İSTANBUL ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

DOKTORA TEZİ

MAKSİMUM AKIŞ PROBLEMİ, GÖRÜNTÜ TAŞINIMI, SAKLANMASI VE ÇAĞRIMI İÇİN BİR ALGORİTMA

Heysem İSTANBULLU
Matematik Anabilim Dalı

Danışman
Prof.Dr. Bedriye M. ZEREN
İkinci Danışman
Prof.Dr. M. Nizamettin ERDURAN

Kasım, 2009

İSTANBUL

Bu çalışma İstanbul Üniversitesi Bilimsel Araştırma Projeleri Yürütücü Sekreterliğinin 1882 numaralı projesi ile desteklenmiştir.

ÖNSÖZ

Yüksek lisans ve doktora çalışmam boyunca gösterdiği her türlü destek ve yardımlarından dolayı danışman hocam sayın Prof. Dr. Bedriye M. ZEREN'e çok teşekkür ederim.

Doktora tez çalışmamda ikinci tez danışmanım olmayı kabül eden doktora tez çalışmam boyunca özellikle bilgisayar konusundaki yardımlarından dolayı hocam sayın Prof. Dr. M. Nizamettin ERDURAN'a çok teşekkür ederim.

Doktora ders ve tez aşamasında gösterdiği her türlü yardım ve desteklerinden dolayı tez izleme komitesi üyesi olan hocam sayın Prof.Dr. Müfit GİRESUNLU'ya çok teşekkür ederim.

Eğitim-öğretim hayatım boyunca maddi manevi her türlü yardımlarımdan dolayı aileme çok teşekkür ederim.

Kasım, 2009 **Heysem İSTANBULLU**

İÇİNDEKİLER

ŞEKİL LİSTESİ

SEMBOL LİSTESİ

N	: Düğümler kümesi
A	: Arklar kümesi
$G = (N ; A)$	: N düğümler kümesi ve A arklar kümesinden oluşan şebeke
$G(x)$	: x akışından sonra arta kalan şebeke
$G(x, \Delta)$	: Δ - arta kalan şebekesi
n	: Şebekedeki düğüm sayısı
m	: Şebekedeki ark sayısı
U	: Şebekedeki kapasitesi en büyük olan arkın kapasitesi
s	: Şebekedeki kaynak düğümü
t	: Şebekedeki kuyu düğümü
i	: Şebekedeki i düğümü
(i, j)	: i düğümünden j düğümüne giden ark
b_{ij}	: (i, j) arkının boş kapasitesi
x_{ij}	: (i, j) arkından geçen akış miktarı
r_{ij}	: (i, j) arkındaki arta kalan kapasite
$d(i)$	: i düğümünün uzaklık etiketi
$e(i)$	: i düğümündeki fazlalık
$\left[S, \overline{S}\right]$	: Bir kesit
$C\left[S, \overline{S}\right]$	: $\left[S, \overline{S}\right]$ kesitinin kapasitesi
$O(\ )$	: Büyük O sembolü, Landau sembolü
γ_i	: Kaynak düğümünden j düğümüne gelen yol üzerinde, j düğümünden önceki düğüm
δ_j	: Kaynak düğümünden j düğümüne gelen yol üzerindeki maksimum boş kapasite
AS	: Ark sayısı
GA	: Geliştirilen Algoritma
KÖA	: Kapasite ölçekleme algoritması
EBİA	: En büyük etiketli akış öncesi gönderme algoritması
CPU	: Central processing unit, merkezi işlem birimi, işlemci
n	:n elemanlı tek kuyu düğümü ve tek kaynak düğümü olan şebekenin matris gösteriminde kuyu düğümü
1	:n elemanlı tek kuyu düğümü ve tek kaynak düğümü olan şebekenin matris gösteriminde kaynak düğümü

ÖZET

MAKSİMUM AKIŞ PROBLEMİ, GÖRÜNTÜ TAŞINIMI, SAKLANMASI VE ÇAĞRIMI İÇİN BİR ALGORİTMA

Şebeke akış problemleri özellikle uygulamalı matematik, bilgisayar bilimleri, mühendislik, işletme ve yöneylem araştırmaları alanları içinde yer almaktadır. Şebeke akışı teorisinin ilk çalışmalarının temel fikri, bazı fiziksel sistemlerin matematiksel nesnelerle ifade edilebilmesidir.

Biz burada "akış göndermek için kullanılabilecek alternatif yolları olan bir şebekede en uygun maliyetli yol hangisidir?" sorusunun cevabıyla ilgileneceğiz.

Uygulamada karşılaşılan birçok problemi çözmek için gerekli olan algoritmaların seçiminde mümkün olan en az işlem sayısıyla çalışan algoritmalar tercih edilmelidir. O halde bu algoritmalar şebeke büyüdükçe çalışma süresinin çok hızlı artmasını engelleyen özelliklere sahip algoritmalar olmalıdır.

Genel kısımlar bölümünde şebekeler ve maksimum akış problemi hakkında önce genel bilgiler ve tanımlar ve daha sonra maksimum akış probleminin bilinen en etkin algoritmaları verilmektedir.

Genel kısımlar bölümünde verilen algoritmaların bulgular bölümünde matris gösterimi yardımıyla nasıl uygulanabileceği ayrıntılı olarak verilmiştir. Ayrıca bu tez çalışması kapsamında maksimum akış problemi için geliştirilen yeni bir algoritma da verilmiştir. Geliştirilen algoritma ile kapasite ölçekleme algoritması ve en büyük etiketli akış öncesi gönderme algoritmasına dayalı kodlar C bilgisayar programlama dilinde yazılmış ve karşılaştırmaları da yapılmıştır. Bu tez çalışmasında geliştirilen algoritma bağımsız bir bilgisayar ağında denenmiş, büyük hacimli görüntü dosyalarının taşınması, saklanması ve geri çağırımı süreçleri için kullanılmıştır.

SUMMARY

AN ALGORTIHM FOR MAXIMUM FLOW PROBLEM, IMAGE TRANSFER, STORAGE AND RETRIEVAL

Network flows is a problem domain that lies especially applied mathematics, computer science, engineering, management and operation research. The early work about the network flow theory was to establish networks as useful mathematical objects for representing some physical systems.

In this work we try to answer the question of how the most cost-effective alternative ways can be established in order to send flows in to networks.

To solve many problems that we encounter in applications, algorithms with minimum run-time must be preferred. In addition these algorithms should not let fast increase in run time as the networks grow.

In the chapter entitled “GENERAL SECTION” of this thesis definitions about network flows and maximum flow problems are given and some well known and effective algorithm for maximum flow problem are presented.

In the chapter entitled “FINDINGS SECTION” of this thesis matrix representation of algorithms is presented. In the “GENERAL SECTION” applicability of these representations are given in details with examples. Moreover, a new improved algorithm for maximum flow problem is developed and introduced in details. This new algorithm, capacity scaling algorithm and highest label algorithm are coded in C computer programming language and their properties are compared in run time environment. This new improved algorithm is then applied on the computer network for image files with very large sizes to transfer, storage and retrieval processes.

1. GİRİŞ

Günümüzde şebeke (network) sözcüğü ile sıkça karşılaşılmaktadır. Örneğin elektrik akım şebekeleri ile evlerimize enerji taşınmakta, telefon şebekeleri ile şehirler ve milletler arası iletişim sağlanmaktadır. Karayolu şebekeleri, demiryolu şebekeleri, havayolu şebekeleri ve denizyolu şebekeleri ile büyük coğrafik yerler arasında gidip gelmeyi, üretim dağıtım şebekeleri ile yiyecek ve tüketim mallarına ulaşımı gerçekleştirmekte, bilgisayar şebekeleri de iş ve kişisel yaşantıda bilgi paylaşımı sağlamaktadır.

Tüm bu problem alanları ve diğer birçok alanda bir düğümden başka bir düğüme mümkün olan en etkin şekilde akış sağlanması istenilmektedir. Şebeke akış problemleri olarak adlandırılan bu tür problemlerin çözümleri için matematiksel algoritmalar geliştirilmiştir [1, 2, 3].

Şebeke akış problemleri özellikle uygulamalı matematik, bilgisayar bilimleri, mühendislik, işletme ve yöneylem araştırmaları alanları içinde yer almaktadır. Bu alan Gustav Kirchhof ve ondan önce elektrik devreleri ile sistematik olarak uğraşan elektrik mühendisleri ve mekanikçilere kadar dayanmaktadır. Şebeke akışı teorisinin ilk çalışmalarının temel fikri, bazı fiziksel sistemlerin matematiksel nesnelerle ifade edilebilmesidir. İlk dönemlerde daha çok "verilen bir şebekede voltaj uygulanırsa sonuçtaki akış ne olur?" sorusunun cevapları aranmaktaydı [1, 2].

Biz burada "akış göndermek için kullanılabilecek alternatif yolları olan bir şebekede en etkin yol hangisidir?" sorusunun cevabıyla ilgileneceğiz. Bu sorunun cevabıyla 1940'ların sonu 1950'lerin başlarında araştırmacıların optimizasyonu bağımsız bir alan olarak geliştirmesiyle başlamıştır [1- 6].

Uygun çözümler kümesinden bir çözüm seçmek en iyi çözümdür, denilebilir. Fakat bir çok uygulanabilir problem için uygun çözümler kümesinin eleman sayısı çok büyük olduğu durumlarla da karşılaşılabilir. Uygulamada karşılaşılan bir çok problemi çözmek için gerekli olan algoritmaların seçiminde mümkün olan en az işlem sayısıyla çalışan

algoritmalar tercih edilmelidir. O halde bu algoritmalar şebeke büyüdükçe işlem sayısının çok hızlı artmasını engelleyen özelliklere sahip algoritmalar olmalıdır.

Şebeke akışı modelleri ile ilgili çalışmalara lineer programlama geliştirilmeden önce başlamıştır. Bu alanda ilk çalışmalar, 1930 da Tolstoĭ[7], 1939 da Kantorovic [8], 1941 de Hitchcock [9] ve 1947 de Koopmans [10] çalışmaları minimum fiyat problemlerinin bir özel hali olan taşıma problemleri ile ilgilidir. Şebeke akışı problemlerine ilgi Dantzing'in 1947 de geliştirdiği simpleks metodu ile artmış ve ayrıca 1951 de Dantzing [11] taşıma problemi için de simpleks metodunu daha ileri götürerek uygulamıştır.

1950'li yıllarda araştırmacılar en kısa yol problemi, maksimum akış problemi ve atama problemlerinin gerçek dünya problemlerine uygulamalarındaki önem nedeniyle bu problemlerle daha fazla ilgilenmeye başlamışlardır. Bu çalışmaların öncüleri Dantzing, Ford ve Fulkerson' dur[12, 13].

Bu tez çalışmasında şebeke akışı problemlerinde önemli bir yere sahip olan maksimum akışı problemi ile ilgileneceğiz.

Genel kısımlar bölümünde şebekeler ve maksimum akış problemi hakkında önce genel bilgiler ve tanımlar ve daha sonra maksimum akış probleminin bilinen en etkin algoritmaları verilmektedir. Bu bölümde Ford ve Fulkerson [1] "Flows in Network", Ahuja ve diğ. [2], "Network Flows" ve Ahlatçıoğlu ve Tiryaki [14] "Kantitatif Karar Verme Teknikleri" kitaplarından yararlanılmıştır.

Bulgular bölümünde ise, şebekeler düğüm - düğüm matris gösterimi ile ifade edilecek, Genel Kısımlar bölümünde verilmiş olan algoritmaların Bulgular bölümde matris gösterimi yardımıyla nasıl kullanılabileceği ayrıntılı olarak verilecektir. Ayrıca bu tez çalışması kapsamında maksimum akış problemi için geliştirilen yeni bir algoritma da verilmiştir. Geliştirilen algoritma, kapasite ölçekleme algoritması ve en büyük etiketli akış öncesi gönderme algoritmasının C bilgisayar programlama dilinde kodları yazılarak karşılaştırmaları da yapılmıştır. Bu tez çalışmasında geliştirilen algoritma bağımsız bir bilgisayar şebekesinde denenmiş, büyük hacimli görüntü dosyalarının taşınması, saklanması ve geri çağırımı süreçleri için kullanılmıştır.

2. GENEL KISIMLAR

2.1. TANIMLAR

Şebeke: Düğümler kümesi ve bu düğümleri birleştiren arkların bir dizilişidir. N düğümler kümesi ile elemanları farklı düğümler olan ikililerden oluşan A arklar kümesinin oluşturduğu şebeke $G = (N ; A)$ ile gösterilir. $G = (N ; A)$ şebekesinin düğüm sayısı N nin elaman sayısıdır, ark sayısı ise A nın eleman sayısıdır.

Düğüm : Şebekedeki arz, talep ve aktarma merkezleridir. Bir şebekede kaynak düğümü, aktarma düğümü ve kuyu düğümü olmak üzere üç ayrı düğüm tipi vardır.

Ark: Şebekede herhangi farklı iki düğüm arasındaki yola ark denir. Gidiş yönü belli olan arklara yönlü arklar, belirsiz olanlara yönsüz arklar denir. Bir yönsüz ark iki yönlü ark çifti olarak düşünülebilir. Şebekede yönlü arklar oklar ile yönsüz arklar doğru parçaları ile gösterilir. i düğümünden j düğümüne giden ark (i, j) ile gösterilir. Ark yönsüz ise $(i, j) = (j, i)$ alınabilir.

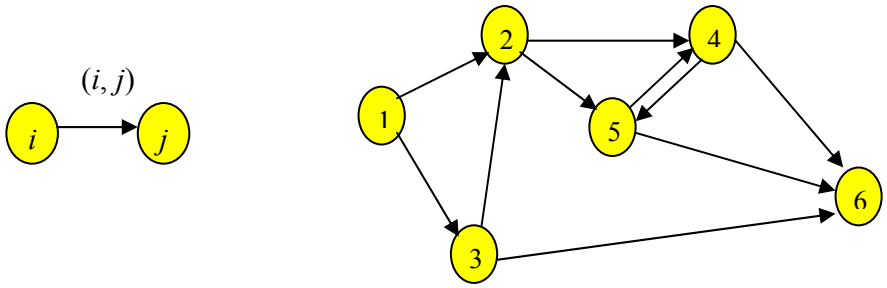
Sonlu Şebeke: Ark sayısı ve düğüm sayısı sonlu olan şebekeye sonlu şebeke denir.

Bu çalışmada düğüm sayısı n, ark sayısı m ile gösterilen sonlu şebekeler ele alınacaktır. Yukarıda verilen tanımlara göre sonlu N düğümler kümesi $N = \{1, 2, ..., n\} \subset \mathbb{N}$ ve A arklar kümesi $A = \{(i, j) : i, j \in N \text{ ve } i \neq j\} \subset N \times N$ olarak da tanımlanabilir.

Bağlantılı Şebeke: Bir şebekede, S düğümlerin boştan farklı bir alt kümesi ve bunun şebekedeki düğümlere göre tümleyeni $\overline{S}$ olsun. $i \in S$ ve $j \in \overline{S}$ olacak şekilde en az bir (i, j) ya da (j, i) arkı varsa "şebeke bağlantılıdır" denir

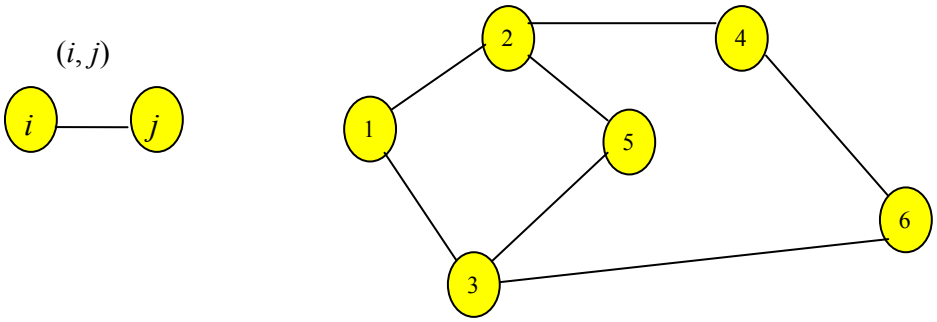
Yönlü Şebeke: $G = (N ; A)$ şebekesinde N düğümler kümesi ve A arklar kümesinin elemanları farklı düğümler olan sıralı ikililerden oluşuyorsa $G = (N ; A)$ şebekesine yönlü şebeke denir. (i, j) bir yönlü arkında i düğümüne başlangıç düğümü ve j

düğümüne bitiş düğümü denir. Örneğin Şekil 2.1 de verilen yönlü şebeke $N=\{1,2,3,4,5,6\}$ ve $A=\{(1,2),(1,3),(2,4),(2,5),(3,2),(3,6),(4,5),(4,6),(5,6)\}$ dir.



Şekil 2.1 Yönlü şebeke

Yönsüz Şebeke: $G = (N ; A)$ şebekesinde N düğümler kümesi ve A arklar kümesinin elemanları farklı düğümler olan sırasız ikililerden oluşuyorsa $G = (N ; A)$ şebekesine yönsüz şebeke denir. i ve j düğümleri arasında bir yönsüz ark varsa bu ark (i, j) ya da (j, i) ile gösterilebilir. Bir yönsüz ark iki yönlü ark çifti olarak da düşünülebilir. Şebekede yönsüz arklar doğru parçaları ile gösterilir. Örneğin Şekil 2.2 de verilen yönsüz şebekede $N=\{1,2,3,4,5,6\}$ ve $A=\{(1,2),(1,3),(2,4),(2,5),(3,5),(3,6),(4,6)\}$ dir.



Şekil 2.2. Yönsüz şebeke

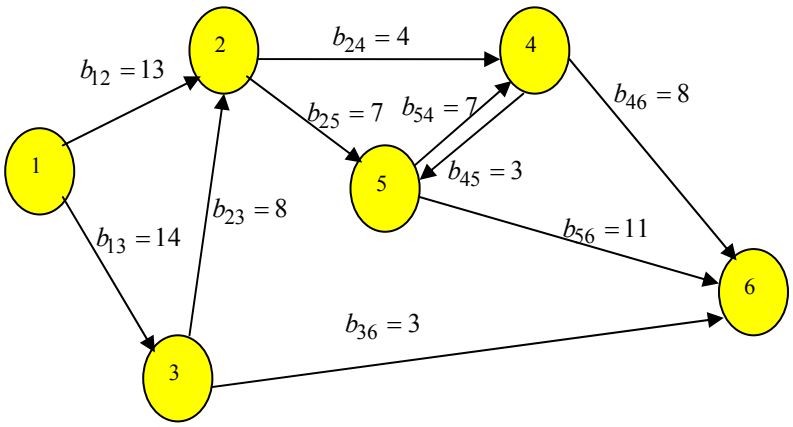
Alt şebeke: $G = (N, A)$ şebekesinde eğer $N' \subseteq N$ ve $A' \subseteq A$ olacak şekilde N', A' kümeleri varsa bu kümelerin oluşturduğu $G'=(N', A')$ şebekesine $G = (N, A)$ şebekesinin bir alt şebekesi denir.

Zincir: İki özel düğümü birleştiren düğüm ve arkların dizilişine zincir denir. Örneğin $1,(1,2),2,\ ...,\ (k-1,k),\ k$ düğüm ve arkların dizilişi, 1 düğümü ile k düğümünü birleştiren bir zincirdir.

Devre: Eğer zincirde başlangıç düğümü ile bitiş düğümü aynı ise bu zincire devre denir. Örneğin $1,(1,2),\ 2,\ ...,\ (k-1,k),\ k$ zincirinde $1 = k$ ise bu zincir devre olur. Hiç bir devre içermeyen zincire basit zincir denir.

Yol: Üzerindeki bütün arkları yönlü olan zincirlere yol denir. Her yol zincirdir, fakat tersi doğru değildir.

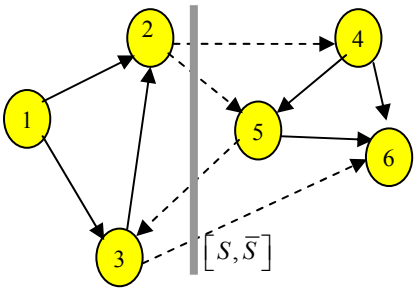
Kapasite: Bir şebekede her $(i,\ j)$ arkından birim zamanda yapılabilecek maksimum akışa kapasite denir ve b_{ij} ile gösterilir. Şekil 2.3 te kapasiteli bir şebeke örneği verilmiştir. Bu örnekteki boş kapasiteler arkların üzerinde gösterilmiştir.



Şekil 2.3 Kapasiteli şebeke

Kesit: Bir kesit N düğüm kümesini S ve $\overline{S} = N - S$ olarak iki ayrık kümeye ayırır. Her kesit başlangıç düğümü S kümesinde bitiş düğümü $\overline{S}$ kümesinde olan arklardan oluşan bir küme tanımlar. Bu küme $\left[S, \overline{S}\right]$ ile gösterilir.

***s – t* kesiti.** Bir $G = (N;\ A)$ şebekesinde verilen $\left[S, \overline{S}\right]$ kesitinde s kaynak düğümü S kümesinde, t kuyu düğümü $\overline{S}$ kümesinde olan kesitlere $s - t$ kesiti denir. Örneğin Şekil 2.4 te verilen şebekede 1 düğümü kaynak düğümü, 6 düğümü kuyu düğümü, $S = \{1,\ 2,\ 3\}$ ve $\overline{S} = \{4,5,6\}$ olmak üzere $\left[S, \overline{S}\right]$ bir $s - t$ kesitidir. Bu kesitteki arklar $[S,\ \overline{S}] = \{(2,4),(2,5),(3,6)\}$ dır.



Şekil 2.4. Bir *s*-*t* kesiti örneği

Bir kesitin kapasitesi: $[S,\overline{S}]$ kesitinin kapasitesi S kümesinden $\overline{S}$ kümesine giden arkların kapasitelerinin toplamıdır ve $C[S,\overline{S}]$ ile gösterilir. Örneğin Şekil 2.4 te verilen kesitin kapasitesi $C[S,\overline{S}] = b_{24} + b_{25} + b_{36}$ dır.

2.2 ŞEBEKENİN DÜĞÜM - ARK MATRİS GÖSTERİMİ

$G = (N, A)$ şebekesi her düğüm için bir satır her ark için bir sütun olmak üzere n satır m sütunlu bir matris şeklinde gösterilebilir. (i, j) arkını gösteren sütunda sıfırdan farklı sadece iki eleman vardır: Bunlar da i düğümünü gösteren satırda +1, j düğümünü gösteren satırda -1 elemanlarıdır. Bu gösterime düğüm-ark matris gösterimi denir. Örneğin Şekil 2.3 te verilen şebeke düğüm-ark matris gösterimi olarak Şekil 2.5 de verilmektedir.

$$
\begin{array}{c}
 & (1,2) & (1,3) & (2,4) & (2,5) & (3,2) & (3,6) & (4,5) & (4,6) & (5,4) & (5,6) \\
1 & \left[\begin{array}{cccccccccc} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right. \\
2 & \begin{array}{cccccccccc} -1 & 0 & 1 & 1 & -1 & 0 & 0 & 0 & 0 & 0 \end{array} \\
3 & \begin{array}{cccccccccc} 0 & -1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \end{array} \\
4 & \begin{array}{cccccccccc} 0 & 0 & -1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \end{array} \\
5 & \begin{array}{cccccccccc} 0 & 0 & 0 & -1 & 0 & 0 & -1 & 0 & -1 & 1 \end{array} \\
6 & \left.\begin{array}{cccccccccc} 0 & 0 & 0 & 0 & 0 & -1 & 0 & -1 & 0 & -1 \end{array}\right]
\end{array}
$$

Şekil 2.5 Şekil 2.3 te verilen şebekenin düğüm-ark matris gösterimi ile gösterimi

2.3 ŞEBEKENİN DÜĞÜM - DÜĞÜM MATRİS GÖSTERİMİ

Düğüm - düğüm matris gösteriminde şebeke n x n matrisi ile iki farklı şekilde gösterilebilir. Birinci gösterimde matrisin birinci satır ve birinci sütunu düğümleri,

matrisin diğer (i, j) elemanlarına (i, j) arklarının kapasiteleri yazılır. Eğer şebekede $i \neq j$ olmak üzere (i, j) arkı yoksa matrisin (i, j) elemanına 0 yazılır. Matrisin (i, i) elemanı boş bırakılır. Örneğin Şekil 2.3 te verilen şebeke düğüm-düğüm matris birinci gösterimi olarak Şekil 2.6 da verilmektedir.

Düğüm	**1**	**2**	**3**	**4**	**5**	**6**
1	---	13	14	0	0	0
2	0	---	0	4	7	0
3	0	8	---	0	0	5
4	0	0	0	---	3	8
5	0	0	0	7	---	11
6	0	0	0	0	0	---

Şekil 2.6 Şekil 2.3 te verilen şebekenin düğüm - düğüm matris birinci gösterimi ile gösterimi

Bu tez çalışmasının Bulgular kısmında şebekeler bu birinci gösterimi ile gösterilecektir.

İkinci düğüm-düğüm matris gösteriminde matriste her bir satır ve sütunda düğümler gösterilir. Bu gösterimde şebekede (i, j) arkı varsa matrisin (i, j) elamanı için 1 yazılır, (i, j) arkı yoksa matrisin (i, j) elamanı için 0 yazılır. Örneğin Şekil 2.3 te verilen şebekenin düğüm - düğüm matris ikinci gösterimi olarak Şekil 2.7 de verilmektedir.

$$\begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{array} \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \begin{bmatrix} 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \end{array}$$

Şekil 2.7 Şekil 2.3 te verilen şebekenin düğüm - düğüm matris ikinci gösterimi ile gösterimi

2.4 MAKSIMUM AKIŞ PROBLEMİ

"Verilen bir şebekede ark kapasitelerini aşmadan iki özel düğüm olan s kaynak düğümü ile t kuyu düğümü arasında mümkün olan en fazla akış nasıl gönderilebilir?" problemine maksimum akış problemi denir.

$G = (N, A)$ şebekesinde (i, j) arkının boş kapasitesi $b_{ij} \geq 0$ olsun. Bu şebekede s kaynak düğümü ve t kuyu düğümü olsun. s kaynak düğümünden t kuyu düğümüne şebekedeki ark kapasitelerini aşmadan gönderilebilecek maksimum akışı bulmak için aşağıdaki notasyon kullanılabilir. (i, j) arkında yapılabilecek akış miktarı x_{ij} değişkenleri ile gösterilirse,

$$\sum_i x_{ij} - \sum_k x_{jk} = \begin{cases} -v & , \quad j = s \text{ ise} \\ 0 & , \quad j \neq s, t \text{ ise} \\ v & , \quad j = t \text{ ise} \end{cases} \tag{2.1}$$

$$0 \leq x_{ij} \leq b_{ij}, \qquad \forall\ (i, j) \in A \tag{2.2}$$

özellikleri yazılabilir. Burada v akış miktarıdır. (2.1) denklemi akışın düğümlerdeki korunumunu verir. Yani kaynak düğümüne hiç giriş olmadığı halde v miktar akış çıkar ($j = s$). Ara düğümlere giren akış çıkan akışa eşittir ($j \neq s, t$). Kuyu düğümüne v miktar akış ulaşır (kuyu düğümünden çıkış yoktur ($j = t$)). (2.2) ise arklarda yapılabilecek akışın, ark kapasitesi ile sınırlı olması gerektiğini vurgular. Şebekede yapılabilecek maksimum akış:

$$v = \sum_j x_{sj} \tag{2.3}$$

amaç fonksiyonunu (2.1) - (2.2) kısıtları altında maksimize eden probleminin çözümü ile elde edilir. Burada v ye akışın değeri denir.

Maksimum akış problemi için geliştirilen algoritmalar iki türlüdür:

1. *Geliştirilmiş yol* (*augmenting path*) *algoritmaları*: Bu algoritmalar, kaynak düğümünden kuyu düğümüne doğru akış yolu boyunca akışı aşamalı olarak artırırlar. Bu algoritmalar kaynak düğümü ve kuyu düğümü hariç şebekedeki düğümlerin denge kısıtlarını korurlar.

