— Bilgisayar Ağlarının Programlanması —

Genetik Algoritmalar

Parçacık Sürü Optimizasyonu Algoritmaları

Yapay Arı Kolonisi Algoritması

Karınca Kolonisi Algoritması

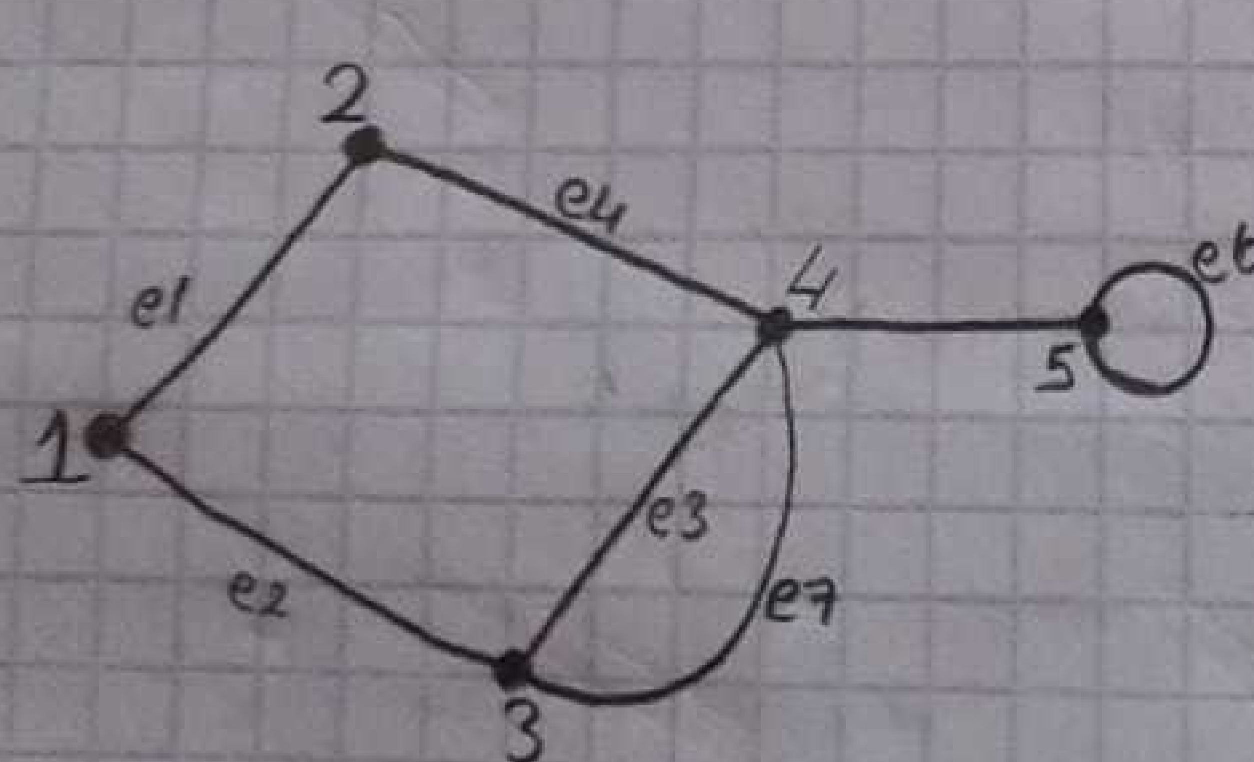
— BÖLÜM 1 —

Graf Teorisi: Elemanların nokta olarak adlandırılan sonlu sayıda (boş olmayan) noktalar kümesi ve elemanlarını kenar olarak adlandırılan sonlu sayıda kenarlar kümesinden oluşan ikili yapıya GRAF adı verilir. Graflar aşağıdaki gibi gösterilir.

