Helsinki University of Technology
Department of Electrical and Communication Engineering

Marko Luoma

Simulation studies of Differentiated Services Networks

Licenciate thesis

Supervisor: Professor Jorma Virtamo, HUT

Tekijä:	Marko Luoma
Työn nimi:	Simulation studies of Differentiated Services Networks
Päivämäärä:	26. lokakuuta 2000
Sivumäärä:	109

Osasto:	Sähkö- ja tietoliikennetekniikan osasto
Professuuri:	S-38 Teletekniikka

Työn valvoja: Professori Jorma Virtamo

Internet on siirtymässä palvelun laatua tarjoavaksi verkoksi. Kehitys on seurausta Internetin rajoittuneista mahdollisuuksista tarjota uusia palveluja vanhaa Best Effort -periaatetta käyttäen. IETF on kehittänyt uusia arkkitehtuureja, jotka kykenevät tarjoamaan parempaa palvelua ja joustavuutta tulevaisuuden tarpeisiin. Uusin suuntaus on luokkapohjainen liikenteenhallinta, jota on sovellettu Differentiated Services (DiffServ) arkkitehtuurin kehittämisessä. DiffServ tarjoaa kevyen ja karkean liikenteenhallinnan, joka näin ollen on skaalautuva kaikkiin mahdollisiin tulevaisuuden tarpeisiin.

DiffServ perustuu mekanismien käsitteelliseen kuvaamiseen. Se ei ota kantaa lopullisiin toteutuksiin tai palveluihin, jotka on tuotettu sen pohjalta. Käytännön toteutukset on jätetty yritysten vastuulle, jotka voivatkin lisätä tuotteisiinsa toiminnallisuuksia, joita ei ole kuvattu suosituksissa. Tämä johtaa helposti yhteensopimattomuuksiin, koska tulkinta- ja sovelluserojen mahdollisuus on suuri.

Tämä opinnäyte on johdatus palvelun laatua käsitteleviin ongelmiin Internetissä. Se esittelee nykyisen Best Effort -verkon sekä Integrated Services ja Differentiated Services -konseptien mukaiset toteutusmahdollisuudet. Työn tavoite on esittää konseptien heikkoudet ja vahvuudet sekä tarjota simulaatiopohjaisia tuloksia DiffServ -verkon erilaisista toteutuksista aiheutuviin ongelmiin. Lisäksi verkon mitoituksen vaikutusta saavutettuun palveluun tarkastellaan muutamien esimerkkien kautta.

Avainsanat: Differentiated Services, Internet, Simulaatio

HELSINKI UNIVERSITY OF TECHNOLOGY

ABSTRACT OF
THE LICENCIATE THESIS

Author:	Marko Luoma
Title of the thesis:	Simulation studies of Differentiated Services Networks
Date:	26th October 2000
Number of pages:	109

Faculty:	Electrical and Communication Engineering
Laboratory:	S-38 Telecommunications Technology

Supervisor: Professor Jorma Virtamo

Internet is moving towards Quality of Service (QoS) networking. This is due to the limited possibilities which Best Effort (BE) networking has. New architectures are being developed by the Internet Engineering Task Force (IETF) to address these limitations. The most recent trend in this development is class based quality separation. Architectural concept, which is developed based on this, is called Differentiated Services (DiffServ). DiffServ offers low overhead, coarse grained traffic control, which should be flexible in all possible dimensions of future evolution.

DiffServ is based on the conceptual formulation of mechanisms. It does not cover implementation or services. Implementation of DiffServ compliant devices is left to the resposibility of individual implementers, which may add other functionalities beyond those described in DiffServ RFCs. This leads easily to non-interoperable services, as there is a great possibility of having different types of services and implementations of same services.

This work serves as an introduction to the service problems of QoS networking in the Internet. It presents Best Effort, Integrated Services and Differentiated Services architectures, and tries to pinpoint their strengths and weaknesses. Simulations of DiffServ provide some insight to the problems of mixed implementations of same service. In addition, difficulties of service provisioning become apparent from the results.

Keywords: Differentiated Services, Internet, Simulation

Foreword

This thesis is based on the research I have carried out during the period of years 1997-2000 in the Helsinki University of Technology / Laboratory of Telecommunications Technology. The research was conducted in two consecutive projects funded by Academy of Finland: Models for Integrated Telecommunications Networks Traffic and Architecture (MITTA), and Models for Integrated Internet and Telecommunications Network Traffic and Architecture (MI^{2}TTA). These projects concentrated on modeling and traffic issues of multiservice networks.

A big part of my research work was carried out in close co-operation with people working in project IPANA, especially M.Sc. Mika Ilvesmäki with whom I have had lucky to publish many joint papers. Other people who deserve special thanks for making this thesis what it is are Doc. Kalevi Kilkki who has been my mentor in understanding often so philosophical questions related to internetworking, and Prof. Jorma Virtamo who has been more than patient in supervising my work.

This work was also influenced by numerous talks which I have had with my colleague M.Sc. Markus Peuhkuri. I don't know if this have ever been possible to do without his help in developing suitable tools and comments on analysing the data. Also I would like to reserve special thanks to the people in COST263 action from fruitful discussions about valuability of Differentiated Services. Those comments opened my eyes for many problems which I would otherwise have neglected.

Last but not least, I want to thank my beloved wife Marion who has been patient and understanding during the course of my work.

Espoo, October 26th 2000

Marko Luoma

Contents

Chapter 1

Introduction

1.1 History and background of the Internet

Internet was born about 20 years ago, trying to connect together a US Defense Department network called the ARPAnet and various other networks using terrestrial, radio and satellite connections. The ARPAnet was an experimental network designed to support military research - in particular, research about how to build networks that could withstand partial outages (like bomb attacks) and still function [KH93]. From this Internet, and especially its network protocol IP, flood all over the academic world to interconnect large and expensive computer facilities.

Most of the architecture and the instrumentation for the accounting and the traffic control in the Internet reflect this historical status - typical government bulk-funded service for the academic community. For this reason, Internet has been a research environment with the usage-insensitive costs that has often been transparent to the end-users. As a result, the current Internet architecture is not able to meet the demands of the current usage. The most significant problems are the lack of mechanisms for allocating network resources among multiple entities or at multiple qualities of service and accounting based on the resource consumption [BBCW94]. Bottom line is that the Internet has no controlling and limiting mechanisms for the use of bandwidth resources. As the amount of the traffic grows and new applications, with fundamentally different traffic characteristics, come into the widespread use, resource contention will become a problem. As an example, multimedia applications produce traffic profiles which are fundamentally different from the data transfer. Multimedia has large volumes of information with much higher mean and lower variance in the bandwidth. In addition, these applications are not designed to share the bandwidth as equitably as the data transfer applications.

These are the reasons for the fact that the quality of service (QoS) architectures are finding the way to the Internet. Deployment of the QoS into the Internet means extensive re-engineering in the architecture and the creation of new service models. Design choices we make for the QoS-Internet are far from being the exclusive concern of a small technical community, these issues will have far-reaching implications for the general public. In particular, these design decisions will play an important role, along with many economic and social factors, in determining the nature of our future telecommunications infrastructure.

1.2 Scope of this study

Differentiated Services (DiffServ) has gained a lot of attention as a means to offer 'first hand' QoS. Operation of the DiffServ is based on the aggregate traffic handling with the possibility to do fine grained flow analysis at the edges of the network. Differentiated Services is a low overhead solution which is aimed to provide a carrier scale solution to the QoS problem. This solution has different functionalities in the access network and the core network. In the core network forwarding treatment is as simple as possible whereas in the access network more time is dedicated to the packet conditioning. Implementation of the services in the DiffServ is based on standard blocks. These blocks, however, are only functional descriptions of the actual operational elements. This means that there is a large amount of freedom in the implementation of the actual forwarding treatment. Freedom brings possibilities and flexibility to scale to the future but also problems of inter-operability and difficulties to formulate coherent services.

This work serves as an introductory review to the problems of the service engineering in the Internet and especially in the DiffServ Internet. Implementation of the functional blocks is evaluated through simulations of the different service scenarios. These scenarios are simple combinations of the standardized forwarding treatments. To become widely used and successful architecture, differentiated services must provide coherent service irrespective of the implementation of actual forwarding treatment. This would make possible to serve customers, based on their demands, on global scale (today this seems not be possible as interoperability and implementation issues are on a vague basis).

1.3 Outline of the Thesis

Chapter 2 will concentrate on the formulation of the terminology which is used in the Internet engineering. Some terms have contradicting meanings and therefore attention is paid to clarify points with possibility of misunderstanding. Chapter 3 concentrates on the architectural issues of the Internet service engineering. The difference between the current Best Effort Internet and proposed QoS architectures is briefly examined. Most time is paid on the issues of the Best Effort and Integrated Services architectures which are not analyzed in this thesis further. Chapter 4 is devoted to the Differentiated Services. Construction blocks of the DiffServ are explained in detail. Some examples are also given on times when they are needed to make some issues clear. Chapter 5 deals with the implementational issues of the Differentiated Services. Differences between implementational aspects are explained and some conceptual difficulties are evaluated more closely. Chapter 6 contains simulation scenarios and results. Finally the work is concluded in Chapter 7.

Chapter 2

Terminology

Terminology of this thesis is strongly affected by two different communication worlds: telephony and Internet. These two worlds used to be two different islands where terminology and the way of doing things evolved to different directions, many times with clear intention. Now, as these two worlds are coming together, many terms used in both of the worlds have contradicting meanings. This chapter covers some of the essential concepts and terms.

2.1 Connection

Term connection is easily combined with the thought of a telephone conversation between two parties. Connection is established at the beginning of the conversation through the network. Network is aware of the connection as long as it is going on, after which it is torn down. Connection in the Internet is not as clear as it is in the telephone network. Internet does not have connections inside the network, but communicating parties may have connection between them. This is on the level of transport protocol, which keeps track of the communication processes between different peers. Network is not aware of these connections; it merely sees packets coming from one user going to, possibly, a number of different users.

Terminology for the connections is inherited from the Open Systems Interconnection (OSI) model [HS88]. In the OSI communication between two parties is divided into seven different protocol layers, see Figure 2.1 for representation of the OSI stack and communicating protocol elements. Each item on these layers

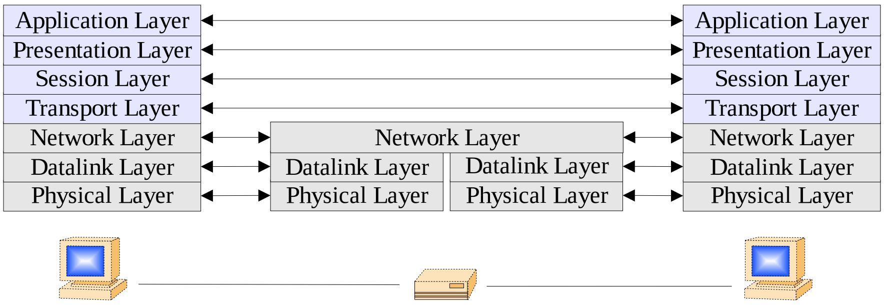


Figure 2.1: Communication in the OSI model

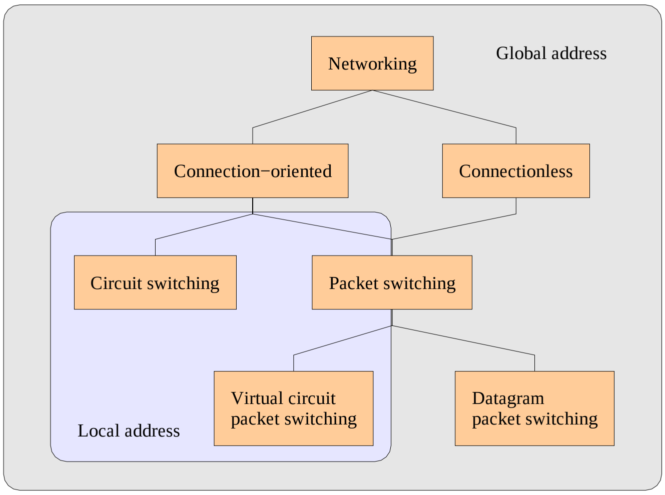


Figure 2.2: Division and differences of the connection-oriented and the connectionless networking

forms a connection with adjacent item in the communicating peer at the same layer. Layers 1-3 are network layers which have only a local meaning, i.e. a connection is formed between two following elements in a communication path through the network. Layers 4-7 are host layers, which communicate through the network with adjacent item of the communicating peer.

This thesis concentrates on the simulation studies of the performance of Differentiated Services. Therefore, the term connection is used for connections in layer 3 of the OSI model (e.g. IP layer in the TCP/IP model). When the term connection is used on other occasions to represent communication between protocol items at the other layers, it is expressed with a *remark*.

