

# **CENG 481 GRAF TEORİ VE UYGULAMALARI**

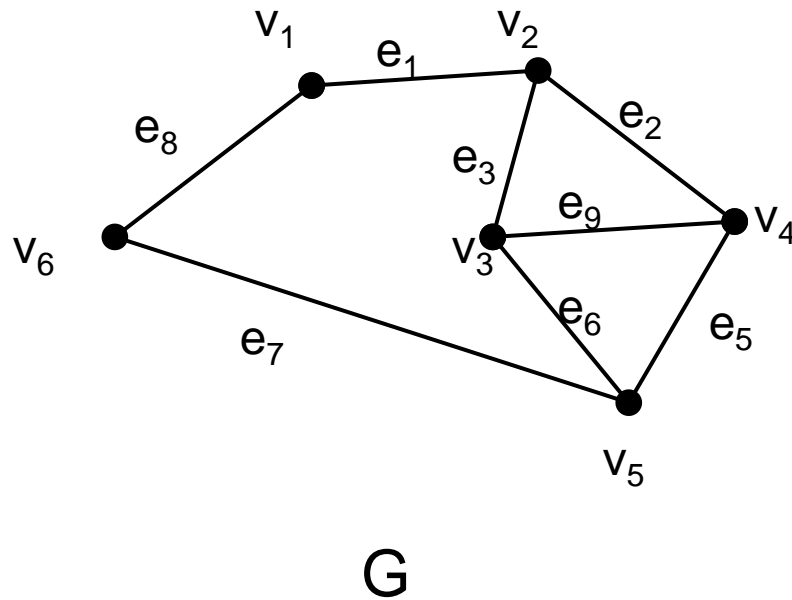
## **Hafta 2**

**Prof. Dr. Tufan TURACI**  
**tturaci@pau.edu.tr**

## Hafta 2 Konular

- 1- Alt graf, Etkilenmiş alt graf
- 2- İki parçalı graf, İki parçalı tam graf
- 3- Tümleyen graf, Düzenli graf
- 4- Graf Parametreleri
  - i-) Bağımsızlık Sayısı ve Algoritması
  - ii-) Örtü Sayısı ve Algoritması

**Alt graf :**  $G=(V,E)$  bir graf olsun.  $V'\subseteq V$ ,  
 $E'\subseteq E$  olmak üzere  $G'=(V',E')$  grafına  $G$  nin  
bir altgrafı denir.



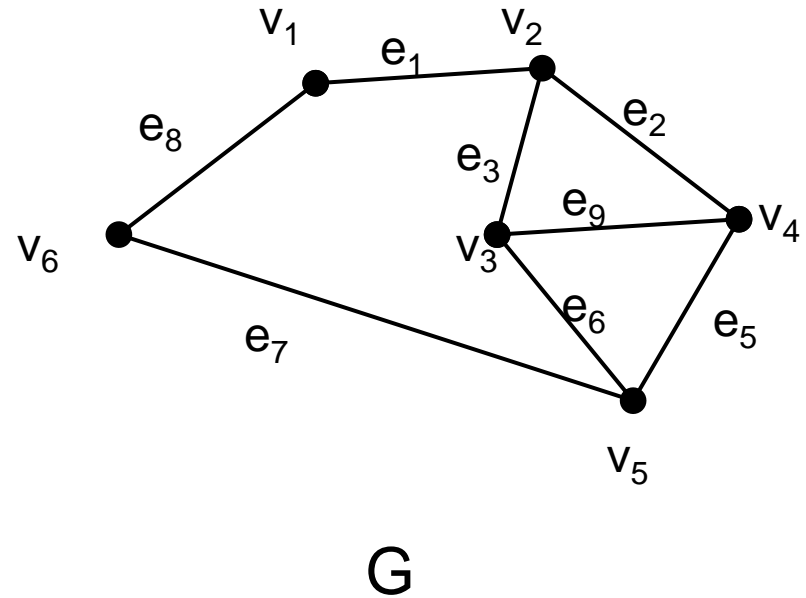
## Alt graf örneği(1)

●  $v_1$

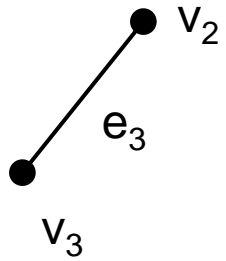
$G'$

$V'=\{v_1\}$

$E'=\emptyset$



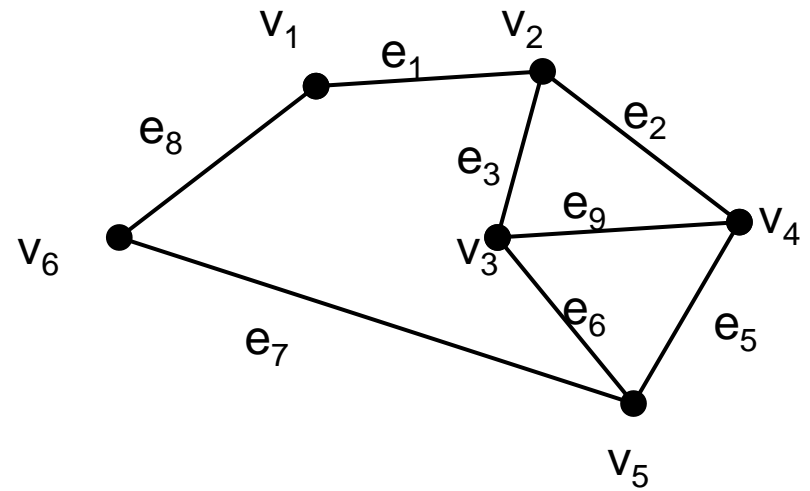
## Alt graf örneği(2):



$G'$

$$V' = \{v_2, v_3\}$$

$$E' = \{e_3\}$$



$G$

## Alt graf örneği(3):

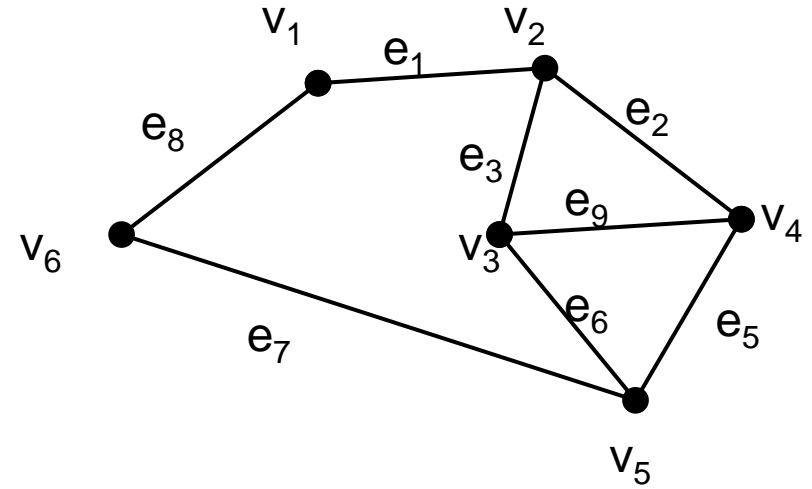
●  $v_1$

$v_4$   
●

$G'$

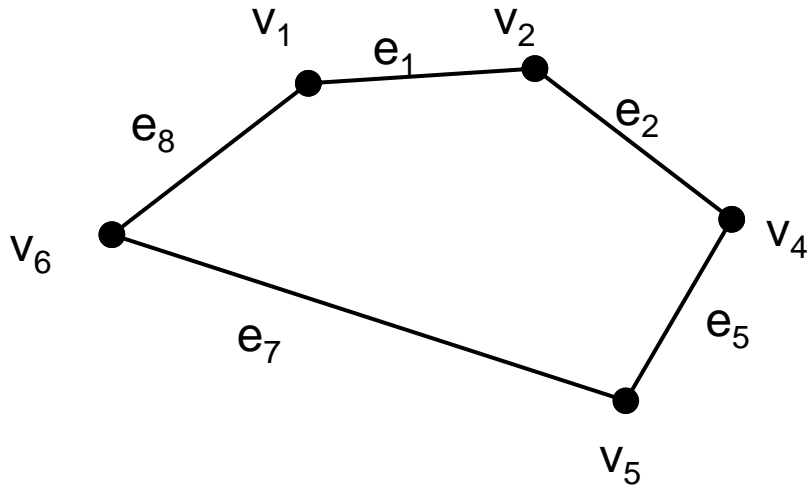
$V'=\{v_1, v_4\}$

$E'=\emptyset$



$G$

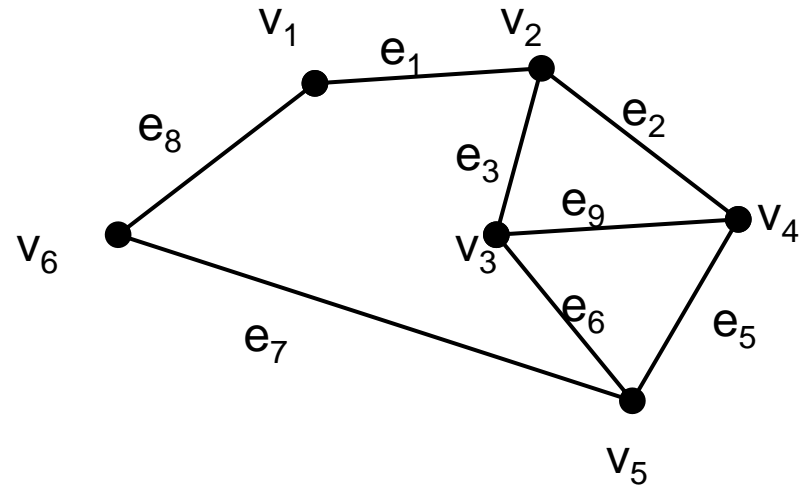
## Alt graf örneği(4):



$G'$

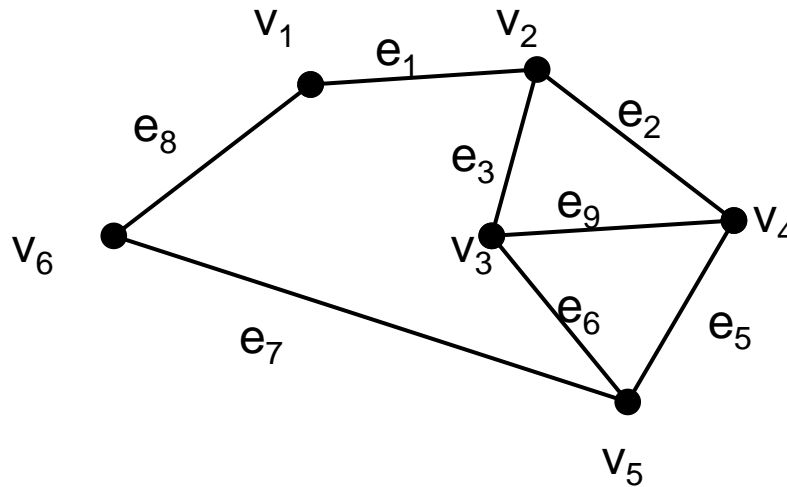
$$V' = \{v_1, v_2, v_4, v_5, v_6\}$$

$$E' = \{e_1, e_2, e_5, e_7, e_8\}$$



$G$

**Etkilenmiş Alt Graf :**  $G=(V,E)$  bir graf olsun.  
 $V' \subseteq V$  alt kümelerindeki tepeler ile, bu tepeler arasında  $G$  de bulunan ayrıtların oluşturduğu grafa etkilenmiş alt graf denir.