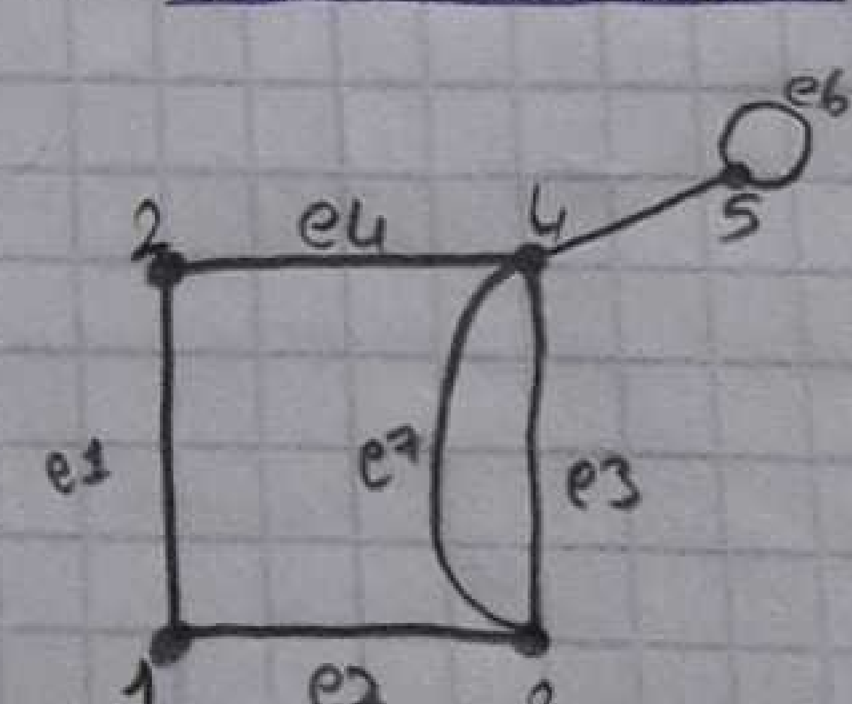
$G = (V, E)$

graf — nokta — kenar

Burada $E = \{\{i, j\} : i, j \in V\}$ şeklinde tanımlanır. Bir graf her zaman içerdiği noktalar ve bu noktaları birbirine bağlayan kenarlar yardımıyla çizilebilir.



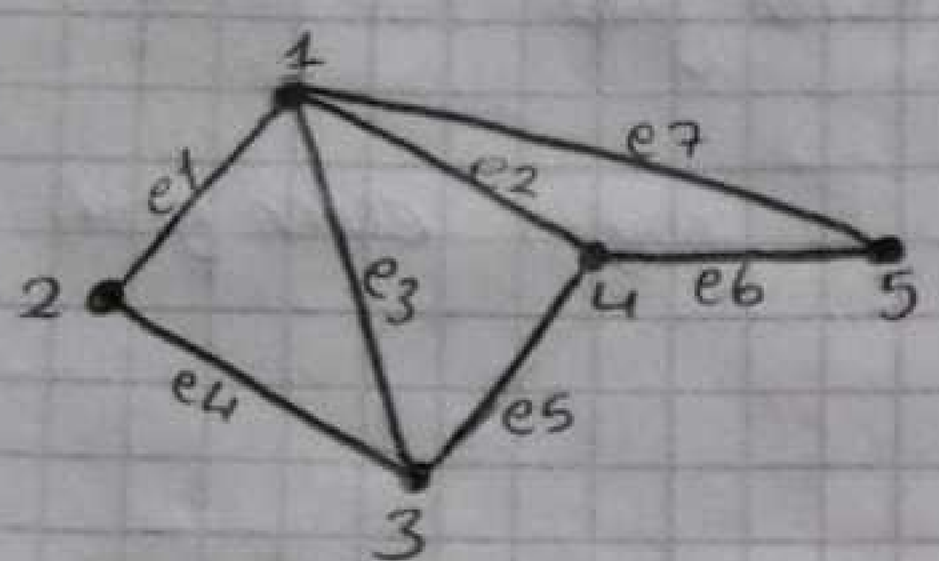
Farklı Gösterimi



$V = \{1, 2, 3, 4, 5\}$

$E = (e1, e2, e3, e4, e6, e7)$

Bir graf üzerindeki i ve j noktaları arasında en az 1 adet Kenar bulunuyorsa i ve j noktaları birbirine komsudur $i \sim j$ seklinde gösterilir Aşağıdaki grafta herbir noktaya göre komsulukları gösterebiliriz.



