VERİ YAPILARI

GRAPH'LAR

Bilgisayar Mühendisliği

GRAPH (ÇİZGE - GRAF)

Terminoloji

Çizge Kullanım Alanları

Çizge Gösterimi

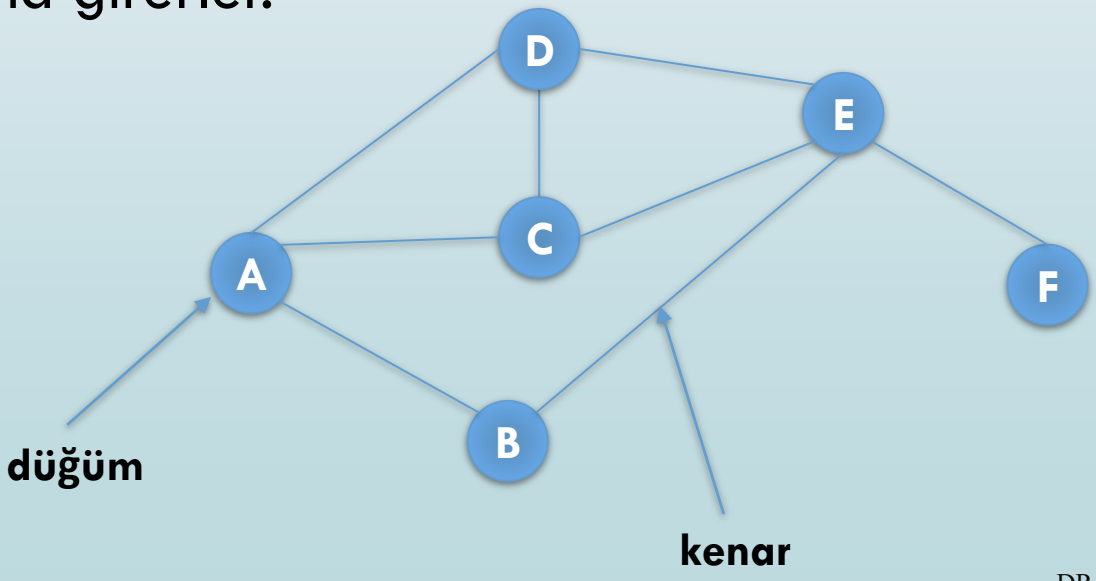
- Komşuluk Matrisi
- Komşuluk Listesi

İlgili Algoritmalar

- Çizge Üzerinde Gezinme
  - DFS
  - BFS
- Dijsktra Algoritması
- Kruskal Algoritması (Minimum Spining Tree)
- Toplojik Sıralama

ÇİZGE (GRAPH) GİRİŞ

- **Köşe** (vertex) isimli **düğümlerden** ve **kenar** (**edge**) isimli köşeleri birbirine bağlayan **bağlantılardan** oluşan veri yapısı modelidir.
- Ağaçlar gibi çizgeler de **doğrusal olmayan** veri yapıları grubuna girerler.



ÇİZGE (GRAPH) GİRİŞ (DEVAM...)

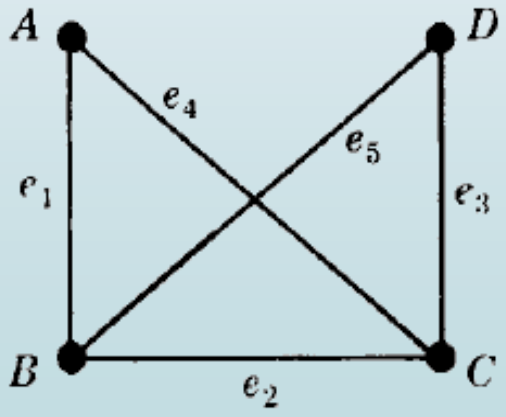
- Bir G çizgesi G(V, E) şeklinde gösterilir:
  a. **V = V(G) Kümesi:** Küme elemanları, G'nin düğümleri (nodes), noktaları (points) veya köşeleri (vertices)
  b. **E = E(G) Kümesi:** Küme elemanları G'nin kenarları (edges) olarak adlandırılan sırasız düğüm ikililerini içerir.

ÇİZGE (GRAPH) GİRİŞ (DEVAM...)

Çizgeler, **düzlemsel diyagramlarla** gösterilir.

V kümesindeki her v düğümü bir nokta (ya da küçük çember) ile temsil edilir ve her e = {v1, v2} kenarı, v1 ve v2 uç noktalarını bağlayan bir çizgi ile gösterilir.

Örnek:



V = {A, B, C, D}

E = {e_1, e_2, e_3, e_4, e_5}

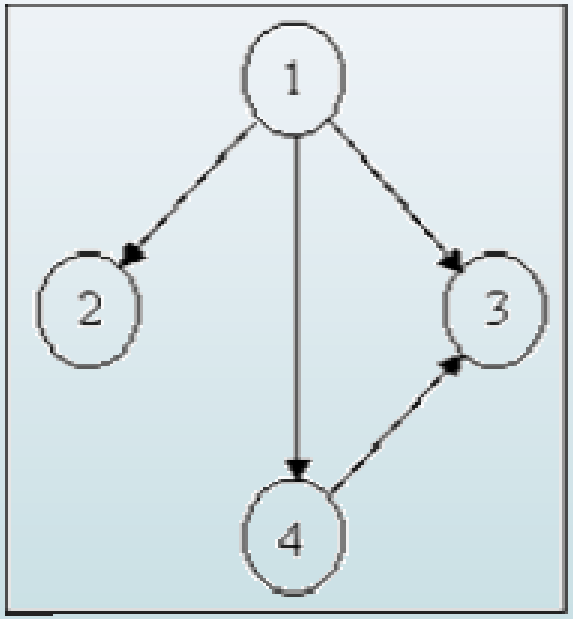
$$e_1 = \{A, B\}, e_2 = \{B, C\}, e_3 = \{C, D\}, e_4 = \{A, C\}, e_5 = \{B, D\}.$$

YÖNLÜ VE YÖNSÜZ KENAR

Yönsüz Kenar (undirected edge):
Çizgi seklinde yönü belirtilmeyen kenarlar yönsüz kenarlardır.

- Yönsüz kenarlarda (v1,v2) olması ile (v2,v1) olması arasında fark yoktur.

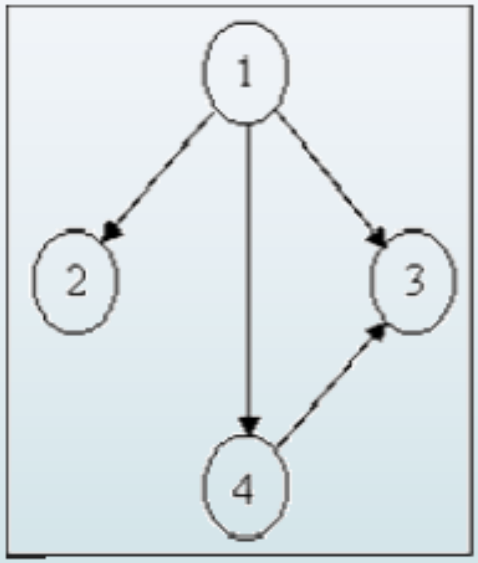
Yönlü Kenar (directed edge): Ok şeklinde gösterilen kenarlar yönlü kenarlardır.



$G_1=(V,E)$
$V_1=\{1,2,3,4\}$
$E_1=\{(1,2),(1,3),(1,4),(4,3)\}$

TERMİNOLOJİ

1. **Komşu Köşeler (Adjacent):** Aralarında doğrudan bağlantı (kenar) bulunan i ve j köşeleri komşudur. Diğer köşe çiftleri komşu değildir.
2. **Bağlantı (incident):** Komşu i ve j köşeleri arasındaki kenar (i, j) bağlantıdır.
3. **Bir Köşenin Derecesi (degree):** Bir köşeye bağlı olan kenarların sayısıdır.



$G_1=(V,E)$
$V_1=\{1,2,3,4\}$
$E_1=\{(1,2),(1,3),(1,4),(4,3)\}$

Soru1: Komşu olmayan köşeler hangileridir?

Soru2: 1'in derecesi kaçtır?

TERMİNOLOJİ (DEVAM...)

4. **Yönsüz Çizge (undirected graph):** Tüm kenarları ***yönsüz*** olan çizgeye yönsüz çizge denilir. Yönsüz çizgede bir köşe çifti arasında en fazla bir kenar olabilir.

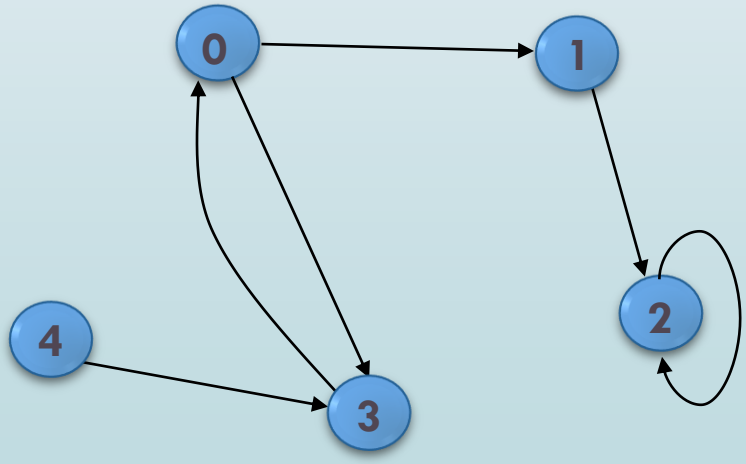
5. **Yönlü Çizge (directed graph, digraph):** Tüm kenarları **yönlü** olan çizgeye yönlü çizge adı verilir. Yönlü çizgede bir köşe çifti arasında ters yönlerde olmak üzere ***en fazla iki kenar*** olabilir.

6. **Döngü (Loop):** (i, i) seklinde gösterilen ve bir köşeyi kendine bağlayan kenar.

TERMİNOLOJİ (DEVAM...)

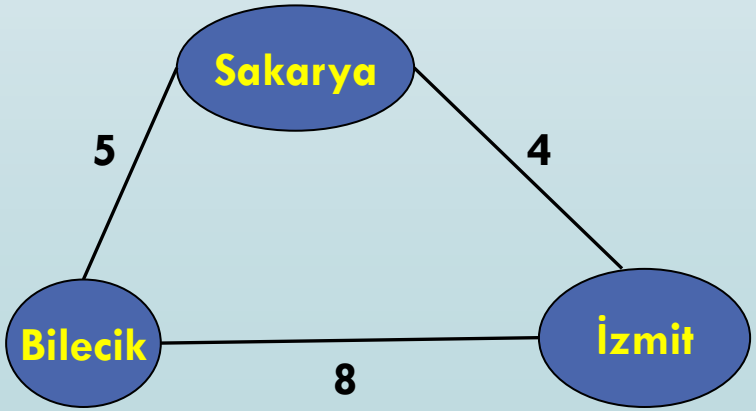
Yönlü Çizge ve Döngü (Loop)

- G=(V, E)
- V={0,1,2,3,4}
- E={(0,1), (1,2), (0,3), <3,0>, <2,2>, <4,3>}



TERMİNOLOJİ (DEVAM...)

