$

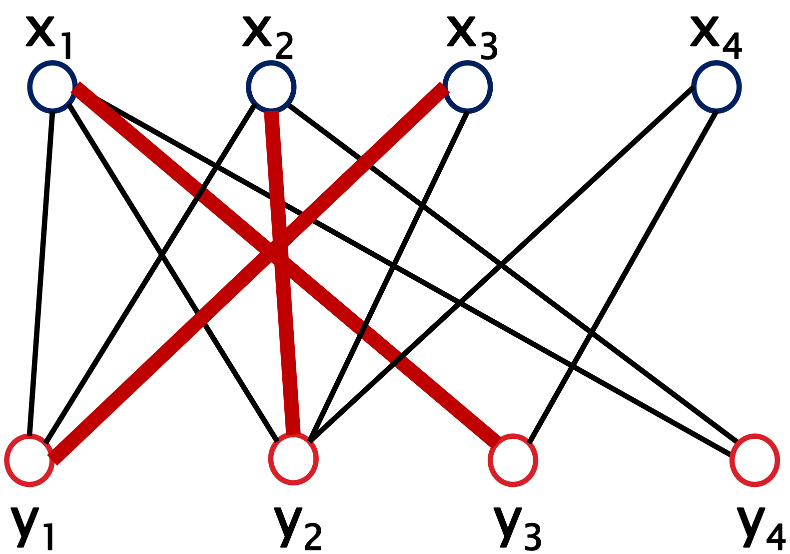


Etapa 4

$u = x_4$



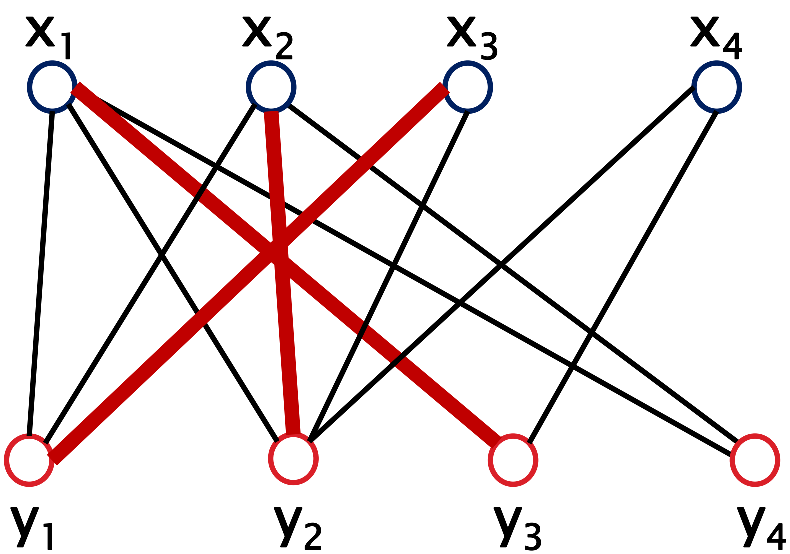
$M = \{x_3y_1,\ x_1y_3,\ x_2y_2\}$

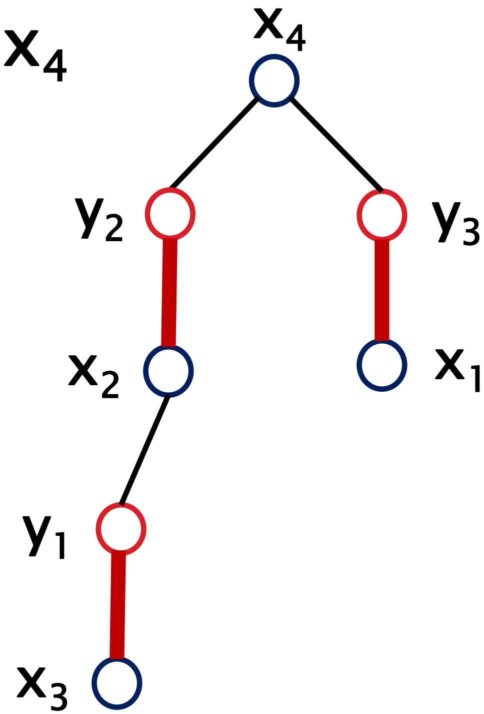


Etapa 4

$M = \{x_3y_1, x_1y_3, x_2y_2\}$

$u = x_4$

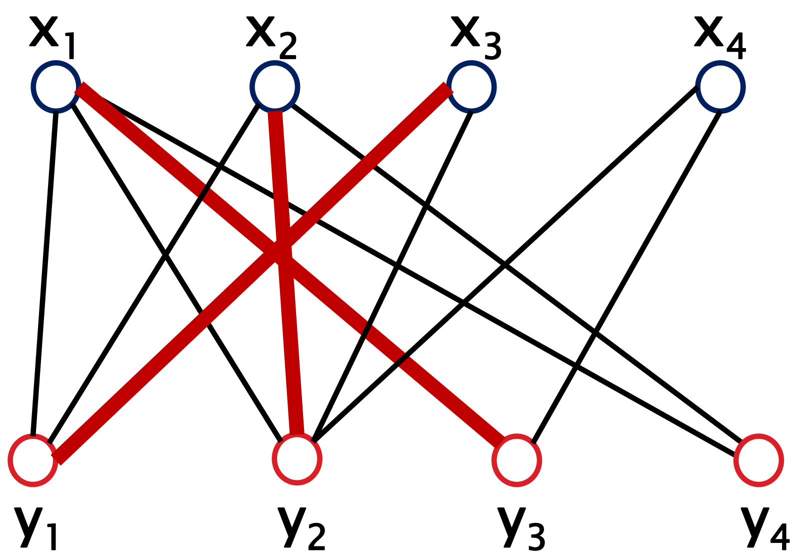


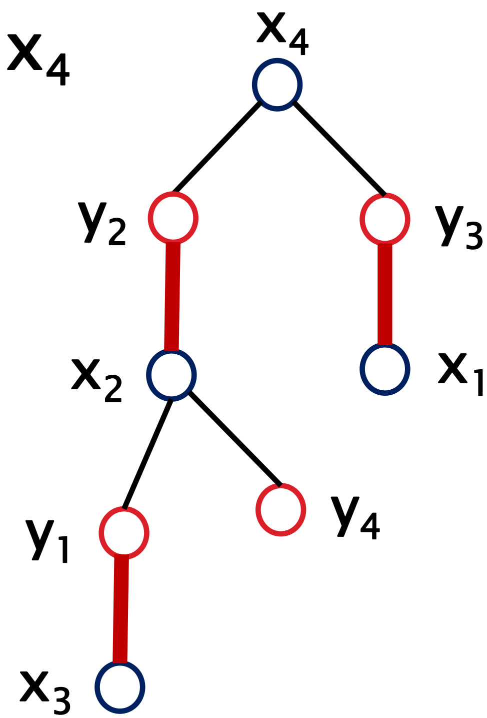


Etapa 4

$u = x_4$

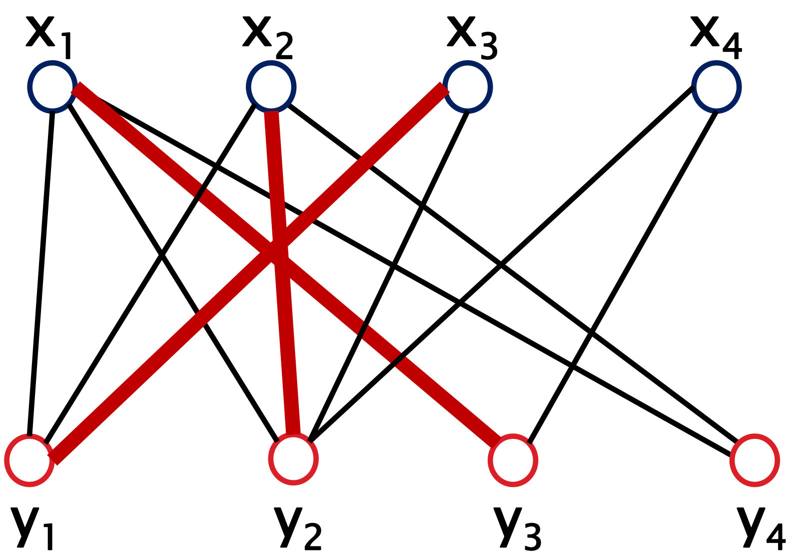
