

DELAY-BOUNDED RATE ADAPTIVE SHAPER FOR
TCP TRAFFIC IN DIFFSERV INTERNET

A THESIS

SUBMITTED TO THE DEPARTMENT OF ELECTRICAL AND

ELECTRONICS ENGINEERING

AND THE INSTITUTE OF ENGINEERING AND SCIENCES

OF BILKENT UNIVERSITY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

MASTER OF SCIENCE

By

Yakup Balkaş

September 2002

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Asst. Prof. Dr. Ezhan Karařan(Supervisor)

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Asst. Prof. Dr. Nail Akar

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Asst. Prof. Dr. İbrahim K rpeođlu

Approved for the Institute of Engineering and Sciences:

Prof. Dr. Mehmet Baray
Director of Institute of Engineering and Sciences

ABSTRACT

DELAY-BOUNDED RATE ADAPTIVE SHAPER FOR TCP TRAFFIC IN DIFFSERV INTERNET

Yakup Balkaş

M.S. in Electrical and Electronics Engineering

Supervisor: Asst. Prof. Dr. Ezhan Karaşan

September 2002

Applications with different quality requirements set out the need for different Qualities of Service (QoS) to be provided in Internet. Differentiated Services (DiffServ) model is an architecture proposed to provide QoS in the Internet in a scalable way. Assured Forwarding Per Hop Behavior (AF PHB) is a QoS service class which provides a loss sensitive service. The DiffServ Service Provider (SP) delivers services to customers where traffic parameters are quantified in a Service Level Agreement (SLA). The incoming traffic from customers are policed in order to make sure that they meet the specifications in the SLA. The portion of traffic that is nonconformant with the SLA is not guaranteed to receive the service quality specified in the SLA. Shapers delay nonconformant packets in order to increase the ratio of traffic that is within the bounds specified in the SLA. If nonconformant traffic is tolerated in the SP network up to some extent, increasing the ratio of traffic that is complying with specifications in the SLA may lead to unnecessary delaying of packets and may decrease throughput. In this thesis, a shaper, called Delay-Bounded Rate-Adaptive Shaper (DBRAS), is introduced which tries to increase the ratio of traffic that conforms to the SLA while satisfying an upper-bound (D_{max}) in the amount of delay it can apply

to incoming packets (shaping delay). By avoiding unnecessarily large shaping delays, it is shown that throughput is increased. In order to have the shaper to adapt to changes in network topology, traffic, and different propagation delays, an adjustment algorithm is proposed where the shaper dynamically adjusts its D_{max} value in order to increase throughput. The resulting shaper is called Dynamic DBRAS (D-DBRAS). The heuristic adjustment algorithm is greedy in that it adapts the maximum shaping delay in the direction where throughput increases. Results obtained from simulations show that throughput of TCP in AF PHB shaped by D-DBRAS can be increased by up to 65% compared with unshaped traffic. Simulations are performed in order to analyze effects of parameters such as propagation delay, buffer threshold levels, and offered traffic on the performance of D-DBRAS. It is also shown through simulations that by using the adjustment algorithm, the maximum shaping delay, D_{max} , converges to regimes where throughput increases in response to changes in offered traffic.

Keywords: DiffServ, AF PHB, TCP, shaper, delay bound, throughput

ÖZET

FARKLILAŞMIŞ HİZMETLER İNTERNETİ'NDE TCP TRAFFİĞİ İÇİN ÖNERİLEN BEKLETME SÜRESİ SINIRLI İLETİM HIZI UYUMLU DÜZENLEYİCİ

Yakup Balkaş

Elektrik ve Elektronik Mühendisliği Bölümü Yüksek Lisans

Tez Yöneticisi: Yrd. Doç. Dr. Ezhan Karaşan

Eylül 2002

Günümüz İnternet'inde sağlanan farklı yeterlilik koşullarına sahip işlemlerin yaşadığı sorunlar İnternet'te farklı özelliklere sahip hizmetler sunulmasını gerekli kılmaktadır. Farklılaşmış Hizmetler İnterneti bu ihtiyacı yaygınlaşma sırasında sorun yaşamayacak bir biçimde karşılamayı amaçlamaktadır. Sağlanması planlanan Şartlı Yönlendirilmeli Ağ Yönlendiricisi Hizmeti (AF PHB), paket kayıplarına duyarlı bir hizmet sunar. Bu ve bunun gibi hizmetleri, Farklılaşmış Hizmetler Sağlayıcıları kullanıcılarına kullanım ve hizmet özelliklerinin ölçülebilir yeterlilik koşullarıyla belirtildiği Hizmet Özellikleri Sözleşmesi'ni (SLA) imzadıktan sonra sunmaya başlarlar. Kullanıcı trafiğinin SLA'da belirtilen kullanım özelliklerinin dışında kalan bölümü hizmetin belirlenmiş yeterlilik koşullarından mahrum bırakılabilir. Düzenleyici SLA'ya uygun olmayan paketleri bekleterek trafiğin SLA'da belirtilen kullanım koşullarına uygun olan bölümünü arttırmaya çalışır. Eğer Farklılaşmış Hizmetler Sağlayıcısı, ağında, SLA'da belirtilen kullanım özellikleri dışı trafiğin yönlendirilmesine belli sınırlar dahilinde izin veriyorsa, düzenleyici paketleri bekleterek trafiğin veri ulaştırma hızının daha yüksek değerlere çıkmasına engel olabilir. Bu tez çalışmasında önerilen düzenleyici -ki

Bekletme Süresi Sınırlı İletim Hızı Uyumlu Düzenleyici (DBRAS) olarak adlandırılmıştır- trafiğin mümkün olduğunca SLA'ya uygun hale getirilmesi için çalışırken paket bekletme süresini de belli bir üst sınır değer -ki bu değer D_{max} olarak adlandırılmıştır- altında tutar. Gereksiz derecede yüksek bekletme sürelerinden kaçınarak yapılan düzenlemenin veri ulaştırma hızını arttırdığı gözlenmiştir. Ağ mimarisi, trafiğin kullanım özellikleri veya paketlerin ağda yayılma süresinde oluşacak değişikliklerin DBRAS'ın, veri ulaştırma hızına yapacağı katkıyı engellememesi için DBRAS'ın D_{max} 'ı veri ulaştırma hızının artışı hedefleyerek değiştirebilmesini sağlayan bir D_{max} ayarlama algoritması önerilmiştir. Ortaya çıkan yeni düzenleyiciye Dinamik DBRAS (D-DBRAS) adı verilmiştir. Önerilen buluşsal algoritma açgözlü bir yaklaşımla D_{max} 'ı veri ulaştırma hızını arttıracak yönde değiştirir. Benzetim sonuçları D-DBRAS'ın, TCP trafiğinin veri ulaştırma hızını, düzenlenmeyen bir trafiğinkine oranla %65 arttırabileceğini göstermiştir. Bunun yanında, paketlerin ağda yayılma süresi, ağ arabelleklerinin sınır seviyeleri ve ağdaki trafik yükü gibi parametrelerin D-DBRAS'ın verimini nasıl etkilediğini incelemek için benzetimler de yapılmıştır. Ağdaki trafik yükünün zaman içinde değiştiği ağlarda yapılan benzetimler, bu değişiklikler karşısında D-DBRAS'ın, TCP trafiğinin veri ulaştırma hızını arttırabildiği rejimlerine yakınsama becerisini göstermiştir.

Anahtar kelimeler: Farklılaşmış Hizmetler İnterneti, AF PHB, TCP, düzenleyici, bekletme süresi üst sınır değeri, veri ulaştırma hızı

ACKNOWLEDGMENTS

I gratefully thank my supervisor Asst. Prof. Dr. Ezhan Karayan for his supervision, enlightening guidance, motivating patience, and improving suggestions throughout my graduate study and the development of this thesis.

I would like to thank Asst. Prof. Dr. Nail Akar for his invaluable contributions to the initial phases of this thesis work and also for his helpful suggestions on my thesis.

I would also like to thank Asst. Prof. Dr. İbrahim Körpeođlu for his helpful suggestions on my thesis.

Contents

1	Introduction	1
2	Differentiated Services Architecture	7
2.1	An Overview of DiffServ Model	9
2.2	Per Hop Behavior Proposals	13
2.2.1	Expedited Forwarding Per Hop Behavior	14
2.2.2	Assured Forwarding Per Hop Behavior	14
2.3	Active Queue Management Schema	15
2.3.1	RED	16
2.3.2	Variants of RED for AF PHB	17
2.4	Proposed Marking Mechanisms	18
2.4.1	Token Bucket Marker (TBM)	18
2.4.2	Other Markers in the Literature	18
2.5	Traffic Shapers	19
2.5.1	Rate Adaptive Shaper (RAS)	20

2.5.2	Green Rate Adaptive Shaper (green RAS)	21
2.5.3	Other Shaper Algorithms in the Literature	22
2.6	Traffic Conditioner Proposals	23
3	Delay-Bounded, Rate-Adaptive Shaper (DBRAS)	25
3.1	Required Capabilities	25
3.2	DBRAS Design	26
3.3	DBRAS Shaping Algorithm	27
3.4	Performance Analysis of DBRAS	31
3.4.1	Simulation Topology	31
3.4.2	Simulation Results	34
4	Dynamic DBRAS (D-DBRAS)	38
4.1	Required Capabilities	38
4.2	D_{max} Adaptation Algorithm	39
4.3	Simulation Topology	41
4.4	Simulation Results	42
4.4.1	Selection of a Proper Value of δ	43
4.4.2	Effects of Some Parameters on Performance of D-DBRAS	67
4.4.3	Dynamic Network Load Simulations	83
5	Conclusions	99

List of Figures

2.1	A typical DiffServ SP network.	11
2.2	Ingress edge router internals.	12
2.3	Internals of a core router.	13
3.1	Design of the DBRAS.	27
3.2	Simulation Topology.	32
3.3	Simulations with $P_D = 20$ msec.	35
3.4	Simulations with $P_D = 5$ msec.	36
4.1	Simulation Topology for D-DBRAS.	41
4.2	D_{max} of s0 vs. time with different initial D_{max} values, $\delta = 0.8$ msec.	44
4.3	D_{max} of s0 vs. time with different initial D_{max} values, $\delta = 2$ msec.	45
4.4	D_{max} of s0 vs. time with different initial D_{max} values, $\delta = 4$ msec.	46
4.5	D_{max} of s0 vs. time with different initial D_{max} values, $\delta = 6$ msec.	47
4.6	D_{max} of s1 vs. time with different initial D_{max} values, $\delta = 0.8$ msec.	48
4.7	D_{max} of s1 vs. time with different initial D_{max} values, $\delta = 2$ msec.	49

4.8	D_{max} of s1 vs. time with different initial D_{max} values, $\delta = 4$ msec.	50
4.9	D_{max} of s1 vs. time with different initial D_{max} values, $\delta = 6$ msec.	51
4.10	D_{max} of s0 vs. time with different initial D_{max} values, $P_D = 5$ msec, $\delta = 0.8$ msec.	52
4.11	D_{max} of s0 vs. time with different initial D_{max} values, $P_D = 5$ msec, $\delta = 0.8$ msec cont.'ed.	53
4.12	D_{max} of s0 vs. time with different initial D_{max} values, $P_D = 5$ msec, $\delta = 2$ msec.	54
4.13	D_{max} of s0 vs. time with different initial D_{max} values, $P_D = 5$ msec, $\delta = 2$ msec cont.'ed.	55
4.14	D_{max} of s0 vs. time with different initial D_{max} values, $P_D = 5$ msec, $\delta = 4$ msec.	56
4.15	D_{max} of s0 vs. time with different initial D_{max} values, $P_D = 5$ msec, $\delta = 4$ msec cont.'ed.	57
4.16	D_{max} of s0 vs. time with different initial D_{max} values, $P_D = 5$ msec, $\delta = 6$ msec.	58
4.17	D_{max} of s0 vs. time with different initial D_{max} values, $P_D = 5$ msec, $\delta = 6$ msec cont.'ed.	59
4.18	D_{max} of s1 vs. time with different initial D_{max} values, $P_D = 5$ msec, $\delta = 0.8$ msec.	60
4.19	D_{max} of s1 vs. time with different initial D_{max} values, $P_D = 5$ msec, $\delta = 0.8$ msec cont.'ed.	61

4.20	D_{max} of s1 vs. time with different initial D_{max} values, $P_D = 5$ msec, $\delta = 2$ msec.	62
4.21	D_{max} of s1 vs. time with different initial D_{max} values, $P_D = 5$ msec, $\delta = 2$ msec cont.'ed.	63
4.22	D_{max} of s1 vs. time with different initial D_{max} values, $P_D = 5$ msec, $\delta = 4$ msec.	64
4.23	D_{max} of s1 vs. time with different initial D_{max} values, $P_D = 5$ msec, $\delta = 4$ msec cont.'ed.	65
4.24	D_{max} of s1 vs. time with different initial D_{max} values, $P_D = 5$ msec, $\delta = 6$ msec.	66
4.25	D_{max} of s1 vs. time with different initial D_{max} values, $P_D = 5$ msec, $\delta = 6$ msec cont.'ed.	67
4.26	Throughput vs. initial D_{max} graph for $P_D = 20$ msec case.	68
4.27	Throughput vs. initial D_{max} graph for $P_D = 5$ msec case.	69
4.28	Green ratio vs. initial D_{max} for $P_D = 20$ msec case.	70
4.29	Green ratio vs. time for $P_D = 5$ msec case.	70
4.30	Throughput vs. initial D_{max} , for the less-congested network and $P_D = 20$ msec.	74
4.31	Throughput vs. initial D_{max} , for the less-congested network and $P_D = 5$ msec.	74
4.32	Green Ratio vs. initial D_{max} , for the less-congested network and $P_D = 20$ msec.	75

4.33 Green Ratio vs. initial D_{max} , for less-congested network and P_D = 5 msec.	76
4.34 Network topology of 0s10u case.	77
4.35 Network topology of 10s0u case.	78
4.36 D_{max} of s0 vs. time with different initial D_{max} values, ap = 200 sec.	84
4.37 Throughput of s0 vs. time with different initial D_{max} values, ap = 200 sec.	85
4.38 D_{max} of s0 vs. time with different initial D_{max} values, ap = 100 sec.	86
4.39 Throughput of s0 vs. time with different initial D_{max} values, ap = 100 sec.	87
4.40 D_{max} of s0 vs. time with different initial D_{max} values, ap = 50 sec.	88
4.41 Throughput of s0 vs. time with different initial D_{max} values, ap = 50 sec.	89
4.42 D_{max} of s0 vs. time with different initial D_{max} values, ap = 25 sec.	90
4.43 Throughput of s0 vs. time with different initial D_{max} values, ap = 25 sec.	91
4.44 D_{max} of D-DBRAS of s0 vs. time for dynamic network load Sce. 1.	93
4.45 Throughput of s0 vs. time for dynamic network load Sce. 1. . . .	94
4.46 D_{max} of D-DBRAS of s0 vs. time for dynamic network load Sce. 2.	95
4.47 Throughput of s0 vs. time for dynamic network load Sce. 2. . . .	96
4.48 D_{max} of D-DBRAS of s0 vs. time for dynamic network load Sce. 3.	97

4.49 Throughput of s0 vs. time for dynamic network load Sce. 3. . . . 98

List of Tables

4.1	Actual TCP window size measurements.	71
4.2	End-to-end delay measurements (in msec).	71
4.3	Shaping delay measurements (in msec).	72
4.4	Dropping probability measurements.	72
4.5	Throughput levels achieved by different ratios of shaped traffic in the network (in Kbps).	79
4.6	Green ratios in topologies with different shaped traffic ratio in the network.	80
4.7	Throughput levels achieved with different RED parameter values (in Kbps).	81
4.8	Dropping probabilities at congested queue of networks with dif- ferent RED parameter values.	82
4.9	Dynamic network load scenarios.	92

To My Family . . .

Chapter 1

Introduction

In its short history, the Internet has evolved in great scales. It was founded as a means of communication among colleagues in different universities and later more services are provided such as file transfer, remote access, e-mail, audio/video on-demand, and audio/video streaming applications. These applications have some similarities while they differ in some aspects. Differences arise from different requirements of these services. For example, file transfers typically last long and consist of a large number of packets each carrying a part that is vital for the concatenation of the file at the receiver. On the other hand, audio and video streaming applications require that packets are delivered in a timely fashion and that the variation of time interval between two consecutively received packets is bounded. If the requirements are re-stated in the Internet terms, file transfer applications require low packet loss rate, whereas real-time audio and video streaming applications require low delay and variation of delay (jitter).

To satisfy the needs stated above, some efforts on placing regulations regarding the transportation of application packets in the Internet are currently carried out. Initial efforts on this subject lead to definitions of a number of transport protocols. Transmission Control Protocol (TCP) was designed to satisfy low (or

no) packet loss. It overcomes the problem through establishing a connection between sender and receiver and making sure that receiver successfully receives packets in order and without loss. The establishment of a connection and reliability mechanisms increase overhead. Also TCP has a mechanism to detect, respond to, and avoid congestion in the Internet. Congestion is the condition that queues of network are fed by traffic amount of which is higher than capacity and forwarding capabilities of queues. Basically, TCP decreases its packet sending rate when it detects a congestion. Then it increases its sending rate until it encounters congestion again. Changes in sending rate of TCP increase delay and jitter. So, TCP is unable to satisfy the low delay and low jitter requirements. A group of transport protocols were used to answer requirements of real-time applications. These protocols evolved from User Datagram Protocol (UDP) which aims to transmit packets as soon as possible, to save time. UDP does not provide any retransmission mechanism for recovering from packet losses. Instead, it is the responsibility of the application layer to support any form of reliability, if necessary.

TCP and UDP, as stated previously, have different considerations. This, however, lead to some problems when they co-exist. UDP traffic affects TCP traffic adversely. While TCP aims to reduce congestion in the Internet by shaping, UDP traffic receives a larger share of network resources (queue occupancy, ratio of bandwidth used). As a result, TCP performance degrades when TCP and UDP share common network resources. This is referred to as the fairness problem in the Internet. Some research has been done to avoid the unfair allocation of network resources among TCP and UDP traffic. There were two methods used for this purpose. In the first approach, UDP is proposed to have a congestion control mechanism similar to TCP. In the second approach, the architecture of Internet is proposed to be modified so that traffic with different quality requirements, e.g. real-time audio/video streaming and data, are separated from each other. This way, Quality of Service (QoS) is introduced into the Internet. There

are two methods of embedding QoS into the Internet. In the first method, resources are reserved in the network in order to satisfy requirements of individual flows. This method is named the Integrated Services (IntServ) model. In the second model, the traffic using the Internet are classified according to their requirements into a small number of classes among which network resources are shared. This method is called the Differentiated Services (DiffServ) model. The fact that DiffServ reservations are done for a small number of aggregate traffic instead of individual flows as in the case of IntServ model makes DiffServ model superior over the IntServ model in terms of scalability.

DiffServ model classifies the network elements (nodes) into two categories: edge nodes and core nodes. In the edge, traffic is mapped to service classes and in the core, packets are served based on the service class they are assigned. In the DiffServ architecture, a service level agreement (SLA) specifies details of the service to be provided by Service Provider (SP) to the customer traffic. SLA specifies service parameters to be provided by the SP such as throughput (the amount of traffic received by receiver in unit time), burst size (the number of packets that can be sent consecutively), delay, jitter, and packet loss ratio (dropping probability). SLAs are typically negotiated between the customer and the SP before any service is initiated. At the node where customer traffic enters the SP network, policing of the traffic is performed by the SP to enforce SLA specifications. SP does not give any guarantee on the portion of traffic that does not conform with SLA (called out-of-profile packets). For this reason, customers try to reduce the amount of nonconformant traffic using shapers which delay packets in order to decrease the number of out-of-profile packets. Delaying may require large buffer space in the shaper. Moreover, in networks where nonconformant traffic is tolerated to some extent, shaping may even decrease the throughput since excessive shaping generates unnecessarily large delays.

In this thesis, a shaper, called Delay-Bounded Rate Adaptive Shaper (DBRAS), is proposed which tries to reduce nonconformant packets while obeying an upper-bound, D_{max} , on the shaping delay which is defined as the time interval between the time when all bytes of a packet are received and the time when all bytes of the the packet are sent downstream. This way, buffer space in the shaper can be limited and the decrease in the throughput due to unnecessarily large shaping delays can be avoided. DBRAS is simulated to work in a network where resources are less than the amount of traffic using the network. Bursty traffic sources with identical distribution of packet generation and QoS requirements are used where some of the traffic streams are shaped while the remaining are unshaped. Numerical results show that average throughput of shaped traffic can be increased up to 75% compared to unshaped traffic if D_{max} is chosen optimally. When the propagation delay in the network is changed, it is observed that the optimum value for D_{max} also changes even for the same network topology and traffic sources. Also it is observed that the gain of shaping (ratio of average throughputs of shaped traffic and unshaped traffic with statistically identical characteristics) makes two peaks as D_{max} increases, i.e., there are local maxima. Exceeding a certain D_{max} value, measured throughput values remain fixed at a value smaller than those two peaks no matter how much D_{max} is increased. This saturating value for D_{max} increases when propagation delay in the network is decreased.

The observation that the optimum value of D_{max} varies for different network topologies and traffic characteristics leads to Dynamic DBRAS (D-DBRAS) where D_{max} is adjusted periodically in order to increase throughput in response to changing network conditions. D-DBRAS uses a greedy algorithm which adjusts D_{max} in constant steps, δ , in a direction that increases the throughput of TCP. If throughput measured in the current period is larger than the one measured in the previous period, then D_{max} is applied the same change (increase or decrease) as the one applied in the previous period. Otherwise, D_{max} is changed

in the opposite direction of the adjustment in the previous period. The shaper also keeps track of the ratio of nonconformant traffic and avoids the case when all packets satisfy specifications in SLA which was found to result in non-optimum throughput. It is observed that in order to have convergence of D_{max} in an acceptable duration, the step size δ has to be determined appropriately.

With value of δ selected properly for convergence, simulations show that throughput for shaped traffic can be increased by 38-58% compared to unshaped traffic depending on the initial value of D_{max} . Hence, the cases where using fixed nonoptimum value of D_{max} that leads to loss of performance compared to no shaping are completely eliminated by using D-DBRAS. When D-DBRAS is used, the ratio of in-profile packets are increased by 23-70% compared to unshaped traffic.

When the propagation delay in the network is changed, similar results are obtained. When the level of congestion in the network is decreased, throughput for traffic shaped by D-DBRAS is increased by 36-42% compared to unshaped traffic. This gain is lower than the more resource constrained case since in the less congested case, most of nonconformant packets can also be delivered to the destination.

It is observed that in a congested network, as the ratio of shaped traffic in the network increases, average throughput of shaped traffic decreases. Nevertheless, when all traffic in the network is shaped by D-DBRAS, average throughput achieved is 4-45% larger than the average throughput achieved when no traffic in the network is shaped. This is the result of more efficient usage of network resources when the traffic is shaped, i.e., more packets can be marked as conformant at the expense of some additional delay. Furthermore, total throughput in the network increases as the ratio of shaped traffic in the network increases.

It is also observed that buffer threshold levels in the SP network that determine how nonconformant packets are handled in the core network can affect the throughput. When nonconformant packets are handled in a less tolerable fashion, the throughput of a shaped traffic can be 65% better than average throughput of unshaped traffic having the same statistical characteristics. On the other hand, when nonconformant packets are treated in a relatively tolerable manner, average throughput of shaped traffic can be up to 24% less than throughput of unshaped traffic with the same traffic characteristics and QoS requirements in a congested network. Therefore, performance of the shaper is significantly affected by how SP network handles out-of-profile packets.

This thesis is organized as follows: In Chapter 2, the DiffServ Architecture is discussed. Mechanisms and devices that comprise DiffServ Architecture are described supplemented with some of their proposed forms. In Chapter 3, the shaper DBRAS is introduced. Required capabilities of DBRAS, its design, and implementation of the simulations for testing its performance are described. In Chapter 4, the shaper D-DBRAS is introduced. The motivations behind the extension from DBRAS to D-DBRAS, algorithm used for D_{max} adaptation are presented. Simulations for performance evaluation of D-DBRAS are described and effects of some parameters on this performance are analyzed.

Chapter 2

Differentiated Services Architecture

The Internet has grown both geographically and in content in great scales during last thirty years. The reason behind this growth is the simplicity of connecting to it. Devices that are connected to the Internet use a protocol named Internet Protocol (IP). All types of traffic using IP are handled in the same manner by the network. There is no guarantee that a packet will reach its destination, neither is there a guarantee on the amount of time it will take until the packet reaches its destination. For different types of traffic, these uncertainties have different importance. For example, a file transfer application requires no packet loss; whereas it does not consider the delay its packets face in the Internet. On the other hand, real-time traffic like audio and video streaming applications are not keenly interested in the individual packet losses, but they are very sensitive to the delay and the variation of delay (jitter) for their packets.

To satisfy the needs of traffic types with different requirements, transport protocols are used: The reliability of delivering packets is satisfied by using

Transmission Control Protocol (TCP) which provides a reliable service by establishing a connection between the source and the receiver and ensuring successful transmissions of packets through the Internet. In addition to this property, TCP has a mechanism that tries to detect, avoid and respond to congestion faced by the packets in the Internet. The delay requirements are currently addressed by using transport protocols that are based on User Datagram Protocol (UDP) which makes no attempt to ensure packet delivery. UDP does not exercise any congestion control mechanism so that packets experience smaller delays with this less complicated transport protocol. Having solved two problems by using two different transport protocols, however, does not lead to a fair Internet service. In fact, TCP traffic may obtain smaller throughput (amount of traffic received as consecutive packets by its receiver in unit time) than UDP traffic especially when the network is congested.

To solve the fairness problem in the Internet, there have been different proposals. Some researchers tried to find the solution while keeping the Internet structure as it is, whereas some work which received the majority of the researcher's support considered modifications to the structure of the Internet. Adding congestion detection and avoidance mechanisms to all traffic using the Internet was an idea from the first group of researchers. The second class of research, adding quality of service (QoS) support into the structure of Internet, is divided into two subclasses. The first subclass aimed to support QoS by reserving requirements of individual traffic flows in the Internet, namely the Integrated Services (IntServ) model. IntServ model requires too much processing in the network core, which makes it unscalable and disadvantageous over the second model, Differentiated Services (DiffServ) model. DiffServ model has reservation in the network core for only a small number of service classes. It assigns traffic flows to one of those service classes based on their requirements at the edges of the network. This gives the property of scalability to DiffServ model.

In Section 2.1, an overview of DiffServ model is presented. In Section 2.2, proposed forms of mechanisms at the core of the DiffServ network, Per Hop Behaviors are described. In Section 2.3, active queue management schema used in the DiffServ architecture are discussed. Some marking mechanisms used in DiffServ networks are introduced in Section 2.4. In Section 2.5, shaping algorithms that have been previously proposed in the literature are presented. Traffic Conditioners proposed in the literature are discussed in Section 2.6.

2.1 An Overview of DiffServ Model

In DiffServ architecture, buffer space and bandwidth resources are distributed among a small number of service classes in every network node [1]. These service classes might satisfy different requirements such as delay and loss. The Service Provider (SP) accepts to serve a customer after making a formal agreement with the customer named as Service Level Agreement (SLA). Major parameters in the SLA specifying the traffic and service required are described below [2].

Contents of an SLA:

1. Scope: Boundaries to identify geographical/topological region in which the QoS is to be enforced uniquely.
2. Flow Id : Identification of IP datagrams of customer traffic such as Type of Service Byte in IPv4 header and Traffic Class Byte in IPv6 header [3] (it will be referred to as DS field in the remaining part).
3. Traffic Conformance Testing: The set of parameters (like sending rate, maximum number of consecutive packets (a burst) in the traffic) and their values that customer agrees to obey and also the algorithm to be used for confirmation of this obedience.

4. Excess Treatment: Method for applying to excess traffic which is the portion of traffic that fails traffic conformance test (also named as out-of-profile packets). Possible methods are dropping, marking, and shaping.
5. Performance Guarantees: Guarantees supplied to the in-profile packets of the customer, i.e., packets obeying parameter values for Traffic Conformance, in terms of
 - (a) delay: The maximum packet transfer delay from ingress (entrance point of the customer traffic into SP network) to egress (departure point of the customer traffic out of SP network) router.
 - (b) jitter: The maximum packet transfer delay variation from ingress to egress router.
 - (c) packet loss: The ratio of the number of in-profile packets lost between ingress and egress routers to the total number of in-profile packets injected at ingress router.
 - (d) Throughput: Rate measured at the destination counting all received packets.

Out-of-profile portion of traffic receives no guarantees.

6. Service Schedule: Statement of time intervals when the service is available in terms of:
 - (a) Time of the day range
 - (b) Day of the week range
 - (c) Month of the year range
7. Reliability: Maximum allowed mean downtime per year (MDT) and the maximum allowed time to repair in case of service breakdown.

Having signed an SLA, depending on the QoS requirements of the service, SP chooses one of service classes it supports in its network as the service class to

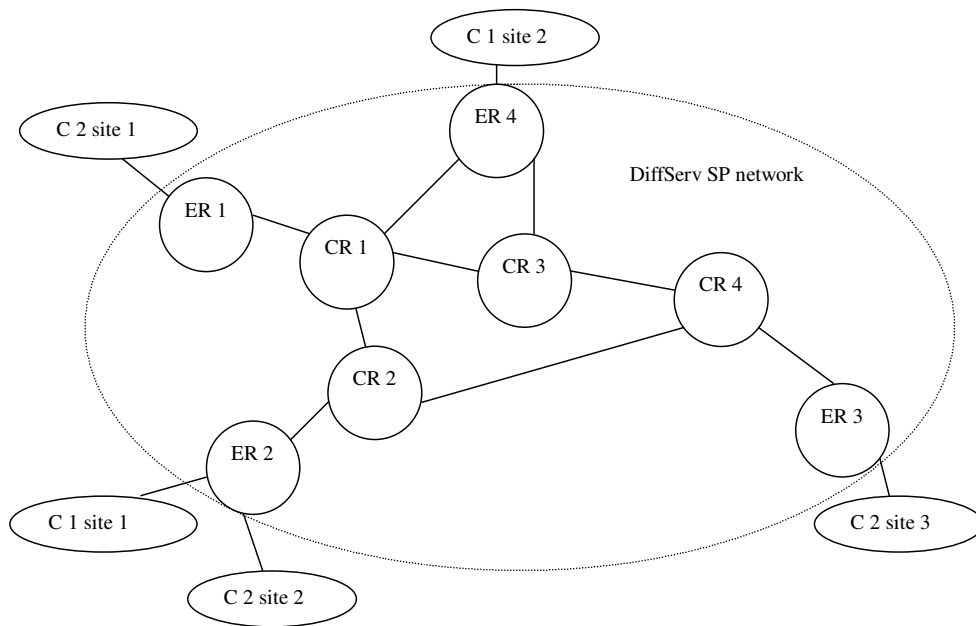


Figure 2.1: A typical DiffServ SP network.

which the customer traffic will be assigned. Later, it configures its edge routers that are ingress routers for that particular traffic so that DS field of packets belonging to that traffic are filled with a value from the set of values specified in the SLA and packets are mapped to the assigned service class. It also places equipment at ingress router for Traffic Conformance Test and Excess Treatment. This equipment can be a group of devices: a Meter, a Marker, and a Shaper (or Dropper). Meters exercise the Traffic Conformance Test per packet; markers do excess treatment of marking packets of customer traffic as in-profile or out-of-profile, and shapers apply delay to packets in order to decrease the number of out-of-profile packets. When a shaper is used, re-marking is done after shaping if an out-of-profile packet is shaped to be in-profile. Droppers just drop out-of-profile packets. Instead of the group of devices, a single device that contains all of those devices can be used which is called Traffic Conditioner [1].

Routers of DiffServ SP network are classified into two classes: Edge routers (ER) that interact with customer sites and core routers (CR) that interact only with routers of the SP. In Figure 2.1, a typical DiffServ SP network is shown.

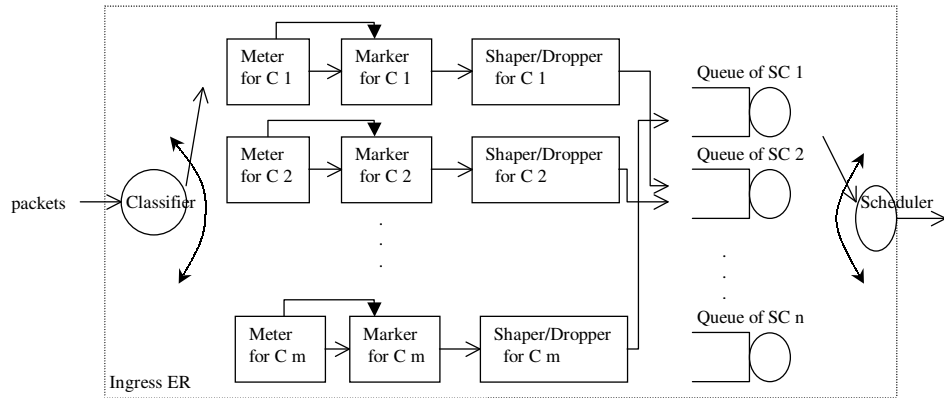


Figure 2.2: Ingress edge router internals.

Mechanisms that work in an ingress edge router for a customer traffic can be seen in Figure 2.2. The Classifier classifies incoming packets with respect to the customer traffic they belong to. In this figure, it is assumed that each of m served traffic belong to different customers (C). Classifier directs packets to the respective Meter. Meter applies Traffic Conformance Test and sends the packet and the result of the test to Marker. Marker marks the packet with respect to the result of traffic conformance test and passes the packet to Shaper/Dropper. If a Shaper is used, Shaper delays the packet (if necessary) to make it in-profile and a re-marking mechanism is applied to modify the marking of the packet. If a Dropper is used, Dropper drops out-of-profile packets. Traffic coming from Shaper/Dropper component of every customer is directed to the queue of the service class (SC) it is assigned to. Scheduler serves queues of all service classes using a scheduling algorithm.

The internals of a core router are shown in Figure 2.3. In a core router, a Classifier, looking at the DS field of IP header of incoming packets, directs them to the queue of service class they are assigned to. And queues of service classes are served by a Scheduler which works with some scheduling algorithm.

Service classes in a SP network are called as Per Hop Behaviors (PHB) in DiffServ terminology [1]. After Internet Engineering Task Force (IETF) DiffServ

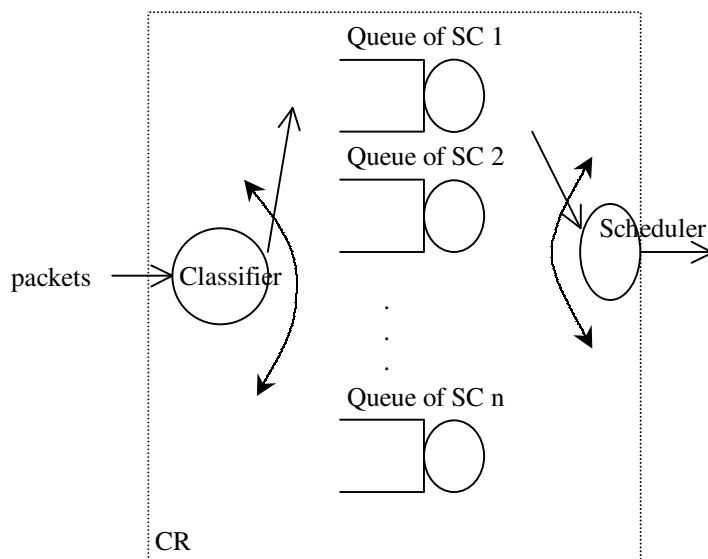


Figure 2.3: Internals of a core router.