1 numaralı düğüm

$1 \sim 2$, $1 \sim 3$, $1 \sim 4$, $1 \sim 5$

$N_1 = \{2, 3, 4, 5\}$

2 numaralı düğüm

$2 \sim 1$, $2 \sim 3$

$N_2 = \{1, 3\}$

3 numaralı düğüm

$3 \sim 1$, $3 \sim 2$, $3 \sim 4$

$N_3 = \{1, 2, 4\}$

4 numaralı düğüm

$4 \sim 1$, $4 \sim 3$, $4 \sim 5$

$N_4 = \{1, 3, 5\}$

5 numaralı düğüm

$5 \sim 1$, $5 \sim 4$

$N_5 = \{1, 4\}$

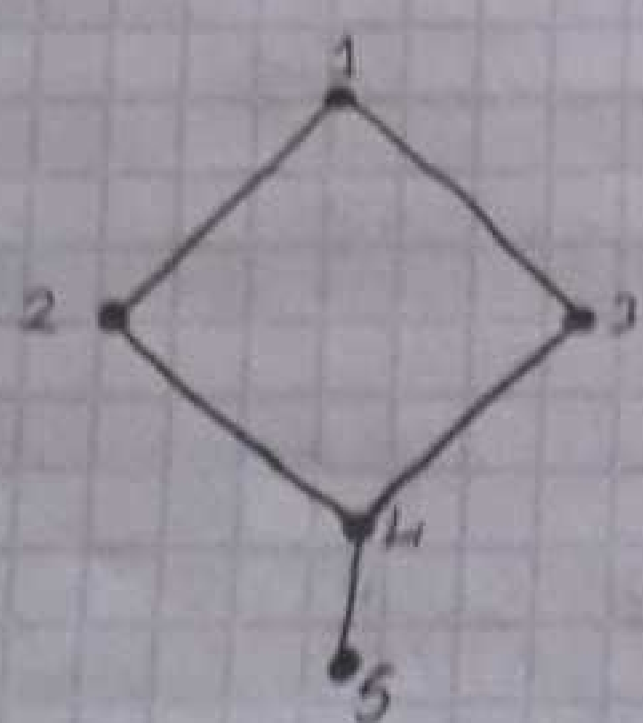
Bölüm 2

Euler Turları ve Hamilton Döngüsü

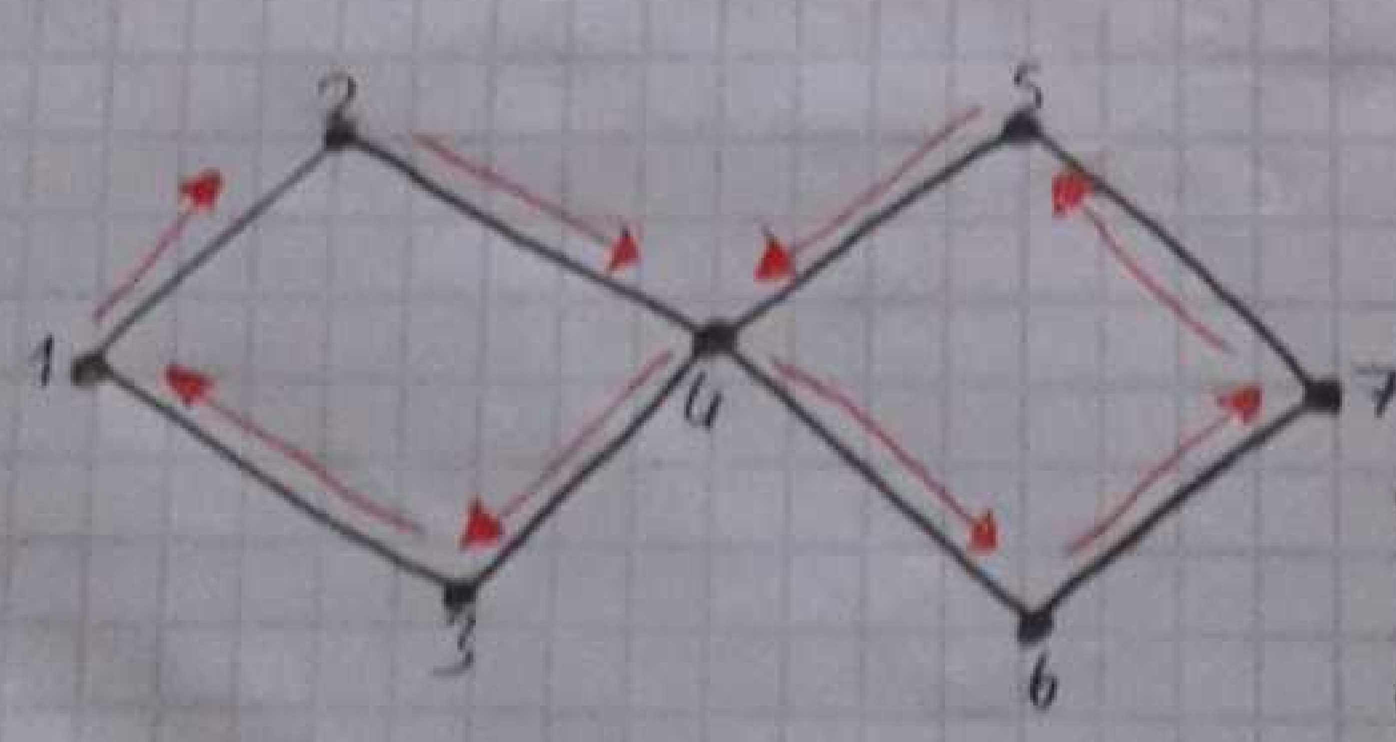
Her kenarın tam olarak bir kez kullanıldığı ve başlangıç noktasına geri dönülen yola euler turu veya euler yolu adı verilir.

Bir graft euler turu içeriyorsa bu grafta euler grafı adı verilir.

