T.C.
KIRIKKALE
ÜNİVERSİTESİ
BİLGİSAYAR
MÜHENDİSLİĞİ

AĞ OPTİMİZASYONU

DR. EVRENCAN ÖZCAN

DERS İÇERİĞİ

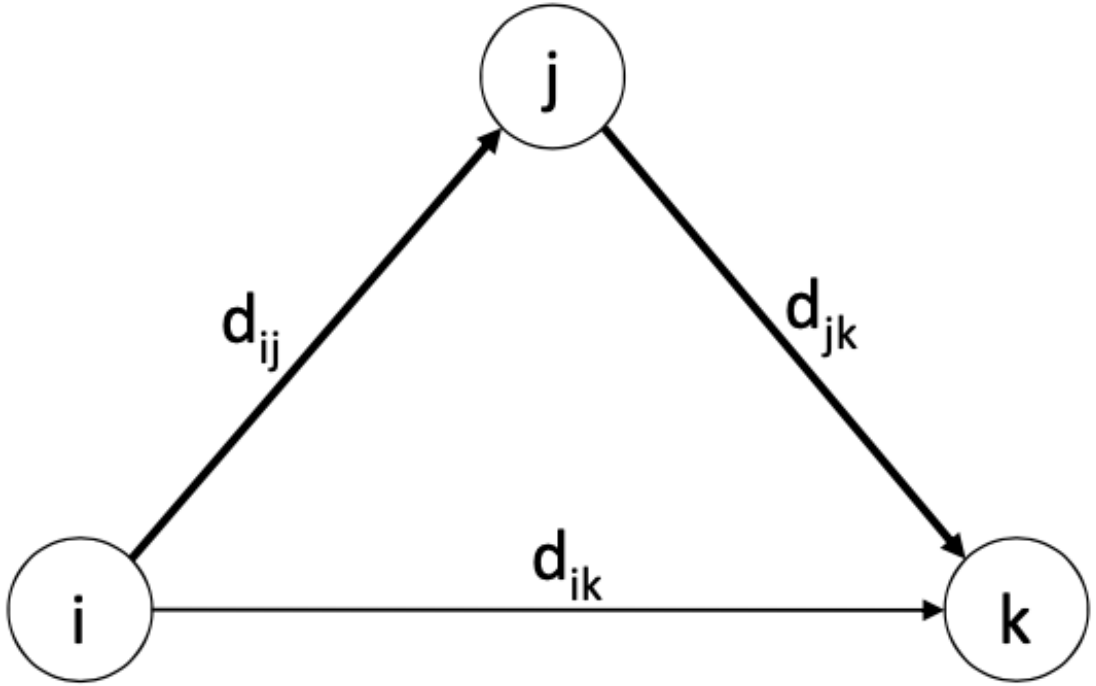
- AĞ OPTİMİZASYONUNA GİRİŞ
  - OPTİMİZASYON KAVRAMI
  - TEMEL ŞEBEKE KAVRAMLARI
  - ŞEBEKE OPTİMİZASYONUNUN UYGULAMA ALANLARI
- MİNİMUM YAYILAN AĞAÇ PROBLEMİ
- EN KISA YOL PROBLEMİ
- MAKSİMUM AKIŞ PROBLEMİ
- PROJE YÖNETİMİ
  - KRİTİK YOL METODU (CPM)
  - PROJE DEĞERLENDİRME VE GÖZDEN GEÇİRME TEKNİĞİ (PERT)
  - PROJE PLANLAMASINDA ZAMAN-MALİYET İLİŞKİSİ

EN KISA YOL PROBLEMİ – FLOYD ALGORİTMASI

- Floyd algoritması Dijkstra algoritmasından daha geneldir. Çünkü şebekedeki herhangi iki düğüm arasındaki en kısa yolu belirler.
- Algoritma, n düğümlü şebekeyi n satırlı ve n sütunlu kare matris olarak gösterir. Matrisin (i, j) elemanı, i. düğümden j. düğüme olan dij uzaklığını verir.
- i doğrudan j'ye bağlı ise dij sonlu, bağlı değilse sonsuzdur.

EN KISA YOL PROBLEMİ – FLOYD ALGORİTMASI

Eğer $d_{ij} + d_{jk} \leq d_{ik}$ ise, i'den başlayıp j'den geçerek k'ya ulaşmak daha kısadır. Bu durumda i'den k'ya doğrudan yolu, i ⇨ j ⇨ k dolaylı yolu ile değiştirmek optimumu verir. Bu üçlü işlem değişimi, aşağıdaki adımlar kullanılarak şebekeye sistematik olarak uygulanır.



EN KISA YOL PROBLEMİ – FLOYD ALGORİTMASI

$D_0 =$

	1	2	. . .	j	. . .	n
1	–	d_{12}		d_{1j}		d_{1n}
2	d_{21}	–	. . .	d_{2j}	. . .	d_{2n}
.	.	.	.	.	.	.
.	.	.	.	.	.	.
.	.	.	.	.	.	.
i	d_{i1}	d_{i2}	. . .	d_{ij}	. . .	d_{in}
.	.	.	.	.	.	.
.	.	.	.	.	.	.
.	.	.	.	.	.	.
n	d_{n1}	d_{n2}	. . .	d_{nj}	. . .	–

