ÇOK DAR BOĞAZLI ATM AĞLARI İÇİN AKIŞ KONTROLÜ ALGORİTMALARININ GELİŞTİRİLMESİ

İnci MUNYAS ELMAS

Yüksek Lisans Tezi

Fen Bilimleri Enstitüsü
Elektrik - Elektronik Mühendisliği Anabilim Dalı

Aralık – 2003

JÜRİ VE ENSTİTÜ ONAYI

İnci MUNYAS ELMAS'ın "Çok Dar Boğazlı ATM Ağları İçin Akış Kontrolü Algoritmalarının Geliştirilmesi" başlıklı Elektrik–Elektronik Mühendisliği Anabilim Dalındaki Yüksek Lisans tezi 23/12/2003 tarihinde, aşağıdaki jüri tarafından Anadolu Üniversitesi Lisansüstü Eğitim–Öğretim ve Sınav Yönetmeliğinin ilgili maddeleri uyarınca değerlendirilerek kabul edilmiştir.

Adı-Soyadı	İmza
Üye (Tez Danışmanı) : Prof. Dr. Altuğ İFTAR	
Üye : Prof. Dr. Hüseyin Akçay	
Üye : Yrd. Doç. Dr. Osman Parlaktuna	

Anadolu Üniversitesi Fen Bilimleri Enstitüsü Yönetim Kurulu'nun 07.01.2004 tarih ve 1/2 sayılı kararıyla onaylanmıştır.

Enstitü Yöneticisi
Prof. Dr. Ömer ÖZER
Fen Bilimleri Enstitüsü
Müdürü

ÖZET

Yüksek Lisans Tezi

ÇOK DAR BOĞAZLI ATM AĞLARI İÇİN AKIŞ KONTROLÜ ALGORİTMALARININ GELİŞTİRİLMESİ

İNCİ MUNYAS ELMAS

Anadolu Üniversitesi
Fen Bilimleri Enstitüsü
Elektrik-Elektronik Mühendisliği Anabilim Dalı

Danışman: Prof. Dr. Altuğ İFTAR
2003, 110 sayfa

Bu tezde, başka bir çalışmada tek dar boğazlı veri iletişim ağlarında akış kontrolü için tasarlanmış olan $\mathcal{H}_\infty$ denetleyicinin, ATM ağlarında çok dar boğazlı durum için gerçeklemesi yapılmıştır. Gerçeklemesi yapılan denetleyiciler, iletim oranı komutlarını, anahtarlardaki kuyruk uzunluğu bilgilerini kullanarak hesaplamaktadırlar. Algoritmanın gerçek ağlara uygulandığında ne kadar verimli çalışacağını görmek için yapılan benzetim çalışmalarında MATLAB / SIMULINK paket programı kullanılmıştır. Aynı çalışmalar, farklı parametre ve ağ koşulları için de tekrarlanmış ve elde edilen sonuçlar karşılaştırılmıştır.

Anahtar Kelimeler: ATM Ağları, Sonsuz Boyutlu Sistemler, Gürbüz Kontrol, Akış Kontrolü

ABSTRACT

Master of Science Thesis

DEVELOPING FLOW CONTROL ALGORITHMS FOR ATM NETWORKS WITH MULTIPLE BOTTLENECKS

İNCİ MUNYAS ELMAS

Anadolu University
Graduate School of Natural and Applied Sciences
Electrical and Electronics Engineering Program

Supervisor: Prof. Dr. Altuğ İFTAR
2003, 110 pages

In this thesis, the implementation of a $\mathcal{H}_\infty$ controller, designed in another work for the flow control in communication networks with a single bottleneck, is considered for ATM networks with multiple bottlenecks. The controller calculates the data sending rates by using the information about the queue lengths at the switches. In the simulation studies realised to observe the performance of the system in real ATM networks, MATLAB / SIMULINK is used. Same studies are also realised for different parameter values and network conditions and the results are compared.

Keywords: ATM Networks, Infinite Dimensional Systems, Robust Control, Flow Control

TEŞEKKÜR

Çalışmalarımda beni yönlediren ve benden yardımlarını esirgemeyen, asistanı olarak kendisiyle çalışmaktan büyük onur duyduğum Prof. Dr. Altuğ İFTAR'a teşekkür ederim. Ayrıca tezim süresince yardım ve önerilerini esirgemeyen sayın Prof. Dr. Hitay ÖZBAY'a ve sayın Prof. Dr. Hüseyin AKÇAY'a da teşekkür ederim. Çalışmalarım sırasında, benden manevi desteklerini esirgemeyen aileme, Yrd. Doç. Dr. Osman PARLAKTUNA'ya ve arkadaşlarıma da teşekkür ederim.

İÇİNDEKİLER

ŞEKİLLER DİZİNİ

ÇİZELGELER DİZİNİ

SEMBOLLER DİZİNİ

$\mathbb{R}$:	Gerçek Sayılar.
$\mathbb{R}_+$:	Negatif Olmayan Gerçek Sayılar, $\{x \in \mathbb{R} : x \geq 0\} = [0, \infty\}$.
$\mathbb{C}$:	Kompleks Düzlem.
$\mathbb{C}_+$:	$\mathbb{C}$'de Açık Sağ Yarı Düzlem, $\{s \in \mathbb{C} : Re(s) > 0\}$.
$\bar{\mathbb{C}}_+$:	$\mathbb{C}$'de Kapalı Sağ Yarı Düzlem, $\{s \in \mathbb{C} : Re(s) \geq 0\}$.
$j\mathbb{R}$:	Sanal Eksen, $\{s \in \mathbb{C} : Re(s) = 0\}$.
$ess\ \sup$:	Lebesgue Ölçüsüne göre temel supremum.
$\|\cdot\|_2$:	$\mathcal{L}_2$ Lebesgue uzayında tanımlı norm.
$\|\cdot\|_\infty$:	$\mathcal{L}_\infty$ Lebesgue uzayında tanımlı norm.
$\mathcal{H}_\infty(\mathbb{C}_+)$:	$\mathbb{C}_+$ kümesinde sınırlı ve analitik olan $\mathcal{L}_\infty(j\mathbb{R})$ fonksiyonlarının Hardy Uzayı.

1 GİRİŞ

Günümüzde, gelişen İnternet teknolojisinin tüm dünyayı bir bilgisayar ağına taşımış olması veri iletişim ağlarının önemini daha da arttırmıştır. Bu teknolojik olanaklardan yararlanan kullanıcı sayısının her geçen gün biraz daha arttığı göz önüne alınırsa, şu andaki iletim kapasitesinde yetersizlikler olacağı açıktır. Ancak, yazılım ve donanım birlikteliği ile sağlanan veri iletiminde, tüm dünya dikkate alındığında kapasite arttırımı için yeni donanımlar eklemek ya da eski donanımları değiştirmek oldukça zahmetli ve yüksek maliyetli bir iştir. Bunun yerine yazılımla, eldeki donanımlardan en üst düzeyde ve verimli şekilde yararlanmanın yolları aranmalıdır. Bu amaca ulaşmak için de veri iletişim ağlarında, tıkanıklık ve akış kontrolü algoritmaları uygulanır.

Bir veri iletişim ağında, veri iletiminin nasıl gerçekleştirileceği protokollerle belirlenir. Ağdaki bilgi akışını ve tıkanıklığı kontrol etmek için kullanılacak denetleyiciler de kullanılan protokole göre tasarlanır. Bilgi akışı ve tıkanıklık kontrolünün temel amacı, performans hedeflerine ulaşabilmek için ağ kaynaklarının verimli kullanılmasını sağlamaktır. Diğer bir amaç da ağ kaynaklarının, hafıza elemanları, bant genişliği gibi, en verimli şekilde kullanılmasını sağlamaktır. Bilgi akışının kontrolünü, ya ağın kendisi ya da kullanıcı yapar. Kontrolün ağ tarafından yapıldığı protokole örnek olarak asenkron iletim modu (Asynchronous Transfer Mode, ATM); kullanıcı tarafından yapıldığı protokole örnek olarak da İnternette kullanılan TCP/IP verilebilir [1, 2].

Bu çalışmanın amacı, birden fazla tıkalı düğüm içeren, veri iletimi için ATM protokolünün kullanıldığı ağlarda akış ve tıkanıklık kontrolü algoritmalarının geliştirilmesidir. Tıkanıklık kontrolünde, açık döngü kontrol ve kapalı döngü kontrol olmak üzere iki farklı kontrol stratejisi uygulanabilir. Açık döngü kontrolde, her bağlantının kullanabileceği bant genişliğinin sınırı,

ağ ve kullanıcının bağlantı kurulmadan önce üzerinde anlaştığı parametreler ile belirlenir. Bağlantı sağlandıktan sonra bu parametrelerin sağlanacağı garanti edilir ve değerleri bağlantı süresince değişmez. Ağ kaynakları yetersiz olduğunda, yeni bağlantı istemleri reddedilir. Dolayısıyla, açık döngü kontrol yöntemi ağ donanımında beklenmedik değişiklikler olmadığı durumda *tıkanıklığı önleyici* bir yaklaşımdır. Ancak bu yöntemin önemli bir dezavantajı vardır: Ağda kullanılmayan bant genişliği olsa bile iletim yapan kaynaklara daha fazla bant genişliği ayrılmaz. Bu nedenle, bu yöntem, kapalı döngü kontrol yaklaşımının tersine ani yığılmaların yaşandığı veri alış verişlerinin kontrolüne uygun değildir. Kapalı döngü kontrol yaklaşımı, kaynakların iletim oranlarını dinamik olarak kontrol ettiği için özellikle ATM ağlarındaki ABR servisi uygulamaları için çok uygundur.

Akış kontrolü algoritmaları oran–tabanlı ya da pencere (kredi)–tabanlı olarak tasarlanabilir. Oran–tabanlı akış kontrolünde, anahtarlama düğümlerindeki denetleyiciler, kaynaklara hangi oranla iletim yapmaları gerektiğini bildirirler. Pencere–tabanlı akış kontrolünde ise geri besleme bilgisi pencere boyutudur. Pencere–tabanlı yaklaşımda, her linkte, farklı bağlantılar için birbirinden bağımsız akış kontrolleri uygulanır ve her bağlantı, her linkte yapacağı iletim için önceden hafıza rezerve etmek zorundadır. Bir bağlantının veri iletimine devam edebilmesi için bir sonraki düğümden kredi alabilmesi gerekir. Bu yaklaşımın avantajı hücre kaybına yol açmamasıdır (teorik olarak); çünkü, kredi alınamaması durumunda kaynaklar veri gönderemez. Oran–tabanlı yaklaşımda ise hücre kaybı olasılığı pencere–tabanlı yaklaşıma göre daha fazladır ve gidiş–dönüş gecikmelerinden daha fazla etkilenir [1, 3].

Oran–tabanlı yaklaşım, ATM ağlarında daha çok kullanım alanı bulur. Bu çalışmada tıkanıklık ve akış kontrolü algoritmaları geliştirilirken kapalı döngü kontrol stratejisi ve oran–tabanlı yaklaşım kullanılacaktır.

Literatürde, veri iletişim ağlarında tıkanıklık ve akış kontrolü üzerine pek çok çalışma bulunmaktadır. Bu çalışmalardan [4]'te ortaya konan yönteme göre akış kontrolü kullanıcılar tarafından yapılır. Burada, çözüme ulaşmak için adaptif kontrol yaklaşımı kullanılmıştır. [5] çalışmasında, datagram ağlarında tek tıkalı dar boğaz durumu için bir tıkanıklık kontrolü tasarımı yapılmış ve analizi verilmiştir. Burada, hem adaptif hem de gürbüz denetleyiciler tasar-

lanmış ve benzetim çalışmaları yapılmıştır. [5] çalışmasındaki kontrol yöntemi çok dar boğazlı duruma [6]'da geliştirilmiştir. Bu yöntem, çok dar boğazlı durum için kararlılık analizinin yapıldığı ilk çalışmadır. Ancak, bu yöntemin gerçeklenebilmesi için zaman içinde dar boğazların değişmemesi, önceden belirlendiği gibi kalması gerekir ki bu da pratikte mümkün değildir. Tek tıkalı dar boğaz durumu için $\mathcal{H}_\infty$ kontrol yaklaşımı kullanılarak denetleyici tasarımı yapılan çalışmalar [7, 8]'de verilmiştir. Bu çalışmalarda, oluşturulan kontrol döngüleri için kararlılık ve performans incelemeleri de yapılmıştır. Bu çalışmalar temel alınarak, [9, 10]'da birden fazla tıkalı düğüm olduğu durum için denetleyici tasarlanmıştır. [9, 10] çalışmalarında çok tıkalı düğüm durumu için tasarımı yapılan denetleyicinin gerçeklemesi [11]'de yapılmıştır. ATM ağlarındaki ABR servisi için önerilmiş olan oran–tabanlı akış kontrolü üzerine ayrıntılı bilgiler [2]'de sunulmuştur. [3] çalışmasının ilk kısmında, ATM Forum'da şu ana kadar sunulan oran–tabanlı yaklaşımların avantaj ve dezavantajları üzerinde durulmuş, son kısmında da EPRCA algoritmasının performansı incelenmiştir. Aynı yazarların çalışmaları olan [12, 13]'te ise aynı algoritma için yatışkan durum ve geçişken durum analizleri yapılmıştır. [5]'te verilen PD denetleyici yapısı ve kontrol algoritması ATM ağlarına, tek tıkalı durum için [14]'de uyarlanmıştır. Ancak burada sistemdeki zaman gecikmeleri göz ardı edilmiştir. ATM ağlarında oran-tabanlı akış kontrolünü sağlamak için tasarladıkları denetleyicilerde temel olarak Smith Predictor kullanan çalışmalar [15, 16, 17, 18]'dir. [15]'te önerilen denetleyici yapısının PRCA ve EPRCA algoritmalarına uygulamaları arasında karşılaştırma da yapılmıştır. [16]'da ise önceden tek dar boğazlı durum için önerilen algoritma, multicast ABR trafiğine uyarlanmıştır. Yapılan benzetim çalışmalarında çok dar boğazlı durum da ele alınmıştır ve dar boğazların yer değiştirmesi gözlenebilmiştir. [17]'de tek dar boğazlı durum için tıkanıklık kontrol algoritması önerilmiştir. Ayrıca, TCP İnternet protokolünün akış performansını arttırmak için Smith Predictor kullanan bir algoritma önerilmiştir. [18]'de ATM ağları ABR servisinde tek dar boğazlı durum için Smith Predictor kullanılarak tıkanıklık kontrolü algoritması tasarlanmıştır. Burada, geri beslemede ABR trafiğinin tahmini de kullanılarak performans iyileştirmesi sağlanmıştır.

Bu tezde, veri iletişim ağlarında tek dar boğazlı durum için [8] çalışmasında

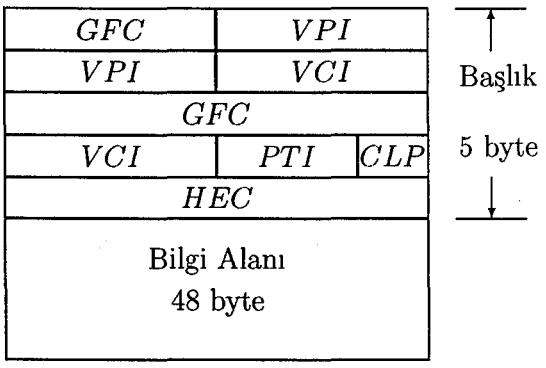
tasarlanmış $\mathcal{H}_\infty$ denetleyici, ATM ağlarında çok dar boğazlı durum için gerçeklenmiştir. Çok dar boğazlı durum için $\mathcal{H}_\infty$ denetleyici tasarımı [9, 10] çalışmalarında, bu denetleyicilerin gerçeklemesi ise [11]'de verilmiştir. Bu çalışmalarda genel veri iletişim ağları ele alınmıştır. Bu da belli bir protokole bağlı kalınmadığı anlamına gelir. Bu çalışmada ise özel olarak ATM ağları üzerinde durulmaktadır. ATM protokolünün temel kontrol yapısı göz önüne alındığında, bu ağlarda çok dar boğazlı durum için $\mathcal{H}_\infty$ denetleyici tasarımı yapılamamıştır. Bu nedenle, sistem üzerinde bazı kabullenmeler yapılarak [8] çalışmasında tasarlanmış $\mathcal{H}_\infty$ denetleyici ATM protokolü kontrol yapısı içinde kullanılmıştır. Gerçeklemesi yapılan denetleyiciler, anahtarlardaki kuyruk uzunluğu bilgilerini kullanarak iletim oranı komutlarını hesaplamaktadırlar. Sistemin gerçek ağlarda nasıl bir performans göstereceğini belirlemek için yapılan benzetim çalışmalarında MATLAB / SIMULINK paket programı kullanılmıştır. Bu çalışmalar, farklı parametre değerleri ve ağ koşulları için tekrarlanmış ve elde edilen sonuçlar karşılaştırılmıştır.

2 ATM AĞLARI

Asenkron İletim Modu (Asynchronous Transfer Mode, ATM) hem sabit bant genişlikli hem de yüksek oranda değişen bant genişliğine sahip servisleri destekleyebilen ilk anahtarlama teknolojisidir. Bu nedenle, ATM, Tümleştirilmiş Genişbantlı Sayısal Servis Ağları (Broadband Integrated Services Digital Networks, B–ISDN) için standart iletim modu olarak seçilmiştir. Buradaki temel amaç, her bir bilgi türünün iletimini en iyi şekilde sağlamaktan öte, farklı bilgi türlerinin birarada olduğu, görüntülü telefon, yüksek–hızlı–veri, video konferans gibi tümleştirilmiş genişbantlı servisleri kullanıcılara iyi bir şekilde sunmaktır [19, 20].

ATM, veri paketlerini sabit uzunluklu, 53 byte, hücrelere bölerek gönderir. Böylesi sabit uzunluklu hücrelerin kullanılmasının getirdiği en önemli avantaj, her bir hücrenin iletim süresinin sabit ve kısa olmasıdır. Bu kısa iletim süresi, ortalama gidiş–dönüş gecikmesi ve bu gecikmelerdeki değişimlerin daha düşük değerler almasını sağlar. Bu parametrelerin değerlerinin düşük olması özellikle paket tabanlı ses ve görüntü iletimi için önem taşır. Bunun yanı sıra, ATM'de kullanılan paket anahtarlaması ve gelişmiş trafik yönetimi algoritmaları, bu teknolojinin veri iletimi için de oldukça uygun olmasını sağlar. ATM'de kullanılan hücre yapısı Şekil 2.1'de görülmektedir [1].

Her bir ATM hücresi, uzunluğu 48 byte olan bilgi alanı ve uzunluğu 5 byte olan başlık alanından oluşur. Başlık alanı, hücrelerin ait oldukları sanal kanalı belirler ve böylece bilgi paketinin ağda doğru bir şekilde yönlendirilmesi sağlanmış olur. Temel ATM hücre yapısında VPI (Virtual Path Identifier), sanal yol belirleyicisi; VCI (Virtual Channel Identifier), sanal kanal belirleyicisi; PTI, (Payload Type Identifier) taşınan bilginin ne bilgisi olduğununun belirleyicisidir ve fiziksel bir iletim ortamında ATM hücresinin tanınmasını sağlarlar. GFC Genel Akış Kontrolu (Generic Flow Control), ağ ile kulanıcılar arasındaki ATM bağlantılarındaki trafik akışını kontrol eden bir



Şekil 2.1: Kullanıcı–Ağ Arayüzündeki (User Network Interface, UNI) ATM hücre yapısı

mekanizmadır. ATM hücresinin başlık alanında bulunan Hücre Kaybı Önceliği (Cell Loss Priority) CLP, biti farklı önceliklere sahip trafik akışları üretir. Ağın performansının yüksek öncelikli hücreler açısından düşmemesi için, gerekli görülen durumlarda, düşük öncelikli hücreler diziden çıkarılır. Başlık Hata Kontrolü (Header Error Control) HEC, hata kontrolünün bilgi alanı yerine başlık alanında yapılmasını sağlar. Böylece, daha çok sayıda byte bilgi alanı için ayrılmış olur.

ATM bağlantı tabanlı olarak çalışır. Buna göre, bir hedefe veri göndermek isteyen bir kaynak, bağlantı isteğini ağa bildirir. Uygulanan bağlantı izin (Call Admission Control, CAC) algoritmaları ile kaynağın istediği servis kalitesinin sağlanıp sağlanamayacağı kontrol edilir. Gerekli şartlar sağlanırsa bağlantının kurulmasına izin verilir. Bu durumda kaynakla hedef arasında bir *sanal devre (Virtual Circuit, VC)* kurulur. Kaynaktan hedefe gidecek paketler, hücreler, sanal kanal bazında korunur.

ATM bağlantıları sağlandığında, kaynaklar, *Servis Kalitesi (Quality of Service, QoS)* parametrelerini tek yönlü olarak ağa bildirir ve bu parametrelerin uzlaşılan değerlerinin sağlanması ağ tarafından garanti edilir. Bu parametrelerin verimliliğini düşürebilecek, aynı ağ kaynaklarını paylaşan başka bağlantı olmadığından da emin olunmalıdır. Böylece, QoS parametrelerindeki düşüşün yaratacağı problemler, gerçekleştirilen trafik kontrolü ile daha ortaya çıkmadan önlenmiş olur. Bazı QoS parametreleri Çizelge 2.1'de görülmektedir [1, 20]. ATM anahtarlama sistemlerinin performansını belirleyen beş parametre vardır:

Parametre	*Anlam*
En Yüksek Hücre Oranı (Peak Cell Rate, PCR)	En yüksek iletim hızı
En Düşük Hücre Oranı (Minimum Cell Rate, MCR)	En düşük iletim hızı
Gecikme Değişimi Toleransı (Cell Delay Variation Tolerance, CDVT)	Kabul edilebilecek en yüksek titreşim değeri
Hücre Kaybı Oranı (Cell Loss Ratio, CLR)	Kaybolan ya da çok geç iletilen hücrelerin oranı
Kayıp RM Hücresi Sayısı (Missing RM–cell Count, CRM)	Yanlış hedefe iletilen RM–hücrelerinin oranı

Çizelge 2.1: Bazı QoS parametreleri

1. Geçirgenlik (Throughput): Hücrelerin anahtarı terk ettikleri orandır ve birim zamanda çıkan hücre sayısı olarak birimlendirilir. Bu parametre, ATM anahtarının teknolojisine ve boyutuna bağlıdır.

2. Bağlantı bloke etme olasılığı (Connection Blocking Probability): Kurulmuş ve kurulabilecek olan bağlantıların kalitesinden emin olunmasını sağlayan, anahtarın giriş ve çıkışı arasındaki kaynakların yeterli olamama olasılığı olarak tanımlanır.

3. Hücre kaybı olasılığı (Cell Loss Probability): ATM anahtarlarında, anahtardaki kuyruğun kaldırabileceğinden çok hücre bu kuyruğa girmeye çalışırsa bazı hücreler kaybedilir. Anahtarın güvenilirliğinden emin olabilmek için de bu olasılık belli sınırlar arasında tutulmalıdır. Bir başka olasılık da hücrelerin ağ içinde yanlış yönlendirilmesidir. Bu şekilde, hücreler olması gerekenden farklı bir hedefe ulaşabilirler ve doğru hedef için kaybedilmiş olurlar.

4. Anahtarlama gecikmesi (Switching Delay): Bir hücrenin anahtarı geçme süresidir ve hücrenin donanım içinde iletimi sırasında oluşan iç gecikmeyi, anahtarlama gecikmesini, ve anahtarda hücre kaybının önlenmesi için hücrelerin hafızada sırada tutulması nedeniyle oluşan gecikmeyi kapsar.

5. Gecikme üzerindeki titreşimler (Jitter on the Delay): Hücre gecikmesindeki değişimler olarak da adlandırılır ve anahtardaki gecikmenin belli bir değeri aşması olasılığı olarak tanımlanır.

ATM ağları farklı uygulamaların ihtiyaç duyduğu QoS parametrelerini sağlamak için farklı servis tiplerini destekler. Şu anda desteklenen servis tipleri şunlardır : CBR, VBR, UBR ve ABR. Bu servisleri ve uygulama alanlarını biraz daha ayrıntılı incelemek gerekirse [21, 22];

- **Sabit Bit Oranı (Constant Bit Rate, CBR):** Bu servis tipi sabit (statik) bant genişliğine ihtiyaç duyulan bağlantılarda kullanılır. Bu da kaynağın iletim oranının (bant genişliğinin), bağlantı kurulu olduğu sürece hep aynı kalan En Yüksek Hücre Oranı (Peak Cell Rate, PCR) değerine eşit olmasıdır. CBR kullanıcıları bant genişliği ile ilgili isteklerini bağlantı kurulumu sırasında bildirirler. Bunun üzerine ağ, bu kullanıcılar için istedikleri kadar bant genişliğini ayırır ve böylece bu kullanıcılar için servis kalitesini sağlamış olur. Servis kalitesini hücre kaybı oranı, anahtarlama gecikmesi, gecikme değişimi gibi faktörler etkiler. Bu servisin kullanıldığı veri iletim uygulamalarına örnek olarak video konferans, interaktif sesli iletişim (telefon, vs), ses /görüntü dağıtımı (televizyon, uzaktan eğitim, vs) ve ses / görüntü geri kazanımı verilebilir.

- **Değişken Bit Oranı (Variable Bit Rate, VBR):** Bu servisin kullanıcıları, ağa, en yüksek veri iletim oran değerlerini ve sürekli bir ortalama oran değerini bildirirler. Bu kullanıcılar, yüklerini, belirttikleri parametre sınırları içinde tuttukları sürece QoS parametrelerinin sağlanması garanti edilir. Bu servis iki gruba ayrılır: Gerçek–zamanlı değişken bit oranı ve gerçek–olmayan–zamanlı bit oranı. Gerçek–zamanlı değişken bit oranı, zamana karşı hassas olan, yani, gecikme ve gecikme değişimleri konusunda çok kesin sınırlamaları olan uygulamalarda kullanılır. Kaynakların zamanla değişen oranlarda bilgi iletmesi beklenir. Telekonferans bu uygulamaya örnek olarak verilebilir. Gerçek–olmayan–zamanlı değişken bit oranı, gecikme ve gecikme değişimleri konusunda çok kesin sınırlamaları olmayan ve trafik

karakteristiği ani değişimler gösteren uygulamalarda kullanılır. Örneğin, havayolu rezervasyonu, bankacılık işlemleri gibi.

Görüldüğü gibi CBR ve VBR servisleri, gecikmelere karşı duyarlı kullanıcılarca kullanılır. Bu servisler tarafından kullanılmayan bant genişliği, gecikmeye karşı duyarlı olmayan kullanıcılar arasında dağıtılır. Bu tür kullanıcılar için de iki farklı servis tipi vardır:

- **Belirlenmemiş Bit Oranı (Unspecified Bit Rate, UBR):** Bu servis, genelde, çok kesin olarak sınırlanmış gecikme ve gecikme değişimi ya da belli bir kalitede servis istenmeyen, kritik olmayan uygulamalarda kullanılır. UBR için ağ, trafikle ilgili servis garantisi vermez. Bu servis tipi, pek çok sıradan metin, veri, görüntü iletimi, mesajlaşma, dağıtım, geri kazanım gibi veri işlemlerinde tercih edilir.

- **Mümkün Olan Bit Oranı (Available Bit Rate, ABR):** Bu servisin kullanıcıları, CBR, VBR ve UBR servisleri tarafından kullanılmadan kalan, mümkün olan bant genişliğini kullanır. Bu tip servis, ağın isteğine göre, bilgi gönderme oranlarını arttırabilen ya da azaltabilen kaynaklarca tercih edilir. Bu özellik, kaynaklara, ATM katlarındaki değişimleri kendi yararlarına kullanma olanağını verir ve böylece mümkün olan her bant genişliğinde bağlantı sürekliliği sağlanır. ABR için minimum geçirgenlik ve hücre kaybı gerekliliği vardır. Bu gerekli koşulların sağlanabilmesi için, ABR trafiği, ağın tıkanıklık durumunu da göz önüne alan kapalı döngü oran kontrol yaklaşımını kullanır. ABR'ın kullanım alanlarına örnek olarak, yerel ağlar (Local Area Network, LAN), kritik veri iletimi (savunma ile ilgili bilgiler, banka işlemleri, vs.), süper bilgisayar uygulamaları ve veri iletişimi verilebilir.

Tüm ağ yapılarında olduğu gibi ATM ağlarında da karşılaşılan en önemli problemlerden biri *ağ kaynaklarının yönetimidir.* Veri iletişim ağlarında kaynak yönetimi, ağdaki trafiğin kontrol edilmesi ile yapılabilir. Trafik kontrolü de akış kontrolü ya da tıkanıklık kontrolü algoritmaları uygulanarak yapılabilir. ATM ağları için trafik kontrolü, desteklenen tüm servis tiplerine ait akışların kontrol edilmesi anlamına gelir. Tüm bu akışların kontrolünün yapılabilmesi

için tıkanıklık kontrolü algoritmaları ile birlikte zamanlama ve bağlantı kabulü algoritmalarının da uygulanması gerekir. Uygulanacak tıkanıklık kontrolü algoritmalarının temel amaçları şu şekilde belirlenmiştir [23]:

> Trafik ve tıkanıklık kontrolü için kullanılan parametre ve prosedürlerin temel amacı, performans hedeflerine ulaşabilmek için ağı ve uç–sistemleri (kaynak ve hedef sistemler) korumaktır. Diğer bir amaç da ağ kaynaklarının kullanımının optimize edilmesidir... Tasarlanan ATM trafik ve tıkanıklık kontrolü algoritmaları, ağın en verimli şekilde kullanılmasını sağlarken, ağ ve uç–sistemlerin yapısal karmaşıklıklarını en aza indirmelidir (ATM Forum Technical Committee, Traffic Management Specification, Version 4.0, ATM Forum / 95–001328, October 1995).

Bir ağ kaynağı için kullanıcılardan gelen istek, o kaynağın kapasitesinden daha büyük olursa tıkanıklık meydana gelir. Böylesi bir tıkanıklık uzun kuyruk gecikmelerine ya da paket kayıplarına neden olur. Bu problemin çözülebilmesi için kaynakların veri akışını dinamik olarak kontrol edecek ve yukarıda verilen amaçlara ulaşılacak şekilde tasarlanan algoritmaların geçerliliğine, ATM Forum'un belirlediği şu kriterlere göre karar verilir [23];

1. Kullanıcı ve anahtarların sayısına, hızlarına ve aralarındaki uzaklığa uyum sağlayabilir olmalıdır;
2. Ağ kaynaklarını, farklı ABR kaynaklarına adil bir şekilde dağıtabilmelidir;
3. Belirsizliklere, bozucu etkilere ve ağın belirlediği şartlara uymak istemeyen kullanıcıların yaratacağı etkilere karşı gürbüz olmalıdır;
4. Sistem yatışkan duruma ulaşmalıdır;
5. Belirli bir anahtar yapısına gereksinim duymadan her türlü ortamda gerçeklenebilir olmalıdır;
6. Ağ kaynaklarını, minimum düzeyde belleğe ihtiyaç duyulacak şekilde kullandırmalıdır.

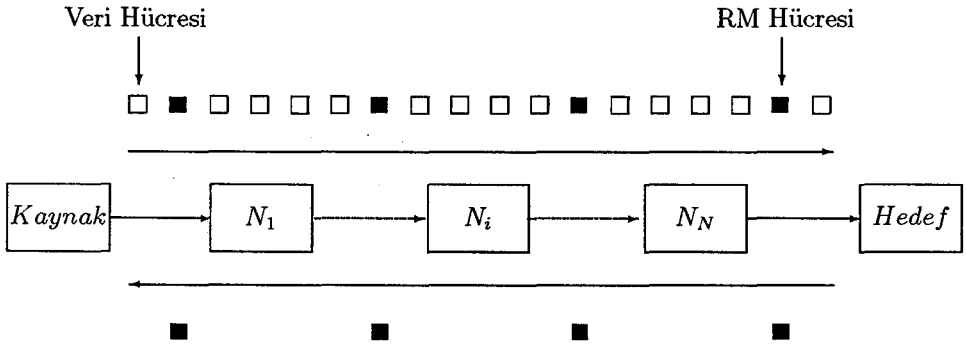
Bu kriterleri sağlayacak algoritmaların tasarımında karşılaşılan en önemli problemler tahmin edilemeyen zaman gecikmelerinin (yayılma, kuyruk, işleme gecikmeleri) nasıl ele alınacağı, trafik akışının karakteristiği (bağlantı süresinin, aktif ve kontrol edilebilen kaynakların sayısının tam olarak bilinememesi) ve gerçekleme kolaylığıdır.

ATM'in desteklediği servisler göz önüne alındığında, ancak ABR servisi kullanıcılarının veri akışlarının dinamik olarak kontrol edilebildiği görülür. Bu nedenle, bu servisi biraz daha ayrıntılı ele almak gerekir.

ABR trafik kontrol yapısının temeli, kaynakların, iletim oranlarını ağın izin verdiği orana göre ayarlayabilmesidir. Bir kaynak bir hedefe veri göndermek istediğinde, ilk olarak, ağa PCR ve MCR parametreleri için istediği değerleri bildirir. Veriler statik yönlendirmeye göre yönlendirileceği için geçecekleri anahtarlar önceden bellidir. Buna göre, eğer yol üzerindeki tüm anahtarlar kaynağın istediği MCR değerini sağlayabiliyorsa bağlantının kurulmasına izin verilir. Sonuçta, kaynağın yeni iletim oranı, verilerin gönderildiği yol üzerindeki tıkanık anahtarın belirlediği orana ayarlanır. Belirlenen bu oranlar, kaynağa, *ağ–kaynakları yönetimi (Resource Management, RM)* hücreleri kullanılarak bildirilir. Bu hücreler, kaynaklar tarafından periyodik olarak, $Nrm - 1$ veri hücresinde bir, üretilir ve bilgi hücreleriyle aynı yolu izleyerek hedefe ulaşırlar. Nrm değeri, oran tabanlı yaklaşım için 32 olarak belirlenmiştir. Pratikte, bu parametrenin alacağı değer, kontrol algoritmasının tepki zamanını ve uç sistemler ile anahtarlardaki hesaplama yükünü etkiler. RM hücreleri hedefe ulaştıktan sonra kaynağa geri gönderilirler. Bu bilgi akışı Şekil 2.2'de verilmiştir.

İleri ve geri yönlü RM hücrelerinin akışı düzenli ise ağdan kaynağa gönderilen geri besleme bilgisi de doğru bir şekilde kaynağa ulaşır. Ancak, ilk RM hücresi geri dönene kadar kaynak iletim oranları ağla uzlaşılan parametrelere göre kontrol edilir. Bu da en azından ilk gidiş–dönüş zamanı bitinceye kadar açık döngü kontrolün uygulanmış olması anlamına gelir.

Her bir RM hücresi, ATM veri hücrelerinde olduğu gibi uzunluğu 5 byte olan başlık alanına sahiptir. RM hücresinin başlık kısmında bulunan ve akış kontrolünde en çok kullanılan alanlar şunlardır:



Şekil 2.2: ABR trafik yönetimi modeli

DIR (DIRection): Hücrenin yönünü belirtir. Bu bit 0 ise RM hücresi ileri, 1 ise geri yönlüdür.

Belirli Oran (Explicit Rate, ER): Bu alan, kaynağın izin verilen iletim oranını, ACR, ağın belirlediği değere ayarlamak için kullanılır.

Anlık Hücre Oranı (Current Cell Rate, CCR): Bu değer, kaynak tarafından anlık iletim hızına ayarlanır. Bu alandaki değer, hücrenin ağ içindeki dolaşımı süresince değişmez.

Tıkanıklık Belirteci (Congestion Indication, CI): Bu bit, ağ elemanlarına, kaynaklar iletim oranlarını azaltsınlar diye oluşan tıkanıklığı bildirme olanağı sağlar.

Artış Yok (No increase, NI): Bu bit, ağ elemanlarına, kaynaklar iletim oranlarını arttırmasınlar diye olması yakın bir tıkanıklığı bildirme olanağı sağlar.

ABR kaynaklarının bir bağlantı süresince yapması gerekenler şu şekilde sıralanabilir:

1. Kaynak, hiç bir zaman hesaplanan ACR oranının üstünde bir oranla iletim yapamaz. Bu değerin alt sınırı MCR, üst sınırı ise PCR'dır ve bu sınırlar dışına çıkılamaz.

2. Bağlantı sağlandıktan sonra, ACR en çok ilk hücre oranı (Initial Cell

Rate, ICR) değerine ayarlanabilir. Gönderilecek ilk hücre RM hücresi olmalıdır.

3. CI = 1 yapılmış bir RM hücresi alındığında, ACR azaltılacak; eğer bu işlem sonunda MCR değerinin altına düşülürse ACR, MCR değerine ayarlanacaktır. Eğer bu RM hücresinde hem CI = 0 hem de NI = 0 ise ACR, PCR değerini geçmemek kaydıyla arttırılabilir. Eğer RM hücresinde NI = 1 ise ACR arttırılamaz.

4. Geri yönlü bir RM hücresi alındığında, önceki kurala göre yapılması gerekenler yapıldıktan sonra ACR, (ACR, ER) değerlerinden hangisi küçükse ona eşit olacak ve bu değer de MCR'dan az olmayacak şekilde ayarlanır.

5. Kaynak, her bir veri hücresini gönderirken *İleri Yönde Belirli Tıkanıklık Belirtme (Explicit Forward Congestion Indication, EFCI)* bitini sıfırlamalıdır.

Hedef nodların bir bağlantı süresince yapması gerekenler ise şu şekilde sıralanabilir:

1. Veri hücresi alındığında, EFCI bitinin değeri, bağlantının EFCI durumu olarak kaydedilir.

2. Bir RM hücresi alındığında, o hücre, kaynağa geri yönlü bir RM hücresi olarak geri gönderilmelidir.

3. Bir hedef, kendi içinde tıkanıklık yaşadığında, anahtarlarda olduğu gibi, ileri yönlü bir RM hücresi almadan da geri yönlü RM hücresi gönderebilir.

Anahtarlama düğümlerinin bir bağlantı süresince yapması gerekenler aşağıda verilmiştir:

1. Anahtarlar, kaynaklara geri besleme bilgilerini dört yolla bildirebilirler;

   - **EFCI işaretleme (EFCI Marking):** Tıkanıklık meydana gelen anahtar, üzerinden geçen her veri hücresinin başlık alanında bulunan EFCI bitini 1 yapabilir. Bu bilgiler, daha sonra, hedef tarafından bir RM hücresine yazılarak kaynağa geri gönderilir.

- **Göreceli oran işaretleme (Relative Rate Marking):** Tıkanıklığın meydana geldiği anahtar, ileri ya da geri yönlü RM hücrelerindeki CI ve NI bitlerini 1 yapabilir.
- **Belirli oran işaretleme (Explicit Rate Marking):** Tıkanıklığın meydana geldiği anahtar, ileri ya da geri yönlü RM hücrelerindeki ER alanında yer alan değeri azaltabilir. Bu anahtarlar, çok yoğun bir tıkanıklıkla karşı karşıya kaldıklarında kendileri de RM hücreleri üretip kaynağa gönderebilirler.
- **VS / VD:** Anahtarlar, ABR kontrol döngüsünü daha küçük parçalara bölebilmek için *sanal kaynak/sanal hedef (Virtual Source/ Virtual Destination, VS/VD)* özelliğini kullanabilirler. Bu durumda anahtar hem bir sanal kaynak hem de bir sanal hedef gibi davranır.

2. Bir anahtar ileri yönlü bir RM hücresi almadan da geri yönlü RM hücresi gönderebilir [2, 23, 14].

3 GÜRBÜZ KONTROL

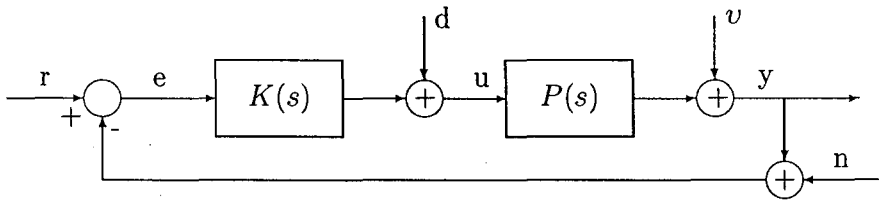
Sistemlerin kontrolü için geri besleme kullanılmasının ana nedeni sistemdeki belirsizlikleri de göz önüne alan bir tasarım yapabilmektir. Bunu gerçekleyecek kontrol sistemi tasarlanırken, öncelikle, kontrol edilecek sistemin matematiksel modeli oluşturulur. Bu tür bir model, temelde, sistemin girdileri ile çıktıları arasındaki ilişkiyi ortaya koyar. Ancak elde edilen her model, karmaşık sistemin basitleştirilmiş hali olduğundan modelleme hataları ile karşı karşıya kalınır. Ayrıca, sisteme, dışardan etkiyen bozucu sinyaller de vardır. Geri beslemeli kontrol, sistemde böyle belirsizlikler varken de kararlılığı ve belli performans kriterlerinin gerçeklenmesini sağlar.