7. **Ağırlıklı (Weighted) Çizge:** Çizge kenarları üzerinde ağırlıkları olabilir. Eğer kenarlar üzerinde ağırlıklar varsa bu tür çizgelere **ağırlıklı/maliyetli** çizge (Weighted Graphs) denir. Ağırlık uygulamadan uygulamaya değişir.
   - Şehirler arasındaki uzaklık
   - Routerler arası bant genişliği



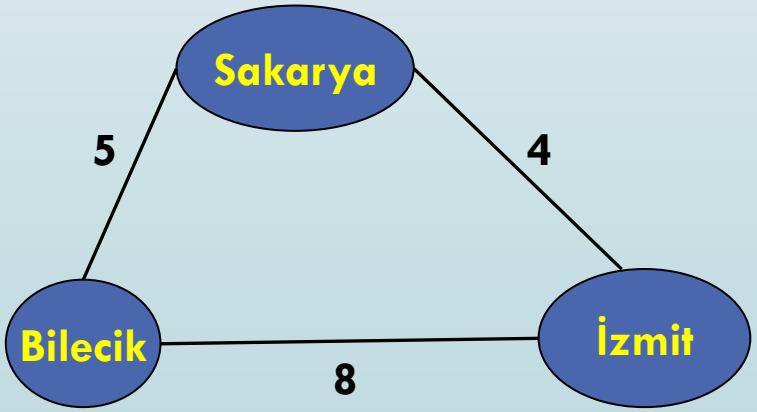
YOL (PATH)

Basit Yol (Simple Path): Tüm düğümlerin farklı olduğu yoldur.

Daire (Cycle): Başlangıç ve bitiş düğümleri aynı olan basit yol. **(Tamamlanmış Graf)**

Uzunluk: Bir yol üzerindeki kenarların sayısının toplamıdır.

Basit Yol: *Sakarya, İzmit veya Sakarya, Bilecik*
Daire: *Sakarya, İzmit, Bilecik, Sakarya*



ÇİZGE KULLANIM ALANLARI

Bilgisayar ağlarında, elektriksel ve diğer ağların analizinde,

Kimyasal bileşiklerin moleküler yapılarının araştırılmasında,

Ulaşım ağlarında (kara, deniz ve havayolları),

Planlama projelerinde,

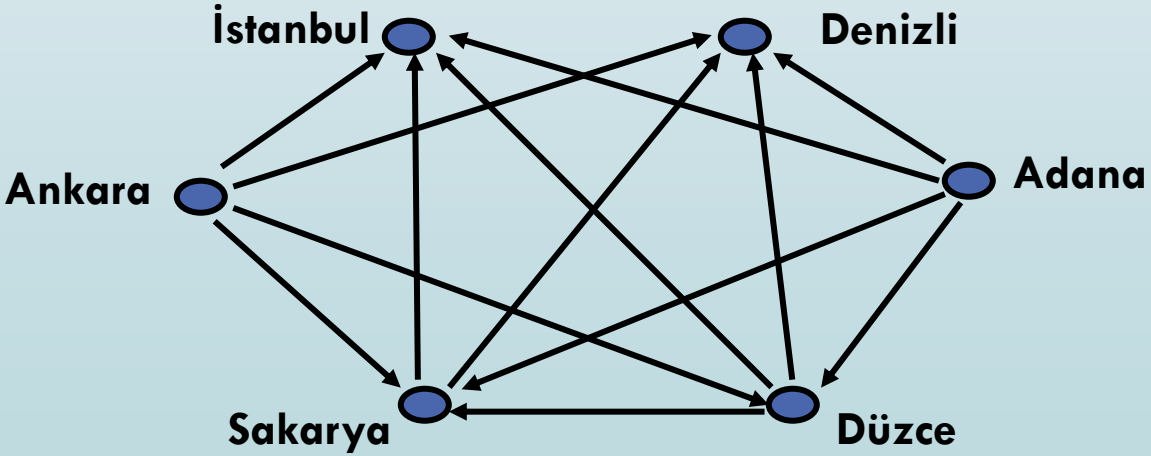
Sosyal alanlarda ve diğer pek çok alanda kullanılmaktadır.

Not: Eğer bir problemin çözümü GRAF veri modeline benzetilebiliyorsa, o problem için algoritmik bir durum elde edilmiş olur.

ÇİZGE KULLANIM ALANLARI (DEVAM...)

Örnek (Generic)

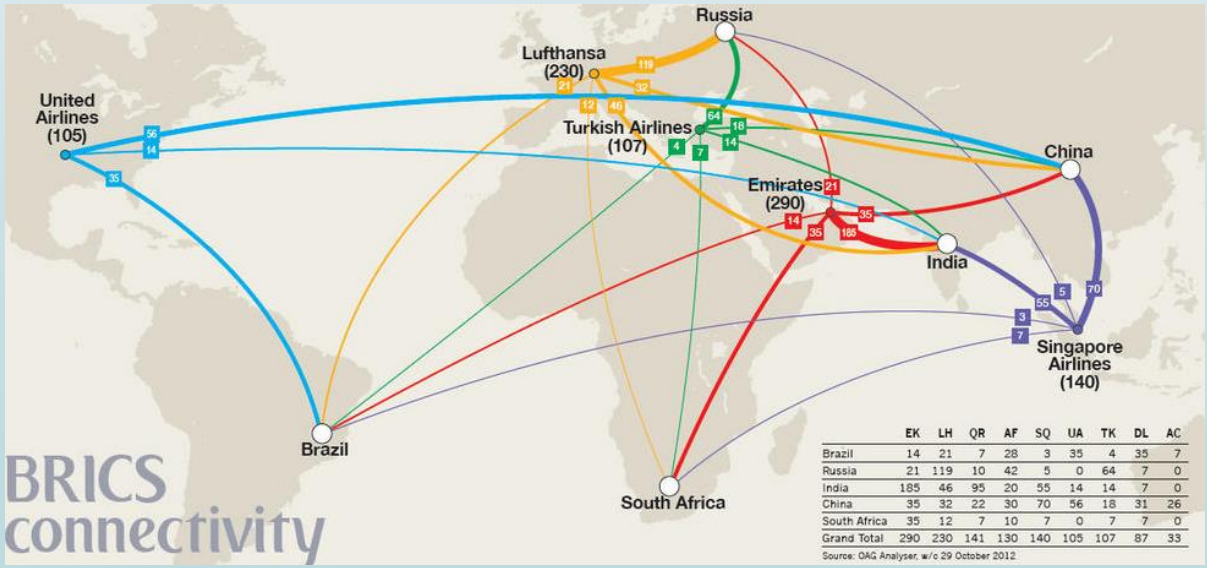
- V = 6 il: sırasıyla mesafeleri:
- Ankara, İstanbul, Adana, Denizli, Sakarya ve Düzce
- 80, 105, 80, 105, 103 ve 90.
- E ={(x, y) | Eğer x, y den daha küçük ise}



ÇİZGE KULLANIM ALANLARI (DEVAM...)

Örnek (Uçuş sistemi)

- Her bir düğüm bir şehri gösterir
- Her bir kenar iki şehir arasındaki doğrudan uçuşu gösterir
- Doğrudan uçuşların sorgulanmasında cevap bir kenardır.
- Bir yere ulaşmak için "A' dan B' ye yol var mı" sorusu sorulur.
- Maliyetleri kenarlara bile ekleyebiliriz. (ağırlıklı), daha sonra "A'dan B' ye en ucuz yol hangisidir?" diye sorabiliriz.



	EK	LH	QR	AF	SQ	UA	TK	DL	AC
Brazil	14	21	7	28	3	35	4	35	7
Russia	21	119	10	42	5	0	64	7	0
India	185	46	95	20	55	14	14	7	0
China	35	32	22	30	70	56	18	31	26
South Africa	35	12	7	10	7	0	7	7	0
Grand Total	290	230	141	130	140	105	107	87	33

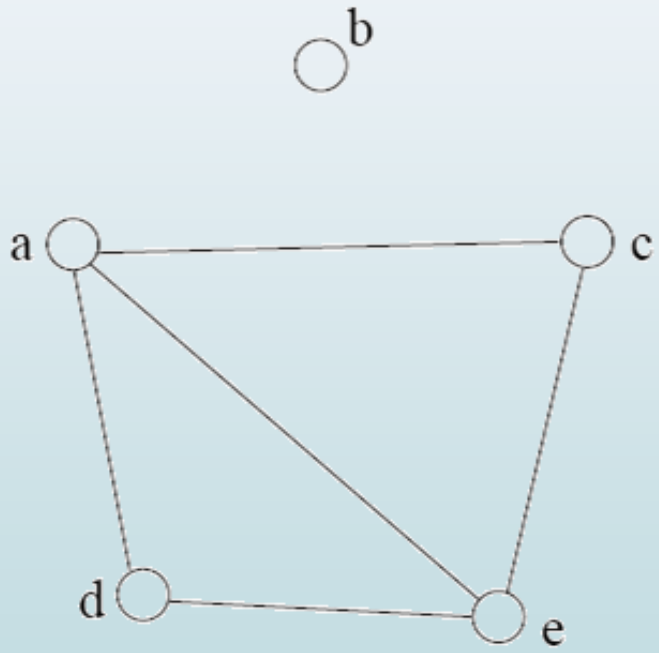
Source: OAG Analyser, w/c 29 October 2012

ÇİZGE GÖSTERİMİ

İki popüler gösterim bulunmaktadır. Her ikisi de farklı yönlerden düğüm ve kenar kümelerini gösterir.

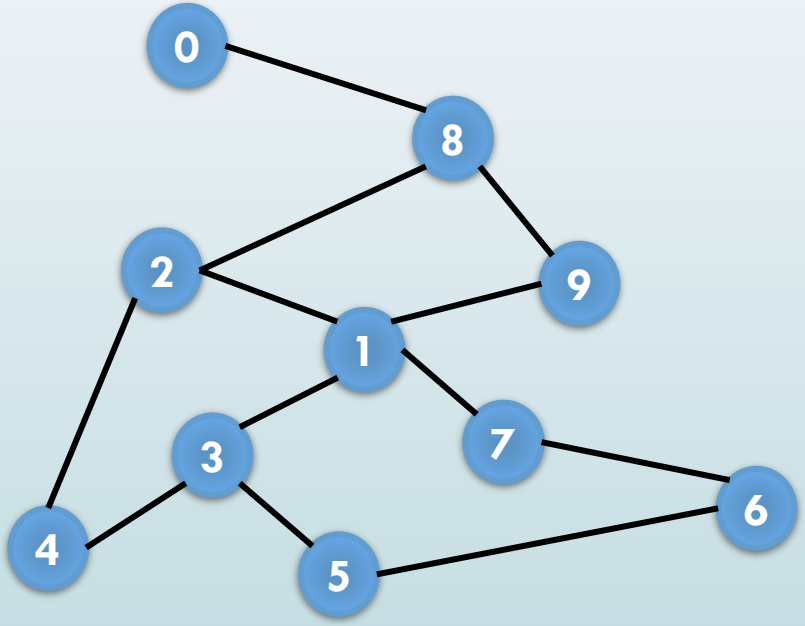
1. **Komşuluk Matrisi:** Çizgeyi göstermek için D matrisi kullanılır.
2. **Komşuluk Listesi:** Bağlantılı listelerin bir boyutlu dizisi kullanılır.

KOMŞULUK MATRİSİ



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

KOMŞULUK MATRİSİ (DEVAM...)



	0	1	2	3	4	5	6	7	8	9
0	0	0	0	0	0	0	0	0	1	0
1	0	0	1	1	0	0	0	1	0	1
2	0	1	0	0	1	0	0	0	1	0
3	0	1	0	0	1	1	0	0	0	0
4	0	0	1	1	0	0	0	0	0	0
5	0	0	0	1	0	0	1	0	0	0
6	0	0	0	0	0	1	0	1	0	0
7	0	1	0	0	0	0	1	0	0	0
8	1	0	1	0	0	0	0	0	0	1
9	0	1	0	0	0	0	0	0	1	0

KOMŞULUK MATRİSİ (DEVAM...)