D_0 = Başlangıç uzaklık matrisi

$S_0 =$

	1	2	. . .	j	. . .	n
1	–	2		j		n
2	1	–	. . .	j	. . .	n
.	.	.	.	.	.	.
.	.	.	.	.	.	.
.	.	.	.	.	.	.
i	1	2	. . .	j	. . .	n
.	.	.	.	.	.	.
.	.	.	.	.	.	.
.	.	.	.	.	.	.
n	1	2	. . .	j	. . .	–

S_0 = Düğüm sırası matrisi

EN KISA YOL PROBLEMİ – FLOYD ALGORİTMASI

k. Genel Adım:

k. satırı ve k. sütunu anahtar (pivot) satır ve anahtar (pivot) sütun olarak tanımla. Tüm i ve j'ler için D_{k-1}'deki her bir d_{ij} elemanına üçlü işlemi uygula. Eğer burada

$d_{ik} + d_{kj} \leq d_{ij}$, $(i \neq k, j \neq k$ ve $i \neq j)$

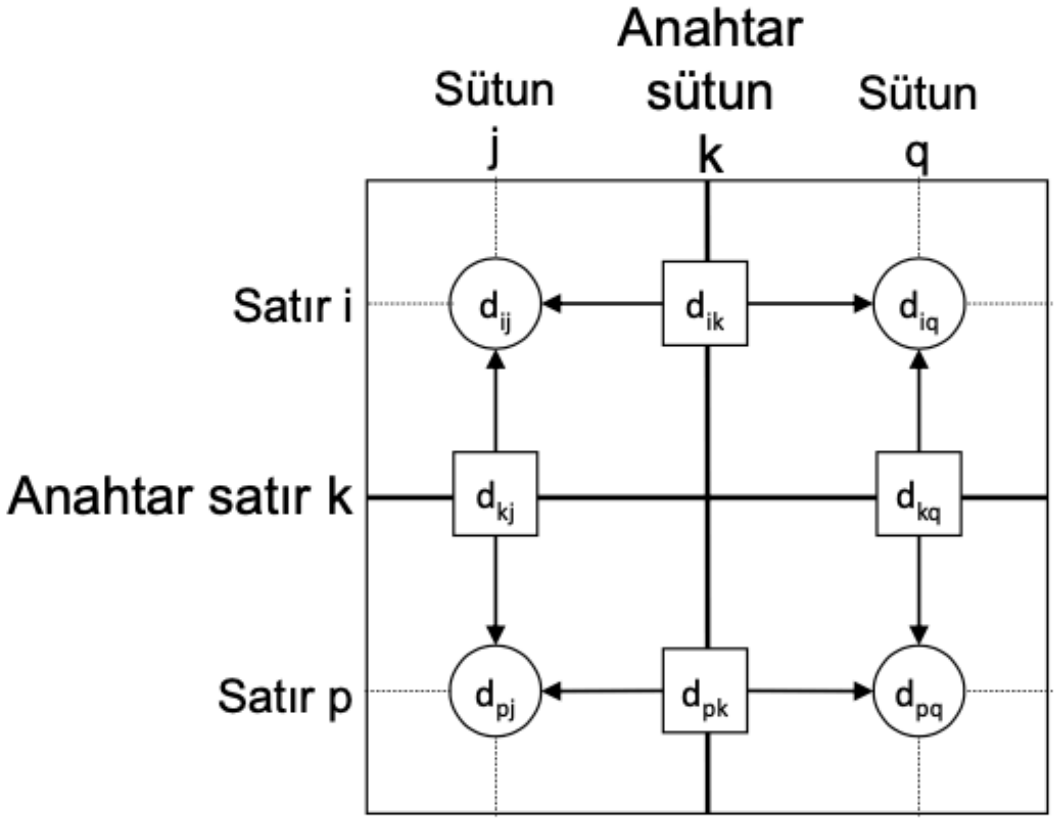
sağlanıyorsa aşağıdaki değişiklikleri yap.

(a) D_{k-1}'de d_{ij}'yi $d_{ik} + d_{kj}$ ile değiştirerek D_k'yı oluştur.

(b) S_{k-1}'de S_{ij}'yi k ile değiştirerek S_k'yı oluştur. k = k + 1 olarak belirle ve k. adımı tekrar et.

EN KISA YOL PROBLEMİ – FLOYD ALGORİTMASI

k. Genel Adım: Eğer anahtar satır ve sütundaki karelerle gösterilen elemanların toplamı, daireyle gösterilen ilgili arakesit elemanından küçükse, arakesit uzaklığı yerine anahtar uzaklıkların toplamını yazmak optimumdur.



EN KISA YOL PROBLEMİ – FLOYD ALGORİTMASI

k. Genel Adım:

n adım sonra i ve j düğümleri arasındaki en kısa yolu D_n ve S_n matrislerinden aşağıdaki kuralları kullanarak belirleriz:

