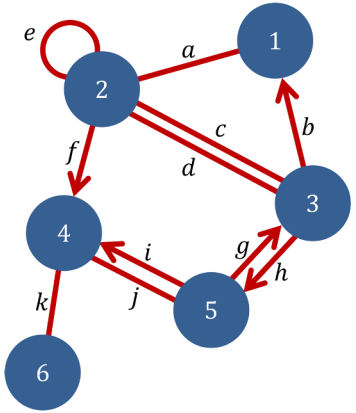
Graf Veri Modeli

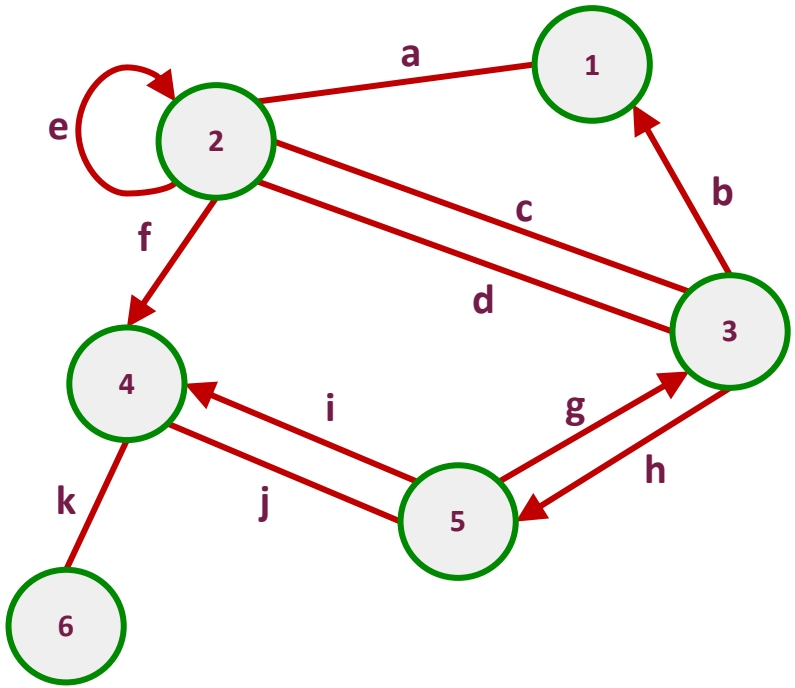


Suhap SAHIN
Onur GÖK

Graf Veri Modeli

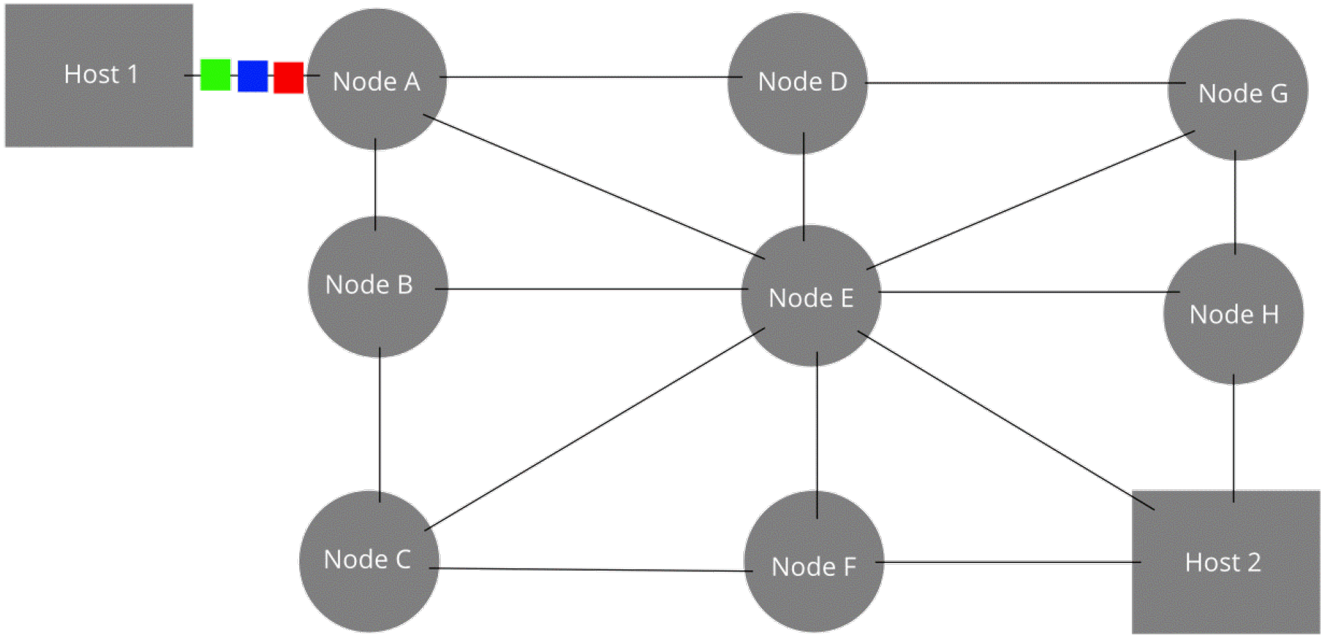
1. Tanım ve Kullanım alanları
2. Grafların Bellekte tutulma sekilleri
3. Graf Renklendirme(Welch-powel alg.)
4. En kısa yol problemi
    1. Dijkstra E.K. Y. A.
    2. Bellman-Ford E. K. Y. A.
5. En kısa Yol Agacı
    1. Prim E. K. Y. A.A.
    2. Kruskal E. K.Y.A.A.

Tanım



Kullanım Alanları

Kullanım Alanları

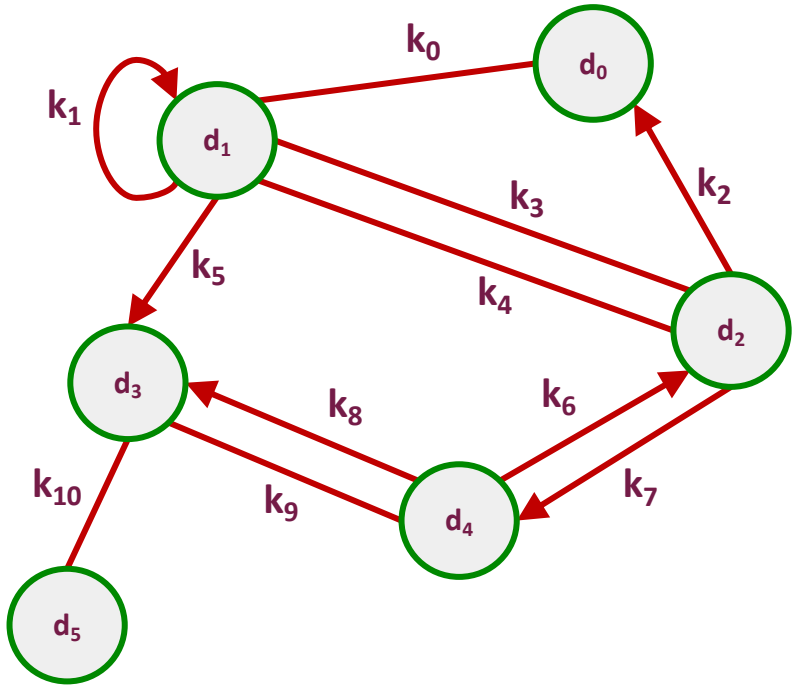


Graf

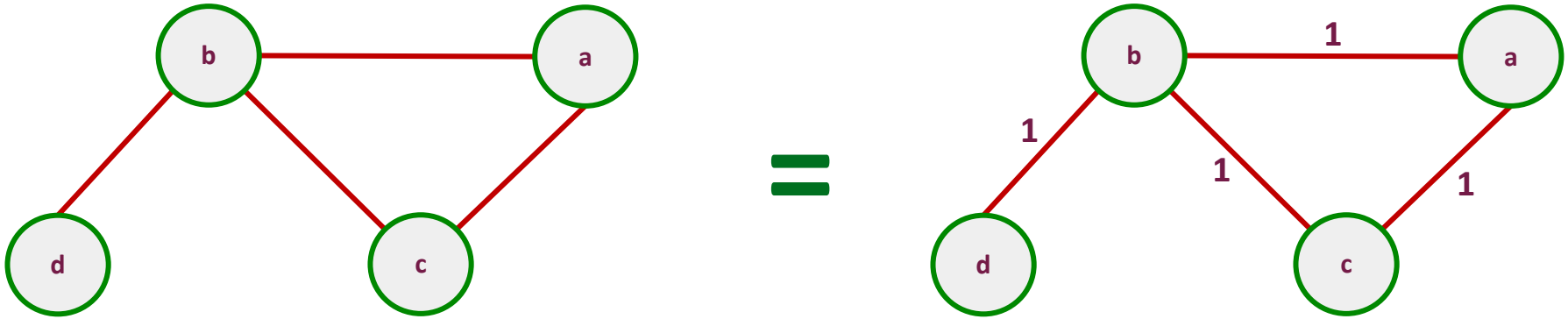
D={d0, d1, d2, d3, d4, d5}

K={ k0, k1, k2, k3, k4, k5, k6, k7, k8, k9, k10 }

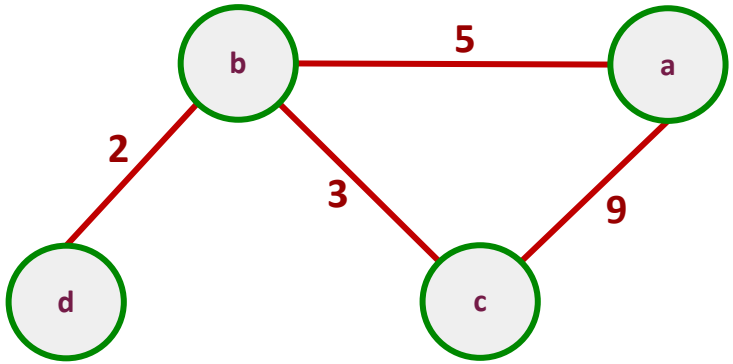
G={D,K}



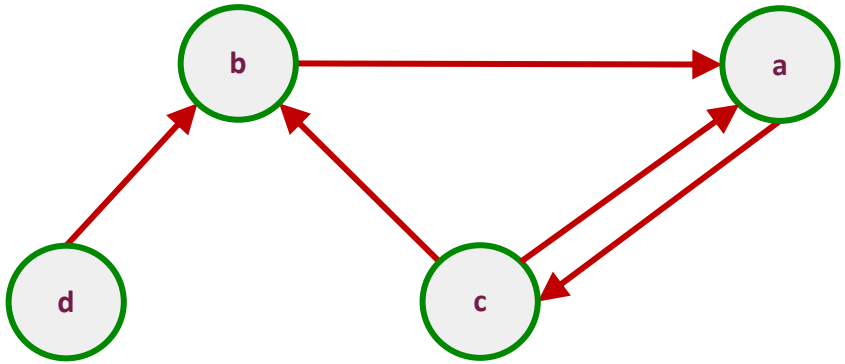
Basit Graf



Maliyetli Graf

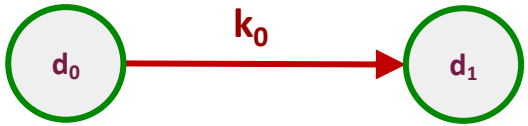


Yönlendirilmis Graf



Komsuluk Bitisiklik

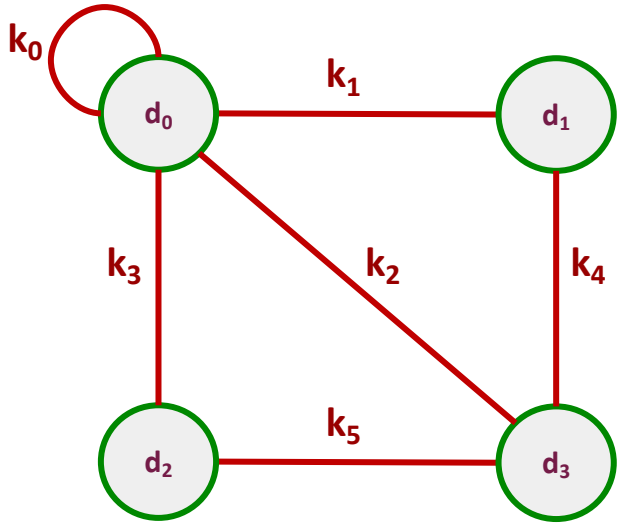
Komsuluk iliskisi $Gdd=\{(d_0,d_1)\}$



Bitisiklik iliskisi

$Gdk=\{(d_0,k_0),(d_1,k_0)\}$

Komsuluk Bitisiklik



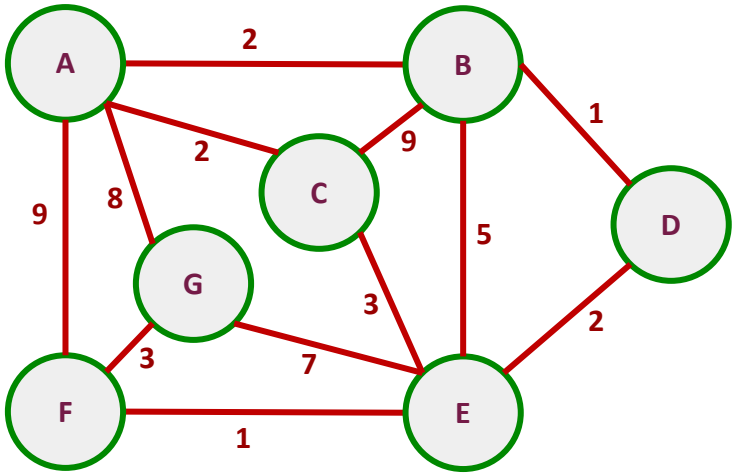
Komsuluk iliskisi

Gdd={(d_0,d_0), (d_0,d_1), (d_0,d_2), (d_0,d_3), (d_1,d_0), (d_1,d_3), (d_2,d_0), (d_2,d_3), (d_3,d_0), (d_3,d_1), (d_3,d_2)}

Bitisiklik iliskisi

Gdk={(d_0,k_0), (d_0,k_1), (d_0,k_2), (d_0,k_3), (d_1,k_1), (d_1,k_4), (d_2,k_3), (d_2,k_5), (d_3,k_2), (d_3,k_4), (d_3,k_5)}

Komsuluk Bitisiklik

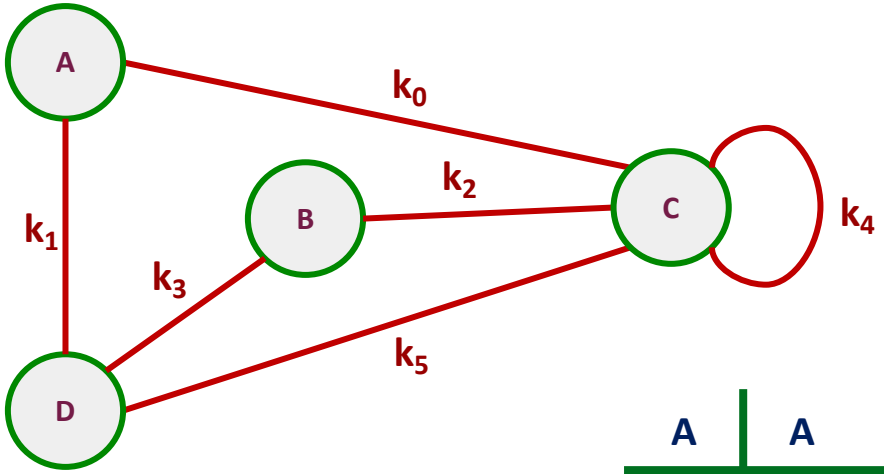


Komsuluk iliskisi

Gdd={(A,B:2), (A,C:2), (A,G:8), (A,F:9), (B,C:9), (B,D:1), (B,E:5), (C,E:3), (D,E:2), (E,G:7),

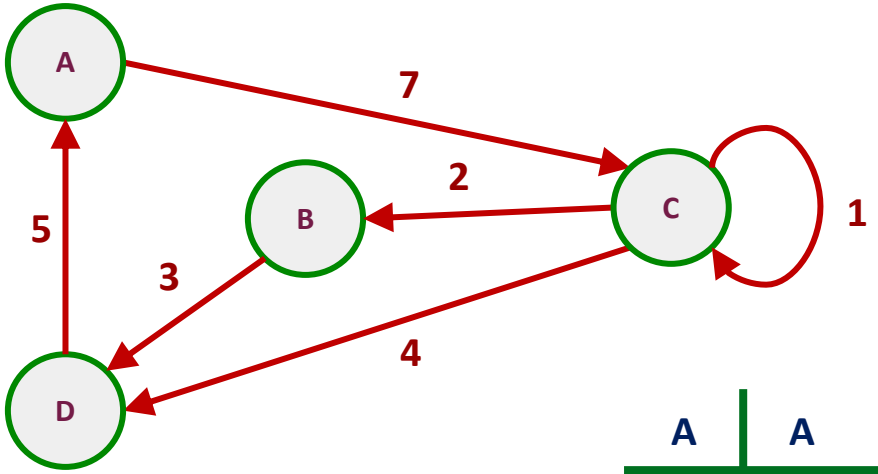
(E,F:1),(F,G:3)}

Komsuluk Matrisi



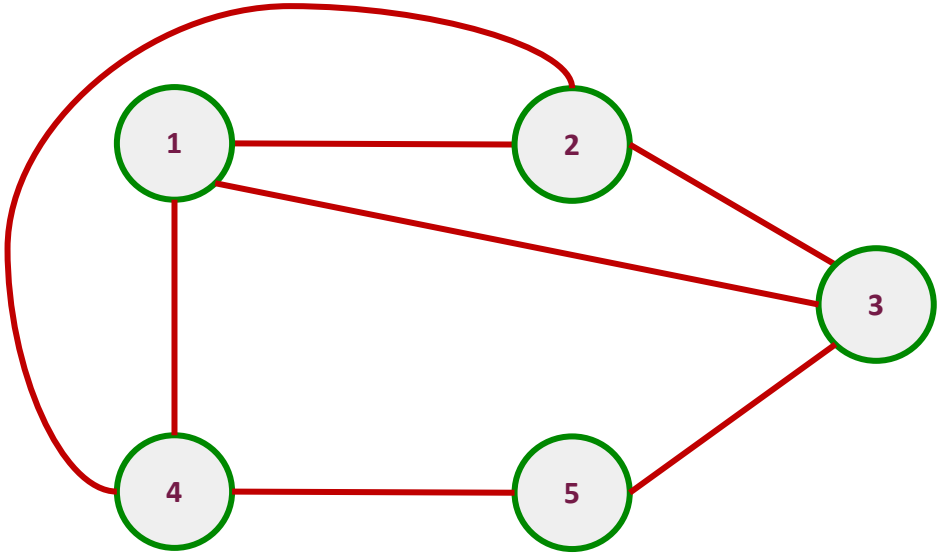
A	A	B	C	D
A	0	0	1	1
B	0	0	1	1
C	1	1	1	1
D	1	1	1	0

Yönlü Komsuluk Matrisi



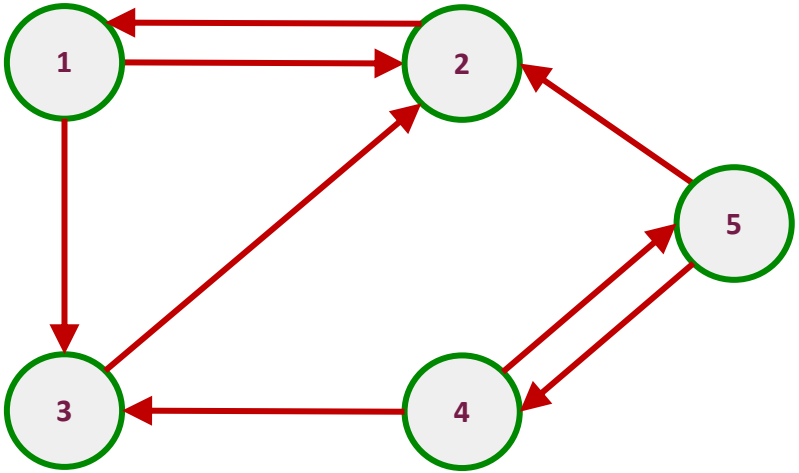
A	A	B	C	D
A	-	-	7	-
B	-	-	-	3
C	-	2	1	4
D	5	-	-	-

Komsuluk Matrisi



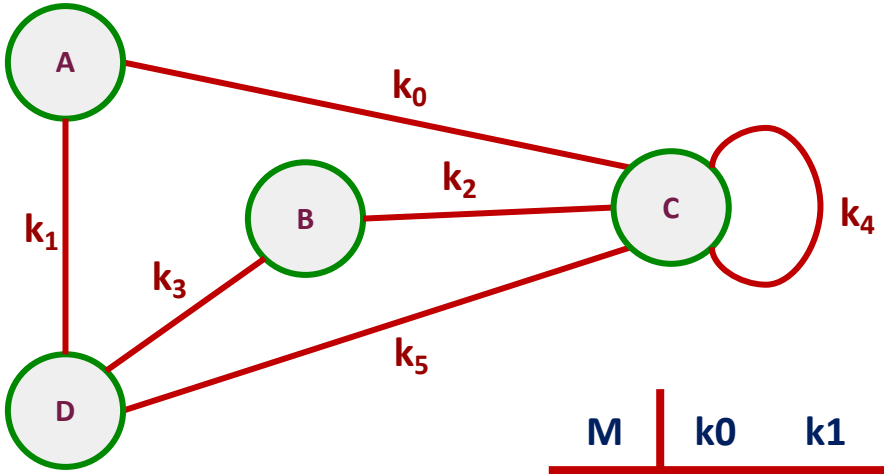
	1	2	3	4	5
1	0	1	1	1	0
2	1	0	1	1	0
3	1	1	0	0	1
4	1	1	0	0	1
5	0	0	1	1	0

Yönlü Komsuluk Matrisi



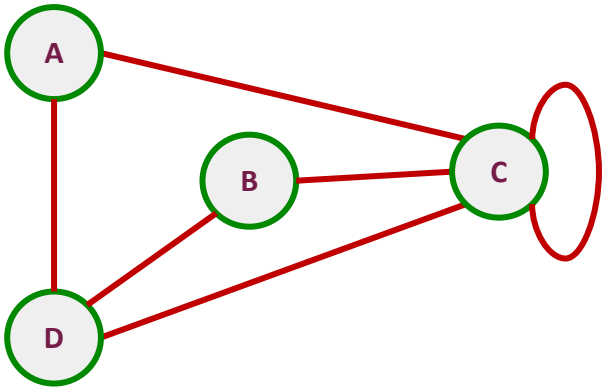
	1	2	3	4	5
1	0	1	1	0	0
2	1	0	0	0	0
3	0	1	0	0	0
4	0	0	1	0	1
5	0	1	0	1	0

Bitisiklik Matrisi



M	k0	k1	k2	k3	k4	k5
A	1	1	0	0	0	0
B	0	0	1	1	0	0
C	1	0	1	0	1	1
D	0	1	0	1	0	1

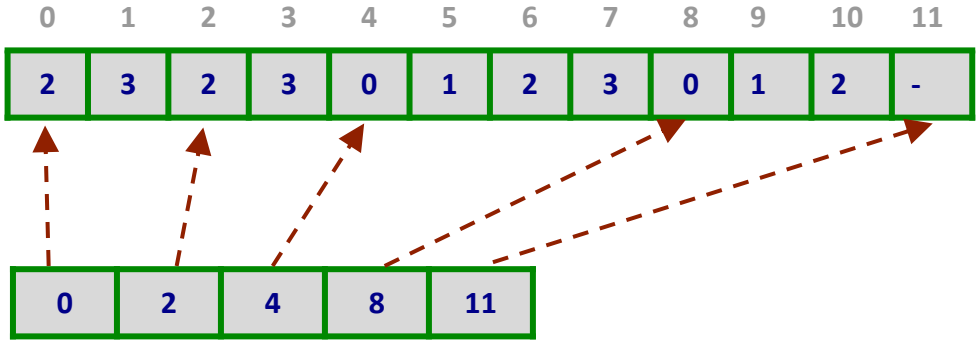
Bellekte Tutulması



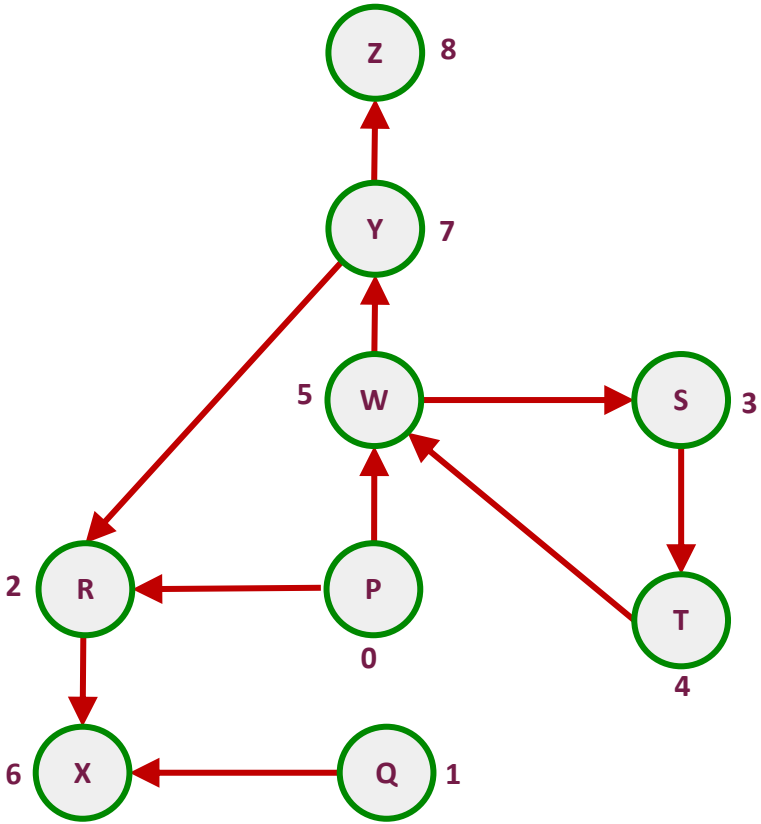
Matris

	A	B	C	D
A	0	0	1	1
B	0	0	1	1
C	1	1	1	1
D	1	1	1	0

Dizi

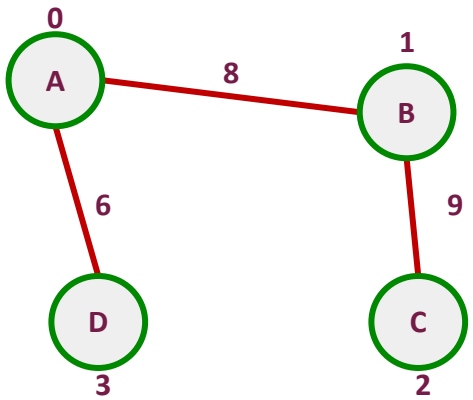


Bellekte Tutulması



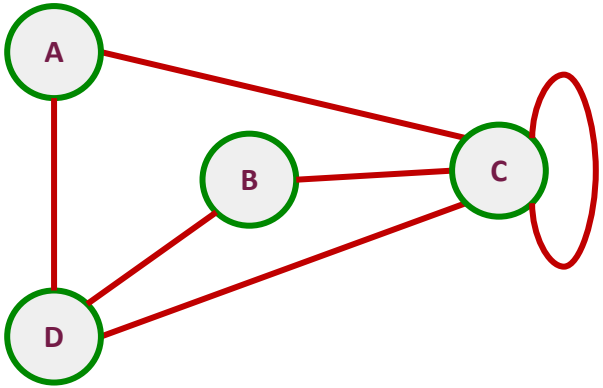
		0	1	2	3	4	5	6	7	8
		P	Q	R	S	T	W	X	Y	Z
0	P	0	0	1	0	0	1	0	0	0
1	Q	0	0	0	0	0	0	1	0	0
2	R	0	0	0	0	0	0	1	0	0
3	S	0	0	0	0	1	0	0	0	0
4	T	0	0	0	0	0	1	0	0	0
5	W	0	0	0	1	0	0	0	1	0
6	X	0	0	0	0	0	0	0	0	0
7	Y	0	0	1	0	0	0	0	0	1
8	Z	0	0	0	0	0	0	0	0	0