2.1.1 Connection-oriented and connectionless networking

Telecommunication networks can be classified into connection-oriented and the connectionless networks. Connection-oriented and connectionless networks differ in the address information which they use in forwarding of the information, see Figure 2.2, and in the nature of communication path within the network.

2.1.1.1 Connection-oriented networking

In a connection-oriented network a connection is established between the communicating parties before any information can be transferred on a session. After the session, this connection is also terminated. Connection establishment phase reserves suitable amount of resources along the path of communication and binds them to the connection identifier.

Association of the resources and the connections may be carried out on the link layer (L2) or on the network layer (L3).

Link layer association means that in the connection establishment phase a logical link address is associated to the connection on a link-by-link basis. Forwarding of the packet is bound to this link layer address. Therefore, there is no need to consult network layer addresses in the forwarding of the information. This is the way, which is used in the PSTN and ATM networks.

Network layer association means that the association of resources is carried out based on the global network addresses; each packet has to be investigated in order to find a connection which packet belongs to. Forwarding is associated to this classification. This is the way, which is used in the Integrated Services Internet architecture.

2.1.1.2 Connectionless networking

Connectionless networks do not have explicit route or association of the resources to the connections. Connectionless networks, therefore, provide service, which can be characterized to be their best attempt to deliver the information so that each packet is treated as an individual connection.

2.1.2 Flow

Definition of the flow dates back to the mid 80´s when packet train phenomena was observed in the token ring local area networks. Packet train was a burst of packets traveling from the same source and heading to the same destination [JR86]. Independently from the packet train observations a proposition for a new building block, flow, was made [Cla88]. Background for this proposition was the evident necessity of the network devices to become aware of the individual connections traveling through them.

Since then several formulations for the flow has been given, with common ground in the packet train observations. Unfortunately, most of them have been concentrating on the traffic in the upper layers of the host protocol (TCP and application). Current definition of the flow is based on the formulation of Claffy [CBP95]. This approach differs from the previous definitions in that the flow is related to the arbitrary measurement point in the network. This type of independent, single point, observation in the IP layer is actually what a router sees from the passing traffic.

Flow is defined as traffic satisfying various temporal and spatial locality conditions. This means that packet train in the observation point falls within predefined time out, granularity[1] and directionality conditions.

Flow is somewhat similar to the connection. It tries to build a similar structure, which the connection has in the connection-oriented networking to connectionless networks, like IP networks. This is possible, if the following condition is valid

> ”Packets of a single session travel across the network along the same path.”

[1] Granularity is the filter defining packets which are member candidates for the flow, they become members if other conditions are also fulfilled. This filter can be based on the IP addresses, transmission layer ports, and protocol identifiers or in general as a combination of any fixed elements of the IP packet.

If this condition is true, it means that dynamic routing of the network operates on the time scales which are longer than the average connection time and therefore a flow can be treated as a connection.

Based on this definition and extensive traffic measurements, [CBP95, CM97] present flow analyses of the Internet traffic.

2.1.3 Connection state

Connection state (state hereafter) is the knowledge of network equipment about the connections passing through. In short, it is the mapping between the protocol addresses of communicating parties - in the Internet, this has often variable granularity among the connections. In the Integrated Services Internet architecture, state information contains information about the resource reservation and route of the connection.

2.1.3.1 Stateless operation

IP is a connectionless protocol. Connectionless means that the network treats every IP packet as an independent unit. Based on the independence, each packet receives similar treatment in the forwarding path of a router. Because forwarding treatment is similar for each packet, there is no need to associate any information about packets to the forwarding treatment. This lack of the association and somewhat black box approach to the traffic is called stateless operation. Stateless operation makes possible to change routing while session is up and sending traffic without any disruption in the service.

2.1.3.2 State based operation

In the connection-oriented service, a connection state is maintained in relevant places of the network. Connection state registers the presence and attributes of the connection. An instance of a connection state is the reservation state by which signaling protocol (RSVP [BZB$^+$97]) enables a connection-oriented service over the traditional IP infrastructure.

There are three ways for a router to become aware of session attributes:

1. **Locally**. If the router uses some form of intelligent processing to extract the knowledge about flows from the passing traffic, it relies heavily on the stability of the routing. The independent process of each individual router constitutes a local state, a state acquired based on the behavioral experiences (e.q. caching most often referred routing table entries [CV96]) or based on the intelligent processing [ILK98].

2. **Soft state reservation**. In the soft state operation an end system signals the flow setup and periodically refreshes this state. The end system may also signal flow tear down, but, in general, this is not required due to the periodic nature of the reservation. There is a distinct knowledge of the

individual sessions and their characteristics in each router by time of the setup and an arrival of successive refresh messages. In the case of a change in the network routing, a new state is formed on the new path with the next refresh (SETUP) message. State of the old reservation times out when regular updates cease to arrive.

3. **Hard state reservation**. In the hard state operation an end system signals the flow setup. The difference to the soft state is that there is no periodic update of the reservation and the end system has to tear the connection down. In addition, route pinning is used to make sure that the reservation attributes are met during the reservation. This way each router has an explicit knowledge of the flows using its resources. Hard state operation assumes an end-to-end state reservation, like in the telephone networks. This way of operation is currently not supported by the Internet service and protocol models.

2.1.4 Other state and connection information

There are lot of information, which has *state* in the Internet, but these has little or no meaning to the connection state.

Each router must be aware of the network topology or at least routes, which are valid in the network. This information has timer and activity flags forming a *state* for this information. Routing *state* information may be thought to be a potential connection space, but without reservation attributes. If routing is stable enough, this *state* information can be used for traffic engineering purposes - like in Multi Protocol Label Switching (MPLS). There is also congestion related *state* information in some router implementations. Router may track the traffic to find out which flows are using the resources most aggressively.

A *state* that has direct relation to the connection state is the *state* in the end system transport protocol e.g. in the Transmission Control Protocol (TCP). TCP uses a controlled *connection* structure in the communication over the network (irrespective of the nature of the network). TCP *connection* has several *states* which represent status of the *connection*, i.e. closed, established, listen, etc. However, from the network point of view a single connection carries multiple TCP *connections*. This is due to the tendency of applications to generate TCP *connections* very liberally.

In general, previous terms are not to be interpreted too strictly. There is no clear connection-oriented operation and connectionless operation. Rather, there are a large spectrum of options. At one end, there is the pure connectionless operation, where the routing is done for each packet independently (this is very time consuming but guarantees the best possible route on the light of received information). At the other end is the PSTN type connection paradigm. The Internet with its different service architectures forms a large overlapping area which aims to offer operation which provides the best parts of both ends while still keeping flexibility.

2.2 Protocols

2.2.1 Internetwork Protocol

The Internet is based on the concept of separation of the application and the network technology. This separation is done by the network layer (L3)[2] in which a single protocol, Internetwork Protocol (IP) [Pos81a], is used. IP provides a common interface for all applications and to all network technologies. Doing this it offers a possibility to use several applications on top of several different networks.

IP is a simple protocol defining only the encapsulation of the information and basic forwarding treatment of a packet, i.e. forwarding and in some form of also the priority[3]. As a connectionless protocol, IP has no awareness of connections using its services and has, therefore, no mechanisms to ensure that the data is delivered correctly to the destination. IP has all the weaknesses of connectionless protocol: forwarding treatment of a packet is independent of other packets, i.e. routing may change from packet to packet; delivery of packets is unreliable, there is no indication of the congestion. Dropping of packets during the congestion has no correlation to the IP protocol, although the IP header contains a field which relates the type of information and the preferred forwarding treatment (low delay, high reliability, and minimum cost).

To deal with losses and delays in the network extra protocols for the communication in the IP-based Internet have been developed.

2.2.2 Transmission Control Protocol

The most popular transport protocol (L4) is the Transmission Control Protocol (TCP) [Pos81b], which defines a reliable, end-to-end, octet streaming protocol with flow control and congestion control procedures. TCP uses a window based transmission control algorithm to restrict transmission rate to a value, which is suitable for the network and the receiver conditions. Original TCP has been amended with several additional algorithms [Ste97] to increase its dynamics from the slow speed modem connections to the modern high-speed networks.

TCP keeps track of individual *connections*, i.e. sessions, end-to-end. Both ends of the communication take part in the control by forming a *state*, i.e. session state for the time of the communication. This session state makes it possible to offer flow control and packet reordering at the receiver.

TCP implements retransmissions to handle losses in the network. To become aware of losses TCP has a timer related to each transmitted segment of data. If the receiver does not acknowledge the data before the timer expires, it is considered lost. The value of the timer is related to the experienced round trip time (RTT) through an algorithm which tries to balance the long term behavior

[2] In the Internet protocol model this is called Internetwork layer

[3] Look Section 3.2 for more information about the priority capabilities of the IP

and the short term fluctuation.[4] Acknowledgments (ACK) of the received data segments are the heartbeat of the communication. Acknowledgments perform clocking for the transmission of new packets so that they are spaced so wide apart that no packet is unnecessarily queued before the lowest speed link. However, this heartbeat is distorted by the option of delayed acknowledgments, where a number of acknowledgments are combined to a single ACK. This makes possible to send bursts of data to the network and cause unnecessary queuing before slow speed links.

TCP is the protocol of choice for reliable stream delivery service and over long-haul circuits, where the amount of data on the fly needs to be regulated. These two areas constitute major part of the Internet communication like web browsing, X-windowing and file transfer. TCP is, however, poor fit for many applications like multicasting (point-to-point nature of TCP sessions) and real-time communication (unnecessary retransmission of lost segments and windowed sending).

2.2.3 User Datagram Protocol

The other widely used transport protocol is the User Datagram Protocol (UDP) [Pos80], which defines an unreliable datagram delivery without flow control or congestion control. UDP is a minimal extension on top of the IP developed to deliver transaction oriented operation with multiplexing and error control properties.

Transaction orientation in the UDP reflects the nature of sessions in the UDP, it does not make a *connection* with the remote UDP client nor does it have an internal *state*. UDP pushes the datagram into the network and accepts incoming datagrams from the network.

UDP is the protocol of choice for the communication if efficiency over the fast networks with short latency is required. It is also suitable for the multicast communication due to the flexibility of the *stateless* operation. In addition, the minimal protocol structure of the UDP makes it an option for real-time communication.

2.2.4 Real-time Transport Protocol

Real-time Transport Protocol (RTP) [SCFJ96] is an addition to the UDP to make it more suitable for the real-time communication. UDP suffers from two major problems with regard to the real-time communication. One is the lack of sequence control and the other is the lack of delay control. RTP provides these functionalities through additional protocol information, specially designed for the real-time operation.

RTP is suitable for so called adaptive real-time applications, which can tolerate losses and delay variations to the some extent by having playout delay compensation and loss concealment. RTP provides tools for implementing these func-

[4] Comprehensive information about TCP/IP protocols and their algorithms is given in the book of Richard Stevens: TCP/IP Illustrated, Volume 1: The Protocols

tionalities. It provides sequence numbers and timestamps for the other end to observe possible misordering of the packets and to correct the time between the playout of the information in packets. Based on the observed loss and delay variation receiver can inform the sender about the status of the connection. This is done through a separate channel and protocol called Real-time Transport Control Protocol (RTCP) which is a part of the overall RTP specification.

2.3 Service related terminology

The term service is used quite often with different meanings. To clarify differences, a short introduction of services and service models is presented here.

2.3.1 Service model

Service model is a group of services, which the service provider offers or could offer to the customers. In a way service model is the strategic vision of the service provider - what is offered and what should be offered to the customers within the limits of applied technology (service architecture). Service model provides a tool for visualizing and grouping services in a common base, to see where they overlap, and also to see whether some important items are not covered at all. Service model is a tool for strategic decisions in management levels of the service provider.

2.3.2 Service

Service is a package that is sold to the customer as a form of Service Level Agreement, see Section 2.4.1. Service formulates the service model of the service provider from a single customer perspective, i.e. what single customer receives if he uses service provider's network facilities. The following are examples of services

- Leased Line Emulation [BBB+99]

  > ”*Leased Line Emulation (LLE) is a quantitative service, which emulates traditional leased line service. LLE delivers traffic with very low latency and very low drop probability, up to a negotiated rate* X*, between fixed set of endpoints [*A,B*]. Above this rate, traffic is dropped.*”

- SIMA [LRK98]

  > ”*SIMA is a qualitative service, which promises to carry customer traffic with a level which is comparable to the ratio of customers subscribed virtual bit rate* X *kbps and momentary bit rate* Y *from ingress point* A *to any egress point. Customer may select the traffic to be real time or non-real time traffic with the distinction*

> *that real time traffic has a higher probability to be dropped but lower delay, in general.*”

- Better than Best Effort [BBB+99]

> ”*Better than Best Effort (BBE) is a qualitative service, which promises to carry specific traffic at a higher priority than competing best-effort traffic. Such a service offers relatively loose (not quantifiable) performance from a given ingress point* A *to any egress point. Amount of the priority traffic is negotiated to be at maximum* X *Mbps*”

2.3.3 Service architecture

Service architecture is a general network platform that gives abstract representation of possible services and/or service models, which can be implemented by using the selected technology. Service architecture is rough functional representation of the network and does not necessarily characterize the actual service, which is delivered by the network. Two popular service architectures are Differentiated Services and Integrated Services. Differentiated Services and Integrated Services are explained more deeply in Section 3. Service architecture offers a generalization of functional tools and elements which are used by the technical personnel of the service provider to implement services in the service model.