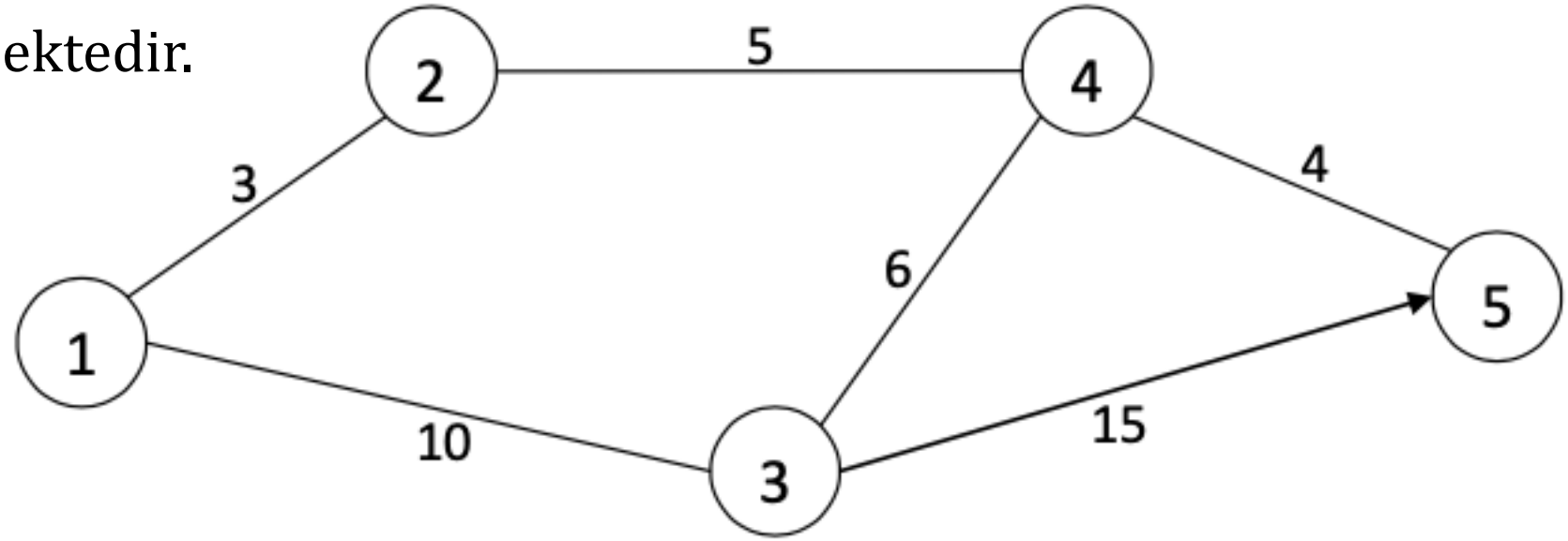
1. D_n'deki d_{ij}, i ve j düğümleri arasındaki en kısa yolu verir.

2. S_n'deki i → k → j yolunu veren $k = S_{ij}$ ara düğümünü belirle. **Eğer $S_{ik} = k$ ve $S_{kj} = j$ ise dur**; yolun tüm ara düğümleri bulunmuştur. Aksi halde i ve k düğümleri ve k ve j düğümleri arasındaki prosedürü tekrar et.

EN KISA YOL PROBLEMİ – FLOYD ALGORİTMASI

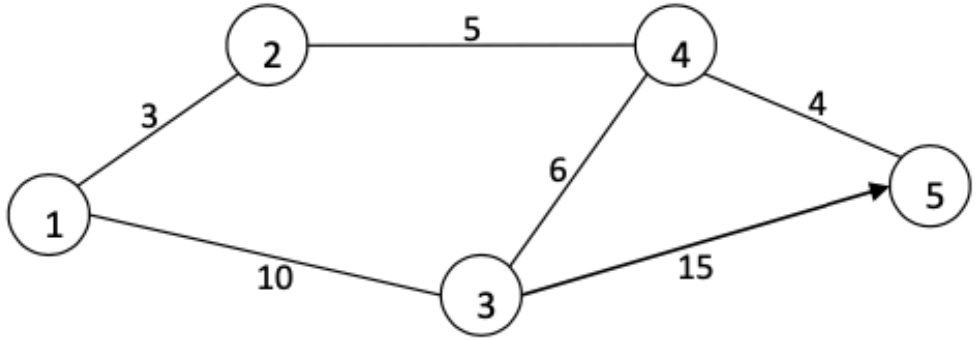
ÖRNEK

Aşağıdaki şebeke için her iki düğüm arasındaki en kısa yolları bulun. Uzaklıklar km cinsinden bağlantıların üzerinde belirtilmiştir. (3, 5) bağlantısı 5. düğümden 3. düğüme trafiğin olmadığı tek yönlü bir bağlantıdır. Diğer düğümlerde iki yönde de trafik akışına izin verilmektedir.



EN KISA YOL PROBLEMİ – FLOYD ALGORİTMASI

0. Yineleme



D_0

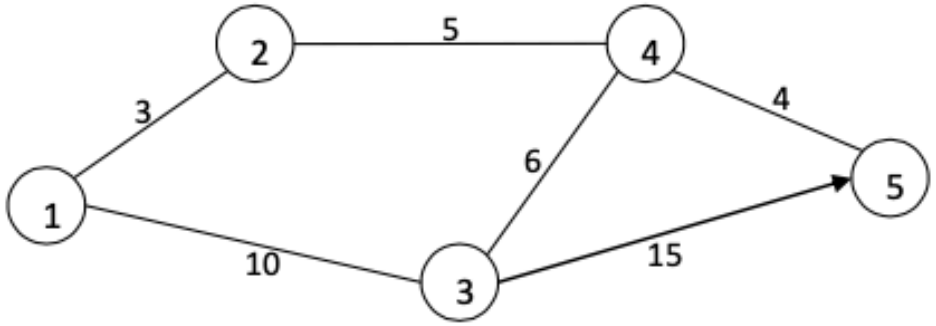
	1	2	3	4	5
1	–	3	10	∞	∞
2	3	–	∞	5	∞
3	10	∞	–	6	15
4	∞	5	6	–	4
5	∞	∞	∞	4	–

