YÖNEYLEM ARAŞTIRMASI

16.10.2020
Saat 9⁰⁰ 11⁵⁰

ŞEBEKE MODELLERİ

Yöneylem Araştırmasında, şebeke (Dallarla birbirine bağlanan düğümler) olarak uygun bir biçimde Modellenip çözülebilen çok sayıda durum vardır.

1) Herhangi bölgedeki kuyuları iç kesimdeki teslim noktalarına bağlayan doğal gaz boru hattı projesinin tasarımı

2) Herhangi bir yol şebekesinde iki şehir arasında en kısa yolun belirlenmesi

3) Boru hattı şebekesinde maksimum kapasitenin belirlenmesi

4) Boru hattı şebekesinin minimum Maliyetle akış çizelgesinin belirlenmesi

5) Herhagi bir projenin faaliyetleri için zaman çizelgesinin belirlenmesi

yukarıda 5 maddeyi optimum şekilde gerçekleyen 5 algoritma verilecektir.

1. Minimum kapsayan ağaç algoritması (1 durum)
2. En kısa yol algoritması (2 durum)
3. Maksimum Akış algoritması (3. durum)
4. Minimum Maliyetli şebeke algoritması (4. durum)
5. kritik yol algoritması (5. durum)

2

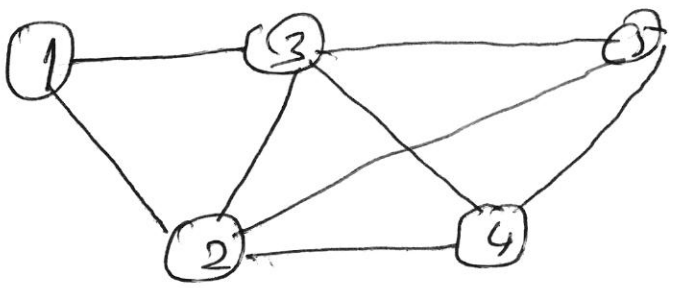
ŞEBEKE TANIMLARI

Bir şebeke bağlantılar (veya dallar) ile birbirine bağlanmış bir dizi düğümden oluşur. Gözönüne aldığımız bir şebeke (N, A) notasyonu ile ifade edilir. Burada N düğümler kümesi, A ise bağlantılar kümesidir.

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

$$A = \{(1,3), (1,2) (2,3) (2,4) (2,5) (3,4) (3,5) (4,5)\}$$

olan bir şebeke aşağıda gösterilmiştir.



(6.1)

Her şebekenin kendine özgü bir akış tipi vardır. (Örneğin petrol ürünleri boru hattından, trafik bir kara yolu şebekesinden akar. Genellikle bir şebekedeki akış, şebekedeki bağlantıların sonlu veya sonsuz olabilen kapasitelerle sınırlıdır.

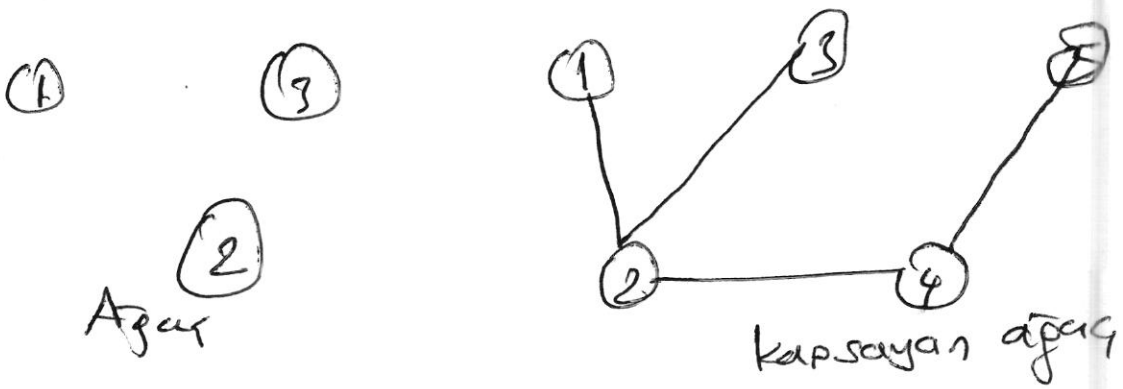
Bir yönde pozitif akışa; diğer yönde sıfır akışa izin veren bir bağlantının yönlendirilmiş veya yönelimli olduğu söylenir. Yönlendirilmiş bir şebekenin tüm dalları yönlendirilmiştir.

