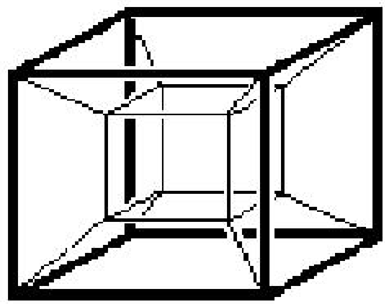
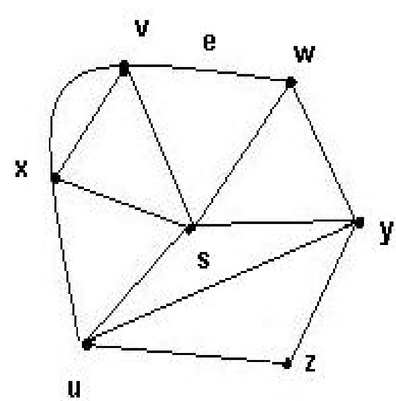
Graf Teorisi
(Graph Theory)



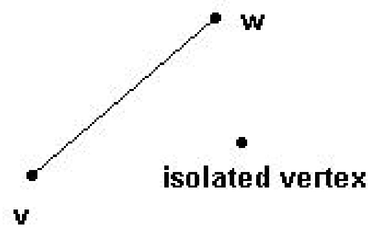
Giris

- G grafi nedir ?
- G = (V, E)
  - V = V(G) = dügümler 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)}



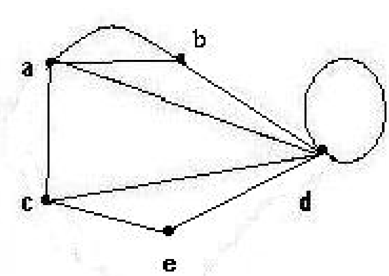
Kenarlar (Edges)

- Kenar bir çift dügüm ile etiketlenmis olup e = (v,w) seklinde gösterilir.
- Ayrik dügüm (Isolated vertex) = a kenar baglantisi olmayan dügümdür.



Özel Kenarlar

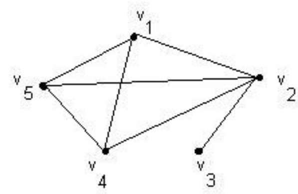
- Paralel kenarlar(Parallel edges)
  - Iki veya daha fazla kenar bir dügüm çifti ile baglanmistir.
    - a ve b iki paralel kenar ile birlesmistir
- Döngüler (Loops)
  - Kenarin baslangiç ve bitis noktasi ayni dügümdür.
    - d dügümü gibi.



Özel Graflar

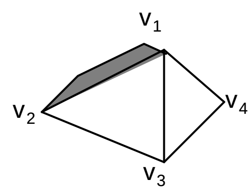
Basit (Simple) Graf

- Yönsüz, paralel kenar olmayan ve döngü içermeyen graflardir.



Çoklu (Multi) Graf

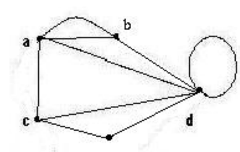
- Basit graflarin yeterli olmadigi durumlarda kullanilir.
- Yönsüz, paralel kenari olan ve döngü içermeyen graflardir.



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

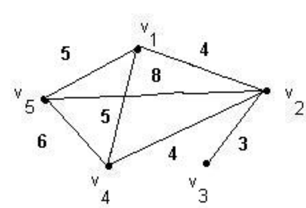
Pseudo Graflar

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



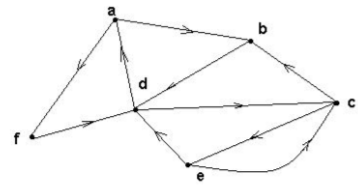
Agirlikli (Weighted) Graf

Her bir kenarina nümerik bir deger, agirlik verilmis bir grafdir.



Yönlü (Directed) Graflar (digraphs)

G, yönlü bir graf (*directed) veya digraph* ise her bir kenari sirali bir dügüm çifti ile iliskilendirilmis 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 alinarak nesneleri siniflandirabiliriz.

