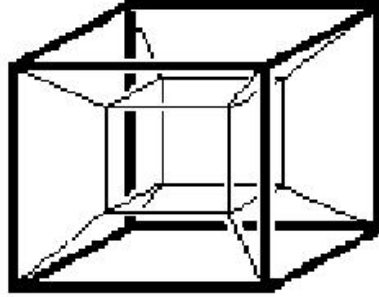
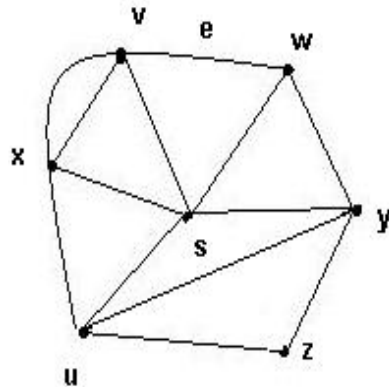


Graf Teorisi (Graph Theory)



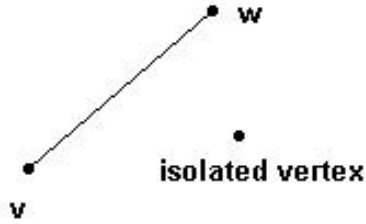
Giris

- G grafi nedir ?
- $G = (V, E)$
 - $V = V(G)$ = düğümler kümesi
 - $E = E(G)$ = kenarlar kümesi
- Örnek:
 - $V = \{s, u, v, w, x, y, z\}$
 - $E = \{(x,s), (x,v)_1, (x,v)_2, (x,u), (v,w), (s,v), (s,u), (s,w), (s,y), (w,y), (u,y), (u,z), (y,z)\}$



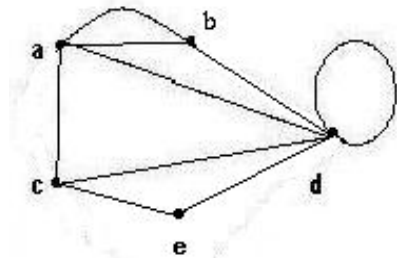
Kenarlar (Edges)

- Kenar bir çift düğüm ile etiketlenmiş olup $e = (v,w)$ şeklinde gösterilir.
- Ayrik düğüm (Isolated vertex) = a kenar bağlantısı olmayan düğümdür.



Özel Kenarlar

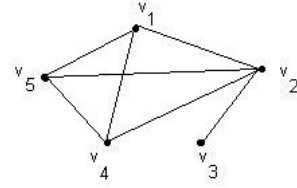
- Paralel kenarlar (Parallel edges)
 - İki veya daha fazla kenar bir düğüm çifti ile bağlanmıştır.
 - a ve b iki paralel kenar ile birleşmiştir
- Döngüler (Loops)
 - Kenarın başlangıç ve bitiş noktası aynı düğümdür.
 - d düğümü gibi.



Özel Graflar

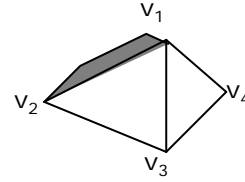
□ Basit (Simple) Graf

- Yönsüz, paralel kenar olmayan ve döngü içermeyen graflardir.



□ Çoklu (Multi) Graf

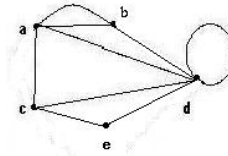
- Basit graflarin yeterli olmadigi durumlarda kullanilir.
- Yönsüz, paralel kenari olan ve döngü içermeyen graflardir.



Basit graflar, çoklu graftir fakat çoklu graflar basit graf degildir.

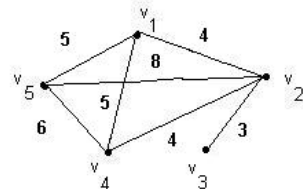
□ Pseudo Graflar

- Çoklu graflarin yeterli olmadigi durumlarda kullanilir.
- Yönsüz, Paralel kenari olan ve döngü içeren graflardir.
- Yönsüz graflarin en temel halidir.