Yol: Her bir daldaki akışın yönüne bakmaksızın iki düğümü birleştiren dalların sırasıdır.
Yol bir düğümü kendisine bağlıyorsa bir döngü veya bir çevrim oluşturur. Örneğin (2,3), (3,4) (4,2) bir döngüdür.

3

Bağlı şebeke: Herhangi iki düğümün en azından bir yolla bağlanmasıdır. Şekil (6.1)'de bu tip şebeke görülmektedir.

Ağaç: Bağlı şebekenin tüm düğümlerinin sadece bir alt kümesini ilişkilendiren bir şebekedir. Öysa kapsayan ağaç, şebekenin tüm düğümlerini hiç bir döngüye izin vermeden birbirine bağlar. Burada kapsayan ağaç şebekenin tüm düğümlerini bağlayan ağaçtır. Şekil (6.2)'de bir ağaç, bir de kapsayan ağaç örneği verilmiştir.



MINIMUM KAPSAYAN AĞAÇ: Bu algoritma doğrudan veya dolaylı olarak dalların en kısa bağlantısını kullanarak şebekenin dallarının biriyle ilişkilendirilmesini ele alır. İki şehir arasındaki bir veya daha fazla kasabayı birbirine bağlayan tali yolların yapımının gerçekleştirilmesi bunun tipik bir uygulamasıdır. Yol sisteminin en ekonomik tasarımı, tali yolların toplam uzunluğunun minimum kılınmasıdır. Prosedürün aşamaları aşağıda verilmiştir.

$N = \{1, 2, \ldots n\}$ Şebekenin düğümler kümesi olsun.

4

C_k = Algoritmanın k.ci iterasyonunda kalıcı olarak bağlanmış düğümler kümesi

$\bar{C}_k$ = Henüz kalıcı olarak bağlanmamış düğümler kümesi olsun.

0. adım: $C_0 = \phi$ ve $\bar{C}_0 = N$ olarak belirle.

1. adım: Bağlanmamış düğümler kümesi $\bar{C}_0$'deki herhangi bir i düğümüyle başlayıp $C_1 = \{i\}$ olarak belirle. Bu otomatik olarak $\bar{C}_1 = N - \{i\}$ sonucunu verir.

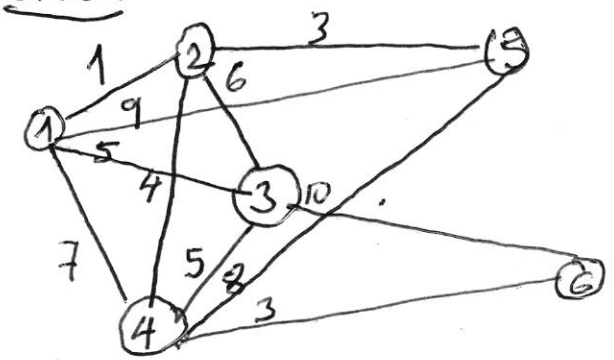
$k = 2$ olarak belirle.

k.genel adım: Bağlanmış düğümler kümesi C_{k-1} deki bir düğüme en kısa bağlantıyı verecek şekilde $\bar{C}_{k-1}$ bağlanmamış düğümler kümesinden bir j^* düğümü seç. Bu j^*'ı C_{k-1}'e kalıcı olarak bağla. $\bar{C}_{k-1}$ kümesinden çıkar.

$$C_k = C_{k-1} + \{j^*\} \qquad \bar{C}_k = \bar{C}_{k-1} - \{j^*\}$$

ifadesiyle gösterilir. Bağlanmamış düğümler kümesi $\bar{C}_k$ boş kümeyse dur. Aksi halde $k = k+1$ olarak belirle ve adımı tekrar et

