

47067

YILDIZ TEKNİK ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

SPC SANTRALLARDA BLOKAJI AZALTAN YENİ
BİR ALGORİTMA

Elekt. ve Hab. Yük.Müh. Sıtkı ÖZTÜRK

F.B.E Telekomünikasyon Anabilim Dalında

hazırlanan

DOKTORA TEZİ

Tez Danışmanı: Prof.Dr. Günsel DURUSOY (İ.T.Ü)

İSTANBUL, Ekim 1995

Y.C. YÜKSEKÖĞRETİM KURULU
TELEKOMÜNİKASYON MERKEZİ

İÇİNDEKİLER

Bölüm 1

1.0	Giriş.....	1
-----	------------	---

Bölüm 2

2.0	Bağlaşma Şebekeleri.....	4
2.1	Bağlaşma Sistemlerinin Sınıflandırılması.....	4
2.2	Bağlaşma.....	5
2.2.1	Kesişmeli (Matrisel) Seçiciler.....	5
2.3	Link Sistemleri.....	10
2.3.1	Çok Katlı Link Sistemleri.....	12
2.3.2	Link Sistemlerinde Tıkanma (Blokaj).....	13
2.3.3	Clos Tipi Link Sistemleri.....	14
2.3.4	Zaman Çoğullamalı Seçicinin Uzay Seçici Eşdeğeri.....	15

Bölüm 3

3.0	Yeniden Düzenleme Teorem Ve Algoritmaları.....	20
3.1	Üç Katlı Şebekede Yeniden Düzenlenme.....	20
3.1.1	Yeniden Düzenleme İçin Maksimum Bağlantı Değişikliği.....	23
3.1.2	Değişiklik Algoritması	27
3.2	Blokaj Kaldırma Yöntemleri.....	34
3.3	Diğer Şebekelere Genelleştirme.....	37
3.4	Üç Katlı Şebekede (N+1) Ortakat Grup Seçicisi İle Blokajın Kaldırılması.....	42
3.5	Bağlantı Zinciri İle Yeniden Düzenleme.....	46
3.5.1	Bağlantı Zinciri Tanımları.....	46
3.5.2	Bağlantı Zinciri İle Yeniden Düzenleme Yöntemi.....	49
3.5.3	Yeniden Düzenleme Algoritması.....	55
3.6	Paull Yönteminin Simülasyon Sonuçları.....	57

BÖLÜM 4

4.1	Teklif Edilen Algoritmanın Dayandığı Temeller.....	60
4.2	Teklif Edilen Algoritma.....	63
4.3.	Yeniden Düzenlemede Ortakat Grup Seçicisinin Bulunması.....	67
4.4.	Geliştirilen Algoritmanın Ve S.Ohta Ve H.Yueda Algoritmalarının Sonuçlarının Karşılaştırılması.....	69

BÖLÜM 5

	SONUÇ.....	72
	Kaynaklar.....	73
E1	300 çağırma isteği.....	73
E2	260 Bağlantı sonucundaki bağlaşma şebekesinin bağlantı durumu.....	80
E3	Yüklü ortakat grup seçicisinden başlayan aramada S.Ohta ve H. Yueda' nın algoritmasının sonuçları	85
E4	Yüklü ortakat grup seçicisinden başlayan aramada teklif edilen algoritmanın sonuçları.....	86
E5	Ardışıl arama yapılarak S.Ohta ve H. Yueda'nın algoritmasıyla elde edilen sonuçları.....	87
E6	Ortakat grup seçicisini ardışıl aramada teklif edilen algoritmanın sonuçlar.....	88
E6	Rastgele bir noktadan başlayıp arayan S.Ohta ve H. Yueda'nın algoritmasının sonuçları.....	89
E6	Rastgele bir noktadan başlayıp arayan teklif edilen algoritmayla elde edilen sonuçlar.....	90
E9	S. Ohta ve H. Yueda' nın 1000 deneme sonucu.....	91
E10	Teklif edilen yöntemle 10000 deneme sonucu.....	92
E11	S.Ohta ve H. Yueda'nın algoritmasının programı.....	93
E12	Teklif edilen algoritmanın programı.....	101

ÖNSÖZ

Bu çalışmamın hazırlanmasında ve ortaya çıkmasında değerli zamanlarını ayırıp, çalışmalarında, beni cesaretlendirerek destekleyen ve çalışmalarımı yakından sabırla takip eden Değerli Hocam Sayın Prof. Dr. Günsel DURUSOY'a minnet duygularımı sunar, teşekkürü borç bilirim.

Ayrıca Çalışma seçiminde beni yönlendiren ve yardımlarını esirgemeyen Değerli Hocam Sayın Prof. Dr. Atilla ATAMAN'a ve sıkıntılarımı paylaşan arkadaşlarıma teşekkür ederim.

Sıtkı ÖZTÜRK

Ekim-1995, İzmit

ÖZET

SPC santrallarda yeniden düzenleme yapılabilir. Çok katlı bir link sisteminde yeniden düzenleme ile orta kat grup seçicilerinin bazıları üzerindeki bağlantılar değiştirilerek blokaj azaltılır. İlk uygulamalarda ortakat grup seçicilerinin ikisi arasında bağlantıları değiştiren algoritmalar kullanılmıştır. Ancak S. Ohtave H. Yueda, daha büyük kapasiteli santrallarda ortakat grup seçicilerinin önerdikleri bir kurala göre seçilmesi halinde, bütün ortakat grup seçicilerinin kullanılabileceğini göstermiş ve bir algoritma geliştirmişlerdir.

Bu tezde geliştirilen algoritma, adı geçen araştırmacıların çalışmalarından hareket edilerek ortaya atılmıştır. Burada, ortakat grup seçicilerinin seçiminde herhangi bir şart ileri sürmeden, bütün ortakat grup seçicileri üzerinden yeniden düzenleme yapıp yapılamıyacağı araştırılarak sonuçta, gerekirse bütün ortakat grup seçicilerinin bağlantılarını değiştiren yeni bir algoritma geliştirilmiştir.

Bu algoritma üç katlı bağlaşma şebekelerinde meydana gelebilecek iç blokajı kaldırmak için, bugüne kadar ortaya atılan algoritmalarından farklıdır. Geliştirilen algoritmanın yazılımı yapılarak simülasyon sonuçları incelenmiş; bu sonuçlar daha önceki çalışmaların sonuçlarıyla karşılaştırılmış ve aralarındaki fark ve önerilen algoritmanın elverişli yanları ortaya konmuştur.

Tez çalışması dört bölümden oluşmaktadır.

Birinci bölüm giriş bölümü olup, bağlaşma şebekelerinin yeniden düzenlenmesi ile ilgili gerekçeler ve temel bilgiler verilmiştir.

İkinci bölümde bağlaşma şebekeleri ile ilgili temel kavramlar anlatılmakta; özellikle üç katlı bağlaşma şebekeleri incelenmektedir. Burada zaman bölmeli seçicilerin, uzay bölmeli seçicilere eşdeğerliği belirtilmekte ve ilerdeki bölümlerde uzay bölmeli seçiciler kullanılarak gerçekleştirilen yeniden düzenleme algoritmalarının, zaman bölmeli seçicilerde de kullanılabileceği gösterilmiştir.

Üçüncü bölümde ise, yeniden düzenlemenin teorik incelemesi yapılmıştır. Burada, iç blokajın kaldırılması için bağlantılardaki değişikliklerin nasıl gerçekleştirilmesi hakkındaki M.C. Paull'un teoremi açıklanmıştır. Ayrıca bugüne kadar geliştirilen yöntemler ve algoritmalarından örnekler verilmiştir.

Dördüncü bölümde, yeni geliştirilen özgün algoritma ayrıntılı olarak açıklanmakta ve bu algoritmanın bilgisayar programı verilmektedir. Yine bu bölümde S.Ohta ve H.Yueda'nın geliştirdiği algoritmanın bilgisayar programı ve buradan alınan sonuçlar yer almaktadır. Geliştirilen algoritmadan ve S. Ohta ve H. Yueda 'nın algoritmasından elde edilen sonuçlar karşılaştırılmakta, tezde teklif edilen algoritmanın elverişli yönleri belirtilmektedir. Yeni geliştirilen algoritma için C++ 'da yazılan bilgisayar programları ek olarak verilmiştir.

Beşinci bölümde ise, çalışmanın sonuçları değerlendirilmiştir.

ABSTRACT

Rearrangement algorithms can be used at the optimization of switching networks, that using communication systems. Rearrangement algorithms, generally were realized that changing of the connection of two middle switches. However, the middle switches that mentioned and emphasized at the studies of S. Ohta and H. Yueda (1987) can be selected according to a determined rule, and can be used all middle switches.

At the new algorithm developed in this thesis, the studies of the predecessor researches were accepted as main idea. However, proposed algorithm is different from conventional algorithms. But, at the selection of the middle switches without any conditions searched; can be a new arrangement or not on the all middle switches, and was developed an original algorithm.

This algorithm is different from the conventional algorithms to eliminate the internal blocking that can occur the three stage switching networks, software of this algorithm are made and simulated. These simulation results were compared by the results of predecessor algorithms and made the differences between the results, clear.

This thesis was completed in four chapters.

The first chapter contains introduction of fundamental of the rearrangement switching networks.

In the second chapter, basic concepts of the switching networks are given. Especially, three-stage switching networks are examined. Details of the space division switches of time division switches were described. Rearrangement algorithms that will be used for the space division switches in the ahead chapter mentioned that these algorithms can be used for the time division switches.

Then, in the third chapter rearrangement was theoretically examined. It was explained that where and how the changes of connection for the eliminating of internal blocking should be realized. Besides, some examples of conventional methods and algorithms of this event were given.

In the fourth chapter, results of previous algorithms software were compared. Advantages and disadvantages of the algorithms were scrutinized. All software was realized in C++ language and was given at the end.

Finally in the fifth chapter of the thesis results, conclusions, and recommendations were given.

BÖLÜM 1

1.0 GİRİŞ

Bir haberleşme şebekesi, kendisine bağlı abonelerin birbiriyle bağlanıp haberleşmesini sağlar. Yerel bir santral, kendi abonelerini birbirlerine bağlaştırmının yanısıra, başka yerel santral abonelerine de bağlaştırmalarına imkan verir.

Bir haberleşme şebekesi bağlaşma düğümleri diye adlandırılan santrallardan ve bunları bağlayan iletim yollarından oluşur. Günümüzde haberleşme şebekelerinde çok çeşitli (elektromekanik, elektronik, paket, ISDN santralları gibi) santrallar bulunur. Bir santralin içinde çeşitli seçme organları ve bunları bağlayan imkanlar, bir şebeke oluşturur. Bu şebeke "bağlaşma şebekesi" veya "bağlaşma sistemi" diye adlandırılır. Bağlaşma düğümleri arasındaki bağlantılar, farklı türdeki kablolardan, radyolink düzeneklerinden, uydulardan ya da tüm bunların karışımından oluşan iletim yolları ile sağlanabilir. Santralların ve yolların verimli olarak kullanılması istenir. Bu amaçla haberleşme şebekesi eniyilenir.

Haberleşme şebekesinin eniyilenmesinde:

- 1- Bağlaşma şebekesinin eniyilenmesi,
- 2- İletim yollarının eniyilenmesi, söz konusudur.

Bağlaşma şebekesi eniyilenmesi, istenen koşullara göre devre sayılarının (hat ya da jonksiyon kapasitelerinin) eniyi biçimde belirlenmesi anlamındadır. İletim yollarının eniyilenmesi problemi, bağlaşma şebekesi eniyilenmesinin çıktılarını kullanır ve iletim sistemiyle ilgili koşullara göre yapılır. Bunun yanında bir haberleşme şebekesinin eniyilenmesi, çok santrallı bir bölgede santral yerlerinin ve lokal abonelerin belirlenmesi veya santrallar arası yol bulmada yol bulma algoritmalarının seçilmesinde de söz konusu olabilir.

Santrallarda blokaj, "imkan bulunamaması nedeni ile çağırma isteğinin karşılanmaması" olarak tanımlanmaktadır. İki türlü blokaj sözkonusu olabilir:

1. Dış blokaj,
2. İç blokaj.

Dış blokaj durumunda, santraldaki tüm eş zamanlı bağlantı imkanları kullanılmıştır. Yeni bir bağlantı isteği karşılanamaz.

İç blokaj ise, santral içindeki bağlantıların dağılımından doğan ve çok katlı bağlaşma şebekelerinde sıkça görülen bir blokaj şeklidir. Bu gibi durumlara rastlayan çağırmalar kaybolduğu için taşınan trafik (offered traffic) yükü sınırlanır. Yük sınırlamasını azaltmak için,

1. Donanım, iç blokaj olasılığını küçük tutacak ve eş zamanlı bağlantı sayısını artıracak yönde düzenlenebilir. Bu, donanımın maliyetini artırır.
2. Günümüz teknolojinin sunduğu Kaydedilmiş Program Kontrol'lu (SPC≡Stored Program Controlled) santrallardaki esneklikten yararlanılır ve yazılıma "Yeniden Düzenleme Programı" eklenir.

Tezde, yeniden düzenleme ile blokajın azaltılması ele alınmıştır. Bu amaçla 2. bölümde bağlaşma ile ilgili kavramlar ve özellikle üç katlı bağlaşma şebekeleri incelenmiştir. 3. bölümde yeniden düzenlemenin teorik incelemesi

yapılmış; 4. bölümde SPC santrallarda üç katlı bağlaşma şebekesinde bağlantıları yeniden düzenleyerek iç blokaj olasılığını azaltan, dolayısı ile taşınan trafiği artıran bir çözüm önerilmiştir.

Bağlaşma şebekelerinde yeniden düzenleme C. Clos (1953)'un [2][4], blokajsız şebekenin teoremi ile başlamış sayılmaktadır. D. Slepian (1952), yaptığı çalışmada n , giriş ve çıkış grup seçicisinin giriş-çıkış sayısı olmak üzere, maksimum yeniden düzenlemedeki bağlantı sayısının $2n-2$ olduğunu; M.C. Paull (1962) [19], bu sınırın $n-1$ olacağını ve iki ortakat grup seçicisinin bağlantılarını değiştiren yeniden düzenleme yöntemlerini ileri sürmüştür. V. E. Benes (1965) [3], yeniden düzenlemenin matematiksel teorisini geliştirmiştir. V. I. Neiman (1969) [14], kurulu olan çağırma şebekesinin yeniden düzenlenmesi için, yalnız giriş ve çıkış grup seçicilerini saklayarak bütün ortakat grup seçicilerinin bağlantılarını yeniden oluşturan bir algoritma vermiştir. G. Hebuterne de kitabında (1987) [7], bu çalışmayı yeniden düzenleme algoritması olarak benimsemiştir. K. Gotoh (1978) [5], zincir notasyonunu kullanarak, ZUZ şebekesinde bir yeniden düzenleme algoritması geliştirmiştir. A.Jajszczyk ve J. Rajski (1980) [10], S. Ohta (1987) [17], seçiciler arasında birden fazla bağlantı olan (PCM) şebekeler için yeniden düzenleme algoritması çalışmaları vardır. S. Ohta ve H. Yueda zincir notasyonunu kullanarak yaptıkları simülasyonda 9000 giriş ve çıkışlı şebeke için yeniden düzenlenen bağlantı sayısının 20'den az olduğunu göstermişlerdir [16][18].

Tez de yeniden düzenleme için farklı bir yaklaşım getirilmiştir. Şöyle ki: Santral bağlantıları rastgele grup seçicileri üzerinden yapıldığından iç blokajı kaldırmak için, üç katlı bağlaşma şebekesinin ortakat grup seçicilerinin tümünü gözönüne alan bir yeniden düzenleme algoritması teklif edilmiş; sonuçları diğer çalışmalarla karşılaştırılmış bazı yönlerden daha elverişli olduğu görülmüştür.

BÖLÜM 2

2.0 BAĞLAŞMA ŞEBEKELERİ

Bağlaşma düğümü veya santral diye adlandırılan bağlaşma merkezleri, iki önemli fonksiyonu yerine getirmektedir. Birincisi, sistemin herhangi bir abonesinin diğer herhangi bir abonesini seçmesini sağlamak; diğeri ise haberleşme sistemindeki haberleşme kanallarının sayısını azaltmaktır. Santral seçme ve bağlama işlemini uç cihazlarının içediğı organlar yardımı ile santraldaki aletlere uzaktan kumanda ederek, yani "otomatik" olarak sağlar. Bir telekominikasyon zincirinde bağlaşma fonksiyonlarını yerine getiren bu alt sistemlere "telekominikasyon bağlaşma sistemleri" adı da verilir.

2.1 BAĞLAŞMA SİSTEMLERİNİN SINIFLANDIRILMASI

Bağlaşma sistemleri "devre bağlaşma sistemleri" ve "hafızalı bağlaşma sistemleri" diye iki ana gruba ayrılmıştır.

Devre bağlaşmasında, haberleşme başlamadan önce uç noktaları arasında devre veya kanal diye adlandırılan bir haberleşme yolu kurulur ve bu yol haberleşme süresince hizmette kalır. Haberleşme yolu uzayda, zaman veya frekans domeninde kurulmuş olabilir.



Hafızalı bağlaşma, işleyiş biçimi nedeni ile "kaydetme (depolama) ve ileriye gönderme" veya kısaca "sakla ilet" yöntemidir. Burada uç noktaları arasında gerçek zamanda bir bağlantı ve bir etkileşim yoktur.

2.2 BAĞLAŞMA

Çağırılan ve çağırılan abonelere ait uç cihazları arasında konuşma işaretlerinin iletimine elverişli bir yol, bir konuşma yolu, çok çeşitli bağlaşma teknikleri ve çok çeşitli bağlaşma organları yardımı ile kurulabilir.

Bağlaşma organlarına "seçici" adı verilir. Seçici bir girişi veya bir çok girişten birini, bir çıkışa veya birçok çıkıştan birine bağlaştıran bir alettir. Genel olarak, uzayda çoğullanmış yollar kuran seçiciler (uzay seçicileri) ve zamanda ayrılmış yollar kuran seçiciler (zaman seçicileri) diye kurulan yollar bakımından ikiye ayrılır. İlk otomatik santrallardan (1892) bu güne kadar çok çeşitli seçiciler kullanılmıştır. Elektronik santrallarda karşımıza, değişik yapılarda matrisler şeklinde düzenlenmiş seçiciler çıkmaktadır.

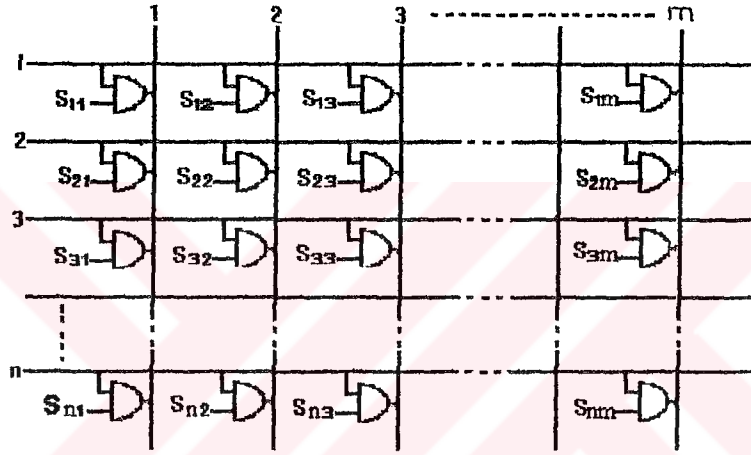
2.2.1 KESİŞMELİ (MATRİSEL) SEÇİCİLER

Kesişmeli seçicilerde yatay ve düşey yollar ve bu yolların kesişme noktalarında yer alan kontaklar bulunur. Kontaklar, mekanik veya elektronik olabilir. Elektronik kontaklar, seçicilerde imkanların daha iyi kullanımı için çalışmaya konu olan yeniden düzenleme gibi yöntemlere imkan verir.

Elektronik Kesişmeli Seçiciler

Şekil 2.1.'de zaman çoğullamalı elektronik kesişmeli seçicinin basitleştirilmiş devresi görülmektedir. Bağlantı, bağlantı kurulacak yolların kesişme

noktasındaki lojik kapının kontroluyla sağlanır. Seçicide, satırlar girişleri sütunlar da çıkışları göstermekte, kesişme noktasındaki lojik kapı, bu satır-sütun numaraları ile belirtilmektedir. Şekilde görüldüğü gibi seçici, "n" satırdan ve "m" sütundan oluşan $n \times m$ boyutunda bir matristir. Bu matrisdeki girişler ve çıkışlar zamanda çoğullanmış işaretler taşıyan hatlar (PCM hatları) olabilir. Zaten sayısal elektronik santrallarda bu tür bağlaşma söz konusu olup, Şekil 2.2.'de de bu kavram açıklanmaktadır.

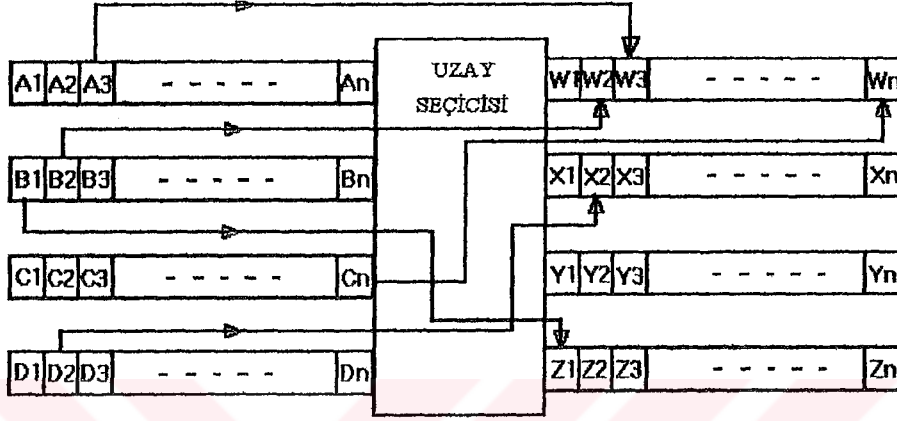


Şekil 2.1. $n \times m$ giriş-çıkıştan oluşan elektronik kesişmeli seçici.

Örneğin "B" girişinin 1. zaman dilimi, "Z" çıkışının 1. zaman dilimine ve "C" girişinin n. zaman dilimi, "W" çıkışının n. zaman dilimine bağlanmıştır. Bağlaşma, elektronik kesişmeli seçicinin, "B" girişini "Z" çıkışına bağlayan lojik kapının 1. zaman dilimi süresince S_{BZ} girişi ve "C" girişini "W" çıkışına bağlayan lojik kapının n. zaman dilimi süresince S_{CW} girişi kontrol edilerek sağlanır. Bu örnekteki bağlaşmalarda görüldüğü gibi giriş-çıkış zaman dilimleri arasında değişiklik söz konusu olmayıp, giriş zaman dilimleri aynı veya farklı çıkışların aynı zaman dilimlerine bağlantısı mümkündür. Uzak seçicilerinde, herhangi bir zaman dilimi süresinde kurulan eş zamanlı bağlantı

sayısı, en fazla küçük boyuta ("n" ve "m" 'den küçük olanına) eşit veya daha az olabilir.

Zamanda çoğullanmış işaretleri bağlaştıran bir kesişmeli seçici, uzayda çoğullanan zaman dilimleri kadar büyüklükte seçici gibi düşünülebilir.



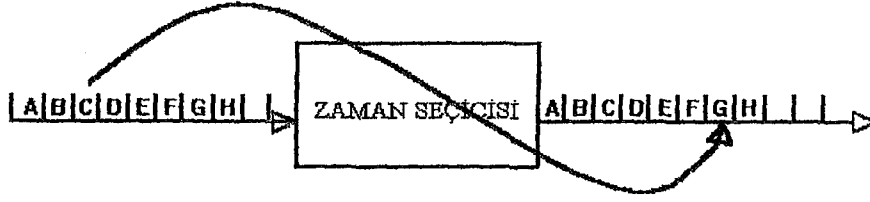
Şekil 2.2. Birden fazla zaman çoğullamalı PCM giriş-çıkış'larının, elektronik uzay seçicisi kullanarak kanal dilimlerinin bağlantısının gerçekleştirilmesi.

Zaman Seçicileri

Bu gün elektronik santralların çok büyük bir kısmı sayısal santrallardır. Bu santrallarda bağlaşma uzayda ve zamanda olmak üzere iki çeşit gerçekleştirilir. Zaman seçicileri belli bir zaman kanalındaki bilginin istenen bir zaman kanalına bağlaştırılmasını sağlar. Yapılan işlem "zaman kanalı değiş tokuşu" diye de adlandırılır.

Bir zaman bağlaşması basit olarak Şekil 2.3. de verilmiştir. PCM sisteminde bir çerçeve 125 μ s'dir. Avrupa standartlarında her çerçevede 30+2 kanal olduğundan, bir kanalın süresi 3.906 μ s'dir. Zaman kanalı değişikliği, bilgi içeren herbir giriş PCM çerçevesinin zaman kanalındaki bilgiler, çıkış PCM çerçevesinde gideceği yere göre zaman kanalına aktarılmasıdır. Dolayısıyla,

seicide uygun řekilde baėlanan ıkıřtaki kanalların iletimi iin yeni bir ıkıř PCM ercevesi oluřturulmuř olur.



řekil 2.3. Bir zaman seicisinde PCM giriř-ıkıřının kanal dilimlerinin baėlantısının gerekleřtirilmesi.

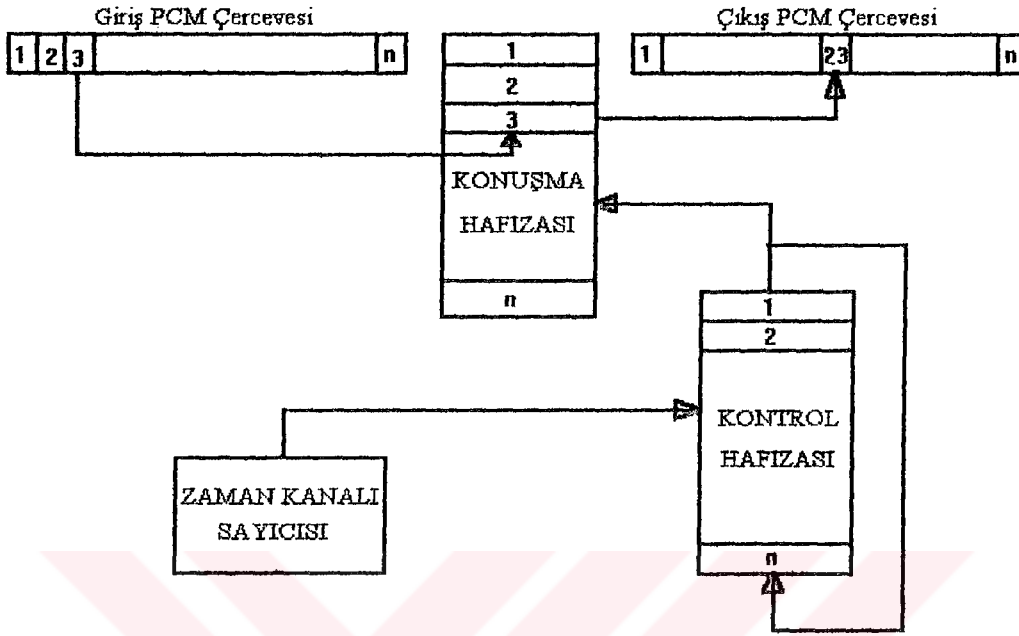
Giriřin belli bir zaman diliminde tařınan bilgi bir hafızaya kaydedilerek, istenen ıkıř zaman dilimi geldiėinde okunur. Bylece ıkıřın istenen zaman kanalına baėlařması yapılmıř olur. Bu iřlemler bazı yntemlerle kontrol edilir ve zaman diliminin serbest veya meřgul olduėu gibi bilgiler bu kontrollerde hafızada tutulur. Zaman baėlařması řekil 2.4.'de grldėu gibi  temel kattan oluřur [4].

- PCM Konuřma kanalları iin hafıza,
- Giriř-ıkıř konuřma kanallarının durumunu gsteren kontrol hafızası,
- Zaman dilimlerini sayan sayıcı veya mikroiřlemci.

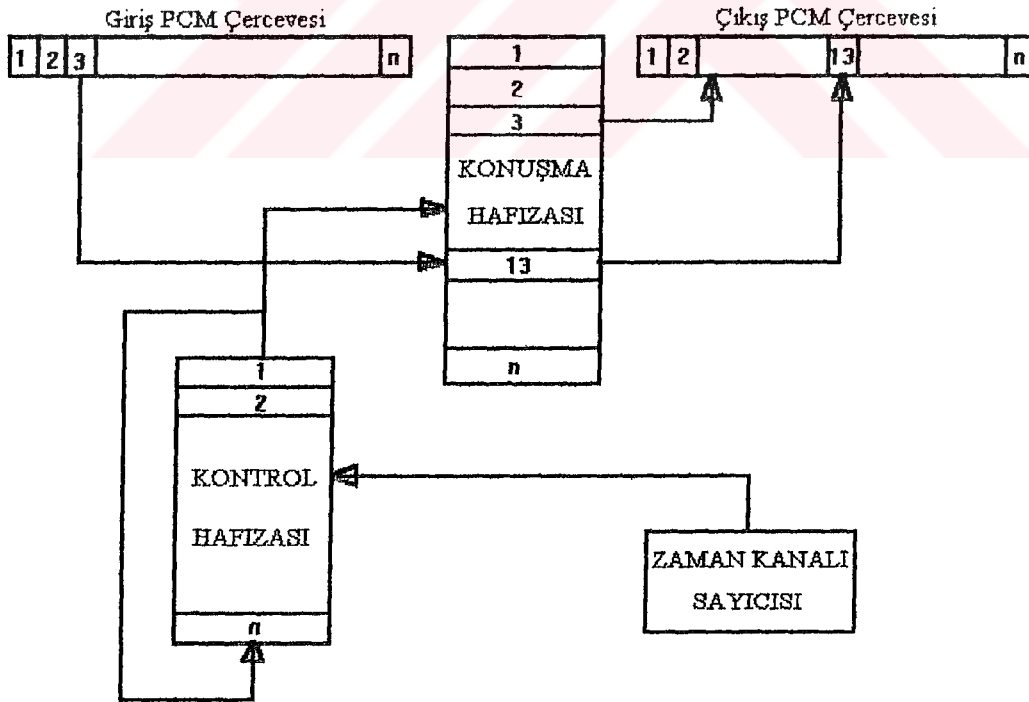
Giriř PCM kanallarının herbiri, řekil 2.4a.'da grldėu gibi ardıřıl veya řekil 2.4B.'de grldėu gibi rastgele olmak zere iki farklı řekilde konuřma hafızasına yazılabilir. Ardıřıl yntemde konuřma hafızasındaki kanalların diziliř sırası, giriř PCM kanal sırasıyla aynı olup, ıkıř PCM kanal sırasına gre rastgeledir (řekil 2.4a.). Rastgele yntemde konuřma hafızasındaki kanalların diziliř sırası, ıkıř PCM kanal sırasıyla aynı olup, giriř PCM kanal sırasına gre rastgeledir (řekil 2.4b.).

Konuřma hafızasında, giriř PCM ercevesinde bulunan zaman kanalı kadar hafıza birimi ayrılmaktadır. Bu hafızalara kanalların yazılımı, zaman kanalı

sayıcısı ile okunması, kontrol hafızasındaki bilgilere göre veya tersine olabilir.