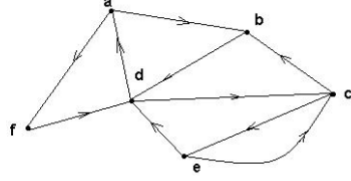
□ Agirlikli (Weighted) Graf

Her bir kenarina nümerik bir deger, agirlik verilmiş bir graftir.



Yönlü (Directed) Graflar (digraphs)

G, yönlü bir graf
(*directed*) veya *digraph*
ise her bir kenari sirali
bir düğüm çifti ile
iliskilendirilmiş ve her
kenari yönlüdür.



Tip	Kenar	Çoklu Kenara Izin ?	Döngüye Izin ?
Basit Graf	Yönsüz	Hayir	Hayir
Çoklu Graf	Yönsüz	Evet	Hayir
Pseudo Graf	Yönsüz	Evet	Evet
Yönlü Graf	Yönlü	Hayir	Evet
Yönlü Çoklu Graf	Yönlü	Evet	Evet

Graflarda Benzerlik (similarity) (1)

Problem: Nesnelerin degisik özellikleri referans alınarak nesnelere siniflandırabiliriz.

Örnek:

- Bilgisayar programlarında üç ayrı özelliğın olduğunu kabul edelim. $k = 1, 2, 3$ gibi:
- 1. Programın satır sayısı
- 2. Kullanılan "return" sayısı
- 3. Çağrılan fonksiyon sayısı

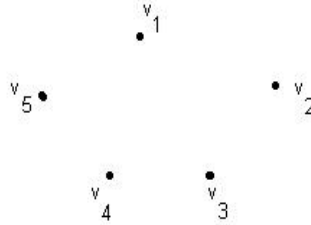
Graflarda benzerlik (2)

Asağıdaki tabloda 5 programın birbirleriyle karşılaştırıldığını farzedelim.

Program	# of lines	# of "return"	# of function calls
1	66	20	1
2	41	10	2
3	68	5	8
4	90	34	5
5	75	12	14

Graflarda benzerlik (3)

- G grafini asagidaki gibi olussun:
 - $V(G)$ programlardan olusan bir küme $\{v_1, v_2, v_3, v_4, v_5\}$.
 - Her düğüm, v_i bir üçlü ile gösterilir (p_1, p_2, p_3) ,
 - burada p_k özellik degerleridir $k = 1, 2, \text{ veya } 3$
 - $v_1 = (66, 20, 1)$
 - $v_2 = (41, 10, 2)$
 - $v_3 = (68, 5, 8)$
 - $v_4 = (90, 34, 5)$
 - $v_5 = (75, 12, 14)$



Benzer olmayan fonksiyonlar (1)

- Benzer olmayan (*dissimilarity function*) bir fonksiyon asagidaki gibi tanimlanir.
- Her bir düğüm çifti $v = (p_1, p_2, p_3)$ ve $w = (q_1, q_2, q_3)$ ile gösterilsin.

$$s(v, w) = \sum_{k=1}^3 |p_k - q_k|$$

- v ve w gibi iki programin *dissimilarity* $s(v, w)$ ile ölçülür.
- N seçilen sabit bir sayi olsun. Eger $s(v, w) < N$ ise v ve w arasindaki kenar eklenir. Sonra:
- Eger $v = w$ veya v ve w arasinda bir yol varsa v ve w nun ayni sinifta oldugunu söyleyebiliriz.

Benzer olmayan fonksiyonlar(2)

□ $N = 25$.

$$s(v_1, v_2) = 36$$

$$s(v_2, v_3) = 38$$

$$s(v_3, v_4) = 54$$

$$s(v_1, v_3) = 24$$

$$s(v_2, v_4) = 76$$

$$s(v_3, v_5) = 20$$

$$s(v_1, v_4) = 42$$

$$s(v_2, v_5) = 48$$

$$s(v_4, v_5) = 46$$

$$s(v_1, v_5) = 30$$

Benzer olmayan fonksiyonlar(2)

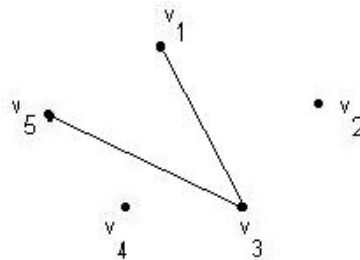
□ $N = 25$.