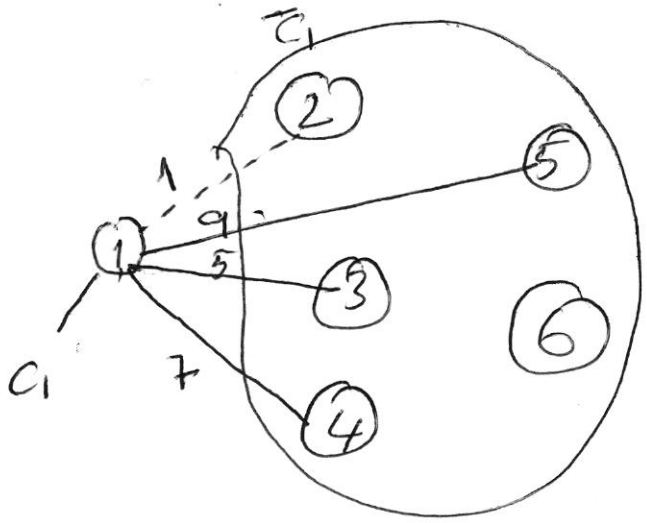
Örnek



Algoritma 1 düğümle başlayalım.

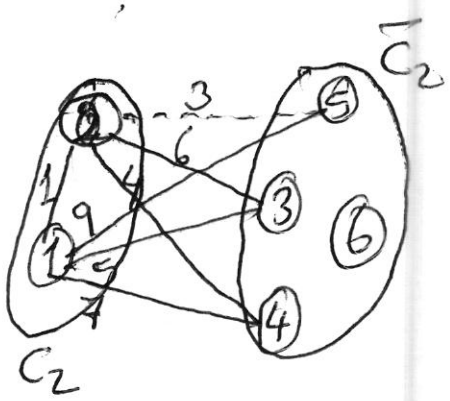
$C_{k-1} = \{1\}$ $\bar{C}_1 = \{2, 3, 4, 5, 6\}$

5



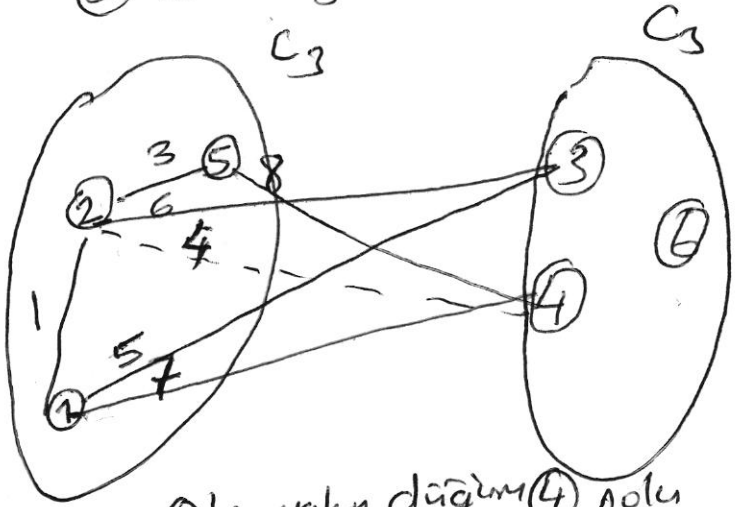
(1)

① nolu düğüme en yakın düğüm ① nolu düğümdür ve uzaklık 1 dir.