Kontrol teorisinin temel problemi, belirsizliklerin etkidiği sistemler için gürbüz performansı sağlamaktır. 1980'li yılların başlarında, temelleri Zames tarafından atılan $\mathcal{H}_\infty$ kontrol teorisi ile gürbüz tasarım yöntemleri ortaya konulmuştur [24]. Kontrol sistemlerinin $\mathcal{H}_\infty$ optimizasyonu, temelde, bazı kapalı döngü transfer fonksiyonlarının aldıkları en büyük değerin minimize edilmesi olarak tanımlanabilir.

$\mathcal{H}_\infty$ optimizasyon probleminin çözümü, spektral ayrıştırma teknikleri [25], durum–uzayı yaklaşımı [26] ya da operatör teorisi [27] kullanılarak yapılabilir.

3.1 Gürbüz Kararlılık ve Performans

Şekil 3.1'de verilen, kapalı döngü sistem ele alınsın. Şekilde, plant $P(s)$, denetleyici ise $K(s)$ transfer fonksiyonları ile ifade edilir. Sistem girdilerinden r referans sinyalini, d hareketlendirici bozucu sinyalini, v çıktı bozucu sinyalini ve n ölçüm gürültüsünü; iç sinyallerden e ölçülen hatayı, u komut girdisini, y de plant çıktısını gösterir. Burada, ulaşılmak istenen performans hedefi, d, v ve n bozucu sinyallerinin varlığında çıktının referans girdisini takip etmesidir. Bunu sağlayacak denetleyici çıktısının büyüklüğü ise sınırlı olmalıdır. Burada,



Şekil 3.1: Geri Besleme Sistemi

dış sinyallerin enerjilerinin sınırlı olduğu kabul edilmiştir.

Tanım 1 *[27] Eğer her bir dış sinyalden her bir iç sinyale olan transfer fonksiyonlarının tümü $\mathcal{H}_\infty$'de ise Şekil 3.1'de verilen kapalı döngü sistem iç kararlıdır.*

Denetleyici tasarımında ulaşılmak istenen en önemli hedef, kapalı döngü sistemin kararlı olmasıdır. Tanım 1'de verilen kararlılık ifadesi, Şekil 3.1'deki kapalı döngü sistemde, $P(s)$'in değişmeyen (nominal) bir plant olduğu durum için geçerlidir. $P(s)$'in, üzerine etkiyen belirsizlikler dolayısıyla değişim gösteren bir plant olduğu durumda ise gürbüz kararlılık ve gürbüz performanstan söz edilir. Bu problemler şu şekilde tanımlanır;

Tanım 2 *[28] Ulaşılmak istenen performans hedefleri ve belirsizliklerin etkimesi sonucu oluşabilecek tüm plantlerin kümesi, Π, verilsin. $P \in \Pi$ oluşturulan modelde elde edilen nominal plant ve K da tasarlanan denetleyici olsun. Buradan, şu tanımlar yazılabilir;*

1. *Eğer K nominal plant P'yi iç kararlılaştırıyorsa, kapalı döngü sistem* nominal kararlıdır.

2. *Eğer K Π kümesindeki her planti iç kararlılaştırıyorsa, kapalı döngü sistem* gürbüz kararlıdır.

3. *Nominal plant P için performans hedeflerine ulaşılmışsa, kapalı döngü sistem için* nominal performans *sağlanmıştır.*

4. *Π kümesindeki her plant için performans hedeflerine ulaşılmışsa, kapalı döngü sistem için* gürbüz performans *sağlanmıştır.*

Plantdeki modelleme hatalarının gösterimi, çarpımsal, toplamsal ya da göreceli asal pertürbasyonlarla yapılabilir [27];

Çarpımsal Pertürbasyon:

$$P_\Delta(s) = P_m(s) = P(s)(1 + \Delta_m(s))$$

Toplamsal Pertürbasyon:

$$P_\Delta(s) = P_a(s) = P(s) + \Delta_a(s)$$

Göreceli asal Pertürbasyon:

$$P_\Delta(s) = P_{cf}(s) = \frac{N(s) + \Delta_N(s)}{D(s) + \Delta_D(s)}$$

Burada $P(s) = N(s)/D(s)$ nominal planti, $\Delta(s)$ ise modelleme hatalarını göstermektedir. Çoğu zaman, $\Delta(s)$'in kesin ifadesi bilinmese de, bu pertürbasyonlar için bir üst sınır bulunabilir. Bu üst sınır, frekansa bağlı bir ağırlık fonksiyonunun büyüklüğü ile verilebilir;

$$\| \Delta(j\omega) \| < | W(j\omega) | \qquad \forall \omega \in \mathbb{R}.$$

Buradan, yukarıda tanımları verilen pertürbasyonlar için

$$1 < ess \sup_\omega \left| \frac{W_m(j\omega)}{\Delta_m(j\omega)} \right| \tag{3.1}$$

$$1 < ess \sup_\omega \left| \frac{W_a(j\omega)}{\Delta_a(j\omega)} \right| \tag{3.2}$$

$$1 < ess \sup_\omega \frac{|W_{cf}(j\omega)|^2}{|\Delta_N(j\omega)|^2 + |\Delta_D(j\omega)|^2} \tag{3.3}$$

yazılabilir. $W_m(s)$, $W_a(s)$ ve $W_{cf}(s)$'in sırasıyla, çarpımsal, toplamsal ve göreceli asal pertürbasyonlar için üst sınır olan, bilinen, frekansa bağlı ağırlık fonksiyonları olduğu kabul edilsin. Çarpımsal, toplamsal ya da göreceli asal pertürbasyonlar için gürbüz kararlılık koşulları Teorem 1 ile verilir;

Teorem 1 *[27] $P = N/D$ nominal plant olmak üzere P_m, P_a ve P_{cf} plantleri (N ve D göreceli asal ve P_m, P_a ve P aynı sayıda sağ yarı düzlem kutbuna sahip) tanımlansın. W_m, W_a ve W_{cf} (3.1), (3.2) ve (3.3) koşullarını sağlayan ağırlık fonksiyonları olsun. $K(s)$ denetleyicisi nominal planti, $P(s)$, kararlılaştırsın. Buradan, $K(s)$,*

Çarpımsal Pertürbasyonlar için:

$$\left\|W_m PK(1+PK)^{-1}\right\|_\infty \leq 1,$$

Toplamsal Pertürbasyonlar için:

$$\left\|W_a K(1+PK)^{-1}\right\|_\infty \leq 1,$$

Göreceli asal Pertürbasyonlar için:

$$\left\| W_{cf} D^{-1} \begin{bmatrix} (1+PK)^{-1} \\ K(1+PK)^{-1} \end{bmatrix} \right\|_\infty \leq 1.$$

gerek ve yeter koşullarını sağlarsa planti gürbüz kararlılaştırır.

Kapalı döngü bir sistemin kararlılığının yanı sıra, göstereceği performans da önemlidir. Ulaşılmak istenen performans hedefi ise referans sinyali takip edilirken yatışkan durum hatasının belli sınırlar içinde kalmasını sağlamaktır. Burada açık döngü sistemin transfer fonksiyonunun L ile gösterildiği kabul edilsin. Referans sinyali r'den hata sinyali e'ye olan transfer fonksiyonu *duyarlılık fonksiyonu* olarak adlandırılır ve

$$S(s) := (1+L(s))^{-1}$$

ile ifade edilir.

$$T(s) := 1 - S(s) = L(s)\,(1+L(s))^{-1}$$

ise *tamamlayıcı duyarlılık fonksiyonu*dur. Sistem iç kararlı olarak kabul edildiği için S düzgün ve kararlı bir transfer fonksiyonudur. Burada, eğer ve yalnız eğer $\sup_{\|r\|_2 \leq 1} \|e\|_2 \leq 1$ sağlanırsa,

$$\|S(s)\|_\infty \leq 1$$

performans kriteri olur. Ancak bu koşulun performans kriteri olarak alınmasının iyi sonuçlar vermediği görülmüştür. Çünkü fiziksel sistemlerin ve denetleyicilerin transfer fonksiyonlarının frekans yanıtları yüksek frekanslarda düşüş gösterir. Bu da duyarlılık fonksiyonunun, bu frekanslarda alabileceği en yüksek değer olan 1'e asimptotik olarak yaklaşması anlamına gelir. Dolayısıyla, duyarlılık fonksiyonunun alçak frekanslarda düşük değere sahip olması, tepe değeri minimizasyonu yapılırken bu frekansların sonuca etki etmemesine neden olur. Ancak, bu frekanslar sistem performansı açısından oldukça önemlidir. Bu nedenle, sadece duyarlılık fonksiyonunun tepe değeri minimizasyonu yerine, frekansa bağlı bir ağırlık fonksiyonu kullanılarak yeni bir performans kriteri tanımı verilir;

$$\| WS \|_\infty = \sup_{w \in \mathbb{R}} |W(jw)S(jw)|$$

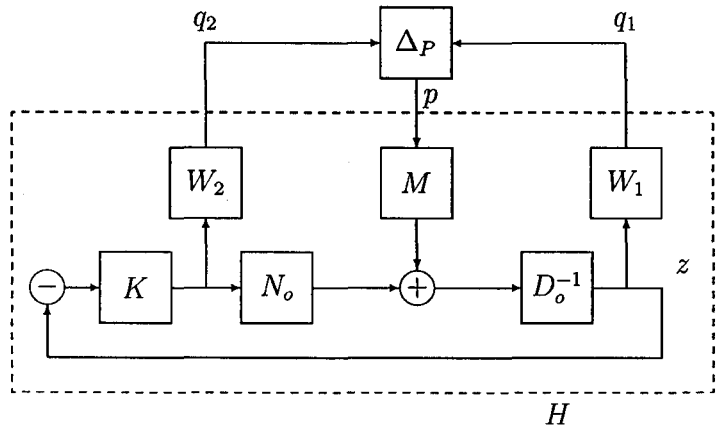
Burada, $W(s)$, büyüklüğü alçak frekanslarda yüksek, yüksek frekanslarda düşük olacak şekilde seçilir. Bu şekilde tanımlanan problem *ağırlıklı duyarlılık fonksiyonu minimizasyonu* olarak anılır [29].

3.2 $\mathcal{H}_\infty$ Optimizasyon Problemi

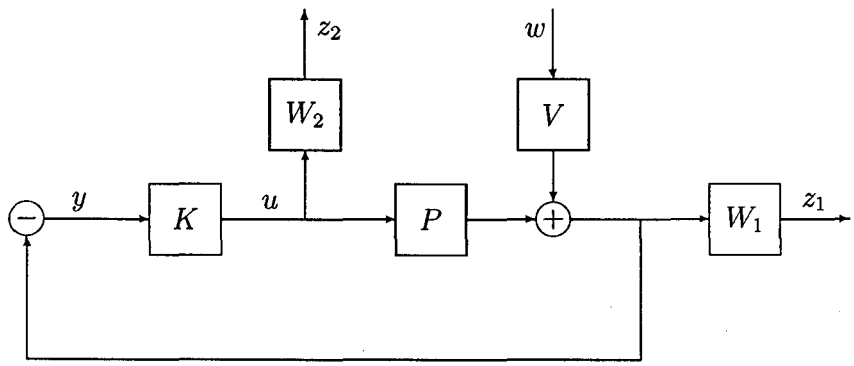
Temel $\mathcal{H}_\infty$ kontrol problemi, sistemin iç kararlılığını sağlayan ve belli bir transfer fonksiyonunun ∞–normunu minimize eden denetleyicilerin bulunması olarak tanımlanır. $\mathcal{H}_\infty$ kontrol teorisinde tanımlanan iki–blok optimizasyon (karma duyarlılık) probleminde gürbüzlük optimize edilmeye çalışılır. Burada, gürbüzlükten kasıt yalnızca kararlılık değildir; aynı anda gürbüz performansın da sağlanmasıdır.

Göreceli asal pertürbasyonların üzerine etkidiği bir planti içeren kapalı döngü sistem Şekil 3.2'de verilmiştir [29]. Burada, $P_0 = \frac{N_0}{D_0}$ nominal plantinin $P = \frac{N_0 + M\Delta_N W_2}{D_0 + M\Delta_D W_1}$ plantine pertürbe edildiği kabul edilmiş ve $\Delta_P = [\ \Delta_N \quad \Delta_D\ ]$ olarak tanımlanmıştır. Burada, $\|\Delta_P\|_\infty < 1$'dir. Şekilde kesikli çizgilerle gösterilmiş olan H bloğunun girişi p ve çıkışı da $q = [q_1 \quad q_2]^T$'dir. Transfer fonksiyonu matrisi H'i bulabilmek için girdi–çıktı ilişkileri yazılırsa,

$$\begin{aligned} \hat{q}_1 &= \frac{W_1 V}{1 + P_0 K}\hat{p} = W_1 S_0 V \hat{p} \\ \hat{q}_2 &= -\frac{W_2 K V}{1 + P_0 K}\hat{p} = -W_2 U_0 V \hat{p} \end{aligned} \tag{3.4}$$



Şekil 3.2: Geri besleme döngüde göreceli asal pertürbasyon modeli



Şekil 3.3: Karma duyarlılık problemi

elde edilir. Burada, $V = D_0^{-1}M$'dir ve

$$S_0 = \frac{1}{1+P_0K}, \qquad U_0 = \frac{K}{1+P_0K}$$

nominal duyarlılık ve nominal giriş duyarlılık fonksiyonlarıdır. Giriş duyarlılık fonksiyonu, bozucu sinyalden plant girişine olan transfer fonksiyonudur. Buradan, H transfer fonksiyonu matrisi,

$$H = \begin{bmatrix} W_1SV \\ -W_2KSV \end{bmatrix}$$

şeklinde bulunur. Yazılan bu ifadelere dayanarak, Şekil 3.2'deki sistem, Şekil 3.3'deki blok diyagramla gösterilebilir. Dolayısıyla, sistemdeki w bozucu sinyalinden çıktıya, $z := [z_1\ z_2]^T$, olan transfer fonksiyonunun, H, $\mathcal{H}_\infty$ normunun minimizasyonu *iki blok optimizasyon* problemini tanımlamaktadır. Bu problem standart $\mathcal{H}_\infty$ kontrol probleminin özel bir halidir. Bir sonraki

altbölümde bu problemin çözümü, yani, $\mathcal{H}_\infty$ denetleyici tasarımı üzerinde durulacaktır.

3.3 $\mathcal{H}_\infty$ Denetleyici Tasarımı

Gürbüz kararlılık ve gürbüz performans problemleri şu iki-blok optimizasyon problemi ile tanımlanabilir [27],

$$\gamma_{opt} : = \inf_{[K,P] \text{ kararlı}} \left\| \begin{bmatrix} W_1(1+PK)^{-1} \\ W_2PK(1+PK)^{-1} \end{bmatrix} \right\|_\infty \tag{3.5}$$

Bu çalışmada, ele alınan plant, gecikme içerdiği için sonsuz boyutludur. Böylesi sistemler için problem tanımları ve çözümleri [27]'de verilmiştir. (3.5)'te verilen problem tanımında W_1 ve W_2 çözülmek istenen kontrol problemine uygun, frekansa bağlı ağırlık fonksiyonlarıdır. Tanımı verilen iki-blok optimizasyon problemindeki plant ve ağırlıklar üzerinde şu kabullenmeler yapılmaktadır:

Kabullenme 3.1 *Tek girdili tek çıktılı (SISO), doğrusal ve zamanla değişmeyen (LTI), ve olası sonsuz boyutlu plant transfer fonksiyonları şu şekilde ifade edilebilir;*

$$P(s) = \frac{M_n(s)N_1(s)N_2(s)}{M_d(s)}$$

Burada $M_n(s)$ herhangi bir (olası sonsuz boyutlu) iç fonksiyon, $M_d(s)$ rasyonel bir iç fonksiyon, $N_1(s)$, $N_1(s)^{-1} \in \mathcal{H}_\infty(\mathbb{C}_+)$ olan olası sonsuz boyutlu bir dış fonksiyon ve $N_2(s)$ ise rasyonel bir dış fonksiyondur. Ayrıca, $P(s)$'in $j\mathbb{R}$ ekseninde sınırlı sayıda nokta dışında sürekli olduğu kabul edilir.

Kabullenme 3.2 *W_1 ağırlığı $W_1^{-1} \in \mathcal{H}_\infty$ olan rasyonel bir fonksiyon ve W_2 ağırlığı ise $(W_2N_2), (W_2N_2)^{-1} \in \mathcal{H}_\infty$ rasyonel olacak şekilde seçilen bir fonksiyon olarak kabul edilir.*

Plantin transfer fonksiyonu, $N(s), D(s) \in \mathcal{H}_\infty$ olmak üzere $P(s) = N(s)/D(s)$ göreceli asal ayrıştırmasına sahip olsun. Bu ayrıştırma sonucu Bezout eşitliğini sağlıyan $X(s), Y(s) \in \mathcal{H}_\infty$ bulunabilsin:

$$N(s)X(s) + D(s)Y(s) = 1 \tag{3.6}$$

Böyle bir ayrıştırmanın olması sistemi kararlaştıran bir denetleyicinin olması için gereklidir [27].

Teorem 2 *[27],[26]: İki $\mathcal{H}_\infty$ fonksiyonun oranı olan K denetleyicisi, planti, eğer ve ancak*

$$K(s) = \frac{X(s) + D(s)Q(s)}{Y(s) - N(s)Q(s)}$$

formunda ise kararlılaştırır. Burada $Q(s) \in \mathcal{H}_\infty$ kararlılık dışındaki tasarım kriterlerine göre seçilecek parametredir.

(3.5) ifadesinde verilen iki–blok optimizasyon problemi [30]'da verilen yöntemle çözülebilir. Bu yönteme göre, plantin, $P(s) = \frac{m_n(s)n_0(s)}{m_d(s)}$ göreceli asal ayrıştırmasına sahip olduğu kabul edilmektedir. Burada, $m_d(s) = \prod_{k=1}^{l} \frac{s-\alpha_k}{s+\alpha_k}$'dir ve $\alpha_1, \ldots, \alpha_l \in \mathbb{C}_+$ birbirinden farklıdır. $m_n \in \mathcal{H}_\infty(\mathbb{C}_+)$ ve $n_o \in \mathcal{H}_\infty(\mathbb{C}_+)$, sırasıyla, muhtemelen sınırsız boyutlu bir iç fonksiyon ve bir dış fonksiyondur. Ağırlıkların rasyonel ve W_1, $(W_2 n_0)$ ve $(W_2 n_0)^{-1} \in \mathcal{H}_\infty$ olduğu kabul edilmiştir. $\eta_1, \ldots, \eta_{n_1} \in \bar{\mathbb{C}}_+$, $n_1 \geq 1$ olmak üzere, $W_1(-s)$'in kutupları olsun.

$$E_\rho(s) := \left(\frac{W_1(s)W_1(-s)}{\rho^2} - 1 \right)$$

tanımlansın. Bu fonksiyonun sıfırları $\beta_1, \ldots, \beta_{2n_1}$ ile gösterilir ve β_i'ler, $\beta_1, \ldots, \beta_{n_1} \in \bar{\mathbb{C}}_+$ ve $\beta_{n_{1+i}} = -\beta_i$ şeklinde numaralandırılır.

$$F_\rho(s) := G_\rho(s) \prod_{k=1}^{n_1} \frac{s - \eta_k}{s + \eta_k}$$

tanımlansın. Burada, $G_\rho(s) \in \mathcal{H}_\infty(\mathbb{C}_+)$ minimum faz bir transfer fonksiyonudur ve aşağıdaki spektral ayrıştırmadan bulunur:

$$G_\rho(s)G_\rho(-s) := \left[1 - E_\rho(s) \left(\frac{W_2(s)W_2(-s)}{\rho^2} - 1 \right) \right]^{-1}$$

Buradan, optimal $\mathcal{H}_\infty$ denetleyici

$$K_{opt}(s) = E_{\gamma_0}(s) m_d(s) \frac{n_o(s)^{-1} F_{\gamma_0}(s) L(s)}{1 + m_n(s) F_{\gamma_0}(s) L(s)}$$

olarak verilir. Burada, $L(s) = L_1(s)/L_2(s)$ sıfırdan farklı bir polinomdur ve $L_1(s)$ ve $L_2(s)$ polinomlarının derecesi $n_1 + l - 1$'e eşittir. Bu polinomlar, $\rho = \gamma_0$ olduğu durumda aşağıdaki denklemleri sağlar;

$$L_1(\beta_k) + m_n(\beta_k) F_{\gamma_0}(\beta_k) L_2(\beta_k) = 0 \qquad k = 1, \ldots, n_1 \quad (3.7)$$

$$L_1(\alpha_k) + m_n(\alpha_k) F_{\gamma_0}(\alpha_k) L_2(\alpha_k) = 0 \qquad k = 1, \ldots, l \quad (3.8)$$

$$L_2(-\beta_k) + m_n(\beta_k) F_{\gamma_0}(\beta_k) L_1(-\beta_k) = 0 \qquad k = 1, \ldots, n_1 \quad (3.9)$$

$$L_2(-\alpha_k) + m_n(\alpha_k) F_{\gamma_0}(\alpha_k) L_1(-\alpha_k) = 0 \qquad k = 1, \ldots, l \quad (3.10)$$

Bu denklemler, payda terimi $(1 + m_n(s)F_{\gamma_0}(s)L(s))$'in, $E_{\gamma_0}(s)m_d(s)$'in kapalı sağ yarı düzlemdeki kutuplarını sadeleştirmesini sağlayacak koşullardır. Optimal $\mathcal{H}_\infty$ performans ölçütü, γ_0, bu denklemlerin matris gösteriminin en küçük tekil değerlerini çizdirerek bulunabilir. Çizimin sıfır olduğu en büyük γ değeri γ_0'yu verir.

3.4 Belli Bir $\mathcal{H}_\infty$ Optimizasyon Probleminin Tanımı ve Çözümü

Bu çalışmada, nodlarda gerçeklenen denetleyiciler, [8]'de tasarımı verilen denetleyicilerdir. Bu altbölümde, bu denetleyicileri bulmak için oluşturulan $\mathcal{H}_\infty$ optimizasyon probleminin, [8]'de verilen tanımı ve çözümü sunulacaktır.

[8] çalışmasında, incelenen geri besleme sistemi bir tıkanık nodu besleyen n kaynak ve bu kaynakların iletim oranlarını kontrol edecek, tıkanık nodda gerçeklenen bir denetleyiciden oluşmaktadır. Tıkanık nodda oluşabilecek kuyruk uzunluğunun dinamiği şu şekilde verilir;

$$\dot{q}(t) = \sum_{j=1}^{n} r_j^b(t) - c(t) \tag{3.11}$$

Burada, $q(t)$ kuyruk uzunluğunun t anındaki değeri, $r_j^b(t)$ j. kaynaktan gönderilen verinin t anında noda giriş oranı, $c(t)$ ise nodun t anındaki çıkış kapasitesidir. Bu modelde, gidiş–dönüş gecikmesi, $\tau_j(t)$, $\tau_j(t) = \tau_j^b(t) + \tau_j^f(t)$ ile verilir ve, $\tau_j^b(t) := h_j^b + \delta_j^b(t)$ denetleyiciden j. kaynağa olan geri yönlü gecikme iken $\tau_j^f(t) := h_j^f + \delta_j^f(t)$, j. kaynaktan noda olan ileri yönlü gecikmedir. Burada, $h_j^{\cdot} > 0$ gecikmenin zamanla değişmeyen ve bilinen kısmı; $\delta_j^{\cdot}(t)$ ise gecikmenin zamanla değişen ve bilinmeyen kısmını gösterir; üstsimge "$\cdot$" ise "f" veya "b" anlamına gelmektedir. Negatif zaman gecikmelerinin önüne geçmek için bilinen $\delta_j^+ > 0$ için $|\delta_j(t)| < \delta_j^+ \leq h_j$ olduğu kabul edilir.

Eğer (3.11) ifadesinin integrali alınacak olursa kuyruk uzunluğu elde edilebilir;

$$q(t) = \int_0^t \left[\sum_{j=1}^{n} r_j^b(t) - c(t) \right] d\nu + q(0) \tag{3.12}$$

Bu ifadedeki, j. kaynaktan t anında alınan verinin oranı,

$$r_j^b(t) = \begin{cases} \left(1 - \dot{\tau}_j^f(t)\right) r_j^s\left(t - \tau_j^f(t)\right), & t - \tau_j^f(t) \geq 0 \\ 0, & t - \tau_j^f(t) < 0 \end{cases} \tag{3.13}$$

olarak bulunur. Burada, $r_j^s(t) := r_j(t - \tau_j^b(t))$, j. kaynaktan noda t anında gönderilen verinin oranıdır ve $r_j(t)$ de denetleyicinin j. kaynak için hesapladığı iletim oranı komutunun t anındaki değeridir. Burada, $\frac{d}{dt}(t-\tau_j^f(t)) > 0$ olduğu, dolayısıyla, $\dot{\tau}_j^f(t) < 1$ ya da $\dot{\delta}_j^f(t) < 1$ kabul edilmiştir. Bu koşul sağlanmazsa, kaynağın gönderdiği bir paket, hedefe, önceden gönderilmiş bir paketten daha önce ulaşabilir. Ayrıca, $\delta_j(t)$ ve $\delta_j^f(t)$'nin şu koşulları sağladığı kabul edilir:

$$\left|\dot{\delta}_j(t)\right| < \beta_j \quad , \quad \left|\dot{\delta}_j^f(t)\right| < \beta_j^f \tag{3.14}$$

Burada $0 \leq \beta_j^f \leq \beta_j < 1$ bilinen sınır değerlerdir. $\dot{\tau}_j(t) = \dot{\delta}_j(t)$ bilgisi de kullanılarak, (3.12)

$$q(t) = \int_0^t \left[\sum_{j=1}^{n} \left(1 - \dot{\delta}_j^f(\nu)\right) r_j\left(\nu - \tau_j(\nu)\right) - c(\nu)\right] d\nu + q(0), \tag{3.15}$$

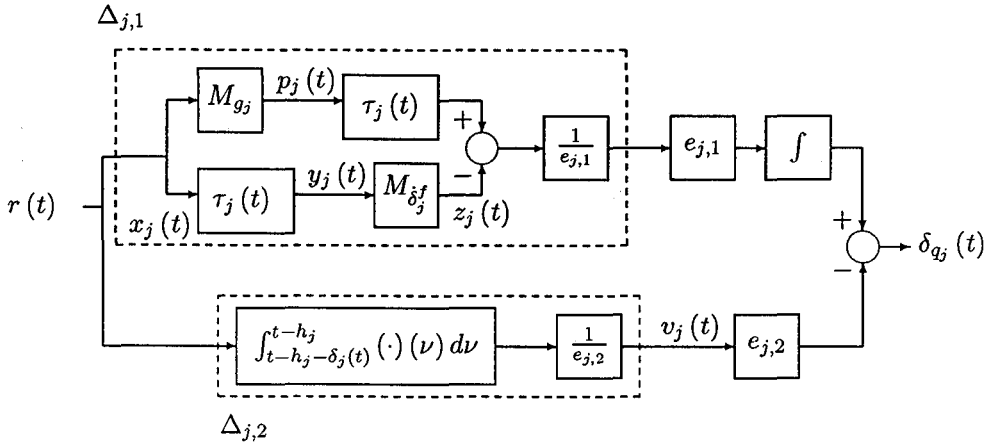
şeklinde yeniden yazılabilir. Buradan, nominal kuyruk uzunluğu ifadesi,

$$q_0(t) = \int_0^t \left[\sum_{j=1}^{n} r_j\left(\nu - h_j\right) - c(\nu)\right] d\nu + q(0), \tag{3.16}$$

olur. $\delta_q(t) := q(t) - q_0(t)$ kuyruk uzunluğundaki belirsizlik olarak ifade edilebilir. Buradan,

$$\delta_q(t) = \sum_{j=1}^{n} \int_0^t \left[(1 - \dot{\delta}_j^f(\nu)) r_j(\nu - \tau_j(\nu)) - r_j(\nu - h_j)\right] d\nu \tag{3.17}$$

olarak elde edilir. $\lambda_j = \nu - h_j - \delta_j(\nu) =: z_j(\nu)$ tanımı yapılır ve $\delta_j(0) = 0$ olarak kabul edilirse (3.17)'deki ifade aşağıdaki gibi düzenlenebilir;



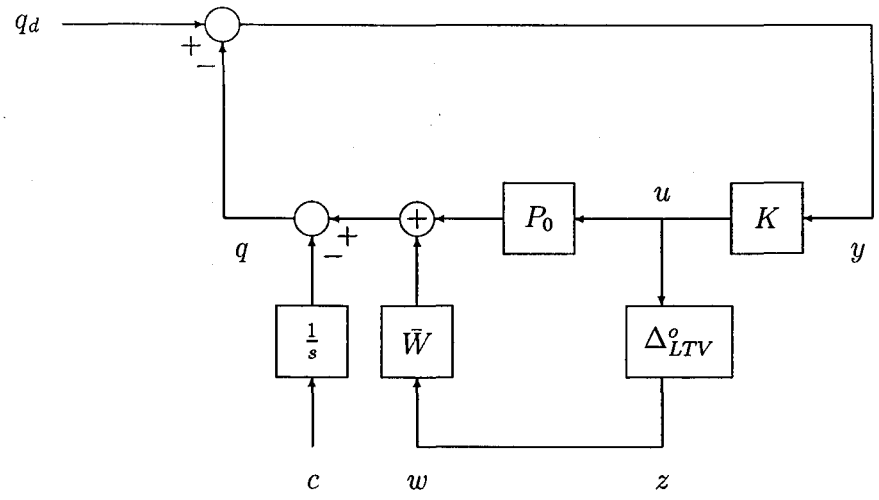
Şekil 3.4: Sistemdeki belirsizlikler, $\delta_{q_j}(t)$

$$
\begin{aligned}
\delta_q(t) &= \sum_{j=1}^{n}\left[\int_0^t (1-\dot{\delta}_j^f(\nu))r_j(\nu-\tau_j(\nu))\,d\nu - \int_{-h_j}^{t-h_j} r_j(\sigma)\,d\sigma\right] \\
&= \sum_{j=1}^{n}\left[\int_0^t (1-\dot{\delta}_j^f(\nu))r_j(\nu-\tau_j(\nu))\,d\nu - \int_{-h_j}^{t-h_j} r_j(\sigma)\,d\sigma\right. \\
&\quad \left. + \int_{-h_j}^{t-h_j-\delta_j(t)} r_j(\sigma)\,d\sigma - \int_{-h_j}^{t-h_j-\delta_j(t)} r_j(\lambda_j)\,d\lambda_j\right] \\
&= \sum_{j=1}^{n}\left[\int_0^t (1-\dot{\delta}_j^f(\nu))r_j(\nu-\tau_j(\nu))\,d\nu - \int_{t-h_j-\delta_j(t)}^{t-h_j} r_j(\nu)\,d\nu\right. \\
&\quad \left. - \int_0^t r_j(\nu-\tau_j(\nu))\left[1-g_j(\nu-\tau_j(\nu))\right]\,d\nu\right] \\
&= \sum_{j=1}^{n}\left[\int_0^t \left[g_j(\nu-\tau_j(\nu))-\dot{\delta}_j^f(\nu)\right] r_j(\nu-\tau_j(\nu))\,d\nu - \int_{t-h_j-\delta_j(t)}^{t-h_j} r_j(\nu)\,d\nu\right]
\end{aligned}
$$

Burada,

$$
\delta_{q_j}(t) = \int_0^t \left[g_j(\nu-\tau_j(\nu))-\dot{\delta}_j^f(\nu)\right] r_j(\nu-\tau_j(\nu))\,d\nu - \int_{t-h_j-\delta_j(t)}^{t-h_j} r_j(\nu)\,d\nu
$$

Dolayısıyla, $\delta_q(t) = \sum_{j=1}^{n}\delta_{q_j}(t)$'dir ve $\delta_{q_j}(t)$ Şekil 3.4'te verilen sistemin çıktısıdır. Şekilde, $\Delta_{j,1}$ ve $\Delta_{j,2}$ doğrusal, zamanla değişen sistemleri göstermektedir. Ayrıca, M_{g_j} ve $M_{\dot{\delta}_j^f}$ ile sırasıyla $p_j(t) = g_j(t)\,r(t)$ ve $z_j(t) = \dot{\delta}_j(t)\,y_j(t)$ ifadeleriyle tanımlanan zamana bağlı doğrusal sistemleri göstermektedir. Eğer, $e_{j,1} = \frac{\beta_j+\beta_j^f}{\sqrt{1-\beta_j}}$ olarak tanımlanırsa $\Delta_{j,1}$ LTV sisteminin endüklenmiş $\mathcal{L}_2$ normu 1'den küçük olacaktır. Benzer şekilde, $e_{j,2} = 2\delta_j^+$



Şekil 3.5: Ağ için oluşturulan suni model

olarak tanımlanırsa, $\Delta_{j,2}$ LTV sisteminin endüklenmiş $\mathcal{L}_2$ normu 1'den küçük olacaktır. Ağı modelleyen sistem Şekil 3.5'te verilmiştir.

Şekilde, $q_d(t)$ ulaşılmak istenen kuyruk uzunluğu değeridir. Ayrıca,

$$P_o(s) = \frac{1}{s}\left[e^{-h_1 s} \cdots e^{-h_n s}\right] \ ,$$

$$\Delta^o_{LTV} = \begin{bmatrix} \begin{bmatrix} \Delta^o_{1,1} \\ \Delta^o_{1,2} \end{bmatrix} & & 0 \\ & \ddots & \\ 0 & & \begin{bmatrix} \Delta^o_{n,1} \\ \Delta^o_{n,2} \end{bmatrix} \end{bmatrix} ,$$

$$\bar{W}(s) = \left[\bar{W}_1(s) \cdots \bar{W}_n(s)\right] \ , \quad \bar{W}_j(s) = \left[\frac{e_{j,1}}{s} \ \ e_{j,2}\right] \ ,$$

Burada, $\Delta^o_{j,k}$ ($j = 1 \cdots n$ ve $k = 1, 2$) normu 1'den küçük olan herhangi LTV sistemlerdir. Bu şekilde, Δ^o_{LTV}'nin endüklenmiş $\mathcal{L}_2$ normu $\sqrt{2}$'den küçük olur.

Nominal plant, $P_0(s)$, için $\mathcal{H}_\infty$'de şu göreceli asal ayrıştırma yapılabilir:

$$P_0(s) = N(s)M^{-1}(s) = \tilde{M}^{-1}(s)\tilde{N}(s)$$

Burada, I_n, $n \times n$ birim matrisi ve $\epsilon > 0$ da gelişi güzel seçilen bir sayı olmak üzere, $N(s) = \tilde{N}(s) = \frac{1}{s+\epsilon}\left[e^{-h_1 s} \cdots e^{-h_n s}\right]$, $M(s) = \frac{s}{s+\epsilon} I_n$, ve $\tilde{M}(s) = \frac{s}{s+\epsilon}$ olarak verilir. Böylesi bir ayrıştırma Şekil 3.5'te verilen sistemi

gürbüz kararlılaştırılacak denetleyicinin bulunmasında kullanılacaktır. Bunu sağlayacak tüm $K(s)$ denetleyicilerinin parametrizasyonu şu şekilde verilir, [28];

$$K(s) = [X(s) + M(s)Q(s)]\,[Y(s) - N(s)Q(s)]^{-1} \tag{3.18}$$

Burada $Q \in \mathcal{H}_\infty$'dir ve $X \in \mathcal{H}_\infty$ ve $Y \in \mathcal{H}_\infty$ Bezout özdeşliğini sağlar;

$$\tilde{M}(s)Y(s) + \tilde{N}(s)X(s) = 1 \tag{3.19}$$

Buradan, $Y(s) = \tilde{M}^{-1}(s)\left[1 - \tilde{N}(s)X(s)\right] = \frac{s+\epsilon}{s} - \frac{\epsilon}{s}\sum_{j=1}^{n}\alpha_j e^{-h_j s}$ olarak bulunur. Burada α_j katsayıları, $\sum_{j=1}^{n}\alpha_j = 1$ eşitliğini sağlayan ağırlıklı eşitlik katsayılarıdır. Bu durumda, tasarlanacak denetleyicinin parametrizasyonundaki tek bilinmeyen $Q_j(s)$'dir. Bu parametre, oluşturulacak $\mathcal{H}_\infty$ probleminin çözümü ile elde edilecektir. İki-blok $\mathcal{H}_\infty$ optimizasyon problemi, şu şekilde yazılabilir;

$$\gamma_{opt} := \inf \left\| \begin{bmatrix} W_s S \\ WKS \end{bmatrix} \right\|_\infty \tag{3.20}$$

$$= \inf_{K,P_0\text{'ı kararlılaştırıyor}} \left\| \begin{bmatrix} W_s(1+P_0K)^{-1} \\ WK(1+P_0K)^{-1} \end{bmatrix} \right\|_\infty \tag{3.21}$$

Burada $W_s(s) = \frac{1}{s}$ ve $W(s) = w(s)I_n$'dir ve,

$$w(s) = \sqrt{2}\left(\frac{1}{s}\sqrt{\sum_{j=1}^{n} e_{j,1}^2} + \sqrt{\sum_{j=1}^{n} e_{j,2}^2}\right) .$$

ile verilir. Ayrıca, infimum, nominal planti kararlılaştıran tüm denetleyiciler üzerinden alınır. Ancak, sistem SISO olmadığı için bu problemin optimal çözümünü bulmak çok zordur; hatta, çözüm bulunamayabilir. Bu nedenle, bir altoptimal çözüm önerilmiştir. Bunun için $Q(s) = [Q_1(s) \cdots Q_n(s)]$ yazılıp;

$$C_j(s) := [X_j(s) + M_j(s)Q_j(s)]\,[Y_j(s) - N_j(s)Q_j(s)]^{-1} \tag{3.22}$$

olarak tanımlanırsa,

$$Q_j(s) = [M_j(s) + C_j(s)N_j(s)]^{-1}\,[C_j(s)Y_j(s) - X_j(s)] \ . \tag{3.23}$$

olarak elde edilir. Burada,

$$Y_j(s) := \frac{s+\epsilon}{s} - \frac{\epsilon}{s}e^{-h_j s} \ , \quad N_j(s) := \frac{1}{\alpha_j(s+\epsilon)}e^{-h_j s} \ , \quad M_j(s) := \frac{s}{s+\epsilon} \ ,$$

ve $\tilde{M}_j(s)Y_j(s) + \tilde{N}_j(s)X_j(s) = 1$ Bezout özdeşliğini sağlayacak $X_j(s) := \alpha_j \epsilon$, olarak tanımlansın. Buradan, gerekli düzenlemeler yapılarak, iki–blok optimizasyon problemi şu şekilde ifade edilebilir;

$$\inf_{C_j \text{ kararlılaştırıyor } P_j} \left\| \begin{bmatrix} n^{-1}W_s\left[1 + P_jC_j\right]^{-1} \\ (n\alpha)^{-1}wC_j\left[1 + P_jC_j\right]^{-1} \end{bmatrix} \right\|_\infty =: \gamma_j \tag{3.24}$$

Elde edilen yeni problemin, (3.24), optimal çözümü [30]'daki yöntem kullanılarak [8]'de şu şekilde verilmiştir:

$$C_j(s) = \frac{n\alpha_j\gamma_j}{2\sqrt{2\sum_{k=1}^{n}\left(\delta_k^+\right)^2}} \left(\frac{sh_j - k_j}{sh_j}\right) \frac{1}{1 + F_j(sh_j)} \tag{3.25}$$

burada

$$F_j(\zeta) = \frac{(\zeta + k_j)(\zeta + a_j)(\zeta^2 + b_j\zeta + c_j) - (\zeta^4 - \hat{\gamma}_j^{-2}) - \frac{\hat{\gamma}_j}{\delta_j}e^{-\zeta}\zeta^2(\zeta - k_j)}{\zeta^4 - \hat{\gamma}_j^{-2}} \tag{3.26}$$

Çözümle ilgili ayrıntılar [8]'de bulunabilir. Buradaki $F_j(sh)$ bir sınırlı darbe tepkili filtredir (finite impulse response (FIR) filter). Filtrenin darbe yanıtının sıfırdan farklı olduğu süre h_j'dir. (3.25) ifadesi (3.23)'de yerine konacak olursa Q_j, buradan da Q elde edilebilir. Q, (3.18)'de yerine konduğunda $K(s) = [K_1(s) \cdots K_n(s)]^T$ bulunabilir. Burada,

$$K_j(s) = \frac{C_j(s)}{1 + C_j(s)P_j(s)} \left(1 - \sum_{k=1}^{n} \alpha_k \frac{P_k(s)C_k(s)}{1 + P_k(s)C_k(s)}\right)^{-1} \tag{3.27}$$

ile verilmektedir. Optimal performans ölçütü, γ_{opt}, ise $\gamma_{\text{opt}} \leq \sum_{j=1}^{n} n\alpha_j\gamma_j$ eşitsizliğini sağlar.