□ $s(v_1, v_3) = 24$, $s(v_3, v_5) = 20$
ve digerleri $s(v_i, v_j) > 25$

□ Üç sınıf vardır:

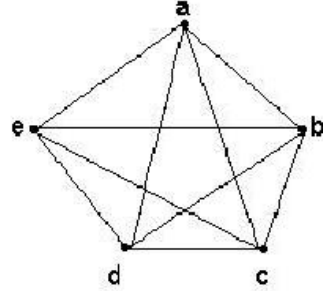
□ $\{v_1, v_3, v_5\}$, $\{v_2\}$ and $\{v_4\}$

□ **similarity graph** sekildeki gibidir.



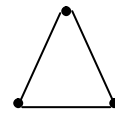
Tam (Complete) Graf K_n

- $n \geq 3$
- *complete graph* K_n : n adet düğüm içeren basit graf yapisindedir. Her düğüm, diğer düğümlere bir kenar ile bağlantılıdır.
- Şekilde K_5 grafi gösterilmiştir.
- Soru: K_3 , K_4 , K_6 graflarını çiziniz.



Cycles (Çember) Graf C_n

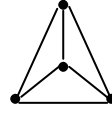
- $n \geq 3$
- *cycles graph* C_n : n adet düğüm ve $\{v_1, v_2\}, \{v_2, v_3\}, \dots, \{v_{n-1}, v_n\}, \{v_n, v_1\}$, düğüm çiftlerinden oluşan kenarlardan meydana gelir.
- Şekilde C_3 grafi gösterilmiştir.
- Soru: C_4 , C_5 , C_6 graflarını çiziniz.



C_3

Wheel (Tekerlek) Graf W_n

- *wheel graph* W_n : Cycle C_n grafına ek bir düğüm eklenerek oluşturulur. Eklenen yeni düğüm, diğer bütün düğümlere bağlıdır.
- Şekilde W_3 grafi gösterilmiştir.
- Soru: W_4, W_5, W_6 graflarını çiziniz.



W_3

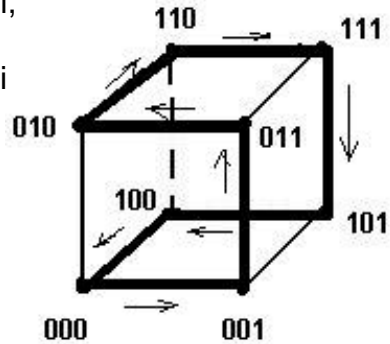
N-Cube (Küp) Graf Q_n

- *N-cube* Q_n : Grafın düğüm noktaları n uzunluğunda 2^n bit stringi ile gösterilir. Düğümlerin string değeri, bir düğümden diğerine geçerken aynı anda sadece bir bitin değerini değiştirmektedir.

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

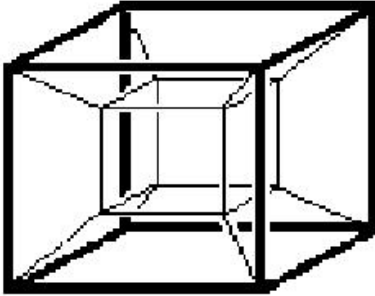
- Şekilde Q_3 grafi gösterilmiştir.

Soru: Q_1, Q_2 graflarını çiziniz.



The 3-cube

hypercube veya 4-cube



16 düğüm, 32 kenar ve 20 yüzey

□ Düğüm etiketleri:

0000 0001 0010 0011

0100 0101 0110 0111

1000 1001 1010 1011

1100 1101 1110 1111

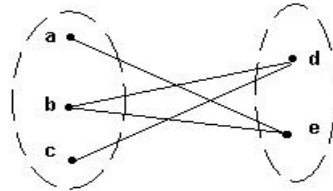
İki Parçalı (Bipartite) Graflar

□ G , bipartite graf ise:

■ $V(G) = V(G_1) \cup V(G_2)$

■ $|V(G_1)| = m, |V(G_2)| = n$

■ $V(G_1) \cap V(G_2) = \emptyset$



• Bir grafi oluşturan düğümleri iki ayrı kümeye bölerek grafi ikiye ayırabiliriz. Bu ayırma işleminde izlenecek yol; bir kenar ile birbirine bağlanabilecek durumda olan düğümleri aynı küme içerisine yerleştirmemektir.