$M = \{x_3y_1, x_1y_3, x_2y_2\}$



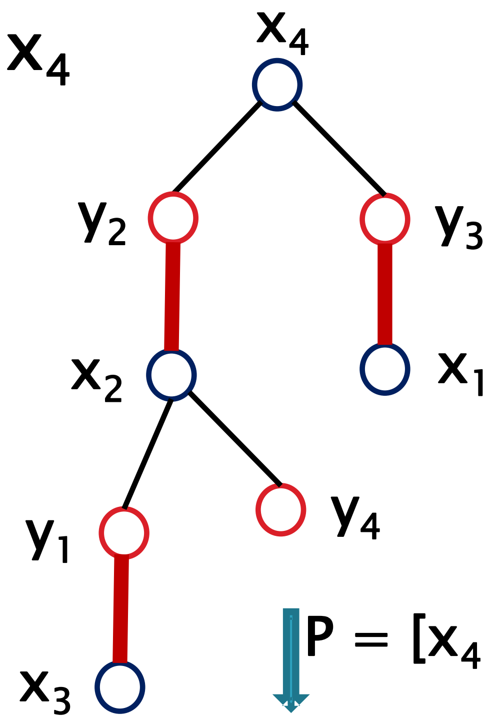


Etapa 4

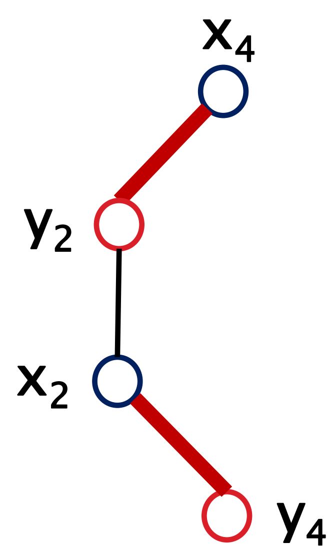
$$M = \{x_3y_1,\ x_1y_3,\ x_2y_2\}$$



$u = x_4$



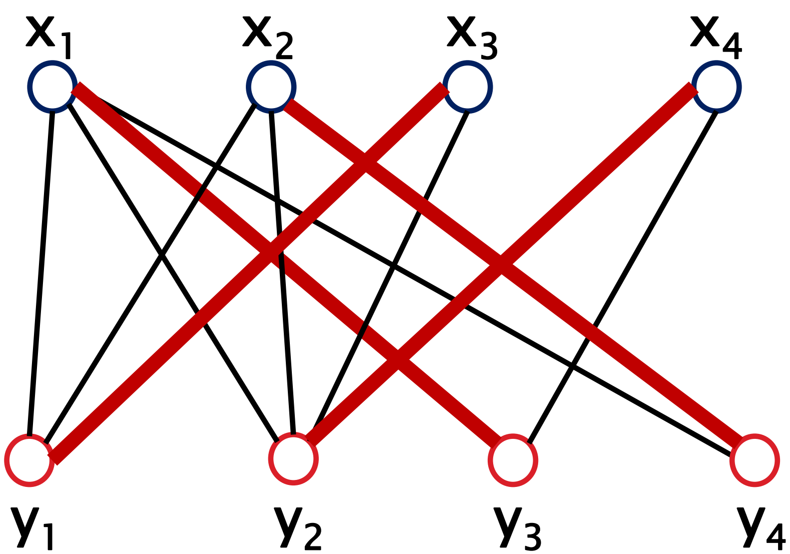
$P = [x_4, y_2, x_2, y_4]$



$$M = \{x_3y_1,\ x_1y_3,\ x_4y_2,\ x_2y_4\}$$

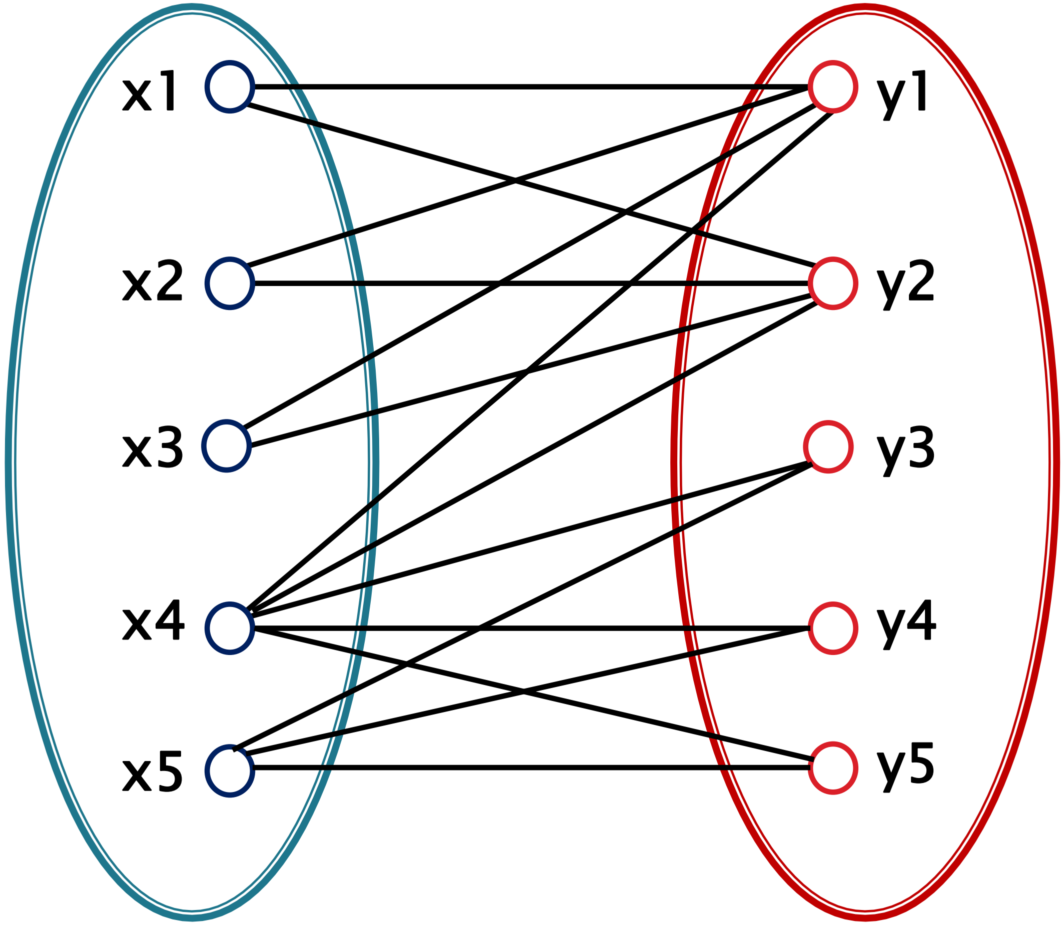
Etapa 4

$M = \{x_3y_1, x_1y_3, x_4y_2, x_2y_4\}$