2. *Akış öncesi gönderme* (*preflow-push*) *algoritmaları*: Bu algoritmalarda bazı düğümler fazlalıklara (excess) sahip olurlar, ya da akış toplarlar. Bu algoritmalar fazla akışa sahip

olan düğümlerden kuyu düğümüne ileriye doğru ya da kaynak düğümüne geriye doğru akış göndererek akışı aşamalı olarak rahatlatırlar.

Burada maksimum akış problemi aşağıdaki kabuller altında kurulacaktır:

Kabul 2.1. Şebekedeki akış yönlüdür, yönsüz arklar çift yönlü olarak alınır.

Bir şebekede (i, j) arkı yönsüz ve kapasitesi b_{ij} olsun. Bu durumda bu şebekede (i, j) ve (j, i) arkları var kabul edilir. (i, j) arkından yapılan akış x_{ij} ve (j, i) arkından yapılan akış x_{ji} ise $x_{ij} + x_{ji} \leq b_{ij}$ eşitsizliği sağlanır.

Kabul 2.2. Arklardaki bütün kapasiteler negatif olmayan tam sayılardır.

Kapasiteler rasyonel sayı olursa bu kapasiteleri yeteri kadar büyük bir tam sayı ile çarparak kapasiteler tam sayılara çevrilebilir. Ayrıca bilgisayar yardımıyla yapılan hesaplamalarda rasyonel sayılar kullanılabildiğinden kapasiteler rasyonel sayı da olabilir.

Kabul 2.3. Herhangi bir (i, j) arkı A ya ait ise (j, i) arkı da A ya aittir.

Eğer şebekede tek yönlü (i, j) arkı varsa (j, i) arkı kapasitesi 0 olan bir ark olarak şebekeye alınır.

Kabul 2.4. Şebekede aynı paralel arklar yoktur. Yani başlangıç düğümü i bitiş düğümü j olan birden fazla ark yoktur.

Şebekede paralel ark olması genelliği bozmaz. Başlangıç ve bitiş düğümleri aynı olan birden fazla ark olursa bunları tek bir ark olarak ve bu arkların kapasitelerinin toplamı yeni kapasite olarak alınabilir.

2.5 AKIŞ VE KESİT

2.5.1 Tanımlar

Doymuş ark. Bir şebekede yapılan bir *x* akışında; eğer (i, j) arkında yapılan akış x_{ij} ve arkın kapasitesi b_{ij} olmak üzere $x_{ij} = b_{ij}$ ise bu arka doymuş ark denir.

Doymamış ark. Bir şebekede yapılan bir *x* akışında; eğer (i, j) arkında yapılan akış x_{ij} ve arkın kapasitesi b_{ij} olmak üzere $x_{ij} < b_{ij}$ ise bu arka doymamış ark denir.

Arta kalan şebeke (residual network). Maksimum akış probleminde Kabul 2.4 ten dolayı şebekede (i, j) arkı varsa (j, i) arkı da vardır. Bu durumda verilen bir *x* akışında, arta kalan kapasitesi (residual capacity) r_{ij} olan her $(i, j) \in A$ arkı için (i, j) arkı ya da (j, i) arkı kullanılarak *i* düğümünden *j* düğümüne maksimum akış artırımı yapılabilir. r_{ij} arta kalan boş kapasitesinin iki bileşeni vardır: **(1)** (i, j) arkında kullanılmayan kapasite $b_{ij} - x_{ij}$, **(2)** (j, i) arkında yapılan akış x_{ji}. Böylece $r_{ij} = (b_{ij} - x_{ij}) + x_{ji}$ olur. *G* şebekesinde yapılan *x* akışından sonra arta kalan boş kapasitelerin oluşturduğu şebeke $G(x)$ ile gösterilir.

***s – t* Kesitinin kapasitesi.** $[S, \overline{S}]$ bir $s - t$ kesiti olmak üzere, $[S, \overline{S}]$ kesitinin kapasitesi

$$C[S, \overline{S}] = \sum_{(i,j) \in (S, \overline{S})} b_{ij} \tag{2.4}$$

olarak tanımlanır. $[S, \overline{S}]$ kesitinin kapasitesi *S* deki düğümlerden $\overline{S}$ deki düğümlere gönderilebilecek maksimum akışın bir üst sınırıdır.

Arta kalan şebekedeki *s – t* kesitinin kapasitesi. $[S, \overline{S}]$ arta kalan şebekedeki bir $s - t$ olmak üzere, $[S, \overline{S}]$ kesitinin kapasitesi

$$r\left[S,\ \overline{S}\right]=\sum_{(i,j)\in\left(S,\overline{S}\right)} r_{ij}$$

olarak tanımlanır.

Minimum kesit. Bütün s – t kesitlerinin içinden kapasitesi minimum olan kesite minimum kesit denir.

2.5.2 Özellikler.

G = (N, A) şebekesinde x bir akış ve $\left[S,\ \overline{S}\right]$ bir s - t kesiti olsun. (2.1) kısıtını S kümesindeki düğümler için uygularsak

$$v=\sum_{i\in S}\left[\sum_{\{j:(i,j)\in A\}} x_{ij}-\sum_{\{j:(j,i)\in A\}} x_{ji}\right]$$

elde edilir. Burada eğer p ve q düğümleri S kümesine ait ve $(p,q)\in A$ ise Kabul 2.4 ten dolayı $(q,p)\in A$ dır. Toplam alınırken $i = p$ ve $j = q$ için $x_{pq}-x_{qp}$ terimi, $i = q$ ve $j = p$ için $x_{qp}-x_{pq}$ terimi gelir ve bu terimlerin toplamından 0 elde edilir. p ve q düğümleri $\overline{S}$ kümesine ait ise x_{pq} toplamın içinde olmayacaktır. Yani başlangıç ve bitiş düğümleri sadece S kümesinde ya da sadece $\overline{S}$ kümesinde olan arklar bu toplama etki etmezler. Böylece yukarıdaki kısıt

$$v=\sum_{(i,j)\in\left[S,\overline{S}\right]} x_{ij}\ -\sum_{(i,j)\in\left[\overline{S},S\right]} x_{ij} \tag{2.5}$$

şeklinde yazılabilir. (2.5) denkleminin sağ tarafı $\left[S,\ \overline{S}\right]$ kesitinin toplam akışını verir. $x_{ij}\le b_{ij}$ ve $0\le x_{ij}$ olduğundan (2.5) denkleminden

$$v\le\sum_{(i,j)\in\left[S,\overline{S}\right]} b_{ij}=C\left[S,\overline{S}\right] \tag{2.6}$$

elde edilir. Böylece aşağıdaki özellik elde edilir.

Özellik 2.1. Herhangi bir akışın değeri herhangi bir *s* - *t* kesitinin kapasitesini aşamaz.

Bu özellikten eğer bir *x* akışının değeri bir $\left[S,\overline{S}\right]$ kesitinin kapasitesine eşit olursa *x* akışı bir maksimum akış, $\left[S,\overline{S}\right]$ kesiti bir minimum kesit olur.

Özellik 2.2. Bir şebekede değeri *v* olan herhangi bir *x* akışı için *s* kaynak düğümünden *t* kuyu düğümüne yapılabilecek ek akış arta kalan şebekedeki *s*–*t* kesitinin kapasitesinden küçük eşittir.

İspat. $\left[S,\overline{S}\right]$ bir *s* - *t* kesiti ve *x* akışının değeri *v* olsun. x' de bir akış ve değeri $v+\Delta v$, $\Delta v \geq 0$ olsun. (2.6) dan

$$v+\Delta v \leq \sum_{(i,j)\in\left[S,\overline{S}\right]} b_{ij} \tag{2.7}$$

dir. (2.5) ve (2.7) den

$$\Delta v \leq \sum_{(i,j)\in\left[S,\overline{S}\right]} \left(b_{ij}-x_{ij}\right) + \sum_{(i,j)\in\left[\overline{S},S\right]} x_{ij} \tag{2.8}$$

elde edilir. Kabul 2.4 ten $\sum_{(i,j)\in\left[\overline{S},S\right]} x_{ij}$ yerine $\sum_{(i,j)\in\left[S,\overline{S}\right]} x_{ji}$ yazılabilir ve

$$\Delta v \leq \sum_{(i,j)\in\left[S,\overline{S}\right]} \left(b_{ij}-x_{ij}+x_{ji}\right) = \sum_{(i,j)\in\left[S,\overline{S}\right]} r_{ij}$$

elde edilir. Böylece özellik elde edilir.

2.6 KARMAŞILIK ANALİZİ VE BİR NOTASYON

$g(x)$ fonksiyonu bütün pozitif reel sayılar kümesinde tanımlanmış ve pozitif bir fonksiyon olsun. S pozitif reel sayılar kümesinin sınırsız bir alt kümesi ve $f(x)$ fonksiyonu S üzerinde tanımlı herhangi bir fonksiyon olmak üzere yeteri kadar büyük her $x \in S$ için

$$\frac{|f(x)|}{g(x)} \leq M$$

olacak şekilde pozitif M sayısı varsa $f(x) = O(g(x))$ ya da $f(x) \in O(g(x))$ yazılır. Bu sembole büyük O sembolü, ya da Landau (Edmund Landau 1877- 1938) sembolü denir. Örneğin $\sin(x) = O(1)$, $x^3 - x^2 + 5 = O\left(x^3\right)$ dir.

Özellik 2.3. Büyük O sembolünde aşağıdaki özellikler geçerlidir.

1. $f(x) = O(g(x))$ ve $h(x) = O(f(x))$ ise $h(x) = O(g(x))$

2. $O\big(f(x)\big) \pm O\big(g(x)\big) = O\big(\text{maks}\big(f(x), g(x)\big)\big)$

3. $f_1(x) = O\big(f(x)\big)$, $g_1(x) = O\big(g(x)\big)$ ise $f_1(x) + g_1(x) = O\big(f(x) + g(x)\big)$ ve $f_1(x)g_1(x) = O\big(f(x)g(x)\big)$ dir.

İspat. 1. $f(x) = O(g(x))$ ise $0 < f(x) \leq M_1 g(x)$ ve $h(x) = O(f(x))$ ise $0 < |h(x)| \leq M_2 f(x)$ olacak şekilde $M_1 > 0$ ve $M_2 > 0$ sayıları vardır. Burada $0 < |h(x)| \leq M_1 M_2 g(x)$ elde edilir. Yani $h(x) = O(g(x))$ dir.

2. $f_1(x) = O(f(x))$ ise $0 < |f_1(x)| \leq M_1 f(x)$ ve $g_1(x) = O(g(x))$ ise $0 < |g_1(x)| \leq M_2 g(x)$ olacak şekilde $M_1 > 0$ ve $M_2 > 0$ sayıları vardır. Buradan $M \geq \text{maks}\big(M_1, M_2\big)$ olmak üzere

$$0 < |f_1(x)| + |g_1(x)| \leq M_1 g(x) + M_2 f(x) \leq M \text{maks}\big(g(x)\,, f(x)\big)$$

elde edilir, yani $O\big(f(x)\big)+O\big(g(x)\big)=O\big(\text{maks}\big(f(x),g(x)\big)\big)$ dir.

3. $f_1(x)=O(f(x))$ ise $0<\big|f_1(x)\big|\le M_1 f(x)$ ve $g_1(x)=O(g(x))$ ise $0<\big|g_1(x)\big|\le M_2 g(x)$ olacak şekilde $M_1>0$ ve $M_2>0$ sayıları vardır. Buradan $M\ge \text{maks}\big(M_1,M_2\big)$ olmak üzere

$$0<\big|f_1(x)\big|+\big|g_1(x)\big|\le M_1 g(x)+M_2 f(x)\le M\big(g(x)+f(x)\big)$$

elde edilir. Yani $f_1(x)+g_1(x)=O\big(f(x)+g(x)\big)$ dir. Ayrıca

$$0<\big|f_1(x)\big|\,\big|g_1(x)\big|\le M_1 g(x)\;M_2 f(x)=M_1M_2\big(g(x)\,f(x)\big)$$

elde edilir. Yani $f_1(x)g_1(x)=O\big(f(x)g(x)\big)$ dir.

Bir algoritmanın işlem sayısı c, n_0 sabit sayıları ve her $n\ge n_0$ için $cf(n)$ ise bu algoritmanın karmaşıklığı $O\big(f(n)\big)$dir denir. Bir algoritmanın karmaşıklığı n nin yeteri kadar büyük değeri için bir üst sınırıdır. Örneğin bir algoritmanın işlem sayısı $100n+n^2+0,0001n^3$ ise $n>10000$ ve Özellik 2.3 (2) den bu algoritmanın karmaşıklığı $O(n^3)$ tür.

Bir algoritmanın karmaşıklığı algoritmanın parametrelerinin bir polinomu ile sınırlı ise bu algoritmaya iyi denir. Bu tür algoritmalara polinomyal algoritmalar denir. Maksimum akış probleminin parametreleri n, m, $\log_2 U$ dur. Burada n düğüm sayısı, m ark sayısı ve U en büyük boş kapasiteye sahip arkın kapasitesidir. Maksimum akış problemi için karmaşıklığı bu parametrelerinin bir polinomu şeklinde olan algoritmalar iyi algoritmalardır. Örneğin maksimum akış problemi için karmaşıklığı $O(n^2)$, $O(nm)$, $O(nm+n^2\log_2 U)$ olan algoritmalar polinomyal algoritmalardır.

Bir algoritmanın karmaşıklığı algoritmanın parametrelerinin bir polinomu şeklinde olamayan bir fonksiyon ile sınırlı olan algoritmalara üstel algoritmalar denir. Örneğin karmaşıklığı $O(2^n)$, $O(n^{\log_2 n})$, $O(n!)$ olan algoritmalar üstel algoritmalardır.

Polinomyal algoritmalar üstel algoritmalarından üstün olduğundan polinomyal algoritmalar üstel algoritmalarına tercih edilir. Örneğin $10^6 \le n$ için n^{1000} 2^n den, $2^{100000} < n$ için n^{10000} $n^{0,1 \log_2 n}$ den daha küçüktür olduğundan karmaşıklı $O(2^n)$ olan algoritmalar yerine karmaşıklı $O(n^2)$ olan algoritmalar, karmaşıklı $O\left(n^{0,1 \log_2 n}\right)$ olan algoritmalar yerine karmaşıklı $O(n^{10000})$ algoritmalar tercih edilir.

2.7 MAKSİMUM AKIŞ PROBLEMLERİNİN UYGULAMALARI

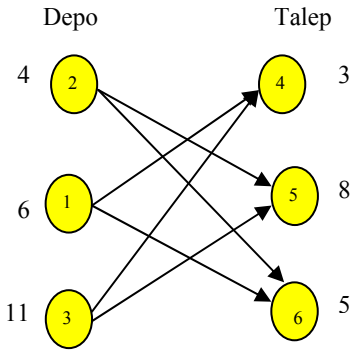
2.7.1. Uygun (Feasible) Akış Problemi

Uygun akış problemi $G = (N, A)$ şebekesinde

$$\sum_{\{j:(i,j)\in A\}} x_{ij} - \sum_{\{j:(j,i)\in A\}} x_{ji} = u(i) \quad \forall i \in N \tag{2.9a}$$

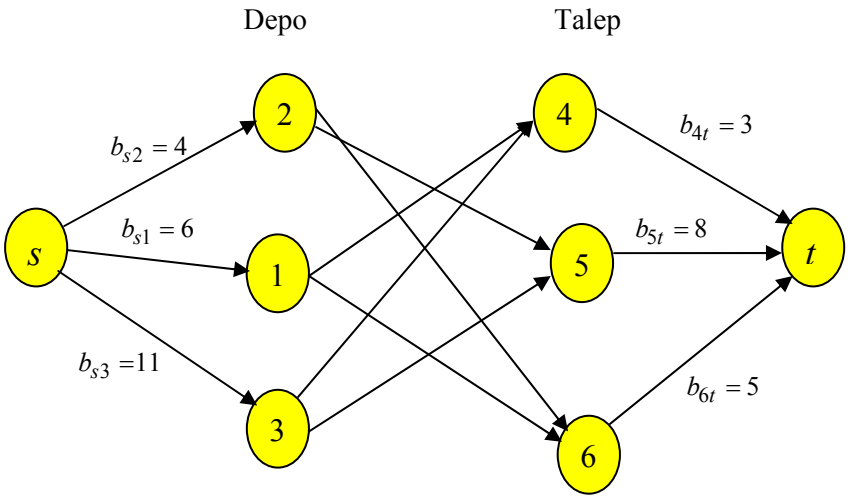
$$0 \le x_{ij} \le b_{ij} \qquad \forall (i,j) \in A \tag{2.9b}$$

kısıtlarını sağlayan bir x akışının bulunmasıdır. Burada $\sum_{i\in N} u(i) = 0$ kabul edilebilir. Bu problem için uygulamada şu örnek verilebilir: Bazı limanlarda var olan ticari ürünlere başka limanlardan talep edilmektedir. Ticari ürünlerin limanlarda stoklandığı ve bu ürünlerden maksimum miktarda deniz yoluyla taşındığını bilinmektedir. Şekil 2.8 de üç depo ve üç talep merkezinin olduğu uygulanabilir akış probleminin bir örneği verilmektedir. Depoların başlarında yazılan sayılar ile depoların sahip oldukları mal miktarı gösterilmektedir, talep düğümlerinin başında ise talep edilen mal miktarı verilmektedir.



Şekil 2.8 Üç depo ve üç talep eden olan uygulanabilir akış problemi örneği

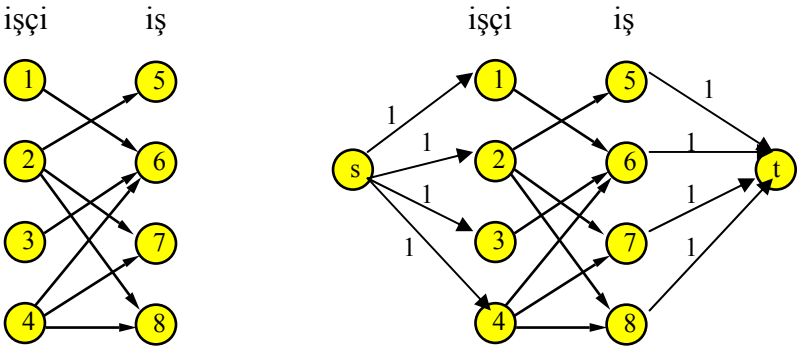
Bu problem yeni bir şebeke tanımlanarak maksimum akış problemi olarak çözülebilir. Bunun için iki yeni düğüm *s* kaynak düğümü ve *t* kuyu düğümü tanımlanır. Her *i* düğümü için $u(i) > 0$ ise kapasitesi $u(i)$ olan (s, i) arkı, $u(i) < 0$ ise kapasitesi $-u(i)$ olan (i, t) arkı eklenir. Sonra bu yeni şebekede *s* kaynak düğümünden *t* kuyu düğümüne gönderilebilecek maksimum akış problemi çözülür. Eğer bulunan maksimum akış kaynak düğümünden çıkan bütün arkları ve kuyu düğümüne gelen bütün arkları doyurursa (2.9) probleminin uygun bir çözümü vardır, aksi halde çözüm yoktur denir. Şekil 2.8 de verilen örneğe bu çözüm uygulanırsa *s* kaynak düğümü ve *t* kuyu düğümü eklenerek Şekil 2.9 da verilen şebeke elde edilir. Depolardaki ürün miktarları kaynak düğümü ile depolar arasındaki arkların kapasitesi, benzer şekilde taleplerdeki istek miktarları talepler ile kuyu düğümü arasındaki arkların kapasitesi olarak alınır. Daha sonra şebeke maksimum akış problemi olarak çözülür.



Şekil 2.9 Şekil 2.8 de verilen örneğin maksimum akış problemine dönüştürülmesi

2.7.2. Atama Problemi

Eleman sayısı eşit olan iki küme arasında bir eşleştirme verilmiş olsun. Bu eşleştirmeden bire-bir eşleştirme elde edilmesi problemine atama problemi denir. Örneğin bir küme n elemanlı işçi diğer küme n elemanlı iş kümesi olsun. Bu iki küme arasında verilen bir eşleşmeden her bir kişinin sadece bir işi, her bir işi sadece bir kişinin yaptığı bir eşleştirme bulunması problemi bir atama problemidir. $B \rightarrow x$ linki B işçisinin x işini yapabildiği anlamında kullanılırsa bu problem bir yönlü şebeke olarak kurulabilir. Bu problemi çözmek için şebekeye bir s kaynak düğümü ve bir t kuyu düğümü eklenir. Kaynak düğümü işçileri (ya da işleri) gösteren düğümlerle kuyu düğümü işleri (ya da işçileri) gösteren düğümlerle kapasiteleri 1 olan arklarla eşleştirilir. Bu şebekede s kaynak düğümünden t kuyu düğümüne yapılabilecek maksimum akış n ise atama probleminin çözümü vardır aksi durumda çözüm yoktur. Örneğin Şekil 2.10 (a) da dört işçi ile dört işin olduğu bir şebeke, Şekil 2.10 (b) de şebekeye yeni kaynak düğümü ve kuyu düğümünün eklenmesiyle oluşan şebeke verilmiştir.



Şekil 2.10. **(a)**Atama probleminin iş-işçi örneği, **(b)** (a) da verilen şebekeye yeni kaynak ve kuyu düğümlerinin eklenmesi

2.7.3 İki İşlemcili Bir Bilgisayarda Dağıtımlı Hesaplama

Bu uygulama iki işlemcili bilgisayar için bir programın modüllerinin (alt programlarının) işlemciler arası iletişim ve hesaplamanın minimum maliyetli bir yol bulunması ile ilgilidir. Birbirine denk olmak zorunda olmayan iki işlemcili bir bilgisayar sisteminde büyük bir program çalıştırılmak istenmektedir. Bu programın her

bir modülünün çalıştırmanın maliyeti önceden bilinir ve bir işlemciden diğerine göre değişik olabilir. Çünkü işlemcilerin bellek, kontrol, hız ve aritmetik kapasiteleri farklı olabilir. i. modülün 1. işlemcide hesaplama maliyeti α_i ve 2. işlemcide hesaplama maliyeti β_i olsun. İşlemciler arası iletişimden dolayı farklı modüllerin farklı işlemcilere atamak yüksek maliyetli olmaktadır. *i.* ve *j.* modülleri farklı modüle atanması halinde işlemciler arasındaki iletişimin maliyeti c_{ij} olsun. Eğer *i.* ve *j.* modülleri aynı işlemciye atanırsa böyle bir maliyet yoktur. Programın modüllerini farklı işlemcilere paylaştırarak toplam maliyeti ve işlemciler arasındaki iletişim maliyetini minimum olması amaçlanmaktadır.

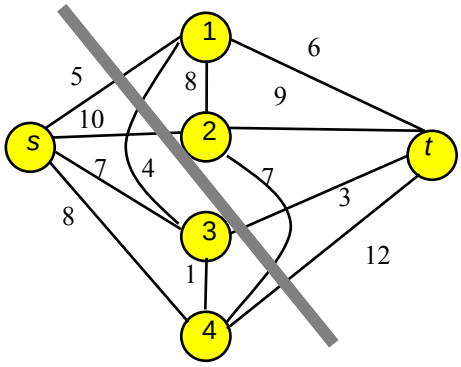
Bu problem *s* kaynak düğümü bir işlemci *t* kuyu düğümü diğer işlemciyi ve her bir modül bir düğüm olarak bir yönsüz şebeke şeklinde kurulabilir. (*s*, *i*) arkının kapasitesi α_i, (*i*, *t*) arkının kapasitesi β_i dir. Programın çalışırken modül *i* ile modül *j* birbirlerini etkilerlerse kapasitesi c_{ij} olan (*i*, *j*) arkı şebekeye eklenir. Birinci işlemciye atanan modüller A_1 ikinci işlemciye atanan modüller A_2 olsun. Bu atamanın toplam maliyeti $\sum_{i\in A_1}\alpha_i+\sum_{i\in A_2}\beta_i+\sum_{(i,j)\in A_1\times A_2}c_{ij}$ dir. *s*-*t* kesiti $\left[\{s\}\cup A_1, \{t\}\cup A_2\right]$ dir. Bu kesit $i\in A_1$ ve kapasitesi α_i olan bir (*i*, *t*) arkı, $i\in A_2$ ve kapasitesi β_i olan bir (*s*, *i*) arkı ve bütün $i\in A_1$, $j\in A_2$ ve kapasitesi c_{ij} olan (*i*, *j*) arklarını kapsamaktadır. A_1 ve A_2 atamalarının maliyetinin $\left[\{s\}\cup A_1, \{t\}\cup A_2\right]$ kesitinin kapasitesine eşittir. Dolayısıyla bu atamanın minimum maliyeti *s* - *t* kesitinin minimumun kapasitesine eşittir. Teorem 2.1 de verilecek olan maksimum akış minimum kesit teoreminden dolayı *s* kaynak düğümünden *t* kuyu düğümüne yapılan maksimum akış A_1 ve A_2 atamalarının maliyetinin minimumunu verir. Bu uygulamanın bir örneği olarak dağıtımlı hesaplama için Şekil 2.11 de verilen veriler Şekil 2.12 de bir şebeke olarak verilmiştir.

$\{c_{ij}\} = 2$

	1	2	3	4
1	0	8	4	0
2	8	0	0	7
3	4	0	0	1
4	0	7	1	0

i	1	2	3	4
α_i	6	9	3	12
β_i	5	10	7	8

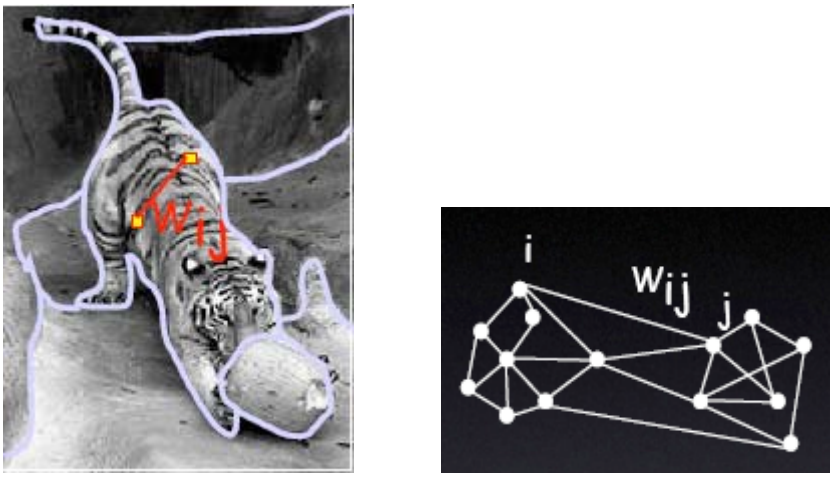
Şekil 2.11. Dağıtımlı hesaplama modeli için veriler



Şekil 2.12. Şekil 2.11 de verilen veriler için dağıtımlı hesaplama modeli için şebeke

2.7.4 Görüntü Bölütleme (Image Segmentation)

Görüntü bölütleme yapısal olmayan görüntüden bir yapısal görünüm ortaya çıkartmaya çalışır. Medikal görüntülemede bu anlamlı parçaların tespit edilmesi anlamına gelir. Görüntü bölütlemenin en büyük problemi nesnenin sınırlarının bulunmasıdır [15]. Bunun için yöntemlerden biri görüntüyü bir şebeke olarak göstermektir. Görüntüyü bir şebeke olarak göstermek için piksellerden oluşan küçük bir grup ya da pikseller düğümleri, pikseller arasındaki benzerlikler (weight) arkları oluşturur. Örneğin Şekil 2.13(a) da verilen resim üzerinde pikseller ve pikseller arasındaki benzerlik (weight), Şekil 2.13. (b) de verilen bu resim için bir şebeke örneği verilmiştir[16]. i düğümü ile j düğümü arasındaki benzerlik w_{ij} ile gösterilir. w_{ij} değeri ne kadar küçük ise i düğümü ile j düğümü arasındaki benzerlik o kadar azdır. Bu şebekede $\left[S, \overline{S}\right]$ bir kesit ise bu kesitin kapasitesi $C\left[S, \overline{S}\right] = \sum_{(i,j)\in A} w_{ij}$ dir. Görüntü bölütlemesinde minimal kesitler aranır. Teorem 2.1 de verilecek olan maksimum akış minimum kesit teorimden bu şebekedeki maksimum akış hesaplanarak minimum kesit bulunabilir [15].