• Mevcut küme içerisindeki düğümler birbirlerine herhangi bir kenar ile bağlanmamalıdır.

• K_3 Bipartite graf midir ?

Hayir

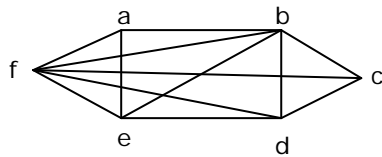
• C_6 Bipartite graf midir?

Evet

{1,3,5} ve {2,4,6}

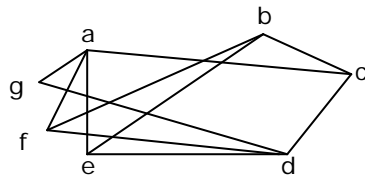
Yandaki graf Bipartite graf midir?

Hayir

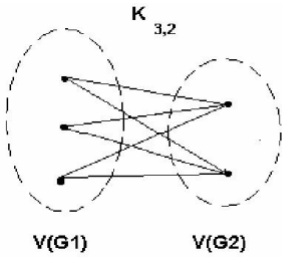


Yandaki graf Bipartite graf midir?

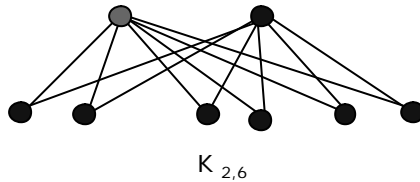
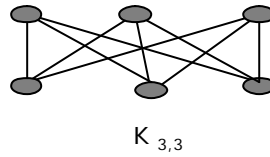
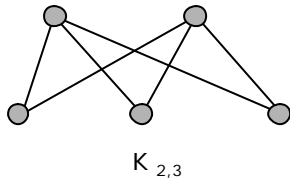
Evet. {a,b,d} ve {c,e,f,g}



Tam (complete) bipartite graph $K_{m,n}$



- *complete* bipartite graf $K_{m,n}$ şeklinde gösterilir. İlgili grafın düğümlerinin kümesi m ve n elemanlı iki alt kümeye ayrılır.
- Bir kenarı birbirine bağlayan iki düğümünde farklı alt kümelerin elemanı olmak zorundadırlar.
- $|V(G_1)| = m$
- $|V(G_2)| = n$



$K_n, C_n, W_n, K_{m,n}, Q_n$ graflarının kenar ve düğüm sayılarını formüle edecek olursak:

K_n n düğüm $n(n-1)/2$ kenar

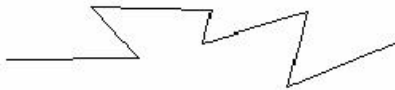
C_n n düğüm n kenar

W_n $n+1$ düğüm $2n$ kenar

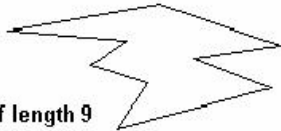
$K_{m,n}$ $m+n$ düğüm $m \cdot n$ kenar

Q_n 2^n düğüm $n2^{n-1}$ kenar

Yollar (Paths) ve Döngüler (Cycles)



Path of length 7



Cycle of length 9

- n uzunluğundaki bir yol'un (path) $n+1$ adet düğümü ve n adet de ardışık kenarı vardır
- Bir döngü içeren yol başladığı düğümde son bulur. Uzunluğu n olan bir döngüde n adet düğüm vardır.

Euler Döngüsü (Euler cycles)

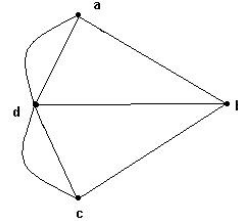
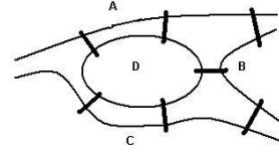
□ G grafi içerisindeki *Euler cycle* basit bir çevrim olup G grafi içerisindeki her kenardan sadece bir kez geçilmesine izin verir.

□ Königsberg köprü problemi:

□ Balangıç ve Bitiş noktası aynıdır, yedi köprüden sadece bir kez geçerek başlangıç noktasına dönmek mümkün müdür?

