SAKARYA ÜNİVERSİTESİ

Ayrık İşlemsel Yapılar

Hafta 11

Doç.Dr. Nilüfer YURTAY

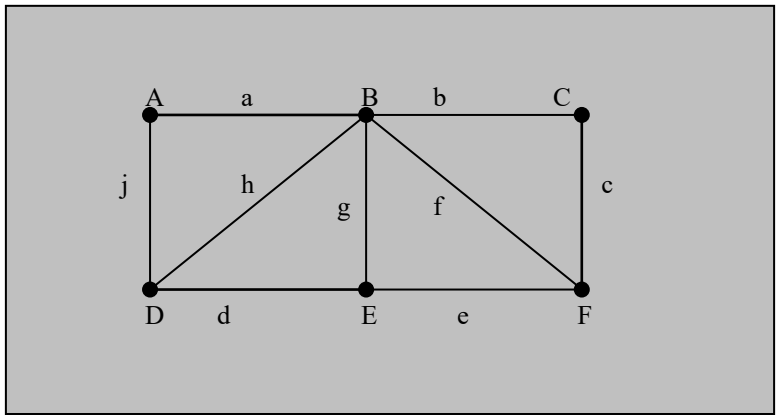
Graflar

11.1 Hamiltonian Döngüsü ve Yolu

Yeni bir G grafının düğümlerinin şehirleri, kenarlarının da bu şehirler arası mevcut yolları gösterdiğini varsayalım. Bir satış temsilcisinin tüm bu şehirleri sadece birer kez ziyaret etmesi gerektiğini düşünelim. Bu durumda zaman ve para açısından en ekonomik yol bu şehirlerden bir kez geçerek turun tamamlandığı yol olacaktır.

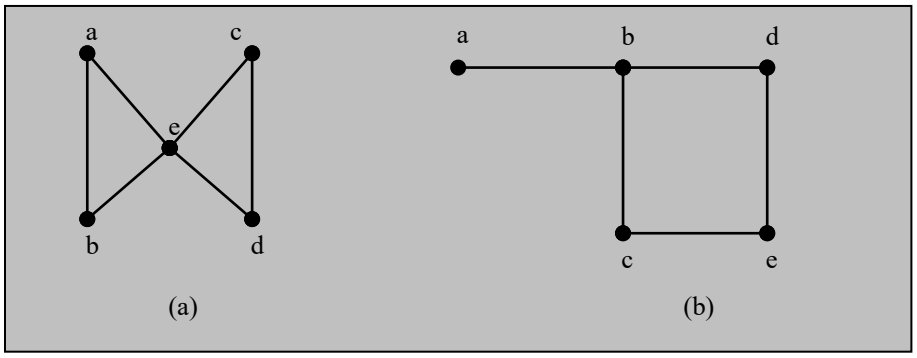
Bir grafta , her bir düğümün sadece bir kez bulunduğu yola hamiltonian yolu , her bir düğümü içeren döngüye de hamiltonian döngüsü adı verilir.

Şekil 11.1' deki grafta a,b,c,e,d,j yolu bir Hamiltonian döngüsüdür.



Şekil 11.1

Şekil 11.2a ve b 'deki graflarda ise Hamiltonian döngüsü yoktur.



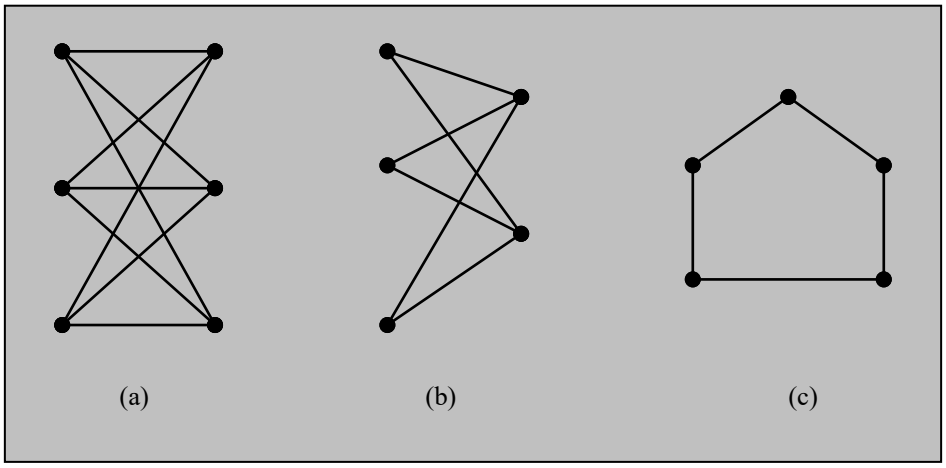
Şekil 11.2

Euler halkası ve yolunun varlığını sınayabilmemiz için nispeten kolay bir kriter mevcuttur. Ayrıca Euler halkasını bulmamız için doğrudan uygulanabilecek bir algoritma da vardır. Ancak aynı durum Hamiltonian döngüsü ve yolu için yoktur. Burada problem , gerek ve yeter koşulların belirlenememesidir.

Genel olarak bir grafın Hamiltonian döngüsünü bulmak zordur. Ancak bazı koşullar bize döngünün varlığını garanti edecektir .

Teorem

n düğümlü (n>3) bir G grafı olsun. Eğer her bir düğümün en az n/2 derecesi var ise G'nin bir Hamiltonian döngüsü vardır.



Şekil 11.3

Şekil 11.3(a) da 6 düğüm vardır ve her bir düğümün derecesi 3 dür. O halde bir Hamiltonian döngüsü vardır. Ancak teorem bunun nasıl bulunacağını bize söylememektedir. Döngüyü bir parça deneme yanılma yoluyla bulabiliriz.

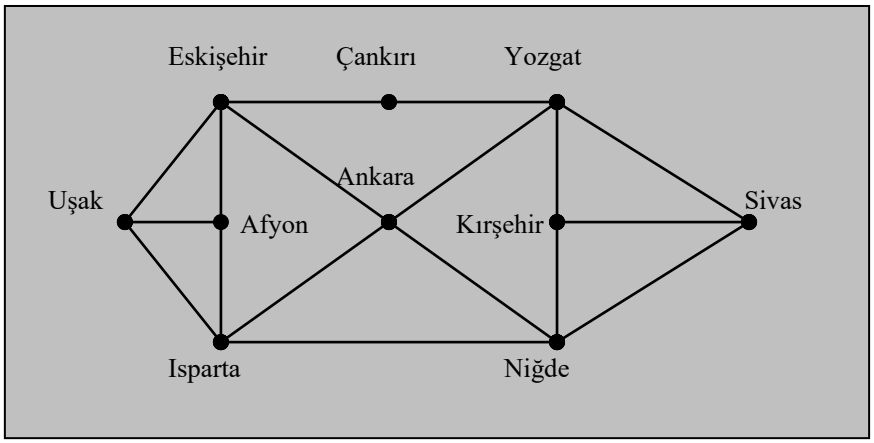
Şekil 11.3(b) de Hamiltonian döngüsü yoktur. Dikkat edilirse bu graf teoremin koşulunu sağlamamaktadır. Derecesi 2<(5/2) olan düğümler vardır.

Ancak şekil 11.3 (c) de de her birinin derecesi 2 olan 5 düğüm vardır ve bir hamiltonian döngüsü de mevcuttur.

Demek ki derecesi n/2 den küçük düğümleri olan graflar için Hamiltonian döngüsünün varlığı yada yokluğu hakkında bir yorum yapılamaz.

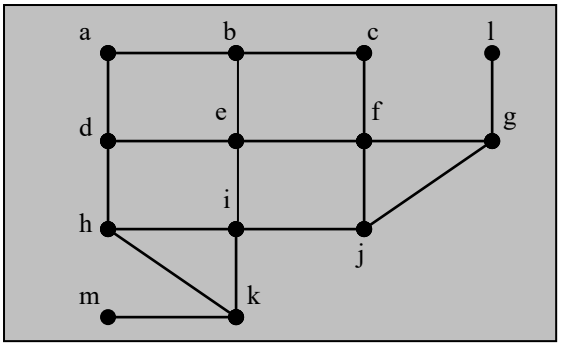
11.2 En Kısa Yol ve Uzaklık

Aşağıdaki graf bir bölgedeki şehirlerarası yol grafiğini gösteriyor olsun. Burada düğümler şehirleri, düğümler arasındaki kenarlar ise bu şehirler arasındaki yolu ifade etsin. Bir şehirden diğer bir şehre ulaşmak için muhtemel çok yol bulunduğu halde en ekonomik biçimde hedefe ulaşmak istenebilir. Bu da bir en kısa yol problemi olarak düşünülebilir.