Aşağıdaki örnek grafta Euler yolu içerir fakat euler turu içermez. Bir grafın euler turu olması için her kenarın üzerinden bir kere geçilmelidir. Bu örnekte 4 ve 5 numaralı düğümler arasında kalan kenar üzerinden iki kere geçilmektedir bu nedenle bu graf bir euler turu içermez.

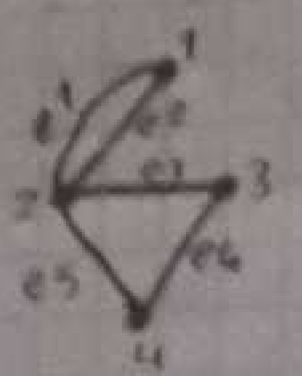


5,4,2,1,3,4
4,2,1,3,4,5 seçenekleri birer euler yoludur fakat euler turu değildir (çünkü bir yoldan 2 kez geçilmez)



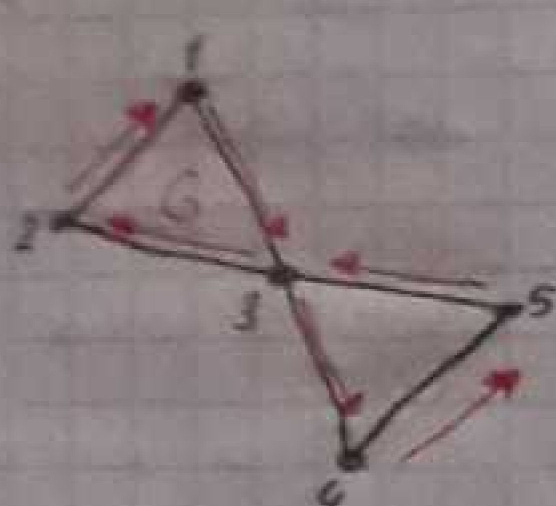
Yandaki graf (1,2,4,6,7,5,4,3,1) hem euler turu hem de euler yolu içerir.

Örnek: Aşağıdaki şekilde verilen graf için bir noktasından başlayıp 4 noktasına biten tüm yolları yazınız



1, e1, e5, 4
1, e2, e5, 4
1, e1, e3, e4, 4
1, e2, e3, e4, 4

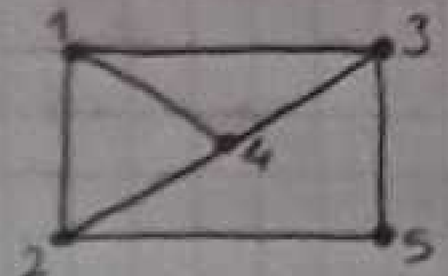
} 4 yolu vardır



→ Şekilde verilen g grafının euler yolu ve euler turu (döngü) sahip olup olmadığını belirleyiniz

1, 3, 4, 5, 3, 2, 1 euler turu (döngü)
euler yolu bulunur

Aşağıdaki grafta hem başlangıç hemde bitiş noktası 5 olan tüm hamilton döngülerini yazınız



5, 3, 1, 4, 2, 5
5, 2, 4, 1, 3, 5
5, 2, 1, 4, 3, 5
5, 3, 4, 1, 2, 5

GRAFLARIN MATRİS GÖSTERİMİ

- Etki Matrisi:

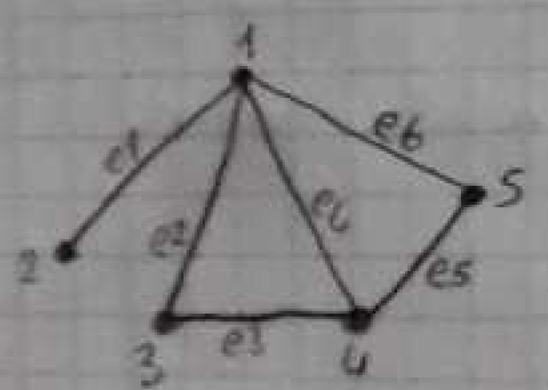
G, n noktalı ve m kenarlı basit bir graf olsun G'nin etki matrisi

$m(G) = (m_{i e_j})_{n \times m}$

Elemanların başlangıç noktası e_j kenarının bitim noktası ise, 1, diğer durumda sıfır şeklinde matris tanımlanmaktadır.

Örnek

Aşağıda verilen grafın etki matrisini hazırlayınız.



$m(G) = (m_{i e_j})_{n \times m}$

i noktasının bağlantılı kenarlarına

$$M(G) = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

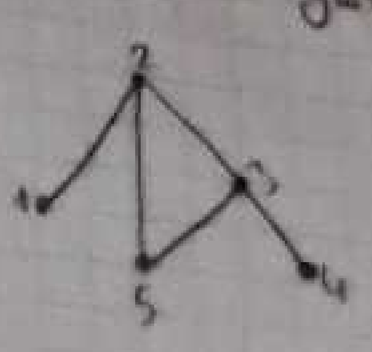
Komşuluk Matrisi

G, n noktalı basit bir graf olsun G'nin komşuluk matrisi $A(G) = (a_{ij})_{n \times n}$ ile gösterilir.