2.3.4 Service class

Service class is a behavioral representation of one possible forwarding treatment in the service architecture. The name service class is used in the Integrated Services architecture, see Section 3.4.1. Closest counterpart in the Differentiated Services is the Per Hop Behavior (PHB), see Section 4.2.2. The following are service class examples from the services presented in the Section 'Service'.

- Leased Line Emulation [BBB+99]

> ”*Packets submitted for leased line service should be marked with the DiffServ codepoint corresponding to the EF PHB*[5]*. From the ingress point* A *to the egress point* B *the provider is promising to carry up to* X *Kbps of traffic. Excess traffic will be discarded.*”

- SIMA [LRK98]

> ”*Packets submitted to SIMA service should be marked with DiffServ codepoint corresponding to DRT/NRT PHB. When the user transmits traffic with the momentary bit rate* Y *Kpbs that equals the virtual one* X *Kpbs, medium drop precedence [*DRT24*] is used. As the momentary bit rate exceeds the virtual one the packets are*

[5] For definition of PHBs see Section 4.2.2

> *marked with a drop precedence [*DRT25,DRT26*], i.e. value indicating a lower relative order. As the momentary bit rate falls below the virtual bit rate packets are marked with a drop precedence [*DRT21,DRT22,DRT23*] indicating a higher relative order.*"

- Better than Best Effort [BBB^{+}99]

> "*Packets submitted for the BBE service should be marked with the DiffServ codepoint corresponding to the AF11 PHB. The provider is promising to carry up to* X *Mbps of traffic from the ingress point* A *to any egress point at a higher priority than best-effort traffic. A lower class of service corresponding to the AF13 PHB will be applied to traffic submitted for the AF11 PHB, in excess of* X *Mbps.*"

2.4 Service Level Agreement

2.4.1 General SLA

Service Level Agreement (SLA) is a contract between the service provider and the customer. This contract describes what the service provider is offering to the customer and what the customer pays for the service. SLA is a long-term agreement, which is negotiated when the customer orders the service or is willing to change the level of service. SLA is not made individually for each connection, but each connection has a level and characteristics agreed in the SLA. SLA has different forms depending on the service offered, network under consideration and legal constrains within the area. Typical SLA covers issues like:

1. From the side of the customer:
   - **Traffic specification**; some level of characterization of the aggregate traffic stream the user is allowed to send into the network. This is often a token bucket filter characterizing the worst-case traffic stream.
   - **Scope of the agreement**; description of the ingress and the egress points having the defined traffic specification and the service level guarantee.
   - **Price paid for the service**
2. From the side of the service provider:
   - **Service description**; some form of behavioral description of the service provided to the customer.
   - **Quality description**; characterization of the resources dedicated to the user and/or quantifiable parameters associated to the service (e.g. throughput, delay, jitter and response time).
   - **Availability of service**; hard downtime (i.e. line broken) and service degradation (resources are not accessible on full scale of the SLA) time during a certain period.

- **Level of technical support**; hot line support numbers and the price that is charged per call or minute from the customer.
- **Compensation**; amount paid by the service provider to the customer under occasions of service degradations (either hard downtime or service degradation)

Service level agreements have gained more consideration during the last two years as the new QoS capable packet networks have arrived and the e-commerce applications have been emerging. This has made the outsourced network connectivity as a strategic asset, which has a great influence to the corporate turnover.[6]

Service level agreement is always a negotiation between the customer and the service provider. Service provider is usually not eager to push the SLA in the form of clear contract. This is due to compensations when the service level is not within the contract. Therefore, a number of templates for the customer use in the SLA negotiation are available on the Internet. They are often produced by the companies manufacturing tools for the SLA verification.[7]

2.4.2 SLA and TCA in DiffServ

Differentiated Services (DiffServ) uses the term SLA with different meanings. Originally, the SLA was thought to be like the general definition of the SLA, presented in the previous section. However, this was realized to be an improper way to approach the problem. DiffServ is not the service which is sold to the customer. It is merely an architectural environment, which the service provider uses in the service provisioning.[8].

New definition divides the SLA into two parts:

1. Service dependent part

   DiffServ SLA is not directly part of the DiffServ, but it does have direct effect on the DiffServ. DiffServ SLA is a part of the service models of the DiffServ service provider. A part of the DiffServ SLA is the Traffic Conditioning Agreement (TCA), which represents a filter to which the specific SLA is bound. This filter is a classifier separating the traffic stream for processing.

2. DiffServ dependent part

   To address the service level aspects that are directly addressable by the DiffServ, a new term Service Level Specification (SLS) was adopted. SLS is defined to be a set of parameters and their values, which together define the service offered to a traffic stream by the DiffServ domain. TCA has also

[6] Implications of the service level commitments to the business are explained in the white paper: 'Service Management for IP Networks', prepared by Renaissance Worldwide Inc, see www.visualnetworks.com/resources/visual-slm-wp.pdf

[7] Templates along with the information about the SLA verification tools can be found from the Internet location: http://www.data.com/issue/990207/sla.html

[8] For more information about relation of the DiffServ to the services, see Section 4.1.

a DiffServ specific version Traffic Conditioning Specification (TCS). TCS is defined to be set of parameters and their values, which together specify a set of classifier rules and the traffic profile [Gro99].

2.5 Utility

2.5.1 Utility in general

Utility is a function, which describes fitness of the network service to the purpose of the communication. Network service in this context is evaluated with quantifiable parameters like bandwidth and delay.

Approaches to the utility analysis are two and they provide different types of functions from the same network service.

1. **Application level utility**, where analysis is purely based on the application behavior. Application level utility expresses how the connection is valued as a function of resources, which are available to the connection. In a sense, this covers the issues of the application design and transport protocol operation. The general assumption for the TCP based applications is that their utility function resembles a logarithmic function. This is due to the uncontrollable delay in the transmission which prohibits design of applications which require timely forwarding. Characteristic property of the logarithmic utility is that logarithmic increase makes it possible to divide resources to infinitesimally small shares and still receive some utility. These utility functions are analyzed in [She95].

2. **User level utility**, where analysis takes into account that one has to pay for the service and therefore expect to receive service which may very well be better than what the application would require. General influence of the monetary incentive to the utility of the network service is negative. This negative influence lowers the threshold for leaving the service if expected utility is low. What this in fact means is that the network service, which may very well be acceptable if no money is paid for the connection, is unacceptable if the connection is charged. Issues of the incentives and the form of the utility functions are analyzed in [CSEZ93].

For competitive operation of the network, network has to be dimensioned and engineered so that the utility levels are always on the tolerable level. Tolerable means, however, different things for different services and different monetary incentives.

2.5.2 Utility and DiffServ

Differentiated Services has some points where the utility analysis is an appropriate tool. One example of such point is the metering and the marking process of the Assured Forwarding (AF) PHB. In this process, some rate is set to be the target rate for the customer. Target rate is the maximum rate, which the user is allowed to send to the AF class with a high priority. However, selection of the transmission protocol influences to the actual outcome of the service. After a packet loss, TCP lowers the transmission rate to the half of the value before the packet loss. This, if the network is full, causes the customer to receive only 66% of the resources, which were dedicated to him. With the utility analysis the effect of different metering and marking mechanisms to the applications and operation of the TCP can be explored, see Section 5.4.3.

2.6 Quality of Service

Quality of Service (QoS) is used today with a large variety of meanings. Service providers use the QoS to express that service, which they deliver, is good[9]. For a common user, this often translates to the availability of service, which fulfills his demands. This demand, however, changes all the time and often has nothing to do with real communication requirements.

Foundation of the network based Quality of Service (QoS) formulation is the thought that communication characteristics can be measured and guaranteed for each individual conversation. Quality of Service is typically defined in a form of directly measurable parameters such as delays: transfer delay and transfer delay variation; bandwidths: minimum, average and/or maximum; and losses.

In the QoS networking, network provides guarantees to the users for the delivery of information in accordance of commonly agreed parameters. These parameters are in the first hand provided by the user in a form of source characteristics (bandwidth requirements) and QoS requirements (delays and losses). Network validates these parameters in the admission control process, where delivered parameters are compared against the capacity restrictions of the network. Based on the judgement of the admission control algorithm an agreement between the user and the network is made for the duration of conversation. This is possible to do within the technological constrains which we have today. It is, however, debatable whether it is wise to do so, and whether this kind of service is good in the framework of the first formulation for the QoS.

Nature of the Quality of Service depends on the mechanisms and parameters used in deriving the contract. There is a long list of different QoS formulations with different propositions for provisioning:

- **Strict QoS:** No loss of the information is allowed in the network. Packets, which are delivered after the maximum delay, are considered lost.
- **Probabilistic QoS:** Some 'controlled' loss of the information is allowed in the network. Packets which are delivered after the maximum delay are considered lost.
- **Relative QoS:** Loss of the information is uncontrolled. Network aims to preserve relative order of importance between connections.

[9] Or better what it used to be

- **No QoS:** Delay and loss characteristics of the communication are totally uncontrolled. Network aims to offer similar QoS for each packet.

Class of Service (CoS) is a way of managing the traffic in the network by grouping similar types of the traffic (for example, e-mail, streaming video, voice, large document file transfer) together and treating each type as a class with its own level of service priority. Unlike the Quality of Service, Class of Service does not guarantee the level of service in terms of bandwidth and delivery time - rather it offers multiple levels of best effort service.

Class of Service and Quality of Service are not competing notions. They should be considered as two objectives, which try to offer the best they can to a selected group of traffic in certain network conditions. CoS offers a coarse grained control. It scales easily in the backbones and delivers suitable level of service for normal computer based communication. QoS offers a fine-grained control. It does not scale with the today's technology. However, it manages in the periphery, where the amount of traffic is lower, and in the delivery of real-time traffic in the backbone (assuming that the volume of real-time traffic is a fraction of the total traffic volume).

2.7 Fairness

When dealing with various aspects of the network resource control and the traffic management one comes to face the question of fairness. Fairness is a term which has many definitions, some originating from the area of networking and some from the area of social economics. In this work fairness is defined according to the max-min fairness criterion [VFJ$^+$00].

Division of resources is fair if and only if:

1. No user receives more resources than it requests.
2. No other allocation scheme satisfying condition 1 has a higher minimal allocation.
3. Condition 2 remains recursively true if we remove the user with the minimal allocation and reduce the total resources accordingly.

Chapter 3

Internet services, architectural models

This chapter gives a broad overview on the problem of the quality separation in the present Internet. It also introduces the different service architectures, which have been proposed and standardized for the use in the Internet.

First a brief outlook to the different value aspects of the Internet services among the different user populations is given. Then these value aspects are evaluated against the current Internet architecture and the new QoS architectures. Last three sections of this chapter are devoted to the different service architectures, namely: best effort, differentiated services and integrated services. At the end, a short summary is given to present the main differences among the service architectures.

3.1 Architectural considerations

3.1.1 Value aspects and user communities

Internet has become a channel where one can offer/sell services/products to a large and possibly geographically disperse user population. This type of commercial usage, with the hard competition of clients, makes content providers to require quality differentiated network services from their service providers. In addition, at the same time, the same infrastructure is used to offer virtual private network (VPN) services for corporations with different service level agreements (SLA). These VPNs, though private in the nature, share capacity of the network, and the service providers intention is to use the excess capacity of the VPNs for the Internet usage.

To manage this reality, division of the customer population, their primary network usage and incentives guiding their network usage into three main groups, is necessary. This division is summarized in Figure 3.1.

1. Corporations
    - Motivation for the quality demand is strategic differentiation from other corporations.

- Main usage are for:
  - (a) Virtual Private Networking, where some well defined set of access points form a closed operation area.
  - (b) Public Internet service, which provides them a way to disseminate products and product information to their customers. If this function is strategic for the corporation, it will require premium level service from the service provider to give the best impression of the corporation to its customers.
- Main incentive is the budgetary cost.