4 ATM AĞLARINDA ÇOK TIKALI DURUM İÇİN $\mathcal{H}_\infty$ DENETLEYİCİ GERÇEKLEMESİ

Bu çalışmada birden fazla tıkanık nodu olan ATM ağları ele alınmıştır. Ağdaki tıkanıklıkları önlemek için bir $\mathcal{H}_\infty$ denetleyici tasarımı yapılması düşünülmüştür. Akış kontrolü algoritması oran–tabanlı yaklaşıma göre oluşturulacaktır. Tasarlanacak denetleyicinin, bağlantılardaki zaman gecikmelerinde görülen belirsizliklere karşı gürbüz kararlı olması, mevcut kapasiteyi farklı bağlantılara farklı oranlarda paylaştırabilmesi, nodlardaki kuyruk uzunluklarını istenen düzeyde tutması beklenmektedir.

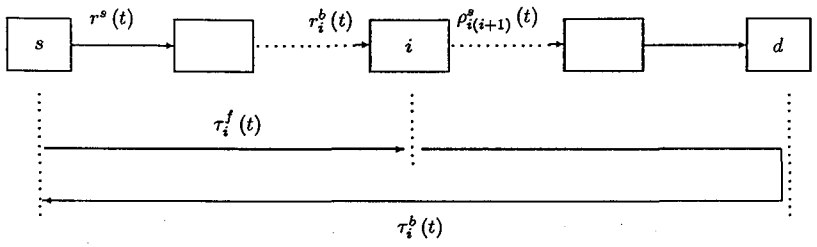
4.1 Tek Kaynak Durumu İçin Problem Tanımı ve Denetleyici Gerçeklemesi

Bu altbölümde, tek bir kaynağın veri iletimi yaptığı bir ağ ele alınacaktır. Şekil 4.1'de bağlantının izlediği yol ve zaman gecikmeleri gösterilmiştir. Ağ üzerinde yapılan kabullenmeler şunlardır;

(a) Ağ $(l-1)$ tane ara noddan ve iletim linklerinden oluşsun. Nodların oluşturduğu kümenin $\mathcal{N} = \{0, 1, \ldots, l\}$ olduğu kabul edilsin.

(b) Ağ trafiği, kaynak–hedef çiftinin, (s,d), arasındaki akışlardan meydana gelir. Kaynak–hedef çifti için *bağlantı* ifadesi kullanılacaktır. Bu şekilde, bir s noduna bağlı olan birden fazla kullanıcının, bir d noduna bağlı olan bir veya birden fazla kullanıcıya gönderdiği trafik, s ve d nodları arasında tek bir bağlantıya karşılık gelir.

(c) $p(k), k = 0, 1, \ldots, l$, bağlantının ağda izlediği nodlar sırasını, yolu, göstersin. Burada, $p(l) = d$'dir. Bağlantının geçtiği yol üzerindeki nodlar numaralandırılırken kaynak 0. nod olarak anılır.



Şekil 4.1: (s, d) bağlantısının ağda izlediği yol ve zaman gecikmeleri

(d) Kaynağın, s, iletim yapmayı istediği oran $r^0(t)$ olsun. Sistem yatışkan duruma ulaştığında bu oranın sabit olduğu; ancak, herhangi bir değer alabileceği ve bu değerin bilinmediği kabul edilsin.

(e) Ağdaki her nodda iletim için bekleyen paketleri tutmak için bir bellek vardır. i. nodun belleğindeki kuyruk uzunluğu $q_i(t)$ ile gösterilsin.

(f) Kontrol Protokolü: Bağlantının, ağa şu oranla veri göndermesine izin verilir,

$$r^s(t) = \min\left\{r^0(t); r_1(t - \tau_1^b(t)); \ldots; r_l(t - \tau_l^b(t))\right\}, \qquad (4.1)$$

(4.1) ifadesi, izin verilen trafiğin iletim oranının, bağlantının yolu üzerindeki nodların kaynak için belirledikleri iletim oranlarının en küçüğüne ya da kaynağın iletim yapmak istediği orana, hangisi daha küçükse, o değere eşit olacağını belirtir. Buradaki $\tau_i^b(t)$ gecikmesi i nodundan, s kaynağına olan geri yöndeki gecikmedir. $r_i(t)$, i nodundaki denetleyicinin bağlantı için belirlediği iletim oranıdır.

(g) Nodlarda çıkış kuyruklaması yapılmaktadır ve her noddaki trafik, *ilk geleni ilk ilet* (First–In–First–Out, FIFO) disiplinine göre iletilir. İletim için bekleyen tüm trafik tipleri, mümkün olan bant genişliğini ağırlıklı eşitlik esasına göre paylaşırlar.

Ağın modellemesinde kullanılan notasyon aşağıda verilmiştir;

$r_i(t)$: i. noddaki denetleyicinin kaynak için t anında belirlediği iletim oranı; $(i = 1, \ldots, l)$

$r^s(t)$: Kaynağın t anındaki gerçek iletim oranı;

$r_i^b(t)$: t anında i. noda veri giriş oranı; $(i = 1, \ldots, l)$

$\tau_i^f(t)$: Kaynaktan i. noda, ileri yöndeki zaman gecikmesinin t anındaki değeri; $(i = 1, \ldots, l)$

$\tau_i^b(t)$: i. noddan kaynağa, geri yöndeki zaman gecikmesinin t anındaki değeri; $(i = 1, \ldots, l)$

$\rho_{i,k}^s(t)$: i. noddan k. noda t anında gönderilen verinin iletim oranı; $(i = 1, \ldots, l, \quad k = 1, \ldots, l, \quad i \neq k)$

$\tau_{i,k}^f(t)$: i. noddan k. noda, ileri yöndeki zaman gecikmesinin t anındaki değeri; $(i = 1, \ldots, l, \quad k = 1, \ldots, l, \quad i \neq k)$

$\tau_{i,k}^b(t)$: k. noddan i. noda, geri yöndeki zaman gecikmesinin t anındaki değeri; $(i = 1, \ldots, l, \quad k = 1, \ldots, l, \quad i \neq k)$

$q_i(t)$: i. noda ait kuyruk uzunluğunun t anındaki değeri; $(i = 1, \ldots, l)$

$q_{d,i}$: i. noda ait ulaşılmak istenen kuyruk uzunluğu; $(i = 1, \ldots, l)$

$c_i(t)$: i. nodun t anındaki çıkış kapasitesi; $(i = 1, \ldots, l)$

Sistemdeki tüm zaman gecikmeleri $\tau(t) = h + \delta(t)$ formundadır. Burada, $h > 0$ gecikmenin zamanla değişmeyen ve bilinen kısmı; $\delta(t)$ ise gecikmenin zamanla değişen ve bilinmeyen kısmını gösterir. Negatif zaman gecikmelerinin önüne geçmek için bilinen $\delta^+ > 0$ için $|\delta(t)| < \delta^+ \leq h$ olduğu kabul edilir.

ABR servisinde paketlerin gönderildiği hedef nodlar da kaynağın iletim hızını düşürebileceğinden, buradaki hedef nod, d, bir anahtar nod gibi ele alınacaktır. Dolayısıyla, bir kaynak–hedef çifti ve ara nodlardan oluşan Şekil 4.1'deki sistem aslında bir kaynak ve l noddan oluşan bir sistem olarak düşünülebilir. Buradan, herhangi bir noddaki kuyruk uzunluğunun zamanla değişimi deterministik akışkanlar kanununa göre yazılacak olursa;

$$\dot{q}_i(t) = r_i^b(t) - \rho_{i,(i+1)}^s(t) \tag{4.2}$$

elde edilir. Eğer (4.2) ifadesinin integrali alınacak olursa kuyruk uzunluğunun denklemi elde edilebilir;

$$q_i(t) = \int_0^t \left[r_i^b(\nu) - \rho_{i,(i+1)}^s(\nu)\right] d\nu + q_i(0) \tag{4.3}$$

Burada, ağ yapısı göz önüne alınarak

$$\int_0^t r_i^b(\nu)d\nu = \begin{cases} \int_0^{t-\tau_{(i-1),i}^f(t)} \rho_{(i-1),i}^s(\theta)d\theta, & t-\tau_{(i-1),i}^f(t) \geq 0 \\ 0, & t-\tau_{(i-1),i}^f(t) < 0 \end{cases} \tag{4.4}$$

olarak elde edilebilir. Eğer (4.4) ifadesinin türevi alınırsa,

$$r_i^b(t) = \begin{cases} \left(1-\dot{\tau}_{(i-1),i}^f(t)\right)\rho_{(i-1),i}^s\left(t-\tau_{(i-1),i}^f(t)\right), & t-\tau_{(i-1),i}^f(t) \geq 0 \\ 0, & t-\tau_{(i-1),i}^f(t) < 0 \end{cases} \tag{4.5}$$

olarak bulunur.

Burada, $i = 1,\ldots,l$ için $\frac{d}{dt}(t-\tau_{(i-1),i}^f(t)) > 0$ olduğu, dolayısıyla, $\dot{\tau}_{(i-1),i}^f(t) < 1$ ya da $\dot{\delta}_{(i-1),i}^f(t) < 1$ kabul edilmiştir. Bu koşul sağlanmazsa, kaynağın gönderdiği bir paket, hedefe, önceden gönderilmiş bir paketten daha önce ulaşabilir. Ayrıca, $\delta_{(i-1),i}(t)$ ve $\delta_{(i-1),i}^f(t)$'nin şu koşulları sağladığı kabul edilir;

$$\left|\dot{\delta}_{(i-1),i}(t)\right| < \beta_i \ , \quad \left|\dot{\delta}_{(i-1),i}^f(t)\right| < \beta_i^f \tag{4.6}$$

Burada $0 \leq \beta_i^f \leq \beta_i < 1$ bilinen sınır değerlerdir. $\dot{\tau}_{(i-1),i}(t) = \dot{\delta}_{(i-1),i}(t)$ bilgisi kullanılarak, (4.5) ifadesini (4.3)'te yerine koyacak olursak,

$$q_i(t) = \int_0^t \left[\left(1-\dot{\delta}_{(i-1),i}^f(\nu)\right)\rho_{(i-1),i}^s\left(\nu-\tau_{(i-1),i}^f(\nu)\right) - \rho_{i,(i+1)}^s(\nu)\right] d\nu + q_i(0), \tag{4.7}$$

olur. Burada, $i = 1$ ise $\rho_{01}^s(t) = r^s(t)$ olur ve (4.1)'de verilen kaynağın veri gönderim oranı (4.7) ifadesinde $i = 1$ için yerine konursa, kuyruk uzunluğu şu şekilde bulunur:

$$q_1(t) = -\int_0^t \rho_{12}^s(\nu)d\nu + q_1(0) + \int_0^t \left(1-\dot{\delta}_1^f(\nu)\right) min\left\{r^0(\nu-\tau_1^f(\nu)); r_1(\nu-\tau_1(\nu)); \ldots; r_l(\nu-\tau_l^b(\nu)-\tau_1^f(\nu))\right\} d\nu \tag{4.8}$$

Ayrıca, $i.$ nodun $(i+1).$ noda göndereceği verinin iletim oranı, $\rho_{i,(i+1)}^s(t)$, $i.$ noda ait kuyruk uzunluğu değerine göre farklı değerler alır;

$$\rho_{i,(i+1)}^s(t) = \begin{cases} c_i(t), & q_i(t) > 0 \\ min\left\{c_i(t); r_i^b(t)\right\}, & q_i(t) = 0 \end{cases} \tag{4.9}$$

Burada, $i = l$ olduğunda, $\rho^s_{l,(l+1)}(t) = \rho^s_l(t)$ olur ve bu oran, hedef nodun veri işleme hızı ile kaynağa geri gönderilen RM hücrelerinin toplam iletim hızını ifade eder. (4.7)'den de görüldüğü gibi $i.$ nodun kuyruk uzunluğu, $(i-1).$ noddan gönderilen verinin oranına ve $(i+1).$ noda gönderilecek verinin oranına bağlıdır. Bu oranlar da nodlarda oluşan kuyruk uzunluklarının değerlerine göre farklı değerler almaktadır. Kuyruk uzunluklarının aldıkları değerler ise nodların tıkanık olup olmamalarına göre değişmektedir, (4.9). Dolayısıyla, her nod için kuyruk uzunluğu ifadesi yazıldığında, bu noddan iteratif olarak geriye doğru tüm nodların tıkanıklık durumlarının incelenmesi gerektiği görülmektedir. Buradan da, $i.$ nodun kuyruk uzunluğunun, bu noddaki denetleyicinin karar verdiği iletim oranı komutu ve $i.$ noda ait gecikmeler cinsinden yazılamamakta olduğu görülür. Dolayısıyla, dış merkezli denetleyici tasarımı yapılamamıştır. Sistem için merkezi bir denetleyici tasarımı da yapılamamıştır; çünkü, sistem doğrusal değildir ve doğrusal olmayan sistemler için şu ana kadar verilmiş olan $\mathcal{H}_\infty$ problem çözümlerindeki yapılara uymamaktadır. Bu nedenle, bu modele göre bir $\mathcal{H}_\infty$ optimizasyon problemi oluşturulamamış; yani, denetleyici tasarlanamamıştır.

Bu yönteme göre, tıkanık nodların hangilerinin olacağı önceden bilinirse, kuyruk uzunluğu denklemlerinde kaynağın yeni iletim oranı, tıkanık olduğu bilinen nodun belirlediği, belli bir geri yönlü gecikmeden sonra, orana eşittir. Buradan, dış merkezli denetleyici tasarımı yapılabilir. Ancak, pratikte böyle bir durum söz konusu olamayacağı için tıkanık nodları önceden belirlemek yerine sistem üzerinde bazı kabullenmeler yapmak daha doğru olur. Buna göre, bağlantının üzerinden geçtiği her nod, bağlantı için tıkanık olan nodun kendisi olduğunu kabul ederek bir iletim oranı hesaplar. Daha sonra tüm nodlarca belirlenen oranların en küçüğü yeni iletim oranı komutu olarak kaynağa bildirilir. Eğer bu oran komutunun değeri, kaynağın iletim yapmayı istediği orandan daha küçükse kaynak bu oranla iletim yapmaya başlar. Tersi durumda ise kaynak, iletim yapmayı istediği oranı yeni iletim oranı olarak atar. Buna göre, her nod, bağlantının yolu üzerindeki en tıkanık nodun kendisi olduğunu kabul ederek oran hesaplayacağından, $i.$ nodun kuyruk

uzunluğu şu şekilde ifade edilebilir;

$$q_i(t) = \int_0^t \left(1 - \dot{\delta}_i^f(t)\right) r_i(\nu - \tau_i(\nu))d\nu - \int_0^t \rho^s_{i,(i+1)}(\nu) + q_i(0) \qquad (4.10)$$

Bu ifadedeki kuyruk uzunluğu dinamiği incelenecek olursa, problemimizdeki çok dar boğazlı durumun tek dar boğaza indirgenmiş olduğu görülür. Buradan, i. nod için nominal kuyruk uzunluğu yazılacak olursa;

$$q_{i,0}(t) = \int_0^t r_i(\nu - h_i)d\nu - \int_0^t \rho^s_{i,(i+1)}(\nu)d\nu + q_i(0) \qquad (4.11)$$

$\delta_{q_i}(t) := q_i(t) - q_{i,0}(t)$ kuyruk uzunluğundaki belirsizlik olarak ifade edilebilir; buradan,

$$\delta_{q_i}(t) = \int_0^t \left[\left(1 - \dot{\delta}_i^f(\nu)\right) r_i(\nu - \tau_i(\nu)) - r_i(\nu - h_i)\right] d\nu \qquad (4.12)$$

olarak elde edilir. Burada, altbölüm 3.4'te verilen çözüm yöntemi, her nod için $n = 1$ alınarak uygulanacak olursa, nodlarda gerçeklenecek denetleyiciler bulunabilir. Ayrıca, ağa sadece bir kaynak veri gönderdiği için, çözüm aşamasında, $\alpha_1 = 1$ alınmalıdır.

4.2 Çok Kaynak Durumu İçin Problem Tanımı ve Denetleyici Gerçeklemesi

Bu altbölümde, l kaynak–hedef çifti ve ara nodlardan oluşan bir ağ ele alınacaktır. Bu yapı, Şekil 4.2'de verilmiştir. Ağ üzerinde yapılan kabullenmeler şunlardır;

(a) Ağdaki nodların oluşturduğu kümenin $\mathcal{N} = \{N_1, N_2, \ldots, N_n\}$ olduğu kabul edilsin.

(b) Ağ trafiği, her bir kaynak–hedef çiftinin, (s,d), arasındaki akışlardan meydana gelir. Her bir kaynak–hedef çifti için *bağlantı* ifadesi kullanılacaktır. Bu şekilde, bir s noduna bağlı olan birden fazla kaynağın, bir d noduna bağlı olan bir veya birden fazla kullanıcıya gönderdiği trafik, s ve d nodları arasında tek bir bağlantıya karşılık gelir. Böylesi bağlantıların oluşturduğu küme $\mathcal{C}$; i. nodun k. çıkış portu üzerinden geçen bağlantıların oluşturduğu küme ise $\mathcal{C}_{i_k}$ olsun. Ayrıca, $j \in \mathcal{C}$, $j = 1, \ldots, l$ bağlantısının kaynak ve hedef nodları sırasıyla $s(j)$ ve $d(j)$ ile gösterilsin. $\mathcal{C}_{i_k}$ kümesinin eleman sayısı ise l_{i_k} ile ifade edilsin.

(c) $p_j = \{N_j^1, \ldots, N_j^{n_j}\}$, $(j = 1, 2, \ldots, l)$, j bağlantısının ağda izlediği nodlar sırasını, yolu, göstersin. Burada, n_j, j bağlantısının yolu üzerindeki nodların sayısıdır ve $p_j(n_j) = N_j^{n_j}$'dir. Örneğin, N_2 nodu j bağlantısının yolu üzerindeki ilk nod ise $N_j^1 = N_2$'dir. Bağlantının geçtiği yol üzerindeki nodlar numaralandırılırken, kaynak 0. nod olarak anılır.

(d) Her bir j bağlantısı için $r^{j,0}(t)$, $s(j)$ kaynağının iletim yapmayı istediği oran olsun. Sistem yatışkan duruma ulaştığında bu oranın sabit olduğu; ancak, herhangi bir değer alabileceği ve bu değerin bilinmediği kabul edilsin.

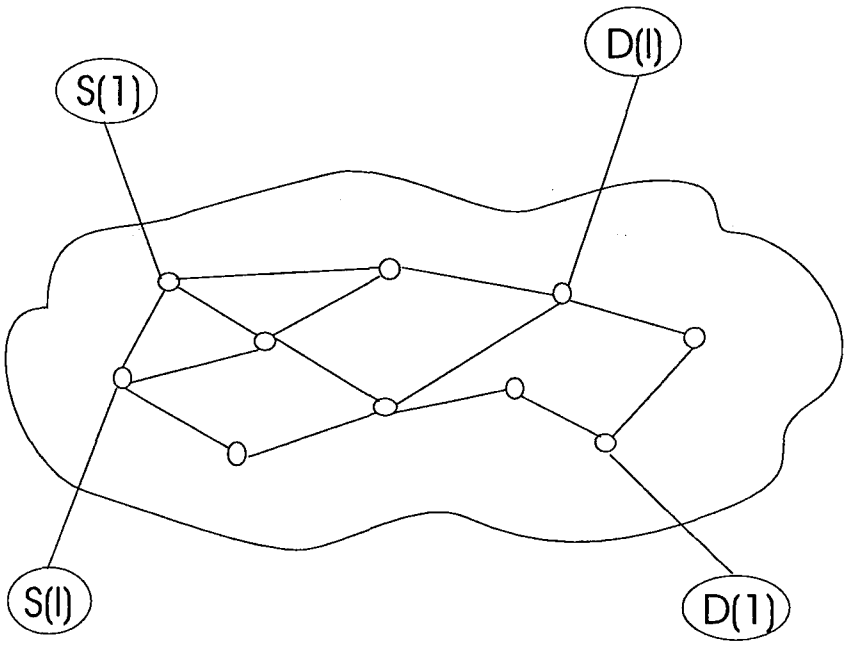
(e) Ağdaki her nodda iletim için bekleyen paketleri tutmak için bellekler vardır. Bu bellekler, nodların çıkış portlarında bulunmaktadır. Herhangi bir i nodundaki çıkış portu sayısı k_i ile; bu noddaki herhangi bir portda, i_k, oluşan kuyruk uzunluğu ise $q_{i_k}(t)$ ile gösterilsin.

(f) Kontrol Protokolü: j bağlantısının, ağa şu oranla veri göndermesine izin verilir,

$$r^{j,s}(t) = \min\left\{r^{j,0}(t); r^j_{N_j^1}(t - \tau^{j,b}_{N_j^1}(t)); \ldots; r^j_{N_j^{n_j}}(t - \tau^{j,b}_{N_j^{n_j}}(t))\right\}, j \in \mathcal{C} \tag{4.13}$$

(4.13) ifadesi, izin verilen trafiğin iletim oranının, bağlantının yolu üzerindeki nodların kaynak için belirledikleri iletim oranlarının en küçüğüne ya da kaynağın iletim yapmak istediği orana, hangisi daha küçükse, o değere eşit olacağını belirtir. Buradaki $\tau^{j,b}_{N_j^m}(t)$, $m = 1, \ldots, n_j$ gecikmesi, j bağlantısının yolu üzerindeki m. noddan $s(j)$ kaynağına olan geri yöndeki gecikmedir. $r^j_{N_j^m}(t)$ ise m. nodda j bağlantısı için belirlenen iletim oran komutudur. Burada, statik yönlendirme yapılmaktadır; yani, bağlantıların geçeceği nodların ve bu nodlarda kullanılacak çıkış portlarının sırası önceden bellidir. Her bağlantı, her nodun sadece bir çıkış portunu kullanacağından, nodlarda belirlenen oran komutları aslında, her bir nodda o bağlantının kullandığı çıkış portunda gerçeklenen denetleyici tarafından hesaplanan orandır; $r^j_{N_j^m}(t) = r^j_{m_k}(t)$.

(g) Nodlarda çıkış kuyruklaması yapılmaktadır ve her noddaki trafik, *ilk geleni ilk ilet* (First–In–First–Out, FIFO) disiplinine göre iletilir. İletim



Şekil 4.2: Bağlantıların ağda izlediği yol

için bekleyen tüm trafik tipleri, mümkün olan bant genişliğini ağırlıklı eşitlik esasına göre paylaşırlar.

Bu altbölümde kullanılacak notasyon aşağıda verilmiştir;

$r_{i_k}^{j}(t)$: i. nodun k. çıkış portundaki denetleyicinin j. kaynak için t anında belirlediği iletim oranı; $(i = 1, \ldots, n, \quad k = 1, \ldots, k_i, \quad j = 1, \ldots, l_{i_k})$

$r^{j,s}(t)$: j. kaynağın t anındaki gerçek iletim oranı; $(j = 1, \ldots, l)$

$r_{i_k}^{j,b}(t)$: j. kaynaktan gönderilen verinin t anında i. nodun k. çıkış portuna giriş oranı; $(i = 1, \ldots, n, \quad k = 1, \ldots, k_i, \quad j = 1, \ldots, l_{i_k})$

$\tau_{i_k}^{j,f}(t)$: j. kaynaktan i. nodun k. çıkış portuna, ileri yöndeki zaman gecikmesinin t anındaki değeri; $(i = 1, \ldots, n, \quad k = 1, \ldots, k_i, \quad j = 1, \ldots, l_{i_k})$

$\tau_{i_k}^{j,b}(t)$: i. nodun k. çıkış portundan j. kaynağa, geri yöndeki zaman gecikmesinin t anındaki değeri; $(i = 1, \ldots, n, \quad k = 1, \ldots, k_i, \quad j = 1, \ldots, l_{i_k})$

$\rho^{s}_{i_k,m_{\bar{k}}}(t)$: i. nodun k. çıkış portundan m. nodun $\bar{k}$. çıkış portuna t anında gönderilen verinin iletim oranı;
$(i = 1,\ldots,n,\ \ m = 1,\ldots,n\ \ , i \neq m,\ \ k = 1,\ldots,k_i,\ \ \bar{k} = 1,\ldots,k_m)$

$\tau^{f}_{i_k,m_{\bar{k}}}(t)$: i. nodun k. çıkış portundan m. nodun $\bar{k}$. çıkış portuna ileri yöndeki zaman gecikmesinin t anındaki değeri;
$(i = 1,\ldots,n,\ \ m = 1,\ldots,n\ \ , i \neq m,\ \ k = 1,\ldots,k_i,\ \ \bar{k} = 1,\ldots,k_m)$

$\tau^{b}_{i_k,m_{\bar{k}}}(t)$: m. nodun $\bar{k}$. çıkış portundan i. nodun k. çıkış portuna geri yöndeki zaman gecikmesinin t anındaki değeri;
$(i = 1,\ldots,n,\ \ m = 1,\ldots,n\ \ , i \neq m,\ \ k = 1,\ldots,k_i,\ \ \bar{k} = 1,\ldots,k_m)$

$q_{i_k}(t)$: i. nodun k. çıkış portuna ait kuyruk uzunluğunun t anındaki değeri;
$(i = 1,\ldots,n,\ \ k = 1,\ldots,k_i)$

q_{d,i_k} : i. nodun k. çıkış portuna ait ulaşılmak istenen kuyruk uzunluğu;
$(i = 1,\ldots,n,\ \ k = 1,\ldots,k_i)$

$c_{i_k}(t)$: i. nodun k. çıkış portunun t anındaki çıkış kapasitesi;
$(i = 1,\ldots,n,\ \ k = 1,\ldots,k_i)$

Sistemdeki tüm zaman gecikmeleri $\tau(t) = h + \delta(t)$ formundadır. Ayrıca, $\tau(t) = \tau^{f}(t) + \tau^{b}(t)$ olarak ifade edilmektedir. Dolayısıyla, $h(t) = h^{f}(t) + h^{b}(t)$ ve $\delta(t) = \delta^{f}(t) + \delta^{b}(t)$ olur. Burada, $h > 0$ gecikmenin zamanla değişmeyen ve bilinen kısmı; $\delta(t)$ ise gecikmenin zamanla değişen ve bilinmeyen kısmını gösterir. Negatif zaman gecikmelerinin önüne geçmek için bilinen $\delta^{+} > 0$ için $|\delta(t)| < \delta^{+} \leq h$ olduğu kabul edilir.

Burada da her hedef nod, $d(j)$, $j = 1,\ldots,l$, bir anahtar nod gibi ele alınacaktır. Herhangi bir i_k portundaki bellek kuyruk uzunluğunun zamanla değişimi deterministik akışkanlar kanununa göre yazılacak olursa;

$$\dot{q}_{i_k}(t) = \sum_{j \in \mathcal{C}_{i_k}} \left[r^{j,b}_{i_k}(t) - \rho^{j,s}_{i_k,i_+}(t) \right] \tag{4.14}$$

olarak elde edilir. Burada, i_+, j bağlantısının yolu üzerinde i nodundan sonra gelen nodu ifade eder. Eğer (4.14) ifadesinin integrali alınacak olursa kuyruk uzunluğunun denklemi elde edilebilir;

$$q_{i_k}(t) = \int_0^t \sum_{j \in \mathcal{C}_{i_k}} \left[r^{j,b}_{i_k}(\nu) - \rho^{j,s}_{i_k,i_+}(\nu) \right] d\nu + q_{i_k}(0) \tag{4.15}$$

Burada, ağ yapısı göz önüne alınarak

$$\int_0^t r_{i_k}^{j,b}(\nu)d\nu = \begin{cases} \int_0^{t-\tau_{i_-,i_k}^{j,f}(t)} \rho_{i_-,i_k}^{j,s}(\theta)d\theta, & t-\tau_{i_-,i_k}^{j,f}(t) \geq 0 \\ 0, & t-\tau_{i_-,i_k}^{j,f}(t) < 0 \end{cases} \tag{4.16}$$

olarak elde edilebilir. Burada, i_-, j bağlantısının yolu üzerinde i nodundan önce gelen nodu ifade eder. Eğer (4.16) ifadesinin türevi alınırsa,

$$r_{i_k}^{j,b}(t) = \begin{cases} \left(1-\dot{\tau}_{i_-,i_k}^{j,f}(t)\right)\rho_{i_-,i_k}^{j,s}\left(t-\tau_{i_-,i_k}^{j,f}(t)\right), & t-\tau_{i_-,i_k}^{j,f}(t) \geq 0 \\ 0, & t-\tau_{i_-,i_k}^{j,f}(t) < 0 \end{cases} \tag{4.17}$$

olarak bulunur. Burada, $i = 1,\ldots,n$, $k = 1,\ldots,k_i$ ve $j = 1,\ldots,l_{i_k}$ için $\frac{d}{dt}(t-\tau_{i_-,i_k}^{j,f}(t)) > 0$ olduğu, dolayısıyla, $\dot{\tau}_{i_-,i_k}^{j,f}(t) < 1$ ya da $\dot{\delta}_{i_-,i_k}^{j,f}(t) < 1$ kabul edilmiştir. Ayrıca, $\delta_{i_-,i_k}^{j}(t)$ ve $\delta_{i_-,i_k}^{j,f}(t)$'nin şu koşulları sağladığı kabul edilir:

$$\left|\dot{\delta}_{i_-,i_k}^{j}(t)\right| < \beta_{i_k}^{j} \quad , \quad \left|\dot{\delta}_{i_-,i_k}^{j,f}(t)\right| < \beta_{i_k}^{j,f} \tag{4.18}$$

Burada $0 \leq \beta_{i_k}^{j,f} \leq \beta_{i_k}^{j} < 1$ bilinen sınır değerlerdir. $\dot{\tau}_{i_-,i_k}^{j}(t) = \dot{\delta}_{i_-,i_k}^{j}(t)$ bilgisi kullanılarak, (4.17) ifadesini (4.15)'te yerine koyacak olursak,

$$q_{i_k}(t) = \int_0^t \sum_{j\in\mathcal{C}_{i_k}} \left[\left(1-\dot{\delta}_{i_-,i_k}^{j,f}(\nu)\right)\rho_{i_-,i_k}^{j,s}\left(\nu-\tau_{i_-,i_k}^{j,f}(\nu)\right) - \rho_{i_k,i_+}^{j,s}(\nu)\right]d\nu + q_{i_k}(0), \tag{4.19}$$

olur. Burada, i nodunun, yolları üzerindeki 1. nod olduğu bağlantılar için, $i = N_j^1$, $\rho_{01_k}^{j,s}(t) = r^{j,s}(t)$ olur ve o bağlantılar için (4.13)'te verilen kaynağın veri gönderim oranı kuyruk uzunluğu denkleminde yerine konur. Böylece, ağdaki herhangi bir nod için kuyruk uzunluğu şu şekilde yazılabilir;

$$\begin{aligned} q_{i_k}(t) = & \int_0^t \sum_{j\in\mathcal{C}_{i_k}^1} \left(1-\dot{\delta}_{i_-,i_k}^{j,f}(\nu)\right) r^{j,s}\left(\nu-\tau_{i_-,i_k}^{j,f}(\nu)\right)d\nu + \\ & \int_0^t \sum_{j\in\mathcal{C}_{i_k}^o} \left[\left(1-\dot{\delta}_{i_-,i_k}^{j,f}(\nu)\right)\rho_{i_-,i_k}^{j,s}\left(\nu-\tau_{i_-,i_k}^{j,f}(\nu)\right) - \rho_{i_k,i_+}^{j,s}(\nu)\right]d\nu + q_{i_k}(0) \end{aligned}$$

Burada, $\mathcal{C}_{i_k}^1$, yollarının üzerindeki ilk nodun i nodu olduğu ve bu nodun k. çıkış portunu kullanan bağlantıların oluşturduğu kümeyi, $\mathcal{C}_{i_k}^o$ ise i nodun k. çıkış portunu kullanan diğer bağlantıların oluşturduğu kümeyi temsil etmektedir. Ayrıca, i. nodun k. çıkış portundan bir sonraki noda gönderilecek toplam

verinin iletim oranı, $\rho^{s}_{i_k,i_+}(t)$, i nodun k. çıkış portuna ait kuyruk uzunluğunun alacağı değerlere göre farklı değerler alır;

$$\rho^{s}_{i_k,i_+}(t) = \begin{cases} c_{i_k}(t), & q_{i_k}(t) > 0 \\ min\left\{c_{i_k}(t); r^{b}_{i_-,i_k}(t)\right\}, & q_{i_k}(t) = 0 \end{cases} \tag{4.20}$$

Burada,

$$\rho^{s}_{i_k,i_+}(t) = \sum_{j \in \mathcal{C}_{i_k}} \rho^{j,s}_{i_k,i_+}(t), \qquad r^{b}_{i_-,i_k}(t) = \textstyle\sum_{j \in \mathcal{C}_{i_k}} r^{j,b}_{i_-,i_k}(t)$$

ile ifade edilmektedir ve

$$\rho^{j,s}_{i_k,i_+}(t) = \begin{cases} \frac{q^{j}_{i_k}(t)}{q_{i_k}(t)} c_{i_k}(t), & q_{i_k}(t) > 0 \\ \frac{r^{j,b}_{i_-,i_k}(t)}{r^{b}_{i_-,i_k}(t)} \rho^{s}_{i_k,i_+}(t), & q_{i_k}(t) = 0 \end{cases}$$

ile hesaplanır. Bu şekilde, noda gelen paketlerin iletiminde her kaynağa eşit hak tanınmış olur.

(4.19)'dan da görüldüğü gibi i_k. çıkış portunda oluşan kuyruk uzunluğu, bir önceki noddan gönderilen verinin oranına ve bir sonraki noda gönderilecek verinin oranına bağlıdır. Bu oranlar da ilgili nodlardaki çıkış portlarında oluşan kuyruk uzunluklarının değerlerine göre farklı değerler almaktadır (4.20). Kuyruk uzunluklarının aldıkları değerler ise çıkış portlarının tıkanık olup olmamalarına göre değişmektedir, (4.20). Dolayısıyla, her çıkış portu için kuyruk uzunluğu ifadesi yazıldığında, bu noddan iteratif olarak geriye doğru tüm nodlardaki portların tıkanıklık durumlarının incelenmesi gerektiği görülmektedir. Buradan da, i_k. portun kuyruk uzunluğunun, bu porttaki denetleyicinin karar verdiği iletim oran komutu ve i_k. noda ait gecikmeler cinsinden yazılamamakta olduğu görülür. Dolayısıyla, dış merkezli denetleyici tasarımı yapılamamaktadır. Sistem için merkezi bir denetleyici tasarımı da yapılamamaktadır; çünkü, sistem doğrusal değildir ve doğrusal olmayan sistemler için şu ana kadar verilmiş olan $\mathcal{H}_\infty$ problem çözümlerindeki yapılara uymamaktadır. Bu nedenle, bu modele göre bir $\mathcal{H}_\infty$ optimizasyon problemi oluşturulamamakta; yani, denetleyici tasarlanamamaktadır.

Burada da tek kaynaklı durumdakine benzer bir kabullenme yapılırsa, her çıkış portu, üzerinden geçen her bağlantının yolu üzerindeki en tıkanık portun kendisi olduğunu kabul ederek oran hesaplayacağından, i_k. portun kuyruk

uzunluğu şu şekilde ifade edilebilir;

$$q_{i_k}(t) = \int_0^t \sum_{j \in \mathcal{C}_{i_k}} \left(1 - \dot{\delta}_{i_k}^{j,f}(t)\right) r_{i_k}^j(\nu - \tau_{i_k}^j(\nu))d\nu - \int_0^t \rho_{i_k,i_+}^s(\nu)d\nu + q_{i_k}(0) \tag{4.21}$$

Bu ifadedeki kuyruk uzunluğu dinamiği incelenecek olursa, problemimizdeki çok dar boğazlı durumun tek dar boğaza indirgenmiş olduğu görülür. Buradan, i_k. çıkış portu için nominal kuyruk uzunluğu yazılacak olursa;

$$q_{i_k,0}(t) = \int_0^t \sum_{j \in \mathcal{C}_{i_k}} r_{i_k}^j(\nu - h_{i_k}^j)d\nu - \int_0^t \rho_{i_k,i_+}^s(\nu)d\nu + q_{i_k}(0) \tag{4.22}$$

$\delta_{q_{i_k}}(t) := q_{i_k}(t) - q_{i_k,0}(t)$ kuyruk uzunluğundaki belirsizlik olarak ifade edilebilir; buradan,

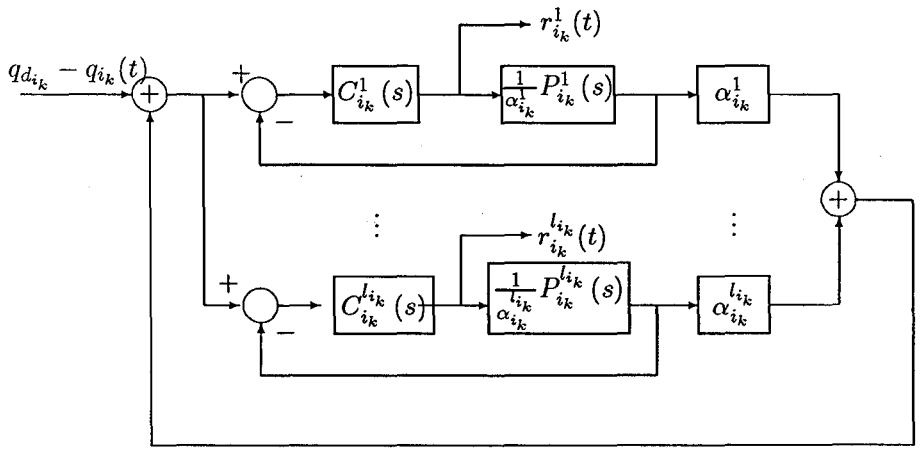
$$\delta_{q_{i_k}}(t) = \int_0^t \sum_{j \in \mathcal{C}_{i_k}} \left[\left(1 - \dot{\delta}_{i_k}^{j,f}(\nu)\right) r_{i_k}(\nu - \tau_{i_k}^j(\nu)) - r_{i_k}^j(\nu - h_{i_k}^j)\right]d\nu \tag{4.23}$$

olarak elde edilir. Burada, altbölüm 3.4'te verilen çözüm yöntemi, her noddaki her çıkış portu için uygulanacak olursa, portlarda gerçeklenecek denetleyiciler bulunabilir. i. nodun k. çıkış portundaki denetleyicinin gerçeklemesi Şekil 4.3'te verilmiştir. Burada, $\alpha_{i_k}^j$, $j = 1, \ldots, l_{i_k}$ katsayıları, her port için belirlenmiş olan ve $\sum_{j=1}^{l_{i_k}} \alpha_{i_k}^j = 1$ ifadesini sağlayan ağırlıklı eşitlik katsayılarıdır ve portların farklı kaynaklar için farklı öncelikler yaratmasını sağlar.

Tek ve çok kaynak durumu için gerçeklemeleri verilen denetleyiciler, ilgili çıkış portundaki kuyruk uzunluğu bilgisini kullanarak, kaynak için iletim oranı hesaplarlar. Her iki durumda da kaynakların, ağa

$$r^{j,s}(t) = \min\left\{r^{j,0}(t); r_1^j(t - \tau_1^{j,b}(t)); \ldots; r_{n_j}^j(t - \tau_{n_j}^{j,b}(t))\right\},$$

oranıyla veri göndermesine izin verilir. Bu şekilde, kaynağın iletim yapacağı yeni oran değerinin, tıkanık nodlarda hesaplanan oranlar ve kaynağın iletim yapmak istediği oran üzerinden minimum alınarak belirlenmesi ABR servisinde uygulanan kontrol yapısıyla da örtüşmektedir. Nodlardaki çıkış portlarında gerçeklenen denetleyiciler, yalnızca ilgili portlarda tıkanıklık söz konusuysa oran komutu hesaplarlar ve yalnız bu durumda hesaplanan oran komutları minimum fonksiyonu içine dahil edilir. Bu durum pratikte RM hücrelerindeki



Şekil 4.3: $K_{i_k}(s)$ Denetleyicisinin Gerçeklemesi.