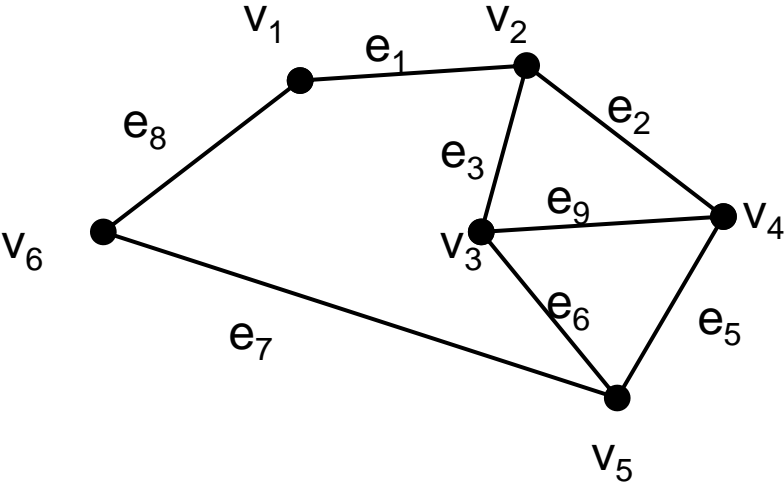
$G$



Etkilenmiş alt graf örneği(1):

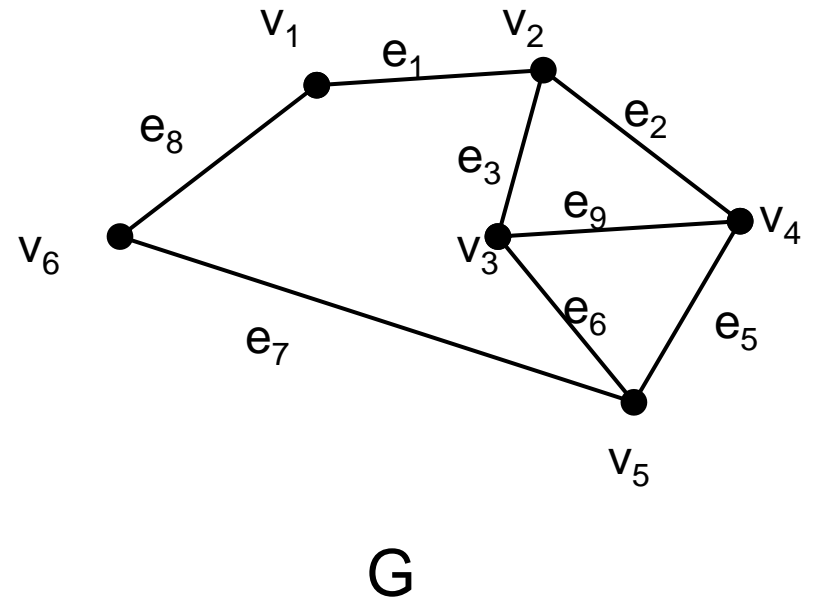
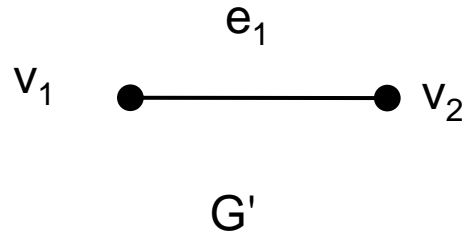
●  $v_1$

$G'$

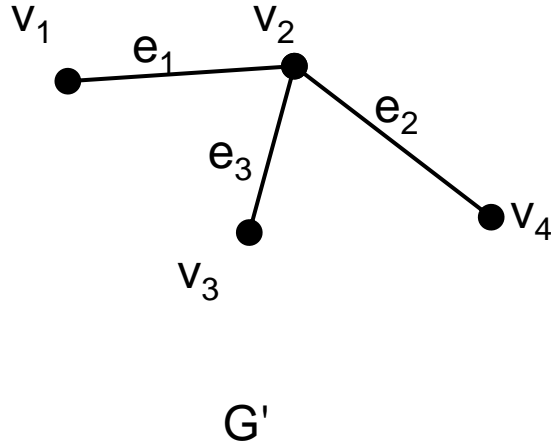


$G$

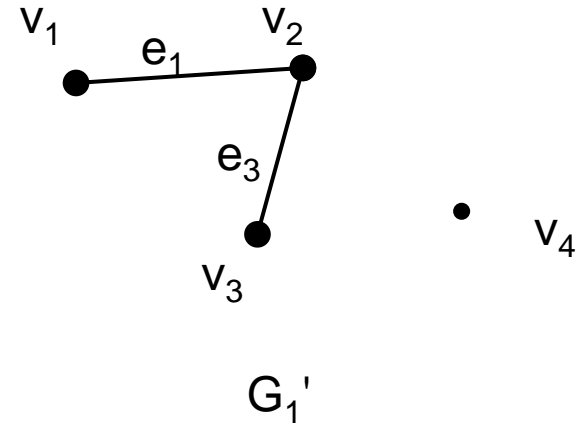
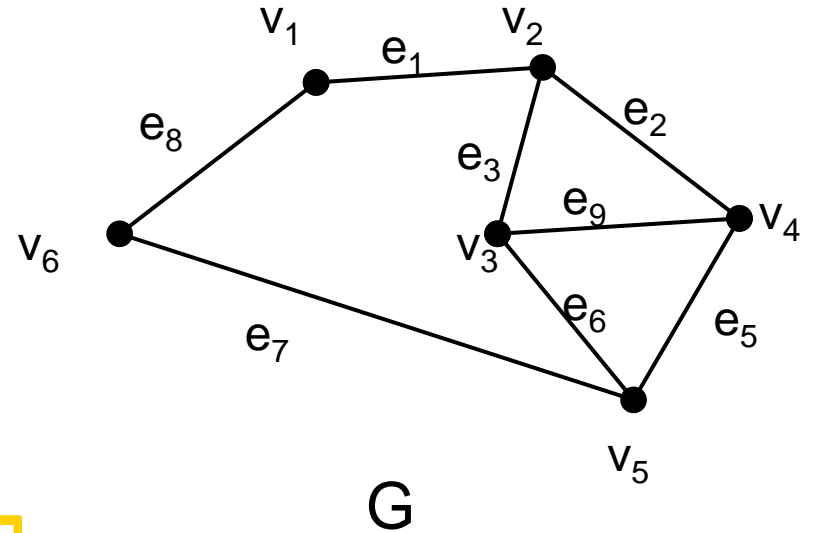
## Etkilenmiş Alt graf örneği(2):



## Etkilenmiş alt graf örneği(3):



Etkilenmiş alt graf değildir.  
Etkilenmiş alt graf olması için  $e_9$  ayrıtı da olmalıydı.



$G_1'$  alt graf olup, etkilenmiş alt graf değildir.

**İki Parçalı Graf:**  $G=(V,E)$  bir graf olsun ve  $V$  kümesi  $V=V_1 \cup V_2$  şeklinde iki kümeye ayrıldığınd

- $V_1$  kümesindeki hiçbir tepe çifti bir ayrıtla birleştirilmemiş ise
- $V_2$  kümesindeki hiçbir tepe çifti bir ayrıtla birleştirilmemiş ise

$G$  ye iki parçalı graf denir.

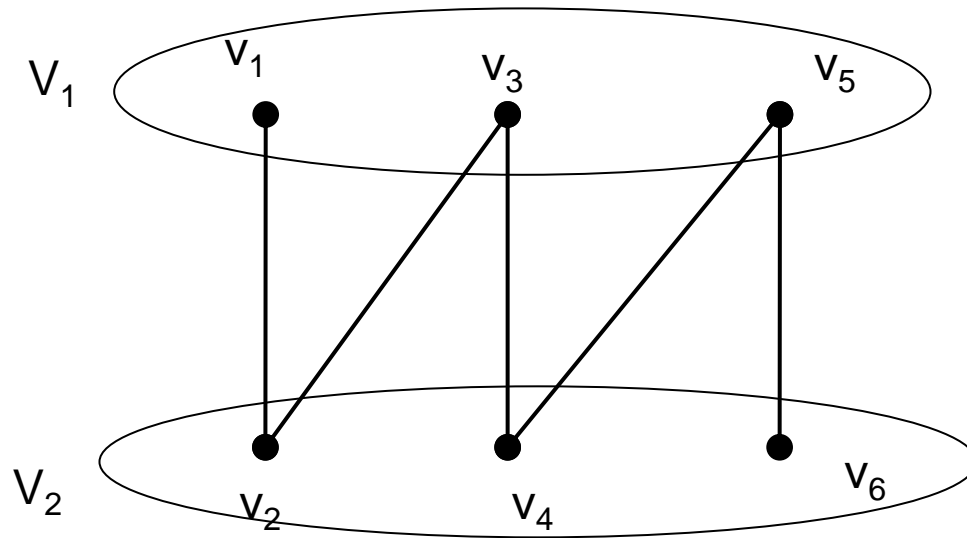


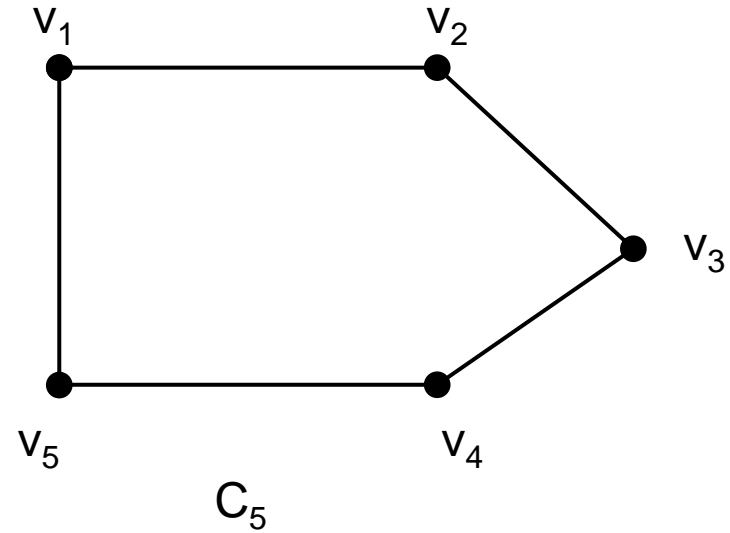
$$V = \{v_1, v_2, v_3, v_4, v_5, v_6\}$$

$$V_1 = \{v_1, v_3, v_5\}$$

$$V_2 = \{v_2, v_4, v_6\}$$

$\Rightarrow P_6$  iki parçalı bir graftır.