2. Academic

   - Motivation for the quality demands are the network performance requirements.
   - Main usage is for the public information retrieval and dissemination among the general population and the fellow institutes.
   - Incentive influences are minimal due to the complex structure of the cost transfer inside the institutions.

3. Residential

   - Value aspects of the quality are on the economical and value added service area.
   - Main usage is the recreational activities and communication to the individual reference group.
   - Incentive is the direct cost of the communication.

Internet, today, offers a single class of best-effort service; that is, there is no admission control and/or resource reservation. The network offers no assurance about when, or even if, packets will be delivered. This uncertainty of the information transfer process makes the Internet unsuitable for the real-time applications.

Thinking of the requirements of different user communities and different applications, the network/router should be able to separate the traffic into classes, which by their nature should be treated differently. However, today such actions are not done and all traffic is treated similarly within a single queue of the router's output port. Real-time applications, implemented on top of the UDP, interfere with the data applications, implemented on top of the TCP, as they compete of the common buffer and the link capacity. Bursty data applications deteriorate the quality of real-time applications by increasing queuing delays of the real-time packets. At the same time real-time applications may act unfairly on the congested low speed links, by taking excessive resources compared to the self-adjusting data applications.

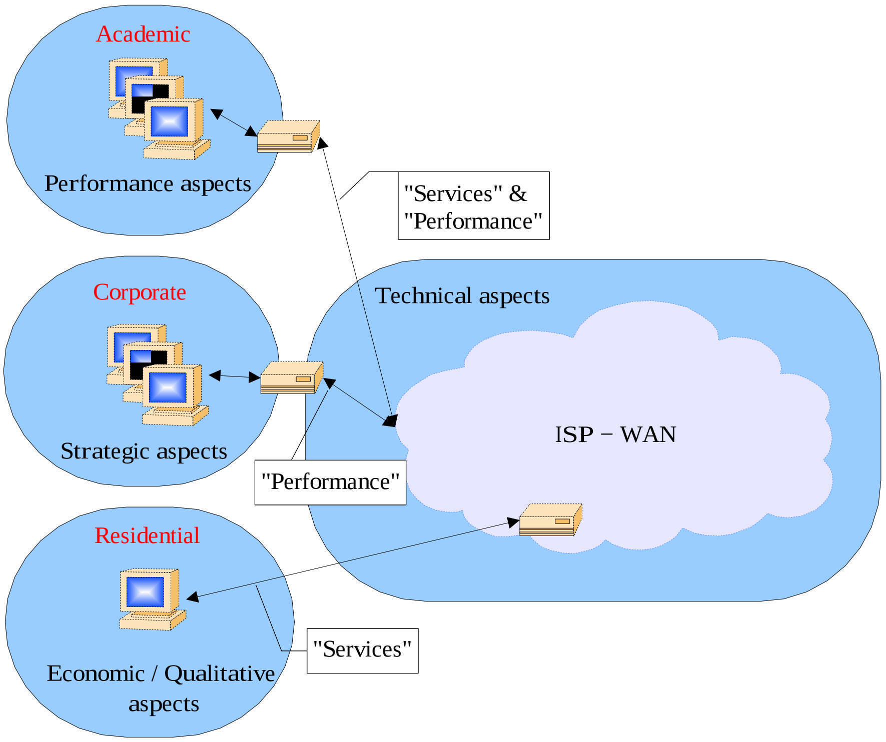


Figure 3.1: Value aspects of the different players of the information market

3.1.2 Solutions for the quality differentiated networking

To address the problems of the traffic interference and demands for the different quality levels, one can either change the Internet service architecture or current router and application behavior.

3.1.2.1 Solutions within current architecture

By improving the operational aspects of typical router implementations one can take full advantage of the possibilities stated in the current 'Requirements for IPv4 router' specification [Bak95]. Today lack of the QoS routing protocols, traffic classification and class dependent scheduling prohibits deployment of the *type of service forwarding*.[1]

Other option to address the problem without changing the architecture is to modify the application implementations rather than the router implementation. This can be done by adding some adaptivity to the delay and loss requirements of the applications. One extreme is to include some sort of flow and congestion control to all existing and forthcoming applications. A disadvantage is that there is no guarantee that there will not be misbehaving individuals.

Nevertheless 'change the implementations but not the architecture' approach has several important advantages:

- No changes are required to any of the network interfaces, so changes can be

[1] *Type of service forwarding* is explained in Section 3.2.

incrementally deployed both at the end hosts and at the routers.

- Network mechanisms (Fair Queuing and its relatives) and application mechanisms (delay adaptation) are relatively well understood.

However, in this approach the network would deliver the same class of service for all users, with no assurances as to the quality of that service[2]. While the network would, possibly, protect the users from each other, it is up to the applications to adjust to the inevitable variations in the packet delay and available bandwidth. There are likely to be limitations to this adaptability. Moreover, because there is no admission control the network must be provisioned so that the fair bandwidth shares are not, except in very rare cases, unreasonably small [She95].

3.1.2.2 New architecture solutions

It is, however, unavoidable that a more solid foundation for the operation in the future is required. This means an extension to the Internet service model - from the single class of best-effort service to include a variety of service classes. The fundamental questions, which apply to all possible service model extensions, are: *how does the architecture decide which service to give a particular flow, and how firm guarantees are given for the traffic?* For the first part of the question, two obvious answers are: the user can pick the service; or the network can pick the service for the user.

Current service model and the service model extensions, Integrated Services and Differentiated Services, are discussed in more detail in the following sections.

3.2 Best Effort Internet Services

There is no official Best Effort Internet service (BE) definition. However, RFC 1812 [Bak95] contains a short definition of the service that should be delivered in the Internet layer of the network.

> ”*IP is a connectionless or datagram internetwork service, providing no end-to-end delivery guarantees. IP datagrams may arrive at the destination host damaged, duplicated, out of order, or not at all. The layers above IP are responsible for reliable delivery service when it is required. The IP protocol includes provision for addressing, type-of-service specification, fragmentation and reassembly, and security. The datagram or connectionless nature of IP is a fundamental and characteristic feature of the Internet architecture.*”

The definition states that the best effort Internet offers a service that is best characterized as the network's best intention to transfer the information between

[2] Type of service forwarding contains possibility to the precedence ordered queuing but this is even less experimented option than the TOS routing

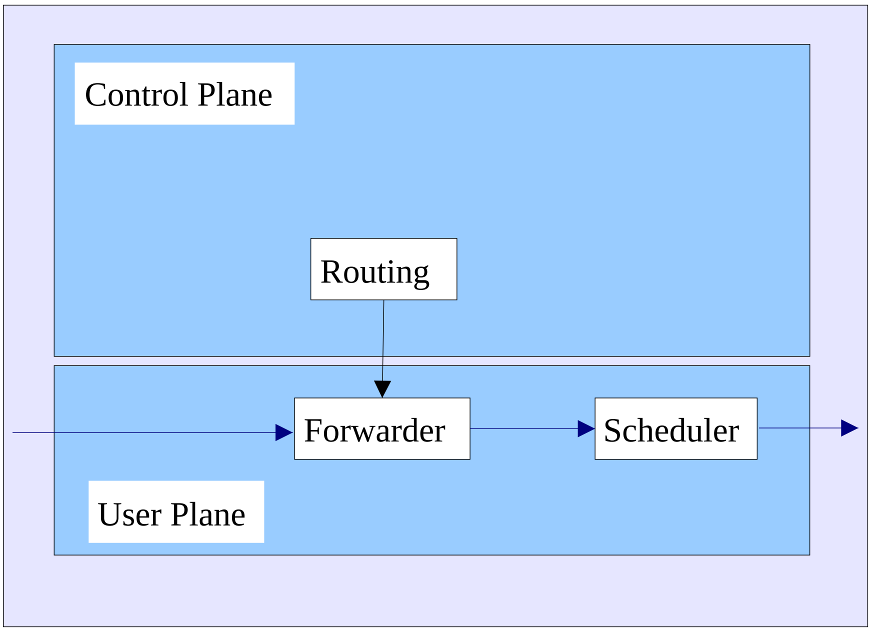


Figure 3.2: Best Effort router, functional blocks

endpoints. Actual forwarding treatment is not characterized at any level, but has evolved by experience to one best effort service class. In a single service class network, there is no separation of the traffic based on the characteristics of the application or the traffic. Separation can only be accomplished through the management processes, for example by dedicating part of the resources for a group of users. This type of management operation, called Virtual Private Networks (VPN), is an operator specific extension to the normal network operation and does not relate to the service architecture.

Best Effort Internet router, presented in Figure 3.2, is technically a simple computer system which forwards packets from one link to another. Forwarding is done based on the destination address lookup, which resolves the proper link for a packet switching. This process is usually executed on First Come First Served (FCFS) basis, meaning that there is no separation of the time critical traffic from the rest of the traffic. The previous quotation, however, refers to the following options:

> ”*. . . The IP protocol includes provision for addressing, type-of-service specification. . .*”

The type of service (TOS) field in the IP header (see Figure 3.3), conceptually, makes it possible to separate the traffic based on the information delivered in the TOS-field [Alm92].

The TOS-field is divided into two different subfields: precedence and type of service.

Precedence is a scheme for allocating resources in the network based on the relative importance of different traffic flows. The basic mechanisms for the precedence processing in a router are the precedence-ordered queue service and the precedence-based congestion control.

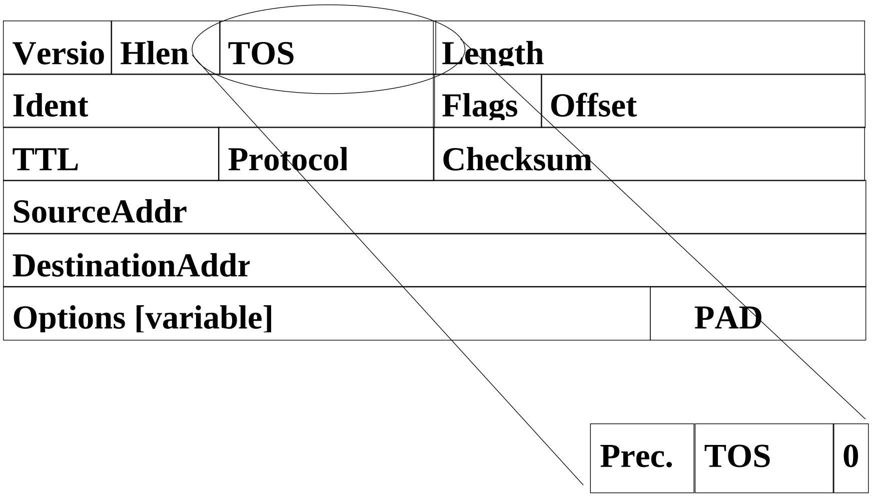


Figure 3.3: IP header with type of service field structure

Type of service expresses the nature of the traffic sent to the network. Internet has no direct knowledge of how to optimize the path for a particular application or a user. Therefore, the IP protocol provides a (rather limited) facility for the upper layer protocols to convey hints to the Internet Layer about how the forwarding tradeoffs should be made for a particular packet:

- minimize delay [1000][3]
- maximize throughput [0100]
- maximize reliability [0010]
- minimize cost [0001]
- normal treatment [0000]

Definitions of 'minimize' and 'maximize' are heuristic of their nature; they don't relate to any actual value in the network nor even in the router. So the forwarding treatment the network offers for the 'minimize delay' traffic can be far from the users' idea of a low delay. It is also important to note that the type of service subfield relates to the selection of paths and not to the queuing or to operation at the time of congestion, as the precedence field does.

For some reason, precedence and type of service subfields are currently not used in the Internet. The reason may be historical; earlier there was no need for this type of operation in the non-military network, or practical; implementation and administration of routes and queues based on this information is a complex and easily misused process.

