



1

## Conexiune

- Teorema lui Menger

- $p$ -conexiune

- Teorema lui König

- Teorema lui Hall

- Teorema lui Dirac

2

## Arbori

- Elemente de bază

- Numărarea arborilor parțiali: Teorema lui Kirchhoff-Trent

3

## Exerciții pentru seminarul 6 (săptămâna 3 - 7 noiembrie)

4

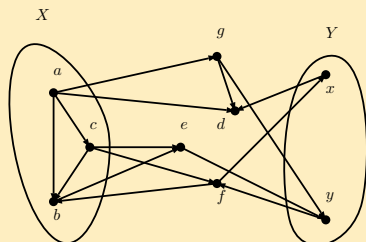
## Anexă - Generarea tuturor arborilor parțiali

## Definiția 1

Fie  $G = (V, E)$  un (di)graf și  $X, Y \subseteq V$ . Un  **$XY$ -drum** este orice drum  $P$  din  $G$  de la un nod  $x \in X$  la un nod  $y \in Y$  astfel încât  $V(P) \cap X = \{x\}$  și  $V(P) \cap Y = \{y\}$ .

Notăm cu  $\mathcal{P}(X, Y; G)$  familia tuturor  **$XY$ -drumurilor** din  $G$ . Observăm că dacă  $x \in X \cap Y$  atunci  $P = \{x\}$ , de lungime 0, este un  $XY$ -drum.

## Exemplu



$XY$ -paths:  $(b, e, y)$ ,  $(c, f, x)$ , and  $(a, g, y)$ ; an  $YX$ -path:  $(y, f, b)$

## Conexiune - Teorema lui Menger și aplicații

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

- Spunem că drumurile  $P_1$  și  $P_2$  sunt **disjuncte (pe noduri)** dacă  $V(P_1) \cap V(P_2) = \emptyset$ .
- Motivată de problemele practice din rețelele de comunicații și de asemeni de studiile teoretice asupra conexiunii în (di)grafuri, este de interes determinarea mulțimilor de cardinal maxim de  $XY$ -drumuri disjuncte.
- **Notăm cu  $p(X, Y; G)$  numărul maxim de  $XY$ -drumuri disjuncte în  $G$ .**
- Teorema care determină acest număr este datorată lui **Menger (1927)** și reprezintă unul dintre rezultatele fundamentale din Teoria grafurilor.

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

## Definiția 2

Fie  $G = (V, E)$  un (di)graf și  $X, Y \subseteq V$ . O mulțime  $XY$ -separatoare în  $G$  este orice submulțime  $Z \subseteq V$  astfel încât

$$V(P) \cap Z \neq \emptyset, \text{ pentru fiecare } P \in \mathcal{P}(X, Y; G).$$

Notăm cu

$$\mathcal{S}(X, Y; G) = \{Z : Z \text{ este mulțime } XY\text{-separatoare din } G\} \text{ și}$$

$$k(X, Y; G) = \min \{|Z| \mid Z \in \mathcal{S}(X, Y; G)\}$$

Din definiție urmează că:

- Dacă  $Z \in \mathcal{S}(X, Y; G)$ , atunci  $\mathcal{P}(X, Y; G \setminus Z) = \emptyset$ .
- $X, Y \in \mathcal{S}(X, Y; G)$ .



## Teorema 1

Teorema lui Menger. Fie  $G = (V, E)$  un (di)graf și  $X, Y \subseteq V$ . Atunci  $p(X, Y; G) = k(X, Y; G)$ .

(I. e., numărul maxim de  $XY$ -drumuri disjuncte = cardinalul minim al unei mulțimi  $XY$ -separatoare.)

Demonstrație:

$k(X, Y; G) \geq p(X, Y; G) = r$ . Fie  $P_1, \dots, P_r$   $XY$ -drumuri disjuncte din  $G$ ;  $Z \cap V(P_i) \neq \emptyset, \forall Z \in \mathcal{S}(X, Y; G)$ . Deoarece drumurile  $P_i$  sunt disjuncte pe noduri:

$$|Z| \geq \left| Z \cap \left( \bigcup_{i=1}^r V(P_i) \right) \right| = \sum_{i=1}^r |Z \cap V(P_i)| \geq \sum_{i=1}^r 1 = r.$$

Astfel,  $|Z| \geq r, \forall Z \in \mathcal{S}(X, Y; G)$ ; astfel  $k(X, Y; G) \geq r$ .