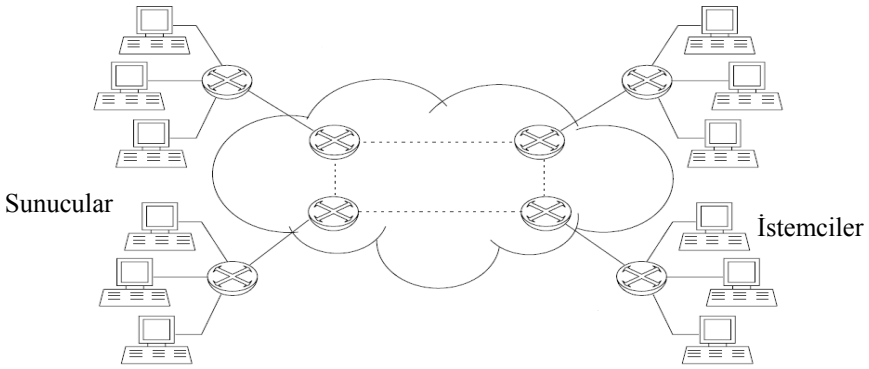


(a) **(b)**

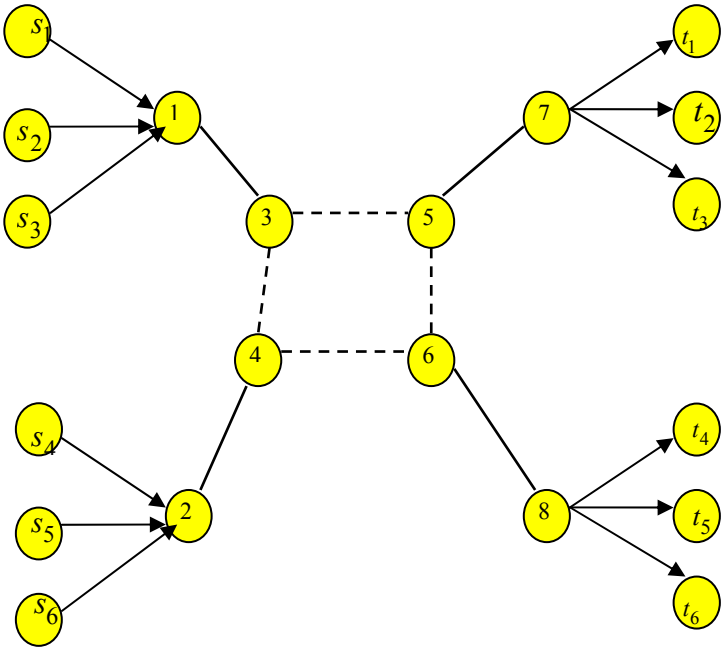
Şekil 2.13. (a) Bir resim üzerinde pikseller ve pikseller arası bezerlikler, **(b)** Resmin şebeke olarak gösterimi [16].

2.7.5. Bilgisayar Ağ Şebekeleri

Bilgisayar ağ şebekelerinde bilgisayarlar arasında veri alışverişi yapılır. Genel bir bilgisayar ağ şebekesi örneği Şekil 2.14 de verilmiştir. Burada istemciler sunucularda bulunan verilere en kısa zamanda ulaşması amaçlanmaktadır. Bunun için bu bilgisayar ağ şebekesini sunucuların kaynak düğümleri istemcilerin kuyu düğümleri ve yönlendiricilerin (routerlarında) ara düğümler olduğu bir şebekede maksimum akış problemi olarak kurulabilir. Oluşturulan şebekede kaynak düğümleri sayısı ve kuyu düğümleri sayısı birden fazla olduğundan Kısım 2.16 da verilecek olan yöntemle şebekedeki maksimum akış bulunur. Şekil 2.14 de verilen bilgisayar ağ şebekesini Şekil 2.15 te $s_1, s_2, s_3, s_4, s_5, s_6$ kaynak düğümleri $t_1, t_2, t_3, t_4, t_5, t_6$ kuyu düğümleri 1, 2, 3, 4, 5, 6, 7, 8 de ara düğümler olarak gösterilmiştir. Kaynak düğümlerinden kuyu düğümlerine yapılabilecek maksimum akış istemcilerin birim zamanda sunuculardan alabileceği maksimum verilerin miktarıdır.



Şekil 2.14. Bilgisayar ağ şebekesi örneği



Şekil 2.15. Şekil 2.14 teki verilen bilgisayar ağ şebekesinin düğüm-ark şebekesi olarak gösterimi

2.8 GELİŞTİRİLMİŞ YOL (AUGMENTING PATH) ALGORİTMASI

Arta kalan şebekede kaynak düğümünden kuyu düğümüne olan bir yönlü yola geliştirilmiş yol (augmenting path) denir. Bir geliştirilmiş yolun arta kalan (residual) kapasitesi bu yoldaki arkların kapasitelerinin minimumuna eşittir. Bir geliştirilmiş yolun

arta kalan kapasitesi her zaman pozitiftir. Bir şebekede geliştirilmiş yol olduğu sürece şebekede akış yapılabilir. Geliştirilmiş yol algoritmasında şebekede geliştirilmiş yollar tükenene kadar akış yapılır. Şekil 2.16 de geliştirilmiş yol algoritması tanımlanmıştır.

Geliştirilmiş yol algoritması;
begin
$x := 0$
while $G(x)$ te s düğümünden t düğümüne bir yol içerir ***do***
begin
$G(x)$ de s düğümünden t düğümüne bir P yolu belirle
$\delta := \min\{r_{ij} : (i,j) \in P\}$
P yolunda δ akış gönder ve $G(x)$ yenile
end;
***end*;**

Şekil 2.16 Geliştirilmiş yol algoritması

2.9 ETİKETLEME ALGORİTMASI VE MAKSİMUM AKIŞ–MİNİMUM KESİT TEOREMİ

Etiketleme algoritması geliştirilmiş yol algoritmasında yolun etiketleme yöntemiyle bulunmasına dayanmaktadır. Etiketleme algoritması iki adımdan oluşur. Adım 1 de düğümler etiketlenir, Adım 2 de akış değerinin artırımı için gerekli iterasyonlar yapılır.

Adım 1:

γ_i : Kaynak düğümünden j düğümüne gelen yol üzerinde, j düğümünden önceki düğüm. j düğümünü etiketlemede faydalanılan düğümün indisi.

δ_j : Kaynak düğümünden j düğümüne gelen yol üzerindeki maksimum boş kapasitedir. $\delta_j > 0$ ise bu yolda j ye akış yapılabilir. $\delta_j = 0$ ise akış gerçekleşmez, j düğümü bu yolla etiketlenmez.

olmak üzere eğer mümkünse, kaynak düğümünden başka, her i düğümü (δ_j , γ_j) ikilisi ile etiketlenir.

Eğer yol üzerinden j düğümünden önceki düğüm i ise $\gamma_j = i$ ve $\delta_j = \min\{\delta_i, b_{ij}\}$ dir. Kaynak düğümü için $\delta_s = \infty$ dur. Kuyu düğümünün etiketi $\delta_t > 0$ ise mevcut yol üzerinde, akış miktarı δ_t kadar artırılabilir. Eğer hiçbir yoldan kuyu düğümü etiketlenemez ise akış miktarı artırılamaz. Dolayısıyla etiketlenen düğümler S, etiketlenmeyen düğümler $\overline{S}$ olmak üzere $[S, \overline{S}]$ minimum kesitinde $C[S, \overline{S}]$ kapasitesi maksimal akışı verir.

Adım 2. Akış miktarını artırmak için iterasyonlar yapılır.

i. Üzerinde δ_t miktar akış yapılan yolun bir elemanı (arkı) (i, j) olsun. Akış yönü i düğümünden j düğümüne olmak üzere,

Yeni akış miktarları: $\hat{x}_{ij} = x_{ij} + \delta_t$

dir. Daha önce zıt yönde bir akış varsa, yani x_{ji} değeri varsa $x_{ij} = -x_{ji}$ alınmalıdır.

Arta kalan kapasiteler: $\hat{r}_{ij} = r_{ij} - \delta_t$ ve $\hat{r}_{ji} = r_{ji} + \delta_t$

dir.

ii. (i, j) arkı akış yapılan yolun bir elemanı değilse, akış ve boş kapasite miktarlarında değişiklik olmaz. Yani,

$$\hat{x}_{ij} = x_{ij} \quad \text{ve} \quad \hat{r}_{ij} = r_{ij}$$

dir.

iii. Yeni toplam akış miktarı

$$\hat{z} = z + \delta_t$$

olur. Yeni değerlerle Adım 1' e dönülür [14].

Etiketleme algoritmasında düğümler etiketlenenler ve etiketlenmeyenler olarak iki ayrık kümeye ayrılabilir. Etiketleme algoritmasında her bir iterasyonda ya akış yapılabilecek bir geliştirilmiş yol bulunur ya da kuyu düğümü etiketlenemez. Kuyu düğümünün

etiketlenemediği durumda maksimum akışa ulaşılmış olur. Bunu göstermek için şebekedeki etiketlenen düğümler S ile etiketlenmeyenler $\overline{S} = N - S$ ile gösterilsin. Bu durumda $s \in S$ ve $t \in \overline{S}$ dir. $\overline{S}$ deki düğümler S deki düğümler tarafından etiketlenemediği için bütün $(i, j) \in [S, \overline{S}]$ arkları için $r_{ij} = 0$ dır. $r_{ij} = \left(b_{ij} - x_{ij}\right) + x_{ji}$, $x_{ij} \le b_{ij}$, $x_{ji} \ge 0$ ve $r_{ij} = 0$ olduğundan dolayı her $(i, j) \in [S, \overline{S}]$ arkı için $x_{ij} = b_{ij}$ ve $x_{ji} = 0$ elde edilir. (2.5) denkleminden

$$v = \sum_{(i,j)\in\left[S,\overline{S}\right]} x_{ij} - \sum_{(i,j)\in\left[\overline{S},S\right]} x_{ij} = \sum_{(i,j)\in\left[S,\overline{S}\right]} b_{ij} = C\left[S,\overline{S}\right]$$

elde edilir. Bu da yapılan x akışının $\left[S,\overline{S}\right]$ kesitinin kapasitesine eşit olduğunu göstermektedir. Özellik 2.1 den dolayı x akışı maksimum akış, $\left[S,\overline{S}\right]$ kesiti de minimum kapasiteye sahip bir kesittir. Buradan aşağıdaki maksimum akış-minimum kesit teoremi ispatlanmış olur.

Teorem 2.1(Maksimum Akış - Minimum Kesit Teoremi). Bir şebekede s kaynak düğümünden t kuyu düğümüne yapılabilecek maksimum akış, şebekedeki bütün $s - t$ kesitlerinin minimum kapasitesine eşittir[13].

Teorem 2.2(Geliştirilmiş Yol Teoremi). x^* akışının maksimum akış olması için gerek ve yeter koşul $G(x^*)$ arta kalan şebekesinde bir geliştirilmiş yolun var olmamasıdır.

İspat. $G(x^*)$ arta kalan şebekesinde bir geliştirilmiş yol var olsa, x^* akışının maksimum olması ile çelişir. Tersine $G(x^*)$ arta kalan şebekesinde bir geliştirilmiş yol olmasa, S etiketlenen düğümler kümesi olmak üzere etiketleme algoritmasında olduğu gibi kapasitesi akışa eşit olan $\left[S,\overline{S}\right]$ kesiti bir $s - t$ kesitidir, dolayısıyla akış maksimum olur.

Sonuç 2.1. Bir $\left[S,\overline{S}\right]$ kesitinin minimal kesit olması için gerek ve yeter koşul her maksimal *x* akışı $\left[S,\overline{S}\right]$ kesitindeki her bir arkı doyuracak ve $\left[\overline{S},S\right]$ kesitindeki bütün arklardaki akış sıfır olacaktır, yani ters yönde akış olmayacaktır .

İspat. *x* akışı $\left[S,\overline{S}\right]$ kesitindeki her bir arkı doyursun ve $\left[\overline{S},S\right]$ kesitindeki bütün arklardaki akış sıfır olsun. Dolayısıyla *x* akışı $\left[S,\overline{S}\right]$ kapasitesi kadar akışı yapmış olacaktır. Maksimum akış-minimum kesit teoreminden *x* akışı maksimal olması için gerek ve yeter koşul $\left[S,\overline{S}\right]$ kesitinin minimal kesit olmasıdır.

Sonuç 2.2. $\left[X,\overline{X}\right]$ ve $\left[Y,\overline{Y}\right]$ iki minimal kesit ise $\left[X\cup Y,\ \overline{X\cup Y}\right]$ ve $\left[X\cap Y,\overline{X\cap Y}\right]$ kesitleri de minimal kesittir.

İspat. Sonuç 2.1 den $\left[X,\overline{X}\right]$, $\left[Y,\overline{Y}\right]$ kesitleri minimal ise bir *x* maksimal akış vardır ki bu minimal kesitlerdeki bütün arklar doymuştur. $\left[X\cup Y,\overline{X\cup Y}\right]$ kesiti minimal olmasa en az bir $i\in X\cup Y$ düğümünden $j\in\overline{X\cup Y}$ düğümüne giden (*i*, *j*) doymamış arkı var olurdu. Buradan $j\in\overline{X}$, $j\in\overline{Y}$ olur. $\left[X,\overline{X}\right]$, $\left[Y,\overline{Y}\right]$ kesitleri minimal olduğundan bu Sonuç 2.1 ile çelişir. Dolayısıyla böyle bir (*i*, *j*) arkı olamaz, yani $\left[X\cup Y,\ \overline{X\cup Y}\right]$ minimal kesittir. Benzer şekilde $\left[X\cap Y,\ \overline{X\cap Y}\right]$ kesiti minimal olmasa en az bir $k\in X\cap Y$ düğümünden $l\in\overline{X\cap Y}$ düğümüne giden (*k*, *l*) doymamış arkı vardır. Buradan $k\in X$, $k\in Y$ olur. $\left[X,\overline{X}\right]$, $\left[Y,\overline{Y}\right]$ kesitleri minimal olduğundan bu Sonuç 2.1 ile çelişir. Dolayısıyla $\left[X\cap Y,\ \overline{X\cap Y}\right]$ minimal kesittir.

Teorem 2.3. $\left[Y,\overline{Y}\right]$ herhangi bir minimal kesit, *x* bir maksimum akış ve $\left[X,\overline{X}\right]$ minimal kesitinde bütün arklarda $x_{ij}=b_{ij}$ ve $x_{ji}=0$ eşitlikleri var olursa $X\subseteq Y$ olur**.**

İspat. X Y nin bir alt kümesi olmasın. $X\cap Y\subset X$ ve Sonuç 5.4 ten $\left[X\cap Y,\ \overline{X\cap Y}\right]$ kesiti minimal kesittir. $s\neq i\in X$ ve $i\notin X\cap Y$ düğümü için $i\in X$ ve $i\neq s$ olduğundan

s düğümünden i düğümüne $s=1, ..., k=i$ şeklinde bir yol vardır. Bu yolda ileri yönde, yani kaynak düğümünden kuyu düğümüne giden bir yolda bulunan, (j, j+1) ($j=1,...,k$) arkları x tarafından doyurulmamış ve ters yöndeki arklarda pozitif akış var olur. Fakat $s \in X \cap Y$ ve $i \in \overline{X \cap Y}$ olduğundan dolayı $j \in X \cap Y$ ve $j+1 \in \overline{X \cap Y}$ olacak şekilde j, j + 1 ($1 \le j < k$) düğüm çifti vardır. Eğer (j, j+1) ileri yönlü bir ark ise $x_{j,j+1} < b_{j,j+1}$ olur, ters yönde bir ark olursa $x_{j,j+1} > 0$ olur ki her durumda Sonuç 2.1 ile çelişir. $X \subseteq Y$ olmak zorundadır.

Teorem 2.4. x bir maksimal akış olsun. X ve $\overline{Y}$ kümeleri aşağıdaki koşulları sağlayacak şekilde tanımlansın.

i. $s \in X$, **ii.** $i \in X$ ve $x_{ij} < b_{ij}$ ise $j \in X$, **iii.** $i \in X$ ve $x_{ji} > 0$ ise $j \in X$

iv. $t \in \overline{Y}$, **v.** $j \in \overline{Y}$ ve $x_{ij} < b_{ij}$ ise $i \in \overline{Y}$, **vi.** $j \in \overline{Y}$ ve $x_{ji} > 0$ ise $i \in \overline{Y}$

Ayrıca her i, j için $b_{ij} > 0$ olsun. $\left[X, \overline{X}\right]$ minimal kesitinin tek olası için gerek ve yeter koşul $\left[X, \overline{X}\right] = \left[Y, \overline{Y}\right]$ olmasıdır.

İspat. Eğer $\left[X, \overline{X}\right] = \left[Y, \overline{Y}\right]$ ve $\left[Z, \overline{Z}\right]$ herhangi bir minimal kesit ise $\left[X, \overline{X}\right] = \left[Z, \overline{Z}\right]$ olduğu gösterilirse teorem ispat edilmiş olur. Eğer $\left[X, \overline{X}\right] = \left[Y, \overline{Y}\right]$ ise $\left[X, \overline{X}\right] = \left[Y, \overline{Y}\right] = \left[X, \overline{Y}\right]$ dir. Teorem 2.3 ten $X \subseteq Y$ dir, böylece $\left[X, \overline{Y}\right] \subseteq \left[Y, \overline{Y}\right]$ olur. Eğer $(a,b) \in \left[X, \overline{X}\right] = \left[Y, \overline{Y}\right]$ ise $a \in X$, $b \in \overline{Y}$ ve $(a,b) \in \left[Y, \overline{Y}\right]$ olur. $\left[Z, \overline{Z}\right]$ herhangi bir minimal kesit ise, bu kesit ile $\left[Y, \overline{Y}\right]$ kesitine Teorem 2.3 ü uygularsak $Y \subseteq Z$ bulunur ve $X \subseteq Y$ olduğundan $X \subseteq Z$ elde edilir. Ayrıca $\overline{Z} \subseteq \overline{Y}$ dir. Böylece $\left[X, \overline{Y}\right] \subseteq \left[Z, \overline{Y}\right] \subseteq \left[Z, \overline{Z}\right]$ elde edilir. Buradan $C\left[X, \overline{Y}\right] \le C\left[Z, \overline{Z}\right]$ bulunur. Eğer $\left[X, \overline{Y}\right] \subset \left[Z, \overline{Z}\right]$ ise ya $\left[Z, \overline{Z}\right]$ de sıfır kapasiteli arklar vardır, ki bu teoremdeki kabulle çelişir, ya da $C\left[X, \overline{Y}\right] < C\left[Z, \overline{Z}\right]$ olur ki bu da $\left[Z, \overline{Z}\right]$ minimalliği ile çelişir. Böylece $\left[X, \overline{Y}\right] = \left[Z, \overline{Y}\right] = \left[Z, \overline{Z}\right]$ dir.

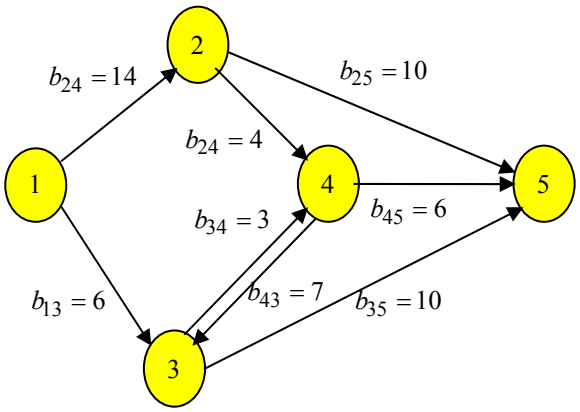
Teorem 2.5. Eğer bir şebekede bütün arkların kapasiteleri tam sayı ise maksimum akış problemi bir tam sayı maksimum akışa sahiptir.

İspat. Artırımların (augmentations) sayısına indüksiyon yöntemi uygulanarak bu teorem ispat edilebilir. Etiketleme algoritması sıfır akışla başlar ve bütün arkların kapasiteleri ve başlangıç arta kalan şebekedeki arklar tam sayıdır. Her bir iterasyondaki artış, bir yolun arta kalan kapasitesinin minimumuna eşittir ve indüksiyon hipotezinden dolayı tam sayıdır. Bir sonraki adımda arta kalan kapasite tam sayıdır. Çünkü r_{ij} arta kalan kapasite ve ark kapasitesi b_{ij} tam sayıdır. Eğer arta kalan kapasiteleri akış olarak çevirirsek, yani arta kalan kapasite kadar akış gönderilirse, arklardaki x_{ij} akışları tam sayı olurlar. Kapasiteler tam sayı olduğundan her bir artırım akışa en az bir birimlik artış sağlar. Maksimum akış hiç bir kesitin kapasitesini geçemeyeceğinden algoritma sonlu iterasyon adımında sonlanır.

Maksimum akış probleminin tam sayı olmayan çözümleri olabilir. Fakat Teorem 2.5. ten maksimum akış probleminin en az bir tam sayı optimal çözümün vardır.

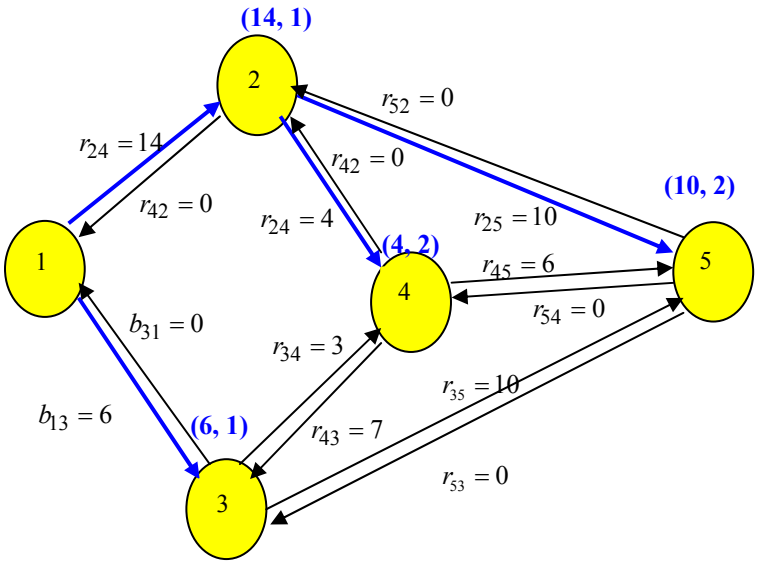
Teorem 2.6. Etiketleme algoritmasının karmaşıklığı $O(mnU)$ dur. [2, syf 186].

Örnek 2.1 Şekil 2.17 te verilen şebeke için 1 kaynak düğümünden 5 kuyu düğümüne yapılabilecek maksimum akış miktarını ve her bir arkta oluşacak boş kapasiteleri etiketleme algoritması ile belirleyelim.



Şekil 2.17. Boş kapasitelerden oluşan bir şebeke

Başlangıçta hiç bir arkta akış olmadığından $\forall(i,j)$ için $x_{ij}=0$ ve $r_{ij}=b_{ij}$ dir. Şekil 2.18. te bu şebekenin arta kalan boş kapasiteler şebekesi ve ilk etiketlemesi yapılmıştır.



Şekil 2.18. Şekil 2.17 de verilen şebekenin arta kalan boş kapasiteler şebekesi ve düğümlerin etiketlenmesi

Adım 1. 1 düğümü kaynak düğümüdür ve $\delta_1=\infty$ dur. 1 düğümüne komşu 2 ve 3 düğümleri vardır ve $r_{12}=14>0$, $r_{13}=6>0$ olduğundan bu düğümler 1 düğümünden etiketlenebilirler.

2 düğümünün etiketi : $\delta_2=\min(\delta_1, r_{12})=\min(\infty, 14)=14$ için $(\delta_2=14, \gamma_2=1)$

3 düğümünün etiketi : $\delta_3=\min(\delta_1, r_{13})=\min(\infty, 6)=6$ için $(\delta_3=6, \gamma_3=1)$

dir. 2 düğümüne komşu etiketsiz düğümler 4 ve 5 düğümleri vardır $r_{24}=4>0$, $r_{25}=10>0$ olduğundan 4 ve 5 düğümleri 2 düğümünden etiketlenebilir.

4 düğümünün etiketi: $\delta_4=\min(\delta_2, r_{24})=\min(14, 4)=4$ için $(\delta_4=4, \gamma_4=2)$

5 düğümünün etiketi: $\delta_5=\min(\delta_3, r_{35})=\min(14, 10)=10$ için $(\delta_5=10, \gamma_5=2)$

dir. Bu etiketler şebeke üstünde Şekil 2.18 de gösterilmiştir. 5 kuyu düğümü etiketlendiğinden kaynak düğümünden kuyu düğümüne kapasitesi $\delta_5 = 10$ olan bir yol bulunmuş olur. Bu yol 1, (1,2), 2, (2,5), 5 tir ve bu yoldan kaynak düğümünden kuyu düğümüne $\delta_5 = 10$ birim akış yapılabilir.

Adım 2.

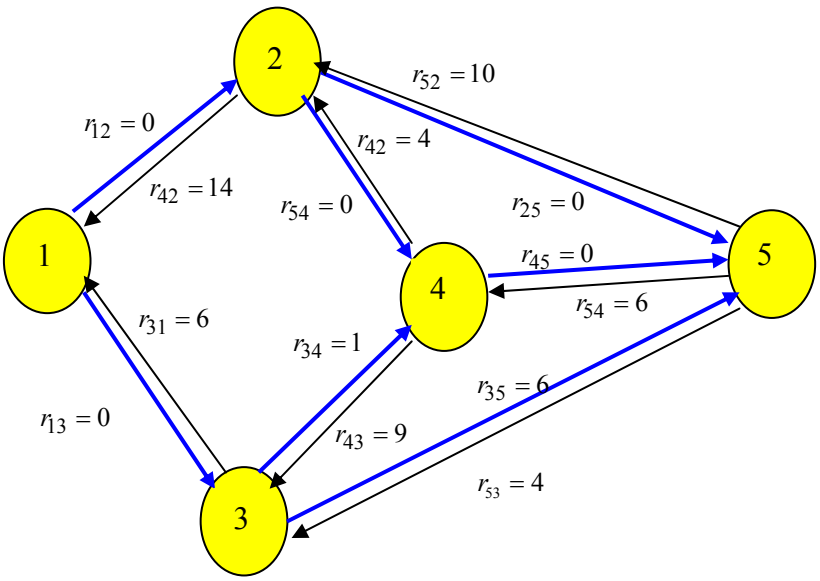
Yeni akış miktarları : $\hat{x}_{12} = x_{12} + \delta_5 = 0+10 = 10$, $\hat{x}_{25} = x_{25} + \delta_5 = 0+10 = 10$

Arta kalan kapasiteler : $\hat{r}_{12} = r_{12} - \delta_5 = 14-10 = 4$, $\hat{r}_{21} = r_{21} + \delta_5 = 0+10 = 10$

$\hat{r}_{25} = r_{25} - \delta_5 = 10-10 = 0$, $\hat{r}_{52} = r_{52} + \delta_5 = 0+10 = 10$

Yeni toplam akış miktarı : $\hat{z} = z + \delta_5 = 0+10 = 10$

Diğer arklarda akış miktarları boş kapasiteler aynı kalır. Yeni bilgileri kullanılarak şebekedeki akışı artırabilmek için düğümleri yeniden etiketlenir ve algoritma aynı yöntemle hesaplamaya devam edilirse sonuçta Şekil 2.19 te verilen şebeke elde edilir.



Şekil 2.19. Şekil 2.17 te verilen şebekenin maksimum akıştan sonra arta kalan şebekesi

Bu şebekeden görüldüğü gibi $r_{12} = 0$ ve $r_{13} = 0$ olduğundan 1 kaynak düğümünden komşu düğümler etiketlenemez. Bu aşamada akışı artırmak mümkün değildir, maksimal akış sağlanmıştır. Maksimal akış sağlandığında

Arklarda akış miktarları : $x_{12}=14, x_{13}=6,\ x_{24}=4,\ x_{25}=10, x_{34}=2,\ x_{35}=4,\ x_{45}=6$

Maksimal akış değeri : $z=20$

dir.