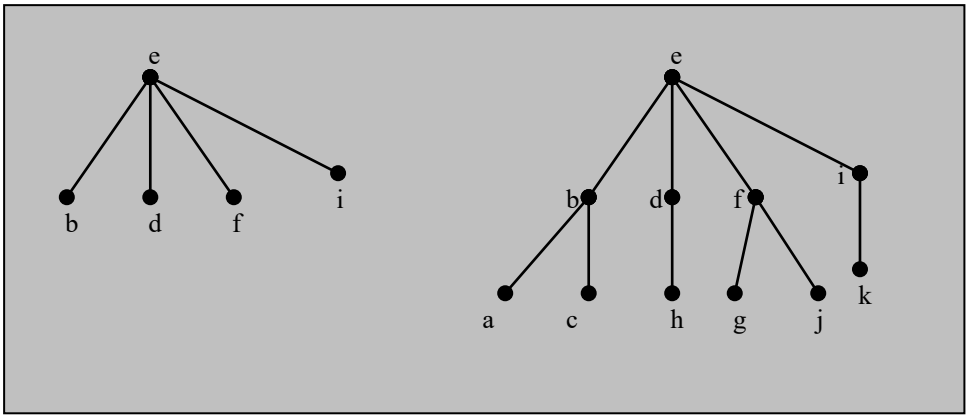


Bu bölümde düğümler arasındaki en kısa yolun bulunması probleminin çeşitli çözümlerini ele alacağız. Bir grafta herhangi bir S ve T düğümleri arasında , en az kenar içeren yola en kısa yol denir. Bu S-T yolu üzerindeki toplam kenar sayısı S 'den T 'ye uzaklık (distance) olarak adlandırılır. En kısa yolu bulmak için yapılacak işlem S düğümünün komşularını bulmak , bu komşulara , komşu olan diğer düğümleri bulmak biçiminde olacaktır. Hangi düğümlerin incelendiğinin kaydından geriye inceleyerek da S'den T'ye en kısa yol bulunacaktır. Bunun için önerilen algoritma Breadth-First Search algoritmasıdır.

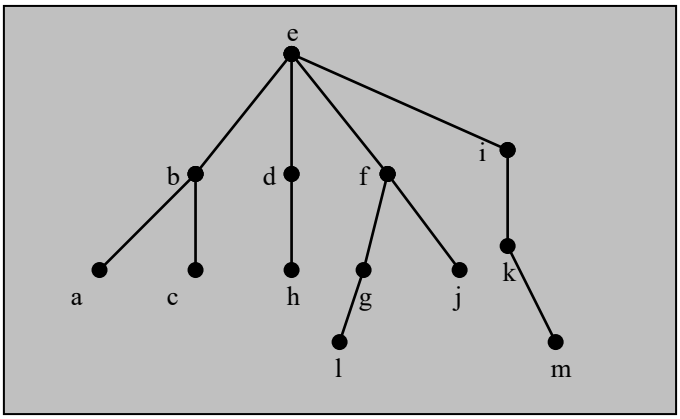
Örnek olarak aşağıdaki şekilde verilen graf için bunları izlemeye çalışalım.



E düğümünü seçelim ve bu e düğümüne bütün diğer düğümlerin mesafelerini belirlemeye çalışalım. e'ye komşu b,d,f ve i düğümleri bulunmaktadır. Bu düğümler ağacın birinci seviyesini oluşturur.



a,c,h,g,j ve k **düğümleri** yeni düğümlerdir ve ağacın ikinci seviyesini oluştururlar. Bunların da komşuları incelenirse;



Böylece bütün komşuluklar elde edilmiş olur. Burada e düğümü ile diğer düğümler arasındaki uzaklıklar bulunmuştur.

11.2.1.BFS Algoritması

Bu algoritma bir S düğümünden T düğümüne en kısa yolu ve uzaklığı bulur. Algoritmadaki L etiketlenmiş düğümler kümesidir. A düğümünün önceli, L kümesinde A 'yı etiketlemek için kullanılan düğümdür. Algoritma aşağıdaki adımlardan oluşmaktadır.

Adım 1 (S ' i etiketle)

a) ***S' e 0 etiketini ver , S'in önceli yok olsun .***

b) ***L={S} ve k=0***

Adım 2 (Düğümleri etiketle)

repeat

Adım 2.1 (etiketi artır)

k=k+1

Adım 2.2 (etiketlemeyi genişlet)

While (L'de k-1 etiketli, L'de olmayan bir W düğümüne komşu olan bir V düğümü var)

a)W düğümüne K etiketini ata

b)V düğümünü W'nin önceli olarak ata

c)W düğümünü L kümesine ekle

endWhile

until (T, L kümesinin içinde yada L'deki düğümlere ,L'de olmayan düğümlerden hiç biri komşu değil)

Adım 3(T'ye en kısa yolu oluştur.)

if(T,L kümesinde)

S düğümüne erişene kadar T düğümünün önceli, öncelinin önceli v.b. üzerinden en kısa yolu oluştur. T 'nin etiketi S 'den T 'ye uzaklığı gösterecektir.

Otherwise

S 'den T 'ye yol yoktur.

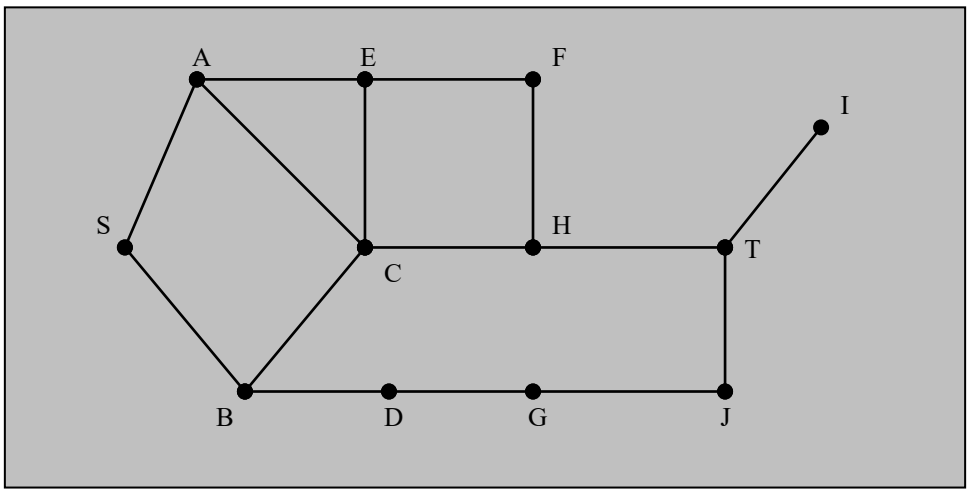
Endif

Bir düğümü etiketlemek ve bir kenarı, komşu kenarı bulmakta kullanmayı, algoritmanın analizinde elemanter işlem olarak düşündüğümüzde, n düğümlü ve e kenarlı bir grafta , her düğüm bir kez etiketlenmekte ve her kenar komşu düğümü bulmak için en çok bir kez kullanılmaktadır. Buna göre en çok n+e işlem yapılacaktır.

n+e<= n+C(n,2)=n+1/2(n.(n-1)) olduğuna göre algoritma en çok n^2 mertebesindedir. **Hata! Yer işareti tanımlanmamış.**

Örnek 11.1

Şekil 11.4'deki çoklu grafta S düğümünden T düğümüne en kısa yolu bulmak için BFS algoritmasını adım adım uygulayınız.



Şekil 11.4

Çözüm:

Adım 1 S etiketi (0), L = {S} k = 0

Adım 2

Repeat (1)

2.1 k =1

2.2 while

- L de 0 etiketli S düğümüne,L de olmayan A komşu

(a) A etiketi 1 (b) A nın önceli S

(c) L={S,A}

- L de 0 etiketli S düğümüne, L de olmayan B komşu

(a)B etiketi 1 (b) B nin önceli S (c)L={S,A,B}

endwhile (0 etiketli S'de L'de olmayan komşu düğüm yok)

Until (T, L de değil ve L deki düğümlere komşu var)

(2)

2.1 k = 2

2.2 while

- L de 1 etiketli A düğümüne L'de olmayan C komşu

(a) C etiketi 2 (b) C nin önceli A

(c) L={S,A,B,C}

- L de 1 etiketli A ya L'de olmayan E komşu

a)E etiketi 2 (b) E nin önceli A

(c) L={S,A,B,C,E}

-L de 1 etiketli B ye L'de olmayan D komşu

(a)D etiketi 2 (b) D nin önceli B

c) L={S,A,B,C,E,D}

endwhile(1 etiketli düğümlere L'de olmayan komşu düğüm yok)

Until (T, L de değil ve L deki düğümlere komşu var)

(3)

2.1 k = 3

2.2 while

-L de 2 etiketli C ye,H komşu,

H→ 3(C),L={S,A,B,C,E,D,H}

-L de 2 etiketli E ye,F komşu,

F→ 3(E),L={S,A,B,C,E,D,H,F}

-L de 2 etiketli D ye,G komşu, G→3(D),L={S,A,B,C,E,D,H,F,G}

endwhile

until (T, L de yok , L nin dışında komşu düğümler var)