## Conexiune - Teorema lui Menger și aplicații

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

$k(X, Y; G) \leq p(X, Y; G)$ . Omisă. (Vom arăta mai târziu că  $\forall G = (V, E)$  și  $\forall X, Y \subseteq V$ ,  $\exists k(X, Y; G)$   $XY$ -drumuri disjuncte în  $G$  folosind fluxuri în anumite rețele.)  $\square$

Menger (1927) a enunțat echivalent teorema de mai sus, utilizând drumuri intern-disjuncte:  $P_1, P_2 \in \mathcal{P}_{st}$  astfel încât  $V(P_1) \cap V(P_2) = \{s, t\}$ :

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

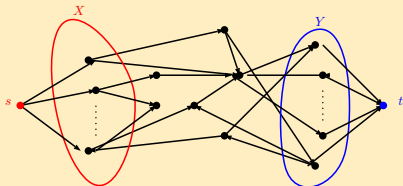
### Teorema 2

Fie  $G = (V, E)$  un (di)graf și  $s, t \in V$ , astfel încât  $s \neq t$ ,  $st \notin E$ . Există  $k$  drumuri intern-disjuncte de la  $s$  la  $t$  în  $G$  dacă și numai dacă există cel puțin un drum de la  $s$  la  $t$  în (di)graful obținut din  $G$  prin ștergerea oricărei mulțime de  $< k$  noduri diferite de  $s$  și  $t$ .

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

## Demonstrația echivalenței:

**Teorema 1  $\Rightarrow$  Teorema 2:** luăm  $X = N_G^+(s)$  ( $N_G(s)$ ) și  $Y = N_G^-(t)$  ( $N_G(t)$ ).  $p(X, Y; G)$  este numărul maxim de drumuri intern-disjuncte de la  $s$  la  $t$  în  $G$ .  $k(X, Y; G)$  este cardinalul minim al unei mulțimi de noduri prin a căror ștergere  $s$  și  $t$  se deconectează.



**Teorema 2  $\Rightarrow$  Teorema 1:** adăugăm două noi noduri  $s$  și  $t$  (di)grafului  $G$ , și toate muchiile (orientate) de la  $s$  la orice nod din  $X$  și de la orice nod din  $Y$  la  $t$ . Numărul maxim de drumuri intern-disjuncte de la  $s$  la  $t$  în  $G$  este  $p(X, Y; G)$ ; cardinalul minim al unei mulțimi de noduri prin a căror ștergere  $s$  și  $t$  se deconectează este  $k(X, Y; G)$ .  $\square$

## Aplicații: $p$ -conexiune

- Un graf  $G$  este  $p$ -conex ( $p \in \mathbb{N}^*$ ) dacă fie  $G = K_p$ , fie  $|G| > p$  și  $G \setminus A$  este conex pentru orice  $A \subseteq V(G)$  cu  $|A| < p$ .
- Din Teorema 2, o caracterizare echivalentă a  $p$ -conexiunii este:  
Un graf  $G$  este  $p$ -conex ( $p \in \mathbb{N}^*$ ) dacă fie  $G = K_p$ , fie  $\forall st \in E(\overline{G})$  există  $p$  drumuri intern-disjuncte de la  $s$  la  $t$  în  $G$ .
- numărul de conexiune pe noduri al grafului  $G$ ,  $k(G)$ , este cel mai mare  $p$  pentru care  $G$  este  $p$ -conex.
- Urmează că, pentru a calcula  $k(G)$ , trebuie aflat

$$\min_{st \notin E(G)} p(\{s\}, \{t\}; G)$$

care poate fi determinat în timp polinomial folosind fluxuri în rețele.

### Aplicații: Teorema lui König

- O **acoperire cu noduri (vertex cover)** a grafului  $G$  este o mulțime  $Z \subseteq V(G)$  de noduri astfel încât  $G - Z$  este un graf nul (orice muchie din  $G$  are cel puțin o extremitate în  $Z$ ).
- Un caz special al Teoremei 1 se obține când  $G$  este bipartit iar  $X = S$  și  $Y = T$  sunt cele două clase ale bipartiției lui  $G$ :

### Teorema 3

(König, 1931) Fie  $G = (S, T; E)$  un graf bipartit. Atunci, cardinalul maxim al unui cuplaj din  $G$  este egal cu cardinalul minim al unei acoperiri cu noduri a lui  $G$ .

**Demonstrație:** Cardinalul maxim al unui cuplaj în  $G$  este  $p(S, T; G) = k(S, T; G)$ , din Teorema 1. Deoarece o mulțime de noduri este o mulțime  $ST$ -separatoare dacă și numai dacă este o acoperire cu noduri, Teorema 3 este dovedită.  $\square$