Elemanları ise

$$a_{ij} \begin{cases} 1; & i \sim j \text{ ise} \\ 0; & \text{aksi durumda} \end{cases}$$

Aşağıda şekilde gösterilen g grafına ait komşuluk matrisi yazın



$$A(G) = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \end{bmatrix}_{5 \times 5}$$

Yönlü grafların Komşuluk Matrisi:

G, n noktalı yönlü bir graf

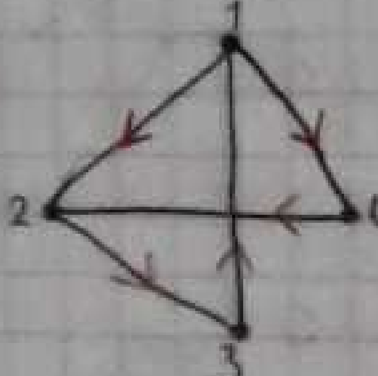
$$A(G) = (a_{ij})_{n \times n}$$ ile gösterilir

Elemanları;

1; ij yönlü kenarı için, yön i'den j'ye ise

0; aksi durumlarda   şeklinde   tanımlanır

Aşağıdaki yönlü grafın komşuluk matrisini hazırlayınız



$$A(G) = \begin{bmatrix} 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}_{4 \times 4}$$

AĞIRLIKLI GRAFLAR

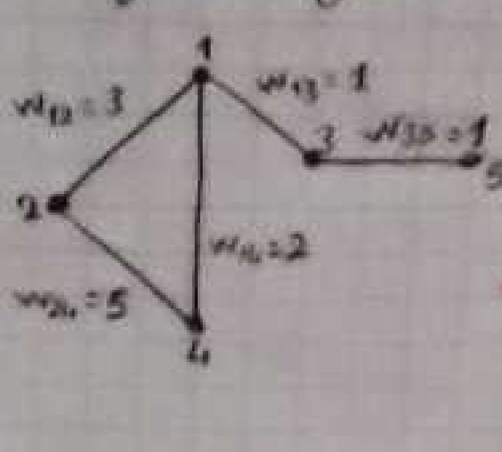
Ağırlıklı grafların Komşuluk Matrisi;

G, n noktalı basit bağlantılı ve ağırlıklı bir graf olsun

G grafın komşuluk matrisi $A(G) = (a_{ij})$ ile gösterilir

Elemanları,

$$a_{ij} = \begin{cases} w_{ij} ; & i \sim j \text{ ise} \\ 0 ; & \text{aksi durumda} \end{cases}$$

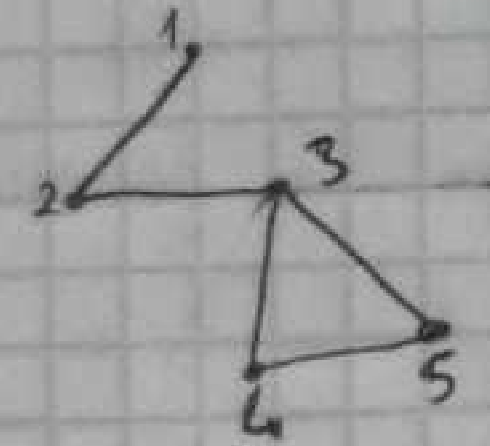


$$A(G) = \begin{array}{c} \begin{matrix} 1 & 2 & 3 & 4 & 5 \end{matrix} \\ \begin{bmatrix} 0 & 3 & 1 & 2 & 0 \\ 3 & 0 & 0 & 5 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 2 & 5 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix} \end{array}$$

örnek:

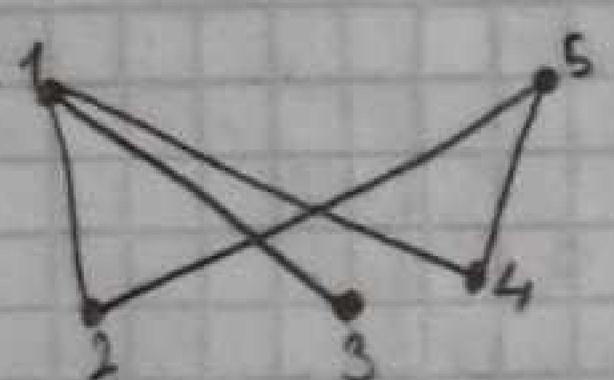
$$A(G) = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \end{bmatrix}$$