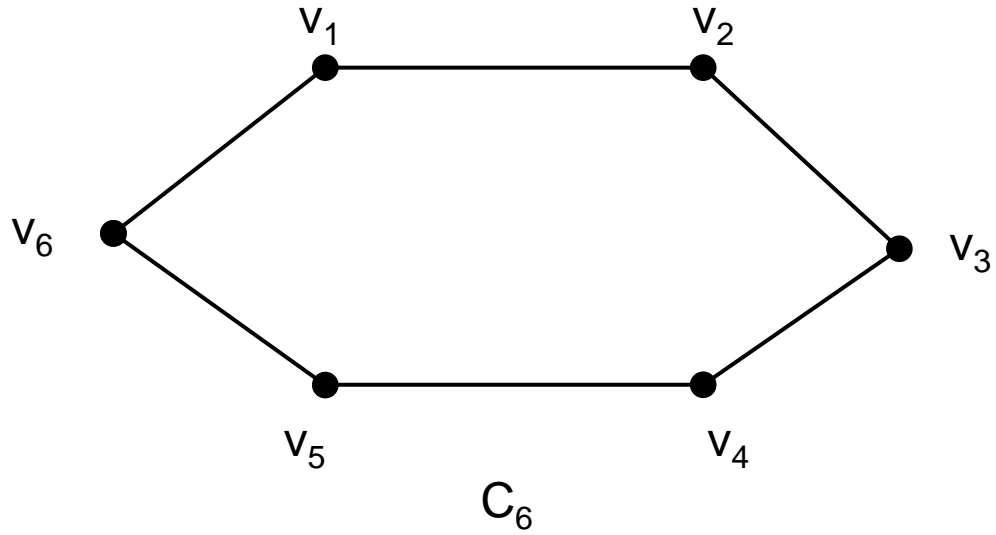


$$V = \{v_1, v_2, v_3, v_4, v_5\}$$

$$V_1 = \{v_1, v_3\}$$

$$V_2 = \{v_2, v_4, v_5\}$$

$C_5$ ; iki parçalı graf değildir.



$$V = \{v_1, v_2, v_3, v_4, v_5, v_6\}$$

$$V_1 = \{v_1, v_3, v_5\}$$

$$V_2 = \{v_2, v_4, v_6\}$$

$C_6$ ; iki-parçalı graftır.

**Sonuç 1:** Her  $P_n (n \geq 2)$  yol grafi 2-parçalı graftır.

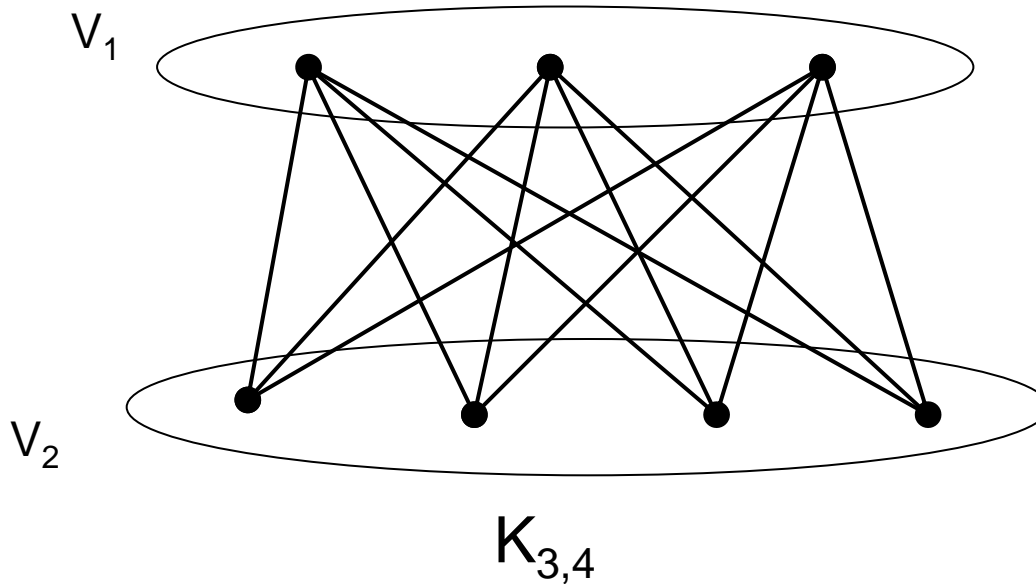
**Sonuç 2:** Her  $C_n (n \text{ çift})$  çevre grafi 2-parçalı graftır.

**Sonuç 3:** Her  $C_n (n \text{ tek})$  çevre grafi 2-parçalı graf değildir.



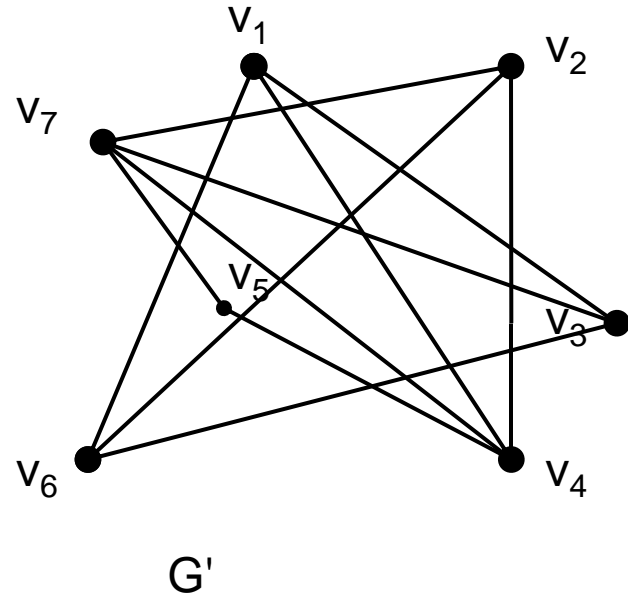
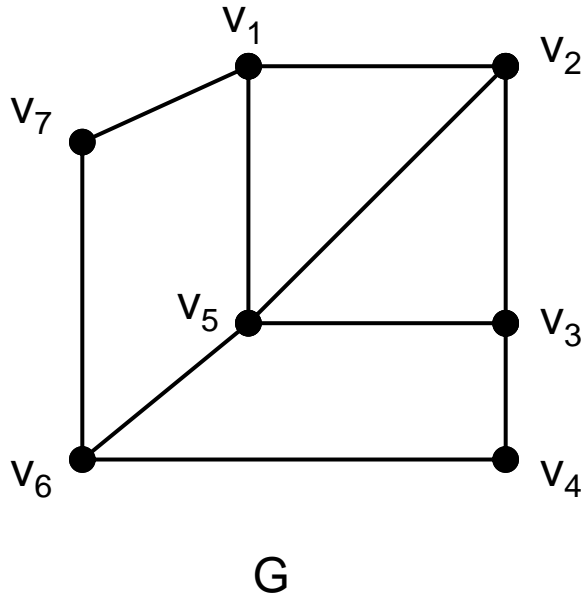


**İki Parçalı Tam Graf:**  $G$  iki parçalı bir graf,  $V$  tepeler kümesi  $V=V_1 \cup V_2$  olsun  $V_1$  deki her bir tepe  $V_2$  deki her bir tepeye bir ayrıtla birleştirilmiş ise bu grafa iki-parçalı tam graf denir.  $V_1=m$ ,  $V_2=n$  olmak üzere iki-parçalı bir tam graf  $K_{m,n}$  ile gösterilir.

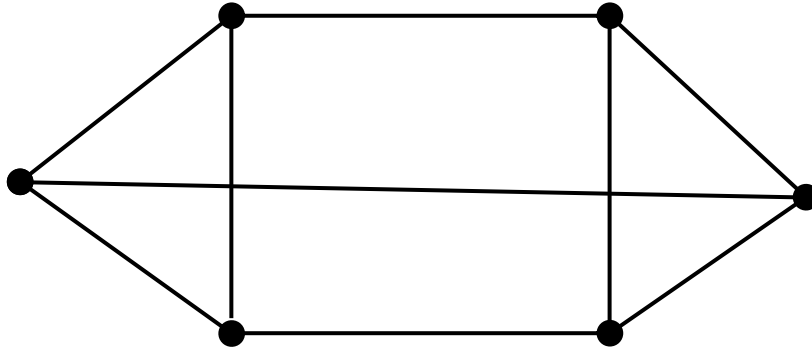


**Tümleyen Graf:**  $G(V,E)$  bir graf olsun.  $G$  grafının tümleyen grafi  $G'(V',E')$  ile gösterilir. Burada  $V'=V$  dir.

$E'$  kümesi ise  $G$  grafında olmayan ayritların oluşturduğu graftır.



**r düzenli (r-regular) graf:** Bir  $G$  grafının tüm tepelerine ait dereceler aynı ise bu grafa düzenli graf denir. Çevre graf ile tam graf regüler graflara örnek graflardır.



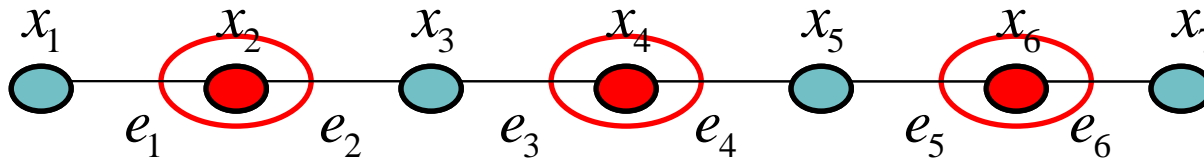
3-düzenli graf

# GRAF PARAMETRELERİ

**Bağımsızlık Sayısı:**  $S \subseteq V(G)$ , bir grafının tepeler kümesinin herhangi bir alt kümesi olsun.  $S$  kümesindeki tepeleri ikişerli aldığımızda bu tepeler,  $G$  grafında bir ayrıta sahip değilse  $S$  kümesine **bağımsız küme** denir. Bir  $G$  grafi birden fazla bağımsız kümeye sahip olabilir. Bu kümeler içinde en çok elemana sahip olan kümenin eleman sayısına  $G$  grafının **bağımsızlık sayısı** (independence number) denir ve  $\beta(G)$  ile gösterilir.



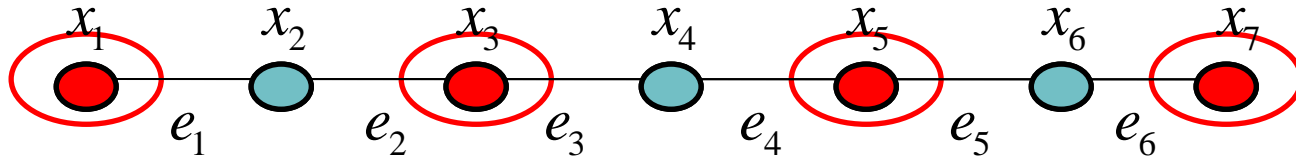
Örnek 1:  $P_7$  grafinın bağımsızlık sayısını bulalım.



$$S_1 = \{x_2, x_6\}$$

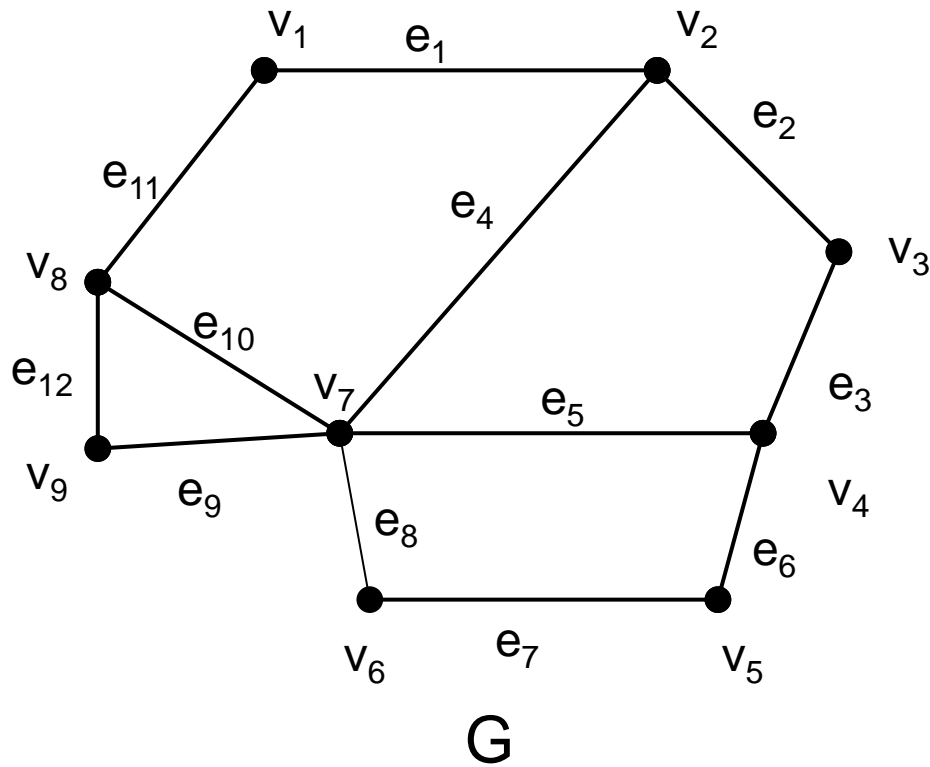
$$S_2 = \{x_2, x_6, x_4\}$$

3' den daha büyük elemanlı bağımsız küme bulunabilir mi?



$$S_3 = \{x_1, x_3, x_5, x_7\}$$

$S_1, S_2$  ve  $S_3$  bağımsız kümelerdir.  $\beta(P_7) = 4$  dir.