CI bitinin değerinin kontrol edilmesiyle yapılabilir. Eğer bir porta gelen bir RM hücresindeki CI biti önceden 1 yapılmışsa ve o portta bir tıkanıklık yoksa denetleyici yeni bir oran hesaplamaz ve CI bitinin ve ER alanının değerini değiştirmeden hücreyi bir sonraki noda gönderir (burada bir sonraki nod yol üzerindeki bir başka anahtar, hedef nod ya da kaynak olabilir). CI biti 1 yapılmışsa ve o portta tıkanıklık vara, denetleyici bir oran komutu hesaplar, bunu RM hücresinin ER alanında yazan değerle karşılaştırır ve daha küçük olanı ER alanına yazarak hücreyi gönderir. CI bitinin değerinin 0 olduğu durumda portda tıkanıklık varsa CI biti 1 yapılır ve ER alanına denetleyicinin hesapladığı değer yazılır. CI bitinin değeri 0 iken portda herhangi bir tıkanıklık yoksa, RM hücresi, CI bitinin ve ER alanının değerleri değiştirilmeden bir sonraki noda gönderilir. Bu yapı da ATM ağlarındaki belirli oran işaretleme yöntemine uymaktadır.

5 BENZETİM ÇALIŞMALARI

Bu çalışmada anlatılan kontrol algoritmasının gerçek ağlara uygulandığında ne kadar verimli çalışacağını görmek için benzetim çalışmaları yapılmıştır. Bu amaçla, farklı sayıda kaynak ve anahtar içeren ağ modelleri için farklı parametre değerleri ve ağ koşulları ele alınarak kontrol uygulamasının benzetimi yapılmıştır. Yapılan benzetimlerde, kapasite ve iletim oranı istem değerleri, tıkanık nod sayısının 1'den fazla olması ve tıkanıklığın zaman içinde nodlar arasında yer değiştirmesi sağlanacak şekilde seçilmiştir. Çalışmalarda MATLAB/SIMULINK paket programı kullanılmıştır. Kullanılan MATLAB programları EK–1, EK–2, EK–3 ve EK–4'te verilmiştir. Bu programlar, [8]'deki benzetim çalışmaları için yazılan programları temel almaktadır. Denetleyicinin SIMULINK gerçeklemesi ise EK–5'te sunulmuştur.

Gerçeklenen benzetim mekanizmasında, ABR kaynaklarının temel özellikleri göz önünde bulundurulmuştur. ABR kaynaklarının iletim oranları sınırlıdır, $MCR \leq ACR \leq PCR$. Benzetim çalışmalarında $MCR = 0$ ve $PCR = 1000$ p/s olarak alınmıştır. Ayrıca, ABR kontrol yapısına göre, kaynak iletim oranı hesaplanırken sadece tıkanık olan nodların hesapladığı, RM hücrelerine yazılan, iletim oranları göz önüne alınır. Bu durum, kontrol mekanizmasında, kuyruk uzunluğu sıfır olan; yani, tıkanık olmayan, nodların hesapladığı iletim oranlarını kaynak girişindeki *min* fonksiyonuna dahil etmeyerek gerçeklenmiştir. Ayrıca, tıkanıklık yokken denetleyici yeni bir iletim oranı hesaplamamalı ya da oran eski değerinde sabit kalmalıdır. Benzetim mekanizmasında, kuyruk uzunluğu sıfır olduğunda, denetleyiciye girilen kuyruk uzunluğu bilgisi yerine, istenen kuyruk uzunluğu değeri girilmekte; böylece, denetleyiciye gelen, düzeltilmesi gereken hata sıfır olmakta ve denetleyicinin hesapladığı iletim oranı eski değerinde sabit kalmaktadır. Kontrol algoritmasında, RM hücrelerindeki CI ve NI bitleri kullanılarak nodlardaki denetleyicilerin gereksiz durumlarda karşılaştırma yapmaması amaçlanmıştır.

i	h_i	δ_i^+	β_i	β_i^f
1	1	4	0.1	0.01
2	1	4	0.2	0.02

Çizelge 5.1: Durum 1.1, 1.3, 1.4 ve 1.5'te Kullanılan Tasarım Parametreleri

i	h_i^b	δ_i^b	h_i^f	δ_i^f
1	0.9	$0.06\sin(\frac{2\pi}{50}t)$	0.1	$0.03\sin(\frac{2\pi}{50}t)$
2	0.5	$0.03\sin(\frac{2\pi}{50}t)$	0.5	$0.06\sin(\frac{2\pi}{50}t)$

Çizelge 5.2: Durum 1.1, 1.3, 1.4 ve 1.5'te Kullanılan Gerçekleme Parametreleri

Bu durum, benzetim mekanizmasında kuyruk uzunluklarının sıfırdan büyük ya da sıfıra eşit olması durumlarının koşul olarak kullanılması ile gerçeklenmiştir.

5.1 Bir Kaynak ve İki Noddan Oluşan Sistem için Yapılan Benzetim Çalışmalarında Elde Edilen Sonuçlar

Benzetim çalışmalarının bu bölümünde ele alınan ağ yapısı, Şekil 4.1'de verilen modelin, tek bir kaynak–hedef çifti ve 1 ara noddan oluşan şeklidir. Bu sistem için yapılan benzetim çalışmasında kullanılan SIMULINK diyagramı Ek–6'da verilmiştir.

Durum 1.1 Bu durum için denetleyici tasarımında kullanılan parametre değerleri Çizelge 5.1'de, gerçeklemede kullanılan değerler ise Çizelge 5.2'de verilmiştir. 1. nod için ulaşılmak istenen kuyruk uzunluğu $q_{d_1} = 100$ paket iken 2. nod için bu değer $q_{d_2} = 150$ paket olarak alınmıştır. Nodların çıkış kapasiteleri ve kaynağın iletim yapmayı istediği oran Şekil 5.2'de yer almaktadır. Bu durum için elde edilen sonuçlar ise Şekil 5.2 ve Şekil 5.3'te verilmiştir. Nodlardaki kuyruk uzunluğu grafiklerinde gözlenen, kuyruğun sıfır olduğu zaman aralıkları, nodlara gelen toplam veri iletim oranının nodun çıkış kapasitesini aşması için geçmesi gereken zaman aralıklarıdır. İlk 50 saniye içinde, kaynağın iletim yapmak istediği oran nodların çıkış kapasitelerinden daha büyüktür. Dolayısıyla, her iki nodda da kuyruk oluşması

beklenir. Nodlarda kuyruk oluşumuyla birlikte, denetleyiciler iletim oranı komutu hesaplamaya başlarlar. İlk anlarda, 1. noda gelen veriler kısa bir kuyruk oluşturup kısa bir süre içinde de diğer noda aktarılırlar. Bu da kuyruk uzunlukluğunun belli bir değere yükselip kısa zaman içinde tekrar sıfır olmasına yol açar. Kuyruk uzunlukluğunda meydana gelen bu değişim, hesaplanan iletim oran komutunda salınıma neden olur. Ancak zamanla, denetleyicilerin, kuyruk uzunluklarını istenen değerlerine getirdiği ilgili grafiklerden de görülmektedir. Kuyrukların sıfır olduğu daha uzun zaman aralıklarında, denetleyicilerin hesapladığı komutların, beklendiği gibi eski değerlerinde sabit kaldığı gözlenmektedir. Bu oranlar ve kaynağın iletim yapmak istediği oran göz önüne alındığında kaynağın gerçek iletim oranının beklendiği gibi gerçekleştiği görülmektedir. Kaynağın iletim hızının denetleyicilerin hesapladığı orana düşürülmesiyle birlikte, yeni iletim oranı, 2. nodun kapasitesinin altında kaldığı için, yaklaşık 50. saniyeden sonra bu nodun kuyruk uzunluğu sıfır olur. Ayrıca, 2. noddaki veri işleme hızı grafiği incelendiğinde, bu oranın, 2. nodun kuyruğunun sıfırdan büyük olduğu anlarda nodun kapasitesine, sıfıra eşit olduğu anlarda da kapasite ile gelen veri oranının minimumuna eşit olduğu görülmektedir.

Durum 1.2 Bu durum için denetleyici tasarımında kullanılan parametre değerleri Çizelge 5.3'te, gerçeklemede kullanılan değerler ise Çizelge 5.4'te verilmiştir. Nodlar için ulaşılmak istenen kuyruk uzunluğu değerleri, nod çıkış kapasiteleri ve kaynağın iletim yapmayı istediği oran Durum 1.1'le aynıdır. Bu durum için elde edilen sonuçlar Şekil 5.4 ve Şekil 5.5'te verilmiştir. 1. durumda olduğu gibi ilk 50 saniye içinde her iki nodda da kuyruk oluşmaktadır. Ancak, gecikmelerin bir önceki duruma göre daha büyük olması nedeniyle, iletim oranı komut grafiklerinde, salınımlar gözlenmektedir. Kuyrukların sıfır olduğu zaman aralıklarında, denetleyicilerin hesapladığı komutlar, eski değerlerinde sabit kalmak için sönümlü salınımlar göstermektedir. 1. nodun iletim oranı komut grafiğinden de görüldüğü gibi, komuttaki salınımlar tam olarak sönümlenmeden

i	h_i	δ_i^+	β_i	β_i^f
1	3	4	0.3	0.03
2	3	4	0.4	0.04

Çizelge 5.3: Durum 1.2'de Kullanılan Tasarım Parametreleri

i	h_i^b	δ_i^b	h_i^f	δ_i^f
1	2.5	$0.4\sin(\frac{2\pi}{50}t)$	0.5	$0.2\sin(\frac{2\pi}{50}t)$
2	1.5	$0.4\sin(\frac{2\pi}{50}t)$	1.5	$0.2\sin(\frac{2\pi}{50}t)$

Çizelge 5.4: Durum 1.2'de Kullanılan Gerçekleme Parametreleri

kuyruk uzunluğunda yeni bir değişim olduğunda, iletim oranı komutu eski değerinde sabitlenemeden yeni bir değere ulaşmaktadır. Sonuçta, kaynağın gerçek iletim oranının, 1. nodun hesapladığı oran değeri daha küçük olduğu için bu değere ayarlandığı görülmektedir. Kaynağın yeni iletim oranı, 2. nodun kapasitesinin altında kaldığı için, bir süre sonra bu nodun kuyruk uzunluğu sıfır olur. Ayrıca, 2. noddaki veri işleme hızı grafiği incelendiğinde, bu oranın, 2. nodun kuyruğunun sıfırdan büyük olduğu anlarda nodun kapasitesine, sıfıra eşit olduğu anlarda da kapasite ile gelen veri oranının minimumuna eşit olduğu görülmektedir.

Durum 1.3 Bu durum için denetleyici tasarımında ve gerçeklemede kullanılan parametre değerleri ve nodlar için ulaşılmak istenen kuyruk uzunluğu değerleri Durum 1.1'de kullanılan değerlerle aynıdır. Nodların çıkış kapasiteleri ve kaynağın iletim yapmayı istediği oran Şekil 5.6'da verilmiştir. Bu durum için elde edilen sonuçlar Şekil 5.6'da ve Şekil 5.7'de verilmiştir. Burada, kaynağın iletim yapmak istediği oran, 50 p/s değerinin üzerine eklenen ortalaması 0 ve varyansı 5 olan bir Gaussian sinyalidir. İlk 40 saniyede, 1. nodun kapasitesi kaynağın iletim oranı isteminden daha büyüktür. Dolayısıyla, bu nodda bu süre içinde kuyruk oluşmayacaktır. Ancak, aynı zaman aralığında, 2. nodun kapasitesi kaynağın iletim yapmak istediği orandan daha küçük olduğu için bu nodda kuyruk oluşur. Kuyruğun oluşmasıyla,

2. noddaki denetleyici iletim oranı komutu hesaplar. Bu oran kaynağın iletim yapmak istediği orandan daha küçük olduğu için yeni iletim oranı olarak atanır. Bu arada, 1. nodda kuyruk oluşmamış olduğu için denetleyici herhangi bir oran hesaplamaz. Kaynağın iletim oranının düşürülmesiyle, 2. noddaki kuyruk da sıfırlanır. Bu kuyruk uzunluğu azaldıkça, denetleyicinin hesapladığı oranda artış gözlenir. Kaynağın iletim oranını azaltan bu komut olduğu için bu komutun değerindeki artış yeni iletim oranının kaynağın istediği orana ayarlanmasını sağlar. 1. nodun kapasitesindeki azalma, kaynağın iletim oranının tekrar azaltılmasına neden olur. 1. noddaki kuyruk istenen değerine ulaştığında, denetleyicinin hesapladığı oran sabit kalır.

Durum 1.4 Bu durumda kullanılan parametre değerleri 1. durumdakilerle aynıdır. Kaynağın iletim yapmak istediği oran 70 p/s olarak alınmıştır. Elde edilen sonuçlar ve nodların çıkış kapasiteleri Şekil 5.8 ve Şekil 5.9'da verilmiştir. Burada, kaynağın iletim yapmak istediği oran, nodların kapasitelerinden büyük olduğu için her iki nodda da kuyruk oluşumu beklenir. Nodların kapasiteleri birer sinüs fonksiyonu olduğu için periyodik olarak artıp azalan kuyruk uzunlukları oluşur. Kuyruk varken hesaplanan iletim oranı komutları, kuyruklar sıfırlandığında eski değerlerinde sabit kalırlar. Hesaplanan komutların değerleri ve kaynağın istemi göz önüne alındığında, kaynağın gerçek iletim oranının beklendiği gibi gerçekleştiği görülmektedir. Kuyruk uzunluğu grafikleri incelendiğinde, 1. noddan 2. noda gönderilen verinin oranı ve 2. noddaki veri işleme hızının beklendiği gibi gerçekleştiği görülmektedir.

Durum 1.5 Bu durumda kullanılan parametre değerleri 1. durumdakilerle aynıdır. Kaynağın iletim yapmak istediği oran ve nodların çıkış kapasiteleri Şekil 5.10'da verilmiştir. Elde edilen sonuçlar ise Şekil 5.10 ve 5.11'de verilmiştir. Burada, kaynağın, sürekli olarak veri göndermediği, belli aralıklarla aktif olduğu kabul edilmiştir. Bu

oran ve nodların çıkış kapasiteleri göz önüne alındığında, kuyruk uzunluklarının beklenen zaman aralıklarında oluştuğu görülmektedir. Ayrıca, denetleyicilerin hesapladığı iletim oranı komutları, kuyruk uzunluklarındaki değişimlere uygun olarak hesaplanmıştır. Dolayısıyla, kaynağın gerçek iletim oranı da beklendiği gibi gerçekleşmiştir.

5.2 Bir Kaynak ve Üç Noddan Oluşan Sistem için Yapılan Benzetim Çalışmalarında Elde Edilen Sonuçlar

Bu bölümde ele alınan ağ yapısı, Şekil 4.1'de verilen modelin, tek bir kaynak–hedef çifti ve 2 ara noddan oluşan şeklidir. Bu sistem için yapılan benzetim çalışmasında kullanılan SIMULINK diyagramı Ek–7'de verilmiştir.

Durum 2.1 Bu durum için denetleyici tasarımında kullanılan parametre değerleri Çizelge 5.5'te, gerçeklemede kullanılan değerler ise Çizelge 5.6'da verilmiştir. Nodlar için ulaşılmak istenen kuyruk uzunluğu değeri $q_{d_1} = 100$ paket, 2. nod için $q_{d_2} = 150$ paket, ve 3. nod için $q_{d_3} = 125$ paket olarak alınmıştır. Kaynağın iletim yapmayı istediği oran ve nodların çıkış kapasiteleri Şekil 5.12'de verilmiştir. Bu durum için elde edilen sonuçlar Şekil 5.12, Şekil 5.13 ve Şekil 5.14'te verilmiştir. Nodlardaki kuyruk uzunluğu grafiklerinde gözlenen, kuyruğun sıfır olduğu zaman aralıkları, nodlara gelen toplam veri iletim oranının nodun çıkış kapasitesini aşması için geçmesi gereken zaman aralıklarıdır. Tüm nodlar için kuyruk uzunluğu ve iletim oran komutu grafikleri birlikte incelendiğinde, kuyruk uzunluklarının ulaşılmak istenen değerlerin üzerine çıkması durumunda, iletim oranı komut değerlerinin azaldığı, kuyruk azalmaya başladığında da bu değerlerin artma eğiliminde olduğu görülmektedir. Kuyruk uzunluklarının sıfır olması, gelen tüm verilerin çıkışa aktarılması anlamına gelir. Dolayısıyla, bu durumda, denetleyicilerin yeni bir iletim oranı hesaplamalarına gerek yoktur. Bu komutlara ait grafiklerinden de görüldüğü gibi böyle zaman aralıklarında denetleyicilerin hesapladığı oranlar eski değerlerinde sabit kalmaktadır. Ayrıca,

i	h_i	δ_i^+	β_i	β_i^f
1	1	4	0.1	0.01
2	1	4	0.2	0.02
3	1	4	0.3	0.03

Çizelge 5.5: Durum 2.1, 2.3, 2.4 ve 2.5'te Kullanılan Tasarım Parametreleri

i	h_i^b	δ_i^b	h_i^f	δ_i^f
1	0.9	$0.055\sin(\frac{2\pi}{50}t)$	0.1	$0.015\sin(\frac{2\pi}{50}t)$
2	0.5	$0.035\sin(\frac{2\pi}{50}t)$	0.5	$0.035\sin(\frac{2\pi}{50}t)$
3	0.2	$0.15\sin(\frac{2\pi}{50}t)$	0.8	$0.055\sin(\frac{2\pi}{50}t)$

Çizelge 5.6: Durum 2.1, 2.3, 2.4 ve 2.5'te Kullanılan Gerçekleme Parametreleri

kaynağın iletim oranının beklendiği gibi tüm oran komutları ve kaynağın iletim yapmak istediği oran arasından en küçüğüne eşit olduğu da görülmektedir. Nodlar arasında gönderilen verilerin oranının ve hedef nodda, 3, veri işleme hızı da beklendiği gibi gerçekleşmiştir.

Durum 2.2 Bu durum için denetleyici tasarımında kullanılan parametre değerleri Çizelge 5.7'de, gerçeklemede kullanılan değerler ise Çizelge 5.8'de verilmiştir. Nodlar için ulaşılmak istenen kuyruk uzunluğu değerleri, kaynağın iletim yapmak istediği oran ve nodların çıkış kapasiteleri bir önceki durumla aynıdır. Bu durum için elde edilen sonuçlar Şekil 5.15 ve Şekil 5.16'da verilmiştir. Sonuçlar 1. durumla karşılaştırıldığında, tüm nodlar için kuyruk uzunluklarındaki değişimlerin yaklaşık aynı zaman aralıklarında oluştuğu görülmektedir. Ancak gecikmelerin bir önceki duruma göre daha büyük olması ve gerçeklemelerdeki gecikmelerin büyüklüklerindeki artışların tasarımda kullanılanlardan daha fazla olması nedeniyle iletim oranı komut grafiklerinde gözlenen salınımların hem sayıları hem de büyüklükleri artmıştır. Kuyrukların sıfır olduğu zaman aralıklarında, denetleyicilerin hesapladığı komutlar, eski değerlerinde sabit kalmak için sönümlü salınımlar göstermektedir. 1. ve 3.

i	h_i	δ_i^+	β_i	β_i^f
1	3	4	0.3	0.03
2	3	4	0.4	0.04
3	3	4	0.5	0.05

Çizelge 5.7: Durum 2.2'de Kullanılan Tasarım Parametreleri

i	h_i^b	δ_i^b	h_i^f	δ_i^f
1	2.5	$0.6\sin(\frac{2\pi}{50}t)$	0.5	$0.3\sin(\frac{2\pi}{50}t)$
2	1.5	$0.5\sin(\frac{2\pi}{50}t)$	1.5	$0.4\sin(\frac{2\pi}{50}t)$
3	1	$0.3\sin(\frac{2\pi}{50}t)$	2	$0.6\sin(\frac{2\pi}{50}t)$

Çizelge 5.8: Durum 2.2'de Kullanılan Gerçekleme Parametreleri

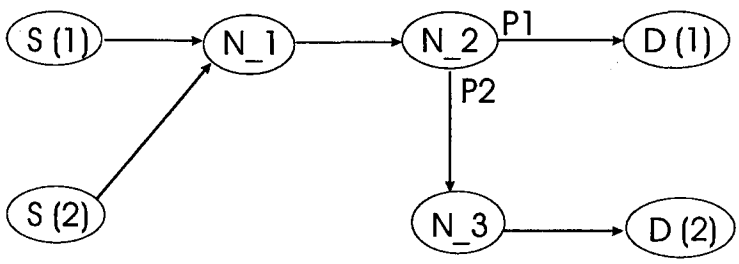
nodların iletim oranı komut grafiklerinde görüldüğü gibi, komutlardaki salınımlar tam olarak sönümlenmeden kuyruk uzunluklarında yeni bir değişim olduğunda, iletim oranı komutları eski değerinde sabitlenemeden yeni bir değere ulaşmaktadır. Dolayısıyla, bu salınımlar, sistemin yatışkan duruma daha geç ulaşmasına yol açmaktadır. Oran komutlarında gözlenen bu salınımlar, kaynağın gerçek iletim oranına da yansımıştır.

Durum 2.3 Bu durumda denetleyici tasarımında ve gerçeklemede kullanılan parametre değerleri ile tüm nodlar için ulaşılmak istenen kuyruk uzunluğu değerleri 1. durumla aynıdır. Kaynağın iletim yapmayı istediği oran Şekil 5.17'de verilmiştir. Bu durum için elde edilen sonuçlar ise Şekil 5.17 ve Şekil 5.18'de verilmiştir. Burada kaynağın iletim yapmak istediği oran, 60 p/s değerinin üzerine ortalaması 0 ve varyansı 40 olan bir Gaussian sinyali eklenerek elde edilmiştir. Grafiklerden de görüldüğü gibi, nodlarda, beklenen zaman aralıklarında kuyruklar oluşmuştur. Kaynağın iletim yapmak istediği oranda gelişigüzel meydana gelen ani değişimler, 1. nodda hesaplanan oran komutunu oldukça etkilemiştir. Çünkü, 0–100 s. arasında, her ne kadar kuyruk uzunluğu grafiğinde belli olmasa da, 1. nodda uzunluğu 2 paketi geçmeyen kuyruklar oluşup

sıfırlanmaktadır. Kuyrukta meydana gelen bu küçük büyüklükteki ani değişimler, denetleyicinin sürekli değişen, büyük değerli oran komutu hesaplamasına neden olur ki bu durum ilgili grafikten de gözlenebilir. Diğer nodlarda oluşan kuyruklar bu şekilde ani değişimler göstermediği için bu nodlardaki denetleyicilerin hesapladığı oranlar daha düzgündür. Kaynağın iletim yapmak istediği oranda gözlenen ani değişimlerin değerlerinin hesaplanan komutlardan daha küçük olduğu anlarda, bu değişimlerin değerleri kaynağın gerçek iletim oranı olur. Bu durum kaynağın gerçek iletim oranının grafiğinden de görülmektedir. Burada, nodların çıkış kapasiteleri Durum 2.1'de kullanılan kapasite değerleriyle aynı alınmıştır.

Durum 2.4 Bu durumda kullanılan parametre değerleri 1. durumdakilerle aynıdır. Kaynağın iletim yapmak istediği oran 70 p/s olarak alınmıştır. Elde edilen sonuçlar ve nodların çıkış kapasiteleri Şekil 5.19, Şekil 5.20 ve Şekil 5.21'de verilmiştir. Burada kaynağın iletim yapmak istediği oran nodların kapasitelerinden büyük olduğu için tüm nodlarda kuyruk oluşumu beklenir. Nodların çıkış kapasiteleri birer sinüs fonksiyonu olduğu için, nodlarda hemen hemen periyodik olarak, artıp azalan ve sıfır olan kuyruk uzunlukları oluşur. Kuyruk varken hesaplanan iletim oranı komutları, kuyruklar sıfırlandığında eski değerlerinde sabit kalırlar. Hesaplanan komutların değerleri ve kaynağın istemi göz önüne alındığında, kaynağın gerçek iletim oranının beklendiği gibi gerçekleştiği görülmektedir. Dolayısıyla, nodlar arası veri alış verişlerindeki iletim hızları da beklendiği gibi gerçekleşmiştir.

Durum 2.5 Bu durumda kullanılan parametre değerleri 1. durumdakilerle aynıdır. Kaynağın iletim yapmak istediği oran Şekil 5.22'de verilmiştir. 1. ve 3. nodların çıkış kapasiteleri Durum 2.1 ile aynıyken, 2. nodun çıkış kapasitesi Şekil 5.22'de verilmiştir. Elde edilen sonuçlar ise Şekil 5.22 ve Şekil 5.23'te verilmiştir. Burada, kaynağın, sürekli olarak veri göndermediği, belli aralıklarla aktif



Şekil 5.1: Gerçeklemesi yapılan ağ modeli

olduğu kabul edilmiştir. Bu oran ve nodların çıkış kapasiteleri göz önüne alındığında, kuyruk uzunluklarının beklenen zaman aralıklarında oluştuğu görülmektedir. Örneğin, veri gönderilmeyen ilk 25 saniyede kuyruk uzunlukları sıfırdır. Veri gönderiminin başlaması ve nodlarda kuyruk oluşmasıyla birlikte denetleyiciler de komut hesaplamaya başlar. Kuyruk uzunlukları tekrar sıfırlandığında, hesaplanan komutlar eski değerlerinde sabitlenir. Dolayısıyla, kaynağın gerçek iletim oranı da bu komutlar ve istenen iletim oranı bilgilerine göre beklendiği gibi elde edilmiş ve sistem yatışkan duruma ulaşmıştır.

5.3 İki Kaynak–Hedef Çifti ve Ara Nodlardan Oluşan Sistem için Yapılan Benzetim Çalışmalarında Elde Edilen Sonuçlar

Bu sistem için yapılan benzetim çalışmasında ele alınan ağ modeli Şekil 5.1'de, SIMULINK diyagramı ise Ek–8'de verilmiştir. Şekil 5.1'deki P_1 ve P_2, sırasıyla, 2. nodun 1. ve 2. çıkış portlarını göstermektedir. Tüm durumlar için $q_{d,1} = q_{d,3} = q_{d,d_2} = 100$ p ve $q_{d,2_1} = q_{d,2_2} = q_{d,d_1} = 150$ p olarak alınmıştır. Bağlantıların ağda izlediği yollar şu şekilde ifade edilebilir;

$p(1) = \{N_{1_1}, N_{2_1}, d_{1_1}\}$
$p(2) = \{N_{1_1}, N_{2_2}, N_{3_1}, d_{2_1}\}$

Burada, 1. ve 2. kaynaklar 1. nodda aynı çıkış portunu, 2. nodda ise farklı portları (1. bağlantıya ait veriler 1. port üzerinden bu bağlantının hedef

noduna, 2. bağlantıya ait veriler ise 2. port üzerinden 3. noda gönderilmektedir) kullanmaktadırlar. Şekillerin alt yazılarında kullanılan notasyonda, sadece bir çıkış portu olan nodların gösterimi için i_k yerine, alt indis k ihmal edilerek, i kullanılmıştır.

Durum 3.1 Bu durum için denetleyici tasarımında kullanılan parametre değerleri Çizelge 5.9'da, gerçeklemede kullanılan değerler ise Çizelge 5.10'da verilmiştir. Kaynakların iletim yapmak istedikleri oranlar ve nodların çıkış kapasiteleri Şekil 5.24 ve Şekil 5.25'te verilmiştir. 2. kaynağın hedef nodunun çıkış kapasitesi 35 p/s olarak alınmıştır. Bu durum için elde edilen sonuçlar Şekil 5.25, Şekil 5.26, ve Şekil 5.27'de verilmiştir. Grafiklerden de görüldüğü gibi portlardaki denetleyiciler, portlarda kuyruk oluştuğu andan itibaren iletim oran komutu hesaplamaktadırlar. Portlarda oluşan kuyruk uzunluklarının istenen değerlerine ulaşaması ve artış göstermesi durumunda hesaplanan oran komutlarının değerlerinin azalma gösterdiği, kuyruk uzunluklarının 1'e düşmesi durumunda komutların sabitlendiği gözlenmektedir. Portlardan kaynaklara gönderilen oran grafikleri incelendiğinde ise kuyruk uzunluklarının değerlerinin 1'den büyük olduğu zaman aralıklarında denetleyicilerin hesapladıkları oranların, diğer aralıklarda ise kaynakların iletim yapmak istedikleri oranların kaynaklara gönderildiği görülmektedir. Bazı portlarda oluşan kuyruklar istenen değerlerinde sabitlenmeden alt sınır değeri olarak seçilen 1'e düşmüştür. Bu durumda da hesaplanan oran komutlarında beklenen artış ve azalışların gerçekleştiği görülmektedir. Portlar arası veri gönderim oranlarının da portların çıkış kapasiteleri ile sınırlı olduğu görülmektedir. Ayrıca, 1. ve 2. nodların portlarındaki denetleyicilerin 1. ve 2. kaynak için hesapladıkları iletim oran komutlarının yatışkan durum değerleri arasındaki oranın ağırlıklı eşitlik katsayılarının oranına eşit olduğu da görülmektedir. Oran komutlarında, dolayısıyla, kaynakların gerçek iletim oranları ve nodlar arası veri gönderim oranlarında gözlenen salınımların temel nedeni, portlarda oluşan kuyruk

i_k, j	$h^j_{i_k}$	$\delta^{j,+}_{i_k}$	$\alpha^j_{i_k}$	$\beta^j_{i_k}$	$\beta^{j,f}_{i_k}$
$1_1, 1$	1.5	2	0.2	0.1	0.01
$1_1, 2$	2.5	2.5	0.8	0.2	0.02
$2_1, 1$	1.5	2	1	0.1	0.01
$2_2, 2$	2.5	2.5	1	0.2	0.02
$3_1, 2$	2.5	2.5	1	0.1	0.01
$d_{11}, 1$	1.5	2	1	0.3	0.03
$d_{21}, 2$	2.5	2.5	1	0.4	0.04

Çizelge 5.9: Durum 3.1, 3.3 ve 3.4'te Kullanılan Tasarım Parametreleri

i_k, j	$h^{j,b}_{i_k}$	$\delta^{j,b}_{i_k}$	$h^{j,f}_{i_k}$	$\delta^{j,f}_{i_k}$
$1_1, 1$	1.3	$0.06\sin(\frac{2\pi}{50}t)$	0.2	$0.03\sin(\frac{2\pi}{50}t)$
$1_1, 2$	2.2	$0.06\sin(\frac{2\pi}{50}t)$	0.3	$0.06\sin(\frac{2\pi}{50}t)$
$2_1, 1$	1	$0.06\sin(\frac{2\pi}{50}t)$	0.5	$0.06\sin(\frac{\pi}{50}t)$
$2_2, 2$	1.9	$0.03\sin(\frac{2\pi}{50}t)$	0.6	$0.03\sin(\frac{\pi}{50}t)$
$3_1, 2$	1.5	$0.03\sin(\frac{2\pi}{50}t)$	1	$0.03\sin(\frac{\pi}{50}t)$
$d_{11}, 1$	0.75	$0.03\sin(\frac{2\pi}{50}t)$	0.75	$0.06\sin(\frac{\pi}{50}t)$
$d_{21}, 2$	1.25	$0.03\sin(\frac{2\pi}{50}t)$	1.25	$0.03\sin(\frac{\pi}{50}t)$

Çizelge 5.10: Durum 3.1, 3.3 ve 3.4'te Kullanılan Gerçekleme Parametreleri

uzunluklarında kısa zaman aralıkları içinde meydana gelen artıp azalmalardır. Küçük değerli kuyruk uzunlukları, istenen değerlere ulaşılabilmesi için denetleyicilerin büyük değerli oran komutları hesaplamasına neden olur. Böylesi kuyruklar kısa süre içinde boşaltılabildiği zaman denetleyicilerin hesapladığı oranların azalması ve sabitlenmesi gerekir. Bu tür değişimlerin fazla gözlenmediği kuyruk uzunlukları sonucunda hesaplanan oran komutlarında bu tür salınımların fazla gözlenmediği elde edilen sonuçlardan da görülebilmektedir. Kaynakların iletim oranlarının yatışkan durumdaki değerleri ağırlıklı eşitlik katsayıları arasındaki oranı sağlamaktadır.

Durum 3.2 Bu durumda kullanılan parametre değerleri Çizelge 5.11'de,

gerçeklemede kullanılan değerler ise Çizelge 5.12'de verilmiştir. Kaynakların iletim yapmak istedikleri oranlar ve nodların çıkış kapasiteleri ise Durum 3.1'de kullanılan değerlerle aynı alınmıştır. 2. nodun çıkış portlarının kapasiteleri ise Şekil 5.28'de verilmiştir. Bu durum için elde edilen sonuçlar Şekil 5.28, Şekil 5.29, Şekil 5.30 ve Şekil 5.31'de verilmiştir. Burada, sistemdeki zaman gecikmeleri bir önceki duruma göre daha büyüktür ve gerçeklemelerdeki gecikmelerin büyüklüklerindeki artışlar tasarımda yapılan artışlardan daha fazladır. Elde edilen sonuçlar 1. durumla karşılaştırıldığında, bu artışların, kuyruk uzunluklarında daha fazla salınım gözlenmesine; dolayısıyla da, iletim oranı komut grafiklerinde daha fazla sayıda ve büyüklükte salınımların gözlenmesine neden olmaktadır. Bu salınımlar da sistemin yatışkan duruma daha geç ulaşmasına yol açmaktadır.

Durum 3.3 Bu durum için kaynakların iletim yapmak istedikleri oranlar ve nodların çıkış kapasiteleri Şekil 5.32'de verilmiştir. 1. kaynağın hedef nodunun çıkış portunun kapasitesi Durum 3.1'de alınan değerlerle aynı alınmışken 2. kaynağın hedef nodunun çıkış portunun kapasitesi 45 p/s alınmıştır. Elde edilen sonuçlar ise Şekil 5.32, Şekil 5.33, Şekil 5.34 ve Şekil 5.35'te verilmiştir. Burada, her iki kaynağın da sürekli olarak veri göndermedikleri, belli aralıklarla aktif oldukları kabul edilmiştir. Toplam veri oranı ve nodların çıkış kapasiteleri göz önüne alındığında, kuyruk uzunluklarının beklenen zaman aralıklarında oluştuğu görülmektedir. Veri gönderiminin başlaması ve nodlarda kuyruk oluşmasıyla birlikte denetleyiciler de komut hesaplamaya başlar. Kuyruk uzunlukları 1'in altına düştüğünde, hesaplanan komutlar eski değerlerinde sabitlenir. Dolayısıyla, kaynağın gerçek iletim oranı da bu komutlar ve istenen iletim oranı bilgilerine göre beklendiği gibi elde edilmiş ve sistem yatışkan duruma ulaşmıştır. Bu durum için de ağırlıklı eşitlik her iki kaynak için de sağlanmaktadır.

Durum 3.4 Bu durum için denetleyici tasarımında ve gerçeklemede alınan

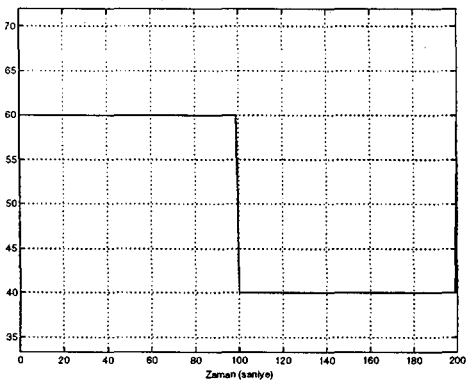
i_k, j	$h^j_{i_k}$	$\delta^{j,+}_{i_k}$	$\alpha^j_{i_k}$	$\beta^j_{i_k}$	$\beta^{j,f}_{i_k}$
$1_1, 1$	2	2	0.2	0.1	0.01
$1_1, 2$	3	3	0.8	0.2	0.02
$2_1, 1$	2	2	1	0.1	0.01
$2_2, 2$	3	2	1	0.2	0.02
$3_1, 2$	3	2	1	0.1	0.01
$d_{11}, 1$	2	2	1	0.3	0.03
$d_{21}, 2$	3	3	1	0.4	0.04

Çizelge 5.11: Durum 3.2'de Kullanılan Tasarım Parametreleri

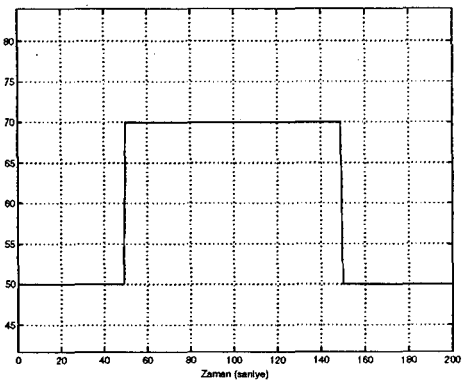
i_k, j	$h^{j,b}_{i_k}$	$\delta^{j,b}_{i_k}$	$h^{j,f}_{i_k}$	$\delta^{j,f}_{i_k}$
$1_1, 1$	1.8	$0.06\sin(\frac{2\pi}{50}t)$	0.2	$0.03\sin(\frac{\pi}{50}t)$
$1_1, 2$	2.7	$0.03\sin(\frac{2\pi}{50}t)$	0.3	$0.03\sin(\frac{\pi}{50}t)$
$2_1, 1$	1.45	$0.06\sin(\frac{2\pi}{50}t)$	0.55	$0.06\sin(\frac{\pi}{50}t)$
$2_2, 2$	2.45	$0.03\sin(\frac{2\pi}{50}t)$	0.55	$0.03\sin(\frac{\pi}{50}t)$
$3_1, 2$	2.35	$0.03\sin(\frac{2\pi}{50}t)$	0.65	$0.03\sin(\frac{\pi}{50}t)$
$d_{11}, 1$	1.5	$0.03\sin(\frac{2\pi}{50}t)$	0.5	$0.06\sin(\frac{\pi}{50}t)$
$d_{21}, 2$	2.75	$0.03\sin(\frac{2\pi}{50}t)$	0.25	$0.03\sin(\frac{\pi}{50}t)$

Çizelge 5.12: Durum 3.2'de Kullanılan Gerçekleme Parametreleri

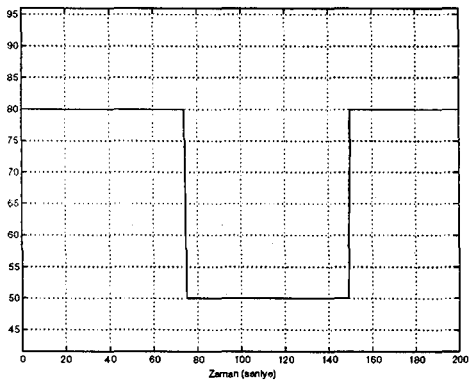
gecikme değerlerinin büyüklükleri Durum 3.1'de kullanılan değerlerle aynıdır. Bu durum için 1. kaynağın iletim yapmak istediği oran ve nodların çıkış kapasiteleri Şekil 5.36'da verilmiştir. 2. kaynağın iletim yapmak istediği oran Durum 3.1'dekiyle aynıdır. Ayrıca, 1. nodun çıkış kapasitesi de Durum 3.2'de bu nod için kullanılan kapasite değeri ile aynı ve 2. kaynağın hedef nodunun kapasitesi $c_{d_2}(t) = 30$ p/s olarak alınmıştır. Elde edilen sonuçlar ise Şekil 5.36, Şekil 5.37, Şekil 5.38 ve Şekil 5.39'da verilmiştir. İlk iki nodun portlarının çıkış kapasitelerinin değerleri ile kaynakların iletim oranı istemlerinin değerleri arasında çok büyük fark yoktur. Dolayısıyla, Her iki nodda da az sayıda paket kuyruğa girmekte ve kısa süre içersinde de iletilmektedir. Ayrıca, 1. kaynağın iletim yapmak istediği oranda gelişigüzel meydana gelen değişimlerin büyüklükleri fazla olmasa da nodların kuyruk uzunluklarında gözlenen değişimlerin oluşmasına etkendir. Kuyruk uzunluklarında gözlenen bu değişimlerin etkisi hesaplanan oran komutlarından da görülmektedir. Kaynakların iletim yapmak istediği oran değerlerinin hesaplanan komutlardan daha küçük olduğu anlarda, bu değerleri kaynakların gerçek iletim oranı olur. Bu durum kaynakların gerçek iletim oran grafiklerinden de görülmektedir. Bu durum için de ağırlıklı eşitlik şartları her iki kaynak için de sağlanmaktadır. Bunun yanı sıra nodlar arası veri gönderim oranları nodların çıkış kapasiteleri ile sınırlı kalmaktadır.