(4)

2.1 k = 4

2.2 while

- L de 3 etiketli H ya komşu, T var =>

T →4(H),L={S,A,B,C,E,D,H,F,G,T}

- L de 3 etiketli F komşu , L de olmayan düğüm yok

- L de 3 etiketli G ye komşu, J var =>

J→ 4(G),L={S,A,B,C,E,D,H,F,G,T,J}

endwhile

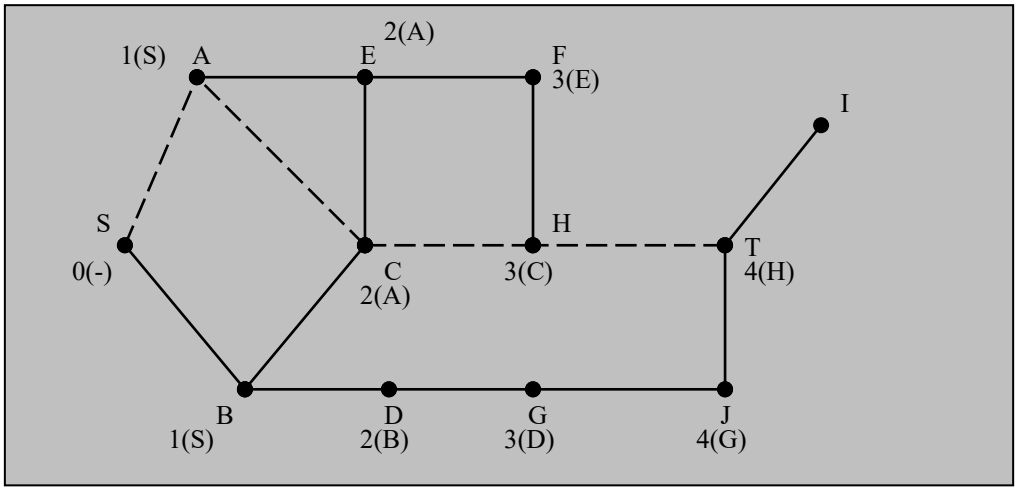
Until (T, L de!)

Adım 3 (T ye en kısa yolu oluştur)

if (T, L kümesinde)

T nin önceli H, H nın önceli C, C nin önceli A, Anın önceli S; En kısa yol S,A,C,H,T olup S ile T arasındaki mesafe 4 kenardır.

Endif

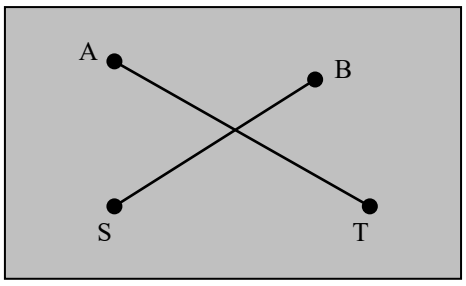


Şekil 11.5

Bu örnekten görüldüğü gibi C nin önceli B Düğümü de olabilirdi. Bu durumda en kısa yol S,B,C,H,T olacaktı.

Örnek 11.2

Şekil 11.6'deki graf için BFS algoritmasını uygulayınız.



Şekil 11.6

Çözüm:

repeat L={S}

2.1 k = 1

2.2while

- L de S e komşu B; B→1(S)

endwhile

Until (T, L de değil, L deki düğümlere komşu düğüm yok)

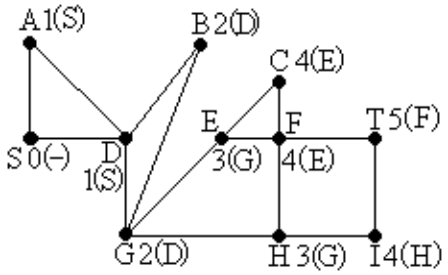
Adım 3

T, L de değil

S den T ye yol yok!

Örnek 11.3

Şekil 11.7a'daki grafta S-T düğümleri arasındaki en kısa yolu bulunuz.



Şekil 11.7a

Çözüm:

Adım 1: S →0 k=0 L={S}

Adım 2: repeat

Adım 2.1: k=1

Adım 2.2: while

- L' de S' e komşu A var L={S,A}
- L' de S' e komşu D var L={S,A,D}

k=2

- L' de A' ya komşu L' de olmayan düğüm yok
- L' de D' ye komşu B,G var L={S,A,D,B,G}

k=3

- L' de B' ye komşu düğüm yok
- L' de G' ye komşu E, H var L={S,A,D,B,G,E,H}

k=4

- L' de E' ye komşu C, F var
- L' de H' ye komşu I var L={S,A,D,E,H,C,F,I}

k=5

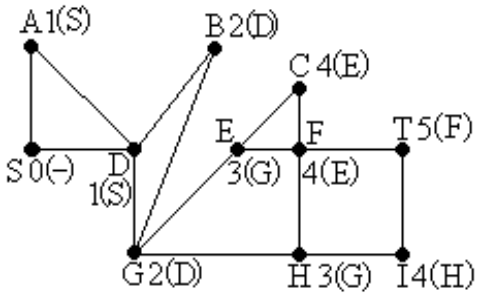
- L’ de C’ ye komşu yok

- L’ de F’ ye komşu T var L={S,A,D,E,H,C,F,I,T}

- L’ de I’ ya komşu yok

T, L’ de bitti.

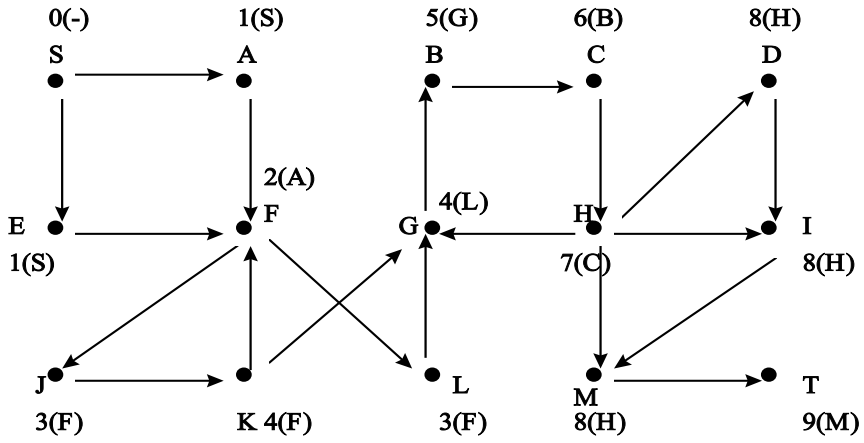
S-T uzaklığı 5, yol S,D,G,E,F,T’dir.



Şekil 11.7b

Örnek 11.4

Şekil 11.8’deki yönlü grafta S’den T’e en kısa yolu ve uzaklığı bulunuz. BFS algoritmasını adapte edeceğiz.



Şekil 11.8

Adım1

S’i etiketle S→0(-)

L={S}, k=0

Adım 2.

Düğümleri etiketle

repeat

2.1 k=k+1=1

2.2 etiketlemeyi genişlet

while (L'de k-1 etiketli, L'de olmayan bir W düğümüne doğru kenarı olan bir V düğümü var)

S düğümünden A ve E düğümlerine yol var.

A→1(S) L={S,A}

E→1(S) L={S,A,E}

until (T,L'de değil ve L'deki düğümlerden L'de olmayan düğümlere yol var.)

k=2

while A düğümünden F'e yol var.

F→2(A) L={S,A,E,F}

while E'den yol yok.

2.2

k=3 (k=2, etiketli F var.)

while F'den J,L ye yol var

J→3(F) L={S,A,E,F,J}

L→3(F) L={S,A,E,F,J,L}

k=4 (k=3 etiketli J,L var)

while J'den K'a yol var

K→4(J) L={S,A,E,F,J,L,K}

while L'den G'e yol var

G→4(L) L={S,A,E,F,J,L,K,G}

k=5, (k=4 etiketli K,G var)

while K'dan yol yok

while G'den B'e yol var.

B→5(G) L={S,A,E,F,J,L,K,G,B}

k=6, (k=5 etiketli B var)

C→6(B) L={S,A,E,F,J,L,K,G,B,C}

k=7

H→7(C) L={S,A,E,F,J,L,K,G,B,C,H}

k=8 H'dan D,I,M'e yol var

D→8(H) L={S,A,E,F,J,L,K,G,B,C,H,D}

I→ (8H) L={S,A,E,F,J,L,K,G,B,C,H,D,I}

M→8(H) L={S,A,E,F,J,L,K,G,B,C,H,D,I,M}

k=9

D'den yol yok

I'dan yol yok

M'den T'e yol var

T→9(M) L={S,A,E,F,J,L,K,G,B,C,H,D,I,M,T}

until (T kümede)

Bitti.