$S=\{v_1, v_2\} \Rightarrow$  Bağımsız küme değil.


$S=\{v_1, v_3\} \Rightarrow$  Bağımsız küme.

$S=\{v_1, v_3, v_5\} \Rightarrow$  Bağımsız küme.

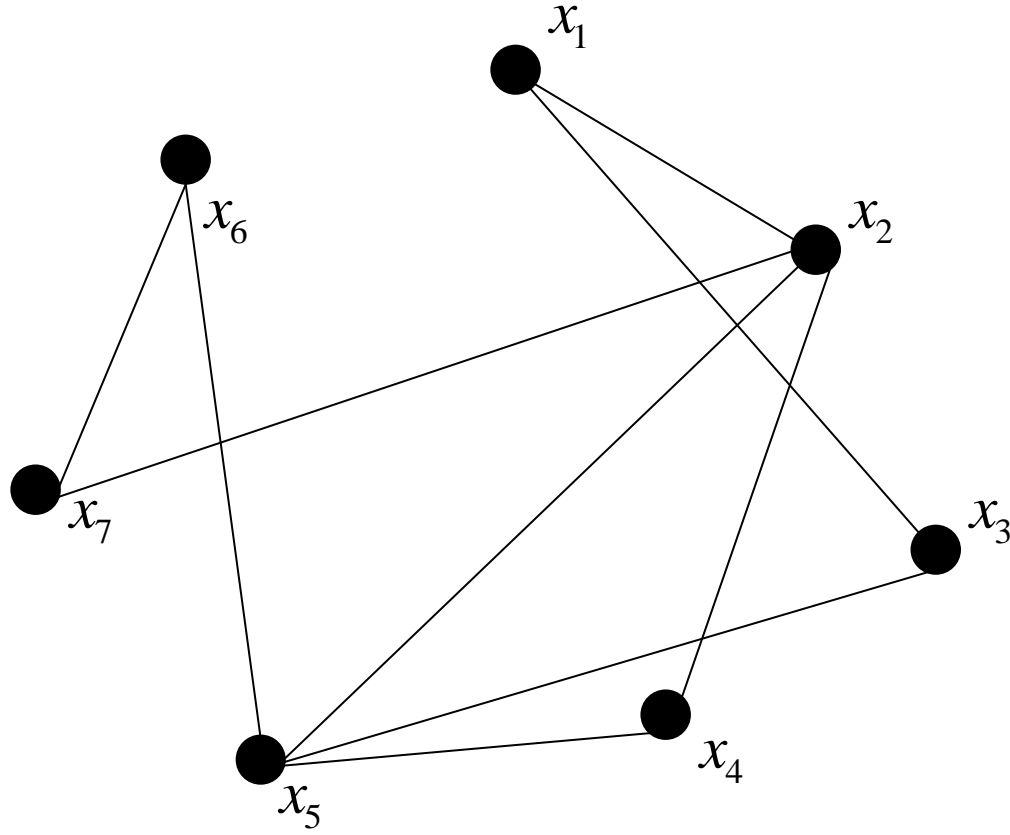
$S=\{v_1, v_3, v_5, v_7\} \Rightarrow$  Bağımsız küme.

$S=\{v_2, v_4, v_6, v_8\} \Rightarrow$  Bağımsız küme.

$\beta(G)= 4$  tir.

**Problem :** Bir askeri malzeme deposunda bazı malzemelerin yan yana bulunması durumunda kimyasal tepki verdikleri bilinmektedir. **Şekil 1'** deki grafta malzemeler birer tepe,  kimyasal tepkiye giren malzemeler ise birer ayrıtla ifade edilmiştir. Bu malzemeleri hiçbir problem olmaksızın bir depoya yerleştirebileceğimiz bir plan hazırlayabilir miyiz?

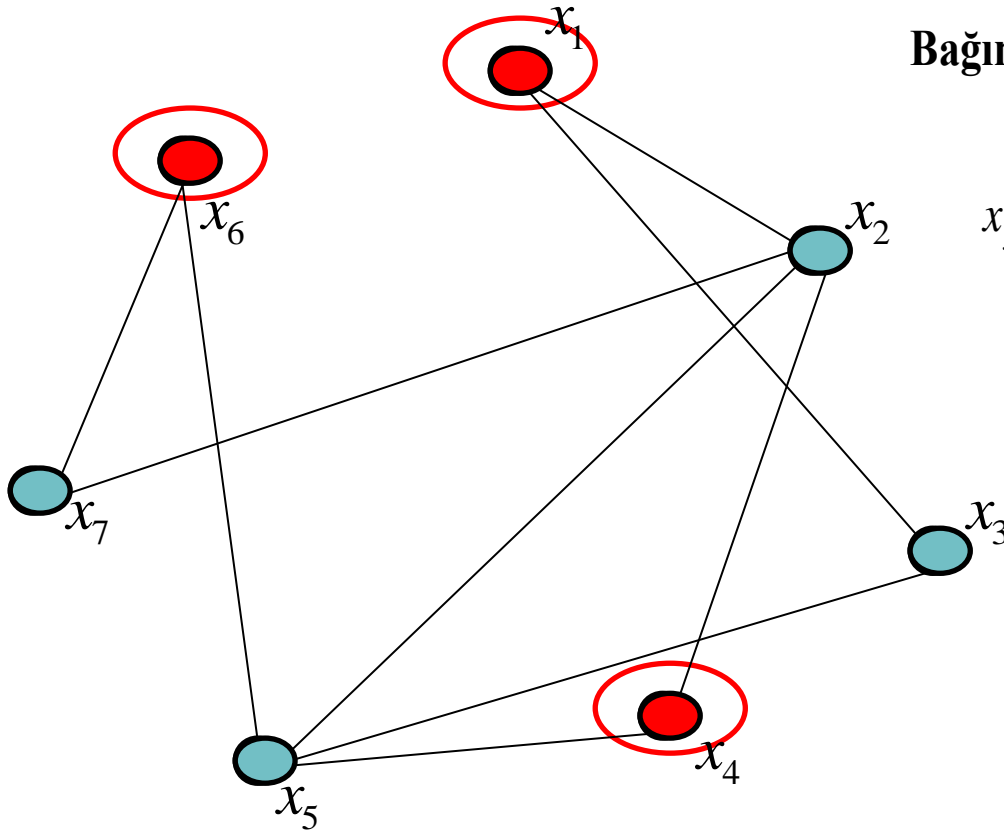




Şekil 1

Burada açıktır ki,  $x_1$  ve  $x_4$  yanyana gelebilir. Fakat  $x_1$  ve  $x_2$  yanyana gelemmez.

## Bağımsız Küme Bulmanın Matematiksel Modeli



$$x_j = \begin{cases} 1 \\ 0 \end{cases}$$

,  $x_j$  tepesi secilmis ise

,  $x_j$  tepesi secilmemis ise

$$x_1 + x_2 \leq 1$$

$$x_1 + x_3 \leq 1$$

$$x_2 + x_4 \leq 1$$

$$x_2 + x_5 \leq 1$$

$$x_2 + x_7 \leq 1$$

$$x_3 + x_5 \leq 1$$

$$x_4 + x_5 \leq 1$$

$$x_5 + x_6 \leq 1$$

$$x_6 + x_7 \leq 1$$

$$k.a., Z_{\max} = \sum_{j=1}^n x_j$$

$$x_1 = 1 \quad x_2 = 0$$

$$x_4 = 1 \quad x_3 = 0$$

$$x_6 = 1 \quad x_5 = 0$$

$$x_7 = 0$$

$$\sum_{j=1}^7 x_j = 3 \Rightarrow \beta(G) = 3$$

Bağımsız küme problemi,

- Yerleştirme(Depolama) problemlerinin çözümünde
- Bağımsız kümeyi bulan bir çok algoritma vardır. En önemli algoritma Paull-Unger Algoritmasıdır.

## Paull - Unger Algoritması

Paul-unger algoritması bir graftaki tüm maksimal bağımsız kümeleri ve bağımsızlık sayısını bulur.

**Tanım 2.2.1 (Prather, 1976)**  $\Sigma = \{\sigma_1, \sigma_2, \dots, \sigma_n\}$  alfabeti ile  $x = \sigma_{i_1} \sigma_{i_2} \dots \sigma_{i_k}$  gibi kelimeleri oluşturabiliriz. ( $\epsilon$ , null uzunluğu sıfır olan bir kelimedir).  $\Sigma^*$  gösterilimi verilen  $\Sigma$  alfabetinden üretilen bütün kelimelerin kümesinin bir yığındır.

$$\Sigma^* = \{x: x, \Sigma \text{ alfabetinin bir kelimesidir} \}$$

Şimdi bu yığını ilişkilendiren bir ikili işlem tanımlayacağız. Bu işlem genellikle kelimelerin birleşimi (bitişikliği) olarak adlandırılır.  $x$  yukarıda verildiği gibi ve  $y = \sigma_{j_1} \sigma_{j_2} \dots \sigma_{j_l}$  olsun. Birinci kelimeyi yazdıktan sonra ikinci kelimenin sembollerini yan yana dizeriz

$$x.y = \sigma_{i_1} \sigma_{i_2} \dots \sigma_{i_k} \sigma_{j_1} \sigma_{j_2} \dots \sigma_{j_l}$$

görüldüğü gibi her  $x$  kelimesi için

$$x.\epsilon = \epsilon.x = x$$

öyleyse  $\Sigma^* = (\Sigma^*, \cdot, \epsilon)$  bir monoid'e dönüştüğü görülür çünkü bitişiklik işleminin birleşmeli olduğu açıktır.  $\Sigma^*$ 'yi  $\Sigma$  alfabelinden elde edilen **free monoid** olarak adlandırırız.

**Örnek 2.2.1:** Eğer  $\Sigma = \{0,1\}$  ise

$\Sigma^* = \{0, 1, 00, 01, 10, 11, 000, 001, \dots\}$  şeklindedir.  $\Sigma^*$ 'deki bitişiklik işleminin bazıları aşağıdaki gibi gösterilebilir.