WorkGroup specified properties of PHBs and the method of submission of proposals for PHBs in [1], there have been some PHB proposals. Per Hop Behaviors that have been proposed to be included in DiffServ model are discussed in the next section.

2.2 Per Hop Behavior Proposals

DiffServ architecture is aimed to contain only a small number of PHBs. This aim was set to have simplicity and scalability in the core of the network [1]. As described previously, PHBs are service classes among which DiffServ SP distributes its network resources. Allocating different levels of queue space and different amount of bandwidth to PHBs, SP can offer services with varying QoS considerations. Each PHB used in DiffServ networks is standardized for both simplifying the establishment of multi-SP services and helping SPs in deciding on the way of distributing network resources among different PHBs in their networks.

In Sections 2.2.1 and 2.2.2, ideas behind Expedited Forwarding Per Hop Behavior (EF PHB) and Assured Forwarding Per Hop Behavior (AF PHB) group are presented.

2.2.1 Expedited Forwarding Per Hop Behavior

EF PHB is specified in [4]. It supplies a low loss, low delay, low jitter, assured bandwidth service. Loss, delay, and jitter arise from queueing in network core. Queues occur in a network router when the short-term arrival rate is higher than departure rate. EF PHB aims to keep sizes of queues used by the traffic small. The bandwidth used by EF PHB in every node of SP network is configured to be at least as high as a constant value which is specified by SP. This forces queue loads seen by the traffic to be below a certain level [4].

2.2.2 Assured Forwarding Per Hop Behavior

AF PHB is specified in [5]. It is presented as a group of 4 PHBs, each supplying a delay-tolerant, loss-sensitive service. More specifically, each gives a customer the assurance that its packets will be forwarded to their destinations with a high probability as long as the subscribed information rate (in SLA) is not exceeded. Otherwise, packets exceeding subscribed information rate are forwarded with a smaller probability compared to packets that obey subscribed information rate. Packets of a customer traffic using AF PHB can be marked in 2 (green, red) or 3 (green, yellow, red) colors. Green packets are in-profile packets, and red (or yellow and red) packets are out-of-profile packets. Forwarding probability of packet colors decreases in the direction of green, yellow, and red. Each queue used by an AF PHB in DiffServ core have 3 virtual queues in it. Those virtual queues are used to apply probabilistic packet dropping to incoming packets while using different dropping probabilities for packets of different colors, so as to satisfy the

relation between forwarding probabilities of packet colors. This is achieved using an active queue management scheme on each virtual queue of AF PHB, which will be described in the next section [5].

In [6], it is reported that AF PHB may fail to supply subscribed information rate to TCP traffic in under-provisioned networks (networks that have less amount of resources than the amount of traffic SP has agreed on SLAs to serve) which carry either only TCP traffic or both TCP and UDP traffic. Nevertheless, in [3], it is stated that AF PHB can protect TCP traffic from UDP traffic by providing a minimum bandwidth.

Information on active queue management schema that can be used in AF PHB is introduced in the next section.

2.3 Active Queue Management Schema

The idea of Active Queue Management (AQM) was initially introduced for the need in routers of the current (Best-Effort) Internet to avoid the performance degradation during congestion as a supplement to the end-system-based congestion avoidance mechanisms used by TCP [7]. AQM drops packets probabilistically and in a distributed way before the occurrence of congestion in the queue so that TCP traffic decreases its congestion window. This way, TCP saves both further traffic from being lost, which makes it more time-consuming for TCP to recover, and prevents network queues from getting congested. The first AQM introduced was Random Early Detect (RED) [7]. Later, AQMs that try to avoid some of the failing points of RED have been proposed. In Section 2.3.1, RED is briefly described. Among different AQMs that exist in the literature, only RED is extended to support AF PHB. In Section 2.3.2, RED variants for AF PHB, namely RED with In and Out- Coupled virtual Queues (RIO-C), RED with In

and Out- Decoupled virtual Queues (RIO-DC), Weighted RED (WRED), and DROP are introduced.

2.3.1 RED

RED keeps track of average queue load as Exponentially Weighted Moving Average (EWMA) upon packet arrivals to the queue and behaves differently to incoming packets depending on the relation between average queue load and two thresholds, min_{th} (minimum threshold) and max_{th} (maximum threshold). If average queue load is smaller than min_{th} , then the incoming packet is enqueued. If average queue load is greater than max_{th} , the packet is dropped (hard drop). If average queue load is between min_{th} and max_{th} , the packet is dropped probabilistically (early drop). The dropping probability increases both as average queue load gets closer to max_{th} and as the number of packets enqueued since the last drop increases. The dropping probability, denoted as p_a , is given by

$$p_a = \frac{p_b}{1 - count \cdot p_b}, \quad (2.1)$$

where

$$p_b = max_p \cdot \frac{q_a - min_{th}}{max_{th} - min_{th}} \quad (2.2)$$

and max_p is the maximum value of p_b , q_a is the average queue load and $count$ is the number of packets enqueued since the last drop [8].

In [8], it is reported that maximum thresholds of queues in current Internet can easily be exceeded. This is due to the fact that the average queue size increases proportional with $N^{2/3}$ where N is the number of active connections served by the queue, until the maximum threshold is reached. Physical queue sizes used in the current Internet cause the average queue size to reach maximum threshold even for small N [8]. In [8] and [9], it is reported that determination of threshold values for RED is not trivial. Moreover, performance obtained

with RED may not be better than performance obtained with First-In/First-Out (FIFO) queue [9]. Stabilized RED (SRED), Dynamic RED (DRED), and BLUE offer different methods to keep the instantaneous queue load over some specified level, which helps TCP during recovery from congestion and increases performance of the queue after congestion [8]. Stochastic Fair BLUE (SFB) extends BLUE to avoid bad effects of UDP traffic on TCP traffic performance [10].

2.3.2 Variants of RED for AF PHB

Since there are at least 2 virtual queues in an AF PHB, RED has to be extended in order to support AF PHB. Variants of RED that support AF PHB differ in the number of threshold values used, the number of average queue load values computed, and the computation method used. Both RIO-C and RIO-DC keeps one average queue load value and one pair of threshold parameters for each virtual queue. Both update average queue loads of those virtual queues on arrival of each packet. They differ in the method they use in the computation of average queue loads. RIO-DC uses only packets with the color for which the computation is made, whereas RIO-C uses all packets with a color having forwarding probability higher than or equal to the color for which the computation is made [11]. This way, RIO-C makes sure that the dropping probability computed for the incoming packet will increase as the forwarding probability of the packet decreases [12]. WRED computes a single average queue load including packets of all-colors in the computation. However, it has different threshold pairs for each color. DROP AQM drops all incoming packets after queue size reaches a minimum threshold [11].

2.4 Proposed Marking Mechanisms

As a component for Excess Treatment, a marker marks packets as in-profile or out-of-profile based on traffic conformance at the ingress router. In this section different markers proposed in the literature are described. In Section 2.4.1, Token Bucket Marker is explained. In Section 2.4.2, some other markers in the literature are presented.

2.4.1 Token Bucket Marker (TBM)

TBM has two parameters: the maximum rate the customer is allowed to send its packets, called Committed Information Rate (*CIR*), and the maximum number of packets that a customer is allowed to send consecutively (in a burst), called Committed Burst Size (*CBS*) [13]. It enforces its policy through emulation of a *token bucket* which has size *CBS* and token production rate of *CIR*. Tokens produced after token bucket becomes full are discarded. A token can color one Byte of a packet. So, an incoming packet is marked as green when there are tokens enough to color every Byte of it. In that case, also the number of tokens in the token bucket is decreased by the number of tokens used. Otherwise, the packet is marked as red and token level in the token bucket remains unchanged.

2.4.2 Other Markers in the Literature

In [14], the Two-Rate Three Color Marker (trTCM) is described. It is based on TBM. It has 2 token buckets with parameter pairs (*CBS*, *CIR*) and (*Peak Burst Size* (*PBS*), *Peak Information Rate* (*PIR*)). If an incoming packet finds enough tokens in both of token buckets, it is marked as green. In [15], Fair Marker is introduced which is a TBM that aims to distribute tokens fairly among flows within the aggregate of customer. Fair Traffic Conditioner, which is proposed in [16], is

the Fair Marker extended to be based on trTCM. TCP Friendly Marker, which is presented in [17], aims to protect small-window flows from packet losses and maintain spacing between packets that are marked as out-of-profile. In achieving these, it distributes tokens of an aggregate customer traffic among individual flows constituting the aggregate. Then it uses these tokens while leaving some time interval between consecutive packets that are marked as out-of-profile. Proportional Marking, which is proposed in [18], marks out-of-profile packets probabilistically as out-of-profile whose probability increases proportionally with the percentage of excess traffic. A modified form of Proportional Marking which considers individual flows within an aggregate, named as New Marking Algorithm, is also introduced [18].

2.5 Traffic Shapers

In addition to the findings stated in Section 2.2.2, there are studies that show the existence of a relation between assured rate (subscribed information rate in SLA), packet size, Round Trip Time (RTT) of TCP, dropping probability for out-of-profile packets, and capability of TCP to achieve its assured rate [19]. In fact, for some values of assured rate, packet size, RTT, and dropping probability, TCP flow can be unable to achieve its assured rate [19]. Moreover, in [20], for a TCP traffic marked by a TBM in AF PHB, it is reported that the service rate obtained by TCP in an AF PHB cannot be specified using only SLA parameters. On the other hand, there are studies on dynamic pricing of Internet usage based on QoS and rate of service. From customers' viewpoint, this arouses the need to adapt to price changes and to control the rate of their usage [21]. These observations set out the specification of a mechanism to shape the TCP traffic that uses AF PHB so as to achieve (or go beyond in the case of over-provisioned network) either its assured rate or the rate which has the affordable price. The second one might be the topic of future, since there exists no deployed example

of Internet Service pricing based on the content or usage profile. But the first mechanism, to shape the TCP traffic that uses AF PHB so as to achieve or go beyond its assured rate, is the starting point of research on shapers.

In Section 2.5.1, Rate Adaptive Shaper (RAS) and in Section 2.5.2, Green Rate Adaptive Shaper (g-RAS) are explained. In fact, ideas of these two shapers gave rise to the shaper proposed in this thesis. It should be noted that both of these shapers are designed to work upstream from meter, which is different from placement seen in Figure 2.2. In terms of this figure, these shapers are placed before Meter. In Section 2.5.3, some other traffic shapers that have been presented in the literature are described.

2.5.1 Rate Adaptive Shaper (RAS)

Rate Adaptive Shaper (RAS) works upstream from a TCM. In fact, RAS that works upstream from a trTCM is named as trRAS. trRAS consists of a Drop-Tail First-In/First-Out (FIFO) queue served by a varying rate server. The rate of server is determined by two factors: queue load and estimated rate of incoming traffic. trRAS has three threshold values on the queue load: Committed Information Rate Threshold (CIR_{th}), Peak Information Rate Threshold (PIR_{th}), and Maximum Information Rate Threshold (MIR_{th}). Usually, these parameters are assigned values of CBS and PBS parameters of the downstream trTCM, and the size of the queue, respectively. There are three parameters related with the rate of the server: Committed Information Rate (CIR) which is the average transmission rate of customer, Peak Information Rate (PIR) which is the maximum transmission rate of the customer, and Maximum Information Rate (MIR) which is the maximum transmission rate allocated by the SP to the customer. Normally, these parameters are assigned CIR and PIR values of the downstream trTCM, and the rate of the downstream link, respectively. The effect of queue

length (b_o) on the rate of server is determined by a function $f(b_o)$ which is expressed by

$$f(b_o) = \begin{cases} CIR, & \text{if } b_o \leq CIR_{th} \\ \frac{b_o - CIR_{th}}{PIR_{th} - CIR_{th}} * (PIR - CIR) + CIR, & \text{if } CIR_{th} < b_o \leq PIR_{th} \\ \frac{b_o - PIR_{th}}{MIR_{th} - PIR_{th}} * (MIR - PIR) + PIR, & \text{if } PIR_{th} < b_o \leq MIR_{th} \\ MIR, & \text{if } b_o > MIR_{th} \end{cases} \quad (2.3)$$

trRAS can estimate the rate of incoming customer traffic using any form of EWMA calculation. For example, it can use $EAR_{new} = [(1 - e^{-T/K}) \times L/T] + e^{-T/K} \times EAR_{old}$, where EAR_{new} is the new value of estimated arrival rate, EAR_{old} is the previous value of estimated arrival rate, T is the amount of time elapsed since the arrival of the previous packet, L is the size of the incoming packet, and K is a constant. The rate of server for trRAS is calculated as $\max(f(b_o), EAR_{new})$ [22].

2.5.2 Green Rate Adaptive Shaper (green RAS)

As an extension to RAS, green RAS considers the status of the downstream marker in shaping. This way, it avoids unnecessary delaying of the packets that can be marked as green by the downstream marker. Green RAS working upstream from a trTCM (g-trRAS) is described below.

g-trRAS computes a time value T_1 as the time to send the currently received packet using the server rate computed as $\max(f(b_o), EAR_{new})$. Additionally, it computes a time value T_2 which is the earliest possible time instant when the current packet can be marked as green by the downstream marker. The calculation method is $\max(t, t + (B - Tc(t))/CIR, t + (B - Tp(t))/PIR)$, where t is the current time, B is the size of the packet (in Bytes), $Tc(t)$ is the number of tokens at time t in the token bucket of trTCM with parameters (CBS, CIR),

and $T_p(t)$ is the number of tokens at time t in the token bucket of trTCM with parameters (PBS, PIR). PIR and CIR are respective parameters of trTCM. If B is greater than any of PBS or CBS, T_2 is set to infinity. The time to send the current packet is determined as $\min(T_1, T_2)$ [22].

Unnecessary delaying of packets can degrade performance of TCP [23]. This lowering effect of delay on the throughput of TCP traffic (the amount of data received as consecutive packets in unit time), which results from increased RTT, can be observed from the approximation of steady-state TCP Reno throughput, $B(p)$, which is stated as [23]

$$B(p) \approx \min \left(\frac{W_{max}}{RTT}, \frac{1}{RTT \sqrt{\frac{2bp}{3}} + T_O \min \left(1, 3\sqrt{\frac{3bp}{8}} \right) p(1 + 32p^2)} \right) \quad (2.4)$$

where W_{max} is the maximum window size of TCP, b is the number of packets that are acknowledged by a received acknowledgement. The value of b is usually 2, p is the probability that a packet is dropped, given that it is not a member (other than first) of consecutive drops, and T_O is the initial timeout duration.

2.5.3 Other Shaper Algorithms in the Literature

In [24], a shaper is introduced to work in ATM networks. It uses the distribution of a random process as a reference to shape the distribution of the incoming traffic. In [25], a traffic shaper is introduced for Best-Effort Internet to avoid congestion occurring in video stream clients. It can determine minimum and maximum rates that customer can use, and the size of each burst. A traffic shaper is used for Best-Effort Internet to shape the customer traffic into the currently available rate in the network in [26]. In [27], a traffic shaper mechanism is introduced to work in Best-Effort Internet and make TCP flows uncorrelated to favor delay-sensitive flows. A traffic shaper for Guaranteed Service of IntServ model is proposed in [28]. This shaper has two rate values to work with. During

a small interval since the beginning of a burst of traffic, it uses the high rate and later it uses the low rate to send MPEG-compressed video.

2.6 Traffic Conditioner Proposals

As stated previously, Traffic Conditioners include a group of devices: a Meter, a Marker, and a Shaper/Dropper. So, a traffic conditioner both applies Traffic Conformance Test and handles Excess Treatment.

In [29], three traffic conditioners are introduced. RTT-Aware Traffic Conditioning aims to distribute excess bandwidth in over-provisioned networks favoring TCP connections with long RTT. Similarly, Target-Aware Traffic Conditioner with 2 Drop Precedences (TATC-2DP) and TATC-3DP aim to distribute excess bandwidth in over-provisioned networks favoring TCP connections with larger subscribed rate.

In this thesis, a shaper is proposed, which determines the sending time of an arriving packet considering the status of the downstream marker and the total shaping delay. The shaping delay is defined as the time interval between the time when all Bytes of a packet are received and the time when all Bytes of the the packet are sent downstream to the marker. The proposed shaper considers an upper bound on the shaping delay it can apply to an arriving packet. By limiting the maximum shaping delay, the proposed shaper tries to prevent unnecessarily large shaping delays resulting in large RTTs which in turn decrease the throughput as predicted by (2.4). Obeying the upper bound on the shaping delay, it tries to increase the number of packets that are marked as green by the downstream marker. By adjusting shaping delay in a controlled manner and decreasing dropping probability that segments face in the DiffServ core, it aims to increase the throughput achieved by TCP. In the next chapter, architecture and the algorithm of the proposed shaper and results of simulations performed

using NS [30] are presented. In Chapter 4, an extension to the shaper is introduced where the shaper periodically modifies the upper-bound on the shaping delay to increase the throughput achieved based on information from the sender TCP. Algorithms used are described in detail. Simulation results that reflect the improvement of this extension are presented.

Chapter 3

Delay-Bounded, Rate-Adaptive Shaper (DBRAS)

In this chapter, we are going to present a description of the shaper proposed in this thesis, DBRAS, in a sequence of steps: In Section 3.1, required capabilities of DBRAS are specified. In Section 3.2, design of DBRAS is presented. In Section 3.3, shaping algorithm of DBRAS is presented. In Section 3.4, simulations for performance evaluation are introduced.

3.1 Required Capabilities

The throughput level achieved by TCP traffic when it is shaped by any shaper should not be less than the level of throughput achieved when the TCP traffic is not shaped. DBRAS aims to delay a packet it received, as long as the shaping delay of the packet is at most equal to an upper bound, D_{max} . Otherwise, it just aims to put the packet out as soon as possible. The upper bound on shaping delay is placed for the following observation. Packets of the traffic can be made all in-profile using an infinite-length queue as long as the mean packet arrival

rate is less than the mean drain rate of the policer, CIR. But depending on the relation between CIR and actual traffic sending rate, this may lead to long queuing delays in the shaper which, in turn, may lead to decrease in the achieved level of throughput. Furthermore, the physical constraints on the queue size in the shaper makes this idea unapplicable. Moreover, it is possible that SP network is lightly loaded at the time of TCP traffic and packets of the traffic can pass through the network irrespective of their color. In that case, shaping the traffic to make a larger ratio of packets marked as green can lead to lower level of achieved throughput than the level that could be achieved when the traffic is not shaped.

3.2 DBRAS Design

The design of DBRAS is shown in Figure 3.1. Packets arrive to DBRAS from the source via a link with a transmission rate of R_{UL} . DBRAS consists of three components: **Scheduler**, which determines the amount of shaping delay to apply to an incoming packet. Then Scheduler sends the packet to the **Buffer**, which is the place where the packet waits until it is ready for transmission. It is actually a queue whose activity- putting the packet at the head of the queue onto the outgoing link- is controlled by the **Transmitter**. Scheduler maintains a list of shaping delay values assigned to each packet. As Scheduler sends the packet to the Buffer, it simultaneously appends a delay value for that packet to the end of the delay list. After a packet becomes the head-of-the-queue packet (first one to go out) in the Buffer, Transmitter releases the packet after a delay equal to the waiting time assigned for this packet which is given in the delay list.

It should be noted that we need to supply the shaper with a downstream link to Marker having a rate at least equal to the transmission rate of the upstream link of the shaper, R_{UL} . Otherwise, inevitably, whatever it does for shaping, there will be accumulation of packets in the Buffer since the rate with which

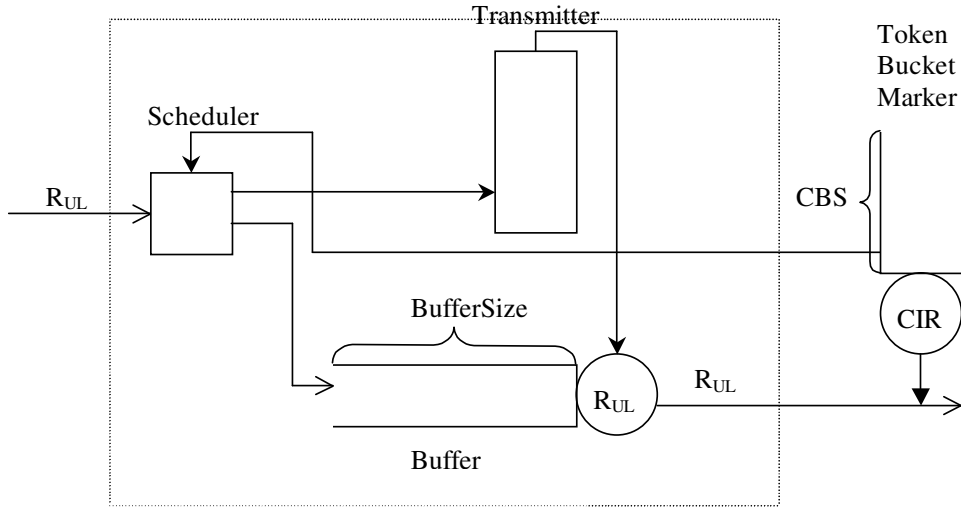


Figure 3.1: Design of the DBRAS.

DBRAS receives data will be larger than the rate with which it can send packets to the downstream Marker. For simplicity, the downstream marker is assumed to be of type TBM.

Scheduler determines the delay for a packet of size L_i Bytes received at time a_i , using the following algorithm which is a modification of the algorithm proposed for the green RAS [22]. Green RAS calculates the delay using the state of the marker at the time when the packet comes to the head of the queue whereas DBRAS shaping algorithm determines the delay using the state of the marker at the time when the packet will reach the marker while also considering D_{max} as an upper-bound on the shaping delay. We assume that the propagation delay of the link between shaper and marker is negligible and only the transmission delay for that link is considered. DBRAS shaping algorithm is described in detail in the following section.

3.3 DBRAS Shaping Algorithm

The shaping algorithm for DBRAS is described below. A packet arriving at DBRAS can find it in either of 2 states: there is no packet in the queue, or there

is at least one packet in the queue.

When a packet is received,

If there is no packet in the queue

 If there are enough tokens to mark the packet as green

 send it immediately after you receive it

 Else

 calculate a delay value in which tokens enough for the packet to be marked
 as green can be produced

 If shaping delay is greater than D_{max}

 send the packet immediately after you receive it (let it be marked
 as red)

 Else

 send the packet so as to satisfy its shaping delay (it will be marked
 as green)

Else

 After sending the last packet currently in the queue, if there will be enough
 tokens to mark the packet as green,

 send the packet immediately after sending the last packet currently in the
 queue

 Else

 calculate a delay value in which tokens enough for the packet to be marked
 as green can be produced

 If shaping delay is greater than D_{max} ,

 send the packet immediately after sending the last packet currently in
 the queue

 Else

 send the packet so as to satisfy its shaping delay (it will be marked as
 green)

The parameters used by DBRAS algorithm are described below.

a_n : arrival time of the last bit of packet n , in sec,
 L_n : size of the packet n , in Bytes,
 R_{UL} : transmission rate of the downstream link (link between shaper and TBM),
in Bytes/sec,
CIR: Committed Information Rate of TBM, in Bytes/sec,
CBS: Committed Burst Size of TBM, in Bytes,
 D_{max} : the maximum value of shaping delay, in sec,
 S_n : sending time of the first bit of packet n (start of transmission), in sec,
 S'_n : sending time of the last bit of packet n (end of transmission), in sec,
 $S'_{lastuse}$: the last time TBM marked a packet as green, i.e., the last time a token
is used, in sec,
 $t_{a_l_u}$: the number of tokens remaining in the token bucket after the latest
packet that is marked as green is transmitted,
 t_p : the number of tokens produced in the time interval from the last token usage
up to the earliest possible arrival of the currently considered packet to TBM.

The shaping algorithm of DBRAS can be stated in a more programming-based approach in the following way:

```

if ( $a_n > S'_{n-1}$ ) /* There is no packet in the queue*/
     $t_p \leftarrow \min(CBS - t_{a\_l\_u}, (a_n - S'_{lastuse} + (L_n/R_{UL})) * CIR)$ 
    if ( $L_n \leq t_{a\_l\_u} + t_p$ ) /*enough tokens to mark as green*/
         $S_n \leftarrow a_n$ 
         $S'_n \leftarrow S_n + (L_n/R_{UL})$ 
         $S'_{lastuse} \leftarrow S'_n$ 
         $t_{a\_l\_u} \leftarrow t_{a\_l\_u} + t_p - L_n$ 
    else
         $d \leftarrow (L_n - (t_{a\_l\_u} + t_p))/CIR$  /*when it can be marked as green*/
        if ( $(d + (L_n/R_{UL})) > D_{max}$ )
             $S_n \leftarrow a_n$ 

```

```

     $S'_n \leftarrow S_n + (L_n/R_{UL})$ 
else
     $S_n \leftarrow a_n + d$ 
     $S'_n \leftarrow S_n + (L_n/R_{UL})$ 
     $S'_{lastuse} \leftarrow S'_n$ 
     $t_{a\_l\_u} \leftarrow (t_{a\_l\_u} + t_p + d * CIR) - L_n$ 
else /* There is at least one packet in the queue*/
     $t_p \leftarrow \min(CBS - t_{a\_l\_u}, (S'_{n-1} - S'_{lastuse} + (L_n/R_{UL})) * CIR)$ 
    if ( $L_n \leq t_{a\_l\_u} + t_p$ ) /*there will be enough tokens to mark as green*/
         $S_n \leftarrow S'_{n-1}$ 
         $S'_n \leftarrow S_n + (L_n/R_{UL})$ 
         $S'_{lastuse} \leftarrow S'_n$ 
         $t_{a\_l\_u} \leftarrow t_{a\_l\_u} + t_p - L_n$ 
    else
         $d \leftarrow (L_n - (t_{a\_l\_u} + t_p))/CIR$ 
        if ( $(S'_{n-1} - a_n + (L_n/R_{UL}) + d) > D_{max}$ ) /*let it be marked as red*/
             $S_n \leftarrow S'_{n-1}$ 
             $S'_n \leftarrow S_n + (L_n/R_{UL})$ 
        else /* introduce delay d to mark as green*/
             $S_n \leftarrow S'_{n-1} + d$ 
             $S'_n \leftarrow S_n + (L_n/R_{UL})$ 
             $S'_{lastuse} \leftarrow S'_n$ 
             $t_{a\_l\_u} \leftarrow (t_{a\_l\_u} + t_p + d * CIR) - L_n$ 
 $S'_{n-1} \leftarrow S'_n$ 

```

In order to evaluate the performance of DBRAS, we implemented DBRAS in NS [30] (version 2.1b7a, using DiffServ Module that is contributed by Nortel Networks and included in NS starting from version 2.1b8), and performed simulations. In Section 3.4, performance analysis of DBRAS is presented.

3.4 Performance Analysis of DBRAS

In the beginning of the chapter, required capabilities for DBRAS were stated. To test whether DBRAS is successful in answering those requirements, a group of simulations are performed. The topology to be used in the simulations should be chosen so as to observe the performance of DBRAS objectively. The topology used in simulations is discussed in Section 3.4.1. In Section 3.4.2, results of simulations are presented and reasoned.

3.4.1 Simulation Topology

The performance of DBRAS is evaluated for a bursty TCP traffic in a congested network, where advantage of using the shaper will be more significant. In simulations, we used 10 TCP connections, 5 of which are shaped by DBRAS, whereas the other 5 are unshaped. We compare the average throughput values achieved by shaped and unshaped traffic as a measure of performance of DBRAS. Simulations are obtained for different values of D_{max} , CBS, and total propagation delay of the traffic from its source to its destination (called as P_D in the remaining of thesis) to observe the effects of these parameters on the performance of DBRAS. Before analyzing the results obtained from simulations, we are going to study the simulation network in 3 parts: Network design, DiffServ configuration, and traffic details.

Network Design

We built the topology shown in Figure 3.2. In this network, nodes (e1, core, e2) represent the network of DiffServ SP. Nodes e1 and e2 are DiffServ edge routers and core node is a DiffServ core router, as their names imply, respectively. Nodes labelled as s# denote nodes of customers which send data into SP network. Node

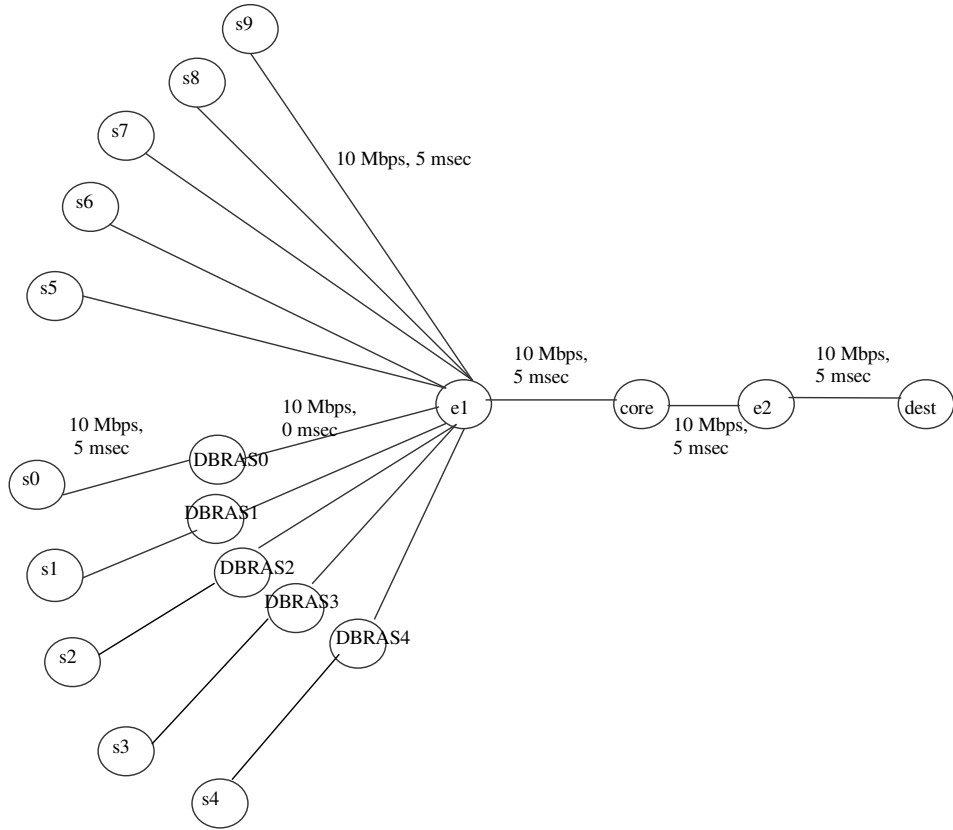


Figure 3.2: Simulation Topology.

dest is the destination of all the traffic through DiffServ core. Nodes labelled as DBRAS# are the nodes where DBRAS is working. Each link in this topology has 10 Mbps rate. Except the links between nodes DBRAS# and e1, all links have 5 msec propagation delays. The links between nodes DBRAS# and e1 are assigned to have 0 msec propagation delay to satisfy the requirement of DBRAS shaping algorithm stated previously. There are queues of Drop Tail type at the entrance of links $(s_i, e1)$ and $(e1, s_i)$ for all $i, 5 \leq i \leq 9$, links (s_i, DBRAS_i) , (DBRAS_i, s_i) , and $(e1, \text{DBRAS}_i)$ for all $i, 0 \leq i \leq 4$, and links $(e2, \text{dest})$ and $(\text{dest}, e2)$. There are queues of DBRASQueue type (our implementation of DBRAS in NS [30]) at the beginning of each of the links $(\text{DBRAS}_i, e1)$ for all $i, 0 \leq i \leq 4$. There are queues of dsred/edge (NS implementation of DiffServ edge router) type at the beginning of simplex links $(e1, \text{core})$ and $(e2, \text{core})$. There are queues of dsred/core (NS implementation of DiffServ core router) type at the beginning of simplex links $(\text{core}, e1)$ and $(\text{core}, e2)$.

DiffServ Configuration

The DiffServ SP network bandwidth is divided evenly between those 10 TCP connections. In fact, each is configured to have CIR of 1 Mbps and the same CBS value which will be specified in particular simulations. Each of 10 traffic streams uses the same AF PHB and a TBM of its own. In a particular simulation, all TBMs in the network have identical CIR and CBS values. Similarly, in a particular simulation, all DBRASSs in the network have the same D_{max} value. In the DiffServ core, RIO-C AQM scheme is used. The parameters $(min_{th}, max_{th}, max_p)$ of the green and red virtual queues are $(20, 40, 0.02)$ and $(10, 20, 0.1)$, respectively. These values are as specified in example simulation files of DiffServ in NS.

Traffic Details

In order to generate bursty traffic, Exponential type traffic generators in NS are used. These traffic generators have exponentially distributed burst occurrence times and burst durations. The rate of traffic production during a burst is set to 10 Mbps. Average burst duration is set to be 20 msec and idle time (time between consecutive bursts) average is set to be 113.3 msec, so that each traffic has a mean rate of 1.5 Mbps. Packet size is 1000 Bytes. In our simulations TCP has a MSS of 1000 Bytes and a congestion window limit of 20 segments.

Traffic is sent using TCP Reno implementation. This implementation uses 3 duplicate acknowledgements (ACKs) as a signal for packet loss (fast retransmission). Fast recovery is executed after fast retransmission by dropping slow-start threshold by half, assigning that value of slow-start threshold as congestion window, increasing congestion window by one for each duplicate ACK received, and assigning slow-start threshold value to congestion window by the reception of a new ACK (ACK acknowledging previously unacknowledged data) [30].

We measure the throughput of a TCP traffic by using EWMA. The formula used in calculation of TCP throughput in k^{th} measurement period, $T(k)$, is

$$T(k) = (1 - \alpha) \times T(k - 1) + \alpha \times ((l_s - f_s)/p_d) \quad (3.1)$$

where α is the EWMA constant with value 0.2, l_s is the sequence number of the last segment that was ACKed by the receiver in the current measurement period (in NS, ACKs are in terms of segments, not Bytes). Similarly, f_s is the sequence number of the first segment that was ACKed by the receiver in the current measurement period, and p_d denotes the duration of measurement period in seconds, which is chosen to be 200 sec. The traffic generators work for 5000.0 sec and measurements are made until time 4400.0 sec.

Having studied the simulation network in detail, we are going to observe and analyze the results obtained from simulations. Simulations of this chapter can be classified into 2 sets in terms of P_D . In the first set of simulations, P_D is 20 msec. In the second set of simulations, the P_D is 5 msec. Within each set, measurements are taken for different values of the pair (CBS, D_{max}).