Şekil 2.4a. Zaman seçicisinde ardışıl yöntem. Ardışıl yazma, rastgele okuma.



Şekil 2:4B. Zaman seçicisinde rastgele yöntem. Rastgele yazma, ardışıl okuma.

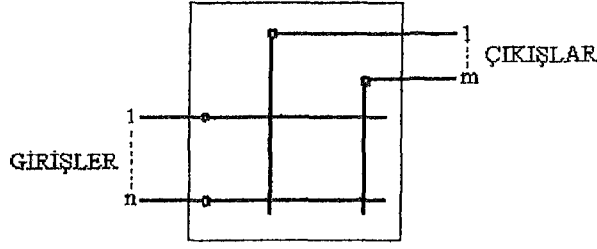
Bir zaman seçicisi giriş zaman dilimlerine eşit sayıda girişi ve çıkışı zaman dilimlerine eşit sayıda çıkışı olan bir uzay seçicisi gibi düşünülebilir. Böylece santralların bağlaşma devrelerini uzay veya zaman katları olup olmadıklarına bakılmaksızın modelleyebilmek mümkün olur.

Genellikle eşzamanlı bağlantı gereksinimi tek bir PCM hattının sunduğu 30 imkandan fazlasını gerektirir ve çok sayıda PCM hattı kullanılır. Problem, "n tane" PCM giriş hattının herbirinin taşıdığı zaman kanallarını, "m tane" ($n=m$ olabilir) çıkış PCM hattının taşıdığı zaman kanallarına bağlaştırmaktır. Örneğin 1.giriş PCM hattının 3. zaman kanalı, 5. çıkış PCM hattının 23. zaman kanalına bağlaştırılabilirdir.

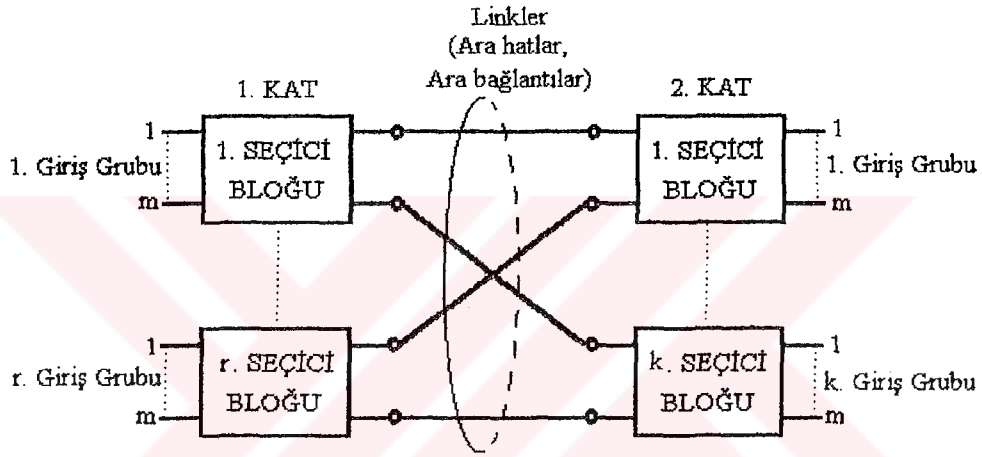
2.3 LİNK SİSTEMLERİ

Seçicilerle kurulan çok katlı bağlaşma devreleri "Link Sistemleri" diye adlandırılır. Link sistemlerinin prensibi, çeşitli seçme katları üzerinden bağlaşmanın eşzamanlı olarak sağlanmasıdır. Bu, özel bir kontrol cihazını ve özel bir kontrol biçimini gerektirir. Crossbar sistemlerinde marker diye bilinen bu cihaz, hangi seçicilerin boş olduğu ve bunların nasıl bağlanabileceğini bilmek zorundadır. Daha sonra geliştirilen elektronik sistemli SPC edilen santrallarda ise işlemci bu görevi üstlenmiştir.

Bu gün link sistemi prensiplerine göre çalışan bağlaşma devrelerinde genellikle eşlenik seçme kullanılmaktadır. Girişlerin herbirinden çıkışların herbirine erişebilen bağlaşma organlarına bağlaşma bloğu, seçici bloğu veya grup seçicisi adı verilir (Şekil 2.5.). Bu seçici blokları ile kurulan devreler, link sistemlerini oluştururlar. Dolayısıyla linkler, trafik gereksinimlerine uygun olarak istenen sayıda giriş ve çıkış elde etmek için seçici bloklarının iki veya daha fazla katının birbiri ile bağlanması ile oluşan devredir (Şekil 2.6.).



Şekil 2.5. n giriş ve m çıkıştan oluşan seçici bloğu.



Şekil 2.6. r giriş k çıkış grubundan oluşan link sistemi.

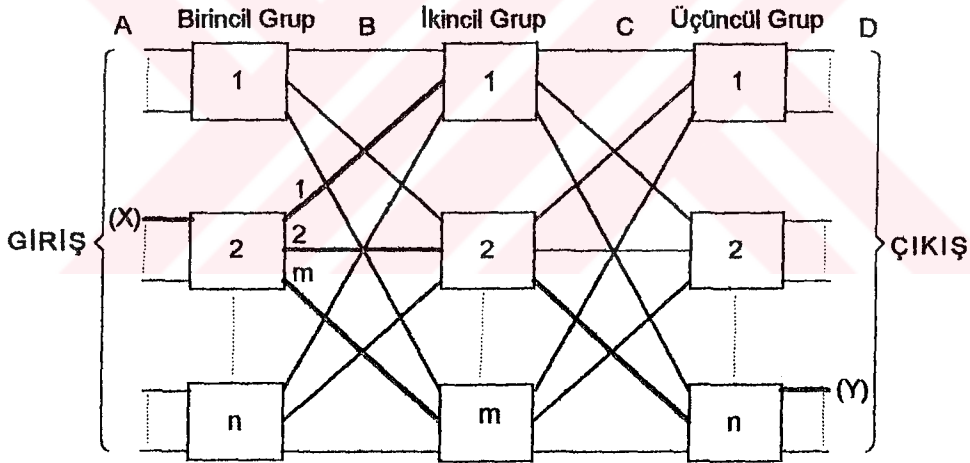
Bir link sisteminde tek bir seçicide olduğu gibi bir girişten bir çıkışa doğrudan erişim mümkün değildir. Bağlantı bir veya daha fazla kat üzerinden geçerek gerçekleştirilir ve eşlenik seçme denilen, girişten çıkış veya çıkışlarına bağlantının ancak ve ancak link sistemi üzerinde bir yol bulmak mümkün ise kurulduğu bir seçme uygulanır.

Link sistemlerinin önemli bir özelliği iki yönlü olabilmesidir. Bağlantı bir giriş ve bir çıkış olan iki uç arasında gerçekleştirileceğinden, bir girişin bir çıkış olarak veya bir çıkışın bir giriş olarak göz önüne alınabilmesi mümkündür. Şekil 2.6.'daki sistem bir giriş ve bir çıkış arasında sadece bir tek yol verir. Bağlaşma blokları genellikle üniform yapıdadır ve link grupları

arasında düzenli olarak dağılmıştır. Bu kontrol ve işletme yönünden büyük kolaylık sağlar.

2.3.1. ÇOK KATLI LİNK SİSTEMLERİ

Trafik gereksinimlerine ve seçici yapısına bağlı olarak bir link sistemi 3,4 ve hatta daha fazla kattan oluşabilir. Yani bir link sistemi üzerinden bağlantı 3,4 veya daha fazla seri kesişme noktası üzerinden kurulabilir. Özellikle küçük boyutlu seçme matrislerinden kurulan elektronik santrallarda kat sayısı fazla olmaktadır. Şekil 2.7.'de üç katlı bir yapı görülmektedir. İkinci kat karıştırma (yöneltme) görevi yapmakta ve bir girişin belli bir çıkışa daha fazla yol üzerinden erişmesine imkan sağlamaktadır.



Şekil 2.7. Üç katlı link sistemi. A, giriş; B, link; C, link ve D, çıkış katlarını göstermektedir.

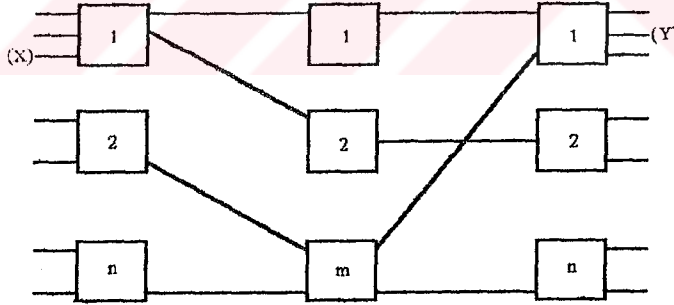
Şekilde, (X) işaretli girişten bir çağırma geldiğini ve kontrol sisteminin saptadığına göre (Y) işaretli çıkışa erişmek istediğini varsayalım. Bu bağlantı (1), (2) veya (m) linkleri üzerinden sağlanabilir. Bir ara kat koymakla belli bir girişten belli bir çıkışa ulaşım imkanları artırılmış, örneğimizde (m)'ye

çıkartılmıştır. Ancak belli bir ikincil gruba ulaşan çağırma için tek bir seçenek vardır.

Sayısal santrallarda yer alan, uzay ve zaman katlarının çeşitli kombinezonlarından oluşan bağlaşma devreleri de bir link sistemi gibi düşünülebilir. Bu kombinezonlar, UZ, ZU, UZU, ZUUUZ, ..., gibi değişik uygulamalar olabilir. Örneğin, ATT'nin No=4 ESS santralında ZUUUUZ, No=3 EAX santralında ise UUZUU ve Northern Telecom'un DMS-100 santralında ZUZU şeklinde seçici katlar kullanılarak link sistemleri gerçekleştirilmiştir[].

2.3.2 LİNK SİSTEMLERİNDE TIKANMA (BLOKAJ)

İki veya daha fazla katlı düzenlemeler, link sistemlerinde tıkanma, yığılma veya blokaj denilen olaya neden olur. Blokaj, bir girişin istenen ve boş olan çıkışa, yol bulunamaması dolayısı ile erişememesi olasılığı olarak tanımlanır.



Şekil 2.8. Üç katlı link sisteminde (X),(Y) çağırmanın blokajı.

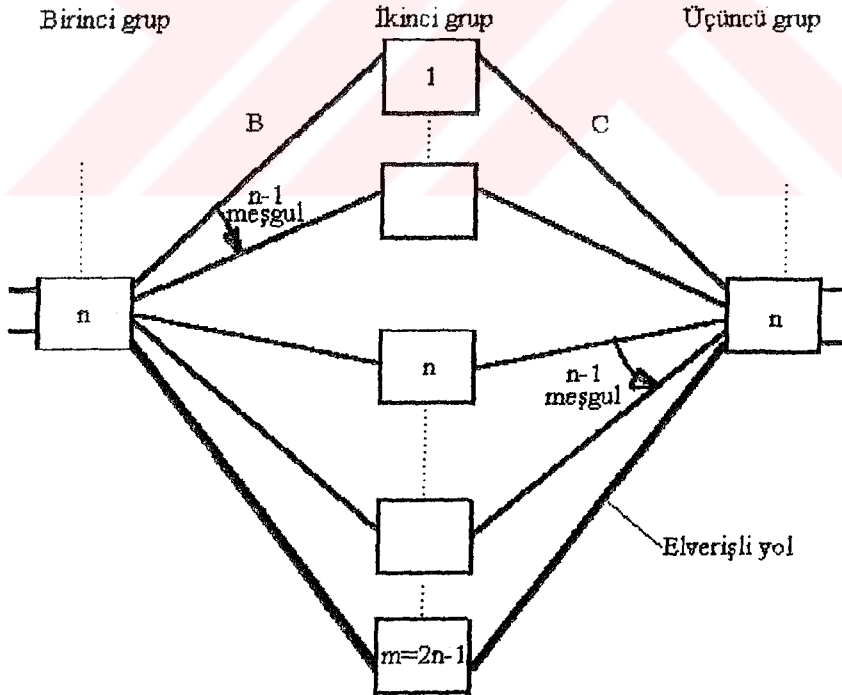
Şekil 2.8.'de bir giriş grubundan gelen çağırma, çağırılan çıkışın bulunduğu ikinci basamak seçişisine erişecek olan link, daha önce başka çağırımlar tarafından meşgul edilmişse yol bulamaz ve blokaj olur. Belli bir doğrultuya olan çıkışlar söz konusu ise yığılma, o doğrultudaki çıkışlardan boş olanların bağlanabildiği linklerin tümü meşgul ise ortaya çıkar.

Tek katlı düzenlemelerde blokaj söz konusu değildir. Bir bağlantının gerçekleştirilmemesi ancak ve ancak istenen çıkışın meşgul olması durumunda ortaya çıkar.

Çok katlı link sistemleri seçicilerin uygun şekilde boyutlandırılması ve seçici sayılarının uygun olması durumunda, blokajsız olarak gerçekleştirilebilir. Bu konuda Clos (1954) giriş trafiği ne olursa olsun blokajsız link sistemin gerçekleştirileceğini göstermiştir.

2.3.3 CLOS TİPİ LİNK SİSTEMLERİ

Clos tipi sistemlerde karesel seçici boyutları $n \times n$ ise ikinci kat seçicilerinin sayısı $2n-1$; birinci ve üçüncü kat seçicileri sırasıyla n giriş $2n-1$ çıkış ve $2n-1$ giriş n çıkışlı olur. Şekil 2.9.'da tıkanma olmaması açıklanmaktadır. Burada

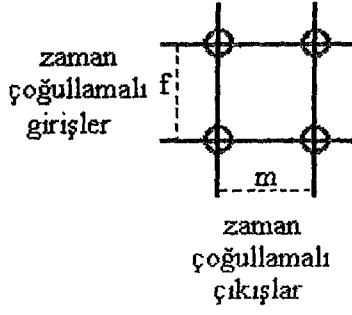


Şekil 2.9. Clos tipi "blokajsız" devre.

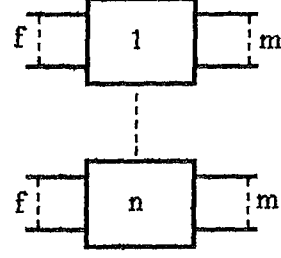
en kötü halde bile devre tıkanmamaktadır. Birinci gruptaki bir seçicinin halen $n-1$ girişi meşgul olsun ve bir n . çağırma gelsin. Çıkışların sayısı $2n-1$ olduğundan bu çağırma $(2n-1)-(n-1)=n$ boş link üzerinden bir ikincil grup seçicisine erişebilir. Çıkıştaki en elverişsiz durum ise, istenen çıkışın halen $(n-1)$ çıkışı meşgul olan bir seçicinin tek boş çıkışını ilgilendirmesidir. Bu boş çıkışa halen bir üçüncü kat seçicisine bağlanmamış olan $(2n-1)-(n-1)=n$ ikinci basamak seçicisi üzerinden n tane C linki ile erişebilir. Birinci basamaktan gelen $n-1$ meşgulliyet ve 3. basamağa erişen $n-1$ meşgulliyet ikinci basamakla tamamem ayrı seçiciler üzerinden geçse dahi yine de en az bir boş ikinci basamak seçicisi vardır ve istenen bağlantı onun üzerinden gerçekleştirilebilir. Bu şekilde Clos ikinci basamak seçicisinin $m \geq 2n-1$ alınması durumunda devrenin tamamen tıkanmasız olduğunu ispatlamıştır. Üç katlı Clos devrelerinde $m \geq n$ alınması durumunda tıkanmanın olacağı ancak, yeniden düzenleme yapılarak tıkanmanın kaldırılacağı daha sonraki yapılan çalışmalarda gösterilmiştir.

2.3.2 ZAMAN ÇOĞULLAMALI SEÇİCİNİN UZAY SEÇİCİ EŞDEĞERİ

Zaman çoğullamalı bağlaşma şebekesinin trafik taşıyıcı karakteristikleri uzay seçicilerine benzetilerek çözülebilir [9]. Öncelikle en basit zaman çoğullamalı Şekil 2.1.'deki bağlaşma şebekesi göz önüne alınabilir. Bu şebekeyi, Şekil 2.10a.'da gösterilen her biri n tane zaman dilimine sahip f girişli, m çıkışlı $f \times m$ kapıdan oluşan zaman çoğullamalı seçici olarak gösterebiliriz. Şekil 2.10a.'daki seçicinin, f girişinin herhangi birindeki bir zaman dilimi m adet çıkışın herhangi birindeki aynı zaman dilimine bağlanabilir. Böyle bir şebeke Şekil 2.10b.'de gösterildiği gibi n tane ayrık $f \times m$ boyutunda uzay seçicili şebekeye denk olacaktır. Burada zaman çoğullamalı kapılarla, farklı iki zaman dilimi arasında hiç bir bağlantının mümkün olmadığı açıktır. Şebeke sadece $f \times m$ boyutunda çalışır ve zaman çoğullamadan kaynaklanan bir avantaj kullanılamaz.



(a)

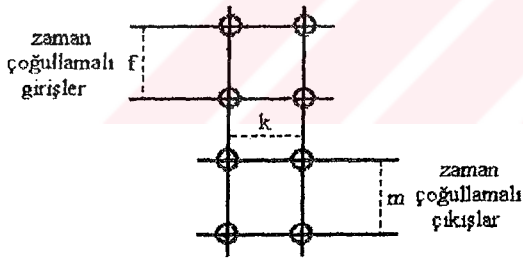


(b)

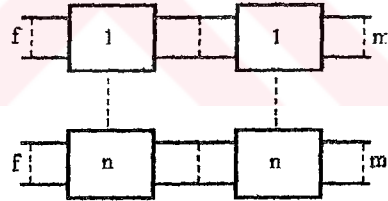
Şekil 2.10a. Zaman çoğullamalı seçici.

b. Zaman çoğullamalı seçicinin uzay seçici eşdeğeri

Şekil 2.10a.'da söz konusu dezavantaj daha açık olarak Şekil 2.11a'den görülebilir. Bu şebekenin uzay seçici eşdeğeri Şekil 2.11b.'de verilmiştir. Uzay seçici eşdeğerinden de görüldüğü gibi giriş veya çıkış katı grup seçicileri arasında bir bağlantı mevcut değildir.



(a)

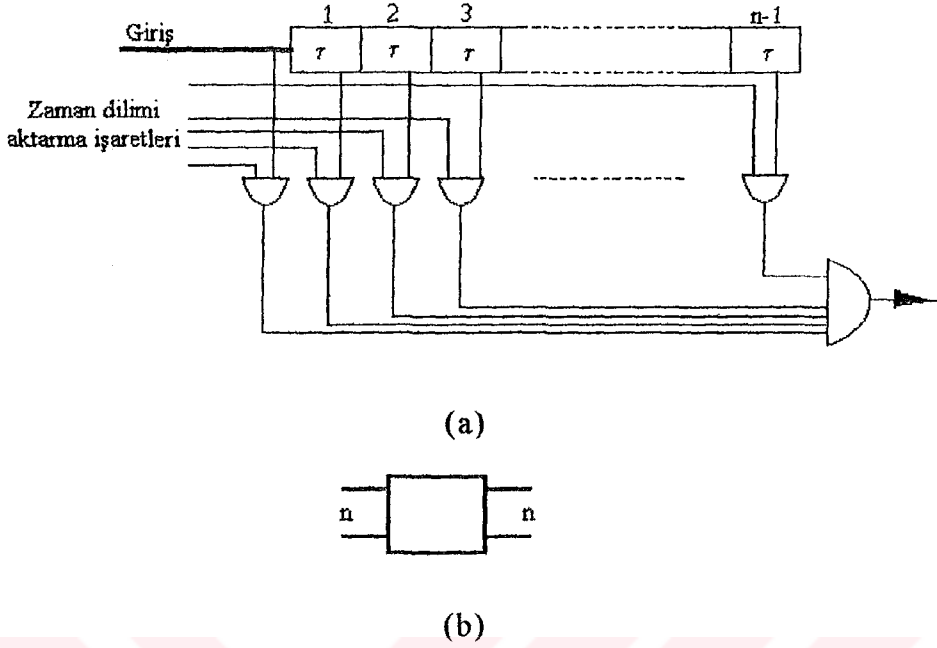


(b)

2.11a. İki katlı zaman çoğullamalı seçici.

b. İki katlı zaman çoğullamalı seçicinin uzay seçici eşdeğeri

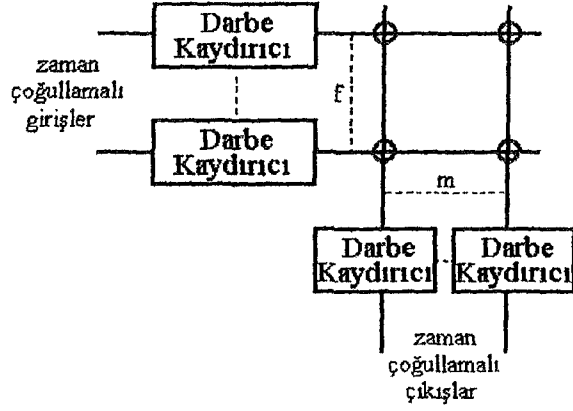
Zaman çoğullamalı şebekeye uygun gecikme devreleri ilave edilerek, esnek olmayan yapı iyileştirilir. Böylece çoğullanmış zaman kanalları 2.12a.'da görüldüğü



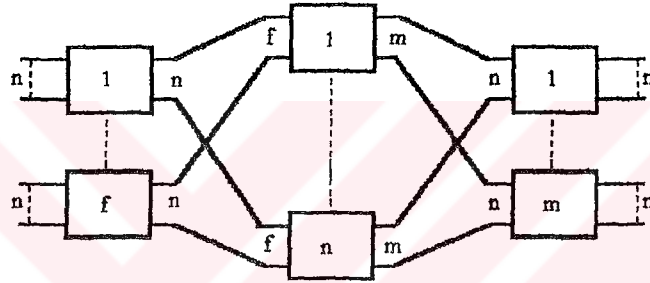
Şekil 2.10a. Darbe kaydırıcı.

b. Uzay seçici eşdeğeri.

gibi farklı zaman dilimine aktarılabilir. Girişdeki rastgele bir zaman dilimi, rastgele bir zaman dilimi ile yerdeğiştirebilen bir darbe kaydırıcı Şekil 2.12b.'de gösterilen $n \times n$ 'lik uzay çoğullamalı şebekeye eşdeğerdir. Şekil 2.10a.'daki şebekenin giriş ve çıkışında darbe kaydırıcı kullanarak (Şekil 2.13a.), girişteki herhangi bir zaman dilimi, çıkışta n tane zaman dilimlerinden herhangi birine kaydırılabilir. Bu şebeke, Şekil 2.13b.'de gösterilen üç katlı uzay seçicilerinden oluşur. Benzer şekilde eğer, Şekil 2.10a.'daki darbe kaydırıcı kat Şekil 2.11a.'daki zaman çoğullamalı katın arasında kullanılır (Şekil 2.14a.) ise Şekil 2.14b.'de gösterilen üç katlı uzay seçicili şebekeye eşdeğer olur. Zaman çoğullamalı bağlaşma şebekelerine ait problemler, yukarıda anlatılan zaman-uzay eşdeğeri kullanılarak rahatca çözülebilir. Örneğin, Şekil 2.14a.'daki şebeke, Şekil 2.14b.'deki uzay seçici eşdeğeri üç katlı blokajsız Clos şebekesi olarak düşünülürse $k \geq 2f-1$ alınır. Benzer şekilde, eğer en az $2n-1$ uzunluğunda bir darbe kaydırıcı, zaman

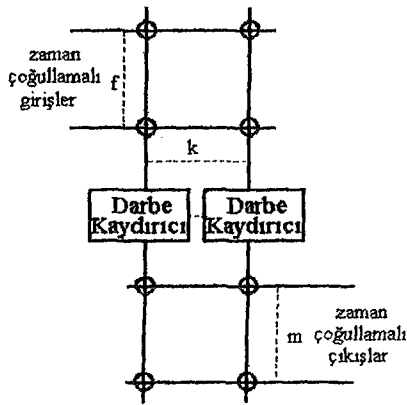


(a)

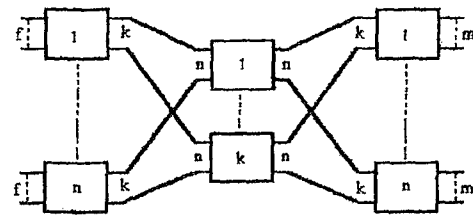


(b)

Şekil 2.13a. Darbe kaydırıcılı zaman çoğullamalı seçici
b. Uzak seçici eşdeğeri

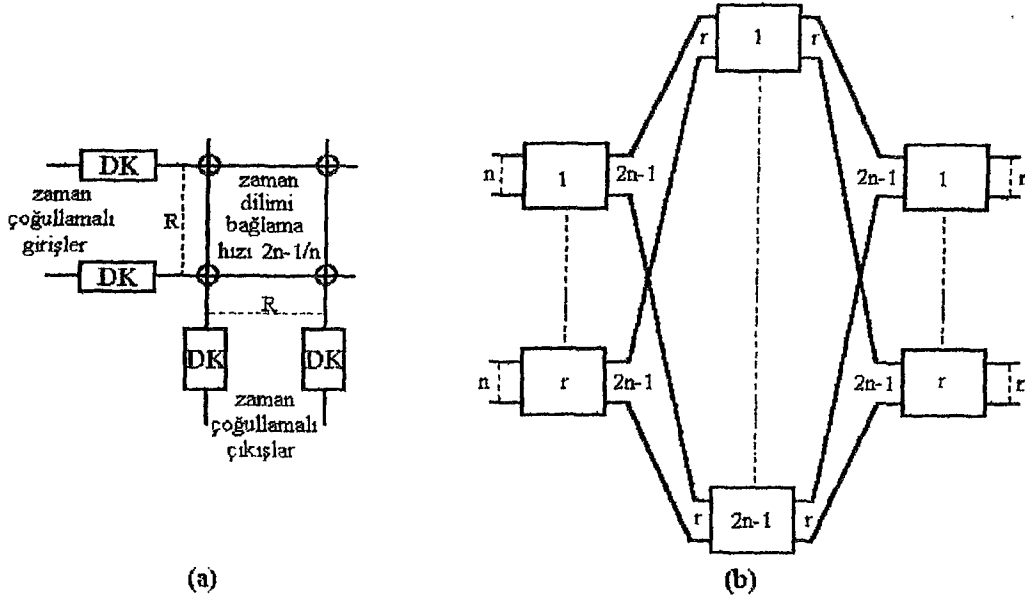


(a)



(b)

Şekil 2.14a. Darbe kaydırıcılı zaman çoğullamalı seçici
b. Uzak seçici eşdeğeri



Şekil 2.15a. Zaman çoğullamalı blokajsız seçicinin
b. Blokajsız uzay seçici eşdeğeri

çoğullamalı seçicinin giriş ve çıkışlarında kullanılır ise, Şekil 2.15a.'daki şebeke de blokajsız olacaktır. Bu şebekede iç saat hızı, dış saat hızından en az $(2n-1)/n$ kez olacaktır.

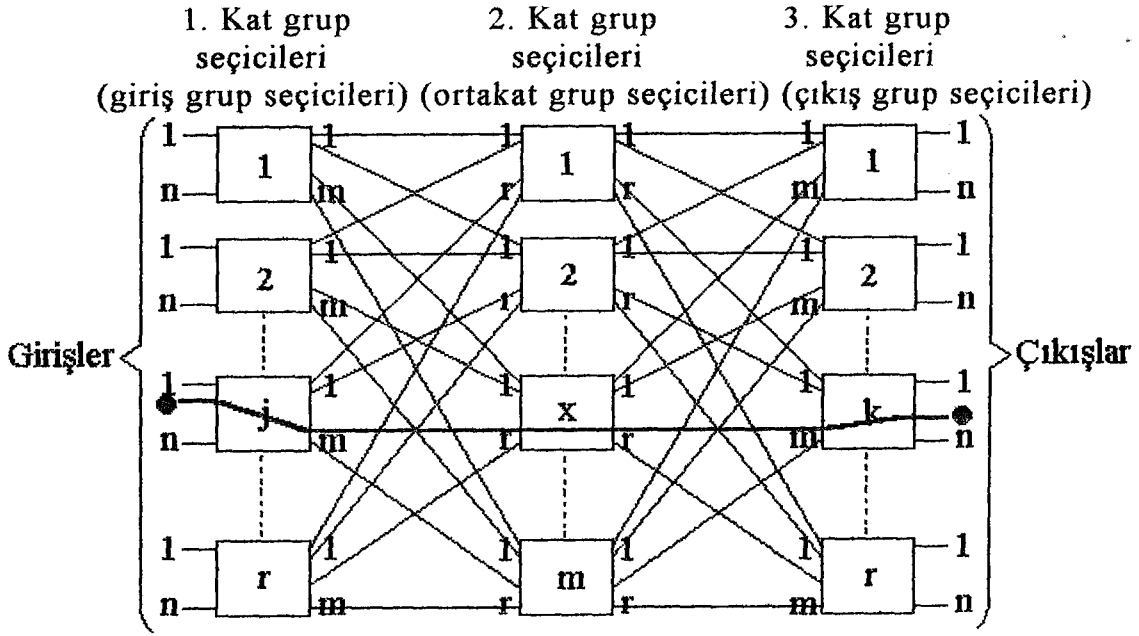
BÖLÜM 3

3.0 YENİDEN DÜZENLEME TEOREM VE ALGORİTMALARI

Bu bölümde konu ile ilgili geçmişteki çalışmalar kronolojik olarak verilmektedir. Öncelikle geçmişteki çalışmalardan en çok referans verilen ve daha sonraki çalışmalara taban oluşturan M.C.PAULL'un [19] çalışması ayrıntılı olarak incelenmekte, diğer çalışmalar da kısmen ele alınmaktadır.