## Aplicații: Teorema lui Hall

- Fie  $I$  și  $S$  mulțimi finite nevide. O familie submulțimi ale lui  $S$  (indexată după  $I$ ) este o funcție  $A : I \rightarrow 2^S$ . Notăm  $\mathcal{A} = (A_i)_{i \in I}$  și (folosind notația funcțională)  $\mathcal{A}(J) = \bigcup_{j \in J} A_j$  (pentru  $J \subseteq I$ ).
- O funcție de reprezentare pentru familia  $\mathcal{A} = (A_i)_{i \in I}$  este orice funcție  $r_{\mathcal{A}} : I \rightarrow S$  cu proprietatea  $r_{\mathcal{A}}(i) \in A_i, \forall i \in I$ ; atunci,  $(r_{\mathcal{A}}(i))_{i \in I}$  este numit un sistem de reprezentanți pentru  $\mathcal{A}$ .
- Dacă funcția de reprezentare,  $r_{\mathcal{A}}$ , este injectivă, atunci  $r_{\mathcal{A}}(I)$  este o submulțime a lui  $S$  și este numită sistem de reprezentanți distincți pentru  $\mathcal{A}$ , sau o transversală a lui  $\mathcal{A}$ .
- Problema centrală în Teoria Transversalelor este de a caracteriza familiile care admit o transversală (cu anumite proprietăți). Teorema lui Hall (1935) este primul rezultat de acest tip.

## Teorema 4

(Hall, 1935) Familia  $\mathcal{A} = (A_i)_{i \in I}$  de submulțimi ale lui  $S$  are o transversală dacă și numai dacă

$$(H) \quad |\mathcal{A}(J)| \geq |J|, \forall J \subseteq I.$$

**Demonstrație:** " $\Rightarrow$ " Dacă  $r_{\mathcal{A}}$  este o funcție de reprezentare injectivă pentru  $\mathcal{A}$ , atunci  $r_{\mathcal{A}}(J) \subseteq \mathcal{A}(J), \forall J \subseteq I$ . Astfel,  $r_{\mathcal{A}}$  fiind injectivă,  $|\mathcal{A}(J)| \geq |r_{\mathcal{A}}(J)| = |J|$ .

" $\Leftarrow$ " Fie  $G_{\mathcal{A}} = (I, S; E)$  graful bipartit asociat familiei  $\mathcal{A}$  (dacă  $I \cap S \neq \emptyset$ , putem considera copii izomorfe disjuncte), unde  $E = \{is | i \in I, s \in A_i\}$ . Se observă că  $N_{G_{\mathcal{A}}}(i) = A_i$ . Mai mult,  $\mathcal{A}$  are o transversală dacă și numai dacă  $G_{\mathcal{A}}$  are un cuplaj de cardinal  $|I|$ .

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

**Demonstrația Teoremei lui Hall (continuare):** Arătăm că dacă relația (H) are loc, atunci orice acoperire cu noduri a lui  $G_A$  are cel puțin  $|I|$  noduri, și - din Teorema lui König -  $G_A$  are un cuplaj de cardinal  $|I|$  ( $I$  este o acoperire cu noduri a lui  $G_A$ ).

Fie  $X = I' \cup S' \subseteq I \cup S$  o acoperire cu noduri a lui  $G_A$ : urmează că  $N_{G_A}(I \setminus I') \subseteq S'$ , adică,  $\mathcal{A}(I \setminus I') \subseteq S'$ . Atunci,

$$|X| = |I'| + |S'| \geq |I'| + |\mathcal{A}(I \setminus I')|.$$

Deoarece are loc (H), obținem

$$|X| \geq |I'| + |\mathcal{A}(I \setminus I')| \geq |I'| + |I \setminus I'| = |I|. \quad \square$$

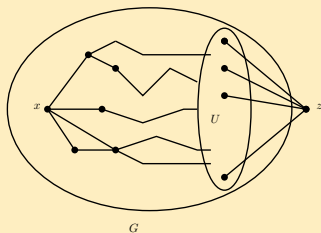
Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru  
- Graph Algorithms \* C. Croitoru - Graph Algorithms \* C. Croitoru - Graph Algorithms \*

## Aplicații: Teorema lui Dirac (structura grafurilor $p$ -conexe)

### Lema 1

Fie  $G = (V, E)$  un graf  $p$ -conex de ordin  $|G| \geq p + 1$ ,  $U \subseteq V$ ,  $|U| = p$  și  $x \in V \setminus U$ . Atunci există  $p$   $xU$ -drumuri astfel încât oricare două dintre ele îl au numai pe  $x$  drept nod comun.

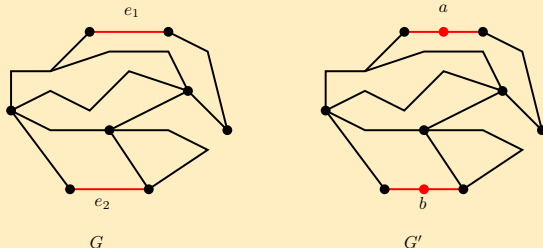
**Demonstrație:** Fie  $G' = (V \cup \{z\}, E')$ , unde  $E' = E \cup \{zu : u \in U\}$ .





**Demonstrație:** Inducție după  $p$ .

Pentru  $p = 2$ , trebuie să arătăm că într-un graf 2-conex,  $G$ , orice două muchii  $e_1$  și  $e_2$  aparțin unui circuit. Fie  $G'$  graful obținut din  $G$  prin inserarea unui nod  $a$  pe  $e_1$  și a unui nod  $b$  pe  $e_2$ :



$G'$  este 2-conex (orice graf de tipul  $G' - v$  este conex). Astfel, există două drumuri intern-disjuncte de la  $a$  la  $b$ , care dau circuitul din  $G$  conținând  $e_1$  și  $e_2$ .