3.4.2 Simulation Results

In the first set of simulations, propagation delay for each non-zero-propagation-delay link is chosen to be 5 msec so that P_D is 20 msec. For this set, 3 CBS values are used: 3, 5, and 8 KBytes. For each CBS value, D_{max} value range is defined so that the throughput values reach a constant, which results from the fact that with this particular D_{max} value, all packets of the traffic can be shaped to be marked as green.

Results of the first set of simulations are shown in Figure 3.3. In this figure, there are 3 pairs of plots for average throughput values of shaped and unshaped sources, one for each CBS value. When we focus on the upper-most and lower-most plots, average throughput for shaped and unshaped traffic with CBS = 3

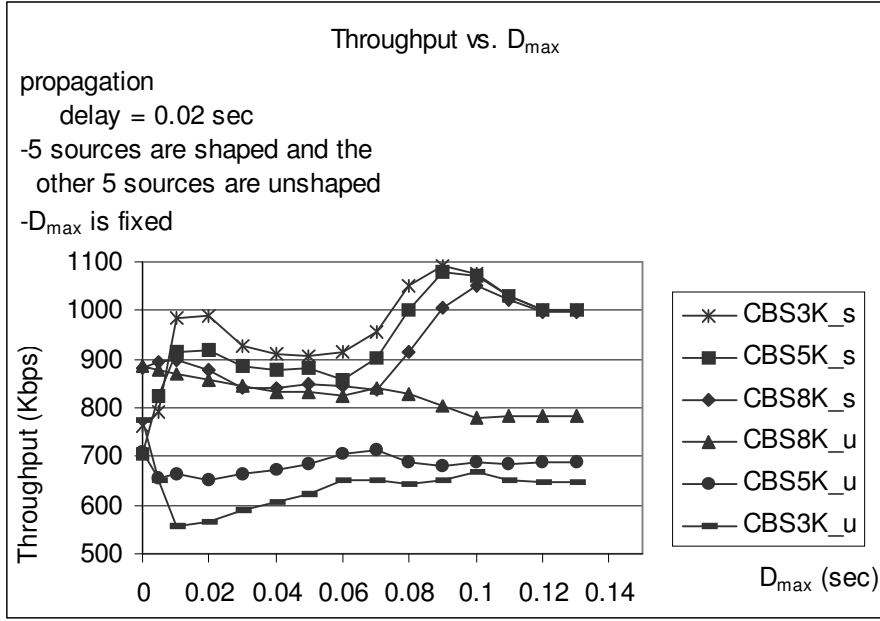


Figure 3.3: Simulations with $P_D = 20$ msec.

KBytes respectively, we observe that as D_{max} increases, the average throughput of shaped traffic goes through consecutive periods of increase, decrease, increase, decrease and finally a constant value. This behavior is consistent with the fact that packet drop probability of TCP traffic is inversely proportional with D_{max} (as D_{max} increases, the number of green packets increases and green packets have lower dropping probability compared to red packets in DiffServ core), and RTT is directly proportional to D_{max} (as D_{max} increases, shaping delay that can be assigned to a packet increases which can lead to an increase in RTT); both of which are inversely proportional to the throughput of TCP as given by equation (2.4). The behavior of average throughput for unshaped traffic is mainly due to changes in the average throughput for shaped traffic, since the amount of total network resources used by all traffic is fixed. If we consider the effect of CBS value on DBRAS, as CBS increases, the average throughput achieved by shaped traffic and the gain of using DBRAS (difference between average throughput for shaped and unshaped traffic) decreases. This is mainly due to the fact that as CBS increases, TBMs for unshaped traffic generate larger number of green

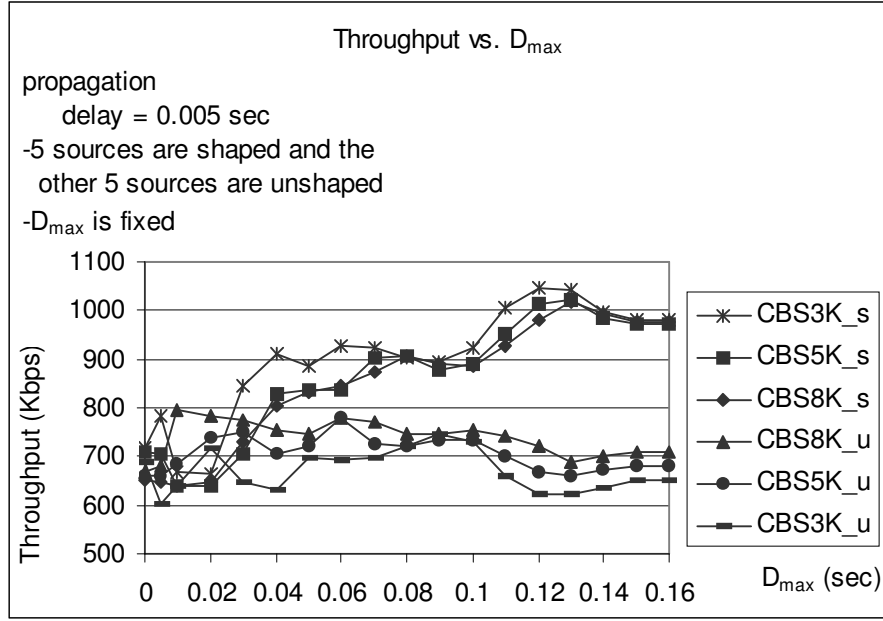


Figure 3.4: Simulations with $P_D = 5$ msec.

packets, i.e., unshaped traffic benefits more from increasing CBS compared to shaped traffic.

In the second set of simulations, propagation delay for each non-zero-propagation-delay link is chosen to be 1.25 msec so that P_D is 5 msec. The same set of CBS values as the previous simulations is used. For each CBS value used, D_{max} is increased until all average throughputs become constant.

The results obtained for the second set of simulations are shown in Figure 3.4. Behaviors of average throughputs of shaped and unshaped traffic as D_{max} increases are roughly the same as the ones observed in Figure 3.3. However, this set of simulations resulted in slightly lower maximum average throughput (maximum taken over D_{max}) for shaped traffic and slightly higher maximum average throughput for unshaped traffic compared with the first set of simulations. In fact, behavior of average throughput of shaped traffic seems contradictory to equation (2.4) describing TCP throughput, where as RTT decreases, we should have obtained higher TCP throughput. The explanation for this situation is as follows: As P_D decreases, sender TCPs can increase their congestion windows

more rapidly. This results in higher queue load in DiffServ core (in particular, queue at the beginning of link (e1, core)). This triggers drops due to exceeding max_{th} (named as hard drops) which will force consecutive packet drops for both shaped and unshaped traffic resulting in smaller congestion window sizes for both shaped and unshaped traffic. This, in turn, increases the number of green packets in both shaped and unshaped traffic. The reason for having a smaller congestion window when hard drops occur is that consecutive packet losses in the same window cause TCP Reno sender to enter the slow start phase instead of doing fast recovery. So, in the steady-state, congestion window sizes in this set of simulations are smaller than congestion window sizes in $P_D = 20$ msec case.

In the simulation results that have been presented so far, we observe that D_{max} value affects the average throughput achieved using DBRAS and the gain of using DBRAS. In fact, in different scenarios (P_D values), the maximum average throughput is achieved at different values of D_{max} , namely around 90 msec in $P_D = 20$ msec case and around 120 msec in $P_D = 5$ msec case. Moreover, dynamic changes in topology and network load may affect the performance of DBRAS with a particular D_{max} value. These imply the need to have a DBRAS where D_{max} changes dynamically, based on measurements obtained from the network. In the next chapter, we are going to present an algorithm for dynamically adjusting D_{max} value used by DBRAS during its operation in order to increase the throughput achieved. DBRAS presented in this chapter will be called DBRAS and the extended form of DBRAS will be called Dynamic DBRAS (D-DBRAS) in the remaining of this thesis.

Chapter 4

Dynamic DBRAS (D-DBRAS)

In this chapter, the extended form of DBRAS, D-DBRAS is presented. Ideas presented in this chapter are built on assumptions that DBRAS is used to shape a traffic which comprises of a single TCP connection and that sender TCP statistics are available to DBRAS. In Section 4.1, design considerations in terms of required capabilities of D-DBRAS are stated. In Section 4.2, the D_{max} Adaptation Algorithm is introduced. In Section 4.3, the topology used in simulations for D-DBRAS is studied. In Section 4.4, simulation results are presented.

4.1 Required Capabilities

Examining Figures 3.3 and 3.4, it can be observed that curves of average throughput values of shaped sources vs. D_{max} have one or more intervals of D_{max} values where the highest average throughput values among the points constituting the curves are achieved. These intervals do not include the last two points where all segments of TCP are marked as green by respective TBMs.

In the light of these observations, D-DBRAS should achieve convergence of D_{max} to one of those local optimum values. Moreover, the sender TCP should

achieve a throughput greater than the throughput value achieved when TCP traffic is unshaped. Furthermore, it should avoid an arbitrarily large D_{max} value for which all TCP segments are marked as green by the TBM (all-green case). By dynamically adjusting D_{max} , we would like to achieve throughputs that are higher than the throughputs obtained by using DBRAS where D_{max} is assigned the starting value of D_{max} for D-DBRAS.

4.2 D_{max} Adaptation Algorithm

The D-DBRAS performs periodic average throughput measurements obtained within a time interval and changes its D_{max} by a fixed amount in the direction which increases throughput of sender TCP. In case throughput measurement of sender TCP remains the same from the last update, it changes its D_{max} in the counter-direction of previous D_{max} change. In this context, the heuristic adaptation algorithm is greedy. Since Figures 3.3 and 3.4 exhibit multiple local optimums for the throughput, this algorithm does not guarantee reaching the globally optimum solution. Also reaching all-green case, it decreases its D_{max} until it leaves the all-green case so that unnecessarily large values of D_{max} are avoided. The average throughput is calculated at the shaper by using the acknowledgement number field in TCP header for segments received at the TCP source from the destination. Before presenting D_{max} adaptation algorithm, we introduce the following quantities.

$D_{max}(k)$: D_{max} value used by D-DBRAS during the k^{th} adaptation period, in sec.

δ : The amount of change applied to D_{max} at the end of an adaptation period, in sec.

ap: The length of an adaptation period, in sec.

ha(k): Highest sequence number that is acknowledged by the end of k^{th} adapta-

tion period, in Bytes.

$gr(k)$: Ratio of TCP segments that are marked as green by the downstream TBM to the total number of TCP segments in the k^{th} adaptation period.

$T(k)$: Throughput measured at the end of k^{th} adaptation period, in bits/sec, for $k > 1$, which is computed as $T(k) = ((ha(k) - ha(k - 1)) * 8.0)/ap$.

The adaptation algorithm can be stated as:

At the end of k^{th} adaptation period,

obtain $ha(k)$

calculate $T(k)$

if ($k = 1$)

$D_{max}(k + 1) \leftarrow D_{max}(k) + \delta$ /*start with an increase*/

else

if ($gr(k) = 1.0$) /*avoid all-green case*/

$D_{max}(k + 1) \leftarrow D_{max}(k) - 2 * \delta$

else

if ($D_{max}(k) > D_{max}(k - 1)$)

if ($T(k) > T(k - 1)$)

$D_{max}(k + 1) \leftarrow D_{max}(k) + \delta$

else

$D_{max}(k + 1) \leftarrow D_{max}(k) - \delta$

if ($D_{max}(k + 1) < 0.0$)

$D_{max}(k + 1) \leftarrow 0.0$

else

if ($T(k) > T(k - 1)$)

$D_{max}(k + 1) \leftarrow D_{max}(k) - \delta$

if ($D_{max}(k + 1) < 0.0$)

$D_{max}(k + 1) \leftarrow 0.0$

else

$D_{max}(k + 1) \leftarrow D_{max}(k) + \delta$

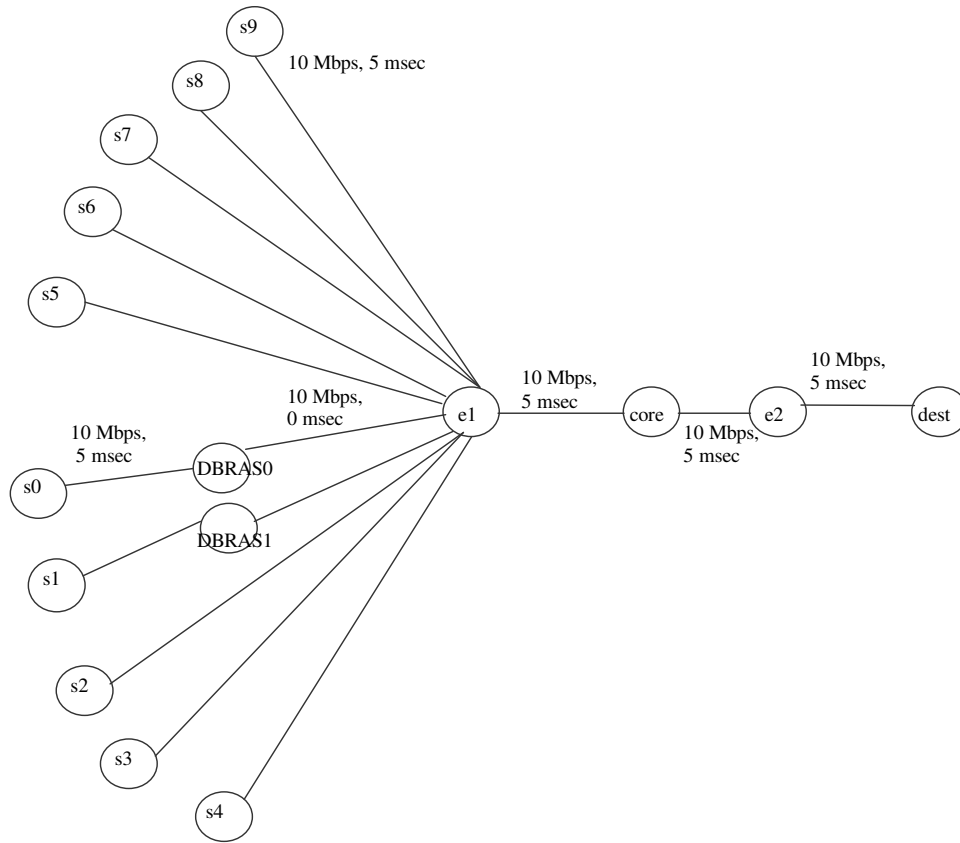


Figure 4.1: Simulation Topology for D-DBRAS.

In Section 4.3, the topology used in simulations is studied. In Section 4.4, simulations performed using D-DBRAS are presented.

4.3 Simulation Topology

Topology used in simulations with D-DBRAS is shown in Figure 4.1. Points where this topology differs from the topology used in Chapter 3 in terms of network design and DiffServ configuration are described below.

Differences in Network Design

Nodes s2, s3, and s4 are connected to e1. Each of the simplex links (si, e1) and (e1, si) for all $i, 2 \leq i \leq 4$ has 10 Mbps bandwidth and 5 msec propagation delay. There is a queue of Drop Tail type at the beginning of each of these links.

Differences in DiffServ Configuration

CBS value of each TBM is set to be 3 KBytes. The adaptation duration parameter of every D-DBRAS instance is set to $ap = 200$ sec, which was used as EWMA computation period in simulations of Chapter 3.

In the next section, details of simulations of D-DBRAS are presented.

4.4 Simulation Results

For evaluating the performance of D-DBRAS, a group of simulations are performed. First, in order to satisfy the requirement of fast convergence, it can be observed from the D_{max} Adaptation Algorithm that a proper value of δ should be used. In Section 4.4.1, simulations performed for finding a proper value for δ are studied. After choosing proper value for δ , simulations are performed to test the performance of D-DBRAS. Effects of P_D , level of congestion, ratio of shaped traffic to total traffic, and RED parameters on performance of D-DBRAS are investigated. Results of these simulations are presented in Section 4.4.2. In Section 4.4.3, results of simulations performed in networks with dynamic network load are presented. In this section, initially, simulations to find a better ap value in terms of D_{max} convergence duration (time it takes for D_{max} of D-DBRAS to converge) are presented. Then, with proper values of δ and ap , some simulations are performed to test the performance of D-DBRAS with traffic dynamic load. Scenarios used and results of simulations are presented.

All averages presented in this chapter are computed using EWMA with $\alpha = 0.2$ unless otherwise stated.

4.4.1 Selection of a Proper Value of δ

In this simulation, a set of δ values $\{0.8, 2, 4, 6\}$ (in msec) are used. It should be noted that transmission time of a TCP segment from DBRAS is 0.8 msec. The simulation is performed with both P_D values, 20 msec and 5 msec. The instances of D-DBRAS in the network are activated with initial values of D_{max} chosen from the range of values used in Chapter 3.

For $P_D = 20$ msec case, it can be observed from Figure 3.3 that there are two locally optimum D_{max} values: one is around 20 msec and the other is around 90 msec. D_{max} values of both instances of D-DBRAS in the simulation are desired to converge to either of these two values by the end of each simulation. Behavior of D_{max} of D-DBRAS instance that shapes “traffic from node s0” (denoted from now on as s0) with δ values of 0.8, 2, 4 and 6 msec are presented in Figures 4.2, 4.3, 4.4 and 4.5 respectively. Similarly, results for D-DBRAS of s1 are presented in Figures 4.6 through 4.9. It can be observed that proper convergence of D_{max} occurs for $\delta = 6$ msec in $P_D = 20$ msec case.

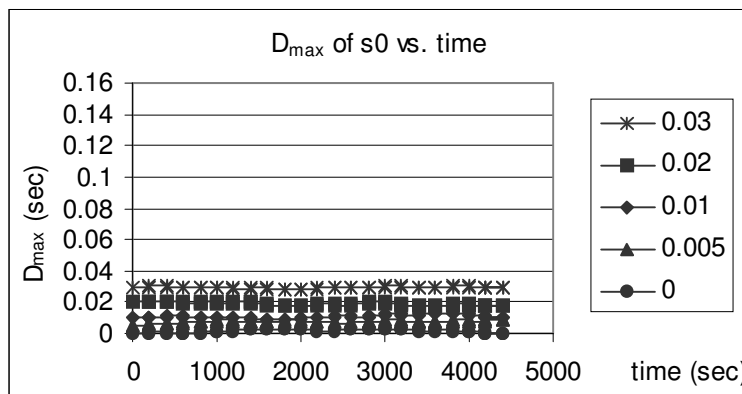
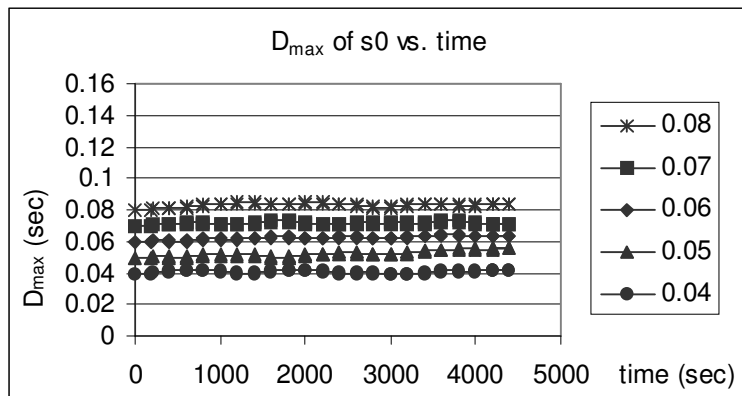
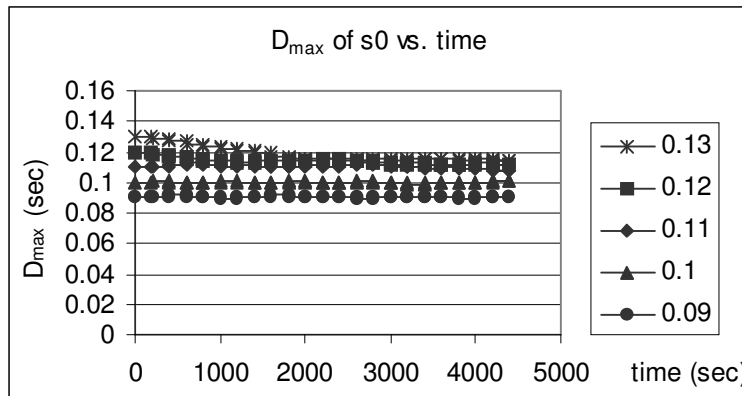
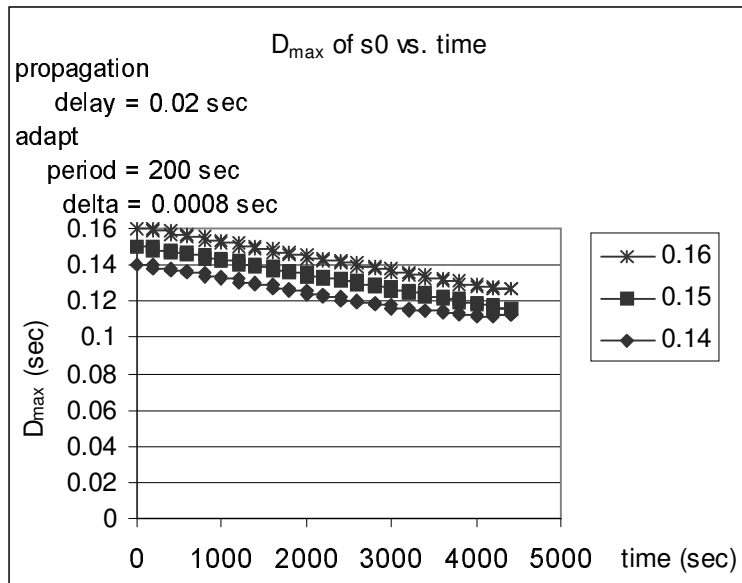


Figure 4.2: D_{max} of s0 vs. time with different initial D_{max} values, $\delta = 0.8$ msec.

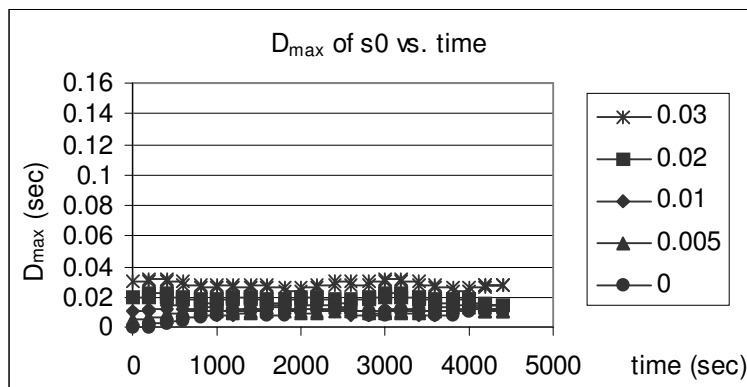
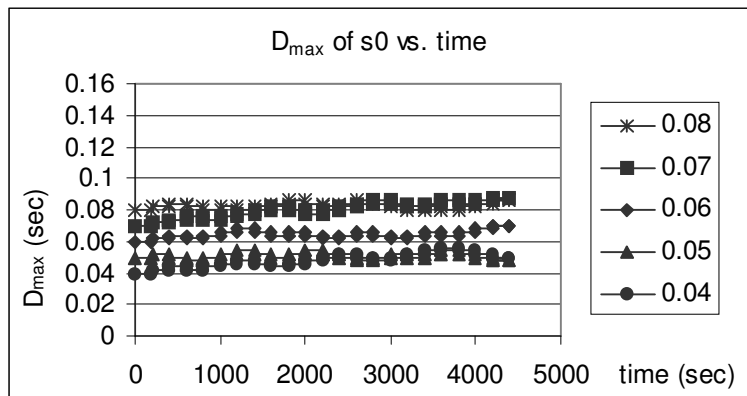
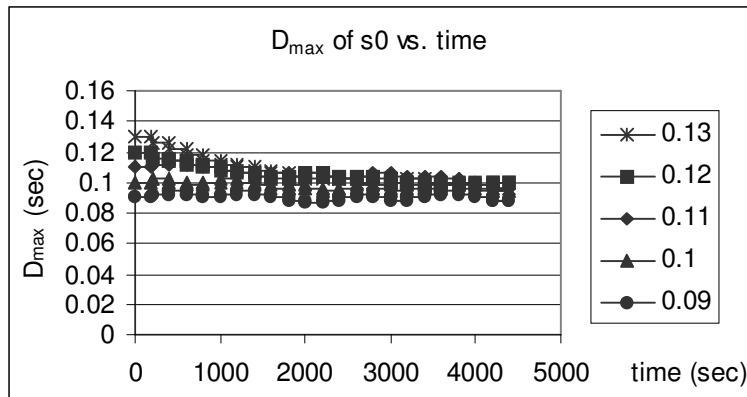
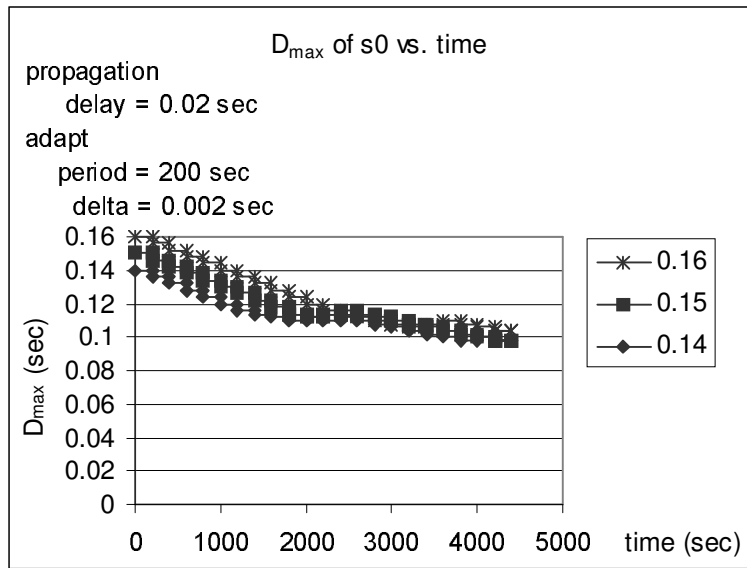


Figure 4.3: D_{max} of s0 vs. time with different initial D_{max} values, $\delta = 2$ msec.

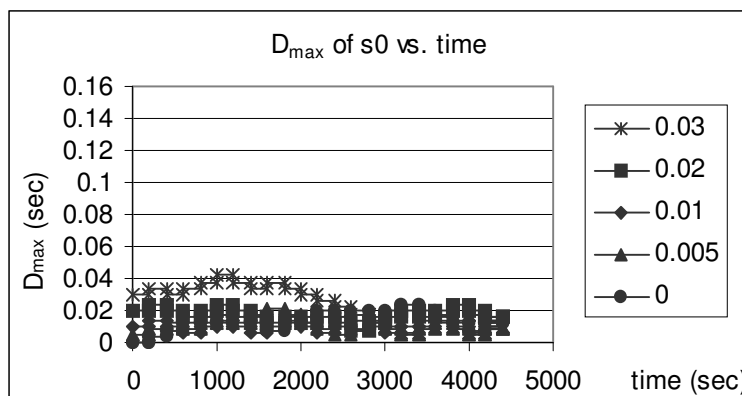
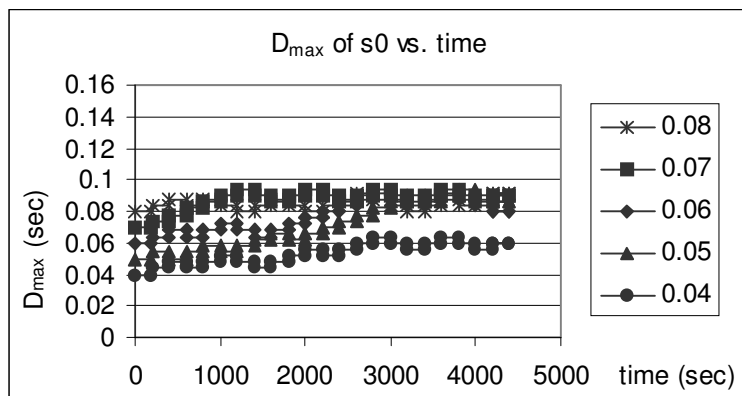
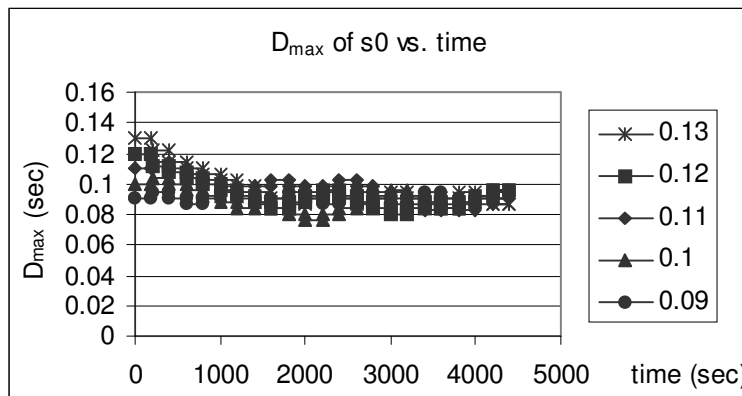
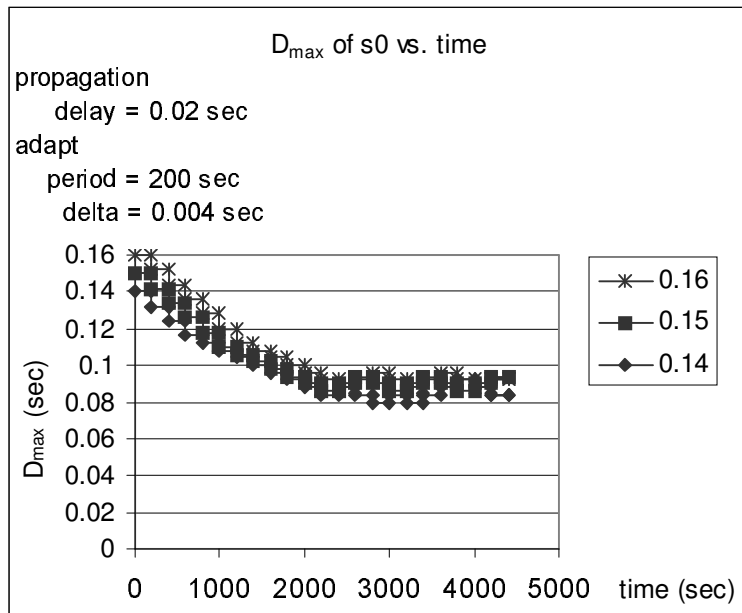


Figure 4.4: D_{max} of s0 vs. time with different initial D_{max} values, $\delta = 4$ msec.

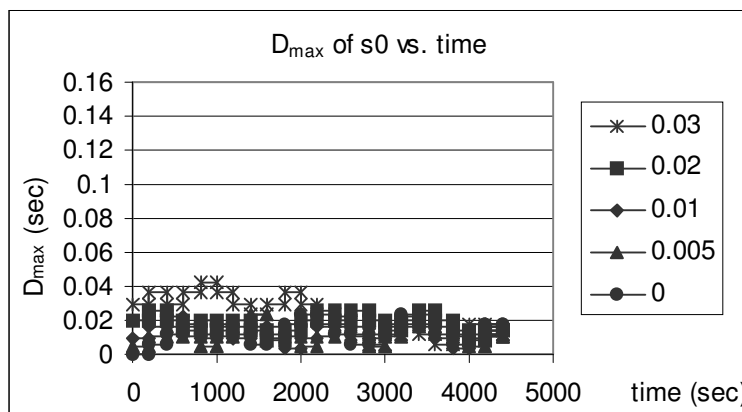
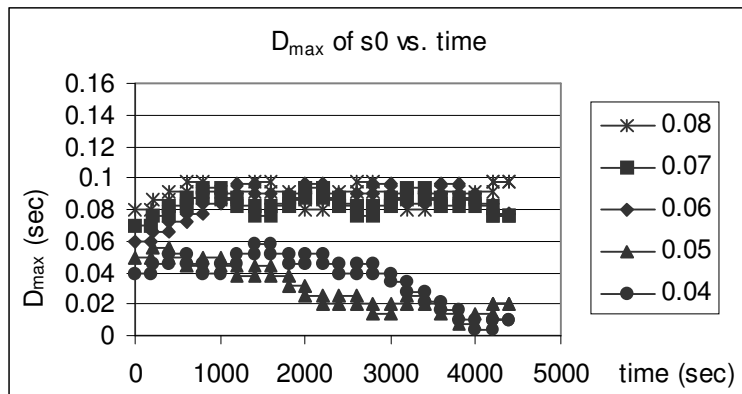
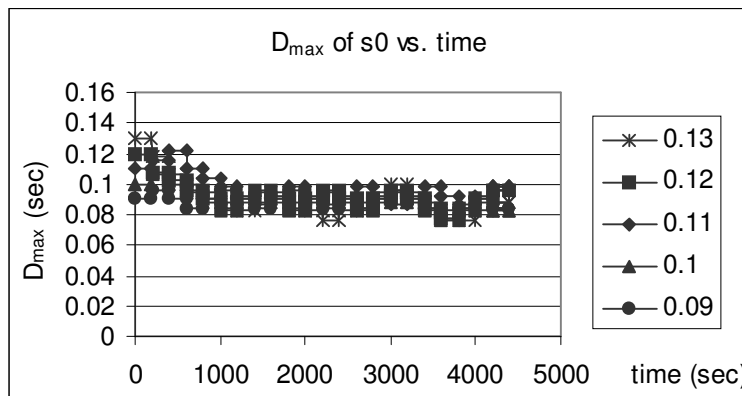
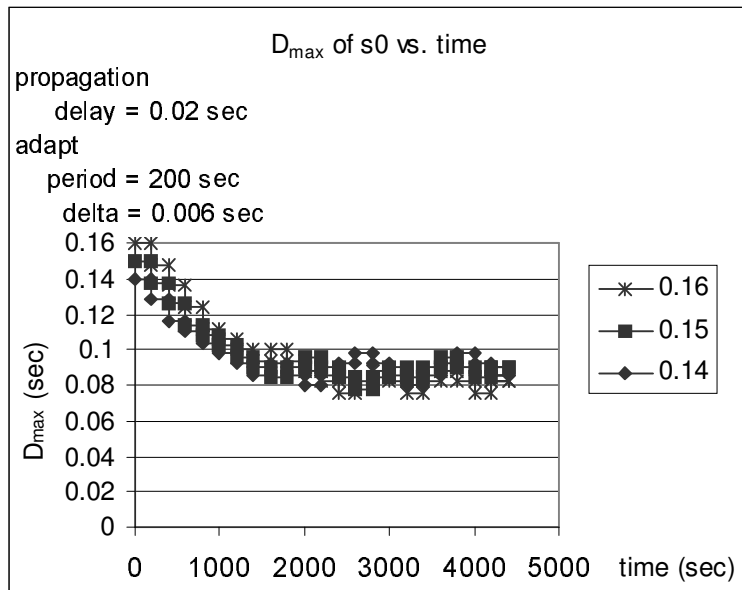


Figure 4.5: D_{max} of s0 vs. time with different initial D_{max} values, $\delta = 6$ msec.