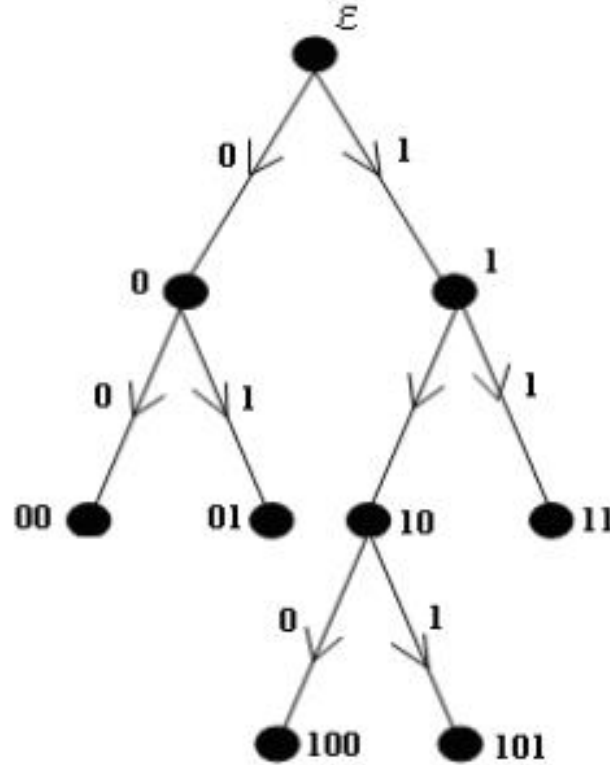
$$10 \cdot 001 = 10001$$

$$111 \cdot 01 = 11101$$

$$10 \cdot \epsilon = 10$$

## ALGORİTMA

$x E y \Leftrightarrow y = x\sigma$  ( $\sigma \in \Sigma$ ) ayrıt bağıntısı aracılığı ile birleştirilmiş bir sonsuz (digraf)  $G = (\Sigma^*, E)$  grafının tepeler kümesi olarak,  $G$  grafının tepelerinin alfabelinden üretilen bütün kelimelerin kümesini  $\Sigma^*$  ile gösterelim. (Prather, 1976).



Şekil 1.a

Şekil 1.a' da  $B=\{0,1\}$  ikili alfabesinden oluşturulan  $B^*$  için sonsuz bir yönlendirilmiş ağaç graf gösterilmiştir. Şekil 1.a' da görüldüğü gibi yönlendirilmiş dallar olarak gösterilen ayrıtları soldan sağa doğru etiketleriz, grafın bütününde  $B^*$ 'ı temsil eden bu sıralamayı gerçekten incelemek istiyorsak bu gerekli olacaktır. Bu yönlendirilmiş ağaç grafta ilk tepe grafın kök tepesi olarak adlandırılır. Bu aynı zamanda  $\Sigma^*$  kümesindeki null kelimesidir.

$V$ ,  $B^*$ 'ın sonlu bir alt kümesi olmak üzere  $T = (V, E)$  şeklinde sonsuz (digraf) bir ikili ağaçtır,

- (i)  $x\sigma \in V \Rightarrow x \in V (\sigma \in B)$
- (ii)  $x E y \Leftrightarrow$  bazı  $\sigma \in B$  için  $y = x\sigma$

(i). madde bize her bir tepeden köke birleşmiş bir yol olduğunu garanti eder. (ii). madde ise daha önce söylendiği gibi ayrıt bağıntısıdır. Aynı zamanda ağacın uçlarındaki düğümlere (tepelere) ait olan kümeleri gösteren alt küme aşağıdaki şekilde tanımlanır.

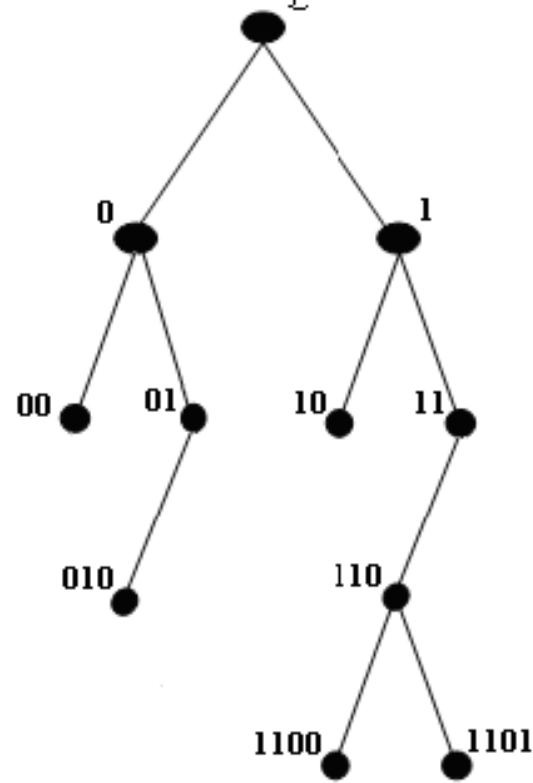
$$L = L(T) = \{x \in V: \text{bütün } \sigma \in B \text{ için } x\sigma \notin V\} \subseteq V$$

**Örnek :** Şekil **1.a' da** bir ikili ağaç gösterilmektedir.

İkili ağacın tanımındaki (i). madde aşağıdaki şekilde örneklendirilmiştir.

$$1101 \in V \Rightarrow 110 \in V \Rightarrow 11 \in V \Rightarrow 1 \in V \Rightarrow \varepsilon \in V$$





**Şekil 1.b.**

Graftaki her bir tepenin gösterdiği  $x$  kelimesinin ( $x \in \Sigma^*$ ) önce gelenlerinin grafta daha önceden var olduğu (i). madde tarafından garanti edilir. Örneğin, **Şekil 1.b.** 'deki 1101 graf tepesi  $1101=110 \cdot 1$  şeklinde oluşturulmuş olup bu tepenin önce geleni 110' dır ve bu grafta daha önceden grafın tepesi olarak vardır. Bu özel T ağacı aşağıdaki düğümler (tepeler) kümesine sahiptir.

$$L = L(T) = \{ 00, 10, 010, 1100, 1101 \}$$

$T = (V, E)$  ikili ağacı algoritmik işlemler içerisinde kullanılabilir. Sadece  $V \subseteq B^*$  'daki sonlu sayıdaki tepeler kullanılmalıdır. Bu yüzden algoritmada  $T$  için en uygun veri tipi dizi veri yapısıdır

**T: array  $B^*$  of A**

Burada  $A$ , tepeler etiketlenirken kullanılan cebirdir.  $B^*$  sonsuz bir küme olduğu için algoritmanın çalıştırılması sırasında bu tepelerin hepsini etiketlendirmek imkânsızdır. Sadece  $V \subseteq B^*$  bir sonlu alt kümesi etiketlendirilecektir. Tepelerin etiketlendirildiği yukarıdan aşağıya doğru sıradan dolayı  $V$  alt kümesinin  $(i)$  durumu sağladığını söyleriz. Yani işlem sonlandığı zaman etiketlendirilmiş bir ikili ağaç elde ederiz.

Etiketlendirilmiş ikili ağaçlar, bir  $G$  grafinin tüm maximal bağımsız kümelerinin bulunmasında ve aynı zamanda  $\beta$  bağımsızlık sayısının hesaplanmasında kullanılır. Bu hesaplamalar ilk M.C.Paull ve S.H.Unger tarafından yapılmıştır. Onların oluşturduğu çözümü aşağıdaki algoritma ile gösterebiliriz.

T: array  $B^*$  of  $\mathcal{P}(V)$                        $\beta, i, j, n$ : positive integer

E: array  $n^+ \times n^+$  of  $B$                       L, M: subset of  $B^*$

v: array  $B$  of  $V$                               x: element of  $B^*$

$\sigma$ : element of  $B^*$

begin T  $\leftarrow \emptyset$ ; T[ $\epsilon$ ]  $\leftarrow V$ ; L  $\leftarrow \{\epsilon\}$ ;

  for  $j \leftarrow 1$  to  $n - 1$  do

    for  $i \leftarrow j + 1$  to  $n$  do

      if  $E[i, j] = 1$  then

        begin M  $\leftarrow \{x \in L: \{v_i, v_j\} \subseteq T[x]\}$ ; L  $\leftarrow L \sim M$ ;

```

 $v[0] \leftarrow v_i; v[1] \leftarrow v_j;$ 
for  $x \in M$  do
  for  $\sigma \in B$  do
    begin  $T[x\sigma] \leftarrow T[x] \sim \{v[\sigma]\};$ 
    if  $((T[x\sigma] \not\subseteq T[y]) \text{ for all } y \in L)$  then
       $L \leftarrow L \cup \{x\sigma\}$ 
    end;
  end;
 $\beta \leftarrow \max_{x \in L} \{|T[x]|\}$ 

```

End.

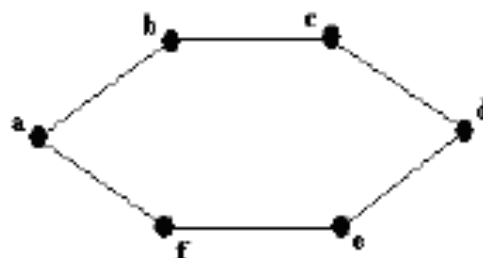
Bu algoritma herhangi bir  $G$  grafi için  $\beta$  bağımsızlık sayısını belirler ve aynı zamanda bütün maksimal bağımsız kümelerin listesini verir. Bunlardan daha büyük bir bağımsız küme içerilmez.

**Örnek :** Paull-Unger algoritmasındaki çeşitli adımları sırasıyla örneklemek için **Şekil 1.c.** deki  $C_6$  grafını inceleyelim. İlk olarak **Şekil 1.d.** (a)' da gösterildiği gibi yalnızca ağacımızın kökünü etiketleyelim.

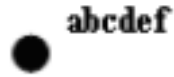
Bu noktada  $L = \{e\}$  dir. Daha sonra alt üçgensel matristeki girdileri inceleyebiliriz.

Alt üçgensel matris,

<b>b</b>	1				
<b>c</b>	0	1			
<b>d</b>	0	0	1		
<b>e</b>	0	0	0	1	
<b>f</b>	1	0	0	0	1
	<b>a</b>	<b>b</b>	<b>c</b>	<b>d</b>	<b>e</b>

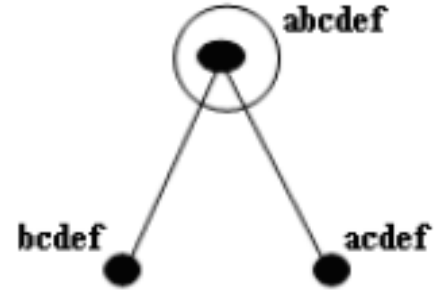


**Şekil 1.c.**



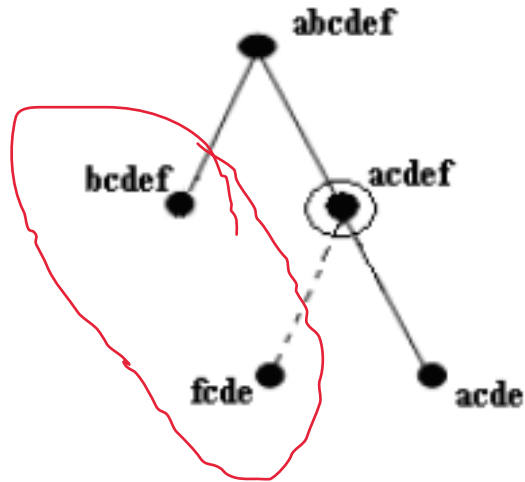
$L = \{\epsilon\}$

(a)