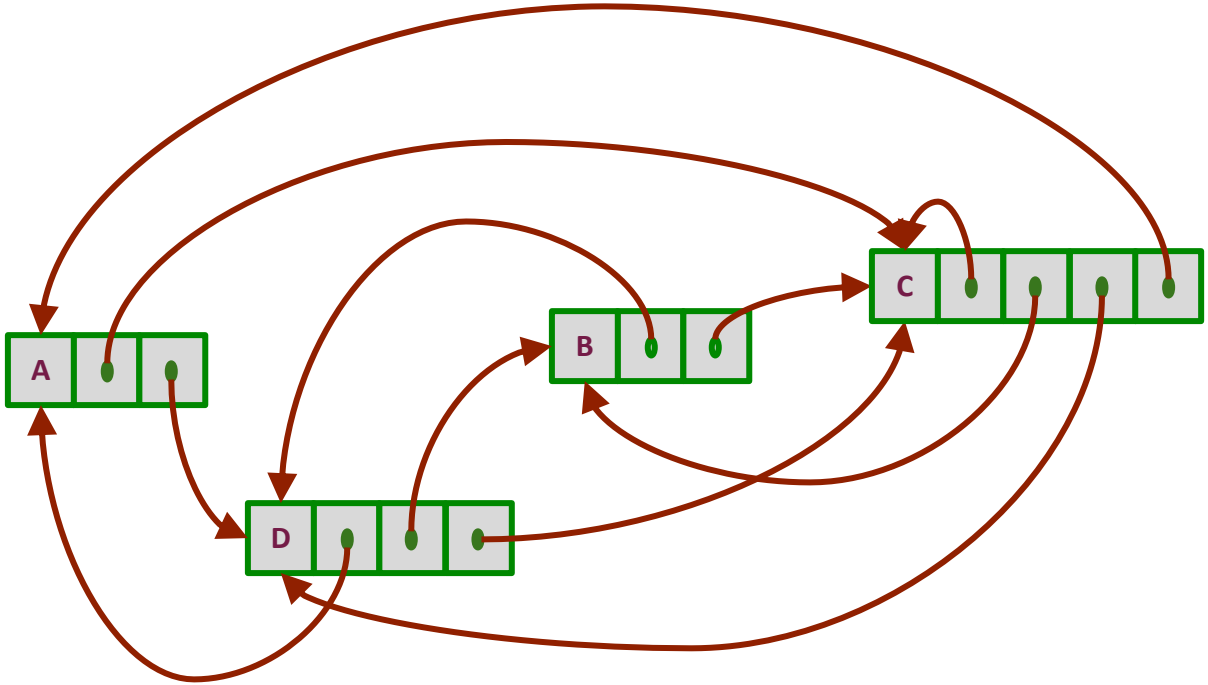
Bellekte Tutulması



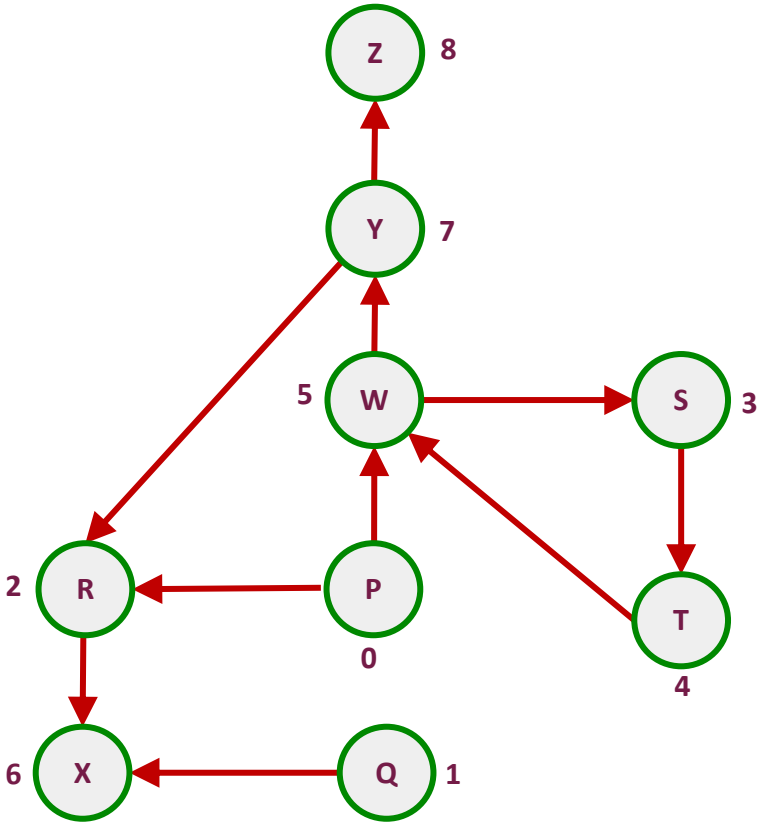
		0	1	2	3
		A	B	C	D
0	A	∞	8	∞	6
1	B	8	∞	9	∞
2	C	∞	9	∞	∞
3	D	6	∞	∞	∞

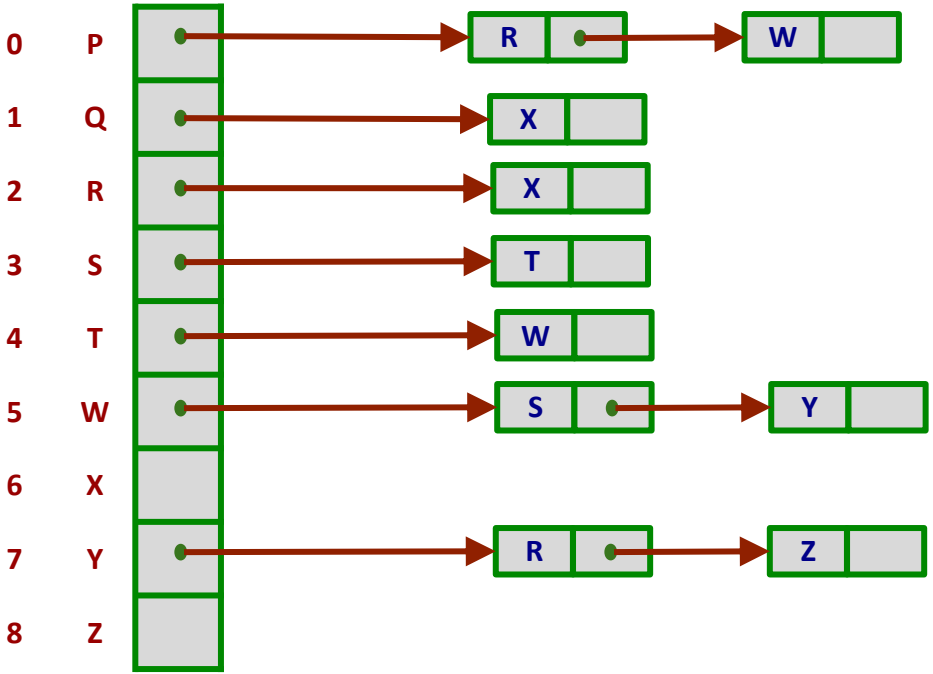
Bellekte Tutulması



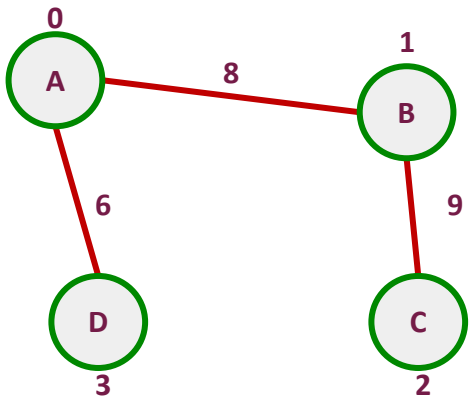


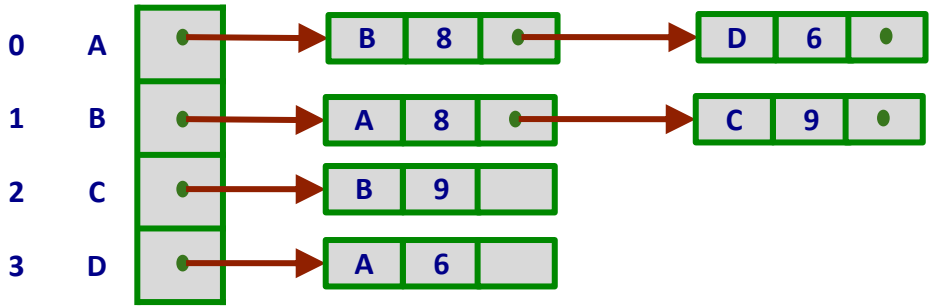
Bellekte Tutulması





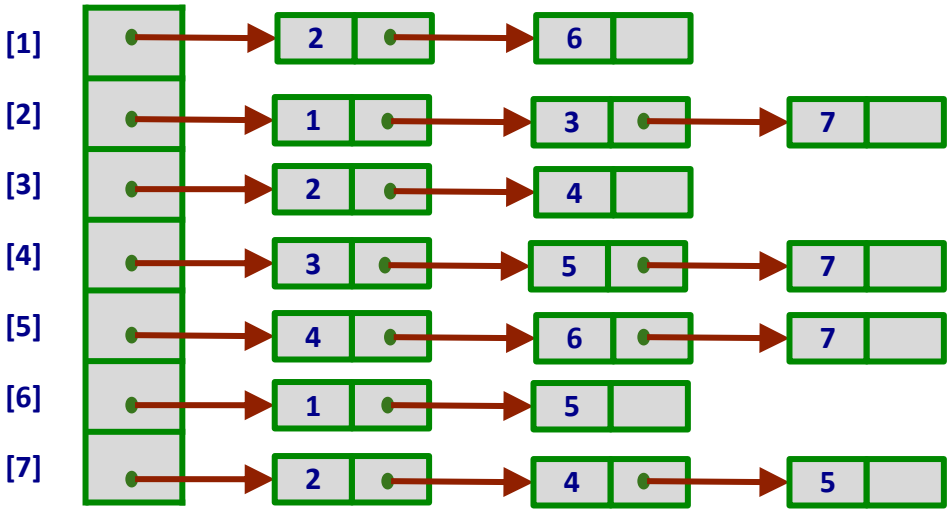
Bellekte Tutulması





Komsuluk Matrisi

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

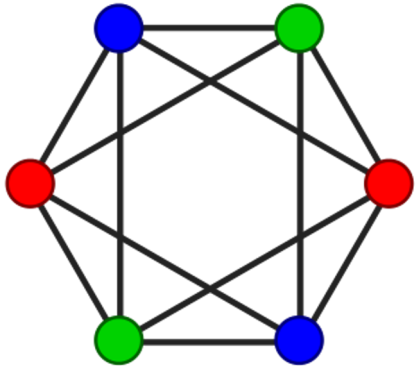


Graf Renklendirme

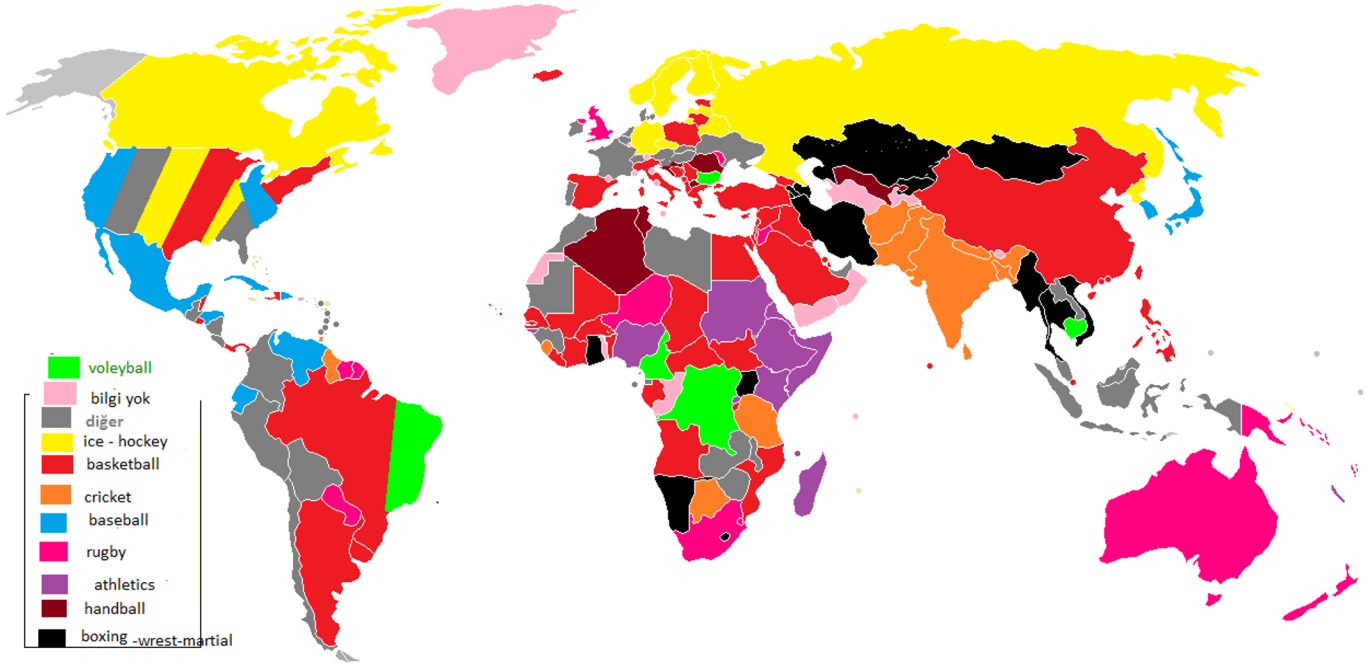
Graf üzerinde birbirine komsu olan dügümlere farklı renk atama

En az sayıda renk kullanılarak tüm dügümlere komsularından farklı renk verme

Kullanılan toplam renk sayısı kromatik (chromatik) sayı olarak adlandırılır.

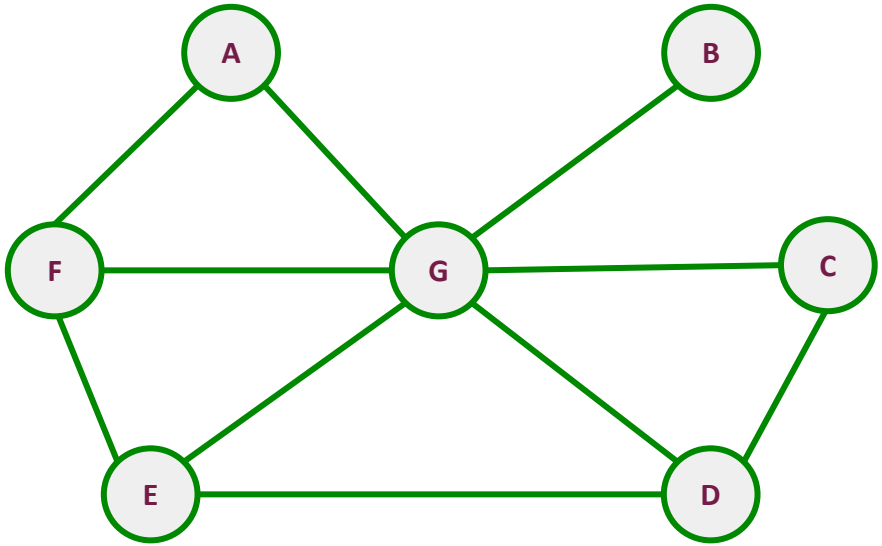


Graf Renklendirme



Welch-Powel Algoritması

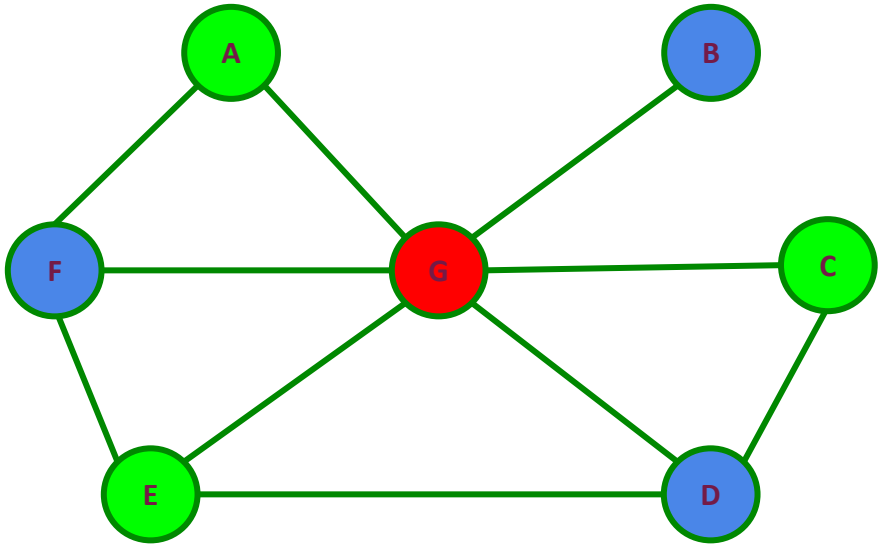
Adım 1: Dügümler derecelerine göre büyükten küçüge dogru sıralanır.



Dügüm	Derece
G	6
D	3
E	3
F	3
C	2
A	2
B	1

Welch-Powel Algoritması

Adım 3: Renk numarası bir artırılır, bu numara daha önce atama yapılmamış dügümlerden derecesi en büyük olana verilir ve adım 2 diger dügümler için tekrarlanır.



Dügüm	Derece
G	6
D	3
E	3
F	3
C	2
A	2
B	1

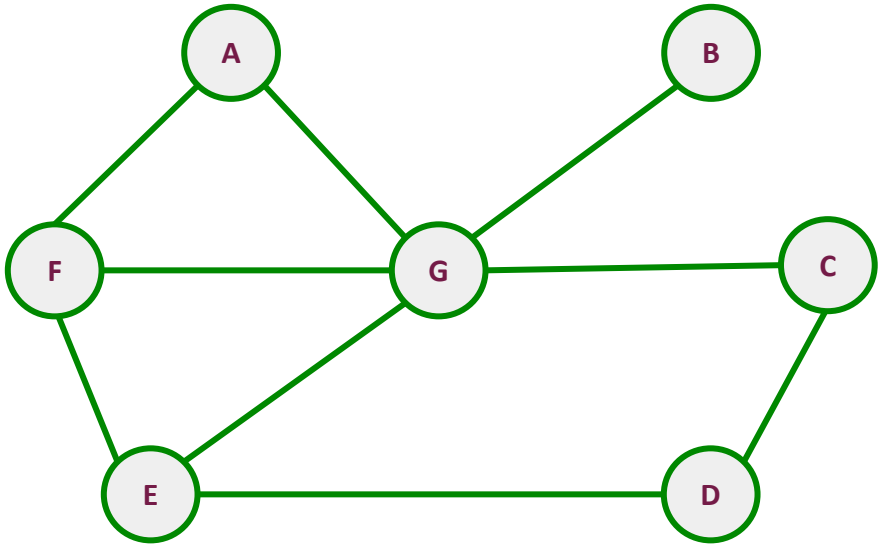
Kırmızı: G

Mavi: D,F,B

Yeşil A,E,C

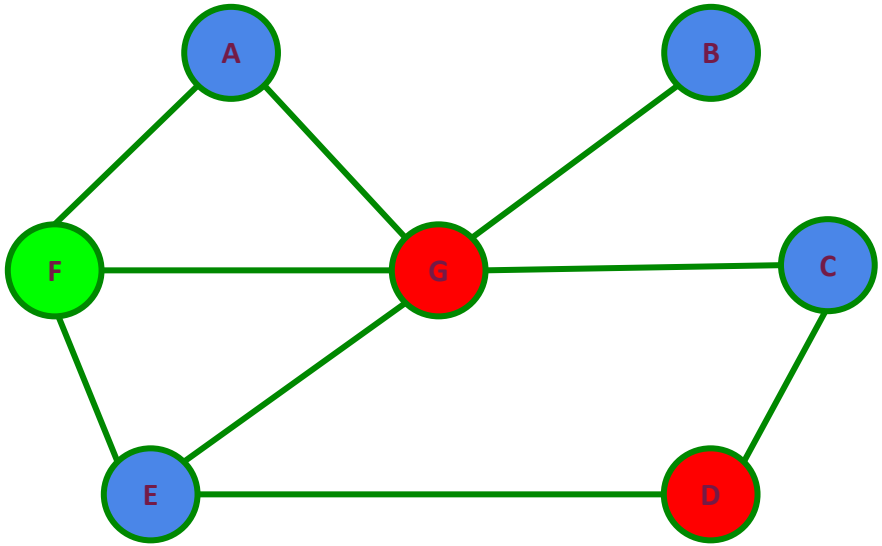
Kromatik sayı: 3

Welch-Powel Algoritması



Dügüm	Derece
G	5
E	3
F	3
D	2
C	2
A	2
B	1

Welch-Powel Algoritması



Dügüm	Derece
G	5
E	3
F	3
D	2
C	2
A	2
B	1

Kırmızı: G,D

Mavi: A,B,C,E

Yesil: F

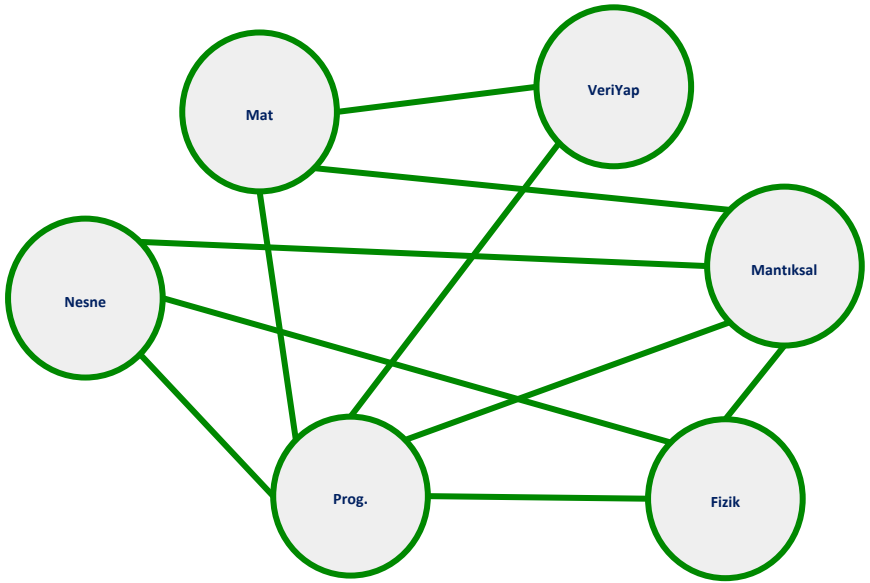
Kromatik sayı: 3

Sınav Çakısması

Ögr1: Mat, VeriYap, Prog.
Ögr2: Mat, Mantıksal, Prog.
Ögr3: Mantıksal, Fizik, Nesne
Ögr4: Fizik,Prog., Nesne

Sınav Çakışması

Ögr1: Mat, VeriYap, Prog.
Ögr2: Mat, Mantıksal, Prog.
Ögr3: Mantıksal, Fizik, Nesne
Ögr4: Fizik, Prog., Nesne



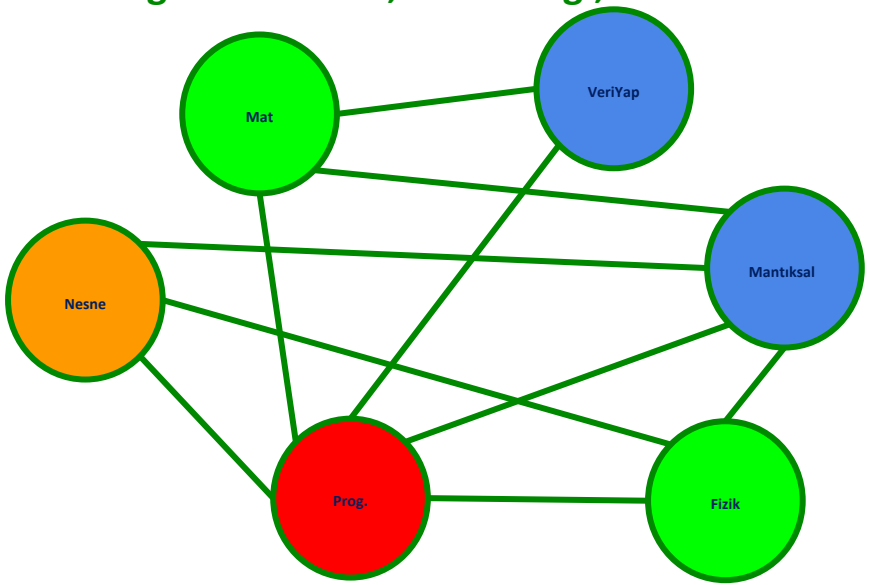
Sınav Çakışması

Ögr1: Mat, VeriYap, Prog.

Ögr2: Mat, Mantıksal, Prog.

Ögr3: Mantıksal, Fizik, Nesne

Ögr4: Fizik, Prog., Nesne



Dügüm	Derece
Prog.	5
Mantıksal	4
Mat	3
Fizik	3
Nesne	2
VeriYap	2

Kırmızı: Prog.

Mavi: VeriYap, Mantıksal

Yesil: Mat, Fizik

Turuncu: Nesne

Kromatik sayı: 4

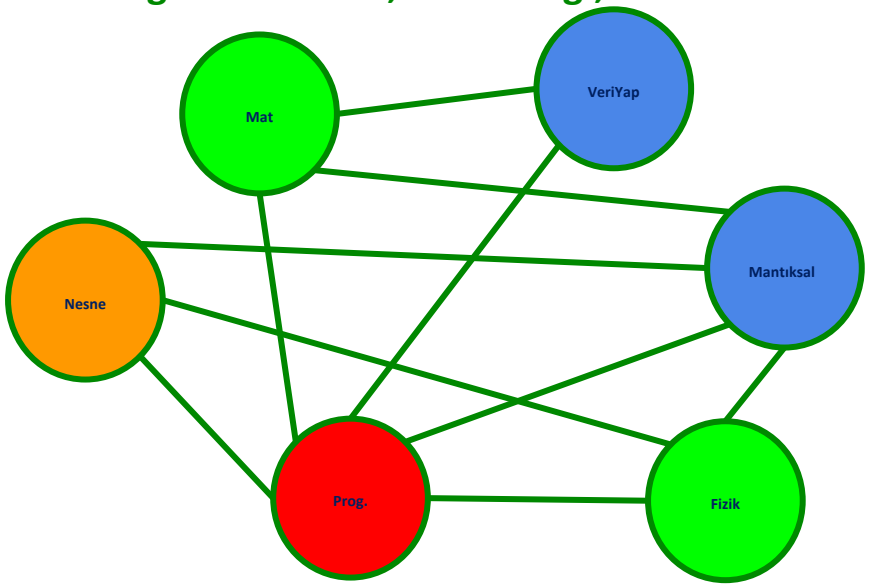
Sınav Çakışması

Ögr1: Mat, VeriYap, Prog.

Ögr2: Mat, Mantıksal, Prog.

Ögr3: Mantıksal, Fizik, Nesne

Ögr4: Fizik, Prog., Nesne



Dügüm	Derece
Prog.	5
Mantıksal	4
Mat	3
Fizik	3
Nesne	2
VeriYap	2

Saat 10:00: Prog.

Saat 12:00: VeriYap, Mantıksal

Saat 14:00: Mat, Fizik

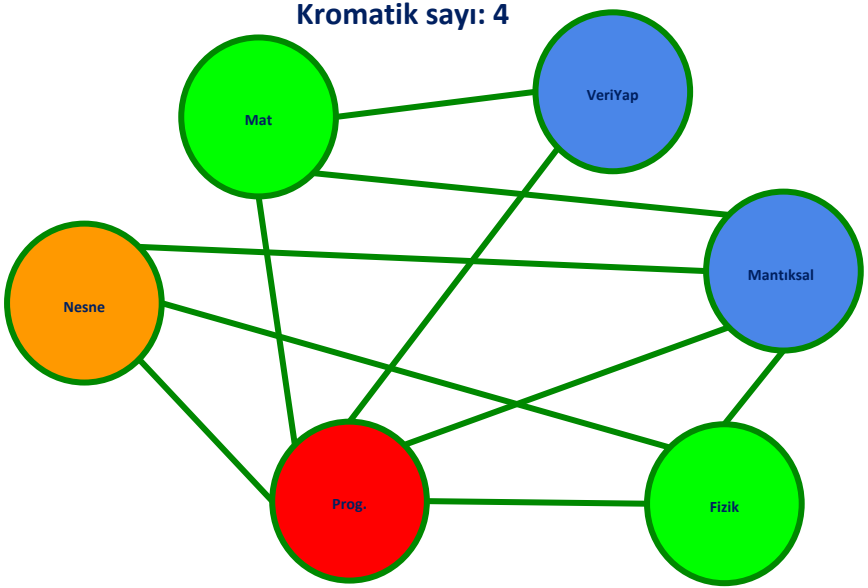
Saat 16:00: Nesne

Welch-Powel Algoritması

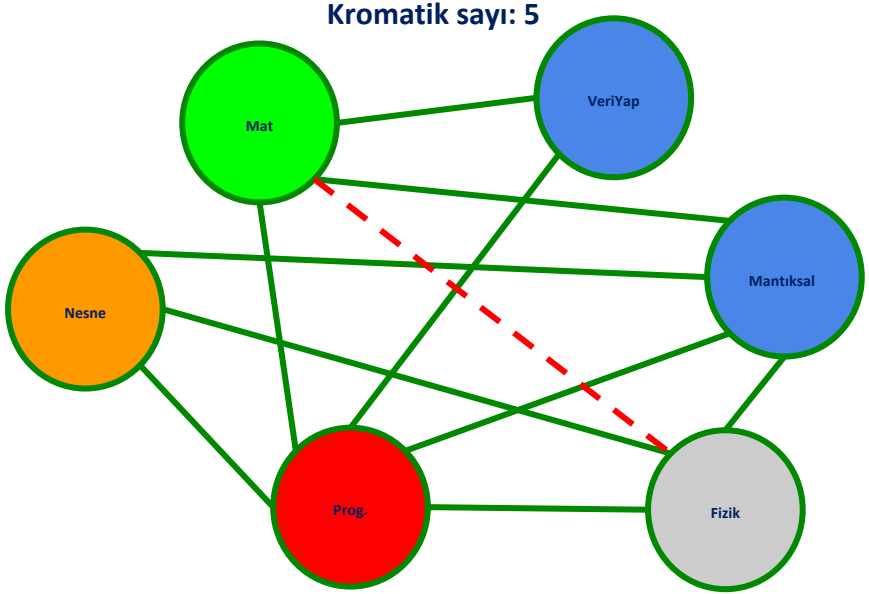
4 < Kromatik Sayı ise Graf 3 boyutludur.