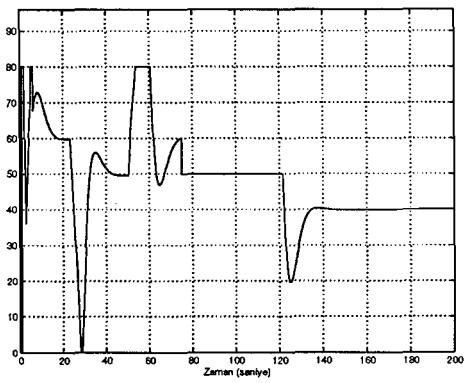
1. nodun çıkış kapasitesi, $c_1(t)$, p/s



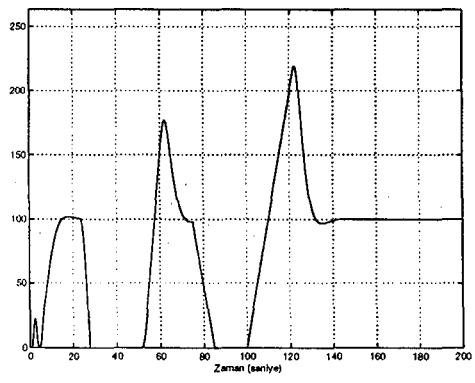
2. nodun çıkış kapasitesi, $c_2(t)$, p/s



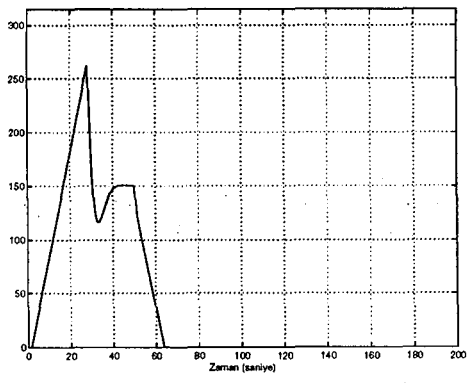
Kaynağın iletim yapmak istediği oran, $r^0(t)$, p/s.



Kaynağın gerçek iletim oranı, $r^s(t)$, p/s.

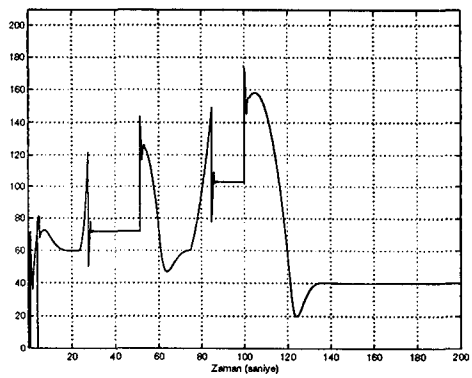


1. nodun kuyruk uzunluğu, $q_1(t)$, p.

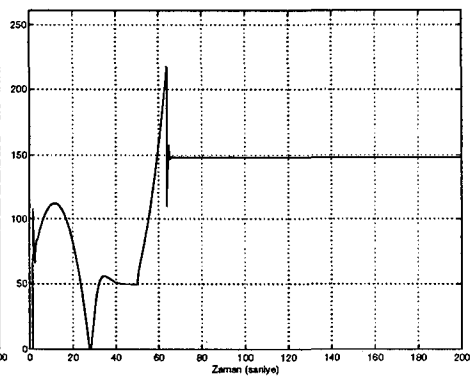


2. nodun kuyruk uzunluğu, $q_2(t)$, p.

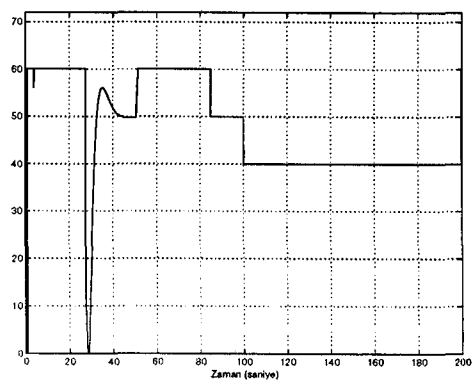
Şekil 5.2: Durum 1.1 için elde edilen sonuçlar



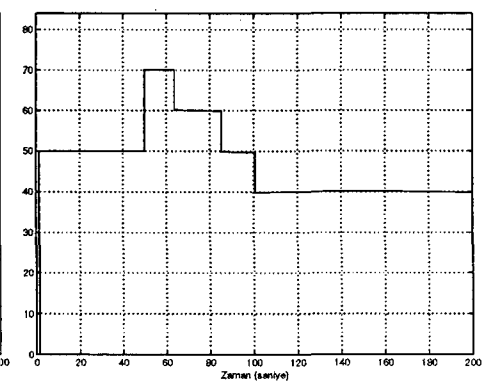
1. nodda hesaplanan iletim oranı komutu, $r_1(t)$, p/s



2. nodda hesaplanan iletim oranı komutu, $r_2(t)$, p/s

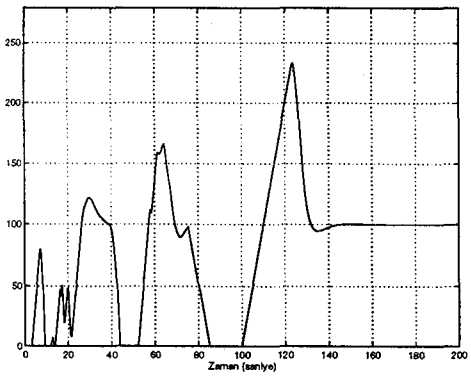


1. noddan 2. noda veri gönderim oranı, $\rho_{12}^s(t)$, p/s.

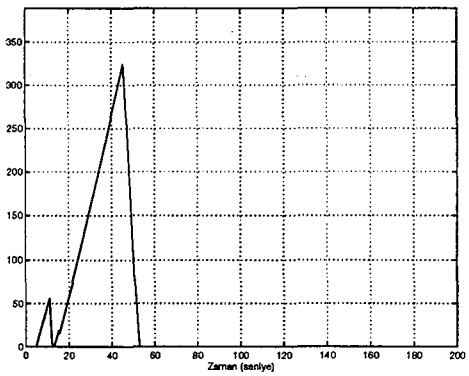


Hedef nodda veri işleme hızı, $\rho_2^s(t)$, p/s.

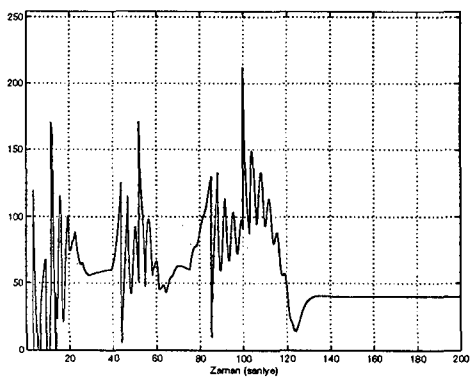
Şekil 5.3: Durum 1.1 için elde edilen sonuçlar (devam)



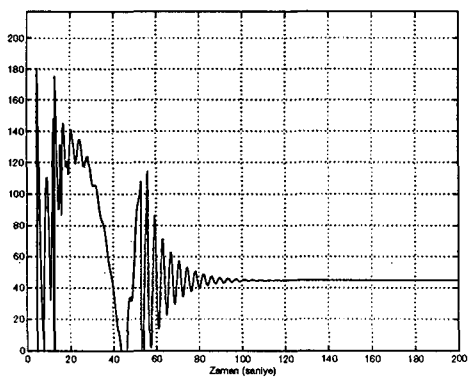
1. nodun kuyruk uzunluğu, $q_1(t)$, p.



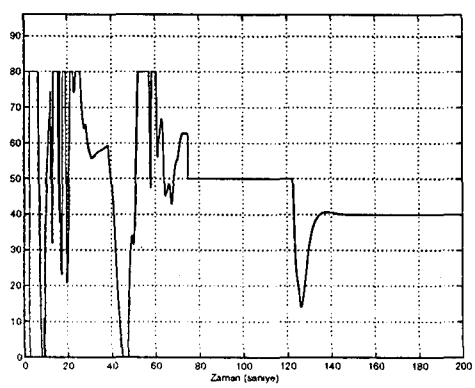
2. nodun kuyruk uzunluğu, $q_2(t)$, p.



1. nodda hesaplanan iletim oranı komutu, $r_1(t)$, p/s

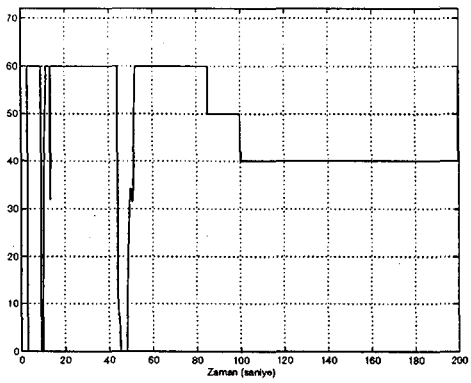


2. nodda hesaplanan iletim oranı komutu, $r_2(t)$, p/s

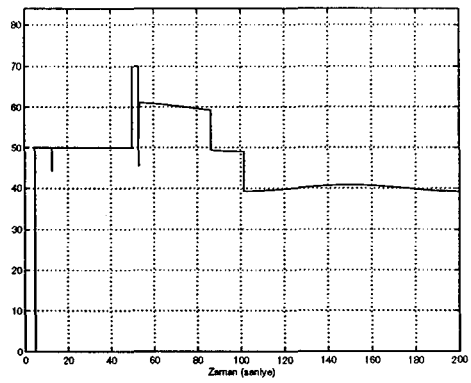


Kaynağın gerçek iletim oranı, $r^s(t)$, p/s.

Şekil 5.4: Durum 1.2 için elde edilen sonuçlar

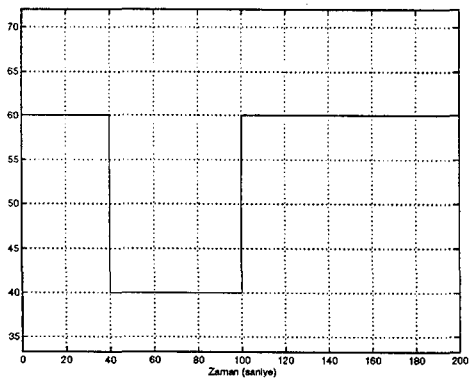


1. noddan 2. noda veri gönderim oranı, $\rho_{12}^{s}(t)$, p/s.

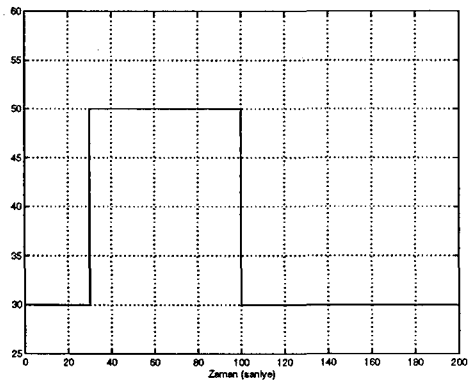


Hedef nodda veri işleme hızı, $\rho_{2}^{s}(t)$, p/s.

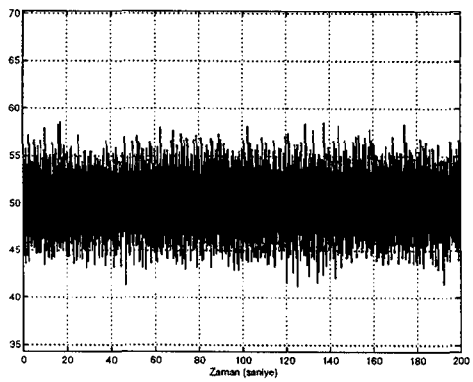
Şekil 5.5: Durum 1.2 için elde edilen sonuçlar (devam)



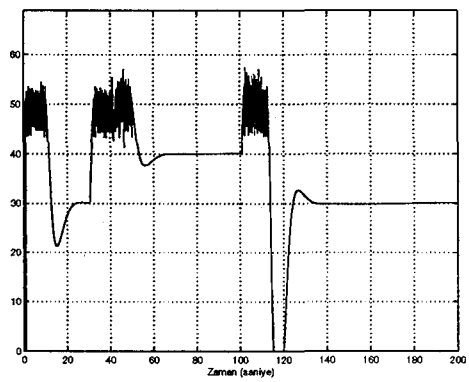
1. nodun çıkış kapasitesi, $c_1(t)$, p/s



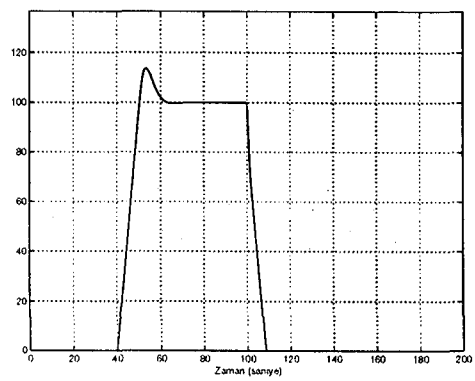
2. nodun çıkış kapasitesi, $c_2(t)$, p/s



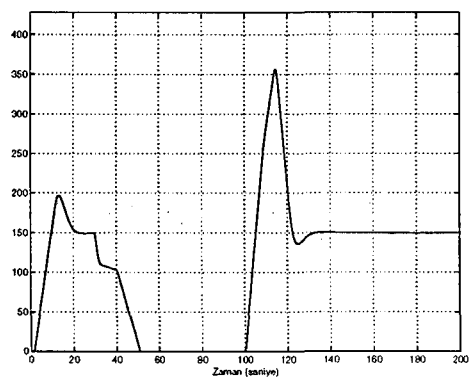
Kaynağın iletim yapmak istediği oran, $r^0(t)$, p/s.



Kaynağın gerçek iletim oranı, $r^s(t)$, p/s.

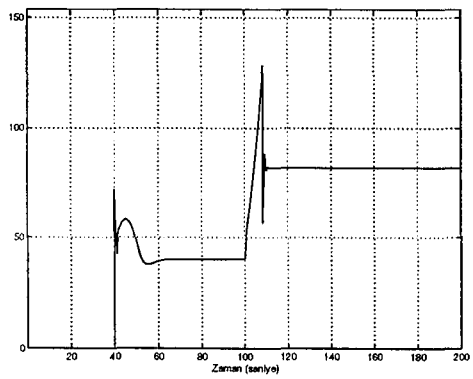


1. nodun kuyruk uzunluğu, $q_1(t)$, p.

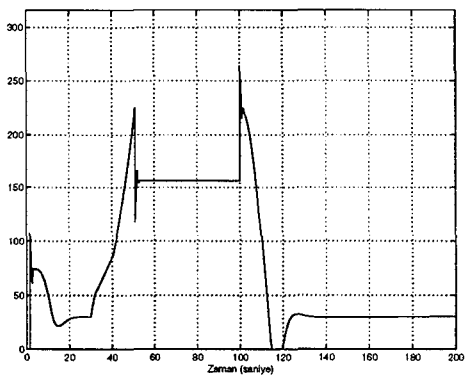


2. nodun kuyruk uzunluğu, $q_2(t)$, p.

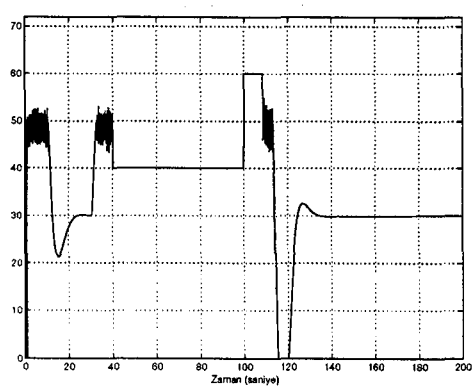
Şekil 5.6: Durum 1.3 için elde edilen sonuçlar



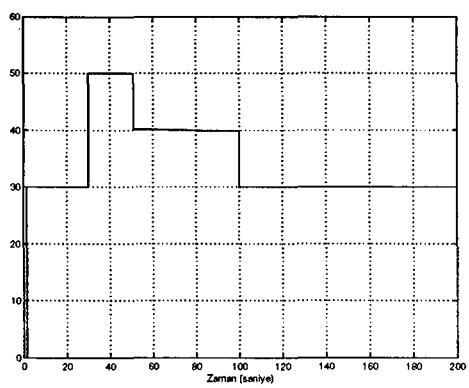
1. nodda hesaplanan iletim oranı komutu, $r_1(t)$, p/s



2. nodda hesaplanan iletim oranı komutu, $r_2(t)$, p/s

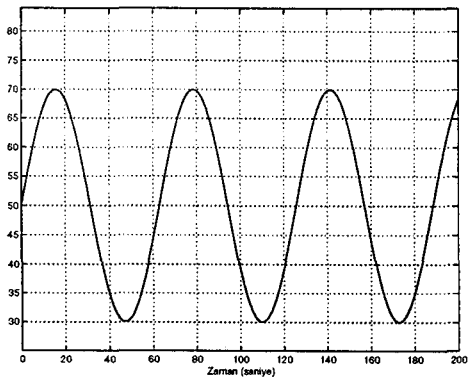


1. noddan 2. noda veri gönderim oranı, $\rho_{12}^s(t)$, p/s.

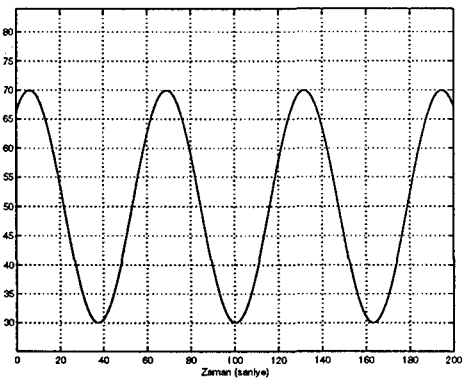


Hedef nodda veri işleme hızı, $\rho_2^s(t)$, p/s.

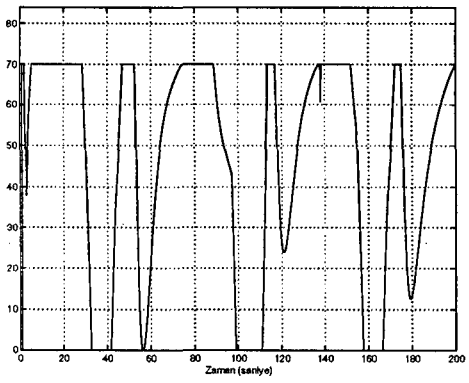
Şekil 5.7: Durum 1.3 için elde edilen sonuçlar (devam)



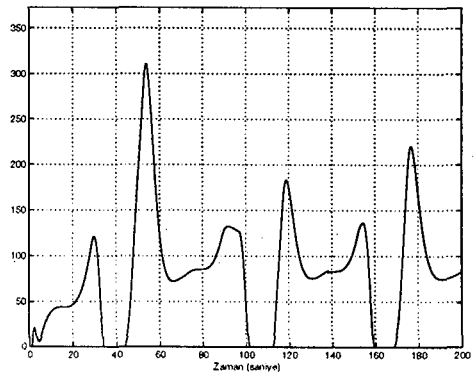
1. nodun çıkış kapasitesi, $c_1(t)$, p/s



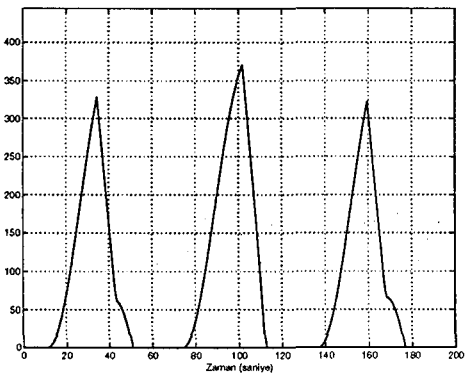
2. nodun çıkış kapasitesi, $c_2(t)$, p/s



Kaynağın gerçek iletim oranı, $r^s(t)$, p/s.

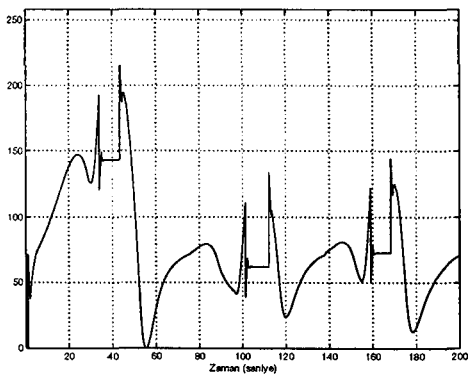


1. nodun kuyruk uzunluğu, $q_1(t)$, p.

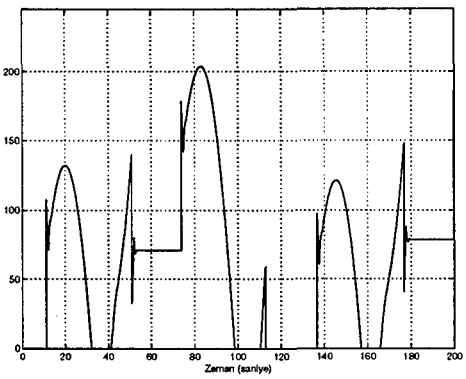


2. nodun kuyruk uzunluğu, $q_2(t)$, p.

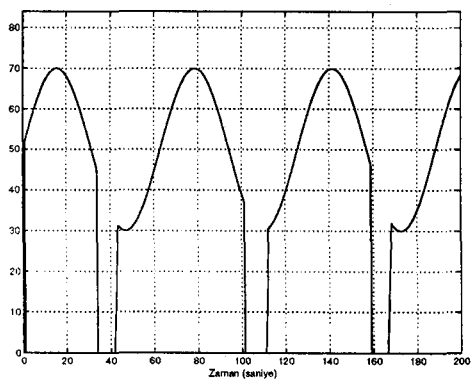
Şekil 5.8: Durum 1.4 için elde edilen sonuçlar



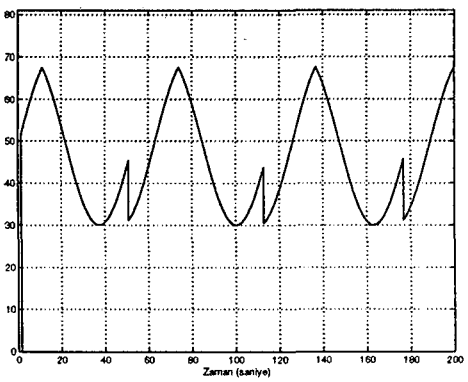
1. nodda hesaplanan iletim oranı komutu, $r_1(t)$, p/s



2. nodda hesaplanan iletim oranı komutu, $r_2(t)$, p/s

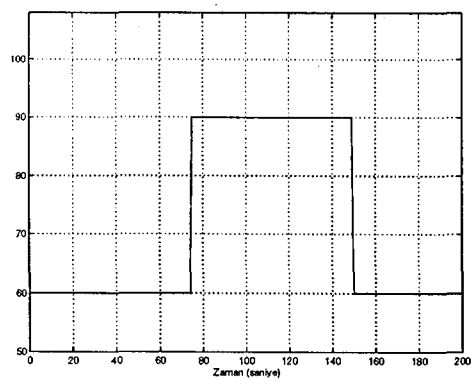


1. noddan 2. noda veri gönderim oranı, $\rho_{12}^s(t)$, p/s.

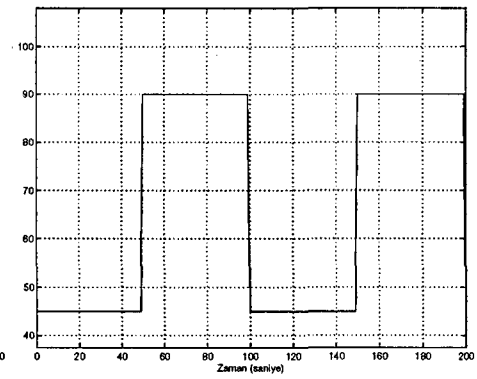


Hedef nodda veri işleme hızı, $\rho_2^s(t)$, p/s.

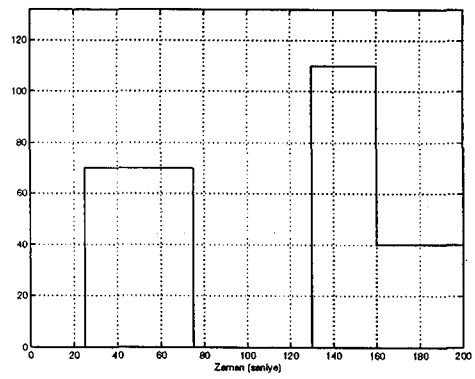
Şekil 5.9: Durum 1.4 için elde edilen sonuçlar (devam)



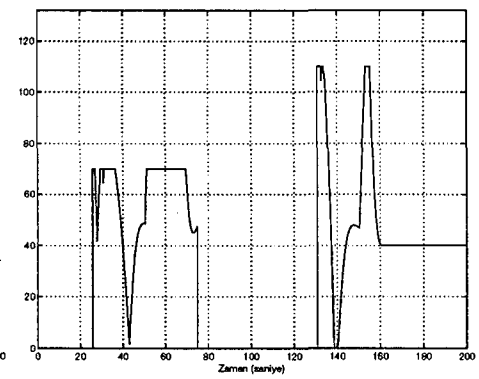
1. nodun çıkış kapasitesi, $c_1(t)$, p/s



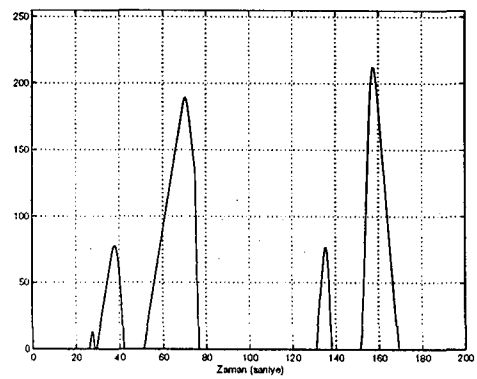
2. nodun çıkış kapasitesi, $c_2(t)$, p/s



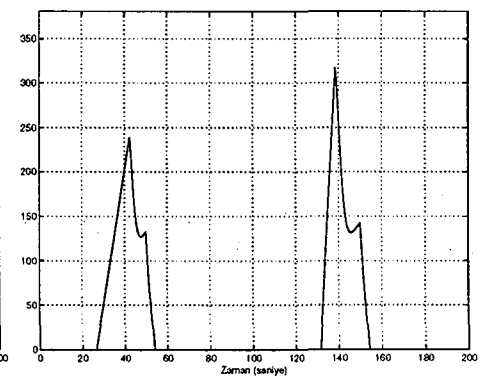
Kaynağın iletim yapmak istediği oran, $r^0(t)$, p/s.



Kaynağın gerçek iletim oranı, $r^s(t)$, p/s.

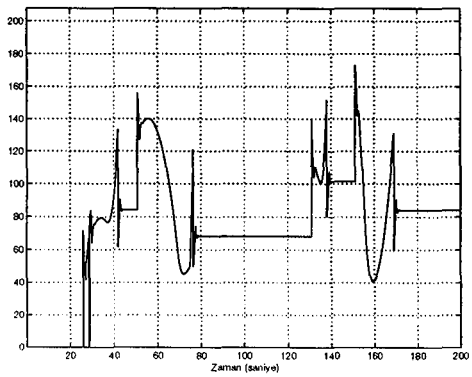


1. nodun kuyruk uzunluğu, $q_1(t)$, p.

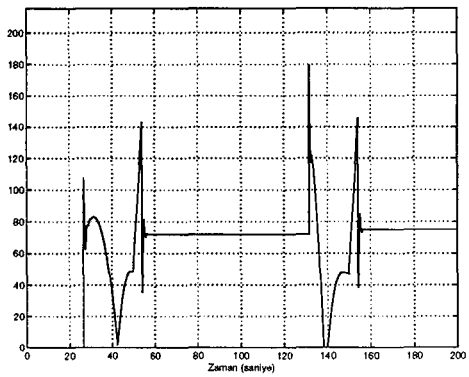


2. nodun kuyruk uzunluğu, $q_2(t)$, p.

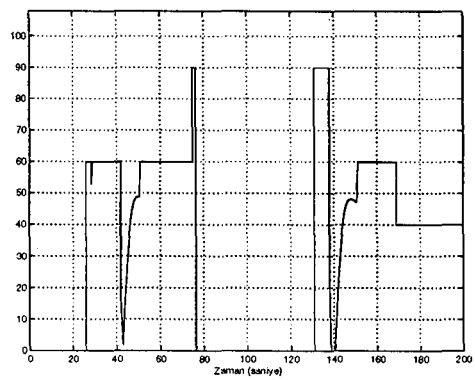
Şekil 5.10: Durum 1.5 için elde edilen sonuçlar



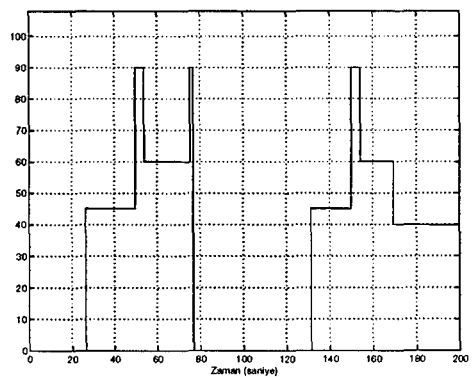
1. nodda hesaplanan iletim oranı komutu, $r_1(t)$, p/s



2. nodda hesaplanan iletim oranı komutu, $r_2(t)$, p/s

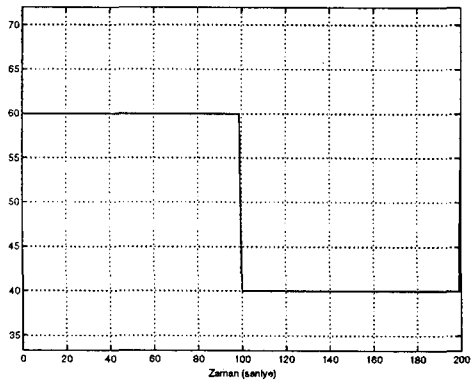


1. noddan 2. noda veri gönderim oranı, $\rho_{12}^s(t)$, p/s.

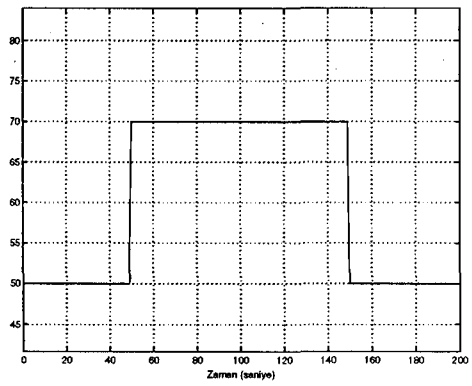


Hedef nodda veri işleme hızı, $\rho_2^s(t)$, p/s.

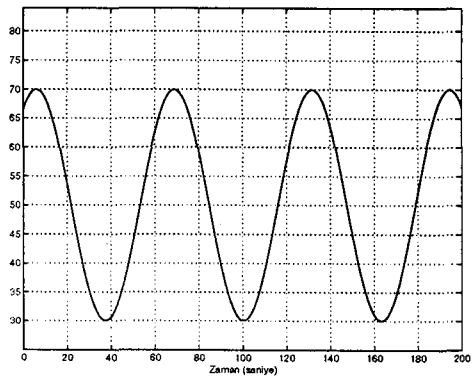
Şekil 5.11: Durum 1.5 için elde edilen sonuçlar (devam)



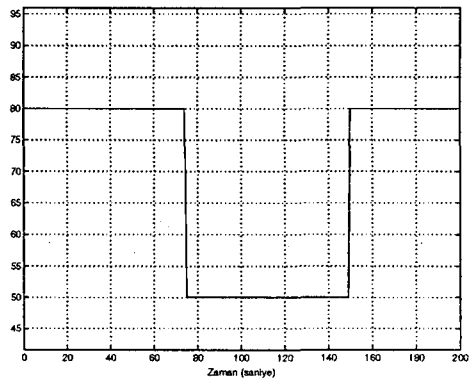
1. nodun çıkış kapasitesi, $c_1(t)$, p/s



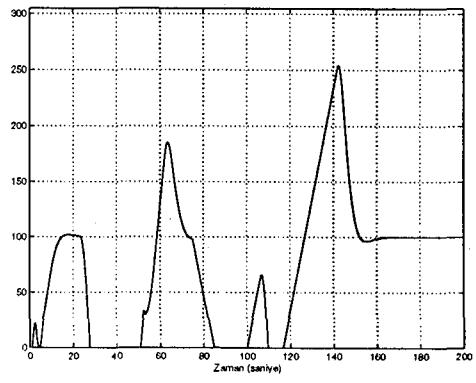
2. nodun çıkış kapasitesi, $c_2(t)$, p/s



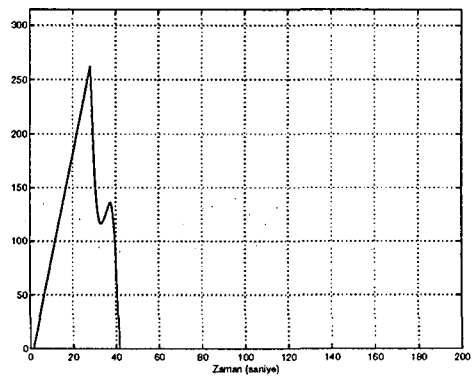
3. nodun çıkış kapasitesi, $c_3(t)$, p/s



Kaynağın iletim yapmak istediği oran, $r^0(t)$, p/s.

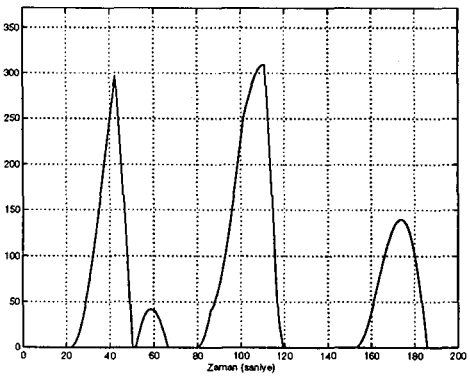


1. nodun kuyruk uzunluğu, $q_1(t)$, p.

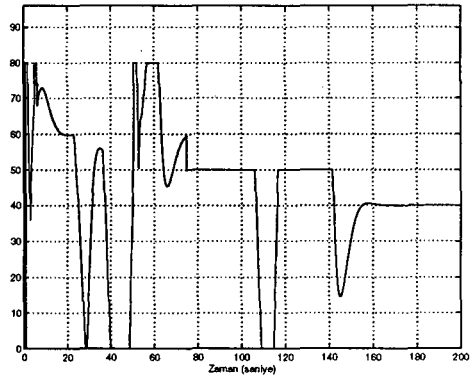


2. nodun kuyruk uzunluğu, $q_2(t)$, p.

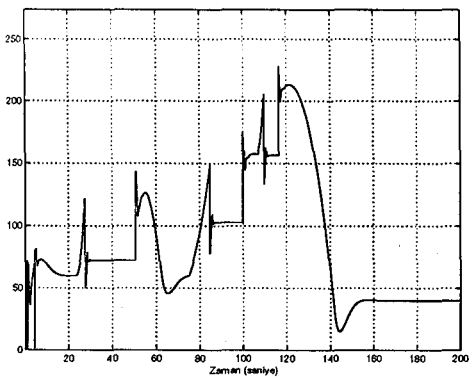
Şekil 5.12: Durum 2.1 için elde edilen sonuçlar



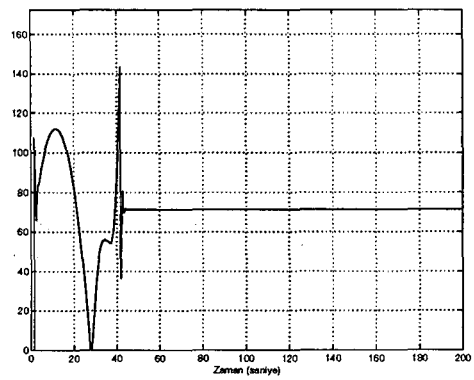
3. nodun kuyruk uzunluğu, $q_3(t)$, p.



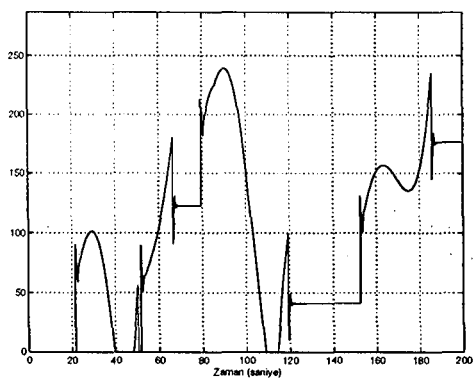
Kaynağın gerçek iletim oranı, $r^s(t)$, p/s.



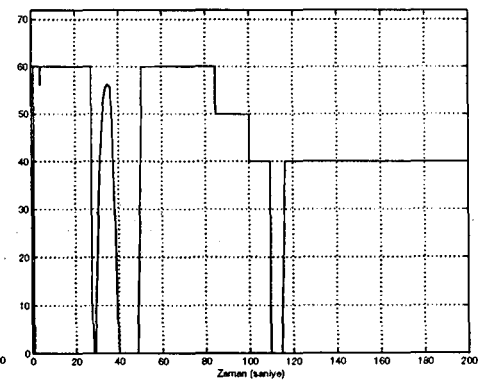
1. nodda hesaplanan iletim oranı komutu, $r_1(t)$, p/s



2. nodda hesaplanan iletim oranı komutu, $r_2(t)$, p/s

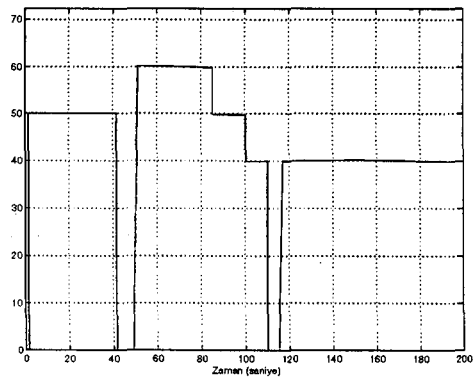


3. nodda hesaplanan iletim oranı komutu, $r_3(t)$, p/s

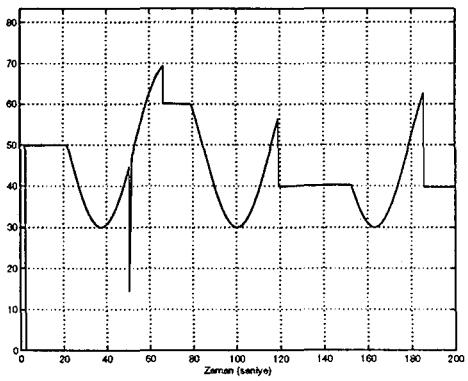


1. noddan 2. noda veri gönderim oranı, $\rho^s_{12}(t)$, p/s.

Şekil 5.13: Durum 2.1 için elde edilen sonuçlar (devam)

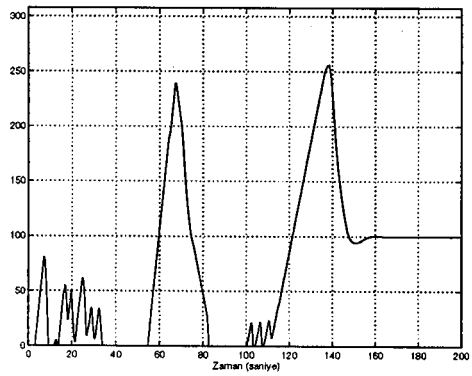


2. noddan 3. noda veri gönderim oranı, $\rho^s_{23}(t)$, p/s.

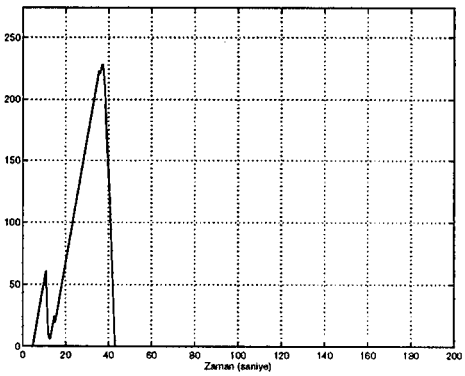


Hedef nodda veri işleme hızı, $\rho^s_3(t)$, p/s.