Örnek:

- Bilgisayar programlarinda üç ayri özelligin oldugunu kabul edelim. k = 1, 2, 3 gibi:
- 1. Programin satir sayisi
- 2. Kullanilan "return" sayisi
- 3. Çagrilan fonksiyon sayisi

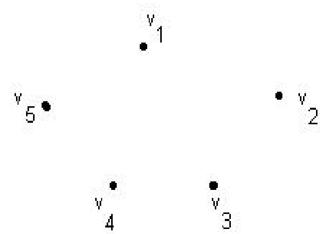
Graflarda benzerlik (2)

Asagidaki tabloda 5 programin birbirleriyle karsilastirildigini 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 dügüm, v_i bir üçlü ile gösterilir (p_1, p_2, p_3),
  - burada p_k özellik degerleridir k = 1, 2, 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 dügüm 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.

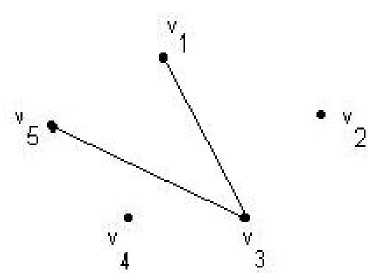
Benzer olmayan fonksiyonlar(2)

- N = 25.

$s(v_1,v_2) = 36$ $s(v_2,v_3) = 38$ $s(v_3,v_4) = 54$
$s(v_1,v_3) = 24$ $s(v_2,v_4) = 76$ $s(v_3,v_5) = 20$
$s(v_1,v_4) = 42$ $s(v_2,v_5) = 48$ $s(v_4,v_5) = 46$
$s(v_1,v_5) = 30$

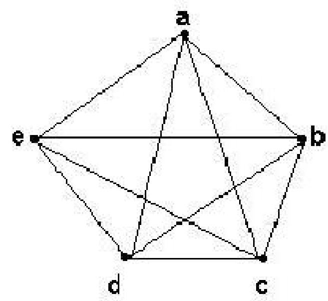
Benzer olmayan fonksiyonlar(2)

- N = 25.
- $s(v_1,v_3) = 24$, $s(v_3,v_5) = 20$ ve digerleri $s(v_i,v_j) > 25$
- Üç sinif vardir:
- $\{v_1,v_3, v_5\}$, $\{v_2\}$ and $\{v_4\}$
- ***similarity graph*** sekildeki gibidir.



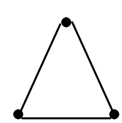
Tam (Complete) Graf K_n

- $n \geq 3$
- *complete graph K_n* : n adet dügüm içeren basit graf yapisindadir. Her dügüm, diger dügümlere bir kenar ile baglantilidir.
- Sekilde K_5 grafi gösterilmistir.
- Soru: K_3, K_4, K_6 graflarini çiziniz.



Cycles (Çember) Graf C_n

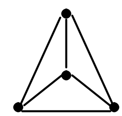
- $n \geq 3$
- *cycles graph C_n* : n adet dügüm ve $\{v_1,v_2\}$, $\{v_2,v_3\}$, …, $\{v_{n-1},v_n\}$, $\{v_n,v_1\}$, dügüm çiftlerinden olusan kenarlardan meydana gelir.
- Sekilde C_3 grafi gösterilmistir.
- Soru: C_4, C_5, C_6 graflarini çiziniz.



C_3

Wheel (Tekerlek) Graf W $_n$

- *wheel graph W_n* : Cycle C_n grafina ek bir dügüm eklenerek olusturulur. Eklenen yeni dügüm, diger bütün dügümlere baglidir.
- Sekilde W_3 grafi gösterilmistir.



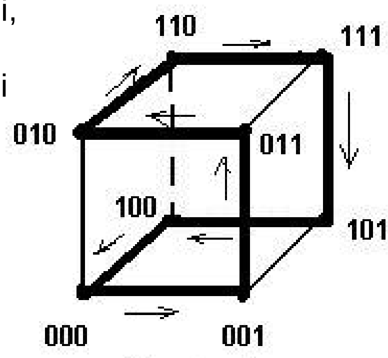
W_3

- Soru: W_4, W_5, W_6 graflarini çiziniz.

N-Cube (Küp) Graf Q $_n$

- *N-cube Q_n* : Grafin dügüm noktalari n uzunlugunda 2^n bit stringi ile gösterilir. Dügümlerin string degeri, bir dügümden digerine geçerken ayni anda sadece bir bitin degerini degistirmektedir.