Kromatik sayı <= 4 ise Graf 2 boyutludur (Düzlemseldir).

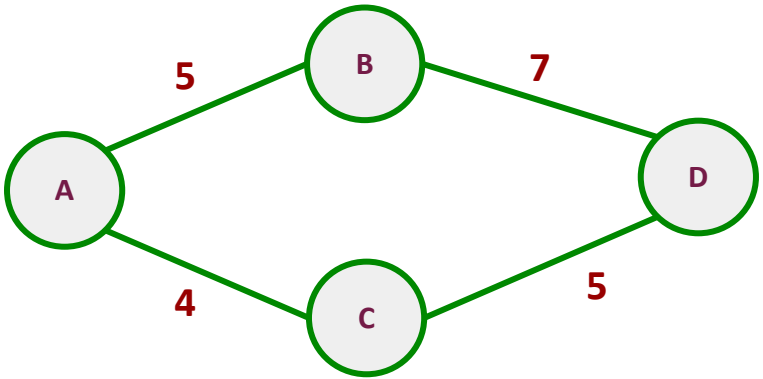
Kromatik sayı: 4



Kromatik sayı: 5

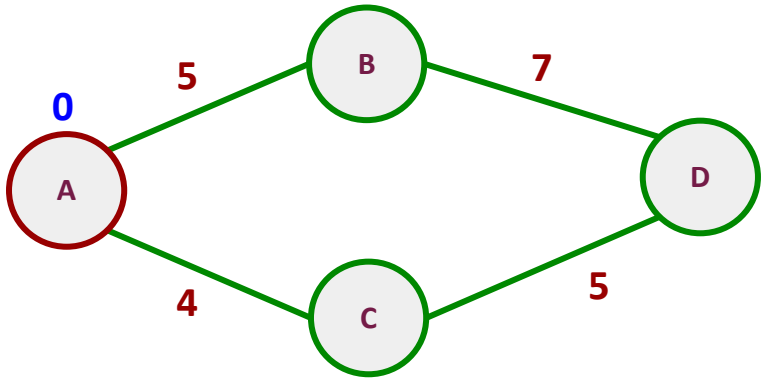


Dijkstra Algoritması

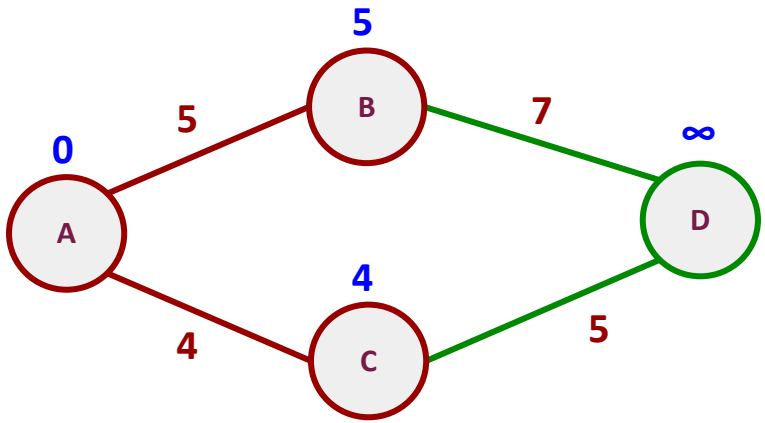


Dijkstra nın algoritması belirli bir başlangıç noktasına göre en kısa yolu belirleyen bir algoritmadır
Bir düğümden, yani başlangıç düğümünden, diğer tüm düğümlere olan en kısa yolu belirler
Ağırlıklı ve yönlü graflar için geliştirilmiş olup graf üzerindeki herbir kenarın ağırlığı sıfır veya sıfırdan büyük sayıdır.