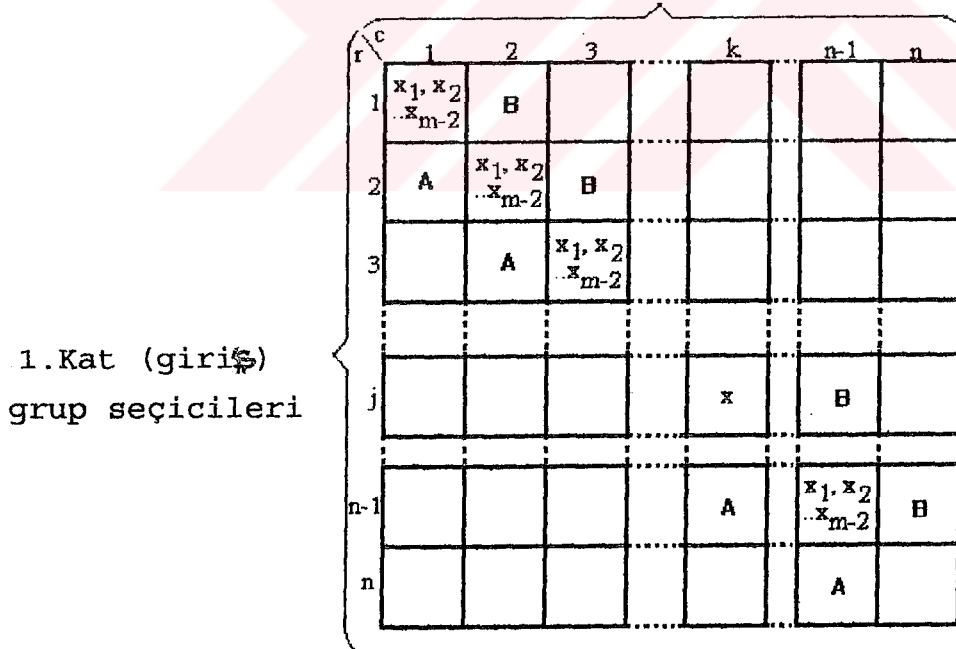
3.1 ÜÇ KATLI ŞEBEKEDEN YENİDEN DÜZENLENME

Şekil 3.1a'da verilen üç katlı genel şebeke örneği, Şekil 3.1b'deki $n \times n$ boyutlu matris eşdeğeri ile gösterilebilir. Matriste satır, bir giriş grup seçicisini, sütun ise bir çıkış grup seçicisini göstermektedir. Dolayısıyla matrisin satır ve sütun sayısı, sırasıyla şebekenin giriş ve çıkış grup seçicilerinin sayısı kadardır. Herbir giriş ve çıkış grup seçicisi, herbir ortakat grup seçicisine sadece bir tek hatla bağlandığından, matrisin herhangi bir satır veya sütununda m 'den daha fazla ortakat grup seçicisini sembolize eden harf olamaz. j 'inci giriş grup seçicisi ile k 'inci çıkış grup seçicisi arasındaki bağlantı; örneğin x ortakat grup seçicisi üzerinden yapılırsa, bu matrisin (j,k) kesişim noktasına x ortakat grup seçicisi yazılır.



(a)

3. Kat (çıkış) grup seçiciler



(b)

Şekil 3.1a. Genel bir bağlaşma şebekesinin blok eşdeğeri

b. Şebekenin matris eşdeğeri

Üç katlı genel bir şebekede $n \times r$ girişten herhangi biri, $n \times r$ çıkıştan herhangi birine bağlantı isteyebilir. Herhangi bir anda, sistemde a giriş grup seçicisinden çıkan bir çağırma, b çıkış grup seçicisine bağlantı isteyebilir. Bu çağırma kısaca (a,b) şeklinde gösterilir. Eğer (a,b) çağırmasının bağlantısı yapılamıyorsa, şebekede teorik olarak iki durumdan biri ortaya çıkmış demektir:

1. (a,b) çağırmasının çıktığı anda, a giriş grup seçicisinden n 'den az ve b çıkış grup seçicisinden n 'den az bağlantı yapılmıştır.

2. (a,b) çağırmasının çıktığı anda, a giriş grup seçicisindeki bütün girişlerin (n -giriş) ve b çıkış grup seçicisinde, bütün çıkışların (n -çıkış) bağlantısı yapılmıştır.

Şebekede birinci durum mevcut ise, buna **dış blokaj** denir ve olasılığı düşüktür. Dış blokaj durumundan önce, giriş grup seçicisinin n -girişi ve çıkış grup seçicisinin n -çıkışı bağlanmıştır. Yeni gelen çağırmanın karşılanabilmesi, bir giriş grup seçicinden $(n+1)$ çağırma çıkması demektir; bu ise imkansızdır.

Şebekede ikinci durum mevcut ise, buna **iç blokaj** denir ve olasılığı yüksektir. Bu durumda giriş grup seçicisinde n -girişten az ve çıkış grup seçicisinde n -çıkıştan az bağlantı yapılmıştır. Yeni gelen çağırmanın bağlanamaması, hem a giriş grup seçicisi için, hem de b çıkış grup seçicisi için, serbest olan bir ortakat grup seçicisinin bulunmamasından kaynaklanmaktadır. Buna karşılık a giriş grup seçicisi ve b çıkış grup seçicisi, farklı ortakat grup seçicilerine erişebilmektedir.

İç blokaj, yeniden düzenleme algoritmaları kullanılarak ortadan kaldırılabilir. Yeniden düzenleme mevcut bağlantıları değiştirerek, giriş grup seçicisi ve çıkış grup seçicisi arasında serbest bir ortakat grup seçicisi bulmayı hedefler.

M.C.Paull [1] bunu bölüm 3.2 'de incelenen dört farklı yöntemle gerçekleştirmiştir.

Daha önce de belirtildiği gibi iç blokaj olasılığı yüksek, dış blokaj olasılığı ise çok düşüktür. Çalışmanın bundan sonraki bölümlerinde **blokaj** deyimini ile iç **blokaj** kastedilecektir.

3.1.1 YENİDEN DÜZENLEME İÇİN MAKSİMUM BAĞLANTI DEĞİŞİKLİĞİ

$n \times n$ boyutunda n adet grup seçicisinden oluşan simetrik bir şebekede, bloke olmuş a giriş grup seçicisi ile b çıkış grup seçicisi arasındaki blokajı kaldırmak için, en fazla $(n-1)$ ortakat grup seçicisinin bağlantısında değişiklik gerekebilir. Bir blokajlı çağırmanın blokajını kaldırmak için gerekli ve yeterli değişikliklerin maksimum sayısı ile ilgili teorem aşağıda verilmiştir.

TEOREM 1: Bir şebekede blokajlı bir çağırmanın blokajını kaldırmak için $(n-1)$ 'den fazla değişiklik gerekmez. $n > 1$ olmak üzere blokajlı şebekenin, blokajını kaldırmak için $(n-1)$ değişiklik yeterlidir.

YETERLİK: Şebeke, giriş grup seçici sayısı n , çıkış grup seçici sayısı n ve her bir grup seçicisi de $n \times n$ boyutunda, üç kattan oluşan simetrik bir yapı olsun. Böyle bir şebekenin eşdeğeri Şekil 3.2. de görüldüğü gibi $n \times n$ boyutunda bir matrisle gösterilebilir. Birinci giriş grup seçicisinden çıkan çağırmanın, birinci çıkış grup seçicisine bağlantı istediği kabul edilsin. Bu çağırma (r_1, c_1) şeklinde gösterilsin ve çağırmanın blokajlı olduğu varsayalım. Birinci giriş grup seçicisi için serbest olup, birinci çıkış grup seçicisi için serbest olmayan (matriste r_1 satırında görülmeyen, c_1 sütununda görülen) ortakat grup seçicisi, örneğin A olsun. Birinci çıkış grup seçicisi için serbest olup, ancak birinci giriş grup seçicisi için serbest olmayan (matriste c_1

sütununda görülmeyen, r_1 satırında görülen) ortakat grup seçicisi, örneğin B olsun. Bu durumda Şekil 3.2.'deki (r_2, c_1) 'de görülen A ortakat grup seçicisi, r_1 giriş grup seçicisi için serbesttir ve (r_1, c_2) 'de görülen B ortakat grup seçicisi, c_1 çıkış grup seçicisi için serbesttir.

Yapılan varsayımlar özetlenirse,

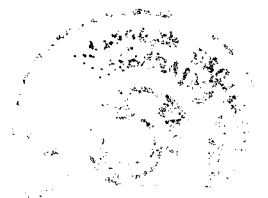
- r_1 , blokajın görüldüğü giriş grup seçicisidir (1. satırda).
- c_1 , blokajın görüldüğü çıkış grup seçicisidir (1. sütunda).
- r_2 giriş grup seçicisi ile c_1 çıkış grup seçicisinin A ortakat grup seçicisi üzerinden bağlantısı vardır.
- c_2 çıkış grup seçicisi ile r_1 giriş grup seçicisinin B ortakat grup seçicisi üzerinden bağlantısı vardır.

$r \backslash c$	1	2	3	...	$n-2$	$n-1$	n
1	x_1, x_2 ... x_{n-2}	B					
2	A	x_1, x_2 ... x_{n-2}	B				
3		A	x_1, x_2 ... x_{n-2}				
...							
$n-2$					x_1, x_2 ... x_{n-2}	B	
$n-1$					A	x_1, x_2 ... x_{n-2}	B
n						A	

Şekil 3.2. $n \times n$ boyutunda n tane seçiciden oluşan simetrik matrisin bağlantı durumu.

Şebekenin diğer durumları da tanımlanırsa,

- r_3 giriş grup seçicisi ile c_2 çıkış grup seçicisi, A ortakat grup seçicisi,
- c_3 çıkış grup seçicisi ile r_2 giriş grup seçicisi, B ortakat grup seçicisi,



- r_4 giriş grup seçicisi ile c_3 çıkış grup seçicisi, A ortakat grup seçicisi,
 - c_4 çıkış grup seçicisi ile r_3 giriş grup seçicisi, B ortakat grup seçicisi
- üzerinden bağlanmışlardır.

Genel olarak, $j > 1$ olmak üzere;

r_j , c_{j-1} çıkış grup seçicisi ile A ortakat grup seçicisi üzerinden bağlantı sağlayan giriş grup seçicisi olarak tanımlanır. Eğer c_{j-1} çıkış grup seçicisi tanımlı ve A ortakat grup seçicisi ile bağlantısı varsa, r_j tanımlanmış, aksi halde tanımlanmamıştır.

c_j , r_{j-1} giriş grup seçicisi ile B ortakat grup seçicisi üzerinden bağlantı sağlayan çıkış grup seçicisi olarak tanımlanır. Eğer r_{j-1} giriş grup seçicisi tanımlı ve B ortakat grup seçicisi ile bağlantısı varsa, c_j tanımlanmış, aksi halde tanımlanmamıştır.

Yukardaki tanımlamaların önemli bir özelliği, r_j ve r_k giriş grup seçicileri tanımlı ve $j \neq k$ olmak üzere $r_j \neq r_k$ olmasıdır. Benzer şekilde c_j ve c_k çıkış grup seçicileri tanımlı ve $j \neq k$ olmak üzere $c_j \neq c_k$ olmaktadır. Buna aşağıdaki şekilde karar verilir.

Giriş ve çıkış grup seçicileri,

$$r_1, c_1, r_2, c_2, r_3, c_3, \dots, r_i, c_i, \dots, r_n, c_n \quad (3.1)$$

şeklinde bir dizi olsun. Dizide ilk giriş grup seçicisi veya çıkış grup seçicisi değişkeninin, önceki giriş grup seçicisi veya çıkış grup seçicisi değişkenine eşit olması halinde;

1. Dizinin ilk değişkeni r_j , hem tanımlı hem de dizinin tanımlanmış önceki, örneğin r_k ($k \neq j$), değişkeni ile aynı olduğu kabul edilsin. Bu tanımlamada $k=1$ olamaz. Çünkü, r_1 giriş grup seçicisi, genel tanımdan dolayı

A ortakat grup seçicisi üzerinden bağlanmadığından, tanımlanmamış ve A ortakat grup seçicisi üzerinden bağlanabilmesi için de $j > 1$ olarak tanımlanmalıdır. Bundan dolayı $r_1 \neq r_j$ ($j > 1$) olmalıdır. Şimdi $k > 1$, $j > 1$ ve $j \neq k$ olsun. Bu durumda (r_k, c_{k-1}) ve (r_j, c_{j-1}) seçicileri A ortakat grup seçicisi üzerinden genel tanım gereği, bağlantı yapar. Dolayısıyla $c_{k-1} = c_{j-1}$ olmadıkça, $j-1 \neq k-1$ olmalıdır ve $r_k = r_j$ ise, bu giriş grup seçicisi iki farklı A ortakat grup seçicisi üzerinden bağlantı yapacaktır. Halbuki bir giriş grup seçicisi, aynı ortakat grup seçicisi üzerinden birden fazla bağlantı yapamayacağından, $c_{k-1} = c_{j-1}$ olur. Ancak bu özelliğe sahip ise, r_j giriş grup seçicisi, dizinin ilk değişkeni olduğu varsayımına ters düşer. Bu da c_j çıkış grup seçicisinin, birinci çıkış grup seçicisi (dizinin ilk değişkeni) olması olasılığını yok eder.

2. Dizinin ilk değişkeni c_j 'nin, hem tanımlı hem de dizinin tanımlanmış önceki, örneğin c_k ($k \neq j$), değişkeni ile aynı olduğu kabul edilsin. Burada $k \neq 1$ 'dir. Çünkü tanım gereği c_1 çıkış grup seçicisi, B ortakat grup seçicisi üzerinden bağlanmamıştır. c_j 'nin tanımı gereği $j > 1$ olur. Eğer $j > 1$, $k > 1$, $k \neq j$ ve $c_j = c_k$ ise bir önceki paragraftakine benzer nedenlerden dolayı $r_{j-1} = r_{k-1}$ 'dir. $c_j = c_k$ olması ise ikinci kabule ters düşer. Dolayısıyla $c_j \neq c_k$ ve $j \neq k$ ise, c_j ve c_k farklı birer çıkış grup seçicileridir.

Tanımlı olan giriş ve çıkış grup seçicilerinin farklı olduğu gösterildikten sonra, ispata aşağıdaki gibi devam edilebilir. Kolaylık için (3.1) dizisi aşağıdaki gibi tekrar yazılmıştır.

$$r_1, c_1, r_2, c_2, r_3, c_3, \dots, r_j, c_j, \dots, r_n, c_n \quad (3.1)$$

A ortakat grup seçicisi üzerinden bağlantı yapan (tanımlı olan) giriş ve çıkış grup seçicileri araştırılır. (3.1) ifadesindeki dizide ilk değişken tanımlıdır. Ancak bunu takip eden değişken, örneğin c_r , tanımlanmamıştır. Çünkü dizinin tanımlanmamış ilk değişkeni r_{r+1} 'dir. r_j 'nin tanımından dolayı, c_r çıkış grup seçicisi, A ortakat grup seçicisi üzerinden bağlantı yapmamıştır. c_j ve r_j



tanımlamalarına göre, aşağıdaki giriş ve çıkış grup seçicileri A ortakat grup seçicisi üzerinden bağlantı yapmışlardır.

$$(r_2, c_1), (r_3, c_2), \dots, (r_f, c_{f-1}) \quad (3.2)$$

Benzer şekilde B ortakat grup seçicisi üzerinden bağlantı yapacak seçici grupları da,

$$(r_1, c_2), (r_2, c_3), \dots, (r_{f-1}, c_f) \quad (3.3)$$

olacaktır.

(r_1, c_1) giriş ve çıkış grup seçicisinde blokajı kaldırmak için aşağıdaki değişiklikler yapılır.

3.1.2 DEĞİŞİKLİK ALGORİTMASI:

$c_j, j=3, 5, \dots, f$; f 'nin tek (veya $j=2, 4, \dots, f$; f 'nin çift) olduğu çıkış grup seçicilerinin hali hazırda B ortakat grup seçicisi üzerindeki bağlantıları, A ortakat grup seçicisi üzerine aktarılır. Yapılan değişiklik, $r_j, j=2, 4, \dots, (f-1)$; f 'nin tek (veya $j=1, 3, \dots, (f-1)$; f 'nin çift) olduğu giriş grup seçicilerinin hali hazırda B ortakat grup seçicisi üzerindeki bağlantılarının, A ortakat grup seçicisi üzerine aktarılmasına eşdeğerdir.

$r_j, j=2, 4, \dots, (f-1)$; f 'nin tek (veya $j=3, 5, \dots, f$; f 'nin çift) olduğu giriş grup seçicilerinin hali hazırda A ortakat grup seçicisi üzerindeki bağlantıları, B ortakat grup seçicisi üzerine aktarılır. Yapılan değişiklik, $c_j, j=1, 3, \dots, (f-2)$; f 'nin tek (veya $j=2, 4, \dots, (f-2)$; f 'nin çift) olduğu çıkış grup seçicilerinin A ortakat grup seçicisi üzerindeki bağlantılarının, B ortakat grup seçicisi

üzerine aktarılmasına eşdeğerdir. Burada, $3,5,7,\dots f$ çıkış ve $2,4,6,\dots(f-1)$ giriş grup seçicilerinde toplam $(f-1)$ değişiklik yapılmıştır.

Eğer f tek ise, değişiklikten sonra (r_1, c_2) giriş ve çıkış grup seçicisinin bağlantısı, B ortakat grup seçicisi üzerinden devam etmektedir; değişiklik yapılmamıştır. Fakat (r_2, c_1) giriş ve çıkış grup seçicilerinin A ortakat grup seçicisi üzerindeki bağlantısı, B ortakat grup seçicisi ile değiştirilmiştir. Böylece (r_1, c_1) giriş ve çıkış grup seçicisinin blokajı kaldırılmıştır ve A ortakat grup seçicisi üzerinden bağlantı yapılmıştır. Benzer yaklaşım, f 'nin çift olduğu durumlarda da geçerlidir.

Bu değişikliğin herhangi bir karışıklığa neden olmadığı, Şekil 3.4.'de verilen şebeke örneği ile açıklanmıştır. Bu örnekte f 'nin tek olduğu kabul edilmiştir (benzer yaklaşım, f 'nin çift olduğu durum için de geçerlidir). Değişiklik A ve B ortakat grup seçicilerinin bağlantılarında yapıldığından, karışıklık da bu ortakatlarda olmalıdır. Bu nedenle olabilecek karışıklık $r_1, r_2, \dots, r_f$ giriş grup seçicilerinde, $c_1, c_2, \dots, c_f$ çıkış grup seçicilerinde olabilir. Çünkü bu giriş ve çıkış grup seçicilerinde değişiklik yapılmış ve aynı zamanda en çok bir giriş ve çıkış grup seçicisi A ortakat grup seçicisi üzerinden bağlanmıştır. Benzer şekilde en çok bir giriş ve çıkış grup seçicisi B ortakat grup seçicisi üzerinden bağlanmıştır. Ayrıca değişiklikten önce, $c_j, j=1,2, 3, \dots(f-1)$ çıkış grup seçicilerinin ve $r_j, j=2,3,\dots f$; giriş grup seçicilerinin, A ortakat grup seçicisi üzerinden bir tane bağlantısı vardır. Değişiklik sonucunda $c_j, j=3,5, \dots f$ çıkış grup seçicileri ve $r_j, j=2, 4, \dots(f-1)$ giriş grup seçicileri A ortakat grup seçicisi üzerinden bağlanmıştır. Dolayısıyla, bu giriş veya çıkış grup seçicilerinde, A ortakat grup seçicisi üzerinden muhtemelen birden fazla bağlantı yapılmış olabilir.

	1	2	3	4	5	6	7
1		B					
2	A ⁺		B [*]				
3		A		B			
4			A ⁺		B [*]		
5				A		B	
6					A ⁺		B [*]
7						A	

(a)

	1	2	3	4	5	6	7
1		B					
2	A ⁺		(A)				
3		A		B			
4			A ⁺	(A)			
5				A		B	
6					A ⁺		(A)
7						A	

(b)

	1	2	3	4	5	6	7
1		B					
2	(B)		(A)				
3		A		B			
4			(B)		(A)		
5				A		B	
6					(B)		(A)
7						A	

(c)

	1	2	3	4	5	6	7
1	(A)	B					
2	(B)		(A)				
3		A		B			
4			(B)		(A)		
5				A		B	
6					(B)		(A)
7						A	

(d)

Şekil 3.4. (1,1) giriş ve çıkış grup seçicisindeki blokajın kaldırılması.

- Şebekenin blokajlı durumu.
- B^{*} bağlantısının, A ortakat grup seçicisine aktarılması.
- A⁺ bağlantısının, B ortakat grup seçicisine aktarılması.
- (1,1) 'in blokajı kaldırılarak A ortakat grup seçicisi üzerinden bağlanması.

Fakat bu giriş ve çıkış grup seçicilerinin herbiri, A ortakat grup seçicisi ile sadece bir tek bağlantı yapmışlardır. Çünkü herbirinin bağlantısı A ortakat



grup seçicisine değiştirilmekle beraber, bunlardaki değişiklikten önceki A ortakat grup seçicisi üzerindeki bağlatıları da, B ortakat grup seçicisi üzerine aktarılmıştır. A ortakat grup seçicisinin bağlandığı bütün çıkış grup seçicilerinin sorunu c_f haricinde çözülmüştür ve c_f çıkış grup seçicisinin zaten başlangıçta A ortakat grup seçicisi üzerinden bağlantısı olmadığından, bağlanması sorun oluşturmaz. Bu arada, r_j , $j=2, 4, \dots(f-1)$ giriş grup seçicilerindeki değişiklikte önce, A ortakat grup seçicisi üzerindeki bağlantılar B ortakat grup seçicisine aktarılmıştır. Bu değişiklik ile A ortakat grup seçicisine aktarılan giriş grup seçicilerinin bağlantılarının sorunu çözülmüştür (Şekil 3.4b.).

Değişiklik sonucunda, $j=2,4,\dots(f-1)$ olmak üzere r_j giriş grup seçicileri ve $j=1,3,5, \dots(f-2)$ olmak üzere c_j çıkış grup seçicileri, B ortakat grup seçicisi üzerinden bağlanmıştır. Dolayısıyla, sadece bu giriş ve çıkış grup seçicileri birden fazla B ortakat grup seçicisi üzerinden bağlantı yapmış olabilir. Fakat $j=3,5, \dots f$ olmak üzere c_j çıkış grup seçicilerinde, başlangıçtaki B ortakat grup seçicisi üzerindeki bağlantı, A ortakat grup seçicisi üzerine aktarılmıştır. Burada B ortakat grup seçicisi ile bağlantısı olmayan c_1 çıkış grup seçicisi dışındaki sorunlar çözülür. $j=2,4,\dots(f-1)$ olmak üzere r_j , giriş grup seçicilerinde, başlangıçtaki B ortakat grup seçicisi üzerindeki bağlantılar, A ortakat grup seçicisine aktarıldığından, bu değişiklikte, B ortakat grup seçicisine aktarılan giriş grup seçicilerinin sorunu çözülür (Şekil 3.4c). Böylece yapılan değişikliklerin çağırımları bozmadığı görülmüş olur (Şekil 3.4d).

(3.1) ifadesinde bütün değişkenler tanımlanmış ise, c_n tanımlanan son elemandır. c_n çıkış grup seçicisi, A ortakat grup seçicisi üzerinden bağlanamaz. Çünkü, r_1 giriş grup seçicisi dışındaki herhangi bir giriş grup seçicisi, A ortakat grup seçicisi üzerinden bağlanabilir. Dolayısıyla r_1 giriş grup seçicisinin dışındaki bütün giriş grup seçicileri, A ortakat grup seçicisi



üzerinden bağlanabilir. Fakat bu giriş grup seçicilerinden hiçbirisi, A ortakat grup seçicisi üzerinden c_n çıkış grup seçicisine bağlanamaz. Bu r_j 'nin tanımından ileri gelmektedir. Buradan genel bir sonuca gidilerek, r_{f+1} 'in (3.1) dizisinin tanımlanmamış ilk değişkeni olduğu söylenebilir.

Dolayısıyla örnekten de görüldüğü gibi, r_j , $j=2, 4, \dots, (f-1)$ ve c_j , $j= 3, 5, \dots, f$ seçicilerinde değişiklik yapılarak (r_1, c_1) giriş ve çıkış grup seçicisi için, serbest ortakat grup seçicisi bulunmuştur. Tanımlı eleman sayısı f yerine n alınırsa, en fazla $(n-1)$ tane değişiklik yapmanın yeterli olduğu görülür.

GEREKLİK: Şebeke de, A, B, $x_1, x_2, \dots, x_{n-2}$ sembolleri ile gösterilen n tane ortakat grup seçicisi vardır. Aşağıdaki şebeke bağlantılarından dolayı $(1,1)$ giriş ve çıkış grup seçicisinin blokajlı olduğu kabul edilmiştir.

Şebekenin bağlantı durumu:

- $i=1, 2, \dots, (n-1)$ olmak üzere, (i,i) giriş ve çıkış grup seçicilerinin herbiri, $x_1, x_2, \dots, x_{n-2}$, ortakat grup seçicisi üzerinden bağlanmıştır.

- $i=1, 2, \dots, (n-1)$ olmak üzere, $(i,i+1)$ giriş ve çıkış grup seçicilerinin her biri B ortakat grup seçicisi üzerinden bağlanmıştır.

- $i=1, 2, \dots, (n-1)$ olmak üzere, $(i+1,i)$ giriş ve çıkış grup seçicilerinin her biri A ortakat grup seçicisi üzerinden bağlanmıştır.

-Şebekede bunlardan başka ortakat grup seçicisi üzerinden bağlantı yapılmamıştır. $(1,1)$ giriş ve çıkış grup seçicisindeki blokajın kaldırılması için $(1,2)$ ve $(2,1)$ giriş ve çıkış grup seçicilerinin aynı ortakat grup seçicisi üzerinden bağlantı yapmak gerekir. Çünkü:

1-Herbir değişiklikten sonra, $(1,1)$ giriş ve çıkış grup seçicisinin, hali hazırda $(n-2)$ farklı ortakat grup seçicisi üzerinden bağlantısı devam etmektedir.

2-Hali hazırda (1,2) giriş ve çıkış grup seçicisinin bağlandığı ortakat grup seçicisi, (1,1) giriş ve çıkış grup seçicisinin bağlı olduğu ortakat grup seçicilerinden farklı olmalıdır.

3-Hali hazırda (2,1) giriş ve çıkış grup seçicisinin bağlandığı ortakat grup seçicisi, (1,1) giriş ve çıkış grup seçicisinin bağlı olduğu ortakat grup seçicilerinden farklı olmalıdır.

4- (1,2) ve (2,1) giriş ve çıkış grup seçicilerinin ortakat grup seçicileri birbirinden farklı ise, birinci giriş ve çıkış grup seçicilerinin bağlı olduğu ortakat grup seçici sayısı toplam n olacaktır. Dolayısıyla, serbest bir ortakat grup seçicisi olmadığından, (1,1) giriş ve çıkış grup seçicisinden blokajı kaldırmak için, düzenleme yapılması gerekecektir.

Şekil 3.5. de görüldüğü gibi, (1,1) giriş ve çıkış grup seçicisinden blokajı kaldırmak için, $i=k-1$ olmak üzere $(i+1, i)$ ve $(i, i+1)$ giriş ve çıkış grup seçicilerinin, w ortakat grup seçicisi üzerinden bağlandığı kabul edilsin. Şimdi $i=k$ olmak üzere, k giriş grup seçicisinde yapılan değişiklikten sonra hali hazırda (i,i) giriş çıkış grup seçicisinin, $x_1, x_2, \dots, x_{n-2}$ den oluşan, $(n-2)$ farklı ortakat grup seçicisi ile bağlantısı devam etmektedir. Dolayısıyla $i=k-1$ olmak üzere, $(i+1,i)$ giriş ve çıkış grup seçicisinin bağlı olduğu w ortakat grup seçicisi, k giriş grup seçicisinin bağlı olduğu $x_1, x_2, \dots, x_{n-2}$ ortakat grup seçicilerinden farklıdır. $i=k$ olmak üzere $(i, i+1)$ giriş ve çıkış grup seçicisinin bağlı olduğu ortakat grup seçicisi, k giriş grup seçicisinin $w, x_1, x_2, \dots, x_{n-2}$ ortakat grup seçicilerine bağlı olduğundan dolayı, $i=k$ olmak üzere $(i,i+1)$ giriş ve çıkış grup seçicisinin bağlı olduğu ortakat grup seçicisi, bunlardan farklı olmalıdır. $(n-1)$ ortakat grup seçicisinin, $w, x_1, x_2, \dots, x_{n-2}$ 'den farklı olabilecek sadece bir ortakat grup seçicisi vardır ki o da z 'dir. Bundan dolayı, $i=k$ olmak üzere $(i,i+1)$ giriş ve çıkış grup seçicisi, z ortakat grup seçicisi üzerinden bağlanmalıdır. Benzer şekilde, $i=k$ olmak üzere, k çıkış grup seçicisinde yapılan değişiklikten sonra ki durumda hala (i,i) giriş çıkış grup seçicisinin $x_1, x_2, \dots, x_{n-2}$ den oluşan $(n-2)$ farklı ortakat grup seçicisi ile

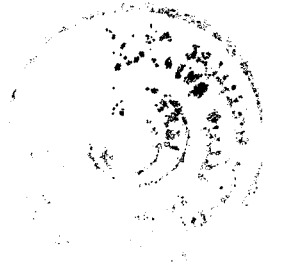


bağlantısı devam etmektedir. Ayrıca $i=k-1$ olmak üzere, $(i,i+1)$ giriş ve çıkış grup seçicisinin bağlı olduğu w ortakat grup seçicisi, k çıkış grup seçicisinin ortakat grup seçicisidir. $i=k$ olmak üzere, $(i+1,i)$ giriş ve çıkış grup seçicisinin bağlı olduğu ortakat grup seçicisi, k çıkış grup seçicisinin $w, x_1, x_2, \dots, x_{n-2}$ ortakat grup seçicilerine bağlı olduğundan dolayı, bunlardan farklı olan bir ortakat grup seçicisi olmalıdır; ki o da z 'dir.

	1	2	3		k	k	k-1	n
1	$x_1, x_2, \dots, x_{n-2}$	B						
2	A	$x_1, x_2, \dots, x_{n-2}$	B					
3		A	$x_1, x_2, \dots, x_{n-2}$		B			
k-1					$x_1, x_2, \dots, x_{n-2}$	w		
k					w	$x_1, x_2, \dots, x_{n-2}$	z	
k+1						z	$x_1, x_2, \dots, x_{n-2}$	B
n							A	

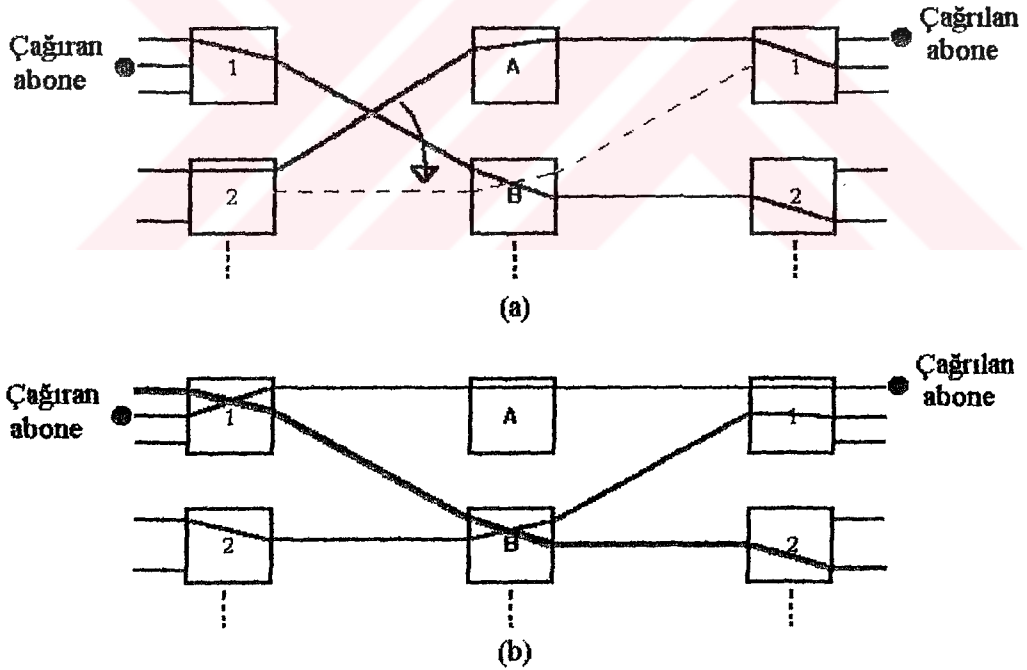
Şekil 3.5. Blokajlı şebekenin, blokajının kaldırılması.