S_0

	1	2	3	4	5
1	–	2	3	4	5
2	1	–	3	4	5
3	1	2	–	4	5
4	1	2	3	–	5
5	1	2	3	4	–

EN KISA YOL PROBLEMİ – FLOYD ALGORİTMASI

1. Yineleme



D_1

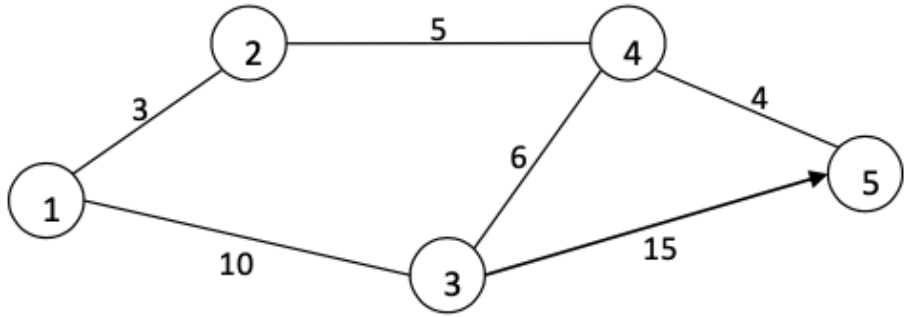
	1	2	3	4	5
1	–	3	10	∞	∞
2	3	–	**13**	5	∞
3	10	**13**	–	6	15
4	∞	5	6	–	4
5	∞	∞	∞	4	–

S_1

	1	2	3	4	5
1	–	2	3	4	5
2	1	–	**1**	4	5
3	1	**1**	–	4	5
4	1	2	3	–	5
5	1	2	3	4	–

EN KISA YOL PROBLEMİ – FLOYD ALGORİTMASI

2. Yineleme



D_2

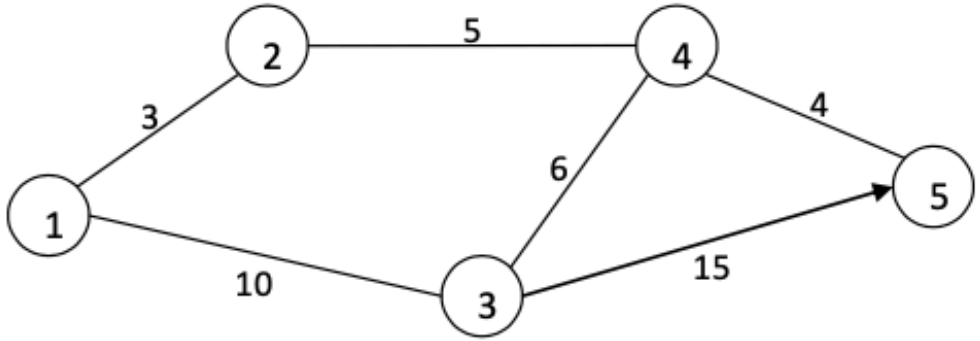
	1	2	3	4	5
1	–	3	10	**8**	∞
2	3	–	13	5	∞
3	10	13	–	6	15
4	**8**	5	6	–	4
5	∞	∞	∞	4	–

S_2

	1	2	3	4	5
1	–	2	3	**2**	5
2	1	–	1	4	5
3	1	1	–	4	5
4	**2**	2	3	–	5
5	1	2	3	4	–

EN KISA YOL PROBLEMİ – FLOYD ALGORİTMASI

3. Yineleme



D_3

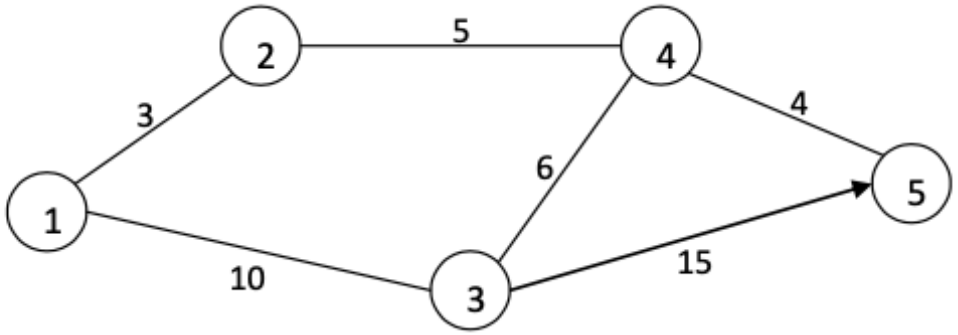
	1	2	3	4	5
1	–	3	10	8	**25**
2	3	–	13	5	**28**
3	10	13	–	6	15
4	8	5	6	–	4
5	∞	∞	∞	4	–

S_3

	1	2	3	4	5
1	–	2	3	2	**3**
2	1	–	1	4	**3**
3	1	1	–	4	5
4	2	2	3	–	5
5	1	2	3	4	–

EN KISA YOL PROBLEMİ – FLOYD ALGORİTMASI

4. Yineleme



D_4

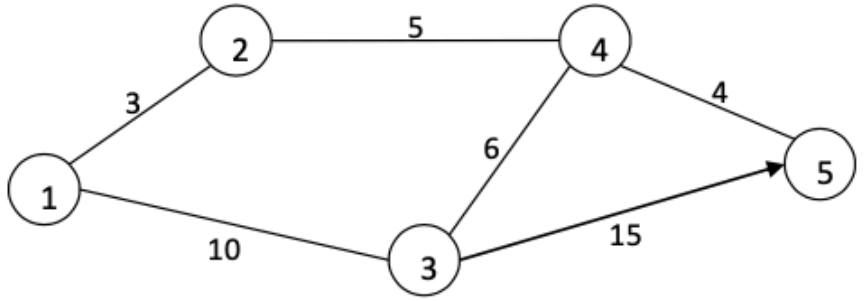
	1	2	3	4	5
1	–	3	10	8	**12**
2	3	–	**11**	5	**9**
3	10	**11**	–	6	**10**
4	8	5	6	–	4
5	**12**	**9**	**10**	4	–