**Demonstrație (continuare).** În pasul inductiv, fie  $p \geq 3$ , presupunem că Propoziția este adevărată pentru orice graf  $p'$ -conex cu  $2 \leq p' < p$ , și considerăm un graf  $p$ -conex  $G$ , două dintre muchiile sale  $e_1$  și  $e_2$  și o mulțime de  $p - 2$  noduri  $\{x_1, x_2, \dots, x_{p-2}\}$ .

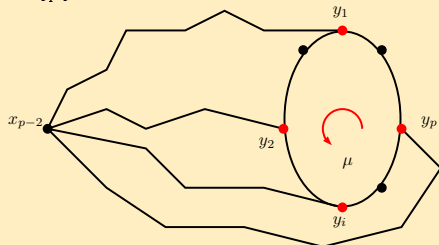
Putem presupune că nicio extremitate  $v$  a lui  $e_1$  sau  $e_2$  nu aparține mulțimii  $\{x_1, x_2, \dots, x_{p-2}\}$  (altfel, aplicăm ipoteza inductivă și obținem că în graful  $(p - 1)$ -conex,  $G$ , există un circuit  $C$  conținând  $e_1, e_2$  și mulțimea de noduri  $\{x_1, x_2, \dots, x_{p-2}\} \setminus \{v\}$ ; iar  $v$  este un nod al lui  $C$  deoarece  $e_1$  și  $e_2$  sunt muchii ale lui  $C$ ).

Graful  $G - x_{p-2}$  este  $(p - 1)$ -conex. Din ipoteza inductivă, există un circuit  $\mu$  care conține  $x_1, x_2, \dots, x_{p-3}, e_1$  și  $e_2$ . Fie  $Y$  mulțimea nodurilor lui  $\mu$ . Evident,  $|Y| \geq p$  (mulțimii de  $p - 3$  noduri  $x_1, x_2, \dots, x_{p-3}$ , îi adăugăm cel puțin trei extremități ale muchiilor  $e_1$  și  $e_2$ ). Din Lema 1, există  $p$   $x_{p-2}$   $Y$ -drumuri astfel încât oricare două dintre ele au în comun doar un singur nod,  $x_{p-2}$ .

## Conexiune - Teorema lui Menger și aplicații

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

**Demonstrație (continuare).** Fie  $P_{x_{p-2}y_1}, P_{x_{p-2}y_2}, \dots, P_{x_{p-2}y_p}$  aceste drumuri, unde ordinea  $y_1, \dots, y_p$  se obține în urma unei parcurgeri a lui  $\mu$ . Nodurile  $y_1, \dots, y_p$  împart circuitul  $\mu$  în drumurile  $P_{y_1y_2}, P_{y_2y_3}, \dots, P_{y_{p-1}y_p}, P_{y_p y_1}$ :



Cel puțin unul dintre drumurile de mai sus nu conține niciun element din mulțimea  $x_1, x_2, \dots, x_{p-3}, e_1$  și  $e_2$  (pigeon hole principle).

Fie  $P_{y_1y_2}$  acest drum (altfel, renumerotăm nodurile  $y_i$ ).



## Conexiune - Teorema lui Menger și aplicații

O aplicație interesantă a Teoremei 5 și Propoziției 1 este următoarea condiție suficientă pentru ca un graf să fie Hamiltonian dată de Erdős și Chvatal.

### Teorema 6

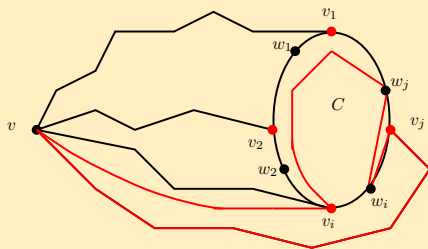
(Erdős-Chvatal, 1972) Fie  $G = (V, E)$  un graf  $p$ -conex. Dacă  $\alpha(G) \leq p$  atunci  $G$  este graf Hamiltonian.

**Demonstrație:** Să presupunem, prin contradicție că  $G$  nu este Hamiltonian. Fie  $C$  un cel mai lung circuit din  $G$ .

Din Teorema lui Dirac  $|C| \geq p$  și din presupunerea noastră, există un nod  $v \in V(G) \setminus V(C) \neq \emptyset$ .

**Demonstrație (continuare).** Cum  $|C| \geq p$ , putem repeta argumentul din demonstrația Propoziției de mai sus pentru a arăta că există  $P_{vv_1}, P_{vv_2}, \dots, P_{vv_p}$ ,  $p$   $vC$ -drumuri care se intersectează două câte două doar în  $v$  și cu extremități  $v_i$  etichetate în ordinea în care apar la o parcurgere a circuitului.