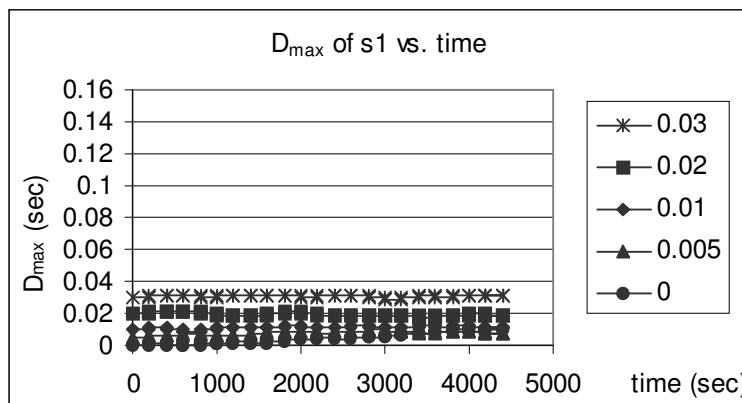
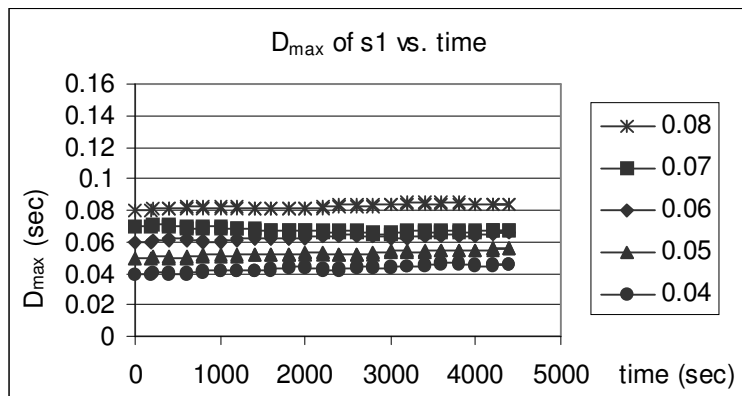
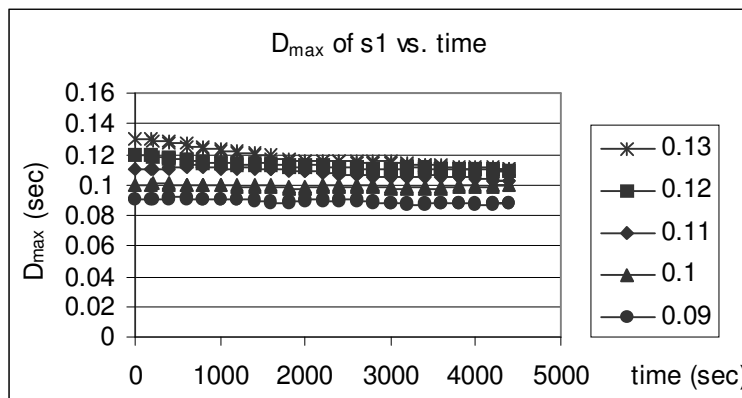
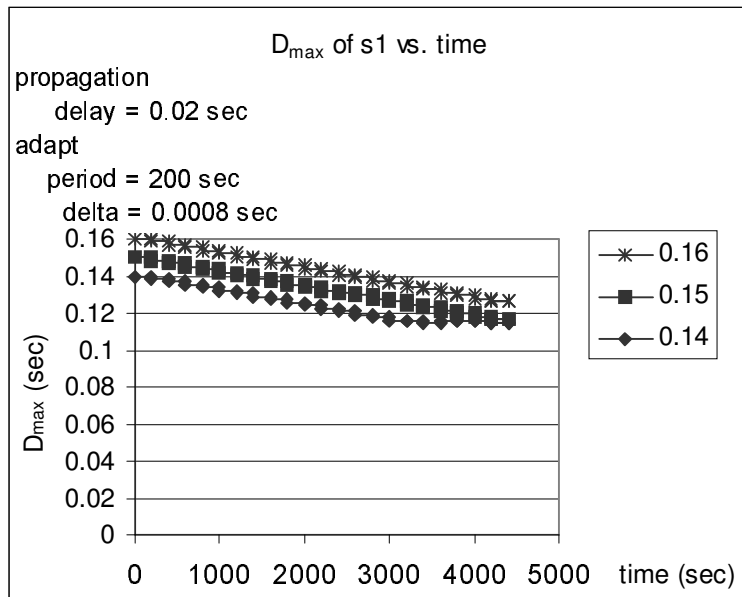


Figure 4.6: D_{max} of s1 vs. time with different initial D_{max} values, $\delta = 0.8$ msec.

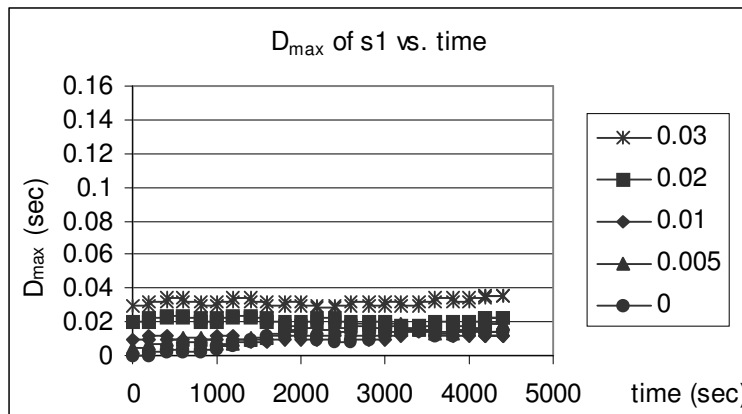
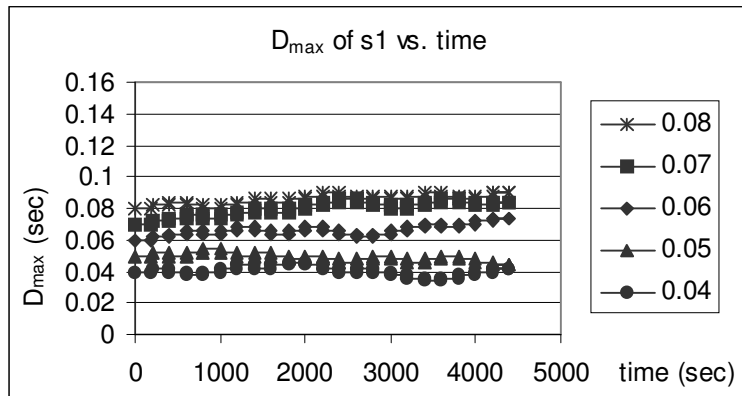
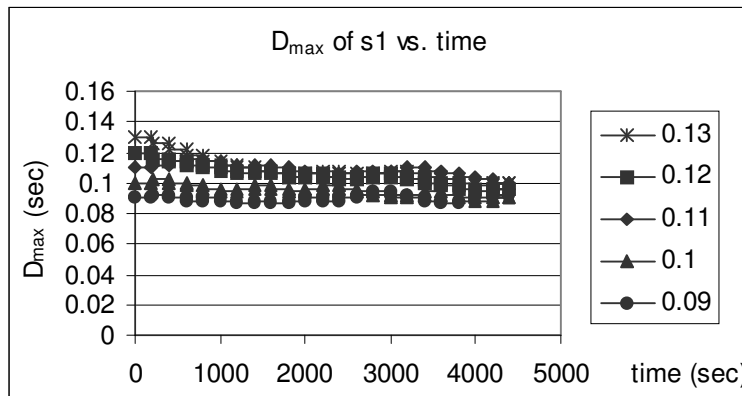
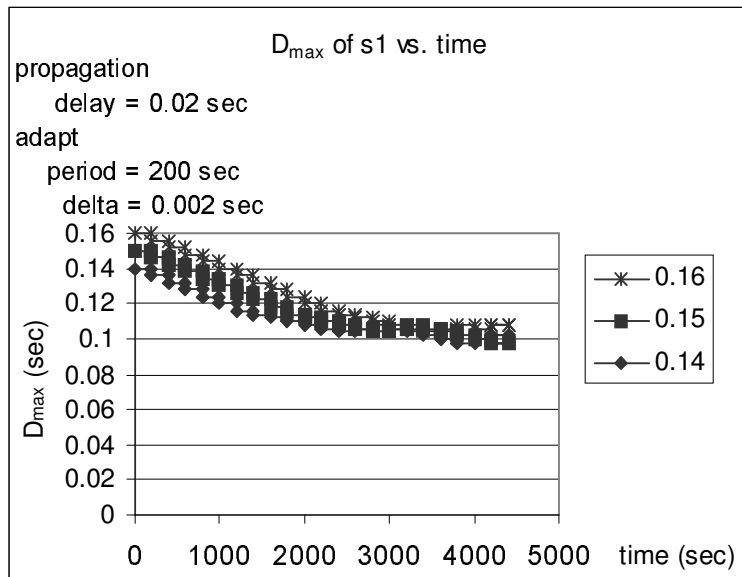


Figure 4.7: D_{max} of s1 vs. time with different initial D_{max} values, $\delta = 2$ msec.

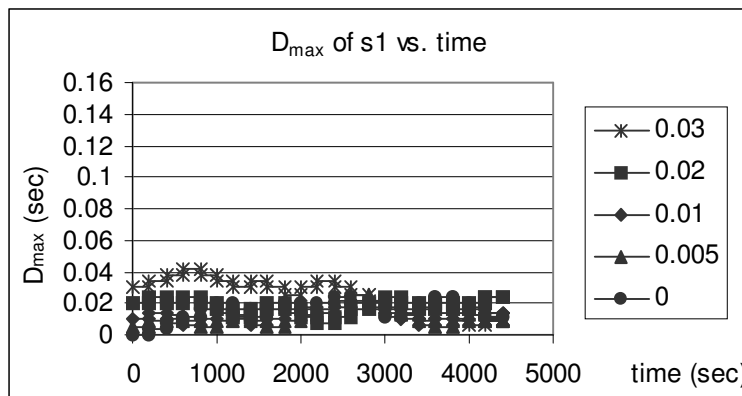
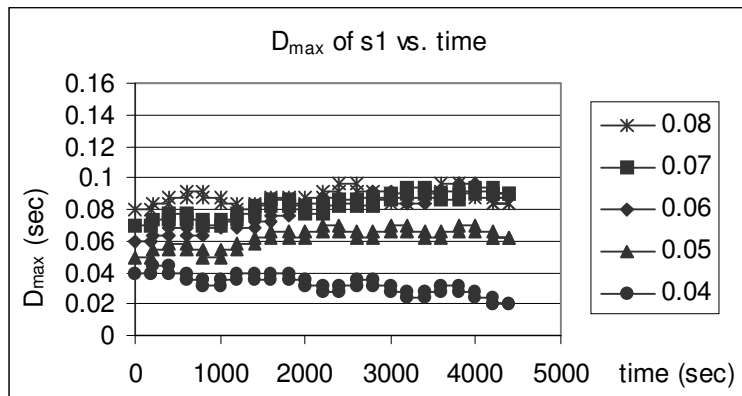
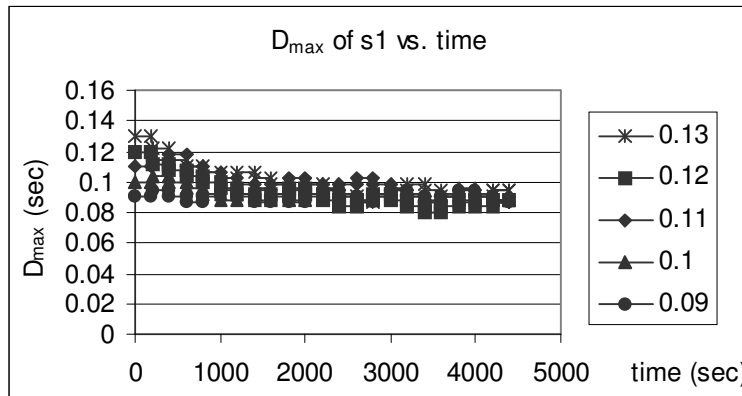
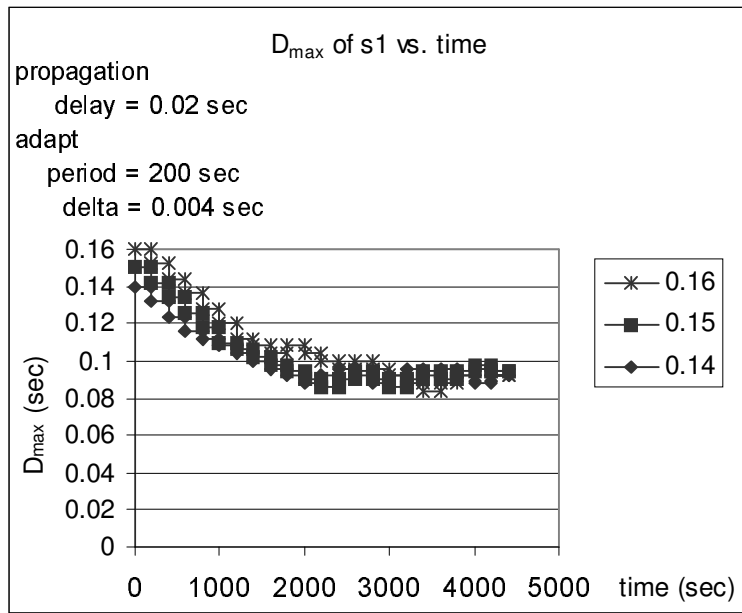


Figure 4.8: D_{max} of s1 vs. time with different initial D_{max} values, $\delta = 4$ msec.

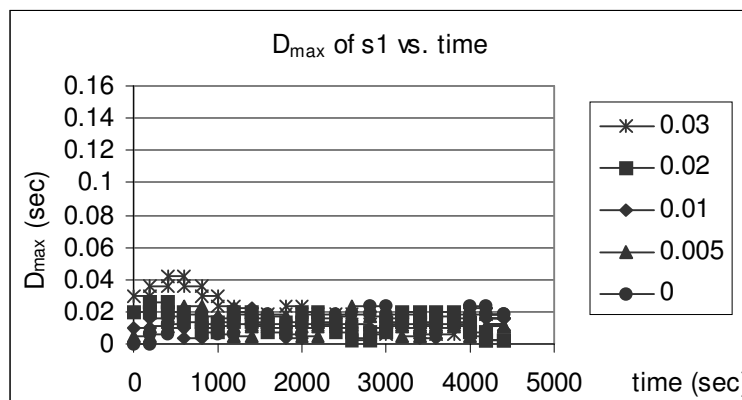
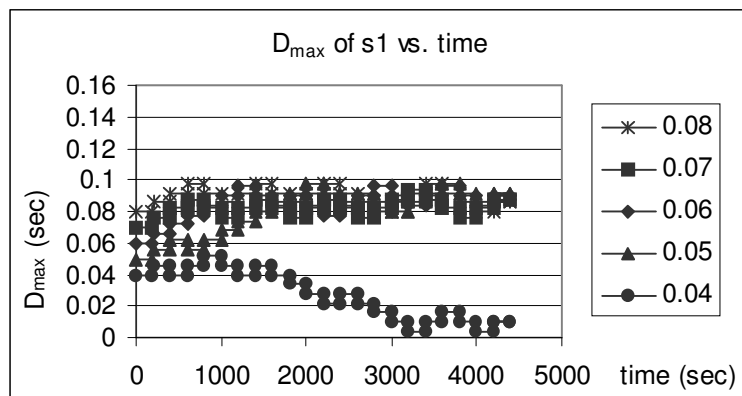
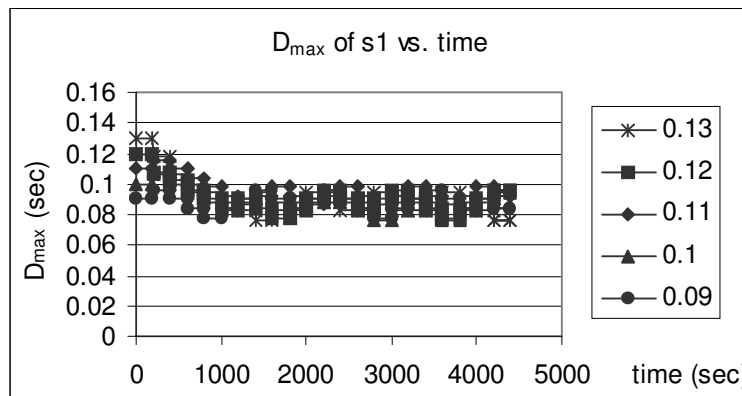
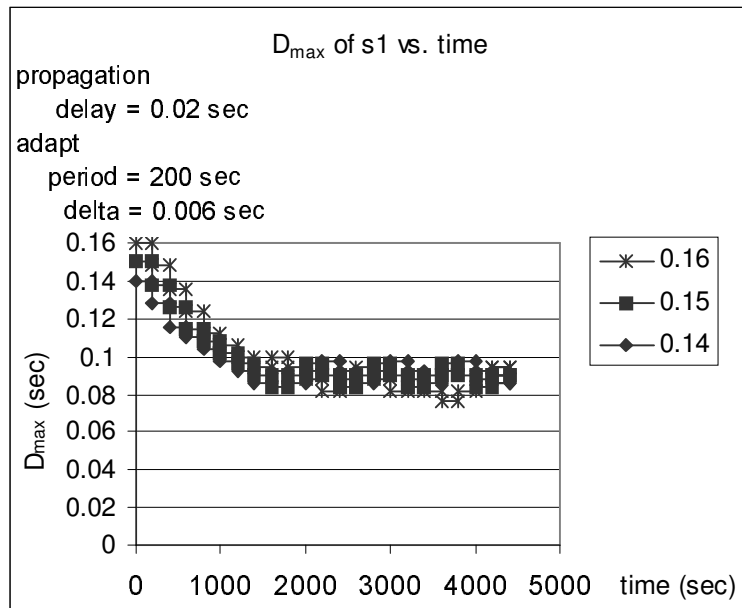


Figure 4.9: D_{max} of s1 vs. time with different initial D_{max} values, $\delta = 6$ msec.

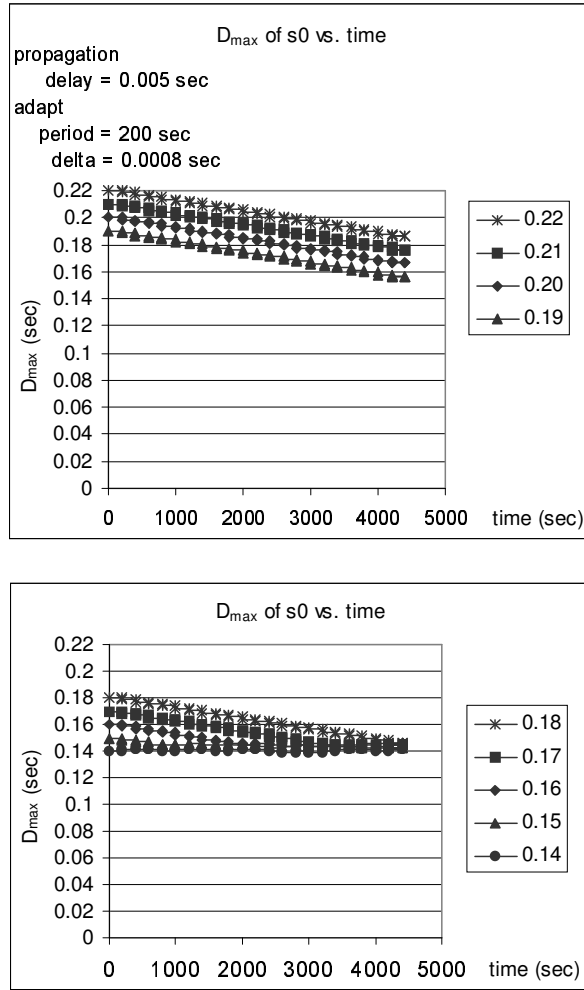


Figure 4.10: D_{max} of s0 vs. time with different initial D_{max} values, $P_D = 5$ msec, $\delta = 0.8$ msec.

In $P_D = 5$ msec case, in Figure 3.4, there are three locally optimum D_{max} values; one around $D_{max} = 5$ msec, another around $D_{max} = 50$ msec, and the last around $D_{max} = 120$ msec. Behavior of D_{max} for s0 with δ values of 0.8, 2, 4 and 6 msec are presented in Figures 4.10 through 4.17. Results for s1 are presented in Figures 4.18 through 4.25. $\delta = 6$ msec gives proper convergence behavior also in $P_D = 5$ msec case. Based on these results, $\delta = 6$ msec is used in all subsequent simulations.

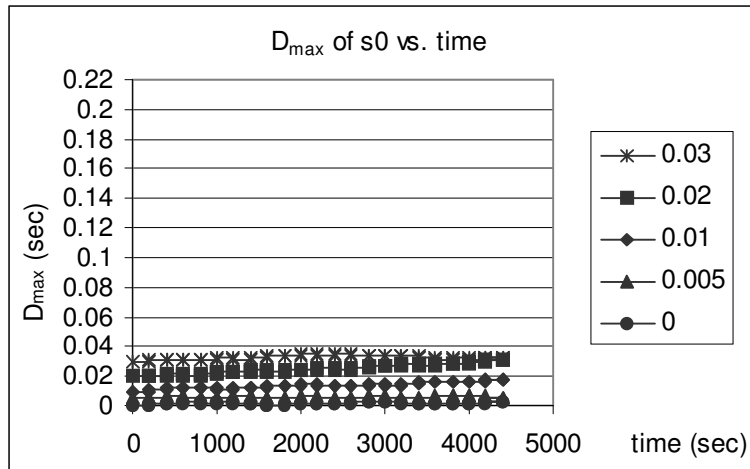
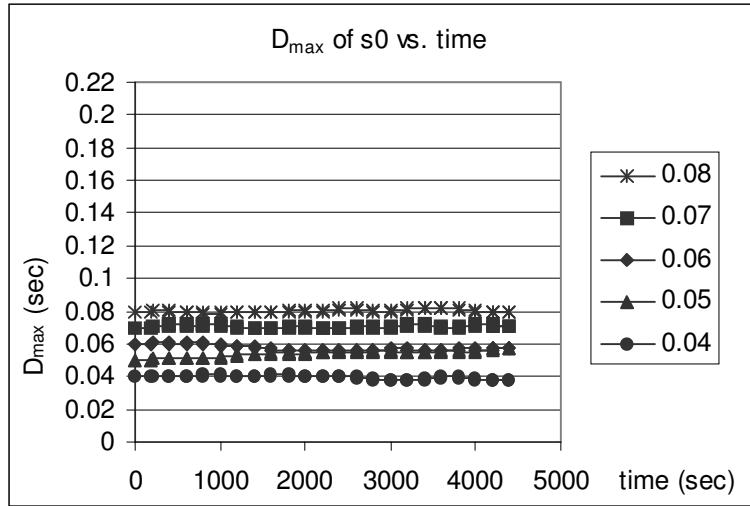
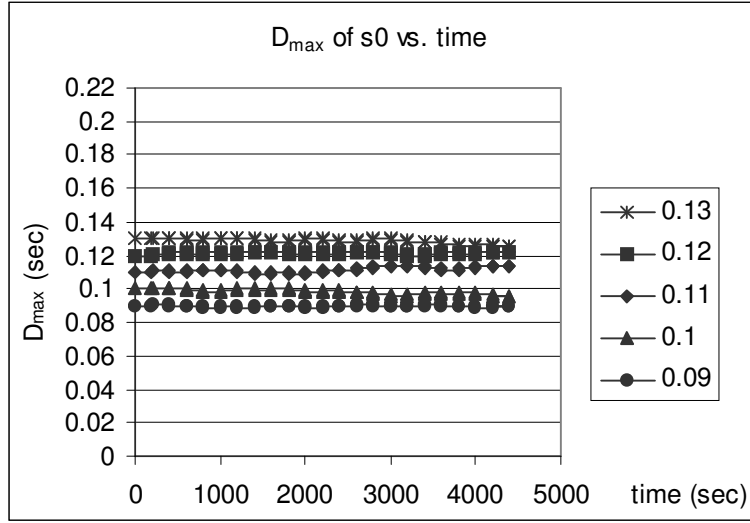


Figure 4.11: D_{max} of s_0 vs. time with different initial D_{max} values, $P_D = 5$ msec, $\delta = 0.8$ msec cont.'ed.

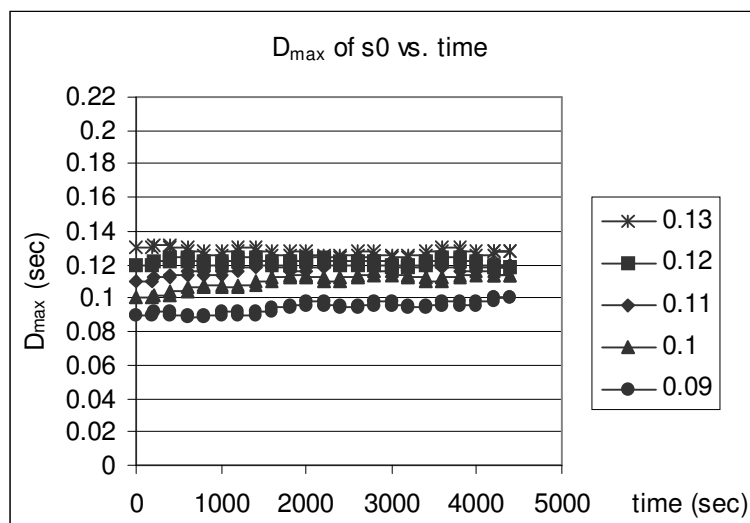
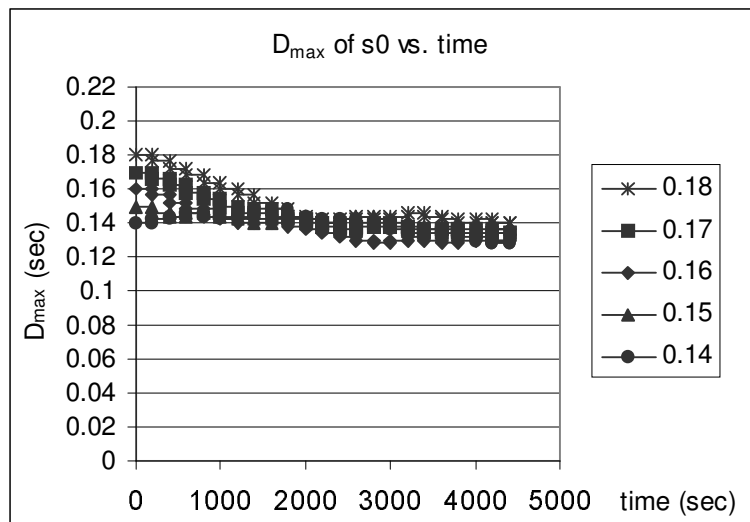
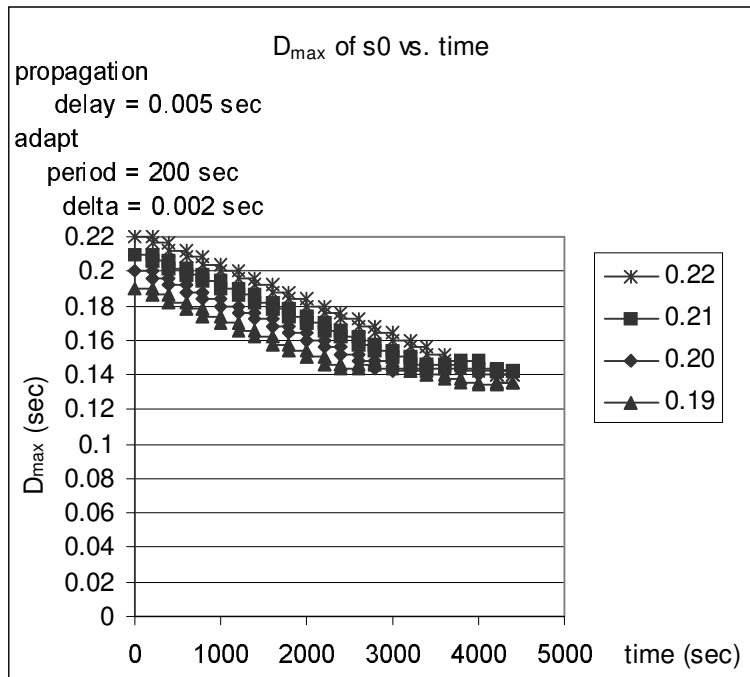


Figure 4.12: D_{max} of s0 vs. time with different initial D_{max} values, $P_D = 5$ msec, $\delta = 2$ msec.

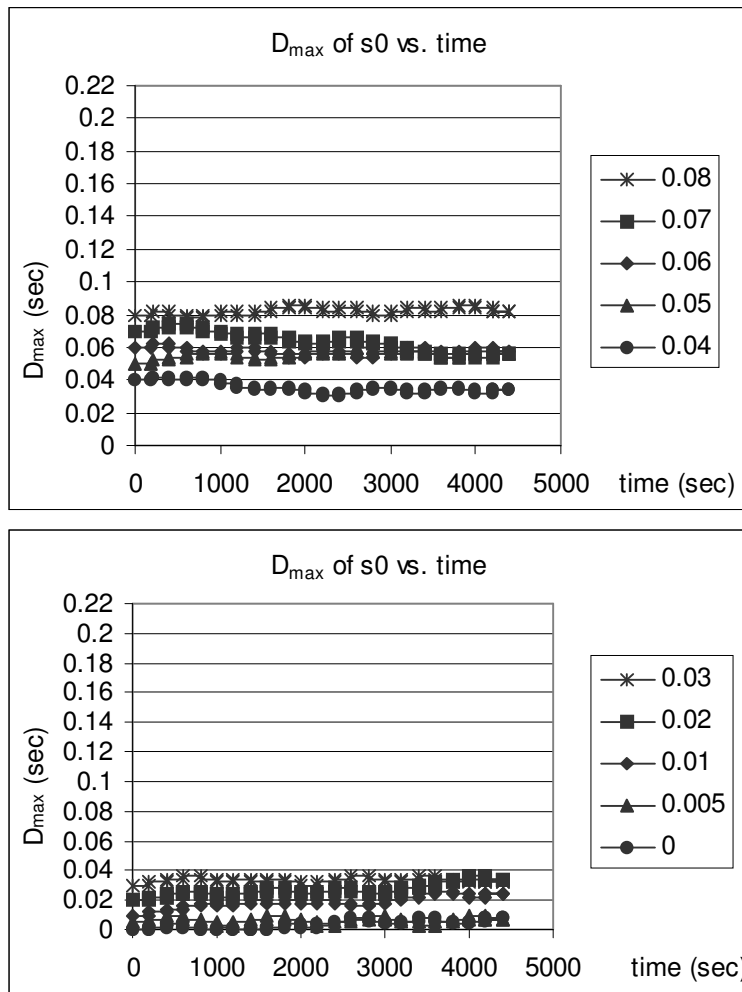


Figure 4.13: D_{max} of s0 vs. time with different initial D_{max} values, $P_D = 5$ msec, $\delta = 2$ msec cont.'ed.

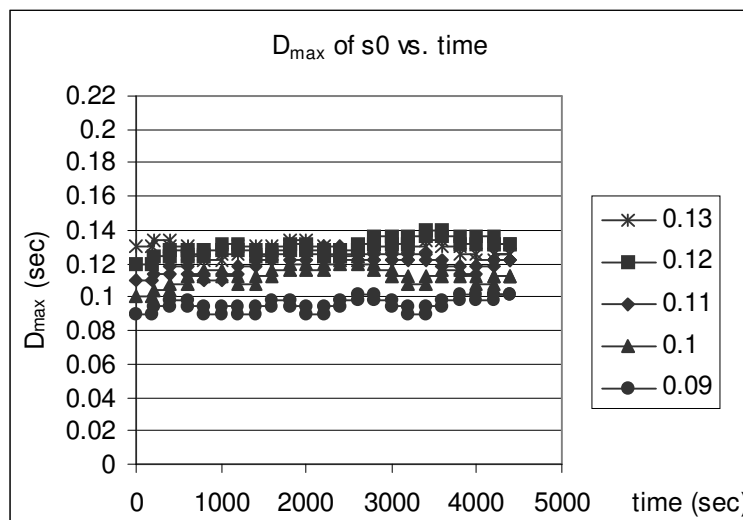
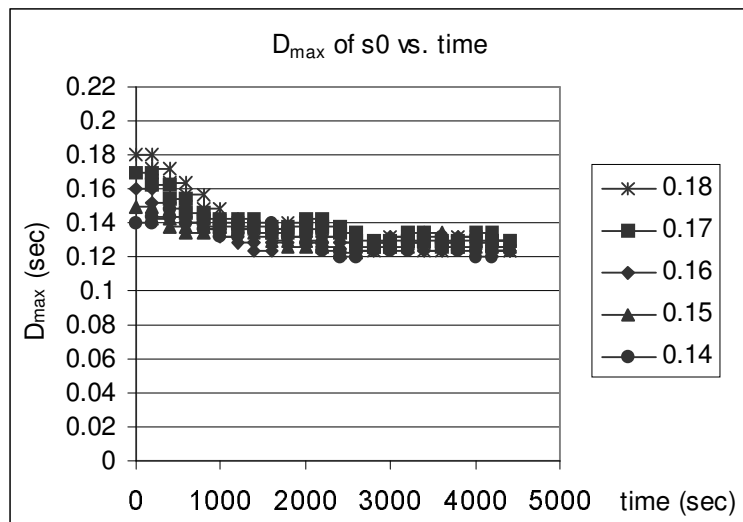
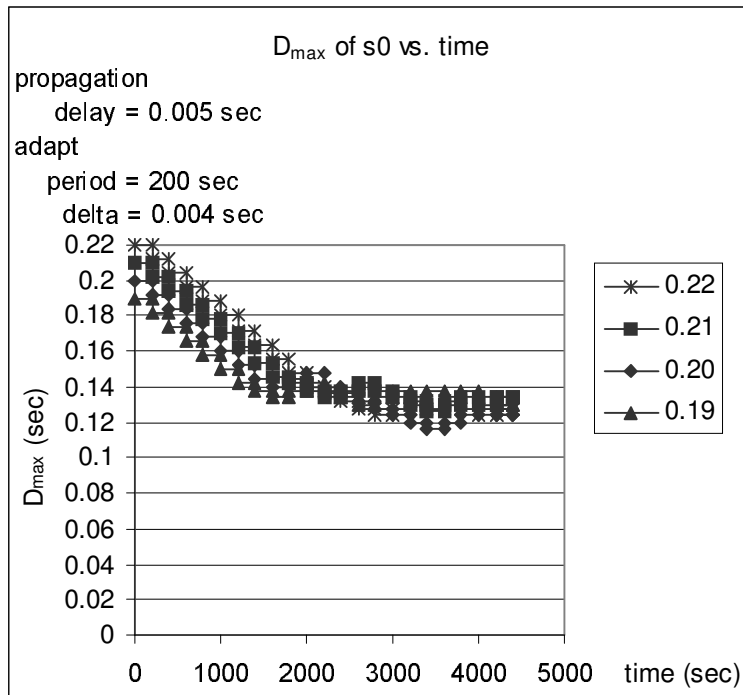


Figure 4.14: D_{max} of s0 vs. time with different initial D_{max} values, $P_D = 5$ msec, $\delta = 4$ msec.