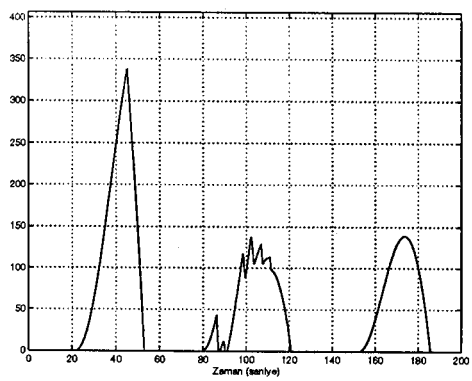
Şekil 5.14: Durum 2.1 için elde edilen sonuçlar (2. devam)



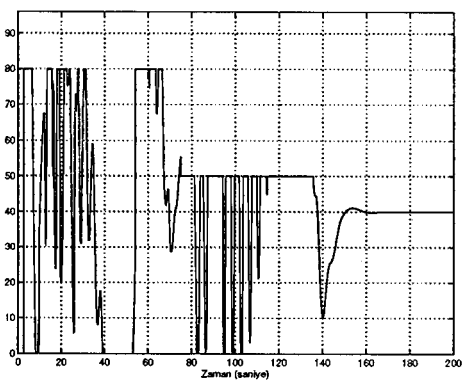
1. nodun kuyruk uzunluğu, $q_1(t)$, p.



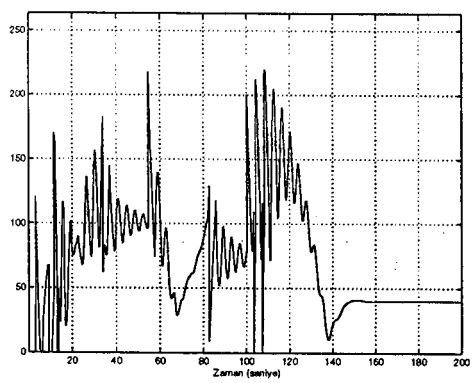
2. nodun kuyruk uzunluğu, $q_2(t)$, p.



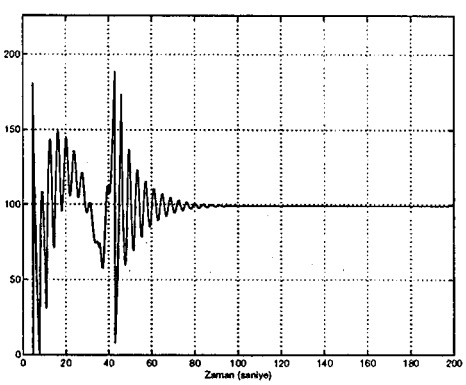
3. nodun kuyruk uzunluğu, $q_3(t)$, p.



Kaynağın gerçek iletim oranı, $r^s(t)$, p/s.

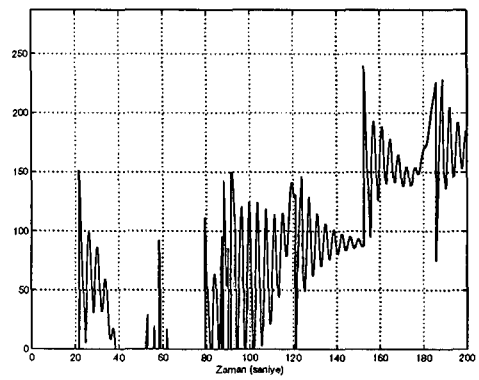


1. nodda hesaplanan iletim oranı komutu, $r_1(t)$, p/s

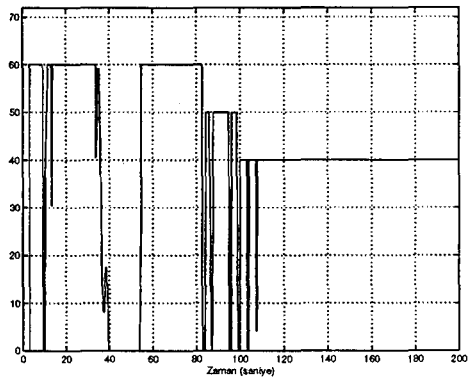


2. nodda hesaplanan iletim oranı komutu, $r_2(t)$, p/s

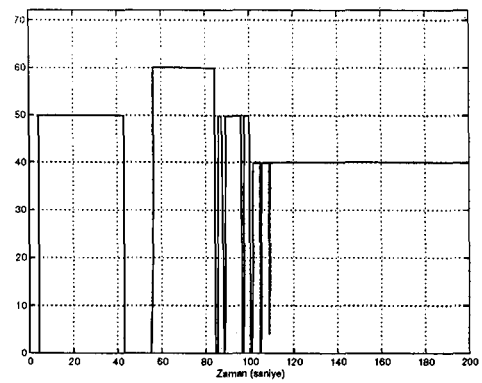
Şekil 5.15: Durum 2.2 için elde edilen sonuçlar



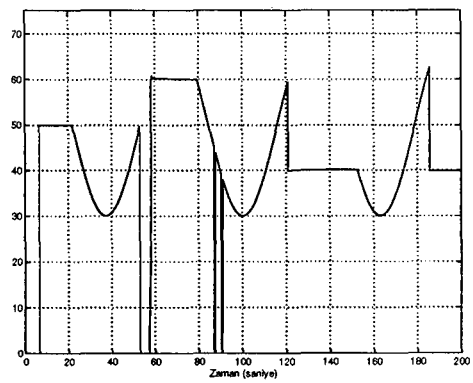
3. nodda hesaplanan iletim oranı komutu, $r_3(t)$, p/s



1. noddan 2. noda veri gönderim oranı, $\rho^s_{12}(t)$, p/s.

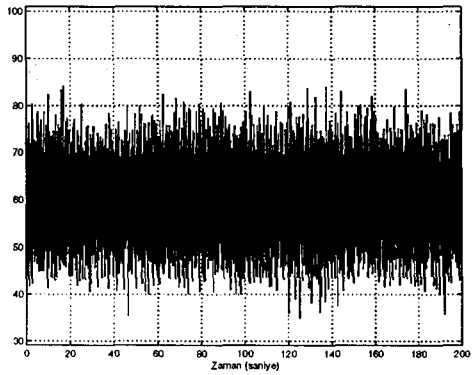


2. noddan 3. noda veri gönderim oranı, $\rho^s_{23}(t)$, p/s.

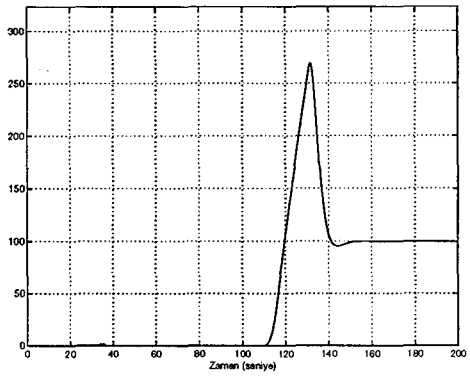


Hedef nodda veri işleme hızı, $\rho^s_3(t)$, p/s.

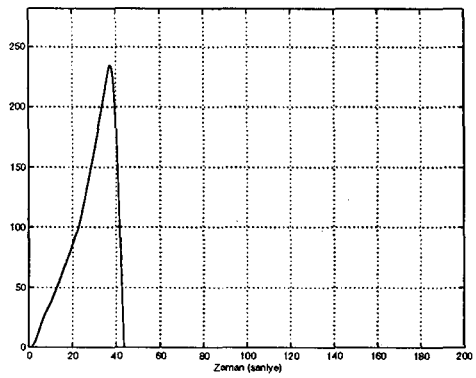
Şekil 5.16: Durum 2.2 için elde edilen sonuçlar (devam)



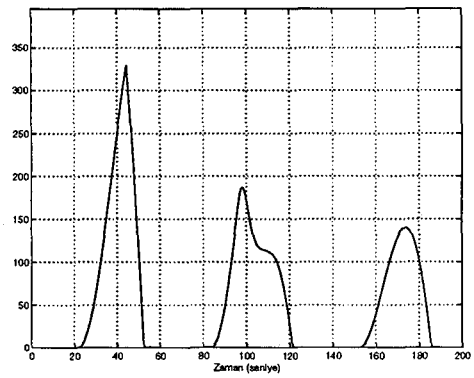
Kaynağın iletim yapmak istediği oran, $r^0(t)$, p/s.



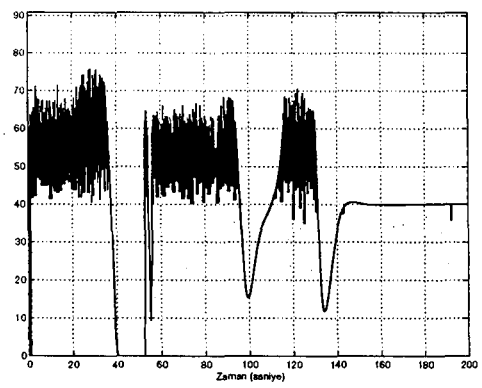
1. nodun kuyruk uzunluğu, $q_1(t)$, p.



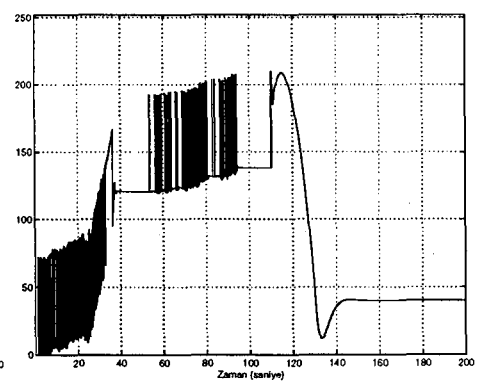
2. nodun kuyruk uzunluğu, $q_2(t)$, p.



3. nodun kuyruk uzunluğu, $q_3(t)$, p.

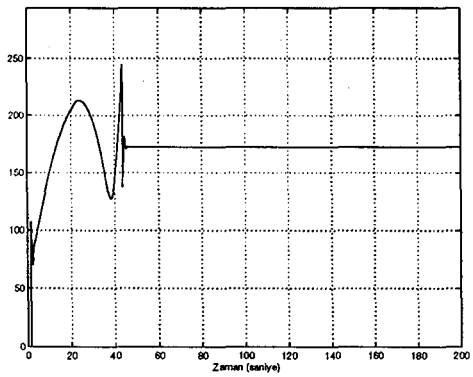


Kaynağın gerçek iletim oranı, $r^s(t)$, p/s.

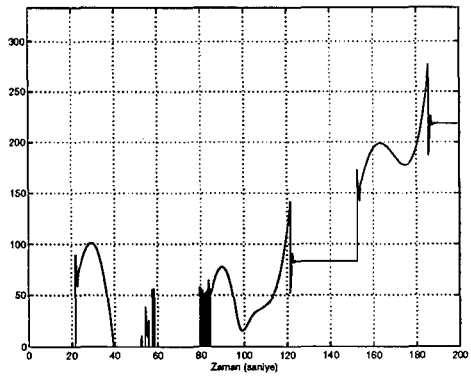


1. nodda hesaplanan iletim oranı komutu, $r_1(t)$, p/s

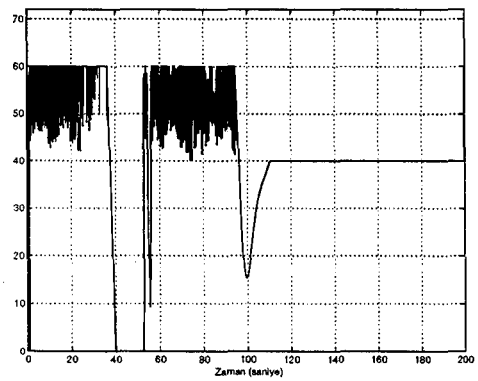
Şekil 5.17: Durum 2.3 için elde edilen sonuçlar



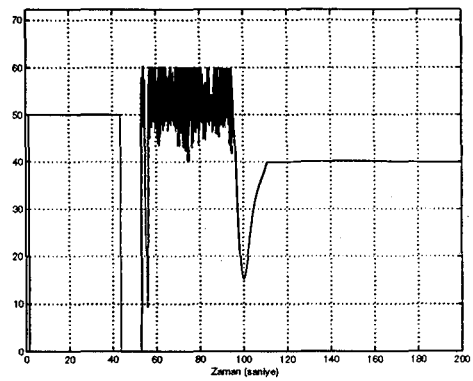
2. nodda hesaplanan iletim oranı komutu, $r_2(t)$, p/s



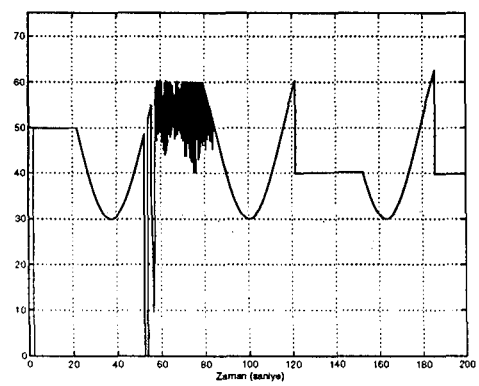
3. nodda hesaplanan iletim oranı komutu, $r_3(t)$, p/s



1. noddan 2. noda veri gönderim oranı, $\rho^s_{12}(t)$, p/s.

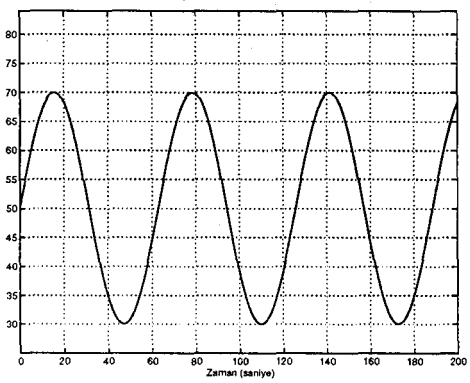


2. noddan 3. noda veri gönderim oranı, $\rho^s_{23}(t)$, p/s.

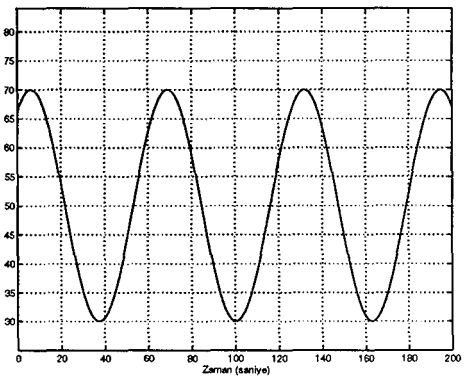


Hedef nodda veri işleme hızı, $\rho^s_3(t)$, p/s.

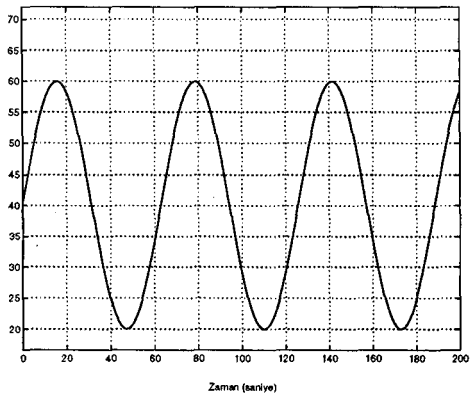
Şekil 5.18: Durum 2.3 için elde edilen sonuçlar (devam)



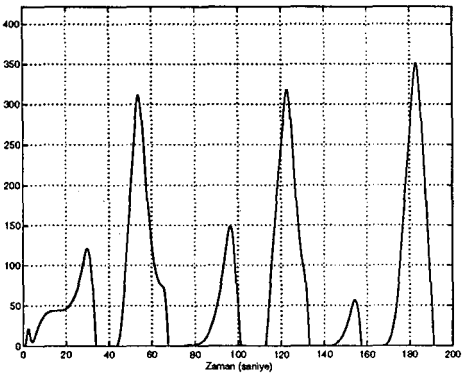
1. nodun çıkış kapasitesi, $c_1(t)$, p/s



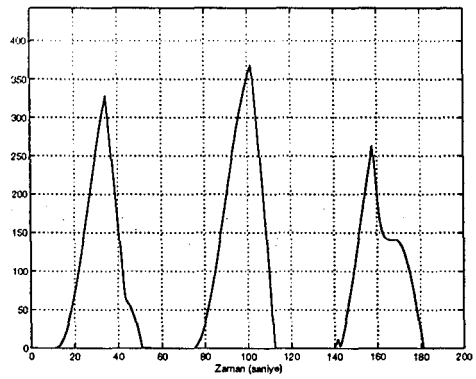
2. nodun çıkış kapasitesi, $c_2(t)$, p/s



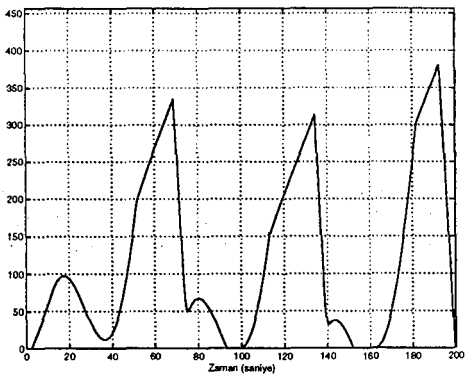
3. nodun çıkış kapasitesi, $c_3(t)$, p/s



1. nodun kuyruk uzunluğu, $q_1(t)$, p.

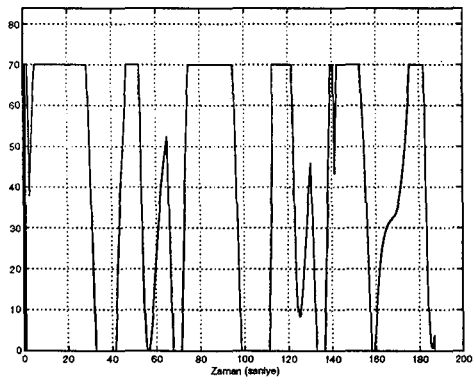


2. nodun kuyruk uzunluğu, $q_2(t)$, p.

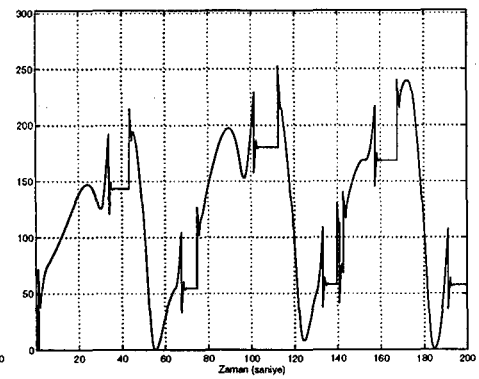


3. nodun kuyruk uzunluğu, $q_3(t)$, p.

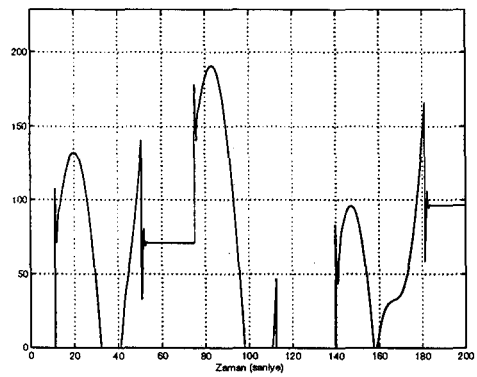
Şekil 5.19: Durum 2.4 için elde edilen sonuçlar



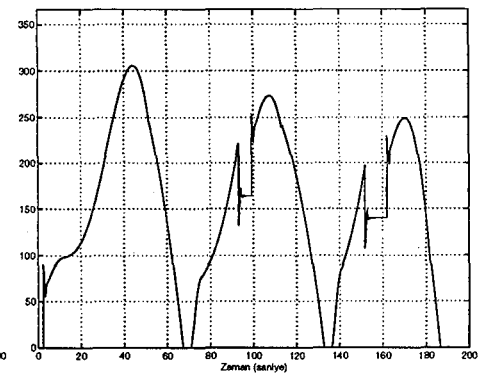
Kaynağın gerçek iletim oranı, $r^s(t)$, p/s.



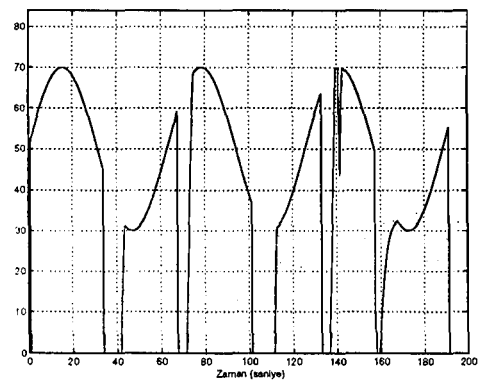
1. nodda hesaplanan iletim oranı komutu, $r_1(t)$, p/s



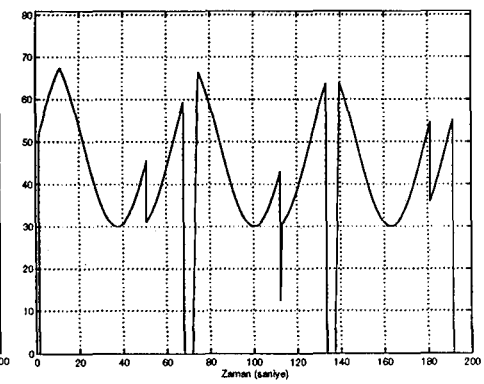
2. nodda hesaplanan iletim oranı komutu, $r_2(t)$, p/s



3. nodda hesaplanan iletim oranı komutu, $r_3(t)$, p/s

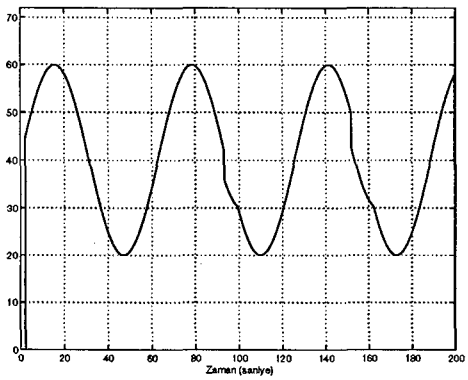


1. noddan 2. noda veri gönderim oranı, $\rho^s_{12}(t)$, p/s.



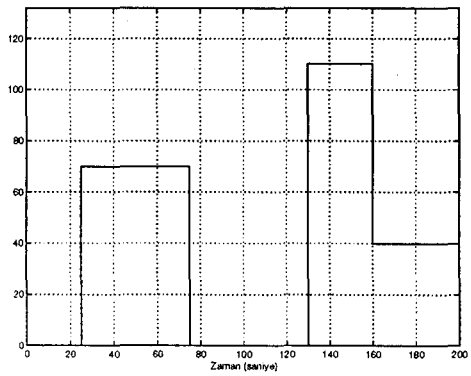
2. noddan 3. noda veri gönderim oranı, $\rho^s_{23}(t)$, p/s.

Şekil 5.20: Durum 2.4 için elde edilen sonuçlar (devam)

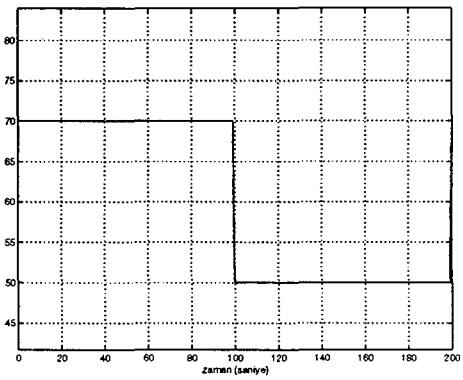


Hedef nodda veri işleme hızı, $\rho_3^s(t)$,
p/s.

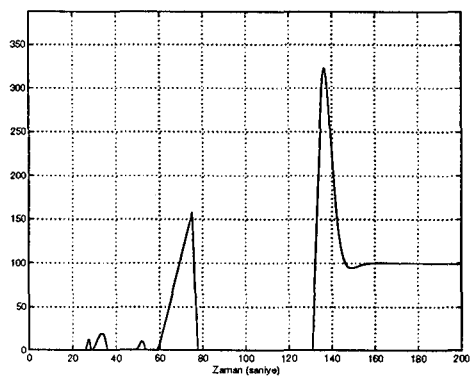
Şekil 5.21: Durum 2.4 için elde edilen sonuçlar (2. devam)



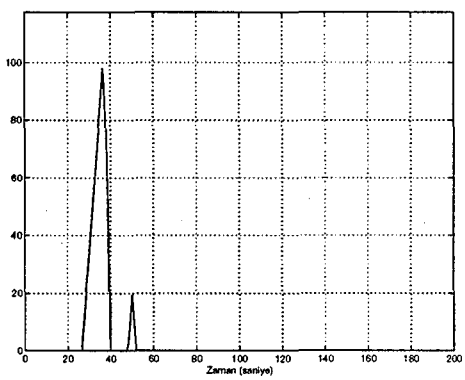
Kaynağın iletim yapmak istediği oran, $r^0(t)$, p/s.



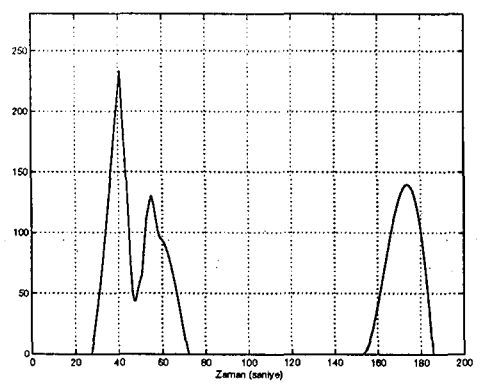
2. nodun çıkış kapasitesi, $c_2(t)$, p/s



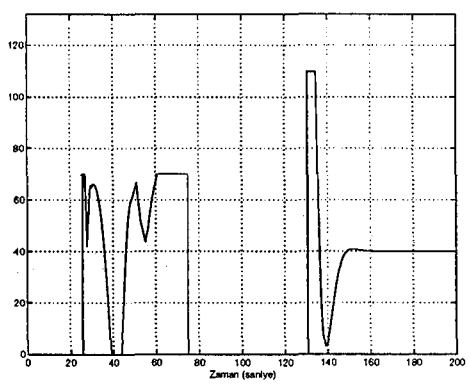
1. nodun kuyruk uzunluğu, $q_1(t)$, p.



2. nodun kuyruk uzunluğu, $q_2(t)$, p.

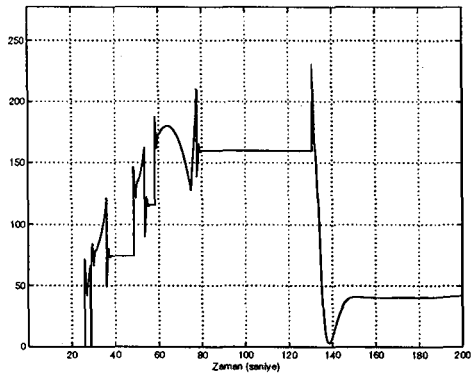


3. nodun kuyruk uzunluğu, $q_3(t)$, p.

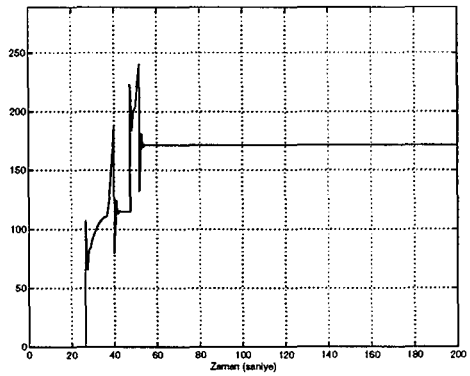


Kaynağın gerçek iletim oranı, $r^s(t)$, p/s.

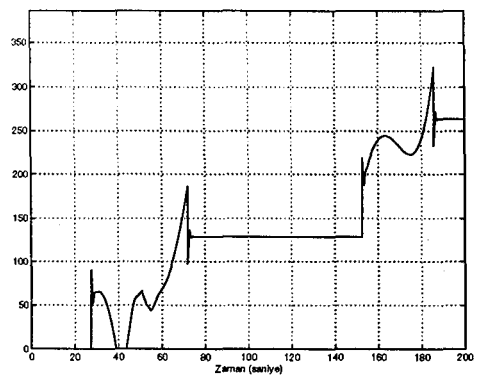
Şekil 5.22: Durum 2.5 için elde edilen sonuçlar



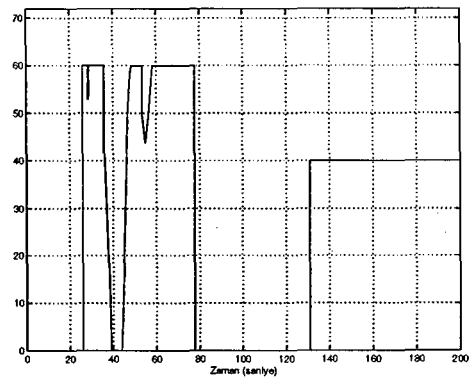
1. nodda hesaplanan iletim oranı komutu, $r_1(t)$, p/s



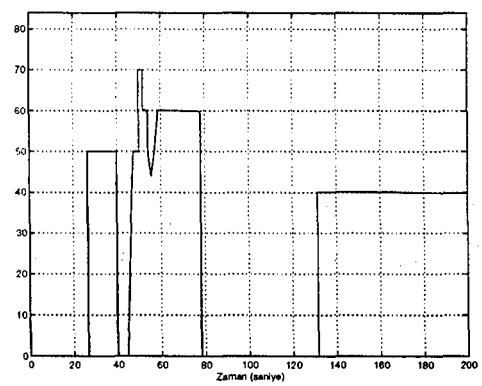
2. nodda hesaplanan iletim oranı komutu, $r_2(t)$, p/s



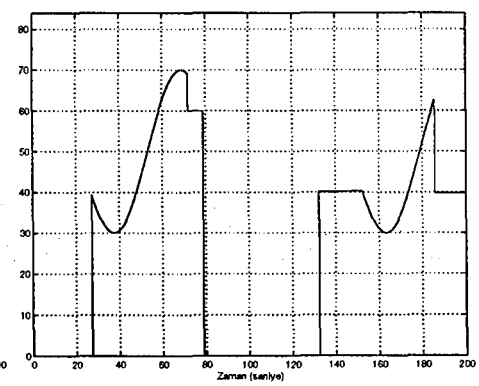
3. nodda hesaplanan iletim oranı komutu, $r_3(t)$, p/s



1. noddan 2. noda veri gönderim oranı, $\rho^s_{12}(t)$, p/s.

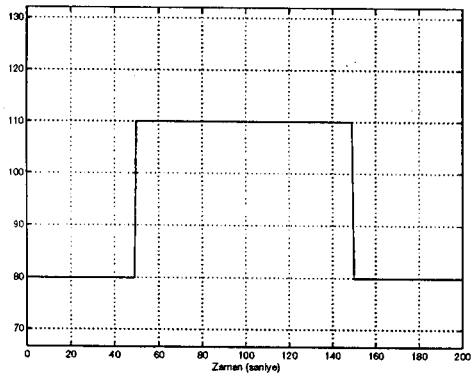


2. noddan 3. noda veri gönderim oranı, $\rho^s_{23}(t)$, p/s.

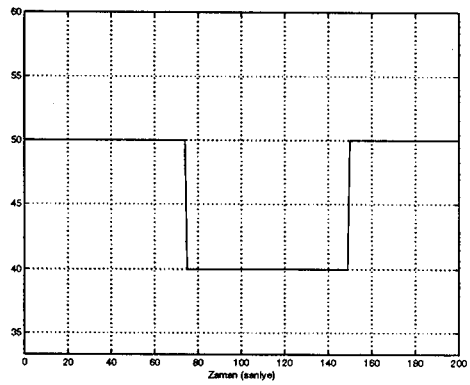


Hedef nodda veri işleme hızı, $\rho^s_3(t)$, p/s.

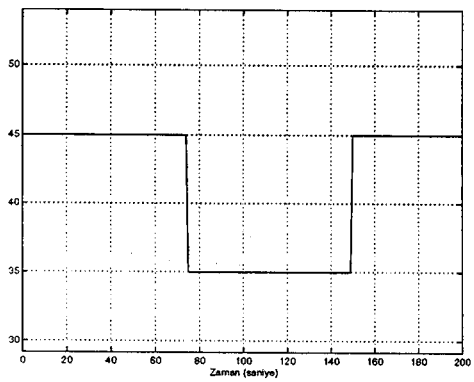
Şekil 5.23: Durum 2.5 için elde edilen sonuçlar (devam)



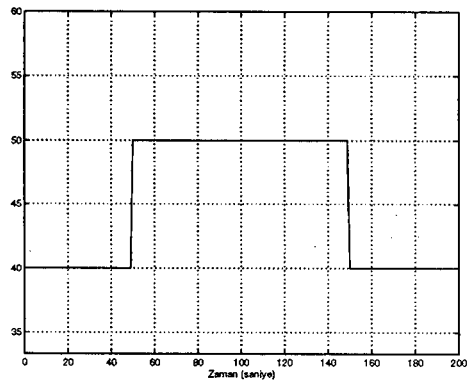
1. nodun çıkış kapasitesi, $c_1(t)$, p/s



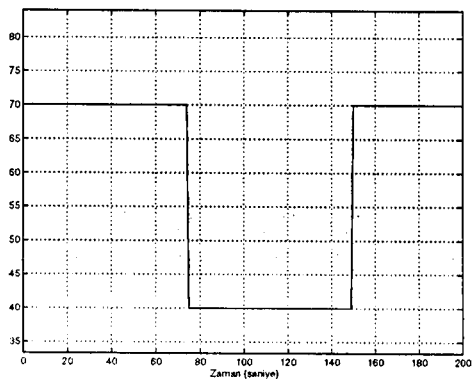
2. nodun 1. portunun çıkış kapasitesi, $c_{2_1}(t)$, p/s



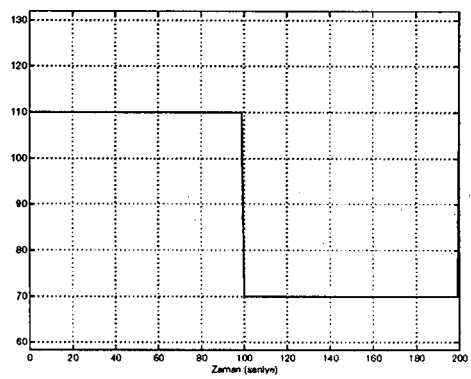
2. nodun 2. portunun çıkış kapasitesi, $c_{2_2}(t)$, p/s



3. nodun çıkış kapasitesi, $c_3(t)$, p/s

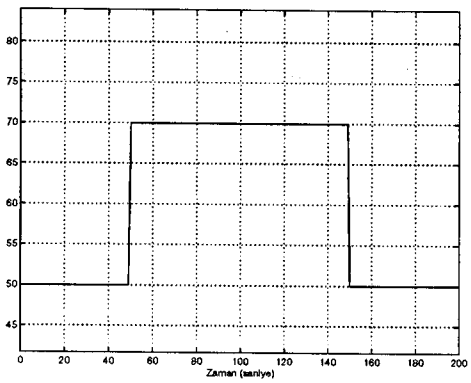


1. kaynağın hedef nodunun çıkış kapasitesi, $c_{d_1}(t)$, p/s

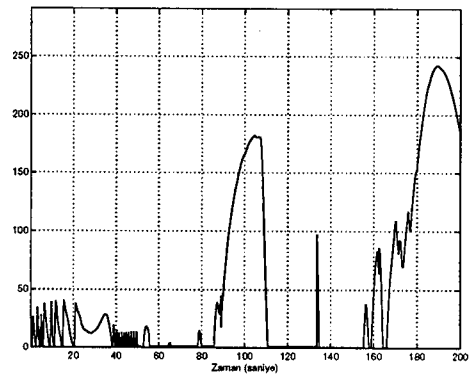


1. Kaynağın iletim yapmak istediği oran, $r^{0,1}(t)$, p/s.

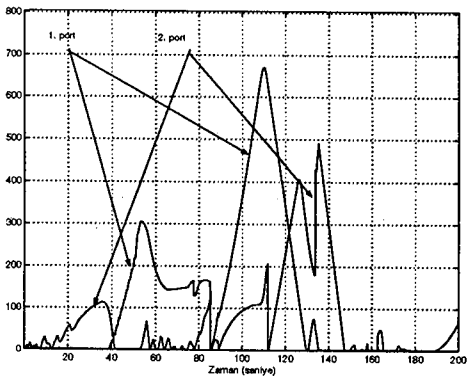
Şekil 5.24: Durum 3.1 için elde edilen sonuçlar



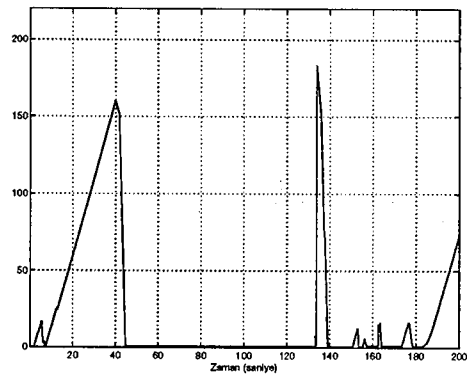
2. Kaynağın iletim yapmak istediği oran, $r^{0,2}(t)$, p/s.



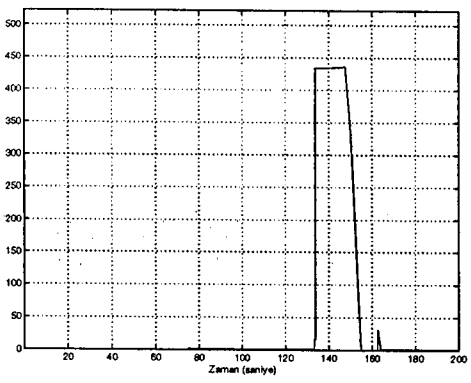
1. nodun kuyruk uzunluğu, $q_1(t)$, p



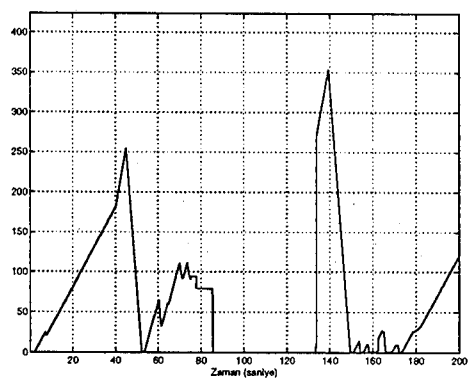
2. nodun kuyruk uzunluğu, $q_2(t)$, p



3. nodun kuyruk uzunluğu, $q_3(t)$, p

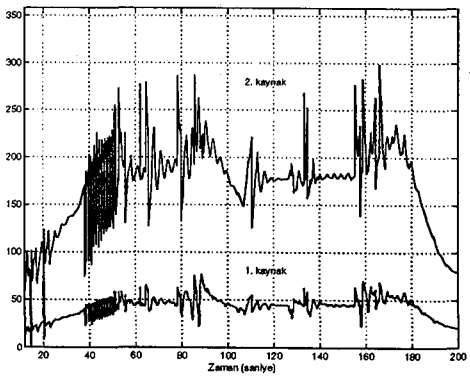


d_1'in kuyruk uzunluğu, $q_{d_1}(t)$, p.

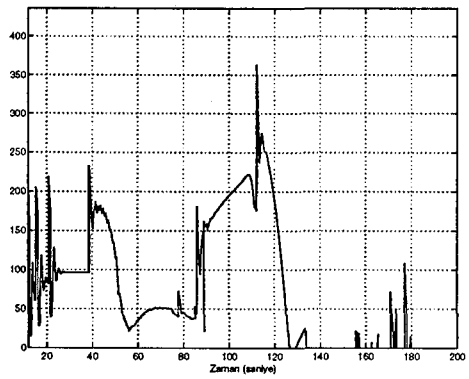


d_2'nin kuyruk uzunluğu, $q_{d_2}(t)$, p.