2.10. ETİKETLEME ALGORİTMASININ MATRİSLE GÖSTERİMİ

Kısım 2.9 da verilen etiketleme algoritmasının matrisle gösteriminde şebeke Kısım 2.3 te verilen birinci düğüm- düğüm matris gösterimi kullanılır. Bu matrise δ_j ve γ_j sütunları eklenir. Burada, aynı etiketleme algoritmasında olduğu gibi, δ_j kaynaktan j düğümüne bir yol boyunca ne kadar boş kapasite kullanılabileceğini, γ_j ise j düğümüne hangi düğümden gelindiğini belirler. Şekil 2.20 de bu matris genel olarak verilmiştir [14].

Düğüm	**1**	**2**	**…**	***j***	**…**	***n***	**Etiketler**	
1	---	r_{12}	⋯	r_{1j}	⋯	r_{1n}	δ	γ
2	r_{21}	---	⋯	r_{2j}	⋯	r_{2n}	δ_1	γ_1
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
i	r_{i1}	r_{i2}	⋯	r_{ij}	⋯	r_{in}	δ_i	γ_i
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
n	r_{n1}	r_{n2}	⋯	r_{jn}	⋯	---	δ_n	γ_n

Şekil 2.20. Etiketleme algoritmasının matris gösterimi

Algoritma birinci satırla başlar. $r_{1j}>0$ olan sütunlar bulunur. Buna karşılık gelen j düğümleri $\delta_j=r_{1j}$, $\gamma_j=1$ ile etiketlenir. n kuyu düğümü etiketlendiğinde birinci aşama son bulur. Eğer kuyu düğümü etiketlenmediyse etiketlenen en küçük indisli satıra geçilir. Bu i. satır olsun. i. satırda $r_{ij}>0$ ve j. satır henüz etiketlenmemiş olsun. Şu halde j düğümünün etiketi

$$\delta_j=\min\left(\delta_i, r_{ij}\right) \quad \text{ve} \quad \gamma_j=i$$

dir. Eğer δ_n belirlenemediyse aynı işlem etiketlenmiş en küçük indisli satırla devam edilir. Böylece sonlu adımda δ_n belirlenemez ise akış değeri maksimumdur. Belirlenebilir ise akış değeri δ_n kadar artırabilir. Eğer şebekede δ_n akış yapılırsa akış yapılan yolda bulunan bütün (i, j) arklarındaki boş kapasite δ_n kadar azalır, ters yöndeki (j, i) arkında ise δ_n kadar artar ve matriste bu (i, j) arklarına karşılık gelen (i, j) elemanlarının değeri δ_n kadar azalır, (j, i) arklarına karşılık gelen (j, i) elemanlarının değeri δ_n kadar artar.

Herhangi bir şebeke için b_{j1} (kaynak düğümüne gelen akış) ve b_{nj} (kuyu düğümünden çıkan akış) kapasiteleri bütün j düğümleri için sıfırdır. Tablodan akış yapılabilecek bir yol kalmadığında tablonun son satırı ya da birinci sütundaki değerin toplamı bu şebekede yapılabilecek maksimum akışı verir. Ayrıca şebekede akışlar bittikten sonra hangi arkta ne kadar akış yapıldığı bulmak için şebekenin akış yapılmadan önceki durumu gösteren matris ile bütün akışlar yapıldıktan sonra arta kalan kapasiteleri gösteren matriste sadece arkları gösteren elemanlar birbirinden çıkartılır. Çıkan matriste (i, j) elemanının değeri pozitif α ise (i, j) arkında α akış yapılmış olur. Eğer α negatifse bu durumda (i, j) arkının tersi yönde α akış yapılmış olur.

Örnek 2.2. Şekil 2.21 de düğüm-düğüm matrisi ile verilen şebekede 1 kaynak düğümü 8 kuyu düğümü olmak üzere kaynak düğümünden kuyu düğümüne yapılabilecek maksimum akışı etiketleme algoritmasının matris yöntemiyle bulalım.

Düğüm	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
1	---	4	12	8	12	0	4	0
2	0	---	18	14	8	4	8	8
3	0	4	---	0	4	4	14	10
4	0	0	5	---	12	8	0	8
5	0	0	8	8	---	8	12	4
6	0	4	5	0	7	---	0	4
7	0	7	1	6	6	9	---	14
8	0	0	0	0	0	0	0	---

Şekil 2.21. Boş kapasitelerden oluşan bir şebeke

Çözüm: Önce verilen matrise etiket sütunları eklenir. Matriste 1. satırda sıfırdan farklı elemanların olduğu sütunlar 2, 3, 4, 5, 7 olduğundan 2, 3, 4, 5, 7 satırları 1. satırdan etiketlenir. Bu düğümlere gelen akışlar, δ_i değerleri, bu satırdaki sıfırdan farklı

sayılardır. Örneğin 2. satırın etiketi $\delta_2 = r_{12} = 4$ ve $\gamma_2 = 1$ dir. 6. satır 1. satırdan etiketlenemediğinden 2 satırdan (etiketlenen en küçük indisli satır 2 olduğundan) etiketlenip etiketlenemeyeceğine bakılır. 2. satır ve 6. sütunun kesişimi $r_{26} = 4 > 0$ olduğunda 6. satır 2. satırdan etiketlenir ve etiketi $\gamma_6 = 2$ $\delta_6 = r_{26} = 4$ dır. Ayrıca $r_{28} = 8 > 0$ olduğundan 8. satır, 2. satırdan etiketlenir ve etiketi $\gamma_8 = 1$ $\delta_8 = \min(\delta_2, r_{28}) = \min(4,8) = 4$ dir. Şebekedeki diğer düğümlerin etiketleri Şekil 2.22 de verilmiştir.

Düğüm	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**Etiketler**	
1	---	4	12	8	12	0	4	0	δ	γ
2	0	---	18	14	8	4	8	8	4	1
3	0	4	---	0	4	4	14	10	12	1
4	0	0	5	---	12	8	0	8	8	1
5	0	0	8	8	---	8	12	4	12	1
6	0	4	5	0	7	---	0	4	4	2
7	0	7	1	6	6	9	---	14	4	1
8	0	0	0	0	0	0	0	---	4	2

Şekil 2.22. Düğümlerin etiketlenmesi

Kuyu düğümü etiketlendiğine göre bu aşamada akış yapılabilecek yol kuyu düğümü, kuyu düğümünü etiketleyen satır olan 2. satır, 2. satırı etiketleyen satır olan 1. satır, yani kaynak düğümü olarak bulunur. Dolayısıyla akış 1, (1, 2), 2, (2, 8), 8 yolunda yapılır ve $\delta_8 = 4$ birim akış yapılır. Akış sonrası şebekedeki arta kalan boş kapasiteler Şekil 2.23 te verilmiştir.

Düğüm	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
1	---	0	12	8	12	0	4	0
2	4	---	18	14	8	4	8	4
3	0	4	---	0	4	4	14	10
4	0	0	5	---	12	8	0	8
5	0	0	8	8	---	8	12	4
6	0	4	5	0	7	---	0	4
7	0	7	1	6	6	9	---	14
8	0	4	0	0	0	0	0	---

Şekil 2.23 Birinci akıştan sonraki arta kalan şebeke

Şekil 2.23 teki şebekede bir önceki akışta yapılan işlemler tekrarlanır. Bu işlemler kuyu düğümü etiketlenemeyene kadar tekrarlanırsa Şekil 2.24 teki şebeke bulunur. Kuyu düğümü etiketlenemediğinden dolayı bu şebekede maksimum akışa ulaşılmış olur. Bu şebekede yapılabilecek maksimum akış 40 birimdir. Bu yöntemle bu şebekedeki maksimum akışa 11 adımda ve toplam 38 arkta akış yapılmıştır.

Düğüm	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
1	---	0	0	0	0	0	0	0
2	4	---	22	14	8	6	6	0
3	12	0	---	4	6	0	14	0
4	8	0	1	---	18	6	0	0
5	12	0	6	2	---	8	12	0
6	0	2	9	2	7	---	0	0
7	4	9	1	6	6	9	---	8
8	0	8	10	8	4	4	6	---

Şekil 2.24. Şebekede maksimum akış yapıldıktan sonra arta kalan şebeke

Bu şebekede maksimum akış yapıldıktan sonra arklarda yapılan akışlar Şekil 2.25 te matris şeklinde verilmiştir. Bu matriste (i, j) elamanın değeri $\alpha = 0$ ise (i, j) arkında akış yapılmamış demektir, $\alpha > 0$ ise (i, j) arkında α birim akış yapılmıştır, $\alpha < 0$ ise (i, j) nin ters yönünde yani (j, i) arkında α birim akış yapılmıştır. Örneğin matrisin (1,3) elemanının değeri 12 olduğunda (1, 3) arkında 12 birimlik, (2,3) elemanının değeri -4 olduğundan (3, 2) arkında 4 birimlik akış yapılmıştır.

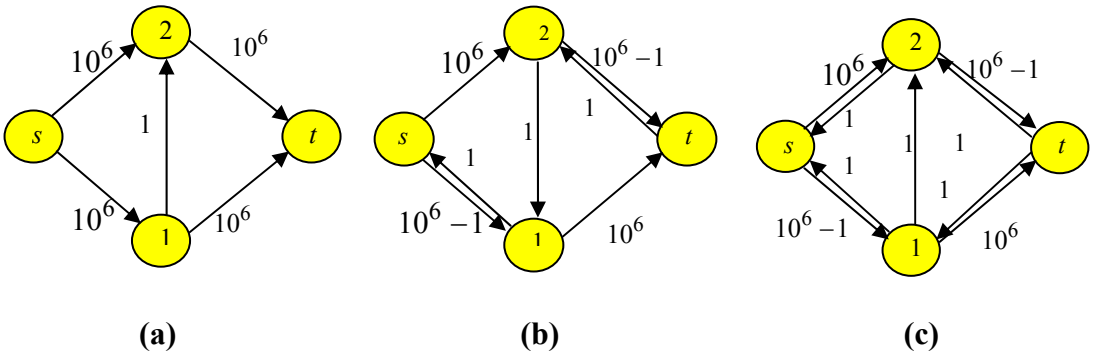
Düğüm	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
1	---	4	12	8	12	0	4	0
2	-4	---	-4	14	0	-2	2	8
3	-12	4	---	-4	-2	4	0	10
4	-8	0	4	---	-6	2	0	8
5	-12	0	2	6	---	0	0	4
6	-0	2	-4	-2	0	---	0	4
7	-4	-2	0	0	0	0	---	6
8	0	-8	-10	-8	-4	-4	-6	---

Şekil 2.25. Şebekede maksimum akış yapıldıktan sonra arklarda yapılan akışlar

Not 2.1. Etiketleme Algoritmasının Zayıflıkları

Etiketleme algoritması maksimum akış probleminin çözümü için en basit algoritmadır. Buna karşılık şebekedeki en büyük kapasiteli arkın kapasitesi U çok büyük olursa örneğin $U = 2^n$, bu durumda algoritmanın maksimum akışı bulması için gerekli işlem sayısını artırabilir. Şekil 2.26 da verilen şebekede algoritma s-1-2-t ve s-2-1-t yolların 10^6 kere seçebilir, her bir artırımda yolda bir birimlik akış taşınır.

Bundan dolayı etiketleme algoritmasını etkinliği garantili ise geliştirilmiş yol çok dikkatli seçilmelidir.

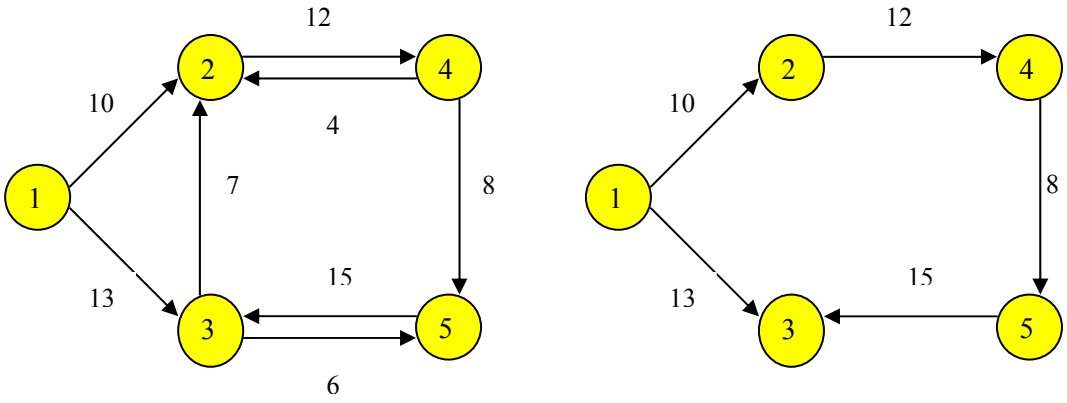


(a) **(b)** **(c)**

Şekil 2.26. Etiketleme algoritmasının zayıflığı için bir örnek: **(a).** sıfır akış için arta kalan şebekede, **(b)** s-1-2-t yolundan 1 birimlik akış yapıldıktan sonraki arta kalan şebeke, **(c)** s-2-1-t yolundan 1 birimlik akış yapıldıktan sonraki arta kalan şebeke.

2.11. KAPASİTE ÖLÇEKLEME (CAPACITY SCALING) ALGORİTMASI

Bu algoritmada her zaman maksimum kalan geliştirilmiş yoldan akış gönderir [2, 17]. Akış artırımı maksimum boş kapasiteli yol yerine yeteri kadar büyük boş kapasiteli bir yoldan yapılır. Çünkü yeteri kadar büyük bir yol $O(m)$ işlemde bulunur. Verilen bir x akışına göre, arkların kapasiteler en az Δ olan arklar ve bu arklara bağlı düğümlerin oluşturduğu alt şebekeye Δ arta kalan şebekesi denir ve $G(x,\Delta)$ ile gösterilir. Ayrıca $G(x,1) = G(x)$ dir. Şekil 2.27 (a) da bir $G(x)$ şebekesi ve (b) de $\Delta = 8$ için $G(x,\Delta)$ şebekesi örneği verilmiştir.



(a) **(b)**

Şekil 2.27. **(a)** $G(x)$ şebekesi, **(b)** $\Delta = 8$ için $G(x, \Delta)$ alt şebekesi.

Bu algoritmada her bir Δ aşamasında en az Δ kadar akış gönderecektir. Algoritma $\Delta = 2^{\lfloor \log_2 U \rfloor}$ ile başlar ve her bir aşamadan sonra Δ 1 olana kadar yarım yarım azalır. Δ aşamaları $1 + \lfloor \log_2 U \rfloor = O(\log_2 U)$ sayıdadır. Son aşamada $\Delta = 1$ ve $G(x, \Delta) = G(x)$ olacaktır. Bu da algoritmanın maksimum akışı bulduğunu gösterir.

Teorem 2.7: Kapasite ölçekleme algoritması maksimum akış problemini $O(m \log_2 U)$ artırımda ve $O(m^2 \log_2 U)$ karmaşıklığında çözer [2, syf. 182, 17].

2.12 UZAKLIK ETİKETLERİ (DISTANCE LABEL)

2.12.1. Tanımlar ve Özellikler

Bir $G = (N, A)$ şebekesi için uzaklık fonksiyonu $d : N \to \mathbb{N} \cup \{0\}$, r_{ij} arta kalan kapasiteye bağlı olarak düğümlerden nonnegatif tam sayılara giden bir fonksiyondur. d fonksiyonuna aşağıdaki koşulları sağlarsa geçerlidir denir:

$i)$ $d(t) = 0$ (2.9)

$ii)$ $d(i) \le d(j) + 1$, $G(x)$ arta kalan şebekesindeki her bir (i, j) arkı için (2.10)

$d(i)$ ye i düğümünün uzaklık etiketi ve (2.9), (2.10) koşullarına geçerlilik koşulları denir.

Kabul edilebilir ark, kabul edilebilir yol. Arta kalan şebekedeki (i, j) arkı $d(i) = d(j) + 1$ koşulunu sağlarsa (i, j) arkına kabul edilebilir ark (admissible) denir. s kaynak düğümünden t kuyu düğümüne giden bir yoldaki bütün arklar kabul edilebilirse bu yola kabul edilebilir yol (admissible path) denir.

Özellik 2.4. Eğer uzaklık etiketi geçerli ise arta kalan şebekede $d(i)$ uzaklık etiketi i düğümünden t kuyu düğümüne olan bir en kısa yolun bir alt sınırıdır [2, syf. 209].

Özellik 2.5. Eğer $d(s) \geq n$ ise arta kalan şebekede s kaynak düğümünden t kuyu düğümüne bir yol yoktur [2, syf. 210].

Özellik 2.6. Bir kabul edilebilir yol kaynak düğümünden kuyu düğümüne giden bir en kısa geliştirilmiş yoldur [2, syf. 210].

2.12.2. Uzaklık Etiketini Yenileme (Relabel)

Başlangıç düğümü i olan arkların kümesini $A(i)$ ile gösterilsin. i düğümü kabul edilebilir bir arka sahip olmasın yani $d(i) = d(j) + 1$ ve $r_{ij} > 0$ olan $(i, j) \in A(i)$ arkı var olmasın. Bu durumda $d'(i) = \min\{d(j) + 1 : (i, j) \in A(i) \text{ ve } r_{ij} > 0\}$ i düğümünün yeni etiketi olarak tanımlanır. Bu durumda (2.10) geçerlilik koşulu $d(i) \leq d(j) + 1$ yerine $d(i) < d(j) + 1$ olur. Buradan $d(i) < d'(i)$ dir. Yani bir düğümün etiketini yenilemek etiketinde bir artış yapmaktadır.

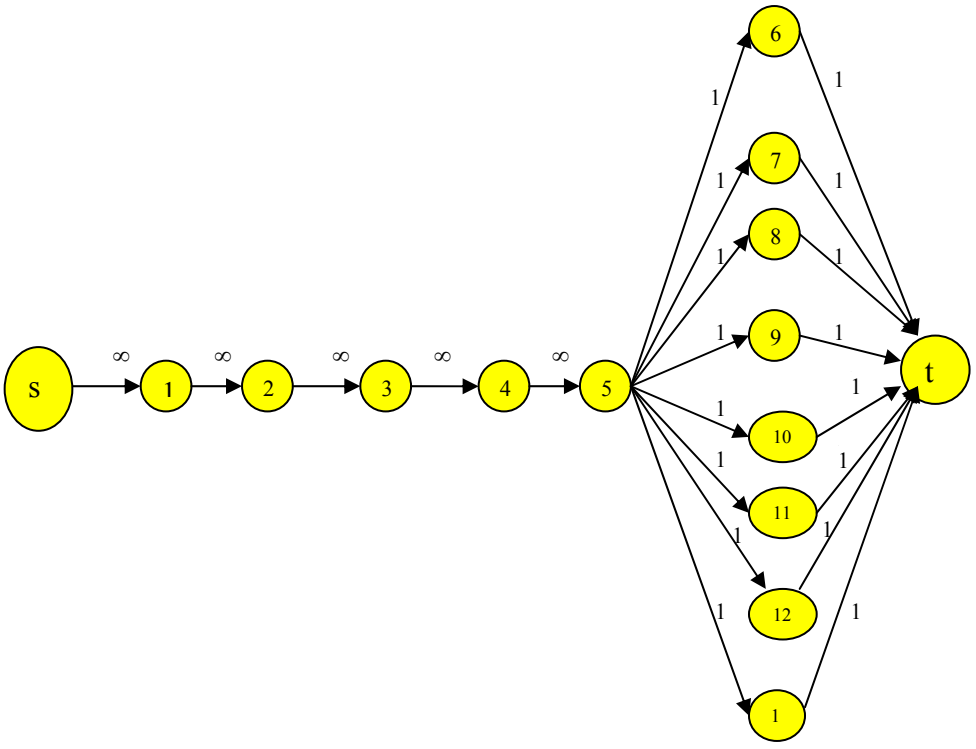
Özellik 2.7. Eğer algoritma herhangi bir düğümü k kez etiketini yenilerse bir kabul edilebilir ark bulma ve etiket yenilemenin için toplam $O\left(k\sum_{i \in N}|A(i)|\right) = O(km)$ işlem yapılır. [2, syf. 217].

Lemma 2.1. Eğer algoritma herhangi bir düğümü en fazla k kere etiketini yenilerse algoritma arkları en fazla $km/2$ işlemde doyurur. [2, syf. 217].

Teorem 2.8. Kapasite ölçekleme algoritmasında eğer geliştirilmiş yol bir kabul edilebilir yol olarak seçilirse kapasite ölçeklendirme algoritmasının karmaşıklığı $O(nm \log_2 U)$ dur. [2, 17].

2.13 GENEL AKIŞ ÖNCESİ GÖNDERME (GENERİC PREFLOW-PUSH) ALGORİTMASI

Bu tür algoritmalar geliştirilmiş yol algoritmalarına göre daha geneldir, daha güçlüdür ve daha esnektir. Geliştirilmiş yol probleminin asıl zayıflığı, eksikliği, bir geliştirilmiş yolda en kötü durumda $O(n)$ işlemde akış göndermesidir. Akış öncesi gönderme algoritmalarında bu zayıflık yoktur ve algoritmanın en kötü durumdaki karmaşıklığını düzeltmektedirler. Şekil 2.28 de verilen arta kalan şebekede hangi geliştirilmiş yol algoritması uygulanırsa uygulansın 8 geliştirilmiş yol, her bir geliştirilmiş yolun uzunluğu, ark sayısı, 7 ve her bir geliştirilmiş yoldan sadece 1 birim akış yapılabilir. *s* kaynak düğümünden 5 düğümüne olan yol bütün bu geliştirilmiş yollarda ortak yol olduğundan *s* kaynak düğümünden 5 düğümüne 8 birim akış gönderilirse daha sonra 5 düğümünden sonraki uzunluğu 2 olan 8 yoldan her birine 1 birimlik akış gönderilirse, baştaki ilk 5 arklık yol tekrar edilmemiş olur. Bu akış öncesi gönderme algoritmalarının temel düşüncesidir.



Şekil 2.28. Arta kalan şebekede geliştirilmiş yol algoritmasının zayıflığı

Geliştirilmiş yol algoritmaları bir geliştirilmiş yol boyunca akış gönderirler. Akış öncesi gönderme algoritmalarında ise belli arklara akış yapılır.

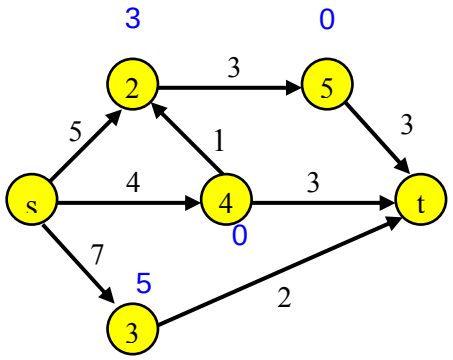
x bir önakış (preflow) ise, $x : A \to \mathbb{R}$ şeklinde bir fonksiyon ve (2.2) kısıtını sağlayan fakat (2.1) ile verilen kısıt yerine

$$\sum_{\{j:(j,i)\in A\}} x_{ji} - \sum_{\{j:(i,j)\in A\}} x_{ij} \geq 0\,, \quad \forall i \in N - \{s, t\}$$

kısıtını sağlar. Verilen bir x önakışta (preflow) her bir düğümdeki fazlalık (excess) $e(i)$

$$e(i) = \sum_{\{j:(j,i)\in A\}} x_{ji} - \sum_{\{j:(i,j)\in A\}} x_{ij} \geq 0$$

şeklinde tanımlanır. Yani bir düğümdeki fazlalık düğüme gelen akıştan düğümden çıkan akışın çıkartılmasıyla bulunur. Bir önakışta $\forall i \in N - \{s, t\}$ için $e(i) \geq 0$ ve t kuyu düğümünde akış çıkışı olmadığından $e(t) \geq 0$ dır. Negatif fazlalığa sadece s kaynak düğümü sahiptir. Fazlalığı pozitif olan düğüme aktif düğüm denir. Şekil 2.29 verilen şebekede fazlılıkları düğümlerin üzerinde gösterilmiştir. Bu şebekedeki fazlalıklar $e(2) = (5+1)-3 = 4\,, e(3) = 5\,, e(4) = e(5) = 0$ ve 2 ve 3 düğümleri aktif düğümlerdir.



Şekil 2.29. Bir şebekedeki fazlalıklar

Akış öncesi gönderme algoritmasında bir aktif düğüm seçilir ve bu aktif düğümdeki fazlalıkları komşu düğümlere, uzaklık etiketleri kendi uzaklık etiketlerinden 1 küçük olan düğümlere, doğru boşaltır. Kuyu düğümüne akış göndermek için kuyu düğümüne en yakın olan düğümler seçilmelidir. Yani kabul edilebilir arklara akış yapılmalıdır.

Kabul edilebilir arklar olmadığında düğümün uzaklık etiketi arttırılır. Algoritma aktif düğümler bitince sonlanır. Şekil 2.30 de bu algoritma verilmiştir.

Preprocess
begin
x :=0;
her bir düğümün uzaklık etiketlerini $d(i)$ hesapla ;
$x_{sj} := b_{sj}$ her bir $(s, j) \in A(s)$ arkı için;
$d(s) := n$;
end
***Procedure* gönderme/etiket yenileme(i);**
begin
if şebeke bir (i, j) kabul edilebilir arka sahipse ***then***
i düğümünden j düğümüne $\delta := \min\{e(i), r_{ij}\}$ birim akış gönder;
else $d(i)$ yi $\min\{d(j)+1:(i, j) \in A(i), r_{ij} > 0\}$ ile değiştir;
end;
akış öncesi gönderme algoritması;
begin
preprocess;
while şebekede bir aktif düğüm var ***do***
begin
bir aktif i düğümü seç;
gönderme /etiket yenileme(i);
end;
end*;*

Şekil 2.30 Akış öncesi gönderme algoritması

i düğümünden j düğümüne δ birim akış yapılırsa $e(i)$ ve r_{ij} değerleri δ kadar azalır, $e(j)$ nin değeri δ kadar artar. Bir (i, j) arkındaki gönderme $\delta = r_{ij}$ ise buna doyuran (saturating), aksi halde doyurmayan (nonsaturating) gönderme denir.

Teorem 2.9. Genel akış öncesi gönderme algoritmasının karmaşıklığı $O(n^2m)$ dir. [2, syf. 229, 18].

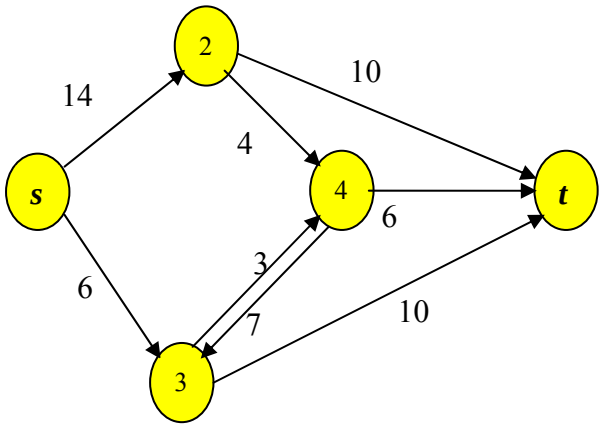
2.14 EN BÜYÜK ETİKETLİ AKIŞ ÖNCESİ GÖNDERME (HIGHEST LABEL PREFLOW-PUSH) ALGORİTMASI

Bu algoritmada genel akış öncesi gönderme algoritmasında yapılan göndermenin en büyük uzaklık etiketine sahip aktif bir düğümden uzaklık etiketi daha küçük olan düğümlere yapılır. Yani $h^* = \text{maks}\{d(i) : i \text{ aktif düğüm}\}$ olarak tanımlanırsa algoritma uzaklık etiketi h^* olan düğümlerden uzaklık etiketi h^*-1, buradaki düğümlerden etiketi h^*-2 olan düğümlere, şeklinde devam ederek aktif düğümler bitene kadar ya da etiket yenilemeye kadar akış gönderir. Etiket yenileme yapılırsa algoritmada aynı işlemler tekrarlanır. Eğer algoritma n ardışık işlemde etiketleme yapamadığı zaman bütün fazlalıklar kuyu düğümüne ulaşmış ve algoritma sonlanmış olur [2 syf. 233, 18].