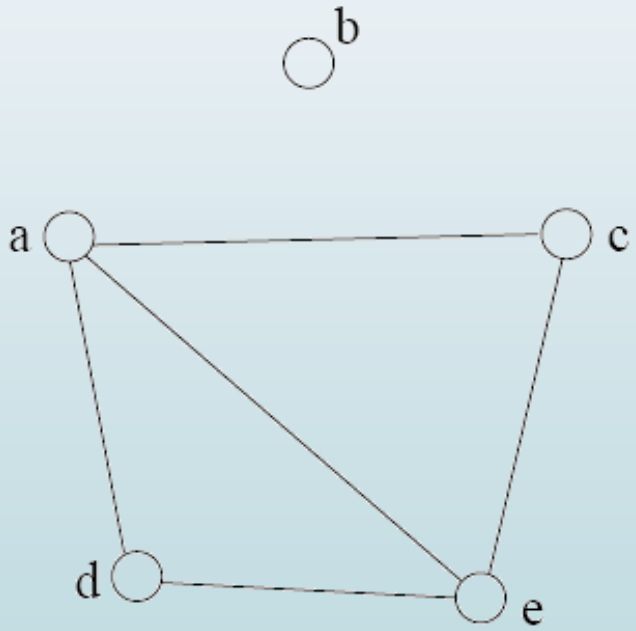
İki boyutlu dizi ile gerçekleştirilebilir.

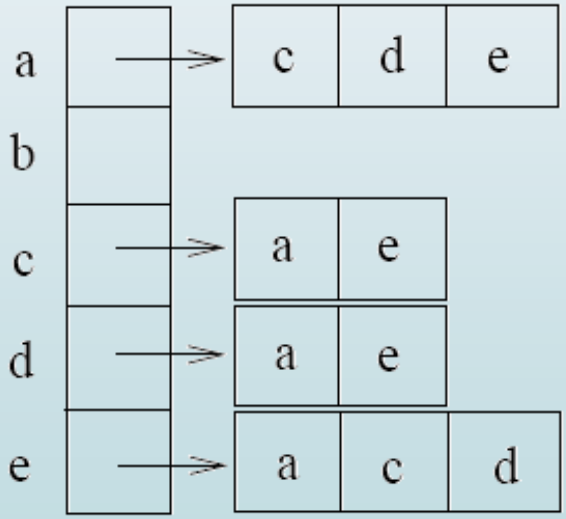
Uygulaması basittir.

- Kenar Oluşturmak ve kaldırmak kolaydır.

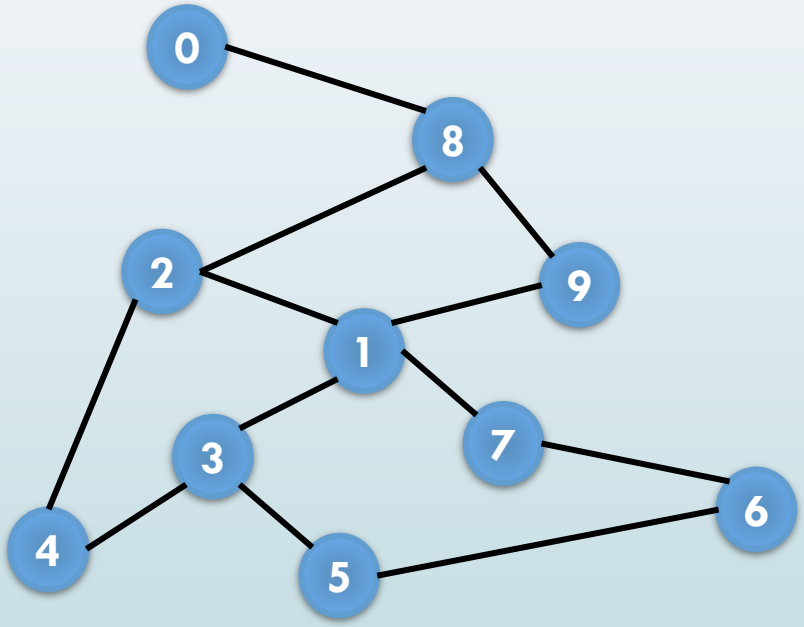
Hafızada fazla yer kaplar.

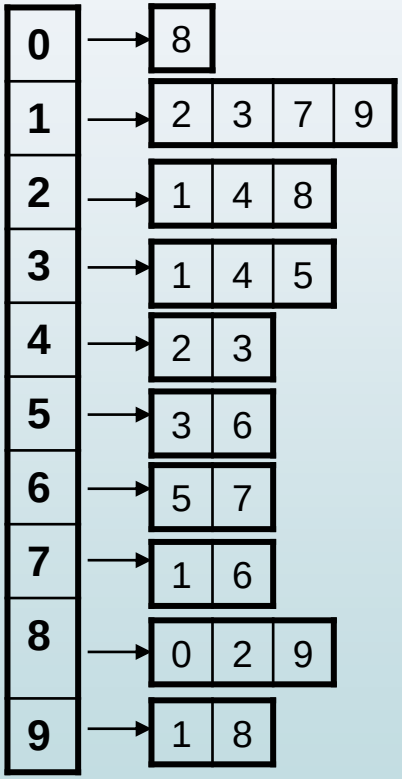
KOMŞULUK LİSTESİ





KOMŞULUK LİSTESİ (DEVAM...)





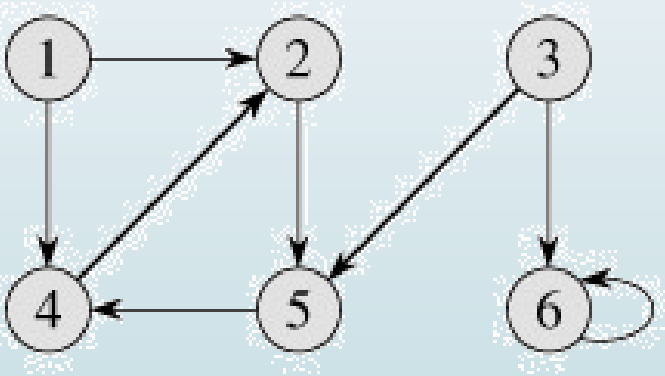
KOMŞULUK LİSTESİ (DEVAM...)

Bağlı liste içeren dizi ile gerçekleştirimi yapılır.

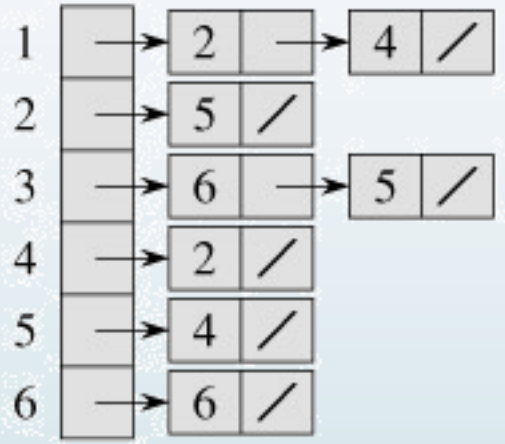
Uygulaması daha karmaşıktır.

Hafıza kullanımı komşuluk matrisine göre daha optimaldir.

ÖRNEK – YÖNLÜ ÇİZGE



Komşuluk Listesi



Komşuluk Matrisi

	1	2	3	4	5	6
1	0	1	0	1	0	0
2	0	0	0	0	1	0
3	0	0	0	0	1	1
4	0	1	0	0	0	0
5	0	0	0	1	0	0
6	0	0	0	0	0	1

GRAFLARIN BELLEK ÜZERİNDE TUTULMASI

	Bellek Gereksinimi	Bağlantı Sorugulama	Bağlantı Ekleme
Matris Üzerinde	N^2*b	O(1)	O(1)
İki Dizi Üzerinde	((2*m-c)+(N+1)*b	O(d)	O(m)
Bağlantılı Liste İle	N*(b+(d*a))	$O(N^2)$	$O(N^2)$
Dizi-Bağlantılı Liste	(N+2*m)*(b+a)	O(d)	O(d)

N= Düğüm Sayısı
m= Kenar sayısı
d= Düğüm derecesi
c= Çevrimli kenar sayısı
b= Veri türü boyutu
a= Bellek adres boyutu

ÇİZGE ÜZERİNDE GEZİNME (TRAVERSE)

Çizge üzerinde dolaşma; çizge düğümleri ve kenarları üzerinde **istenen bir işi yapacak** veya bir *problemi çözecek* biçimde hareket etmektir.

Çizge üzerinde dolaşma yapan **birçok yaklaşım ve yöntem** bulunmaktadır. En önemli iki tanesi aşağıdaki gibidir:

- **BFS (Breadth First Search) (Sığ Öncelikli Arama) Yöntemi**
- **DFS (Depth First Search) (Derin Öncelikli Arama) Yöntemi**

ÇİZGE ÜZERİNDE GEZİNME (DEVAM...)

Depth-First Search (DFS)

Bir düğümden başla, düğümün bir kenarında o kenar üzerinde gidilebilecek **en uzak** düğüme kadar sürdür.

Geri gel (backtracking) ve diğer kenarı dene.

Tüm düğümler gezilene kadar devam et.

Breadth-First Search (BFS)

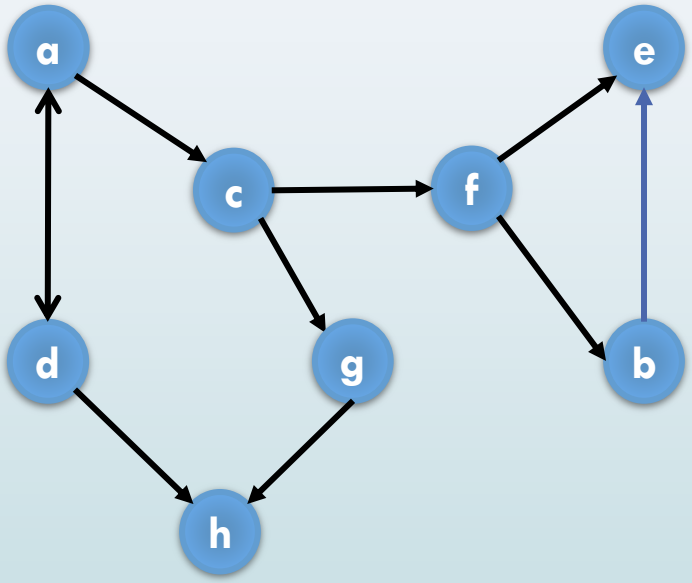
Başlangıç düğümünden başla ve tüm komşuları **ziyaret et**.

Daha sonra **komşunun** komşularını ziyaret et.

Başlangıç düğümünden başlayıp ***dışa doğru*** dalga gibi ilerle.

DFS GÖSTERİM

Çizge

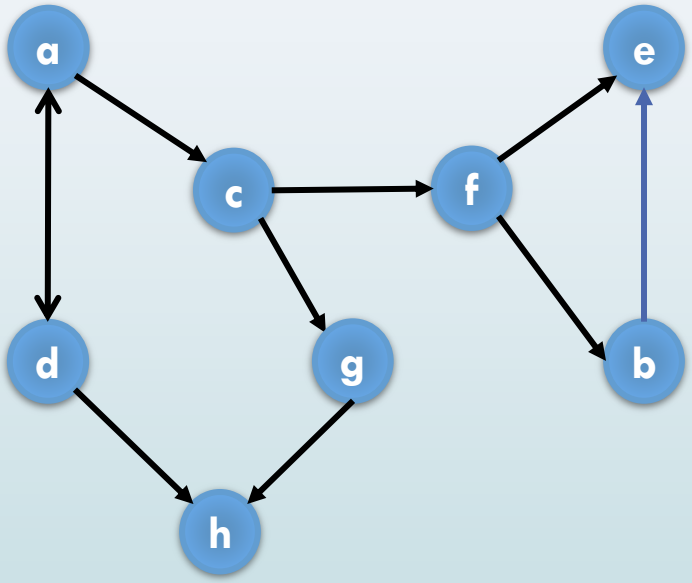


ÇIKTI : a c f e b g h d

STACK

BFS GÖSTERİM

Çizge

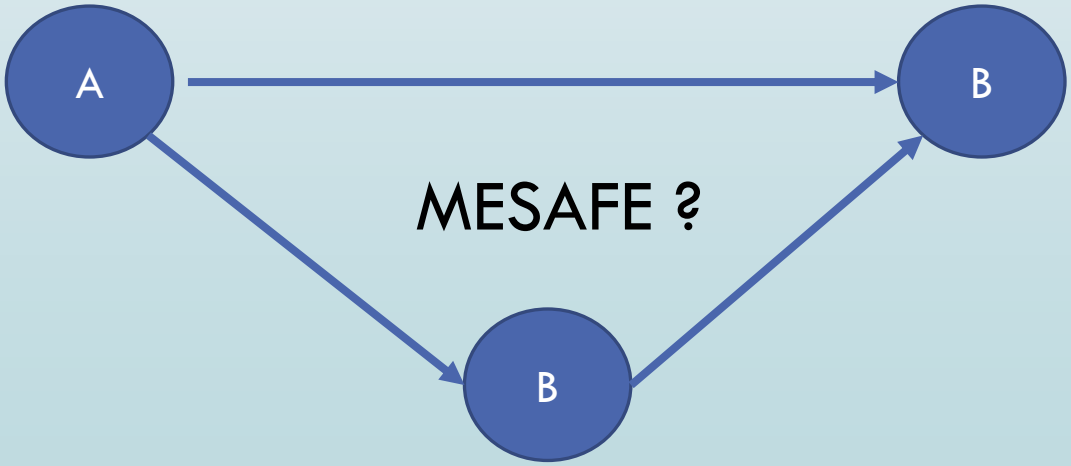


ÇIKTI : a c d f g h e b

QUEUE

DİJKSTRA ALGORİTMASI

Dijkstra algoritması, ağırlıklı bir graf üzerinde yani kenarları (edge) belli bir metrik değere sahip olan herhangi iki düğüm arası en kısa mesafeyi bulmamızı sağlayan bir algoritmadır.