[3] Square brackets represent the actual bit value of the attribute

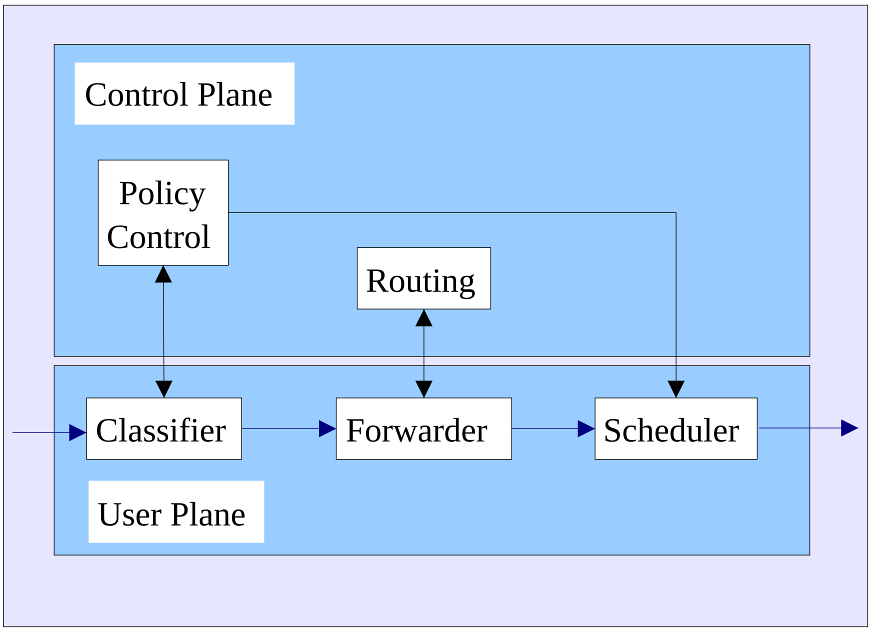


Figure 3.4: DiffServ router, functional blocks

3.3 Differentiated Services

Differentiated Services (DiffServ)[BBB$^+$99] was more or less developed from the experiences of the use of class based queuing (CBQ)[4] to do the traffic/priority distribution in a single expensive congested link [WGC$^+$95]. DiffServ is a combination of classification and related scheduling, see Figure 3.4, like the CBQ but in a more general terms. The idea of the DiffServ is to offer a low level of granularity in the quality differentiation. Flows are aggregated to a small number of classes which makes forwarding as easy as possible in the core network. Required processing, i.e. policing and assigning flows into the classes, is done at the edge of the network. Due to aggregate handling inside the network, the quality which the DiffServ offers is dependable on the network provisioning and the traffic distribution.

In Differentiated Services either the user explicitly or the network implicitly chooses the appropriate service class for the flow without actually reserving resources to the individual flows. What this means concretely is that the application sends its packets, possibly without stating anything about its service requirements, and the network then classifies the packets[5] into the proper service class and handles them accordingly. From the networking ideology point of view, this resembles current best effort Internet and stateless operation which it makes use of. This allows good scalability even in cases of millions of flows.

Advantages (and at same time disadvantages) of this approach are:

1. There is no service level negotiations. Applications do not specify their desired service level to the network, and the network does not, in turn,

[4] See Section 5.2.6 for more information about CBQ

[5] This classification is done anyway in order to make sure that the user is conforming to the agreed SLA

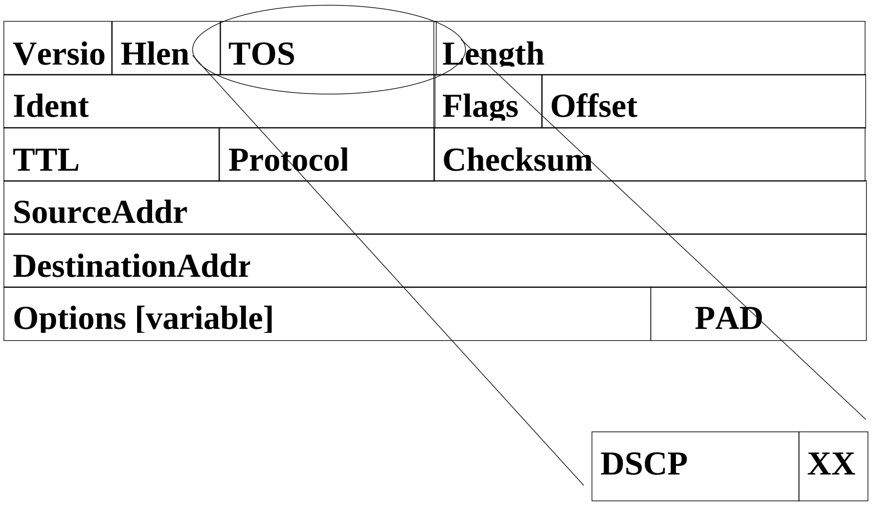


Figure 3.5: IP header with differentiated services field (DSCP)

describe the delivered service to the applications.

2. Mapping of the applications and/or the users to a particular service class and the nature of the service delivered to the each service class need not be uniform across routers or stable over time. This is possible as there is no explicit commitment to a given service level.

From an individual session's point of view, the service still somewhat resembles the best effort. Sessions in a class are not isolated from each other, and there is no admission control to limit the number of sessions within a class [BV98].

In the DiffServ, quality class of a packet is coded in the DiffServ code point (DSCP). This code point is a reformatted TOS-field of the IP header. This time six of the eight bits in the field are used to express the class for the packet, see Figure 3.5. This class specifies both forwarding treatment (scheduling) and path selection (routing). Forwarding treatment is a set of rules defining importance of a class compared to the other classes. Rules characterize the relative amount of resources, which should be dedicated for a particular class in the scheduler, and the packet discarding order during the congestion.

Mixture of the traffic in different classes is related to the question *'who codes the DSCP ?* If coding is done by the user, the mixture of the traffic in different classes in the network is uncontrollable by the ISP. However, the ISP can re-mark packets entering the network, but this prohibits user control over the QoS of the traffic which he is generating. User may also let the network to do the DSCP coding. This allows, for the ISP, full control over the traffic in the network. If the ISP uses this control wisely, it may provide better service with fewer resources. This is the best interest of the user, better service, and the ISP, lower operating costs.

A more detailed analysis of the DiffServ is presented in Chapter 4.

3.4 Integrated Internet Services

Integrated Internet Services (IIS or IntServ) is a extension to the best effort service model which allows an user to select the service class and the quality of service parameters which best fit to his communication requirements. This selection is done on flow-by-flow basis.

In the IntServ, network offers a set of service classes, which have profoundly different characteristics. User, or many times the application that is using the network for a task set by the user, explicitly selects the service class. This maximizes the user control over the communication process. User control over the service class selection allows using other criteria than just suitability. User may select service class based on monetary, performance or reliability reasons, which are incomprehensible for other users or the network.

The IntServ network uses state-based reservation of resources for each individual flow. This reservation is done by a control protocol, RSVP[6], and an admission control agent operating on the network devices. Admission control is used to guarantee the quality of connections by doing à priori analysis of the network resources against sum of the used resources plus a new connection request. If the network has not enough resources to guarantee the quality of the ongoing connections plus the new request, it will deny the access from the new connection. IntServ uses periodical refreshment of the reservation, i.e. state reservation is soft. If refreshment messages are not received by the network, it will free up the resources from the particular connection. In case refreshment messages do not contain changes to the original reservation, admission control is not triggered. The network must provide some system to encourage the users to request the proper service classes for their applications. Pricing of the service classes is one approach. Charging more for the higher quality service class will ensure that only the performance-sensitive applications will request it.

Implication which an explicit service class reservation has is that the service classes must be known to the users and/or the applications. This may sometimes be problematic, like when different ISPs use the IntServ to build proprietary services, which may not be universally known or visible to the users.

3.4.1 Service classes

There are a number of service class propositions for the IntServ. However, only two of them have been standardized. Service classes are generalizations of the forwarding treatment - reflecting the nature of the communication for which they are designed.

3.4.1.1 Guaranteed Service

Traditional real-time service models have been designed based on two assumptions: first, traffic sources can be well characterized, and second, receivers require

[6] See Section 3.4.2 for information about resource reservation protocol (RSVP)

strict delay bound. When end system requests the traditional real-time service, it must characterize its traffic so that the network can make the admission control decision. As a result of admission control, a connection is either rejected, when there are not enough resources; or accepted, and the required resources are allocated to the connection.

If the allocation of resources is done with a single parameter, it is called a peak rate allocation, but if probabilistic loss and delay guarantees are accepted, it is called probabilistic allocation. Probabilistic allocation makes it possible to use benefits of the statistical multiplexing. These benefits, however, depend on the amount of the connections and their relative consumption of the resources.

Guaranteed service, since it is likely to be expensive, is suited for mission critical applications. Some examples are stock marketing and surveillance information delivery.

- Guaranteed Service

  Real-time service, which provides a hard or absolute bound on the delay for every packet and offers zero packet loss, is usually called guaranteed or deterministic guaranteed service. Guaranteed service is one of the standardized service classes in the IntServ [SPG97]. It is offered for applications which require absolute delay bound and/or zero loss probability. This service commitment leads to a low utilization of the network resources, if the traffic is bursty, and, therefore, is very expensive for the user. Managing of the guaranteed service is relatively easy because no multiplexing is done and no losses are allowed - there are no probabilities related to the allocation of resources.

- Probabilistic Service

  Probabilistic service is closely related to the guaranteed service with the exception that it allows controlled losses in the communications. Controlled means that the user admits some level of losses for which the network makes a commitment. In the probabilistic service, bandwidth for a new connection is not allocated based on the peak rate; rather, the allocated bandwidth is less than the peak rate of the source. Consequently, the sum of all peak rates may be greater than the capacity of the output link (this is called statistical multiplexing). Statistical allocation makes economic sense when dealing with bursty sources, but it is difficult to carry out effectively. This is because of difficulties in the characterization of the arrival process and the lack of understanding on how the arrival processes are modified deep in the network. This shaping is at this point a harmful phenomenon; it loses the controllability of the source traffic to the extent that resources have to be allocated very much like in the peak rate allocation. Probabilistic service should be a cheaper version of the guaranteed service for the user, because the user has voluntarily allowed the network to remove packets in the case of congestion. Depending on the amount of the multiplexing and allowed losses, the utilization of the networking resources may be high.

3.4.1.2 Predictive Service

Predictive service model [CSZ92] is based on the use of traffic measurements in the service allocation [Jam96]. Traffic is characterized by the traffic filter but the tolerance of the filter is allowed to be coarse. This coarse allocation is then corrected by traffic measurements made on the aggregate traffic. Because the measured information is always past information the predictive service gives less tight delay bound than the guaranteed service.

Load estimation, which is based on the measured information, allows conservative traffic specifications without lower network utilization in the end. Effect of the conservative traffic specification is decayed away, depending on the parameter values in the load estimator, quite quickly. The predictive service, thus, allows its admission control algorithm to admit more flows and to attain a higher network utilization. This higher utilization makes it possible to offer the predictive service at a lower price. Applications which are candidates to the predictive class are all interactive applications which allow guarantee to be broken occasionally.

3.4.1.3 Controlled Load Service

Controlled load service, which is the other standardized service class [Wro97a], is intended to support a broad class of applications which have been developed for the Internet, but are sensitive to the overloaded conditions. The concept of the controlled load service is to provide a level of service that is comparable to what would be achieved with best effort network which is not overloaded. The controlled load service offers guarantees that:

1. A very high percentage of the transmitted packets is successfully delivered to the end nodes[7].
2. Transit delay experienced by a very high percentage of the *delivered* packets does not greatly exceed the minimum transit delay experienced by any packet.

These guarantees and the service definition leave a lot of room for the implementation of the service. This is a clear advantage as it gives a lot of flexibility in the provisioning of the service. However, flexibility is also a disadvantage. Without clear formulation of the service commitment, received quality can be no better than offered by the DiffServ[8].

3.4.1.4 Best Effort Service

Best and 'better' effort classes are the plain old Internet and some extensions of it, such as local policy based networking, like the Differentiated Services, where

[7] This guarantee is not quantifiable but the quality of the transmission media is indicative for the guarantee.

[8] DiffServ implements the same level of service with considerably lower complexity

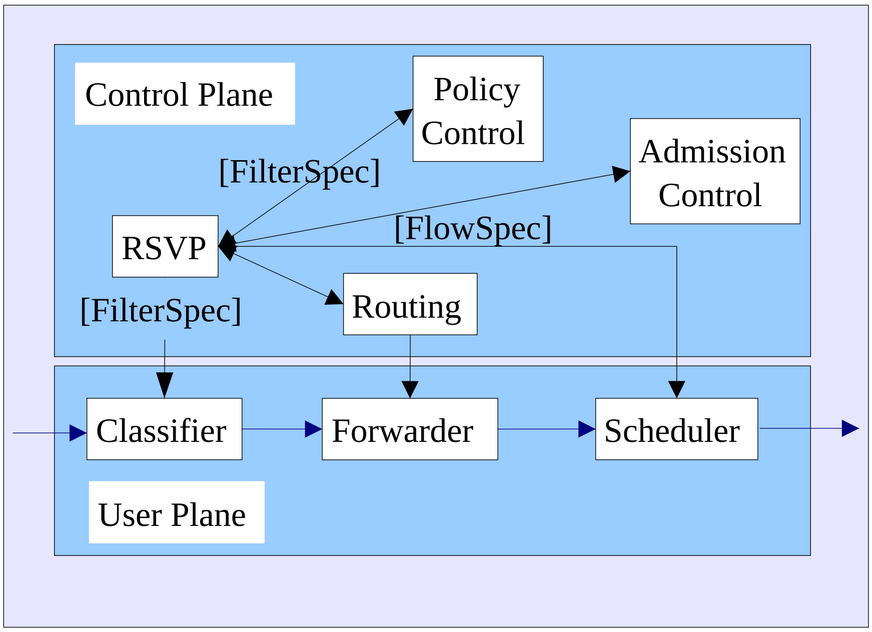


Figure 3.6: IntServ router, functional blocks

a filter separates traffic into a relatively small number of parallel traffic classes. These parallel classes get a share of the capacity on some controlled order.