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

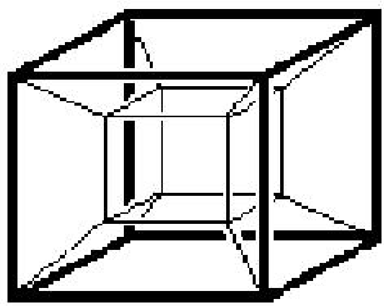


The 3-cube

- Sekilde Q_3 grafi gösterilmistir.

Soru: Q_1, Q_2 graflarini çiziniz.

hypercube veya 4-cube



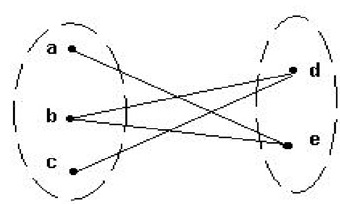
16 dügüm, 32 kenar ve 20 yüzey

- Dügüm etiketleri:

0000 0001 0010 0011
0100 0101 0110 0111
1000 1001 1010 1011
1100 1101 1110 1111

Iki Parçali (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 olusturan dügümleri iki ayri kümeye bölerek grafi ikiye ayirabiliriz. Bu ayirma isleminde izlenecek yol; bir kenar ile birbirine baglanabilecek durumda olan dügümleri ayni küme içerisine yerlestirmemektir.

• Mevcut küme içerisindeki dügümler birbirlerine herhangi bir kenar ile baglanmamalidir.

- K_3 Bipartite graf midir ?

Hayir

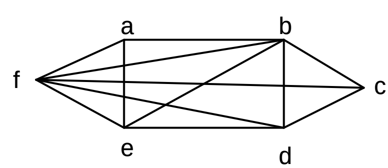
- C_6 Bipartite graf midir?

Evet

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

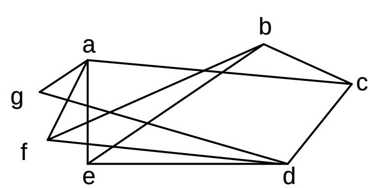
Yandaki graf Bipartite graf midir?

Hayir

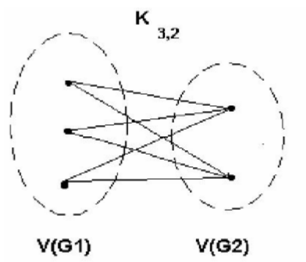


Yandaki graf Bipartite graf midir?

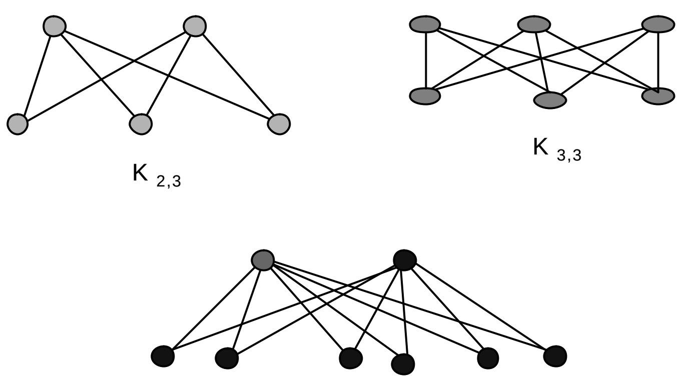
Evet. {a,b,d} ve {c,e,f,g}



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



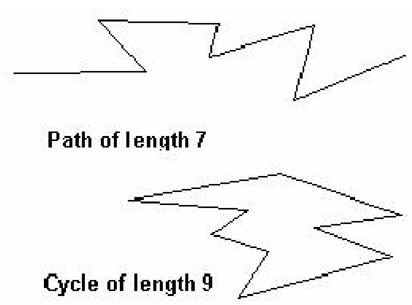
- *complete* bipartite graf $K_{m,n}$ seklinde gösterilir. Ilgili grafin dügümlerinin kümesi m ve n elemanli iki alt kümeye ayrilir.
- Bir kenari birbirine baglayan iki dügümünde farkli alt kümelerin elemani olmak zorundadirlar.
- $|V(G_1)| = m$
- $|V(G_2)| = n$



K_n, C_n, W_n, $K_{m,n}$, Q_n graflarinin kenar ve dügüm sayilarini formüle edecek olursak:

K_n	n dügüm	n(n-1)/2 kenar
C_n	n dügüm	n kenar
W_n	n+1 dügüm	2n kenar
$K_{m,n}$	m+n dügüm	m*n kenar
Q_n	2^n dügüm	$n2^{n-1}$ kenar

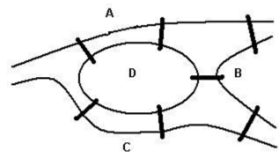
Yollar (Paths) ve Döngüler(Cycles)

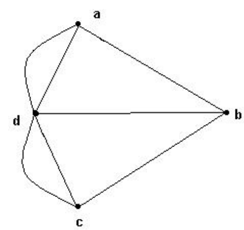


- **n** uzunlugundaki bir yol'un (path) **n+1** adet dügümü ve n adet de ardisik kenari vardir
- Bir döngü içeren yol basladigi dügümde son bulur. Uzunlugu **n** olan bir döngüde **n** adet dügüm vardir.

Euler Döngüsü (Euler cycles)

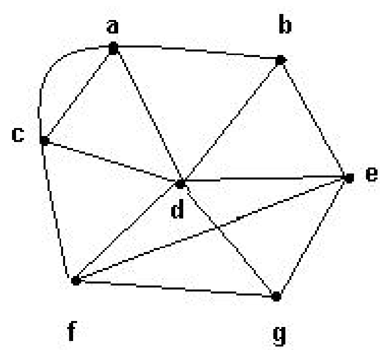
- G grafi içerisindeki *Euler cycle* basit bir çevrim olup G grafi içerisindeki her kenardan sadece bir kez geçilmesine izin verir.
- Königsberg köprü problemi:
  - Balangiç ve Bitis noktasi ayndir, yedi köprüden sadece bir kez geçerek baslangiç noktasina dönmek mümkün müdür?
- Bu problemi grafa indirgeyelim.
- Kenarlar köprüleri ve dügüm noktalarida bölgeleri göstersin.



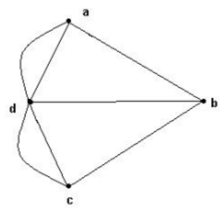


Bir dügümün derecesi

- **v** dügümünün derecesi $\delta(v)$ ile gösterilir ve bu da yönsüz bir grafta dügüme gelen kenarlar toplamidir. Dügüm noktalarindaki döngü dügüm derecesine 2 kez katilir.
- **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'*ına sahip ise *Euler Grafi* adini alir.
- Euler grafinda tüm dügümlerin derecesi çifttir.
- Konigsberg bridge problemi bir Euler grafi degildir.
- Konigsberg bridge probleminin çözümü yoktur.

Grafin dügüm derecelerinin toplami

- Sifir dereceli bir dügüm ***isolated*** olarak adlandirilir. Isolated olan bir dügümden, baska bir dügüme yol yoktur.
- Dügüm derecesi bir olan dügüme ***pendant*** denir.

- **Teorem**: *Handshaking*

e adet kenarli ve ***n*** adet dügümlü bir grafin G(V,E) dügümlerinin toplami kenar sayisinin iki katidir.

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

Örnek: Her birinin derecesi 6 olan 10 dügümlü bir grafin kaç tane kenari vardir.

e=30

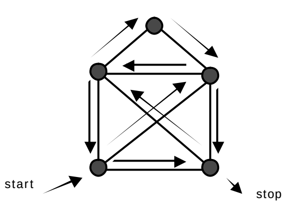
❑ G grafinda (v,w) yönlü bir kenar olsun ve yön *v*'den *w*'ya verilsin. *v* ***initial vertex***, *w*'da ***terminal*** veya ***end vertex*** olarak adlandirilir. Bir dügüm noktasinda döngü söz konusu ise bu dügümün *initial vertex*'i ve *end vertex*'i birbirinin aynidir.

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

❑ Yönlü bir grafin in_degree ve out_degree'lerinin toplami birbirinin aynidir.

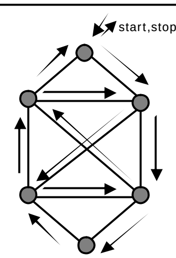
$$\sum_{v \in V} \delta^-(v) = \sum_{w \in V} \delta^+(v)$$

Örnek: Asagida verilmis olan graflardan hangilerinde her kenardan en az bir kez geçirilerek graf gezilmistir, hangileri Euler grafidir, eger degilse sebebi nedir ?