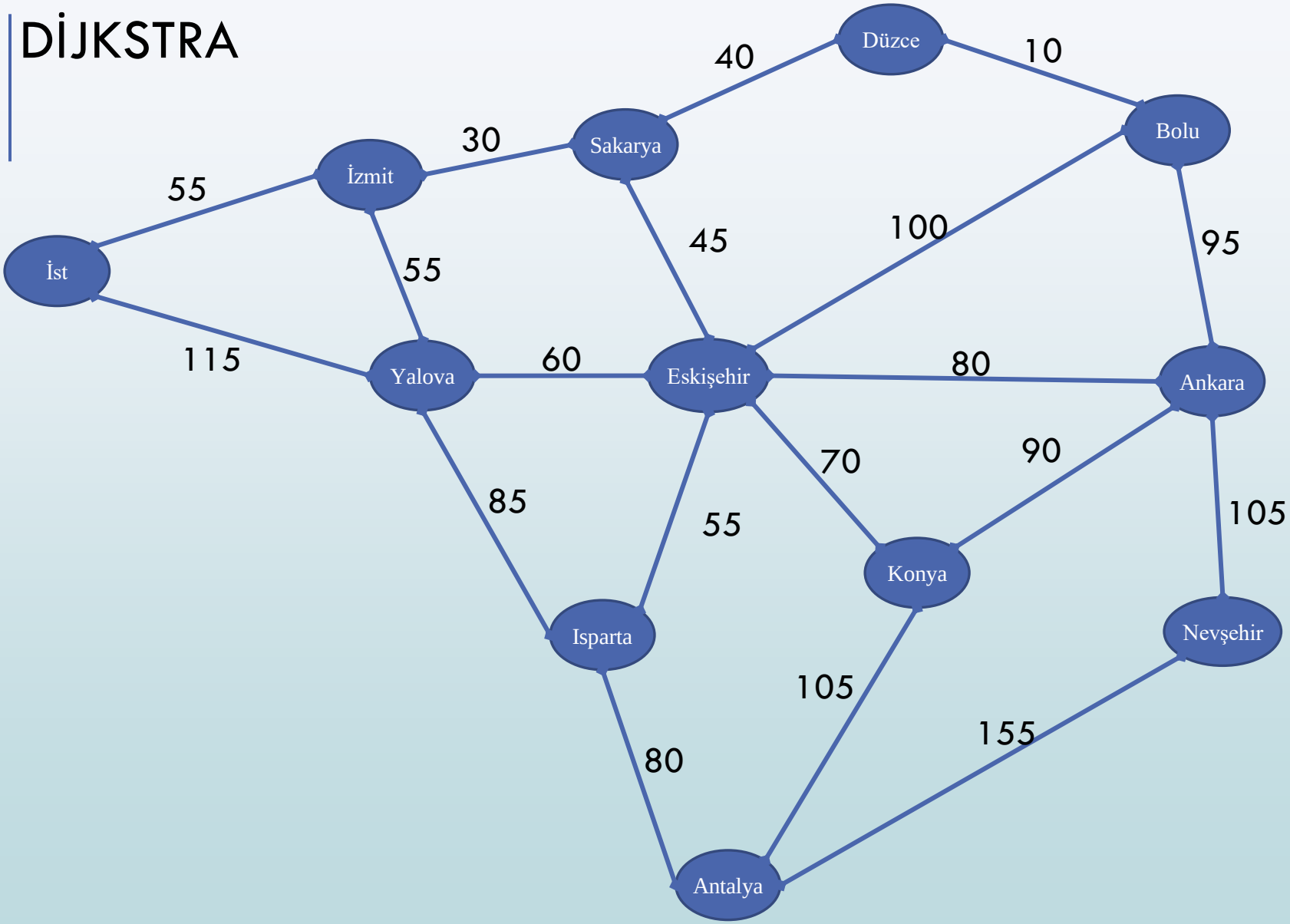


DİJKSTRA ALGORİTMASI

Dijkstra algoritmasını kullanmamız için;

- Grafımız ağırlık ve yönlü olmalı.
- Kenarların ağırlık değeri sıfır ya da sıfırdan büyük bir değer olmalıdır.
- Eğer kenar değerleri sıfırdan küçükse Bellamn-Ford algoritması kullanılabilir.
- Dijkstra algoritmasının zaman karmaşıklığı yani Big O notasyonu O(MlogN)'dir.

DİJKSTRA


Düzce
Bolu
Sakarya
İzmit
İst
Yalova
Eskişehir
Ankara
Konya
Isparta
Nevşehir
Antalya
40
10
30
55
55
45
100
95
115
60
80
85
55
70
90
105
105
80
155

DİJKSTRA

	İzmit ∞	Yalova ∞	Sakarya ∞	Eskişehir ∞	Düzce ∞	Bolu ∞	Ankara ∞	Nevşehir ∞	Konya ∞	Isparta ∞	Antalya ∞

DİJKSTRA

	İzmit ∞	Yalova ∞	Sakarya ∞	Eskişehir ∞	Düzce ∞	Bolu ∞	Ankara ∞	Nevşehir ∞	Konya ∞	Isparta ∞	Antalya ∞
İstanbul	55(İs)	115(İs)	∞	∞	∞	∞	∞	∞	∞	∞	∞

DİJKSTRA

	İzmit ∞	Yalova ∞	Sakarya ∞	Eskişehir ∞	Düzce ∞	Bolu ∞	Ankara ∞	Nevşehir ∞	Konya ∞	Isparta ∞	Antalya ∞
İstanbul	55(İs)	115(İs)	∞	∞	∞	∞	∞	∞	∞	∞	∞
İzmit	-	55(İz) 110(İs)	30(iz) 85(İs)	∞	∞	∞	∞	∞	∞	∞	∞

DİJKSTRA

	İzmit ∞	Yalova ∞	Sakarya ∞	Eskişehir ∞	Düzce ∞	Bolu ∞	Ankara ∞	Nevşehir ∞	Konya ∞	Isparta ∞	Antalya ∞
İstanbul	55(İs)	115(İs)	∞	∞	∞	∞	∞	∞	∞	∞	∞
İzmit	-	55(İz) 110(İs)	30(iz) 85(İs)	∞	∞	∞	∞	∞	∞	∞	∞
Sakarya	-	55(İz) 110(İs)	-	45(S) 130(İs)	40(S) 125(İs)	∞	∞	∞	∞	∞	∞

DİJKSTRA

	İzmit ∞	Yalova ∞	Sakarya ∞	Eskişehir ∞	Düzce ∞	Bolu ∞	Ankara ∞	Nevşehir ∞	Konya ∞	Isparta ∞	Antalya ∞
İstanbul	55(İs)	115(İs)	∞	∞	∞	∞	∞	∞	∞	∞	∞
İzmit	-	55(İz) 110(İs)	30(iz) 85(İs)	∞	∞	∞	∞	∞	∞	∞	∞
Sakarya	-	55(İz) 110(İs)	-	45(S) 130(İs)	40(S) 125(İs)	∞	∞	∞	∞	∞	∞
Yalova	-	-	-	45(S) 130(İs)	40(S) 125(İs)	∞	∞	∞	∞	85(Y) 195(İs)	∞

DİJKSTRA

	İzmit ∞	Yalova ∞	Sakarya ∞	Eskişehir ∞	Düzce ∞	Bolu ∞	Ankara ∞	Nevşehir ∞	Konya ∞	Isparta ∞	Antalya ∞
İstanbul	55(İs)	115(İs)	∞	∞	∞	∞	∞	∞	∞	∞	∞
İzmit	-	55(İz) 110(İs)	30(iz) 85(İs)	∞	∞	∞	∞	∞	∞	∞	∞
Sakarya	-	55(İz) 110(İs)	-	45(S) 130(İs)	40(S) 125(İs)	∞	∞	∞	∞	∞	∞
Yalova	-	-	-	45(S) 130(İs)	40(S) 125(İs)	∞	∞	∞	∞	85(Y) 195(İs)	∞
Düzce	-	-	-	45(S) 130(İs)	-	10(D) 135(İs)	∞	∞	∞	85(Y) 195(İs)	∞

DİJKSTRA

	İzmit ∞	Yalova ∞	Sakarya ∞	Eskişehir ∞	Düzce ∞	Bolu ∞	Ankara ∞	Nevşehir ∞	Konya ∞	Isparta ∞	Antalya ∞
İstanbul	55(İs)	115(İs)	∞	∞	∞	∞	∞	∞	∞	∞	∞
İzmit	-	55(İz) 110(İs)	30(iz) 85(İs)	∞	∞	∞	∞	∞	∞	∞	∞
Sakarya	-	55(İz) 110(İs)	-	45(S) 130(İs)	40(S) 125(İs)	∞	∞	∞	∞	∞	∞
Yalova	-	-	-	45(S) 130(İs)	40(S) 125(İs)	∞	∞	∞	∞	85(Y) 195(İs)	∞
Düzce	-	-	-	45(S) 130(İs)	-	10(D) 135(İs)	∞	∞	∞	85(Y) 195(İs)	∞
Eskişehir	-	-	-	-	-	10(D) 135(İs)	80(E) 210(İs)	∞	70(E) 200(İs)	55(E) 185(İs)	∞

DİJKSTRA

	İzmit ∞	Yalova ∞	Sakarya ∞	Eskişehir ∞	Düzce ∞	Bolu ∞	Ankara ∞	Nevşehir ∞	Konya ∞	Isparta ∞	Antalya ∞
İstanbul	55(İs)	115(İs)	∞	∞	∞	∞	∞	∞	∞	∞	∞
İzmit	-	55(İz) 110(İs)	30(iz) 85(İs)	∞	∞	∞	∞	∞	∞	∞	∞
Sakarya	-	55(İz) 110(İs)	-	45(S) 130(İs)	40(S) 125(İs)	∞	∞	∞	∞	∞	∞
Yalova	-	-	-	45(S) 130(İs)	40(S) 125(İs)	∞	∞	∞	∞	85(Y) 195(İs)	∞
Düzce	-	-	-	45(S) 130(İs)	-	10(D) 135(İs)	∞	∞	∞	85(Y) 195(İs)	∞
Eskişehir	-	-	-	-	-	10(D) 135(İs)	80(E) 210(İs)	∞	70(E) 200(İs)	55(E) 185(İs)	∞
Bolu	-	-	-	-	-	-	80(E) 210(İs)	∞	70(E) 200(İs)	55(E) 185(İs)	∞