3.4.2 Resource reservation

IntServ network uses resource reservations to make guarantees for the connections. These reservations are based on the soft-state[9] operation of the network. Mechanism that implements the reservation handling is the resource reservation protocol (RSVP)[BZB$^+$97, BO97, Wro97b]. RSVP activates the reservation state, which enables the connection-oriented service over the connectionless IP infrastructure.

RSVP reservation request contains a flow descriptor, which consists of two separate items: a flowspec and a filterspec. The flowspec specifies the traffic and desired QoS. Information of the flowspec carries expression of a service class and two sets of numeric parameters: Rspec, which defines the desired QoS, and Tspec, which describes the characteristics of the data flow. This information is used in the admission decision and in setting up the scheduler parameters. Filterspec is used to pinpoint the flow to which the flowspec should be applied by setting up the parameters in the classifier. This information is also used by the policy control to determine administrative rights for the resource reservation of the requesting user. The structure of an IntServ router and the signaling information used in various functional blocs are presented in Figure 3.6.

For multicast communication, RSVP holds also possibility to choose the reservation style. Reservation style expresses how the sending parties' traffic should be handled within the reserved session/flow. There are two dimensions of control over the reservation style. Sending parties can be explicitly specified or the defi-

[9] Look Section 2 for more information about states

nition of the senders can be left open. At the same time there is also control over the resource sharing. Resources can be dedicated to each sender individually or they may be shared among all senders. These two dimensions make possible four different operations (three of which are meaningful).

- **Fixed-Filter**: a distinct set of senders have each their own reservation of resources.
- **Shared-Explicit**: a distinct set of senders shares one reservation within the network.
- **Wildcard-Filter**: an unspecified set of senders shares one reservation.

3.5 Summary

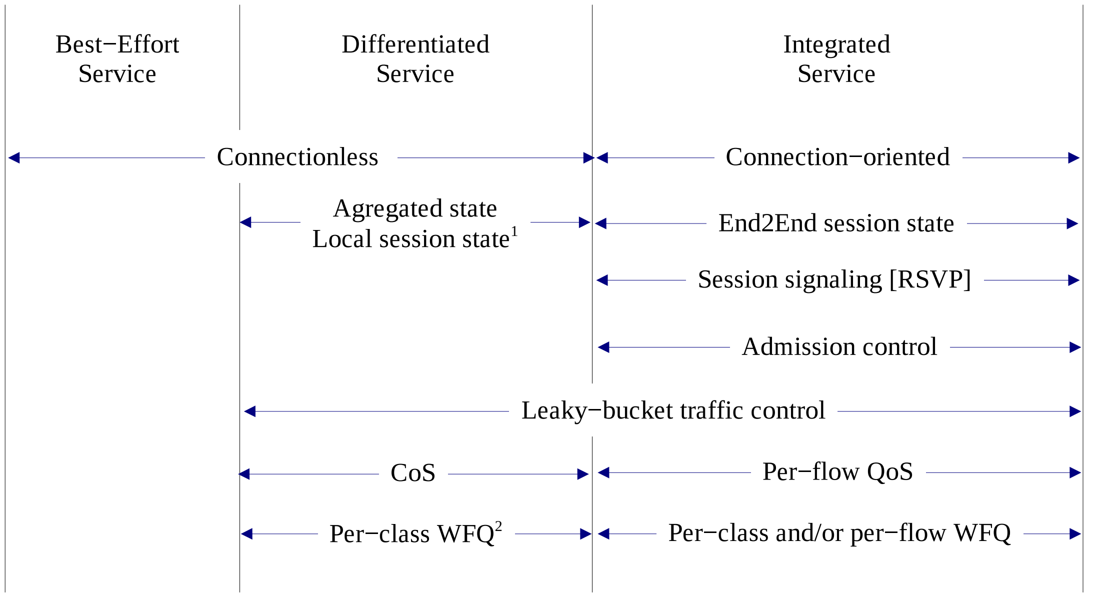
Internet is moving from a single service class, best effort, network to a multiple service class network. This is happening now through the introduction of services, which offer better service for a set of customers, primarily corporates in the form of VPNs.

There are two competing extensions to the best effort model.

One is the Differentiated Services, which offers aggregated traffic handling. Aggregated traffic handling operates on the level of service classes. All packets which are sent to the particular service class are handled similarly. This does not allow per flow quality guarantees, which may be necessary for the real-time applications, like IP-telephones.

The other extension to the best effort model is the Integrated Services, which offers a per flow traffic handling. This means that each flow is treated individually inside the network. This allows guarantees, which make it possible to operate any application in the Internet. However, there is also a price for doing such thing. This price is the complexity of implementation which comes from the per flow traffic management in the core network.

Despite of the differences of these two models they both aim at the same goal, offering the QoS to the users/applications which require it. It is probable that both of these models are needed in a long run, but for the moment Differentiated Services seems to provide sufficient amount of tools for the traffic management. Figure 3.7 presents key differences between the Best Effort, Differentiated Services and Integrated Services.



[1] Border routers may keep track individual sessions if required by policing or multifield classification.
[2] Scheduling depends on per hop behavior [PHB]. Minimum requirement is FIFO with multilevel RED.

Figure 3.7: Comparison of the key issues in Best Effort, Differentiated Services and Integrated Services service models

Chapter 4

Differentiated Services

Differentiated services (DiffServ) is a mechanism by which the network service provider can offer different levels of network service to the different traffic streams. In doing so, it provides quality of service (QoS) to the customers. Differentiated services is a service provider based concept which has significance inside a single provider's network. The foundation of the DiffServ network is that routers within the network handle packets from the different traffic flows by applying different per-hop behaviors (PHBs). The PHB to be applied is specified by the DiffServ code-point (DSCP) in the IP header of each packet (Type Of Service (TOS) field in IPv4, see Figure 3.5, and Traffic Class octet in IPv6).

Advantages of per packet coding are:

- Classification of the packets is done only once (in the access point of the packet to the network).
- Forwarding of the packets is based only on the class of packet (i.e. there is a limited number of different forwarding behaviors).
- There is neither signaling nor state based reservation of resources inside the network.

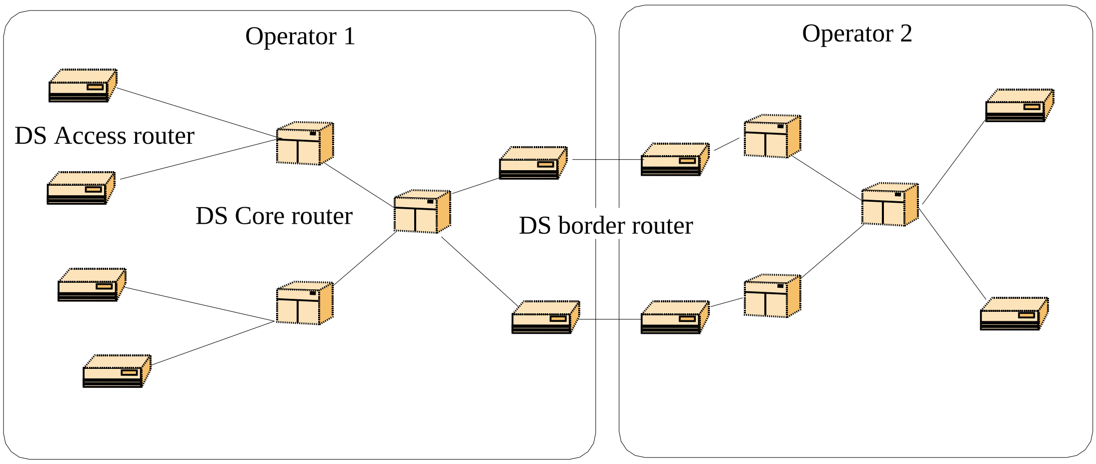


Figure 4.1: Differentiated Services based network domain and grouping of active components

DiffServ network has in general two types of devices, see Figure 4.1: access routers and core routers.

Access routers are responsible of preprocessing of the packet stream. This means policing and classification of the traffic into the proper service class. Access area has, in general, lower traffic volumes than the core network. Therefore, more time can be used for the packet conditioning in the access network than in the core network.

Core routers do high-speed forwarding based on the information coded in the DSCP. This minimizes the required processing in the core network and allows building high performance routers with a low cost.

The bottleneck of the DiffServ network is the interface between two network providers. Routers interfacing two providers need to perform the actions of the core router and the access router at the same time. They have to forward outgoing packets as fast as possible, while at the same time they have to police and classify the incoming traffic from the other network. These routers are called border routers and the way they actually do interoperability actions between network providers is yet largely open. It may become necessary to do DSCP translation into a form of special interoperability code-point. However, it is unavoidable that the border routers do both of the actions of the core router and the access router.

4.1 DiffServ architecture

Differentiated Services is a layer between the service model and the network infrastructure. It maps the service level agreement (SLA) to a conceptual network model, which is then applicable to the configuration of individual elements within the network. This requires translations from the SLA to a more detailed service level specification (SLS). Also a part of the SLA, the traffic conditioning agreement (TCA), is translated into the DiffServ specific traffic conditioning specification (TCS)[1]. TCS, which is still far from the device specific format of the configuration information, is translated to a form, which is understandable for the network elements (i.e. policy information base (PIB) and management information base (MIB)). This is illustrated in Figure 4.2.

The combination of the traffic conditioning at the ingress interface to the DiffServ network and the PHB treatment at the following interfaces constitute the DiffServ service. DiffServ service is domain specific and has no end-to-end meaning. If end-to-end services are pursued, each service provider has to use the same PHBs with the same policies and parameters. However, this is not likely to happen and, therefore, the DiffServ services are only meaningful within a single provider domain.

In order to support the DiffServ certain functionality is required from the access routers which reside at the ingress and egress points to and from the DiffServ network. Figure 4.3 illustrates the major functional blocks of the DiffServ router. Following sections, explain the logic and the operation behind these elements.

[1] For relation of SLA, SLS, TCA and TCS see Section 2.4

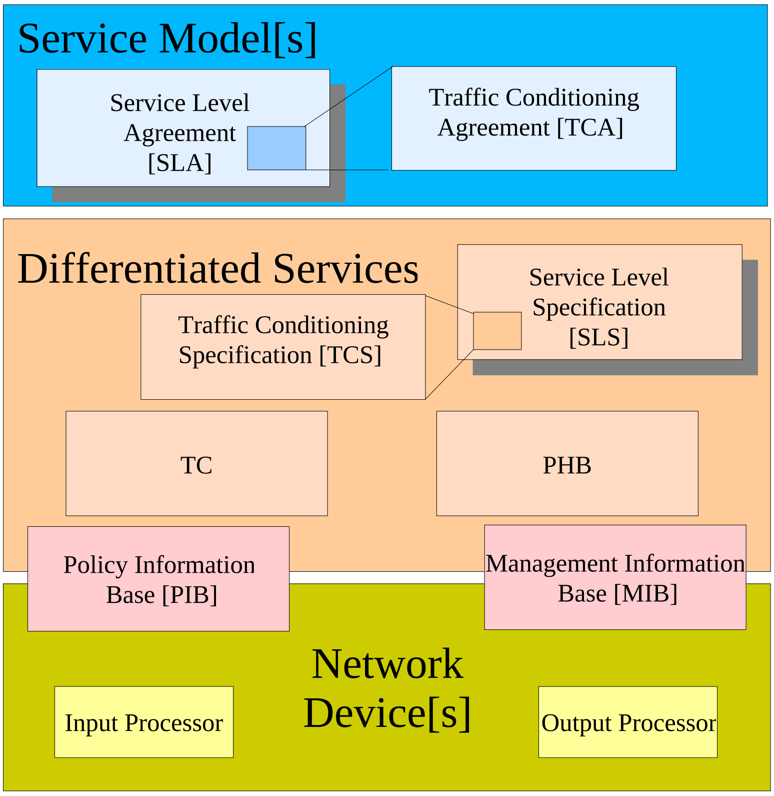

Service Model[s]
Service Level Agreement [SLA]
Traffic Conditioning Agreement [TCA]
Differentiated Services
Service Level Specification [SLS]
Traffic Conditioning Specification [TCS]
TC
PHB
Policy Information Base [PIB]
Management Information Base [MIB]
Network Device[s]
Input Processor
Output Processor