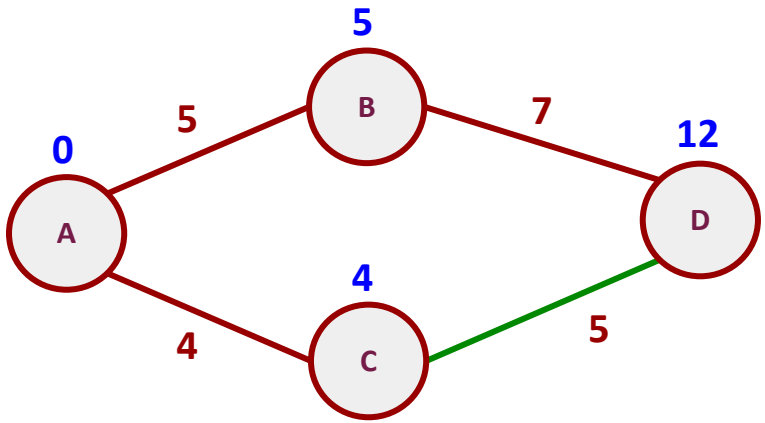
Dijkstra Algoritması



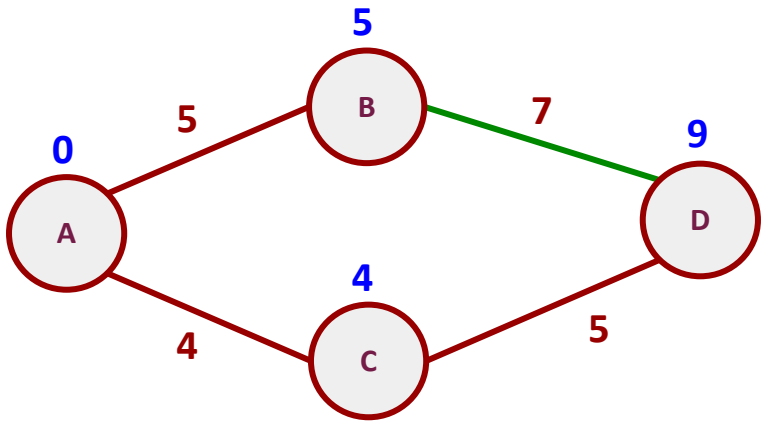
Dijkstra Algoritması



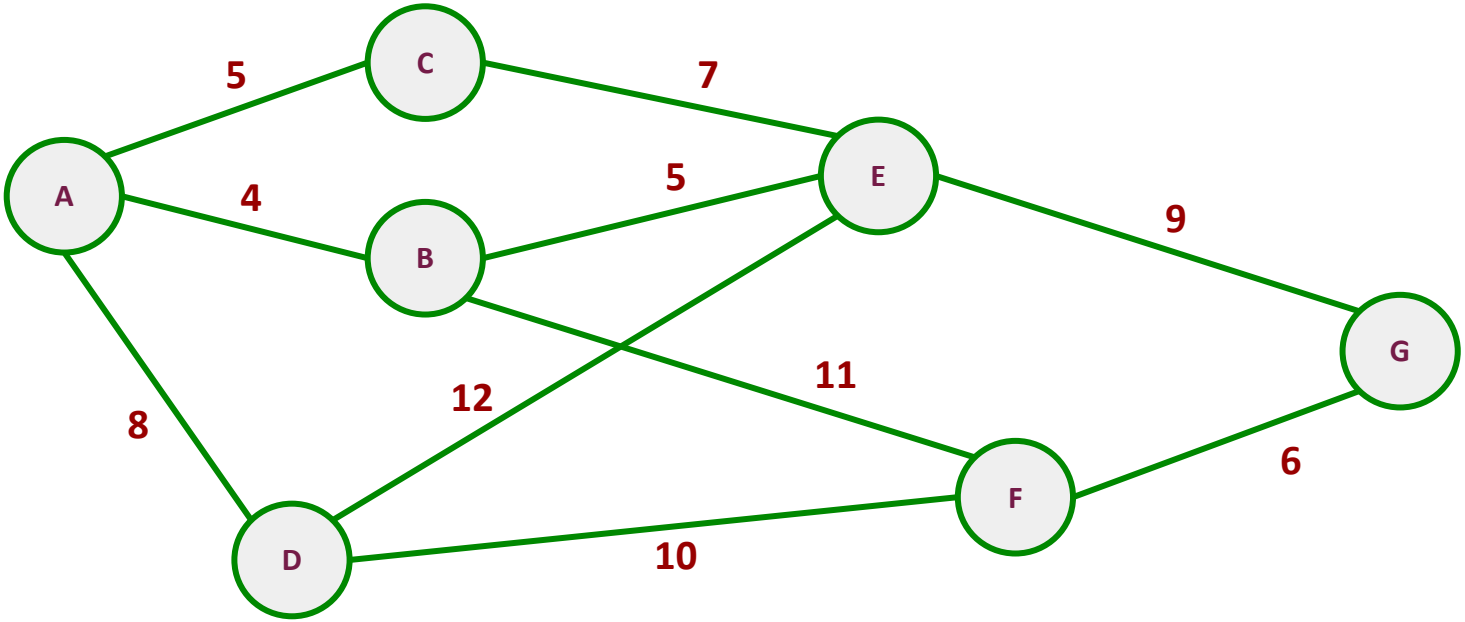
Dijkstra Algoritması



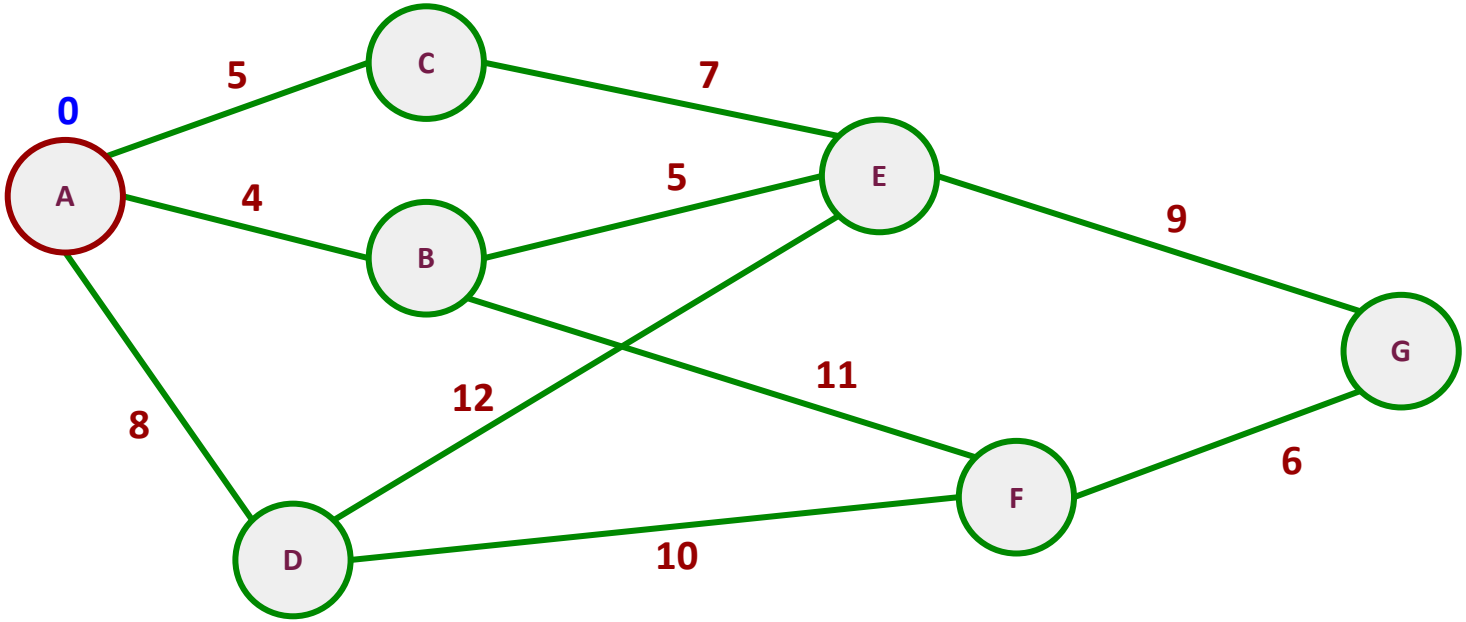
Dijkstra Algoritması



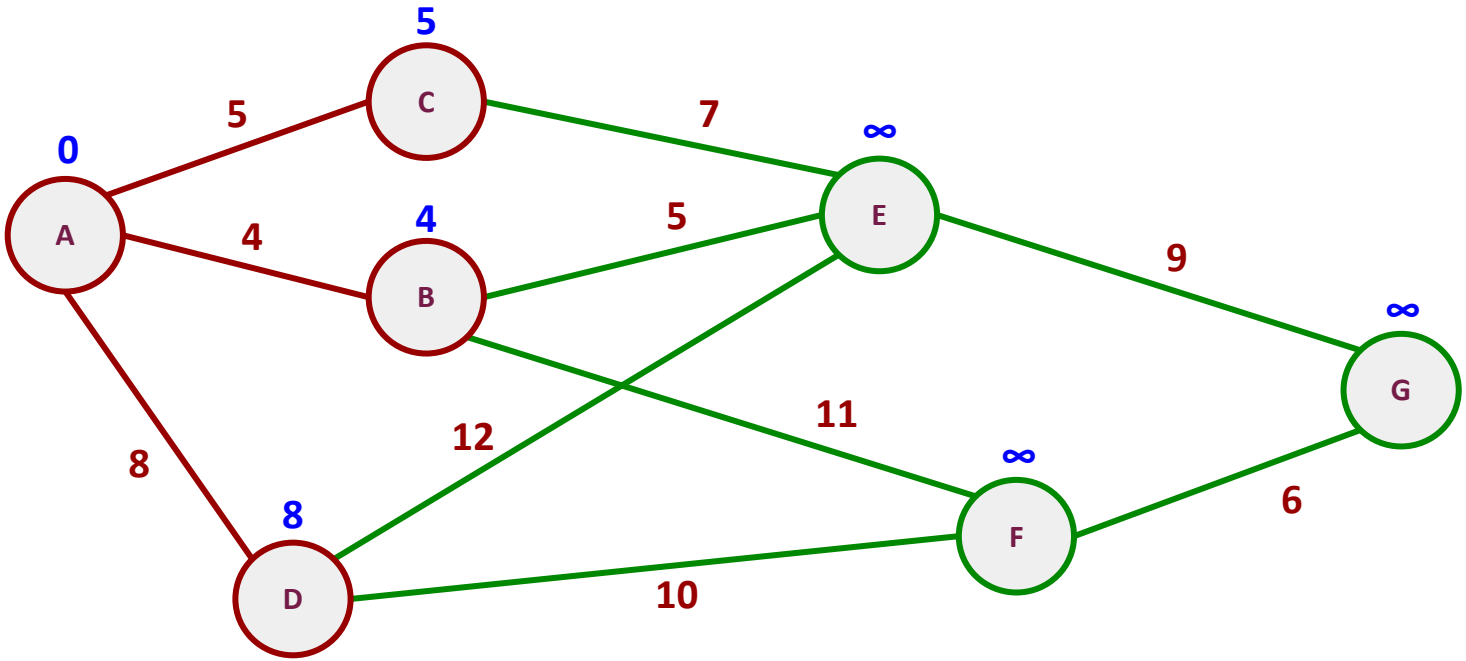
Dijkstra Algoritması



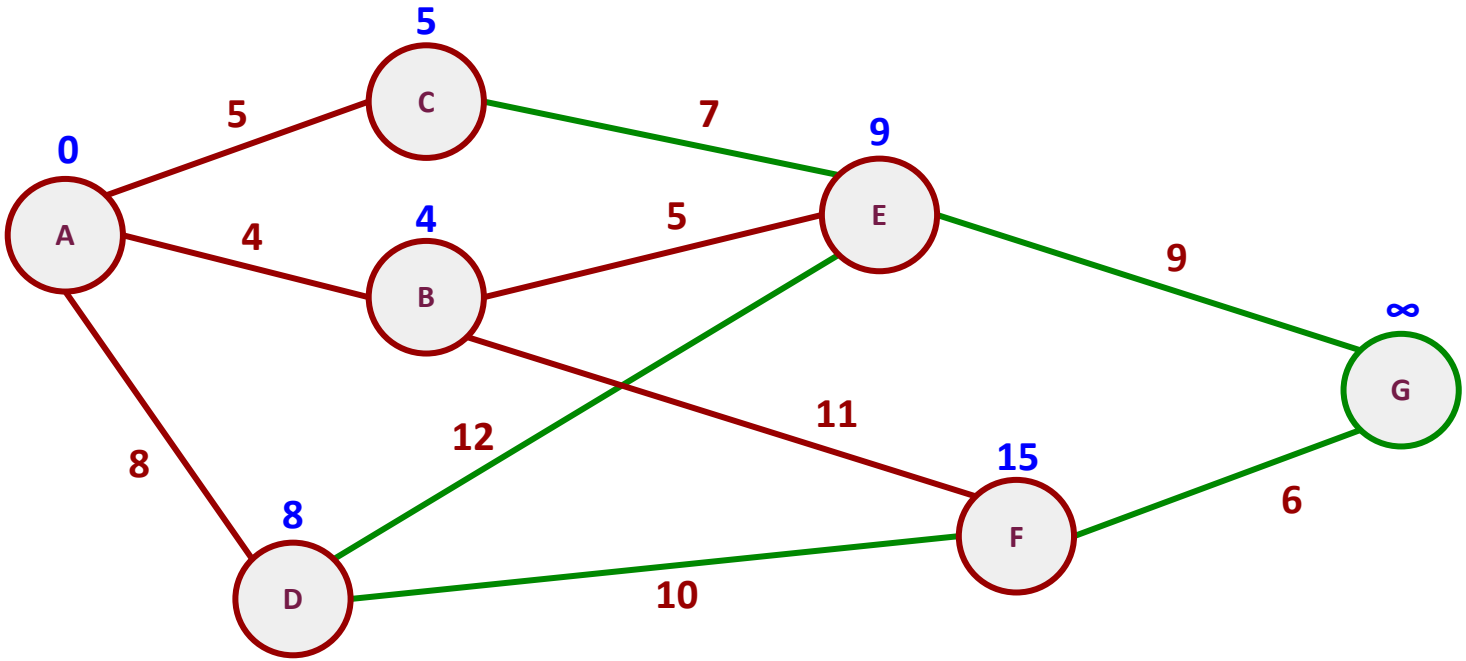
Dijkstra Algoritması



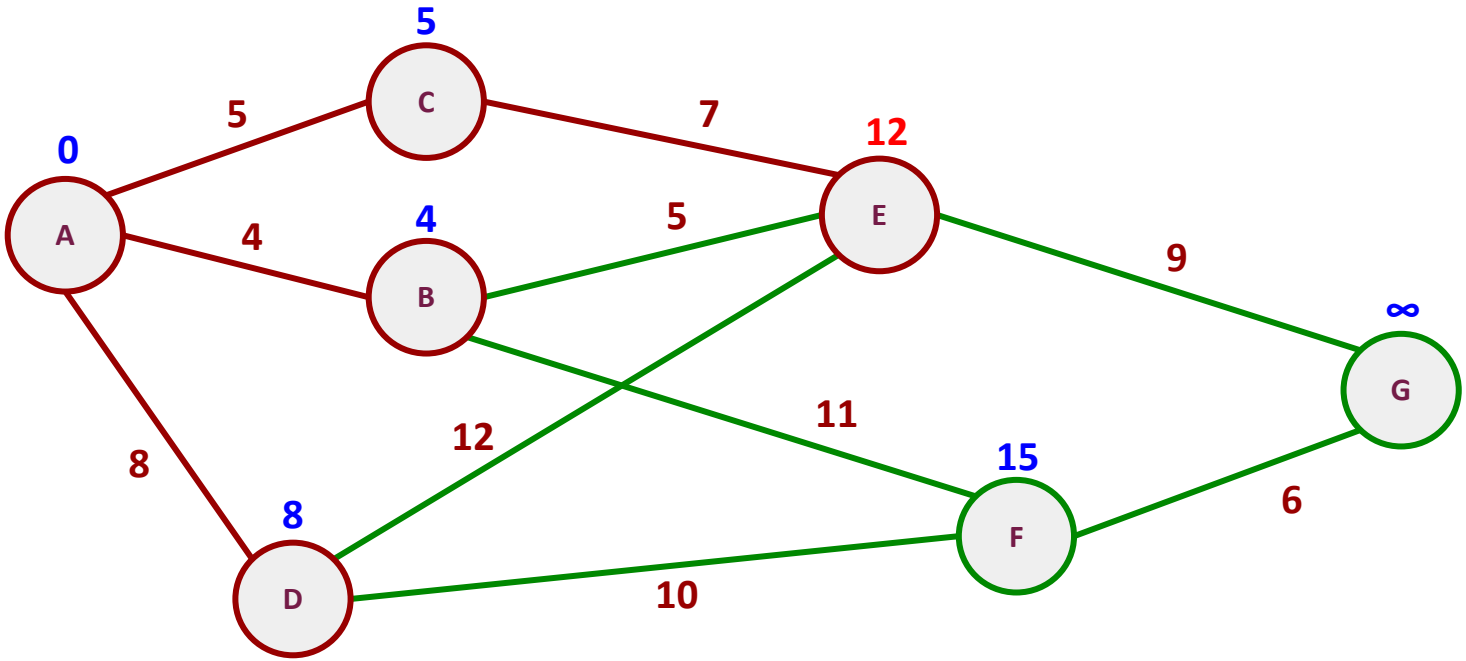
Dijkstra Algoritması



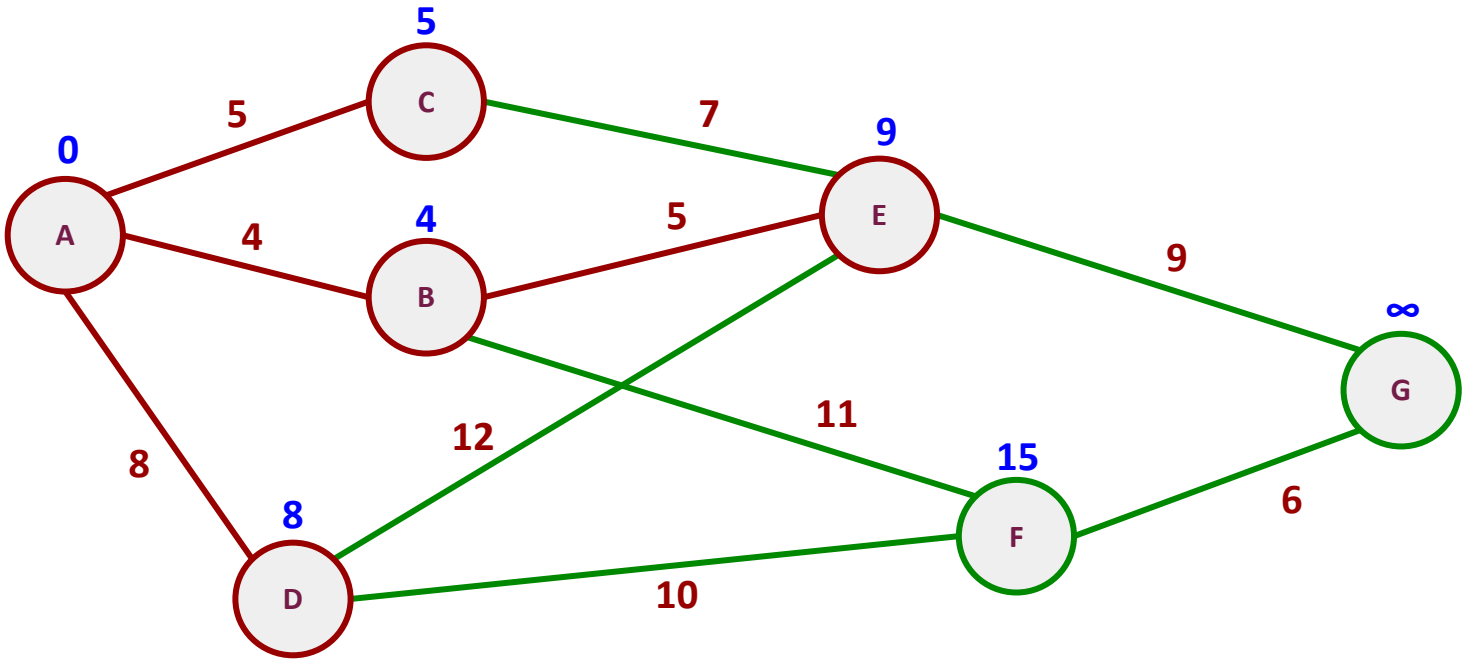
Dijkstra Algoritması



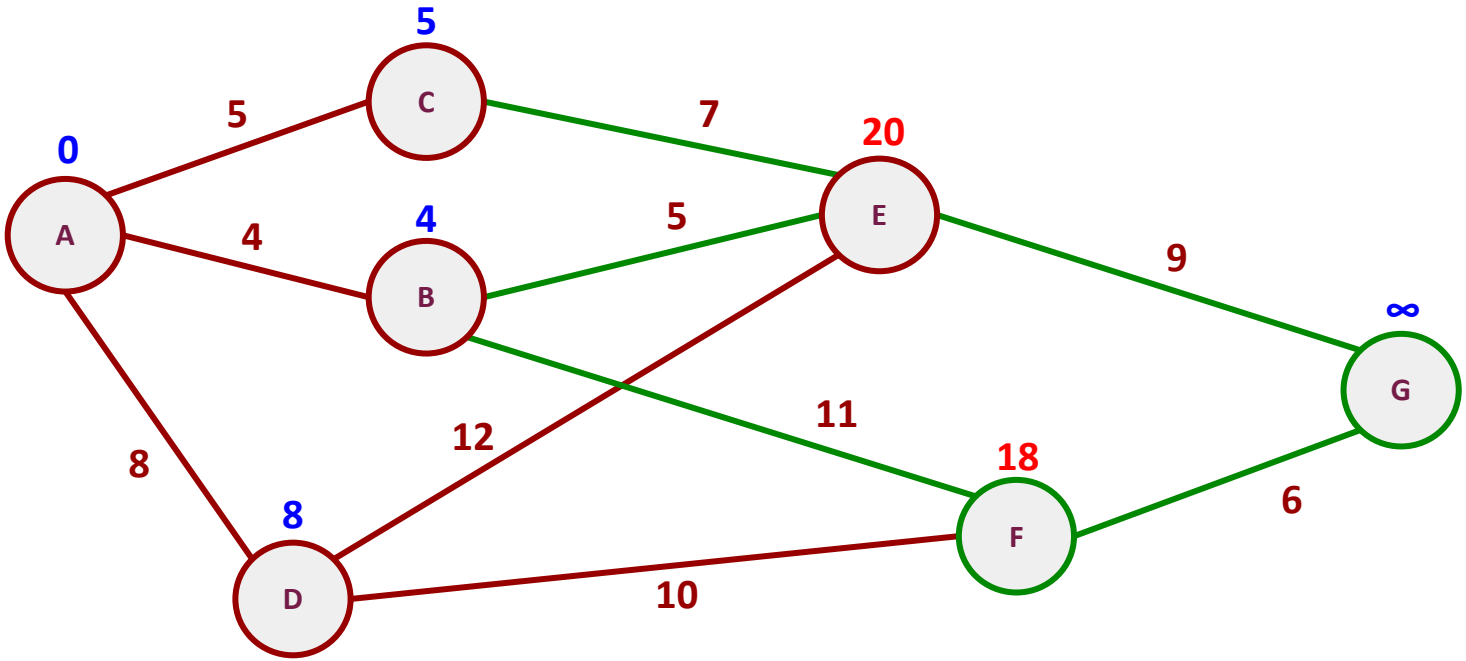
Dijkstra Algoritması



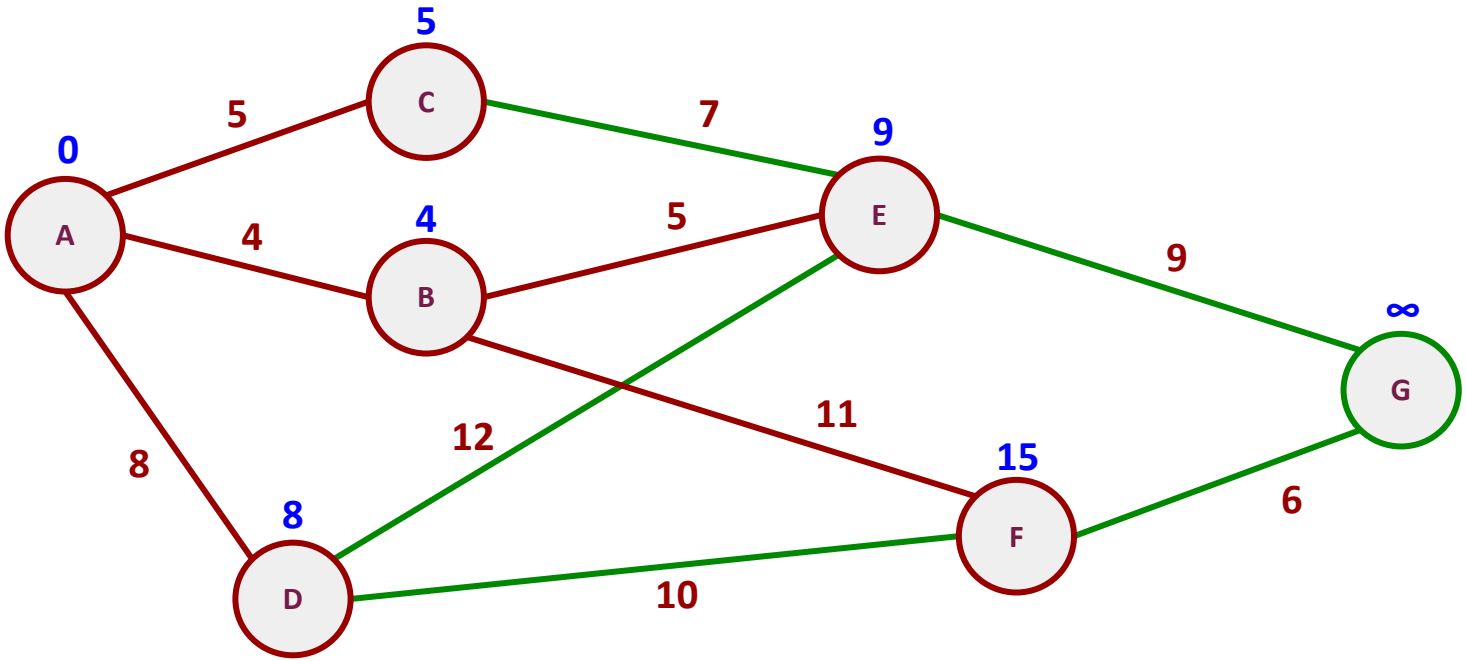
Dijkstra Algoritması



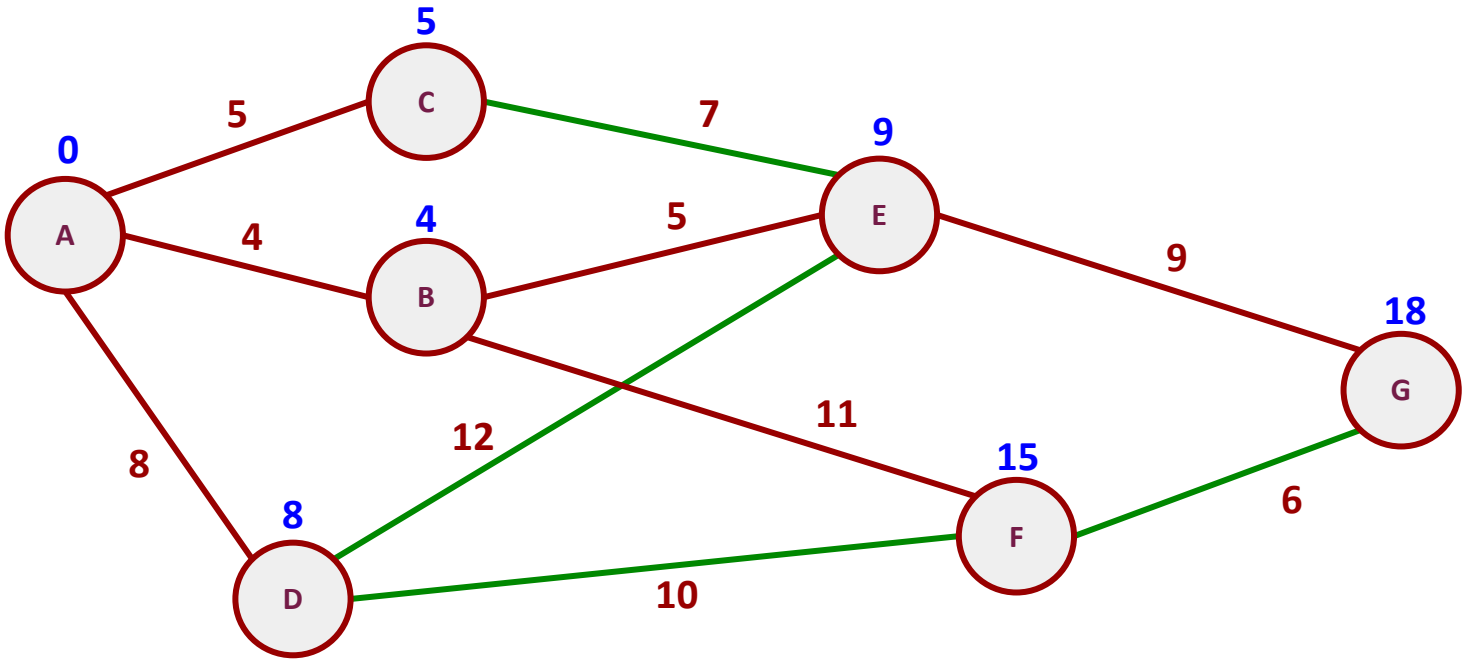
Dijkstra Algoritması



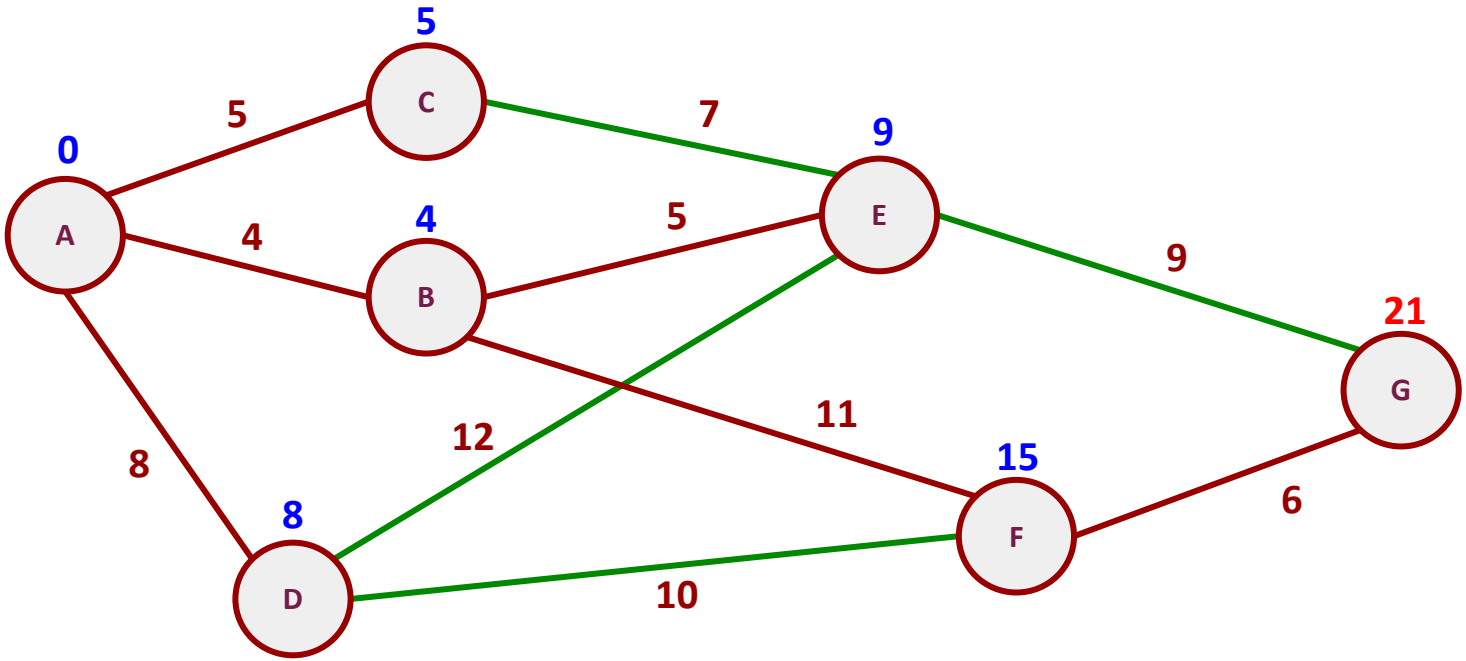
Dijkstra Algoritması



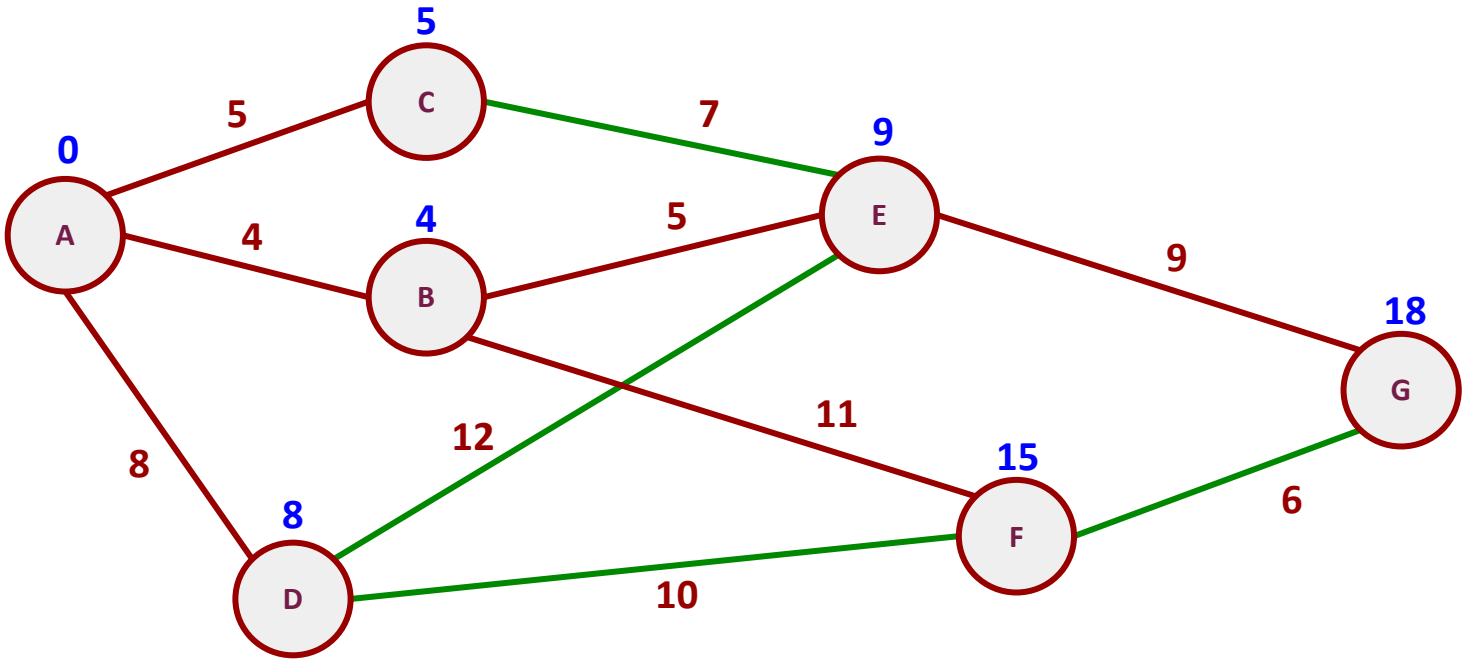
Dijkstra Algoritması



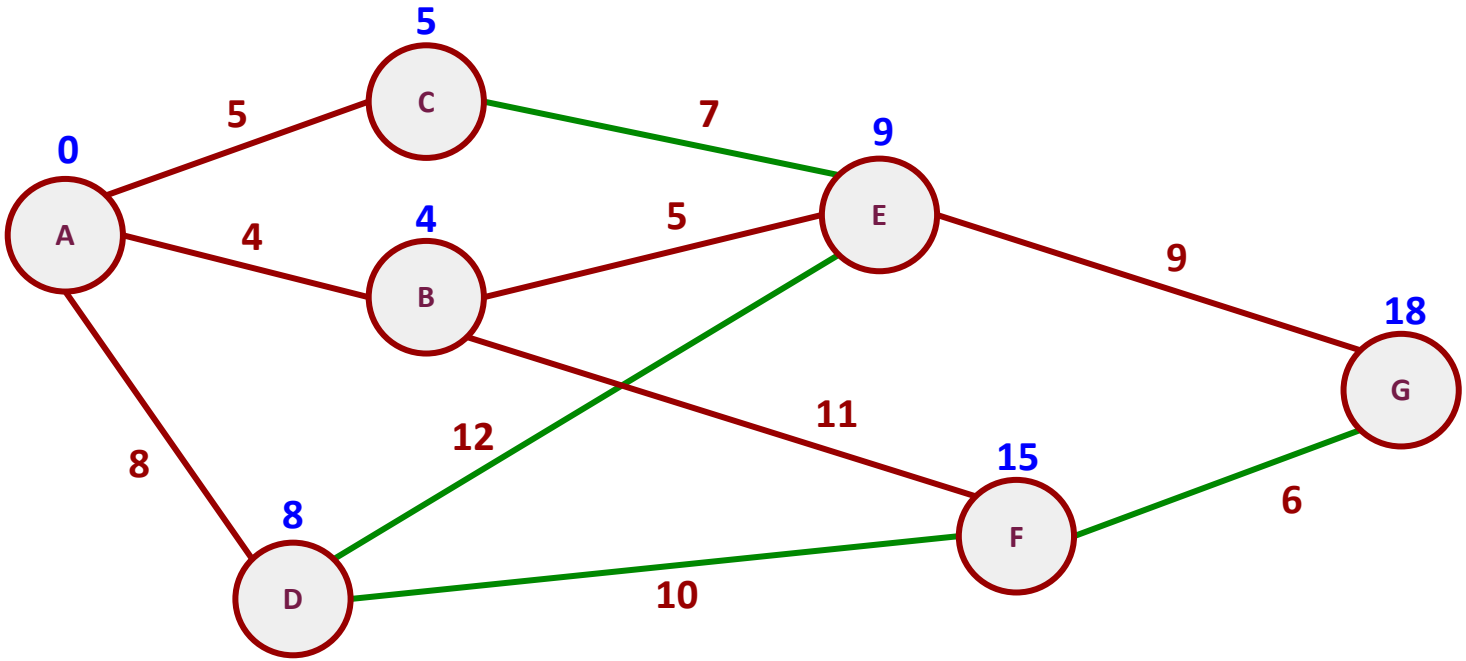
Dijkstra Algoritması



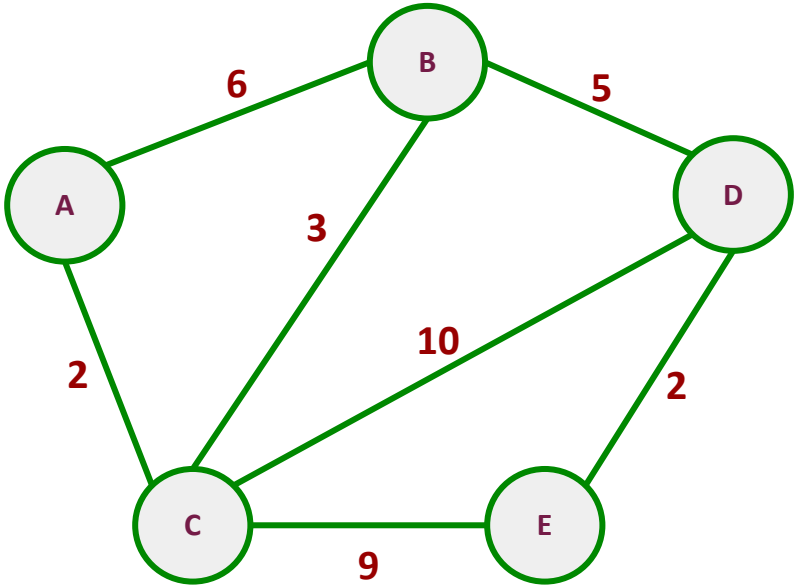
Dijkstra Algoritması



Dijkstra Algoritması

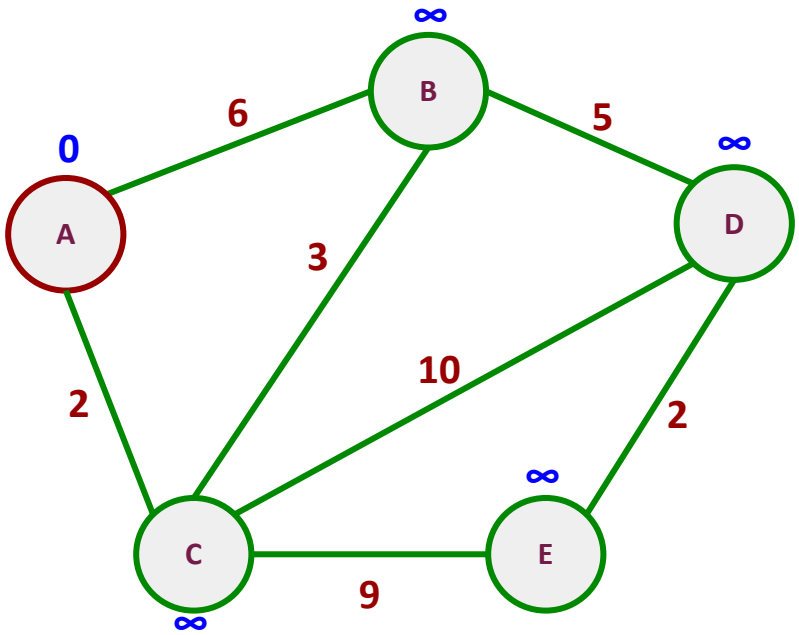


Dijkstra Algoritması



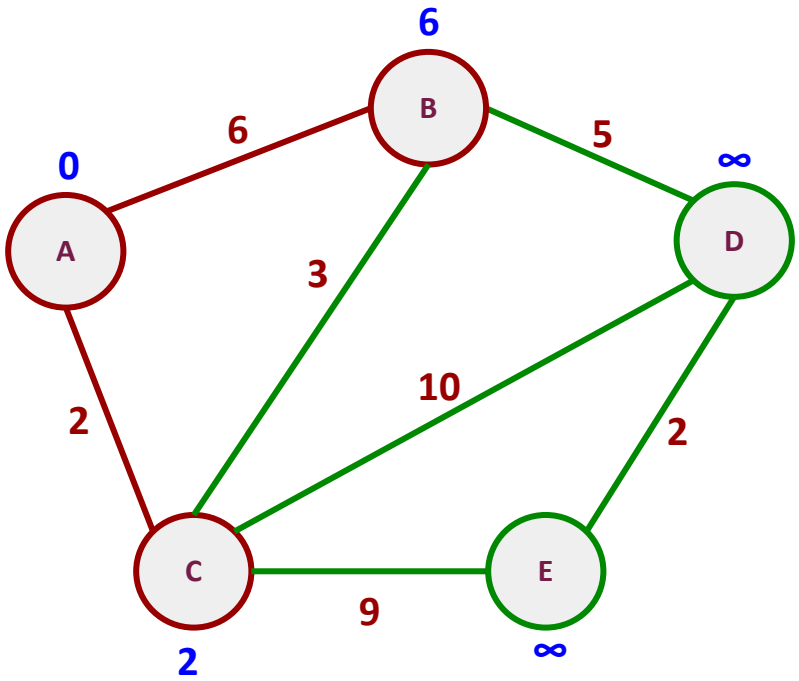
	A	B	C*	D	E

Dijkstra Algoritması



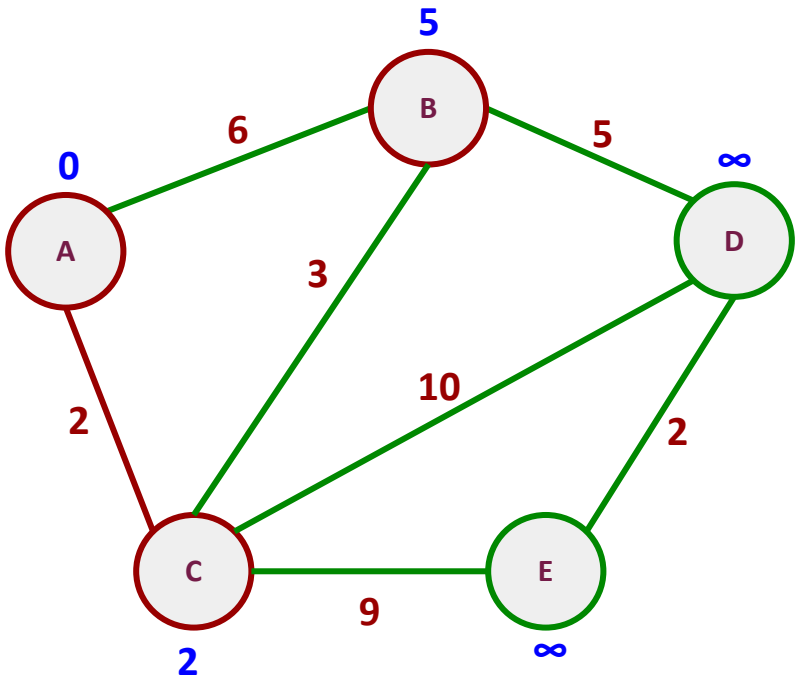
	A	B	C*	D	E
	0	∞	∞	∞	∞

Dijkstra Algoritması



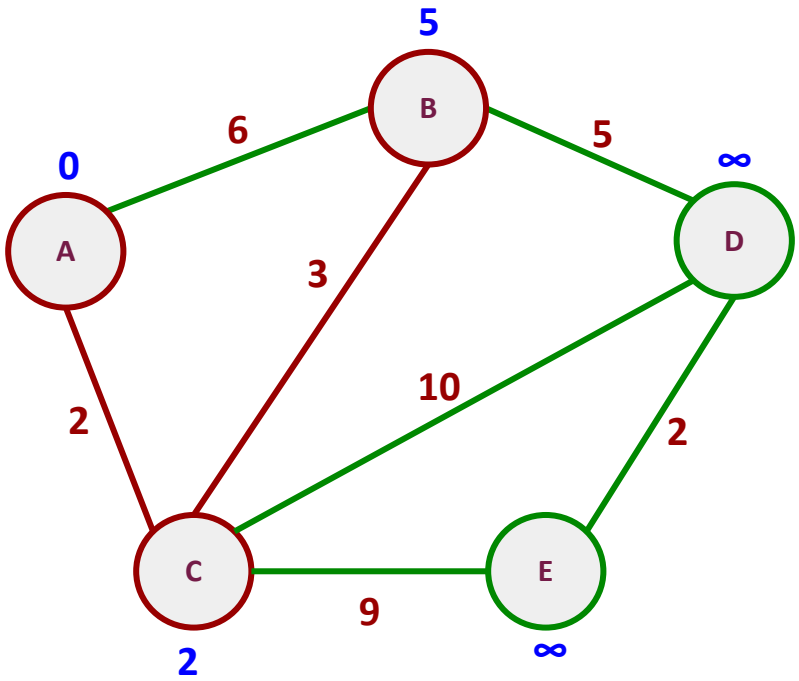
	A*	B	C*	D	E
	0	∞	∞	∞	∞
A	0	6 (A)	2 (A)	∞	∞

Dijkstra Algoritması



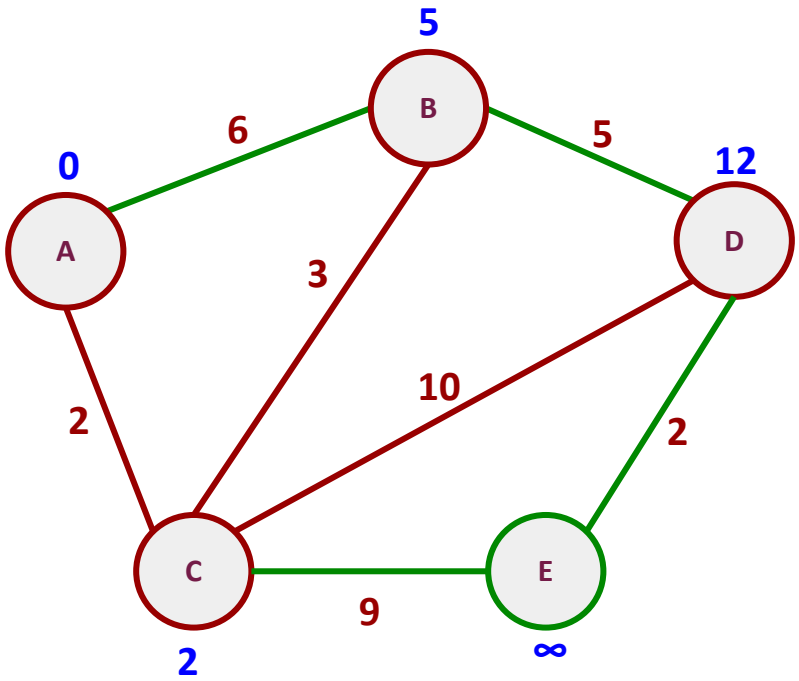
	A*	B	C*	D	E
	0	∞	∞	∞	∞
A	0	6 (A)	2 (A)	∞	∞
C	0		2 (A)		

Dijkstra Algoritması



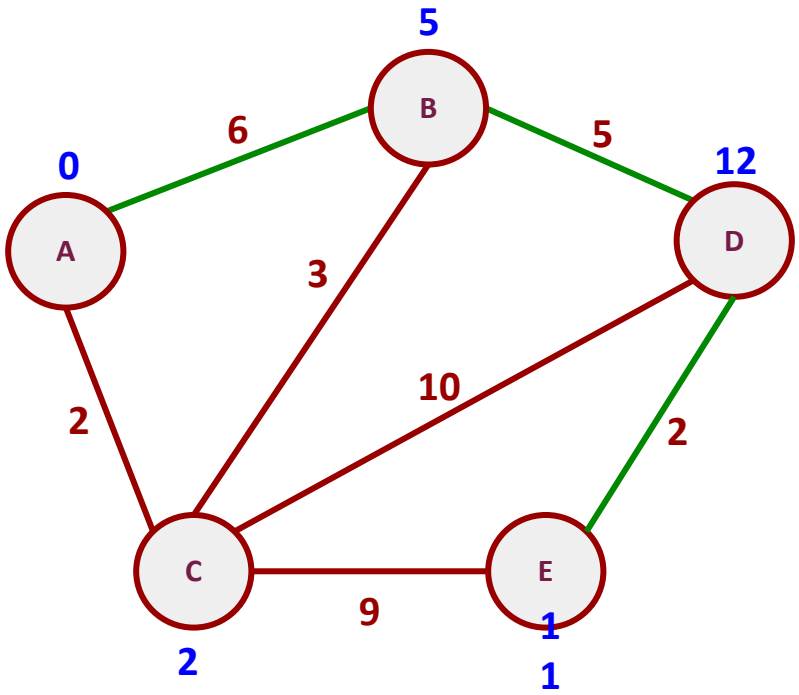
	A*	B	C*	D	E
	0	∞	∞	∞	∞
A	0	6 (A)	2 (A)	∞	∞
C	0	5 (A-C)	2 (A)		

Dijkstra Algoritması



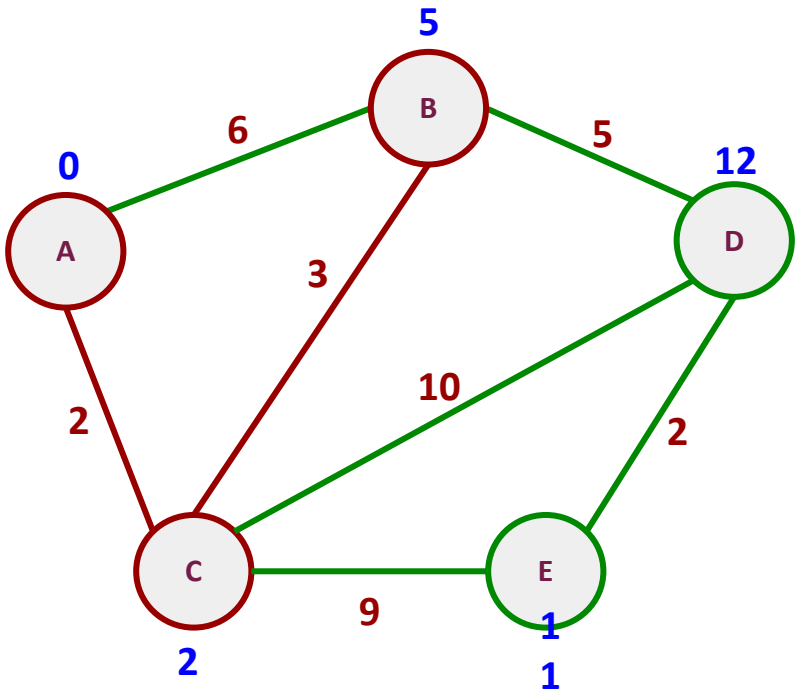
	A*	B	C*	D	E
	0	∞	∞	∞	∞
A	0	6 (A)	2 (A)	∞	∞
C	0	5 (A-C)	2 (A)	12 (A-C-D)	