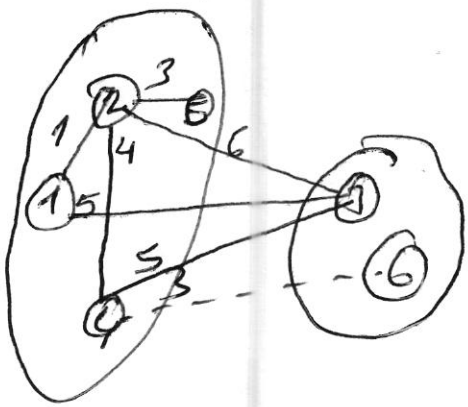


(2)

① yada ② nolu düğüme en yakın düğüm 5 dir.
(2,5) uzaklığı 3 dür. 1+3=4

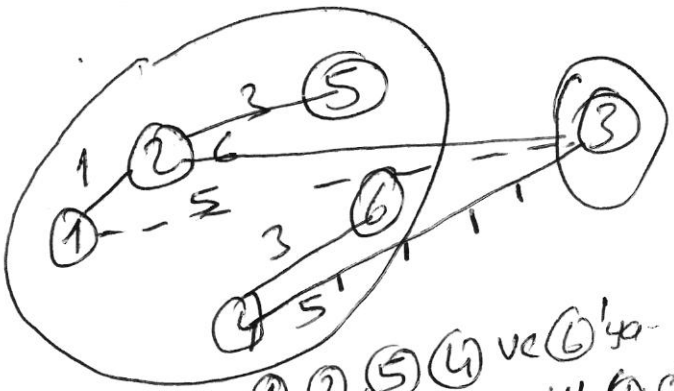


①, ② ve ⑤'e en yakın düğüm ④ nolu düğümdür (3) uzaklık 4'dür (2,4) 4'de daha önceki uzaklık 8 birim oldu.

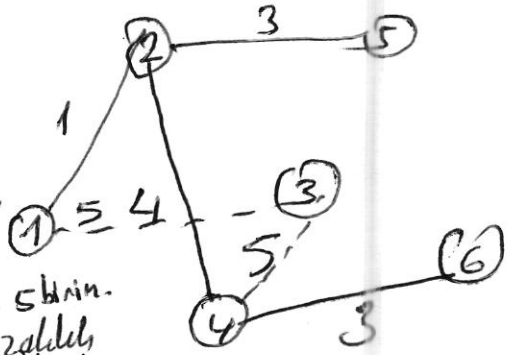


(4)

①, ②, ⑤ ve ④'e en yakın uzaklık 3 birim (4,6) olup. toplam uzaklık 11 olur.

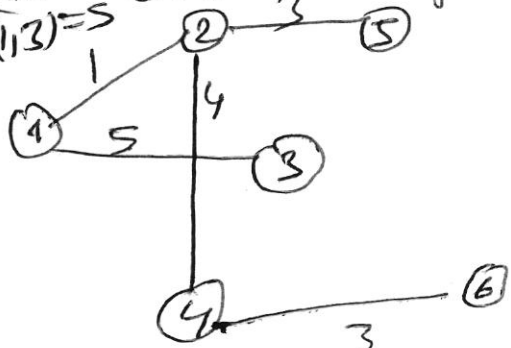


①, ②, ⑤, ④ ve ⑥'ya en yakın uzaklık ④, ③ 5 birim. ③ düğümü uzaklık ⑥ birim.

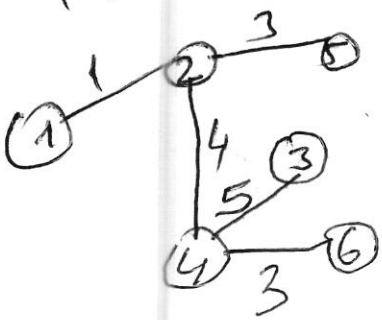


(6)

(5)
yada (1,3) = 5



yada



6 En kısa yol algoritmaları

Bu bölümde döngü içeren yada döngü içermeyen şebekeler için iki algoritma tanıtılacaktır.

1. Dijkstra
2) Floyd

Dijkstra Algoritması, kaynak düğümüyle şebekedeki diğer düğümler arasındaki en kısa yolu belirlemek için tasarlanmıştır. Floyd ise herhangi iki düğüm şebekedeki en kısa yolun belirlenmesine izin verdiği için daha geneldir.