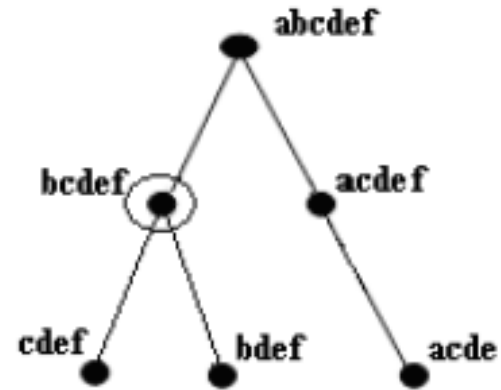
$L = \{0, 1\}$

(b)



$L = \{0, 11\}$

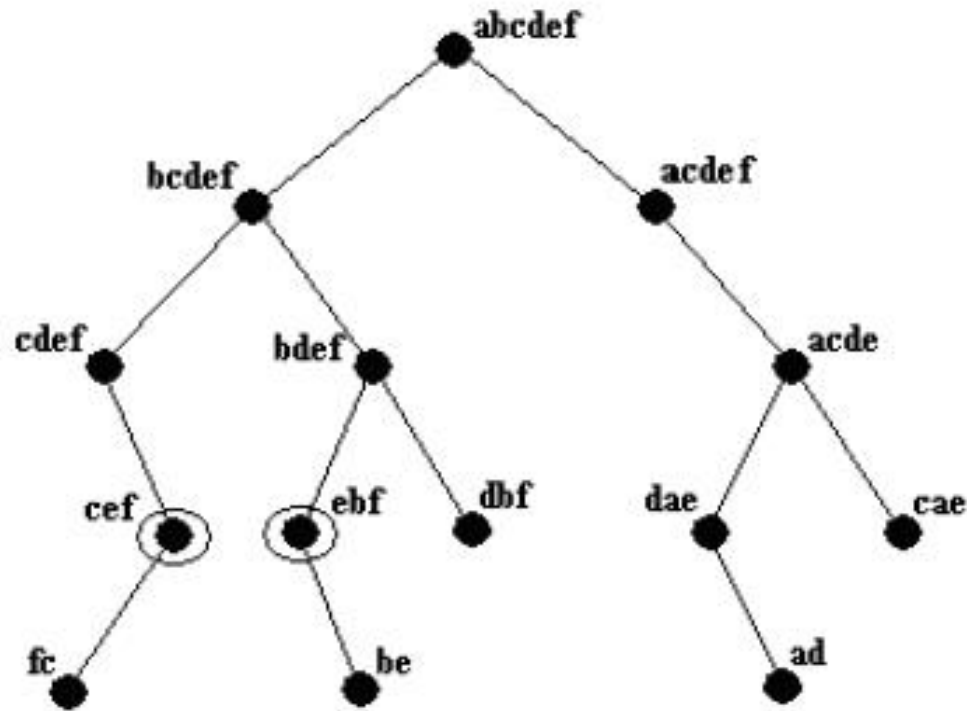
(c)



$L = \{00, 01, 11\}$

(d)

Şekil 1.d.



Şekil 1.e.

Yuvarlak içine alınmış düğümler, matriste son olarak bulunmuş 1-girişi (eEf) tanımlandığı zaman işaretlenmiş M' lerdir. Bununla birlikte sonuç olarak,

$$L=\{0010, 0101, 011, 1101, 111\} \text{ dir.}$$

Böylece,  $\beta = \max_{x \in L} \{|T[x]|\} = 3$  tür.

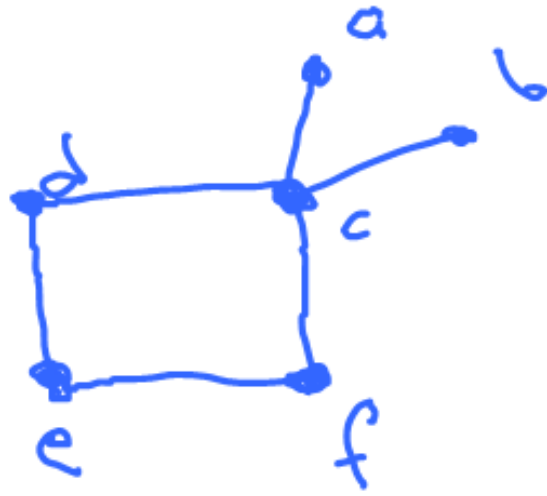
Algoritmanın uygulaması sonucunda elde edilen bütün dalların etiketleri maksimal bağımsız kümelerle eşleşir. Sırasıyla bütün bağımsız alt kümeler,

$$\{f, c\}, \{b, e\}, \{d, b, f\}, \{a, d\}, \{c, a, e\}$$

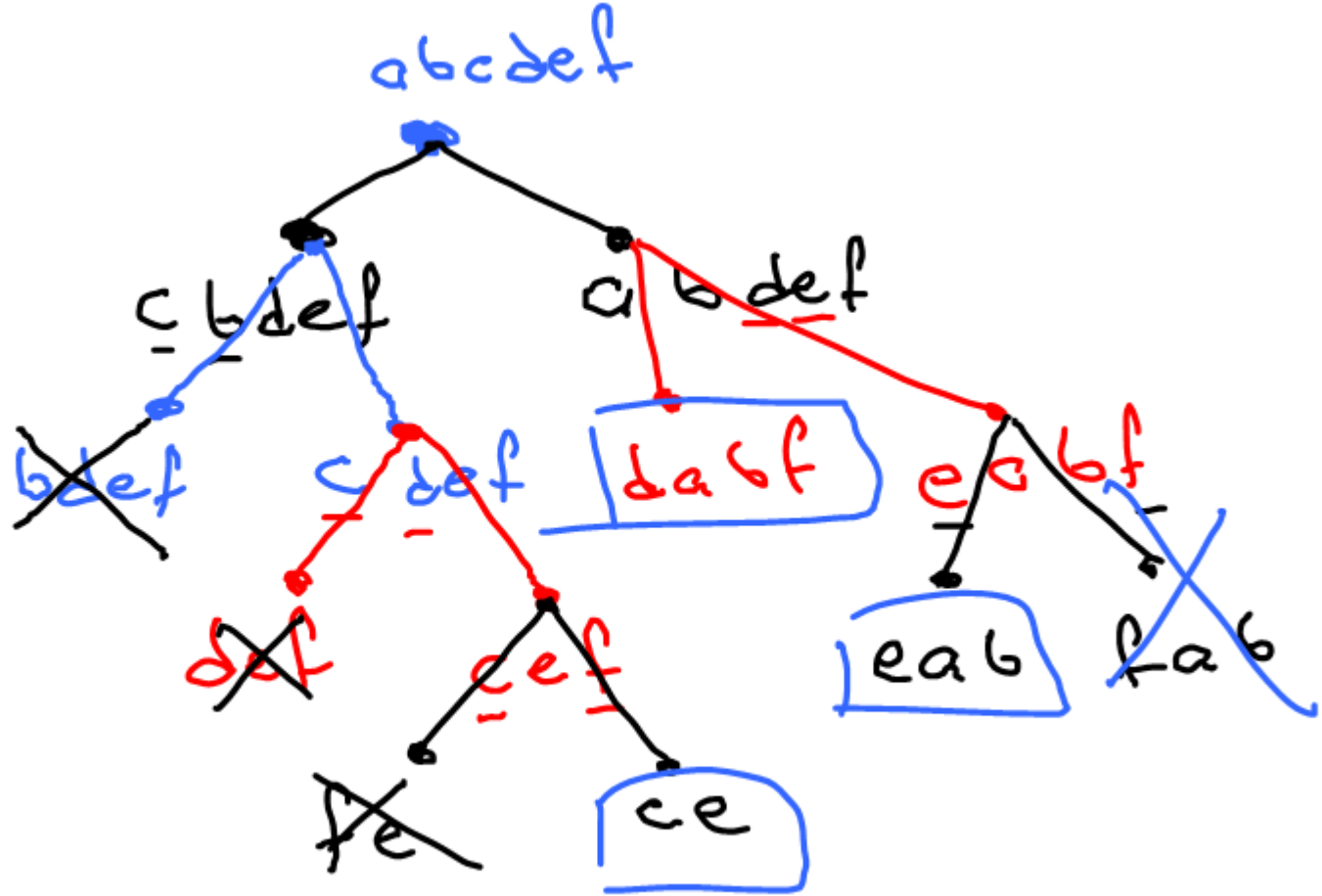
şeklindedir.



rank



$$A(G) = \begin{bmatrix} a & b & c & d & e & f \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & -1 & 1 & 0 \\ 0 & 0 & 0 & 0 & -1 & 1 \end{bmatrix}$$



Maximal besuchte K mmer.

$\{c, e\}, \{e, a, b\}, \{a, b, d, f\}$

$$B(G) = 4$$

## Önemli Graflar için Bağımsızlık Sayıları

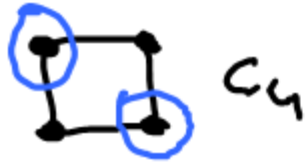
1-) Yol Graf

$$\beta(P_n) = \begin{cases} \frac{n+1}{2} & , n \text{ tek} ; \\ \frac{n}{2} & , n \text{ çift} . \end{cases}$$



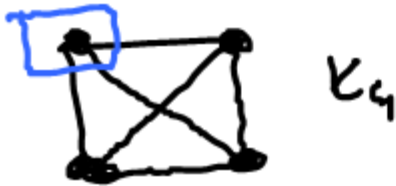
2-) Gevre graf

$$\beta(C_n) = \begin{cases} \frac{n-1}{2} & \text{in tek;} \\ \frac{n}{2} & \text{in çift.} \end{cases}$$



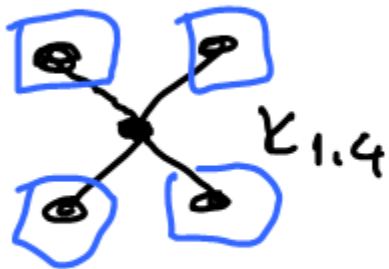
3-) Tam Graf

$$\beta(K_n) = 1$$



4) Yıldız Graf

$$\beta(K_{1,n}) = n$$



5) Teknokutlu graf

$$\beta(W_{1,n}) = \beta(C_n) = \begin{cases} \frac{n}{2} & , n \text{ çift} \\ \frac{n-1}{2} & , n \text{ tek} \end{cases}$$