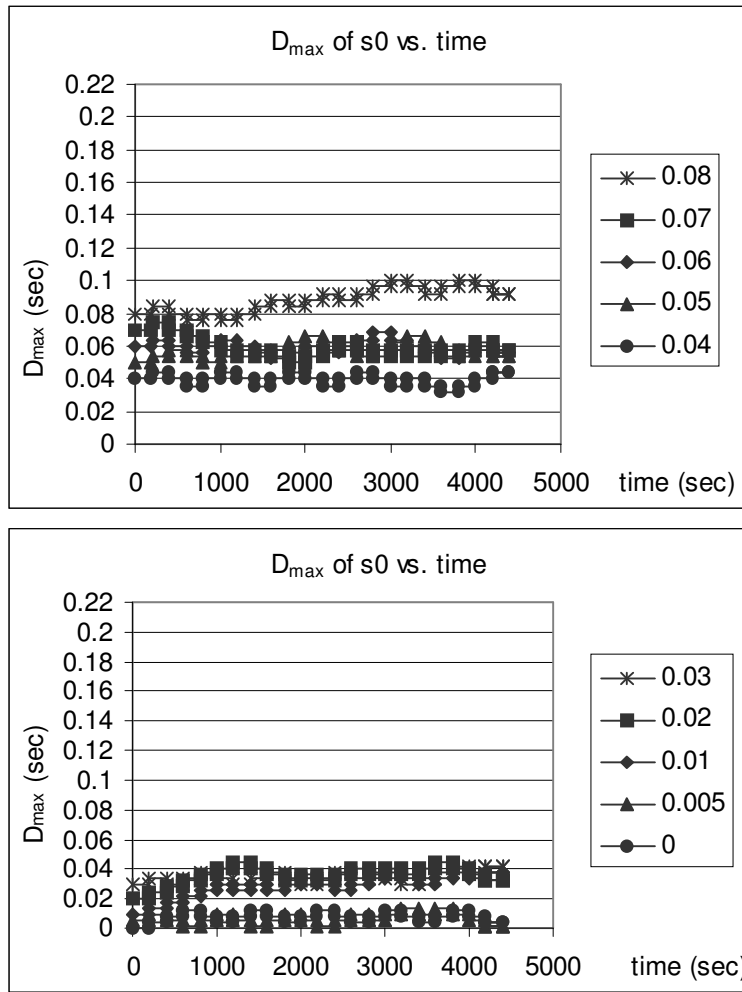


Figure 4.15: D_{max} of s0 vs. time with different initial D_{max} values, $P_D = 5$ msec, $\delta = 4$ msec cont.'ed.

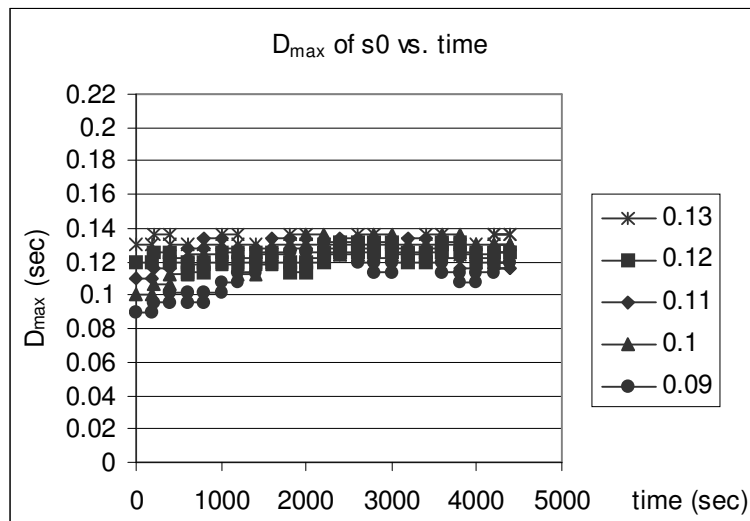
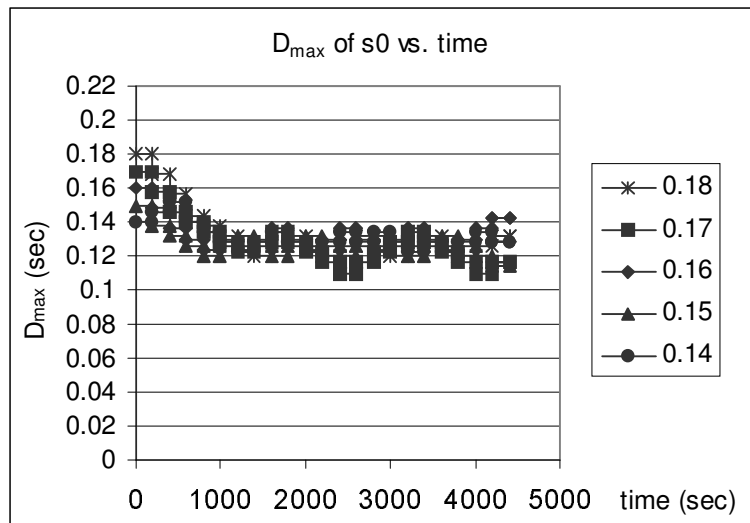
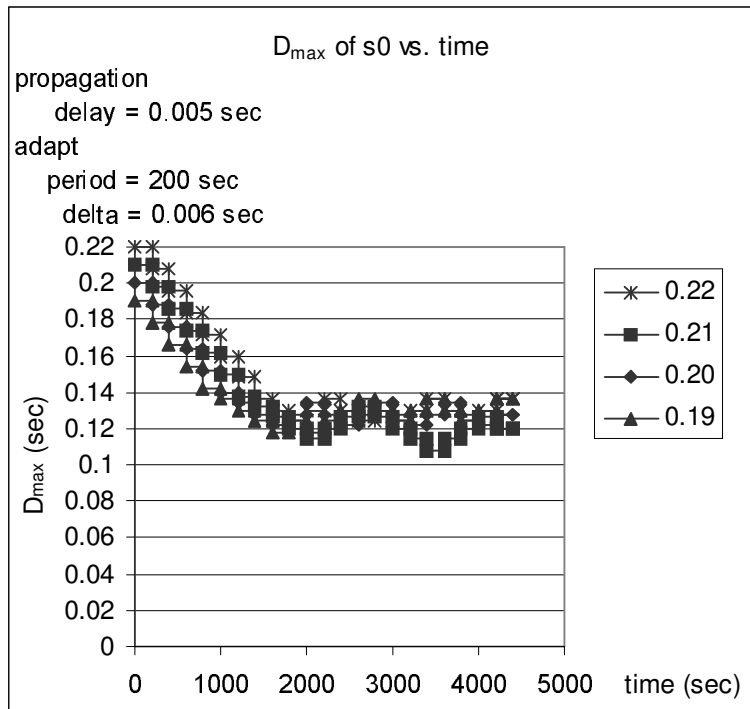


Figure 4.16: D_{max} of s0 vs. time with different initial D_{max} values, $P_D = 5$ msec, $\delta = 6$ msec.

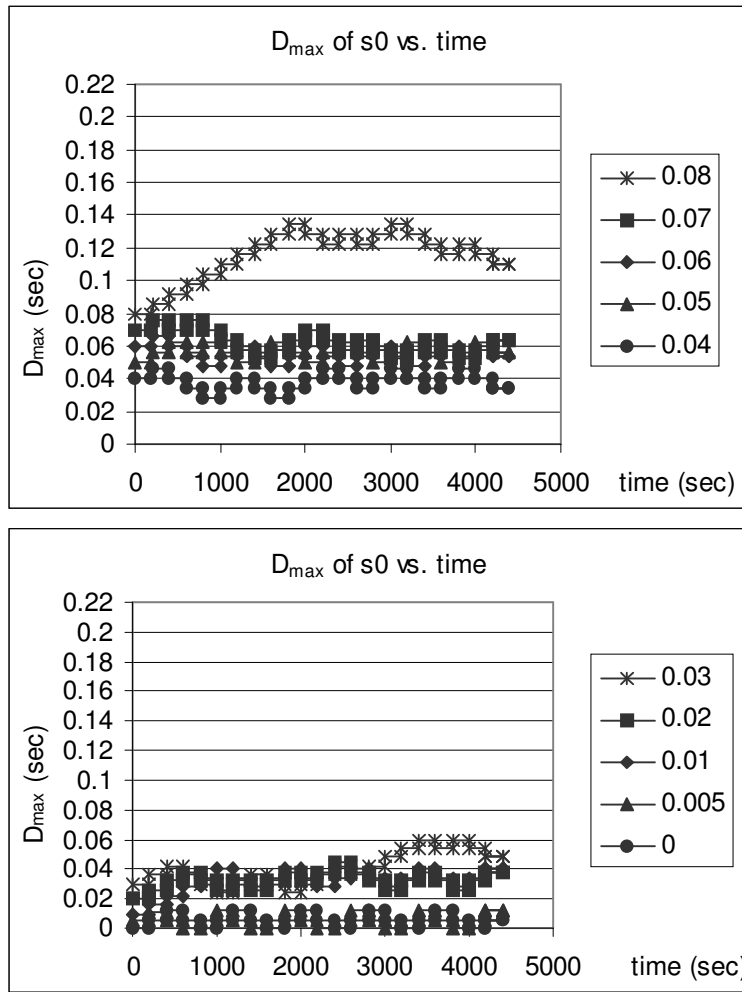


Figure 4.17: D_{max} of s_0 vs. time with different initial D_{max} values, $P_D = 5$ msec, $\delta = 6$ msec cont.'ed.

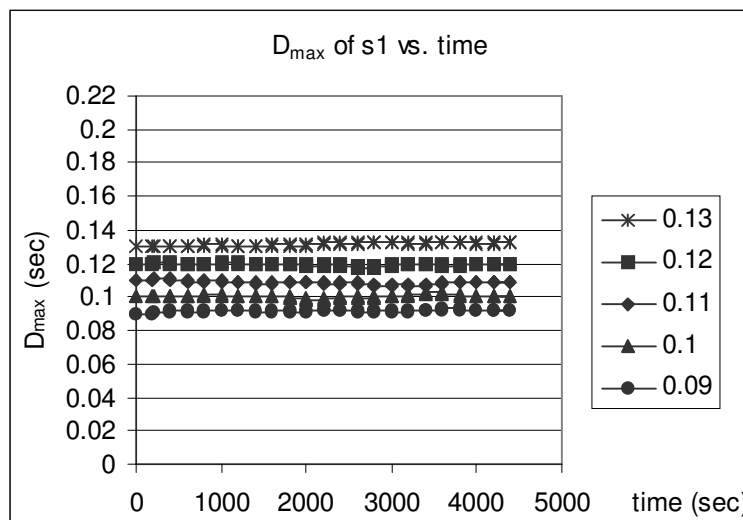
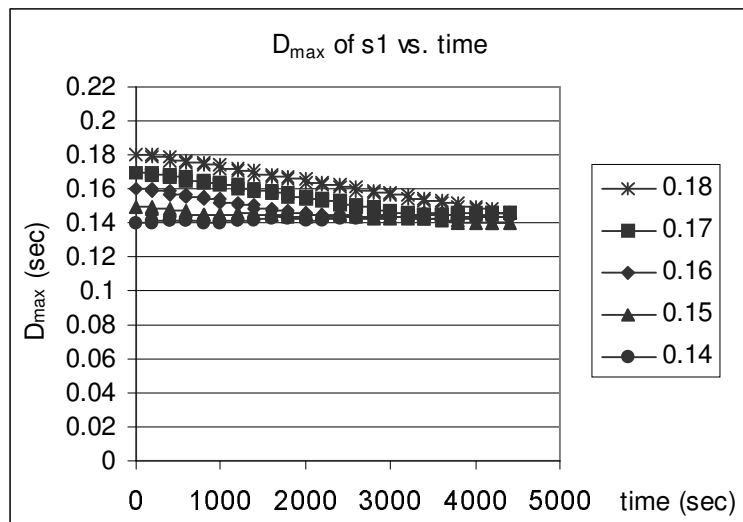
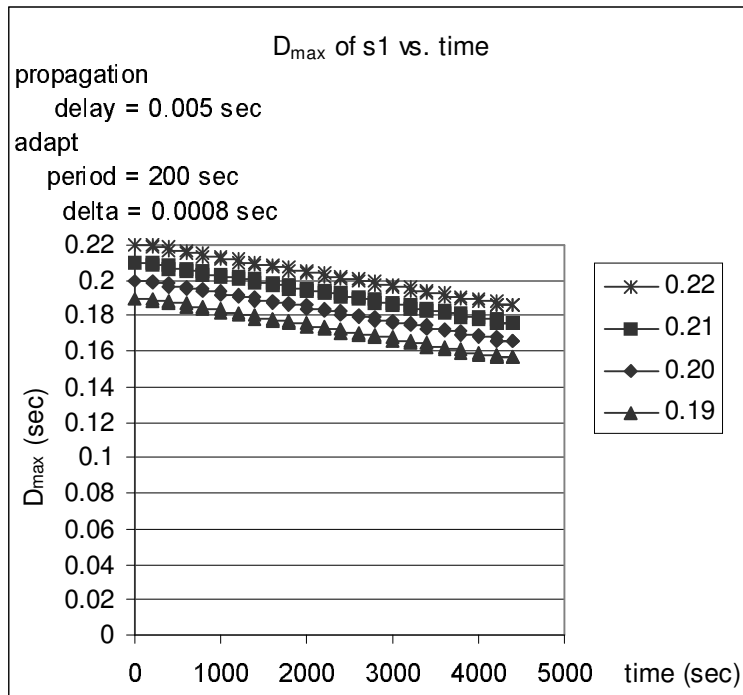


Figure 4.18: D_{max} of s1 vs. time with different initial D_{max} values, $P_D = 5$ msec, $\delta = 0.8$ msec.

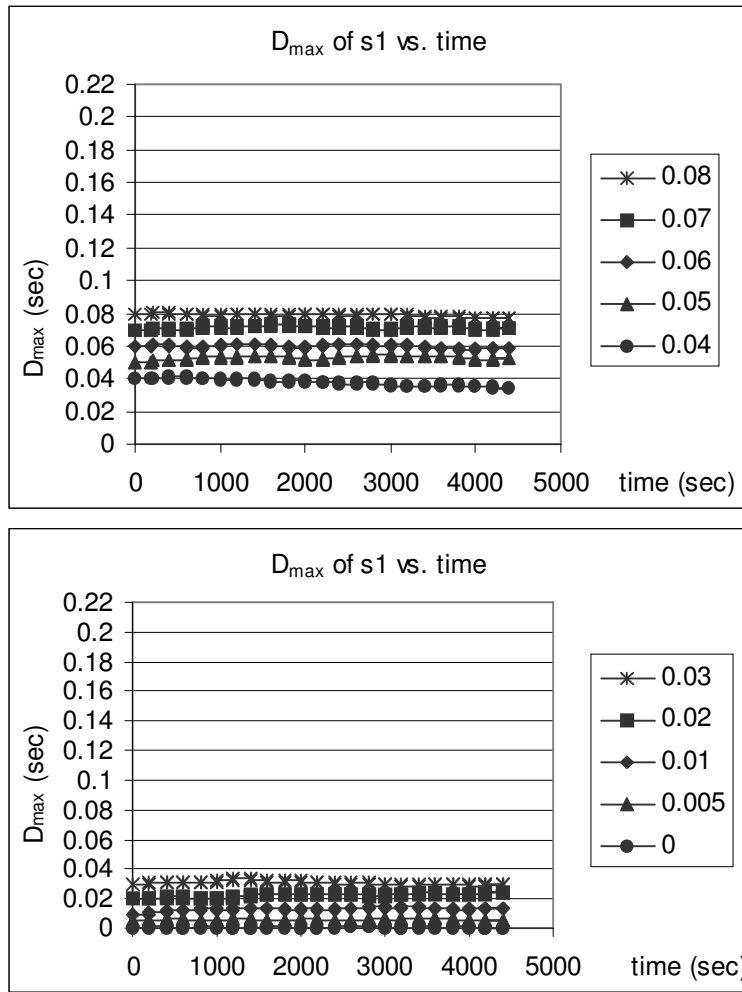


Figure 4.19: D_{max} of s1 vs. time with different initial D_{max} values, $P_D = 5$ msec, $\delta = 0.8$ msec cont.'ed.

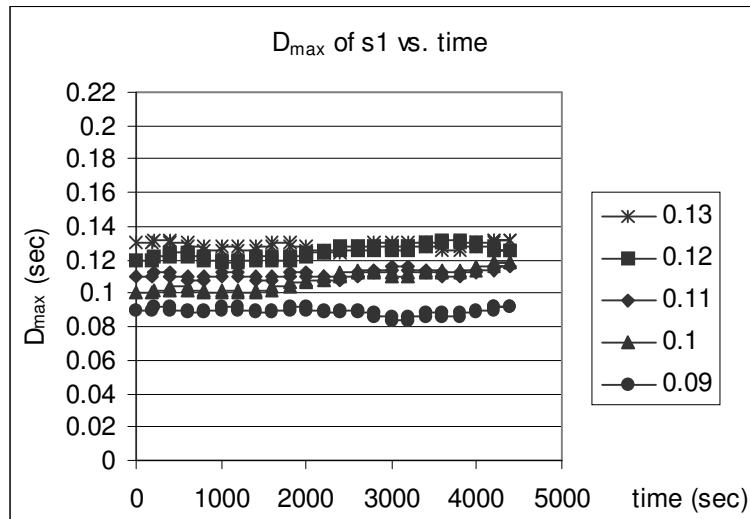
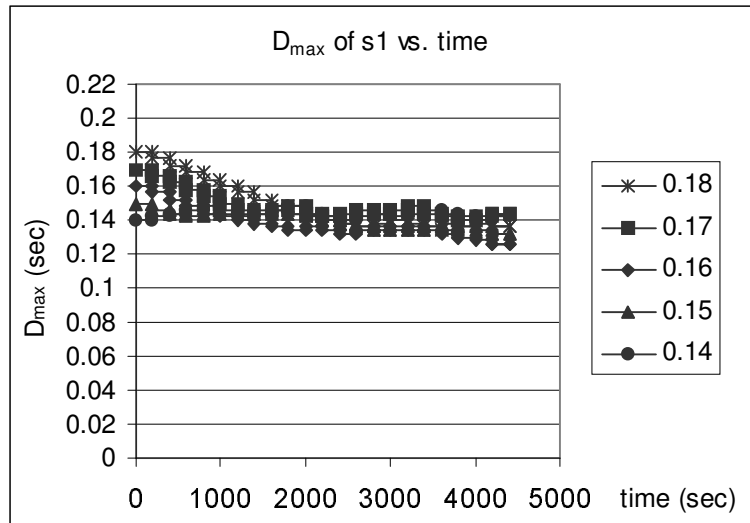
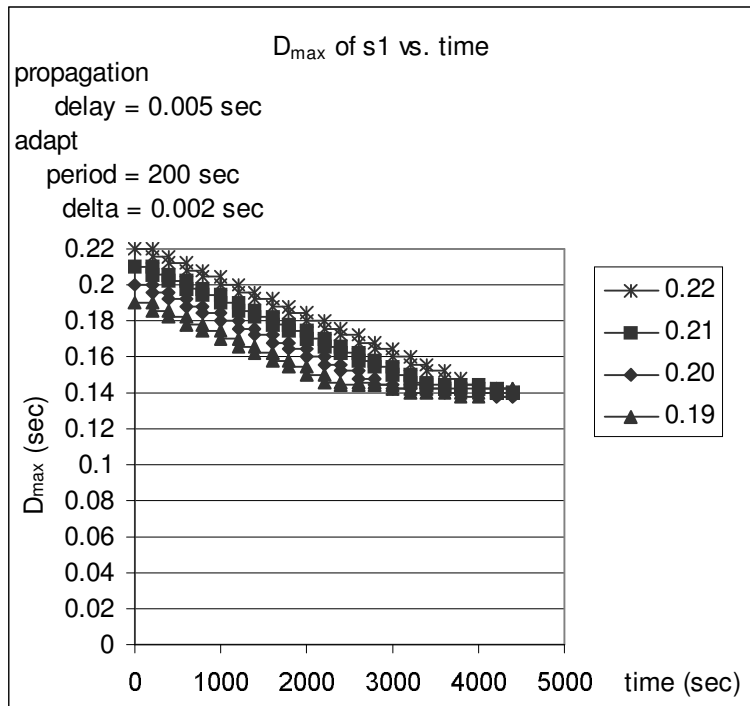


Figure 4.20: D_{max} of s1 vs. time with different initial D_{max} values, $P_D = 5$ msec, $\delta = 2$ msec.

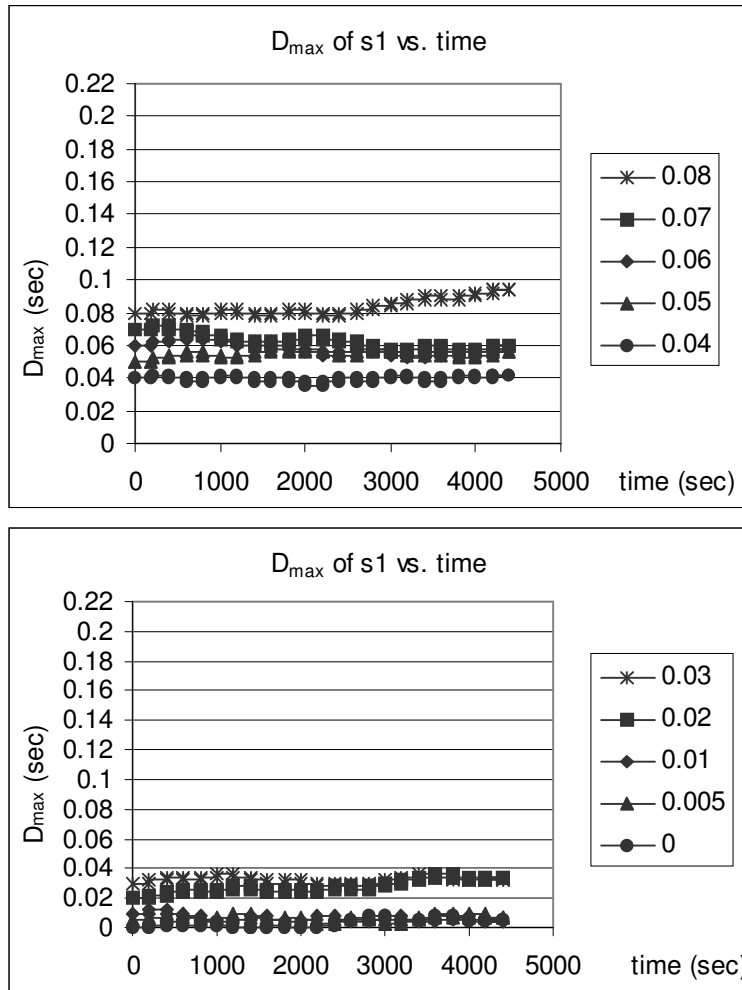


Figure 4.21: D_{max} of s1 vs. time with different initial D_{max} values, $P_D = 5$ msec, $\delta = 2$ msec cont.'ed.

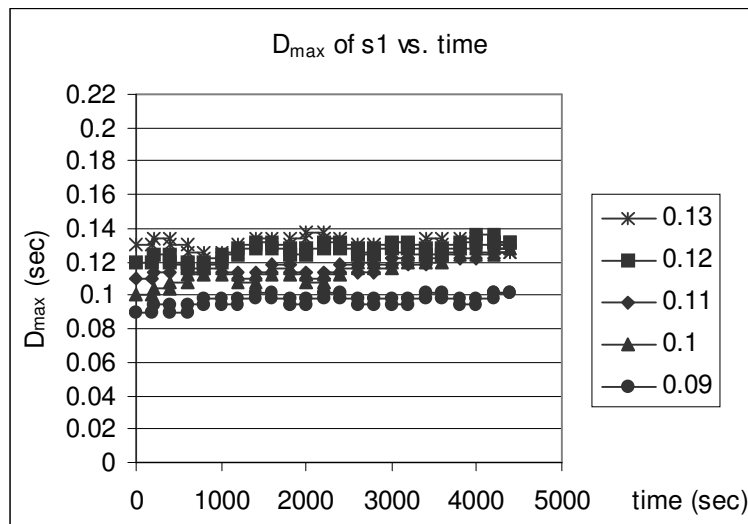
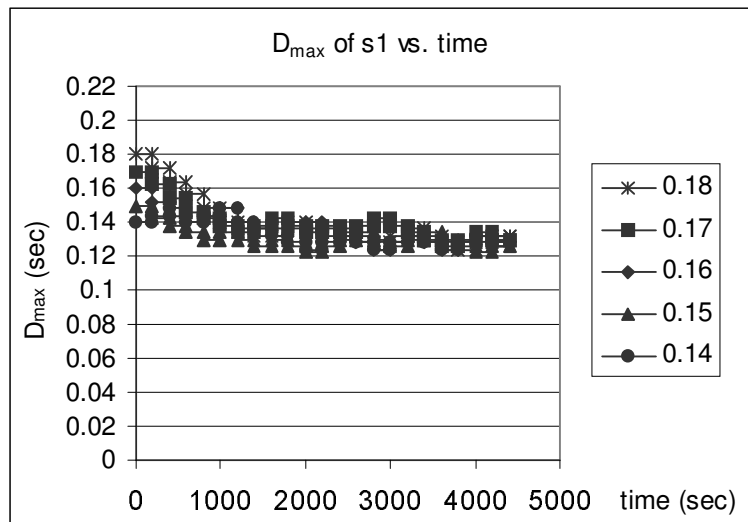
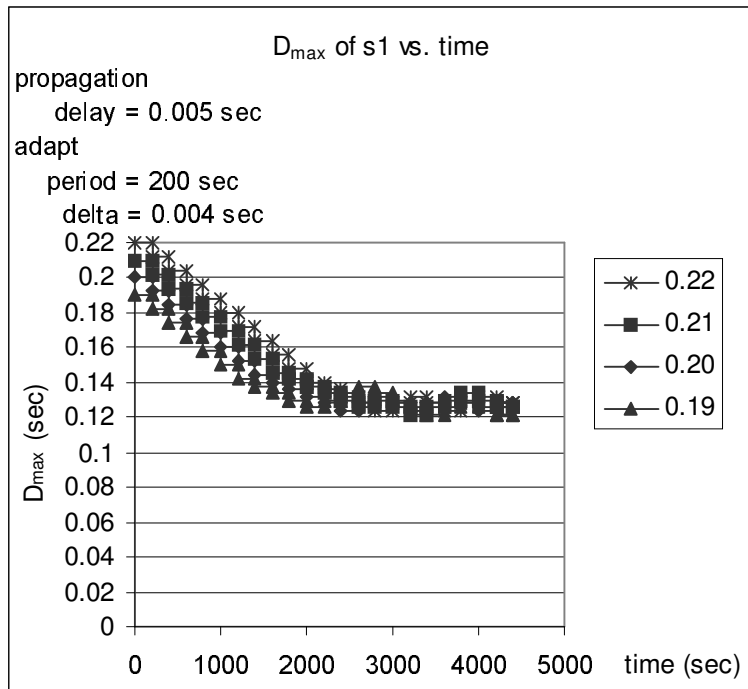


Figure 4.22: D_{max} of s1 vs. time with different initial D_{max} values, $P_D = 5$ msec, $\delta = 4$ msec.

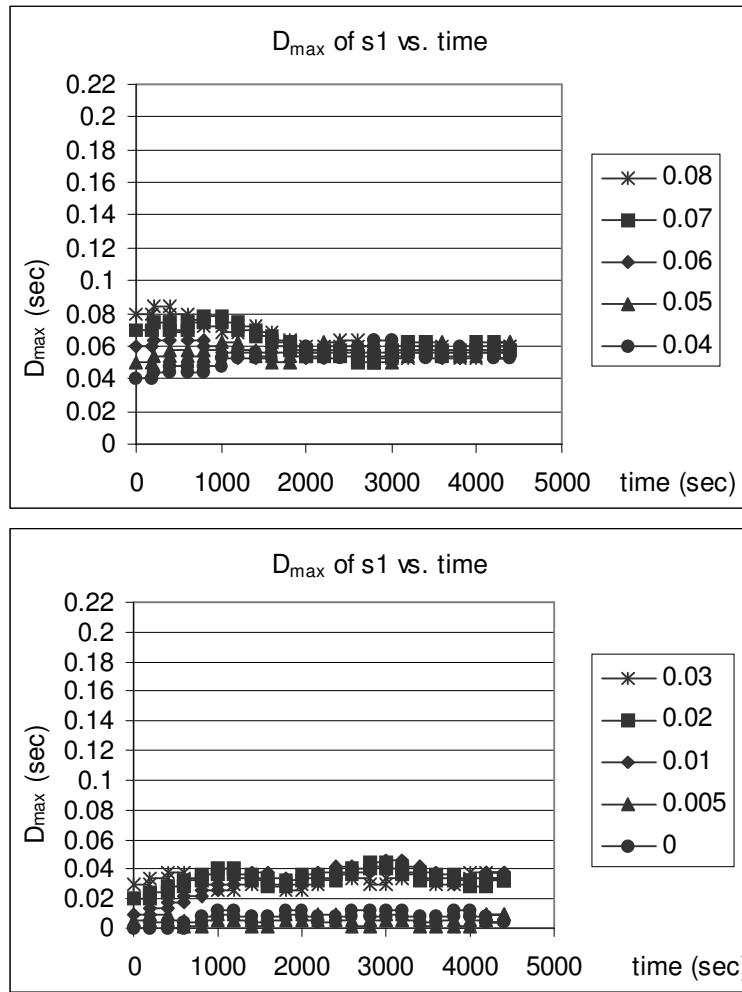


Figure 4.23: D_{max} of s1 vs. time with different initial D_{max} values, $P_D = 5$ msec, $\delta = 4$ msec cont.'ed.

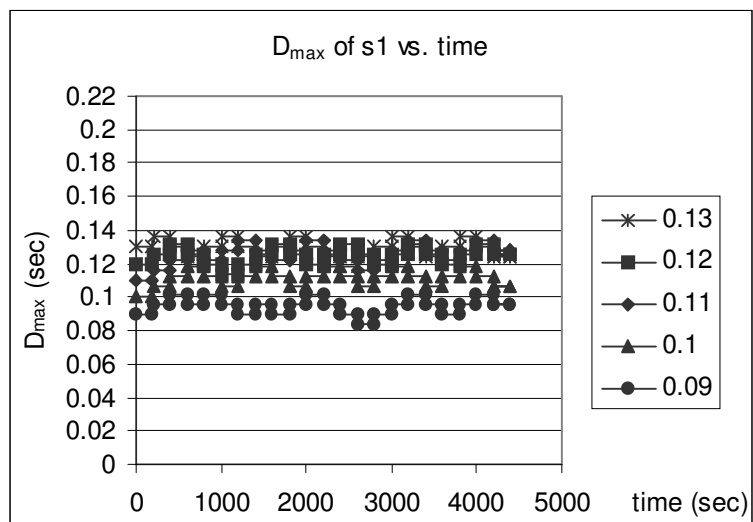
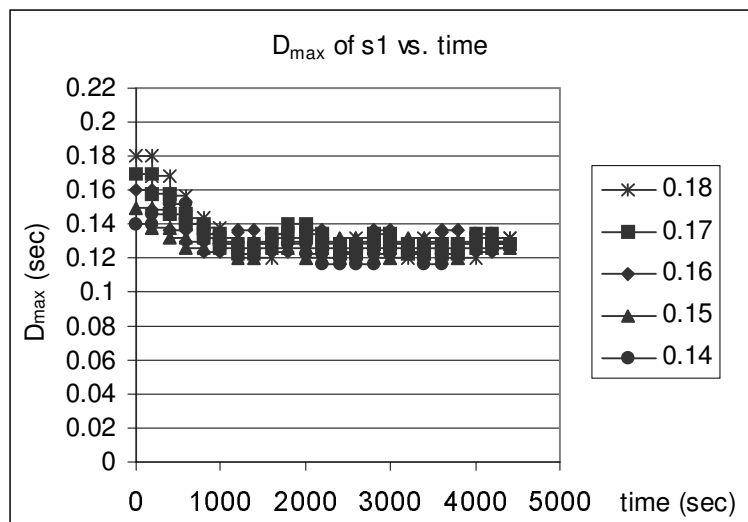
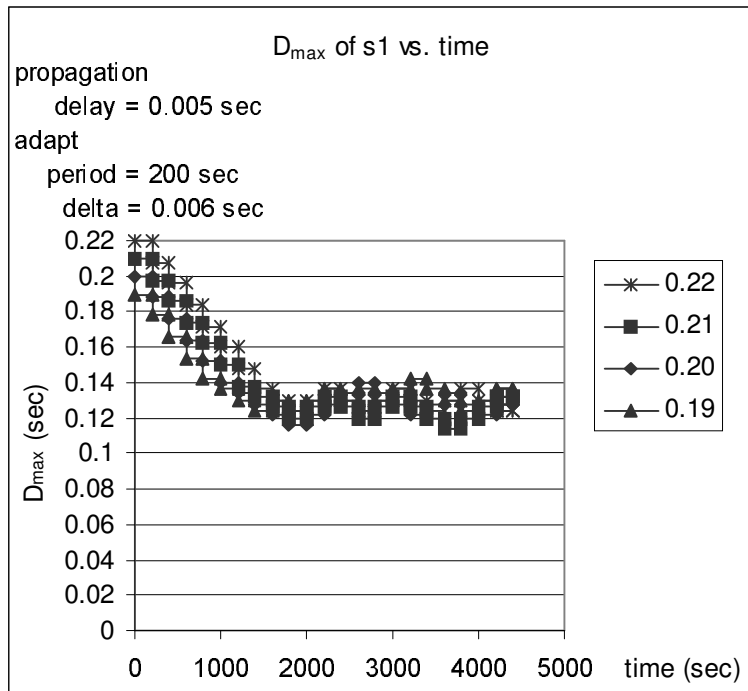


Figure 4.24: D_{max} of s1 vs. time with different initial D_{max} values, $P_D = 5$ msec, $\delta = 6$ msec.