Fie  $w_i$  succesorul nodului  $v_i$  pe circuit.





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

Un **arbore** este un graf conex fără circuite.

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

## Teorema 7

Fie  $G = (V, E)$  un graf. Atunci următoarele afirmații sunt echivalente:

- (i)  $G$  este un arbore (este conex și nu are circuite).
- (ii)  $G$  este conex și este minimal cu această proprietate.
- (iii)  $G$  nu are circuite și este maximal cu această proprietate.

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

**Demonstrație:** Omisă.  $\square$

Minimalitatea și maximalitatea din afirmațiile de mai sus sunt relativ la relația de ordine parțială dată de incluziune pe submulțimile muchii. Mai precis afirmațiile (ii) și (iii) înseamnă:

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

- (ii)  $G$  este **conex** și  $\forall e \in E$ ,  $G - e$  nu este conex.
- (iii)  $G$  **nu are circuite** și  $\forall e \notin E$ ,  $G + e$  are un circuit.

## Definiție

Fie  $G = (V, E)$  un (multi)graf. Un **arbore parțial**  $G$  este un graf parțial al lui  $G$ ,  $T = (V, E')$  ( $E' \subseteq E$ ), care este arbore. Notăm cu  $\mathcal{T}_G$  mulțimea tuturor arborilor parțiali ai lui  $G$ .

## Remarci

1.  $\mathcal{T}_G \neq \emptyset$  dacă și numai dacă  $G$  este conex. Într-adevăr, dacă  $\mathcal{T}_G \neq \emptyset$ , atunci există un arbore parțial  $T = (V, E')$  al lui  $G$ .  $T$  este conex, deci între orice două noduri ale lui  $G$  există este un drum  $P$  în  $T$ . Deoarece  $E' \subseteq E$ ,  $P$  este un drum și în  $G$ , deci  $G$  este conex.

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

Reciproc, dacă  $G$  este conex, atunci considerăm următorul algoritm:

```
 $T \leftarrow G;$   
while ( $\exists e \in E(T)$  astfel încât  $T - e$  este conex) do  
     $T \leftarrow T - e;$ 
```

Din construcție,  $T$  este graf parțial al lui  $G$ , și are loc afirmația (ii) din Teorema 7, deci  $T$  este un arbore.

2. O altă demonstrație constructivă (dacă  $G$  este conex atunci  $\mathcal{T}_G \neq \emptyset$ ) se bazează pe observația că **există o muchie în cross între cele două clase ale oricărei bipartiții a lui  $V$ :  $\exists e = v_1 v_2 \in E$  cu  $v_i \in V_i, i = \overline{1, 2}$ .**

Dacă  $|V| = n > 0$  atunci următorul algoritm construiește un arbore parțial al grafului conex  $G = (V, E)$ :

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

```

 $k \leftarrow 1; T_1 \leftarrow (\{v\}, \emptyset); // v \in V$ 
while ( $k < n$ ) do
  fie  $xy \in E$  cu  $x \in V(T_k), y \in V \setminus V(T_k)$ ;
  // o astfel de muchie există din conexiunea lui  $G$ 
   $V(T_{k+1}) \leftarrow V(T_k) \cup \{y\}$ ;
   $E(T_{k+1}) \leftarrow E(T_k) \cup \{xy\}$ ;
   $k ++$ ;

```

Evident,  $T_k$  este un arbore  $\forall k = \overline{1, n}$  (inductiv, dacă  $T_k$  este un arbore atunci, din construcție,  $T_{k+1}$  este conex și nu are circuite). Mai mult, avem  $|V(T_k)| = k$  și  $|E(T_k)| = k - 1, \forall k = \overline{1, n}$ .

3. Dacă această construcție este aplicată unui arbore  $G$  cu  $n$  noduri, vom obține că  $G$  are  $n - 1$  muchii. Această proprietate poate fi folosită pentru a extinde Teorema 7 cu alte caracterizări ale arborilor:



Fie  $G = (V, E)$  un multi-graf cu  $V = \{1, 2, \dots, n\}$ , și matricea de adiacență  $A = (a_{ij})_{n \times n}$  ( $a_{ij}$  = multiplicitatea muchiei  $ij$  dacă  $ij \in E$ , 0 altfel). Fie

$$D = \text{diag}(d_G(1), d_G(2), \dots, d_G(n)) = \begin{pmatrix} d_G(1) & 0 & \dots & 0 \\ 0 & d_G(2) & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & d_G(n) \end{pmatrix}.$$

Matricea Laplaciană a lui  $G$  (sau Laplacianul) este definită ca fiind:

$$L[G] = D - A.$$

Observăm că suma tuturor elementelor din fiecare linie sau din fiecare coloană a lui  $L[G]$  este 0. Notăm cu  $L[G]_{ij}$  minorul matricii  $L[G]$  obținut prin ștergerea liniei  $i$  și a coloanei  $j$ .