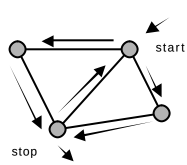


Path var, Euler grafi degil

Dügüm dereceleri çift degil

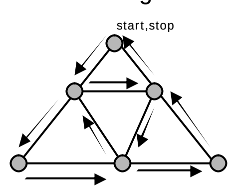


Euler grafi



Path var, Euler grafi degil

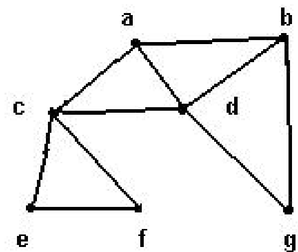
Dügüm dereceleri çift degil



Euler grafi

Hamilton Döngüsü (Hamiltonian Cycles)

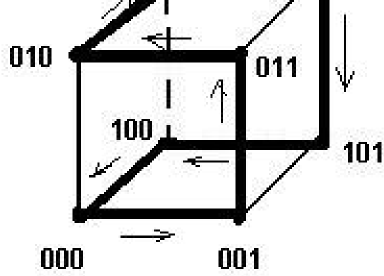
- G grafinin üzerindeki her dügümden yanliz bir kez geçmek sarti ile kapali bir yol olusturabilen graflardir (*Traveling salesperson)*
- Bu kapali yol *Hamiltonian cycle* olarak adlandirilir.
- *Hamiltonian cycle* sahip bir G grafi *Hamiltonian* graf olarak adlandirilir.



A non-Hamiltonian graph

3-cube

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



The 3-cube

Graf Modelleri

Farkli alanlarda farkli graf modelleri kullanilir.

Niche Overlap Graf : Eko sistem içerisindeki farkli grublari modellemede kullanilir.

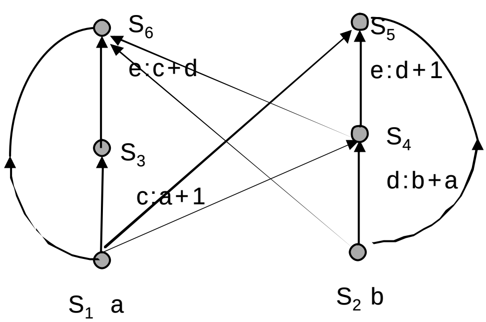
Influence Graf: Grup çalismalarinda, grup içerisindeki kisilerin birbirlerini etkilemesini modellemede kullanilir.

Round-Robin Tournament Graf: Turnuvada yer alan her takimin, hangi takimla karsilastigini ve oyunu kimin kazandigini göstermede kullanilir.

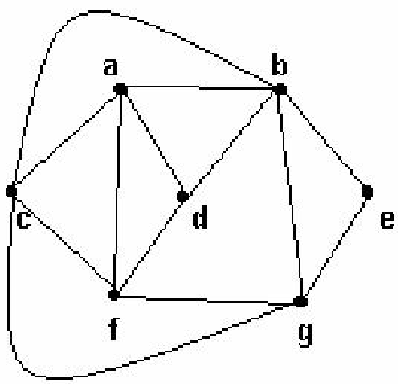
Precedence Graf: Bir islemin sonucu, kendisinden önce gelen islemin sonucuna bagli olarak degisen sistemleri modellemede kullanilir.

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**



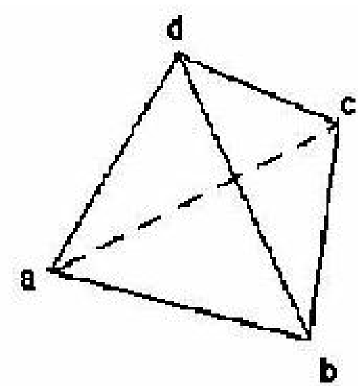
Planar Graflar



Bir G grafinin kenarlari birbirlerini kesmeyecek sekilde çizilebiliyorsa ***Planar*** graf olarak adlandirilir.