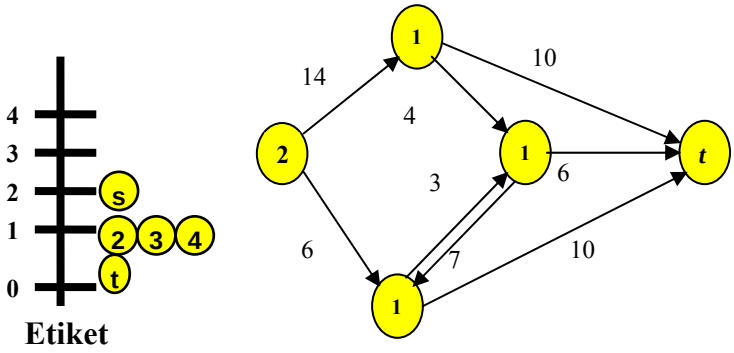
En büyük etiketli akış öncesi gönderme algoritması en az doyurmayan akış sayısına sahip olduğundan uygulamada maksimum akış problemini çözen en etkili algoritmalardan biridir.

Teorem 2.10. En büyük etiketli akış öncesi gönderme algoritmasının karmaşıklığı $O(n^2\sqrt{m})$ dir [2, 18, 19].

Örnek 2.3. Örnek 2.1 de verilen şebekede maksimum akışı bulmak için en büyük etiketli akış öncesi gönderme algoritması ile çözelim: Bunun için Şekil 2.31 de verilen boş kapasiteleri arkların üzerinde gösterilen şebekede, Şekil 2.32 de düğümler yerine uzaklık etiketlerini yazıldı ve solda etiket tablosu yapıldı.

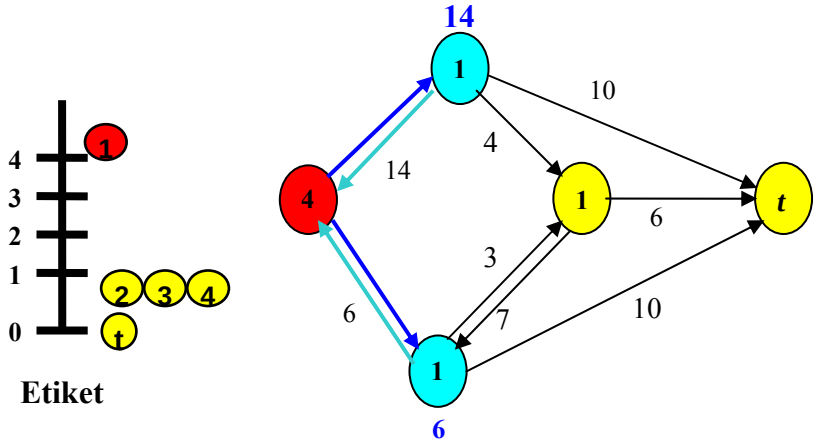


Şekil 2.31. Örnek 2.1 de verilen şebeke



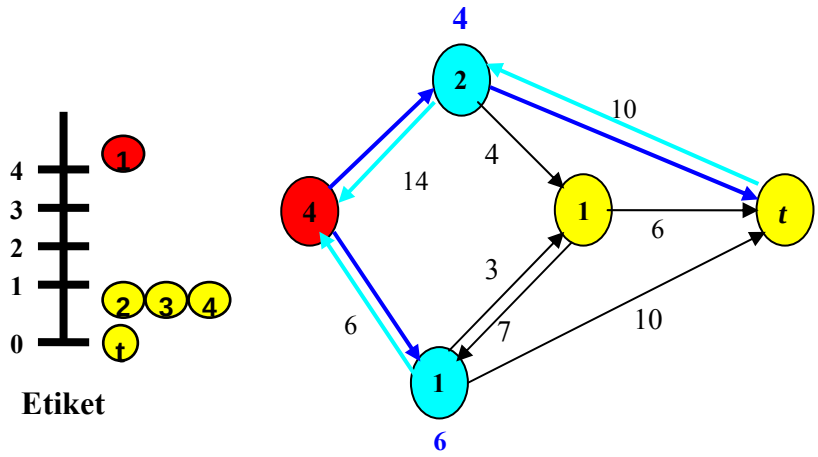
Şekil 2.32. Şekil 2.31 de verilen şebekenin uzaklık etiketleri ve şebekede gösterimi

Başlangıçta aktif olan tek düğüm kaynak düğümüdür. İlk önce kaynak düğümüne bağlı arklar doldurulur. Her bir doyuran göndermeden sonra kaynak düğümünün etiketi bir artar. Kaynağın etiketi 6 olur. Bu durumda şebekedeki göndermeler ve etiketler Şekil 2.33 te verilmiştir.



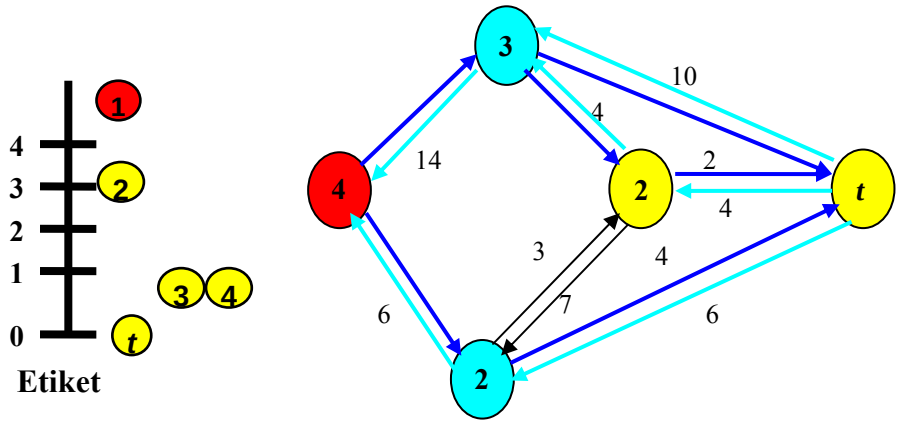
Şekil 2.33. Şekil 2.31 de verilen şebekenin kaynak düğümüne bağlı arklara yapılan doyuran göndermeler

Bir sonraki gönderme için etiketi en büyük olan aktif bir düğüm seçilir. 2 düğümü ve 3 düğümü aktif ve uzaklık etiketleri aynı olduğundan herhangi biri seçilebilir. 2 düğümü seçilirse bu düğümden sadece *t* kuyu düğümüne 10 birimlik gönderme yapılabilir. Bu gönderme Şekil 2.34 gösterilmektedir.



Şekil 2.34. Şebekede 2 düğümünden *t* kuyu düğümüne yapılan gönderme

Ve böyle devam edilerek ya fazlalıklar bitine kadar, ya da bütün arklar dolana kadar devam edilir. Şebekenin en son hali Şekil 2.35 te verilmiştir. Bu şebekede kaynak düğümünden kuyu düğümüne 20 birim akış yapılmıştır.



Şekil 2.35. Şekil 2.31 de verilen şebekede maksimum akış yapıldıktan sonra arta kalan şebeke

2.15 FAZLALIK ÖLÇEKLEME (EXCESS SCALING) ALGORİTMASI

Bu algoritmada Δ ölçek (scaling) parametresi $e_{\text{maks}} = \text{maks}\{e(i) : i \text{ aktif}\}$ nin bir üst sınırı olarak seçilir. Δ-ölçek aşamasında her j düğümü için $e(j) \leq \Delta$ dır. Gönderme $e(i) \geq \Delta/2 \geq e_{\text{maks}}/2$ fazlalığa sahip uzaklık etiketi en küçük olan düğümlerden başlar ve kuyu düğümüne doğru göndermeler yapılır. Akış öncesi gönderme algoritmasında i

düğümünden j düğümüne $\delta = \min(e(i), r_{ij})$ birim akış gönderilirken bu algoritmada $\delta = \min(e(i), r_{ij}, \Delta - e(j))$ birim akış gönderilir. Böylece şebekede Δ dan daha büyük fazlalığa sahip bir düğüm olmaz. $\Delta = 2^{\lceil \log_2 U \rceil}$ ile başlanır. $U \le \Delta \le 2U$ dur. Δ aşaması boyunca $\Delta / 2 < e_{\text{maks}} \le \Delta$ dır ve e_{maks} ın değeri artar ya da azalır. $e_{\text{maks}} < \Delta / 2$ olduğunda $\Delta / 2$ - ölçek aşamasının başlangıcına geçilmiş olur. En son ölçek aşaması 1-ölçek aşamasıdır, her j düğümü için $e(j) \le 1$ dir. 1- ölçek aşamasının sonunda optimal akışa ulaşmış oluruz. Algoritma $\lceil \log_2 U \rceil + 1$ tane ölçek aşamasından sonra e_{maks} ın değeri sıfır olur ve maksimum akışa ulaşılmış olur. Şekil 2.36 da bu fazlalık ölçekleme algoritması verilmiştir [2, 20].

Fazlalık Ölçekleme Algoritması;

begin
preprocess
$\Delta := 2^{\lceil \log_2 U \rceil}$;
while $\Delta \ge 1$ ***do***
begin
while şebekede $e(j) \ge \Delta / 2$ sağlayan bir j düğümü var ***do***
begin
$e(j) \ge \Delta / 2$ sağlayan bütün j düğümleri içinde minimum uzaklık etiketine sahip j düğümünü seç ve hiç bir düğümü fazlalığı Δ geçmeyecek şekilde gönderme yap, ya da j düğümünün etiketini yenile
end;
$\Delta := \Delta / 2$
end;
end;

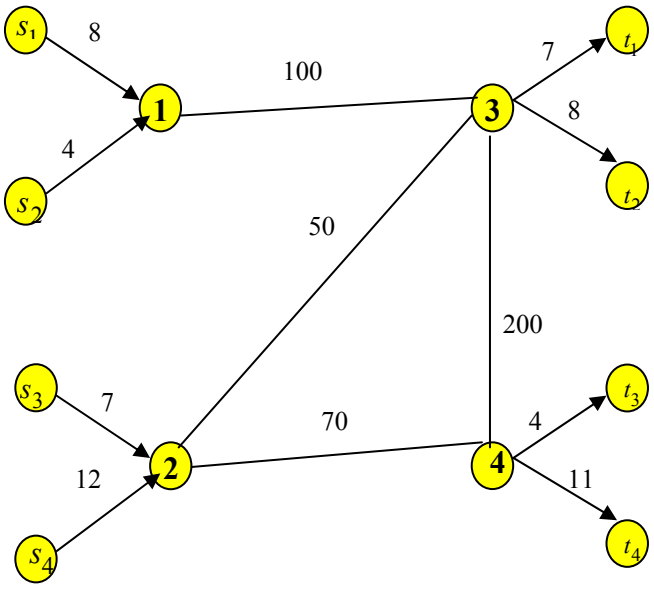
Şekil 2.36. Fazlalık ölçekleme algoritması

Teorem 2.11. Fazlalık ölçekleme algoritmasının karmaşıklığı $O\left(nm + n^2 \log_2 U\right)$ dir [2, 20].

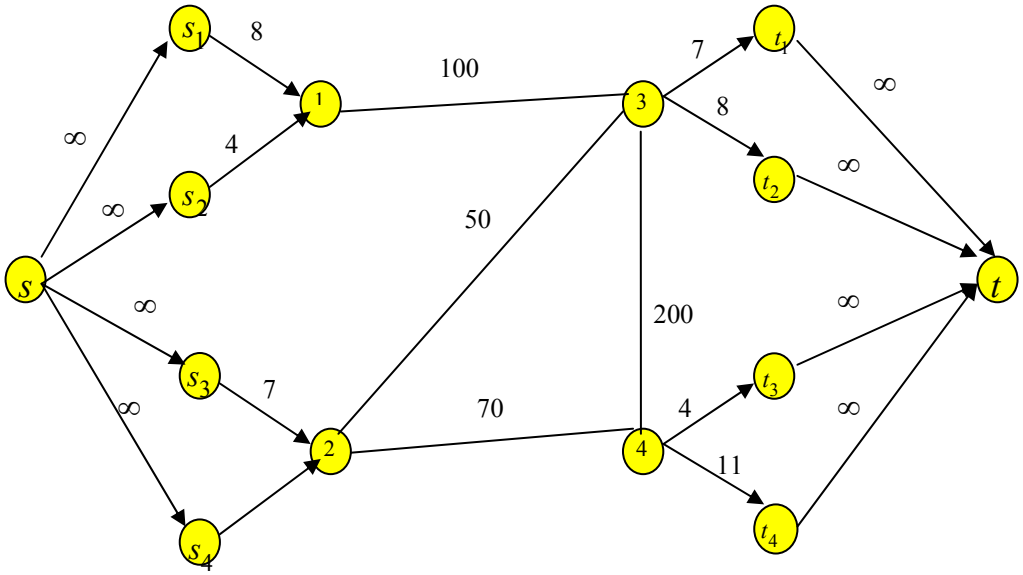
2.16 BİRDEN ÇOK KAYNAK DÜĞÜMÜ VE KUYU DÜĞÜMÜ OLAN ŞEBEKELERDE MAKSİMUM AKIŞIN HESAPLANMASI

Kaynak ya da kuyu sayısı birden fazla olan şebekelerde maksimum akışı bulmak için şebekeye yeni kaynak düğümü ve yeni kuyu düğümü eklenir. Şebekedeki kaynak düğümleri yeni kaynak düğümüne kuyu düğümleri yeni kuyu düğümüne yeteri kadar büyük kapasiteye sahip arklarla birleştirilir. Bu kapasiteler ∞ da alınabilir. Böylece şebeke bir kaynak düğümü ve bir kuyu düğümüne sahip bir şebekeye olur. Bu durumda şebekede bir kaynak düğümünden bir kuyu düğümüne gönderilebilecek maksimum akış problemine döner. Bu durumda herhangi bir algoritmayla maksimum akış problemi çözülür.

Örneğin Şekil 2.37 te verilen şebekede s_1, s_2, s_3, s_4 kaynak düğümleri t_1, t_2, t_3, t_4 kuyu düğümleri olsun. Arkların üzerindeki sayılar arkaların kapasitesi olsun. Kaynak düğümlerinden kuyu düğümlerine gönderilebilecek maksimum akışı bulmak için şebekeye yeni kaynak düğümü s ve yeni kuyu düğümü t eklenir. Yeni kaynak düğümü ile s_1, s_2, s_3, s_4 kaynak düğümleri aralarındaki ark kapasiteleri yeteri kadar büyük olacak şekilde aralarında arklar yerleştirilir. Benzer şekilde yeni kuyu düğümü de t_1, t_2, t_3, t_4 kuyu düğümleri ile aralarındaki ark kapasitesi yeteri kadar büyük olacak şekilde aralarına arklar yerleştirilir. Bu Şekil 2.38 de gösterilmektedir.



Şekil 2.37. Kaynak düğümleri ve kuyu düğümleri birden fazla olan şebeke



Şekil 2.38. Şekil 2.37 de verilen şebekeye yeni kaynak düğümü ve yeni kuyu düğümünün eklenmesi

3. MALZEME VE YÖNTEM

Bu tez çalışmasında maksimum akış probleminin bir uygulaması olarak büyük görüntü dosyalarının dağıtımı, saklanması ve geri çağırımı deneyi için bağımsız bir bilgisayar ağı ile birlikte INTEL Core2 Duo E8400 3GHZ işlemci ve 4GB 800MHz DDR2 RAM özelliklerine sahip ve SUSE 11.1 işletim sistemi altında çalışan iki sunucu bilgisayar kullanılmıştır. Dosyaların optimize boyutlarda parçalara ayrılması, Kısım 4.8 verilecek olan yöntemle diğer bilgisayarlara dağıtılması ve daha sonra bu parçaların birleştirilmesi ve geri çağırımı C bilgisayar programlama dilinde yazılan programlar ve NFS (Network File System) protokolü kullanılarak gerçekleştirilmiştir.

NFS (Network File System) bir bilgisayarın ağ üzerinde bulunan diğer bir bilgisayarın diskindeki bir bölümüne sanki kendi yerel diski gibi paylaştırarak, okuma ve yazma gibi işlemlerin hızlı ve kolayca yapılabilmesini sağlayan bir sistemdir. Ulaşılmak istenen bilgilerin bulunduğu bilgisayar sunucu, ulaşmak isteyen bilgisayar da istemci olarak adlandırılır. Paylaştırılan dosya sisteminde her bir dosyanın en az bir kopyası mevcuttur, bu durum veri depolamada esneklik ve tutarlılık sağlar. Dosyalara tüm ağdan son derece saydam bir şekilde ulaşılabilir. Ayrıca LINUX işletim sistemi altında TCP/IP protokol desteği ile çalışarak sistem yönetiminde kolaylık sağlar.

4. BULGULAR

Genel kısımlar bölümünde maksimum akış problemi ve bu problemin çözümü için algoritmalar verilmiştir. Bulgular bölümünde şebekeler Kısım 2.3 te tanımlandığı gibi düğüm – düğüm matris gösteriminin birinci gösterimi ile gösterilecektir. Bu gösterim maksimum akış probleminin çözümünde kolaylıklar sağlamaktadır. Matris gösterimi problemin bilgisayar program kodlamasının verimliliğini artırmakta ve daha etkin kullanımına yardımcı olmaktadır. Bulgular bölümünde genel kısımlar bölümünde verilen algoritmaların matris gösterimleri verilecektir. Ayrıca maksimum akış problemi için geliştirilen yeni bir algoritma bu bulgular bölümünde ayrıntılı ve karşılaştırmalı olarak verilecektir. Bulunan algoritma, kapasite ölçekleme algoritması ve en büyük etiketli akış öncesi gönderme algoritmasının C bilgisayar programlama dilinde kodları yazılarak karşılaştırmaları da yapılmıştır. Geliştirilen bu yeni algoritma ile bir uygulama alanı olarak bir bilgisayar şebekesi göz önüne alınmış büyük hacimli görüntü dosyalarının taşınması, saklanması ve geri çağırımı süreçleri için kullanılmıştır.

4.1. KAPASİTE ÖLÇEKLEME (CAPACITY SCALING) ALGORİTMASININ MATRİSLE GÖSTERİMİ

Kısım 2.11. de verilen kapasite ölçekleme algoritmasını matrisle gösteriminde başlangıçta matriste $\Delta = 2^{\lfloor \log_2 U \rfloor}$ sayısından küçük sayılar yerine sıfır yazılarak $G(x,\Delta)$ alt şebekesi elde edilir. Bu alt şebekede etiketleme algoritmasındaki gibi kuyu düğümü etiketlenemeyene kadar akış yapılır. Her bir akıştan sonra şebekedeki arta kalan kapasiteler belirlenir. $G(x,\Delta)$ şebekesinde akış yapılamadığında bir sonraki Δ aşaması olan $\Delta/2$ aşamasında geçilir. Bu aşamada şebekeyi gösteren matristeki elemanlar $\Delta/2$ den küçük olanların sıfır yapılarak $G(x, \Delta/2)$ alt şebekesinde akışlar yapılır. Aynı işlemler $\Delta = 1$ olana kadar devam ettirilir.

Burada kabul edilebilir yolu varlığı Kısım 2.10 da verilen etiketleme algoritmasının matrisle gösterimindeki gibi aranabilir. Ayrıca daha kullanışlı olan kabul edilebilir yol bularak da şebekede akış yapılabilir. Bunun için ya kaynak düğümünden kuyu

düğümüne ya da kuyu düğümünden kaynak düğümüne doğru hareket ederek kabul edilebilir yol aranır.

Kaynak düğümünden kuyu düğümüne yol aranırken önce matrisin birinci satırında sıfırdan farklı ilk sayının sütun indisi alınır, bu i_1 olsun. i_1. satırda sıfırdan farklı ilk elemana karşılık gelen sütun indisi i_2 olsun. i_2 satırda sıfırdan farklı ve sütun indisi i_1 den farklı olan ilk elemanın sütun indisi i_3 olsun. Böyle devam ederek i_k. satırda sütun indisleri $i_1, i_2, ..., i_{k-1}$ den farklı ve elemanları sıfırdan farklı bir eleman aranır. Eğer öyle bir eleman yoksa bir önceki i_{k-1} indisinden kuyu düğümüne gidecek kabul edilebilir bir yol yoktur, bu durumda i_{k-1} indisini değiştirilmesi gerekir. Bu da i_{k-2} satırında sütun indisi i_{k-1} den büyük sıfırdan farklı ve önceki sütun indisleri $i_1, i_2, ..., i_{k-3}$ ten farklı bir sütun indisine sahip elemanlar aranır. Eğer böyle bir eleman bulunamazsa bir önceki düğüme gidilerek aynı işlemler tekrarlanır. Kuyu düğümüne ulaşıldığında kabul edilebilir bir yol bulunmuş olur. $j \le n-2$ için $1 = i_0, i_1, i_2, ..., i_j = n$ bir yol olsun. Bu yoldan kaynak düğümünden kuyu düğümüne bu yol üzerinde bulunan arklardaki kapasitelerin minimumu $\delta = \min\{(i_k, i_{k+1}) : k = 0, ..., j-1\}$ kadar akış gönderilebilir. Akış yapıldıktan sonra şebeke yeniden düzenlenir. Eğer kaynak düğümünden kuyu düğümüne kabul edilebilir yol bulunamazsa maksimum akış elde edilmiş olur.

Kuyu düğümünden kaynak düğümüne kabul edilebilir bir yol bulmak için *n.* sütunda sıfırdan farklı ilk elamanın satır indisine bakılır. Bu indis j_1 olsun. j_1. sütunundaki elemanlardan sıfırdan farklı ilk elemanın satır indisi j_2 olsun. j_2 sütunundaki elemanlardan sıfırdan farklı ve satır indisi j_1 den farklı olan ilk elemanın satır indisi j_3 olsun. Böyle devam ederek j_k sütununda satır indisleri $j_1, j_2, ..., j_{k-1}$ den farklı ve sıfırdan farklı ilk eleman j_{k+1} olsun. Eğer böyle bir eleman yoksa j_{k-1} düğümünden kuyu düğümüne akış yapılamaz. Bu durumda bir önceki indis olan j_{k-2} indisli sütunda satır indisleri $j_1, j_2, ..., j_{k-3}$ den farklı ve satır indisi j_k dan büyük olan sıfırdan farklı elemanlara bakılır. Eğer sıfırdan farklı eleman yoksa bir önceki indise aynı işlemler

yapılır. Bu durumda sıfırdan farklı elemanlar sonlu adımda bulunmazsa kaynak düğümünden kuyu düğümüne akış yapabilecek kabul edilebilir bir yol yoktur. Eğer sıfırdan faklı elemanlar varsa en sonunda kuyu düğümüne ulaşılacaktır. $i \le n-2$ için $n = j_0, j_1, j_2, ..., j_i = 1$ bir yol olsun. Bu yoldan kaynak düğümünden kuyu düğümüne bu yol üzerinde bulunan arklardaki kapasitelerin minimumu, $\delta = \min\{(j_k, j_{k+1}) : k = 0, ..., i-1\}$ kadar akış gönderilebilir. Akış yapıldıktan sonra şebeke tekrar düzenlenir. Eğer kaynak düğümünden kuyu düğümüne kabul edilebilir yol bulunamazsa maksimum akış elde edilmiş olur.

Örnek 4.1. Örnek 2.2 de verilen şebekeyi kapasite ölçekleme algoritması ile çözelim. Bu şebekede U = 18, ve $\Delta = 2^{\lfloor \log U \rfloor} = 16$ dır. Şekil 4.1 de boş kapasiteler şebekesi Şekil 4.2 de $G(x, 16)$ alt şebekesi verilmiştir.

Düğüm	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
1	---	4	12	8	12	0	4	0
2	0	---	18	14	8	4	8	8
3	0	4	---	0	4	4	14	10
4	0	0	5	---	12	8	0	8
5	0	0	8	8	---	8	12	4
6	0	4	5	0	7	---	0	4
7	0	7	1	6	6	9	---	14
8	0	0	0	0	0	0	0	---

Şekil 4.1. Boş kapasitelerden oluşan bir şebeke

Düğüm	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
1	---	0	0	0	0	0	0	0
2	0	---	18	0	0	0	0	0
3	0	0	---	0	0	0	0	0
4	0	0	0	---	0	0	0	0
5	0	0	0	0	---	0	0	0
6	0	0	0	0	0	---	0	0
7	0	0	0	0	0	0	---	0
8	0	0	0	0	0	0	0	---

Şekil 4.2. Şekil 4.1 de verilen şebekenin $G(x, 16)$ alt şebekesi

Şekil 4.2 de görüldüğü gibi $G(x, 16)$ şebekesinde akış yapılamaz. Bu yüzden $\Delta / 2 = 8$ aşamasına geçilir. $G(x, 8)$ alt şebekesi Şekil 4.6 de verilmiştir.

Düğüm	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
1	---	0	12	8	12	0	0	0
2	0	---	18	14	8	0	8	8
3	0	0	---	0	0	0	14	10
4	0	0	0	---	12	8	0	8
5	0	0	8	8	---	8	12	0
6	0	0	0	0	0	---	0	0
7	0	0	0	0	0	9	---	14
8	0	0	0	0	0	0	0	---

Şekil 4.3. Şekil 4.1 de verilen şebekenin $G(x, 8)$ alt şebekesi

Kabul edilebilir yolu kuyu düğümünden kaynak düğümüne yöntemiyle bulmak için 8. sütundan sıfırda farklı ilk elemanın satır indisi 2 dir. Buradan 2. sütuna bakılır, sıfırdan farklı eleman olmadığından 8. sütunda satır indisi 2 den büyük sıfırdan farklı ilk elemanın satır indisi 3 tür. 3. sütunda sıfırdan farklı ilk elemanın satır indisi 1, yani kuyu düğümü olduğundan kuyu düğümü ile kaynak düğümü arasında kabul edilebilir bir yol bulunmuş olur. Bu yol 1, (1,3), 3, (3,8), 8 dir. Bu yoldan yapılabilecek maksimum akış, $\min(10,12) = 10$ birimdir. Bu akıştan sonra arta kalan şebeke ve $G(x,8)$ alt şebekesi hesaplanır. Şekil 4.4 bu akıştan sonra arta kalan şebeke, Şekil 4.5 da $G(x, 8)$ alt şebekesi verilmiştir.

Düğüm	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
1	---	4	2	8	12	0	4	0
2	0	---	18	14	8	4	8	8
3	10	4	---	0	4	4	14	0
4	0	0	5	---	12	8	0	8
5	0	0	8	8	---	8	12	4
6	0	4	5	0	7	---	0	4
7	0	7	1	6	6	9	---	14
8	0	0	10	0	0	0	0	---

Şekil 4.4. Birinci akıştan sonra arta kalan şebeke

Düğüm	1	2	3	4	5	6	7	8
1	---	0	0	8	12	0	0	0
2	0	---	18	14	8	0	8	8
3	10	0	---	0	0	0	14	0
4	0	0	0	---	12	8	0	8
5	0	0	8	8	---	8	12	0
6	0	0	0	0	0	---	0	0
7	0	0	0	0	0	9	---	14
8	0	0	10	0	0	0	0	---

Şekil 4.5. Birinci akıştan sonra $G(x, 8)$ alt şebekesi

Maksimum akışa ulaşmak için aynı işlemler kaynaktan kuyuya kabul edilebilir yol bulunmayana kadar tekrarlanır. Sonuçta 7 akışta toplam 24 arkta akış yapılarak bu şebekede 40 birimlik akış yapılır. Maksimum akış yapıldıktan sonra arta kalan şebeke Şekil 4.6 da verilmiştir. Etiketleme algoritmasında bu yöntemden 4 fazla adımda ve 16 fazla arkta akış yapılmıştı. Bu yöntem etiketleme algoritmasından daha az işlem yapılarak sonuca ulaşılmaktadır.