## Teorema 9

(Kirchhoff-Trent). Fie  $G$  un (multi)graf cu mulțimea nodurilor  $\{1, \dots, n\}$  și Laplacianul  $L[G]$ . Atunci, numărul arborilor parțiali ai lui  $G$  este:  $|\mathcal{T}_G| = \det(L[G]_{ii}), \forall 1 \leq i \leq n$ .

Demonstrație: Omisă.  $\square$

## Corolar

(Formula lui Cayley).  $|\mathcal{T}_{K_n}| = n^{n-2}$ .

Demonstrație:

$$L[K_n] = \begin{pmatrix} n-1 & -1 & \dots & -1 \\ -1 & n-1 & \dots & -1 \\ \vdots & \vdots & \ddots & \vdots \\ -1 & -1 & \dots & n-1 \end{pmatrix}.$$

## Arbori - Numărarea arborilor parțiali

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

Astfel:

$$\det(L[K_n]_{11}) = \begin{vmatrix} n-1 & -1 & \dots & -1 \\ -1 & n-1 & \dots & -1 \\ \vdots & \vdots & \ddots & \vdots \\ -1 & -1 & \dots & n-1 \end{vmatrix}$$

Dacă adunăm toate liniile la prima obținem

$$\begin{vmatrix} 1 & 1 & \dots & 1 \\ -1 & n-1 & \dots & -1 \\ \vdots & \vdots & \ddots & \vdots \\ -1 & -1 & \dots & n-1 \end{vmatrix} = n^{n-2}. \text{ (De ce?)}$$

□

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

**Exercițiul 1.** Fie  $G = (V, E)$  un graf conex și  $v \in V$  astfel încât  $N_G(v) \neq V \setminus \{v\}$ . Pentru  $X \subseteq V$  notăm  $N_G(X) = \left( \bigcup_{v \in X} N_G(v) \right) \setminus X$ .

Evident, mulțimea  $A = \{v\}$  satisface următoarele proprietăți:

- (i)  $v \in A$  și  $[A]_G$  este conex.
- (ii)  $N = N_G(A) \neq \emptyset$ .
- (iii)  $R = V \setminus (A \cup N) \neq \emptyset$ .
- (a) Arătați că, dacă  $A \subseteq V$  este orice mulțime noduri care satisface (i) - (iii) și maximală (relativ la " $\subseteq$ ") cu aceste proprietăți, atunci  $\forall x \in R$  și  $\forall y \in N$  avem  $xy \in E$ .
- (b) Dovediți că dacă  $A$  este ca la (a) și  $G$  este  $\{C_k\}_{k \geq 4}$ -free, atunci  $N$  este o clică în  $G$ .
- (c) Deduceți că  $K_n$  ( $n \in \mathbb{N}^*$ ) sunt singurele grafuri regulate, triangulate și conexe.

## Exerciții pentru seminarul 6

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

**Exercițiul 2.** Un graf de ordin cel puțin trei este numit **confidențial conex** dacă, pentru orice trei noduri distincte  $a, b$  și  $c$ , există un drum de la  $a$  la  $b$  astfel încât  $c$  este diferit de și nu este adiacent cu niciun nod intern (dacă există) al acestui drum. (Un exemplu de graf confidențial conex este graful complet  $K_n$ , cu  $n \geq 3$ .)

Arătați că un graf conex, necomplet,  $G = (V, E)$ , cu cel puțin trei noduri este confidențial conex dacă și numai dacă:

- (i) pentru orice  $v \in V$ ,  $N_G(v) \neq \emptyset$  și induce un subgraf conex;
- (ii) orice muchie a lui  $G$  face parte dintr-un  $C_4$  indus sau este muchia mediană a unui  $P_4$  indus.

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

**Exercițiul 3.** Dovediți că un graf conex,  $p$ -regulat și bipartit este 2-conex.

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

**Exercițiul 4.** Fie  $G = (V, E)$  un digraf. Demonstrați că:

- $G$  este tare conex dacă și numai dacă pentru orice  $S \subsetneq V$ ,  $S \neq \emptyset$ , există măcar un arc care pleacă din  $S$ .
- Dacă  $G$  este tare conex și poate fi deconectat prin ștergerea a cel mult  $p$  arce (i. e.,  $\exists A \subseteq E$ ,  $|A| \leq p$  astfel încât  $G - A$  nu este tare conex), atunci  $G$  poate fi deconectat prin inversarea a cel mult  $p$  arce (adică  $\exists B \subseteq E$ ,  $|B| \leq p$  astfel încât  $G' = (V, (E \setminus B) \cup \{uv : vu \in B\})$  nu este tare conex).

**Exercițiul 5.** Fie  $G$  un graf 2-muchie-conex ( $G - e$  este conex,  $\forall e \in E(G)$ ). Definim următoarea relație binară  $e \asymp f$  dacă  $e = f$  or  $G - \{e, f\}$  nu este conex.