Dijkstra Algoritmasını Algoritmanın hesaplamaları, özel bir etiketleme prosedürü kullanılarak, i düğümünden en kısa uzaklık olan ve $d_{ij} (\geq 0)$ (i,j) bağlantısının uzunluğu olarak tanımlansın. Sonrasında j düğümü için etiket

$$[u_j, i] = [u_i + d_{ij}, i] \quad d_{ij} \geq 0$$

şeklinde tanımlanır.

Dijkstra algoritmasında düğüm etiketleri geçici ve kalıcı olmak üzere iki tiptir. Geçici etiket aynı düğüme daha kısa bir yol bulunursa başka bir etiketle değiştirilebilir. Daha iyi başka bir yolun bulunamıyacağının açıkça ortaya çıktığı durumda geçici etiketin statüsü kalıcıya dönüşür.

Algoritmanın adımları

0. adım kaynak düğümü (1 düğüm) kalıcı etiket $[0,-]$ ile etiketle $i=1$ olarak belirle.

7

i. adım

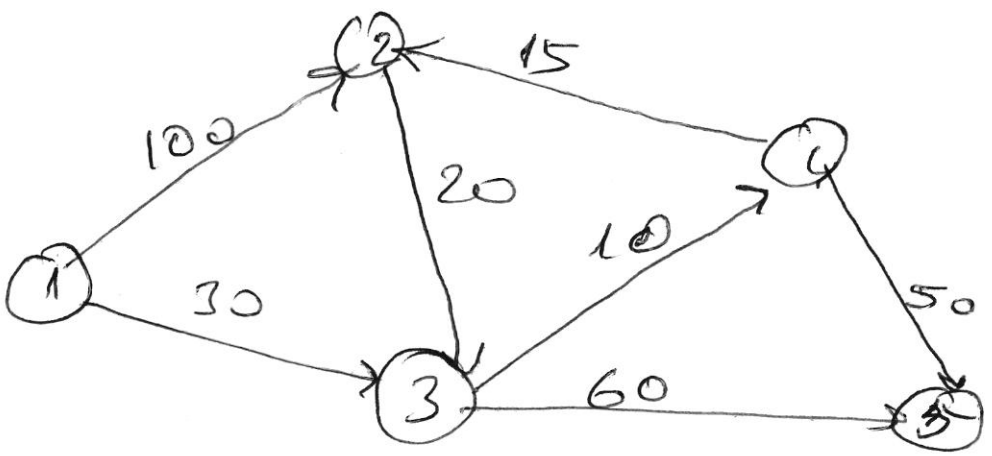
(a) j'nin kalıcı etiketlenmemiş olması koşuluyla

· i. düğümden ulaşılabilen her j. düğümü için geçici $[u_i + d_{ij}, i]$ etiketlerini hesapla j düğümü başka bir k düğümü içersinde $[u_j, k]$ ile zaten etiketlenmişse $u_i + d_{ij} < u_j$ ise ise. $[u_j, k]$'yi $[u_i + d_{ij}, i]$ ile değiştir.

(b) Tüm düğümlerde kalıcı etiket var ise dur.

Aksi halde tüm geçici etiketler arasından $[u_r, s]$'nin en kısa mesafesi ($= u_r$) olanını seç. (eşitlik durumunda herhangi birini rasgele seç) r'yi olarak belirle ve i. adımı tekrarla.

Örnek aşağıdaki şekilde 1 şehir (1. düğüm) ile diğer dört şehir. (2. düğümden 5. düğüme) arasındaki yollar ve bunların km cinsinden uzaklıklarını göstermektedir. 1. şehirden geriye kalan 4 şehire en kısa yolu belirlemek istiyoruz.



0. adım kalıcı etiket $[0, -]$'yi 1. düğüme ata

8 1 adım: 2ci ve 3cü düğümlere (en son kalıcı olarak etiketlenen) 1. düğümden ulaşılabilir. Böylece etiketlenen düğümlerin (geçici ve kalıcı) listesi aşağıdaki hale gelir.