şeklinde olan grafı çiziniz



$1 \sim 2$
$2 \sim 1,\ 2 \sim 3$
$3 \sim 2,\ 3 \sim 4,\ 3 \sim 5$
$4 \sim 3,\ 4 \sim 5$
$5 \sim 3,\ 5 \sim 4$

- Aşağıdaki grafa ait 'etki matrisini' yazınız
kenarlara ilişkin

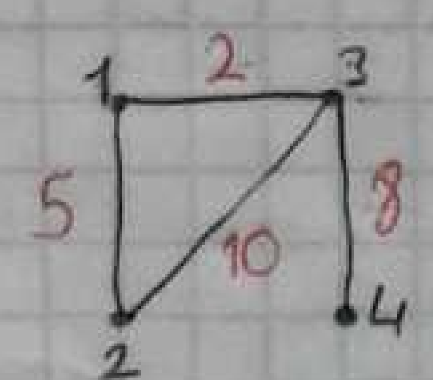


$$M(G) = \begin{bmatrix} 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \end{bmatrix}_{5 \times 5}$$
noktalardan

- $w_{12} = 5$
$w_{13} = 2$
$w_{23} = 10$
$w_{34} = 8$
noktalar
kenarlar

ağırlıkları yandaki olduğu gibi tanımlanan graf çizerek komşuluk matrisi bulunuz

$1 \sim 2,\ 1 \sim 3$
$2 \sim 1,\ 2 \sim 3$
$3 \sim 1,\ 3 \sim 2,\ 3 \sim 4$
$4 \sim 3$



$$A(G) = \begin{bmatrix} 0 & 5 & 3 & 0 \\ 0 & 5 & 10 & 0 \\ 2 & 10 & 0 & 8 \\ 0 & 0 & 8 & 0 \end{bmatrix}_{4 \times 4}$$
komşuluk matrisi

GRAFLARDA ARAMA ALGORİTMALARI

Birçok bilgisayarlı uygulamada graf modellenmesi yaygın olarak kullanılmaktadır. Graflarla temsil edilen çoğu problemde çözümü bulunması için graflar üzerinde farklı şekillerde gezinilmesi istenmektedir. Bu gezinme işlemi grafların özel hali olan ağaç yapılarında sıklıkla kullanılmaktadır.

Graflar üzerinde en sık kullanılan arama algoritmaları enine ve derinine arama algoritmalarıdır.

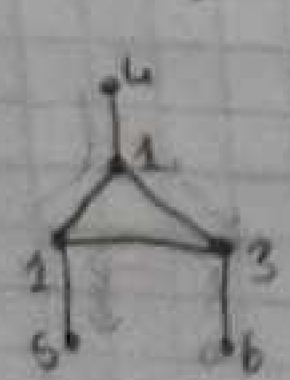
1-) Derinine arama algoritması

Derinine arama algoritmasında arama ağacı fazla dallanmadan kökten gidebilecek en uzak düğüme kadar ilerler. Kök olarak aramanın başlanması gereken herhangi bir düğüm düşünülebilir. Genel olarak bu yöntemde iki durum söz konusudur.

a-) Eğer değerlendirilmiş durum önceki seviyelerde açılmış (ziyaret) ebeveyn düğüme geri dönülerek diğer çocuk düğümler incelenir.

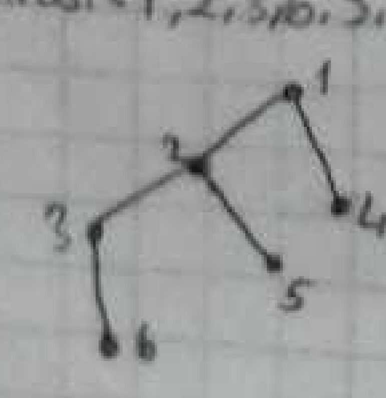
b-) Eğer yeni değerlendirilmiş duruma daha önce rastlanmışsa bu düğümü ebeveyn düğüm kabul edilerek bir sonraki derinlikteki düğümler açılmaya çalışılır.

Aşağıdaki grafın komşuluk matrisini yazarak ağacı üzerindeki ziyaret sırası ve ağaç şeklini hazırlayınız. (Kök düğüm 1 olacak)



$$A(G)=\begin{bmatrix} 0&1&1&1&0&0\\ 1&0&1&0&1&0\\ 1&1&0&0&0&1\\ 1&0&0&0&0&0\\ 0&1&0&0&0&0\\ 0&0&1&0&0&0 \end{bmatrix}$$