□ Bu problemi grafa indirgeyelim.

□ Kenarlar köprüleri ve düğüm noktaları da bölgeleri gösterebilir.

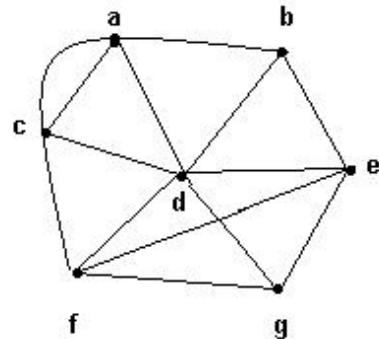


Bir düğümün derecesi

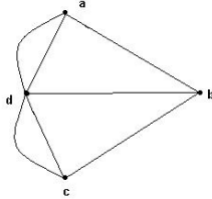
□ v düğümünün derecesi $\delta(v)$ ile gösterilir ve bu da yönsüz bir grafta düğüme gelen kenarlar toplamıdır. Düğüm noktalarındaki döngü düğüm derecesine 2 kez katılır.

□ Örnek:

- $\delta(a) = 4$, $\delta(b) = 3$,
- $\delta(c) = 4$, $\delta(d) = 6$,
- $\delta(e) = 4$, $\delta(f) = 4$,
- $\delta(g) = 3$.



Euler Grafi



- Bir **G** grafi *Euler cycle*'na sahip ise *Euler Grafi* adini alır.
- Euler grafında tüm düğümlerin derecesi çifttir.
- Königsberg bridge problemi bir Euler grafi değildir.
- Königsberg bridge probleminin çözümü yoktur.

Grafın düğüm derecelerinin toplamı

- Sifir dereceli bir düğüm *isolated* olarak adlandırılır. Isolated olan bir düğümden, baska bir düğüme yol yoktur.
- Düğüm derecesi bir olan düğüme *pendant* denir.
- **Teorem: Handshaking**
e adet kenarlı ve **n** adet düğümlü bir grafın $G(V,E)$ düğümlerinin toplamı kenar sayısının iki katıdır.

$$\sum_{i=1}^n \delta(v_i) = 2e$$

Örnek: Her birinin derecesi 6 olan 10 düğümlü bir grafın kaç tane kenarı vardır.

$$e=30$$

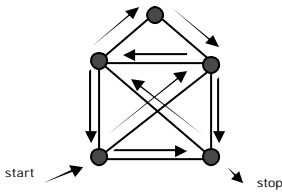
□ G grafında (v,w) yönlü bir kenar olsun ve yön v 'den w 'ya verilsin. v **initial vertex**, w 'da **terminal** veya **end vertex** olarak adlandırılır. Bir düğüm noktasında döngü söz konusu ise bu düğümün **initial vertex**i ve **end vertex**i birbirinin aynıdır.

□ Yönlü bir grafta, herhangi bir düğümün **in_degree**'si $\delta^-(v)$, **out_degree**'si $\delta^+(v)$ olarak gösterilir.

□ Yönlü bir grafın in_degree ve out_degree'lerinin toplamı birbirinin aynıdır.

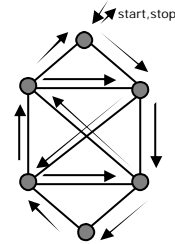
$$\sum_{v \in V} \delta^-(v) = \sum_{w \in V} \delta^+(w)$$

Örnek: Asagıda verilmiş olan graflardan hangilerinde her kenardan en az bir kez geçirilerek graf gezilmiştir, hangileri Euler grafidir, eğer değilse sebebi nedir ?