Adım 3. T L kümesinde, S'den T'e uzaklık 9, geriye izlenirse

T→M→H→C→B>→G→L→F→A→S

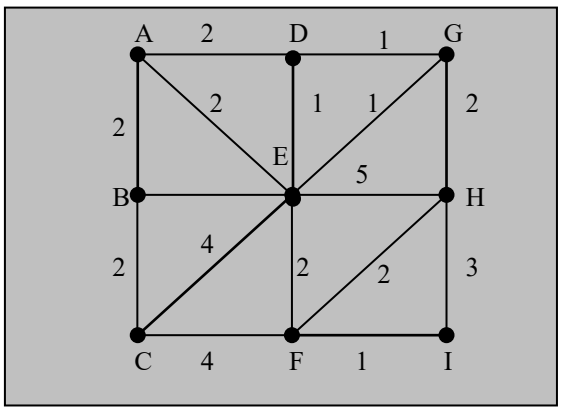
S,A,F,L,G,B,C,H,M,T en kısa yoldur.

11.2.2 Ağırlıklı Graflar (Weighted Graphs)

Çoğunlukla graflar objeler arasındaki ilişkileri, bağlantıları tanımlamak için kullanıldığında her bir kenara bir sayı atanır. Örneğin, bir graf şehirler arası yolları gösteren bir graf ise, kenarların üzerine mesafeler yazılır. Bir ağırlıklı graf (Weighted graph) her bir kenarına ağırlık (weight) denilen bir sayının atıldığı bir graftır. Bir yolun ağırlığı ise o yol üzerindeki kenarların ağırlıklarının toplamıdır.

Örnek 11.5

Şekil 11.9'da bir ağırlıklı graf örneği verilmiştir. AEHI yolunun ağırlığı 2+5+3=10 dur. CFEDG yolu ise 4+2+1+1=8 ağırlığındadır.



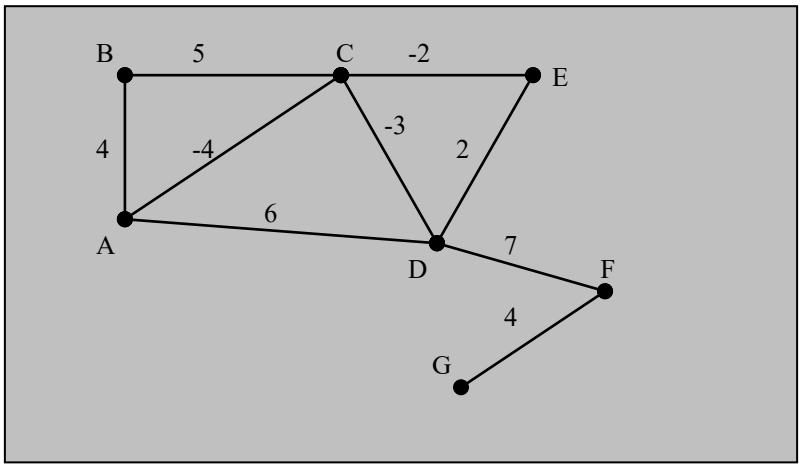
Şekil 11.9

Bir çok uygulamada en küçük ağırlıklı yolu bulmak isteriz. Gerçi bunun her zaman olması gerekmez. Böyle bir durum negatif ağırlıklı bir döngü varsa karşımıza çıkar.

Örnek 11.6

Şekil 11.10'daki grafı ele alalım. ABCDE yolunun ağırlığı 4+5-3+2=8 dir. ABCEDAC yolu ise 4+5-2+2+6-4=11 ağırlığındadır. Görüldüğü gibi ABCEDAC döngüsü tekrarlandıkça ağırlık giderek azalacaktır. Sonuç olarak ADC ve DCE düğümleri arasında en küçük ağırlıklı yol yoktur.

Bu durumda, açık olarak tersi belirtilmedikçe ağırlıklı grafın, negatif ağırlıklı bir döngüsünün olmadığını varsayıyoruz. Bu varsayım eğer iki düğüm arasında yol varsa, en küçük ağırlıklı olanın her zaman olabileceğini garanti edecektir. Daha da ötesi olarak bu varsayım sonucu, en küçük ağırlıklı yolun bir basit yol olacağını da söyleyebiliriz. Zira sıfır ağırlıklı bir döngünün çıkarılabileceği açıktır.



Şekil 11.10

En küçük ağırlıklı yola en kısa yol ve yolun ağırlığına da uzaklık denir. Graftaki tüm ağırlıklar pozitif ise S ve T düğümleri arasındaki en kısa yolu bulan bir algoritmayı yani Dijkstra Algoritması inceleyelim.

11.2.3. Dijkstra Algoritması

G, birden fazla düğümü olan, tüm ağırlıkları pozitif olan, bir ağırlıklı graf olsun. Algoritma S düğümünden, G deki diğer bir düğüme en kısa yolu ve uzaklığı bulur. Algoritmada P sabit etiketli düğümler kümesini gösterir. A düğümünün önceli P de A yı etiketlemek için kullanılan düğümdür. W(U,V), U ve V düğümleri arasındaki kenarın ağırlığıdır U,V arasında kenar yoksa W(U,V) = ∞ yazılır.

Algoritma aşağıdaki adımlardan oluşur.

Adım 1 (S düğümünü etiketle)

(a)S' e 0 etiketini ver S in önceli yok olsun

(b)P ={S}

Adım 2 (Düğümleri etiketle)

P de olmayan (geçici olabilir) her bir V düğümüne W(S,V) etiketini ata ve V nin önceli de S olsun

Adım 3 (P'yi genişlet gözden geçir.)

Repeat

Adım 3.1: (Başka etiketi sabit yap.)

P' de olmayan en küçük etiketli U düğümünü P' ye ekle. (Böyle birden fazla düğüm varsa keyfi olarak birini seç.)

Adım 3.2: (Geçici etiketleri revize et.)

P' de olmayan U' ya komşu olan her X düğümü için, X' in eski etiketi ile U' nun etiketi ve W(U,X)' in toplamından hangisi küçük ise X' in etiketini onunla değiştir. Eğer X' in etiketi değişmiş ise U' nun düğümünü X' in önceli yap.

until (P' de G' deki tüm düğümler var.)

Adım 4 (Mesafeleri ve en kısa yolu bul).

Bir Y düğümü üzerindeki etiket S düğümüne uzaklığı göstermektedir. Eğer Y' nin etiketi ∞ ise yol yok demektir. Aksi halde S' den Y' ye yol Y' nin önceli,

öncelinin önceli vb. , Y' den S' e doğru ters sırada izlenerek elde edilir.

Dijkstra Algoritmasının karmaşıklığı;

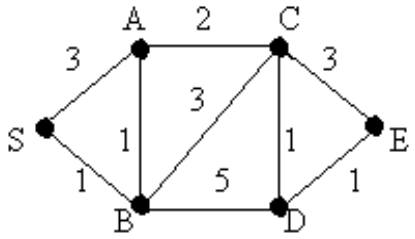
Algoritma analiz edildiğinde, n düğüm var ise bir düğümle ilgili atama işlemlerini bir işlem diye düşünürsek adım 1' de bir, adım 2' de n-1 işlem vardır. Adım 3, n-1 kere yapılmaktadır, her birinde en çok n-2 karşılaştırma yapılmakta, etiketleri gözden geçirmede n-1 düğüm sınanmakta ve her birinde toplama, karşılaştırma ve 2 atama ile birlikte 4 işlem yapılmaktadır. Böylece en çok (n-1)[n-2+1+4(n-1)]=(n-1)(5n-5) işlem olmaktadır. Adım 4' de en çok n-1 öncel izlenmekte olup n işlem yapılmaktadır. Buna göre toplam

$1+n-1+(n-1)(5n-5)+n=5n^2-8n+5$ işlem olup, karmaşıklığı $n^{2'}$ dir.

Örnek 11.7

Şekil 11.11a'daki grafa Dijkstra algoritmasını uygulayarak S düğümünden E düğümüne en kısa yolu bulunuz.

Çözüm:



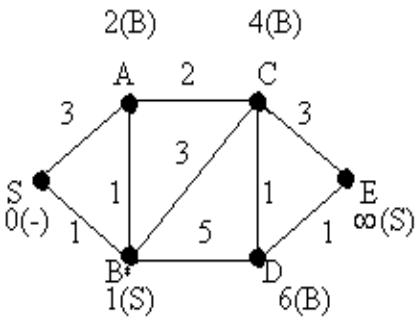
Şekil 11.11a

Adım 1: S' in etiketi 0 ve P={S} (Graf üzerinde S^*, S'in P' de olduğunu göstermekte altında yazılı rakam etiketi, parantez içi önceli göstermektedir).