Euler'in formülü

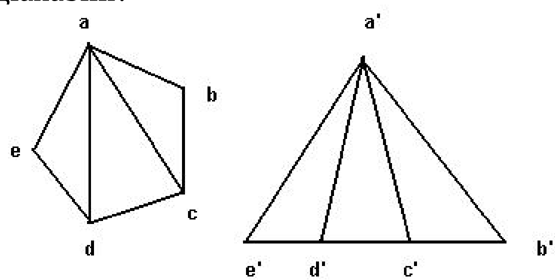
- Eger G bir *planar* graph is
  - v = dügüm sayisi
  - e = kenar sayisi
  - f = yüzey sayisi
- Öyleyse v – e + f = 2



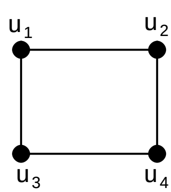
Izomorfik (Isomorphic) Graflar

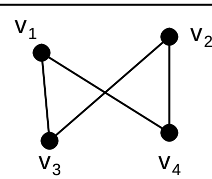
Iki grafin izomorfik olup olmadigi nasil kontrol edilir ?

- Kenar sayilari ayni olmalidir.
- Dügüm sayilari ayni olmalidir.
- Dügüm dereceleri ayni olmalidir.
- Dügümler arasindaki iliskiyi gösteren matrisler ayni olmalidir. Bu matrislerdeki benzerlik satir ve sütunlardaki yer degisikligi ile de saglanabilir.



Örnek





Bu iki graf izomorfik midir?

EVET

Her iki grafinda 4 dügümü, 4 kenari ve her dügümünün de derecesi 2 dir

	u1	u2	u3	u4
u1	0	1	1	0
u2	1	0	0	1
u3	1	0	0	1
u4	0	1	1	0

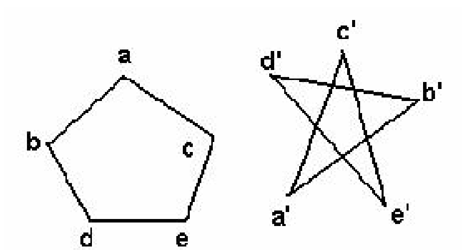
	v1	v2	v3	v4
v1	0	0	1	1
v2	0	0	1	1
v3	1	1	0	0
v4	1	1	0	0

u_2 ve u_4 satir ve sütunlar yerdegistiriyor

Örnek

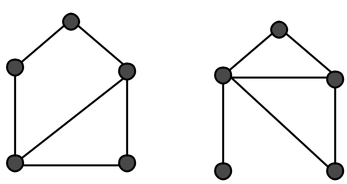
- Asagida verilmis olan iki graf izomorfik midir?

EVET



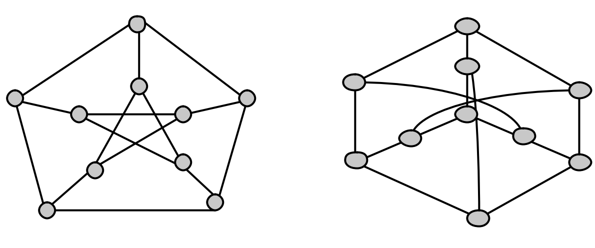
	a	b	c	d	e
a	0	1	1	0	0
b	1	0	0	1	0
c	1	0	0	0	1
d	0	1	0	0	1
e	0	0	1	1	0

Örnek



Bu iki graf izomorfik midir ?

HAYIR



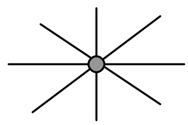
Bu iki graf izomorfik midir ? EVET

Özel Tip Graflar

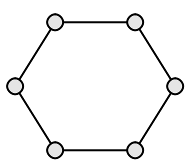
- Özel tip graflar genellikle veri iletisimi ve paralel veri isleme uygulamalarinda kullanilir.

Local Area Network : Bir bina içerisindeki midi ve pc gibi farkli bilgisayarlari ve çevrebirimlerini birbirine baglamak için kullanilir. Bu aglarin farkli topolojileri mevcuttur.

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



« **Ring Topology :** Bu modelde her cihaz diger iki farkli cihaz ile birbirine baglidir. n-cycles C_n modelidir.



« **Hybrid Topology :** Star ve Ring topology'sini birlikte kullanir. Bu tekrarlilik network'ün daha güvenli olmasini saglar. Whell, W_n graf modeline karsilik gelir.