Dijkstra Algoritması



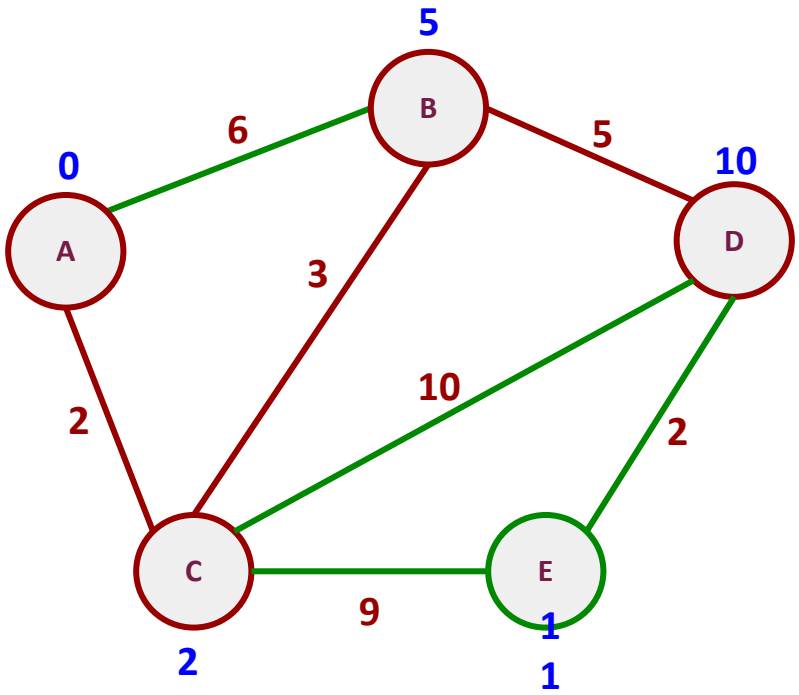
	A*	B	C*	D	E
	0	∞	∞	∞	∞
A	0	6 (A)	2 (A)	∞	∞
C	0	5 (A-C)	2 (A)	12 (A-C-D)	11 (A-C-E)

Dijkstra Algoritması



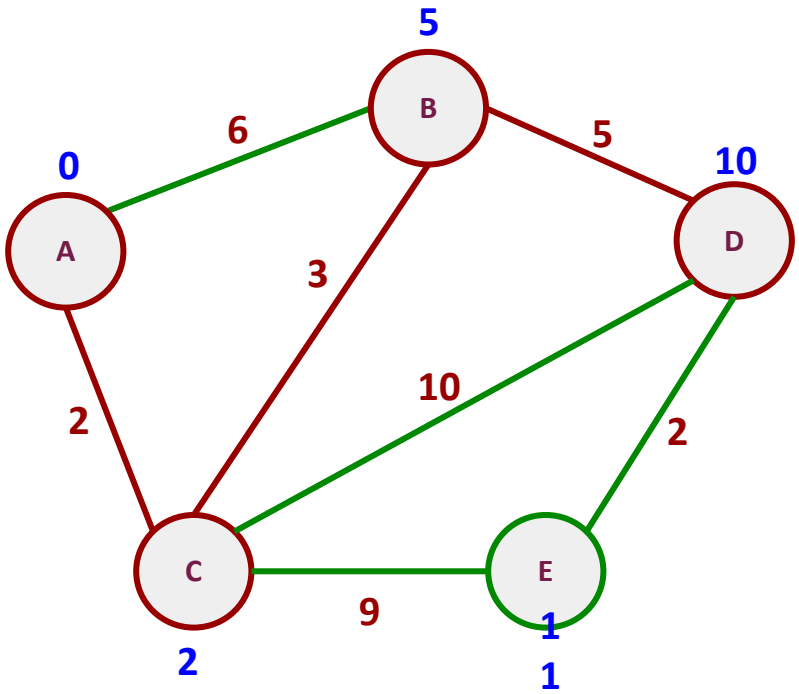
	A*	B	C*	D	E
	0	∞	∞	∞	∞
A	0	6 (A)	2 (A)	∞	∞
C	0	5 (A-C)	2 (A)	12 (A-C-D)	11 (A-C-E)
B	0	5 (A-C)	2 (A)		

Dijkstra Algoritması



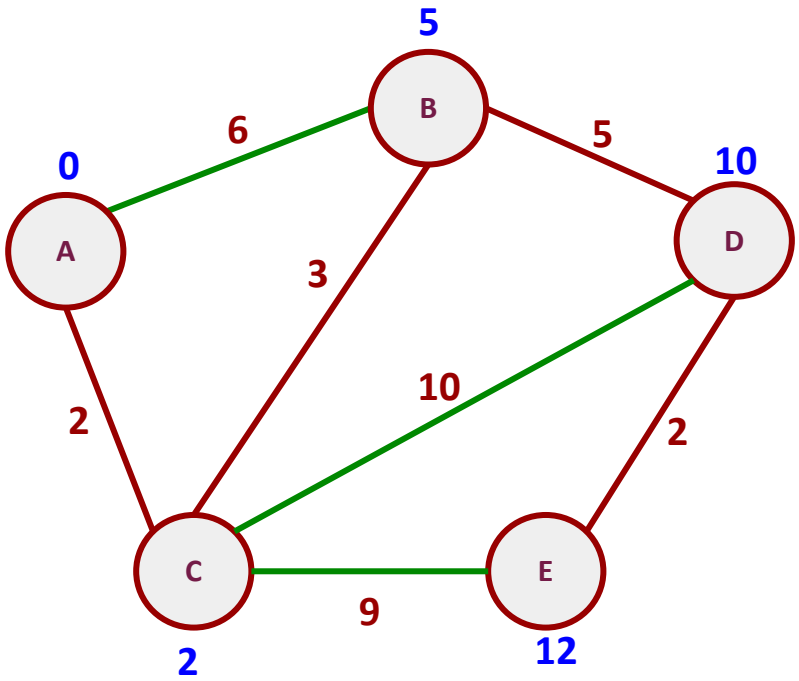
	A*	B*	C*	D	E
	0	∞	∞	∞	∞
A	0	6 (A)	2 (A)	∞	∞
C	0	5 (A-C)	2 (A)	12 (A-C-D)	11 (A-C-E)
B	0	5 (A-C)	2 (A)	10 (A-C-B-D)	11 (A-C-E)

Dijkstra Algoritması



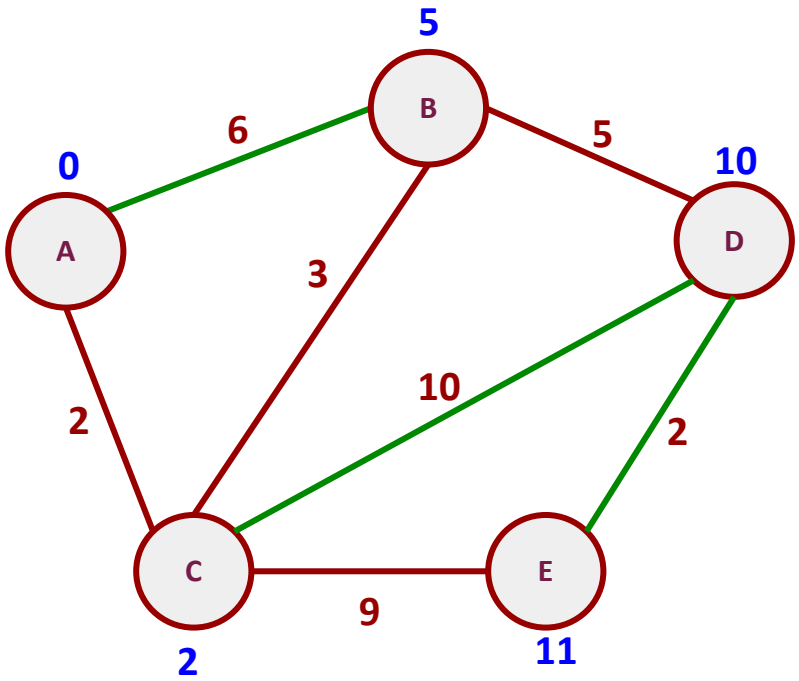
	A*	B*	C*	D*	E
	0	∞	∞	∞	∞
A	0	6 (A)	2 (A)	∞	∞
C	0	5 (A-C)	2 (A)	12 (A-C-D)	11 (A-C-E)
B	0	5 (A-C)	2 (A)	10 (A-C-B-D)	11 (A-C-E)
D	0	5 (A-C)	2 (A)	10 (A-C-B-D)	

Dijkstra Algoritması



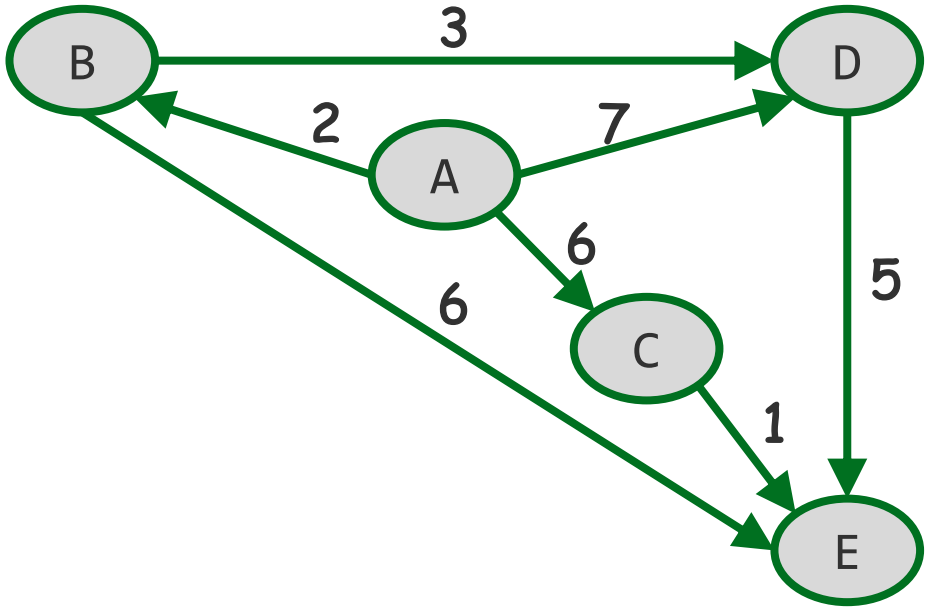
	A*	B*	C*	D*	E
	0	∞	∞	∞	∞
A	0	6 (A)	2 (A)	∞	∞
C	0	5 (A-C)	2 (A)	12 (A-C-D)	11 (A-C-E)
B	0	5 (A-C)	2 (A)	10 (A-C-B-D)	11 (A-C-E)
D	0	5 (A-C)	2 (A)	10 (A-C-B-D)	

Dijkstra Algoritması

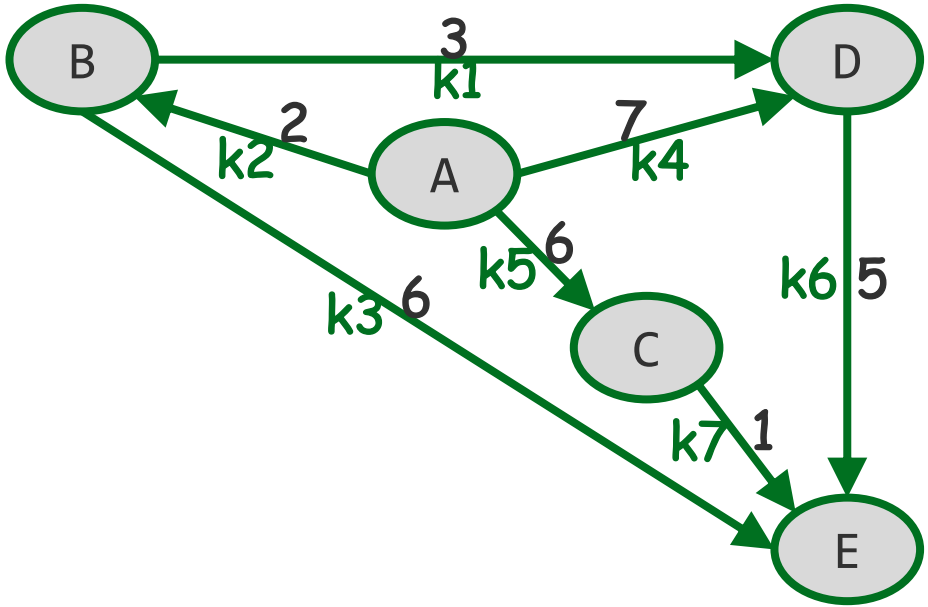


	A*	B*	C*	D*	E
	0	∞	∞	∞	∞
A	0	6 (A)	2 (A)	∞	∞
C	0	5 (A-C)	2 (A)	12 (A-C-D)	11 (A-C-E)
B	0	5 (A-C)	2 (A)	10 (A-C-B-D)	11 (A-C-E)
D	0	5 (A-C)	2 (A)	10 (A-C-B-D)	11 (A-C-E)

Bellman-Ford Algoritması

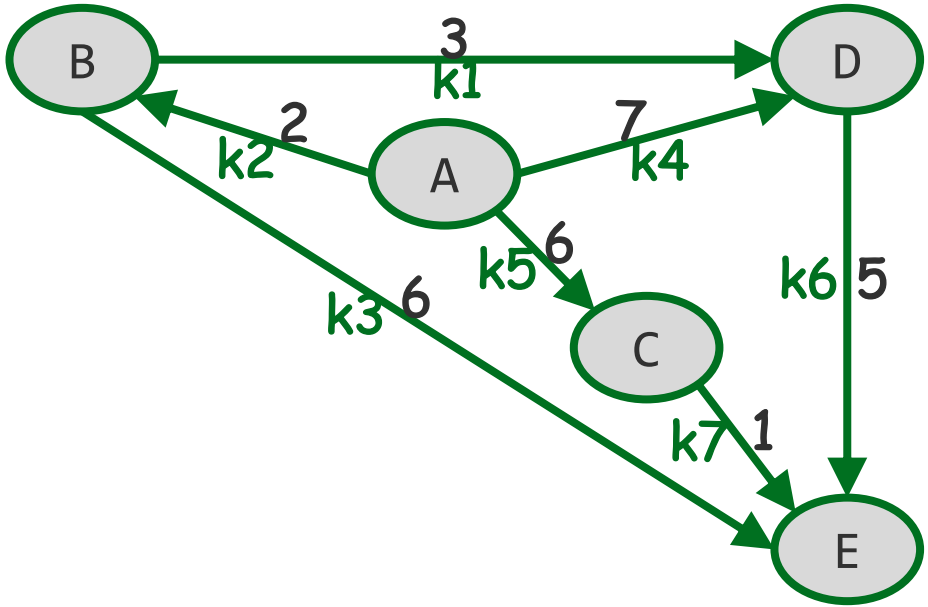


Bellman-Ford Algoritması



k1 : B --> D
k2 : A --> B
k3 : B --> E
k4 : A --> D
k5 : A --> C
k6 : D --> E
k7 : C --> E

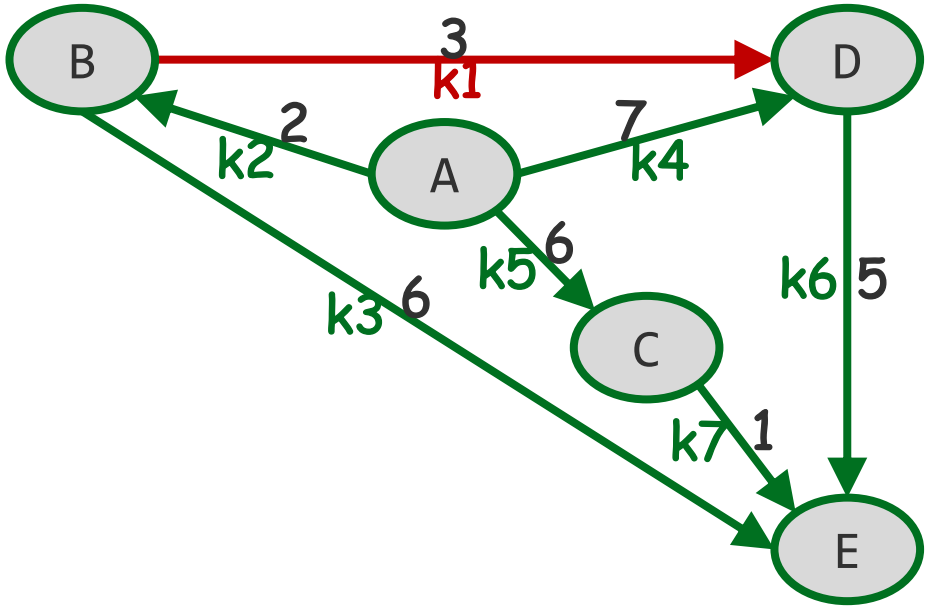
Bellman-Ford Algoritması



k1 : B --> D
k2 : A --> B
k3 : B --> E
k4 : A --> D
k5 : A --> C
k6 : D --> E
k7 : C --> E

	A	B	C	D	E
uzaklık	0	∞	∞	∞	∞
yol					

Bellman-Ford Algoritması



k1 :

B= ∞ D= ∞ yol= 3

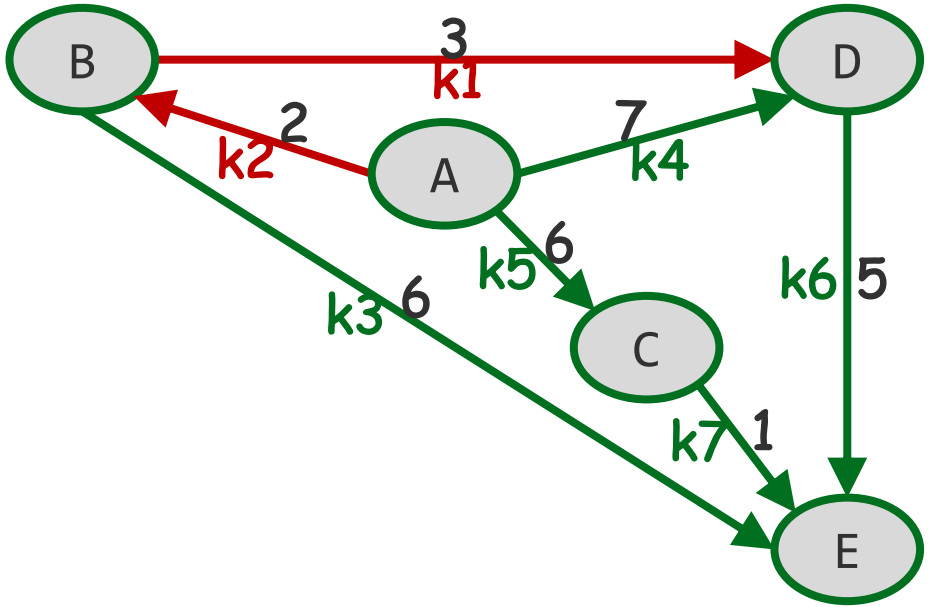
u(A,D)= min(u(B,B)+yol,u(A,D))

= min(∞+3,∞)

= ∞

	A	B	C	D	E
uzaklık	0	∞	∞	∞	∞
yol					

Bellman-Ford Algoritması



k2 :

A= 0 B= ∞ yol= 2

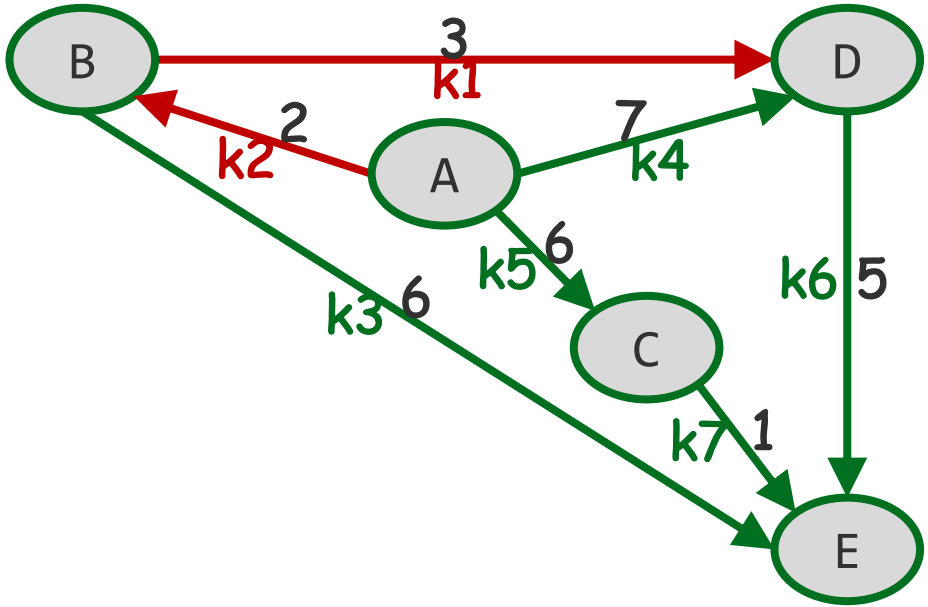
u(A,B)= min(u(A,A)+yol,u(A,B))

= min(0+2,∞)

= 2

	A	B	C	D	E
uzaklık	0	∞	∞	∞	∞
yol					

Bellman-Ford Algoritması



k2 :

A= 0 B= ∞ yol= 2

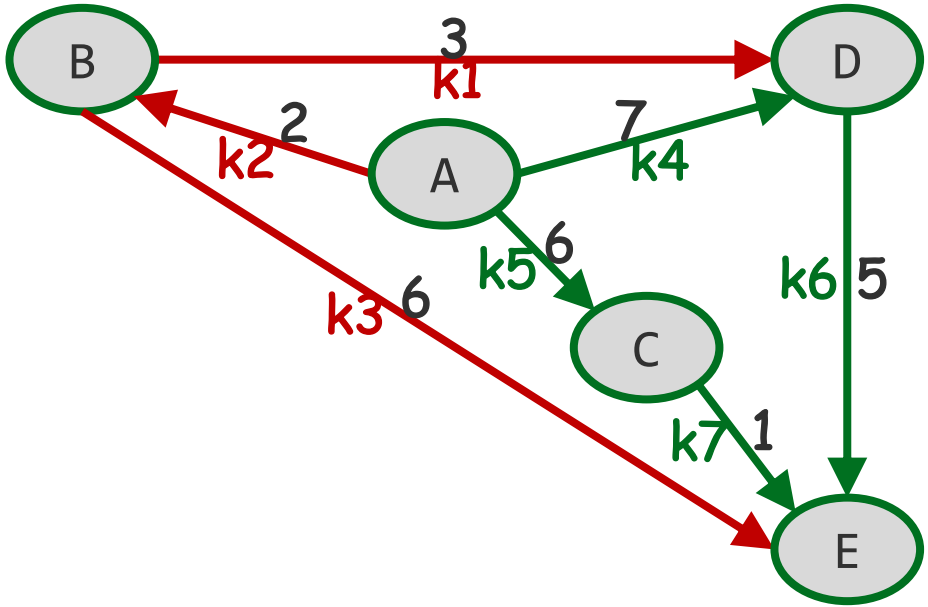
u(A,B)= min(u(A,A)+yol,u(A,B))

= min(0+2,∞)

= 2

	A	B	C	D	E
uzaklık	0	2	∞	∞	∞
yol		A			

Bellman-Ford Algoritması



k3 :

B= 2 E= ∞ yol= 6

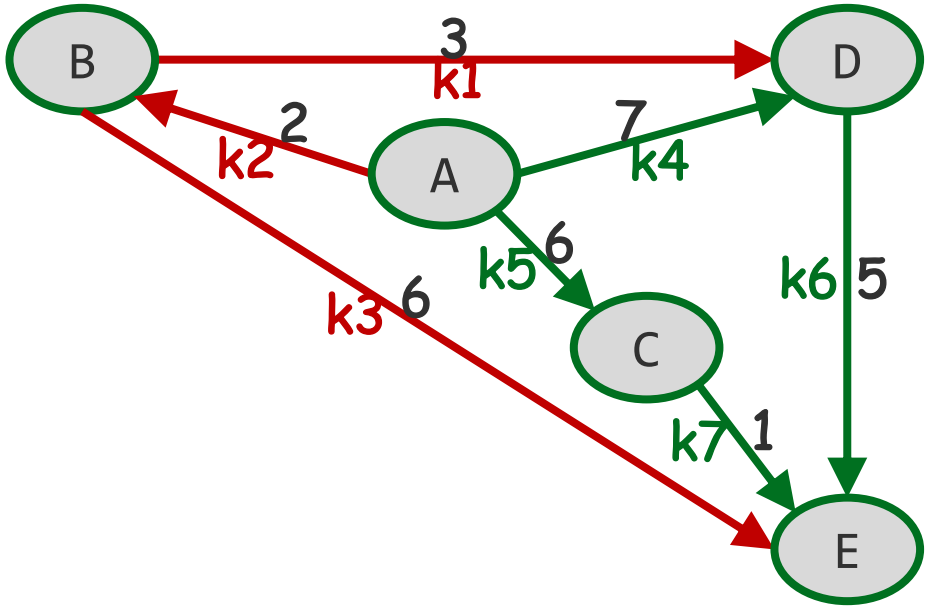
u(A,E)= min(u(A,B)+yol,u(A,E))

= min(2+6,∞)

= 8

	A	B	C	D	E
uzaklık	0	2	∞	∞	∞
yol		A			

Bellman-Ford Algoritması



k3 :

B= 2 E= ∞ yol= 6

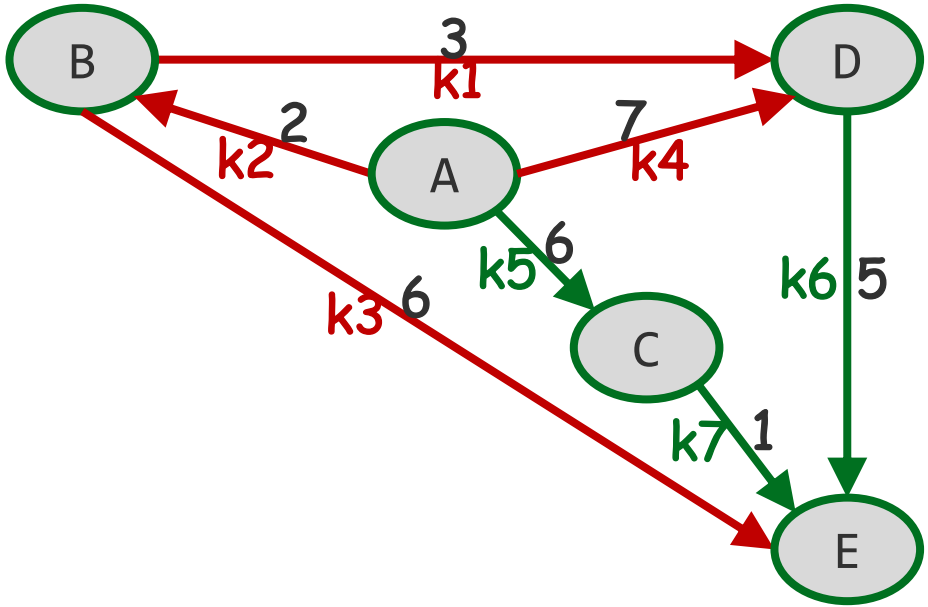
u(A,E)= min(u(A,B)+yol,u(A,E))

= min(2+6,∞)

= 8

	A	B	C	D	E
uzaklık	0	2	∞	∞	8
yol		A			A-B

Bellman-Ford Algoritması



k4 :

A= 0 D= ∞ yol= 7

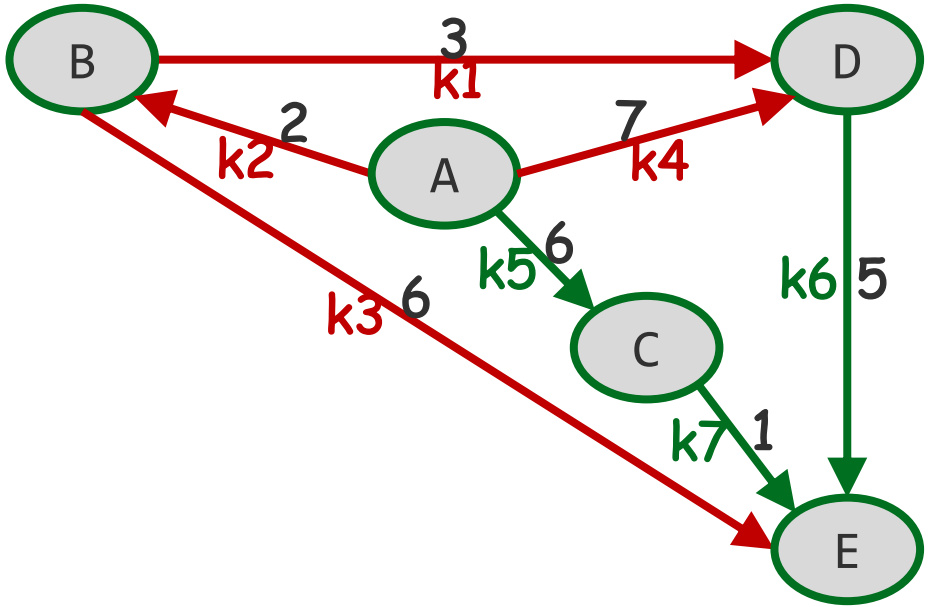
u(A,D)= min(u(A,A)+yol,u(A,D))