- Arătați că  $e \asymp f$  dacă și numai dacă  $e$  și  $f$  aparțin aceluiași circuit.
- Arătați că o clasă de echivalență  $[e]_{\asymp}$  este inclusă într-un circuit.
- Ștergând toate muchiile dintr-o clasă de echivalență  $[e]_{\asymp}$ , componentele conexe ale grafului rămas sunt grafuri 2-muchie-conexe.

### Exercițiul 6.

- (a) Fie  $G$  un graf cu cel puțin 3 noduri. Dacă  $G$  este 2-conex, atunci putem să-i orientăm muchiile așa încât graful orientat rezultat să fie tare conex.
- (b) Reciproca afirmației de mai sus este adevărată?

### Exercițiul 7.

- (a) Fie  $G$  un graf 2-conex, necomplet și  $xy \in E(G)$ . Arătați că  $G - xy$  sau  $G|xy$  este 2-conex.
- (b) Dați câte un exemplu de un graf  $G$  și o muchie  $xy \in E(G)$  astfel ca: (b1)  $G - xy$  și  $G|xy$  sunt 2-conexe; (b2)  $G - xy$  nu este 2-conex dar  $G|xy$  este 2-conex; (b3)  $G - xy$  este 2-conex dar  $G|xy$  nu este 2-conex;

**Exercițiul 8.** Pentru un graf conex  $G$  dat aplicăm următorul algoritm:

```

 $\mathcal{Q} \leftarrow \{G\};$  //  $\mathcal{Q}$  este o coadă;
while ( $\mathcal{Q} \neq \emptyset$ ) {
     $H \leftarrow \text{pop}(\mathcal{Q});$ 
    fie  $A \subseteq V(H)$  o mulțime de articulație minimală din  $H$ ;
    fie  $G_1, \dots, G_k$  componentele conexe ale lui  $H - A$ ;
    pentru ( $j = 1$  to  $k$ )
        push( $\mathcal{Q}, [A \cup V(G_j)]_G$ );
    }

```

Observăm că dacă  $G$  este a graf complet, atunci în  $\mathcal{Q}$  nu se mai adaugă vreun alt graf.

- Arătați că orice graf adăugat în  $\mathcal{Q}$  este conex.
- Dovediți că numărul total de grafuri adăugate la coada  $\mathcal{Q}$  este cel mult  $|G|^2$ .

## Exerciții pentru seminarul 6

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

**Exercițiul 9.** Fie  $G = (V, E)$  un graf conex și  $T_1, T_2$  doi arbori parțiali ai lui  $G$  ( $T_1, T_2 \in \mathcal{T}_G$ ).

- Dovediți că  $T_1$  poate fi transformat în  $T_2$  prin aplicarea repetată a următoarei proceduri: șterge o muchie și adaugă o altă muchie arborelui curent.
- Dacă, în plus,  $G$  este 2-conex arătați că  $T_1$  poate fi transformat în  $T_2$  prin aplicarea repetată a următoarei proceduri: șterge o muchie  $uv$  și adaugă o altă muchie  $uw$  arborelui curent.

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

**Exercițiul 10.** Demonstrați că mulțimea de muchii a unui graf complet  $K_n$  ( $n \geq 2$ ) poate fi partiționată în  $\lceil n/2 \rceil$  submulțimi fiecare reprezentând mulțimea de muchii ale unui arbore (subgraf al lui  $K_n$ ).

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



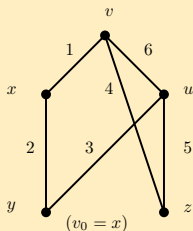
**Exercițiul 13.** Se consideră rețeaua stradală a unui oraș. Se știe că putem îndepărta posibilitatea de a merge în cerc în oraș pe această rețea prin blocarea a cel mult  $p$  sensuri de străzi (prin blocarea unui singur sens se înțelege obstrucționarea sensului vizat de pe stradă). Arătați că putem îndepărta posibilitatea de a merge în cerc în oraș pe rețeaua sa stradală prin inversarea a cel mult  $p$  sensuri de străzi.

(Inversarea unui sens pe o stradă cu două sensuri implică păstrarea doar a celuilalt sens; inversarea sensului unei străzi cu un singur sens implică introducerea celuilalt sens.)

**Exercițiul 14.** Câți arbori parțiali are un graf bipartit complet  $K_{n,n}$ ?  
Aceeși întrebare pentru  $K_{p,q}$ .

**Exercițiul 15.** Arătați că un graf cu cel puțin trei noduri este 2-conex dacă și numai dacă orice două noduri distincte ale sale se găsesc pe un circuit.