Düğüm	Etiket	Statü
1	[0 -]	kalıcı
2	[0+100, 1] = [100, 1]	Geçici
3	[0+30, 1] = [30, 1] ← geçici	

2. ve 3. düğümün etiketleri olan ikisi geçici etiketten [100, 1] ve [30, 1]'e bakıldığında 3 düğümün daha kısa uzaklığı verdiği görülür ($U_3 = 30$) Dolayısıyla 3. Düğümün etiketi kalıcı olarak değiştirilir.

2 adım: 4. ve 5. düğümlere 3. düğümden ulaşılabilir. ve etiketlenen düğümler listesi aşağıdaki gibi oluşturulur.

Düğüm	Etiket	statü.
1	[0 -]	kalıcı
2	[100, 1]	geçici
3	[30, 1]	kalıcı
4	[30+10, 3]	← geçici
5	[30+60, 3]	geçici

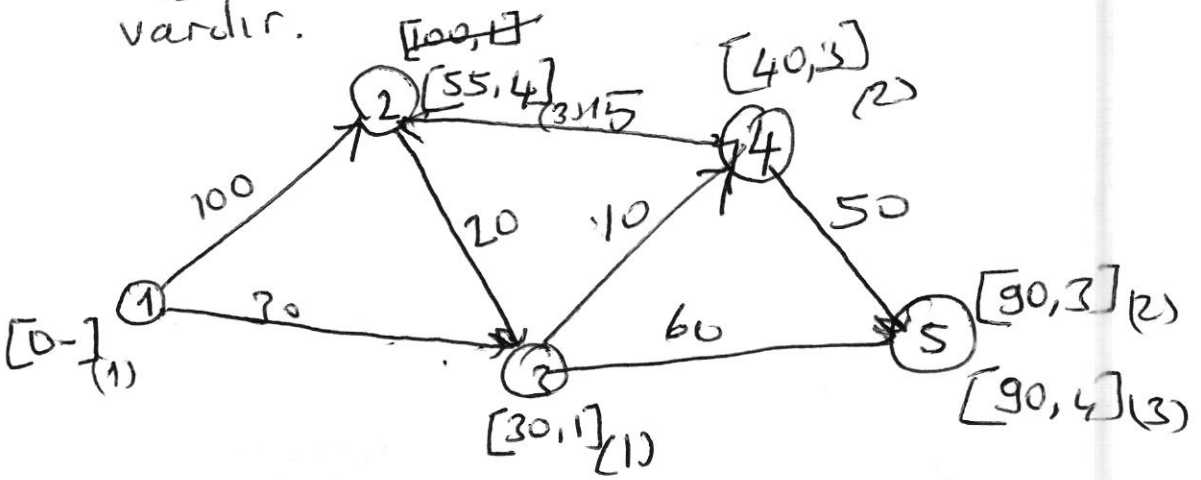
4 düğümdeki [40, 3] etiketin statüsü kalıcı olarak değiştirilir.

[9]

3. Adım. 2. ve 5. düğümlere 4. düğümden ulaşılabilir. Böylece etiketlenmemiş düğümlerin listesi aşağıdaki gibi güncellestirilir.

Düğüm	Etiket	Statü
1	[0 -]	kalıcı
2	[40+15, 4]	geçici
3	[30, 1]	kalıcı
4	[40, 3]	kalıcı
5	[90, 3] veya [90, 4]	Geçici idi.

2 adımda 2. düğümün geçici etiketi [100, 1], 3 adımda 4 düğümden 2. düğüme daha kısa yol bulunduğu için [55, 4] olarak 2 düğüm değiştirilir. Ayrıca 3. cü adımda 5. ci düğümün aynı uzaklığa sahip. (3 düğümdende 4'ü de düğümlerde 90 km) $a_5 = 90$ km iki alternatifi vardır.



1 düğüm ile şebekedeki başka düğüm arasındaki en kısa yolu belirlemek için istenen varış düğümünden başlanır ve kalıcı etiketlerin verdiği bilgi kullanılarak düğümlerden geriye doğru gidilir. 5. ci düğümden 2. ci düğüme en kısa yol 2 → [55, 4] – 4 → [40, 3] → 3 → [30, 1] ⇒ 1