= min(0+7,∞)

= 7

	A	B	C	D	E
uzaklık	0	2	∞	∞	8
yol		A			A-B

Bellman-Ford Algoritması



k4 :

A= 0 D= ∞ yol= 7

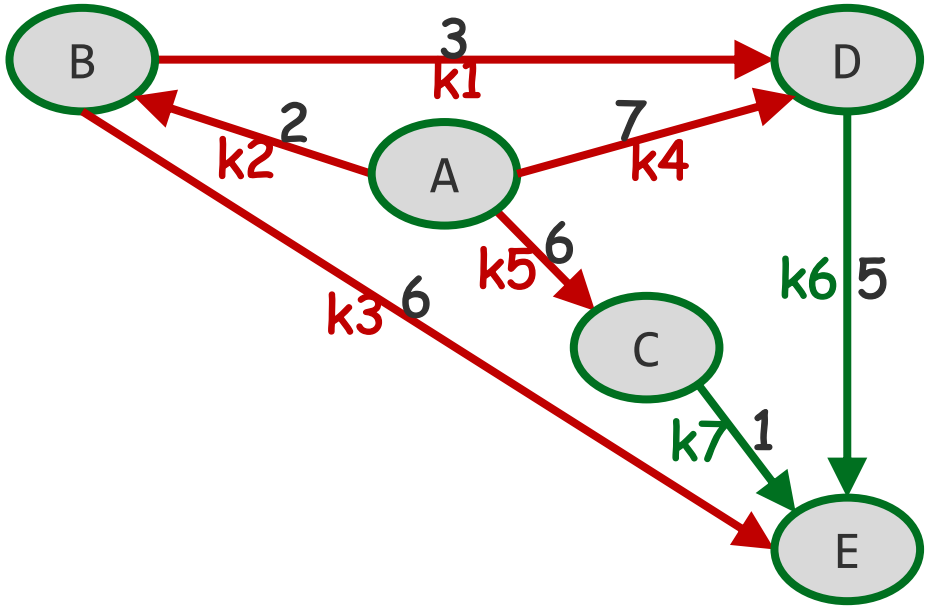
u(A,D)= min(u(A,A)+yol,u(A,D))

= min(0+7,∞)

= 7

	A	B	C	D	E
uzaklık	0	2	∞	7	8
yol		A		A	A-B

Bellman-Ford Algoritması



k5 :

A= 0 C= ∞ yol= 6

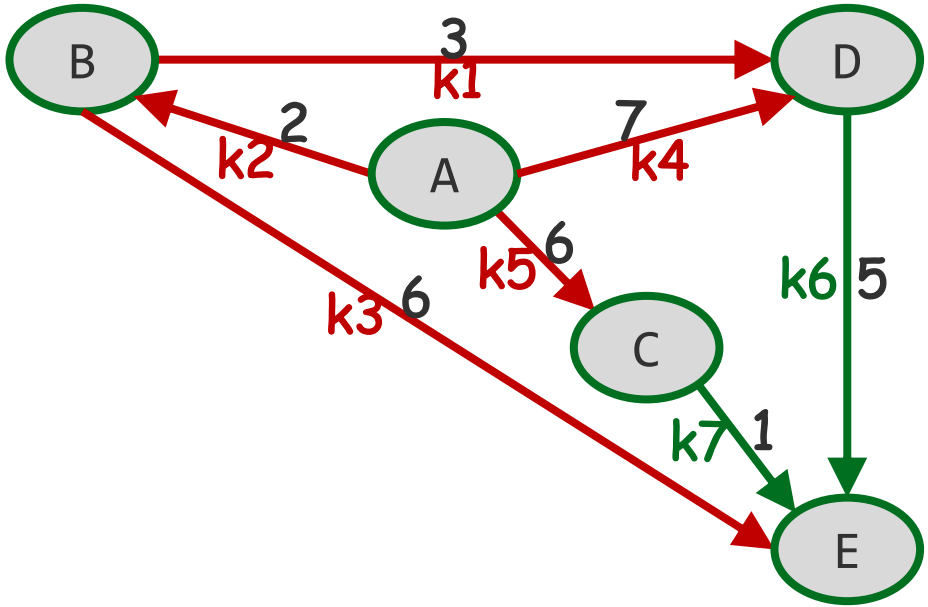
u(A,C)= min(u(A,A)+yol,u(A,C))

= min(0+6,∞)

= 6

	A	B	C	D	E
uzaklık	0	2	∞	7	8
yol		A		A	A-B

Bellman-Ford Algoritması



k5 :

A= 0 C= ∞ yol= 6

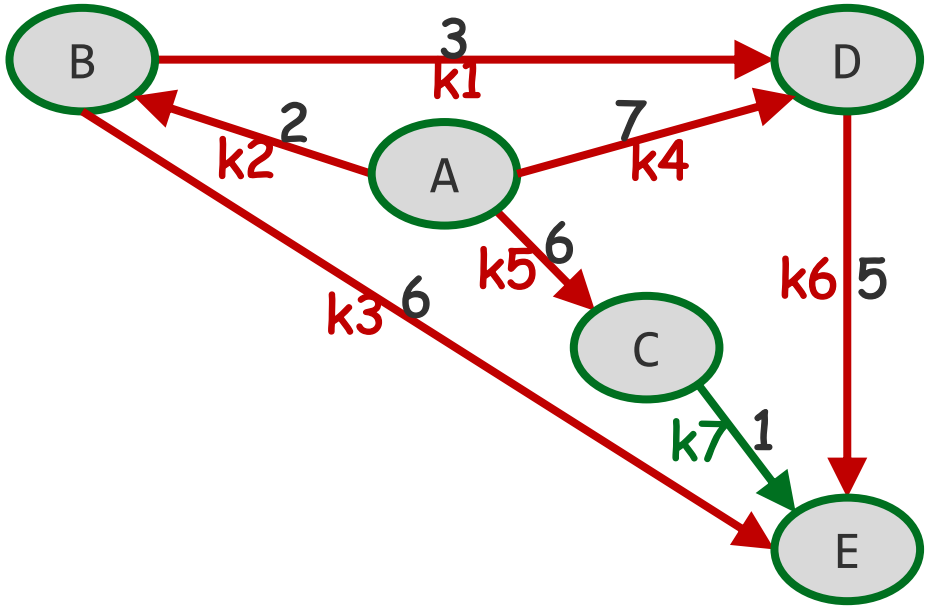
u(A,C)= min(u(A,A)+yol,u(A,C))

= min(0+6,∞)

= 6

	A	B	C	D	E
uzaklık	0	2	6	7	8
yol		A	A	A	A-B

Bellman-Ford Algoritması



k6 :

D= 7 E= 8 yol= 5

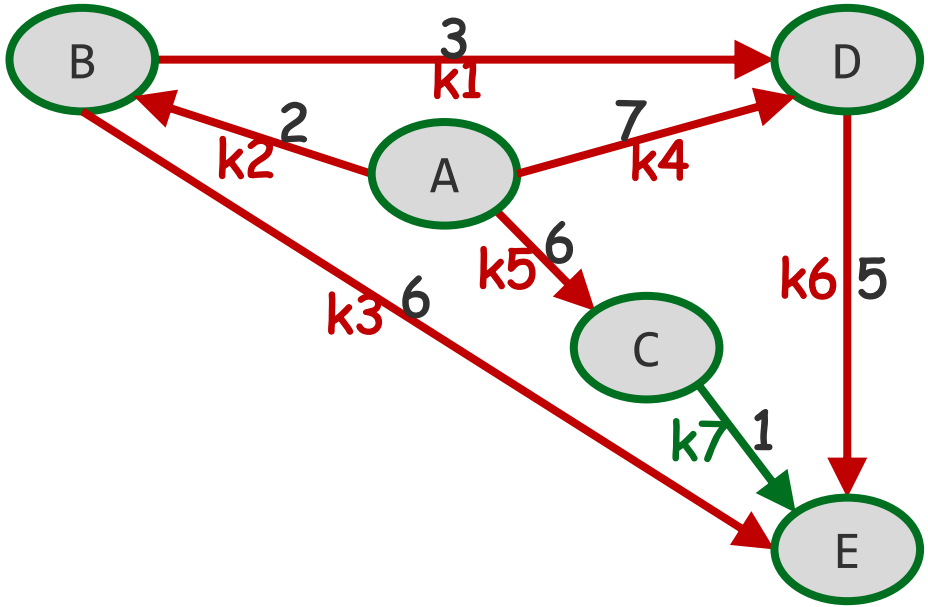
u(A,E)= min(u(A,D)+yol,u(A,E))

= min(7+5,8)

= 8 (Değişmedi)

	A	B	C	D	E
uzaklık	0	2	6	7	8
yol		A	A	A	A-B

Bellman-Ford Algoritması



k6 :

D= 7 E= 8 yol= 5

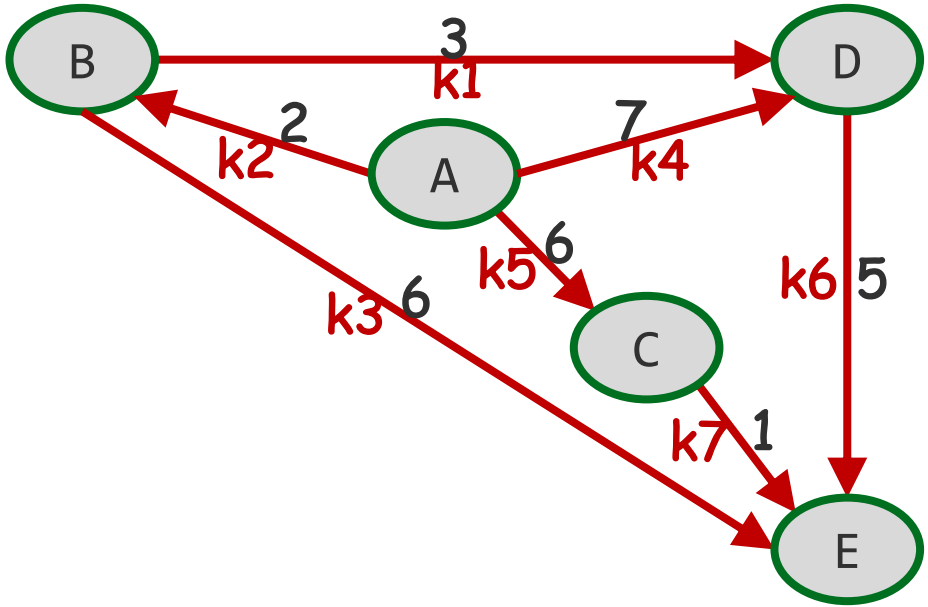
u(A,E)= min(u(A,D)+yol,u(A,E))

= min(7+5,8)

= 8 (Değişmedi)

	A	B	C	D	E
uzaklık	0	2	6	7	8
yol		A	A	A	A-B

Bellman-Ford Algoritması



k7 :

C= 6 D= 7 yol= 1

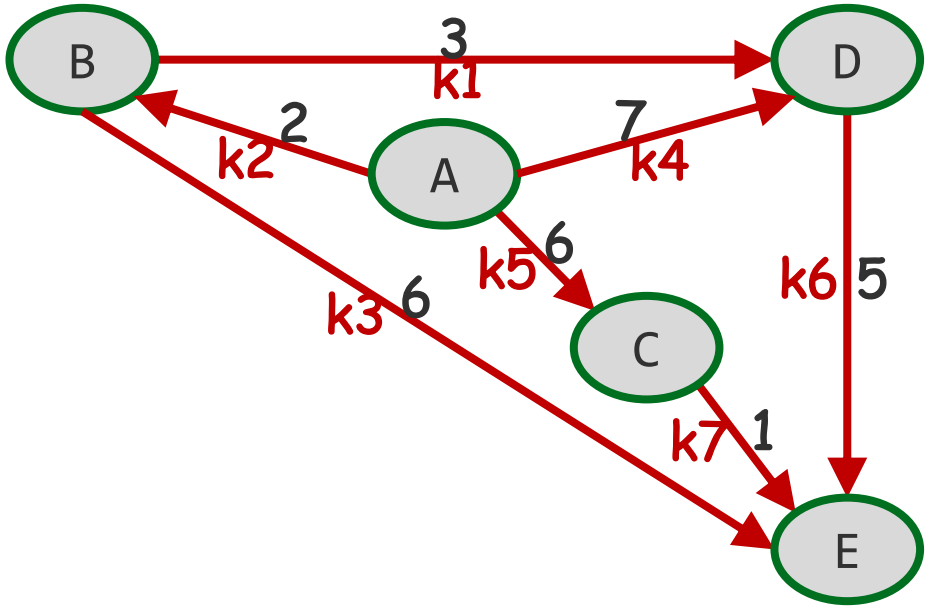
u(A,E)= min(u(A,C)+yol,u(A,E))

= min(6+1,7)

= 7 (Değişmedi)

	A	B	C	D	E
uzaklık	0	2	6	7	8
yol		A	A	A	A-B

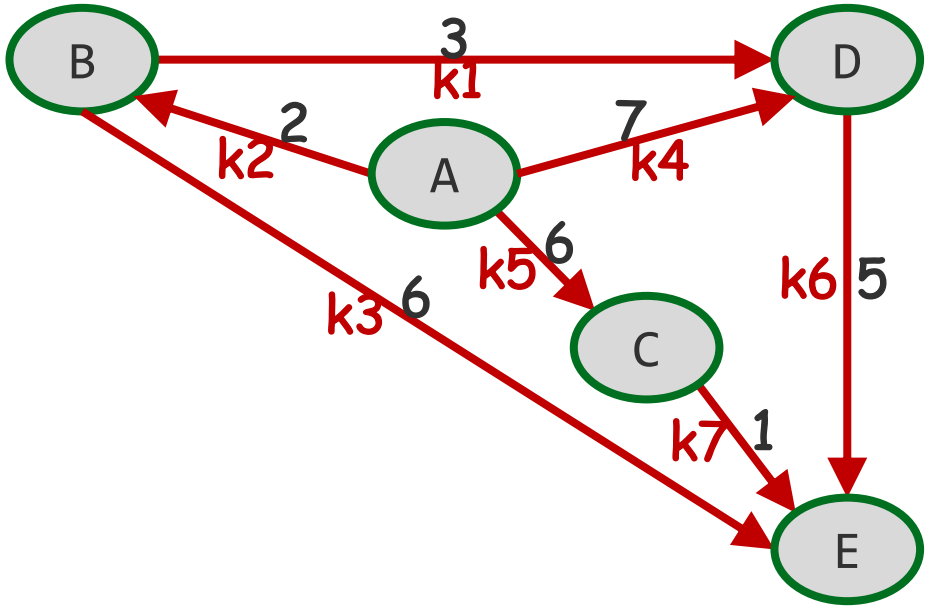
Bellman-Ford Algoritması



Düğümlerin değeri değişmediğinde döngü durur.

	A	B	C	D	E
uzaklık	0	2	6	7	8
yol		A	A	A	A-B

Bellman-Ford Algoritması



A'dan B'ye : 2 (A-B)

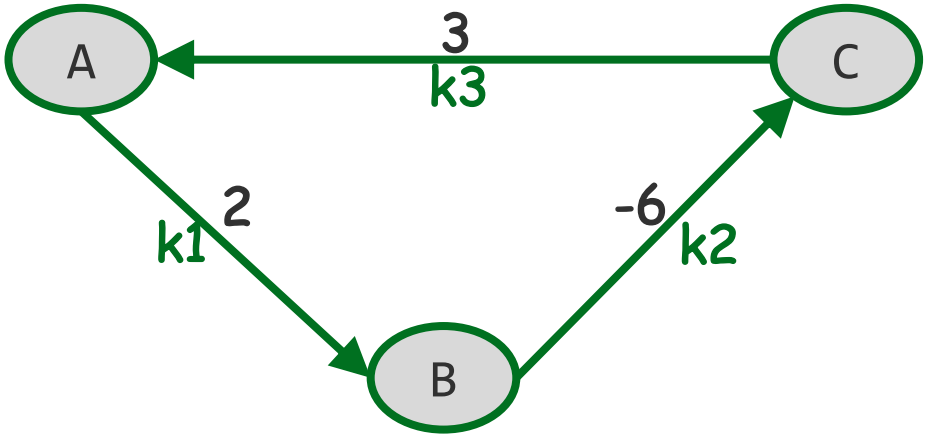
A'dan C'ye : 6 (A-C)

A'dan D'ye : 5 (A-B-D)

A'dan E'ye : 7 (A-C-E)

	A	B	C	D	E
uzaklık	0	2	6	7	8
yol		A	A	A	A-B

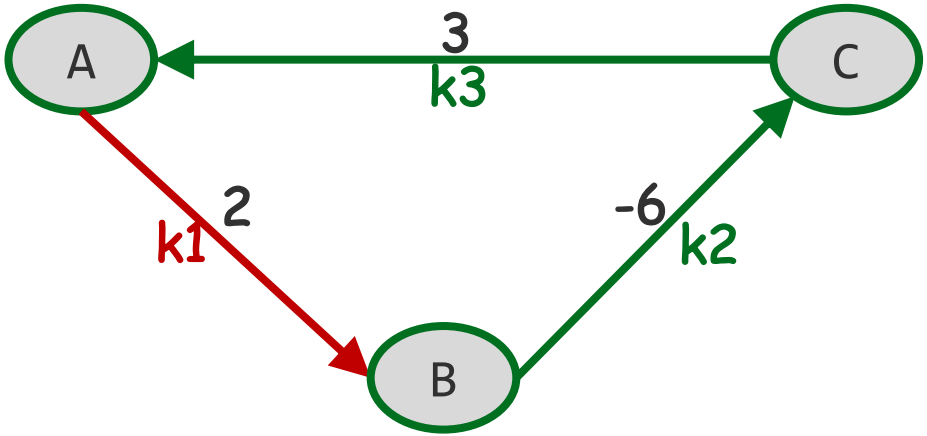
Bellman-Ford Algoritması



k1 : A --> B
k2 : B --> C
k3 : C --> A

	A	B	C
uzaklık	0	∞	∞
yol			

Bellman-Ford Algoritması



k1 :

A= 0 B= ∞ yol= 2

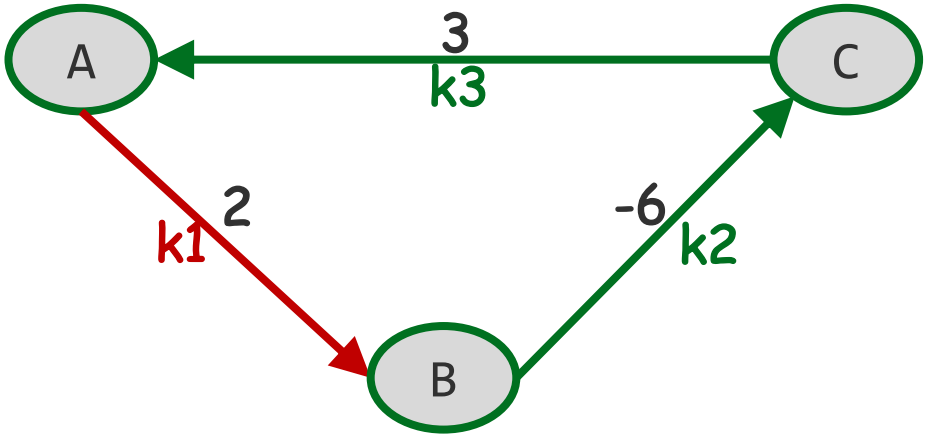
u(A,B)= min(u(A,A)+yol,u(A,B))

= min(0+2,∞)

= 2

	A	B	C
uzaklık	0	∞	∞
yol			

Bellman-Ford Algoritması



k1 :

A= 0 B= ∞ yol= 2

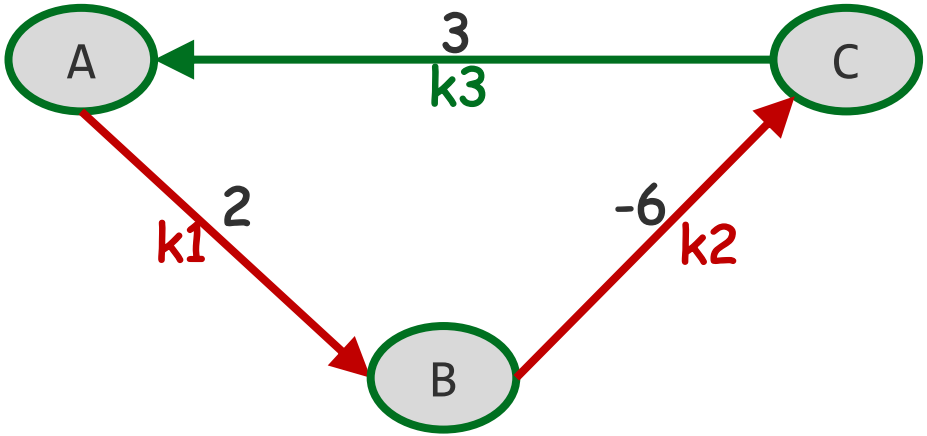
u(A,B)= min(u(A,A)+yol,u(A,B))

= min(0+2,∞)

= 2

	A	B	C
uzaklık	0	2	∞
yol		A	

Bellman-Ford Algoritması



k2 :

B= 2 C= ∞ yol= -6

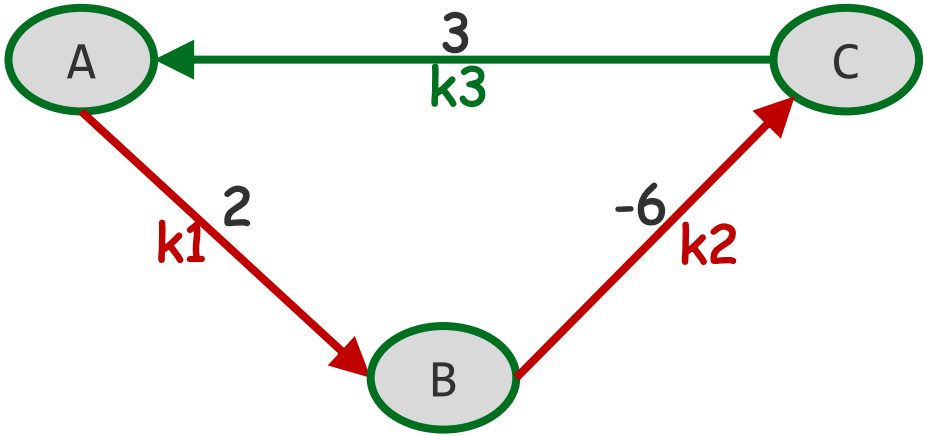
u(A,C)= min(u(A,B)+yol,u(A,C))

= min(2+(-6),∞)

= -4

	A	B	C
uzaklık	0	2	∞
yol		A	

Bellman-Ford Algoritması



k2 :

B= 2 C= ∞ yol= -6

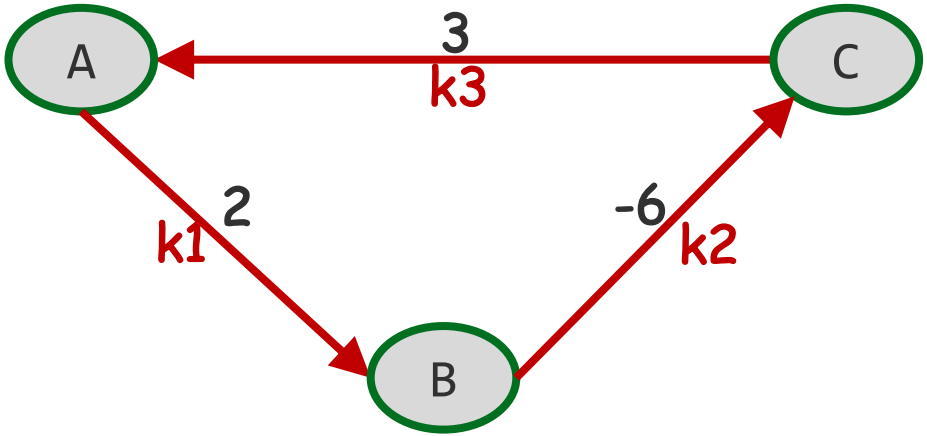
u(A,C)= min(u(A,B)+yol,u(A,C))

= min(2+(-6),∞)

= -4

	A	B	C
uzaklık	0	2	-4
yol		A	A-B

Bellman-Ford Algoritması



k3 :

C= 2 A= 0 yol= 3

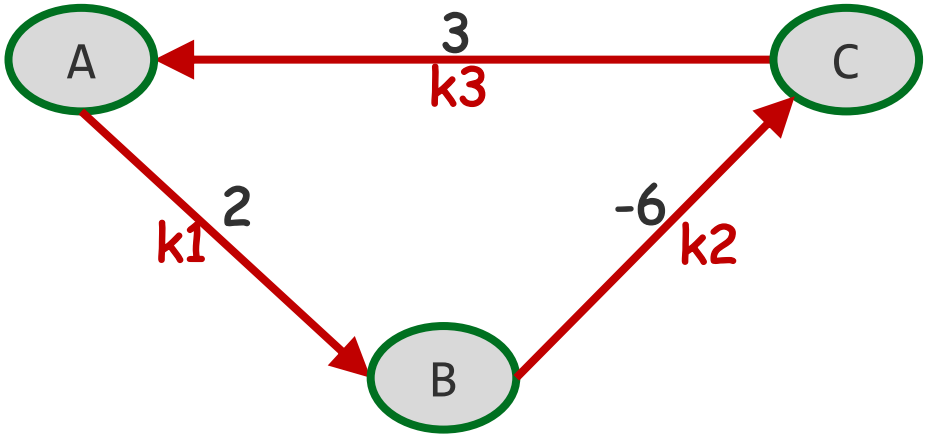
u(A,A)= min(u(A,C)+yol,u(A,A))

= min((-4)+3,0)

= -1

	A	B	C
uzaklık	0	2	-4
yol		A	A-B

Bellman-Ford Algoritması



k3 :

C= 2 A= 0 yol= 3

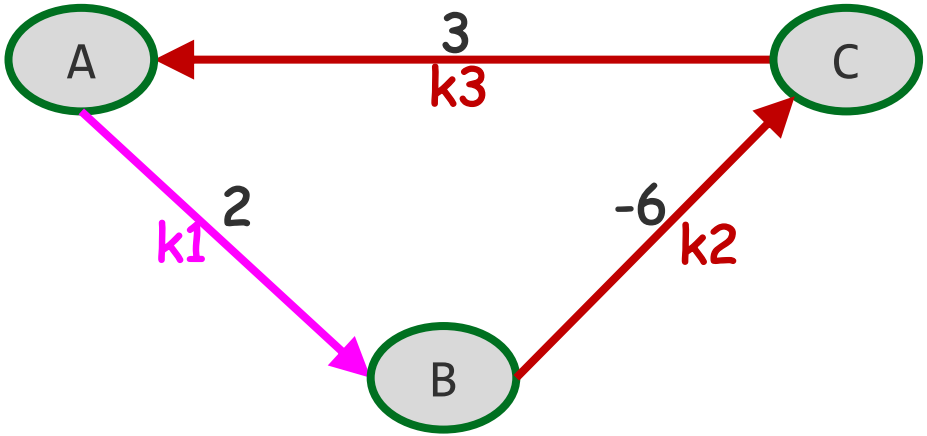
u(A,A)= min(u(A,C)+yol,u(A,A))

= min((-4)+3,0)

= -1

	A	B	C
uzaklık	-1	2	-4
yol	A-B-C	A	A-B

Bellman-Ford Algoritması



NEGATİF ÇEVRİM:

- Aynı sırayla kenarlar dolaş
- Farklı bir değer çıkıyor mu?
- En kısa yol hesaplanmaz

k1 :

C= -4 A= -1 yol= 2

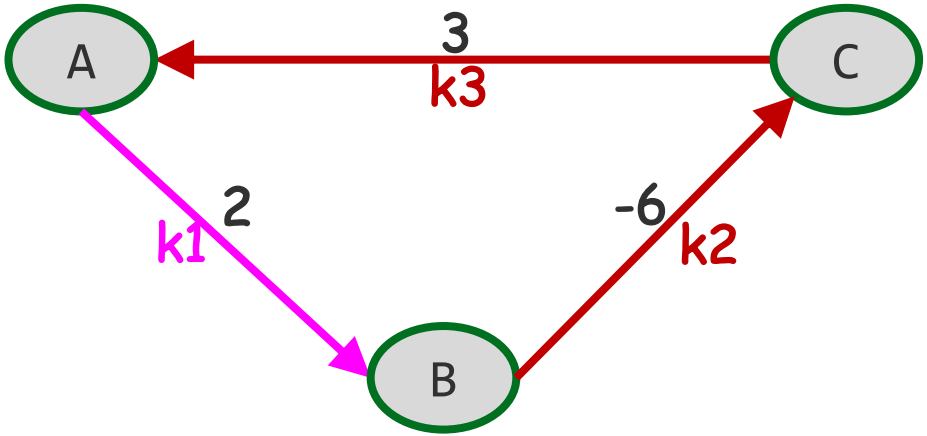
u(A,B)= min(u(A,A)+yol,u(A,B))