- Descrیم o metodă simplă de tip **backtracking** pentru a genera toți arborii parțiali ai unui graf conex  $G = (V, E)$ , unde  $V = \{1, \dots, n\}$ ,  $|E| = m$ .
- Mulțimea de muchii,  $E$ , va fi reprezentată cu un tablou  $E[1..2, 1..m]$  cu elemente din  $V$ , cu semnificația: dacă  $v = E[1, i]$  și  $w = E[2, i]$ , atunci  $vw$  este muchia  $i$  a lui  $G$ . Mai mult, vom presupune că primele  $d_G(v_0)$  coloane din tabloul  $E$  au  $v_0$  în linia 1 ( $E[1, i] = v_0$ ,  $\forall i = \overline{1, d_G(v_0)}$ ), pentru  $v_0 \in V$ .



	1	2	3	4	5	6
1	$x$	$x$	$y$	$z$	$z$	$u$
2	$v$	$y$	$u$	$v$	$u$	$v$

- Un arbore parțial  $T \in \mathcal{T}_G$  va fi reprezentat ca o mulțime de  $n - 1$  indecși (în ordine crescătoare) ai coloanelor din tabloul  $E$  (desemnându-i muchiile).
- Întimpul generării, menținem un vector  $T[1..n - 1]$  cu elemente din  $\{1, \dots, m\}$  și o variabilă flag  $i \in \{1, \dots, n\}$  cu următoarele semnificații:  
Căutăm toți arborii parțiali ai lui  $G$ , cu proprietatea că cele mai mici  $i - 1$  muchii sunt:  $T[1] < T[2] < \dots < T[i - 1]$ .
- Pentru exemplul de mai sus, dacă  $i = 3$ ,  $T[1] = 1$ , și  $T[2] = 2$ , atunci arborii parțiali care vor fi găsiți sunt  $\{1, 2, 3, 4\}$ ,  $\{1, 2, 3, 5\}$ ,  $\{1, 2, 4, 5\}$ ,  $\{1, 2, 4, 6\}$  și  $\{1, 2, 5, 6\}$ . Dar dacă  $i = 3$ ,  $T[1] = 3$  și  $T[2] = 5$ , niciun arbore parțial nu va fi găsit.

```
ALL-ST-Gen( $i$ )
```

```
// sunt generați toți arborii parțiali  $G$ , cu cele mai mici  $i - 1$  muchii:  $T[1], \dots, T[i - 1]$ 
```

```
if ( $i = n$ ) then
```

```
    //  $\{T[1], \dots, T[n - 1]\}$  este arbore parțial
```

```
    process( $T$ ); // printează, memorează etc
```

```
else
```

```
    if ( $i = 1$ ) then
```

```
        for ( $j = 1, \overline{d_G(v_0)}$ ) do
```

```
             $T[i] \leftarrow j$ ; A All-ST-Gen( $i + 1$ ) B
```

```
    else
```

```
        for ( $j = \overline{T[i - 1] + 1, m - (n - 1) + i}$ ) do
```

```
            if ( $\langle \{T[1], \dots, T[i - 1]\} \cup \{j\} \rangle_G$  has no circuit) then
```

```
                 $T[i] \leftarrow j$ ; A All-ST-Gen( $i + 1$ ) B
```

- Prin apelul All-ST-Gen(1) obținem  $\mathcal{T}_G$ .
- Pentru a testa dacă graful  $\langle \{T[1], \dots, T[i-1]\} \cup \{j\} \rangle_G$  nu are circuite, observăm că, din construcție,

$$\langle \{T[1], \dots, T[i-1]\} \rangle_G$$

nu are circuite, deci este o pădure (fiecare componentă conexă este un arbore).

- Fie  $root[1..n]$  un vector (global) cu elemente din  $V$  și semnificația:  $root[v]$  = rădăcina componentei conexe care conține  $v$  (unul dintre nodurile sale).
- Înaintea apelului All-ST-Gen(1), vectorul  $root$  este inițializat pentru a satisface proprietatea:  $root[v] \leftarrow v$  ( $\forall v \in V$ ) (deoarece atunci,  $\{T[1], \dots, T[i-1]\} = \emptyset$ ).

- În timpul apelurilor recursive, când se testează dacă muchia  $j$  poate fi adăugată mulțimii  $\{T[1], \dots, T[i-1]\}$  fără a crea vreun circuit, fie  $v = E[1, j]$  și  $w = E[2, j]$ . Atunci,  $\langle \{T[1], \dots, T[i-1]\} \cup \{j\} \rangle_G$  nu are circuite dacă și numai dacă  $v$  și  $w$  sunt în componente conexe diferite ale pădurii, i.e.,  $root[v] \neq root[w]$ .
- Pentru a actualiza vectorul  $root$ , în locurile marcate cu **A** și **B** din algoritm, trebuie făcute următoarele modificări.
- în loc de **A**:
 

```

      S ← ∅; x ← root[v];
      for (u ∈ V) do
        if (root[u] = x) then
          S ← S ∪ {u}; root[u] ← root[w];
      
```