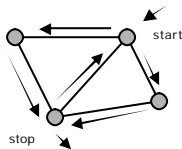


Path var, Euler grafi değil

Düğüm dereceleri çift değil

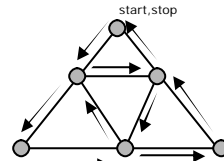


Euler grafi



Path var, Euler grafi değil

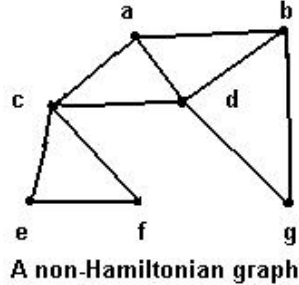
Düğüm dereceleri çift değil



Euler grafi

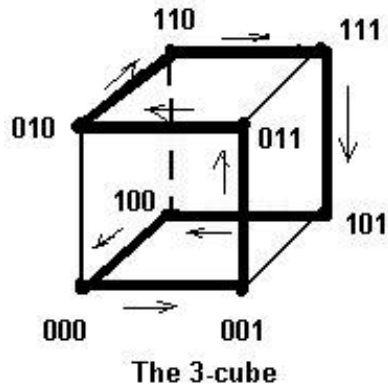
Hamilton Döngüsü (Hamiltonian Cycles)

- G grafinin üzerindeki her düğümden yalnız bir kez geçmek şartı ile kapalı bir yol oluşturabilen graflardır (*Traveling salesperson*)
- Bu kapalı yol *Hamiltonian cycle* olarak adlandırılır.
- *Hamiltonian cycle* sahip bir G grafi *Hamiltonian* graf olarak adlandırılır.



3-cube

Hamiltonian cycle
(000, 001, 011, 010,
110, 111, 101, 100,
000) örnek bir graf
3-cube olarak
verilebilir.



Graf Modelleri

Farklı alanlarda farklı graf modelleri kullanılır.

Niche Overlap Graf : Eko sistem içerisindeki farklı grupları modellemede kullanılır.

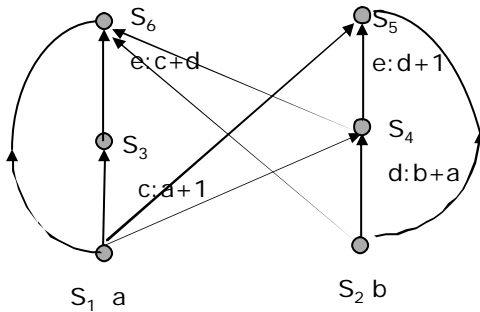
Influence Graf: Grup çalışmalarında, grup içerisindeki kişilerin birbirlerini etkilemesini modellemede kullanılır.

Round-Robin Tournament Graf: Turnuvada yer alan her takımın, hangi takımla karşılaştığını ve oyunu kimin kazandığını göstermede kullanılır.

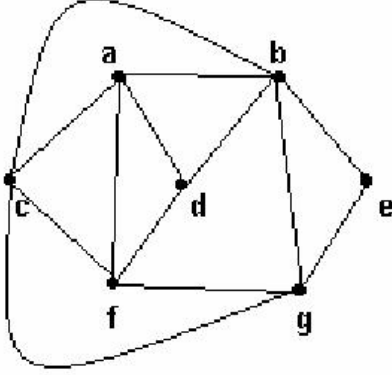
Precedence Graf: Bir işlemin sonucu, kendisinden önce gelen işlemin sonucuna bağlı olarak değişen sistemleri modellemede kullanılır.

Precedence grafa örnek...

S_1 a:0 S_2 b:1 S_3 c:a+1 S_4 d:b+a S_5 e:d+1 S_6 e:c+d



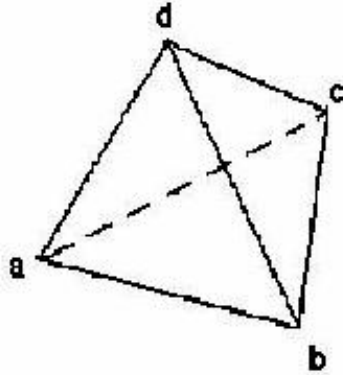
Planar Graflar



Bir G grafinin kenarları birbirlerini kesmeyecek şekilde çizilebiliyorsa **Planar** graf olarak adlandırılır.