(i,i) giriş ve çıkış grup seçicilerinde, $(n-2)$ farklı ortakat grup seçicisi bağlı olduğundan dolayı $(i+1,i)$ ve $(i,i+1)$ giriş ve çıkış seçici çiftleri aynı ortakat grup seçicisi üzerinden bağlanır ise $(1,1)$ 'in blokajı kalkar. Böylece, başlangıçta $i=1,2,\dots,(n-1)$ olmak üzere $(i+1,i)$ ile $(i,i+1)$ giriş ve çıkış grup seçicileri farklı ortakat grup seçicilerine bağlandığından, $(1,1)$ 'in blokajını kaldırmak için en az $(n-1)$ değişiklik gereklidir.



3.2 BLOKAJ KALDIRMA YÖNTEMLERİ

YÖNTEM 1: Teorem 1'in ispatında blokajlı bir çağırmanın, blokajını kaldırmak için gerekli değişiklik anlatılmıştır. Bu yöntem iki ortakat grup seçicisi (örneğin A ve B) üzerinden bağlantıyı gerçekleştiren bir yöntemdir. Yöntemin esası (r_1, c_1) giriş ve çıkış grup seçicisi blokajlı ise, r_1 giriş grup seçicisinin bağlı olduğu, c_1 çıkış grup seçicisinin bağlı olmadığı ve c_1 çıkış grup seçicisinin bağlı olduğu, r_1 giriş grup seçicisinin bağlı olmadığı ortakat grup seçicilerini bulup; teoremden anlatılan değişiklik algoritmasının uygulanmasıdır. Şekil 3.6. 'da blokajı kaldırmak için A ortakat grup seçicisi üzerindeki bağlantının, B ortakat grup seçicisi üzerine aktarılması (kesik çizgi ile) gösterilmiştir.



Şekil 3.6a. (1,1) blokajlı şebekeye Yöntem 1'in uygulanması.

- b. Blokaj kaldırılarak, A üzerinden bağlantının gerçekleştirilmesi.

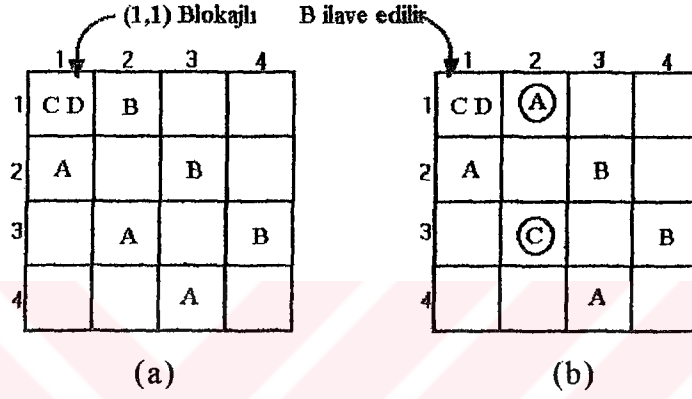
YÖNTEM 2: Bu yöntem, Yöntem 1 'in daha geliştirilmiş durumudur. r_1 giriş grup seçicisinin bağlı olduğu, c_1 çıkış grup seçicisinin bağlı olmadığı ortakat grup seçicisi A ve c_1 çıkış grup seçicisinin bağlı olduğu, r_1 giriş grup seçicisinin bağlı olmadığı ortakat grup seçicisi B olsun. Bunları bütün ortakat grup seçicileri için deneyerek, en az değişiklik gerektirecek ortakat grup seçici çiftleri saptanır. Uygun çifte göre değişiklik algoritması uygulanır.

YÖNTEM 3: Yöntem 1 ve Yöntem 2'de iki ortakat grup seçicisine göre değişiklik algoritması uygulanmaktadır. Ortakat grup seçicisinin sayısında sınırlama getirmeyen değişiklik algoritmaları da geliştirilebilir. Yöntem 3 buna dayanmaktadır.

(r_1, c_1) giriş ve çıkış grup seçicisinin blokajlı olduğu kabul edilsin. A ortakat grup seçicisinin r_1 giriş grup seçicisine bağlandığını, c_1 çıkış grup seçicisine bağlanmadığını; B ortakat grup seçicisinin de c_1 çıkış grup seçicisine bağlandığını, r_1 giriş grup seçicisine bağlanmadığı varsayalım. Teorem 1'in ispatında olduğu gibi, c_r (3.1) ifadesinde ilk çıkış grup seçicisi değişkeni olsun. A ve B ortakat grup seçicilerinde $(f-1)$ değişiklik yapılarak, (r_1, c_1) giriş ve çıkış grup seçicisinin blokajı kaldırılır. Eğer c_j çıkış grup seçicisi ve r_j giriş grup seçicisinin $(j < f)$ tanımlayan A ve B ortakat grup seçicilerinin hali hazırdaki bağlantılarını çözmeden, başka bir ortakat grup seçicisi (örneğin C) ile değiştirilebilirse, $(f-1)$ 'den daha az değişiklik yapılarak (r_1, c_1) giriş ve çıkış grup seçicilerinin blokajı kaldırılabilir. Bu durum Şekil 3.7. 'de gösterilmektedir. Bu yöntem aşağıdaki gibi özetlenebilir:

- r_1 giriş grup seçicisinin bağlı olmadığı ve c_1 çıkış grup seçicisinin bağlı olduğu ortakat grup seçicisi A ile c_1 çıkış grup seçicisinin bağlı olmadığı ve r_1 giriş grup seçicisinin bağlı olduğu ortakat grup seçicisi B 'nin bulunması.
- (3.1) ifadesindeki dizinin son tanımlanan giriş ve çıkış grup seçicisinin bulunması.

- (3.1) ifadesinin belirttiği A ve B ortakat grup seçicilerinin başka bir ortakat grup seçicisi (örneğin C) ile, değiştirilebilirliğinin araştırılması. Mamafih, eğer r_j ve c_j 'yi tanımlamaya yarayan A ve B 'lerin bir kısmı çağırmaı bozmaksızın bir başka ortakat grup seçicisi ile, örneğin C, değiştirilebilirse, (f-1) 'den daha az değişiklikle (r_1, c_1)



Şekil 3.7a. (1,1) blokajlı şebekenin bağlantısı.

b. Yöntem 3 ile (1,1) blokajının kaldırılması.

Daire (○) içindekiler değiştirilen ortakat grup seçicileridir.

blokajsız hale getirilebilir. Böyle bir ortakat grup seçicisi yoksa, Yöntem 1 veya Yöntem 2 uygulanabilir, varsa, o zaman bu değişikliği gerçekleştiren algoritmanın gerektirdiği şekilde c_j ($j < k$) çıkış grup seçicilerinde ve r_j ($j < k$) giriş grup seçicilerinde değişiklik yapılır.

YÖNTEM 4: A, r_1 giriş grup seçicisinin bağlı olduğu ve c_1 çıkış grup seçicisinin bağlı olmadığı, ortakat grup seçicisi ile B, c_1 çıkış grup seçicisinin bağlı olduğu ve r_1 giriş grup seçicisinin bağlı olmadığı, ortakat grup seçicisi olsun. Burada bütün (A,B) çiftleri için Yöntem 3 uygulanır. Gerçekte en az değişiklik gerektiren çiftler bulunur ve bu çiftler ile değişiklik gerçekleştirilerek blokaj kaldırılır. Şekil 3.8. ve Şekil 3.9.'da bu yöntemlerin



uygulaması gösterilmiştir.

	1	2	3	4	5	6
1	E,F	B,D				
2	A,C		B			
3		A	CDEF	B		
4			A		B	
5				A	CDEF	B
6					A	

(a)

D ilave edilir.

	1	2	3	4	5	6
1	E,F	B,D				
2	(B),C		(A)			
3		A	CDEF	B		
4			(B)		(A)	
5				A	CDEF	B
6					(E)	

(b)

A ilave edilir.

	1	2	3	4	5	6
1	E,F	B,(C)				
2	A,C		B			
3		A	CDEF	B		
4			A		B	
5				A	CDEF	B
6					A	

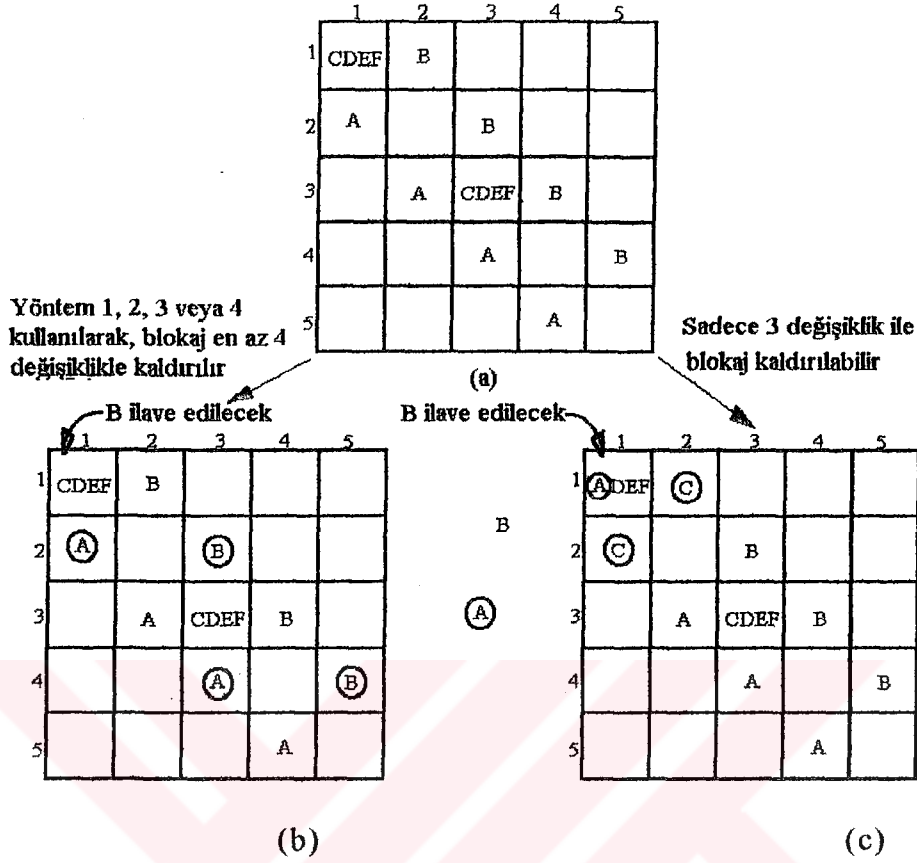
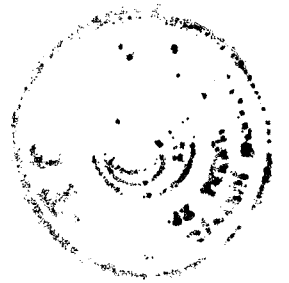
(c)

Şekil 3.8a. Şebekenin blokajlı bağlantısı.

- b. 1. veya 3. yöntemiyle beş değişiklikle blokaj kaldırılması.
- c. 2. veya 4. yöntemiyle dört değişiklikle blokaj kaldırılması.

3.3 DİĞER ŞEBEKELERE GENELLEŞTİRME

Önceki bölümlerde her bir katın grup seçicilerinin eşit sayıda yani, sistemin simetrik yapıda olduğu kabul edilerek çözümler geliştirilmiştir. Şekil 3.10.'da görüldüğü gibi şebeke, ortakat grup seçicilerinin sayısı m , giriş grup seçicilerinki ise n alınarak, $m \times n$ şeklinde genelleştirilebilir.



Şekil 3.9a. Blokajlı şebekenin bağlantısı.

- b. Dört yöntemden birisinin uygulanması ile blokajın kaldırılması.
- c. Dört yöntemin dışında bir yöntemle blokajın kaldırılması. Değiştirilen ortakat grup seçicileri, daire (○) içinde gösterilmiştir.

Simetrik şebeke için verilen iç blokaj düzenlemeleri, genel şebekeler için de verilebilir. Eğer a giriş grup seçicisi ve b çıkış grup seçicisi n tane ortakat grup seçicisine bağlanmış ise, yeni çıkan (a,b) giriş ve çıkış grup seçicisindeki çağırma dış blokaja uğrar. Bu durum konumuzun dışındadır. a giriş grup seçicisi ve b çıkış grup seçicisin 'den az ortakat grup seçicisine bağlanmış ise, iç blokaj oluşmuştur; şebekede yeniden düzenleme yapılarak, blokajın kaldırılması söz konusu olabilir.

$m=n$ olan kare şebekede iç blokajın giderilmesi için $(n-1)'$ den daha az değişiklik gerektiği gösterilmiştir. Clos $m=2n-1$ alınmasıyla sistemin blokajsız olacağını ispat etmiştir [1]. Böyle bir sistemde, her durumda giriş ve çıkış grup seçicileri için serbest bir ortakat grup seçicisi bulunur. Eğer $m=2n-2$ ise blokajın giderilmesi için en fazla 1; genel olarak $m=2n-j$ ise en fazla $j-1$ bağlantının değiştirilmesi gerekir.

$m=2n-2$ lik bir şebeke için yapılması gereken değişik sayısının sınırını bir teoremle ispatlanabilir.

TEOREM 2: $m=n+k$ ve (a,b) giriş ve çıkış grup seçicisindeki çağırma blokajlı ise, b çıkış grup seçicisinin bağlanmadığı, a giriş grup seçicisinin bağlandığı en az $k+1$ ortakat grup seçicisi; a giriş grup seçicisinin bağlanmadığı, b çıkış grup seçicisinin bağlandığı en az $k+1$ ortakat grup seçicisi vardır.

İSPAT: (a,b) giriş ve çıkış grup seçicilerinin blokajlı olduğu varsayımından dolayı, a ve b toplam $n+k$ ortakat grup seçicisine bağlanmıştır. Aynı zamanda blokaj şartından a , n 'den az sayıda ve b , n 'den az ortakat grup seçicisine bağlanmıştır.

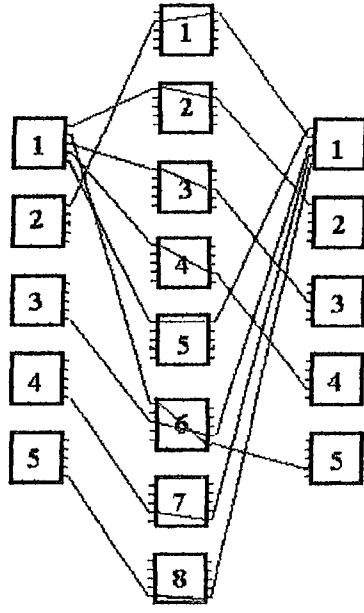
R: b çıkış grup seçicisinin bağlanmadığı, a giriş grup seçicisinin bağlandığı ortakat grup seçicisini,

C: a giriş grup seçicisinin bağlanmadığı, b çıkış grup seçicisinin bağlandığı ortakat grup seçicisini,

X: (a,b) giriş ve çıkış grup seçicisinin bağlı olduğu ortakat grup seçicilerini,

B: a giriş grup seçicisinin ve b çıkış grup seçicisinin ortak bağlanmadıkları aynı ortakat grup seçicilerini

gösterebilir (Şekil 3.10.).



(a)

	1	2	3	4	5
1	5	2	3	4	6
2	1				
3	6				
4	7				
5	8				

(b)

Şekil 3.10. $n+k$ ($5+3$) ortakat grup seçiciden oluşan şebeke (Birinci giriş ve çıkış grup seçicisi için, R: 1,7,8; C: 2,3,4; B: 6; X: 5 ortakat grup seçicilerini içerir)

- Şebekenin mevcut bağlantısı
- (a)'da verilen şebekenin matris eşdeğeri

Bu durumda,

$$B+C+R+X=n+k \quad (3.4)$$

$$R+B+X < n \quad (3.5)$$

$$C+B+X < n \quad (3.6)$$

(3.5) ve (3.6) ifadelerini (3.4) 'de yerine yazılırsa;

$$C > k$$

$$R > k$$

bulunur.

Bu teoremin isbatı Clos şebekesinin blokajsız olduğu gözönüne alınarak yapılabilir. Clos şebekesinde $k=n-1$ dir. (a,b) giriş ve çıkış grup seçicisi blokajlı ise, teorem 2 gereği a giriş grup seçicisi $k+1$ ortakat grup seçicisine bağlandığından, $C=k+1=n-1+1=n$ 'dir. Bu durumda, dış blokaj meydana gelir ve bu da açıklanan konunun dışındadır.

$k=n-2$ olmak üzere, teorem 2 gereği (a,b) giriş ve çıkış grup seçicisindeki çağırma blokajlı ise $C=k+1=n-2+1=n-1$ ve $R=k+1=n-2+1=n-1$ 'dir. Tüm bağlantılar bittikten sonra (n-1) 'inci bağlantı blokajlı ise, a giriş grup seçicisindeki C ile gösterilen ortakat grup seçicilerinden (C'nin tümüne örneğin $A_1, A_2, \dots, A_{n-1}=A$ diyelim) veya b çıkış grup seçicisindeki R ile gösterilen ortakat grup seçicilerinden (R'nin tümüne örneğin $B_1, B_2, \dots, B_{n-1}=B$ diyelim) birinin bağlantısının değiştirilmesiyle blokaj çözülür. Yani, a 'nın bağlı olduğu, çıkış grup seçicilerinin her biri n-1 tane B üzerinden; b 'nin bağlı olduğu giriş grup seçicilerinin her biri n-1 tane A üzerinden bağlantı yapmıştır. Burada herhangi bir giriş veya çıkış grup seçicisi, n ortakat grup seçicisine bağlanmış ise, (a,b) için ortakat grup seçicisi bulunamaz. a ve b farklı ortakat grup seçicileri üzerinden, farklı giriş ve farklı çıkış grup seçicilerine bağlanmıştır. k 'ıncı giriş grup seçicisine bakıldığında, bir tane B ve n-1 tane A ortakat grup seçicisi üzerinden bağlantı yapılmıştır. k 'nın bağlı olduğu A ortakat grup seçicisinden biri $p \neq a$ olmak üzere p çıkış grup seçicisine bağlanmıştır. Bununla birlikte p, hali hazırda bir A ve n-1 tane B ortakat grup seçicisi üzerinden bağlanmıştır. Böylece p, k 'nın bağlı olduğu A 'lardan biri ile başka giriş grup seçicilerinin bağlı olduğu A 'lardan biri ve n-1 tane B, olmak üzere toplam n+1 ortakat grup seçicisi ile bağlantı yapmış olur. Bu duruma izin verilemez. Dolayısıyla $m=2n-2$ durumunda blokajı çözmek için sadece bir değişiklik her zaman yeterlidir.



3.4 ÜÇ KATLI ŞEBEKEDEN $(n+1)$ ORTAKAT GRUPO SEÇİCİSİ İLE BLOKAJIN KALDIRILMASI

M. C. PAULL [1] makelesinde $m \geq n$ olmak üzere üç katlı şebekelerde (bunlara bazen Clos tipi şebeke de denilir) mevcut bağlantıları değiştirerek, blokajı kaldırmak için bazı algoritmalar vermiştir. Bu algoritmalar, şebekede mevcut bağlantılardan bazılarının geçici olarak bozulmasına neden olur. Paull, bu makelede bir öneri yapmış; şebeke ve yöntemlerde küçük bir değişikliklerle, mevcut bağlantıları bozmaksızın blokajsız bir yapı elde edilebileceğini göstermiştir.

Temel üç katlı şebeke, herbir kat $n \times n$ boyutunda n tane grup seçicisinden oluşsun. Bu şebekenin, birinci katı $n \times (n+1)$ boyutunda ve n tane grup seçicisinden, ikinci katı $n \times n$ boyutunda ve $(n+1)$ tane grup seçicisinden, üçüncü katı da $(n+1) \times n$ boyutunda ve n tane grup seçicisinden oluşacak şekilde yeniden düzenlensin. Birinci katın (giriş katı) her bir grup seçicisi ve üçüncü katın (çıkış katı) her bir grup seçicisi, ikinci kat (orta kat) grup seçicilerine bir hat ile bağlanmış olsun.

Şebekedeki bağlantılar $n \times n$ boyutlu matris ile sembolize edilebilir. Matriste, her bir giriş grup seçicisi bir satır ve her bir çıkış grup seçicisi de bir sütun ile gösterilir. Matrisin satır ve sütun sayısı, sırasıyla şebekenin giriş-çıkış grup seçicilerin sayısı kadardır. j 'inci giriş grup seçicisi ile k 'inci çıkış grup seçicisi arasındaki bağlantı, matrisin (j,k) kesişim alanına yazılır. Bunlar için serbest olan ortakat grup seçicisi örneğin A ise, bu noktaya yazılacaktır. Matrisin herhangi bir satır veya sütununda $n+1$ 'den daha fazla ortakat grup seçicisini sembolize eden değişken olamaz; bunlar da $n+1$ ortakat grup seçicileri içinden seçilen n tane değişkendir.

Bu şebekenin blokajını kaldırmak için algoritma aşağıdaki temel adımlardan oluşabilir:

- Şebekede blokajlı durumla karşılaşılana kadar birinci ortakat grup seçicisi üzerinden bağlantı yapılır.

- Bu şebeke, $n \times n$ boyutunda, n tane grup seçicisinden oluşan blokajlı şebeke ile aynı durumdadır. Böyle bir şebekede, ikiden fazla ortakat grup seçicisinin bağlantısının değiştirilmesi ile blokajın kaldırılabilceği yukarda gösterilmiştir. Blokajı kaldırmak için değiştirilmesi gereken bağlantılar C ve D ortakat grup seçicilerinde olsun. Değiştirme algoritmasına göre C 'den D 'ye ve D 'den C 'ye belli bağlantıların değiştirilmesi gerekmektedir. Bu işlem ya C 'nin ya da D 'nin bağlantısı ile şebekenin blokajını çözecektir. C ve D 'den hangisinin serbest hale geleceği, düzenlenecek bağlantı şemasına göre değişir.

Bu şekilde yapılan yeniden düzenlemeler esnasında, ortakat grup seçicilerinin bağlantılarında yapılacak değişiklikler, çağırımların kesilmesine neden olur. Söz konusu şebekede, bağlantılar yeniden düzenlenirken çağırımı muhafaza edecek fazladan bir ortakat grup seçicisi (örneğin A) kullanılabilir. Bu şebeke için geliştirilen algoritma Şekil 3.11.'de gösterilmektedir.

Algoritmaya göre önce C ve D ortakat grup seçicilerinde blokajı çözerek yeniden düzenlenmesi gerekli bağlantı kümesi belirlenir. Daha sonra aşağıdaki adımlar gerçekleştirilir.

1. Yeniden düzenleme için belirlenmiş C 'nin bağlantıları, A ortakat grup seçicisinde de yapılır. Bu durum matriste, değişiklik yapılacak C ortakat grup seçicilerinin bulunduğu yerlere aynı zamanda A ortakat grup seçicisinin de yazılmasıyla gösterilir.

2. C ortakat grup seçicisinde yeniden düzenleme için seçilen tüm bağlantılar çözülür. Bu bağlantılar A ortakat grup seçicisi üzerinden devam eder.

3. Yeniden düzenleme için belirlenmiş D ortakat grup seçicisindeki her bir bağlantı için, C ortakat grup seçicisinde de bağlantı yapılır. Bu bağlantı matriste, değişiklik yapılacak D ortakat grup seçicisinin bulunduğu yerlere, aynı zamanda C ortakat grup seçicisinin de yazılmasıyla gösterilir. Bu durum daima mümkündür; çünkü önceden ispatlandığı gibi, blokajlı bağlantı C de bulunduğu ve C ortakat grup seçicisindeki tüm bağlantılar A ortakat grup seçicisi üzerine aktarıldığından dolayı, söz konusu blokaj olmayacaktır.

4. D ortakat grup seçicisinde yeniden düzenleme için seçilen tüm bağlantılar çözülür ve bunlar C ortakat grup seçicisi üzerinden devam eder.

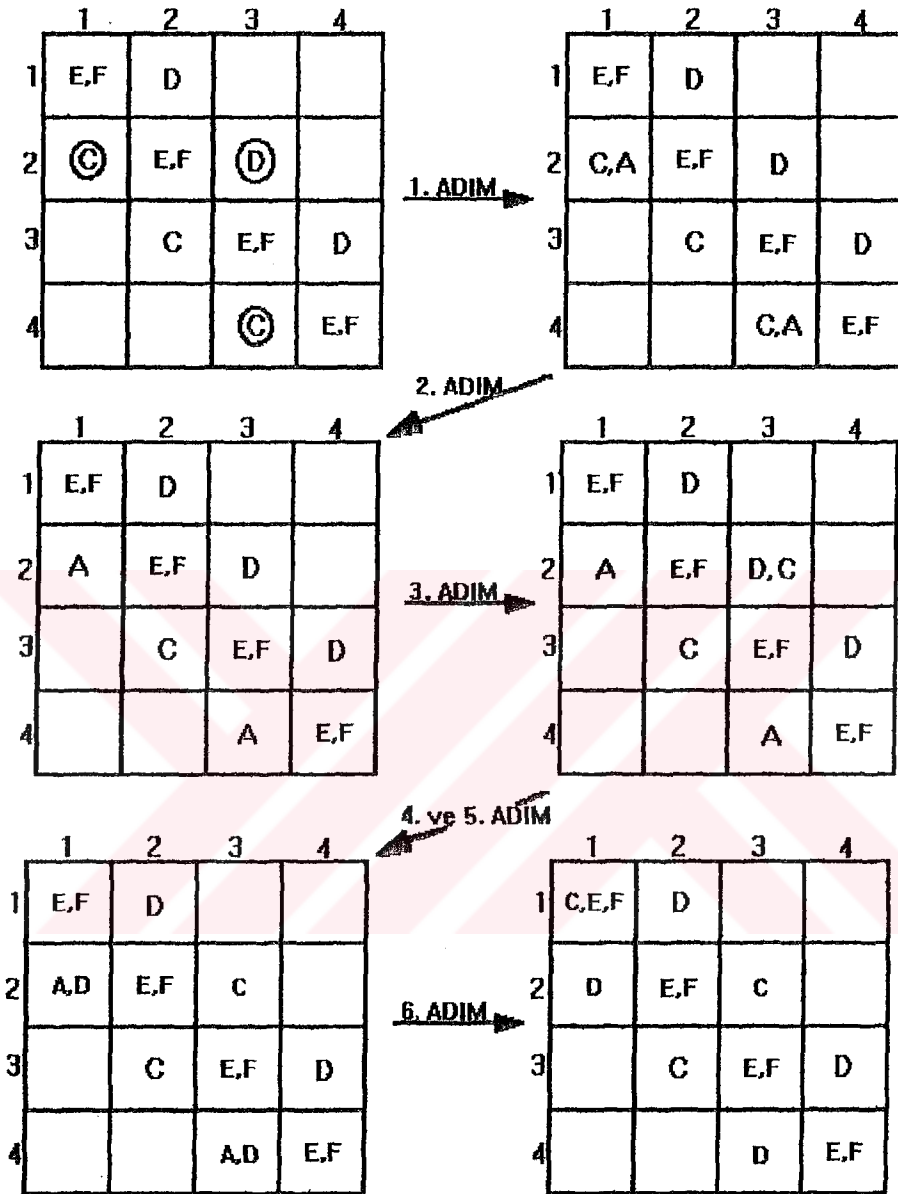
5. A ortakat grup seçicisinin tüm bağlantıları, D ortakat grup seçicisinde de yapılır. Bu bağlantı matriste, değişiklik yapılacak A ortakat grup seçicisinin bulunduğu yerlere aynı zamanda D ortakat grup seçicisinin de yazılmasıyla gösterilir.

6. Son olarak A ortakat grup seçicisindeki tüm bağlantılar çözülür.

Bu değiştirme algoritması ile blokaj kaldırılmış olur. Algoritma gereği A ortakat grup seçicisi her zaman yeni bir blokajlı bağlantının blokajını çözmek için serbest olarak bekletilir.

Blokajlı çağırma, ikinci adımdan sonra blokajsız hale gelir ve istenen bağlantı C ortakat grup seçicisi üzerinden gerçekleştirilir. Blokajlı çağırmanın giriş veya çıkış grup seçicilerinde hem C hem de D ortakat grup seçicileri değiştirildiğinden, A ortakat grup seçicisindeki bağlantının bunlardan (C ve D) hangisine aktarılacağına, ikinci adımda, blokaj çözülecek şekilde, karar

verilir.



Şekil 3.11. Çağırmaı bozmadan yeniden düzenleme yöntemi.

Yeniden düzenleme sırasında, çağırmaı bozulmadan algoritma ile blokaj çözülmek istenirse ve bir tek ortakat grup seçicisi (örneğin A) kullanılacak ise, önceden bu seçiciyle hiçbir bağlantı yapılmamış olmalıdır. Bu alıritmada

blokajın çözülmesi ilk adımlarda sağlandığı halde, bir sonraki blokajın çözülmesi için algoritmanın geri kalan kısmı tamamlanmalıdır.

Burada A fazladan bir ortakat grup seçicisi olarak, yani bir tür bağlantı belleği gibi, şebekenin yüklü durumda ki blokaj karakteristiklerinin iyileştirilmesinde kullanılabilmektedir.

A ortakat grup seçicisine benzer şekilde, ilave ortakat grup seçicileri yardımıyla şebekedeki ölü zamanın daha etkin kullanılması yoluna gidilebilir. Yani A ve B gibi iki tane ilave ortakat grup seçicisi olursa; A ortakat grup seçicisinin yeni bir blokajlı çağırmanın çözülmesi için boşaltılmasını beklemeye gerek kalmayacaktır. Çünkü B ortakat grup seçicisi serbest beklemektedir. Ancak, A ortakat grup seçicisi ilk blokajdan sonra gelecek bir başka blokajlı çağırma durumuna kadar boşaltılmalıdır.

3.5 BAĞLANTI ZİNCİRİ İLE YENİDEN DÜZENLEME

Üç katlı bağlaşma şebekelerinde yeniden düzenleme yöntemlerinden bir diğeri S. OHTA ve H. VEDA [16] 'nın kullandığı yöntemdir. Bu yöntem temel olarak, M.C. PAULL [19] yöntemine benzemekle birlikte, kullanılan bağlantı algoritması farklıdır.

Algoritmada kullanılan bağlantılar, bağlantı zinciri ile gösterilmektedir.

3.5.1 BAĞLANTI ZİNCİRİ TANIMLARI

Yeniden düzenleme yöntemlerinde zincir notasyonu da kullanılabilir. Üç katlı şebekede yeniden düzenleme yöntemlerini açıklamak için bağlantı zinciri notasyonunda kullanılan temel kavramlar şöyle tanımlanabilir:

TANIM 1: I_i giriş grup seçicisi ile Q_j çıkış grup seçicisinin, x ortakat grup seçicisi üzerinden yapmış olduğu bağlantı;

$$I_i \xrightarrow{x} Q_j$$

şeklinde gösterilir. İki ortakat grup seçicisi x_1 ve x_2 üzerinden yapılan bağlantı;

$$I_i \xrightarrow{x_1} Q_j \xrightarrow{x_2} I_k \xrightarrow{x_1} \dots$$

notasyonu ile gösterilip, x_1 ve x_2 ortakat grup seçicisi üzerinden gerçekleştirilen bağlantı zinciri olarak tanımlanır.