DİJKSTRA

	İzmit ∞	Yalova ∞	Sakarya ∞	Eskişehir ∞	Düzce ∞	Bolu ∞	Ankara ∞	Nevşehir ∞	Konya ∞	Isparta ∞	Antalya ∞
İstanbul	55(İs)	115(İs)	∞	∞	∞	∞	∞	∞	∞	∞	∞
İzmit	-	55(İz) 110(İs)	30(iz) 85(İs)	∞	∞	∞	∞	∞	∞	∞	∞
Sakarya	-	55(İz) 110(İs)	-	45(S) 130(İs)	40(S) 125(İs)	∞	∞	∞	∞	∞	∞
Yalova	-	-	-	45(S) 130(İs)	40(S) 125(İs)	∞	∞	∞	∞	85(Y) 195(İs)	∞
Düzce	-	-	-	45(S) 130(İs)	-	10(D) 135(İs)	∞	∞	∞	85(Y) 195(İs)	∞
Eskişehir	-	-	-	-	-	10(D) 135(İs)	80(E) 210(İs)	∞	70(E) 200(İs)	55(E) 185(İs)	∞
Bolu	-	-	-	-	-	-	80(E) 210(İs)	∞	70(E) 200(İs)	55(E) 185(İs)	∞
Isparta	-	-	-	-	-	-	80(E) 210(İs)	∞	70(E) 200(İs)	-	80(I) 265(İs)

DİJKSTRA

	İzmit ∞	Yalova ∞	Sakarya ∞	Eskişehir ∞	Düzce ∞	Bolu ∞	Ankara ∞	Nevşehir ∞	Konya ∞	Isparta ∞	Antalya ∞
İstanbul	55(İs)	115(İs)	∞	∞	∞	∞	∞	∞	∞	∞	∞
İzmit	-	55(İz) 110(İs)	30(iz) 85(İs)	∞	∞	∞	∞	∞	∞	∞	∞
Sakarya	-	55(İz) 110(İs)	-	45(S) 130(İs)	40(S) 125(İs)	∞	∞	∞	∞	∞	∞
Yalova	-	-	-	45(S) 130(İs)	40(S) 125(İs)	∞	∞	∞	∞	85(Y) 195(İs)	∞
Düzce	-	-	-	45(S) 130(İs)	-	10(D) 135(İs)	∞	∞	∞	85(Y) 195(İs)	∞
Eskişehir	-	-	-	-	-	10(D) 135(İs)	80(E) 210(İs)	∞	70(E) 200(İs)	55(E) 185(İs)	∞
Bolu	-	-	-	-	-	-	80(E) 210(İs)	∞	70(E) 200(İs)	55(E) 185(İs)	∞
Isparta	-	-	-	-	-	-	80(E) 210(İs)	∞	70(E) 200(İs)	-	80(I) 265(İs)
Konya	-	-	-	-	-	-	80(E) 210(İs)	∞	-	-	80(I) 265(İs)

DİJKSTRA

	İzmit ∞	Yalova ∞	Sakarya ∞	Eskişehir ∞	Düzce ∞	Bolu ∞	Ankara ∞	Nevşehir ∞	Konya ∞	Isparta ∞	Antalya ∞
İstanbul	55(İs)	115(İs)	∞	∞	∞	∞	∞	∞	∞	∞	∞
İzmit	-	55(İz) 110(İs)	30(iz) 85(İs)	∞	∞	∞	∞	∞	∞	∞	∞
Sakarya	-	55(İz) 110(İs)	-	45(S) 130(İs)	40(S) 125(İs)	∞	∞	∞	∞	∞	∞
Yalova	-	-	-	45(S) 130(İs)	40(S) 125(İs)	∞	∞	∞	∞	85(Y) 195(İs)	∞
Düzce	-	-	-	45(S) 130(İs)	-	10(D) 135(İs)	∞	∞	∞	85(Y) 195(İs)	∞
Eskişehir	-	-	-	-	-	10(D) 135(İs)	80(E) 210(İs)	∞	70(E) 200(İs)	55(E) 185(İs)	∞
Bolu	-	-	-	-	-	-	80(E) 210(İs)	∞	70(E) 200(İs)	55(E) 185(İs)	∞
Isparta	-	-	-	-	-	-	80(E) 210(İs)	∞	70(E) 200(İs)	-	80(I) 265(İs)
Konya	-	-	-	-	-	-	80(E) 210(İs)	∞	-	-	80(I) 265(İs)
Ankara	-	-	-	-	-	-	-	105(A) 315(İs)	-	-	80(I) 265(İs)

DİJKSTRA

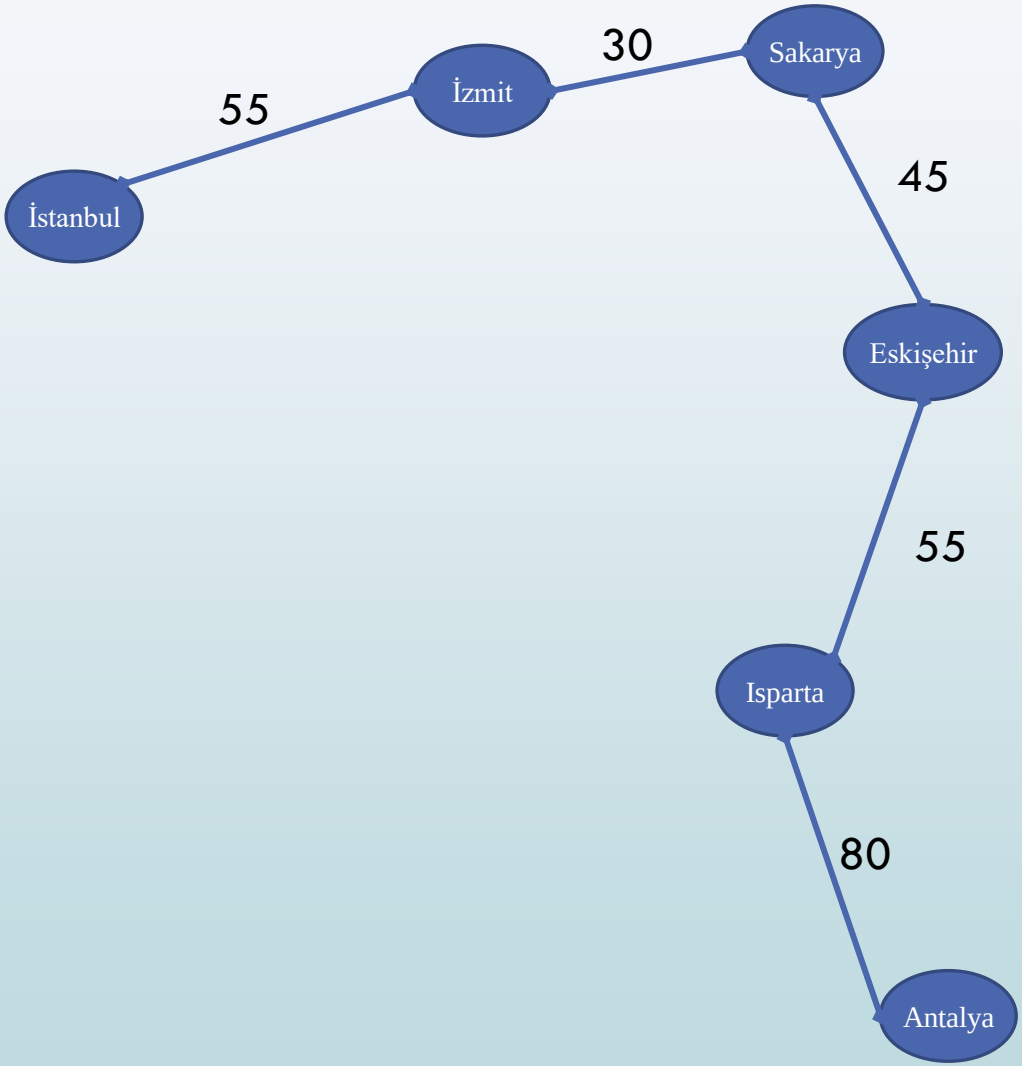
	İzmit ∞	Yalova ∞	Sakarya ∞	Eskişehir ∞	Düzce ∞	Bolu ∞	Ankara ∞	Nevşehir ∞	Konya ∞	Isparta ∞	Antalya ∞
İstanbul	55(İs)	115(İs)	∞	∞	∞	∞	∞	∞	∞	∞	∞
İzmit	-	55(İz) 110(İs)	30(iz) 85(İs)	∞	∞	∞	∞	∞	∞	∞	∞
Sakarya	-	55(İz) 110(İs)	-	45(S) 130(İs)	40(S) 125(İs)	∞	∞	∞	∞	∞	∞
Yalova	-	-	-	45(S) 130(İs)	40(S) 125(İs)	∞	∞	∞	∞	85(Y) 195(İs)	∞
Düzce	-	-	-	45(S) 130(İs)	-	10(D) 135(İs)	∞	∞	∞	85(Y) 195(İs)	∞
Eskişehir	-	-	-	-	-	10(D) 135(İs)	80(E) 210(İs)	∞	70(E) 200(İs)	55(E) 185(İs)	∞
Bolu	-	-	-	-	-	-	80(E) 210(İs)	∞	70(E) 200(İs)	55(E) 185(İs)	∞
Isparta	-	-	-	-	-	-	80(E) 210(İs)	∞	70(E) 200(İs)	-	80(I) 265(İs)
Konya	-	-	-	-	-	-	80(E) 210(İs)	∞	-	-	80(I) 265(İs)
Ankara	-	-	-	-	-	-	-	105(A) 315(İs)	-	-	80(I) 265(İs)
Antalya	-	-	-	-	-	-	-	105(A) 315(İs)	-	-	-

DİJKSTRA

	İzmit ∞	Yalova ∞	Sakarya ∞	Eskişehir ∞	Düzce ∞	Bolu ∞	Ankara ∞	Nevşehir ∞	Konya ∞	Isparta ∞	Antalya ∞
İstanbul	55(İs)	115(İs)	∞	∞	∞	∞	∞	∞	∞	∞	∞
İzmit	-	55(İz) 110(İs)	30(iz) 85(İs)	∞	∞	∞	∞	∞	∞	∞	∞
Sakarya	-	55(İz) 110(İs)	-	45(S) 130(İs)	40(S) 125(İs)	∞	∞	∞	∞	∞	∞
Yalova	-	-	-	45(S) 130(İs)	40(S) 125(İs)	∞	∞	∞	∞	85(Y) 195(İs)	∞
Düzce	-	-	-	45(S) 130(İs)	-	10(D) 135(İs)	∞	∞	∞	85(Y) 195(İs)	∞
Eskişehir	-	-	-	-	-	10(D) 135(İs)	80(E) 210(İs)	∞	70(E) 200(İs)	55(E) 185(İs)	∞
Bolu	-	-	-	-	-	-	80(E) 210(İs)	∞	70(E) 200(İs)	55(E) 185(İs)	∞
Isparta	-	-	-	-	-	-	80(E) 210(İs)	∞	70(E) 200(İs)	-	80(I) 265(İs)
Konya	-	-	-	-	-	-	80(E) 210(İs)	∞	-	-	80(I) 265(İs)
Ankara	-	-	-	-	-	-	-	105(A) 315(İs)	-	-	80(I) 265(İs)
Antalya	-	-	-	-	-	-	-	105(A) 315(İs)	-	-	-
Nevşehir	-	-	-	-	-	-	-	-	-	-	-