Adım 2: P' de olmayan her V düğüme W(S,V) etiket

$A \rightarrow 3(S)$, $B \rightarrow 1(S)$, $C \rightarrow \infty(S)$,

$D \rightarrow \infty(S)$, $E \rightarrow \infty(S)$



Şekil 11.11b

Adım 3: (1)

repeat

Adım 3.1: P' de olmayan en küçük etiketli düğüm B, P={S,B}

Adım 3.2: B' ye komşu A,C,D düğümleri var

Düğüm X eski etiket B etiketi+W(B,X) Minimum

A	3	1+1=2	$A \rightarrow 2(B)$
C	∞	1+3=4	$C \rightarrow 4(B)$
D	∞	1+5=6	$D \rightarrow 6(B)$

until P' de tüm düğümler yok

Adım 3: (2)

Adım 3.1: P' de olmayan en küçük etiketli düğüm A P={S,B,A}

Adım 3.2: A' ya komşu P' de olmayan C düğümü var.

Düğüm X	eski etiket	A etiketi+W(A,X)	Minimum
C	4	2+2=4	değişiklik yok

until P' de tüm düğümler yok

Adım 3: (3)

Adım 3.1: P' de olmayan en küçük etiketli düğüm C,

P={S,B,A,C}

Adım 3.2: C' ye komşu D,E var

Düğüm X	eski etiket	C etiketi+W(C,X)	Minimum
D	6	4+1=5	D $\rightarrow$ 5(C)
E	∞	4+3=7	E $\rightarrow$ 7(C)

until P' de tüm düğümler yok

Adım 3: (4)

Adım 3.1: P' de olmayan en küçük etiketli düğüm D,

P={S,A,B,C,D}

Adım 3.2: D' ye komşu E var

Düğüm X	eski etiket	B etiketi+W(B,X)	Minimum
E	7	5+1=6	E $\rightarrow$ 6(D)

until P' de tüm düğümler yok

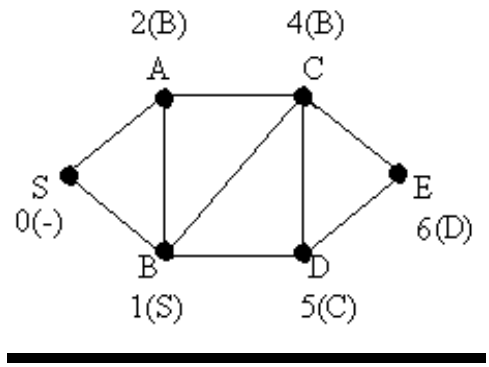
Adım 3: (5)

Adım 3.1: P' de olmayan E var P={S,A,B,C,D,E}

Adım 3.2: E' ye komşu düğüm yok

until (Tüm düğümler P' de)

P en kısa yoldur ve şekil 3.29b'deki gibidir.



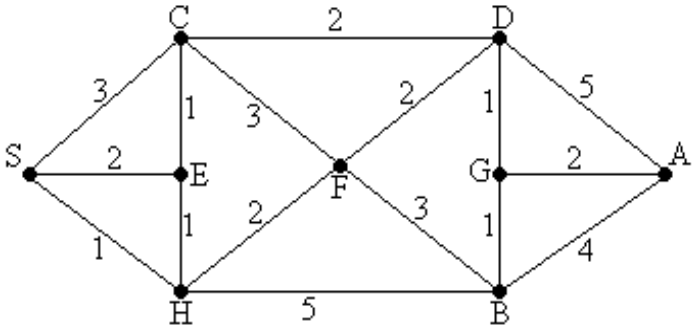
Örneğin S' in E düğümüne en kısa yolu 6 ağırlıklı olup geriye izlenirse S,B,C,D,E olduğu bulunur.

Şekil 11.12

Görüldüğü gibi algoritma tüm düğümlerin S düğümüne uzaklıklarını hesaplamaktadır.

Örnek 11.8

Şekil 11.13'deki grafta S' den A' ya en kısa yolu bulunuz.



Şekil 11.13

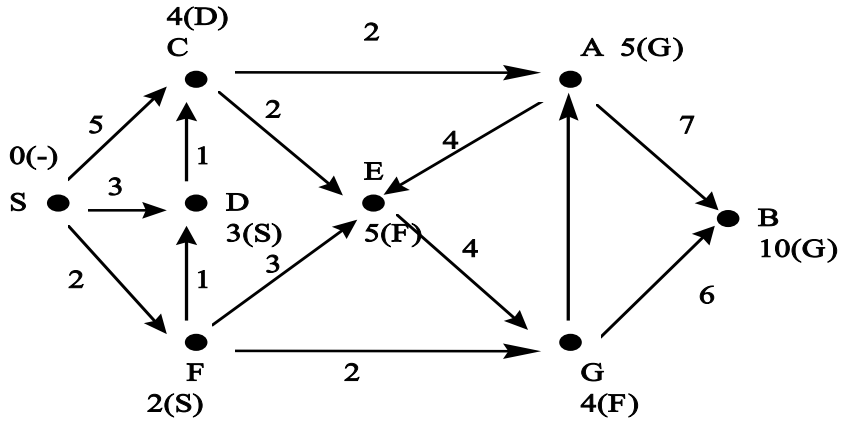
Çözüm:

S $\rightarrow$ 0(-); P={S}

S	C	E	H	F	D	G	B	A	P
0(-)	3(S)	2(S)	1(S)	∞(S)	∞(S)	∞(S)	∞(S)	∞(S)	S
				3(H)			6(H)		H
									E
					5(C)				C
							6(F)		F
						6(D)	10(D)		D
								8(G)	G
									B
									A

En kısa yol: S,C,D,G,A uzaklık 8

Örnek 11.9



Şekil 11.14

S düğümünden tüm düğümlere uzaklığı uzaklığı saptayıp S'den A düğümüne en kısa yolu bulunuz. Dijkstra algoritmasını adapte ediniz. W(U,V) U'dan V düğümüne doğru kenarın ağırlığıdır.U,V arasında kenar yoksa ya da ters yönde ise (V'den U'a) ∞ yazılır.

Çözüm:

Adım 1.

$S \rightarrow 0(-)$ P={S}

Adım 2.

$C \rightarrow 5(S), D \rightarrow 3(S), F \rightarrow 2(S), E,A,G,B \rightarrow \infty(S)$

Adım3.

repeat

3.1P'de olmayan en küçük etiketli F düğümü var. 3(S) P={S,F}

3.2 F'e komşu P'de olmayan D,E,G var.

Düğüm	Eski	F'nin etiketi+W(F,X)	Min	Sonuç
D	3	2+1	---	3(S)
E	∞	2+3	5	5(F)
G	∞	2+2	4	4(F)
C				5(S)

A,B	∞(S)

3.1 En küçük etiketli D düğümü var. 3(S) P{S,F,D}

3.2 D'e komşu P'de olmayan C var.

Eski 5(S) Yeni 3+1=4 Sonuç C→4(D)

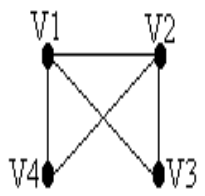
S	C	D	F	E	A	G	B	P'e eklenen
0(-)	5(S)	3(S)	2(S)	∞(S)	∞(S)	∞(S)	∞(S)	S
---	5(S)	3(S)	---	5(F)	∞(S)	4(F)	∞(S)	F
---	4(D)	---	---	5(F)	∞(S)	4(F)	∞(S)	D
---	---	---	---	5(F)	6(C)	4(F)	∞(S)	C
---	---	---	---	5(F)	5(G)	---	10(G)	G
---	---	---	---	---	5(G)	---	10(G)	E
---	---	---	---	---	---	---	10(G)	A
---	---	---	---	---	---	---	---	B
Uzaklıklar	4(D)	3(S)	2(S)	5(F)	5(G)	4(F)	10(G)	

S'den A'a en kısa yol S,F,G,A yolu,uzaklık=5

Teorem

Bir G grafının $V_1,V_2,V_3,...,V_n$ etiketli n düğümü ve komşuluk matrisi de A olsun. V_i ve V_j düğümleri arsındaki m uzunluklu yolların sayısı A^m matrisinin i,j. elemanına eşittir.

Teoremin doğruluğunu Şekil 11.15' de verilen graf için m=1,2,3 olarak görelim.



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

2 uzunluklu yolların sayısı için $A*A=A^2$ matrisini hesaplayalım.

Şekil 11.15

$$A^2 = \begin{bmatrix} 3 & 2 & 1 & 1 \\ 2 & 3 & 1 & 1 \\ 1 & 1 & 2 & 2 \\ 1 & 1 & 2 & 2 \end{bmatrix}$$

3.4 elemanının 2 olması V3 ile V4 arasında iki uzunluklu iki yol var demektir; $V_3\ V_2\ V_4$ ve $V_3\ V_1\ V_4$. Benzer şekilde 1.3 elemanının 1 olması V_1'den V_3'e 2 uzunluklu bir yol yani V_1, V_2, V_3 yolu var demektir. m=3 için A^3 hesaplanırsa;

$$A^3 = \begin{bmatrix} 4 & 5 & 5 & 5 \\ 5 & 4 & 5 & 5 \\ 5 & 5 & 2 & 2 \\ 5 & 5 & 2 & 2 \end{bmatrix}$$

Biçiminde elde edilir. 1,2 elemanı 5 olduğuna göre V_1'den V_2'ye 3 uzunluklu 5 yol var.