Normal olarak, gerek bir giriş grup seçicisi gerekse bir çıkış grup seçicisi bir ortakat grup seçicisine birden fazla bağlantı yapamaz. Sonuçta;

$$I_i \xrightarrow{x_1} Q_j \xrightarrow{x_1} I_k$$

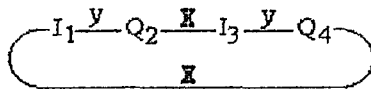
gibi bir bağlantı zinciri oluşturulamaz.

TANIM 2: Bir bağlantı zinciri içindeki giriş ve çıkışı bağlayan zincirlerin sayısı, **zincir uzunluğu** olarak tanımlanır. Örneğin, uzunluğu üç olan bağlantı zinciri;

$$I_1 \xrightarrow{x_1} Q_2 \xrightarrow{x_2} I_3 \xrightarrow{x_1} Q_4$$

şeklinde olur.

TANIM 3: Kapalı bir bağlantı zinciri aşağıdaki gibi tanımlanabilir.



TANIM 4: I_0 ve Q_0 arasında iç blokaj olduğu zaman, I_0 için serbest ortakat grup seçicisi x , Q_0 için serbest ortakat grup seçicisi y olsun. y - x sırası ile bağlanmış ve I_0 ile başlayan zincir, I_0 'nun S_1 zinciri ve benzer şekilde, x - y sırasında bağlanmış ve Q_0 ile başlayan zincir ise Q_0 'nun S_0 zinciri olarak aşağıdaki gibi tanımlanır.

$$S_1: I_0 \xrightarrow{y} Q_1 \xrightarrow{x} I_2 \xrightarrow{y} \dots$$

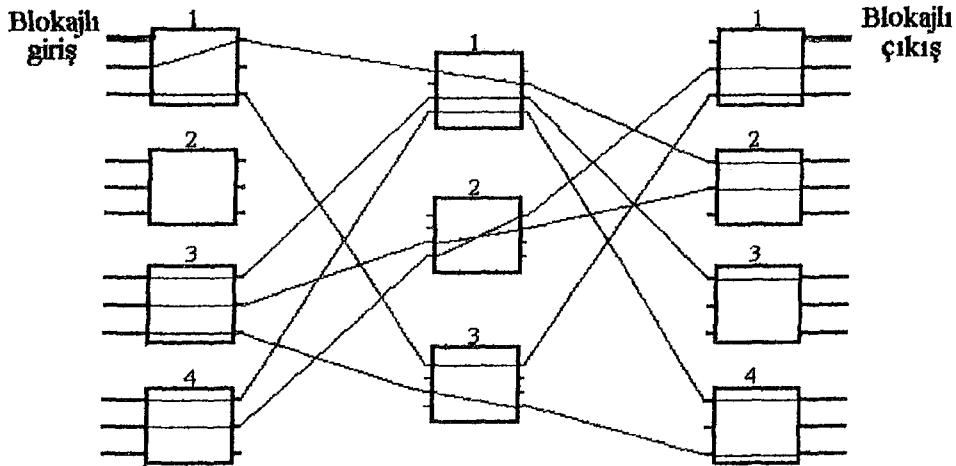
$$S_0: Q_0 \xrightarrow{x} I_1 \xrightarrow{y} Q_2 \xrightarrow{x} \dots$$

Örneğin, Şekil 3.12. deki üç katlı şebekede birinci giriş grup seçicisinden, birinci çıkış grup seçicisine blokajlı olan çağırmanın S_1 ve S_0 bağlantı zincirleri,

$$S_1: I_1 \xrightarrow{x_1} Q_2 \xrightarrow{x_2} I_3 \xrightarrow{x_3} Q_4$$

$$S_0: Q_1 \xrightarrow{x_2} I_4$$

şeklinde yazılır.



Şekil 3.12. S_1 ve S_0 bağlantı zinciri oluşturulan üç katlı şebekenin bağlantı durumu.

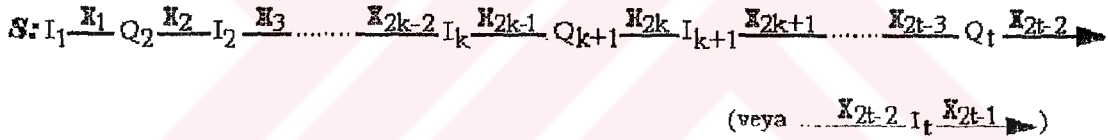
3.5.2 BAĞLANTI ZİNCİRİ İLE YENİDEN DÜZENLEME YÖNTEMİ

TEOREM 3. Giriş grup seçicisi I_1 , ($1 \leq I_1 \leq r$) ile çıkış grup seçicisi Q_1 , ($1 \leq Q_1 \leq r$) arasında iç blokaj olduğunu kabul edelim. Şekil 3.13.'de görüldüğü gibi, I_1 giriş grup seçicisinden başlayan bağlantı zinciri S'yi ele alalım. Eğer giriş ve çıkış grup seçicilerini gösteren değişkenler I_k ve Q_k , ($k=1,2,\dots, 1 \leq I_k \leq r$ ve $1 \leq Q_k \leq r$) ve,

$$I_i \neq I_j \quad (i \neq j \quad i,j = 1,2,\dots) \quad (3.7)$$

$$Q_i \neq Q_j \quad (i \neq j \quad i,j = 1,2,\dots) \quad (3.8)$$

şartlarını gerçekleştirir ve ortakat grup seçicisini gösteren x_i , ($i=1,2,\dots, 1 \leq x_i \leq m$) değişkeni Tablo 3.1'deki şartları sağlar ise, giriş ve çıkış grup seçicileri arasındaki bağlantıyı bozmadan, S 'nin yeniden düzenlenmesi halinde iç blokaj önlenir.



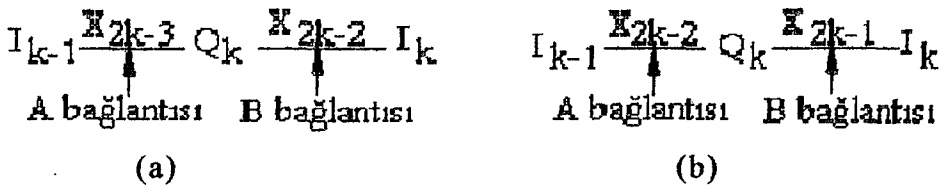
Şekil 3.13. S bağlantı zinciri.

Tablo 3.1 İç blokajlı şebekede S bağlantı zinciri için şartlar.

x_1	Q_1 'in serbest ortakat grup seçicisi.
x_2	I_1 'in serbest ortakat grup seçicisi.
x_{2k} ($k=2, 3, \dots$)	Ya I_k 'nın serbest ortakat grup seçicisi ya da x_{2k-2} ortakat grup seçicisidir.
x_{2k-1} ($k=2, 3, \dots$)	Ya Q_k 'nın serbest ortakat grup seçicisi ya da x_{2k-3} ortakat grup seçicisidir.

İSPAT: Bağlantı zinciri S'de ortakat grup seçicisi x_i ile yapılan bağlantının x_{i+1} ortakat grup seçicisine yeniden düzenlendiği kabul edilsin. Eğer, (3.7), (3.8) eşitlikleri ve Tablo 3.1 'deki şartlar sağlanıyor ise, giriş ve çıkış grup seçicileri arasında var olan bağlantılar aşağıdaki nedenlerden dolayı değişmez.

Önce S bağlantı zincirindeki çıkış grup seçicisi Q_k ($k=2, 3..$) 'da bulunan bağlantılar dikkate alınsın. (3.8) eşitliğinden görüldüğü üzere Q_k 'daki bağlantılar arasında, yeniden düzenlenen bağlantılar, Şekil 3.14.'de gösterildiği gibi iki şekilde ifade edilebilir. I_{k-1} ve Q_k arasındaki bağlantı A, I_k ve Q_k arasındaki bağlantı B olsun. S bağlantı zinciri, Q_k 'da son bulduğunda B bağlantısı meydana gelmez. Bu, Şekil 3.13.'de $k=t$ değişken dönüşümü yapıldığında da açıkça görülmektedir. A bağlantısının yeniden düzenlenmesinin Q_k 'daki diğer bağlantıları etkilemeyeceği açıktır. Çünkü, A'nın ortakat grup seçicisi, x_{2k-3} 'den x_{2k-2} 'ye yeniden düzenlenmiştir. Bu yeniden düzenlemeyle etkilenen bağlantılar, Q_k çıkış grup seçicisinden x_{2k-2} ortakat grup seçicisine yapılan bağlantıdır. Bağlantı B 'dir, fakat B de etkilenmez; zira B bağlantısı, yeniden düzenlenmiş x_{2k-2} ortakat grup seçicisi yerine x_{2k-1} seçicisi geçmiştir. x_{2k-3} ve x_{2k-1} 'in farklı olduğu açıktır. Çünkü I_k 'daki bağlantılar iki farklı çıkış grup seçicisine aynı ortakat grup seçicisi üzerinden bağlanacağından, bu ortakat grup seçicileri farklı olmak zorundadır.



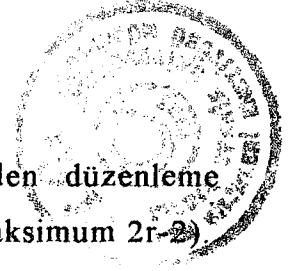
Şekil 3.14. Q_k çıkış grup seçicisindeki bağlantıların yeniden düzenlenmesi.

- Yeniden düzenlemeden önceki bağlantı zinciri.
- Yeniden düzenlemeden sonraki bağlantı zinciri.

Benzer şekilde, B bağlantısının yeniden düzenlenmesinin, Q_k 'daki diğer bağlantıları etkilemeyeceği görülür. B bağlantısı, x_{2k-2} ortakat grup seçicisinden x_{2k-1} 'e yeniden düzenlenir. Tablo 3.1'deki şartlar sağlandığında ya x_{2k-1} , x_{2k-3} ile aynıdır yada Q_k 'nın serbest ortakat grup seçicisidir. x_{2k-1} ile x_{2k-3} 'nın aynı olduğunu kabul edilsin. x_{2k-1} ortakat grup seçicisi üzerinden etkilenebilecek bağlantı A bağlantısıdır, fakat bu bağlantı da etkilenmez. Çünkü A, aynı zamanda x_{2k-1} 'den farklı olan x_{2k-2} ortakat grup seçicisine yeniden düzenlenmiştir. Şimdi de x_{2k-2} 'nin Q_k 'nın serbest ortakat grup seçicisi olduğu kabul edilsin. Bu durumda Q_k ile x_{2k-1} arasında bir bağlantı bulunmamaktadır. A bağlantısı, x_{2k-1} 'den farklı olan bir ortakat grup seçicisi olarak yeniden düzenlendiğinden, B bağlantısının yeniden düzenlenmesi diğer bağlantıları etkilemez.

Böylece Q_k çıkış grup seçicisindeki bağlantılar yeni düzenleme ile etkilenmezler. Benzer şekilde I_k ($k=2,3,\dots$) 'daki bağlantıların yeni düzenlemeden etkilenmedikleri gösterilebilir. I_1 'deki bağlantılar arasındaki yeniden düzenlenen bağlantı, I_1 ile Q_2 arasındaki bağlantılardan biridir. Bu bağlantı, I_1 için serbest olan x_2 ortakat grup seçicisine yeniden düzenlendiğinden diğer bağlantılardan bağımsızdır. Böylece x_i 'nin x_{i+1} 'e yeniden düzenlenmesi esnasında, giriş ve çıkış grup seçicileri arasındaki bağlantı ilişkileri sabit kalır.

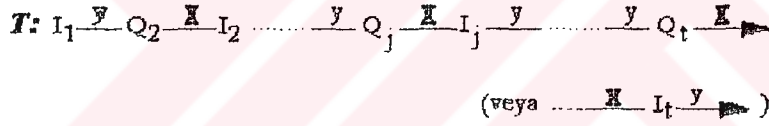
(3.8) eşitliğine göre, Q_1 bağlantı zincirinde bulunmaz; sonuç olarak, Q_1 'deki bağlantı yeniden düzenlenmez. Buradan da yeniden düzenleme sonrasında x_1 , Q_1 'in serbest ortakat grup seçicisi olarak kalır. Diğer taraftan I_1 'in x_1 ortakat grup seçicisi ile olan bağlantısı, x_2 olarak yeniden düzenlenir ve x_1 , I_1 'in serbest ortakat grup seçicisi olur. Diğer bir değişle yeniden düzenleme sonucunda, I_1 ve Q_1 'in serbest ortakat grup seçicisi olan x_1 üzerinde bağlantı sağlanarak iç blokajın önlendiği görülür. (3.7) ve (3.8) eşitliğinden görüldüğü gibi, S bağlantı zincirindeki maksimum giriş grup seçici sayısı r ve maksimum



çıkış grup seçici sayısı $r-1$ 'dir. Bunun sonucunda yeniden düzenleme prosedürü, sınırlı sayıda yolun değiştirilmesiyle tamamlanır (maksimum $2r-2$).

Teorem 3, bağlantı zincirindeki yeniden düzenleme ile ortadan kaldırılabilecek iç blokajın yeter şartlarını gösterir. Ortakat grup seçicisi değişkenlerince sağlanan şartlara bağlı olarak, bağlantı zincirini sağlayan yeter şart, bir tek olmak zorunda değildir. Aşağıdaki teorem bağlantı zincirlerinden birine uygulanır.

TEOREM 4. İç blokajın, giriş grup seçicisi I_1 ($1 \leq I_1 \leq r$) ile çıkış grup seçicisi Q_1 ($1 \leq Q_1 \leq r$) arasında olduğu kabul edilsin. Şekil 3.15.'de gösterilen T bağlantı zinciri, I_1 için x serbest ortakat grup seçicisi ve Q_1 için y serbest ortakat grup seçicisi ($1 \leq x, y \leq m$ olmak üzere) Tablo 1 ve (3.7), (3.8) eşitliklerdeki şartları sağlar.



Şekil 3.15. iki ortakat grup seçicisi ile T bağlantı zinciri.

İSPAT: T bağlantı zinciri, Şekil 3.13.'deki S bağlantı zincirinin bir özel durumudur. Ortakat grup seçicisini gösteren x_i değişkeni aşağıdaki şartları sağlar.

$$x_1 = x_3 = \dots = x_{2k-1} = \dots = y \quad (k=1,2,3,\dots) \quad (3.9)$$

$$x_2 = x_4 = \dots = x_{2k} = \dots = x \quad (k=1,2,3,\dots) \quad (3.10)$$

x ve y 'nin tanımlarına göre x_1 ve x_2 'nin Tablo 3.1 'deki şartı sağladığı açıktır. $x_{2k-1} = x_{2k-3} = y$ ve $x_{2k} = x_{2k-2} = x$ olduğundan Tablo 3.1'deki şart aynı zamanda x_i , $(3 \leq i)$ için de geçerlidir.

I_k ve Q_k 'nn $(k=1,2,...)$, (3.7) ve (3.8) eşitliklerini sağladıkları görülür. Şekil 3.15.'den görüldüğü gibi I_j $(2 \leq j)$ giriş grup seçicisinden Q_j 'ye x ortakat grup seçicisi üzerinden bir bağlantı yoktur. Bunun sonucu olarak,

$$I_i \neq I_j \quad (2 \leq j) \quad (3.11)$$

dir. Eğer,

$$I_i = I_j \quad (2 \leq i < j) \quad (3.12)$$

olursa, x ortakat grup seçicisi üzerinden, I_i ve I_j giriş grup seçicileri ile bağlanan çıkış grup seçicileri içinde

$$Q_i = Q_j \quad (3.13)$$

eşitliği geçerlidir. (3.13) eşitliğinden y ortakat grup seçicisi üzerinden bağlanan, Q_i ve Q_j çıkış grup seçicileri ile giriş grup seçicileri aynı olsun. Bu kabulden,

$$I_{i-1} = I_{j-1} \quad (3.14)$$

olur. Benzer şekilde, aşağıdaki sonuç çıkarılabilir:

$$I_1 = I_{j-i+1} \quad (3.15)$$

Bu son eşitlik (3.11) 'e zıttır. Dolayısıyla

$$I_i \neq I_j \quad (2 \leq i < j) \quad (3.16)$$

dır. Benzer şekilde,

$$Q_i \neq Q_j \quad (2 \leq i < j) \quad (3.17)$$

dir. I_k , ($k = 1, 2, \dots$) 'nın (3.7) eşitliğini sağladığı, (3.11) ve (3.16) eşitliklerinden yararlanarak aşağıdaki gibi gösterilebilir.

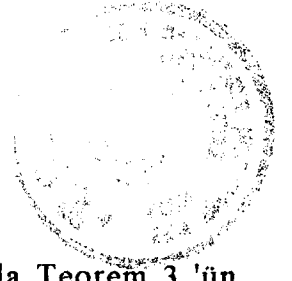
Q_1 çıkış grup seçicisi ile y ortakat grup seçicisi arasında bir bağlantı olmamakla beraber, Q_j , ($2 \leq j$) çıkış grup seçicisi daima y ortakat grup seçicisi üzerinden I_{j-1} 'e bağlıdır. Sonuç olarak;

$$Q_1 \neq Q_j \quad (2 \leq j) \quad (3.18)$$

dir. Q_k , ($k = 1, 2, \dots$) 'nın (3.8) eşitliğini sağladığı, (3.17) ve (3.18) eşitliklerinden çıkarılabilir.

İki ortakat grup seçicisi x ve y kullanılarak gerçekleştirilen bağlantı zinciri, T ile verilmiştir. Teorem 4, iç blokajı kaldırmak için Teorem 3 'e göre yeter şartı sağlayan bağlantı zincirlerinden birisini gösterir.

İç blokajı ortadan kaldırmak için yeter şartı sağlayan, yani bazı k 'lar için x_{2k-1} veya x_{2k} ($k = 2, 3, \dots$), Q_k ve I_k 'nın serbest ortakat grup seçicileri, bağlantı durumuna bağlı olarak T 'den başka bağlantı zinciri olmayabilir. Örneğin, bu durumda, x_1 ve x_2 ortakat grup seçicileri haricinde giriş ve çıkış grup seçicileri için başka serbest ortakat grup seçicisi yoktur. Bağlantı zinciri tanımlarının özelliklerden görülebileceği gibi, bağlantı durumlarından bağımsız olarak, Teorem 3 'ün yeter şartını sağlayan en az bir bağlantı zinciri vardır.



3.5.3 YENİDEN DÜZENLEME ALGORİTMASI

Yeniden düzenleme algoritmasına göre, iç blokaj oluştuğunda Teorem 3 'ün yeter şartını sağlayan bir bağlantı zinciri araştırılır ve yeniden düzenleme yapılır. I_1 giriş grup seçicisi ve Q_1 çıkış grup seçicisi arasındaki iç blokajın, T bağlantı zincirindeki yeniden düzenleme ile kaldırılabilceği, daha önce ispatlanmıştır. Bu yeniden düzenleme algoritması, T bağlantı zincirini araştırır ve yeniden düzenlemeyi yapar. Şekil 3.14.'de verilen bağlantı zincirine ait algoritmanın akışı, Şekil 3.16.'da gösterilmiştir.

T bağlantı zincirinden farklı olarak, 3.17.'de gösterilen T^* bağlantı zincirini ele alalım. Burada T^* , T 'nin I_1 için x serbest ortakat grup seçicisi ve Q_1 için y serbest ortakat grup seçicisi üzerinden bağlantı kurulmuş şeklindedir. Fakat T^* 'ın son bağlantısında x ve y ortakat grup seçicileri değil de z ortakat grup seçicisi kullanılmasıyla, T 'den ayrılır. T^* bağlantı zincirinin, iç blokajı kaldıran Teorem 3'ün yeter şartını sağladığı aşağıda gösterilmektedir.

$$T^*: I_1 \xrightarrow{y} Q_2 \xrightarrow{x} I_2 \xrightarrow{y} \dots \xrightarrow{x} I_{t-1} \xrightarrow{y} Q_t \xrightarrow{z} \rightarrow$$

Q_t çıkış grup seçicisi için serbest ortakat grup seçicisi z 'dir.

$$(veya \dots \xrightarrow{y} Q_t \xrightarrow{x} I_t \xrightarrow{z} \rightarrow)$$

I_{t-1} giriş grup seçicisi için serbest ortakat grup seçicisi z 'dir.

Şekil 3.16. T^* bağlantı zinciri.

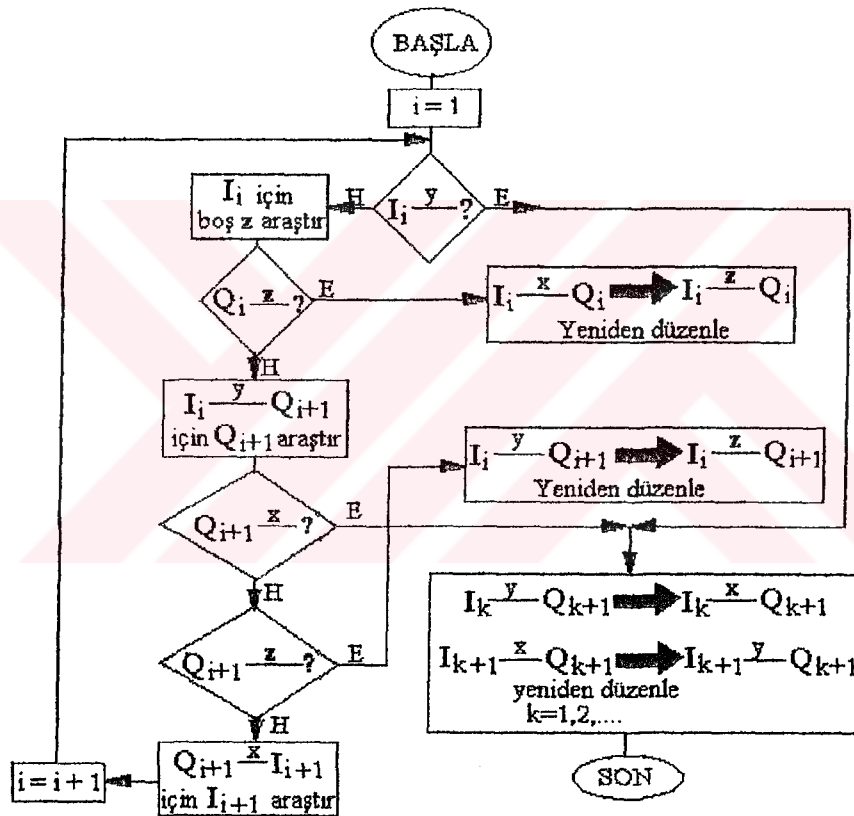
T 'de kurulan yollar T^* ile aynıdır. Bu nedenle (3.7) ve (3.8) eşitlikleri, I_k ve Q_k giriş ve çıkış grup seçicilerine T bağlantı zincirinde olduğu gibi uygulanır. Ortakat grup seçicisi için T 'dekinden tek fark, sonunda z ortakat grup seçicisinin bulunmasıdır. z ortakat grup seçicisinin dışındaki x ve y ortakat grup seçicileri, Tablo 1 'deki şartı sağlar. Şekil 3.16. 'daki T^* 'ın tanımından açıkça görüldüğü gibi, z ortakat grup seçicisi için Tablo 3.1'deki şart sağlanır.

Böylece yeniden düzenlenmiş T^* bağlantı zincirinin, iç blokajı kaldırdığı görülür (Şekil 3.17.).

$$T^*: I_1 \xrightarrow{x} Q_2 \xrightarrow{y} I_2 \xrightarrow{x} \dots \xrightarrow{y} I_{t-1} \xrightarrow{z} Q_t$$

$$(veya \dots \xrightarrow{x} Q_t \xrightarrow{z} I_t)$$

Şekil 3.17. Yeniden düzenlenmiş bağlantı zinciri T^* .



Şekil 3.18. T^* algoritması

Şekil 3.18, C yoksa T 'yi araştıran, T^* bağlantı zincirinin algoritmasının akışını gösterir. T^* bağlantı zinciri yönteminde üçüncü ortakat grup seçicisinin bağlantı zincirinin sonunda olup olmadığına bakılmaksızın, T

bağlantı zinciri yönteminde olduğu gibi, bağlantı zinciri x ve y ortakat grup seçicisi üzerindeki bağlantılar araştırılır.

T* kullanıldığında işlem zamanı, yeniden düzenlenen bağlantıların sayısı ile orantılıdır. T* ve T bağlantı zincirleri ile gerçekleştirilen yeniden düzenleme yöntemleri karşılaştırınca, aşağıdaki belirtilen özellikler ortaya çıkar.

1- Aynı bağlantı durumundaki şebekenin blokajının kaldırılacağı düşünölsün. T* bağlantı zinciri ile gerçekleştirilen yeniden düzenleme yöntemindeki bağlantıların sayısı γ ve T bağlantı zinciri ile gerçekleştirilen yeniden düzenleme yöntemindeki bağlantıların sayısı σ ise

$$\gamma \leq \sigma \leq 2n-2$$

dir. Dolayısıyla T* bağlantı zinciri ile gerçekleştirilen yeniden düzenleme yöntemindeki bağlantıların sayısı daha azdır.

2- Şekil 3.18.'de göröldüğü gibi T* bağlantı zincirinin araştırılması, T bağlantı zinciri ile gerçekleştirilen yeniden düzenleme yönteminden daha karmaşıktır. Aynı sayıda yeniden düzenlenen bağlantıyı bulmak için harcanan zaman, T bağlantı zincirini ile gerçekleştirilen yeniden düzenleme yöntemindeki iki katı kadardır.

Buna göre hangi yöntemin kullanılacağına karar verilmesi için dikkatli bir inceleme gerekir.

3.6 PAULL YÖNTEMİNİN SİMÜLASYON SONUÇLARI

Bu bölümde verilen çeşitli yeniden düzenleme algoritmaları ile blokajın kaldırılabilceğı görölmüştür. Burada değışiklik yapılacak ortakat grup

seçicileri keyfi seçilmiştir. Bununla birlikte ortakat grup seçicilerinin seçilme kurallarının, yeniden düzenleme yöntemlerini etkilediği görülmüştür. Yöntem 1, karmaşık olmaması nedeniyle, gerçek bağlaşma sistemlerinde kolayca kullanılabilir. Bu yöntemle ilgili simülasyon sonuçları [10] 'da verilmiştir.

Burada yeniden düzenleme algoritması 3 ana adım içerir.

1. İki ortakat grup seçicisinin seçimi,
2. Yeniden düzenleme için bağlantı dizisinin araştırılması,
3. Yeniden düzenlemenin yürütülmesi.

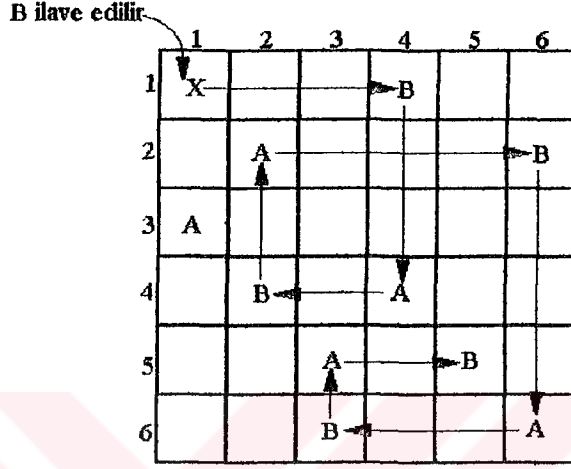
Şekil 3.19.'da (3,3) 'deki blokajı kaldırmak için;

- Önce değişiklik yapılacak ortakat grup seçicileri daire içine alınır.
- Daire içindeki A 'lar B 'ye değiştirilir.
- Daire içindeki B 'ler A 'ya değiştirilir.

		Çıkış grup seçici					
		C ₂	C ₅	C ₁	C ₃	C ₄	
		1	2	3	4	5	6
Giriş grup seçici	1						
	R ₃ 2	A				B	
	R ₄ 3		(B)		(A)		
	R ₁ 4	B		X			
	R ₂ 5			(A)	(B)		
	6						

Şekil 3.19. (3,3)'ün (X) blokajının Yöntem 1 ile kaldırılması (R₅ tanımlanmamış).

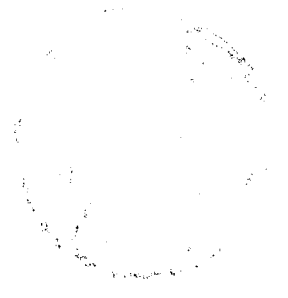
Böylece (R_1, C_1) , A üzerinden bağlanarak blokaj kaldırılır. Yöntem 1, Şekil 3.20. 'deki gibi de uygulanabilir [10]. Burada değişikliğe karar verilen ortakat grup seçicileri sırası ile B 'yi A ile, A 'yı B ile değiştirilerek blokaj kaldırılır.



Şekil 3.20. Yöntem 1 'in uygulanması, ile X blokajının kaldırılması

Yeniden düzenleme, en az kullanılan ortakatlar üzerinde uygulanmasıyla, yeniden düzenleme zamanını azaltacağı söylenebilir. Aynı şekilde zincir yöntemi de belirli ortakat grup seçicileri kullandığından önce bağlantı zincirinin bulunması ve daha sonra yeniden düzenleme yönteminin uygulanması düzenleme zamanını etkiler.

Neiman [14][15] yönteminde değişiklik daha az gibi görülmesine rağmen işlem zamanı açısından, şebeke boyutları küçük olan sistemlere uygulandığından daha verimli olacağı söylenebilir. Bu yöntemin $n=m=4$, $r=10$ için simülasyon sonucu [15] verilmiştir.



BÖLÜM 4

4.1 TEKLİF EDİLEN ALGORİTMANIN DAYANDIĞI TEMELLER

Yeniden düzenleme algoritmalarının kullanılması kurulu olan şebekeden daha fazla verim almayı amaçlar. Özellikle tektaş (monolitik) veya dağınık SPC santrallarda bu algoritmalar kullanılarak verim artırılabilir.