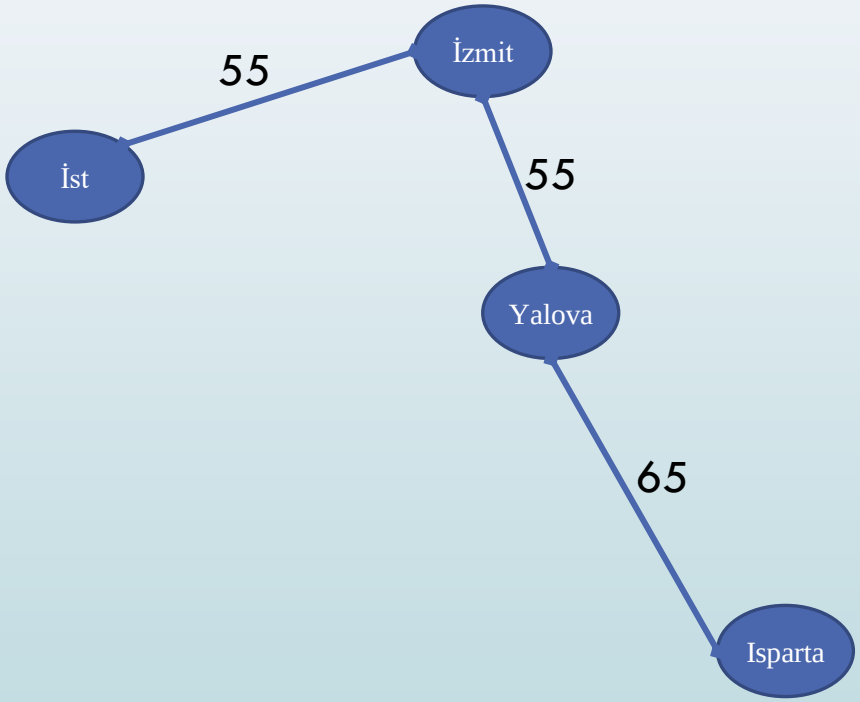
DİJKSTRA

İSTANBUL-ANTALYA=265

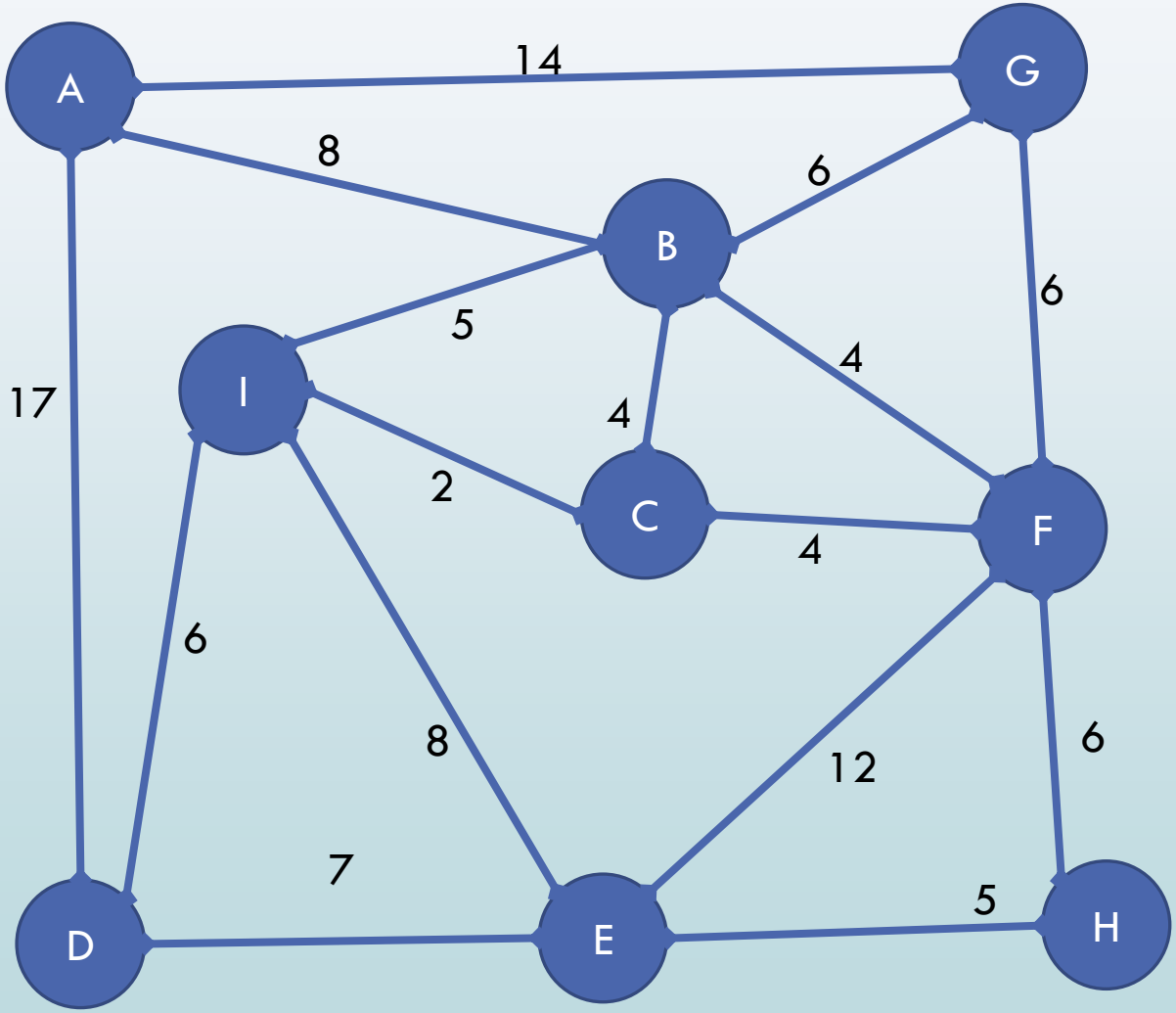


DİJKSTRA

İSTANBUL-ISPARTA=175

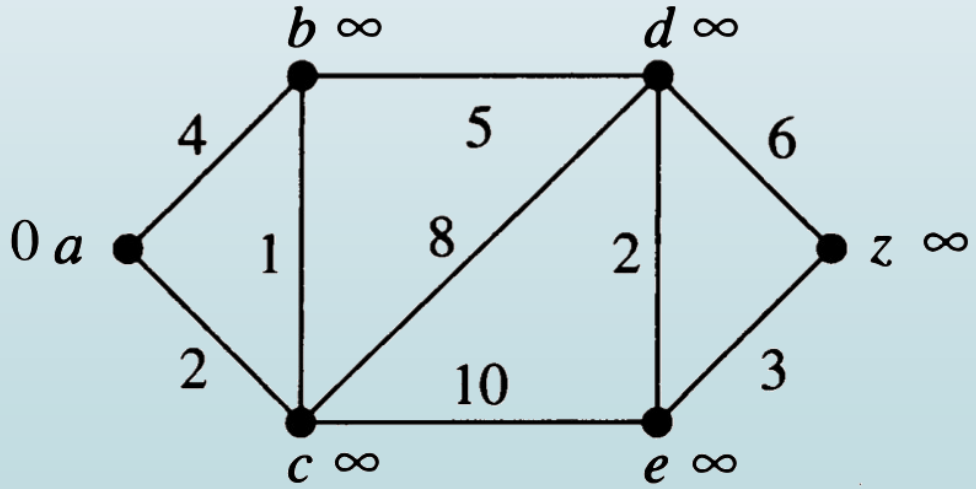


DİJKSTRA ÖRNEK



DİJKSTRA ALGORİTMASI

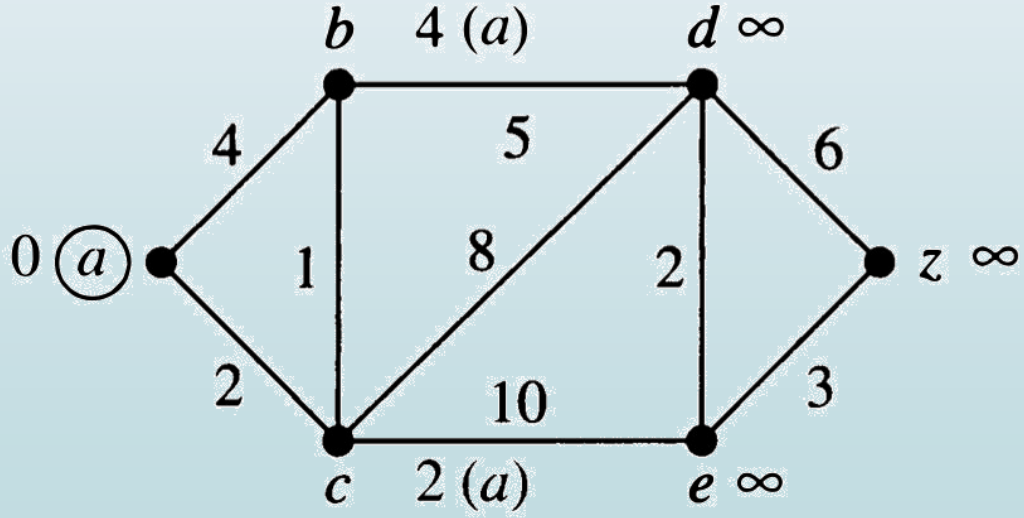
Algoritmanın başlangıç düğümü (node) A olsun. A düğümünün henüz hiçbir düğüme erişimi olmadığını kabul ederek her bir düğüme ulaşımını sonsuz (∞) olarak atıyoruz.



(a)

DİJKSTRA ALGORİTMASI

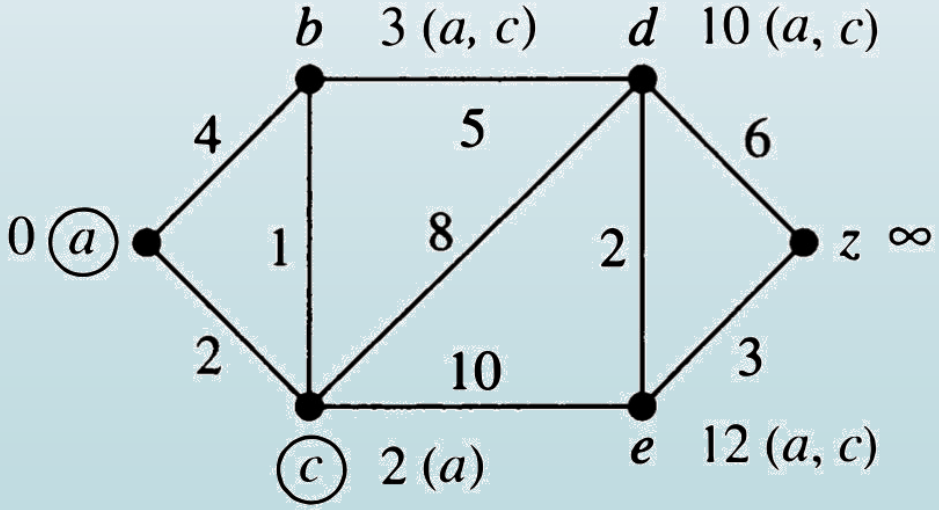
Daha sonra başlangıç düğümünün komşusu olan bütün düğümlere giderek mesafemizi güncelliyoruz. Burada 'b' ve **'c'** düğümlerine erişim sağlıyor.