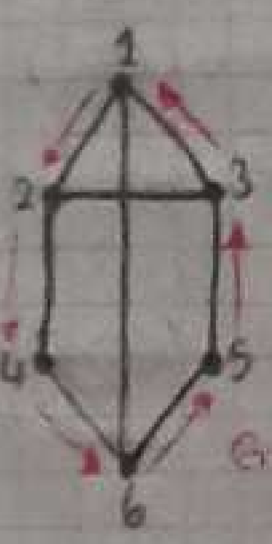
Ziyaret Sırası = 1,2,3,6,5,4



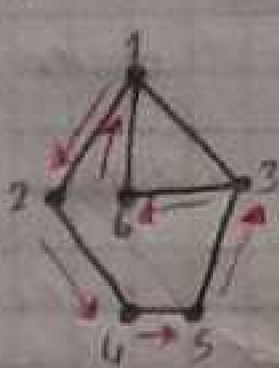
Hamilton Döngüsü:

Bir grafın her bir noktasının tam olarak bir kez kullanıldığı yola "hamilton yolu" denir. Bir grafın her bir noktasından geçen döngüye "hamilton döngüsü" adı verilir. Bir graf hamilton döngüsü varsa bu grafa "hamilton grafı" adı verilir.

Not: Ağırlıksız graflar için hamilton döngüsünde kenar sayısı döngünün uzunluğunu verir. Ağırlıklı graflar için ise hamilton döngüsünde kenar ağırlıklarının toplamı döngünün uzunluğunu vermektedir.

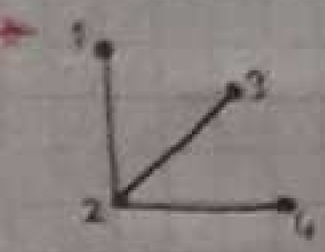


G1

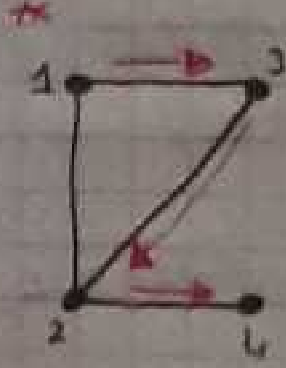


G2

G1, G2 grafları Hamilton graflarıdır



hamilton yolu
" döngüsü
yok



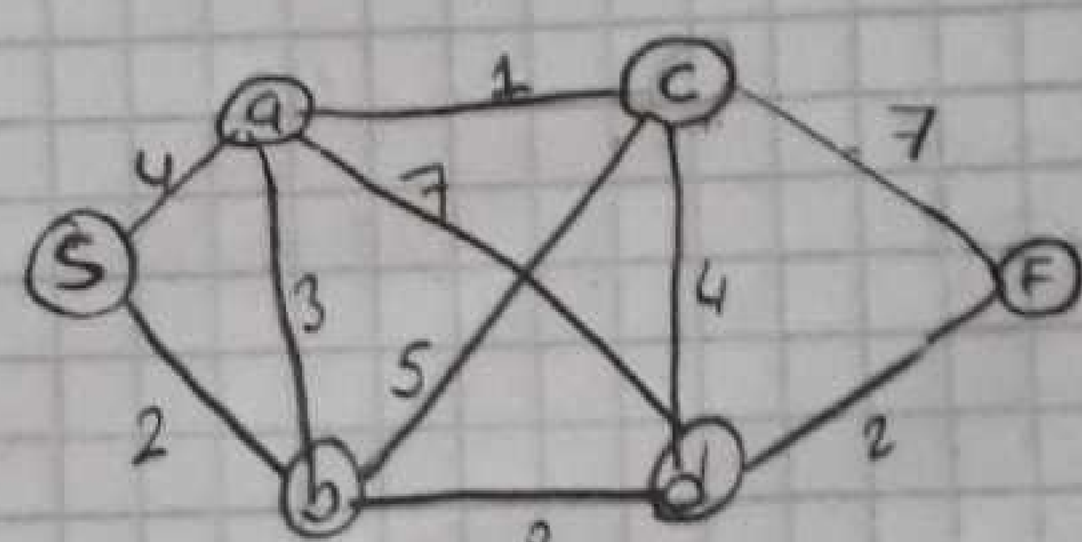
1,3,2,4
hamilton yolu
" döngüsü
yok
(başlangıca dönülmüyor)

Minimum yol (shortest Path) Problemi:

Komşuluk matrisi üzerinden herhangi bir grafın bağlantı yolları oluşturabilir. Bu bağlantı yollarından hareketle yolların ağırlıkları hesaplanarak iki düğüm arasındaki en kısa yol bulunabilir. Buna Dijkstra algoritması denir. Aşağıdaki örnekte S düğümünden f düğümüne en kısa yoldan geçilmesi gerektiğini düşünelim. Komşuluk matrisi aşağıda gösterilmiştir (Dijkstra Al. bütün düğümler birkere ziyaret edilerek yapılır)

A=

	s	a	b	c	d	f
s	0	4	2	∞	∞	∞
a	4	0	3	1	7	∞
b	2	3	0	5	8	∞
c	∞	1	5	0	4	7
d	∞	7	8	4	0	2
f	∞	∞	∞	7	2	0