= min((-1)+2,2)

= 1

	A	B	C
uzaklık	-1	1	-4
yol	A-B-C	A	A-B

Bellman-Ford Algoritması



NEGATİF ÇEVRİM:

- Aynı sırayla kenarlar dolaş
- Farklı bir değer çıkıyor mu?
- En kısa yol hesaplanmaz

k1 :

C= -4 A= -1 yol= 2

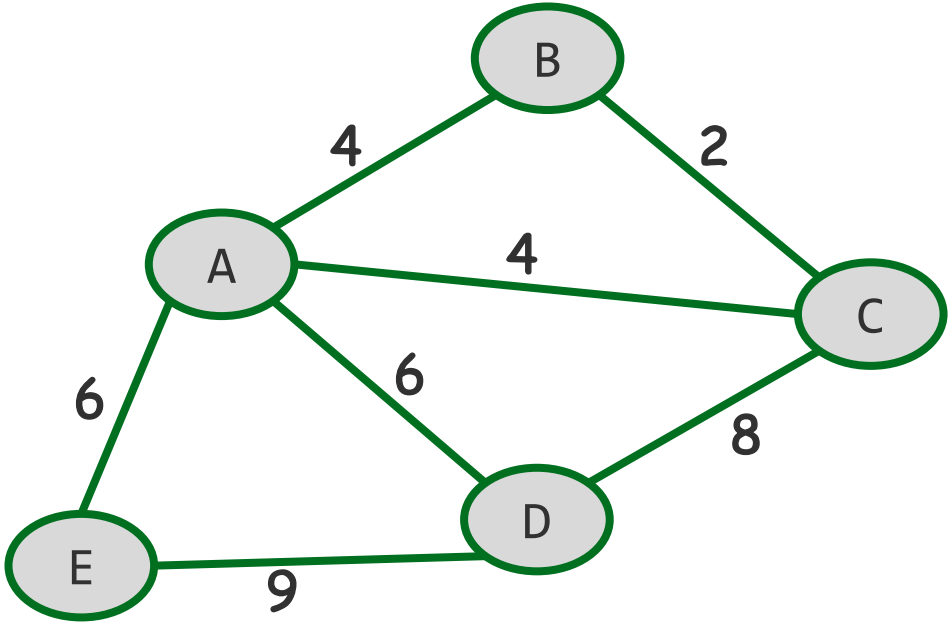
u(A,B)= min(u(A,A)+yol,u(A,B))

= min((-1)+2,2)

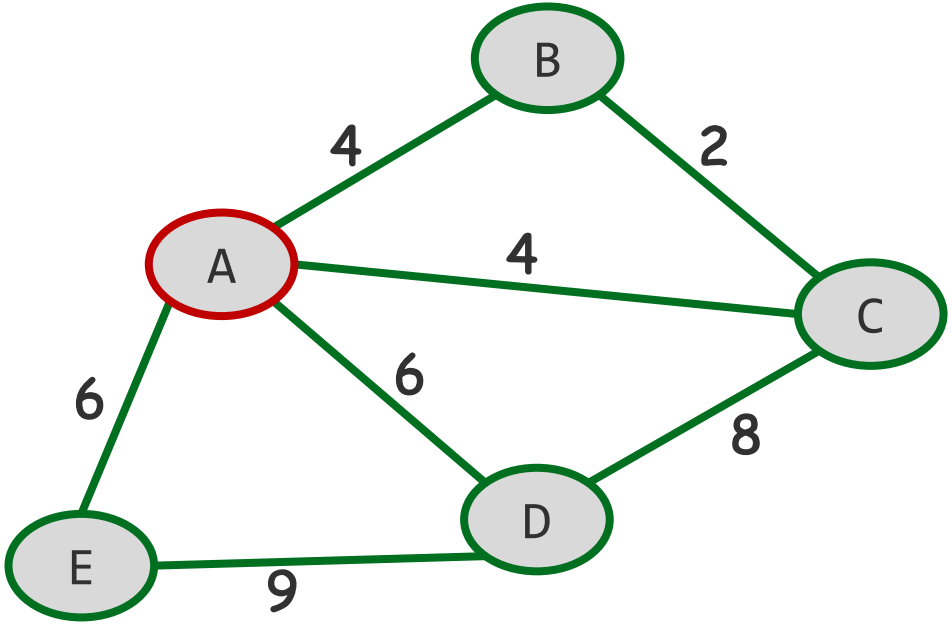
= 1

	A	B	C
uzaklık	-1	1	-4
yol	A-B-C	A	A-B

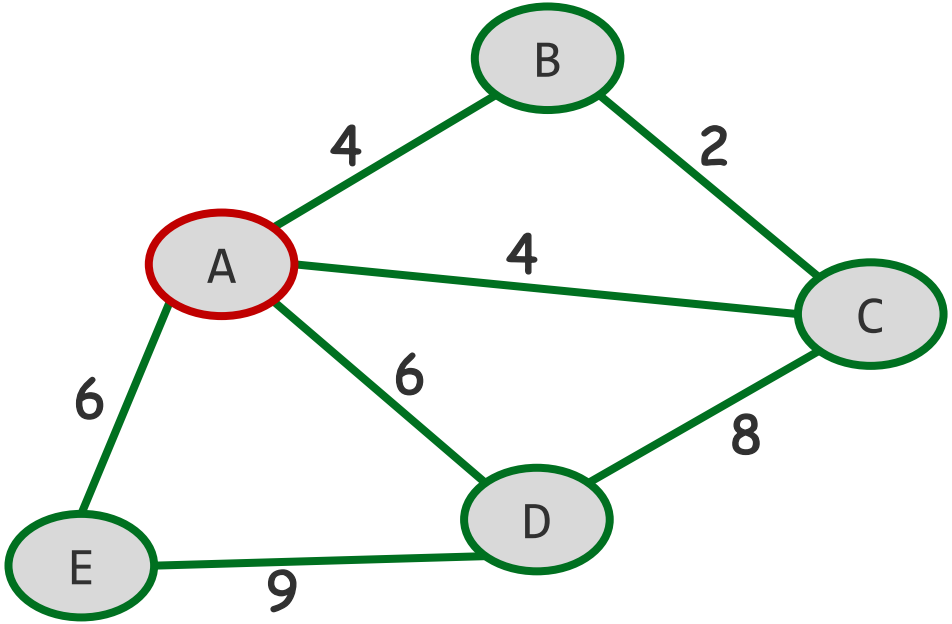
Prim En Kısa Yol Ağacı Algoritması



Prim En Kısa Yol Ağacı Algoritması

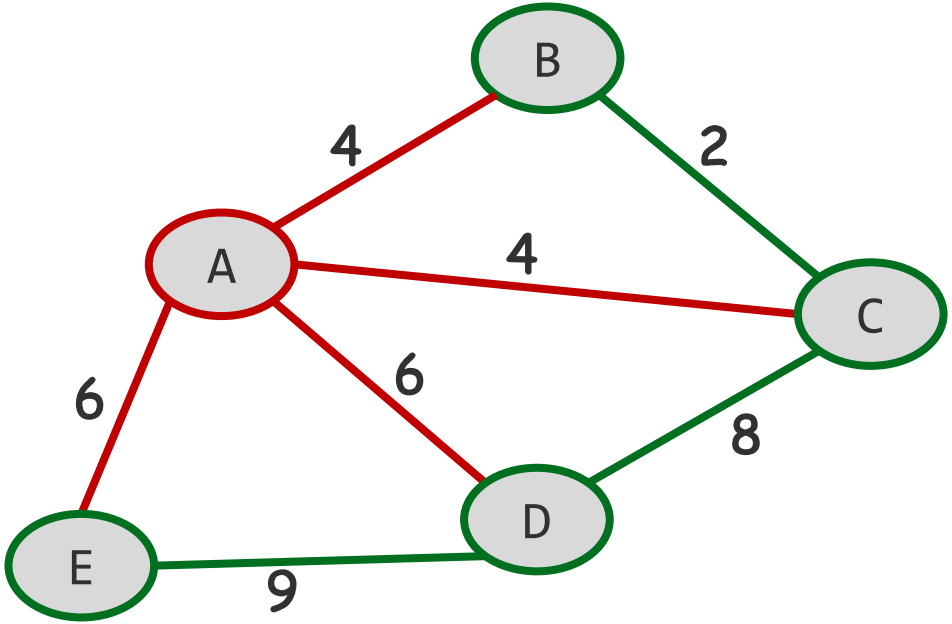


Prim En Kısa Yol Ağacı Algoritması



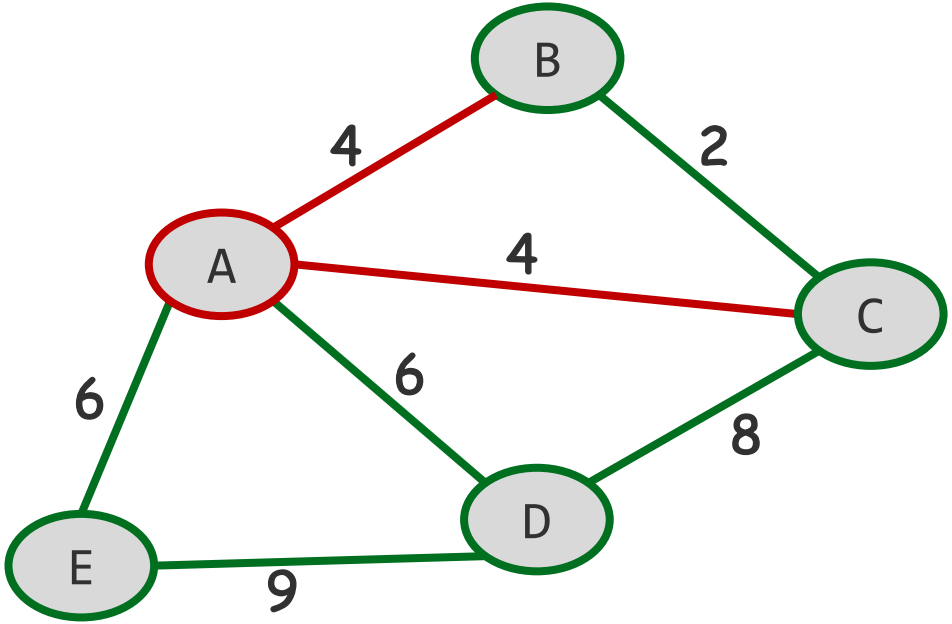
Ziyaret Edilmişler	A	B	C		

Prim En Kısa Yol Ağacı Algoritması



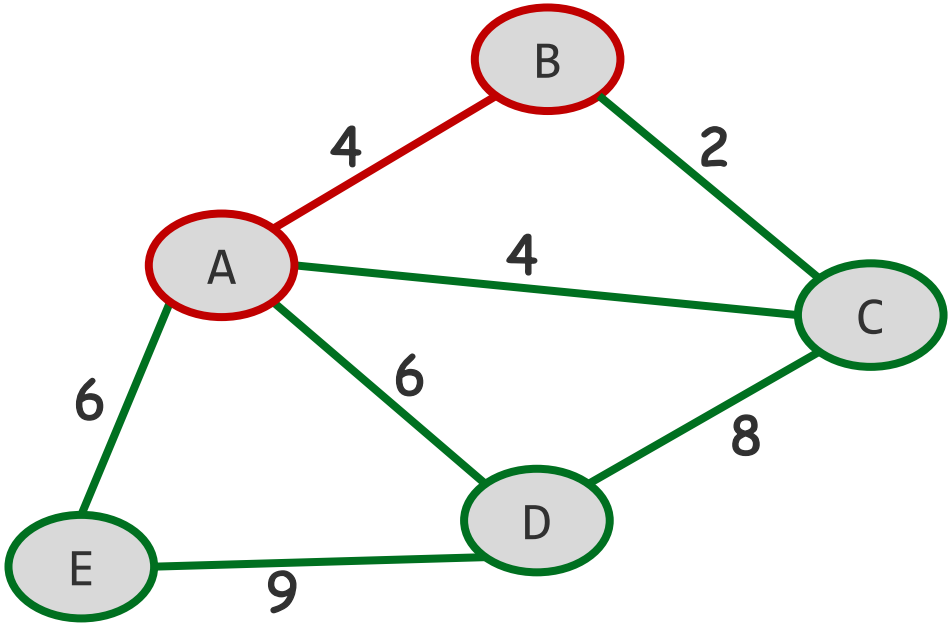
Ziyaret Edilmişler	A	B	C		

Prim En Kısa Yol Ağacı Algoritması



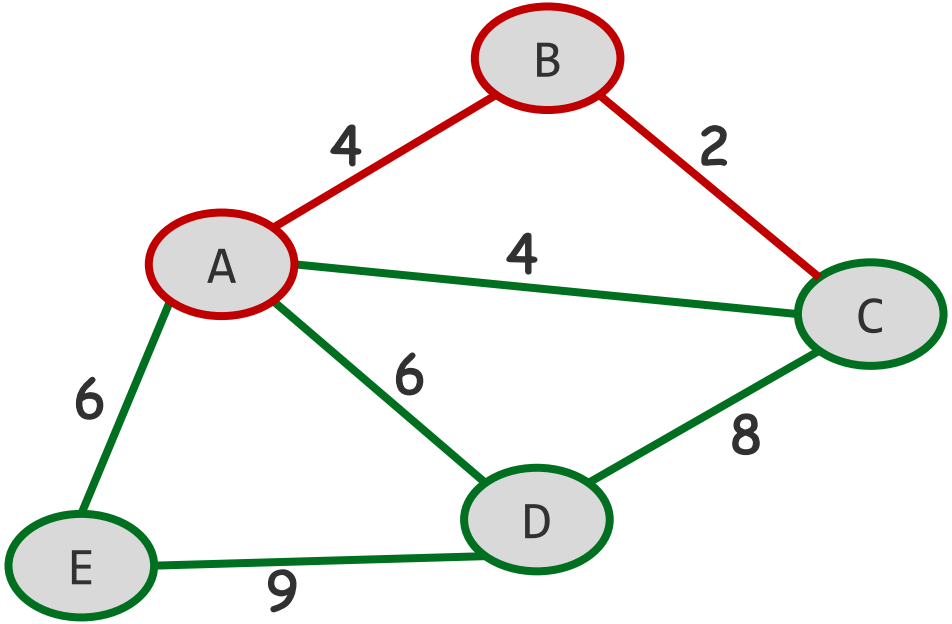
Ziyaret Edilmişler	A	B	C		

Prim En Kısa Yol Ağacı Algoritması



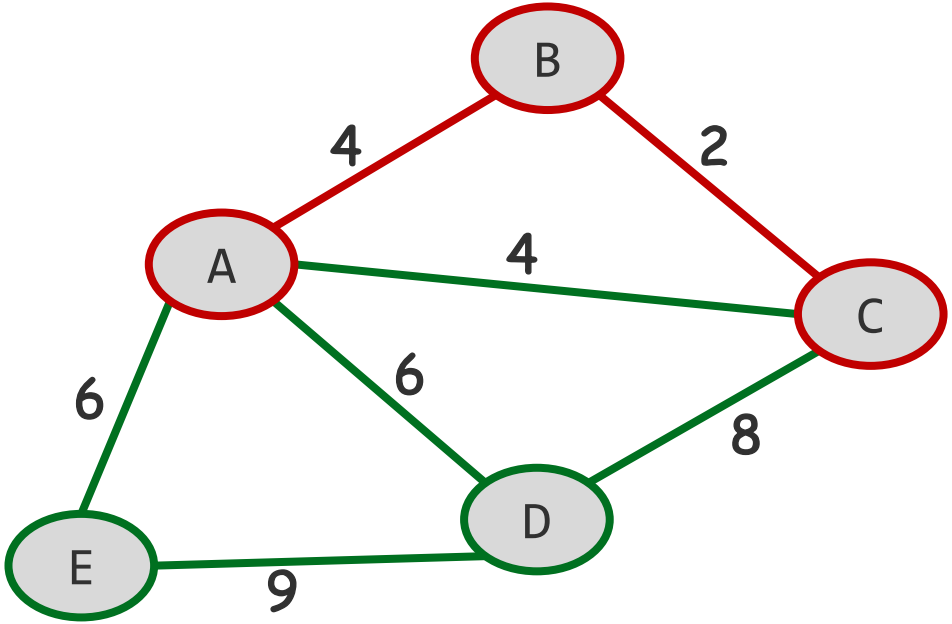
Ziyaret Edilmişler	A	B	C		

Prim En Kısa Yol Ağacı Algoritması



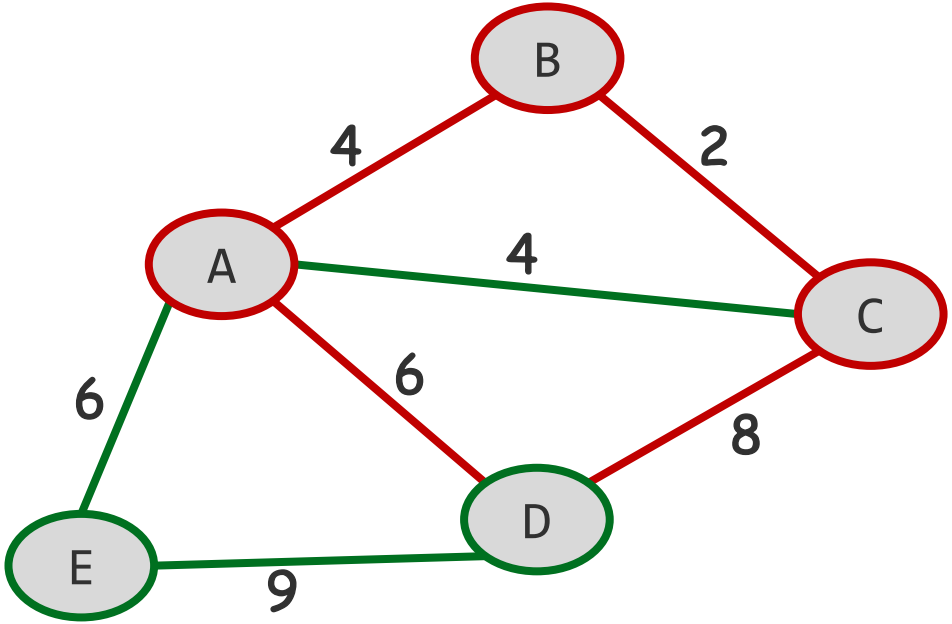
Ziyaret Edilmişler	A	B	C		

Prim En Kısa Yol Ağacı Algoritması



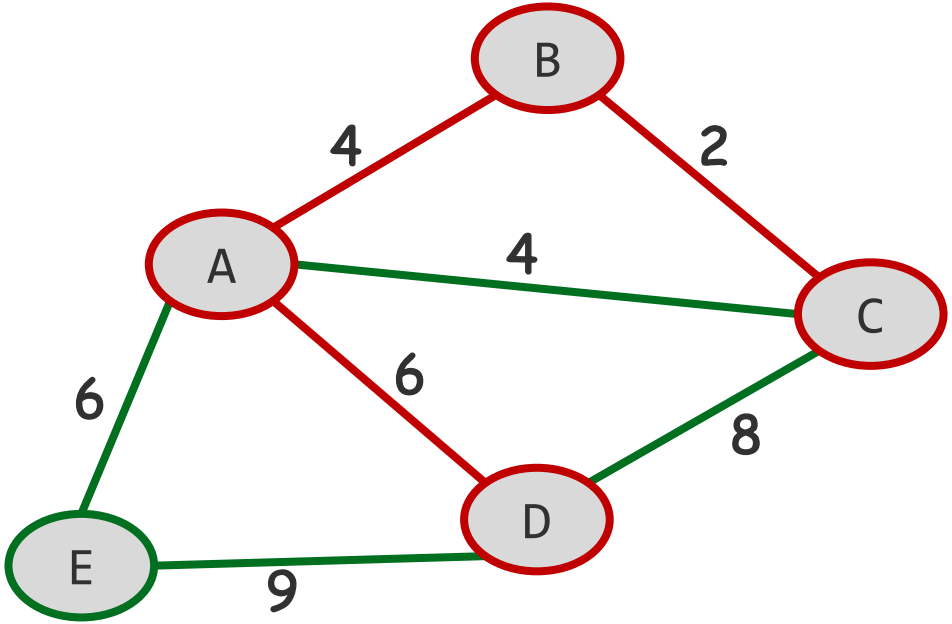
Ziyaret Edilmişler	A	B	C		

Prim En Kısa Yol Ağacı Algoritması



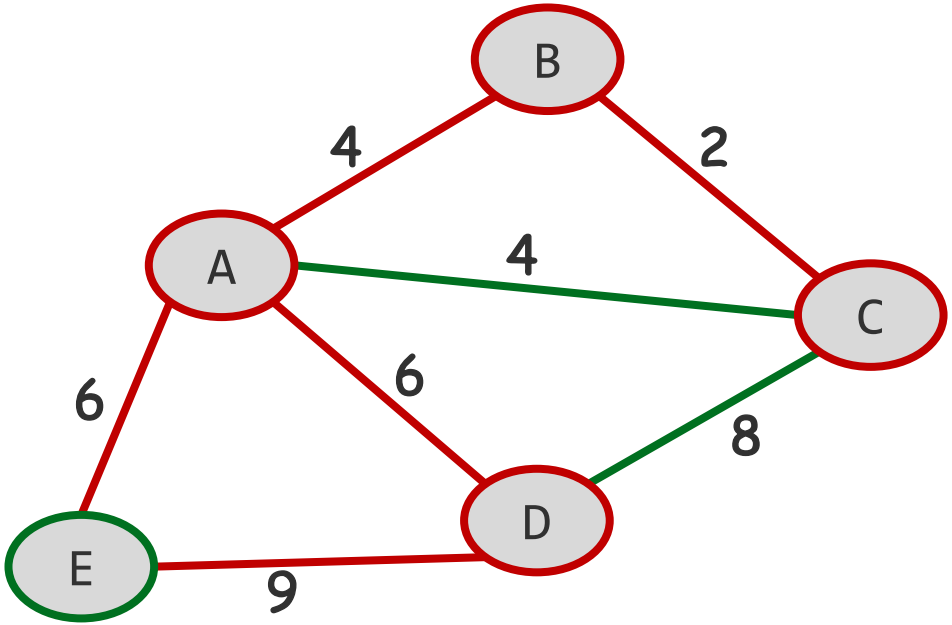
Ziyaret Edilmişler	A	B	C		

Prim En Kısa Yol Ağacı Algoritması



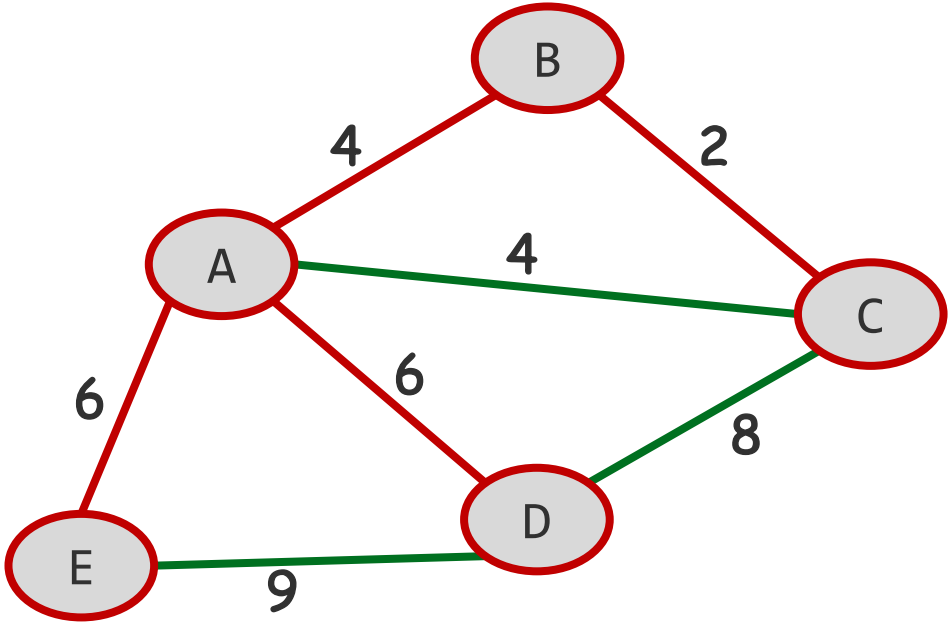
Ziyaret Edilmişler	A	B	C	D	

Prim En Kısa Yol Ağacı Algoritması



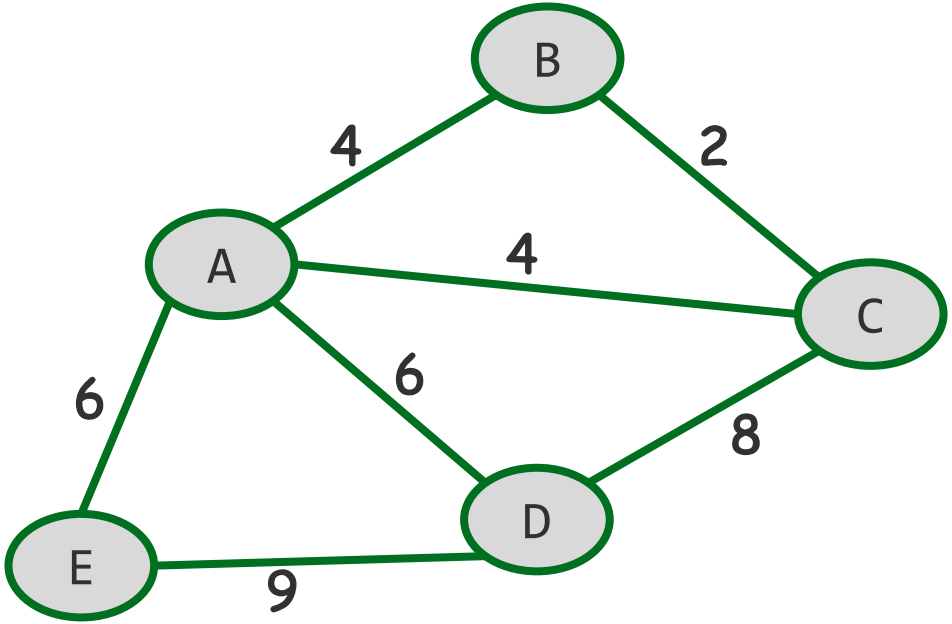
Ziyaret Edilmişler	A	B	C	D	

Prim En Kısa Yol Ağacı Algoritması

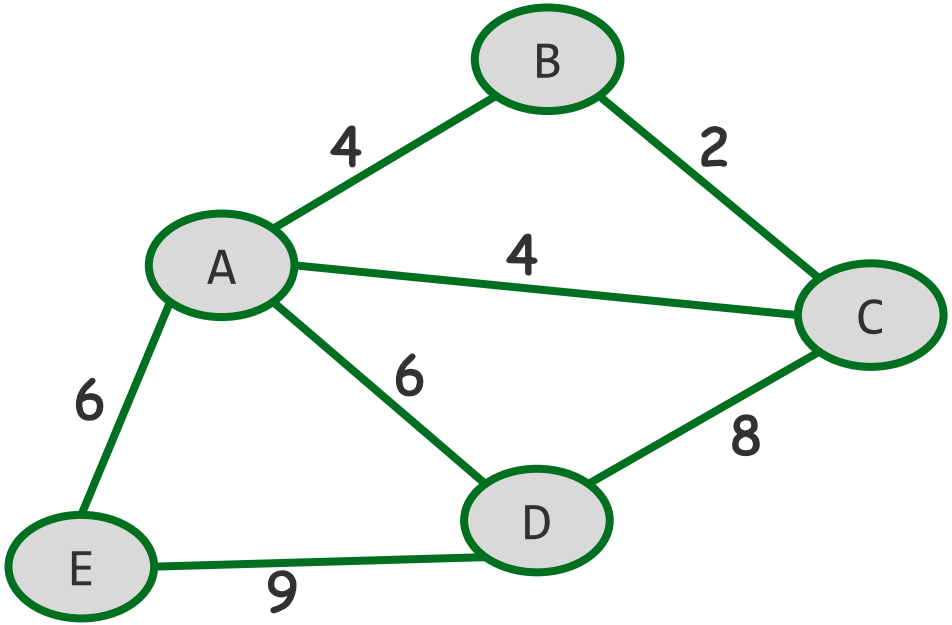


Ziyaret Edilmişler	A	B	C	D	E

Kruskal En Kısa Yol Ağacı Algoritması

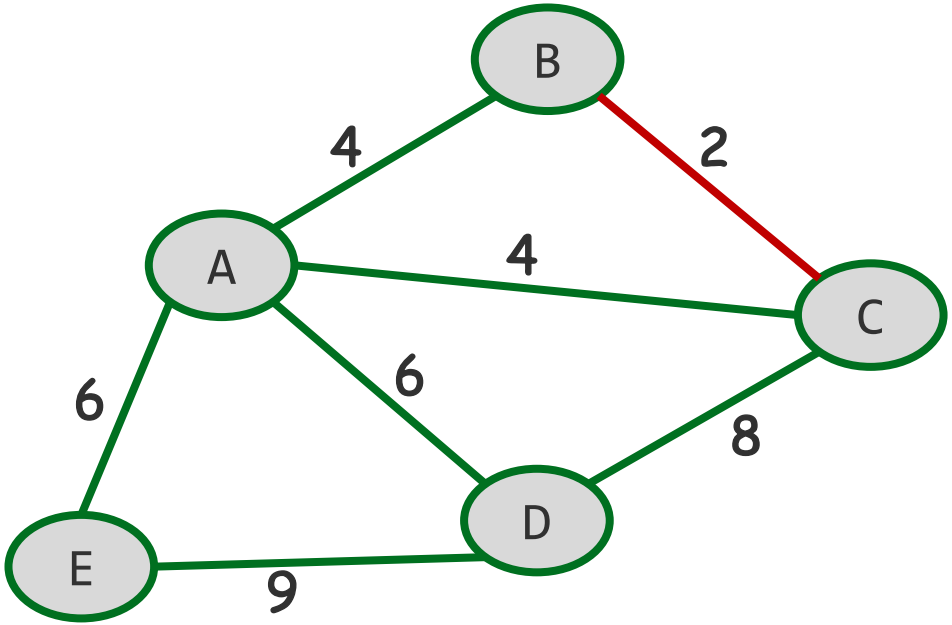


Kruskal En Kısa Yol Ağacı Algoritması



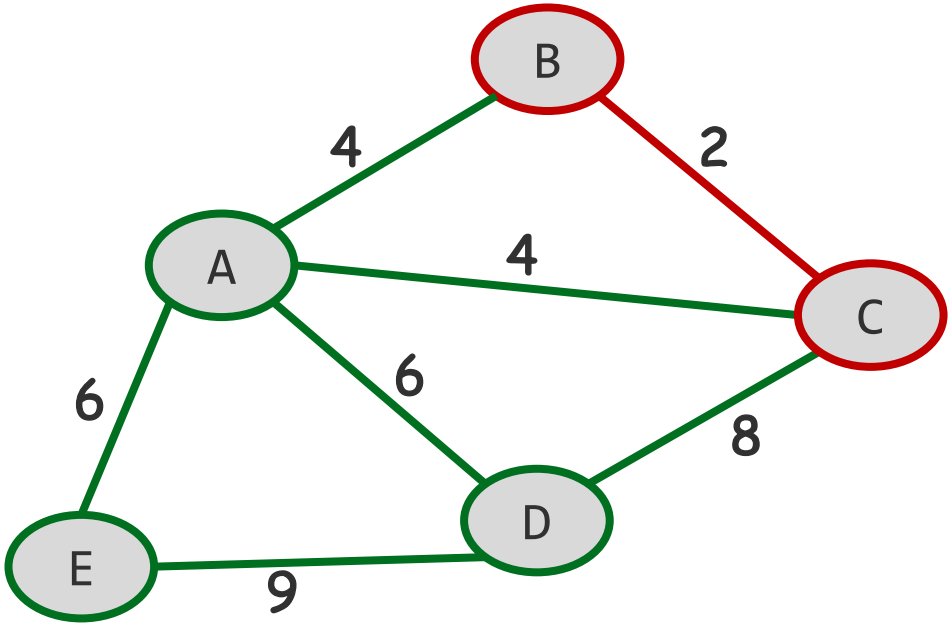
Kenarlar	B-C	A-B	A-C	A-D	A-E	C-D	D-E
Maliyet	2	4	4	6	6	8	9