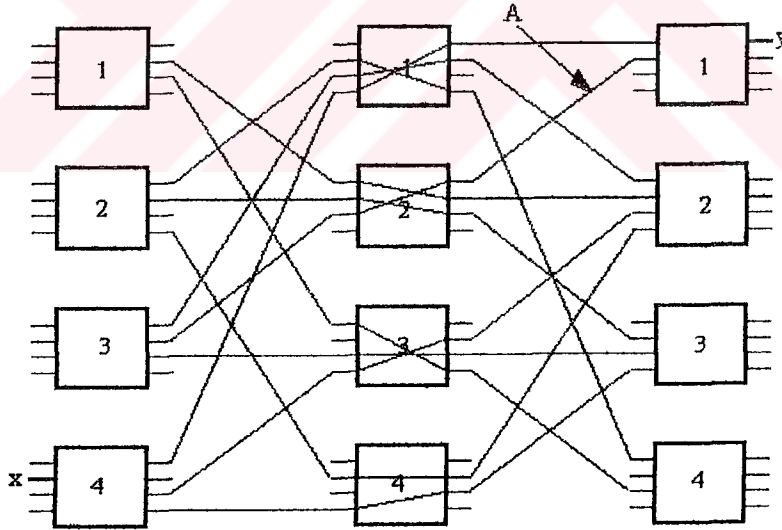
Yeniden düzenleme çalışmaları C. Clos (1953)'un, blokajsız şebekenin teoremini vermesiyle başlamıştır. D. Slepian (1952), yaptığı çalışmada n , giriş veya çıkış grup seçicisinin giriş veya çıkış sayısı olmak üzere, maksimum yeniden düzenlemedeki bağlantı sayısının $2n-2$ olduğunu ortaya atmış; M.C. Paull (1962), bölüm 3.1.1'de verilen teoreme bu sınırın $n-1$ olacağını göstermiştir. Paull ayrıca, bölüm 3.2 de anlatılan iki ortakat grup seçicisinin bağlantılarını değiştiren yeniden düzenleme yöntemlerini ileri sürmüştür. Daha sonra V. E. Benes (1965), yeniden düzenlemenin matematiksel teorisini geliştirmiştir. V. I. Neiman (1969), kurulu olan çağrılmaların yeniden düzenlenmesi için, yalnız giriş ve çıkış grup seçicilerini saklayarak bütün ortakat grup seçicilerinin bağlantılarını yeniden oluşturan bir algoritma vermiştir. G. Hebuterne de kitabında (1987), bu çalışmayı yeniden düzenleme algoritması olarak benimsemiştir. Yine bu çalışma da geçen algoritmayı Nelson T. Tsao-Wu (1974)'de, kullanmıştır ve makelesinde $m=n=4$ ve $r=10$

olan 40x40 boyutunda bağlaşma şebekesini PDP-II bilgisayarı ile gerçekleştirerek bir bağlantı durumunu vermiştir. K. Gotoh (1978), bölüm 3.5 de anlatılan zincir notasyonunu kullanarak, ZUZ şebekesinde bir yeniden düzenleme algoritması geliştirmiştir. A.Jajszczyk ve J. Rajski (1980), bölüm 3.2 de anlatılan Yöntem 1'i kullanarak $n=m=32$ ve $r=8$ olan şebeke üzerinde çalışmış, en az kullanılan ortakat grup seçicileri ile düzenleme yapılması durumunda iyi sonuçlar alındığı göstermiştir. C.M. Melas (1983), 1024 girişli ve 1024 çıkışlı, 16x31 boyutunda 64 giriş ve çıkış grup seçicisi ile 64x64 boyutunda 31 ortakat grup seçicisinden oluşan ($n=16$, $m=31$, $r=64$) bağlaşma şebekesi gerçekleştirerek, Z-80A ile 1200 byte'lık yeniden düzenleme programı kullanmıştır. Bu programla 16 çağırma için yeniden düzenleme bir saniyeden daha az sürmüştür. S. Ohta (1987), seçiciler arasında birden fazla bağlantı olan (PCM) şebekeler için yeniden düzenleme algoritması geliştirmiştir. S. Ohta ve H. Ueda zincir notasyonunu kullanarak yaptıkları simülasyonda $m=n$ ve $m=n+2$ olan 9000 giriş ve çıkışlı şebeke için bölüm 3.5.3 de verilen yeniden düzenlenebilir kontrol algoritması ile yeniden düzenlenen bağlantı sayısının 20'den az olduğunu göstermişlerdir.

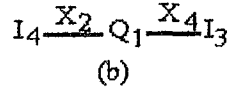
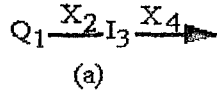
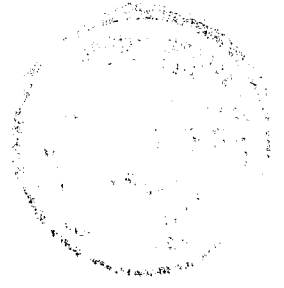
Yapılan çalışmalarda, yeniden düzenleme algoritmaları genellikle iki ortakat grup seçicisinin bağlantılarının değiştirilmesiyle gerçekleştirilmiştir. Ancak S. Ohta ve H. Ueda çalışmalarında, söz edilen ortakat grup seçicilerini belli kurala göre seçerek bütün ortakat grup seçicilerinin kullanılabileceğini vurgulamışlardır. Tezdeki çalışma, bu makaleden hareket edilerek ortaya atılmıştır ve ortakat grup seçicilerinin seçiminde herhangi bir şart ileri sürmeden bütün ortakat grup seçicileri üzerinden yeniden düzenleme yapıp yapılamayacağı araştırılmıştır. Ortakat grup seçicilerinin, yeniden düzenlemede belli şartlarda seçilmesi ancak; çağırmanın kurulması aşamasında da bu şartlar göz önüne alınırsa gerçekleşebilir. Bu ise, çağırmanın rastgele olmasından dolayı çok zordur. Eğer çağırma istekleri belli şartlarda ortakat grup seçicisine bağlanır ise, yeniden düzenlemede buna

uygun bir seçme yöntemi uygulanabilir. Çalışmadaki teklif edilen algorithmada, bağlantılar rastgele ortakat grup seçicileri üzerinden yapıldığından düzenlemede de buna uygun olarak, rastgele ortakat grup seçicileri seçilmiştir.

Örnek olarak $n=m=r=4$ olan şebekede çağırıcı ve çağrılan giriş ve çıkış grup seçicileri sırasıyla $I=\{2, 3, 4, 1, 2, 4, 3, 4, 3, 2, 1, 4, 3, 2, 1, 1\}$ ve $Q=\{4, 2, 1, 2, 3, 2, 3, 3, 1, 2, 4, 1, 3, 4, 4, 1\}$ kabul edilsin. Bağlantılar, birinci çağırıcıdan başlayarak, sırasıyla birinci ortakat grup seçicisinden başlayan denemelerle kurulsun. Şebekeye (4,1) çağırması geldiğinde Şekil 4.1'de görüldüğü gibi blokaj oluşur. Blokajı anında, I_4 'ün serbest ortakat grup seçicisi x_2 ve Q_1 'in serbest ortakat grup seçicileri x_3 ve x_4 'tür. Q_1 çıkış grup seçicisine göre bağlantı zinciri Şekil 4.2'deki gibi elde edilir. Bu düzenleme,



Şekil 4.1 (4,1) çağırması (x,y) için blokajlı şebekenin bağlantı durumu



Şekil 4.2 (a) Birinci çıkış grup seçicisinin bağlantı zinciri.
(b) Yeniden düzenlenen bağlantı zinciri.

Şekil 4.1'de A bağlantısının dördüncü ortakat grup seçicisi üzerine aktarılmasıyla gerçekleştirilir. I_4 ve Q_1 giriş ve çıkış grup seçicisi için bağlantı, Şekil 4.2 (b)'den de görüldüğü gibi x_2 ortakat grup seçicisi üzerinden sağlanır.

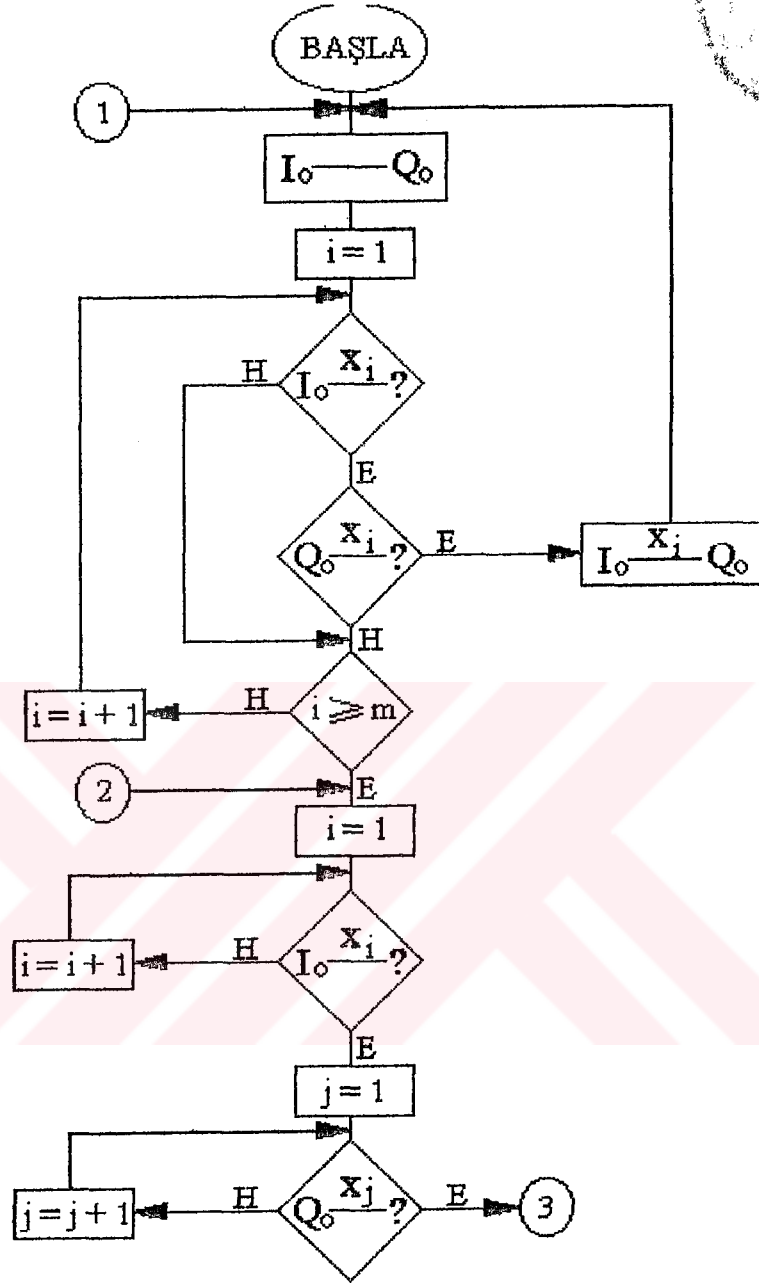
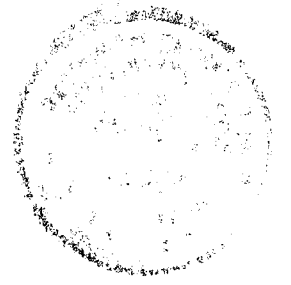
4.2 TEKLİF EDİLEN ALGORİTMA

Teklif edilen ve Şekil 4.3 (a) ve (b)'de akış diyagramı verilen algoritma, genel olarak iki bölüme ayrılabilir. Birinci bölümde çağırma istekleri üretilir ve uygun ortakat grup seçicisi bulunması halinde çağırma kurulur. İkinci bölümde ise, kurulamayan çağırmaların kurulması için önceki kurulu çağırmalar yeniden düzenlenir.

Birinci bölüm iki adımdan oluşur:

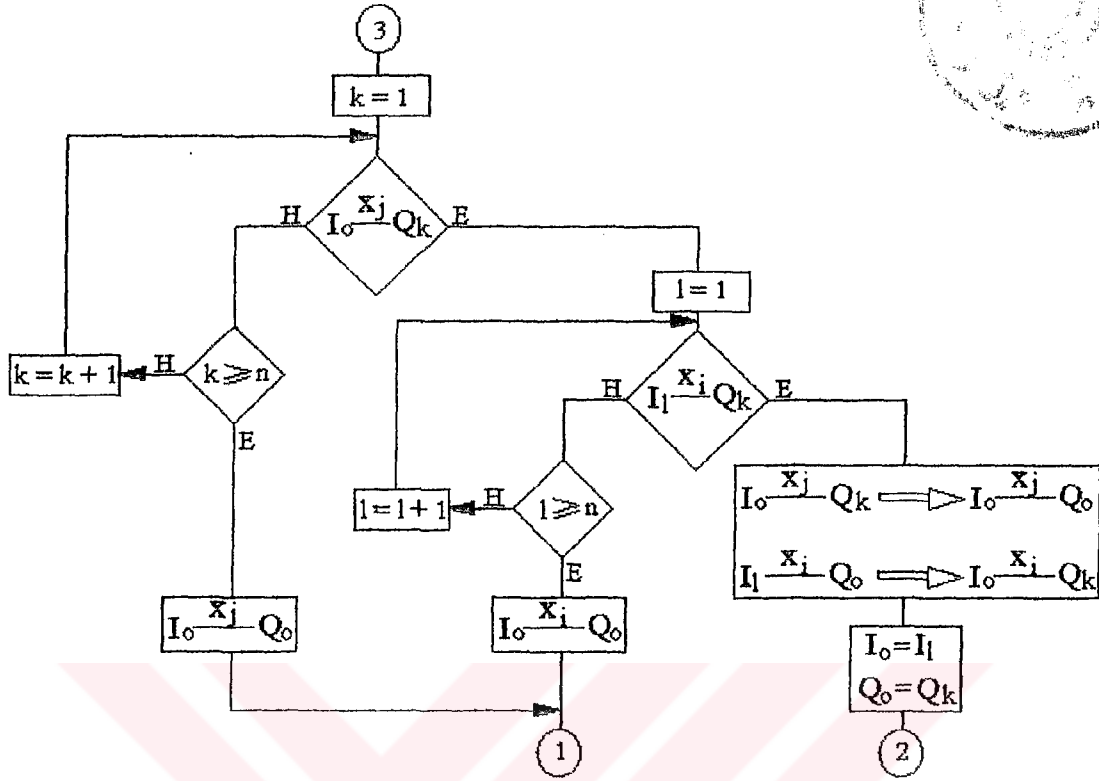
1. Adım: Çağırma isteklerinin üretilmesi: Çağırma istekleri 9000 giriş arasından rastgele üretilerek, bellekte bir dizide saklanır.

2. Adım: Çağırma isteklerinin kurulması: Dizideki çağırma istekleri diziliş sırasına göre kurulur. 30 giriş grup seçicisinin her birinde 300 çağırma olduğundan, toplam 9000 çağırma söz konusudur. Çağırmanın bağlanışında, bellek adresindeki bit değerine bakılarak meşgul yada serbest durumuna karar verilir.



Şekil 4.3(a) Teklif edilen algoritma

Bağlantı haritası: Teklif edilen algorithmada, 300x300 boyutundaki giriş ve çıkış grup seçicilerinden 30'ar adet vardır. Üç katlı bağlaşma şebekesinde ortakat grup seçicisi yöneltme amacıyla kullanılacağından, mn koşulunu sağlayacak şekilde alınmış ve m=n=300 seçilmiştir. Şebekedeki bir çağırma, 30 giriş grup seçicisinin herhangi birinden çıkıp, 300 ortakat grup seçicisinin



Şekil 4.3(b) Teklif edilen algoritma.

herhangi biri üzerinden, çağrılan 30 çıkış grup seçicisinden herhangi birine bağlanabilir. Örneğin 1. giriş grup seçicisi, k. ortakat grup seçicisi üzerinden 1. çıkış grup seçicisine bağlanabilir. Bu bağlantı $(1,k,1)$, $k=1,\dots,300$, şeklinde gösterilir.

Yukarıdan da görülebileceği gibi, oluşturulacak bağlantı haritası üç boyutlu bir matris olarak alınabilir. Matrisin boyutları $(30,300,30)$ olmalıdır. Matrisin eleman sayısı $30 \times 300 \times 30 = 270000$ 'dir. Her bir bağlantının durumu 1 byte'ta saklanır ise, bu matris için 270 Kbyte'lık bellek gerekir. Ancak bu kadar büyük bir bellek uygun görülse de bu boyuttaki matrisin programda çalıştırılması bazı üst seviye dilleri için mümkün değildir. Örneğin, PASCAL

5.0'da bu boyuttaki matrisi çalıştırmak mümkün olmamıştır. Dolayısıyla bit değerine göre kontrol yapılır ise bu bellek alanı $270000 \div 8 = 33750$ byte olur.

Teklif edilen algoritmada 9000 çağırma için 33750 byte'lık bir bellek gerekmektedir. Programda, çağırma giriş grup seçicisi ile çağırılan çıkış grup seçicisi arasında daha önceden bağlantının olup olmadığı araştırılır. Eğer bağlantı yoksa, uygun ortakat grup seçicisi bulunur ve bu giriş-ortakat-çıkış grup seçicilerini gösteren bit adresi 1 yapılarak bağlantı kurulur. Eğer bu çağırma için uygun ortakat grup seçicisi m denemede bulunamazsa, algoritmanın ikinci bölümündeki yeniden düzenleme aşamasında bağlantının kurulmasına çalışılır.

İkinci bölüm ise dört adımdan meydana gelmektedir:

1. Adım: Çağırma çıkan giriş grup seçicisi (I_0)'a uygun ortakat grup seçicisi (x_i) araştırılır. Bu araştırmada, birinci ortakat grup seçicisinden veya son ortakat grup seçicisinden başlayacak şekilde iki farklı yöntem uygulanmıştır. Aynı şekilde çağırılan çıkış grup seçicisi (Q_0) için de uygun ortakat grup seçicisi (x_j) araştırılır.

2. Adım: I_0 giriş grup seçicisinin, x_i ortakat grup seçicisi üzerinden Q_0 çıkış grup seçicisine bağlanabilmesi için, Q_0 çıkış grup seçicisi ile x_i ortakat grup seçicisi üzerinden kurulu olan çağırmanın hangi giriş grup seçicisi (örneğin I_1) ile yapıldığı araştırılır. Kurulu bir bağlantı yoksa, çağırma I_0 , x_i üzerinden Q_0 'a bağlanır ve yeni bir çağırma isteğine geçilir. Eğer bağlantı varsa, bu bağlantı çözülür; I_0 ile Q_0 bağlantısı x_i üzerinden kurulur.

3. Adım: İkinci adımda çözülen giriş ve çıkış grup seçicisinin bağlantısının kurulması: İkinci adımda çözülen I_1 giriş grup seçicisi ile Q_0 çıkış grup seçicisinin bağlantısının birinci adımda bulunan x_j serbest ortakat grup seçicisi üzerinden kurulması aşağıdaki gibi gerçekleştirilir: I_1 giriş grup seçicisi ile x_j ortakat grup seçicisi arasındaki kurulu olan bağlantının hangi çıkış grup seçicisi (Q_k) ile olduğu araştırılır. Kurulu bir bağlantı yoksa,

çağırma I_i , x_j üzerinden Q_o 'a, bağlanır ve yeni bir çağırma isteğine geçilir. Eğer bağlantı varsa, bu bağlantı çözülür. I_i , x_j üzerinden Q_o 'a, bağlanır.

4. Adım: İkinci adımda çözülen I_i giriş grup seçicisi ile üçüncü adımda çözülen Q_k çıkış grup seçicisi yeni bir çağırma isteği gibi düşünülür ($I_i \Rightarrow I_o$, $Q_k \Rightarrow Q_o$) ve birinci adıma dönülerek algoritmanın ikinci bölümü tekrarlanır.

4.3. YENİDEN DÜZENLEMEDE ORTAKAT GRUP SEÇİCİSİNİN BULUNMASI

Teklif edilen algoritmanın 4. adımında, blokajlı çağırmanın bağlanmasından sonra çözülen (I_i, Q_o) veya (I_i, Q_k) için yeni uygun ortakat grup seçicisinin bulunması iki yöntemle olabilir:

1. (I_i, Q_o) çağırmasının bağlanmasında I_i ve Q_o için ayrı ayrı uygun ortakat grup seçicisi bulunur. Bu ortakat grup seçicisi her ikisi için de uygun ise, bağlantı yapılarak yeni çağırma isteği beklenir. Aksi durumda, bulunan ortakat grup seçicisi üzerindeki bağlantı çözülüp (I_i, Q_o) 'ın bağlantısı yapılarak, yeni çözülen bağlantının başka yoldan kurulmasına çalışılır.

2. Birinci yöntemde çözülen çağırmadaki giriş ve çıkış grup seçicilerine uygun ortakat grup seçicisinin araştırması, bulunan ilk ortakat grup seçicisinde son bulur. Bundan sonra uygun ortakat seçicisi varsa bile, bunun araştırılması yapılmaz. Çünkü, bu işlemin zaman açısından bir bedeli olduğundan, yeniden düzenlemede önemli miktarda gecikme olabilir. Bu aşamada, ortakat grup seçicisinin aranma yöntemini değiştirmenin mümkün olup olmayacağı araştırılabilir. Eğer bu işlem 9000 giriş-çıkış için $30 \times 300 \times 30$ boyutunda bir bağlantı haritası kullanılarak yapılırsa, 300 ortakat grup seçicisine ayrı ayrı bağlantı yapıp yapılmayacağının kontrolü gerektiğinde en fazla 300 çevrim söz konusudur. Diğer bir yol, 9000 giriş çıkışlı bağlantı şebekesini $300 \times 30 \times 300$ boyutunda alarak, ortakat grup seçicilerinin sayısını

300'den 30'a indirmek ve uygun ortakat grup seçicisini arama zamanını azaltmak olabilir. Bu yolun kullanımı, $300 \times 30 \times 300 = 2,700,000$ bitlik bir hafıza gerektirir.

Bu düşünce de, bağlantı haritasının tek bir değişkenle gösterilmesinden yola çıkılmıştır. Eğer bağlantı haritası birden fazla değişken ile gösterilebilirse nasıl bir yol tutulmalıdır?

Bunu çözmek için üç ayrı değişken tanımlanabilir: Birinci değişken (I), her bir giriş grup seçicisinin bağlandığı; ikinci değişken (Q); her bir çıkış grup seçicisinin bağlandığı ortakat grup seçicilerini gösterebilir. Bir üçüncüsü (B)'de bağlaşma şebekesinin eş zamanlı bağlantısı kadar uzunlukta olan, bağlantı yapılan çağrılarını gösteren değişken olsun. Örneğin bir çağrımda, 10. giriş grup seçicisinden, 20. çıkış grup seçicisine, 30. ortakat grup seçicisi üzerinden bağlantı yapılacak ise (10, 30, 20); I değişkeninin 10. elemanının 30. biti 1, Q değişkeninin 20. elemanının 30. biti 1 olur ve B değişkeni de 103020 yapılır.

Belli bir giriş ve çıkış için uygun ortakat grup seçicisinin bulunmasında, bu seçicilere bağlı olmayan ortakat grup seçicilerini bulmak için I ve Q nun bitlerinin tümleyeni alınır ve I, Q ile AND' lenir, elde edilen 1'ler bağlantı yapılabilecek orta kat grup seçicilerini gösterir.

C ' programlama dilinde bir tam sayıya 2 byte uzunluğunda yer ayrılır. Simülasyonda kullanılan bağlaşma şebekesinde, her bir giriş veya çıkış grup seçicisinin ortakata bağlantısı $300/16=19$ kelimelik yerde saklanır. Böylece 300 orta kat grup seçicisinin araştırılması, maksimum $19 \times 16=304$ adımda yapılırken, hem zamandan kazanılmış, hem de bütün ortakat grup seçicilerinin araştırılması yapılmış olur. Bu şebeke için gerekli olan bellek kapasitesi ise, toplam $19 \times 30 + 19 \times 30 + 9000 \times 4 = 37.140$ kByte 'tır. Bu kapasite ilk teklif edilen

bağlantı haritası için 16.875 kByte 'lık bellek kapasitesine göre %100 artmıştır. Ancak arama adım sayısı 300'den 35'e inmiştir.

Teklif edilen algoritma ile her iki bağlantı haritası denenerek ikinci bağlantı haritasının daha az sürede yeniden düzenlemeyi gerçekleştirdiği görülmüştür.

İkinci bağlantı haritasıyla yeniden düzenlemenin bir örneği, 10 tane 30x30 giriş çıkış grup seçicisinden oluşan bağlaşma şebekesi için 1. yeniden düzenleme arama yöntemine göre yapılmış ve çağırma istekleri Tablo E.1 ile, sonuçlar da Tablo E.2.'de verilmiştir. Bu tabloda 260-269, 270-279, 280-289 ve 290-299 çağırımları için şebekenin bağlantı durumu ve yeniden düzenleme gerektiren çağırımların bağlantı değişikliği sayısı ile ayrıca yeniden düzenlemede geçen zaman verilmiştir.

4.4. GELİŞTİRİLEN ALGORİTMANIN VE S.OHTA VE H.YUEDA ALGORİTMALARININ SONUÇLARININ KARŞILAŞTIRILMASI

Bağlaşma şebekelerinde çağırmanın bağlanması için mevcut yollardan birisinin seçiminde çeşitli kurallar kullanılabilir [7][5].

1. Bağlantıların homojen bir dağılım göstermesi amacıyla ardışıl bir kural (sequential rule) kullanılabilir. Bu kural, bir başlangıç noktasından başlayarak, uygun ortakat grup seçicisi bulunduğunda bağlantıyı gerçekleştiren ve daha sonra gelen bir çağırmanın bağlantısı için ise, aramaya bir sonraki ortakat grup seçicisinden başlayarak devam eden bir yöntemdir.

2. Aynı noktadan başlayarak bağlantının gerçekleştirilmesi yüklü ortakat grup seçicisi kullanma kuralıdır. Böylece boş matrisleri zor durumlarda kullanmak imkanı sağlanır.

3. Rastgele bir ortakat grup seçicisinden başlayarak bağlantının gerçekleştirilmesi kuralıyla, bağlantının belli bir ortakat grup seçicisinde yığılma olasılığı azaltılır.

4. Çağırma yeniden paketleme (call-repacking) kuralı. Bağlaşma şebekesinde blokaj olduğunda, bütün bağlantıların ardaşıl kural kullanılarak yeniden bağlaştırılması yöntemidir.

Geliştirilen algoritmada ortakat grup seçicilerinin seçiminde ilk üç kural kullanılarak alınan sonuçlar Tablo 4.6. -Tablo 4.11. de verilmiştir. Tablolarda 8841-8880, 881-8920, 8921-8960 ve 8961-9000 çağırma bağlantıları kurulurken, yeniden düzenleme süresinin belli bir değeri aşma olasılığı verilmiştir.

Yüklü ortakat grup seçicisinden başlayarak yapılan bağlaşmada; S.Ohta ve H.Yueda'nın algoritmasının simülasyonunda 8881-8920 arasındaki bağlantı denemesinde yeniden düzenleme süresinin 0.2 saniyelik belli değeri aşma olasılığı %3'tür. Aynı şartlarda, tezde geliştirilen algoritma ile bu olasılık "0" bulunmuştur (Tablo E.3. ve Tablo E.4.). 8961-9000 arasındaki bağlantılarda, yeniden düzenlemede ortalama bağlantı değiştirme sayısı iki algoritmada hemen hemen eşittir. Diğer durumlarda geliştirilen algoritma daha iyidir.

Ortakat grup seçicisi seçiminde ardışıl kural kullanıldığında ise; S.Ohta ve H.Yueda 'nın algoritma sonuçları yeniden düzenlenmenin 0.2 saniyeden uzun sürmesi olasılığı 8961-9000 için %5, 8881-8920 için %8, 8921-8960 ve 8961-9000 için %15'dir. Geliştirilen algoritmanın yine 0.2 'den daha kısa sürede bağlantıyı gerçekleştirdiği görülmüştür. Geliştirilen algoritmanın yeniden düzenlemedeki ortalama bağlantı değişikliği de adı geçen algoritmanın sonuçlarından daha iyidir (Tablo E.5. ve Tablo E.6.).

Ortakat grup seçicisi seçiminde rastgele kural kullanıldığında, S. Ohta ve H. Yueda'nın algoritmanın yeniden düzenleme süresinin 0.5 saniyeden daha uzun sürme olasılığı %2 civarında olurken, geliştirilen algorithmada ise 0.2 saniyenin üzerine çıkma olasılığı bile sıfırdır. S.Ohta ve H.Yueda 'nın algoritmasıyla 8941-9000 için yeniden düzenlemede bağlantı değiştirme sayısı 11 olduğu halde geliştirilen algorithmada bu değer 8 civarındadır (Tablo E.7. ve Tablo E.8.).

Bir başka similasyon da, 9000 çağırmandan 8950 çağırmanın bağlandığı durum alınmış ve sonra rastgele bir tanesi çözülerek kalan 51 çağırmandan rastgele birinden üretilen bu çağırmanın bağlanabilmesi için gerekli yeniden düzenleme sayısı ve yeniden düzenleme süresi incelenmiştir. Buna uygun olarak yapılan 1000 denemenin sözü edilen her iki algorithmayla elde edilen sonuçları, Tablo E.9. ve Tablo E.10.'de verilmiştir. Sözü edilen algorithmalarda 1000 çağırmandan yeniden düzenleme isteği yaklaşık aynı olup, ortalama 700 civarındadır. Buna karşı teklif edilen algorithmanın, yeniden düzenlenen çağırmalardaki ortalama bağlantı değişikliği sayısı 9 iken, diğer algorithmanın ki 14 civarındadır. Yeniden düzenlemenin 0.1 saniyeden daha uzun sürede olması olasılığı, ise geliştirilen algorithmayla sıfır bulunurken S.Ohta ve H. Yueda'nın algoritmasında %3'lardedir.

Dolayısıyla her iki karşılaştırma açısından da geliştirilen algoritma daha iyidir.

Geliştirilen algoritma ile elde edilen diğer bir sonuç ise, Tablo E.8.'dan da görüldüğü gibi ortakat grup seçicilerinin seçiminde "rastgele seçme kuralını" kullanmanın daha iyi olacağıdır.



BÖLÜM 5

SONUÇ

Telefon haberleşme sistemlerinde santrallarda yer elen çok katlı seçme organları bir bağlaşma şebekesi oluşturur. Bağlaşma şebekelerinin daha az blokaj ile hizmet vermesi, bağlaşma şebekesinin iyi düzenlenmesine ve işletilmesine bağlıdır. Örneğin, üç katlı bir şebekede giriş ve çıkış grup seçicileri ile ortakat grup seçicileri arasındaki bağlantılar yeniden düzenlenerek blokaj olasılığı azaltılır.

Yeniden düzenleme konusunda çeşitli algoritmalar geliştirilmiştir. Bu çalışmaların en önemlilerinden bir tanesi de S. Ohta ve H. Yueda 'nın çalışmasıdır. S. Ohta ve H. Yueda'nın geliştirdiği algoritma ile önceki yöntemlere göre yeniden düzenlemedeki bağlaşma sayısı ve zamanında büyük bir iyileştirme sağlanmıştır.

Bu çalışmada teklif edilen algoritma, yukarıda bahsedilen çalışma temel alınarak geliştirilmiştir. Teklif edilen algoritma ile temel alınan algoritmanın bilgisayar programları çalıştırılmış ve sonuçları 4. bölümde verilmiştir. Bu sonuçlar karşılaştırıldığında, teklif edilen algoritmanın daha az yeniden düzenleme sayısı ve zaman gerektirdiği açıkça görülmektedir. 1000 denemede yeniden düzenleme gerektiren çağrımaların ortalama bağlantı değişikliği sayısı S. Ohta ve H. Yueda algoritması için 15 civarında iken, teklif edilen algortmada bu sayı 9 civarındadır. Yine buna paralel olarak bu bağlantılar için gereksinim duyulan ortalama süreler de, söz konusu algoritmalar için sırasıyla 0.018 ve 0.006 saniye mertebesindeir.