(b)

DİJKSTRA ALGORİTMASI

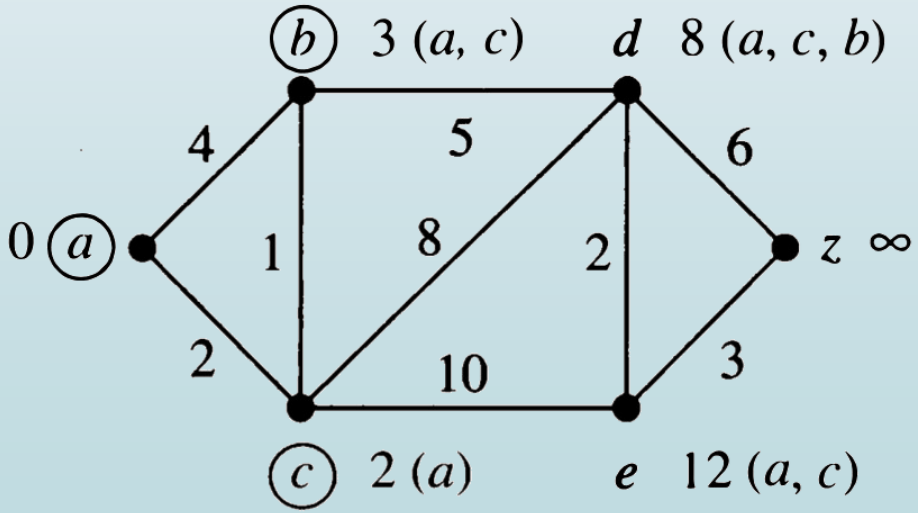
A düğümüyle işlemimiz bitti. Şimdi sıra farketmeksizin yani **ister** 'b' **ister** 'c' düğümünden başlayarak tıpkı 'a' düğümünde yaptığımız gibi mesafeleri güncellememiz gerekmektedir. İlk güncellememize C düğümünden başlarsak;



(c)

DİJKSTRA ALGORİTMASI

Şimdi burada dikkat edilmesi gereken husus a-b arası 4 birim uzaklıktaydı. Artık C düğümünü güncellediğimiz için a-b düğümleri arasını c üzerinden(a-c-b) gidersek 3 birim uzaklıkta olduğu için 'b' düğümüne ulaşımımız 3 olarak güncellemimiz gerekiyor.

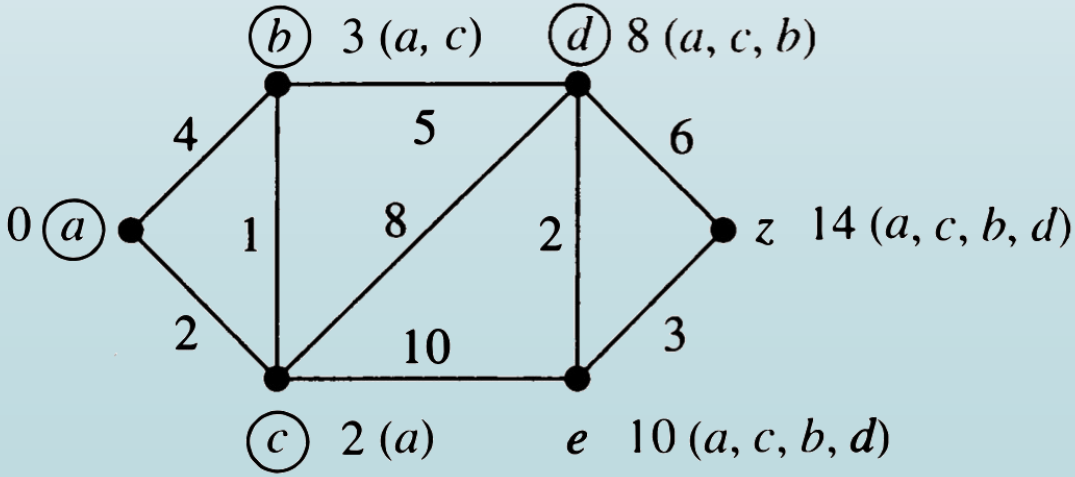


(d)

DİJKSTRA ALGORİTMASI

Sıra geldi ‘b’ düğümünün komşu düğümlerini güncellemeye. Komşu olarak c ve d düğümleri var. C düğümüne ulaşımımız a-b-c üzerinden 4+1=5 birim ama biz a-c arası mesafemiz 2 olduğu için güncellememize gerek yoktur.Çünkü daha fazla maliyetlidir.

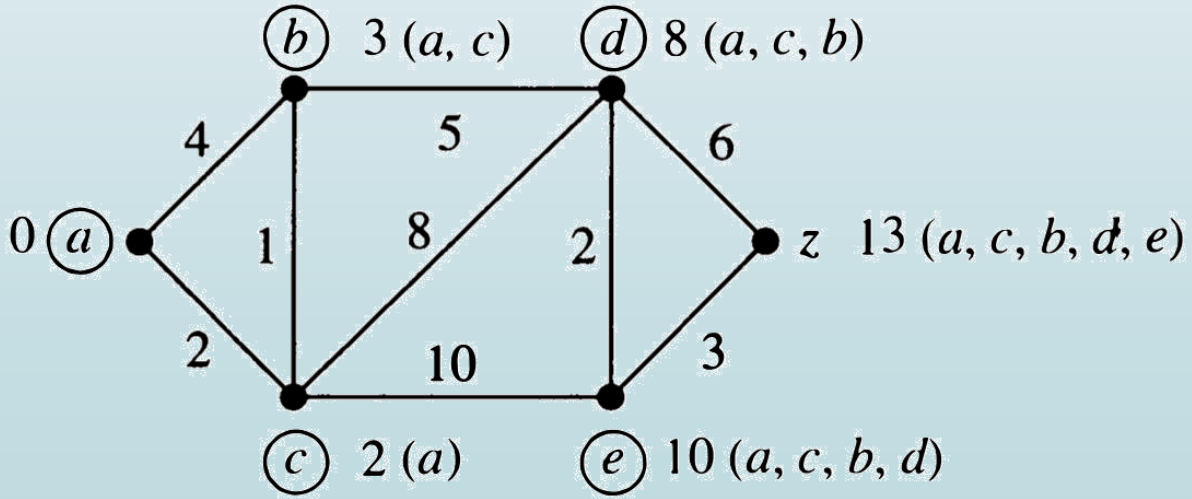
B düğümünün diğer komşusu olan d düğümüne mesafesi a-c-b-d üzeriden 2+1+5=8’dir. Fakat a-c-d üzerinden mesafe 2+8=10 olduğu için **güncellememiz gerekecektir.**



(e)

DİJKSTRA ALGORİTMASI

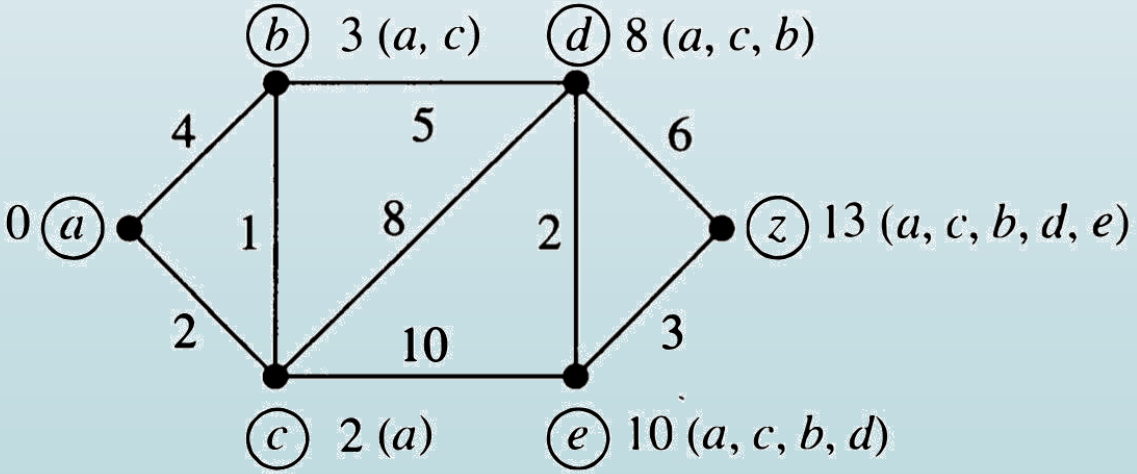
Şimdi ‘d’ düğümünün komşularına bakalım. E düğümüne a-c-e üzerinden ulaşım 10+2=12 birim uzaklıktaydı, fakat a-c-b-d-e üzerinden ulaşım **2+1+5+2=10** birim uzaklıkta olacaktır. ve ‘e’ düğümüne bundan başka daha az mesafede ulaşamayız.



(f)

DİJKSTRA ALGORİTMASI

E düğümünün komşu düğümlerini güncelleyelim. Komşu düğüm olarak güncellememiz gerek tek düğüm kaldı o da ‘z’ düğümü. Bir önceki örnekte d üzerinden maliyetimiz 8+6=14 idi. Fakat ‘e’ düğümü üzerinden 10+3=13 birim uzaklıkta olduğu için güncellememiz gerekecektir.



(g)

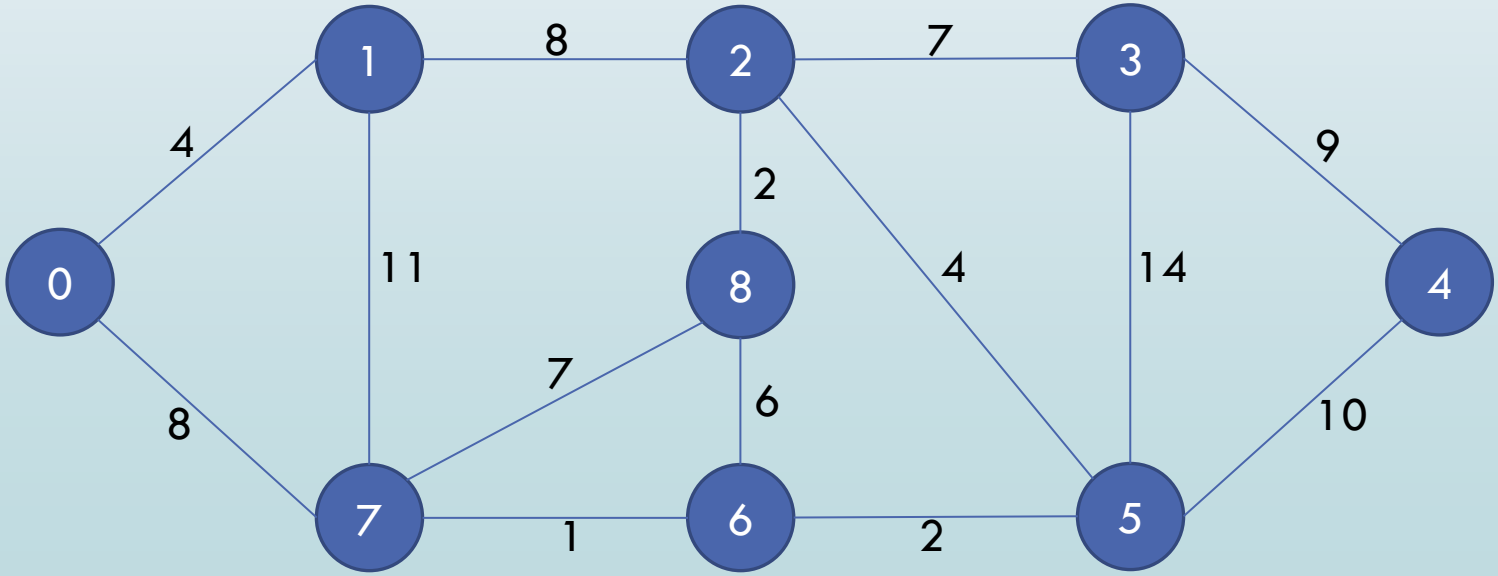
DİJKSTRA ALGORİTMASI

Artık mesafe olarak güncellememiz gereken bir düğüm kalmadığı için algoritma bu graf için sonlanır.