1. adım = $S \to B$ (2)
2. adım = $b \to a$ (3)
3. adım = $a \to c$ (1)
4. adım = $c \to d$ (4)
5. adım = $d \to F$ (2)

yolun maliyeti = 12

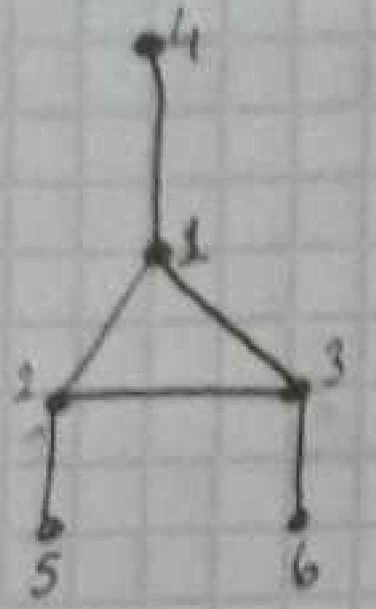
Derinne Arama Algoritması

```
daa (düğümd)
{
    d düğümü ziyaret et
    for(i=0; i<düğüm sayısı; i++)
    {
        if(düğüm ziyaret edilmemisse)
        {
            daa(düğüm i);
        }
    }
}
```

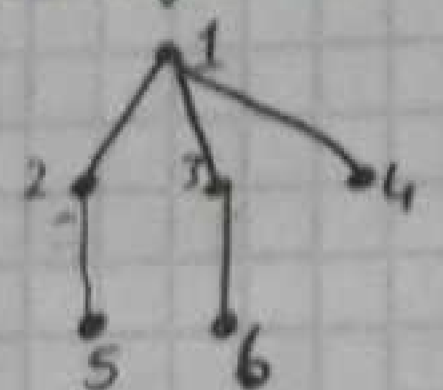
2-) Enine Arama Algoritması

Enine Arama algoritmasında verilen kök düğüme komsu olan düğümlerin hepsi 1. derinlikte incelenmektedir. Belirli bir derinlikte bütün düğümler tek tek incelendikten sonra bir sonraki düğümün degerleri incelenmektedir. Sonlu durumlar söz konusu oldugunda enine arama her zaman köke en yakın çözümü vermektedir. fakat Çözüme ulasılana kadar o derinlikteki tüm düğümler ziyaret edilir.

Geçen hafta derinine arama algoritmasında kullanılan örnek üzerinden yola çıkıldığında kök düğüm olarak 1 numaralı düğümün seçildiğini varsayalım. Bir düğümün ziyaret edildikten sonra bu düğümün komsuları olan 2, 3, 4 düğümleri ardısık olarak ziyaret edilir. Daha sonra 2 numaralı düğüm kuyruktan çekilerek arama listesinde bulunur ve komşusu olan 5 numaralı düğüm ziyaret edilir. Daha sonra ise 3 numaralı düğüm arama listesine konur ve komşusu olan 6 numaralı düğüm ziyaret edilir.



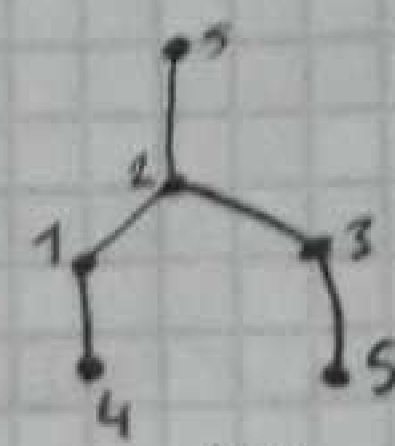
Kök düğüm = 1



Ağaç

Ziyaret Sırası: 1,(2,3,4) 5,6

Kök düğüm = 5



Ağaç

Ziyaret Sırası: 5,2(1,3)4,6

Enine Arama Algoritmasının Adım Adım İşletilmesi:

1-) Bir başlangıç düğümü seçilir ve bu düğüm işaretlenir.

2-) Bu düğümün komşuları sırasıyla "T" listesine eklenir.

3-) Düğümler tek tek ziyaret edilir ve işaretlenerek "L" listesine atılır.

4-) T listesindeki ilk düğüme gidilir ve işaretlendikten (ziyaret) sonra L listesinden silinir.

5-) Silinen düğümün komşuları T listesine eklenir (Bunu yaparken komşu düğüm daha önce ziyaret edilmiş ise bu düğüm L listesinden kontrol edilir ve ekleme yapılmaz.)

6-) "L" listesine bu düğümler eklenir.

7-) İlk 6 adım T listesindeki tüm düğümler silinene kadar devam ettirilir.