S_4

	1	2	3	4	5
1	–	2	3	2	**4**
2	1	–	**4**	4	**4**
3	1	**4**	–	4	**4**
4	2	2	3	–	5
5	**4**	**4**	**4**	4	–

EN KISA YOL PROBLEMİ – FLOYD ALGORİTMASI

4. Yineleme



D_4

	1	2	3	4	5
1	–	3	10	8	**12**
2	3	–	**11**	5	**9**
3	10	**11**	–	6	**10**
4	8	5	6	–	4
5	**12**	**9**	**10**	4	–

S_4

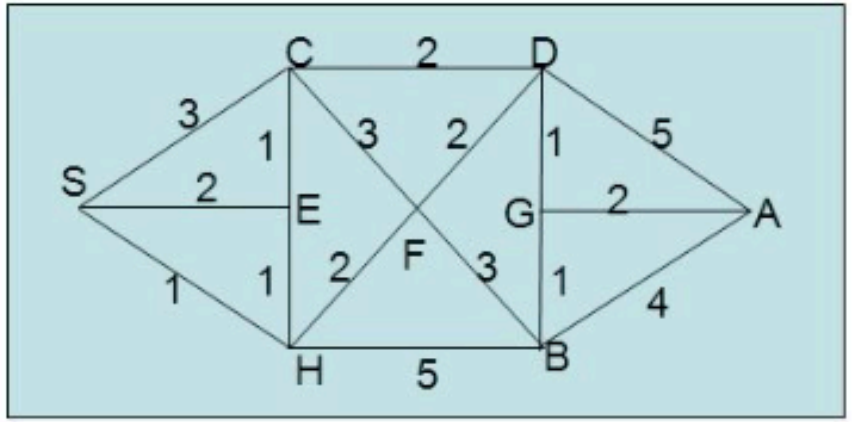
	1	2	3	4	5
1	–	2	3	2	**4**
2	1	–	**4**	4	**4**
3	1	**4**	–	4	**4**
4	2	2	3	–	5
5	**4**	**4**	**4**	4	–

Örneğin 1 düğümünden 5 düğümüne en kısa uzaklık $d_{15} = 12$'dir.

Yol: $S_{15} = 4$; $S_{14} = 2$; $S_{12} = 2$

O halde: 1'den 5'e yol 1 → 2 → 4 → 5

EN KISA YOL PROBLEMİ – FLOYD ALGORİTMASI



0. yineleme: D_0:

	A	B	C	D	E	F	G	H	S
A	-	4	∞	5	∞	∞	2	∞	∞
B	4	-	∞	∞	∞	3	1	5	∞
C	∞	∞	-	2	1	3	∞	∞	3
D	5	∞	2	-	∞	2	1	∞	∞
E	∞	∞	1	∞	-	∞	∞	1	2
F	∞	3	3	2	∞	-	∞	2	∞
G	2	1	∞	1	∞	∞	-	∞	∞
H	∞	5	∞	∞	1	2	∞	-	1
S	∞	∞	3	∞	2	∞	∞	1	-

S_0:

	A	B	C	D	E	F	G	H	S
A	-	2	3	4	5	6	7	8	9
B	1	-	3	4	5	6	7	8	9
C	1	2	-	4	5	6	7	8	9
D	1	2	3	-	5	6	7	8	9
E	1	2	3	4	-	6	7	8	9
F	1	2	3	4	5	-	7	8	9
G	1	2	3	4	5	6	-	8	9
H	1	2	3	4	5	6	7	-	9
S	1	2	3	4	5	6	7	8	-

1. yineleme:

D1:

	A	B	C	D	E	F	G	H	S
A	-	4	∞	5	∞	∞	2	∞	∞
B	4	-	∞	9	∞	3	1	5	∞
C	∞	∞	-	2	1	3	∞	∞	3
D	5	9	2	-	∞	2	1	∞	∞
E	∞	∞	1	∞	-	∞	∞	1	2
F	∞	3	3	2	∞	-	∞	2	∞
G	2	1	∞	1	∞	∞	-	∞	∞
H	∞	5	∞	∞	1	2	∞	-	1
S	∞	∞	3	∞	2	∞	∞	1	-

S1:

	A	B	C	D	E	F	G	H	S
A	-	2	3	4	5	6	7	8	9
B	1	-	3	1	5	6	7	8	9
C	1	2	-	4	5	6	7	8	9
D	1	1	3	-	5	6	7	8	9
E	1	2	3	4	-	6	7	8	9
F	1	2	3	4	5	-	7	8	9
G	1	2	3	4	5	6	-	8	9
H	1	2	3	4	5	6	7	-	9
S	1	2	3	4	5	6	7	8	-

2. yineleme:

D2:

	A	B	C	D	E	F	G	H	S
A	-	4	∞	5	∞	7	2	9	∞
B	4	-	∞	9	∞	3	1	5	∞
C	∞	∞	-	2	1	3	∞	∞	3
D	5	9	2	-	∞	2	1	14	∞
E	∞	∞	1	∞	-	∞	∞	1	2
F	7	3	3	2	∞	-	4	2	∞
G	2	1	∞	1	∞	4	-	6	∞
H	9	5	∞	14	1	2	6	-	1
S	∞	∞	3	∞	2	∞	∞	1	-

S2:

	A	B	C	D	E	F	G	H	S
A	-	2	3	4	5	2	7	2	9
B	1	-	3	1	5	6	7	8	9
C	1	2	-	4	5	6	7	8	9
D	1	1	3	-	5	6	7	2	9
E	1	2	3	4	-	6	7	8	9
F	2	2	3	4	5	-	2	8	9
G	1	2	3	4	5	2	-	2	9
H	2	2	3	2	5	6	2	-	9
S	1	2	3	4	5	6	7	8	-

3. yineleme:

D3:

	A	B	C	D	E	F	G	H	S
A	-	4	∞	5	∞	7	2	9	∞
B	4	-	∞	9	∞	3	1	5	∞
C	∞	∞	-	2	1	3	∞	∞	3
D	5	9	2	-	3	2	1	14	5
E	∞	∞	1	3	-	4	∞	1	2
F	7	3	3	2	4	-	4	2	6
G	2	1	∞	1	∞	4	-	6	∞
H	9	5	∞	14	1	2	6	-	1
S	∞	∞	3	5	2	6	∞		-

S3:

	A	B	C	D	E	F	G	H	S
A	-	2	3	4	5	2	7	2	9
B	1	-	3	1	5	6	7	8	9
C	1	2	-	4	5	6	7	8	9
D	1	1	3	-	3	6	7	2	3
E	1	2	3	3	-	3	7	8	9
F	2	2	3	4	3	-	2	8	3
G	1	2	3	4	5	2	-	2	9
H	2	2	3	2	5	6	2	-	9
S	1	2	3	3	5	3	7	8	-

4. yineleme:

D4:

	A	B	C	D	E	F	G	H	S
A	-	4	7	5	8	7	2	9	10
B	4	-	11	9	12	3	1	5	14
C	7	11	-	2	1	3	3	16	3
D	5	9	2	-	3	2	1	14	5
E	8	12	1	3	-	4	4	1	2
F	7	3	3	2	4	-	3	2	6
G	2	1	3	1	4	3	-	6	6
H	9	5	16	14	1	2	6	-	1
S	10	14	3	5	2	6	6	1	-

S4:

	A	B	C	D	E	F	G	H	S
A	-	2	4	4	4	2	7	2	4
B	1	-	4	1	4	6	7	8	4
C	4	4	-	4	5	6	4	4	9
D	1	1	3	-	3	6	7	2	3
E	4	4	3	3	-	3	4	8	9
F	2	2	3	4	3	-	4	8	3
G	1	2	4	4	4	4	-	2	4
H	2	2	4	2	5	6	2	-	9
S	4	4	3	3	5	3	4	8	-

5. yineleme:

D_5:

	A	B	C	D	E	F	G	H	S
A	-	4	7	5	8	7	2	9	10
B	4	-	11	9	12	3	1	5	14
C	7	11	-	2	1	3	3	2	3
D	5	9	2	-	3	2	1	4	5
E	8	12	1	3	-	4	4	1	2
F	7	3	3	2	4	-	3	2	6
G	2	1	3	1	4	3	-	5	6
H	9	5	2	4	1	2	5	-	1
S	10	14	3	5	2	6	6	1	-

S_5:

	A	B	C	D	E	F	G	H	S
A	-	2	4	4	4	2	7	2	4
B	1	-	4	1	4	6	7	8	4
C	4	4	-	4	5	6	4	5	9
D	1	1	3	-	3	6	7	5	3
E	4	4	3	3	-	3	4	8	9
F	2	2	3	4	3	-	4	8	3
G	1	2	4	4	4	4	-	5	4
H	2	2	5	5	5	6	5	-	9
S	4	4	3	3	5	3	4	8	-

6. yineleme:

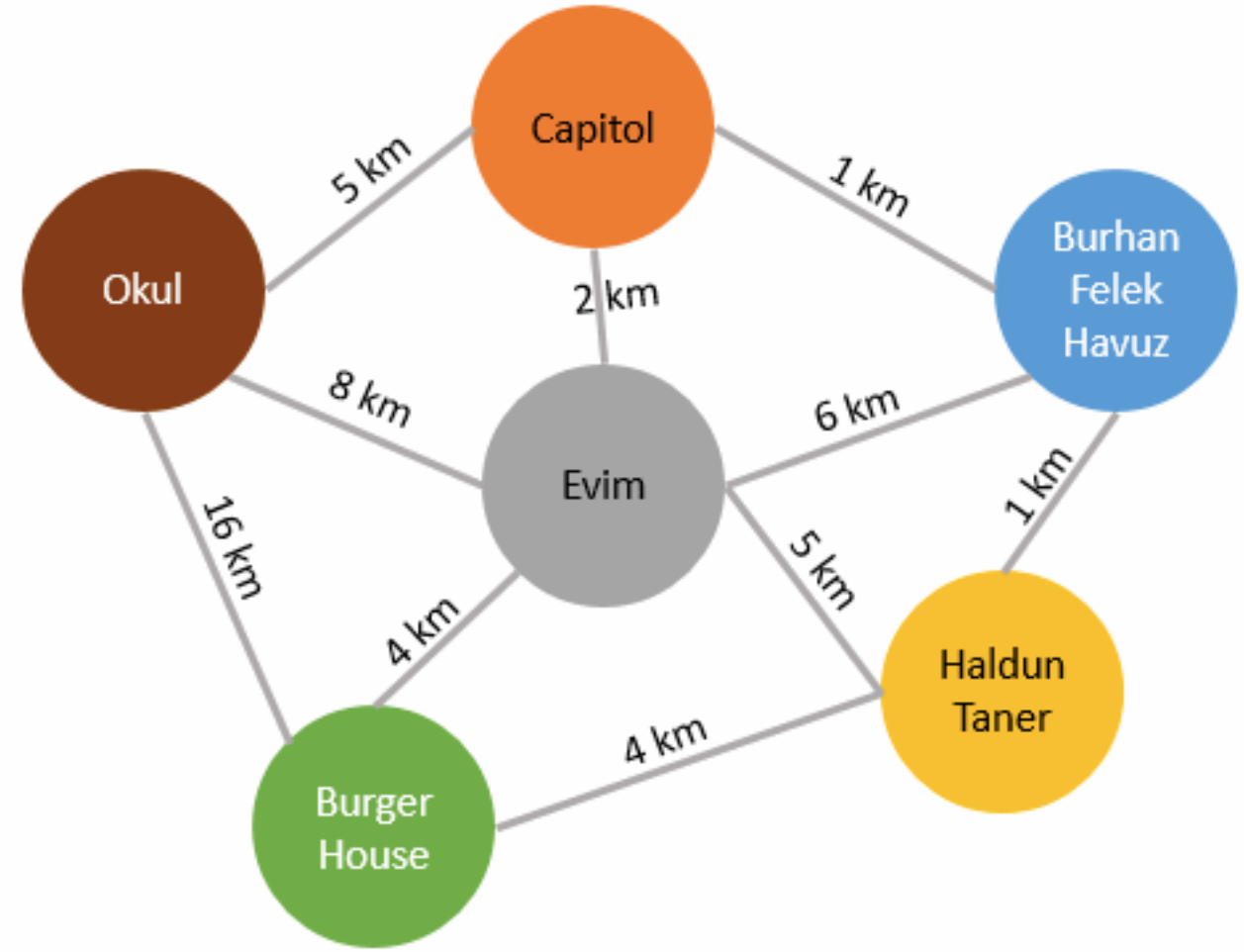
D_6:

	A	B	C	D	E	F	G	H	S
A	-	4	7	5	8	7	2	9	10
B	4	-	6	5	7	3	1	5	9
C	7	6	-	2	1	3	3	2	3
D	5	5	2	-	3	2	1	4	5
E	8	7	1	3	-	4	4	1	2
F	7	3	3	2	4	-	3	2	6
G	2	1	3	1	4	3	-	5	6
H	9	5	2	4	1	2	5	-	1
S	10	9	3	5	2	6	6	1	-

S_6:

	A	B	C	D	E	F	G	H	S
A	-	2	4	4	4	2	7	2	4
B	1	-	6	6	6	6	7	8	6
C	4	6	-	4	5	6	4	5	9
D	1	6	3	-	3	6	7	5	3
E	4	6	3	3	-	3	4	8	9
F	2	2	3	4	3	-	4	8	3
G	1	2	4	4	4	4	-	5	4
H	2	2	5	5	5	6	5	-	9
S	4	6	3	3	5	3	4	8	-

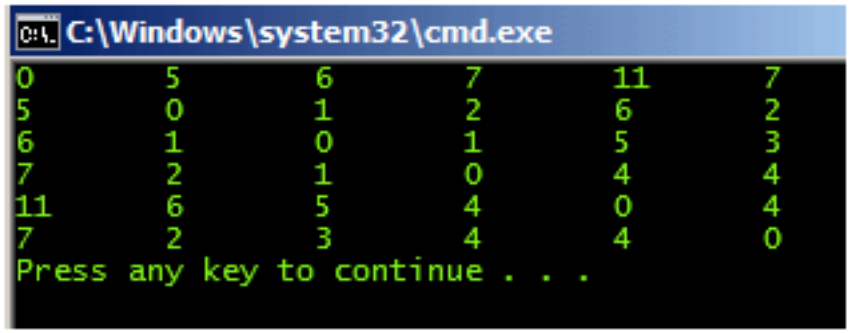
EN KISA YOL PROBLEMİ – FLOYD ALGORİTMASI