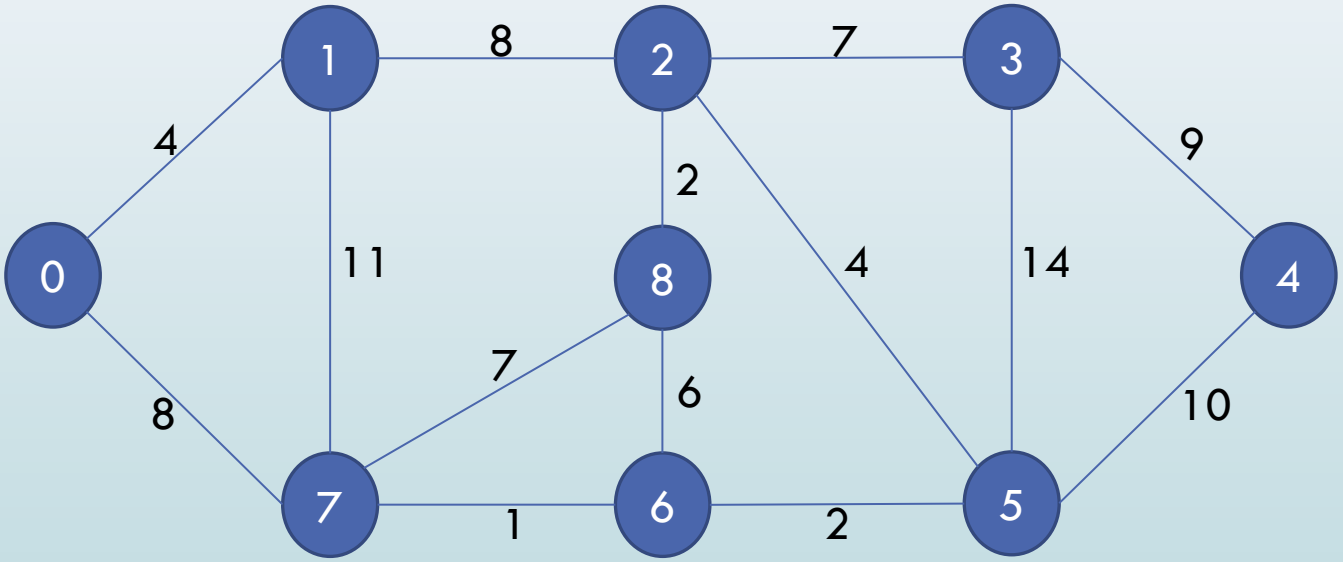
«Bu algoritmanın amacı: Bir düğümden başlayarak o düğümün tüm graf üzerinde bulunan düğümlerine en kısa mesafede ulaşmasını garanti eder. Ayrıca bundan daha kısa bir yol bulunmayacağını iddia eder. Eşit mesafe olabilir.»

KRUSKAL ALGORİTHM (MİN. SPANNİNG TREE)

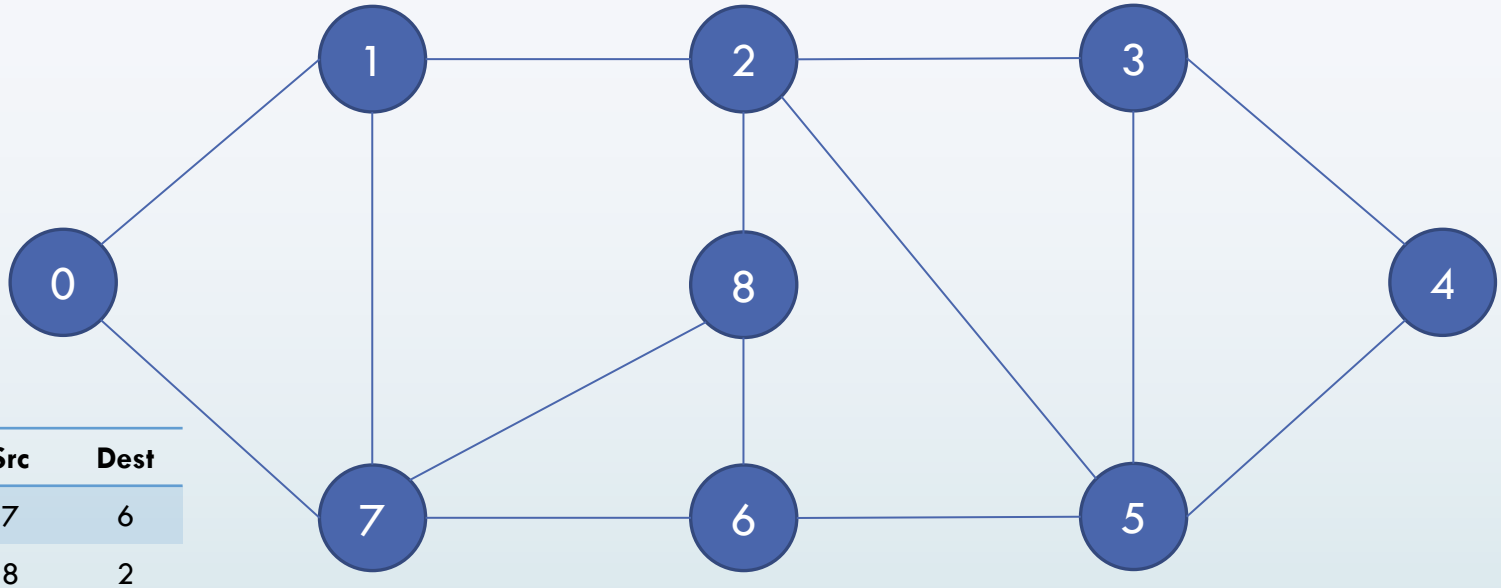
Kruskal algoritması bağlı düğümler içerisinde en kısa şekilde tüm düğümleri dolaşmayı sağlar (Minimum Spanning Tree Solving).



KRUSKAL ALGORİTHM (MİN. SPANNİNG TREE)



Weight	Src	Dest
1	7	6
2	8	2
2	6	5
4	0	1
4	2	5
6	8	6
7	2	3
7	7	8
8	0	7
8	1	2
9	3	4
10	5	4
11	1	7
14	3	5



Weight	Src	Dest
1	7	6
2	8	2
2	6	5
4	0	1
4	2	5
6	8	6
7	2	3
7	7	8
8	0	7
8	1	2
9	3	4
10	5	4
11	1	7
14	3	5

1. 7-6:

2. 8-2:

3. 6-5:

4. *0-1:*

5. *2-5:*

6. *8-6:* *Döngü var*

7. *2-3:*

8. *7-8:* *Döngü var*

9. *0-7:*

10. *1-2:* *Döngü var*

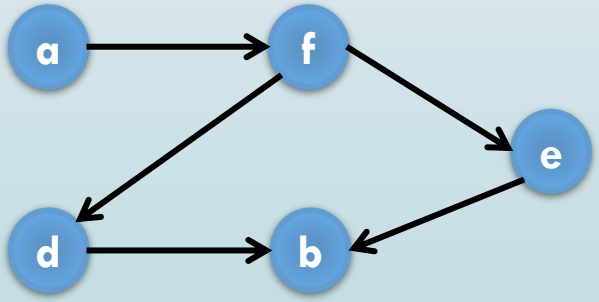
11. *3-4:* *SON*

TOPLOJİK SIRALAMA

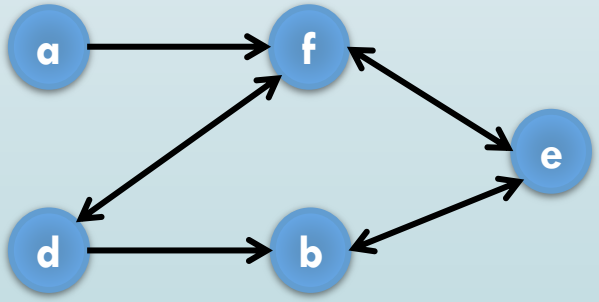
Bu algoritma yönlü döngüsüz graflar üzerinde tanımlanan bir algoritmadır.

Hatırlayalım:

Döngüsüz



Döngülü



ÖRNEK:

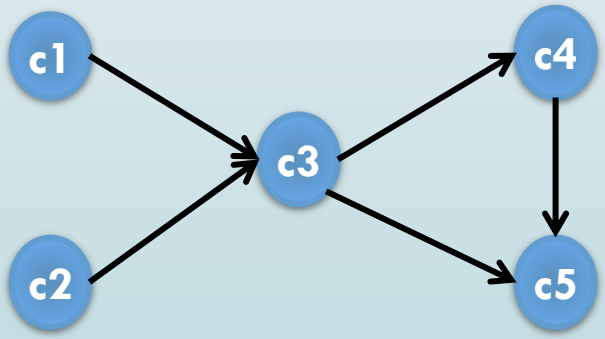
C1,C2,C3,C4,C5 alınması gereken 5 adet ders olsun. Buna bağlı olarak;

- C1 ve C2 nin ön şartı yok,
- C3, C1 ve C2 yi ön şart alıyor,
- C4, C3 ü ön şart alıyor,
- C5, C3 ve C4 ü ön şart alıyor,

Bu şartlara bağlı olarak öğrencilerin dersleri alma sırasını oluşturalım.

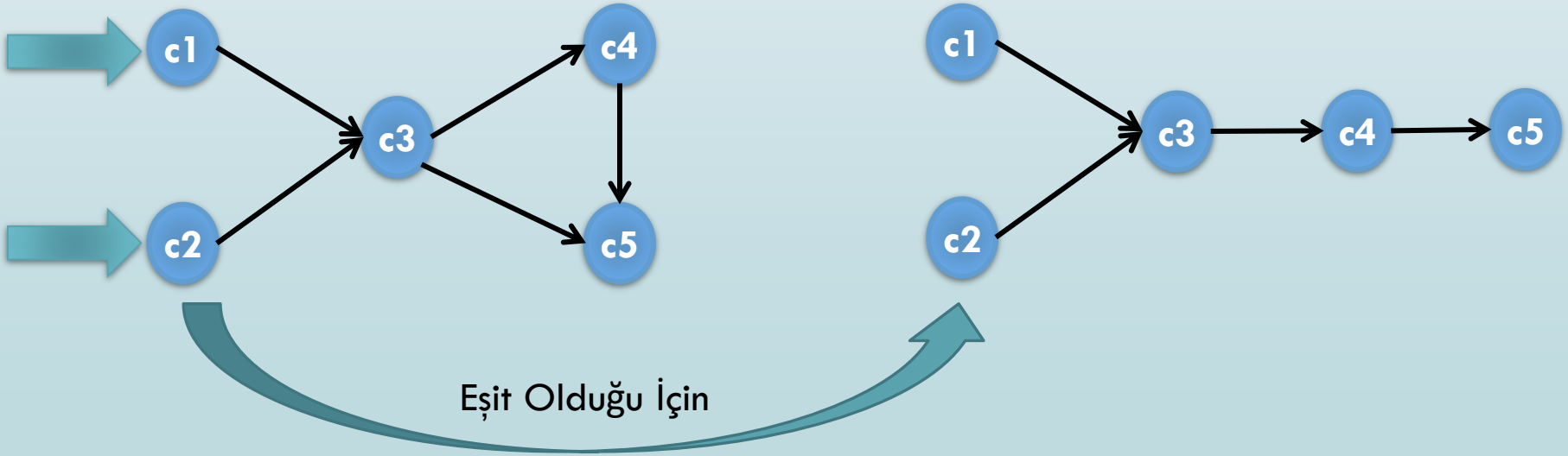
ÖRNEK: (DEVAM...)

Dersleri grafların köşeleri, ön şartları da yönlü kenarlar kabul ederek, bir graf oluşturalım;



ÖRNEK: (DEVAM...)

NASIL SIRALANACAK: İndirge ve Çöz yöntemiyle kendisine gelen ve kenar bulunmayan bir düğümü kaynak olarak seçip graftan çıkarıp işlemler (n-1) düğümlü graf ile devam ettirilir. Graf ta hiç düğüm kalmayınca çözüm tamamlanmış olur.



BİTTİ

YARARLANILAN KAYNAKLAR

- **Ders Kitabı:**
  - Data Structures through JAVA, V.V.Muniswamy
- **Yardımcı Okumalar:**
  - Algorithms, Robert Sedgewick
  - Yrd. Doç. Dr. Deniz KILINÇ, Celal Bayar Üniversitesi
  - http://ensarkarabudak.com/muhendislik/dijkstra-en-kisa-yol-algoritmasi-dijkstras-shortest-path-algorithm/
  - http://www.geeksforgeeks.org/greedy-algorithms-set-2-kruskals-minimum-spanning-tree-mst/