Düğüm	1	2	3	4	5	6	7	8
1	---	0	0	0	0	0	0	0
2	4	---	22	14	2	6	8	0
3	12	0	---	4	10	1	10	0
4	8	0	1	---	16	8	0	0
5	12	2	2	4	---	8	12	0
6	0	2	9	0	7	---	0	2
7	4	7	5	6	6	9	---	6
8	0	8	10	8	4	2	8	---

Şekil 4.6. Maksimum akıştan sonra arta kalan şebeke

4.2 GENEL AKIŞ ÖNCESİ GÖNDERME (GENERIC PREFLOW-PUSH) ALGORİTMASININ MATRİSLE GÖSTERİMİ

Kısım 2.13 te verilen genel akış öncesi gönderme algoritmasının matris gösteriminde matrise iki satır eklenir. Eklenen birinci satıra düğümlerin uzaklık etiketlerini ikinci satıra düğümlerdeki fazlalıklar yazılır. Şebekede ilk göndermeler kaynak düğümünden bitişik düğümlere akış göndermeleri yapılır. Şebekede bu göndermelerden sonra

kaynağın fazlalık satırına -(toplam göndermeler) olarak yazılır ve kaynağın uzaklık etiketi yenilenir. Bu aşamadan sonra aktif olan bir düğüm seçilir ve bu düğümden kuyu düğümüne doğru akış yapılır. i düğümünden j düğümüne α akış itilirse matrisin (i, j) elemanın değeri α azalır ve (j, i) elemanının değeri α artar. Ayrıca fazlalık satırında i sütununda fazlalık değeri α azalır j sütunundaki fazlalık değeri α artar. Bu akıştan sonra i düğümünün etiketi yenilenir. Algoritma düğümlerdeki fazlalıklar bitene ya da kabul edilebilir yollar bitene kadar gönderme ve etiket yenileme işlemleri tekrarlanırsa maksimum akış hesaplanmış olur.

Örnek 4.2. Örnek 4.1 de verilen şebekeyi akış öncesi gönderme algoritmasının matris gösterimi ile çözmek için matrise etiket ve fazlalık satırları eklenir. Bu matris Şekil 4.7 de gösterilmiştir. Başlangıçta düğümlerde fazlalık olmadığında düğümlerin fazlalıkları sıfırdır. Sadece kaynağın fazlalığı başlangıçta ∞ kabul edilir.

Düğüm	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
1	---	4	12	8	12	0	4	0
2	0	---	18	14	8	4	8	8
3	0	4	---	0	4	4	14	10
4	0	0	5	---	12	8	0	8
5	0	0	8	8	---	8	12	4
6	0	4	5	0	7	---	0	4
7	0	7	1	6	6	9	---	14
8	0	0	0	0	0	0	0	---
***d*(*i*)**	2	1	1	1	1	1	1	0
***e*(*i*)**	∞	0	0	0	0	0	0	0

Şekil 4.7. Örnek 4.1 şebekesinin akış öncesi gönderme algoritması ile çözümü için başlangıç gösterimi ve düğümlerin etiketleri

Şebekede ilk göndermeler kaynak düğümünden bitişik düğümlere yapılır. Bu göndermelerden sonra kaynakta fazlalık satırında -(toplam göndermeler) olarak yazılır ve kaynak düğümünün etiketi yenilenir. Şekil 4.8 de kaynaktan bitişik düğümlere yapılan göndermeler ve bunun sonucunda düğümlerdeki fazlalıklar gösterilmektedir. Örneğin 2 düğümünü incelersek, kaynaktan 2 düğümüne 4 birim akış gönderilebilir. Bu göndermeden sonra kaynak ile 2 düğümü arasındaki arta kalan kapasite $r_{12} = 0$ olur,

ters yöndeki arta kalan kapasite r_{21} 4 birim artar. Ayrıca 2 düğümündeki fazlalık 4 birim olur.

Düğüm	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
1	---	0	0	0	0	0	0	0
2	4	---	18	14	8	4	8	8
3	12	4	---	0	4	4	14	10
4	8	0	5	---	12	8	0	8
5	12	0	8	8	---	8	12	4
6	0	4	5	0	7	---	0	4
7	4	7	1	6	6	9	---	14
8	0	0	0	0	0	0	0	---
***d*(*i*)**	7	1	1	1	1	1	1	0
***e*(*i*)**	-40	4	12	8	12	0	4	0

Şekil 4.8. Kaynaktan bitişik düğümlere yapılan göndermelerin matriste gösterimi

Bu aşamadan sonra aktif olan bir düğüm seçilir ve bu düğümden kuyu düğümüne doğru akış yapılır. 2 düğümü seçilirse, 2 düğümünün fazlalığı 4 ve uzaklık etiketi 1 dir. Bu durumda 2 düğümünden kuyu düğümüne akış gönderilir. 2 düğümü ile kuyu düğümü arasındaki arta kalan kapasite $r_{12} = 8$ olduğundan bu düğümdeki tüm fazlalık kuyu düğümüne gönderilir ve 2 düğümünün fazlalığı sıfır olur. Kuyu düğümündeki fazlalık 4 olur. Bu gönderim Şekil 4.9 da gösterilmektedir.

Düğüm	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
1	---	0	0	0	0	0	0	0
2	4	---	18	14	8	4	8	4
3	12	4	---	0	4	4	14	10
4	8	0	5	---	12	8	0	8
5	12	0	8	8	---	8	12	4
6	0	4	5	0	7	---	0	4
7	4	7	1	6	6	9	---	14
8	0	4	0	0	0	0	0	---
***d*(*i*)**	7	1	1	1	1	1	1	0
***e*(*i*)**	-40	0	12	8	12	0	4	4

Şekil 4.9. 2 düğümünden kuyu düğümüne yapılan göndermenin matriste gösterimi

Bu aşamadan sonra 3 düğümü seçilirse kuyu düğümüne $\min(r_{38}, e(3)) = \min(10,12) = 10$ akış yapılabilir. Bu akıştan sonra 3 düğümü ile kuyu düğümü arasında

arta kalan boş kapasite $r_{38} = 0$ olduğundan 3 düğümünün uzaklık etiketi 1 attırılır. Bu işlemler Şekil 4.10 da gösterilmiştir.

Düğüm	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
1	---	0	0	0	0	0	0	0
2	4	---	18	14	8	4	8	4
3	12	4	---	0	4	4	14	0
4	8	0	5	---	12	8	0	8
5	12	0	8	8	---	8	12	4
6	0	4	5	0	7	---	0	4
7	4	7	1	6	6	9	---	14
8	0	4	10	0	0	0	0	---
***d*(*i*)**	7	1	2	1	1	1	1	0
***e*(*i*)**	-40	0	2	8	12	0	4	14

Şekil 4.10. 3 düğümünden kuyu düğümüne yapılan gönderme ve 3 düğümünün etiketinin yenilenmesi

Algoritma düğümlerdeki fazlalıklar bitene ya da kabul edilebilir yollar bitene kadar gönderme ve etiket yenileme işlemleri yapılırsa sonuçta Şekil 4.11 verilen şebeke elde edilir.

Düğüm	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
1	---	0	0	0	0	0	0	0
2	4	---	20	14	8	8	6	0
3	12	2	---	0	4	4	14	0
4	8	0	5	---	12	8	0	0
5	12	0	8	8	---	0	12	0
6	0	0	5	0	15	---	0	0
7	4	9	1	6	6	9	---	8
8	0	8	10	8	4	4	2	---
***d*(*i*)**	7	2	2	2	2	3	1	0
***e*(*i*)**	-40	0	0	0	0	0	0	40

Şekil 4.11. Şebekenin akış öncesi gönderme algoritması ile maksimum akıştan sonra arta kalan şebeke, düğümlerin etiketleri ve düğümlerdeki fazlalıklar.

Bu algoritmada maksimum akış için 18 gönderme ve 11 etiket yenileme işlemi yapılmıştır.

4.3. EN BÜYÜK ETİKETLİ AKIŞ ÖNCESİ GÖNDERME (HIGHEST LABEL PREFLOW-PUSH) ALGORİTMASININ MATRİSLE GÖSTERİMİ

Kısım 2.14 te verilen en büyük etiketli akış öncesi gönderme algoritmasının matrisle gösteriminde ilk aşamada genel akış öncesi gönderme algoritmasında olduğu gibi matrise etiket ve fazlalık satırları eklenir. Şebekedeki ilk göndermeler kaynak düğümünden bitişik düğümlere yapılır. Şebekede bu göndermelerden sonra kaynağın fazlalık satırına -(toplam göndermeler) olarak yazılır ve kaynağın uzaklık etiketi yenilenir. Bu aşamadan sonra yapılacak bütün göndermeler, $h^* = \text{maks}\{d(i) : i \text{ aktif düğüm}\}$ olarak tanımlanırsa algoritma etiketi h^* olan düğümlerden uzaklık etiketi h^*-1, buradaki düğümlerden uzaklık etiketi h^*-2 olan düğümlere, şeklinde devam ederek aktif düğümler bitene kadar ya da etiket yenilemeye kadar akış göndermeleri yapılır. Etiket yenileme yapılırsa algoritmada aynı işlemler tekrarlanır. i düğümünden j düğümüne α akış göndermesi yapılırsa matrisin (i, j) elemanın değeri α azalır ve (j, i) elemanının değeri α artar. Ayrıca fazlalık satırında i sütununda fazlalık değeri α azalır, j sütunundaki fazlalık değeri α artar. Bu akıştan sonra i düğümünün etiketi yenilenir. Eğer algoritma n ardışık işlemde etiketleme yapamadığı zaman bütün fazlalıkları kuyu düğümüne ulaşmış ve algoritma sonlanmış olur.

Örnek 4.3. Örnek 4.1 de verilen şebekede maksimum akışı en büyük etiketli akış öncesi gönderme algoritması ile çözümünde ilk aşamada genel akış öncesi gönderme algoritmasında olduğu gibi matrise etiket ve fazlalık satırları eklenir. Düğümlerin etiketlerinin olduğu başlangıç matrisi Şekil 4.12 de gösterilmiştir. Şebekede ilk göndermeler kaynaktan bitişik düğümlere yapılır ve bu göndermelerden sonra kaynağın uzaklık etiketi yenilenir. Bu göndermelerden sonra matriste arklarda arta kalan boş kapasiteler düzenlenir. Bu aşama Şekil 4.13 te gösterilmiştir.

Düğüm	1	2	3	4	5	6	7	8
1	---	4	12	8	12	0	4	0
2	0	---	18	14	8	4	8	8
3	0	4	---	0	4	4	14	10
4	0	0	5	---	12	8	0	8
5	0	0	8	8	---	8	12	4
6	0	4	5	0	7	---	0	4
7	0	7	1	6	6	9	---	14
8	0	0	0	0	0	0	0	---
***d*(*i*)**	2	1	1	1	1	1	1	0
***e*(*i*)**	∞	0	0	0	0	0	0	0

Şekil 4.12. Örnek 4.1 de verilen şebekenin en büyük etiketli akış öncesi gönderme algoritması ile çözümü için başlangıç gösterimi

Düğüm	1	2	3	4	5	6	7	8
1	---	0	0	0	0	0	0	0
2	4	---	18	14	8	4	8	8
3	12	4	---	0	4	4	14	10
4	8	0	5	---	12	8	0	8
5	12	0	8	8	---	8	12	4
6	0	4	5	0	7	---	0	4
7	4	7	1	6	6	9	---	14
8	0	0	0	0	0	0	0	---
***d*(*i*)**	7	1	1	1	1	1	1	0
***e*(*i*)**	-40	4	12	8	12	0	4	0

Şekil 4.13. Kaynak düğümünden bitişik düğümlere yapılan göndermelerin matriste gösterimi

Bu aşamadan sonra uzaklık etiketi en büyük olan fazlalığa sahip olan düğümden kuyu düğümüne doğru akış yapılır. 2, 3, 4, 5, 7 aktif düğümler ve hepsinin uzaklık etiketleri 1 dir. Bu düğümlerden herhangi birisi seçilebilir. 2 düğümü seçilirse, 2 düğümünden uzaklık etiketi 1 olduğundan 2 düğümünden kaynak düğümüne gönderme yapılır. Bu gönderme Şekil 4.14 te gösterilmektedir.

Düğüm	1	2	3	4	5	6	7	8
1	---	0	0	0	0	0	0	0
2	4	---	18	14	8	4	8	4
3	12	4	---	0	4	4	14	10
4	8	0	5	---	12	8	0	8
5	12	0	8	8	---	8	12	4
6	0	4	5	0	7	---	0	4
7	4	7	1	6	6	9	---	14
8	0	4	0	0	0	0	0	---
***d*(*i*)**	7	1	1	1	1	1	1	0
***e*(*i*)**	-40	0	12	8	12	0	4	4

Şekil 4.14. 2 düğümünden kuyu düğümüne yapılan göndermenin matriste gösterimi

Bu göndermeden sonra aktif düğümlerden uzaklık etiketi en büyük düğüm seçilir. Aktif düğümler aynı uzaklık etiketlerine sahip olduklarından bu düğümlerden herhangi biri seçilebilir. 3 düğümü seçilirse, 3 düğümünün uzaklık etiketi 1 olduğundan, 3 düğümünden kuyuya $\min(r_{38}, e(3)) = \min(10,12) = 10$ birim gönderme yapılır. Bu göndermeden sonra 3 düğümünün uzaklık etiketi yenilenir. Bu işlemler Şekil 4.15 te gösterilmiştir.

Düğüm	1	2	3	4	5	6	7	8
1	---	0	0	0	0	0	0	0
2	4	---	18	14	8	4	8	4
3	12	4	---	0	4	4	14	0
4	8	0	5	---	12	8	0	8
5	12	0	8	8	---	8	12	4
6	0	4	5	0	7	---	0	4
7	4	7	1	6	6	9	---	14
8	0	4	10	0	0	0	0	---
***d*(*i*)**	7	1	2	1	1	1	1	0
***e*(*i*)**	-40	0	2	8	12	0	4	14

Şekil 4.15. 3 düğümünden yapılan gönderme ve 3 düğümünün uzaklık etiketinin yenilenmesi

Bu göndermeden sonra etiketi en büyük aktif düğüm 3 düğümüdür. Bu düğümden uzaklık etiketi 3 düğümünün uzaklık etiketinden 1 küçük olan ve 3 düğümü ile aralarından kabul edilebilir ark olan bir düğüme gönderme yapılır. Bu düğümler 2, 4, 5,

6, 7 dir. Bunlardan herhangi biri seçilebilir. 2 düğümü seçilirse, yapılan gönderme Şekil 4.16 da gösterilmektedir.

Düğüm	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
1	---	0	0	0	0	0	0	0
2	4	---	20	14	8	4	8	4
3	12	2	---	0	4	4	14	0
4	8	0	5	---	12	8	0	8
5	12	0	8	8	---	8	12	4
6	0	4	5	0	7	---	0	4
7	4	7	1	6	6	9	---	14
8	0	4	10	0	0	0	0	---
***d*(*i*)**	7	1	2	1	1	1	1	0
***e*(*i*)**	-40	2	0	8	12	0	4	14

Şekil 4.16. 3 düğümünden 2 düğümüne yapılan gönderme

Maksimum akışı bulmak için düğümlerdeki fazlalıklar bitene kadar ya da kuyuya giden arklar doyana aynı işlemler tekrarlanır ve sonuçta Şekil 4.17 de verilen şebeke elde edilir.

Düğüm	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
1	---	0	0	0	0	0	0	0
2	4	---	20	14	8	4	8	2
3	12	2	---	0	4	4	14	0
4	8	0	5	---	12	8	0	0
5	12	0	8	8	---	8	4	0
6	0	4	5	0	7	---	0	4
7	4	7	1	6	14	9	---	2
8	0	6	10	8	4	0	12	---
***d*(*i*)**	7	1	2	2	2	1	1	0
***e*(*i*)**	-40	0	0	0	0	0	0	40

Şekil 4.17.Şebekenin en büyük etiketli akış öncesi gönderme algoritması ile maksimum akıştan sonra arta kalan şebeke, düğümlerin etiketleri ve düğümlerdeki fazlalıklar.

Bu algoritmada toplam 13 gönderme ve 8 etiket yenileme işleminden sonra maksimum akış elde edilmiştir.

4.4. FAZLALIK ÖLÇEKLEME (EXCESS SCALING) ALGORİTMASININ MATRİSLE GÖSTERİMİ

Kısım 2.15. te verilen fazlalık ölçekleme algoritmasını matrisle gösteriminde genel akış öncesi gönderme algoritmasındaki gibi başlangıç matrisine uzaklık etiketi ve fazlalık etiketleri satırları eklenir. $\Delta = 2^{\lceil \log_2 U \rceil}$ olmak üzere şebekede gönderme $e(i) \geq \Delta/2 \geq e_{\text{maks}}/2$ fazlalığa sahip uzaklık etiketi en küçük olan düğümden fazlalığı $\Delta/2$ den küçük ve uzaklık etiketi daha küçük olan düğüme $\delta = \min(e(i), r_{ij}, \Delta - e(j))$ birim akış yapılır. i düğümünden j düğümüne δ birim akış göndermesi yapılırsa matrisin (i, j) elemanı δ eksilir, (j, i) elamanı δ artar, $e(i)$ δ eksilir, $e(j)$ δ artar. $e_{\text{maks}} < \Delta/2$ olduğunda $\Delta/2$ - ölçek aşamasının başlangıcına geçilmiş olur. Her j için $e(j) \leq 1$ olduğunda, en son ölçek aşaması 1-ölçek aşamasının sonunda şebekede kaynak düğümünden kuyu düğümüne yapılabilecek maksimum akış hesaplanmış olur.

Örnek 4.4. Örnek 4.1 de verilen şebekede maksimum akışı fazlalık ölçekleme algoritmasının matris gösterimi ile çözümünde önce Şekil 4.18 de başlangıç matrisi ve düğümlerin uzaklık etiketleri verilmiştir.

Düğüm	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
1	---	4	12	8	12	0	4	0
2	0	---	18	14	8	4	8	8
3	0	4	---	0	4	4	14	10
4	0	0	5	---	12	8	0	8
5	0	0	8	8	---	8	12	4
6	0	4	5	0	7	---	0	4
7	0	7	1	6	6	9	---	14
8	0	0	0	0	0	0	0	---
***d*(*i*)**	2	1	1	1	1	1	1	0
***e*(*i*)**	∞	0	0	0	0	0	0	0

Şekil 4.18. Örnek 4.1 de verilen şebekenin fazlalık ölçekleme algoritması için başlangıç matrisi

Δ-ölçek aşaması $\Delta = 2^{\lceil \log_2 18 \rceil} = 2^5 = 32$ olduğundan fazlalığı 32 den büyük olan düğümler içinden etiketi en küçük düğüm seçilir. Başlangıçta sadece kaynak düğümünde fazlalık olduğundan bitişik düğüme akış yapılır. Kaynaktan çıkan arkların

kapasiteleri 32 den küçük olduğundan 32-ölçek aşamasında kaynağa komşu düğümlere akış göndermeleri yapılır. Bu göndermeler Şekil 4.19 da gösterilmektedir.

Düğüm	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
1	---	0	0	0	0	0	0	0
2	4	---	18	14	8	4	8	8
3	12	4	---	0	4	4	14	10
4	8	0	5	---	12	8	0	8
5	12	0	8	8	---	8	12	4
6	0	4	5	0	7	---	0	4
7	4	7	1	6	6	9	---	14
8	0	0	0	0	0	0	0	---
***d*(*i*)**	7	1	1	1	1	1	1	0
***e*(*i*)**	-40	4	12	8	12	0	4	0

Şekil 4.19. Kaynak düğümünden bitişik düğümlere yapılan göndermelerin matriste gösterimi

Fakat fazlalığı 16 dan büyük düğüm olmadığından bir sonraki aşamaya $\Delta/2=16$-ölçek aşamasına geçilir. 3, 4, 5 düğümlerinin sahip oldukları fazlalık 8 den büyüktür ve uzaklık etiketleri 1 dir. Bu durumda bu düğümlerden herhangi biri seçilebilir. 3 düğümü seçilirse 3 düğümünün uzaklık etiketi 1 olduğundan $\delta=\min(e(3), r_{38}, \Delta-e(8))=$ $=\min(12,10,16-0)=8$ birim akış 3 düğümünden kuyu düğümüne yapılır. Bu göndermeden sonra şebeke yeniden düzenlenirse Şekil 4.20 deki şebeke elde edilir.

Düğüm	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
1	---	0	0	0	0	0	0	0
2	4	---	18	14	8	4	8	8
3	12	4	---	0	4	4	14	2
4	8	0	5	---	12	8	0	8
5	12	0	8	8	---	8	12	4
6	0	4	5	0	7	---	0	4
7	4	7	1	6	6	9	---	14
8	0	0	8	0	0	0	0	---
***d*(*i*)**	7	1	1	1	1	1	1	0
***e*(*i*)**	-40	4	4	8	12	0	4	8

Şekil 4.20. 3 düğümünden kuyu düğümüne yapılan gönderme

Bu göndermeden sonra fazlalığı 8 den büyük olan düğümlerden uzaklık etiketi en küçük düğüm seçilir. 4 ve 5 düğümlerinin uzaklık etiketleri 1 ve sahip oldukları fazlalık 8 den

büyük olduğundan bu düğümlerden herhangi biri seçilebilir. 4 düğümü seçilirse, uzaklık etiketi 1 olduğundan $\delta = \min(e(4),\ r_{48},\ \Delta - e(8)) = \min(8, 8, 16-0) = 8$ birimlik akış 4 düğümünden kuyu düğümüne gönderilir. Bu göndermeden sonra şebeke yeniden düzenlenirse Şekil 4.21 deki şebeke elde edilir.

Düğüm	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
1	---	0	0	0	0	0	0	0
2	4	---	18	14	8	4	8	8
3	12	4	---	0	4	4	14	2
4	8	0	5	---	12	8	0	0
5	12	0	8	8	---	8	12	4
6	0	4	5	0	7	---	0	4
7	4	7	1	6	6	9	---	14
8	0	0	8	8	0	0	0	---
***d*(*i*)**	7	1	1	1	1	1	1	0
***e*(*i*)**	-40	4	4	0	12	0	4	16

Şekil 4.21. 4 düğümünden kuyu düğümüne yapılan gönderme

Bu göndermeden sonra 8 den büyük fazlalığa sahip düğümlerden uzaklık etiketi en küçük düğüm seçilir. Bu kurala uyan tek düğüm 5 düğümüdür. 5 düğümünün uzaklık etiketi 1 olduğundan $\delta = \min(e(5),\ r_{58},\ \Delta - e(8)) = \min(12, 4, 16-0) = 4$ birimlik akış 5 düğümünden kuyu düğümüne gönderilir. Bu göndermeden sonra 5 düğümü ile kuyu düğümü arasındaki arkta boş kapasite sıfır olduğundan 5 düğümünün uzaklık etiketi 1 artırılır ve 2 olur. Bu göndermeden sonra şebeke yeniden düzenlenirse Şekil 4.22 deki şebeke elde edilir.

Düğüm	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
1	---	0	0	0	0	0	0	0
2	4	---	18	14	8	4	8	8
3	12	4	---	0	4	4	14	2
4	8	0	5	---	12	8	0	0
5	12	0	8	8	---	8	12	0
6	0	4	5	0	7	---	0	4
7	4	7	1	6	6	9	---	14
8	0	0	8	8	4	0	0	---
***d*(*i*)**	7	1	1	1	2	1	1	0
***e*(*i*)**	-40	4	4	0	8	0	4	20

Şekil 4.22. 5 düğümünden kuyu düğümüne yapılan gönderme ve 5 düğümünün uzaklık etiketinin yenilenmesi

Bu göndermeden sonra 8 den büyük fazlalığa sahip düğümlerden uzaklık etiketi en küçük düğüm seçilir. Bu kurala uyan tek düğüm 5 düğümdür. 5 düğümünün uzaklık etiketi 2 olduğundan 5 düğümünden uzaklık etiketi 1 olan ve 5 düğümü ile aralarında kabul edilebilir ark olan düğümlere akış yapılır. 3, 4, 6, 7 düğümlerinin uzaklık etiketleri 1 ve 5 düğümü ile aralarında kabul edilebilir ark vardır. Bu düğümlerden herhangi biri seçilebilir. 3 düğümü seçilirse $\delta = \min(e(5), r_{53}, \Delta - e(3)) =$ $= \min(8, 8, 8-4) = 4$ birimlik akış 5 düğümünden 3 düğümüne gönderilir. Bu göndermeden sonra şebeke yeniden düzenlenirse Şekil 4.23 teki şebeke elde edilir.

Düğüm	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
1	---	0	0	0	0	0	0	0
2	4	---	18	14	8	4	8	8
3	12	4	---	0	8	4	14	2
4	8	0	5	---	12	8	0	0
5	12	0	4	8	---	8	12	0
6	0	4	5	0	7	---	0	4
7	4	7	1	6	6	9	---	14
8	0	0	8	8	4	0	0	---
***d*(*i*)**	7	1	1	1	2	1	1	0
***e*(*i*)**	-40	4	8	0	4	0	4	20

Şekil 4.23. 5 düğümünden 3 düğümüne yapılan gönderme

Bu göndermeden sonra 8 den büyük fazlalığa sahip tek düğüm 3 düğümüdür. 3 düğümünün uzaklık etiketi 1 olduğundan $\delta = \min(e(3), r_{38}, \Delta - e(8)) =$ $= \min(8, 2, 8-0) = 2$ birimlik akış 3 düğümünden kuyu düğümüne gönderilir. Bu akıştan sonra 3 düğümü ile kuyu düğümü arasındaki arkta arta kalan kapasite sıfır olduğundan 3 düğümünün etiketi 1 arttırılır. Bu göndermeden sonra Şekil 4.24 te verilen şebeke elde edilir.

Düğüm	1	2	3	4	5	6	7	8
1	---	0	0	0	0	0	0	0
2	4	---	18	14	8	4	8	8
3	12	4	---	0	8	4	14	0
4	8	0	5	---	12	8	0	0
5	12	0	4	8	---	8	12	0
6	0	4	5	0	7	---	0	4
7	4	7	1	6	6	9	---	14
8	0	0	10	8	0	0	0	---
***d*(*i*)**	7	1	2	1	2	1	1	0
***e*(*i*)**	-40	4	6	0	4	0	4	22

Şekil 4.24. 3 düğümünden kuyu düğümüne yapılan gönderme ve 3 düğümünün uzaklık etiketinin yenilenmesi

Bu göndermeden sonra 8 den büyük fazlalığa sahip düğüm kalmadığından bir sonraki Δ ölçek aşaması olan 8-ölçek aşamasına geçilir. Aynı işlemler tekrarlanırsa maksimum akışa 18 gönderme ve 10 etiket yenileme işleminden sonra ulaşılır. Maksimum akış yapıldıktan sonra arklardaki arta kalan boş kapasiteler Şekil 4.25 te verilmiştir.

Düğüm	1	2	3	4	5	6	7	8
1	---	0	0	0	0	0	0	0
2	4	---	22	14	8	4	8	0
3	12	0	---	0	8	4	12	0
4	8	0	5	---	12	8	0	0
5	12	0	4	8	---	4	12	0
6	0	4	5	0	11	---	0	0
7	4	9	1	6	6	9	---	8
8	0	8	10	8	4	4	6	---
***d*(*i*)**	7	2	2	2	2	2	1	0
***e*(*i*)**	-40	0	0	0	0	0	0	40

Şekil 4.25. Şebekenin kapasite ölçekleme algoritması ile maksimum akıştan sonra arta kalan şebekesi, düğümlerin uzaklık etiketleri ve düğümlerdeki fazlalıklar.

4.5. GELİŞTİRİLEN ALGORİTMA

Bu algoritma bir geliştirilmiş yol algoritmasıdır. Kapasite ölçekleme algoritması gibi Δ ölçeği kullanılır, fakat geliştirilmiş yol farklı bir şekilde bulunur. Bu algoritmada şebekeler düğüm - düğüm matris birinci gösterimi ile gösterilecektir. Şebekedeki

arkların en büyük kapasitesi U olmak üzere Δ nın ilk değeri $\Delta = 2^{\lfloor \log_2 U \rfloor}$ alınır. Verilen şebeke matrisinde başlangıçta Δ sayısından küçük sayılar yerine sıfır yazılarak $G(x,\Delta)$ alt şebekesi elde edilir. $G(x,\Delta)$ alt şebekesinin matris gösteriminde matrise bir satır eklenir ve bu satıra $G(x,\Delta)$ alt şebekesinde düğümlerin uzaklık etiketleri yazılır. Kapasite ölçekleme algoritmasında olduğu gibi $G(x,\Delta)$ da kabul edilebilir yollardan kaynak düğümünden kuyu düğümüne akışlar yapılır. Kaynak düğümünü 1 ile ve kuyu düğümünü n ile gösterilirse, kaynak düğümü etiketlenebiliyorsa, yani $0 < d(1) < n$ ise, 1 kaynak düğümünden n kuyu düğümüne kabul edilebilir bir yol vardır. Eğer 1 kaynak düğümü etiketlenemiyorsa 1 kaynak düğümünden n kuyu düğümüne kabul edilebilir yol yoktur, bu durumda bir sonraki aşama olan $\Delta/2$ aşamasına geçilir. Benzer şekilde devam edilerek Δ nın sürekli yarısı alınarak en son aşama olan $\Delta = 1$ aşamasının sonunda maksimum akışa ulaşılmış olur.