Figure 4.2: Differentiated Services, a layer between the service model and the network infrastructure

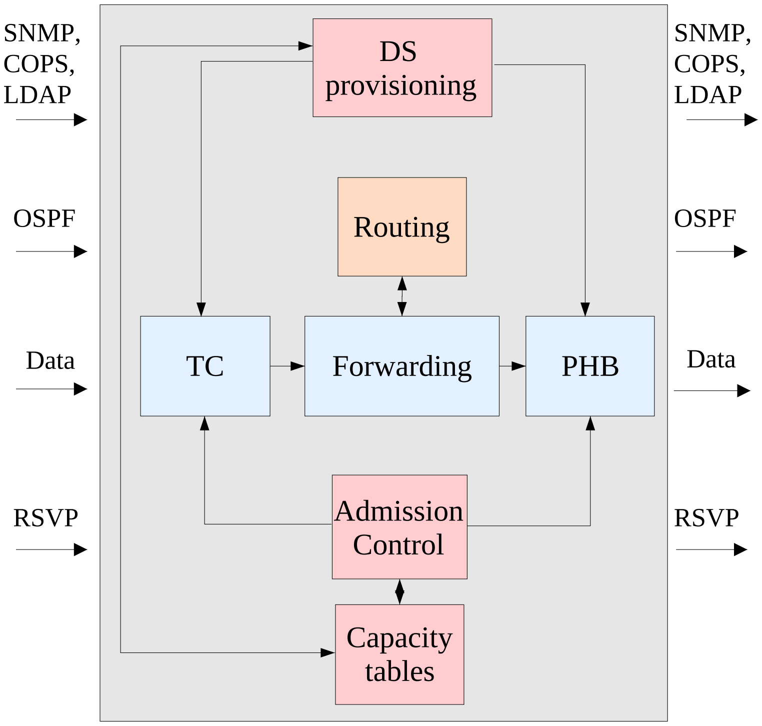


Figure 4.3: DiffServ router and information elements it transfers

4.2 Forwarding path of the DiffServ router

An input processor, forwarding component and an output processor form the forwarding path of the DiffServ router. The components of interest on these interfaces are the traffic conditioning (TC) on the input processor and the per hop behavior (PHB) on the output processor, see the middle section of Figure 4.3.

4.2.1 Traffic conditioning block

Traffic conditioning is a necessary function for any device that treats some traffic differently from the other traffic. The very nature of the DiffServ router is that it treats traffic in a differentiated way.

In the DiffServ, traffic conditioning has different meanings in access routers and core routers. In an access router, conditioning means enforcement of the traffic rules specified in the TCS. In core routers, conditioning means separating traffic streams for proper forwarding. Conditioning is executed by the Traffic Conditioning Block (TCB) with two sets of functional elements:

1. **Classifiers**. Classifier is a packet filter, which selects packets from the incoming packet stream according to predefined rules. These rules have two levels:

   (a) DiffServ code-point. Only the DiffServ code-point is examined to match the packet to some rule of the classifier. This type of classifier is called Behavior Aggregate (BA) classifier.

   (b) Arbitrary bit pattern. Arbitrary number of fields of the IP-header or some fixed location bit pattern is matched against the rules of the classifier. This type of classifier is called Multi-Field (MF) classifier.

2. **Traffic Conditioners**. Conditioner is a functional element which may contain sub-elements, like

   - Meters
   - Markers
   - Shapers
   - Droppers

   The function of the conditioner is to verify that the offered traffic is in compliance to the agreed profile. A meter is used to measure the rate at which traffic is being offered. This rate is then compared against the traffic profile, which is a part of the TCS. Based on the results of the comparison, meter judges particular packets to be either conforming or non-conforming to the profile. An appropriate marking action, such as:

   - marking
   - shaping

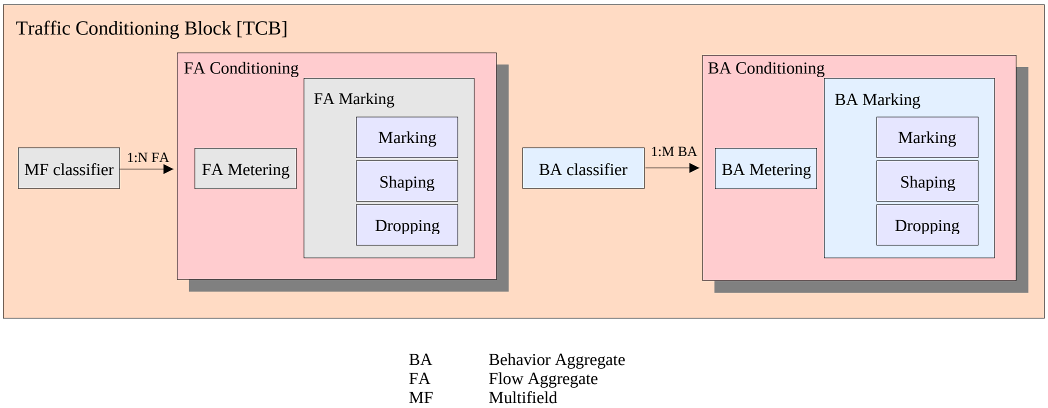


Figure 4.4: Traffic conditioning block with flow aggregate and behavior aggregate conditioners

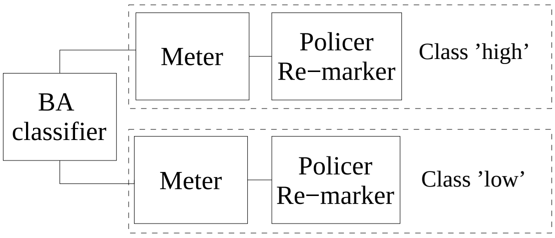


Figure 4.5: Behavior aggregate conditioner

- dropping

is then applied to these packets.

Two types of conditioners exist: Behavior Aggregate (BA) and Flow Aggregate (FA). BA conditioner is used for the inter-class isolation; i.e. it is not able to do fine grained traffic conditioning on the application and user level. For this reason, FA conditioner has been developed to assist the network provider in providing a value added service in the form of fine grained traffic conditioning.

4.2.1.1 Behavior aggregate conditioner

Behavior Aggregate (BA) conditioner consists of two elements: BA classifier and BA traffic conditioner.

BA classifier separates the packet stream into aggregates which correspond to individual service classes, defined by the DiffServ code-point. This functionality is all what is required from the core routers. In the access routers, these aggregates are then conditioned in the BA traffic conditioner. Traffic conditioning starts with metering the aggregate traffic stream. Metering result is compared against the traffic profile in the TCS. Based on the comparison, conformance status of the packet is decided. Non-conforming packets are re-marked for a lower service-level, shaped to conform to the TCS or dropped.

BA classifier suffers from certain limitations, it is only able to separate traffic based on the DiffServ code-point in the submitted packets. If traffic from multiple

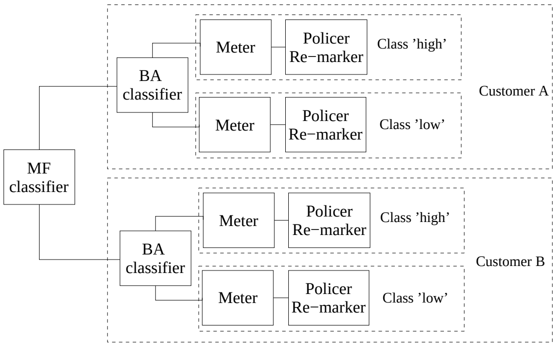
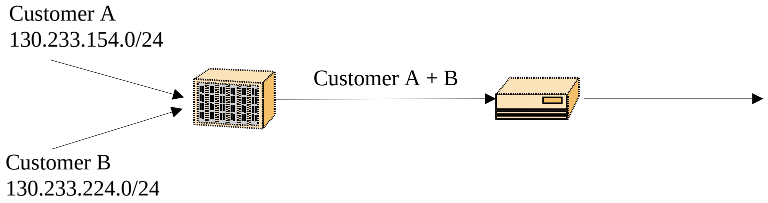


Figure 4.6: Flow aggregate conditioner



Network Address	Netmask	TCS
130.233.154.0	255.255.255.0	1
130.233.224.0	255.255.255.0	2

Figure 4.7: Filtering rules for MF classifier in example network

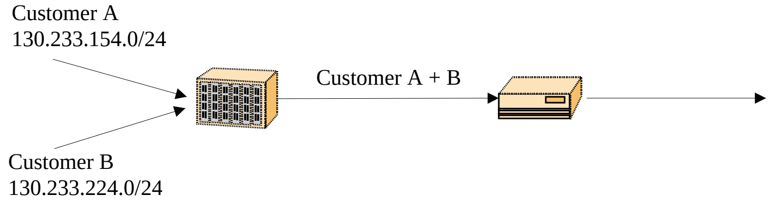
customers is submitted on the same interface, like in Figure 4.7, BA classifier will be unable to separate traffic by the customer. Since the TCAs are specified on a per-customer basis, TC components will be unable to select the appropriate TCS to be applied to an individual packet.

4.2.1.2 Flow aggregate conditioner

Flow aggregate conditioner is a finer granularity conditioner, which is able to overcome limitations of the behavior aggregate conditioner in the access point. The difference is in the classifier, which this time is a multi-field (MF) classifier. MF classifier is able to separate the traffic stream into substreams based on arbitrary filtering rules. These rules are either simple field recognition or logical combinations of several filtering decisions.

As an example, a MF classifier could be used to separate the traffic according to the customer and apply an appropriate TCS for each customer. However, traffic has to pass also the BA conditioner, which in this occasion controls isolation between the classes. To concretize our example; 'Customer A' has a class C network in the address space 130.233.154.0 and 'Customer B' has similar network in the address space 130.233.224.0. MF classifier in this case is a filter having a selection criterion of source address space, see Figure 4.7.

More complex filter configurations can enable value-added services such as provider marking, i.e. network provider does the fine-grained classification of packets for the customer based on some information, and provider shaping. In provider marking, network provider shapes the traffic in the real-time class to



Network Address	Netmask	Protocol	S-port	TCS
130.233.154.0	255.255.255.0	6	80	1.1
130.233.154.0	255.255.255.0	*	*	1.2
130.233.224.0	255.255.255.0	17	22555	2.1
130.233.224.0	255.255.255.0	*	*	2.2

Figure 4.8: Filtering rules for the added value MF classifier in the example network

conform the traffic contract in case of non-conformance. The essence of these services is that the customer relies on the network provider to apply per-flow processing to the customer's traffic. This requires the network provider to classify beyond the minimum level of granularity. This gives a means for the customer to use several applications requiring different type of QoS from the network without having to worry about the classification. To point these applications out some miscellaneous classification criterion is required at the access point of the network service. This criterion can be based on the TCP port address space of an individual application. To concretize this by our example; 'Customer A' has a class C network in the address space 130.233.154.0 from which he provides web hosting service. To maintain good quality for its own customers 'Customer A' has decided to invest on the added value service which gives 'high' quality for the connections from the web-servers (protocol:6 and S-port:80). 'Customer B' has a class C network in the address space 130.233.224.0. 'Customer B' uses IP-telephone application (Vocaltec) which uses UDP port 22555. These telephone conversations are vital for the business of 'Customer B' and he has decided to invest on the added value service which gives 'high' quality for the telephone traffic. MF classifier in this case is a filter having a criterion based on the address space and protocol/application space, see Figure 4.8.

4.2.2 Per Hop Behavior

DiffServ routers implement the per hop behaviors (PHB) that are used to forward the traffic of different service levels with differing behaviors. PHBs are generally implemented as queues, queue space management algorithms and schedulers that reside at the router's output processors, see Figure 4.9. Routers will generally provide support for a limited number of PHB groups. Supported PHB groups vary depending on media type, hardware support and software algorithms.

There are many proposed PHB groups, two of which have already been adopted as standards - namely the Expedited Forwarding (EF) and the Assured Forwarding (AF). Starting points of PHB propositions are somewhat different. However, they all aim to provide controlled operation and service discrimination based on the common policy. Here is a short review of the proposed PHB's.