Euler'in formülü

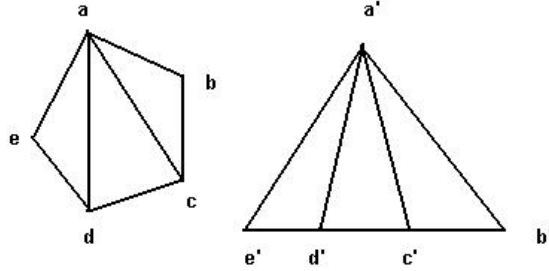
- Eger G bir *planar* graph is
 - v = düğüm sayısı
 - e = kenar sayısı
 - f = yüzey sayısı
- Öyleyse $v - e + f = 2$



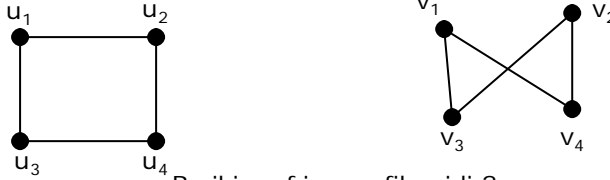
Izomorfik (Isomorphic) Graflar

İki grafın izomorfik olup olmadığı nasıl kontrol edilir ?

- Kenar sayıları aynı olmalıdır.
- Düğüm sayıları aynı olmalıdır.
- Düğüm dereceleri aynı olmalıdır.
- Dğümler arasındaki ilişkiyi gösteren matrisler aynı olmalıdır. Bu matrislerdeki benzerlik satır ve sütunlardaki yer değişikliği ile de sağlanabilir.



Örnek



Bu iki graf izomorfik midir?

EYET

Her iki grafında 4 düğümü, 4 kenarı ve her düğümünün de derecesi 2

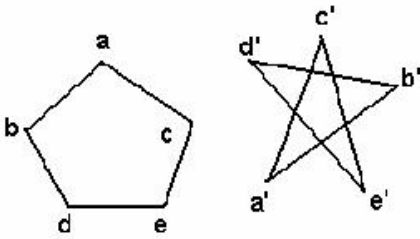
dir

	u1	u2	u3	u4	v1	v2	v3	v4	
u1	0	1	1	0	v1	0	0	1	1
u2	1	0	0	1	v2	0	0	1	1
u3	1	0	0	1	v3	1	1	0	0
u4	0	1	1	0	v4	1	1	0	0

u₂ ve u₄ satır ve sütunlar yerdegistiriyor

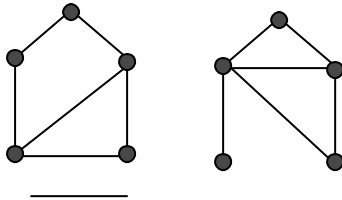
Örnek

□ Asagida verilmis olan iki graf izomorfik midir? EVET



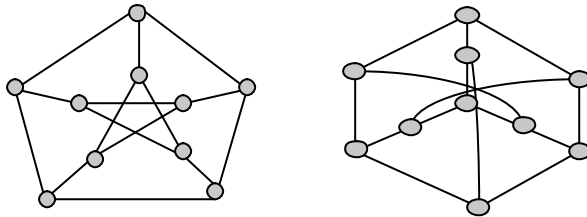
	a	b	c	d	e
a	0	1	1	0	0
b	1	0	0	1	0
c	1	0	0	0	1
d	0	1	0	0	1
e	0	0	1	1	0

Örnek



Bu iki graf izomorfik midir ?

HAYIR



Bu iki graf izomorfik midir ?

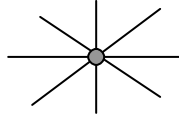
EVET

Özel Tip Graflar

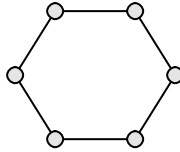
□ Özel tip graflar genellikle veri iletişimi ve paralel veri işleme uygulamalarında kullanılır.

Local Area Network : Bir bina içerisindeki midi ve pc gibi farklı bilgisayarları ve çevre birimlerini birbirine bağlamak için kullanılır. Bu ağların farklı topolojileri mevcuttur.

« **Star Topology** : Bütün cihazlar, merkezdeki cihaz üzerinden birbirlerine bağlanırlar. $K_{1,n}$ complete Bipartite Graf kullanılır.



« **Ring Topology** : Bu modelde her cihaz diğer iki farklı cihaz ile birbirine bağlıdır. n-cycles C_n modelidir.



« **Hybrid Topology** : Star ve Ring topologiesini birlikte kullanır. Bu tekrarlılık network'ün daha güvenli olmasını sağlar. Wheel, W_n graf modeline karşılık gelir.