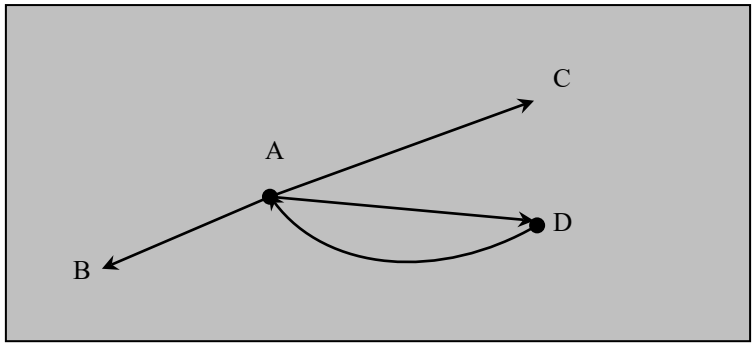
(1) V_1,V_2,V_1,V_2 (2) V_1,V_2,V_4,V_2

(3) V_1,V_2,V_3,V_2 (4) V_1,V_3,V_1,V_2

(5) V_1,V_4,V_1,V_2

11.3 Yönlü Graflar ve Çoklu Graflar

Şimdiye kadar gördüğümüz graflarda iki düğüm arasında kenar iki yönü de temsil etmekte idi. Ancak birçok durumda kenar, iki düğüm arasında tek yönlü bir ilişkiyi gösterme durumunda olabilir. Bu durumda kenarın da yönlü bir çizgi ile gösterilmesi gerekecektir. Örnek olarak şehir içi yolların büyük bir kısmı tek yönlüdür. Şekil 3.36 de bir yönlü graf örneği verilmiştir.



Şekil 11.16

Şekilden de görüldüğü gibi; A düğümünden B düğümüne yol olduğu halde B' den A' ya gidiş yoktur. A,D düğümleri arasında iki yönde ilişki mevcuttur.

Tanım
Bir yönlü graf (directed graph), sonlu boş olmayan bir V kümesi ve V'nin ayrık elemanlarının, sıralı çiftlerinin bir E kümesidir. V' nin elemanları düğümler, E'nin elemanları ise yönlü kenarlar olarak anılır.

Şekil 11.16'de A,B,C,D düğümleri ve (A,B),(A,D),(D,A),(A,C) yönlü kenarları vardır.

Yönlü bir grafta bir A düğümünden çıkan kenarların sayısına A' nın çıkış derecesi (outdegree of A) denir ve outdeg(A) ile gösterilir. Benzer biçimde A düğümüne gelen kenarların sayısı da A' nın giriş derecesi (indegree of A) olup indeg(A) olarak gösterilir. Şekil 11.16'de outdeg(A)=3 indeg(A)=1, outdeg(C)=0, indeg(C)=1, outdeg(D)=indeg(D)=1 v.b dir.

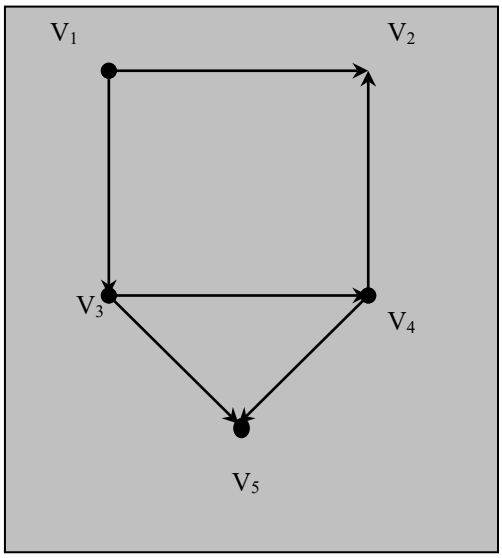
Teorem

Bir yönlü grafta ; giriş dereceleri toplamı, çıkış dereceleri toplamı ve yönlü kenar sayısı birbirine eşittir.

Graflarda olduğu gibi, yönlü bir graf matris ile gösterilebilir. Buna göre $V_1, V_2, \ldots, V_n$ düğümleri bir D yönlü grafının matris gösteriliminde, A(D) n*n komşuluk matrisi olup $A_{i,j}$ elemanı eğer V_i düğümünden V_j düğümüne doğru bir kenar var ise 1 aksi halde 0 olacaktır.

Örnek 11.10

Şekil 11.17'deki yönlü grafı ele alalım.



Şekil 11.17

$$A(D)= \begin{array}{c} \\ V1 \\ V2 \\ V3 \\ V4 \\ V5 \end{array} \begin{array}{c} \begin{array}{ccccc} V1 & V2 & V3 & V4 & V5 \end{array} \\ \begin{pmatrix} 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array}$$

A(D) matrisinde ikinci ve beşinci satırların tüm elemanları 0' dır. Çünkü V_2 ve V_5 düğümlerinden öteye hiç bir kenar yoktur. Benzer şekilde matrisin birinci sütunundan da V_1 düğümüne gelen hiçbir kenar olmadığını göstermektedir.

Teorem

Yönlü bir grafın komşuluk matrisinde i. Satırın toplamı o satıra karşılık gelen V_i düğümünün çıkış derecesini, j. Sütunun toplamı ise V_j düğümündeki giriş derecesini göstermektedir.

A(D) matrisinde, 3. Satırın toplamı 2 olduğuna göre outdeg(V_3)= 2, 2. Sütunun toplamı 2 olduğuna göre indeg(V_2)=1 olacaktır.

Yönlü graflar da komşuluk listesi ile gösterilebilirler. Burada listenin düzenlenmesinde her satırda, ona karşılık gelen düğümden giden kenarlara ilişkin düğümler listelenir. Şekil 3.37'deki graf için komşuluk listesi;

V_1:V_2, V_5

V_2:-

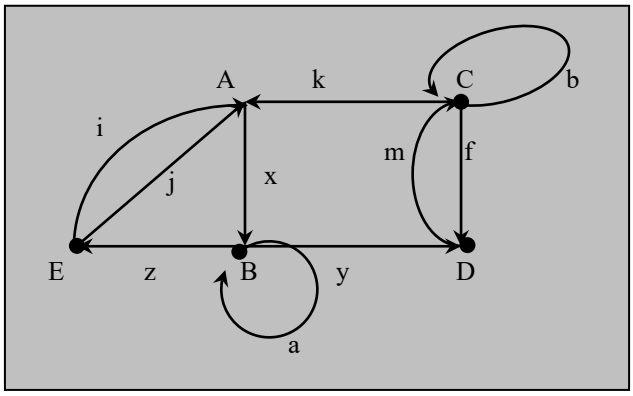
V_3:V_4,V_5

V_4:V_2,V_5

$V_{5:-}$ olacaktır.

11.3.1 Yönlü Çoklu Graflar

Daha önceki bölümlerde ele aldığımız çoklu graf, ağırlıklı graf, yol, basit yol, döngü gibi kavramların benzerleri yönlü çoklu graflar için de geçerlidir. Bunları göstermek için şekil 11.18'deki diyagramı ele alalım.



Şekil 11.18

C ve B düğümlerinde, b ve a yönlü çevrimler (directed loop) vardır. i,j kenarları ise E' den A'ya paralel yönlü kenarlar (parallel directed edges) adını alırlar. Burada dikkat edilirse f ve m kenarları aynı yönde olmadıkları için paralel yönlü kenar değildirler.

Bir $V_1,e_1,V_2,e_2,\ldots, V_n,e_n,V_{n+1}$ alterne dizisi V_1' den V_{n+1}' e yönlü yol adını (directed path) alır. Burada $e_i=(V_i,V_{i+1})$, $1\leq i \leq n$ yönlü yolun uzunluğu yol üzerindeki toplam kenar sayısıdır. Buna göre EjAxByD yolu E'den D'ye 3 uzunluklu bir yönlü yoldur. Bu yolu EABD ya da j,x,y olarak yazabiliriz.

EiAkC dizisine bakarsak bu E'den C'ye bir yönlü yol değildir. Çünkü k kenarı ters yöndedir.

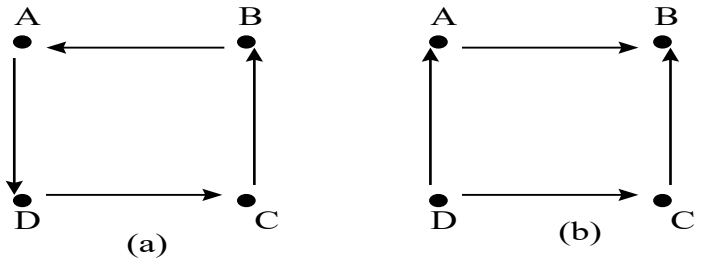
Yine yönlü çoklu graflarda, bir yol üzerinde hiçbir düğüm tekrarlanmıyorsa, bu yola basit yönlü yol (Simple directed path) denir. Şekil 11.18'de jxy yolu E'den D'ye basit bir yoldur.