KAYNAKLAR

- 1-Ackroyd, M.H., Call Repacking in Connecting Networks, IEEE Transactions on Communications, Vol.Com-27, No. 3, March 1979, pp 589-591.
- 2- Bear, D., Principles of Telecommunication-Traffic Engineering, IEE Telecommunications Series 2
- 3- Benes, V.E., Mathematical Theory of Connecting Networks and Telephone Traffic, Academic Press, New York, 1965.
- 4- Bellamy, J., Digital Telephony, John Wiley & Sons, New York, 1982.
- 5- Gotoh, K., Rearrangement Characteristics in Time-Division T-S-T Networks, Electronics and Communications in Japan, Vol. 61-B, No. 3, 1978, pp 76-82.
- 6- DURUSOY, G., Telekomünikasyon Bağlaşma Sistemleri, 1990
- 7- Hébuterne, G., Traffic Flow in Switching Systems, Artech House, Norwood, 1987
- 8- Hwang, F.K., Control Algorithms for Rearrangeable Clos Networks, IEEE Transactions on Communications, Vol.Com-31, No. 8, August 1983, pp 952-954.
- 8- Inose, H., An Introduction to Digital Integrated Communications Systems, Peter Peregrinus Ltd., Stevenage, UK, 1981
- 9- Jajszczyk, A., and Rajski, J., 1980. The effect of choosing the switches for rearrangements in switching networks, IEEE Transactions on Communications Vol. COM-28, No. 10, October 1980, pp 1832-1835
- 10- Jajszczyk, A., On Nonblocking Switching Networks Composed of Digital Symmetrical Matrices, IEEE Transactions on Communications, Vol.Com-31, No. 1, January 1983, pp 2-9.

- 11- Melas, C.M., Path Rearranging in a Data Switching Network, IEEE Transactions on Communications, Vol.Com-31, No. 1, January 1983, pp 155-157.
- 12- Martin, J., 1972. System Analysis for Data Transmission, Prentice-Hall, Englewood Cliffs, New Jersey
- 13- Neiman, V.I., 1969. Structure et commande optimales des réseaux connexion sans blocage, Annals des Telecommun., t. 24, no. 7-8, July/Aug. 1969, pp 232-238.
- 14- Nelson T. Tsao-Wu, "On Neiman's Algorithm for the Control of rearrangeable switching networks," IEEE Transactions on Communications Vol. COM-22, No. 6, June 1974, pp 737-742.
- 15- Ohta, S., Ueda, H., A Rearrangement Algorithm for Three-Stage Switching Networks, Electronics and Communications in Japan, Part 1, Vol. 70, No. 9, 1978, pp 68-77.
- 16- Ohta, S., A Simple Control Algorithm for Rearrangeable Switching Networks with Time Division Multiplexed Links, IEEE Journal on Selected Areas in Communications, Vol. SAC-5, No. 8, October 1987, pp 1302-1308.
- 17- ÖZTÜRK, S., DURUSOY, G., Yeniden düzenlenebilir üç katlı bağlaşma şebekelerinde kontrol algoritması, Elektrik Mühendisliği & Ulusal Kongresi 11-17 Eylül 1995 Bursa, sayfa 752-755.
- 18- Paull, M.C., Reswitching of Connection Networks, The Bell System Technical Journal, May 1962, pp 833-855.
- 19- Skaperda, N.J., Some Architectural Alternatives in the Design of a Digital Switch, IEEE Transactions on Communications, Vol.Com-27, No. 7, July 1979, pp 961-972.

Tablo E.1 300 Giriş-çıkış isteği

ÇAĞ. SIRASI	ÇAĞIR. ABONE	GİRİŞ GRUP SEÇİCİSİ	ÇAĞRILAN ABONE	ÇIKIŞ GRUP SEÇİCİSİ	ÇAĞ. SIRASI	ÇAĞIR. ABONE	GİRİŞ GRUP SEÇİCİSİ	ÇAĞRILAN ABONE	ÇIKIŞ GRUP SEÇİCİSİ
0	4	1	207	7	32	217	8	83	3
1	2	1	147	5	33	35	2	283	10
2	101	4	21	1	34	122	5	176	6
3	10	1	67	3	35	37	2	205	7
4	107	4	252	9	36	202	7	299	10
5	66	3	20	1	37	143	5	122	5
6	162	6	53	2	38	148	5	29	1
7	59	2	46	2	39	104	4	135	5
8	211	8	197	7	40	261	9	89	3
9	285	10	127	5	41	80	3	267	9
10	83	3	212	8	42	54	2	296	10
11	134	5	80	3	43	127	5	232	8
12	33	2	198	7	44	209	7	17	1
13	210	8	129	5	45	194	7	199	7
14	170	6	164	6	46	188	7	253	9
15	13	1	5	1	47	1	1	144	5
16	50	2	22	1	48	237	8	262	9
17	245	9	30	2	49	81	3	77	3
18	206	7	69	3	50	139	5	36	2
19	230	8	265	9	51	116	4	119	4
20	249	9	4	1	52	114	4	90	4
21	288	10	11	1	53	175	6	132	5
22	129	5	178	6	54	84	3	78	3
23	286	10	248	9	55	52	2	227	8
24	252	9	206	7	56	242	9	190	7
25	277	10	223	8	57	89	3	106	4
26	244	9	26	1	58	123	5	292	10
27	136	5	60	3	59	73	3	288	10
28	182	7	51	2	60	125	5	31	2
29	199	7	282	10	61	17	1	16	1
30	180	7	215	8	62	149	5	221	8
31	165	6	167	6	63	296	10	159	6



Tablo E.1 300 Giriş-çıkış isteği (Devam)

ÇAĞ. SİRASI	ÇAĞIR. ABONE	GİRİŞ GRUP SEÇİCİSİ	ÇAĞIRILAN ABONE	ÇIKIŞ GRUP SEÇİCİSİ	ÇAĞ. SİRASI	ÇAĞIR. ABONE	GİRİŞ GRUP SEÇİCİSİ	ÇAĞIRILAN ABONE	ÇIKIŞ GRUP SEÇİCİSİ
64	21	1	168	6	96	51	2	286	10
65	23	1	189	7	97	100	4	95	4
66	77	3	241	9	98	248	9	274	10
67	293	10	71	3	99	287	10	193	7
68	60	3	13	1	100	298	10	261	9
69	64	3	113	4	101	295	10	209	7
70	75	3	109	4	102	222	8	211	8
71	228	8	153	6	103	153	6	139	5
72	267	9	161	6	104	216	8	203	7
73	272	10	91	4	105	86	3	290	10
74	221	8	62	3	106	192	7	271	10
75	160	6	160	6	107	53	2	279	10
76	11	1	240	9	108	168	6	117	4
77	40	2	33	2	109	233	8	289	10
78	138	5	54	2	110	117	4	187	7
79	146	5	225	8	111	196	7	208	7
80	76	3	134	5	112	39	2	293	10
81	195	7	142	5	113	44	2	186	7
82	266	9	28	1	114	276	10	18	1
83	263	9	128	5	115	197	7	276	10
84	167	6	158	6	116	213	8	166	6
85	154	6	66	3	117	300	11	210	8
86	187	7	25	1	118	169	6	85	3
87	232	8	280	10	119	5	1	169	6
88	71	3	150	6	120	283	10	32	2
89	185	7	201	7	121	72	3	155	6
90	243	9	277	10	122	164	6	192	7
91	189	7	287	10	123	130	5	52	2
92	215	8	146	5	124	251	9	238	8
93	220	8	269	9	125	297	10	2	1
94	166	6	24	1	126	106	4	48	2
95	172	6	228	8	127	282	10	27	1



Tablo E.1 300 Giriş-çıkış isteği (Devam)

ÇAĞ. SIRASI	ÇAĞIR. ABONE	GİRİŞ GRUP SEÇİCİSİ	ÇAĞIRILAN ABONE	ÇIKIŞ GRUP SEÇİCİSİ	ÇAĞ. SIRASI	ÇAĞIR. ABONE	GİRİŞ GRUP SEÇİCİSİ	ÇAĞIRILAN ABONE	ÇIKIŞ GRUP SEÇİCİSİ
128	68	3	94	4	160	299	10	126	5
129	93	4	242	9	161	163	6	184	7
130	270	10	148	5	162	67	3	173	6
131	22	1	123	5	163	62	3	249	9
132	184	7	200	7	164	3	1	47	2
133	25	1	136	5	165	140	5	170	6
134	255	9	39	2	166	137	5	7	1
135	219	8	98	4	167	55	2	3	1
136	203	7	177	6	168	36	2	226	8
137	278	10	92	4	169	18	1	273	10
138	205	7	116	4	170	6	1	105	4
139	227	8	8	1	171	201	7	216	8
140	294	10	182	7	172	24	1	291	10
141	74	3	297	10	173	9	1	270	10
142	124	5	179	6	174	224	8	181	7
143	275	10	120	5	175	47	2	73	3
144	126	5	55	2	176	69	3	196	7
145	102	4	68	3	177	258	9	14	1
146	97	4	258	9	178	274	10	234	8
147	46	2	99	4	179	218	8	219	8
148	31	2	264	9	180	235	8	40	2
149	26	1	43	2	181	262	9	61	3
150	159	6	88	3	182	183	7	163	6
151	264	9	266	9	183	90	4	213	8
152	269	9	247	9	184	292	10	154	6
153	155	6	64	3	185	177	6	137	5
154	247	9	110	4	186	94	4	254	9
155	20	1	204	7	187	63	3	87	3
156	111	4	6	1	188	257	9	250	9
157	30	2	251	9	189	115	4	185	7
158	152	6	256	9	190	57	2	140	5
159	7	1	174	6	191	280	10	15	1

Tablo E.1 300 Giriş-çıkış isteği (Devam)

ÇAĞ. SIRASI	ÇAĞIR. ABONE	GİRİŞ GRUP SEÇİCİSİ	ÇAĞRILAN ABONE	ÇIKIŞ GRUP SEÇİCİSİ	ÇAĞ. SIRASI	ÇAĞIR. ABONE	GİRİŞ GRUP SEÇİCİSİ	ÇAĞRILAN ABONE	ÇIKIŞ GRUP SEÇİCİSİ
192	147	5	59	2	224	253	9	102	4
193	28	1	236	8	225	161	6	194	7
194	105	4	275	10	226	158	6	278	10
195	79	3	23	1	227	112	4	259	9
196	142	5	112	4	228	289	10	156	6
197	190	7	130	5	229	193	7	96	4
198	259	9	115	4	230	279	10	1	1
199	290	10	217	8	231	254	9	257	9
200	135	5	121	5	232	150	6	114	4
201	208	7	74	3	233	265	9	65	3
202	200	7	175	6	234	212	8	298	10
203	56	2	10	1	235	156	6	220	8
204	99	4	202	7	237	29	1	141	5
205	120	5	243	9	238	246	9	57	2
206	178	6	86	3	239	225	8	229	8
207	88	3	138	5	240	131	5	244	9
208	32	2	224	8	241	19	1	239	8
209	179	6	125	5	242	43	2	84	3
210	239	8	37	2	243	144	5	58	2
211	119	4	218	8	244	176	6	81	3
212	191	7	284	10	245	85	3	12	1
213	113	4	152	6	246	98	4	82	3
214	171	6	180	7	247	128	5	281	10
215	181	7	246	9	248	186	7	34	2
216	284	10	70	3	249	103	4	214	8
217	273	10	79	3	250	238	8	260	9
218	49	2	9	1	251	82	3	272	10
219	121	5	131	5	252	108	4	45	2
220	38	2	124	5	253	58	2	75	3
221	157	6	49	2	254	223	8	76	3
222	91	4	41	2	255	27	1	222	8
223	8	1	118	4	256	34	2	195	7

Tablo E.1 300 Giriş-çıkış isteği (Devam)

ÇAĞ. SİRASİ	ÇAĞIR. ABONE	GİRİŞ GRUP SEÇİCİSİ	ÇAĞRILAN ABONE	ÇIKIŞ GRUP SEÇİCİSİ	ÇAĞ. SİRASİ	ÇAĞIR. ABONE	GİRİŞ GRUP SEÇİCİSİ	ÇAĞRILAN ABONE	ÇIKIŞ GRUP SEÇİCİSİ
257	214	8	230	8	259	250	9	231	8
258	204	7	183	7					

Tablo E.1 300 Giriş-çıkış isteği, bağlantı değişikliği ve süresi (Devam)

ÇAĞIRMA SİRASİ	ÇAĞIRAN ABONE	GİRİŞ GRUP SEÇİCİSİ	ÇAĞRILAN ABONE	ÇIKIŞ GRUP SEÇİCİSİ	BAĞLA. DEĞİŞİK.	Y. DÜZ. SÜRESİ
260	207	7	111	4	0	0.0000
261	198	7	165	6	0	0.0000
262	132	5	295	10	0	0.0000
263	16	1	235	8	0	0.0000
264	241	9	72	3	0	0.0000
265	109	4	42	2	0	0.0000
266	240	9	294	10	0	0.0000
267	95	4	19	1	0	0.0000
268	271	10	233	8	0	0.0000
269	48	2	133	5	0	0.0000
270	92	4	104	4	0	0.0000
271	110	4	44	2	0	0.0000
272	256	9	145	5	0	0.0000
273	174	6	143	5	0	0.0000
274	173	6	157	6	0	0.0000
275	226	8	237	8	3	0.0000
276	291	10	245	9	2	0.0000
277	145	5	149	5	0	0.0000
278	70	3	50	2	0	0.0000
279	45	2	101	4	0	0.0000
280	118	4	263	9	0	0.0000
281	87	3	93	4	2	0.0000
282	12	1	268	9	0	0.0000
283	260	9	35	2	3	0.0000
284	268	9	171	6	2	0.0000

Tablo E.1 300 Giriş-çıkış isteği, bağlantı değişikliği ve süresi (Devam)

ÇAĞIRMA SIRASI	ÇAĞIRAN ABONE	GİRİŞ GRUP SEÇİCİSİ	ÇAĞRILAN ABONE	ÇIKIŞ GRUP SEÇİCİSİ	BAĞLA. DEĞİŞİK.	Y. DÜZ. SÜRESİ
285	96	4	300	11	0	0.0000
286	229	8	38	2	5	0.0000
287	41	2	108	4	0	0.0000
288	15	1	188	7	2	0.0000
289	236	8	97	4	3	0.0000
290	141	5	162	6	8	0.0000
291	61	3	100	4	2	0.0000
292	65	3	56	2	0	0.0000
293	42	2	151	6	0	0.0000
294	151	6	63	3	2	0.0000
295	133	5	172	6	8	0.0000
296	231	8	103	4	9	0.0000
297	14	1	255	9	26	0.0000
298	234	8	107	4	3	0.0000
299	281	10	285	10	27	0.0000

Tablo E.2. 260 Bağlantı sonucundaki bağlaşma şebekesinin bağlantı durumu

	1. Çıkış Grup seç.	2. Çıkış Grup seç.	3. Çıkış Grup seç.	4. Çıkış Grup seç.	5. Çıkış Grup seç.	6. Çıkış Grup seç.	7. Çıkış Grup seç.	8. Çıkış Grup seç.	9. Çıkış Grup seç.	10. Çıkış Grup seç.
1. Giriş Grup seçicisi	4,11	17,18	3	20,24	2,5,14,15, 26	7,12,13	1,8,19	6,9,10,28	16,23	21,22
2. Giriş Grup seçicisi	5,12,22,26, 27	2,9	19,23,24	1	17,28		3,6,15,29	8,14,20	13	4,7,10,16, 18
3. Giriş Grup seçicisi	2,25,29		6,8,16	5,9,10, 15	11,12,24	14,17	23,27	1,18	4,7,19	3,13,20,26
4. Giriş Grup seçicisi	1,17	13,20,26	7,12,27	3,6	4	21	10,24,25	19,23	2,9,11,14, 15,22	5
5. Giriş Grup seçicisi	9,21	4,8,10,12, 14,16,19,24	1	22,23	6,20,25	3,5,15,18		7,11,13	26	2,27
6. Giriş Grup seçicisi	15	1,6	11,13,14, 17,20,25	12	7,8,19,23	2,4,9,10,22	16,21,26	3,5,27	18	24
7. Giriş Grup seçicisi	10,14	5,27	2,21	17,26	3,9,22	16,20,24	7,11,13,18, 28	4	8,25	1,6,12,15 19,23
8. Giriş Grup seçicisi	20	3,15	4,10,28	7	13	8,11	2,14,22	12,16,21, 24	1,5,6,27	9,17,25
9. Giriş Grup seçicisi	3,6,8,13, 23	7,25	5,18,22	2,19,21	10,16	1	4,9	15,26	12,17,20, 24	11,14
10. Giriş Grup seçicisi	7,16,18,19, 24,28	11	9,15,26	4,8,13	1,21	6,23,27	5,12,17,20	2,22,25	3,10	

Tablo E.2. 270 Bağlantı sonucundaki bağlaşma şebekesinin bağlantı durumu ("*" yeni çağrılar) (Devam)

	1. Çıkış Grup seç.	2. Çıkış Grup seç.	3. Çıkış Grup seç.	4. Çıkış Grup seç.	5. Çıkış Grup seç.	6. Çıkış Grup seç.	7. Çıkış Grup seç.	8. Çıkış Grup seç.	9. Çıkış Grup seç.	10. Çıkış Grup seç.
1. Giriş Grup seçicisi	4,11	17,18	3	20,24	2,5,14,15, 26	7,12,13	1,8,19	6,9,10,28 29 *	16,23	21,22
2. Giriş Grup seçicisi	5,12,22,26, 27	2,9	19,23,24	1	17,28,30 *		3,6,15,29	8,14,20	13	4,7,10,16, 18
3. Giriş Grup seçicisi	2,25,29		6,8,16	5,9,10 ,15	11,12,24	14,17	23,27	1,18	4,7,19	3,13,20,26
4. Giriş Grup seçicisi	1,17,30 *	13,20,26, 28 *	7,12,27	3,6	4	21	10,24,25	19,23	2,9,11,14, 15,22	5
5. Giriş Grup seçicisi	9,21	4,8,10,12, 14,16,19,24	1	22,23	6,20,25	3,5,15,18		7,11,13	26	2,27,28 *
6. Giriş Grup seçicisi	15	1,6	11,13,14, 17,20,25	12	7,8,19,23	2,4,9,10,22	16,21,26	3,5,27	18	24
7. Giriş Grup seçicisi	10,14	5,27	2,21	17,26,29 *	3,9,22	16,20,24, 30 *	7,11,13,18, 28	4	8,25	1,6,12,15 19,23
8. Giriş Grup seçicisi	20	3,15	4,10,28	7	13	8,11	2,14,22	12,16,21, 24	1,5,6,27	9,17,25, 29 *
9. Giriş Grup seçicisi	3,6,8,13,23	7,25	5,18,22 ,29 *	2,19,21	10,16	1	4,9	15,26	12,17,20, 24	11,14
10. Giriş Grup seçicisi	7,16,18,19, 24,28	11	9,15,26	4,8,13	1,21	6,23,27	5,12,17,20	2,22,25,30 *	3,10	

Tablo E.2. 280 Bağlantı sonucundaki bağlaşma şebekesinin bağlantı durumu ("*" yeni çağrılar) (Devam)

	1.Çıkış Grup seç.	2.Çıkış Grup seç.	3.Çıkış Grup seç.	4.Çıkış Grup seç.	5.Çıkış Grup seç.	6.Çıkış Grup seç.	7.Çıkış Grup seç.	8.Çıkış Grup seç.	9.Çıkış Grup seç.	10.Çıkış Grup seç.
1.Giriş Grup seçicisi	4,11	17,18	3	20,24	2,5,14,15, 26	7,12,13	1,8,19	6,9,10,28 29	16,23	21,22
2.Giriş Grup seçicisi	5,12,22,26, 27	2,9	19,23,24	1,11 *	17,18,28		3,6,15,29	8,14,20	13	4,7,10,16, 30
3.Giriş Grup seçicisi	2,25,29	21 *	6,8,16	5,9,10 ,15	11,12,24	14,17	23,27	1,18	4,7,19	3,13,20,26
4.Giriş Grup seçicisi	1,17,30	13,20,26, 28,29 *	7,12,27	3,6,16 *	4	21	10,24,25	19,23	2,9,11,14, 15,22	5
5.Giriş Grup seçicisi	9,21	4,8,10,12, 14,16,19,24	1	22,23	6,20,25,30 *	3,5,15,18		7,11,13	26	2,27,28
6.Giriş Grup seçicisi	15	1,6	11,13,14, 17,20,25	12	7,8,19,23, 29 *	2,4,9,10,22 ,28 *	16,21,26	3,5,27	18	24
7.Giriş Grup seçicisi	10,14	5,27	2,21	17,26,29	3,9 22	16,20,24, 30	7,11,13,18, 28	4	8,25	1,6,12,15 19,23
8.Giriş Grup seçicisi	20	3,15	4,10,28	7	13	8,11	2,14,22	12,16,21, 24,30	1,5,6,27	9,17,25, 29
9.Giriş Grup seçicisi	3,6,8,13,23	7,25	5,18,22,29	2,19,21	10,16,27 *	1	4,9	15,26 *	12,17,20, 24	11,14
10.Giriş Grup seçicisi	7,16,18,19, 24,28	11	9,15,26	4,8,13	1,21	6,23,27	5,12,20,30	2,22,17,25	3,10,29 *	

Tablo E.2. 290 Bağlantı sonucundaki bağlaşma şebekesinin bağlantı durumu ("*" yeni çağrılar) (Devam)

	1. Çıkış Grup seç.	2. Çıkış Grup seç.	3. Çıkış Grup seç.	4. Çıkış Grup seç.	5. Çıkış Grup seç.	6. Çıkış Grup seç.	7. Çıkış Grup seç.	8. Çıkış Grup seç.	9. Çıkış Grup seç.	10. Çıkış Grup seç.
1. Giriş Grup seçicisi	4,11	17,18	3	20,24	2,5,14,15, 26	7,12,13	1,8,19,27 *	6,9,10,28 29	16,23,30 *	21,22
2. Giriş Grup seçicisi	5,12,22,26, 27	2,9	19,23,24	1,11,25 *	17,18,28		3,6,15,29	8,14,20	13	4,7,10,16, 30
3. Giriş Grup seçicisi	2,25,29	21	6,8,16	5,9,10,15, 28 *	11,12,24	14,30	17,23	1,18	4,7,19	3,13,20,26
4. Giriş Grup seçicisi	1,17,30	13,20,26, 28,29	7,12,27	3,6,16	4	21	10,24,25	19,23	2,9,11,14, 15,18,22 *	5,8 *
5. Giriş Grup seçicisi	9,21	4,8,10,12, 14,16,19,24	1	22,23	6,20,25,30	3,5,15,18		7,11,13	26	2,27,28
6. Giriş Grup seçicisi	15	1,6	11,13,14, 17,20,25	12	7,8,19,23, 29	2,4,9,10,22 26	16,21,30	3,5,27	28	24
7. Giriş Grup seçicisi	10,14	5,27	2,21	17,26,30	3,9 22	16,20,24, 29	7,11,13,18, 28	4	8,25	1,6,12,15 19,23
8. Giriş Grup seçicisi	20	3,15,23 *	4,10,28	7,18 *	13	8,11	2,14,22	12,16,21, 24,30	1,5,6,27	9,17,25, 29
9. Giriş Grup seçicisi	3,6,8,13, 23	7,25,30 *	5,18,22,29	2,19,21	10,16,27	1,28 *	4,9	15,26	12,17,20, 24	11,14
10. Giriş Grup seçicisi	7,16,18,19, 24,28	11	9,15,30	4,8,13	1,21	6,23,27	5,12,20,26	2,22,17,25	3,10,29	

Tablo E.2. 300 Bağlantı sonucundaki bağlaşma şebekesinin bağlantı durumu ("*" yeni çağrılar) (Devam)

	1.Çıkış Grup seç.	2.Çıkış Grup seç.	3.Çıkış Grup seç.	4.Çıkış Grup seç.	5.Çıkış Grup seç.	6.Çıkış Grup seç.	7.Çıkış Grup seç.	8.Çıkış Grup seç.	9.Çıkış Grup seç.	10.Çıkış Grup seç.
1. Giriş Grup seçicisi	4,11	17,25	3	20,24	2,5,14,15, 26	7,12,13	1,8,19,27	6,9,10,28 29	16,18,23, 30 *	21,22
2. Giriş Grup seçicisi	5,12,22,26, 27	2,9	19,23,24	1,11,25	17,18,28	21 *	3,6,15,29	8,14,20	13	4,7,10,16, 30
3. Giriş Grup seçicisi	2,25,29	21,22 *	6,8,16	5,9,10,15, 27,28 *	11,12,24	14,30	17,23	1,18	4,7,19	3,13,20,26
4. Giriş Grup seçicisi	1,17,30	13,18,20, 26, 28	7,12,27	3,6,16	4	25	10,21,24	19,23	2,9,11,14, 15,22,29	5,8
5. Giriş Grup seçicisi	9,21	4,8,10,12, 14,16,19,24	1	22,23	6,20,25,30	3,5,15,17 18,29 **		7,11,13	26	2,27,28
6. Giriş Grup seçicisi	15	1,6	11,13,14,18 7,20,29*	12	7,8,19,21 23	2,4,9,10,22 26	16,25,30	3,5,27	28	24
7. Giriş Grup seçicisi	10,14	5,27	2,26	17,21,30	3,9 22	16,19,20, 24	7,11,13,18, 28	4	8,25	1,6,12,15 23,29
8. Giriş Grup seçicisi	20	3,15,23	4,10,28	7,18,26,29 **	13	8,11	2,14,22	12,16,21, 24,30	1,5,6,27	9,17,19,25
9. Giriş Grup seçicisi	3,6,8,13,23	7,29,30	5,21,22, 25	2,14,19	10,16,27	1,28	4,9	15,26	12,17,20, 24	11,18
10. Giriş Grup seçicisi	7,16,18,19, 24,28	11	9,15,30	4,8,13	1,29	6,23,27	5,12,20,26	2,22,17,25	3,10,21	14 **

Tablo E.3. Yüklü orkat grup seçicisinden başlayan aramada S. Ohta ve H. Yueda' nın algoritmasının sonuçları

Deneme Aralığı	YD'deki maksimum bağlantı	Y.D'deki değiştirilen bağlantıların ortalaması	Y.D'deki ortalama bekleme zamanı $E[t]$	Y.D'deki standart sapma $[\sigma]$	$0.2 / E[t]$	$0.5 / E[t]$	$(E[t] / \sigma)^2$	$t > 0.2$	$t > 0.5$
7841	37	2.975	.0123626	.0287806	16.17779	40.44448	.1845103	0	0
8880	47	3.1	.0164835	.0392386	12.13335	30.33336	.1764706	0	0
	41	1.55	.0151099	.0368326	13.23638	33.09094	.1682893	0	0
8881	39	3.175	.0384615	.0509539	5.200005	13.00001	.5697674	0.02	0
8920	45	4.825	.0315934	.0474050	6.330441	15.82610	.4441646	0.02	0
	25	3.875	.0260989	.0442768	7.663166	19.15791	.3474495	0.02	0
8921	53	9.125	.0576923	.0520530	3.466670	8.666675	1.228412	0.03	0
8960	39	9.3	.0686813	.0532003	2.912003	7.280007	1.666667	0.04	0.000001
	47	8.725	.0631868	.0514697	3.165221	7.913051	1.507123	0.03	0
8961	53	11.1	.076923	.0503579	2.600003	6.500007	2.333333	0.03	0
9000	47	16.6	.0534832	.0534832	2.348389	5.870974	2.535620	0.05	0
	45	12.75	.0810439	.0475639	2.467799	6.169498	2.903253	0.03	0

* Y.D. :yeniden düzenleme

Tablo E.4. Yüklü ortakat grup seçicisinden başlayan aramada teklif edilen algoritmanın sonuçları

Deneme Aralığı	YD'deki maksimum bağlantı	Y.D'deki değiştirilen bağlantıların ortalaması	Y.D'den maksimum süre	Y.D'deki ortalama beklene zamanı $E[t]$	Y.D'deki standart sapma $[\sigma]$	$0.2 / E[t]$	$0.5 / E[t]$	$(E[t] / \sigma)^2$	$t > 0.2$	$t > 0.5$
7841	2	0.15	0	0	0	0	0	0	0	0
8880	3	0.125	0	0	0	0	0	0	0	0
	4	0.325	0	0	0	0	0	0	0	0
	3	0.325	0	0	0	0	0	0	0	0
8881	1	1.225	0	0	0	0	0	0	0	0
8920	25	2.625	0.054945	0.00137363	0.00857829	145.6001	364.0004	0.0256410	0	0
	5	1.475	0	0	0	0	0	0	0	0
	11	1.825	0.054945	0.00274725	0.01197499	72.80007	182.0002	0.0526316	0	0
8921	36	5.475	0.054945	0.01447204	0.01447204	48.53338	121.3335	0.0810811	0	0
8960	47	8.175	0.054945	0.01447204	0.01447204	48.53338	121.3335	0.0810811	0	0
	30	7.85	0.054945	0.0054945	0.0164835	36.40004	91.00009	0.1111111	0	0
	29	5.425	0.054945	0.01447204	0.01447204	48.53338	121.3335	0.0810811	0	0
8961	41	14.725	0.054945	0.00824175	0.01961929	24.26669	60.66673	0.1764706	0	0
9000	51	14.725	0.054945	0.00686813	0.01817135	29.12003	72.80007	0.1428571	0	0
	61	16.3	0.054945	0.00686813	0.01817135	29.12003	72.80007	0.1428571	0	0
	100	20.25	0.054945	0.00686813	0.01817135	29.12003	72.80007	0.1428571	0	0

* Y.D. :yeniden düzenleme

Tablo E.5. Ardışıl arama yapılarak S.Ohta ve H. Yueda'nın algoritmasıyla elde edilen sonuçları

Deneme Aralığı	YD'deki maksimum bağlantı	Y.D'deki değiştirilen bağlantıların ortalaması	Y.D'deki ortalama bekleme zamanı $E[t]$	Y.D'deki standart sapma $[\sigma]$	$0.2 / E[t]$	$0.5 / E[t]$	$(E[t] / \sigma)^2$	$t > 0.2$	$t > 0.5$
8841	31	1.65	.0164835	.0392386	12.13335	30.33336	.1764706		0
8880	33	2.625	.0521978	.1453448	3.831583	9.578957	.1289746	0.08	0
	31	3.75	.0343406	.0941209	5.824006	14.56001	.1331203	0.05	0
8881	31	4.575	.0686813	.1220905	2.912003	7.280007	.3164557	0.11	0.02
8920	21	2.575	.0412088	.0941710	4.853338	12.13335	.1914894	0.06	0
	41	4.65	.032967	.0503579	6.066673	15.16668	.4285714	0.02	0
8921	41	7.475	.0796703	.0918995	2.510347	6.275868	.7515639	0.11	0.01
8960	33	7.7	.1043955	.1367016	1.915791	4.789478	.5831987	0.17	0.03
	39	6.1	.1277471	.1848585	1.565593	3.913982	.4775551	0.21	0.05
8961	53	12.675	.0920329	.0432419	2.173137	5.432841	4.529768	0.3	0
9000	47	13.525	.0857829	.0857829	2.080002	5.200005	1.256410	0.12	0.01
	55	16.55	.1102243	.1102243	1.797533	4.493832	1.018947	0.17	0.02

* Y.D. :yeniden düzenleme

Tablo E.6. Ortakat grup seçicisini ardışıl aramada tekliif edilen algoritmanın sonuçlar