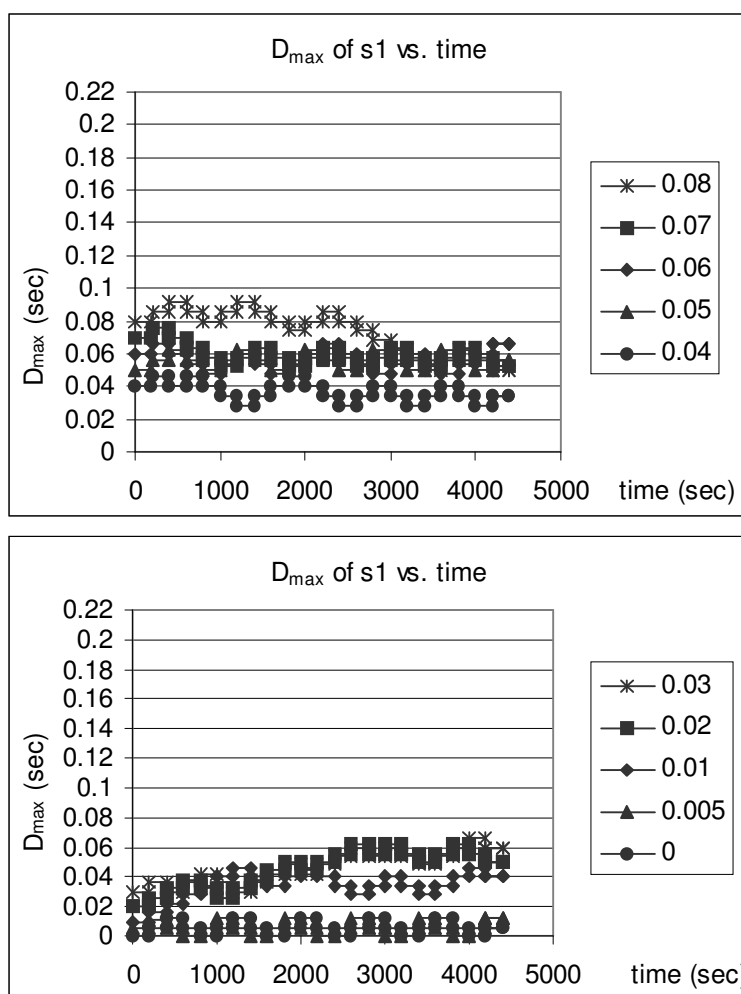


Figure 4.25: D_{max} of s1 vs. time with different initial D_{max} values, $P_D = 5$ msec, $\delta = 6$ msec cont.'ed.

After finding proper value for δ , effects of some parameters on performance of D-DBRAS are investigated in the next section.

4.4.2 Effects of Some Parameters on Performance of D-DBRAS

Effects of P_D , level of congestion in the network, ratio of shaped traffic to total traffic, and RED parameters on performance of D-DBRAS are investigated. In this section, in graphs, curves labelled as $(d$ or $f)$ - si where $i \in \{0, 1\}$ belong to the TCP connection from node si in simulation where both of shapers are either

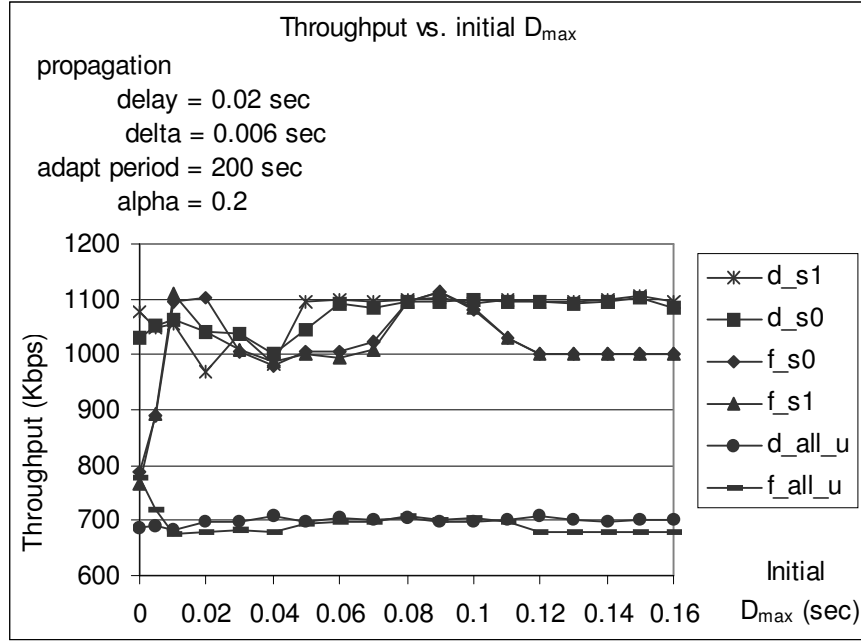


Figure 4.26: Throughput vs. initial D_{max} graph for $P_D = 20$ msec case.

D-DBRAS (d) or DBRAS (f). Curves labelled as (d or f) $_{all_u}$ belong to the average of (all of the) remaining eight unshaped TCP connections.

The Effect of Propagation Delay

To observe the effect of P_D on the performance of D-DBRAS, 2 simulations are done with each of $P_D = 20$ and 5 msec. For each P_D , one simulation is done using DBRAS and the other is done using D-DBRAS. The average throughputs achieved by individual shaped TCP traffic streams and average throughput for unshaped TCP traffic are observed. Green ratios (ratio of the number of green packets to total number of packets of a traffic) of individual shaped traffic and total unshaped traffic are observed. The throughput vs. initial D_{max} graph for $P_D = 20$ msec case is shown in Figure 4.26. The throughput vs. initial D_{max} graph for $P_D = 5$ msec is shown in Figure 4.27.

In both graphs, the behavior of throughput of shaped traffic with DBRAS as D_{max} increases is similar to the one observed in Chapter 3. Using D-DBRAS,

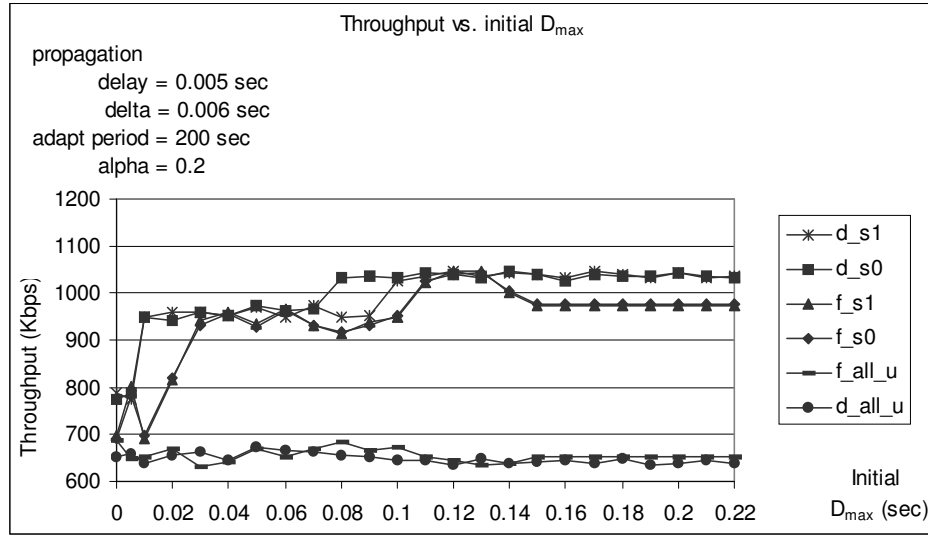


Figure 4.27: Throughput vs. initial D_{max} graph for $P_D = 5$ msec case.

it can be observed that the behavior is smoother than the behavior obtained using DBRAS and fluctuations in throughput as initial D_{max} increases are more gradual than those seen with DBRAS. In both graphs, it can be observed that with most of the initial D_{max} values, D-DBRAS manages to converge to one of optimum D_{max} values, which is implied by the fact that its throughput levels reach locally optimum values. It can be observed further that throughput levels that can be achieved in $P_D = 5$ msec case are lower than those that can be achieved in $P_D = 20$ msec case, as observed and explained in Chapter 3. Since unnecessarily large values of D_{max} are avoided, D-DBRAS does not have the performance degradation for large values of D_{max} which occurs in DBRAS. Average throughput for unshaped TCP traffic behaves in response to changes in throughput of shaped TCP traffic.

The green ratio vs. initial D_{max} graph for $P_D = 20$ msec and $P_D = 5$ msec cases can be seen in Figures 4.28 and 4.29, respectively. Both of these figures give insight for the convergence of D_{max} in D-DBRAS and they are consistent with graphs for throughput.

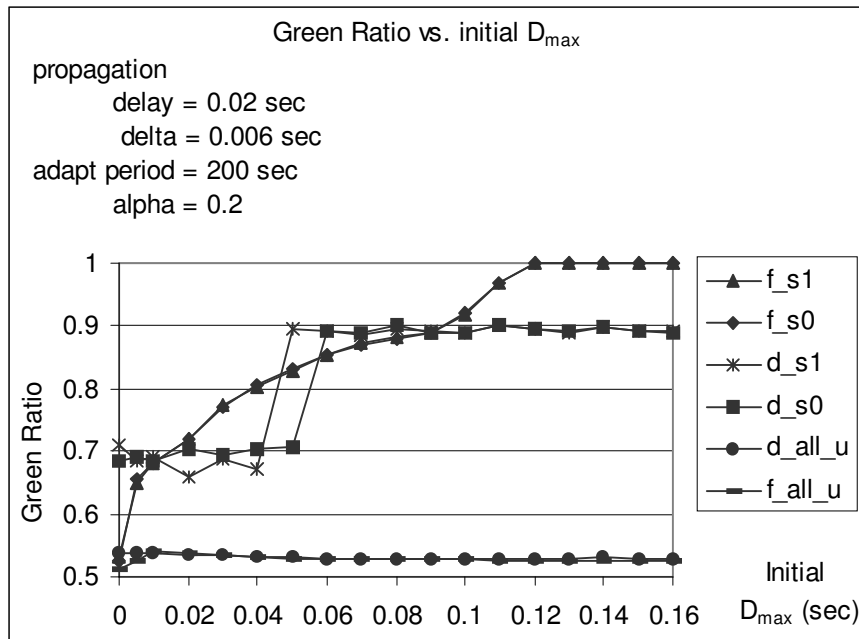


Figure 4.28: Green ratio vs. initial D_{max} for $P_D = 20$ msec case.

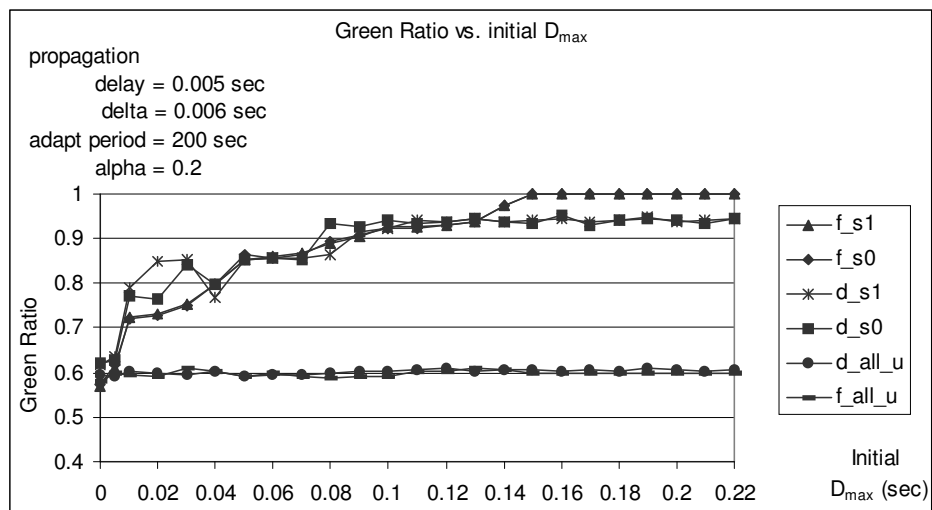


Figure 4.29: Green ratio vs. time for $P_D = 5$ msec case.

P_D (msec)	initial D_{max} (msec)	s0 (segments)	s1 (segments)	o8 (segments)
5	50	11.29	11.08	6.05
5	120	17.66	17.80	6.00
20	20	11.91	11.06	8.33
20	90	17.74	17.93	8.43

Table 4.1: Actual TCP window size measurements.

P_D (msec)	initial D_{max} (msec)	s0		s1		o8	
		f	d	f	d	f	d
5	50	57	63	57	62	26	26
5	120	116	124	116	125	26	27
20	20	47	44	47	41	37	37
20	90	110	104	110	106	37	37

Table 4.2: End-to-end delay measurements (in msec).

The optimum values of two P_D cases are further analyzed in terms of actual window size of TCP (the number of unacknowledged TCP segments), average end-to-end delay (delay from sender TCP to receiver TCP), average shaping delay, and dropping probabilities of packets with respect to their color in the congested link (e1, core).

Measurements for actual TCP window sizes are shown in Table 4.1. The column labelled as *o8* (appears also in subsequent tables) refers to the average of 8 unshaped TCP traffic. The difference between window sizes for shaped and unshaped traffic in each row of Table 4.1 explains the difference between average throughputs achieved by shaped and unshaped traffic as shown in Figures 4.26 and 4.27.

Measurements for end-to-end delay (in msec) and shaping delay (in msec) can be seen in Tables 4.2 and 4.3, respectively. It can be observed that shaping delay constitutes a significant portion of total end-to-end delay.

Measurements for dropping probability at the congested link (e1, core) for simulations with D-DBRAS can be seen in Table 4.4. For $P_D = 5$ msec case, it can be seen that one of the reasons of having a greater throughput value for shaped

P_D (msec)	initial D_{max} (msec)	s0		s1	
		f	d	f	d
5	50	37	43	37	42
5	120	96	103	96	104
20	20	13	10	13	8
20	90	77	72	77	74

Table 4.3: Shaping delay measurements (in msec).

P_D (msec)	initial D_{max} (msec)	green early drop prob.		
		s0	s1	o8
5	50	0.0	0.0	0.0
5	120	0.0	0.0	0.0
20	20	0.0	0.0	0.0
20	90	0.0	0.0	0.0
		green hard drop prob.		
5	50	1.3e-3	1.2e-3	1.0e-3
5	120	1.3e-3	1.2e-3	1.0e-3
20	20	2.1e-5	1.5e-5	0.3e-5
20	90	0.2e-5	0.7e-5	0.0
		red early drop prob.		
5	50	3.1e-2	3.0e-2	3.1e-2
5	120	2.2e-2	2.3e-2	3.0e-2
20	20	2.1e-2	2.1e-2	2.5e-2
20	90	1.7e-2	1.6e-2	2.4e-2
		red hard drop prob.		
5	50	9.2e-2	9.4e-2	1.3e-1
5	120	6.2e-3	5.3e-3	1.4e-1
20	20	4.1e-2	4.6e-2	7.1e-2
20	90	6.3e-3	5.9e-3	7.5e-2

Table 4.4: Dropping probability measurements.

traffic with initial D_{max} of 120 msec than the throughput value with initial D_{max} value of 50 msec is the decrease in dropping probabilities of red packets. Similar reasoning can be done for $P_D = 20$ msec case, between initial D_{max} values 90 and 20 msec. As shaping increases, queue load decreases due to packets waiting longer in D-DBRAS. This also decreases the dropping probability of red packets of shaped traffic since dropping probability for red packets is more sensitive to queue load due to lower RED buffer threshold parameters.

In the following part, the effect of the level of congestion in the SP network on performance of D-DBRAS is investigated.

The Effect of Level of Congestion in the Network

To observe this effect, a comparison is made between two cases: when the network is heavily-congested and when the network has a lower level of congestion. The first case corresponds to the simulations carried out in the previous part, which is referred to as original network. Similar to the original network case, less-congested network case is analyzed with those two P_D cases. The original network topology is modified in 2 of the 3 parts presented previously to obtain the less-congested network case. In network design part, links (e1, core), (core, e2), (e2, dest), (dest, e2), (e2, core), and (core, e1) are modified so that each has 15 Mbps bandwidth. This way, queue load of link (e1, core) decreases, compared to its load in the original network. In DiffServ configuration part, each TBM's CIR is modified to be 1.5 Mbps, the evenly-divided network bandwidth share. Throughput levels and green ratios are observed with respect to initial D_{max} in these simulations.

The throughput level achieved vs. initial D_{max} curves for $P_D = 20$ msec case are shown in Figure 4.30. It can be observed that shaped traffic manages to reach its average sending rate of 1.5 Mbps with most of D_{max} values using DBRAS. Nevertheless, it should be noted that for some small initial D_{max} values, D-DBRAS is able to increase throughput level achieved significantly. The throughput level achieved vs. initial D_{max} curves for $P_D = 5$ msec case are shown in Figure 4.31, where similar observations can be drawn.

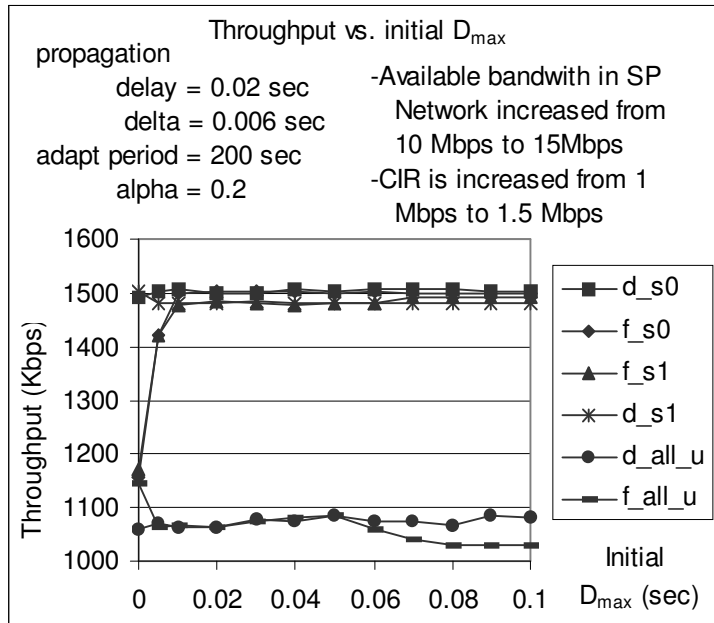


Figure 4.30: Throughput vs. initial D_{max} , for the less-congested network and $P_D = 20$ msec.

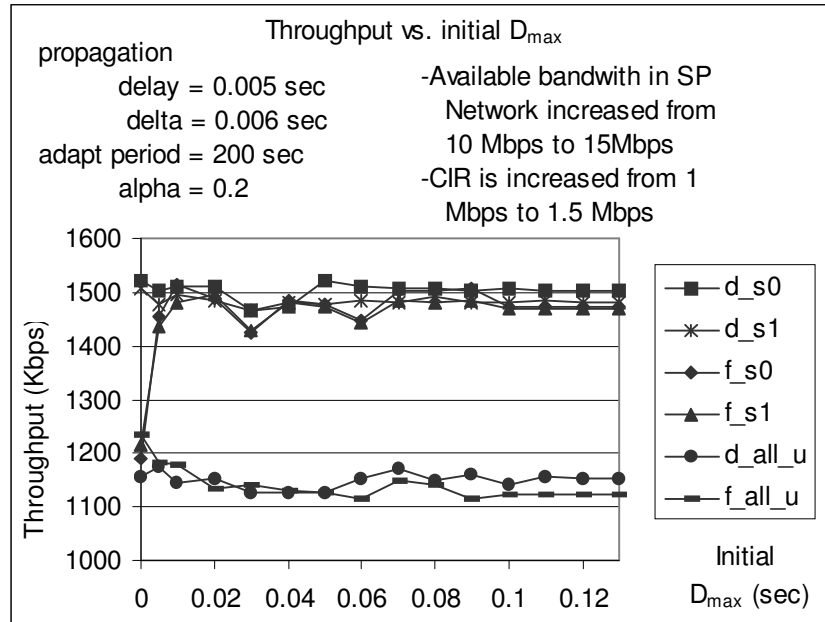


Figure 4.31: Throughput vs. initial D_{max} , for the less-congested network and $P_D = 5$ msec.

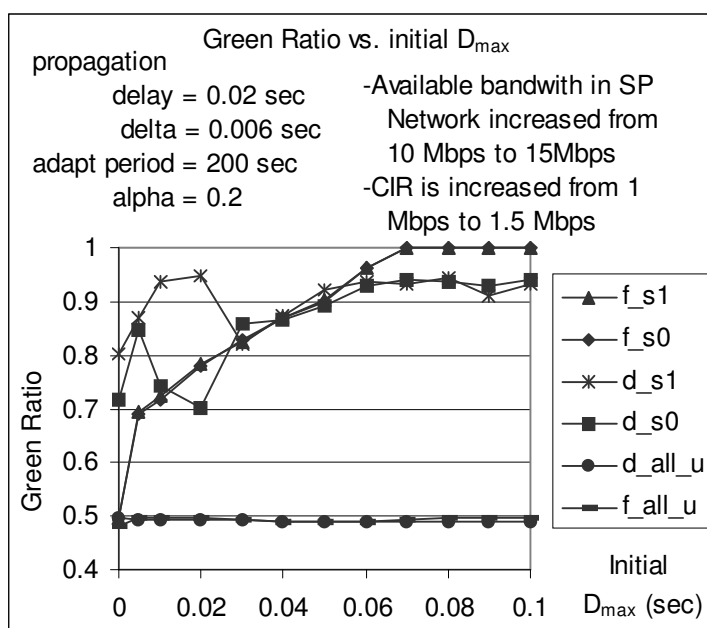


Figure 4.32: Green Ratio vs. initial D_{max} , for the less-congested network and $P_D = 20$ msec.

Green ratio vs. initial D_{max} graphs for $P_D = 20$ and 5 msec cases in the less-congested network are shown in Figures 4.32, and 4.33, respectively. The behavior of green ratios as initial D_{max} increases is roughly the same as the one in green ratio vs. initial D_{max} graphs for the original network (Figures 4.28 and 4.29). It should be noted that initial D_{max} values by which traffic enters all-green case in the less-congested network is smaller than those values for the original network. This is thought to be because of the maximum congestion window size that can be reached in the less-congested network being bigger than that of the original network, which results from decreased queue load at congested link and less number of drops.

In the next part, simulations for observing the effect of ratio of shaped traffic to total traffic in the network are presented.

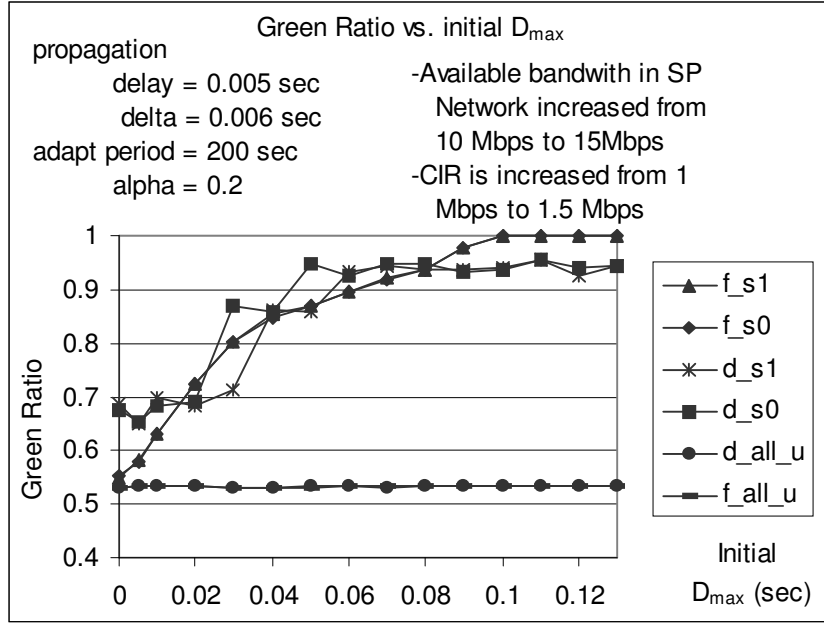


Figure 4.33: Green Ratio vs. initial D_{max} , for less-congested network and $P_D = 5$ msec.

The Effect of Ratio of Shaped Traffic to Total Traffic in the Network

To observe this effect, a group of network topologies that differ in the number of shaped TCP traffic out of 10 TCP traffic are used. In the first topology, which is shown in Figure 4.34, none of 10 TCP streams is shaped. This topology can be visualized as removing two shaper nodes from Figure 4.1 and using links $(s_i, e1)$ and $(e1, s_i)$ where $i \in \{0, 1\}$ with 10 Mbps rate and 5 msec propagation delay. This case is referred to as *0s10u* case. The second topology was used in previous simulations and it was shown in Figure 4.1 where two of 10 traffic are shaped and the other 8 are not shaped. This case is referred to as *2s8u* case. Third topology used is the topology shown in Figure 3.2 where 5 of 10 traffic are shaped and the remaining 5 are not shaped. This case is referred to as *5s5u* case. The fourth and the last topology used is shown in Figure 4.35 where all of 10 traffic are shaped. In this configuration each one of simplex links $(s_i, DBRAS_i)$ and $(DBRAS_i, s_i)$ for all i , $0 \leq i \leq 9$ has 10 Mbps rate, 5 msec propagation delay, and a Drop-Tail type queue. Each one of simplex links $(DBRAS_i, e1)$ for all i , $0 \leq i \leq 9$ has 10 Mbps rate, 0 msec propagation delay, and a queue of

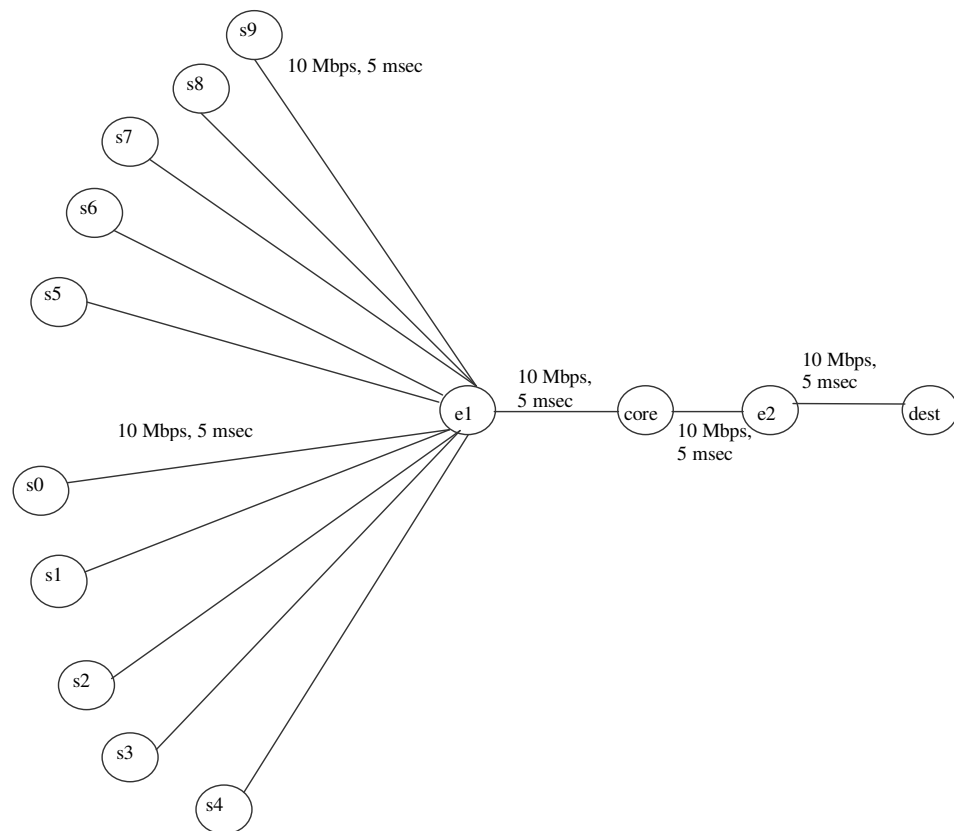


Figure 4.34: Network topology of 0s10u case.

DBRASQueue type. Each one of links $(e1, \text{DBRAS}_i)$ for all $i, 0 \leq i \leq 9$ has 10 Mbps rate, 0 msec propagation delay, and a queue of Drop-Tail type. This case is referred to as *10s0u* case. Every DBRAS is configured with the same parameter values as described previously.

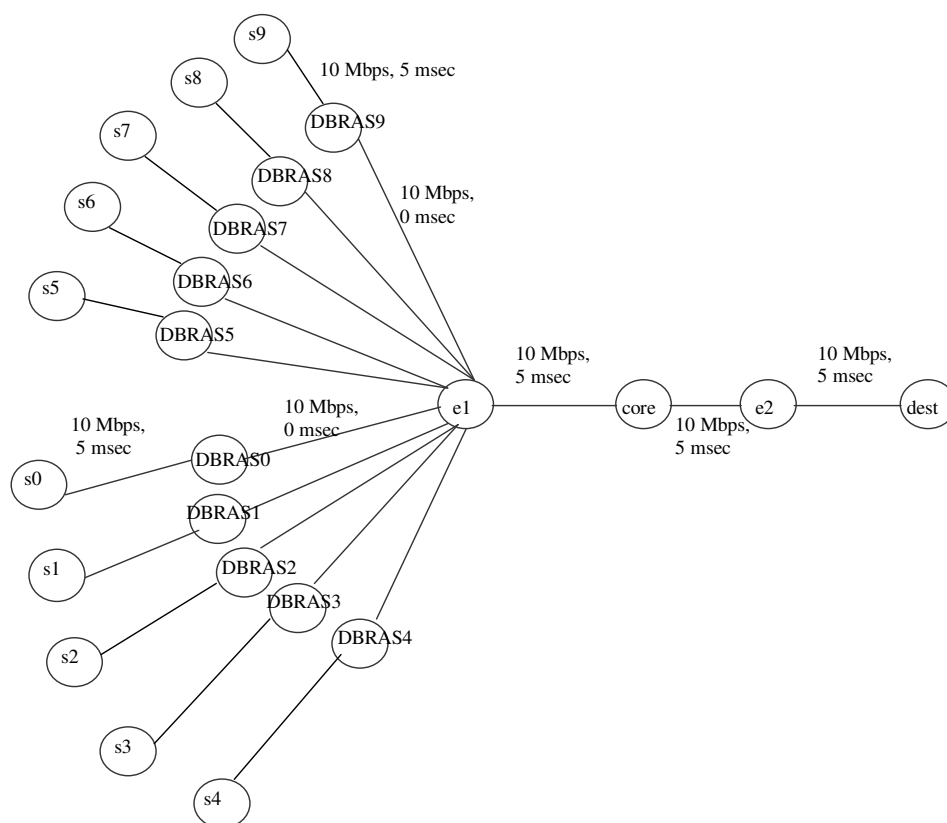


Figure 4.35: Network topology of 10s0u case.

Simulations are performed for two P_D cases and optimum initial D_{max} values of each one. Average levels of throughput achieved by shaped (s) and unshaped (u) traffic (in Kbps) and also corresponding green ratios are shown in Tables 4.5 and 4.6, respectively.

The “total” column in Table 4.5 refers to the sum of individual traffic throughput values. For both of the P_D cases, as the ratio of shaped traffic increases (from 2s8u to 10s0u case), the average throughput level achieved by shaped sources decreases. This is because of the fact that as the number of shaped traffic in the network increases, the relative advantages for shaped traffic in SP network queues is shared among a greater number of traffic flows. Nevertheless, the average throughput of shaped traffic in 10s0u case is better than the average throughput of unshaped traffic in 0s10u case. Since shaping increases utilization of network resources through higher queue load and less drops at congested

Topology used	P_D (msec)	Initial D_{max} (msec)					
		50			120		
		s	u	total	s	u	total
0s10u	5	-	687.23	6872.3	-	687.23	6872.3
2s8u	5	971.41	672.42	7322.18	1042.83	634.14	7158.78
5s5u	5	928.21	693.16	8106.85	1041.48	623.65	8325.65
10s0u	5	903.80	-	9038.0	997.66	-	9976.6
		20			90		
0s10u	20	-	791.17	7911.7	-	791.17	7911.7
2s8u	20	1004.22	697.99	7592.36	1098.65	699.39	7792.42
5s5u	20	952.77	594.32	7735.45	1084.46	652.23	8683.45
10s0u	20	839.30	-	8393.0	992.72	-	9927.2

Table 4.5: Throughput levels achieved by different ratios of shaped traffic in the network (in Kbps).

link (e1, core), the total throughput in the network increases as ratio of shaped sources increases from 0 (0s10u case) to 100 percent (10s0u case).

It is noted in Table 4.6 that as the ratio of shaped traffic in the network increases, the green ratio of unshaped traffic increases. This can be explained in the following way: As the number of shaped sources increases, more green packets arrive at the core router which increases the queue load and thus increases dropping probability for red packets. Since red packets are more likely originating from unshaped sources, packet drops lead to smaller congestion windows (smaller throughput) for unshaped sources resulting in higher ratios of packets marked as green.

In the next part, the effect of RED parameters on performance of D-DBRAS is investigated.

The Effect of RED Parameters

The effects of RED parameters on dropping probabilities in SP network and throughput levels are investigated for the 2s8u case. Simulations are performed

Topology used	P_D (msec)	initial D_{max} (msec)			
		50		120	
		s	u	s	u
0s10u	5	-	0.578	-	0.578
2s8u	5	0.852	0.590	0.938	0.608
5s5u	5	0.883	0.606	0.943	0.639
10s0u	5	0.869	-	0.994	-
		0.020		0.090	
0s10u	20	-	0.506	-	0.506
2s8u	20	0.682	0.535	0.889	0.530
5s5u	20	0.741	0.581	0.912	0.564
10s0u	20	0.883	-	0.992	-

Table 4.6: Green ratios in topologies with different shaped traffic ratio in the network.

for the two P_D cases with their optimum initial D_{max} values. Two RED parameter value sets are used. The six-tuple ($green_min_{th}$, $green_max_{th}$, $green_max_p$, red_min_{th} , red_max_{th} , red_max_p) is set to (20, 40, 0.02, 10, 20, 0.1) which was used in previous simulations and is called `old_red_parameters` (`o_r_p`) and the tuple (20, 40, 0.02, 20, 30, 0.1) which is called `new_red_parameters` (`n_r_p`). The parameter values in `n_r_p` are chosen so that RED is more tolerant on red packets. This way, advantages of shaping are less pronounced. Our focus in this study is to see if there exists a case where shaping does not provide any advantages (or even provides disadvantages) when RED parameters are not appropriately selected. Results for throughput measurements are shown in Table 4.7 and results for measurements of dropping probabilities at congested queue are shown in Table 4.8.

In Table 4.7, there are some combinations of parameters where throughput of unshaped traffic is higher than throughput of shaped traffic, namely cases (`n_r_p`, 5, 120), (`n_r_p`, 20, 20), and (`n_r_p`, 20, 90). These cases show that the optimum initial D_{max} value for a particular P_D is affected by RED parameters used in the SP network. It can be concluded that in those cases, D-DBRAS does not provide any benefits. The reason that unshaped traffic achieves higher throughput compared to shaped traffic can be explained by using the observation

RED param. used	P_D (msec)	initial D_{max} (sec)	Throughput (Kbps)			
			s0	s1	u	total
o.r.p	5	50	972.59	970.23	672.42	7322.18
n.r.p	5	50	1036.61	992.23	944.52	9585.00
o.r.p	5	120	1040.28	1045.38	634.14	7158.78
n.r.p	5	120	955.50	966.31	960.93	9609.25
o.r.p	20	20	1041.40	967.04	697.99	7592.36
n.r.p	20	20	864.97	845.88	912.12	9007.81
o.r.p	20	90	1096.20	1101.10	699.39	7792.42
n.r.p	20	90	740.26	738.33	918.73	8828.43

Table 4.7: Throughput levels achieved with different RED parameter values (in Kbps).

from Table 4.8: In those cases where shaping is not fruitful, hard drops of green packets for shaped traffic are more likely compared to unshaped traffic. This is mainly due to increased buffer levels and less disadvantageous handling of red packets in the core network. If RED parameters are not adjusted so that red packets are not properly penalized, shaping does not introduce any increase in throughput. These results show the impact of decision of SP on the level of misbehavior to apply to out-of-profile portion of customer traffic on the service it can supply to in-profile portion of customer traffic.