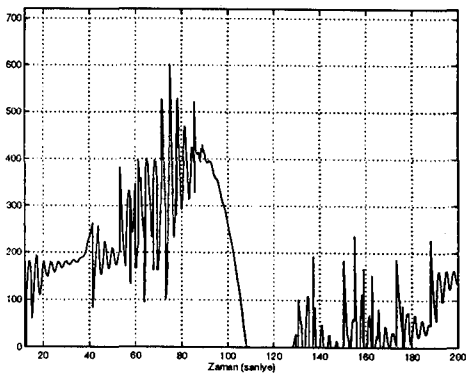
Şekil 5.25: Durum 3.1 için elde edilen sonuçlar (devam)



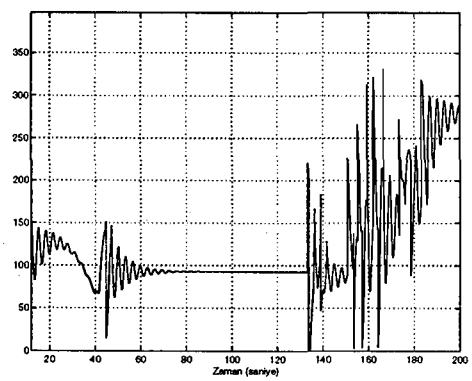
1. nodda hesaplanan iletim oranı komutları, $r_1(t)$, p/s



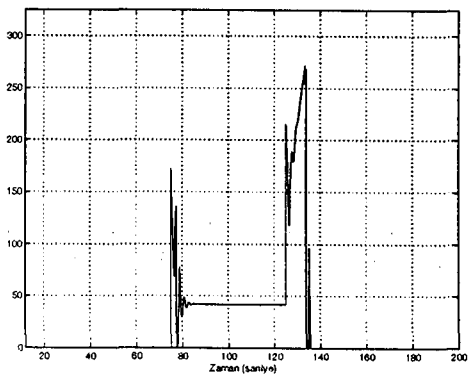
2. nodun 1. portunda hesaplanan iletim oranı komutu, $r_{2_1}^1(t)$, p/s



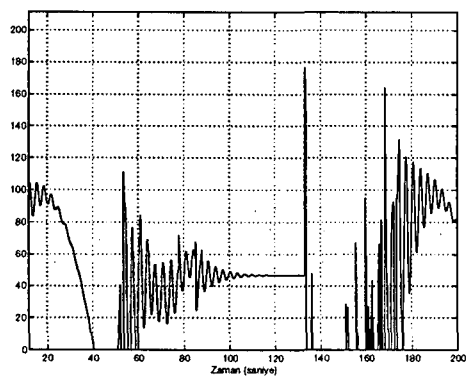
2. nodun 2. portunda hesaplanan iletim oranı komutu, $r_{2_2}^2(t)$, p/s



2. kaynak için 3. nodda hesaplanan iletim oranı komutu, $r_3^2(t)$, p/s

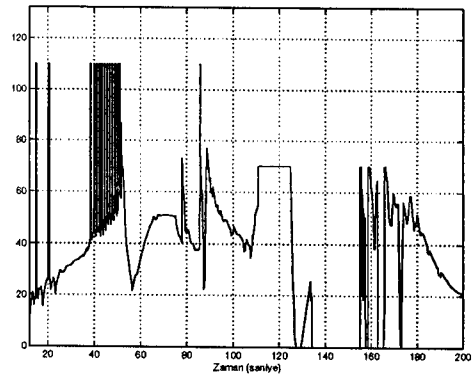


1. kaynak için d_1 nodunda hesaplanan iletim oranı komutu, $r_{d_1}^1(t)$, p/s

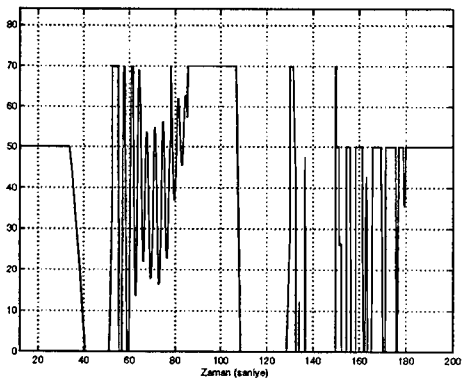


2. kaynak için d_2 nodunda hesaplanan iletim oranı komutu, $r_{d_2}^2(t)$, p/s

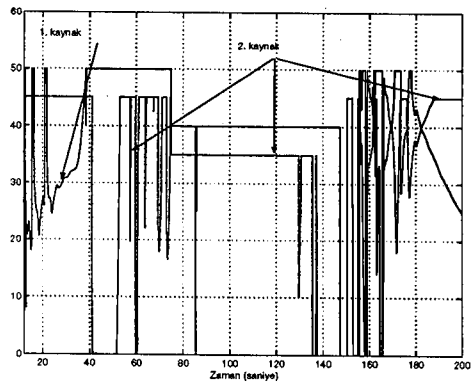
Şekil 5.26: Durum 3.1 için elde edilen sonuçlar (2. devam)



1. Kaynağın gerçek iletim oranı, $r^{1,s}(t)$, p/s.

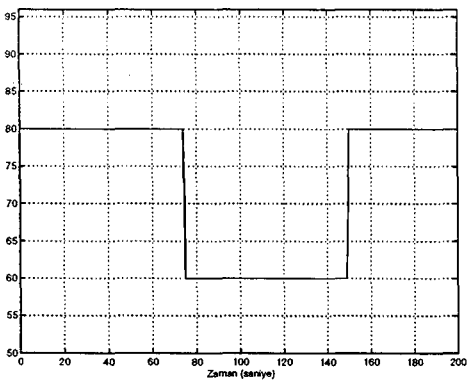


2. Kaynağın gerçek iletim oranı, $r^{2,s}(t)$, p/s.

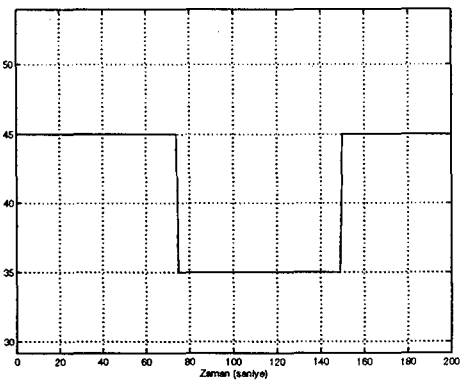


2. noddan gönderilen veri oranı, $\rho_2^s(t)$, p/s.

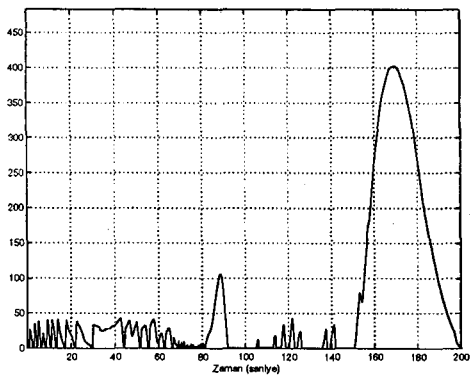
Şekil 5.27: Durum 3.1 için elde edilen sonuçlar (3. devam)



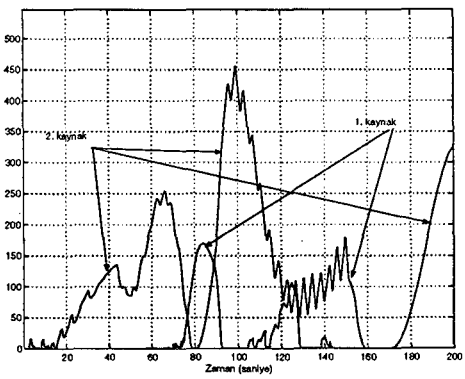
2. nodun 1. portunun çıkış kapasitesi, $c_{2_1}(t)$, p/s



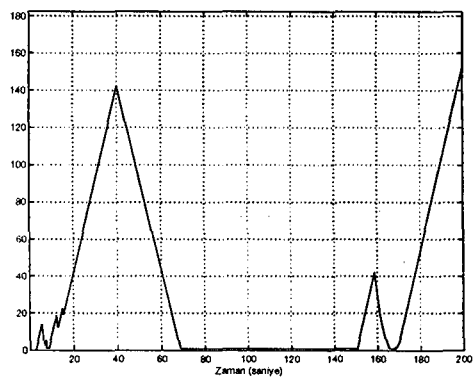
2. nodun 2. portunun çıkış kapasitesi, $c_{2_2}(t)$, p/s



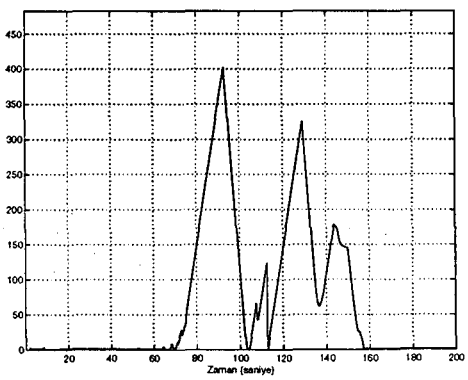
1. nodun kuyruk uzunluğu, $q_1(t)$, p



2. nodun kuyruk uzunluğu, $q_2(t)$, p

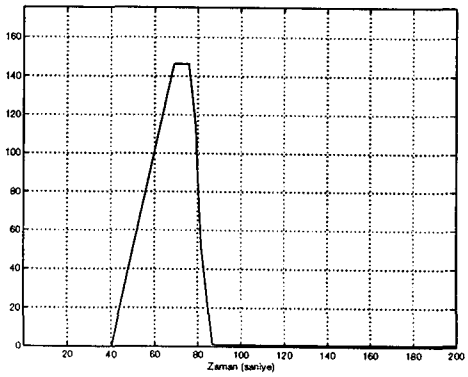


3. nodun kuyruk uzunluğu, $q_3(t)$, p

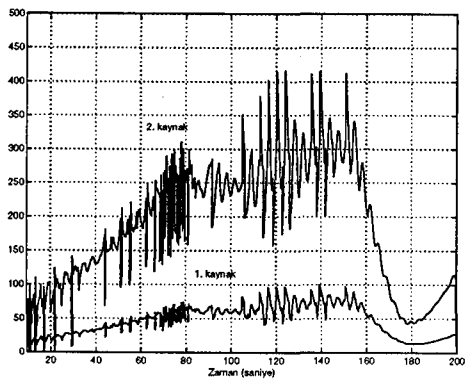


d_1'in kuyruk uzunluğu, $q_{d_1}(t)$, p.

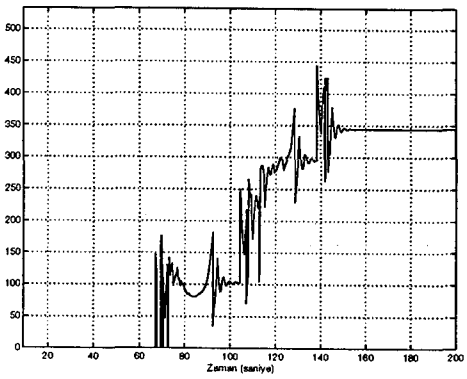
Şekil 5.28: Durum 3.2 için elde edilen sonuçlar



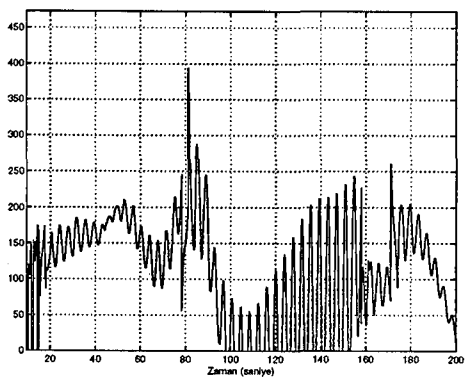
d_2'nin kuyruk uzunluğu, $q_{d_2}(t)$, p.



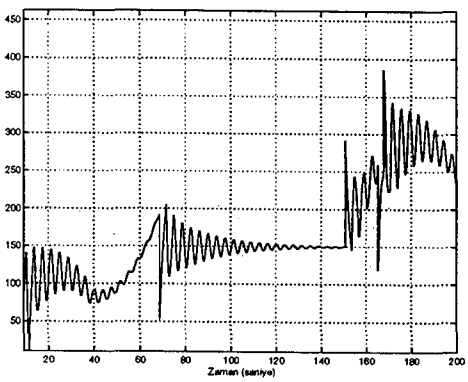
1. nodda hesaplanan iletim oranı komutları, $r_1(t)$, p/s



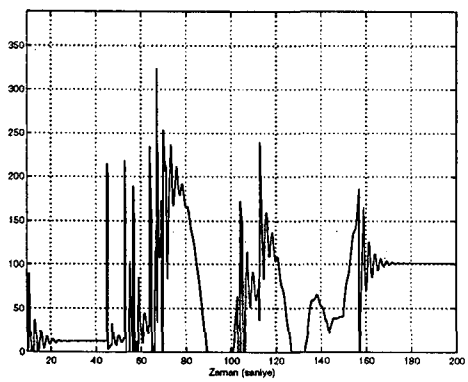
2. nodun 1. portunda hesaplanan iletim oranı komutu, $r^1_{2_1}(t)$, p/s



2. nodun 2. portunda hesaplanan iletim oranı komutu, $r^2_{2_2}(t)$, p/s

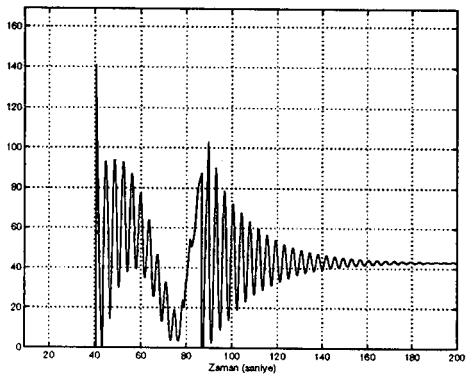


2. kaynak için 3. nodda hesaplanan iletim oranı komutu, $r^2_3(t)$, p/s

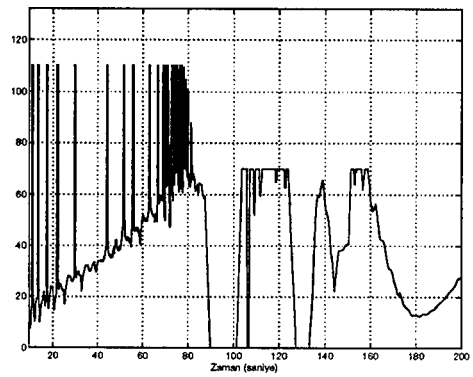


1. kaynak için d_1 nodunda hesaplanan iletim oranı komutu, $r^1_{d_1}(t)$, p/s

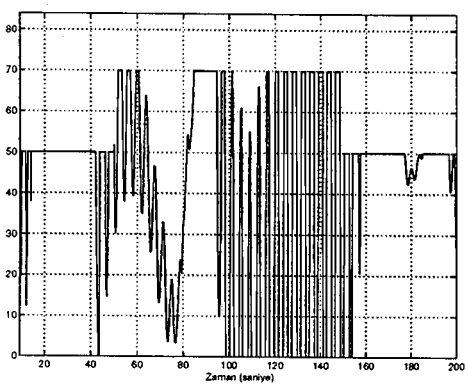
Şekil 5.29: Durum 3.2 için elde edilen sonuçlar (devam)



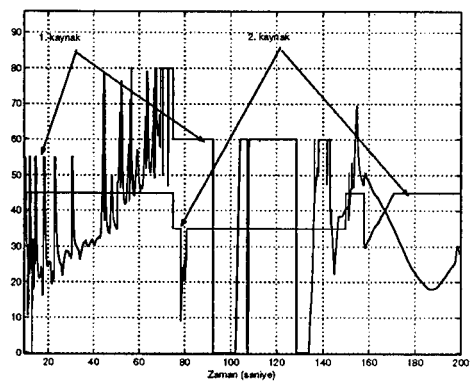
2. kaynak için d_2 nodunda hesaplanan iletim oranı komutu, $r^2_{d_2}(t)$, p/s



1. Kaynağın gerçek iletim oranı, $r^{1,s}(t)$, p/s.

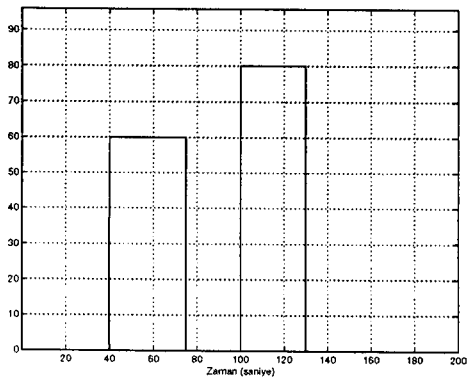


2. Kaynağın gerçek iletim oranı, $r^{2,s}(t)$, p/s.

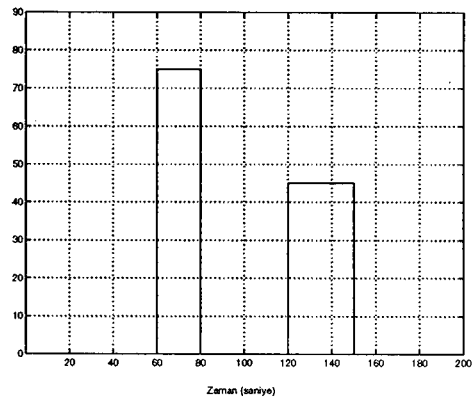


2. noddan gönderilen veri oranı, $\rho^s_2(t)$, p/s.

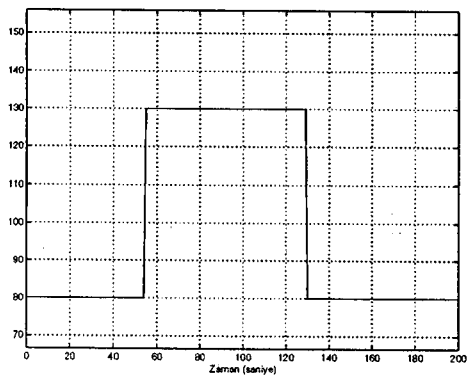
Şekil 5.30: Durum 3.2 için elde edilen sonuçlar (2. devam)



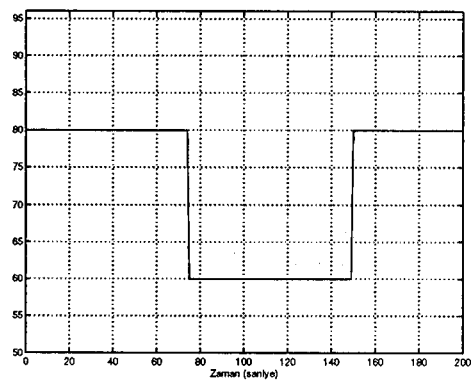
1. Kaynağın iletim yapmak istediği oran, $r^{0,1}(t)$, p/s.



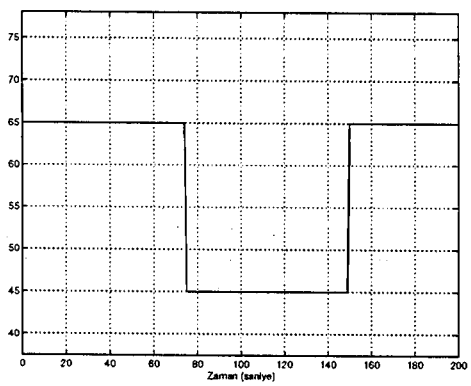
2. Kaynağın iletim yapmak istediği oran, $r^{0,2}(t)$, p/s.



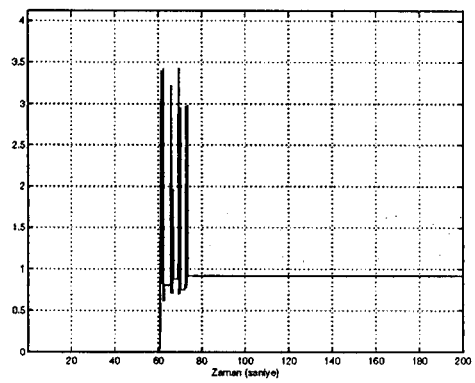
1. nodun çıkış kapasitesi, $c_1(t)$, p/s



2. nodun 1. portunun çıkış kapasitesi, $c_{2_1}(t)$, p/s

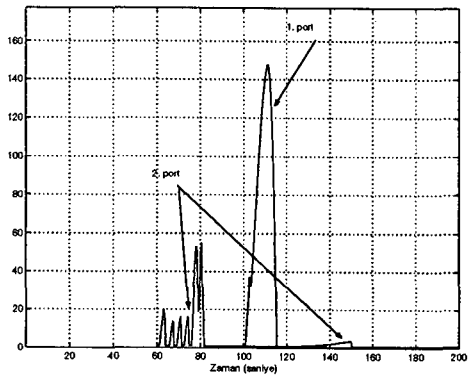


2. nodun 2. portunun çıkış kapasitesi, $c_{2_2}(t)$, p/s

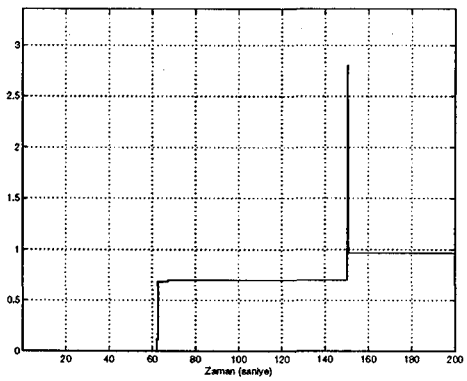


1. nodun kuyruk uzunluğu, $q_1(t)$, p

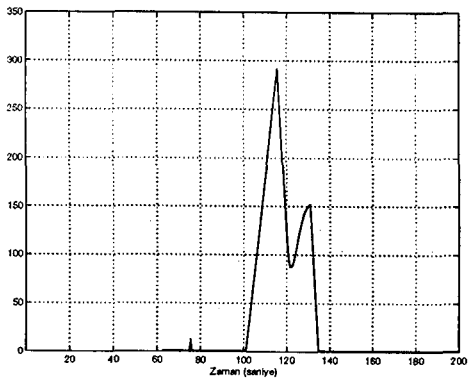
Şekil 5.31: Durum 3.3 için elde edilen sonuçlar



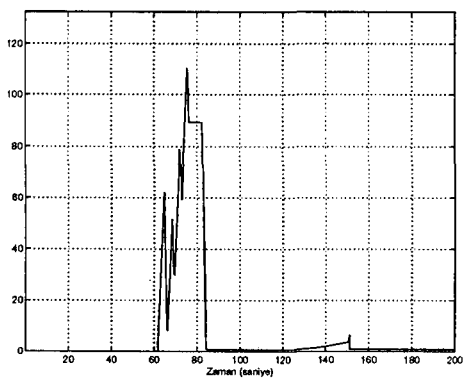
2. nodun kuyruk uzunluğu, $q_2(t)$, p



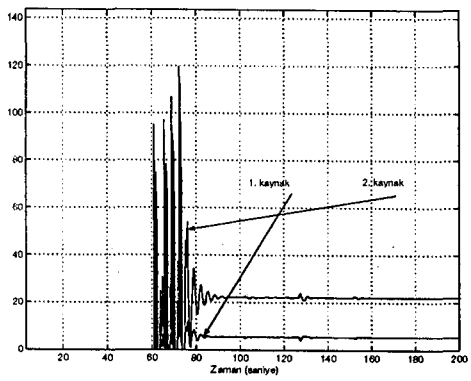
3. nodun kuyruk uzunluğu, $q_3(t)$, p



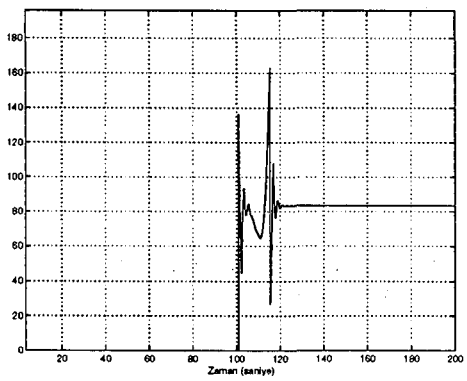
d_1'in kuyruk uzunluğu, $q_{d_1}(t)$, p.



d_2'nin kuyruk uzunluğu, $q_{d_2}(t)$, p.

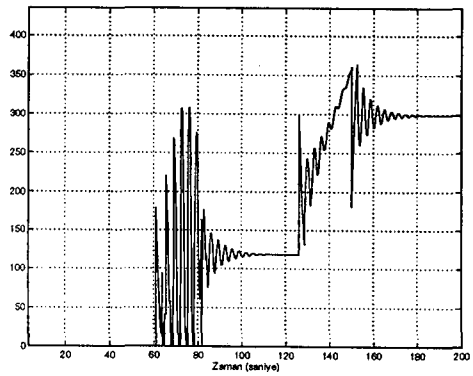


1. nodda hesaplanan iletim oranı komutları, $r_1(t)$, p/s

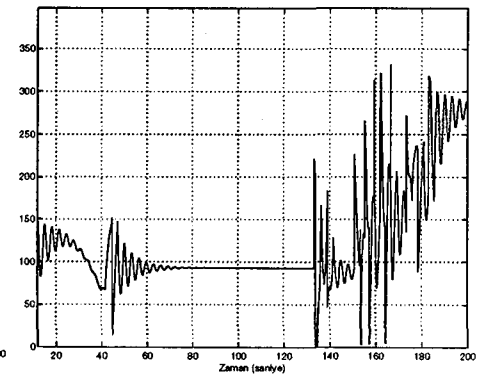


2. nodun 1. portunda hesaplanan iletim oranı komutu, $r^1_{2_1}(t)$, p/s

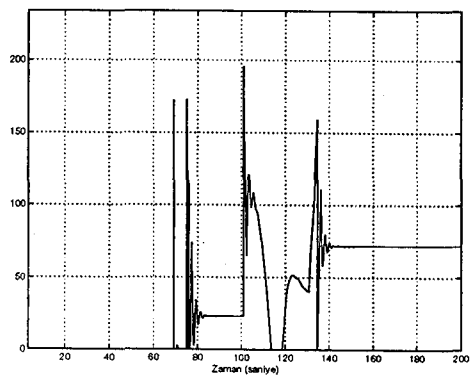
Şekil 5.32: Durum 3.3 için elde edilen sonuçlar (devam)



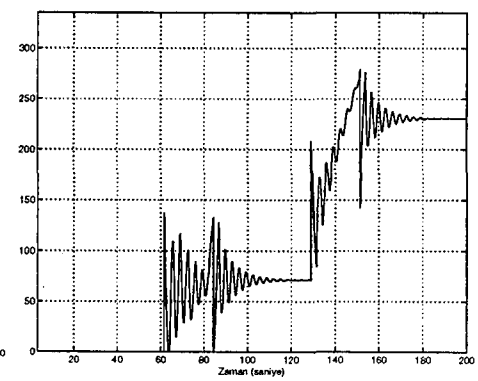
2. nodun 2. portunda hesaplanan iletim oranı komutu, $r^2_{2_2}(t)$, p/s



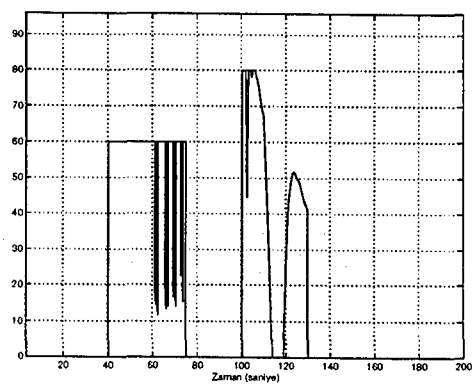
2. kaynak için 3. nodda hesaplanan iletim oranı komutu, $r^2_3(t)$, p/s



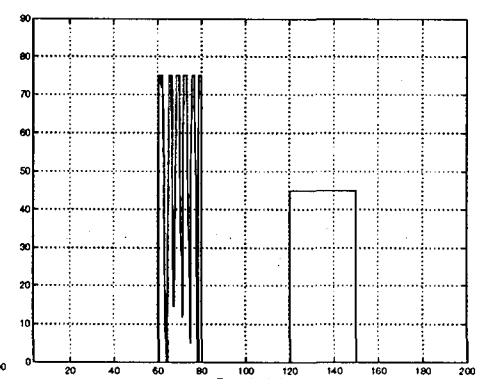
1. kaynak için d_1 nodunda hesaplanan iletim oranı komutu, $r^1_{d_1}(t)$, p/s



2. kaynak için d_2 nodunda hesaplanan iletim oranı komutu, $r^2_{d_2}(t)$, p/s

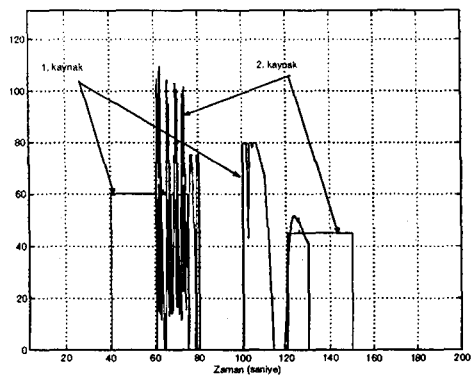


1. Kaynağın gerçek iletim oranı, $r^{1,s}(t)$, p/s.

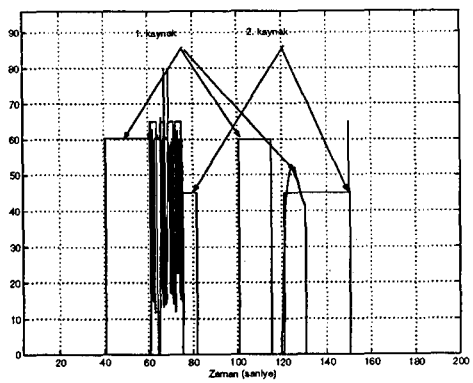


2. Kaynağın gerçek iletim oranı, $r^{2,s}(t)$, p/s.

Şekil 5.33: Durum 3.3 için elde edilen sonuçlar (2. devam)

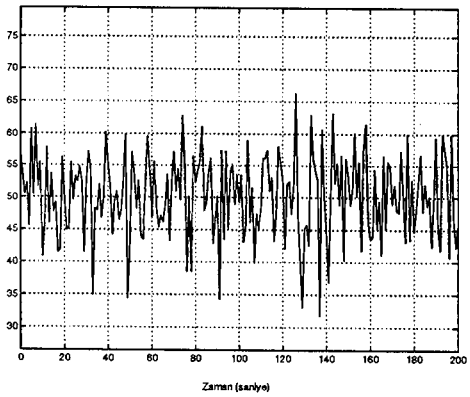


1. noddan 2. noda veri gönderim oranı, $\rho_{12}^{s}(t)$, p/s.

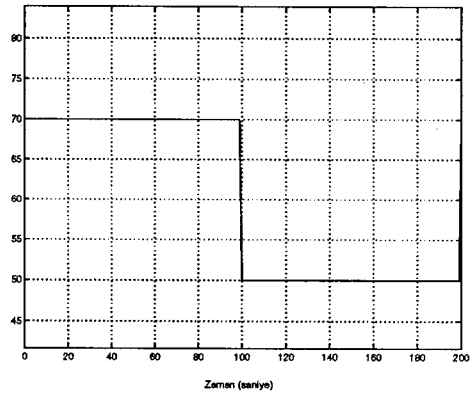


2. noddan gönderilen veri oranı, $\rho_{2}^{s}(t)$, p/s.

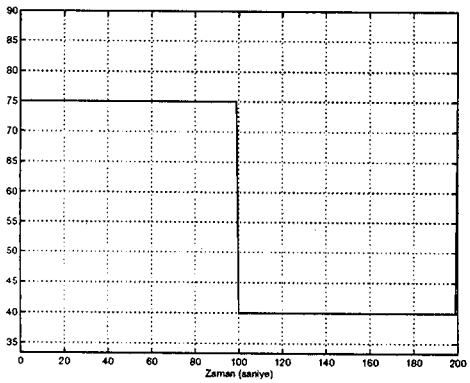
Şekil 5.34: Durum 3.3 için elde edilen sonuçlar (3. devam)



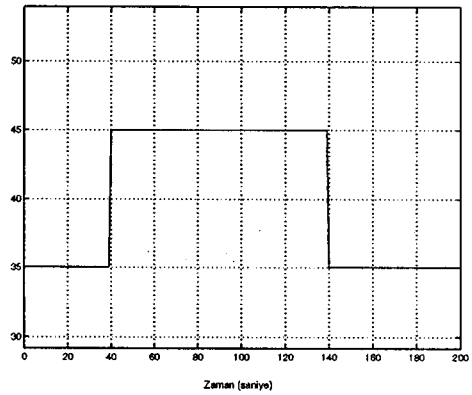
1. Kaynağın iletim yapmak istediği oran, $r^{0,1}(t)$, p/s.



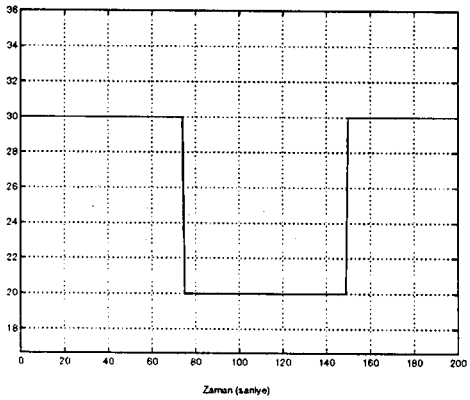
2. nodun 1. portunun çıkış kapasitesi, $c^1_{2_1}(t)$, p/s



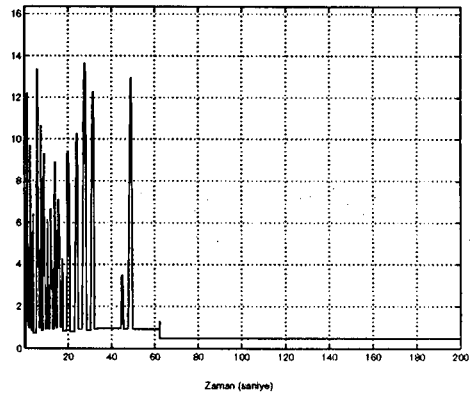
2. nodun 2. portunun çıkış kapasitesi, $c^2_{2_2}(t)$, p/s



3. nodun çıkış kapasitesi, $c_3(t)$, p/s

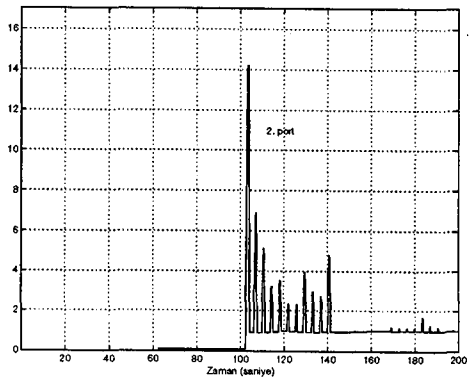


d_1'nin çıkış kapasitesi, $c_{d_1}(t)$, p/s

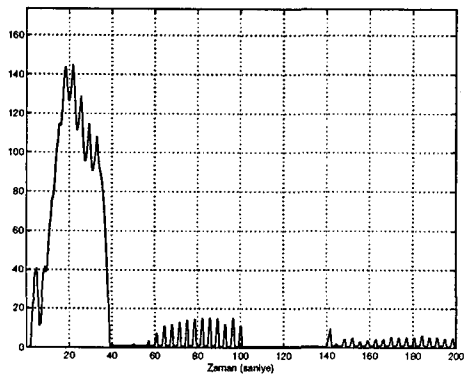


1. nodun kuyruk uzunluğu, $q_1(t)$, p

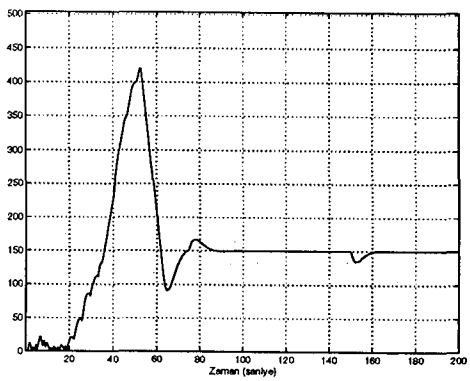
Şekil 5.35: Durum 3.4 için elde edilen sonuçlar



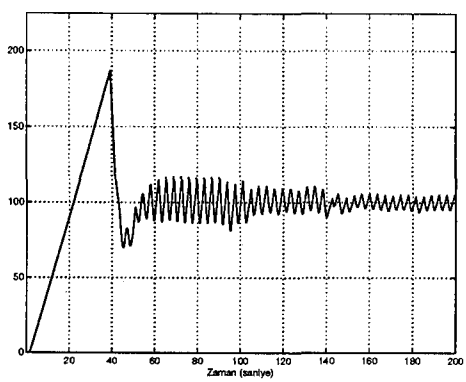
2. nodun kuyruk uzunluğu, $q_2(t)$, p



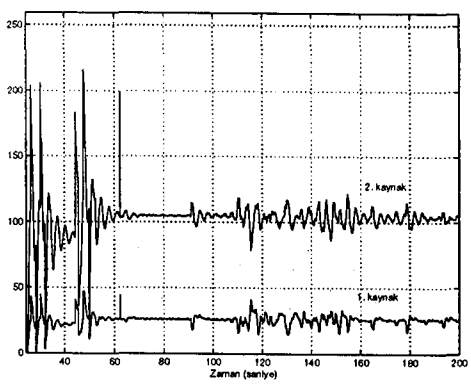
3. nodun kuyruk uzunluğu, $q_3(t)$, p



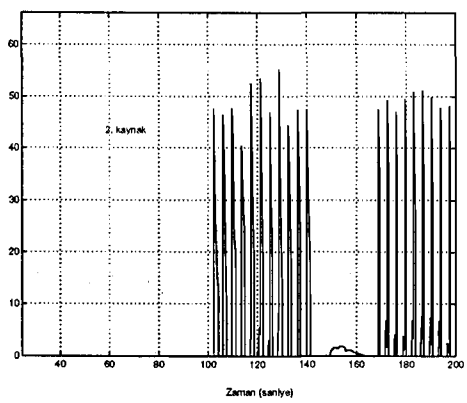
d_1'in kuyruk uzunluğu, $q_{d_1}(t)$, p.



d_2'nin kuyruk uzunluğu, $q_{d_2}(t)$, p.

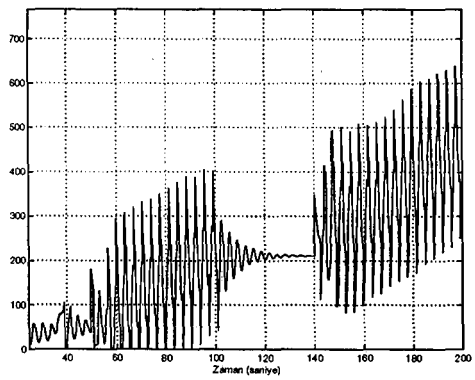


1. nodda hesaplanan iletim oranı komutları, $r_1(t)$, p/s

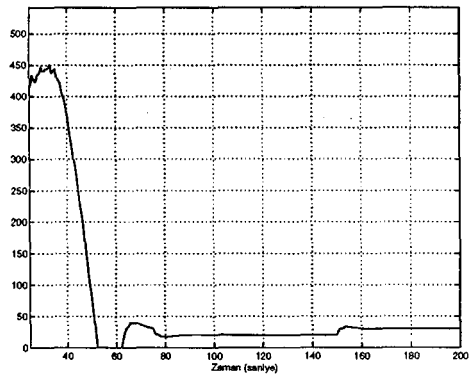


2. nodun portlarında hesaplanan iletim oranı komutları, $r_{2_k}^j(t)$, p/s

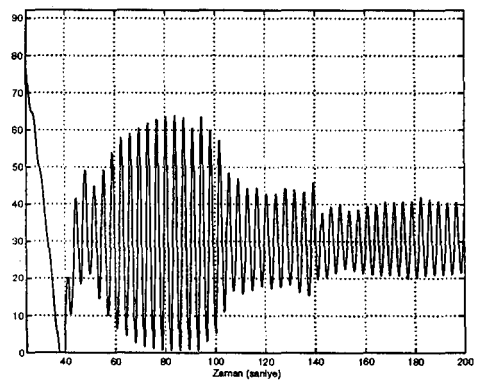
Şekil 5.36: Durum 3.4 için elde edilen sonuçlar (devam)



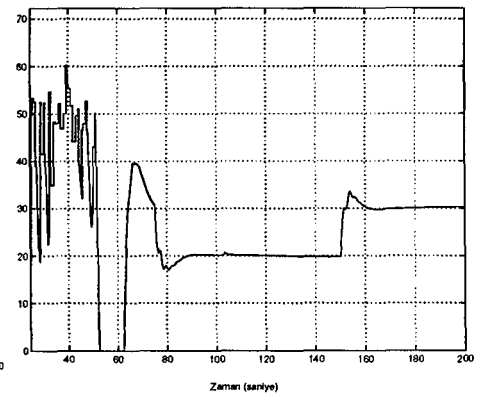
2. kaynak için 3. nodda hesaplanan iletim oranı komutu, $r_3^2(t)$, p/s



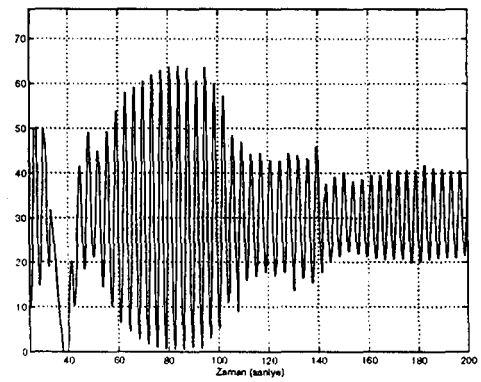
1. kaynak için d_1 nodunda hesaplanan iletim oranı komutu, $r_{d_1}^1(t)$, p/s



2. kaynak için d_2 nodunda hesaplanan iletim oranı komutu, $r_{d_2}^2(t)$, p/s

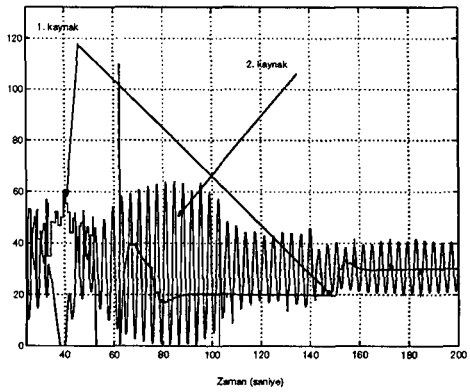


1. Kaynağın gerçek iletim oranı, $r^{1,s}(t)$, p/s.

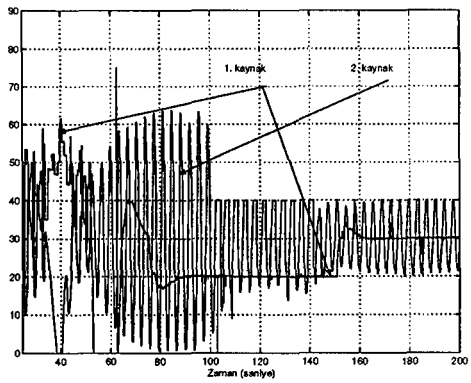


2. Kaynağın gerçek iletim oranı, $r^{2,s}(t)$, p/s.

Şekil 5.37: Durum 3.4 için elde edilen sonuçlar (2. devam)



1. noddan 2. noda gönderilen veri oranı, $\rho^s_{12}(t)$, p/s.