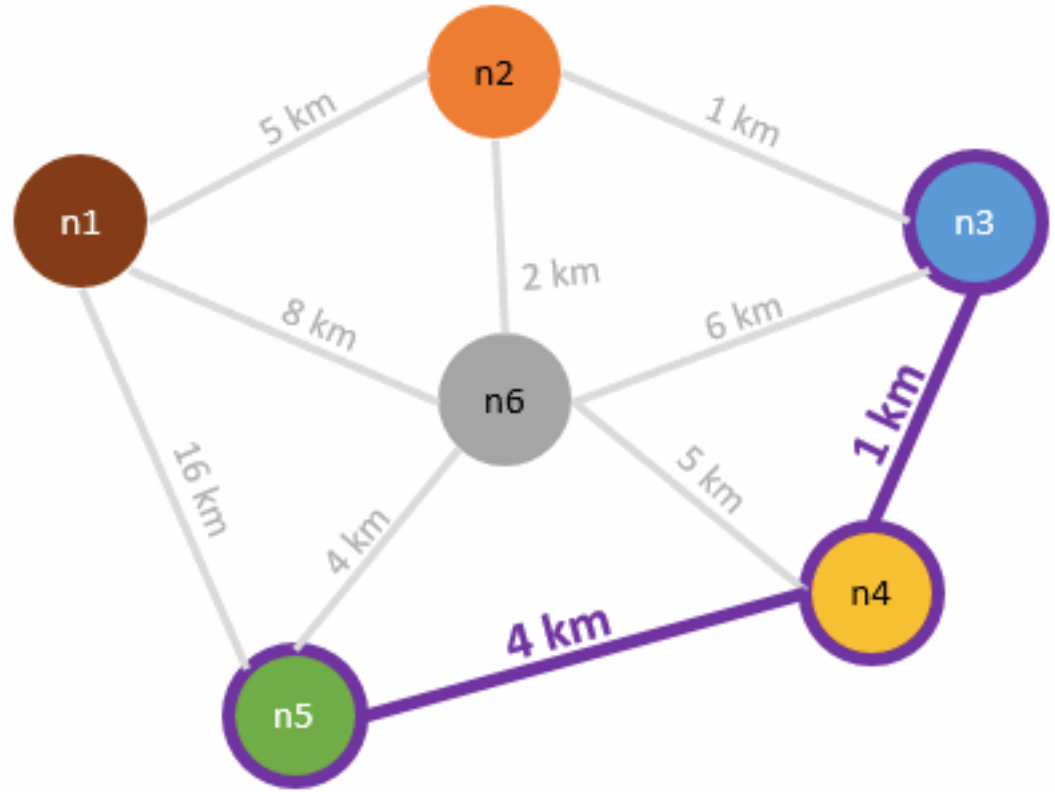


	n1	n2	n3	n4	n5	n6
n1	0	5	∞	∞	16	8
n2	5	0	1	∞	∞	2
n3	∞	1	0	1	∞	6
n4	∞	∞	1	0	4	5
n5	16	∞	∞	4	0	4
n6	8	2	6	5	4	0

```
using System;
using System.Linq;

namespace FloydWarshallCode
{
    class Program
    {
        static void Main(string[] args)
        {
            double[][] proximityMatrix = PrepareFirstState();
            Solve(ref proximityMatrix);
            Dump(proximityMatrix);
        }

        public static void Solve(ref double[][] matrix)
        {
            int size = matrix.Count();

            for (int i = 0; i < size; i++)
            {
                for (int j = 0; j < size; j++)
                {
                    for (int k = 0; k < size; k++)
                    {
                        matrix[j][k] = Math.Min(matrix[j][k], matrix[j][i] + matrix[i][k]);
                    }
                }
            }
        }

        private static double[][] PrepareFirstState()
        {
            double[][] matrix = new double[6][]{
                new double[6],
                new double[6],
                new double[6],
                new double[6],
                new double[6],
                new double[6]
            };

            matrix[0][0] = 0;
            matrix[0][1] = 5;
            matrix[0][2] = double.PositiveInfinity;
            matrix[0][3] = double.PositiveInfinity;
            matrix[0][4] = 16;
            matrix[0][5] = 8;

            matrix[1][0] = 5;
            matrix[1][1] = 0;
            matrix[1][2] = 1;
            matrix[1][3] = double.PositiveInfinity;
            matrix[1][4] = double.PositiveInfinity;
            matrix[1][5] = 2;

            matrix[2][0] = double.PositiveInfinity;
            matrix[2][1] = 1;
            matrix[2][2] = 0;
            matrix[2][3] = 1;
            matrix[2][4] = double.PositiveInfinity;
            matrix[2][5] = 6;

            matrix[3][0] = double.PositiveInfinity;
            matrix[3][1] = double.PositiveInfinity;
            matrix[3][2] = 1;
            matrix[3][3] = 0;
            matrix[3][4] = 4;
            matrix[3][5] = 5;

            matrix[4][0] = 16;
            matrix[4][1] = double.PositiveInfinity;
            matrix[4][2] = double.PositiveInfinity;
            matrix[4][3] = 4;
            matrix[4][4] = 0;
            matrix[4][5] = 4;

            matrix[5][0] = 8;
            matrix[5][1] = 2;
            matrix[5][2] = 6;
            matrix[5][3] = 5;
            matrix[5][4] = 4;
            matrix[5][5] = 0;
```

```
84              return matrix;
85          }
86
87          public static void Dump(double[][] matrix)
88          {
89              int size = matrix.Count();
90
91              for (int i = 0; i < size; i++)
92              {
93                  for (int j = 0; j < size; j++)
94                  {
95                      Console.Write("{0}\t", matrix[i][j]);
96                  }
97                  Console.WriteLine();
98              }
99          }
100     }
101 }
```





	n1	n2	n3	n4	n5	n6
n1	0	5	6	7	11	7
n2	5	0	1	2	6	2
n3	6	1	0	1	5	3
n4	7	2	1	0	4	4
n5	11	6	5	4	0	4
n6	7	2	3	4	4	0

T.C.
KIRIKKALE
ÜNİVERSİTESİ
BİLGİSAYAR
MÜHENDİSLİĞİ

AĞ OPTİMİZASYONU

DR. EVRENCAN ÖZCAN
OFİS:275
EVRENCAN.OZCAN@KKU.EDU.TR