RED param. used	P_D (msec)	initial D_{max} (msec)	green early drop prob.		
			s0	s1	u
o_r_p	5	50	0.0	0.0	0.0
n_r_p	5	50	0.0	0.0	0.0
o_r_p	5	120	0.0	0.0	0.0
n_r_p	5	120	0.0	0.4e-5	0.2e-5
o_r_p	20	20	0.0	0.0	0.0
n_r_p	20	20	0.0	0.0	0.0
o_r_p	20	90	0.0	0.0	0.0
n_r_p	20	90	0.0	0.0	0.0
			green hard drop prob.		
o_r_p	5	50	1.3e-3	1.2e-3	1.0e-3
n_r_p	5	50	4.4e-3	4.6e-3	3.0e-3
o_r_p	5	120	1.3e-3	1.2e-3	1.0e-3
n_r_p	5	120	4.7e-3	4.5e-3	2.8e-3
o_r_p	20	20	2.1e-5	1.5e-5	0.3e-5
n_r_p	20	20	1.1e-2	1.1e-2	1.6e-3
o_r_p	20	90	0.2e-5	0.7e-5	0.0
n_r_p	20	90	1.5e-2	1.6e-2	1.8e-3
			red early drop prob.		
o_r_p	5	50	3.1e-2	3.0e-2	3.1e-2
n_r_p	5	50	3.7e-2	4.0e-2	3.9e-2
o_r_p	5	120	2.2e-2	2.3e-2	3.0e-2
n_r_p	5	120	3.4e-2	3.0e-2	3.9e-2
o_r_p	20	20	2.1e-2	2.1e-2	2.5e-2
n_r_p	20	20	1.8e-2	1.8e-2	2.3e-2
o_r_p	20	90	1.7e-2	1.6e-2	2.4e-2
n_r_p	20	90	5.8e-3	6.2e-3	2.1e-2
			red hard drop prob.		
o_r_p	5	50	9.2e-2	9.4e-2	1.3e-1
n_r_p	5	50	3.1e-2	3.1e-2	3.8e-2
o_r_p	5	120	6.2e-3	5.3e-3	1.4e-1
n_r_p	5	120	2.2e-2	1.5e-2	3.6e-2
o_r_p	20	20	4.1e-2	4.6e-2	7.1e-2
n_r_p	20	20	2.4e-2	2.4e-2	1.9e-2
o_r_p	20	90	6.3e-3	5.9e-3	7.5e-2
n_r_p	20	90	4.1e-3	6.9e-3	2.3e-2

Table 4.8: Dropping probabilities at congested queue of networks with different RED parameter values.

In the next section, simulations carried out for testing performance of D-DBRAS in networks with dynamic network load are presented.

4.4.3 Dynamic Network Load Simulations

In dynamic load networks, the load in the network changes dynamically in time, which is more similar to Internet traffic. For a good performance in dynamic-load networks, the a_p parameter value should be as small as possible, while achieving acceptably good throughput levels. In this part, firstly, simulations performed to reach an a_p value smaller than 200 sec are presented. Then scenarios of dynamic network load are introduced and results obtained are presented.

Search for a Smaller a_p value

The criterion used in determination of proper a_p value is convergence of D_{max} value to one of locally optimum D_{max} values within the first 1000 sec of simulation. We use four different values of a_p , 200, 100, 50, and 25 sec. Behaviors of D_{max} and throughput of s_0 in time are observed for $P_D = 20$ msec. D_{max} vs. time and throughput of s_0 vs. time plots for a_p value of 200 sec are shown in Figures 4.36 and 4.37, plots for a_p value of 100 sec are shown in Figures 4.38 and 4.39, plots for a_p value of 50 sec are shown in Figures 4.40 and 4.41, and plots for a_p value of 25 sec are shown in Figures 4.42 and 4.43, respectively. From these figures, it is concluded that for $a_p = 100$ sec proper convergence of D_{max} for different initial conditions occurs. The results for $a_p = 50$ sec and 25 sec are affected from fluctuations in the traffic.

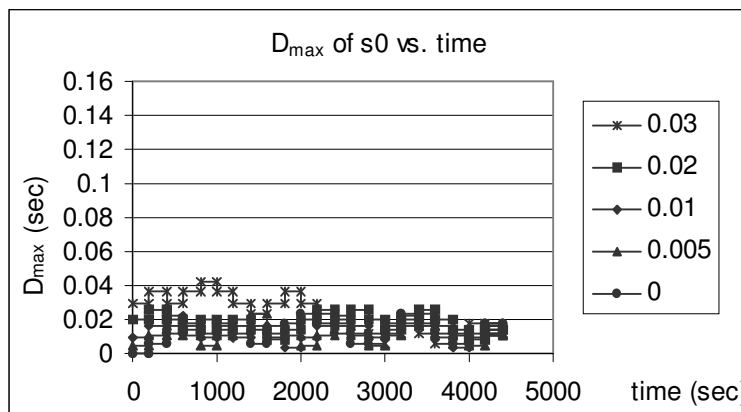
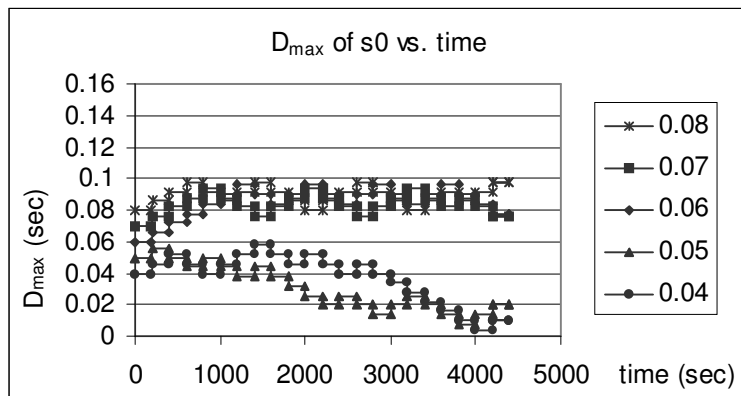
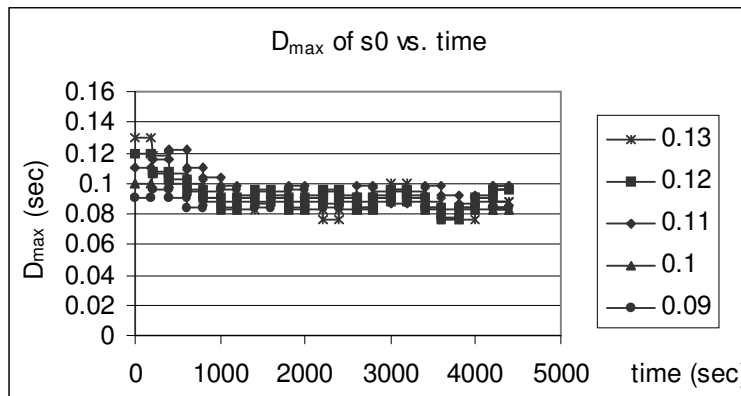
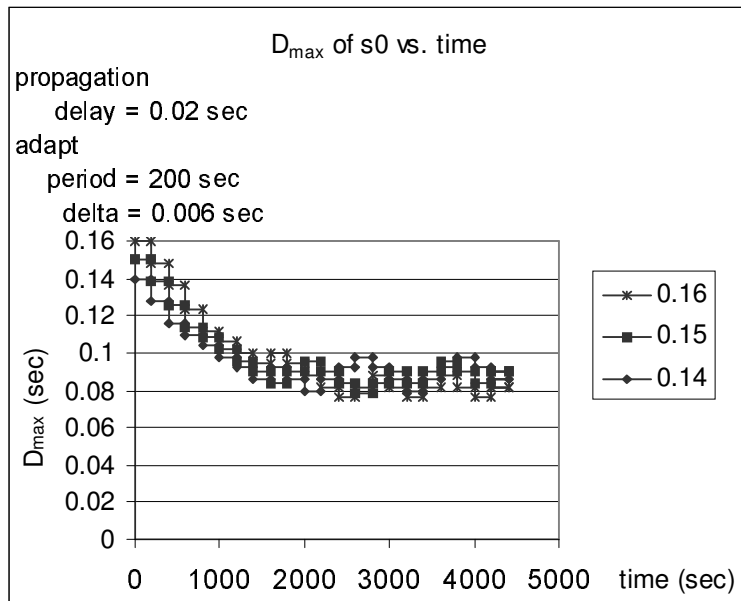


Figure 4.36: D_{max} of s_0 vs. time with different initial D_{max} values, $\text{ap} = 200$ sec.

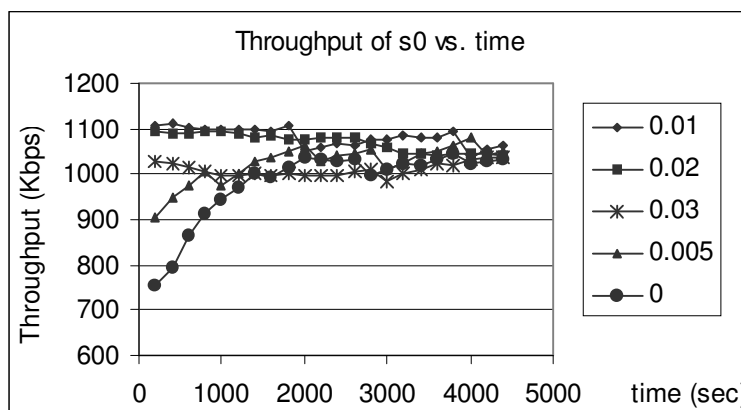
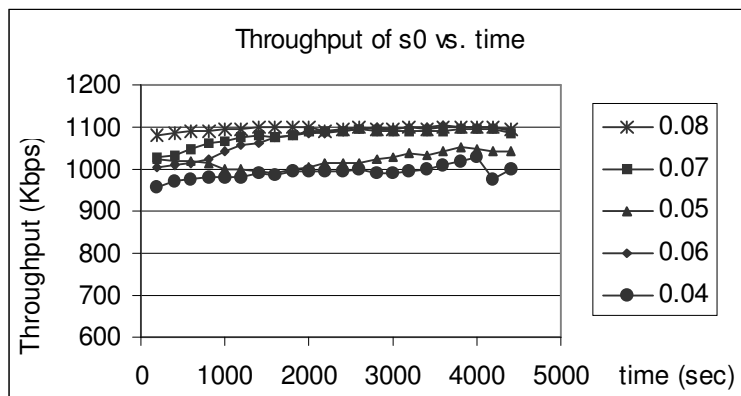
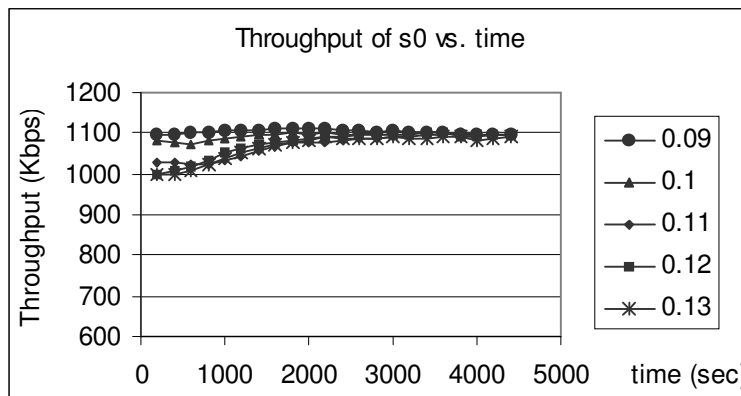
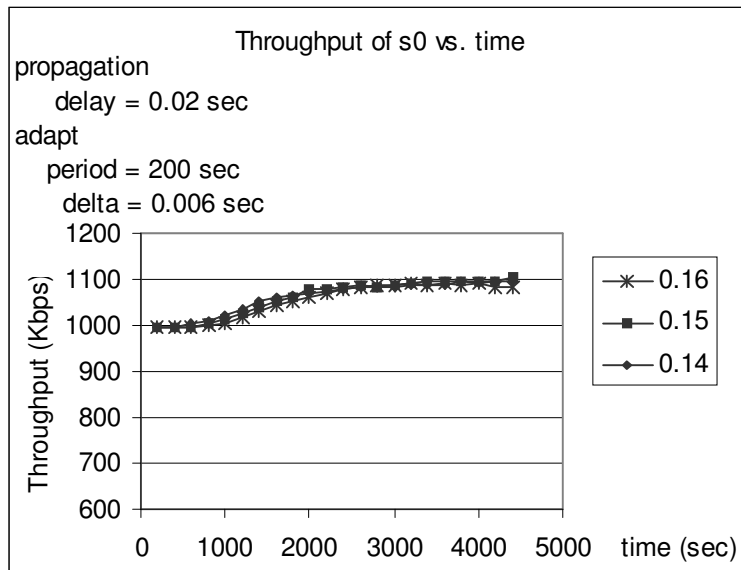


Figure 4.37: Throughput of s0 vs. time with different initial D_{max} values, ap = 200 sec.

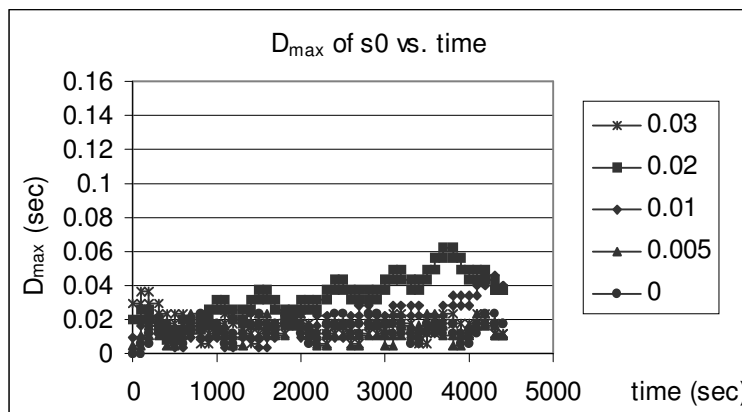
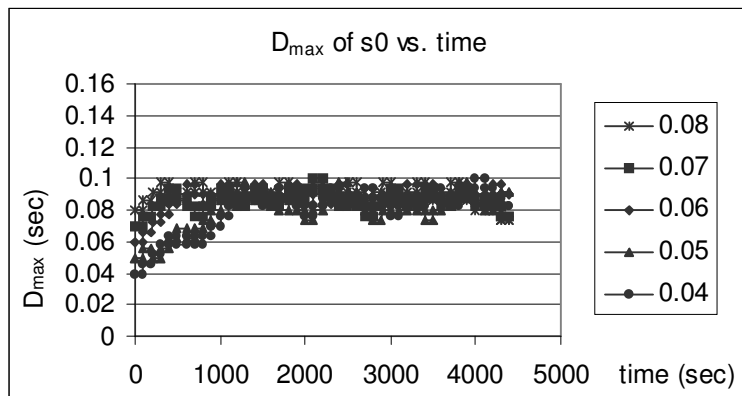
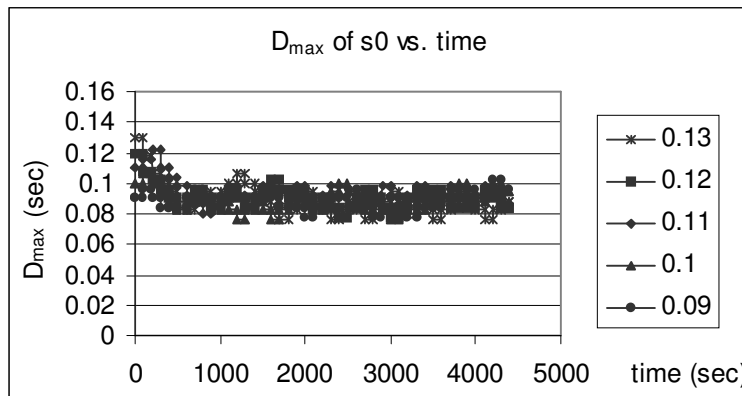
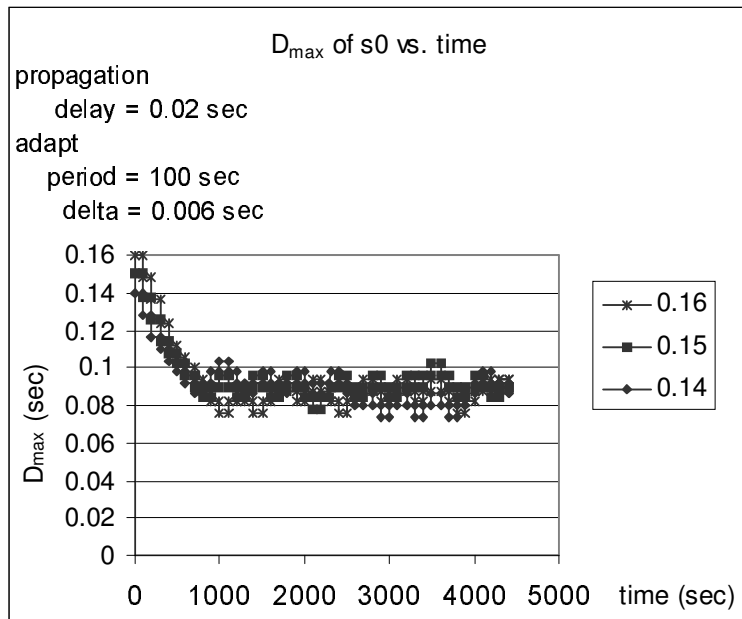


Figure 4.38: D_{max} of s0 vs. time with different initial D_{max} values, ap = 100 sec.

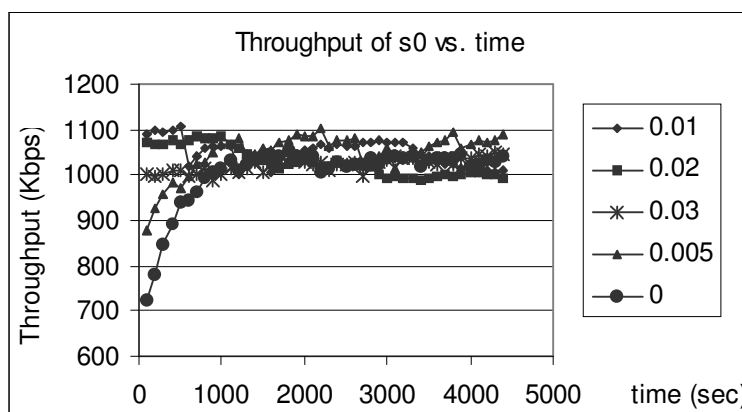
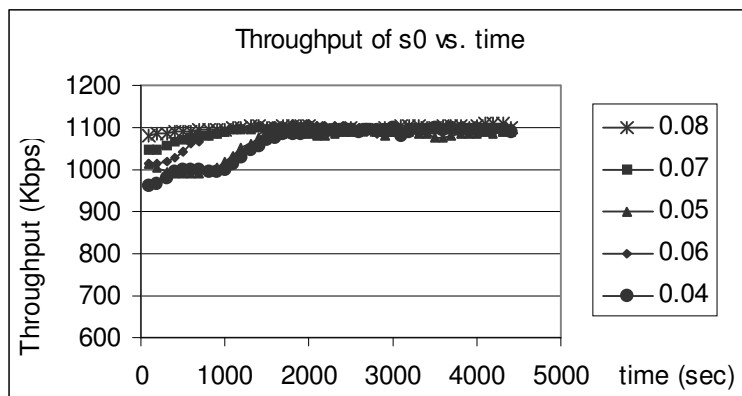
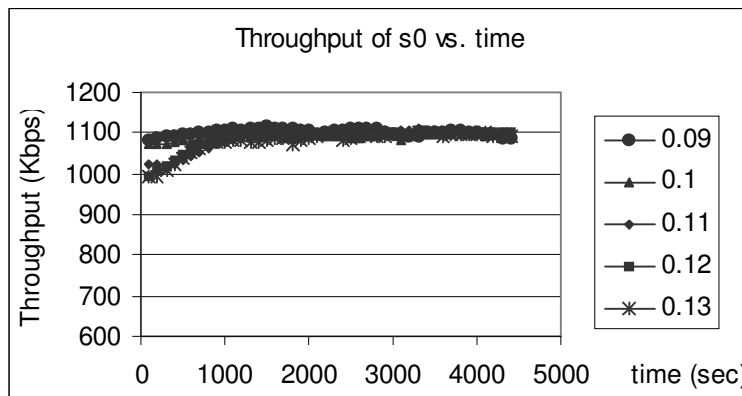
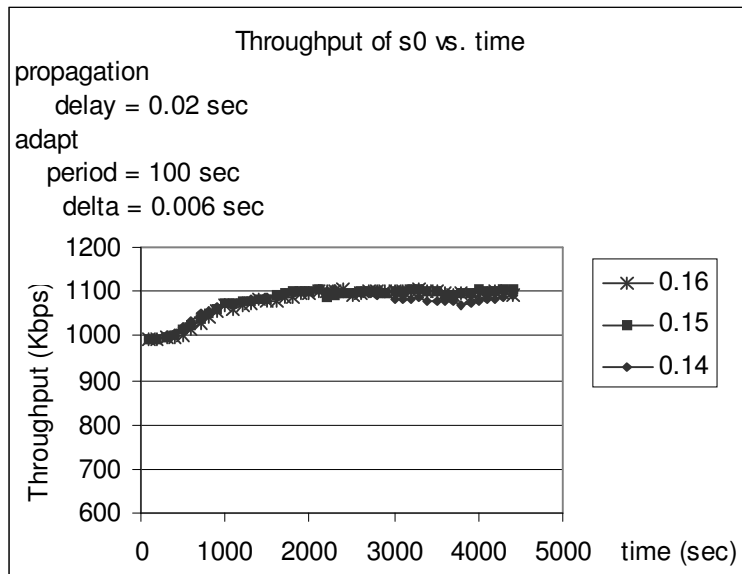


Figure 4.39: Throughput of s0 vs. time with different initial D_{max} values, ap = 100 sec.

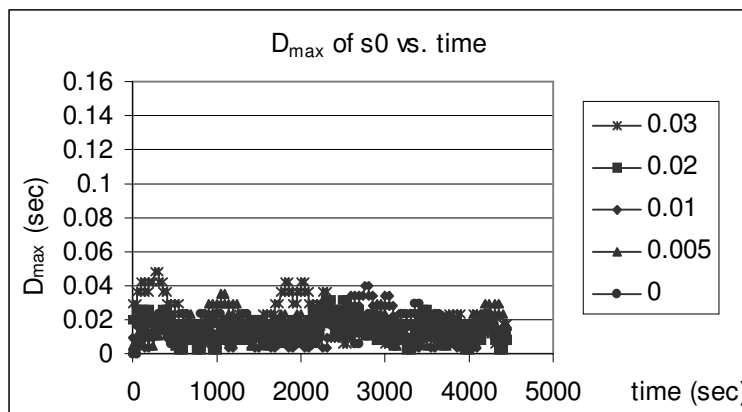
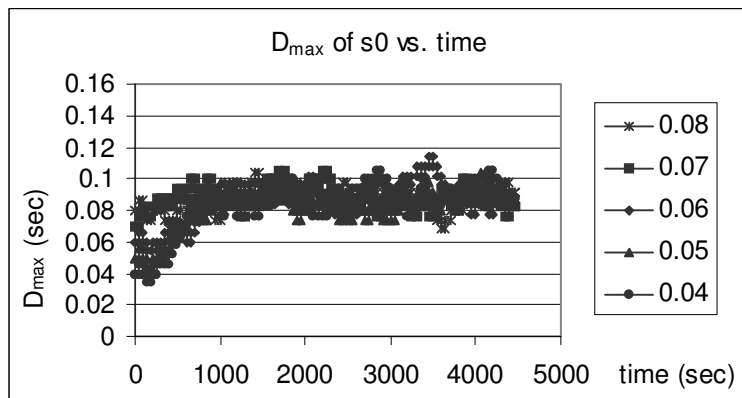
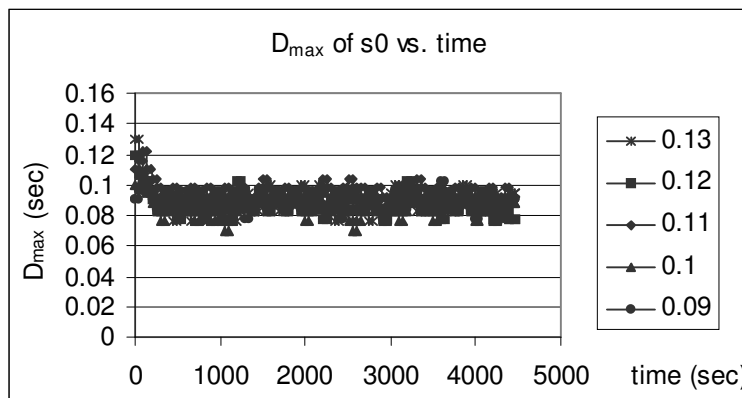
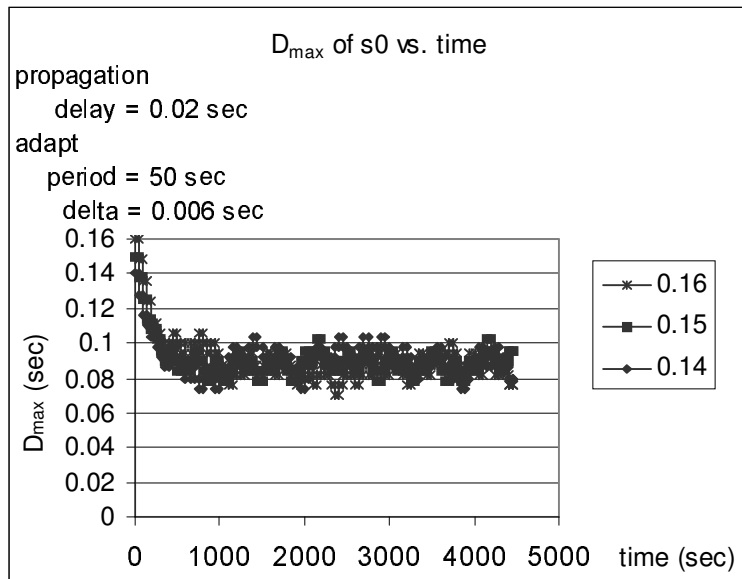


Figure 4.40: D_{max} of s_0 vs. time with different initial D_{max} values, $ap = 50$ sec.

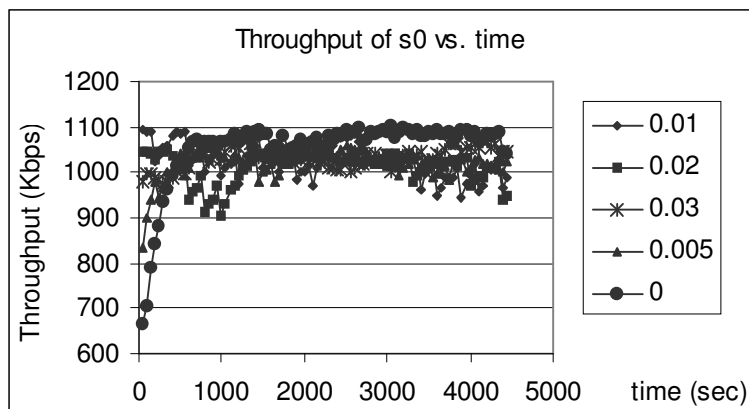
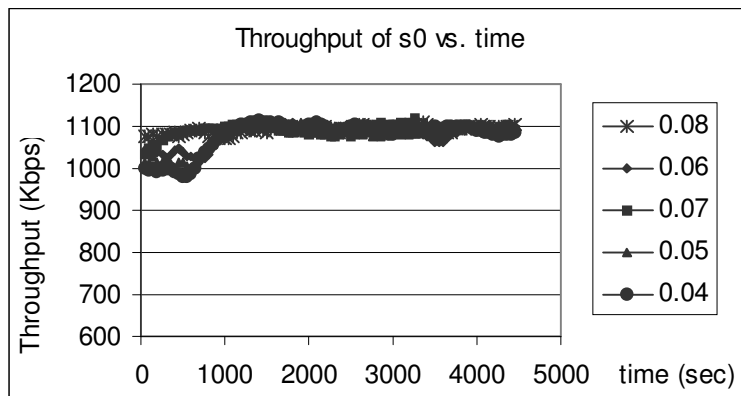
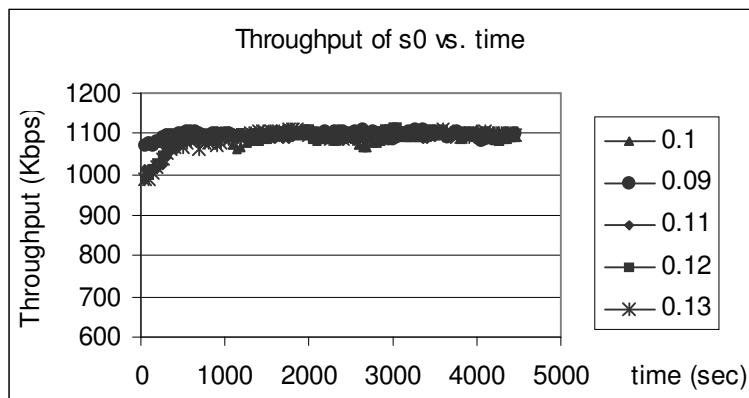
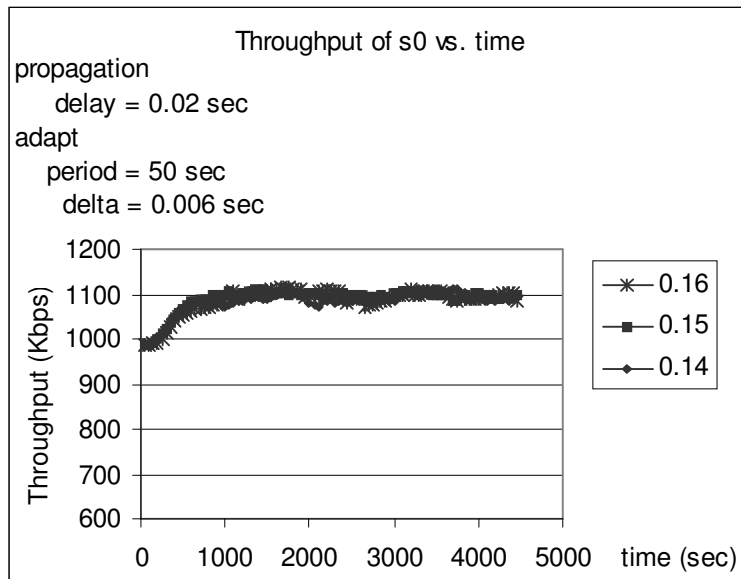


Figure 4.41: Throughput of s0 vs. time with different initial D_{max} values, ap = 50 sec.

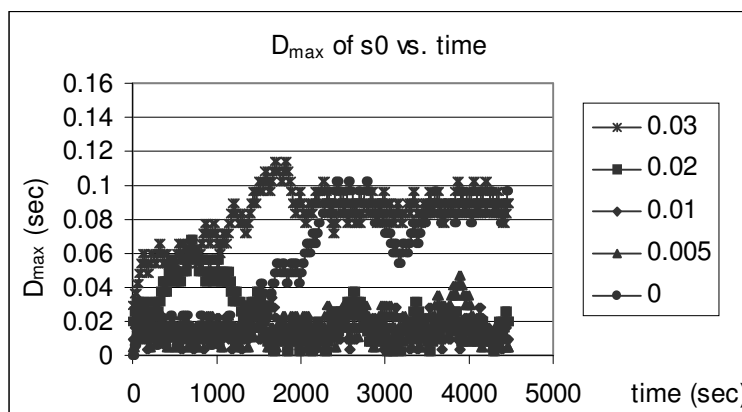
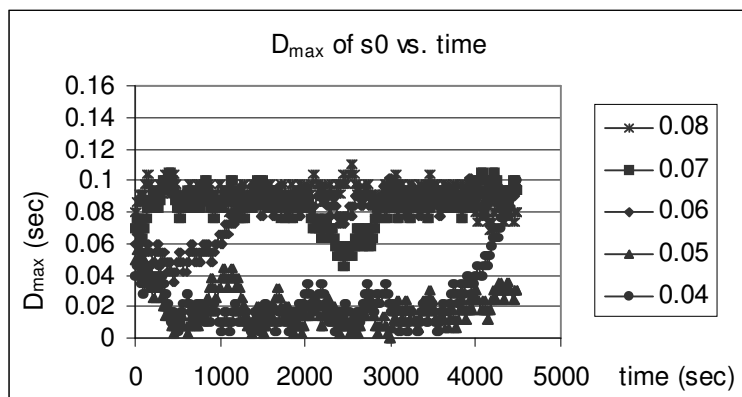
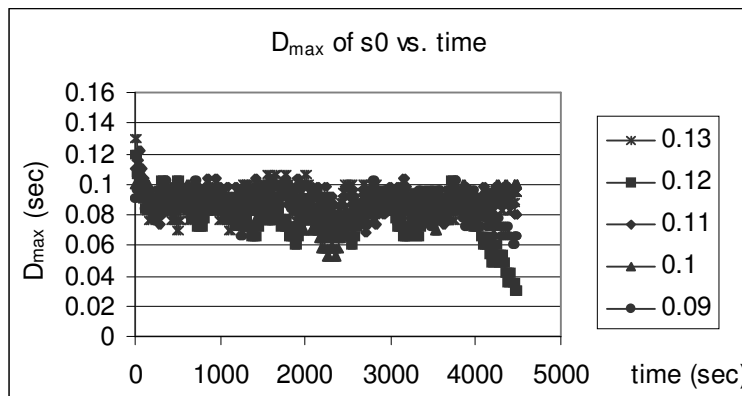
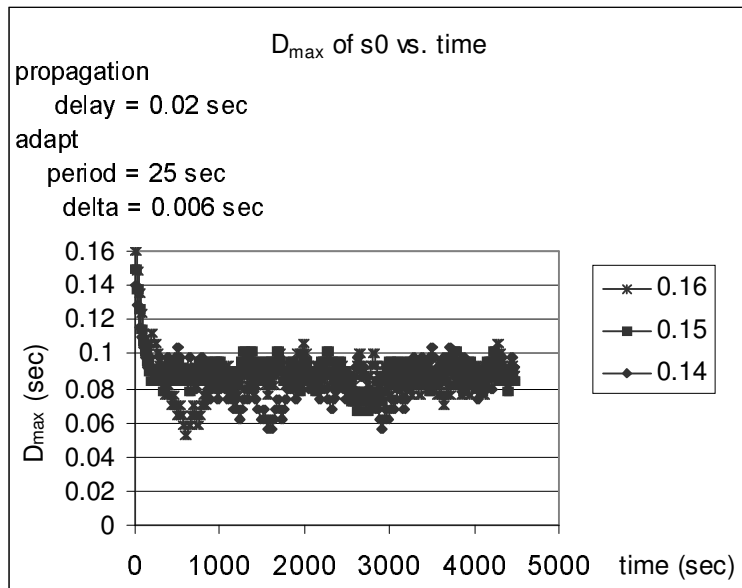


Figure 4.42: D_{max} of s0 vs. time with different initial D_{max} values, ap = 25 sec.