Deneme Aralığı	YD'deki maksimum bağlantı	Y.D'deki değıştirilen bağlantıların ortalaması	Y.D'den maksimum süre	Y.D'deki ortalama bekleme zamanı E[t]	Y.D'deki standart sapma $[\sigma]$	0.2 / E[t]	0.5 / E[t]	$(E[t] / \sigma)^2$	$t > 0.2$	$t > 0.5$
7841	1	0.05	0	0	0				0	0
8880	2	0.175	0	0	0				0	0
	2	0.175	0	0	0				0	0
	2	0.125	0	0	0				0	0
8881	0	0.9	0.054945	0.00274725	0.0119749	72.80002	182.0002	0.0526316	0	0
8920	7	1.125	0.054945	0.00274725	0.0119749	72.80002	182.0002	0.0526316	0	0
	2	0.45	0	0	0				0	0
	5	0.6	0	0	0				0	0
8921	19	4.35	0.054945	0.00412088	0.01447204	48.53338	121.3335	0.0810811	0	0
8960	13	2.225	0.054945	0.0013736	0.00857829	145.6001	364.0004	0.0256410	0	0
	17	2.8	0.054945	0.00412088	0.01447204	48.53338	121.3335	0.0810811	0	0
	15	2.9	0.054945	0.00412088	0.01447204	48.53338	121.3335	0.0810811	0	0
8961	61	9.55	0.054945	0.00274725	0.01197499	72.80007	182.0002	0.0526316	0	0
9000	63	14.825	0.054945	0.00961538	0.02087729	20.80002	52.00005	0.2121212	0	0
	56	8.75	0.054945	0.00824175	0.01961929	24.26669	60.66673	0.1764706	0	0
	50	10.975	0.054945	0.00274725	0.01197499	72.80007	182.0002	0.0526316	0	0

* Y.D. :yeniden düzenleme

Tablo E.7. Rastgele bir noktadan başlayıp arayan S.Ohta ve H. Yueda'nın algoritmasının sonuçları

Deneme Aralığı	YD'deki maksimum bağlantı	Y.D'deki değiştirilen bağlantıların ortalaması	Y.D'deki ortalama bekleme zamanı $E[t]$	Y.D'deki standart sapma $[\sigma]$	$0.2 / E[t]$	$0.5 / E[t]$	$(E[t] / \sigma)^2$	$t > 0.2$	$t > 0.5$
8841	33	5.35	.0714285	.1444333	2.800003	7.000007	2445731	.89	.97
8880	41	3.9	.0950088	.0950088	4.044448	10.11112	.2709030	.93	1.0
	45	4.325	.0941710	.0941710	4.853338	12.13335	.1914894	.94	1.0
8881	21	3.75	.0934065	.1401905	2.141179	5.352947	.4439324	.85	.97
8920	45	6.875	.0741758	.1213464	2.696299	6.740747	.3736545	.89	.97
	41	6.175	.0686813	.1220905	2.912003	7.280007	.3164557	.89	.97
8921	53	9.575	.1195054	.1524105	1.663565	4.183912	.6148160	.55	.96
8960	43	8.825	.1126373	.1334327	1.775612	4.439029	.7125901	.57	.97
	33	6.525	.1401098	.1611069	1.427452	3.568631	.7563245	.76	.95
8961	47	13.675	.120879	.1058315	1.654547	4.136368	1.304582	.81	.98
9000	45	12.65	.1071428	.0876971	1.866669	4.666671	1.492640	.86	.99
	45	11.25	.1126373	.1105405	1.775612	4.439029	1.038295	.82	.98

* Y.D. :yeniden düzenleme

Tablo E.8 Rastgele bir noktadan başlayıp arayan teklif edilen algoritmayla elde edilen sonuçlar

Deneme Aralığı	YD'deki maksimum bağlantı	Y.D'deki değiştirilen bağlantıların ortalaması	Y.D'den maksimum süre	Y.D'deki ortalama bekleme zamanı $E[t]$	Y.D'deki standart sapma $[\sigma]$	$0.2 / E[t]$	$0.5 / E[t]$	$(E[t] / \sigma)^2$	$t > 0.2$	$t > 0.5$
7841	0	0	0	0	0	0	0	0	0	0
8880	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0
8881	0	0	0	0	0	0	0	0	0	0
8920	1	0.025	0	0	0	0	0	0	0	0
	2	0.05	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0
8921	3	0.5	0.054945	0	0	0	0	0	0	0
8960	4	0.875	0	0	0	0	0	0	0	0
	9	0.1625	0	0	0	0	0	0	0	0
	10	1.75	0.054945	0.00137363	0.00857829	145.6001	364.0004	0.0256410	0	0
8961	48	8.95	0.054945	0.00412088	0.0144720	48.53338	121.3335	0.0810811	0	0
9000	38	6.775	0.054945	0.00274725	0.0119749	72.80007	182.0002	0.0526316	0	0
	57	10.75	0.054945	0.00412088	0.0144720	48.53338	121.3335	0.0810811	0	0
	10	1.9487	0.054945	0.00140885	0.0086847	141.9601	354.9004	0.0263158	0	0

* Y.D. yeniden düzenleme

Tablo E.9. S. Ohta ve H. Yueda' nın yöntemiyle bağlaşma şebekesindeki 9000 abonenin 8950' sinin bağlantısı yapılarak, kalan 50' si içinden 1000 bağlantı deneme sonucu

Ortakat grup seçicilerin seçilme yöntemi	Yeniden düzenleme ile bağlanılan çağırma	YD'deki maksimum bağlantı	Y.D'deki değiştirilen ortalama bağlantı	Y.D'deki çağırmanın ortalama bekleme zamanı $E[t]$	Y.D' deki çağırma- nın standart sapması $[\sigma]$	$0.1 / E[t]$	$0.2 / E[t]$	$0.5 / E[t]$	$(E[t] / \sigma)^2$	$t > 0.2$	$t > 0.5$
En fazla kullanılan grup seç.	726	53	13.82920	.0176339	.0340138	5.67090	11.34181	28.35453	.2687721	0	0
	694	53	15.56484	.0176552	.0290011	5.66404	11.32808	28.32021	.3706113	0	0
	708	55	14.98305	.0185478	.0439172	5.39147	10.78294	26.95735	.1783677	0	0
Rastgele	839	55	14.50894	.0185333	.0292360	5.39569	10.79139	26.97847	.4018555	0	0
	816	59	14.63971	.0195944	.0293745	5.10351	10.20702	25.51755	.4449611	0	0
	819	57	14.82662	.0191871	.0286145	5.21182	10.42365	26.05912	.4496213	0	0
Sıralı	698	57	14.57020	.0169243	.0266960	5.90865	11.81731	29.54329	.4019111	0	0
	736	53	14.89946	.0189620	.0285231	5.40865	10.54741	26.36853	.4419510	0	0
	748	55	15.18449	.0175560	.0272982	5.69607	11.39215	28.48036	.4135996	0	0

* Y.D. :yeniden düzenleme



Tablo E.10. Teklif edilen yöntemle bağlaşma şebekesindeki 9000 abonenin 8950 'sinin bağlantısı yapılarak, kalan 50 'si içinden 1000 bağlantı deneme sonucu

Ortakat grup seçicilerin seçilme yöntemi	Yeniden düzenleme ile bağlanılan çağırma	YD 'deki maksimum bağlantı	Y.D 'deki değiştirilen ortalama bağlantı	Y.D 'deki çağırma ortalama bekleme zamanı E[t]	Y.D 'deki çağırma sapma zamanı [σ]	0.1 / E[t]	0.2 / E[t]	0.5 / E[t]	(E[t] / σ) ²	t > 0.2	t > 0.5
En fazla kullanılan grup seç.	720	56	8.863889	0.00663919	0.01790842	15.06208	30.12417	75.31042	0.1374408	0	0
	726	56	8.8744656	0.00514636	0.01600881	19.43119	38.86239	97.15598	0.1033435	0	0
	718	48	8.661560	0.00512718	0.01598202	19.50390	39.00780	97.51950	0.1029186	0	0
	699	59	8.277539	0.00691725	0.01822691	14.45661	28.91321	72.28303	0.1440262	0	0
Rastgele	734	54	8.756131	0.0058388	0.0169329	17.12668	34.25337	85.63342	0.1189024	0	0
	753	60	8.884462	0.0056915	0.0167430	17.57002	35.14004	87.85009	0.1155556	0	0
	705	62	9.621277	0.0072481	0.0185933	13.79679	27.59358	68.98394	0.15196608	0	0
	754	65	9.177719	0.0057568	0.0168276	17.37065	34.74130	86.85325	0.1170370	0	0
Sıralı	672	58	8.700893	0.0057234	0.0167844	17.472010	34.94403	87.36009	0.1162791	0	0
	744	51	8.990591	0.0065727	0.0178308	15.214390	30.42879	76.07199	0.1358779	0	0
	675	50	8.826667	0.0057794	0.0168567	17.302830	34.60567	86.51417	0.1175497	0	0
	718	54	8.983287	0.0054333	0.0164016	18.405090	36.81018	92.02544	0.1097372	0	0

* Y.D. :yeniden düzenleme

E 11. S.OHTA ve H. YUEDA'NIN ALGORİTMALARININ C++ PROGRAMI

```
#include <alloc.h>
#include <dos.h>
#include <conio.h>
#include <stdlib.h>
#include <time.h>
#include <stdio.h>
#include <math.h>
```

```
void dosya(void);
void sifirla(void);
void call(void);
void called(void);
void rear_i1(int);
void rear_i2(int);
void route(int);
void rear_x2(int,int);
void rear_z2(int,int);
void oku(int,int,int);
void cozulme(int,int,int);
void baglasma(int,int,int);
void new_call_called(int);
```

```
int huge *h;
unsigned int huge *p;
```

```
int const n=300, r=30, m=302, kk=8840;
int cikis[9000], giris[9000], repeat=40;
FILE *di,*call_f,*called_f,*estab;
```

```
void main()
{
    int f,x,y,x2,y2,i,j,jj,t,q,top1,top2;
    char *string1;
```

```
clrscr();
```

```
p=(unsigned int huge *) farcalloc((16990),sizeof(int));
h=(int huge *) farcalloc(1,sizeof(int));
```

```
sifirla();
printf("cagırmaların uretilmesi\n");
```

```

//      call();
        call_f=fopen("cagri.dat", "rb");
        fread(giris,sizeof(int),9000,call_f);
        fclose (call_f);

        randomize();
        printf("cagrilanlarin uretilmesi\n");
//      called();
        called_f=fopen("istek.dat", "rb");
        fread(cikis,sizeof(int),9000,called_f);
        fclose(called_f);

        j=-1;
        while (j!=-1)
        {
            printf("random baglantilar olusturuluyor\n");

do { j++;
    gotoxy(30,10);
    printf("kurulan=%d ",j);
        x=(giris[j]-1)/n;
        y=(cikis[j]-1)/n;
        jj=-1;
        i=random(m);
        do { top1=0;q=-1;
            if (i<=1 || i>=m ) i=-1;
                i++;jj++;
            do { q++;
                oku(q,i,y);
                top1+=*h;
            } while (q<(r-1) && top1==0);
            top2=1;
            while (top1==0)
            { top1=1;top2=0;q=-1;
                do { q++;
                    oku(x,i,q);
                    top2+=*h;
                } while (q<(r-1) && top2==0);
            }
        } while (jj<(m-1) && top2!=0);

        oku(x,i,y);
        if (top2==0 && *h==0) baglasma(x,i,y);
        else
        {
            o {
                rear_i1(x); i=*h;
                rear_x2(i,y); x2=*h;

```

```

        if (x2==1) { baglasma(x,i,y);
                    continue;
                }
        else { cozulme(x2,i,y);
              baglasma(x,i,y);
              x=x2;
            }
        rear_i2(y); i=*h;
        rear_z2(x,i); y2=*h;

        if (y2==1) { baglasma(x,i,y);
                    continue;
                }
        else { cozulme(x,i,y2);
              baglasma(x,i,y);
              y=y2;
            }
        } while (x2>-1 && y2>-1);
    } // (top2==0 && *h==0) else
} while (j<(kk-1));
}

    estab=fopen("doktor.dat","rb");
    fread(p,sizeof(int),16990,estab);
    fclose(estab);

di=fopen("dok.dat", "w");

for (f=0; f<4; f++)
{

    j=f*repeat+kk-1;
    for(t=0; t<=repeat-1; t++)
    { j++;
      gotoxy(30,12);
      printf("tekrar kurulmakta= %d, k=%5d f=%d ",j,t,f);
      new_call_called(j);
      route(j);
    }
    fprintf(di,"===== %d\n",f+1);
} //for f
fclose(di);
farfree(p);
farfree(h);

```

```

}

/* _____ */

void sifirla()
{   int i;
    for (i=0; i<=n*r-1; i++) { giris[i]=0; cikis[i]=0;}

}

void call()
{   int i,j,gir;
    randomize();
    for (j=0; j<=(kk-1); j++)
        { gotoxy(30,5); printf("j= %d",j);
          i=-1;
          while (i==1)
              { gir=random(n*r)+1;
                do i++;
                while (i<j && giris[i]!=gir);
                if (gir==giris[i]) i=-1;
              }
          giris[j]=gir;
        }
}

void called()
{   int i,j,gir;
    randomize();
    for (j=0; j<=(kk-1); j++)
        { gotoxy(30,5);printf("j= %d",j);
          i=-1;
          while (i==1)
              { gir=random(n*r)+1;
                do i++;
                while (i<j && cikis[i]!=gir);
                if (gir==cikis[i]) i=-1;
              }
          cikis[j]=gir;
        }
}

void oku(int x1,int y1,int z1)
{   unsigned int s,u;
    div_t xyl;
    div_t xy;

```

```

        xy1=div(906*x1,16); // m*r=906*10
        xy=div(30*y1+z1+10*xy1.rem,16);
        u=10*xy1.quot+xy.quot;
        s=(p+u) & ((unsigned int)pow(2,xy.rem));
        if (s!=0) *h=1;
        else *h=0;
    }

```

```

void baglasma(int x1,int y1,int z1)
{
    unsigned int u;
    div_t xy1;
    div_t xy;

    xy1 = div(906*x1,16);
    xy = div(30*y1+z1+10*xy1.rem,16);
    u=10*xy1.quot+xy.quot;
    *(p+u)|=((unsigned int)pow(2,xy.rem));
}

```

```

void cozulme(int x1,int y1,int z1)
{
    unsigned int u;
    div_t xy1;
    div_t xy;

    xy1 = div(906*x1,16);
    xy = div(30*y1+z1+10*xy1.rem,16);
    u=10*xy1.quot+xy.quot;
    *(p+u)^=((unsigned int)pow(2,xy.rem));
}

```

```

/* x1 giris y1 orta z1 cikis katlarini gosterir */
/* once x1 ve y1 sabit birakip z1 degistirilecek */
/* baglanti yoksa y1 ve z1 sabit birakilip x1 degistirilecek */

```

```

void rear_il(int x1)
{
    int y1,q,top1;
    y1=-1;
    do { q=-1;top1=0; y1++;
        do { q++;
            oku(x1,y1,q);
            top1+=*h;
        } while (q<(r-1) && *h==0);
    } while (top1!=0);
    *h=y1;
}

```

```
void rear_x2(int y1,int z1)
```

```
{
    int q;
    q=-1;
    do { q++;
        oku(q,y1,z1);
    } while (q<(r-1) && *h==0);
    if (*h==0) q=-1;
    *h=q;
}
```

```
void rear_i2(int z2)
```

```
{
    int y1,q,top1;
    y1=-1;
    do { q=-1;top1=0; y1++;
        do { q++;
            oku(q,y1,z2);
            top1+=*h;
        } while (q<(r-1) && *h==0);
    } while (top1!=0);
    *h=y1;
}
```

```
void rear_z2(int x1,int y1)
```

```
{
    int q;
    q=-1;
    do { q++;
        oku(x1,y1,q);
    } while (q<(r-1) && *h==0);
    if (*h==0) q=-1;
    *h=q;
}
```

```
void new_call_called(int cag)
```

```
{
    int i,gir;
    i=-1;randomize();
    while (i== -1)
    { gir=random(n*r)+1;
        do i++;
        while (i<(cag-1) && giris[i]!=gir);
        if (gir==giris[i]) i=-1;
    }
    giris[cag]=gir;
}
```

```

i=-1; randomize();
while (i==1)
    { gir=random(n*r)+1;
      do i++;
      while (i<(cag-1) && cikis[i]!=gir);
      if (gir==cikis[i]) i=-1;
    }
cikis[cag]=gir;
}

void route(int cag)
{
    clock_t start, end;
    char *str,si[4],string[4],star[6],en[6]; double num;
    int x,y,jj,l,i,q,tp1,tp2,x2,y2,dec,sign,ndig = 6,ii[m],oo[m];
        x=(giris[cag]-1)/n;
        y=(cikis[cag]-1)/n;
    jj=-1;
    randomize();
    i=random(m);
    do {
        if (i<=-1 || i>= m) i=-1;

        i++;jj++;tp1=0;q=-1;
        do { q++;
            oku(q,i,y);
            tp1+=*h;
        } while (q<(r-1) && tp1==0);

        tp2=1;q=-1;
        while (tp1==0)
            { tp1=1;tp2=0;
              do { q++;
                  oku(x,i,q);
                  tp2+=*h;
              } while (q<(r-1) && tp2==0);
            }
        } while (jj<(m-1) && tp2!=0);
    oku(x,i,y);tp1=0;
    if (tp2==0 && *h==0)
        { baglasma(x,i,y);
          start=0; end=0;
        }
    // rearrange yapilacaktır
    else { i=0;x2=0;y2=0;l=0;
          for (i=0; i<=m-1; i++) { ii[i]=0; oo[i]=0; }

```


E 12. TEKLİF EDİLEN ALGORİTMANIN C++ PROGRAMI

```

#include <alloc.h>
#include <dos.h>
#include <conio.h>
#include <stdlib.h>
#include <time.h>
#include <stdio.h>
#include <math.h>

void dolu_bos(int,int);
void baglasma(int,int,int);
void baglimi(int,int,int);
void hangi_x_y(int);
void hangi_x_yk(int,int);
void hangi_z_y(int);
void hangi_z_yk(int,int);
void hangi_x_yz(int,int);
void hangi_xy_z(int,int);
void cozul(int,int,int);
void kurulu_orta(int,int);
int huge *h;
unsigned int huge *p1;
unsigned int huge *p2;
unsigned int huge *giris;
unsigned int huge *cikis;
unsigned long int kurulu[9000];
int const n=300, r=30, m=300, kk=9000-4*40;
FILE *di,*estab;

void main()
{
void sifirla(void);
void call();
void called(void);
void new_call_called(int);
void route(int);
FILE *call_f,*called_f;
    int f,x,y,x2,i,j,k,z,z2,top1,repeat=40;
    char *string1;
    ldiv_t xy,xy1;
clrscr();
p1=(unsigned int huge *) farcalloc(580,sizeof(int));
p2=(unsigned int huge *) farcalloc(580,sizeof(int));
giris=(unsigned huge *) farcalloc(9000,sizeof(int));
cikis=(unsigned huge *) farcalloc(9000,sizeof(int));

```

```

h=(int huge *) farcalloc(1,sizeof(int));
sifirla();
printf("cagrilarin uretilmesi\n");
// call();
        call_f=fopen("cagri40.dat", "rb");
        fread(giris,sizeof(int),9000,call_f);
        fclose (call_f);

        printf("cagrilanlarin uretilmesi\n");
// called();
        called_f=fopen("istek40.dat", "rb");
        fread(cikis,sizeof(int),9000,called_f);
        fclose(called_f);

j=-1;
printf("random baglantilar olusturuluyor\n");
do { j++;
gotoxy(30,10);
printf("kurulan=%d ",j);
        x=*(giris+j)-1)/n;
        z=*(cikis+j)-1)/n;

        randomize();
        y=random(m);
        if (y<=-1 || y>=m) y=0;

        baglimi(x,y,z); y=*h;
        if ( y > -1 ) baglasma(x,y,z);
        else { z2=0;x2=0;
                do { hangi_x_y(x); y=*h;           //ortakat bulundu
                    hangi_x_yz(y,z); x2=*h;         //x2-y-z de x2 bul
                    if (x2==1) baglasma(x,y,z);
                    else { cozul(x2,y,z);
                            baglasma(x,y,z);
                            x=x2;
                    }
                    hangi_z_y(z); y=*h;           // x2-y2-z2 de y2 bulunuyor
                    hangi_xy_z(x,y);z2=*h;         //x2-y2-z2 de z2 bulunuyor

                    if (z2==1) baglasma(x,y,z);
                    else { cozul(x,y,z2);
                            baglasma(x,y,z);
                            z=z2;
                    }
                }
        } while (x2>-1 && z2>-1);
} while (j<kk-1);

```

```

    di=fopen("kalanbag.dat", "w");

    for(f=0; f<4; f++)
    {
        gotoxy(30,10);
        printf("random 40 cagri kuruluyor ** %2d**  bekleyiniz\n",f);
        for(k=0; k<=repeat-1; k++)
        {
            j++;
            new_call_called(j);
            route(j);
        } //for k
        fprintf(di,"-----% \n",f+1);
    } //for f

fclose(di);
farfree(p1);
farfree(p2);
farfree(giris);
farfree(cikis);
farfree(h);

} // main
/* _____ */

void sifirla()
{
    int i,k;
    unsigned int s;
    div_t xy;
    for (i=0; i<=n*r-1; i++)  kurulu[i]=0;

        for (i=0; i<30; i++)
        for (k=300; k<304; k++)
        {
            xy = div(k,16);
            s=(unsigned int) pow(2,xy.rem);
            *(p1+i*19+xy.quot)=s;
        }

    for (i=0; i<30; i++)
    for (k=300; k<304; k++)
    {
        xy = div(k,16);
        s=(unsigned int)pow(2,xy.rem);
        *(p2+i*19+xy.quot)=s;
    }
}

```

```

void call()
{
    int i,j,gir;
    for (j=0; j<=(kk-1); j++)
        { gotoxy(30,5); printf("j= %d",j);
          i=-1;
          while (i==1)
              { gir=random(n*r)+1;
                do i++;
                while (i<j && (*(giris+i))!=gir);
                if (gir==(*(giris+i))) i=-1;
              }
          *(giris+i)=gir;
        }
}

```

```

void called()
{
    int i,j,gir;
    randomize();
    for (j=0; j<=(kk-1); j++)
        { gotoxy(30,5);printf("j= %d",j);
          i=-1;
          while (i==1)
              { gir=random(n*r)+1;
                do i++;
                while (i<j && (*(cikis+i))!=gir);
                if (gir==(*(cikis+i))) i=-1;
              }
          *(cikis+i)=gir;
        }
}

```

```

void dolu_bos(int x1,int z1)
{
    unsigned int s,w;
    int k,u,v;
    u=-1;x1=x1*19; z1=z1*19;
    do{ u++;
        s=(~(*(p1+x1+u))) & (~(*(p2+z1+u)));
    }while (u<18 && s==0);
    if (s==0) *h=-1; //baglanti kurulamaz
    else { k=-1;
          do { k++;v=(unsigned int) pow(2,k);
              w=s & v;
          } while (w == 0);
          *h=u*16+k; //baglanti kurulacak ortakat g.s.
        }
}

```

```

    }
}

```

```

void baglimi(int x1,int y1, int z1)
{ unsigned int s,k,v;
  div_t xy;

  xy = div(y1,16);
  s=(unsigned int)pow(2,xy.rem);
  k= *(p1+x1*19+xy.quot) & s;
  v= *(p2+z1*19+xy.quot) & s;
  if(kurulu[r*y1+z1] == 0 && k==0 && v==0) *h=y1;
  else *h=-1;
}

```

```

void baglasma(int x1,int y1, int z1)
{ unsigned int s;
  long int u;
  div_t xy;
  xy = div(y1,16);
  s=(unsigned int)pow(2,xy.rem);
  *(p1+x1*19+xy.quot)|=s;
  *(p2+z1*19+xy.quot)|=s;

  u=1000000;
  u=u*(x1+1)+(y1+1)*100+(z1+1);
  kurulu[r*y1+z1]=u;
}

```

```

void cozul(int x1,int y1,int z1)
{ unsigned int s;
  div_t xy;
  xy = div(y1,16);
  s=(unsigned int)pow(2,xy.rem);
  *(p1+x1*19+xy.quot)^=s;
  *(p2+z1*19+xy.quot)^=s;
  kurulu[r*y1+z1]=0;
}

```

```

void hangi_x_y(int x1)
{ int y1,q;
  unsigned int s,k,v;
  y1=-1;
  do { y1++;
    s=~(*(p1+x1*19+y1));

```

```

    } while(s<=0 && y1<18);
    q=-1;k=0;
    while ( q<15 && s>0 && k==0 )
    { q++;
      v=(unsigned int) pow(2,q);
      k=s & v;
    }
    *h=y1*16+q;
}

```

```

void hangi_x_yk(int x1,int y1)
{ int q;
  unsigned int s,k,v;
  div_t xy;
  xy = div(y1,16);
  s=~(*(p1+x1*19+xy.quot));
  q=xy.rem; k=0; y1=xy.quot;
  while ( q<15 && s>0 && k==0)
  { q++;v=(unsigned int) pow(2,q);
    k=s & v;
  }
  while (k==0 && y1<18)
  {
    do { y1++;
        s=~(*(p1+x1*19+y1));
      } while(s<=0 && y1<18);
    q=-1;k=0;
    while ( q<15 && s>0 && k==0)
    { q++; v=(unsigned int) pow(2,q);
      k=s & v;
    }
  }
  if (k!=0 && q<=15 && y1<=18) *h=y1*16+q;
  else *h=-1;
}

```

```

void kurulu_orta(int x1, int z1)
{ int y1;
  long int u;
  y1=-1;
  do{ y1++;
      u=100000;
      u=u*(x1+1)+(y1+1)*100+(z1+1);
    } while (kurulu[r*y1+z1] != u && y1<m-1);
  if (y1>m-1) y1=-1;
}

```

```

    *h=y1;
}

```

```

void hangi_z_y(int z1)
{ int y1,q;
  unsigned int s,k,v;
  y1=-1;
  do { y1++;
      s=~*(p2+z1*19+y1));
    } while(s<=0 && y1<18);
  q=-1;k=0;
  while ( q<15 && s>0 && k==0)
  { q++;
    v=(unsigned int) pow(2,q);
    k=s & v;
  }
  *h=y1*16+q;
}

```

```

void hangi_z_yk(int y1,int z1)
{ int q;
  unsigned int s,k,v;
  div_t xy;
  xy = div(y1,16);
  s=~*(p2+z1*19+xy.quot));
  q=xy.rem; k=0;y1=xy.quot;
  while ( q<15 && s>0 && k==0)
  { q++;v=(unsigned int) pow(2,q);
    k=s & v;
  }
  while (k==0 && y1<18)
  {
    do { y1++;
        s=~*(p2+z1*19+y1));
      } while(s<=0 && y1<18);
    q=-1;k=0;
    while ( q<15 && s>0 && k==0)
    { q++;v=(unsigned int) pow(2,q);
      k=s & v;
    }
  }
  if (k!=0 && q<=15 && y1<=18) *h=y1*16+q;
  else *h=-1;
}

```

```

void hangi_x_yz(int y1, int z1)
{ ldiv_t xy;
  xy=ldiv(kurulu[r*y1+z1],100000);
  *h=xy.quot-1;
}

```

```

void hangi_xy_z(int x1, int y1)
{ int q;
long int u;

```

```

  ldiv_t xy;
  q=-1; u=1000;
  u=u*(x1+1)+(y1+1);
  do { q++;
    xy=ldiv(kurulu[r*y1+q],100);
  } while (u!=xy.quot && q<r-1);
  if (u==xy.quot) *h=xy.rem-1;
  else *h=-1;
}

```

```

void new_call_called(int cag)
{ int i,gir;
  i=-1; *(giris+cag)=0;
  gotoxy(30,14);
  printf("istek uretiliyor ");
  randomize();
  while (i==1)
  { gir=random(n*r)+1;
    do i++;
    while (i<cag-1 && *(giris+i)!=gir);
    if (gir==*(giris+i)) i=-1;
  }
  *(giris+cag)=gir;
  randomize();
  i=-1; *(cikis+cag)=0;
  while (i==1)
  { gir=random(n*r)+1;
    do i++;
    while (i<cag-1 && *(cikis+i)!=gir);
    if (gir==*(cikis+i)) i=-1;
  }
  *(cikis+cag)=gir;
}

```

```

void route(int cag)

```

```

{
    clock_t start, end;
    char *str,si[4],string[4],star[6],en[6]; double num;
    int x,z,y,top1,z2,x2,y2,dec,sign,ndig = 6;;

        x=*(giris+cag)-1)/n;
        z=*(cikis+cag)-1)/n;
        randomize();
        y=random(m);
        if (y<=-1 || y>=m) y=0;

        baglimi(x,y,z); y=*h; top1=0;
    if ( y > -1 ){ baglasma(x,y,z);start=0; end=0;}
    else { x2=0;z2=0;
        start=clock();
        do { hangi_x_y(x); y=*h; //ortakat bulundu
            hangi_x_yz(y,z); x2=*h; //x2-y-z de x2 bul

            if (x2==1) baglasma(x,y,z);
            else { y2=0;
                while (y2!=-1 && x2!=-1 && y<m-1)
                {
                    hangi_x_yk(x,y);y2=*h; //x e uygun ikinci bir ortakat bulundu
                    if(y2!=-1) { hangi_x_yz(y2,z);
                        x2=*h; y=y2;
                    }
                }
            if (x2==1 ) baglasma(x,y,z);
            else { cozul(x2,y,z);
                baglasma(x,y,z);
                x=x2;top1++;
                hangi_z_y(z); y=*h; // x2-y2-z2 de y2 bulunuyor
                hangi_xy_z(x,y);z2=*h; //x2-y2-z2 de z2 bulunuyor
            if (z2==1) baglasma(x,y,z);
            else { y2=0;
                while (y2!=-1 && z2!=-1 && y<m-1)
                {
                    hangi_z_yk(y,z);y2=*h; //z e uygun ikinci bir ortakat bulundu
                    if(y2!=-1) { hangi_xy_z(x,y2);
                        z2=*h; y=y2;
                    }
                }
            if (z2==1) baglasma(x,y,z);
            else { cozul(x,y,z2);
                baglasma(x,y,z);
                z=z2;top1++;
            }
        }
    }
}

```



ÖZGEÇMİŞ

SITKI ÖZTÜRK

Kocaeli Üniversitesi Mühendislik Fakültesi

Elektronik ve Haberleşme Müh. Bölümü

İZMİT

KİŞİSEL BİLGİLER

Doğum Tarihi: 15 Haziran 1959

Doğum Yeri : Giresun

Medeni Hali : Evli, iki çocuk sahibi

ÖĞRENİM

ORTA: Bursa Atatürk Lisesi (Orta kısım)

Bursa Tophane Teknik Lisesi

YÜKSEK: Yıldız Teknik Üniversitesi (Lisans) 1978-1982

Elektronik ve Haberleşme Müh. Bölümü

Yıldız Teknik Üniversitesi (Yük. Lisans) 1985-1987

Elektronik ve Haberleşme Müh. Bölümü

Yıldız Teknik Üniversitesi (Doktora, halen 1988-
devam ediyor)

Fen Bilimleri Enstitüsü

Elektronik ve Haberleşme Müh. Bölümü

ÇALIŞMALAR

Teletaş'ta 2-MBit Sayısal Radyo'nun Temel

Bandının Gerçekleştirilmesi (Yük. Lisans tezi)

Santrallarda Oluşan Blokajın Düzeltilerek

Sistemin Maksimum Verimle Çalışmasının

Sağlanması (Doktora tezi. Halen devam ediyor.)

LİSANS BİLGİSİ:

İngilizce (iyi düzeyde)