$G(x,\Delta)$ alt şebekesinde 1 kaynak düğümünden n kuyu düğümüne geliştirilmiş yol bulunması:

1. Adım: $G(x,\Delta)$ alt şebekesinde kaynak düğümünden kuyu düğümüne kabul edilebilir bir geliştirilmiş yol bulmak için birinci adımda satır indisleri uzaklık etiketi 1 olan düğümler, sütun indisi kaynak düğümü olan bir matris düzenlenir. Bu matrisin (i, j) elemanları uzaklık etiketi 1 olan düğümler ile kuyu düğümü arasındaki arkların boş kapasiteleri olan bir matris düzenlenir. Ayrıca bu matrise Kapasite ve Etiket sütunları eklenir. Birinci adımda Kapasite sütununda satırda bulunun düğümden kuyu düğümüne yapılabilecek maksimum akış miktarı, yani uzaklık etiketi 1 olan düğümler ile kuyu düğümü arasındaki arkların boş kapasiteleri yazılır. Şekil 4.26 da bu matris genel olarak gösterilmiştir. Şekil 4.26 da uzaklık etiketi 1 olan düğümler $i_1, i_2, ..., i_k$ ile gösterilmiştir.

Düğüm	n	Kapasite	Etiket
i_1	$r_{i_1,n}$	$r_{i_1,n}$	n
i_2	$r_{i_2,n}$	$r_{i_2,n}$	n
$\vdots$	$\vdots$	$\vdots$	$\vdots$
i_k	$r_{i_k,n}$	$r_{i_k,n}$	n

Şekil 4.26. Geliştirilmiş yol bulmada 1. adım matrisi

2. Adım: İkinci adımda uzaklık etiketi 2 olan her bir düğümden kuyu düğümüne maksimum akış yapılabilecek yollar bulunur. Bunun için satır indisleri uzaklık etiketi 2 olan düğümler, sütun indisleri uzaklık etiketi 1 olan düğümler olan bir matris düzenlenir. Matrisin (i, j) ara elamanlarına satırda bulunan i düğümünden sütunda bulunan j düğümüne giden arkın boş kapasitesi ile bir önceki adımda bulunan j düğümünden kuyuya gönderilebilecek maksimum akışın minimumu yazılır. Bu matrise Kapasite ve Etiket sütunları eklenir. Her bir satır için satırda bulunan elemanların maksimumu alınır ve bu maksimum Kapasite sütununda yazılır, bu maksimum değerin bulunduğu yerin sütun indisi Etiket sütununa yazılır. Kapasite sütununda bulunan sayı satır indisi olan düğümden kuyu düğümüne giden maksimum kapasiteli yolun kapasitesini, Etiket sütunundaki bulunan sayı ise bu maksimum akışın yapılabileceği yoldaki satır indisi olan düğümden bir sonra gelen düğümü gösterir. Genel olarak bu matris Şekil 4.27 de verilmiştir. Birinci adımda uzaklık etiketi 1 olan düğümler $i_1, i_2, ..., i_k$ ile gösterilmişti. Bu adımda uzaklık etiketi 2 olan düğümler $j_1, j_2, ..., j_h$ ile gösterilmiştir. Örneğin Şekil 4.27 deki matriste j_1 satırını incelersek, j_1 düğümünden kuyu düğümüne yapılabilecek olan maksimum akış $\underset{1\le l\le k}{\text{maks}}\left\{\min(r_{i_l,n},\ r_{j_1,i_l})\right\}$ dir, bu akışın yapılabileceği yolda j_1 düğümünden sonra gelen düğüm γ_{j_1} düğümüdür.

Düğüm	i_1	⋯	i_k	Kapasite	Etiket
j_1	$\min(r_{i_1,n}, r_{j_1,i_1})$	⋯	$\min(r_{i_k,n}, r_{j_1,i_k})$	$\underset{1\le l\le k}{\text{maks}}\left\{\min(r_{i_l,n}, r_{j_1,i_l})\right\}$	γ_{j_1}
⋮	⋮	⋮	⋮	⋮	⋮
j_h	$\min(r_{i_1,n}, r_{j_h,i_1})$	⋯	$\min(r_{i_k,n}, r_{j_h,i_k})$	$\underset{1\le l\le k}{\text{maks}}\left\{\min(r_{i_l,n}, r_{j_h,i_l})\right\}$	γ_{j_h}

Şekil 4.27. Geliştirilmiş yol bulmada 2. adım matrisi

Benzer şekilde devam edilerek $1 \le u \le d(1)$ olmak üzere u. adıma gelinir.

***u*. Adım:** Bu adımda uzaklık etiketi u olan düğümlerden kuyu düğümüne yapılabilecek olan maksimum akış ve bu akışın yapılabileceği yollar bulunur. Bunun için satır indisleri uzaklık etiketi u olan düğümler, sütun indisleri uzaklık etiketleri (u-1) olan düğümler olan bir matris oluşturulur. (u-1). adımda uzaklık etiketi (u-1) olan

düğümlerden kuyu düğümüne yapılabilecek maksimum akış miktarı ve bu düğümlerden kuyuya giden kabul edilebilir yollar bulunmuştu. Matrisin her (i, j) ara elemanı satırda bulunan i düğümünden sütunda bulunan j düğümüne giden arkın boş kapasitesi ile (u-1). adımda bulunan j düğümünden kuyuya giden maksimum kapasiteli yolun kapasitesinin minimumu yazılır. Kapasite sütununda ise satırdaki elemanların maksimumu, maksimumun bulunduğu yerin sütun indisi ise Etiket sütununa yazılır. Şekil 4.28 de bu adım genel olarak verilmiştir. Şekil 4.28 de verilen matriste uzaklık etiketi u olan düğümler $k_1, k_2, ..., k_j$ ile, uzaklık etiketi (u-1) olan düğümler $h_1, h_2, ..., h_i$ ile , uzaklık etiketi (u-1) olan düğümlerden kuyuya giden maksimum kapasiteli yolların kapasiteleri ise $z_{h_1}, z_{h_2}, ..., z_{h_i}$ ile gösterilmiştir. Örneğin Şekil 4.28 deki matrisin k_1. satırını incelersek, k_1 düğümünden kuyu düğümüne yapılabilecek olan maksimum akış $\underset{1\le l\le i}{\text{maks}}\left\{\min(r_{k_1,h_l}, z_{h_l})\right\}$ dir, bu akışın yapılabileceği yolda k_1 düğümünden sonra gelen düğüm γ_{k_1} düğümüdür.

Düğüm	h_1	…	h_i	Kapasite	Etiket
k_1	$\min(r_{k_1,h_1}, z_{h_1})$	…	$\min(r_{k_1,h_i}, z_{h_i})$	$\underset{1\le l\le i}{\text{maks}}\left\{\min(r_{k_1,h_l}, z_{h_l})\right\}$	γ_{k_1}
⋮	⋮	⋮	⋮	⋮	⋮
k_j	$\min(r_{k_j,h_1}, z_{h_1})$	…	$\min(r_{k_j,h_i}, z_{h_i})$	$\underset{1\le l\le i}{\text{maks}}\left\{\min(r_{k_j,h_l}, z_{h_l})\right\}$	γ_{k_j}

Şekil 4.28. Geliştirilmiş yol bulmada u. adım matrisi

Benzer şekilde devam edilerek $d(1)$ adıma gelinir.

***d*(1). Adım:** Bu adımda uzaklık etiketi $d(1)$ olan düğümlerden kuyu düğümüne yapılabilecek olan maksimum akış ve bu akışın yapılabileceği yollar bulunur. Bunun için uzaklık etiketi $d(1)$ olan düğümler satır indisleri, uzaklık etiketleri $(d(1)-1)$ olan düğümler sütun indisleri olan bir matris oluşturulur. $(d(1)-1)$. adımda uzaklık etiketi $(d(1)-1)$ olan düğümlerden kuyu düğümüne yapılabilecek maksimum akış miktarı ve bu düğümlerden kuyuya giden yollar bulunmuştu. Matrisin her (i, j) ara elemanı satırda

bulunan *i* düğümünden sütunda bulunan *j* düğümüne giden arkın boş kapasitesi ile $(d(1)-1)$. adımda bulunan *j* düğümünden kuyuya giden maksimum kapasiteli yolun kapasitesinin minimumu yazılır. Kapasite sütununda ise satırdaki elemanların maksimumu, maksimumun bulunduğu yerin sütun indisi ise Etiket sütununa yazılır. *d*(1). adımda hem kaynak düğümünden kuyu düğümüne giden bir kabul edilebilir yol bulunur hem de uzaklık etiketi *d*(1) olan düğümler ile kuyu düğümü arasında akış yapılabilecek yollar da bulunmuş olur. Bu adımda bulunan kaynak düğümü ile kuyu düğümü arasındaki kabul edilebilir yolun kapasitesi diğer yolların kapasitesinden küçük ise $(d(1)+1)$. adım yapılır. Şekil 4.29 da *d*(1). adım genel olarak verilmiştir. Şekil 4.29 da verilen matriste uzaklık etiketi *d*(1) olan düğümler $1, p_1, ..., p_{j_1}$ ile, uzaklık etiketi (*d*(1)-1) olan düğümler $q_1, q_2, ..., q_{i_1}$ ile, uzaklık etiketi (*d*(1)-1) olan düğümlerden kuyuya giden maksimum kapasiteli yolların kapasiteleri ise $z_{q_1}, z_{q_2}, ..., z_{q_{i_1}}$ ile gösterilmiştir. Örneğin Şekil 4.29 daki matrisin 1. satırını incelersek, 1 düğümünden kuyu düğümüne yapılabilecek olan maksimum akış $\underset{1\le l\le i_1}{\text{maks}}\left\{\min(r_{1,q_l}, z_{q_l})\right\}$ dir, bu akışın yapılabileceği yolda 1 düğümünden sonra gelen düğüm γ_1 düğümüdür.

Düğüm	q_1	…	q_{i_1}	Kapasite	Etiket
1	$\min(r_{1,q_1}, z_{q_1})$	…	$\min(r_{1,q_{i_1}}, z_{q_{i_1}})$	$\underset{1\le l\le i_1}{\text{maks}}\left\{\min(r_{1,p_l}, z_{p_l})\right\}$	γ_1
⋮	⋮	⋮	⋮	⋮	⋮
p_{j_1}	$\min(r_{p_{j_1},p_1}, z_{p_1})$	…	$\min(r_{p_{j_1},q_{i_1}}, z_{q_{i_1}})$	$\underset{1\le l\le i_1}{\text{maks}}\left\{\min(r_{p_{j_1},p_l}, z_{p_l})\right\}$	$\gamma_{p_{j_1}}$

Şekil 4.29. Geliştirilmiş yol bulmada *d*(1). adım matrisi

(*d*(1) +1). Adım: Bu adım $d(1)$. adımda bulunan kaynak düğümünden kuyu düğümüne kabul edilebilir yolun kapasitesi uzaklık etiketi $d(1)$ olan düğümler ile kuyu düğümü arasında yolların kapasitelerinden küçükse yapılır. $(d(1)+1)$. adımda satır indisi kaynak düğümü, sütun indisi uzaklık etiketi $d(1)$e eşit olan düğümler olan bir matris düzenlenir. Matrisin her (*i*, *j*) ara elemanı kaynak düğümünden sütunda bulunan *j* düğümüne giden arkın boş kapasitesi ile $d(1)$. adımda bulunan sütundaki düğümünden

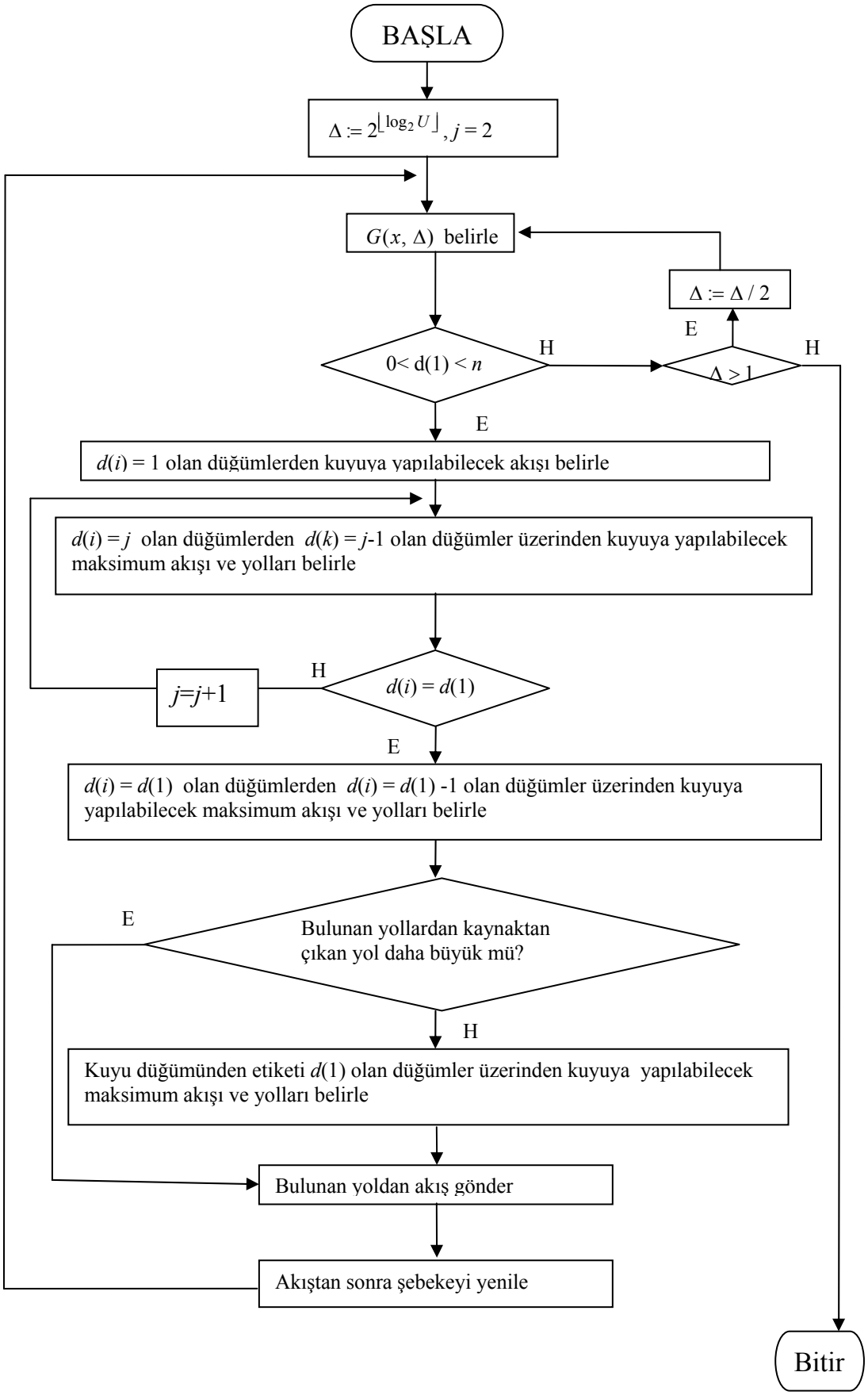
kuyuya gönderilebilecek maksimum akışın minimumu yazılır. Kapasite sütununda ise satırdaki elemanların maksimumu, maksimumun bulunduğu yerin sütun indisi ise Etiket sütununa yazılır. Böylece bu adımda kaynak düğümünden kuyu düğümüne giden bir kabul edilebilir yol bulunmuş olur. Şekil 4.30 da bu adım genel olarak verilmiştir. Bir önceki adımda uzaklık etiketi $d(1)$ olan düğümler $1, p_1$ ..., p_{j_1} ile gösterilmişti. Bir önceki adımda bulunan uzaklık etiketi $d(1)$ olan düğümlerden kuyuya giden maksimum kapasiteli yolların kapasiteleri Şekil 4.30 da $z_{p_1}, z_{p_2}, ..., z_{p_{j_1}}$ ile gösterilmiştir. Şekil 4.30 daki matrisin 1. satırını incelersek, 1 kaynak düğümünden kuyu düğümüne yapılabilecek olan maksimum akış $\underset{1\le l\le j_1}{\text{maks}}\left\{\min(r_{1,p_l}, z_{p_l})\right\}$ dir, bu akışın yapılabileceği yolda 1 kaynak düğümünden sonra gelen düğüm γ_1 düğümüdür.

Düğüm	p_1	⋯	p_{j_1}	Kapasite	Etiket
1	$\min(r_{1,p_1}, z_{p_1})$	⋯	$\min(r_{1,p_{j_1}}, z_{p_{j_1}})$	$\underset{1\le l\le j_1}{\text{maks}}\left\{\min(r_{1,p_l}, z_{p_l})\right\}$	γ_1

Şekil 4.30. Geliştirilmiş yol bulmada ($d(1)$+1). adım matrisi

$(d(1)+1)$. adımda bulunan kaynak düğümünden kuyu düğümüne giden kabul edilebilir yolun kapasitesi ile $d(1)$. adımda bulunan kaynak düğümünden kuyu düğümüne giden kabul edilebilir yolun kapasitesi karşılaştırılarak kapasitesi büyük olan yoldan akış gönderilir. Eğer $(d(1)+1)$. adımındaki yol daha büyükse bu yol kaynak düğümü, $(d(1)+1)$. adımda Etiket sütununda bulunan düğüm γ_1, γ_1 düğümünün $d(1)$. adımda Etiket sütununda karşı gelen düğüm γ_{γ_1}, ..., en sonda 1. adımda kuyu düğümü ve bu düğümler arasındaki arkalardan oluşur. Eğer $d(1)$. adımındaki yol daha büyükse bu yol kaynak düğümü, $d(1)$. adımda kaynak düğümünün Etiket sütununda karşı gelen düğüm γ_1, γ_1 düğümünün $(d(1)-1)$. adımda Etiket sütununda karşı gelen düğüm γ_{γ_1}, ..., en sonda 1. adımda kuyu düğümü ve bu düğümler arasındaki arkalardan oluşur.

Bu şebekenin akış diyagramı Şekil 4.31 de verilmiştir.



Şekil 4. 31. Geliştirilen algoritmanın akış diyagramı

Teorem 4.1. Geliştirilen algoritmanın her bir Δ ölçek aşamasında $O(n)$ akış ve her bir akış için $O\left(n^2\right)$ işlem yapıldığından dolayı bu algoritmanın karmaşıklığı $O\left(n^3 \log_2 U\right)$ dur.

İspat. Geliştirilen algoritmada Δ ölçeklerinin sayısı $\left(1+\lfloor \log_2 U \rfloor\right)$ dur. Her bir Δ ölçek aşamasında kaynak düğümünden kuyu düğümüne her bir akışta en az Δ birim akış gönderilir. Kaynak düğümden çıkan ark sayısı en fazla (n-1) olduğundan her bir Δ ölçek aşamasında en fazla (n-1) defa akış yapılabilir. Her bir akışta en fazla $(d(1)+1)$ adım yapılmaktadır. Şebekedeki düğüm sayısı n olduğundan bir akış için en fazla işlem sayısı sadece iki farklı uzaklık etiketindeki düğüm sayısı $\left[n-\left(d(1)-2\right)\pm 1\right]/2$ düğüm, diğer uzaklık etiketlerinde ise sadece 1 düğüm olduğu durumdur. Bu durumda bir akışta en fazla

$$\left(\frac{\left[n-\left(d(1)-2\right)\pm 1\right]}{2}\right)\left(\frac{\left[n-\left(d(1)-2\right)\pm 1\right]}{2}\right)+2\left(\frac{\left[n-\left(d(1)-2\right)\pm 1\right]}{2}\right)=O\left(n^2\right)$$

işlem yapılabilir. Böylece geliştirilen algoritmanın karmaşıklığı $O\left(n^3 \log_2 U\right)$ dur.

Örnek 4.5. Örnek 4.1 de verilen şebekede maksimum akışı bu geliştirilen algoritma ile çözelim. Şekil 4.32 de bu şebeke verilmiştir. Bu şebekede $U = 18$ ve $\Delta = 2^{\lfloor \log_2 U \rfloor} = 16$ dır. Şekil 4.33 te $G(x, 16)$ şebekesi verilmiştir.

Düğüm	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
1	---	4	12	8	12	0	4	0
2	0	---	18	14	8	4	8	8
3	0	4	---	0	4	4	14	10
4	0	0	5	---	12	8	0	8
5	0	0	8	8	---	8	12	4
6	0	4	5	0	7	---	0	4
7	0	7	1	6	6	9	---	14
8	0	0	0	0	0	0	0	---

Şekil 4.32. Örnek 4.1 de verilen boş kapasitelerden oluşan şebeke

Düğüm	1	2	3	4	5	6	7	8
1	---	0	0	0	0	0	0	0
2	0	---	18	0	0	0	0	0
3	0	0	---	0	0	0	0	0
4	0	0	0	---	0	0	0	0
5	0	0	0	0	---	0	0	0
6	0	0	0	0	0	---	0	0
7	0	0	0	0	0	0	---	0
8	0	0	0	0	0	0	0	---

Şekil 4.33. Şekil 4.32 de verilen şebekenin $G(x, 16)$ alt şebekesi

$\Delta = 16$ ölçek aşamasında akış yapılamaz. Bu yüzden $\Delta / 2 = 8$ aşamasına geçilir. $G(x, 8)$ alt şebekesi ve düğümlerin uzaklık etiketleri Şekil 4.34 te verilmiştir.

Düğüm	1	2	3	4	5	6	7	8
1	---	0	12	8	12	0	0	0
2	0	---	18	14	8	0	8	8
3	0	0	---	0	0	0	14	10
4	0	0	0	---	12	8	0	8
5	0	0	8	8	---	8	12	0
6	0	0	0	0	0	---	0	0
7	0	0	0	0	0	9	---	14
8	0	0	0	0	0	0	0	---
***d*(*i*)**	2	1	1	1	2		1	

Şekil 4.34. Şekil 4.32 de verilen şebekenin $G(x, 8)$ alt şebekesi ve düğümlerin uzaklık etiketleri

Birinci akış için 1. Adım. Bu adımda kaynak düğümüne akış yapan, yani uzaklık etiketi 1 olan düğümler ile kaynak düğümü arasındaki arkların kapasiteleri matris halinde yazılır. Bu matris Şekil 4.35 te gösterilmiştir.

Düğümler	8	Kapasite	Etiket
2	8	8	8
3	10	10	8
4	8	8	8
7	14	14	8

Şekil 4.35. Birinci akış için 1. Adım matrisi

Birinci akış için 2. Adım. Bu adımda uzaklık etiketi 2 olan düğümlerden kuyu düğümüne olan yollar ve bu yolların kapasiteleri bulunur. Bu matrisin (1,2) elemanı $\min(r_{2t}, r_{12}) = \min(8,0) = 0$ tür. Yani 1 kaynak düğümünden 2 düğümü üzerinden giden bir yolda en fazla 0 birim akış gönderilebilir, yani akış gönderilemez. Benzer şekilde diğer elemanlar aynı şekilde hesaplanırsa Şekil 4.36 da verilen matris bulunur. Kapasite sütunundaki eleman bu satırdaki elemanların maksimumudur, Etiket sütunundaki eleman ise bu maksimum olan elamanın sütun indisidir.

Düğümler	**2**	**3**	**4**	**7**	**Kapasite**	**Etiket**
1	0	10	8	0	10	3
5	0	8	8	12	12	7

Şekil 4.36. Birinci akış için 2. Adım matrisi

Şekil 4.36 da verilen matrise göre kaynaktan 3 düğümü üzerinden 10 birim akış gönderilebilir. Şekil 4.36 daki matriste görüldüğü gibi 1 ve 5 düğümlerinin uzaklık etiketleri aynıdır. Fakat Kapasite sütununda görüldüğü gibi 5 düğümünde diğer yollardan daha fazla akış yapılabilir. Bu durumda bir sonraki aşamaya geçilir.

Birinci akış için 3. Adım. Bu adımda kaynaktan 5 düğümü üzerinden giden yolun kapasitesi 2. adımda bulunan kapasiteden daha büyük kapasiteye sahip olup olmadığına bakılır ve kapasitesi büyük olan yol seçilir. Şekil 4.37 de verilen matriste görüldüğü gibi kaynak düğümünden kuyu düğümüne 5 düğümü üzeriden geçen yoldan 12 birimlik akış yapılabilir.

Düğüm	**5**	**Kapasite**	**Etiket**
1	12	12	5

Şekil 4.37. Birinci akış için 3. Adım matrisi

Bu adımda bulunan yol bir önceki adımda bulunan yoldan daha fazla akış gönderilebildiğinden dolayı bu adımda bulunan yol tercih edilir. Bu yol 1, (1, 5), 5, (5, 7), (7, 8), 8 dir ve bu yolundan 12 birim akış gönderilebilir. Bu akıştan sonra şebekedeki arta kalan boş kapasiteler ve $G(x,8)$ alt şebekesi hesaplanır. Şekil 4.38 de bu akıştan sonra arta kalan şebeke, Şekil 4.39 da $G(x, 8)$ alt şebekesi ve düğümlerin uzaklık etiketleri verilmiştir.

Düğüm	1	2	3	4	5	6	7	8
1	---	4	12	8	0	0	4	0
2	0	---	18	14	8	4	8	8
3	0	4	---	0	4	4	14	10
4	0	0	5	---	12	8	0	8
5	12	0	8	8	---	8	0	4
6	0	4	5	0	7	---	0	4
7	0	7	1	6	18	9	---	2
8	0	0	0	0	0	0	12	---

Şekil 4.38. Birinci akıştan sonra arta kalan şebeke

Düğüm	1	2	3	4	5	6	7	8
1	---	0	12	8	0	0	0	0
2	0	---	18	14	8	0	8	8
3	0	0	---	0	0	0	14	10
4	0	0	0	---	12	8	0	8
5	12	0	8	8	---	8	0	0
6	0	0	0	0	0	---	0	0
7	0	0	0	0	12	9	---	0
8	0	0	0	0	0	0	12	---
***d*(*i*)**	2	1	1	1	2		3	

Şekil 4.39. Birinci akıştan sonra $G(x, 8)$ alt şebekesinin arta kalan şebekesi ve düğümlerin uzaklık etiketleri

Bu şebeke için ikinci akış için adımlar:

İkinci akış için 1. Adım. Bu adımda kuyu düğümüne akış yapan, yani uzaklık etiketi 1 olan düğümler ile kuyu düğümü arasındaki arkların kapasiteleri matris halinde yazılır. Bu matris Şekil 4.40 ta gösterilmiştir.

Düğümler	8	Kapasite	Etiket
2	8	8	8
3	10	10	8
4	8	8	8

Şekil 4.40 İkinci akış için 1. Adım matrisi

İkinci akış için 2.Adım. Bu adımda uzaklık etiketi 2 olan düğümlerden kuyu düğümüne olan yollar ve bu yolların kapasiteleri hesaplanırsa Şekil 2.41 de verilen matris bulunur.

Düğüm	2	3	4	Kapasite	Etiket
1	0	10	8	10	3
5	0	8	8	8	3

Şekil 4.41. İkinci akış için 2. Adım matrisi

Bu matriste kuyu düğümü etiketlenmiş ve Kapasite sütununa göre en büyük kapasiteli yola sahiptir. Dolayısıyla bu yoldan akış yapılır. Bu yol 1, (1, 3), 3, (3, 8), 8 dir ve bu yoldan 10 birim akış gönderilebilir. Bu akıştan sonra şebekede arta kalan boş kapasiteler ve $G(x,8)$ alt şebekesi hesaplanır. Şekil 4.42 de bu akıştan sonra arta kalan şebeke, Şekil 4.43 te $G(x, 8)$ alt şebekesi ve düğümlerin uzaklık etiketleri verilmiştir.