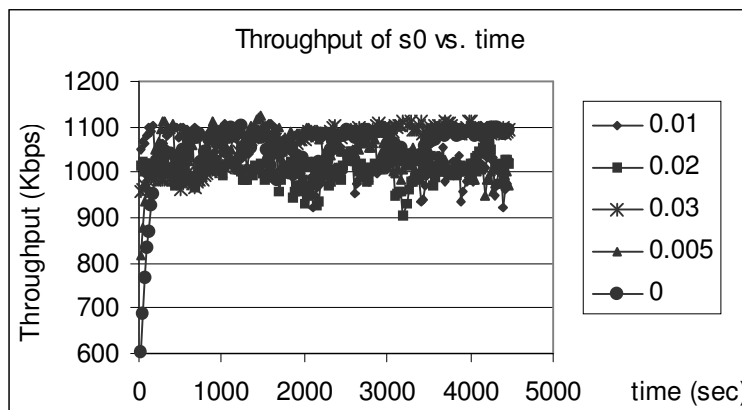
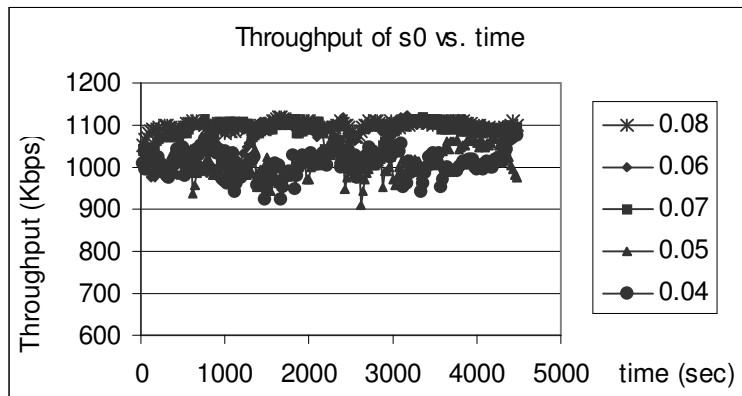
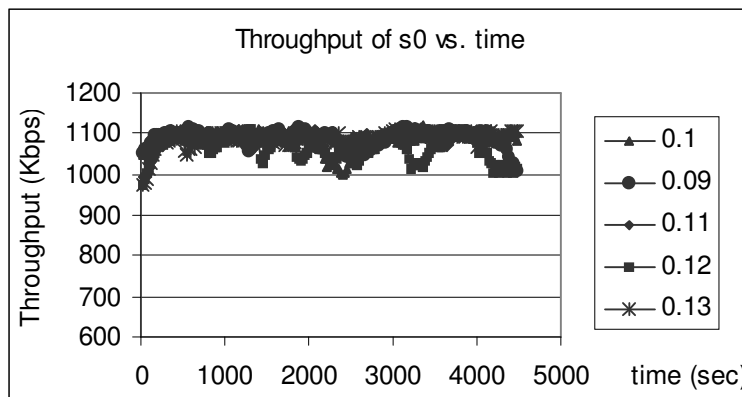
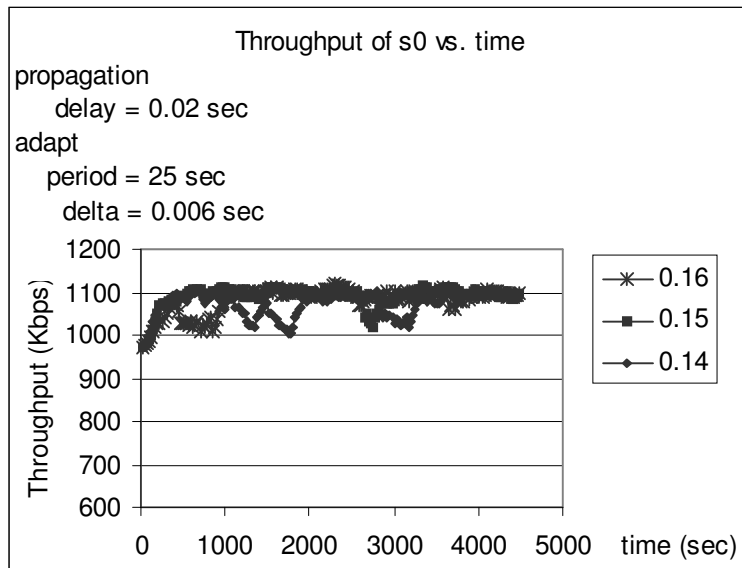


Figure 4.43: Throughput of s0 vs. time with different initial D_{max} values, ap = 25 sec.

Scenario No.	first 2000 sec	remaining 2400 sec
Sc. 1	2s4u	2s8u
Sc. 2	3s3u	5s5u
Sc. 3	5s0u	7s0u

Table 4.9: Dynamic network load scenarios.

Dynamic Network Load Scenarios

Three scenarios are set up using three topologies; namely 2s8u, 5s5u, and 10s0u cases described previously. Load variation during simulation is achieved by starting and terminating various shaped and unshaped TCP connections. The simulation duration is kept as 4400 sec and a load variation is done at time = 2000.0 sec. The scenarios are shown in Table 4.9 specifying the number of active shaped (s) and unshaped (u) traffic sources within an interval. One traffic, namely traffic from node s0, is kept active during the complete simulation. Other traffic sources are either active or inactive during the run of the simulation as specified in Table 4.9. $P_D = 20$ msec is used and initial D_{max} applied ranges from 0 msec to 100 msec for all D-DBRAS in the topology. D_{max} of D-DBRAS of s0 and throughput of s0 are observed in time. Results obtained are presented below.

Dynamic Network Load Simulation Results

We use three different scenarios for dynamic traffic. Graphs obtained for Sc. 1 are shown in Figures 4.44 and 4.45. In the first 2000 sec interval, network resources are enough for those six traffic (mean traffic rates of those six sum up to 9 Mbps). For this reason, in Figure 4.44, we see convergence to smaller D_{max} values, one around 50 msec and another one around 10 msec and a throughput close to 1.5 Mbps for small D_{max} values. In the remaining 2400 seconds, 2s8u case is used. After 2000 sec, the throughput for s0 converges to about 1 Mbps which is expected from our previous simulations.

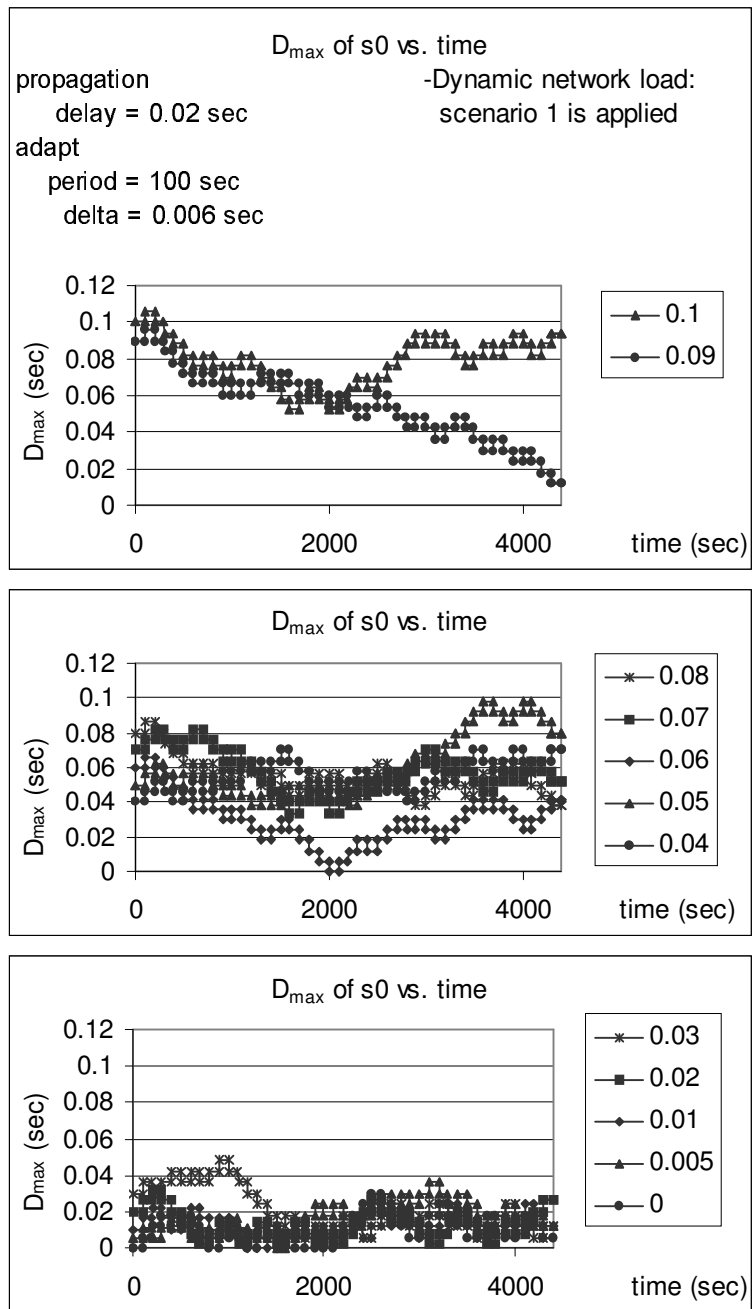


Figure 4.44: D_{max} of D-DBRAS of s0 vs. time for dynamic network load Sce. 1.

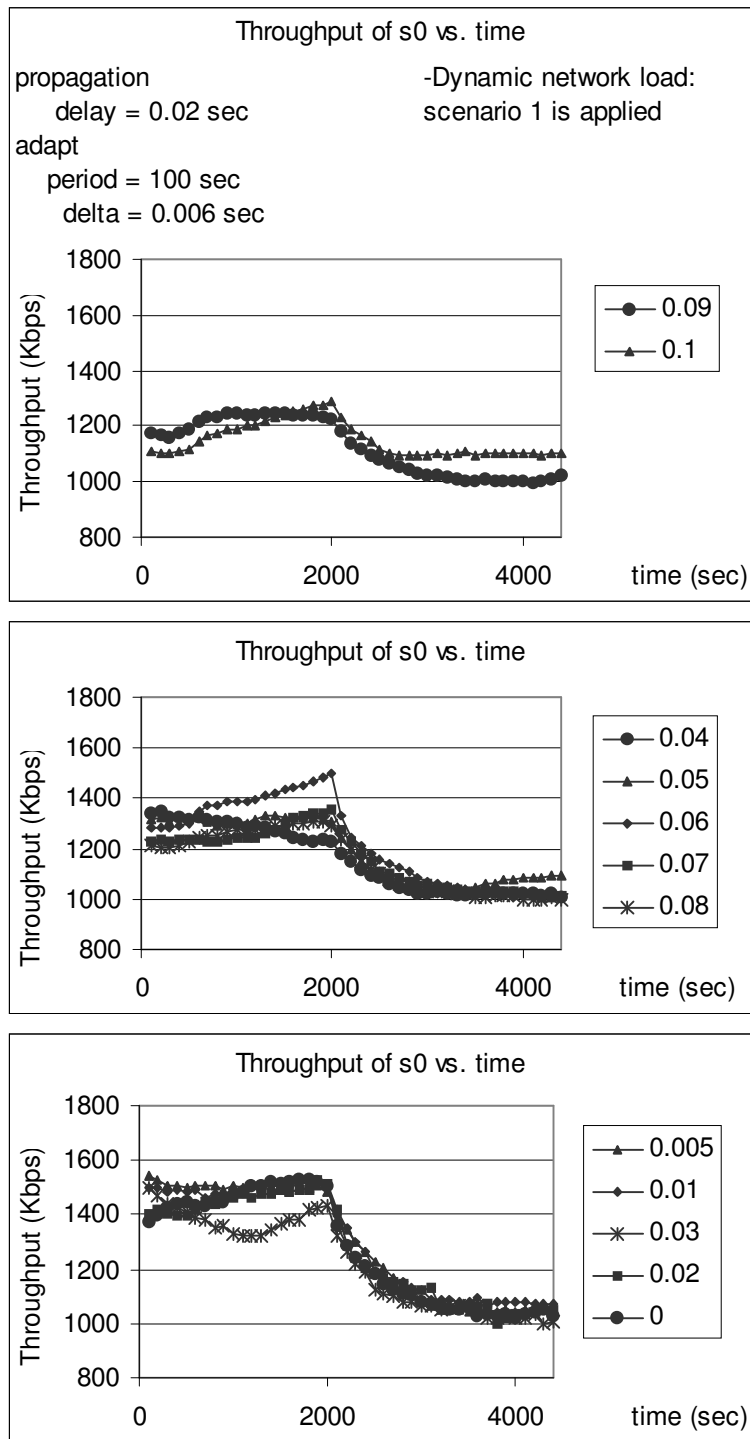


Figure 4.45: Throughput of s0 vs. time for dynamic network load Sce. 1.

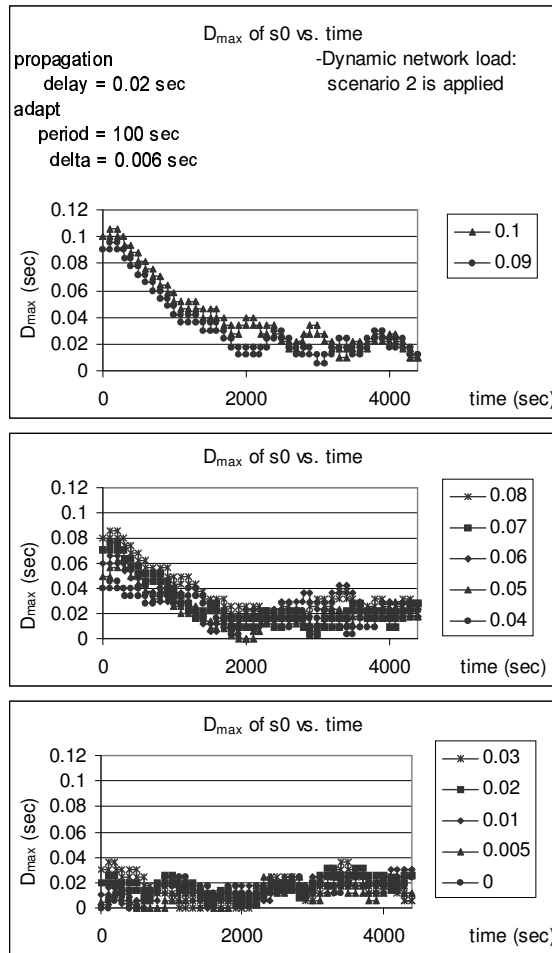


Figure 4.46: D_{max} of D-DBRAS of s0 vs. time for dynamic network load Sce. 2.

Graphs obtained for Sce. 2 are shown in Figures 4.46 and 4.47. Behaviors are similar to graphs for Sce. 1. In the first 2000 sec, the throughput achieved by s0 is about 1.5 Mbps. In remaining 2400 sec, the throughput converges to a value which is slightly lower than 1 Mbps expected from Table 4.5.

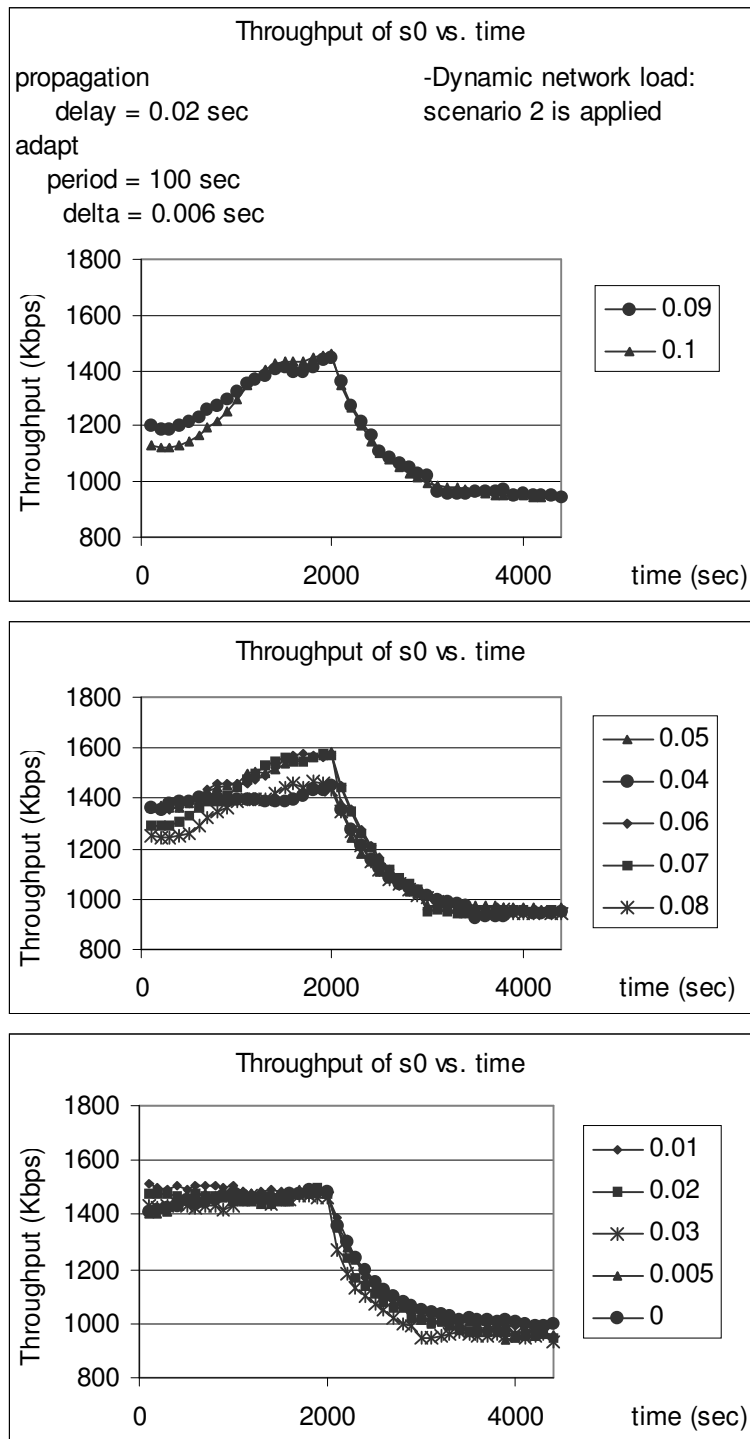


Figure 4.47: Throughput of s0 vs. time for dynamic network load Sce. 2.

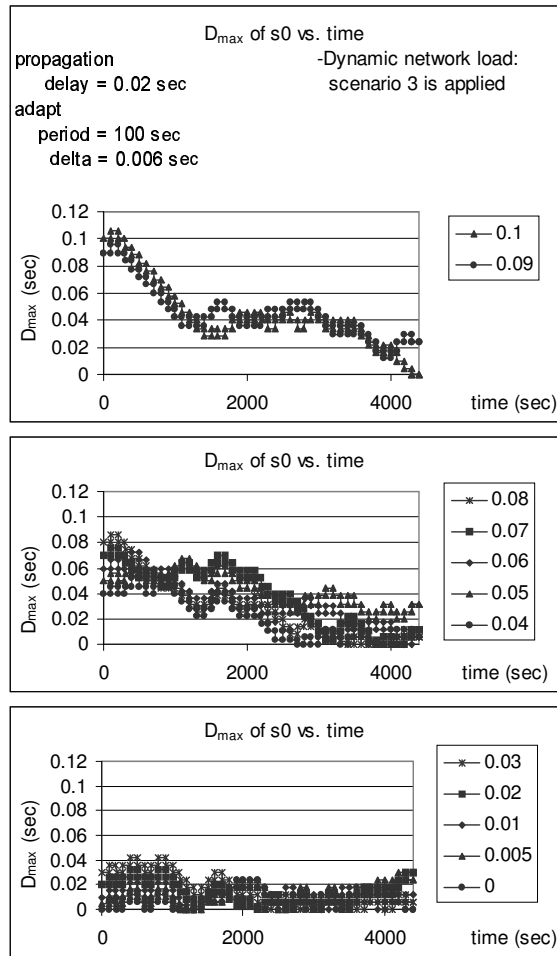


Figure 4.48: D_{max} of D-DBRAS of s0 vs. time for dynamic network load Sce. 3.

Similar reasoning can be done for results of Sce. 3, which are shown in Figures 4.48 and 4.49. In Figure 4.49, the first 2000 sec interval behavior is thought to be because of the over-provisioned network and increase in network resource utilization due to higher number of shaped traffic. In the remaining 2400 sec interval, there is less congestion than both of the previous scenarios, which is the reason of increased throughput compared to results of previous two scenarios.

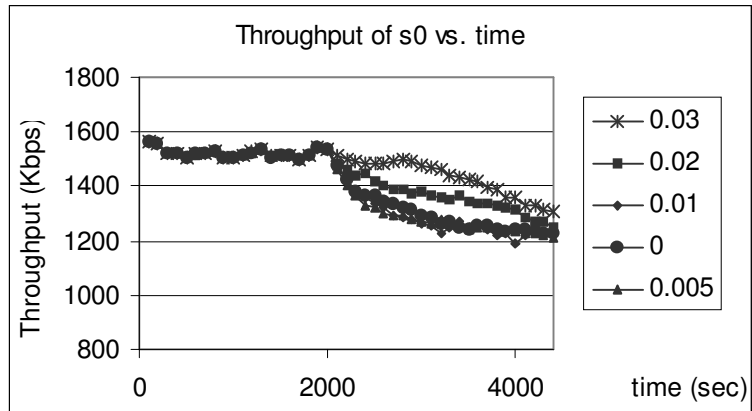
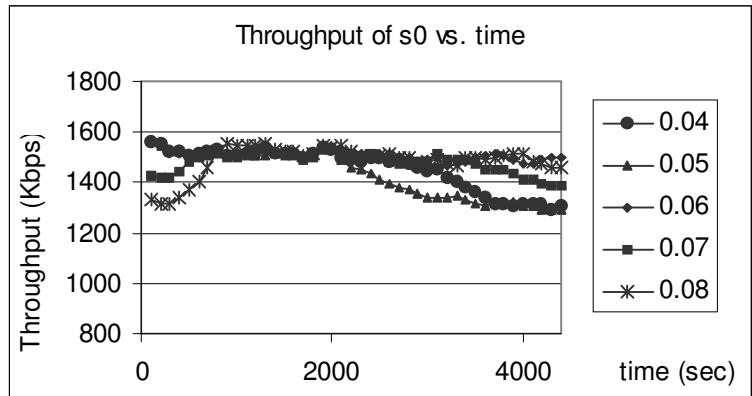
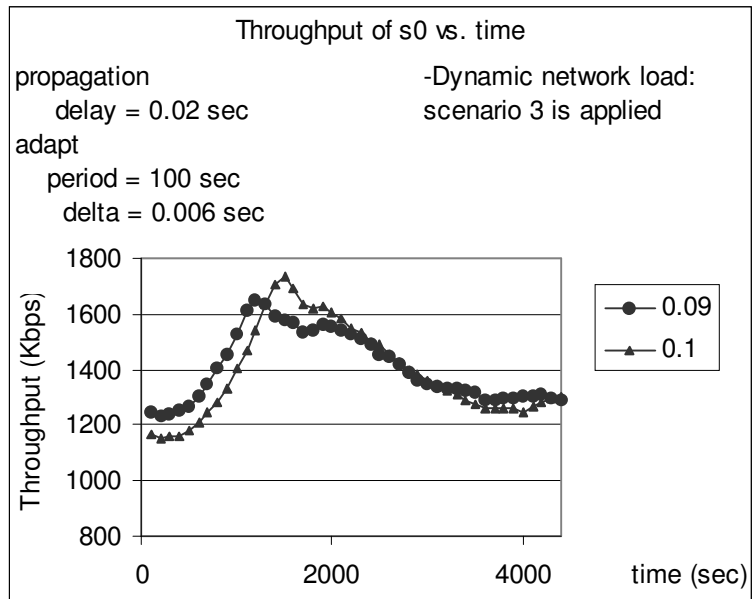


Figure 4.49: Throughput of s0 vs. time for dynamic network load Sce. 3.

Chapter 5

Conclusions

DiffServ model keeps the processing in the network core simple, by moving the processing to edges of the network. In the core of the DiffServ network, resources are distributed among a small number of PHBs by SP in the light of QoS it aims to supply. Traffic of customers are classified into PHBs at edges based on their QoS requirements. They sign an SLA for the specification of service and usage criteria. AF PHB supplies a guarantee on the packet loss ratio of traffic as long as throughput and maximum burst size of traffic satisfy values specified in the SLA. SP applies marking to traffic to enforce the AF PHB in the core of the network. It was found out that AF PHB can be unable to supply the subscribed rate to the traffic based on network load and level of provisioning in the network. Shapers were developed to solve those limitations by increasing the in-profile portion of traffic through delaying out-of-profile packets. Buffer size in the shaper should be chosen according to the maximum shaping delay to be applied to packets. Moreover, in cases where SP is generous in routing out-of-profile packets in its network up to some extent, shaping can lower throughput due to additional delay.

Currently existing shapers do not place any restriction on the maximum shaping delay. DBRAS tries to solve these problems by putting a fixed upper bound,

D_{max} , on the shaping delay. It applies a delay to an incoming packet, only if it is out-of-profile and can be marked by downstream TBM as in-profile by delaying its arrival to TBM by an amount less than or equal to D_{max} . Simulations are done in a congested network where average throughputs of equal number of shaped and unshaped TCP Reno connections with the same bursty traffic behavior are compared. During simulations, effects of differences in values of some parameters that are thought to influence the performance of DBRAS are investigated. Different D_{max} values result in a peaky throughput vs. D_{max} curve, where average throughput of shaped sources takes values in the range 98% to 175% relative to average throughput of unshaped sources. The reason of this fluctuation is the fact that DBRAS decreases dropping probability at the expense of increasing end-to-end delay which is indirectly proportional to throughput of TCP. Moreover, shaping the traffic to be all in-profile does not give the best achievable throughput. With smaller P_D , the D_{max} value which gives the best average throughput for shaped traffic changes. Furthermore, the average throughput of shaped traffic can be as low as 80% of average throughput of unshaped traffic. As CBS increases, the gain of shaping decreases, which results from the natural increase in the in-profile portion of traffic that in turn reduces the advantage of shaping.

The extension of DBRAS to D-DBRAS is proposed to decrease the influence of those parameters on the performance of DBRAS. D-DBRAS is built on the assumptions that D-DBRAS shapes one TCP connection and statistics of sender TCP are available to D-DBRAS. D-DBRAS uses a dynamic D_{max} value. It periodically modifies its D_{max} value by a constant step size, δ , in consecutive periods of time during its activity, in the way to increase throughput which is calculated using ACK numbers received by the sender TCP. The selection of δ directly affects convergence of D_{max} . The performance of D-DBRAS is analyzed with respect to parameters including those used for DBRAS and some other parameters. Using D-DBRAS, as D_{max} varies, throughput of traffic shaped

by D-DBRAS is increased 38-58% relative to average throughput of unshaped traffic. When P_D is decreased, the advantage of shaping in increasing throughput becomes relatively smaller.

If the level of congestion in the network is decreased, the throughput vs. initial D_{max} curve for shaped traffic takes value in the range of 136-142% relative to the average of unshaped traffic in the network. This small decrease in shaping gain results from increased tolerance to out-of-profile packets compared to the initial form of the network.

As the ratio of shaped traffic in the network increases, the throughput of shaped traffic decreases. But the average throughput in all-shaped traffic case is 4-45% larger than that of all-unshaped traffic case. This shows that shaping is socially profitable, however, the individual profit decreases as the ratio of shaped traffic increases.

The change in RED parameters can lead to configurations where average throughput of unshaped traffic is as much as 24% larger than that of shaped traffic. So, a SP that handles out-of-profile packets liberally cannot take the full advantage of shaping in increasing throughput. It is also shown through simulations that D-DBRAS performs well in dynamic network load scenarios.

As a possible future research topic, the adjustment algorithm can be modified so that local maxima for throughput are avoided and globally optimum solution is attained. Improving D-DBRAS so that it converges for short-duration TCP connections is another possible research path. Furthermore, extending D-DBRAS for shaping aggregate of TCP connections is required. Moreover, the calibration of D_{max} can take into consideration RED parameters used in the network. This way, avoidance of those bad configurations which lead to performance degradation can be sought.

Bibliography

- [1] D. Black, S. Blake, M. Carlson, E. Davies, Z. Wang, and W. Weiss, “An Architecture for Differentiated Services,” IETF RFC 2475, December 1998.
- [2] R. Egan, P. Georgatsos, L. Georgiadis, D. Goderis, D. Griffin, C. Jacquenet, G. Memenios, G. Pavlou, Y. T’joens, and C. Zaccone, “Service Level Specification Semantics, Parameters and Negotiation Requirements,” IETF Draft, July 2000.
- [3] F. Baumgartner, T. Braun, and C. Siebel, “Fairness of Assured Service,” Proc. of the European Simulation Multiconference - ESM’99, June 1999.
- [4] V. Jacobson, K. Nichols, K. Poduri, “An Expedited Forwarding PHB,” IETF RFC 2598, June 1999.
- [5] F. Baker, J. Heinanen, W. Weiss, and J. Wroclawski, “Assured Forwarding PHB Group,” IETF RFC 2597, June 1999.
- [6] J. Ibanez, and K. Nichols, “Preliminary Simulation Evaluation of an Assured Service,” IETF Draft, August 1998.
- [7] B. Braden, D. Clark, J. Crowcroft, B. Davie, S. Deering, D. Estrin, S. Floyd, V. Jacobson, G. Minshall, C. Partridge, L. Peterson, K. Ramakrishnan, S. Shenker, J. Wroclawski, and L. Zhang, “Recommendations on queue management and congestion avoidance,” IETF RFC 2309, April 1998.

- [8] J. Aweya, M. Ouellette, D. Y. Montuno, O. W. W. Yang, and C. Zhu, "A Comparison of Active Queue Management Algorithms Using the OPNET Modeler," *IEEE Communications Magazine*, vol. 40, no. 6, pp. 158–167, June 2002.
- [9] J. Bolot, C. Diot, B. Lyles, and M. May, "Reasons not to deploy RED," *Proc. of the IEEE/IFIP International Workshop on Quality of Service - IWQoS'99*, June 1999.
- [10] W. Feng, D. D. Kandlur, D. Saha, and K. G. Shin, "Stochastic Fair Blue: A Queue Management Algorithm for Enforcing Fairness," *INFOCOM 2001*, April 2001.
- [11] M. Baines, J. Ethridge, P. Piedad, and F. Shallwani, "A Network Simulator Differentiated Services Implementation," *OpenIP*, Nortel Networks, July 2000.
- [12] J. Babiarz, I. Lambadaris, R. Makkar, B. Nandy, J. H. Salim, and N. Seddigh, "Empirical Study of Buffer Management Schemes for DiffServ Assured Forwarding PHB," *Nortel Computing Technology Lab (CTL) Technical Report*, May 2000.
- [13] H. Kim, "A Fair Marker," *IETF Draft*, April 1999.
- [14] S. De Cnodder, "Rate Adaptive Shapers for Data Traffic in DiffServ Networks," *NetWorld + Interop 2000 Engineers Conference*, May 2000.
- [15] I. B. H. de A. Alves, L. F. M. de Moraes, and J. F. de Rezende, "Evaluating Fairness in Aggregated Traffic Marking," *IEEE Global Telecommunications Conference - GLOBECOM'00*, November 2000.
- [16] I. Andrikopoulos, G. Pavlou, and L. Wood, "A Fair Traffic Conditioner For the Assured Service in a Differentiated Services Internet," *Proc. of the IEEE International Conference on Communications - ICC'99*, June 1999.

- [17] A. Feroz, S. Kalyanaraman, and A. Rao, "A TCP-Friendly Traffic Marker for IP Differentiated Services," Proc. of the IEEE/IFIP Eighth International Workshop on Quality of Service - IWQoS'2000, June 2000.
- [18] A. L. N. Reddy, and I. Yeom, "Impact of Marking Strategy on Aggregated Flows in a Differentiated Services Network," IWQoS'99, June 1999.
- [19] A. L. N. Reddy, and I. Yeom, "Modeling TCP Behavior in a Differentiated Services Network," IEEE Conference on Network Protocols (ICNP), May 1999.
- [20] C. Diot, V. Firoiu, P. Nain, S. Sahu, and D. Towsley, "On Achievable Service Differentiation with Token Bucket Marking for TCP," ACM SIGMETRICS 2000, August 2000.
- [21] H. Schulzrinne, and X. Wang, "Pricing Network Resources for Adaptive Applications in a Differentiated Services Network," INFOCOM 2001, April 2001.
- [22] O. Bonaventure, and S. De Cnodder, "A Rate Adaptive Shaper for Differentiated Services," IETF RFC 2963, October 2000.
- [23] V. Firoiu, J. Kurose, J. Padhye, and D. Towsley, "Modeling TCP throughput: A Simple Model and its Empirical Validation," ACM SIGCOMM'98, September 1998.
- [24] D. Abendroth, and U. Killat, "Intelligent Shaping: Well Shaped Throughout the Entire Network?," IEEE INFOCOM 2002, June 2002.
- [25] Y. B. Lee, and P. C. Wong, "Design and Performance Evaluation of a Multimedia Web Server," Journal of Visual Communication and Image Representation, vol. 9, no. 3, pp. 183–193. September, 1998.

- [26] M. Hassan, and A. A. Mustafa, "End to End IP Rate Control," 8th International Conference on Advanced Computing and Communications (ADCOM2000), December 2000.
- [27] K. Chandra, and C. You, "Time Series Models for Internet Data Traffic," Proc. of 24th Conf. on Local Computer Networks, LCN-99, October 1999.
- [28] M. F. Alam, M. Atiquzzaman, and M. A. Karim, "Efficient MPEG Video Traffic Shaping for the Next Generation Internet," IEEE Global Telecommunications Conference - Globecom'99, December 1999.
- [29] J. Ethridge, B. Nandy, P. Piedad, and N. Seddigh, "Intelligent Traffic Conditioners for Assured Forwarding Based Differentiated Services Networks," ver. 7, Proc. of High Performance Networking Conference - HPN 2000, May 2000.
- [30] S. McCanne, and S. Floyd, ns-*Network Simulator* (version 2.1b7a). <http://www.isi.edu/nsnam/ns>.