Cuplaj al lui X în Y

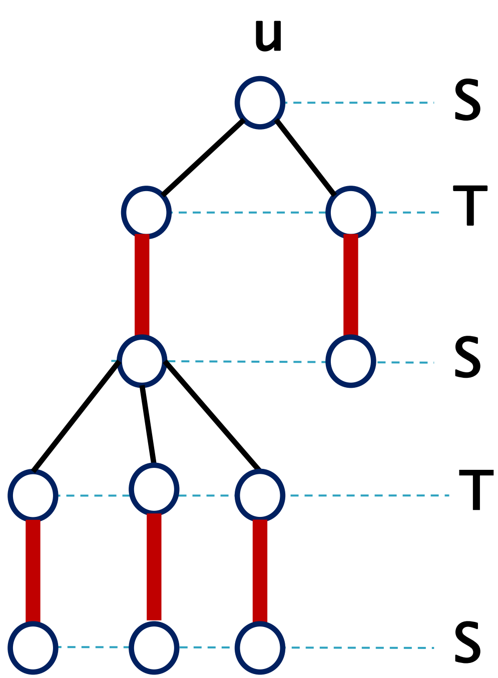
STOP



x1
x2
x3
x4
x5
y1
y2
y3
y4
y5

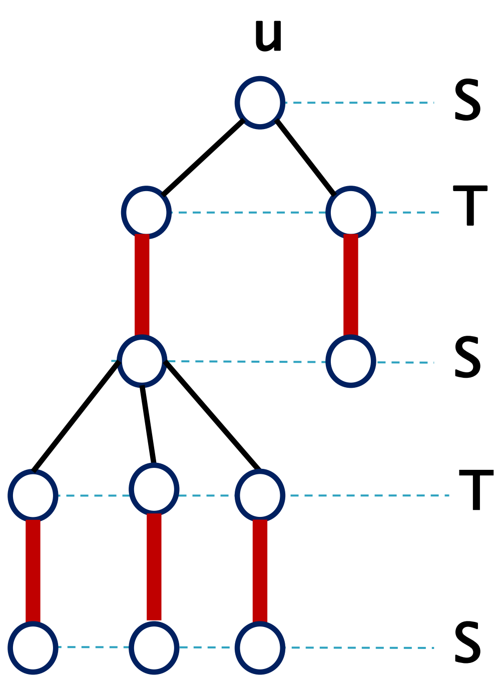
Corectitudine

Dacă arborele M-alternant de la pasul curent nu mai poate fi extins, atunci graful nu admite un cuplaj al lui X în Y.



Corectitudine

Dacă arborele M-alternant de la pasul curent nu mai poate fi extins, atunci graful nu admite un cuplaj al lui X în Y.



Demonstraţie

- $N(S) = T$
- $|N(S)| = |T| = |S|-1$
- Teorema lui Hall

Observație

Dacă arborele M-alternant de la pasul curent nu mai poate fi extins, algoritmul se încheie. **Cuplajul M determinat până la acest pas nu este însă neapărat un cuplaj de cardinal maxim în G**

Generalizare

Algoritmul ungar se poate modifica astfel încât să determine un **cuplaj de cardinal maxim** în G.

Generalizare

Algoritmul ungar se poate modifica astfel încât să determine un **cuplaj de cardinal maxim** în G.

- Astfel, dacă arborele M-alternant de la pasul curent nu mai poate fi extins, algoritmul nu se va opri, ci se reia construcția unui arbore M-alternant cu rădăcina în alt vârf nesaturat din X, **neconsiderat anterior**.