2. noddan gönderilen veri oranı, $\rho^s_2(t)$, p/s.

Şekil 5.38: Durum 3.4 için elde edilen sonuçlar (3. devam)

6 SONUÇ

Bu tezde, çok dar boğazlı ATM ağları için akış kontrolü algoritması tasarımı ele alınmıştır. Algoritma tasarımında kapalı döngü kontrol stratejisi ve oran–tabanlı yaklaşım kullanılmıştır. Kapalı döngü kontrol stratejisinin kullanılmasının temel amacı, sistem performansını kötü yönde etkileyebilecek belirsizliklerin olduğu durumlarda da sistem kararlılığını ve istenen performansı sağlamaktır. Oran–tabanlı yaklaşımın getirdiği avantaj ise anahtarlardaki kuyruklar, kurulan bağlantı bazında oluşturulmadığı için anahtar yapılarının çok karmaşık olmamasıdır. Bu nedenle, ATM ağlarında akış kontrolü için tercih edilmektedir. Ayrıca, akış kontrolü algoritmasının uygulanacağı ve anahtarlarda gerçeklenecek olan denetleyicilerin tasarımı için $\mathcal{H}_\infty$ kontrol teorisinin kullanılması amaçlanmıştır.

ATM ağlarının kullanıcılara sunduğu servisler içinde sadece ABR kullanıcılarının iletim oranları kontrol edilebilir. Bu servis için belirlenen kontrol yaklaşımına göre, bağlantının yeni iletim oranı, bağlantının yolu üzerindeki tıkanık anahtarların belirlediği oranlar ve kaynağın iletim yapmak istediği oran arasından en küçük değer olarak belirlenir. Bu değerin belirlenebilmesi için bağlantının üzerinden geçtiği anahtarlarda bazı karşılaştırmaların yapılması gerekmektedir. Buna göre, eğer karşılaştırma işlemleri yapılmadan önce, RM hücrelerindeki CI alanındaki bit değeri bu işlemin yapılıp yapılmaması için bir ön koşul olarak kullanılırsa, anahtarlar için gereksiz hesap yükü de ortadan kalkar. Eğer gelen RM hücresinde $CI = 1$ yapılmışsa, önceki anahtarlardan birinde tıkanıklık vardır demektir ve anahtar, kendi hesapladığı değerle RM hücresinde yazan değeri karşılaştırmalı ve küçük olan değeri ER alanına yazıp hücreyi bir sonraki noda göndermelidir. Eğer gelen RM hücresinde $CI = 0$ ise, ve anahtarda tıkanıklık söz konusuysa, kendi hesapladığı değeri RM hücresine yazar. Böylece, gerekli olmayan durumlarda da böyle bir karşılaştırmanın yapılmasının önüne geçilmiş olur. Ayrıca,

tıkanıklık yaşayan her anahtar, CI bitinin değerini 1 yapar; yaşamıyorsa da bu bitin değerini değiştirmez ve gelen RM hücrelerindeki CI bitlerinin değeri ne olursa olsun ER alanındaki bilgiyle herhangi bir karşılaştırma yapmadan ve bu alandaki bilgiyi değiştirmeden hücreyi bir sonraki noda gönderir.

Belirlenen iletim oranlarının minimumunun bulunması işlemi ABR kontrol algoritmasının temel özelliklerinden biridir. Ağın modellemesinde, iletim oran komutu ifadelerinde yer alan bu *min* fonksiyonu nedeniyle, oluşturulan model doğrusal değildir. Ancak doğrusal olmayan bu model, şu ana kadar, bu tür sistemler için $\mathcal{H}_\infty$ kontrol yaklaşımı kullanılarak verilmiş olan çözümlerdeki yapılara uymamaktadır. Dolayısıyla, sistem için merkezi bir denetleyici tasarlanamamıştır. Ayrıca, bir anahtardaki denetleyici, kaynak için yeni veri iletim oranı hesaplarken, kendinden bir önceki nodun veri gönderim oranı bilgisine ihtiyaç duymaktadır. Bu oran bilgisi de, o anahtarın tıkanık olup olmamasına göre farklı değerler alır. Bu nedenle, sistem için dış merkezli bir denetleyici tasarımı da yapılamamıştır. Ancak burada, hangi anahtarların tıkanık olacağı önceden bilinirse, kaynağın yeni iletim oranı, en tıkanık olduğu bilinen nodun belirlediği, belli bir geri yönlü gecikmeden sonra, orana eşittir. Buradan, dış merkezli denetleyici tasarımı yapılabilir. Ancak, pratikte böyle bir durum söz konusu olamayacağı için tıkanık nodları önceden belirlemek yerine sistem üzerinde bazı kabullenmeler yapılmıştır. Buna göre, bağlantının üzerinden geçtiği her nod, bağlantı için tıkanık olan nodun kendisi olduğunu kabul ederek bir iletim oranı hesaplar. Daha sonra tüm nodlarca belirlenen oranların en küçüğü yeni iletim oranı komutu olarak kaynağa bildirilir. Eğer bu oran komutunun değeri, kaynağın iletim yapmayı istediği orandan daha küçükse, kaynak bu oranla iletim yapmaya başlar. Tersi durumda ise, kaynak, iletim yapmayı istediği oranı yeni iletim oranı olarak atar. Bu kabullenme ile, çok dar boğazlı durum için denetleyici tasarımı problemi tek dar boğazlı durum için denetleyici tasarımı problemine indirgenmiş olur.

Veri iletişim ağlarında tek dar boğazlı durum için $\mathcal{H}_\infty$ denetleyici tasarımı [8] çalışmasında yapılmıştır. Dolayısıyla, bu tezde, [8] çalışmasında tasarlanmış $\mathcal{H}_\infty$ denetleyicilerin, ATM ağlarında çok dar boğazlı durum için gerçeklemesi yapılmıştır. Gerçeklemesi yapılan denetleyiciler, anahtarlardaki kuyruk uzunluğu bilgilerini kullanarak iletim oranı komutlarını

hesaplamaktadırlar. Kontrol algoritmasının gerçek ağlarda nasıl bir performans göstereceğini belirlemek için yapılan benzetim çalışmalarında MATLAB / SIMULINK paket programı kullanılmıştır. Bu çalışmalar, farklı parametre değerleri ve ağ koşulları için tekrarlanmış ve elde edilen sonuçlar karşılaştırılmıştır.

Yapılan benzetim çalışmalarında, benzer ağ koşulları ve parametre değerleri farklı sayıda kaynak ve anahtar içeren modellere uygulanmıştır. Buradaki amaç, kontrol algoritmasının performansının kaynak ya da anahtar sayıları arttığında nasıl değiştiğinin de gözlenebilmesidir. Elde edilen sonuçlardan, bağlantının yolu üzerindeki anahtar sayısındaki artışın sistem performansını kötü yönde etkilemediği görülmüştür. Ancak, bağlantının yolu üzerinde aralarındaki uzaklığın fazla olduğu çok sayıda anahtar varsa, bu durum, iletimde karşı karşıya kalınan zaman gecikmelerinin ve bu gecikmelerdeki değişimlerin artmasına yol açar. Bu durumun, oran komutlarındaki salınımların sayılarını ve büyüklüklerini arttırdığı; dolayısıyla da, sistemin yatışkan duruma çok daha geç ulaşmasına neden olduğu benzetim çalışmalarında elde edilen sonuçlarda da görülmüştür. Ağa veri gönderen kaynak sayısındaki artışın etkisinin, gönderilen verilerin toplam oranı ve nodların çıkış kapasitelerine bağlı olarak değişim gösterdiği gözlenmiştir. Gönderilen verilerin toplam oranı nod çıkış kapasitesinden çok fazla olsa da eğer kapasite çok hızlı değişimler göstermiyorsa sistem yatışkan duruma uygun bir zaman dilimi içinde ulaşmaktadır. Ancak, gelen veri oranı kapasiteden çok büyükse, yani, çok yoğun bir tıkanıklık söz konusuysa, elde edilen sonuçlarda sık aralıklarla ve büyük değerli salınımlar gözlenmiştir ve sistem uzun bir süre sonunda yatışkan duruma ulaşmıştır.

KAYNAKLAR

[1] TANNENBAUM A. *Computer Networks.* Prentice Hall Inc., (1995).

[2] BONOMI F., FENDICK K. W. *The rate–based flow control framework for the available bit rate ATM service.* IEEE Network Magazine, **25**, 24–39, (1995).

[3] OHSAKI H., MURATA M., SUZUKI H., IKEDA C, MIYAHARA H. *Rate-based congestion control for ATM networks.* ACM SIGCOMM Computer Communication Review, **25**, 60–72, (1995).

[4] ALTMAN E., BAŞAR T. *Multi–user rate–based flow control.* Research report UILU–ENG–96–2227, DC–176, Univ. of Illinois at Urbana–Champaign, (Oct. 1996).

[5] BENMOHAMED L., MEERKOV S. *Feedback control of congestion in store–and–forward datagram networks: The case of a single congested node.* IEEE / ACM Trans. on Networking, **1**, 693–708, (1993).

[6] BENMOHAMED L., MEERKOV S. *Feedback control of congestion in packet switching networks: The case of multiple congested nodes.* International Journal on Communication Systems, **10**, 227–246, (1997).

[7] ÖZBAY H., KALYANARAMAN S., İFTAR A. *On rate–based congestion control in high–speed networks: Design of an $\mathcal{H}_\infty$ based flow controller for single bottleneck.* Proc. of the American Control Conference, Chicago, IL, 2376–2380, (June 1998).

[8] QUET P.–F., ATAŞLAR B., İFTAR A., ÖZBAY H., KALYANARAMAN S., KANG T. *Rate–based flow controllers for communication networks in the presence of uncertain time–varying multiple time–delays.* Automatica, **38**, 917–928, (2002).

[9] BİBEROVİÇ E. *Yüksek Hızlı Veri İletişim Ağlarında Akış Kontrolü.* Yüksek Lisans Tezi, Fen Bilimleri Enstitüsü Elektrik Elektronik Anabilim Dalı, Anadolu Üniversitesi, Eskişehir, 2001.

[10] BİBEROVİÇ E., İFTAR A., ÖZBAY H. *A solution to the robust flow control problem for networks with multiple bottlenecks.* Proc. of the 40th IEEE Conference on Decision and Control, Orlando, FL, U.S.A., 2303–2308, (December 2001).

[11] MUNYAS İ, YELBAŞI Ö, İFTAR A. *Decentralized robust flow controller design for networks with multiple bottlenecks.* Proc. of the European Control Conference, Cambridge, U.K., (September 2003).

[12] OHSAKI H., MURATA M., SUZUKI H., IKEDA C, MIYAHARA H. *Analysis of rate–based congestion control algorithms for ATM networks– Part I: Steady state analysis.* Proc. of GLOBECOM'95, Singapore, 296–303, (November 1995).

[13] OHSAKI H., MURATA M., SUZUKI H., IKEDA C, MIYAHARA H. *Analysis of rate–based congestion control algorithms for ATM networks– Part II: Initial transient state analysis.* Proc. of GLOBECOM'95, Singapore, 1089–1094, (November 1995).

[14] LENGLIZ I, KAMOUN F. *A rate–based flow control method for ABR service in ATM networks.* Computer Networks, **34**, 129–138, (2000).

[15] MASCOLO S., CAVENDISH D. *ATM rate based congestion control using a Smith predictor: An EPRCA implementation.* Proc. of INFOCOM'96, San Fransisco, 569–576, (1996).

[16] CAVENDISH D., MASCOLO S., GERLA M. *Rate based congestion control for multicast ABR traffic.* Proc. of GLOBECOM'96, England, 1114–1118, (November 1996).

[17] MASCOLO S. *Smith's principle for congestion control in high–speed ATM networks.* Proc. of the IEEE Conferance on Decision and Control, San Diego, California, 4595–4600, (December 1997).

[18] GOMEZ–STERN F., FORNES J. M., RUBIO F. R. *Dead–time compensation for ABR traffic control over ATM networks.* Control Engineering Practice, **10**, 481–491, (2002).

[19] http://www.eeng.dcu.ie/ murphy/publ/worse/worse.html

[20] http://hegel.ittc.ukans.edu/projects/abr.html

[21] http://www2.rad.com/networks/1999/atm/serv.htm

[22] JAIN, R. Data flies by ABR service. http://www.cis.ohio-state.edu/ jain/papers/networkw.htm

[23] KALYANARAMAN S. *Traffic Management for the Available Bit Rate (ABR) Service in Asynchronous Transfer Mode (ATM) Networks.* Doktora Tezi, Ohio State University, Columbus, Ohio, (1997).

[24] ZAMES G. *Feedback and Optimal Sensitivity: Model Reference Transformations, Multiplicative Seminorms, and Approximate Inverses.* IEEE Trans. Aut.Control, **26**, 301–320, (1981).

[25] MEINSMA G., ZWART H. *On $\mathcal{H}_\infty$ control for dead–time systems.* IEEE Trans. Aut.Control, **45**, 272–285, (2000).

[26] FRANCIS B. *A Course in $\mathcal{H}_\infty$ Control Theory.* Springer–Verlag, (1987).

[27] FOIAS C., ÖZBAY H., TANNENBAUM A. *Robust Control of Infinite Dimension Systems Frequency Domain Methods.* Springer–Verlag, (1996).

[28] ZHOU K., DOYLE J.C., GLOVER K. *Robust and Optimal Control.* Prentice Hall Inc., (1995).

[29] KWAKERNAAK H. *Robust control and $\mathcal{H}_\infty$ optimization — tutorial paper.* Automatica, **29**, 255–273, (1993).

[30] TOKER O., ÖZBAY H. *$\mathcal{H}_\infty$ optimal and suboptimal controllers for infinite dimensional SISO plants.* IEEE Transactions on Automatic Control, **40**, 751–755, (1995).

EKLER:

Ek–1: Tasarım Parametrelerinin Girildiği ve Alt Fonksiyonların Çağrıldığı Ana Program

Ek–2: Optimal γ'yı ve FIR Filtrelerin Darbe Yanıtını Hesaplayan Program

Ek–3: Optimal γ'nın Hesaplanmasında Kullanılan Alt Fonksiyon

Ek-4: FIR Filtrelerinin Durum Uzayı Gösterimlerinde Çıktı Matrislerinin Elemanlarının Hesaplanmasında Kullanılan Alt Fonksiyon

Ek–5: Nodlarda Gerçeklenen Denetleyicinin SIMULINK Diyagramı

Ek–6: Bir Kaynak ve İki Noddan Oluşan Sistemin SIMULINK Gerçeklemesini Gösteren Diyagram

Ek–7: Bir Kaynak ve Üç Noddan Oluşan Sistemin SIMULINK Gerçeklemesini Gösteren Diyagram

Ek–8: İki Kaynak–Hedef Çifti ve Ara Nodlardan Oluşan Sistemin SIMULINK Gerçeklemesini Gösteren Diyagram

Ek--1: Tasarım Parametrelerinin Girildiği ve Alt Fonksiyonların Çağrıldığı Ana Program

```
% tmain.m

clear all;
close all;
deltat = 0.01;
qdesired1 = 100;
qdesired2 = 100;

n = 2;
nm = [1;1];
h = [1; 2];
delp = [2; 4];
alfa = [1; 1];
beta = [0.1; 0.2];
betaF = [0.01; 0.01];

% actual parameters:
% forward:
hf = [0.5; 1];
delfc = [0; 0];
Af = [0.03; 0.03];
freqf = [pi/50; pi/50];

% backward:
hb = [0.5; 1];
delbc = [0; 0];
Ab = [0.03; 0.03];
freqb = [2*pi/50; 2*pi/50];

topt
sim('sim_1s2bn', 200)
```

Ek--2: Optimal γ'yı ve FIR Filtrelerinin Darbe Yanıtını Hesaplayan Program

```
% topt.m

real_gamma = [];      gamma = [];

for i = 1 : n
      for q = 1 : nm(i,1)
            ar = [];        br = [];        kr = [];        rho = [];

            b = sqrt(sum(((beta + betaF).^2)./((1-beta))));
            d = sqrt(sum(delp.^2));

            beta_h(i,q) = (1/(h(i,q)^2))*sqrt(2)*b;
            delta_h(i,q) = (1/(h(i,q)^3))*sqrt(2)*d;

            options = optimset; options.Display = 'off';
            [gam,val,flag] = fzero('fct1', [beta_h(i,q)+1e-3; 1e5], options, ...
                                                    ...beta_h(i,q), delta_h(i,q));
            if flag < 0 disp('No solution found for gamma'); end
            gamma(i,q) = gam;
            gamma_hat = gam;
            real_gamma(i,q) = gamma(i,q)*(h(i,q)^2)/nm(i,1);

            % optimal gammanın ve w'nun bulunması için ar,br,cr,kr parametrelerinin
              yeniden hesaplanması

            betahat(i,q) = beta_h(i,q);
            delhat(i,q) = delta_h(i,q);

            p1(i,:) = [1 1/gam^2 (1-betahat(i,q)^2/gam^2) betahat(i,q)^2 delhat(i,q)^4
                        ... -(1-betahat(i,q)^2/gam^2)^2/delhat(i,q)^4];
            s1(:,i) = roots(p1(i,:));
            for ii=1:3,
                  if (real(s1(ii,i)) > 0.9999*abs(s1(ii,i)))
                        xp(i,q) = s1(ii,i);
                  end
            end

            cr(i,q) = xp(i,q)^{1/2};
            ar(i,q) = 1/cr(i,q)/delhat(i,q)*(1-betahat(i,q)^2/gam^2)^{1/2};
            br(i,q) = (betahat(i,q)^2/delhat(i,q)^2+2*cr(i,q)-ar(i,q)^2)^{1/2};
            ig = 1/gam^{1/2};
            rho(i,q) = exp(-ig)/delhat(i,q)/(ig+ar(i,1))/(ig^2+br(i,1)*ig+cr(i,q));
            kr(i,q) = (rho(i,q)-1)/gam^{1/2}/(rho(i,q)+1);
            j = (-1)^{1/2};
            ss = j/gam^{1/2};
            v = 1-gam/delhat(i,q)*exp(-ss)*ss^2*(ss-kr(i,q))/(ss+kr(i,q))/(ss+ar(i,1))/...
                  ...(ss^2+br(i,1)*ss+cr(i,q));
```

```
error_i = abs(v);

nE = [h(i,q)^4 0 0 0 -1/gam^2];
dE = [-h(i,q)^4 0 0 0 0];

nF = [gam/delhat(i,q)*h(i,q)^2 0.0000  0.0000];
dF = conv([h(i,q) ar(i,q)],[h(i,q)^2 br(i,q)*h(i,q) cr(i,q)]);

sg = -1;
 nL = sg*[h(i,q) -kr(i,q)];
dL = [h(i,q) kr(i,q)];

n1 = -nF(1,1)*nL;
d1 = [h(i,q) 0];

nn2 = conv(dF,dL) - nE;
np = max(size(nn2));
n2 = nn2(1,2:np);
d2 = nE;

n3 = conv(nF,nL);
d3 = nE;

k1p = gam/delhat(i,q);
k1s = -(gam*kr(i,q))/(delhat(i,q)*h(i,q));
if i == 1
        if q == 1,
                k1p11 = k1p;
                k1s11 = k1s;
        else
                k1p12 = k1p;
                k1s12 = k1s;
        end
 else
        if q == 1,
                k1p21 = k1p;
                k1s21 = k1s;
        else
                k1p22 = k1p;
                k1s22 = k1s;
        end
 end

  % FIR Filter

  [HR,HP,HK] = residue(n2,d2);
  cons = h(i,q)*const;

  conA = zeros(cons-1,cons-1);
  conB = zeros(cons-1,1);
  conC = zeros(1,cons-1);
```

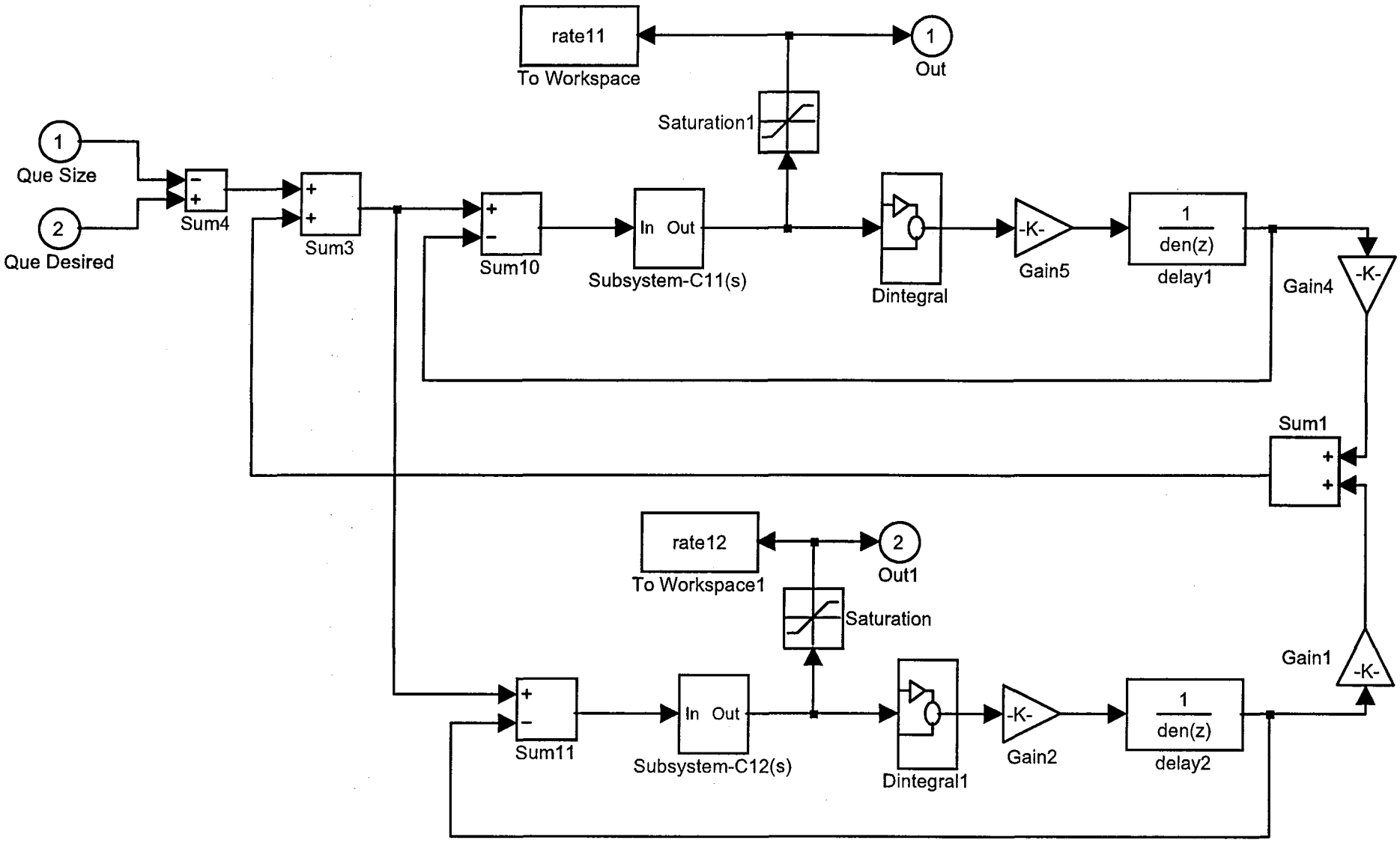
```
            conD = zeros(1,1);

            for coni = 1:cons-2,
                conA(coni,coni+1) = 1;
            end

            conB(cons-1,1) = 1;

            for coni = 1:cons-1,
                conC(1,coni) = himp3((cons-1-coni+1)*deltat,HR,HP);
            end

            conC = deltat*conC;
            conD = conD*deltat;
            if i == 1
                if q == 1,
                    conA11 = conA;
                    conB11 = conB;
                    conC11 = conC;
                    conD11 = conD;
                else
                    conA12 = conA;
                    conB12 = conB;
                    conC12 = conC;
                    conD12 = conD;
                end
            else
                if q ==1,
                    conA21 = conA;
                    conB21 = conB;
                    conC21 = conC;
                    conD21 = conD;
                else
                    conA22 = conA;
                    conB22 = conB;
                    conC22 = conC;
                    conD22 = conD;
                end
            end
        end
        rgamma = [rgamma; rgammai];
        gamm = [gamm; gammai];
end
```

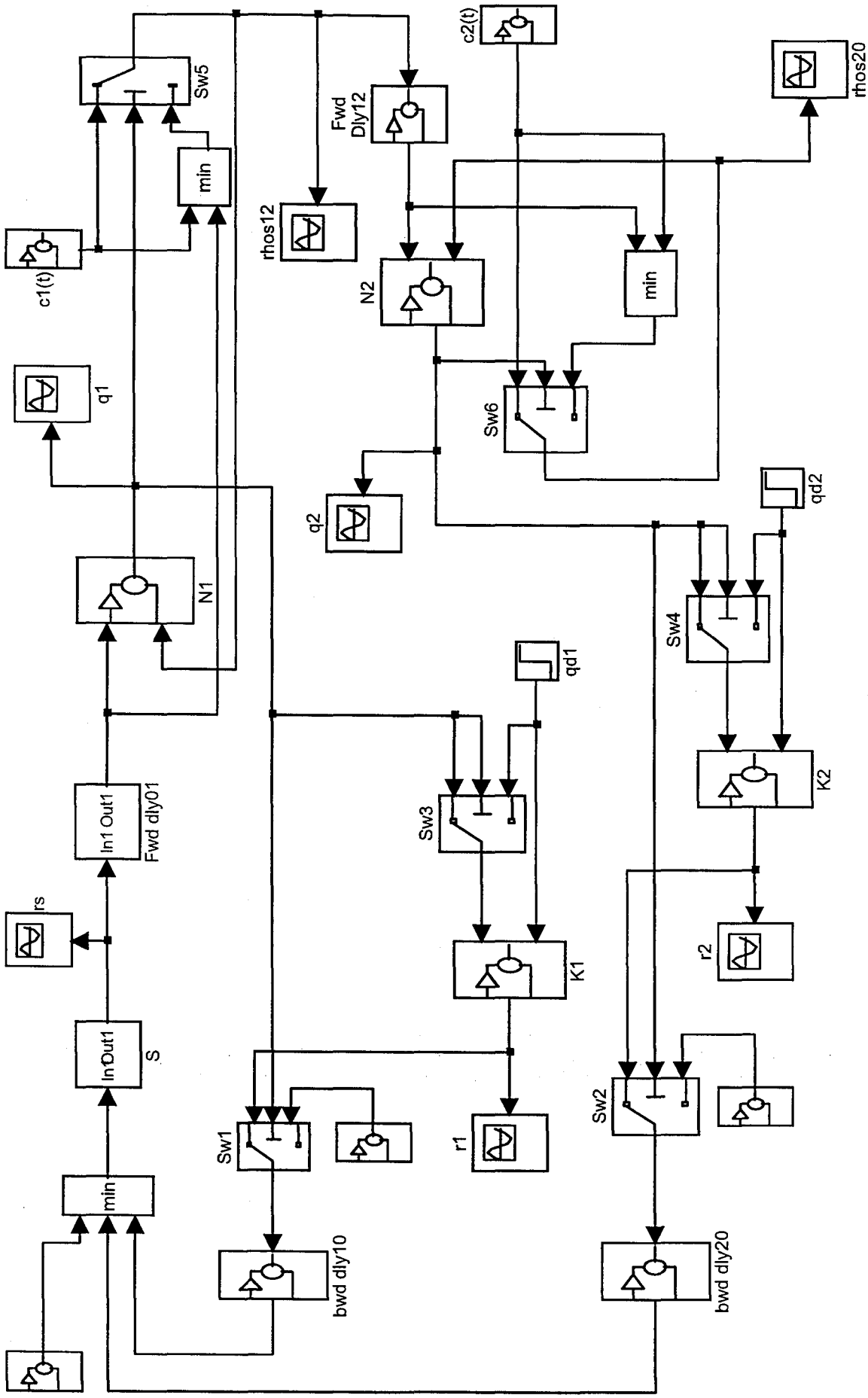
Ek—3: Optimal γ'nın Hesaplanmasında Kullanılan Alt Fonksiyon

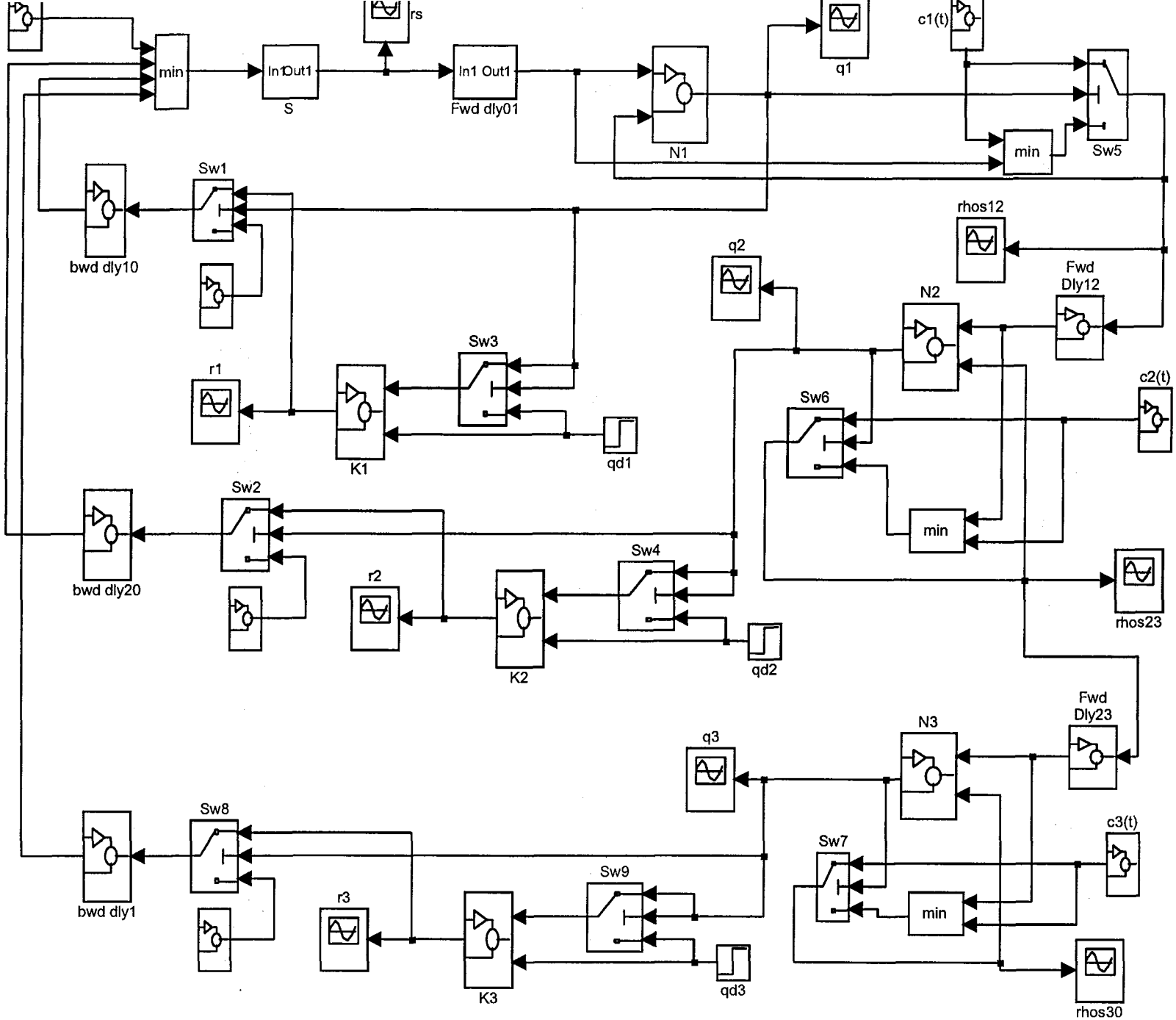
```
% fct1.m

function [F] = fct1(gamma, beta_hiq, delta_hiq);

t = roots([1 1/(gamma^2) (1-(beta_hiq^2)/(gamma^2))*((beta_hiq^2)/(delta_hiq^4))- ...
                ((1-(beta_hiq^2)/(gamma^2))^2)/(delta_hiq^4)]);
x = t(find(real(t)>0));              %3. dereceden polinomun tek koku

c = sqrt(x);
a = (1/(c*delta_hiq))*sqrt(1-(beta_hiq^2)/(gamma^2));
b = sqrt((beta_hiq^2)/(delta_hiq^2)+2*c-a^2);
ro = exp(-1/sqrt(gamma))/delta_hiq/(1/sqrt(gamma)+a)/(1/gamma+c+b/sqrt(gamma));
k = (ro-1)/(sqrt(gamma)*(ro+1));

F = (-1/sqrt(gamma)+angle((i/sqrt(gamma)-k)/(i/sqrt(gamma)+k))-
angle(i/sqrt(gamma)+a)-...     ...angle(c-1/gamma+i*b/sqrt(gamma))+pi);
```

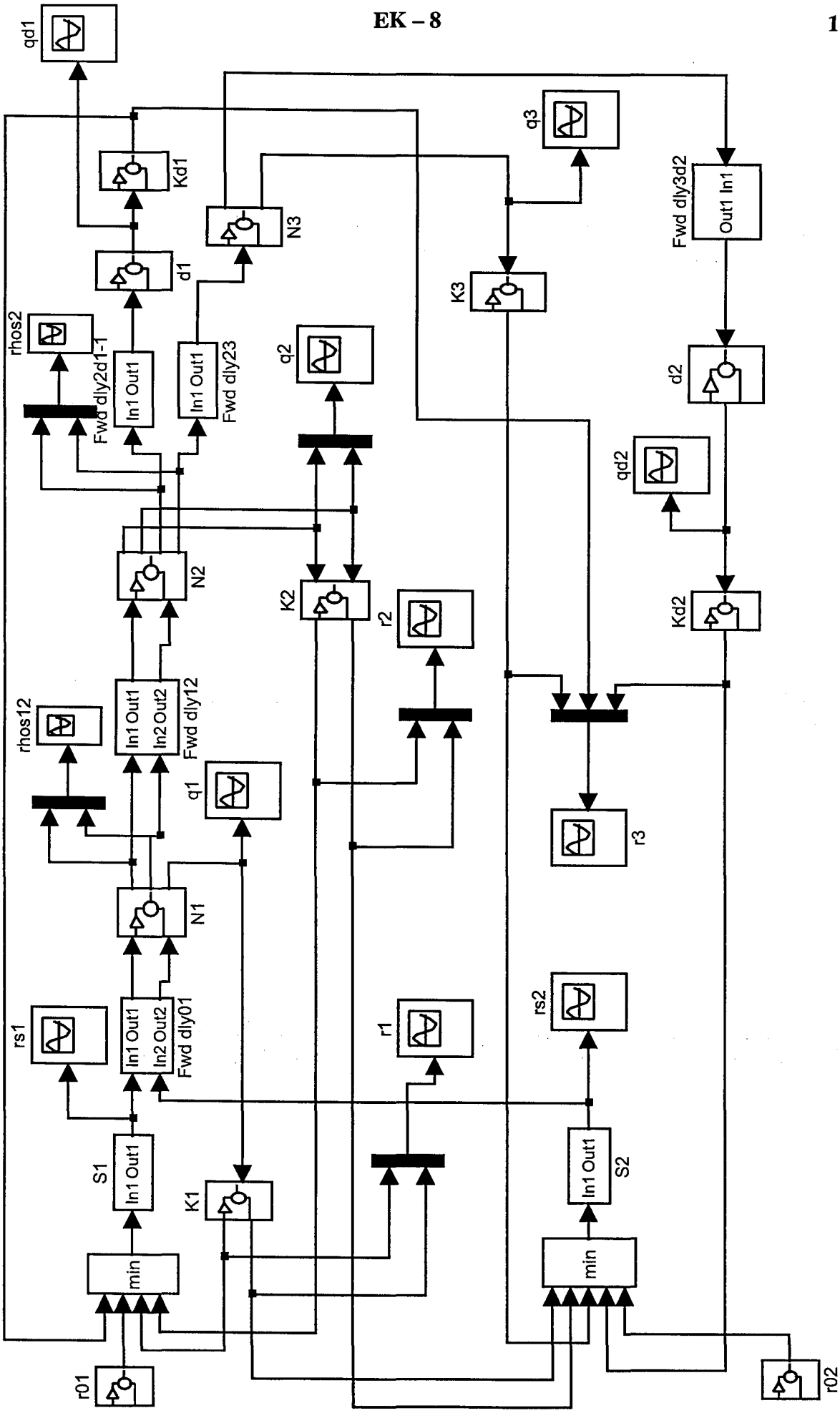
Ek—4: FIR Filtrelerinin Durum Uzayı Gösterimlerinde Çıktı Matrislerinin Elemanlarının Hesaplanmasında Kullanılan Alt Fonksiyon

```
% himp3.m

function [v] = himp3(t,HR,HP)

v = real(HR(1)*exp(HP(1)*t)+HR(2)*exp(HP(2)*t)+HR(3)*exp(HP(3)*t)...

+HR(4)*exp(HP(4)*t));
```


rate11
To Workspace
1
Out
Saturation1
1
Que Size
2
Que Desired
Sum4
Sum3
Sum10
In Out
Subsystem-C11(s)
Dintegral
-K-
Gain5
1
den(z)
delay1
Gain4
-K-
Sum1
rate12
To Workspace1
2
Out1
Saturation
Gain1
-K-
Sum11
In Out
Subsystem-C12(s)
Dintegral1
-K-
Gain2
1
den(z)
delay2


Sw5
min
c1(t)
rhos12
Fwd Dly12
c2(t)
rhos20
N2
min
q1
Sw6
q2
qd2
N1
Sw4
qd1
In1 Out1
Fwd dly01
Sw3
K2
rs
K1
r2
In1Out1
S
Sw1
r1
Sw2
min
bwd dly10
bwd dly20


min
In1 Out1
S
rs
In1 Out1
Fwd dly01
N1
q1
c1(t)
min
Sw5
Sw1
bwd dly10
rhos12
q2
Fwd
Dly12
N2
Sw3
r1
K1
qd1
c2(t)
Sw6
Sw2
bwd dly20
r2
Sw4
K2
qd2
min
rhos23
Fwd
Dly23
N3
q3
Sw8
bwd dly1
r3
Sw9
K3
qd3
Sw7
c3(t)
min
rhos30


qd1
Kd1
d1
N3
q3
K3
Fwd dly3d2
Out1 In1
d2
rhos2
Fwd dly2d1-1
In1 Out1
In1 Out1
Fwd dly23
q2
qd2
Kd2
N2
K2
r2
rhos12
In1 Out1
In2 Out2
Fwd dly12
q1
r3
N1
rs1
In1 Out1
In2 Out2
Fwd dly01
r1
rs2
S1
In1 Out1
K1
S2
In1 Out1
min
min
r01
r02