Kruskal En Kısa Yol Ağacı Algoritması



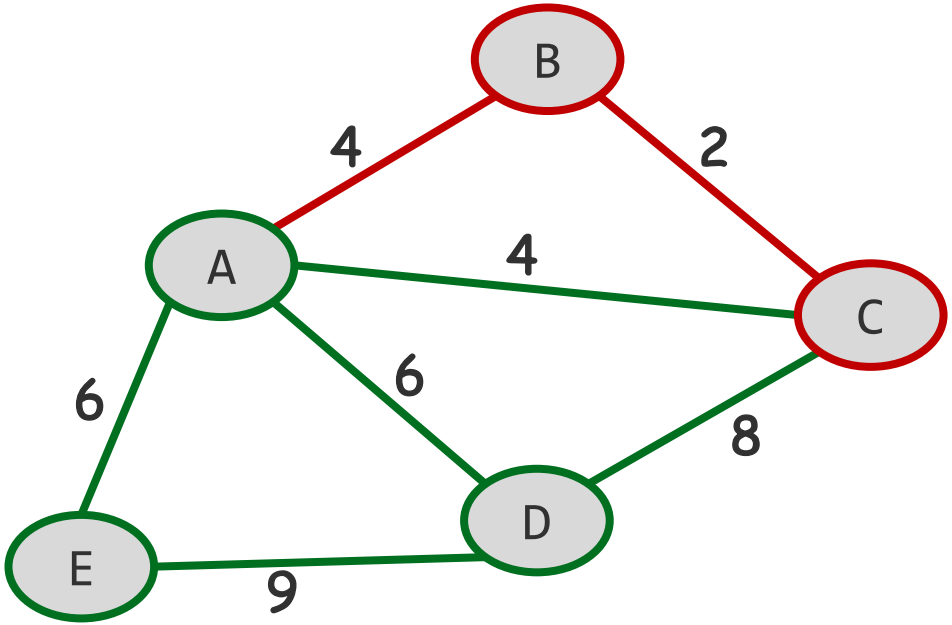
Kenarlar	B-C	A-B	A-C	A-D	A-E	C-D	D-E
Maliyet	2	4	4	6	6	8	9

Kruskal En Kısa Yol Ağacı Algoritması



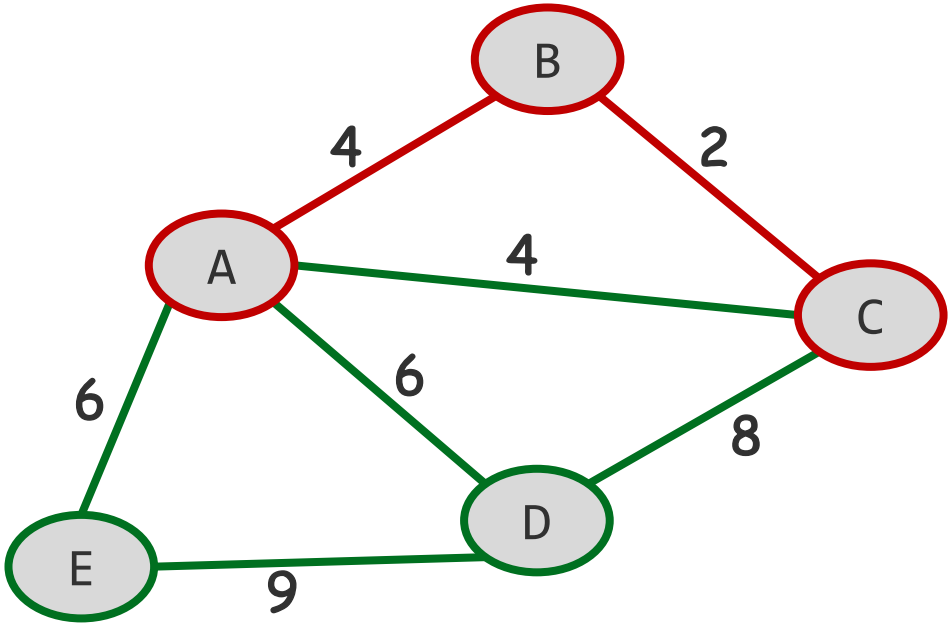
Kenarlar	B-C	A-B	A-C	A-D	A-E	C-D	D-E
Maliyet	2	4	4	6	6	8	9

Kruskal En Kısa Yol Ağacı Algoritması



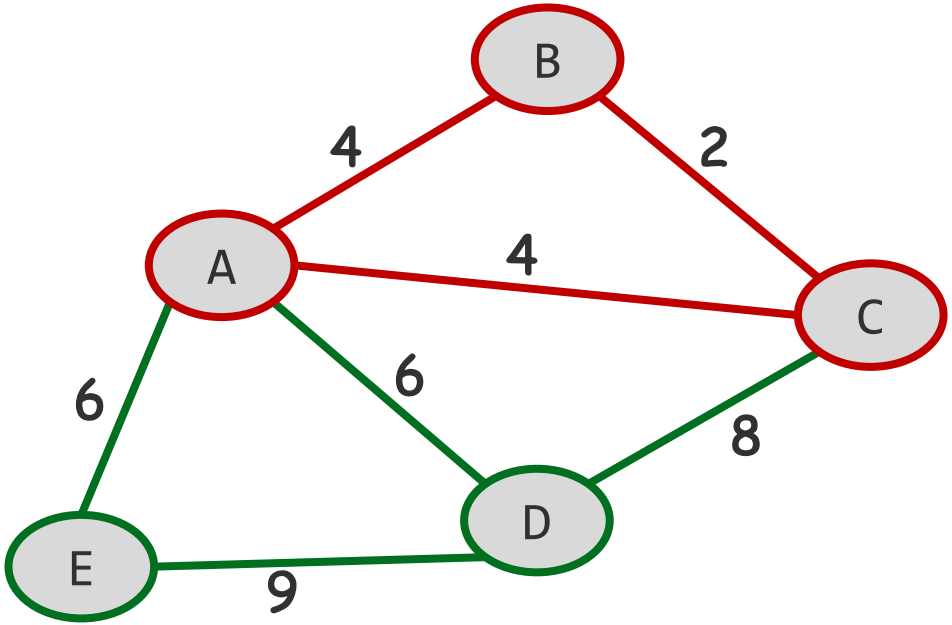
Kenarlar	B-C	A-B	A-C	A-D	A-E	C-D	D-E
Maliyet	2	4	4	6	6	8	9

Kruskal En Kısa Yol Ağacı Algoritması



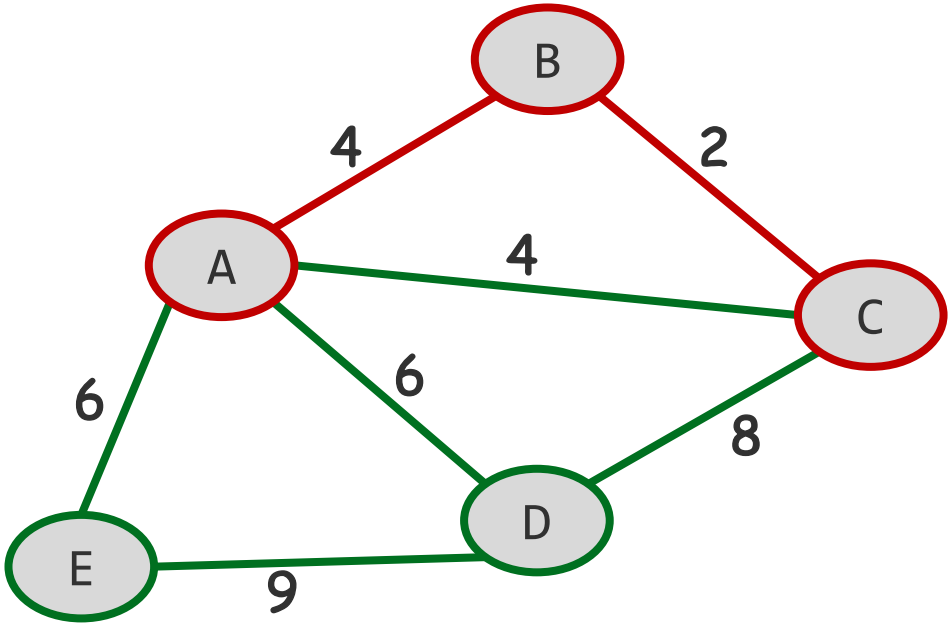
Kenarlar	B-C	A-B	A-C	A-D	A-E	C-D	D-E
Maliyet	2	4	4	6	6	8	9

Kruskal En Kısa Yol Ağacı Algoritması



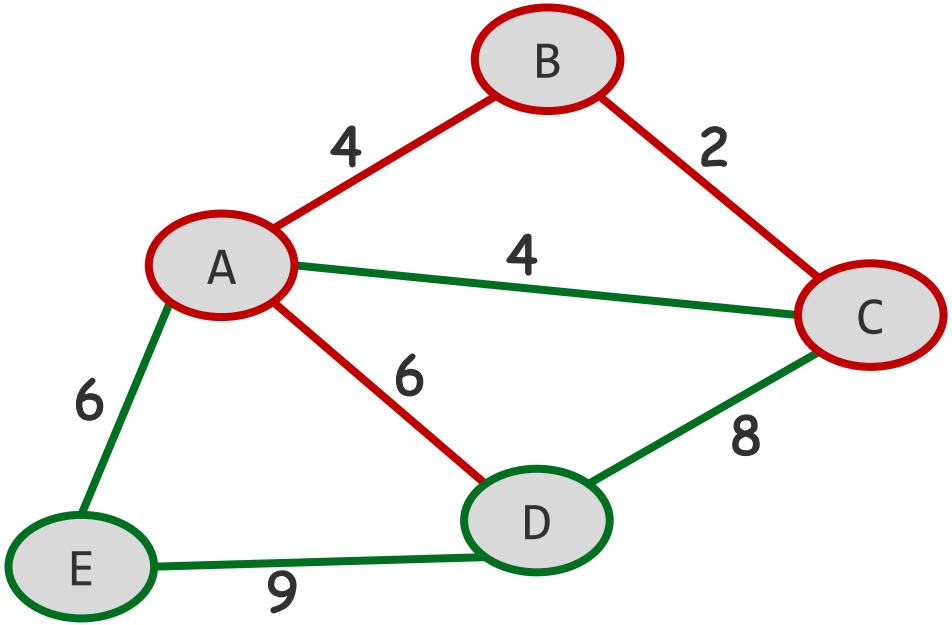
Kenarlar	B-C	A-B	A-C	A-D	A-E	C-D	D-E
Maliyet	2	4	4	6	6	8	9

Kruskal En Kısa Yol Ağacı Algoritması



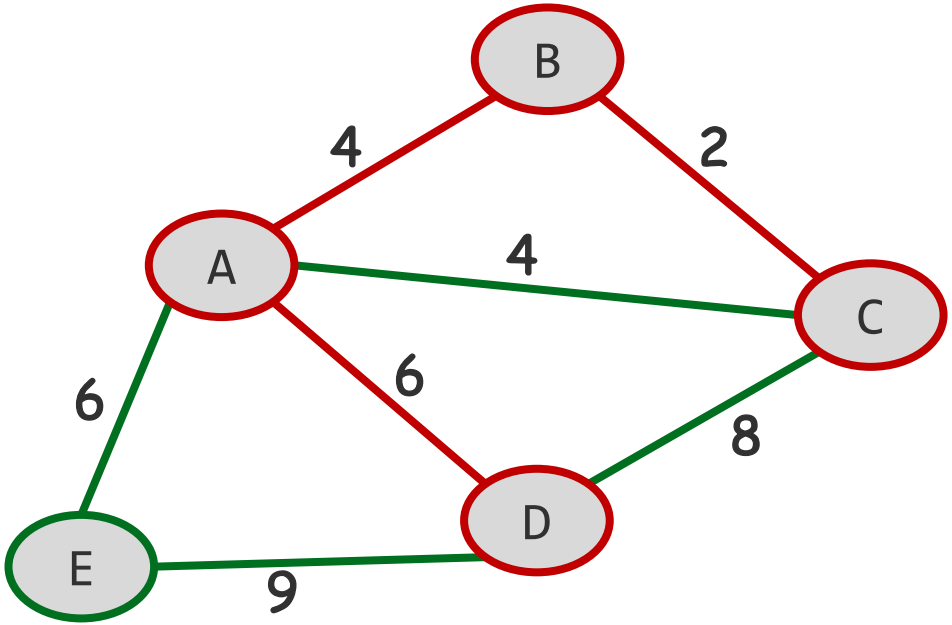
Kenarlar	B-C	A-B	A-C	A-D	A-E	C-D	D-E
Maliyet	2	4	4	6	6	8	9

Kruskal En Kısa Yol Ağacı Algoritması



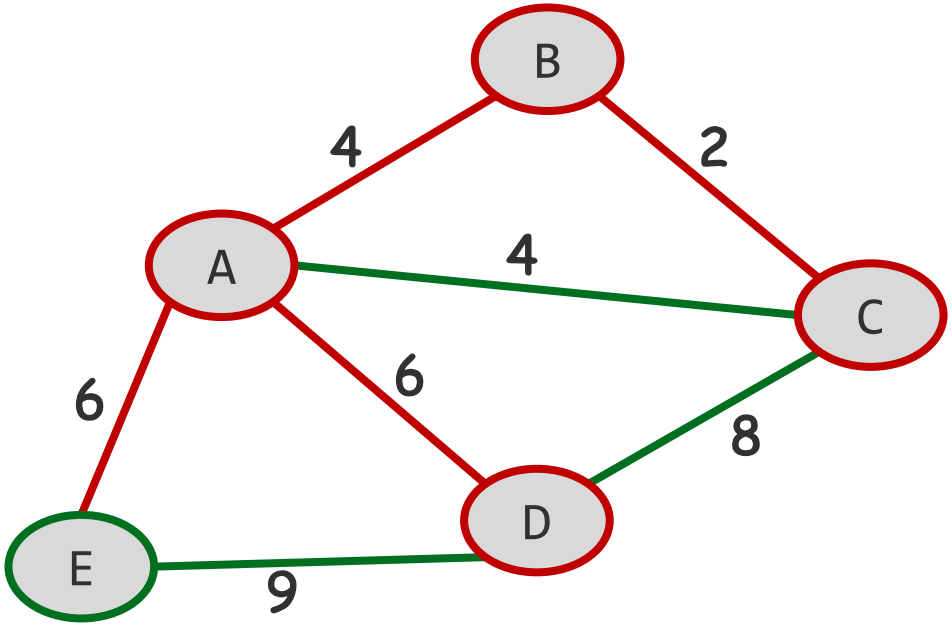
Kenarlar	B-C	A-B	A-C	A-D	A-E	C-D	D-E
Maliyet	2	4	4	6	6	8	9

Kruskal En Kısa Yol Ağacı Algoritması



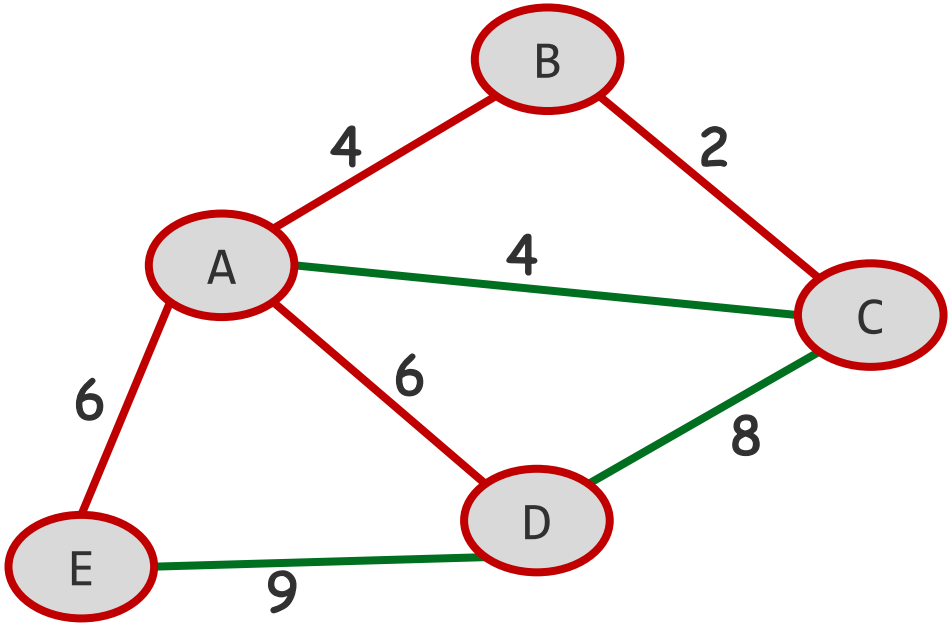
Kenarlar	B-C	A-B	A-C	A-D	A-E	C-D	D-E
Maliyet	2	4	4	6	6	8	9

Kruskal En Kısa Yol Ağacı Algoritması



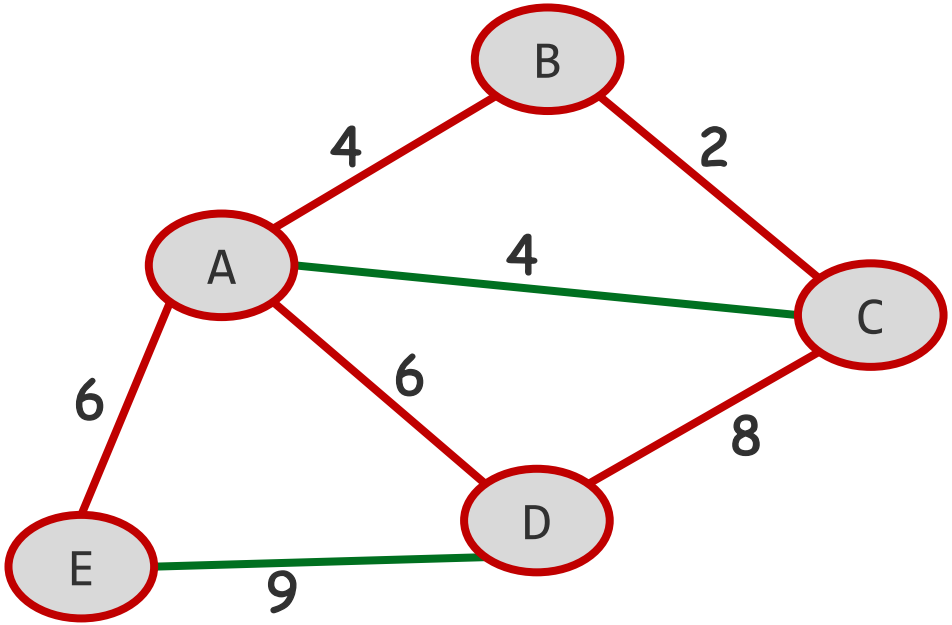
Kenarlar	B-C	A-B	A-C	A-D	A-E	C-D	D-E
Maliyet	2	4	4	6	6	8	9

Kruskal En Kısa Yol Ağacı Algoritması



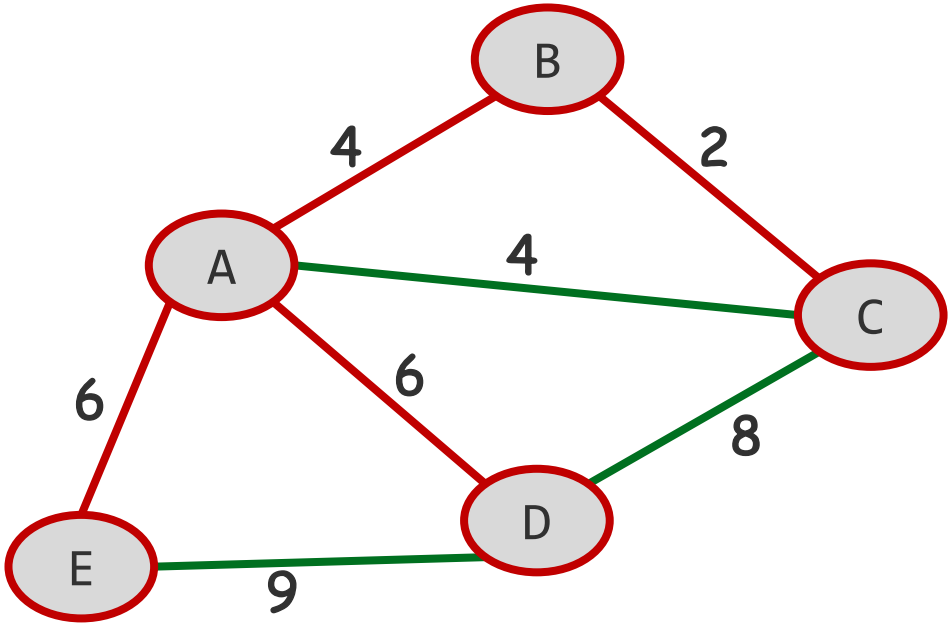
Kenarlar	B-C	A-B	A-C	A-D	A-E	C-D	D-E
Maliyet	2	4	4	6	6	8	9

Kruskal En Kısa Yol Ağacı Algoritması



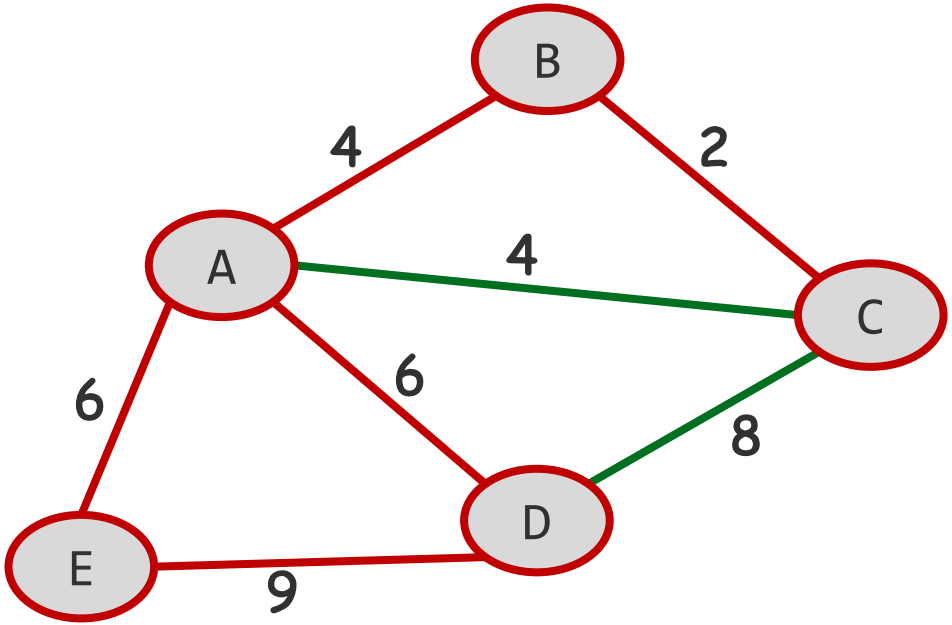
Kenarlar	B-C	A-B	A-C	A-D	A-E	C-D	D-E
Maliyet	2	4	4	6	6	8	9

Kruskal En Kısa Yol Ağacı Algoritması



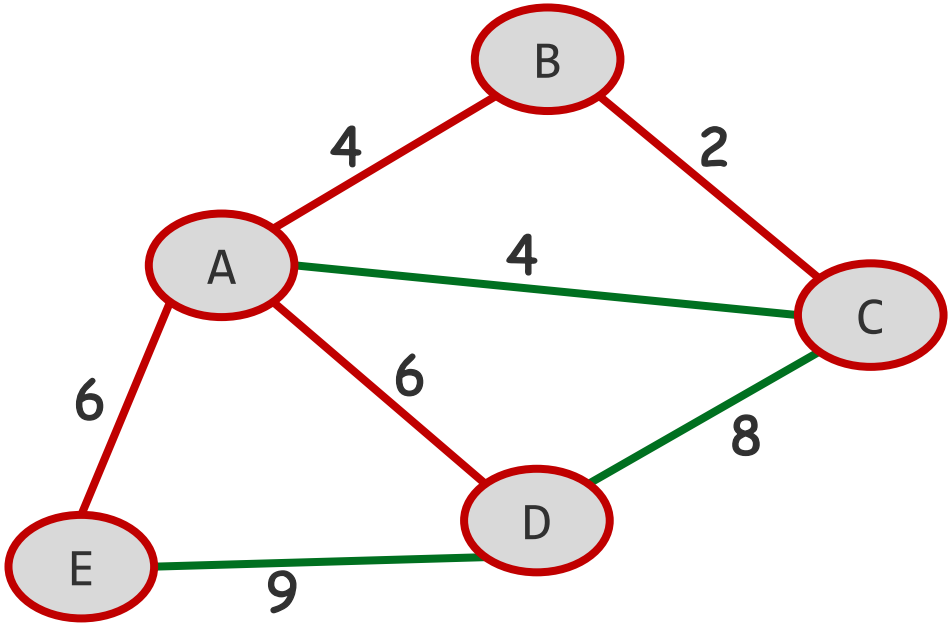
Kenarlar	B-C	A-B	A-C	A-D	A-E	C-D	D-E
Maliyet	2	4	4	6	6	8	9

Kruskal En Kısa Yol Ağacı Algoritması



Kenarlar	B-C	A-B	A-C	A-D	A-E	C-D	D-E
Maliyet	2	4	4	6	6	8	9

Kruskal En Kısa Yol Ağacı Algoritması



Kenarlar	B-C	A-B	A-C	A-D	A-E	C-D	D-E
Maliyet	2	4	4	6	6	8	9