Düğüm	1	2	3	4	5	6	7	8
1	---	4	2	8	0	0	4	0
2	0	---	18	14	8	4	8	8
3	10	4	---	0	4	4	14	0
4	0	0	5	---	12	8	0	8
5	12	0	8	8	---	8	0	4
6	0	4	5	0	7	---	0	4
7	0	7	1	6	18	9	---	2
8	0	0	10	0	0	0	12	---

Şekil 4.42. İkinci akıştan sonra arta kalan şebeke

Düğüm	1	2	3	4	5	6	7	8
1	---	0	0	8	0	0	0	0
2	0	---	18	14	8	0	8	8
3	10	0	---	0	0	0	14	0
4	0	0	0	---	12	8	0	8
5	12	0	8	8	---	8	0	0
6	0	0	0	0	0	---	0	0
7	0	0	0	0	12	9	---	0
8	0	0	10	0	0	0	12	---
***d*(*i*)**	2	1	4	1	2		3	

Şekil 4.43. İkinci akıştan sonra $G(x, 8)$ alt şebekenin arta kalan şebekesi ve düğümlerin uzaklık etiketleri

Aynı işlemleri tekrarlanırsa 6 kabul edilebilir yolda toplam 15 arkta akış yapıldıktan sonra maksimum akışa ulaşılır. Maksimum akışa ulaşıldıktan sonra arta kalan şebeke Şekil 4.44 te verilmiştir.

Düğüm	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
1	---	0	0	0	0	0	0	0
2	4	---	18	14	8	4	12	0
3	10	4	---	0	2	4	14	0
4	8	0	5	---	12	8	0	0
5	12	0	8	8	---	8	0	0
6	0	4	5	0	7	---	0	4
7	4	3	1	6	18	9	---	2
8	0	8	10	8	2	0	12	---

Şekil 4.44. Maksimum akış bulunduktan sonra arta kalan şebeke

4.6. BİRDEN ÇOK KAYNAK DÜĞÜMÜ VE KUYU DÜĞÜMÜ OLAN ŞEBEKELERDE MAKSİMUM AKIŞIN MATRİS GÖSTERİMİ İLE HESAPLANMASI

Kaynak ya da kuyu sayısı birden fazla olan şebekelerde maksimum akışı bulmak için Kısım 2.16 da da gösterildiği gibi şebekeye yeni düğüm kaynak düğümü ve yeni kuyu düğümü eklenir. Şebekedeki kaynak düğümleri yeni kaynak düğümüne kuyu düğümleri yeni kuyu düğümüne kapasiteleri yeteri kadar büyük arklarla birleştirilir. Böylece şebeke bir kaynak düğümü ve bir kuyu düğümüne sahip bir şebekeye olur. Bu durumda şebekede bir kaynak düğümünden bir kuyu düğümüne gönderilebilecek maksimum akış problemine döner. Bu durumda herhangi bir algoritmayla maksimum akış problemi çözülür.

Örnek 4.6. Şekil 4.45 te verilen şebekede 2, 3, 4 üç kaynak düğümünden 9, 10, 11 üç kuyu düğümüne gönderilebilecek maksimum akışı bulmak için önce şebekeye yeni kaynak düğüm ile yeni kuyu düğümü eklenir. Şebekede yeni kaynak düğümü 1 ile yeni kuyu düğümünü 12 ile gösterilmektedir. Ayrıca yeni kaynak düğümü 2, 3, 4 kaynak düğümleriyle, yeni kuyu düğümü 9, 10, 11 düğümleriyle ark kapasiteleri sonsuz olan arklarla birleştirilir. Bu matris Şekil 4.46 da gösterilmiştir. Böylece şebekede bir kaynak düğüm bir kuyu düğümü vardır ve maksimum akış problemi bir kaynak düğümünden

bir kuyu düğümüne yapılabilecek maksimum akış problemine dönüşür. Bu şebekede yapılabilecek maksimum akış Kısım 4.4 te verilen algoritma ile bulunması:

Düğüm	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**
2	---	0	0	2	4	10	6	0	0	0
3	0	---	0	2	6	2	6	0	0	0
4	0	0	---	10	12	16	0	0	0	0
5	0	0	0	---	0	16	6	14	10	4
6	0	0	0	1	---	2	10	6	10	8
7	0	0	0	8	1	---	10	2	14	12
8	0	0	0	4	4	5	---	14	8	12
9	0	0	0	0	0	0	0	---	0	0
10	0	0	0	0	0	0	0	0	---	0
11	0	0	0	0	0	0	0	0	0	---

Şekil 4.45. Üç kaynak düğümü ve üç kuyu düğümü olan şebeke

Düğüm	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**
1	---	∞	∞	∞	0	0	0	0	0	0	0	0
2	0	---	0	0	2	4	10	6	0	0	0	0
3	0	0	---	0	2	6	2	6	0	0	0	0
4	0	0	0	---	10	12	16	0	0	0	0	0
5	0	0	0	0	---	0	16	6	14	10	4	0
6	0	0	0	0	1	---	2	10	6	10	8	0
7	0	0	0	0	8	1	---	10	2	14	12	0
8	0	0	0	0	4	4	5	---	14	8	12	0
9	0	0	0	0	0	0	0	0	---	0	0	∞
10	0	0	0	0	0	0	0	0	0	---	0	∞
11	0	0	0	0	0	0	0	0	0	0	---	∞
12	0	0	0	0	0	0	0	0	0	0	0	---

Şekil 4.46. Üç kaynak düğümü ve üç kuyu düğümü olan şebekeye yeni kaynak ve kuyu düğümünün eklendikten sonra oluşan şebeke

$U = 16$ olduğunda $\Delta = 16$ dır. $G(x,16)$ alt şebekesi Şekil 4.47 de verilmiştir. Şekilde verilen matriste kaynak düğümünden kuyu düğümüne kabul edilebilir bir yol

olmadığından akış yapılamaz. $\Delta = 16/2 = 8$ ölçeğine geçilir. $G(x,8)$ alt şebekesi ve düğümlerin uzaklık etiketleri Şekil 4.48 de verilmiştir.

Düğüm	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**
1	---	∞	∞	∞	0	0	0	0	0	0	0	0
2	0	---	0	0	0	0	0	0	0	0	0	0
3	0	0	---	0	0	0	0	0	0	0	0	0
4	0	0	0	---	0	0	16	0	0	0	0	0
5	0	0	0	0	---	0	16	0	0	0	0	0
6	0	0	0	0	0	---	0	0	0	0	0	0
7	0	0	0	0	0	0	---	0	0	0	0	0
8	0	0	0	0	0	0	0	---	0	0	0	0
9	0	0	0	0	0	0	0	0	---	0	0	∞
10	0	0	0	0	0	0	0	0	0	---	0	∞
11	0	0	0	0	0	0	0	0	0	0	---	∞
12	0	0	0	0	0	0	0	0	0	0	0	---

Şekil 4.47. $G(x,16)$ alt şebekesi

Düğüm	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**
1	---	∞	∞	∞	0	0	0	0	0	0	0	0
2	0	---	0	0	0	0	10	0	0	0	0	0
3	0	0	---	0	0	0	0	0	0	0	0	0
4	0	0	0	---	10	12	16	0	0	0	0	0
5	0	0	0	0	---	0	16	0	14	10	0	0
6	0	0	0	0	0	---	0	10	0	10	8	0
7	0	0	0	0	8	0	---	10	0	14	12	0
8	0	0	0	0	0	0	0	---	14	8	12	0
9	0	0	0	0	0	0	0	0	---	0	0	∞
10	0	0	0	0	0	0	0	0	0	---	0	∞
11	0	0	0	0	0	0	0	0	0	0	---	∞
12	0	0	0	0	0	0	0	0	0	0	0	---
***d*(*i*)**	4	3		3	2	2	2	2	1	1	1	

Şekil 4.48. $G(x,8)$ alt şebekesi ve düğümlerin uzaklık etiketleri

Birinci akış için 1. Adım. Bu adımda uzaklık etiketi 1 olan düğümler ve bu düğümlerin kuyu yapabilecekleri akış miktarları yazılır. Bu adım bütün akışlar için aynıdır. Çünkü kuyuya akış yapabilecek üç düğüm vardır ve aralarındaki arkların kapasiteleri ∞ dur. Bu matris Şekil 4.49 da verilmiştir.

Düğüm	12	Kapasite	Etiket
9	∞	∞	12
10	∞	∞	12
11	∞	∞	12

Şekil 4.49. Birinci akış için 1. Adım matrisi

Birinci akış için 2. Adım. Bu adımda uzaklık etiketi 2 olan düğümlerden kuyu düğümüne yapılabilecek maksimum akışlar ve bu akışların geçtiği yollar bulunur. Şekil 4.50 de bu matris verilmiştir.

Düğüm	9	10	11	Kapasite	Etiket
5	14	10	0	14	9
6	0	10	8	10	10
7	0	14	12	14	10
8	14	8	12	14	9

Şekil 4.48. Birinci akış için 2. Adım matrisi

Birinci akış için 3. Adım. Bu adımda uzaklık etiketi 3 olan düğümlerden kuyuya yapılabilecek maksimum akışlar ve bu akışların geçtiği yollar bulunur. Şekil 4.51 de bu matris verilmiştir.

Düğüm	5	6	7	8	Kapasite	Etiket
2	0	0	10	0	10	7
4	10	10	14	0	14	7

Şekil 4.51. Birinci akış için 3. Adım matrisi

Birinci akış için 4. Adım. Bu adımda uzaklık etiketi 4 olan düğümlerden kuyuya yapılabilecek maksimum akışlar ve bu akışların geçtiği yollar bulunur. Şekil 4.52 de bu matris verilmiştir.

Düğüm	2	4	Kapasite	Etiket
1	10	14	14	4

Şekil 4.52. Birinci akış için 4. Adım matrisi

Bu adımda kaynak etiketlendiğinden kaynak düğümünden kuyu düğümüne bir yol bulunmuş olur. Bu yol 1, (1, 4), 4, (4, 7), 7, (7, 10), 10, (10, 12), 12 dir ve bu yolundan 14 birim akış yapılabilir. Akıştan sonra arta kalan şebeke düzenlenir ve düğümlerin etiketleri yenilenir.

Hesaplamaya aynı şekilde devam edildiğinde maksimum akış yapıldıktan sonra şebekedeki arta kalan boş kapasiteler Şekil 4.53 te verilmiştir. Maksimum akışa ulaşmak için 13 akışta toplam 52 arkta akış yapılmıştır. Şekil 4.53 te verilen matrise göre 9 kuyu düğümüne 24 birim, 10 kuyu düğümüne 28 birim, 11 kuyu düğümüne 24 birim akış yapılmış toplamda şebekede 76 birim akış yapılmıştır.

Düğüm	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**
1	---	∞	∞	∞	0	0	0	0	0	0	0	0
2	22	---	0	0	0	0	0	0	0	0	0	0
3	16	0	---	0	0	0	0	0	0	0	0	0
4	38	0	0	---	0	0	0	0	0	0	0	0
5	0	2	4	10	---	0	16	6	4	6	4	0
6	0	4	6	12	1	---	2	10	0	0	2	0
7	0	10	2	16	8	1	---	10	0	0	0	0
8	0	6	6	0	4	4	5	---	8	8	6	0
9	0	0	0	0	10	6	2	6	---	0	0	∞
10	0	0	0	0	4	10	14	0	0	---	0	∞
11	0	0	0	0	0	6	12	6	0	0	---	∞
12	0	0	0	0	0	0	0	0	24	28	24	---

Şekil 4.53. Maksimum akış bulunduktan sonra arta kalan şebeke

4.7. MAKSİMUM AKIŞ ALGORİTMALARININ KARŞILAŞTIRILMASI

Kapasite ölçekleme algoritması, en büyük etiketli akış öncesi gönderme algoritması ve geliştirilen algoritmanın C bilgisayar programlama dilinde kodları yazılarak bu algoritmaların CPU (Central Processing Unit, Merkezi işlem birimi, işlemci) süresi saniye olarak ve akış yapılan arkların sayısı üzerinden karşılaştırmaları Şekil 4.54 de verilmiştir.

	KÖA		EBGA		GA	
	CPU	**AS**	**CPU**	**AS**	**CPU**	**AS**
***n* = 100, *m*= 9900, *U* = 100**	1,024	5794	0,016	713	0,019	457
***n* = 200, *m*= 39800, *U* = 100**	2,702	45705	0,287	1310	0,298	608
***n* = 500, *m*= 249500, *U* = 100**	61,875	212766	3,933	4457	3,875	2713
***n* = 1000, *m*= 999000, *U* = 100**	162,170	1072659	24,869	9250	24,390	5809
***n* = 100, *m*= 9900, $U = 2^{20}$**	1,929	12019	0,019	415	0,021	340
***n* = 200, *m*= 39800, $U = 2^{20}$**	2,519	119506	0,020	777	0,023	856
***n* = 500, *m*= 249500, $U = 2^{20}$**	73,601	491635	3,89	4003	3,920	1704
***n* = 1000, *m*= 999000, $U = 2^{20}$**	173,17	2072659	25,046	13914	25,845	8909
***n* = 100, *m*= 990, *U* = 100**	1,38	3145	0,013	447	0,018	321
***n* = 200, *m*= 3980, *U* = 100**	2,12	23754	0,221	986	0,235	880
***n* = 500, *m*= 24950, *U* = 100**	16,435	168846	3,818	2522	3,728	2019
***n* = 1000, *m*= 99900, *U* = 100**	147, 86	704510	18,692	5139	20,247	4021

Şekil 4.54. Algoritmaların karşılaştırılması, burada KÖA: Kapasite ölçekleme algoritması, EBGA: En büyük etiketli akış öncesi gönderme algoritması, GA: Geliştirilen algoritma, AS: Ark sayısı, *n*: Düğüm sayısı, *m*: Ark sayısı, *U*: Arklardaki boş kapasitelerin üst sınırı olarak alınmıştır.

Bu programlarda şebekedeki arkların boş kapasiteleri C bilgisayar programlama diline ait standart rand() rastgele sayı üreten fonksiyon kullanılarak belirlenmiş ve tam sayı türünden iki boyutlu dizinler (array) olarak alınmıştır. Karşılaştırmalarda n düğüm sayısı 100, 200, 500 ve 1000 olarak alındı. Şekil 4.54 te verilen tabloda m ark sayısı, n düğüm sayısı olmak üzere ilk dört satırda $m = n(n\text{-}1)$ olarak ve ark kapasitelerinin üst sınır $U = 100$, 5, 6, 7 ve 8. inci satırlarda $m = n(n\text{-}1)$ ve $U = 2^{20}$, son dört satırda $m = n(n-1)/10$ ve $U = 100$ alınmıştır.

Maksimum akış bulunurken şebekelerin en büyük kapasiteli arkın kapasitesi olan U nun küçük ya da büyük olması çözümün süresinin uzamasına ve akış yapılan ark sayısına çokça etki etmediği Şekil 4.54 te görülmektedir. Maksimum akış probleminin çözüm süresine ve akış yapılan ark sayısına sadece şebekenin düğüm ve ark sayısının etki ettiği görülmektedir.

Şekil 4.54 te görüldüğü gibi geliştirilen algoritmanın, şebekelerin en büyük kapasiteli arkının kapasitesinin büyüklüğüne, şebekedeki ark ve düğüm sayısına bağlı olmaksızın maksimum akışı hesaplamada kapasite ölçekleme algoritmasına göre daha az zamanda ve daha az sayıdaki ark üzerinden akış yaparak hesapladığı görülmektedir. Fakat en büyük etiketli akış öncesi gönderme algoritmasına göre geliştirilen algoritma genelde daha az arkta akış yapmasına karşın neredeyse aynı sürede, bazen daha uzun sürede maksimum akışı hesaplamaktadır. Bunun nedeni olarak, geliştirilen algoritmanın kabul edilebilir yol bulurken zaman kaybetmesi, bu karşın daha az sayıda ark üzerinden akış yaparak toplam süreyi kısaltması gösterilebilir. Şebekedeki ark sayısının maksimum akışın bulunmasında toplam süreye çokça etki etmediği görülmüştür. Bunun nedeni şebekelerin matris gösteriminde düğümler arasında arklar olmadığında matriste bu arkların kapasiteleri sıfır olarak yazılmaktadır. Ayrıca diğer durumlardaki gibi ark sayısının az olduğu durumda da şebeke matrisleri bellekte aynı büyüklükte yer kaplamaktadır ve şebekelerin belleğe yazma ve okuma aynı sürede gerçekleşmektedir. Ark sayısı az olduğu durumda matriste sıfırların sayısı çoktur ve algoritmalarda matriste sıfırdan farklı elamanların bulunmasında zaman kaybedildiği düşünülmektedir.

4.8. GÖRÜNTÜ TAŞINIMI, SAKLANMASI VE GERİ ÇAĞIRIMI

Biyomedikal uygulamalarda kullanılan görüntü dosyaları çok büyük hacimde olabilmektedir. Böyle büyük dosyaların saklanması ve gerektiğinde üzerinde işlemler yapabilmek için bu dosyalara kolay ve hızlı ulaşmak gerekir. Bu büyük hacimli dosyaların daha küçük boyutlara sahip parçalara ayırımı ve bu parçaları kullanıcılar arasında paylaşımı sağlanarak hem depolamada hem de bu dosyalara kısa süre içinde ulaşılmasında kolaylıklar getirir.

Kısım 2.7.5 te gösterildiği gibi bilgisayar ağ şebekeleri maksimum akış problemine uyarlanabilir. Bunun için kullanıcılar kaynak düğümlerini ve kuyu düğümlerini, anahtarlar (switch), yönlendiricilerin (router) ara düğümler olarak alınır. Kullanıcılar istemci olduklarında kuyu düğümleri, sahip oldukları veriler diğer kullanıcılar tarafından istenildiğinde kaynak düğümleri olarak alınırlar. Elde edilen şebekede kaynak düğümü sayısı ve kuyu düğümü sayısı birden fazla olduğundan Kısım 4.6 daki yöntem ve Kısım 4.5 teki algoritma kullanılarak çözülebilir.

Bu amaca uygun olarak bağımsız bir bilgisayar ağı test amacıyla kullanıldı. Bu ağa bağlı bilgisayarlar Linux işletim sistemi altında ve NFS (Network File System) protokolü ile dosya transfer işlemlerini gerçekleştirildi. Burada NFS data transfer portlarının tanımlanmasında kolaylık sağladığı için kullanıldı. Düğümler arası kapasiteler, test ağındaki dosya transferlerine yapay gecikmeler bindirilerek belirlendi. Test ağındaki büyük medikal kaynaklı dosyaların transferi bu yöntem ve bu çalışma kapsamında geliştirilen algoritma ile gerçekleştirildi. Büyük medikal dosyalar C programlama dili kullanılarak yazılan bir programla her biri optimize edilmiş büyüklükte parçalara ayrıldı. Geliştirilmiş olan algoritmanın gereği olarak sunucular ve istemeci arasındaki kapasite büyüklüğüne göre karar verici bir kontrol programı yoluyla transfer işlemleri gerçekleştirildi. Bu ana kontrol programı aynı zamanda istenilen dosyaların hangi sunucularda bulunduğunu sürekli olarak izler ve istemcilerdeki yapay transfer gecikmelerini de belirler. Bu ana kontrol programı C programlama dili kullanılarak yazılmıştır.

5. TARTIŞTIMA VE SONUÇ

Bu çalışmada maksimum akış problemi için bilinen ektin algoritmaların şebekelerin düğüm - düğüm matris gösterimine uygulanışları verilmiştir. Ayrıca düğüm - düğüm matris gösterimine dayalı yeni bir algoritma da geliştirilmiştir. Bu matris gösterimi tekniği bilgisayar kodlamasında verimliliği artırmakta ve daha etkin kullanıma yardımcı olmaktadır. Bu çalışma kapsamında kapasite ölçekleme algoritması, en büyük etiketli akış öncesi gönderme algoritması ve geliştirilen algoritmanın C bilgisayar programlama dilinde kodları yazılarak karşılaştırmaları yapılmıştır. Kodlamalarda şebekeler tam sayı türünden iki boyutlu dizinler (array) olarak alınmıştır. Bu dizinlerin her iki boyutu da *n* düğüm sayısı olarak alınmıştır. Geliştirilen algoritmanın, şebekelerin en büyük kapasiteli arkının kapasitesinin büyüklüğüne, şebekedeki ark ve düğüm sayısına bağlı olmaksızın maksimum akışı hesaplamada kapasite ölçekleme algoritmasına göre daha az zamanda ve daha az sayıdaki ark üzerinden akış yaparak hesapladığı görülmektedir. Geliştirilen algoritmanın karmaşıklığı, en büyük etiketli akış öncesi gönderme algoritmasına göre büyüktür. Bunun yanında en büyük etiketli akış öncesi gönderme algoritmasıyla karşılaştırıldığında, geliştirilmiş olan algoritmanın genelde daha az sayıda arkta akış yapmasına karşın, neredeyse aynı sürede, bazen de daha uzun sürede maksimum akışı hesapladığı gözlemlenmiştir. Bunun nedeni olarak, geliştirilen algoritmanın kabul edilebilir yol bulurken zaman kaybetmesi, bu karşın daha az sayıda ark üzerinden akış yaparak toplam süreyi kısaltması gösterilebilir. Şebekedeki ark sayısının maksimum akışın bulunmasında toplam süreye çokça etki etmediği görülmüştür.

	KÖA		**EBGA**		**GA**	
	CPU	**AS**	**CPU**	**AS**	**CPU**	**AS**
***m*= 999000**	162,170	1072659	24,869	9250	24,390	5809
***m*= 99900**	147, 86	704510	18,692	5139	20,247	4021

Şekil 5.1. n = 1000, U = 100 için algoritmaların karşılaştırılması, burada KÖA: Kapasite ölçekleme algoritması, EBGA: En büyük etiketli akış öncesi gönderme algoritması, GA: Geliştirilen algoritma, AS: Ark sayısı, *m*: Ark sayısı olarak alınmıştır.

Şekil 5.1 de şebekelerin düğüm sayısının n = 1000, en büyük kapasiteli arkın kapasitesinin U = 100 olması durumunda ark sayısının m = 999000 ve m = 99900 olduğu durumlar için algoritmaların karşılaştırmaları CPU (Central Process Unit, Merkezi işlem birimi, işlemci) süresi saniye olarak ve akış yapılan arkların sayıları ile verilmiştir. Şekil 5.1 de şebekelerdeki ark sayısının azaldığı durumda algoritmaların akış yapılan ark sayılarının azaldığı fakat CPU sürelerinin benzer oranda azalmadığı görülmektedir. Bunun nedeni şebekelerin matris gösteriminde düğümler arasında arkların bulunmadığı durumlarda matristeki bu arkların kapasiteleri için sıfır değeri atanmış olmasıdır. Ayrıca diğer durumlardaki gibi ark sayısının az olduğu durumda da şebeke matrisleri bellekte aynı büyüklükte yer kaplamaktadır ve şebekelerin belleğe yazma ve bellekten okuma işlemleri aynı sürede gerçekleşmektedir. Hesaplama süresinin uzamasının nedeni olarak, ark sayısının az olduğu durumlarda matriste sıfır değeri atanan elemanların sayısının artması ve sıfırdan farklı matris elamanları ararken oluşan kaybedilen zaman olduğu düşünülmektedir.

Ulaştırmada, taşımacılık ve iletişimde geniş uygulama alanı bulmuş olan maksimum akış algoritması, bu tez çalışmasında büyük hacimli görüntü dosyalarının bağımsız bir bilgisayar ağında taşınması, saklanması ve geri çağrılmasına uygulanmıştır. Bu tez çalışmasında geliştirilen algoritmanın en kısa yol problemine, minimum maliyet kapasiteli akış problemine ve birçok alandaki problemlere uygulanabileceği önerilebilir. Maksimum akış problemi için diğer optimizasyon yöntemleri kullanılarak daha etkin algoritmalar geliştirilebileceği düşünülmektedir.

KAYNAKLAR

1. FORD L.R., D.R. FULKERSON 1962, *Flows in network*, Princeton University Pres, Princeton, New Jersey, 0691079625

2. AHUJA, R.K., T.J. MAGNANTI, J.B. ORLIN, 1993, *Network flows*, Prentice Hall, Englewood Cliffs, New Jersey, 0-13-617-549-X

3. BAZARAA, M.S., J.J. JARVIS , H.D. SHERALI, 2005, *Linear programming and network flows*, A John Wiley & Sons, Inc., Publication, New Jersey, 0-471-48599-3

4. DANTZING, G. B. , M.N.,THAPA, 2003 *Linear Programming, 1: Introduction Springer* – Verlag, New York,, 0-387-94833-3

5. DANTZING, G. B. , M.N.,THAPA, 2003 *Linear Programming 2: Theory and Extensions*, Springer-Verlag , New York, 0-387-98613-8

6. BELLMAN, R., 1957, *Dynamic programming*, Princeton University Press, New Jersey, 069107951X

7. TOLSTOĬ, A.N., 1930, Metody nakhozhdeniya naimen'shego summovogo ilometrazha pri planirovanii perevozok v prostranstve [Russian; Methods of _nding the minimal total kilometrage in cargo transportation planning in space], in: *Planirovanie Perevozok, Sbornik pervy* [Russian; *Transportation Planning,* Volume I], Transpechat' NKPS [TransPress of the National Commissariat of Transportation], Moscow, syf. 23-55.

8. KANTOROVIC, L.V., 1939, Mathematical methods in the organization and planning of production, Publication House of Leningrad University. *Translated in Management Science* 6 (1960) 366-422.

9. HITCHCOCK , F.L., 1941, The distribution of a product from several source to numerous facilities, *Journal of Mathematical Physics*, 20, 224 -230

10. KOOPMANS, T.C., 1947, Optimum utilization of the transportation system, *Proceeding of the International Statics Conference*, Washington, DC. Also in *Econometrica* 17(1949)

11. DANTZING, G.B., 1951, Application of the simplex methods to a transportation problem, *In Activity Analysis and Production and Allocation*, edited by T.C. Koopmans. Wiley, New York, pp. 359-373.

12. FULKERSON, D.R., DANTZING G.B., 1955, Computation on maximal flow in networks, *Naval Research Logistic Quarterly* 2, 277-283.

13. FORD L.R., D.R. FULKERSON, 1956, Maximal flow through a network, *Canadian Journal of Mathematics,* 8, pp. 399-404.

14. AHLATÇIOĞLU, M., TİRYAKİ, F., 1998, *Kantitatif Karar Verme Teknikleri*, Yıldız Teknik Üniversitesi, İstanbul, 975-461-078-9.

15. BOYKOV Y., O. VEKSLER, 2006, *Graph Cuts in Vision and Graphics*, Handbook of Mathematical Models in Computer Vision (chapter 04), Springer, ISBN 0387263713,

16. SHI, J., C. FOWLESS, D. MARTIN, E. SHARON, *Graph Based Image Segmentation Tutorial*[online], http://www.cis.upenn.edu/~jshi/GraphTutorial/Tutorial-ImageSegmentationGraph-cut1-Shi.pdf Ziyaret tarihi: 1.09.2009

17. AHUJA, R.K., J.B. ORLIN, 1991, Distance-directed augmenting path algorithms for maximum flow and parametric maximum flow problems, *Naval Research Logistic Quarterly*, 38, 413-430.

18. GOLDBERG, A.V., R.E. TARJAN, 1986, A new approach to the maximum flow problem, *Journal of ACM,* 35(1988) , 921-940.

19. CHERIYAN, J., K. MEHLHORN, 1999, An Analysis of the Highest-Level Selection Rule in the Preflow-Push Max-Flow Algorithm , *Information Processing Letters* 69, pp. 239-242

20. AHUJA, R.K., ORLIN, J.B., TARJAN, R.E., 1989, Improved Time Bounds for the Maximum Flow Problem, *SIAM J. COMPUT.*, Vol. 18, No. 5, pp. 939-954,

5.ÖZGEÇMİŞ

Antakya doğdum. İlk, orta ve lise öğrenimimi Antakya'da okudum. 1993 te İstanbul Üniversitesi Fen Fakültesi Matematik Bölümünü kazandım ve kayıt oldum. 2000 de aynı bölümde yüksek lisans sınavını kazandım ve 2004 te yüksek lisansımı başarı ile tamamladım. 2004 de aynı bölümde doktora sınavını kazandım ve kayıt oldum. Aralık 2001 den beri İ.Ü. Fen Bilimleri Enstitüsü Matematik Anabilim dalında araştırma görevlisi olarak çalışmaktayım.