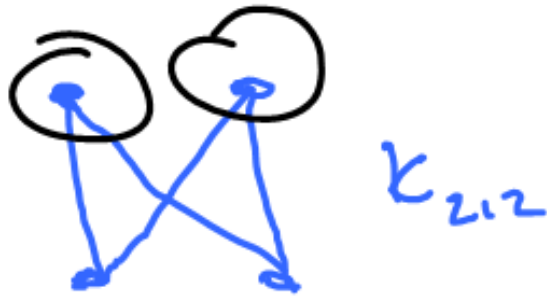
$W_{1,4}$



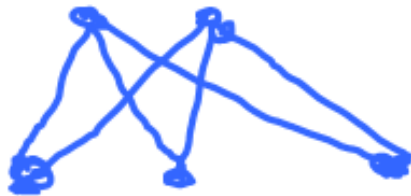
$W_{1,5}$

6) iki parçlı tan grf

$$\beta(k_{m,n}) = \max \{m, n\}$$



$k_{2,2}$



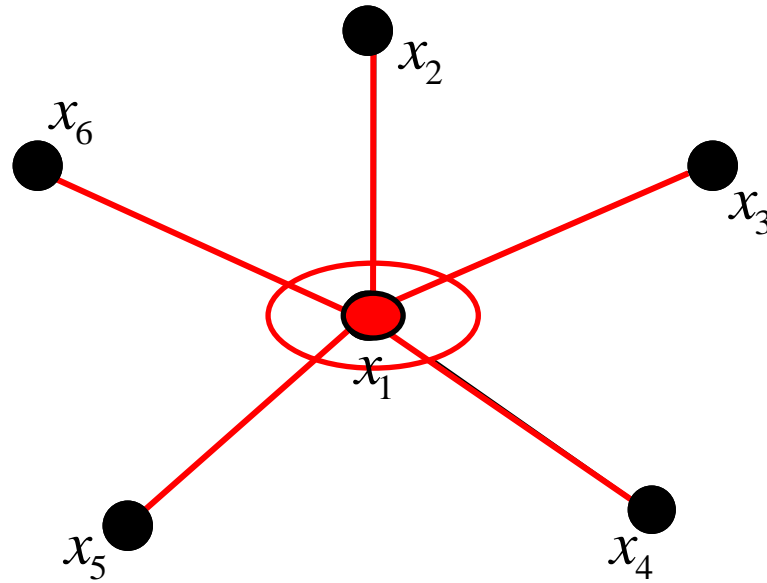
$k_{2,3}$



$k_{3,2}$

**Örtü Sayısı:**  $S \subseteq V(G)$ , bir  $G$  grafinin tepeler kümesinin boş olmayan bir alt kümesi olsun.  $G$  grafindaki her bir ayrıtın en az bir uç noktası bu  $S$  kümesinde ise bu kümeye **örtü kümesi** denir. Bir  $G$  grafi birden çok örtü kümesine sahip ola bilir. Bu kümeler içinde en az elemana sahip olan kümenin eleman sayısına  $G$  grafinin **örtü sayısı** (covering number)denir ve  $\alpha(G)$  ile gösterilir.

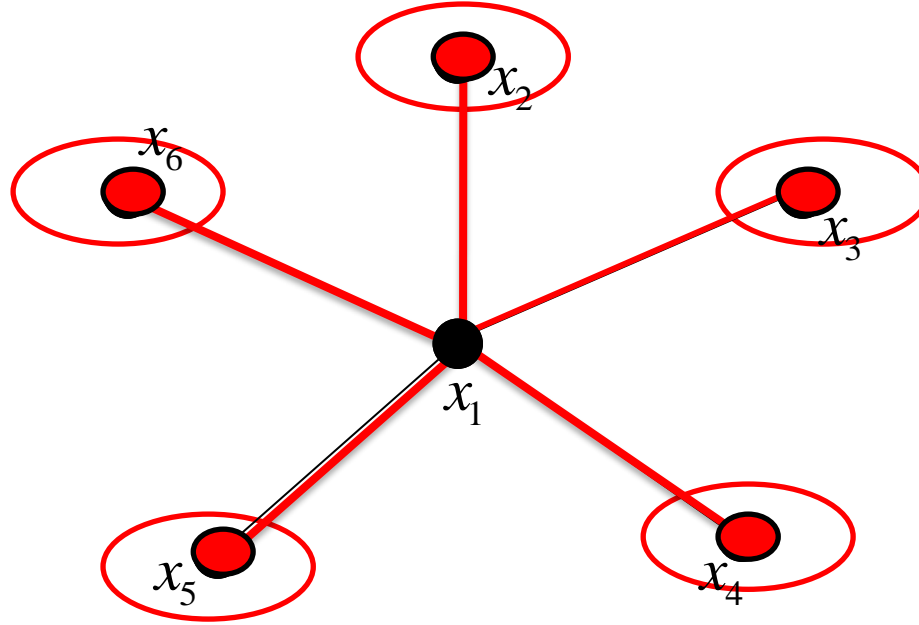
Örnek 4.2:  $K_{1,5}$  grafının örtü sayısını bulalım.



$$S_1 = \{x_1\}$$

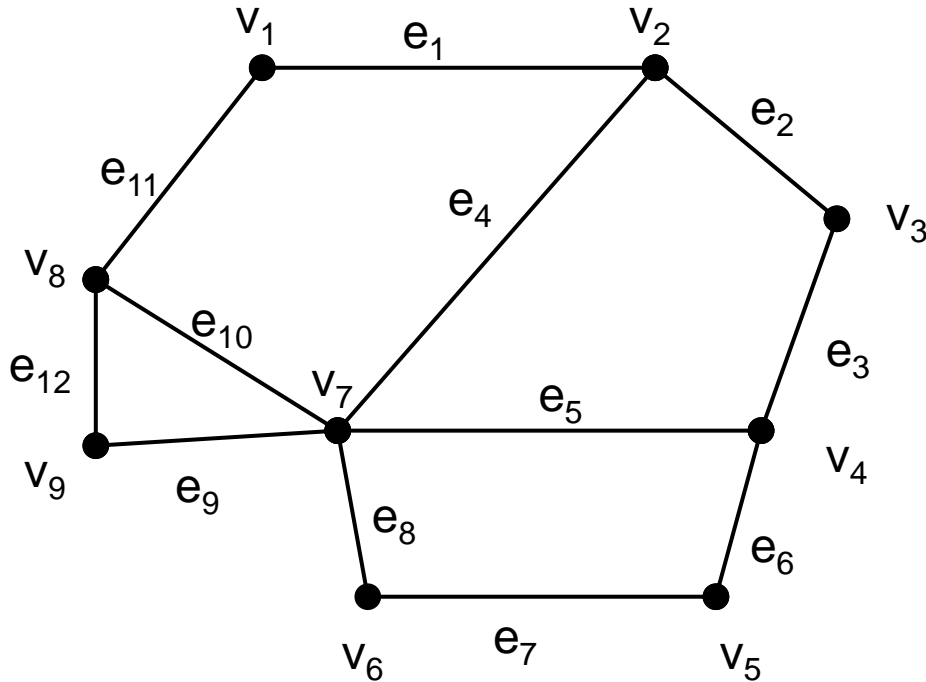


Benzer şekilde başka bir örtü kümesi,



$$S_2 = \{x_2, x_3, x_4, x_5, x_6\}$$

$S_1$  ve  $S_2$  örtü kümeleridir.  $K_{1,5}$  grafının örtü sayısı  $\alpha(K_{1,5}) = 1$  dir.



$$V(G) = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_9\}$$

$$S = \{v_1, v_3, v_5\} \Rightarrow \text{Örtü kümesi değildir.}$$

$$S = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8\} \Rightarrow \text{Örtü kümesidir.}$$

$$S = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_9\} \Rightarrow \text{Örtü kümesidir.}$$

$$S = \{v_2, v_4, v_6, v_7, v_8\} \Rightarrow \text{Örtü kümesidir.}$$

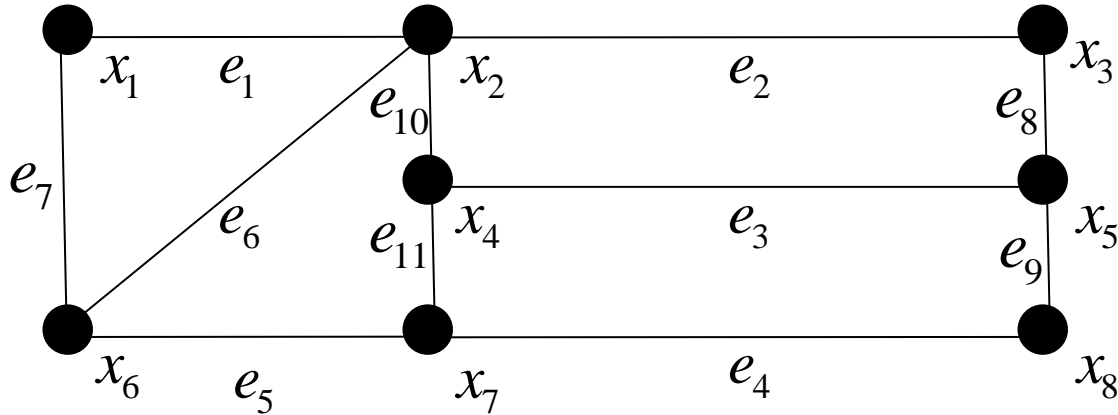
En az elemanlı kümenin eleman sayısı=5 dir ve  $\alpha(G) = 5$  tir.

Örtü kümesi problemi,

- Güvenlik problemlerinin çözümünde
- Yerleştirme Problemlerinin(Ambulans, İtfaiye) çözümünde

**Problem 4.2:** Öğrencilerin gün içi ihtiyaçlarını sağlamak için bir üniversite kampüsüne acil telefonlar yerleştirilmek isteniyor. [ Şekil 4.2' kampüs bir graf ile modellenmiştir.]

Bununla beraber kampüsün her ana sokağına en azından bir telefonla hizmet getirilmek isteniyor. Bu iş için **en az** kaç telefona ihtiyacımız vardır?



$G$  grafı

Şekil 4.2

### Problemin Matematiksel Modeli

$$x_j = \begin{cases} 1, & j.ci \text{ yere telefon konursa} \\ 0, & j.ci \text{ yere telefon konmazsa} \end{cases}$$

$$e_1 \text{ sokağı için, } x_1 + x_2 \geq 1$$

$$e_2 \text{ sokağı için, } x_2 + x_3 \geq 1$$

$$e_3 \text{ sokağı için, } x_4 + x_5 \geq 1$$

$$e_4 \text{ sokağı için, } x_7 + x_8 \geq 1$$

$$e_5 \text{ sokağı için, } x_6 + x_7 \geq 1$$

$$e_6 \text{ sokağı için, } x_2 + x_6 \geq 1$$

$$e_7 \text{ sokağı için, } x_1 + x_6 \geq 1$$

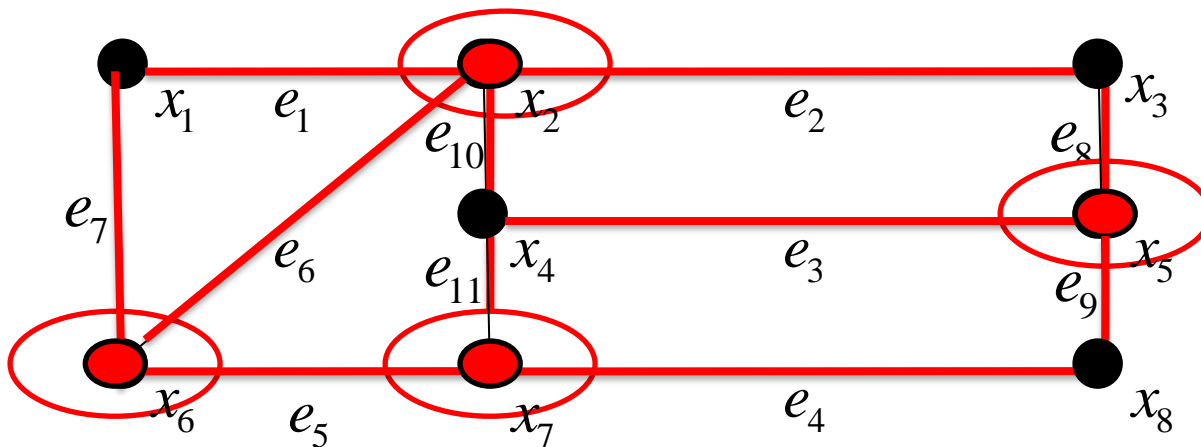
$$e_8 \text{ sokağı için, } x_3 + x_5 \geq 1$$

$$e_9 \text{ sokağı için, } x_5 + x_8 \geq 1$$

$$e_{10} \text{ sokağı için, } x_2 + x_4 \geq 1$$

$$e_{11} \text{ sokağı için, } x_4 + x_7 \geq 1$$

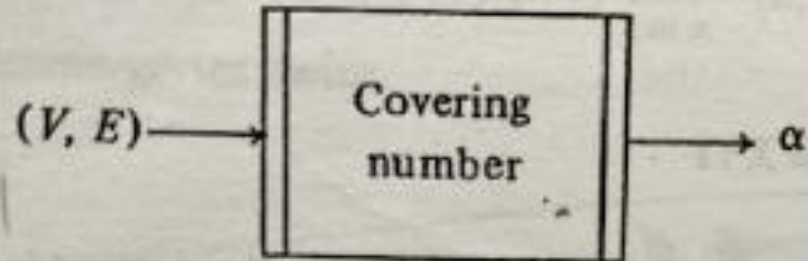
$$\text{k.a., } Z_{\min} = x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_7 + x_8$$