Teorem

Her U-V yönlü yolu bir U-V basit yönlü yolu içerir.

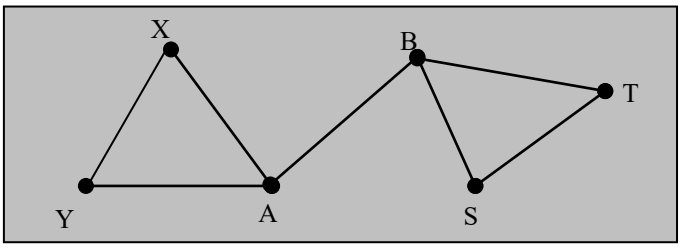
Şekil 11.18'de xymk yolu, yönlü döngüdür (directed cycle). Çünkü A düğümünden tekrar A düğümüne diğer hiçbir düğümden tekrar geçmeden gidilmektedir. yfkx ise yönlü döngü değildir. Çünkü yf kenarları ters yöndedir. Hem a,b kenarları, hem de f,m kenarları yönlü döngü sayılırlar. zjxym yönlü yolunda B düğümünden iki kez geçildiği için yönlü döngü değildirler.

Bir yönlü çoklu grafta, her bir A,B düğümü arasında bir yönlü yol var ise bu grafa sıkı bağlı(strongle connected) dır denir. Şekil 11.19(a) daki graf sıkı bağlı bir graf olduğu halde şekil 3.39(b) deki graf sıkı bağlı değildir. Çünkü, örneğin A' dan C' ye bir yol yoktur.



Şekil 11.19

Eğer herhangi bir kenarı graftan çıkardığımızda graf ikiye bölünüyorsa, (bağlı graf olmuyorsa) bu durumda bu grafı sıkı bağlı bir graf yapamayız. Şekil 3.40'deki grafta A,B kenarının kalkması grafı iki parçaya bölmektedir. Bu durumda bu graf sıkı bağlı yönlü grafa dönüştürülemez.



Şekil 11.20

11.3.2 Yönlü Euler Halkası ve yolu

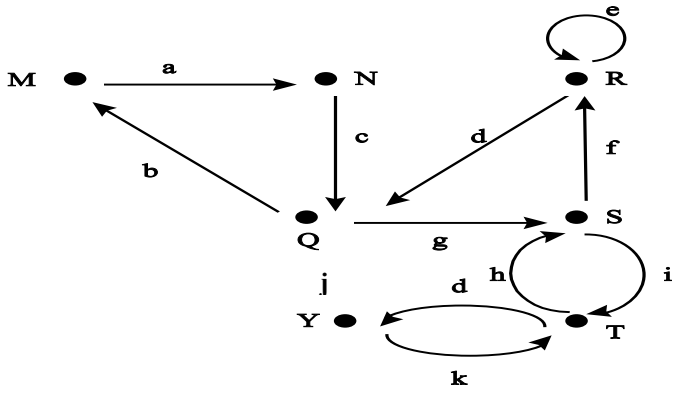
Çoklu graflardaki Euler Halkası ve Yolu tanımlarını yönlü çoklu graflarda da uygulayabiliriz. Bir D yönlü çoklu grafında, bir yönlü yol eğer D grafındaki tüm kenarlardan sadece bir kez geçiyor ve başlangıç ve bitiş düğümleri farklı ise bu yola yönlü Euler Yolu, başlangıç bitiş düğümleri aynı ise bu yola yönlü Euler Halkası adı verilir.

Teorem

Bir yönlü D grafında, kenarların yönlerini ihmal ettiğimizde, elde edilecek grafın bağlı olduğunu varsayalım. Bu durumda D yönlü grafı, ancak ve ancak D nin B gibi bir düğümünün çıkış derecesi,giriş derecesinden 1 fazla, C gibi bir düğümünün giriş derecesi çıkış derecesinden 1 fazla ve geri kalan tüm düğümlerin giriş dereceleri,çıkış derecelerine eşit ise, bu durumda D yönlü grafının B düğümünden başlayıp C düğümünde biten bir yönlü Euler Yolu vardır.

Daha önce ele aldığımız Euler Halkası algoritması, doğal olarak(düğümü terk eden yönlü kenarı seçmek biçiminde) yönlü kenarlara göre değiştirilerek kullanılabilir.

Örnek 11.11



Şekil 11.21

Şekil 11.21'deki yönlü çoklu grafta bir yönlü Euler halkası ya da yolu var mıdır? Eğer var ise uygun bir algoritmayla bu halkayı bulunuz.

a,c,g,h,j,k,i,f,e,d,b

D grafı yönleri düşünülmediğinde bağlı bir graftır. Herhangi iki düğüm arasında yol var ve her bir düğümün giriş derecesi birbirine eşittir. Bu yönlü grafta Euler halkası vardır.

Adım 1: a) E={a,b,c,d,e,f,g,h,i,j,k}

b)C={M}

Adım 2.1 M(den öteye) a kenarı ile N'e bağlı

a)A=M

b)P=M

2.2 a)B=M

b)while (E'de B=M'den öteye kenar var)

a)a kenarı

E={b,c,d,e,f,g,h,i,j,k}

b)B=N

c)P={M,a,N}

while B=N'den öteye c var

a)c kenarı

E={b,,d,e,f,g,h,i,j,k}

b)B=Q

c)P={M,a,N,c,Q}

while Q'dan öteye b,g var

a)b kenarı

E={d,e,f,g,h,i,j,k}

b)B=M

c)P={M,a,N,c,Q,b,M}

endwhile.

2.3 C={M,a,N,c,Q,b,M} E={d,e,f,g,h,i,j,k}

2.1 while C'de Q'dan öteye E'da g var

2.2 B=Q

while Q'dan öteye g var

a)g kenarı E={d,e,f,h,i,j,k}

b)B=S

c)P={Q,g,S}

while S'den öteye f,i var

a)i kenarı E={d,e,f,h,j,k}

b)B=T

c)P={Q,g,S,i,T}

while T'den öteye j var

a)j kenarı E={d,e,f,h,k}

b)B=Y

c)P={Q,g,S,i,T,j,Y}

while Y'den öteye k var

a)k kenarı E={d,e,f,h}

b)B=T

c)P={Q,g,S,i,T,j,Y,k,T}

while T'den öteye h var

a)h kenarı E={d,e,f}

b)B=S

c)P={Q,g,S,i,T,j,Y,k,T,h,S}

while S'den öteye f var

a)f kenarı E={d,e}

b)B=R

c)P={Q,g,S,i,T,j,Y,k,T,h,S,f,R}

while R'den öteye e,d var

a)e kenarı E={d}

b)B=R

c)P={Q,g,S,i,T,j,Y,k,T,h,S,f,R,e,R}

while R'den öteye d var

a)d kenarı E={ }

b)B=Q

c)P={Q,g,S,i,T,j,Y,k,T,h,S,f,R,e,R,d,Q}

endwhile.

2.3 C={M,a,N,c,Q,g,S,i,T,j,Y,k,T,h,S,f,R,e,R,d,Q,b,M}

endwhile.

C={a,c,g,i,j,k,h,f,e,d,b} kenarları Yönlü Euler halkasını oluşturur.

11.3.3 Yönlü Hamiltonian Döngüsü ve Yolları

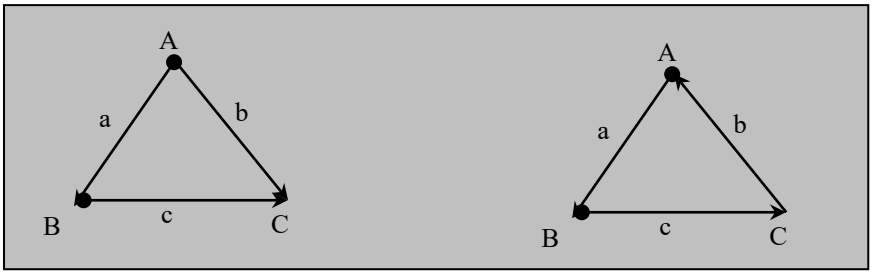
Bir yönlü Hamiltonian Döngüsü(yolu), her bir düğümü sadece bir kez içeren bir yönlü döngüdür(yol). Hamiltonian döngüsünde yönlü çevrimlere ve paralel kenarlara gerek olmadığı için burada yönlü çoklu graf yerine, yönlü graf ele alınacaktır. Graflarda olduğu gibi yönlü graflarda da yönlü Hamiltonian döngüsünün varlığına karar vermek ve bulmak zordur.