Algoritmul ungar – Generalizare

- **Idee**:
  1. Fie M un cuplaj arbitrar în G (eventual $\emptyset$)
  2. Cât timp M nu saturează toate vârfurile din X execută
     - Alege u un vârf M-nesaturat din X neconsiderat anterior
     - Construieşte un arbore M-alternant cu rădăcina în u până când
       a. un vârf nesaturat y este adăugat la arbore;
          fie P unicul u-y lanţ elementar din arbore
          $M \leftarrow M \,\Delta\, E(P)$
       b. Arborele nu mai poate fi extins. NU există cuplaj al lui X în Y, dar algoritmul continuă cu alegerea unui nou u

Aplicaţii

Aplicaţie: Sistem de reprezentanţi distincţi pentru submulţimi

Problemă – sistem de reprezentanţi distincţi

Fie A – mulţime finită

$B_1, B_2, \ldots, B_m \subseteq A$

S.n. **sistem de reprezentanţi distincţi** pentru colecţia de submulţimi $(B_1, B_2, \ldots, B_m)$ un vector $(r_1, r_2, \ldots, r_m)$ cu proprietățile

- $r_i \in B_i, \forall i=1,\ldots,m$
- $r_i \neq r_j, \forall i, j=1,\ldots,m, i \neq j$

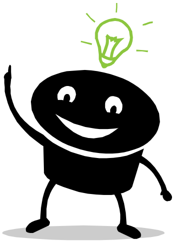
Problemă - sistem de reprezentanţi distincţi

Nu orice colecție de submulțimi admite un sistem de reprezentanţi distincţi.

Problemă - sistem de reprezentanţi distincţi

Condiții necesare și suficiente pentru existența unui sistem de reprezentanți distincți ai unei colecții de submulțimi din A

Problemă – sistem de reprezentanţi distincţi



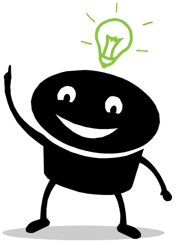
Modelăm problema cu ajutorul uni graf bipartit:

– asociem câte un vârf i fiecărei submulțimi $B_{i,}$ $i=1,\dots,m \Rightarrow$ mulțimea **X** de vârfuri

– asociem câte un vârf a_i fiecărui element din A,

$j = 1,\dots,n$, unde $n = |A| \Rightarrow$ mulțimea **Y** de vârfuri

Problemă – sistem de reprezentanţi distincţi



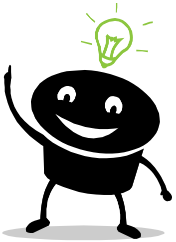
Modelăm problema cu ajutorul uni graf bipartit:

– asociem câte un vârf i fiecărei submulțimi $B_{i,}$ $i=1,\ldots,m \Rightarrow$ mulțimea **X** de vârfuri

– asociem câte un vârf a_j fiecărui element din A, $j = 1,\ldots,n$, unde $n = |A| \Rightarrow$ mulțimea **Y** de vârfuri

– construim muchie de la i la $a_j \Leftrightarrow$ elementul a_j se află în mulțimea B_i

Problemă – sistem de reprezentanţi distincţi



Exemplu:

$A = \{1, 2, 3, 4\}$

- $B_1 = \{2, 3\}$
- $B_2 = \{1, 3, 4\}$
- $B_3 = \{2, 4\}$

Observaţie

Există un sistem de reprezentanți pentru colecția de submulțimi (B_1, B_2, …, B_m) ale lui A $\Leftrightarrow$

există un cuplaj al lui X în Y în graful asociat

- Are loc astfel următorul rezultat

Teoremă –existenţa unui sistem de reprezentanţi distincţi

Fie A o mulțime finită și $(B_1, B_2, \ldots, B_m)$ o colecţie de submulţimi din A.

Colecția **nu** are un sistem de reprezentanți distincți $\Leftrightarrow$

$\exists$ k submulțimi în colecție a căror reuniune are mai puțin de k elemente

Aplicaţie:
Matrice de permutări

Problemă

Pe o tablă de tip șah de dimensiuni nxn sunt așezate ture, astfel încât pe fiecare linie și fiecare coloană sunt **aceleași număr de ture**. Să se arate că se pot păstra pe tablă n dintre aceste ture, care nu se atacă două câte două