$G$  grafi

$$S = \{x_2, x_5, x_6, x_7\} \Rightarrow \alpha(G) = 4$$

## Örtü Sayısı bulmak için bir algoritma



$i, j$ : element of  $n^+$

$\alpha$ : nonnegative integer

$f$ : element of  $L(n) = (L(n), +, \cdot)$

begin  $f \leftarrow 1$ ;

for  $j \leftarrow 1$  to  $n - 1$  do

for  $i \leftarrow j + 1$  to  $n$  do

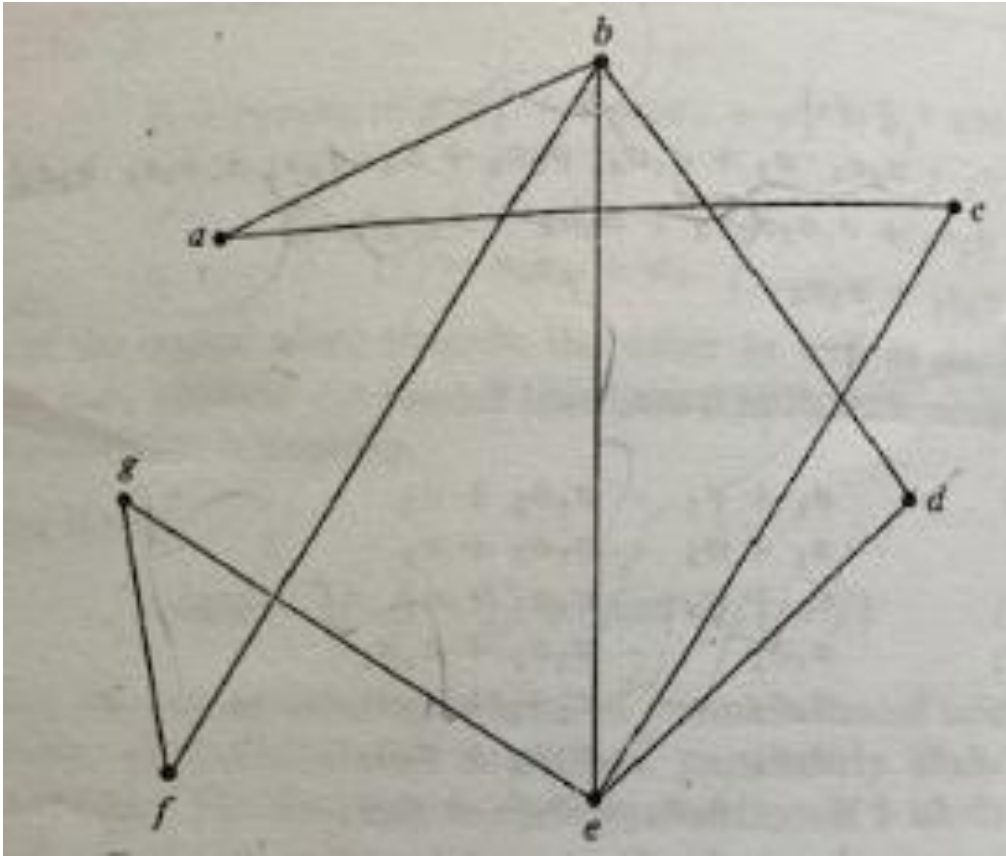
if  $v_i E v_j$  then

$f \leftarrow f \cdot (v_i + v_j)$ ;

$\alpha \leftarrow \min_{x \in f} \{|x|\}$

end

Örnek:



Şekilde verilen grafin örtü kümelerini ve örtü sayısını bulunuz.



$$\begin{aligned}
 f &= (a + b)(a + c)(b + d)(b + e)(b + f)(c + e)(d + e)(e + g)(f + g) \\
 &= (a + bc)(b + d) \dots \\
 &= (ab + bc + ad)(b + e) \dots \\
 &= (ab + bc + ade)(b + f) \dots \\
 &= (ab + bc + adef)(c + e) \dots \\
 &= (bc + abe + adef)(d + e) \dots \\
 &= (bcd + adef + bce + abe)(e + g) \dots \\
 &= (ade + bce + abe + bcdg)(f + g) \\
 &= adef + bcef + abef + bceg + abeg + bcdg
 \end{aligned}$$

Böylece örtü sayısı 4 bulunur.



$$\begin{aligned}
 f &= (a+b) \cdot (b+c) \cdot (b+d) \cdot (b+e) \cdot (e+f) \cdot (a+f) \\
 &= (\cancel{a} \cdot b + a \cdot \cancel{c} + \underline{b \cdot b} + \cancel{b \cdot c}) (b+d) (b+e) (e+f) (a+f) \\
 &= (b + a \cdot c) (b+d) (b+e) (e+f) (a+f) \\
 &= (\underline{b \cdot b} + \cancel{b \cdot d} + \cancel{a \cdot b \cdot c} + a \cdot c \cdot d) (b+e) (e+f) (a+f) \\
 &= (b + a \cdot c \cdot d) (b+e) (e+f) (a+f)
 \end{aligned}$$

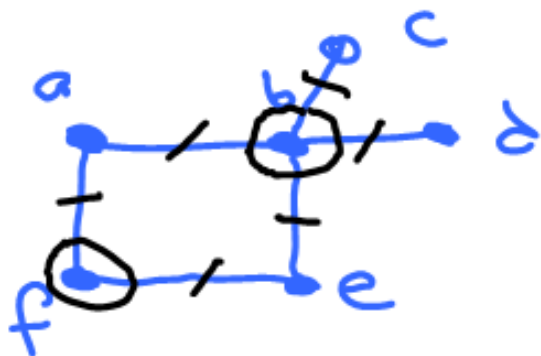
$$= (b + \cancel{be} + \cancel{acde} + acde)(e+f)(a+f)$$

$$= (b + acde) \cdot (e+f)(a+f)$$

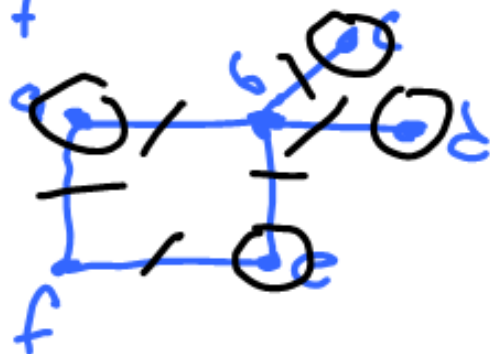
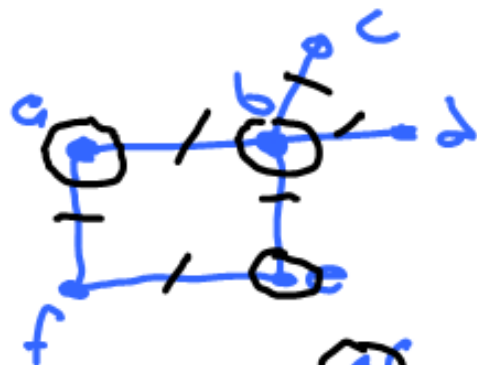
$$= (be + bf + acde + \cancel{acdef})(a+f)$$

$$= (abe + \cancel{abf} + acde + \cancel{acf} + \underline{bf} + \cancel{acdef})$$

$$= (bf + abe + acde) \rightarrow \underline{\text{örküs kümesi}}$$



$$\boxed{\alpha(G) = 2}$$



# Önemli Graflerin Ortü Sayları

$$= \left\lfloor \frac{n}{2} \right\rfloor$$

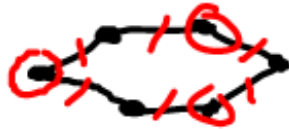
1) Yol graf

$$\alpha(P_n) = \begin{cases} \frac{n-1}{2}, & n \text{ tek} \\ \frac{n}{2}, & n \text{ çift} \end{cases}$$



2) Çevre graf

$$\alpha(C_n) = \begin{cases} \frac{n+1}{2}, & n \text{ tek} \\ \frac{n}{2}, & n \text{ çift} \end{cases} \rightarrow \left\lfloor \frac{n}{2} \right\rfloor$$



3) Yildiz graf

$$\alpha(k, m) = 1$$



4) Tam graf

$$\alpha(k_1) = n - 1$$



5) Tekerlek graf



$W_{1,5}$



$W_{1,6}$

$$\alpha(W_{1,n}) = \begin{cases} 1 + \frac{n+1}{2}, & n \text{ tek} \\ 1 + \frac{n}{2}, & n \text{ çift} \end{cases}$$

6) iki parçalı tam graf



$$\alpha(K_{m,n}) = \min\{m, n\}$$

**Teorem:**  $n$ - tepeli bir  $G$  grafi için  $\alpha(G) + \beta(G) = n$  dir.

İspat:  $G$  grafinin boyutlu sayısı

$\beta(G) = k$  olsun. Bu durumda öyle bir  $S \subset V(G)$  var ki  $|S| = k$  ve  $S$  'deki hiç bir tepe çifti arasında ayrılmaz.



$S$  'deki tepe çiftleri  
arasında ayrılmaz!!!

$$S' = G - S$$



1-)  $S'$  deki topolar arasında bağlantı olabilir.

2-)  $S$  ve  $S'$  arasında bağlantı olabilir.

Bunun anlamı; grafin her bir ağırlarının en az bir uç noktası  $S'$  de olmalıdır.

Böylece  $S'$  öntü kümesidir.

$S$  kümesi maksimum elemanlı e.b.d. den

$S'$  minimum elemanlı olur.

$|S| = k \rightarrow$  bağımsızlık sayısı

$$|S'| = n - k$$

$$|S| + |S'| = \beta(G) + \alpha(G) = k + n - k$$

$$\beta(G) + \alpha(G) = n$$

Bağımsızlık Sayısı ve Örtü Sayısı günlük hayatta nerelerde kullanılabilirler?

Araştırınız...

## KAYNAKLAR

- [1] Chartrand, G.-Lesniak, L., (1986) : *Graphs and Digraphs*, Wadsworth & Brooks, California
- [2] West D.B. (2001) : *Introduction to Graph Theory*, Prentice Hall, USA.
- [3] Graf Teoriye Giriş, Şerife Büyükköse ve Gülistan Kaya Gök, Nobel Yayıncılık
- [4] Discrete Mathematical Structures for Computer Science, Ronald E. Prather, Houghton Mifflin Company, (1976).
- [5] Christofides, N., 1986. Graph Theory an Algorithmic Approach, Academic Press, London
- [6] Algoritmalar (Teoriden Uygulamalara), Vasif V. NABİYEV, Seçkin Yayıncılık