Bu kavramların kullanıldığı bir uygulama Carak round ro türü yarışma verilebilir. Bu tür yarışmada her takım diğerleriyle bir kez maç yapar ve beraberlik yoktur. Böyle bir yarışma bir yönlü modellenebilir. Takımlar düğümlere karşılık düşürülür ve kenarlarda iki düğüm arasındaki maç sonucu belirtecek

şekilde yönlendirilir. Örneğin A takımı B takımını yenmişse AB kenarı A dan B ye doğru yönlenecektir. Bu tür graflara turnuva adı verilir. Buna göre graf K_n bağlı graf olup, bir kenar yönlendirilmiştir.

Örnek 11.12

A,B,C gibi üç takım olsun. Şekil 11.22a'ya ya göre A takımı B ve C yi, B takımı da C yi yenmiştir. 3.42b'de ise A,Byi; B,C yi; C de A yı yenmiştir.



(a) (b)

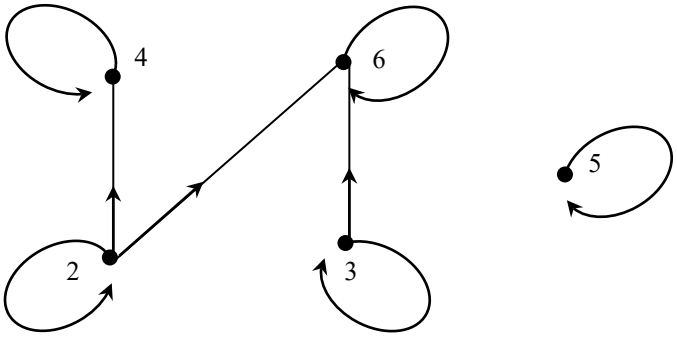
Şekil 11.22

Turnuvada Hamiltonian yolu, birinci takımın ikinciyi yendiği, ikincinin üçüncüyü yendiği, v.b bir sıralamayı göstermektedir. Şekil 3.42a' da bir tek Hamiltınian yolu a,c yolu vardır. Şekil 3.42b'de ise üç ayrı Hamiltonian Yolu , ac,cb,ba yani üç ayrı sıralama vardır. Genellersek, bir turnuvada yönlü döngü yoksa, takımların tek sıralamasını veren sadece bir Hamiltonian yolu vardır.

Önceki bölümlerde, kümelerde bağıntıları görmüştük küme elemanları arasındaki ilişkileri temsil etmek için yönlü grafları kullanabiliriz. Bu yönlü çoklu grafta düğümler küme elemanlarına x düğümünden y düğümüne yönlü kenar ise xRy ilişkisini temsil edecektir.

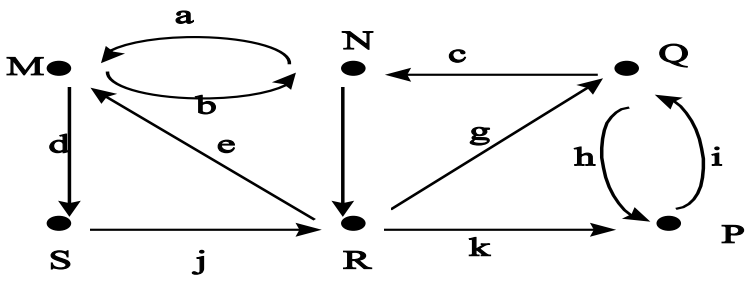
Örnek 11.13

S={2,3,4,5,6} kümesinde xRy ilişkisi " x sayısı y ye bölünebilir" olsun, buna göre SxS kümesinden
R={(2,2),(2,4),(2,6),(3,3),(4,4),(5,5),(6,6)}
elde edilecektir. Bu ilişkiyi yönlü çoklu grafla gösterirsek şekil 11.23'daki gibi olacaktır.



Şekil 11.23

Örnek 11.14



Şekil 11.24

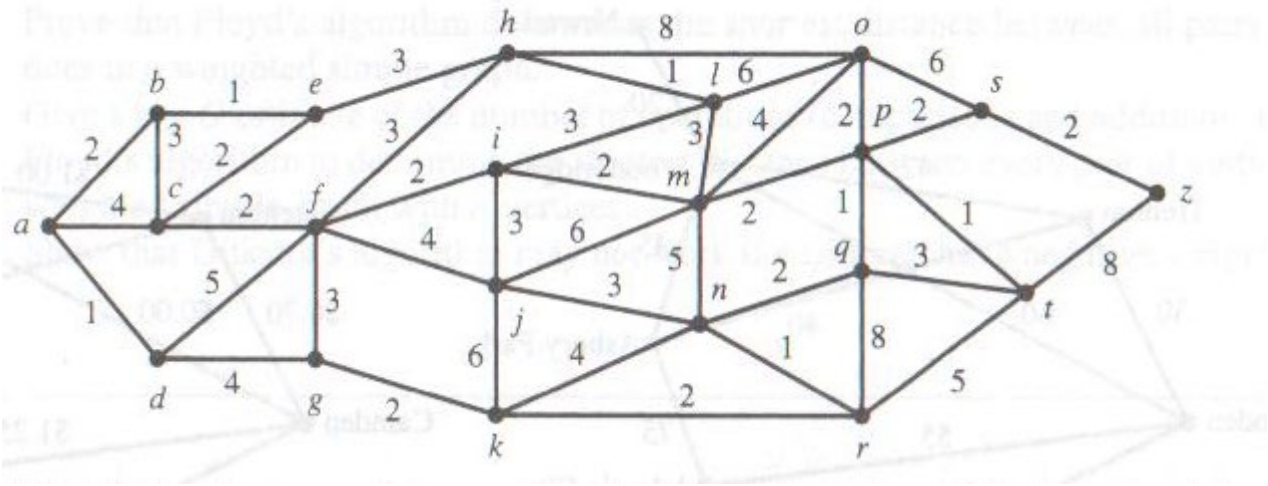
Çözüm:

Şekil 11.24'deki yönlü çoklu grafta Euler halkası ya da yolu var mıdır? Varsa bulunuz.

Düğüm	indegree	outdegree	indegree-outdegree
M	2	2	0
N	3	1	2
Q	2	2	0
P	2	1	1
R	1	4	-3
S	1	1	0

N,P,R düğümlerinin giriş ve çıkış dereceleri arasındaki farklar 2,1 ve -3 olduğu için bu grafta Euler halkası ya da yolu yoktur.

Ödev



Yukarıdaki graf için a-z arasındaki en kısa yolu dijkstra tablosu ile bulunuz.

Kaynaklar

F.Selçuk,N.Yurtay,N.Yumuşak,Ayrık İşlemsel Yapılar, Sakarya Kitabevi,2005.
İ.Kara, Olasılık, Bilim Teknik Yayınevi, Eskişehir, 2000.
"Soyut Matematik", S.Aktaş,H.Hacısalihoğlu,Z.Özel,A.Sabuncuoğlu, Gazi Ünv.Yayınları,1984,Ankara.
"Applied Combinatorics", Alan Tucker, John Wiley&Sons Inc, 1994.
"Applications of Discrete Mathematics", John G. Michaels, Kenneth H. Rosen, McGraw-Hill International Edition, 1991.
"Discrete Mathematics", Paul F. Dierker and William L.Voxman, Harcourt Brace Jovanovich International Edition, 1986.
"Discrete Mathematic and Its Applications", Kenneth H. Rosen, McGraw-Hill International Editions, 5th Edition, 1999.
"Discrete Mathematics", Richard Johnson Baugh, Prentice Hall, Fifth Edition, 2001.
"Discrete Mathematics with Graph Theory" , Edgar G. Goodaire, Michael M. Parmenter, Prentice Hall, 2nd Edition, 2001.
"Discrete Mathematics Using a Computer", Cordelia Hall and John O'Donnell, Springer, 2000.
"Discrete Mathematics with Combinatorics", James A. Anderson, Prentice Hall, 2000.
"Discrete and Combinatorial Mathematics", Ralph P. Grimaldi, Addison-Wesley, 1998.
"Discrete Mathematics", John A. Dossey, Albert D. Otto, Lawrence E. Spence, C. Vanden Eynden, Pearson Addison Wesley; 4th edition 2001.
"Essence of Discrete Mathematics", Neville Dean, Prentice Hall PTR, 1st Edition, 1996.
"Mathematics:A Discrete Introduction", Edvard R. Schneiderman, Brooks Cole; 1st edition, 2000.
"Mathematics for Computer Science", A.Arnold and I.Guessarian, Prentice Hall, 1996.
"Theory and Problems of Discrete Mathematics", Seymour Lipschuts, Marc. L. Lipson, Shaum's Outline Series, McGraw-Hill Book Company, 1997.
"2000 Solved Problems in Discrete Mathematics", Seymour Lipschuts, McGraw- Hill Trade, 1991.