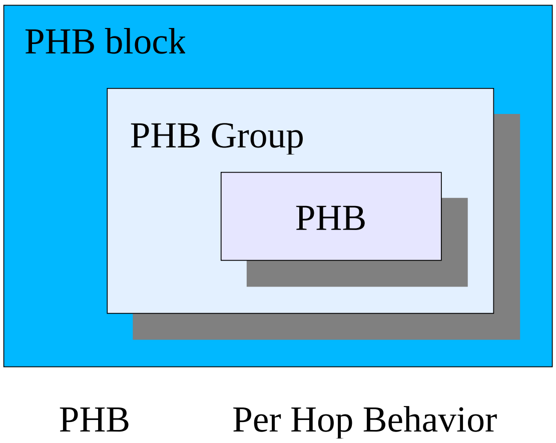


Figure 4.9: Structure of PHB

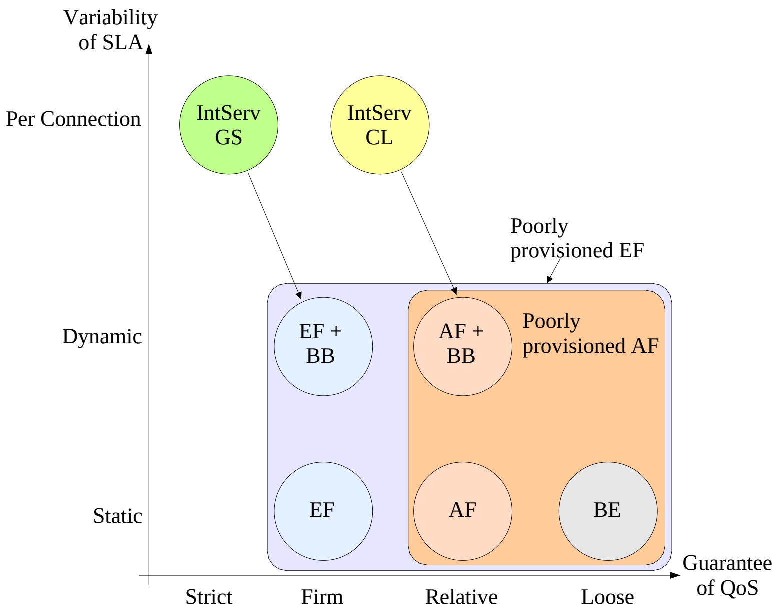


Figure 4.10: Goals for different PHBs and comparison to IntServ service classes

4.2.2.1 Expedited Forwarding

Expedited Forwarding (EF) PHB [JNP99] is intended to build a low loss, low latency, low jitter, assured bandwidth end-to-end service that appears to the endpoints like a virtual leased line. Comparison to the leased line holds also for the connection structure and provisioning time frame of the EF service. Provisioning of the EF service is done on point-to-point manner, meaning that there is a distinct ingress and egress point for the connection. Point-to-multipoint connections are constructed similarly; they have a fixed root and leave structure. EF service is considered to be rather static from the provisioning point of view, connections are normally provisioned for a days or months at a time. However, there are occasions when more dynamics is required, like in the case of IntServ/DiffServ interoperability service. This requires external mechanisms to adjust the DiffServ domain properties and/or to reject excessive connections. These mechanisms are admission control and/or bandwidth broker (BB), see Section 4.3.1.2.

Ensuring low delay, jitter and loss means that the traffic in the EF class sees no or very small queues in the network. Ensuring small queues is equivalent to the bounding of incoming rate of the class aggregate to the departure rate of the class on the route through the network. However, DiffServ does not contain mechanisms for this, unless RSVP is implemented to do this task. What this means is that successful provisioning of the EF service requires fixed pipes through the network with control only at the network edge (ingress point of the pipe). DiffServ has no route pinning, i.e. no fixed routing over the lifetime of the connection; this is due to the connectionless and stateless operation. This makes the previous idea of fixed pipes problematic, as route changes are possible. Due to these restrictions and tight requirements of the service, it is not foreseeable that the EF will be the only PHB in the network. Rather a small percentage of the traffic is likely to be delivered in the EF PHB. This allows the EF traffic to receive their contracted service, even in highly variable routing. However, there have to be mechanisms which take care that the EF traffic does not preempt resources of other service classes (by accident or intentionally). This requires policing of the traffic at the access point of the network and rate limitation of the EF class with some scheduling algorithms, like strict priority queuing.

4.2.2.2 Assured forwarding

The Assured Forwarding (AF) PHB [HBWW99] group provides delivery of IP packets in four independently forwarded AF classes. Within each AF class, an IP packet can be assigned one of three different levels of drop precedence. There are no quantifiable timing requirements associated with the forwarding of packets within the AF PHB. A DiffServ node is not allowed to reorder IP packets of the same microflow, if all packets within a microflow belong to the same AF class. This means that packets are processed and forwarded in the order they arrive at the network, irrespective of their drop precedence in the queue. In fact this prohibits the use of drop precedence ordered queuing, which was one of the early ideas behind the type of service (TOS) field definitions in the IP packet header[2].

[2] More information about the TOS definitions is given in Section 3.2

Packets in one AF class are forwarded independently from packets in other AF classes (full isolation between the classes). Queue space management is assumed to be based on a RED-like algorithm[3] with individual threshold levels for each precedence in the class. This guarantees that packets within a single class preserve their ordering.

Provisioning of the AF service is not clear in many cases. As there is no clear end-to-end semantics on this service, it makes difficult to predict the quality delivered by this service. Actual quality level depends largely on the service model, which is build upon the AF service. If the service model is based on the wildcard filter specification, it means that distribution of the traffic and the load level within the network is not predictable. This makes it hard to realize the benefits of this service. Number of independent classes, which can be four, makes it possible to offer a wide range of services on top AF. However, at this point there is no clear idea what these services could be. Therefore, it is probable that only one or two (one for the non-real-time traffic and one for the real-time traffic) classes will be implemented.

4.2.2.3 Dynamic RT/NRT PHB Group

Dynamic RT/NRT (DRT) PHB group [LRK98] is based on the SIMA model [RK99]. DRT has two service classes, both having six different precedence levels. Precedence level for each packet is calculated independently as a function of the instantaneous bit rate and nominal bit rate (NBR), i.e. subscribed rate. As the DRT is a PHB group, it means that the real-time and the non-real-time classes are two separate PHBs. This means that there has to be at least two capacity subscriptions: one for the real-time class and another for the non-real-time class.

Provisioning of the DRT service is assumed to be done in a way that, under reasonable operating conditions and traffic loads, packets in the real-time class will have smaller delays and delay variations than packets in the non-real-time class. Even in a highly congested network the delay of the delivered packets (note that not every packet gets to be delivered under congestion) in the real-time class should not exceed the delay under normal networking conditions. This means that the size of the buffer in the real-time class is negligible compared to the non-real-time class. However, DRT is not quantifiable service class definition. This means that difference in the delay between the real-time class and the non-real-time class is not quantifiable. It is presumed that having a buffer small enough and hard priority scheduling between the classes satisfies previous condition without any numerical requirements.

Implementation of the DRT PHB group can be based on the FA conditioner and/or the BA conditioner. If there is only separation between the real-time and the non-real-time service classes, the implementation can be based only on the BA conditioner. BA conditioner has a metering process which measures the user traffic and does the calculation of the precedence level based on the logarithmic ratio of the instantaneous rate and NBR. However, if more isolation is required then the FA conditioner is needed to do finer granularity flow separation. At this

[3] RED algorithm is explained is Section 5.3.1

point, the FA conditioner uses the same mechanisms but with multiple (different) NBR values. NBR values for each flow group (application space) are agreed in the SLA. However, a BA conditioner is used after the FA conditioner to make sure that the total traffic emitted is within limits of the class or user provisioning. Provisioning of the DRT service presents similar problems as the AF service. However, DRT has some benefits over the AF. The biggest of the benefits is the proportional sharing based service. Proportional sharing means that resources are divided on a congested link systematically based on the information carried in the DSCP, i.e. service class and precedence. This gives better fairness in comparison to random resource division of the AF PHB group.

4.2.2.4 Class-Based Service Differentiation

The differentiating factor in the Class-based Service Differentiation (CBSD) [Dov98] is the relative forwarding quality in the DiffServ capable router. CBSD refers to the relative service level, which should be interpreted only locally in the statistical manner in terms of traffic load, queuing delay or packet drop probability. Relative performance constraints of the classes are not to be quantitatively defined. Network operators can attach different performance ratings to the classes depending on the implementations and traffic management policies they use.

In CBSD, the network makes the rules, by defining a service level hierarchy which is general, simple and application independent. User and/or application match their needs of service to the parameters of a class based on the performance and cost constraints they have. Basic implementation consists of eight classes with relative order so that in the long run each class has at least the same service level it had if it were processed in a FIFO queue in a non-DiffServ capable router[4].

4.3 Configuration and provisioning of DiffServ service

4.3.1 Intra-domain configuration and provisioning

4.3.1.1 Intra-domain configuration

Configuration of the Differentiated Services routers is done through the policy information base (PIB) and the management information base (MIB) with protocols such as simple network management protocol (SNMP), local directory access protocol (LDAP) and/or common policy protocol (COPS).

Configuration information is a translation of the DiffServ service and traffic conditioning specification to a device specific format. Items that require configuration are:

[4] The question is whether each class can have better service than in FIFO. One can argue that if some classes are served with higher priority than other classes, these other classes will receive service which is lower than in the case where all of the classes were served with the same priority.

1. **Filters**, specify the criteria which classifiers use in classifying the submitted packets.

    - BA filters: a six-bit DiffServ code-point.
    - MF filters: arbitrarily complex, specifying multiple classification fields and corresponding masks.

2. **Profiles** of the user traffic specify criteria which are used in marking of the submitted packets.

    - Set of token bucket[5] parameters reflecting the contracted packet rate of the user. When configuration information is related to the FA conditioning, profile information is per application or flow. Otherwise, information is related to the BA conditioning which reflects the total capacity that is admissible to each class by the particular user.

3. **Schedulers**, quantitative parameters associated with the inter-class scheduling.

4. **Queues**, quantitative parameters associated with the intra-class handling.

4.3.1.2 Intra-domain provisioning

In the most basic form, the intra-domain provisioning is static. Static refers to the situation where the rate of new SLA/SLS subscriptions is so small that configuration of the network can be accomplished by manual intervention of the network operation center (NOC). However, this may not be a long run solution, as is expressed in [THD$^+$99]. There may be some dynamics in the negotiation of the SLA/SLS that requires an automate to adjust the configuration of the network devices. The speed of the SLA/SLS variation and thus required signaling for the configuration changes limits the scalability of the DiffServ. In the worst case, if every flow has its own SLA/SLS, scalability falls down to the level of the IntServ. The component which makes the dynamics possible is the bandwidth broker (BB). Bandwidth broker is an item which translates the user SLA/SLS change request to a network understandable form and passes the information to the network with protocols presented before.

Intra-domain operation requires that each subnet has an item called subnet bandwidth manager (SBM), which operates as a reservation agent between the user and the bandwidth broker. SBM receives resource requests from the hosts with different protocols (RSVP) and translates them to the form of resource allocation request (RAR). SBM passes the RAR to the bandwidth broker, which processes it and configures the network devices based on the new distribution of service classes in the network.

[5] For information about token bucket metering, see Section 5.4.2

4.3.2 Inter-domain configuration and provisioning

4.3.2.1 Inter-domain configuration

Inter-domain operation of the Differentiated Services does not need any additional configuration functionality compared to the intra-domain case. Same conditioning and forwarding treatment adjustment is required in the case of changing the SLA/SLS as in the local case. However, some extra functionality is required for the accounting purposes. These functions are still under research and the future will tell what direction the development will take.

4.3.2.2 Inter-domain provisioning

Provisioning of the inter-domain service can also be static as in the local case. However, at this point the end-to-end nature of the QoS is lost. There is only limited statistical guarantee for the QoS that a single user receives, when only aggregates are used in the provisioning. This is, however, a tractable way to provision the network as it requires minimal co-operation between two service providers. Statistical accuracy of guarantee is lower and lower as the communication path travels through several service providers. Cascaded mixing of the local traffic with the inter-provider traffic leads to a service which cannot be characterized in a parametrized way, only relative behavior can be expressed (even that on a limited level).

A more firm service provisioning of the inter-domain communication can be achieved if a special PHB is used for this purpose. This PHB is called 'Interoperability PHB Group (PHB-i)' [KR99]. In the PHB-i common metrics are used on the border of two domains, so preference values of the packets are clearly expressed in the form of 'temporary DiffServ code points'. This makes it possible to transfer some information about the forwarding treatment requirements of the packet from one domain to another without the need to communicate properties of a single domain to all other domains.

Dynamic inter-domain provisioning requires that bandwidth brokers are able to communicate with each other and form a chain of coherently operating mechanisms in order to provide the QoS service in an end-to-end manner. Level of guarantee and the price the user is willing to pay for the service limits the operation of the BBs. As the DiffServ is intended to networks with relative QoS and high flexibility, a large amount of freedom in the operation of inter-domain reservation is possible. Issues related to the granularity of the control between domains are studied in [GB99].

4.3.3 IntServ/DiffServ interoperability

One way to provide scalable end-to-end services without excessive signaling in the core network is to use the IntServ in the local subnet and the DiffServ in the service provider network. This, however, requires that local subnet is fully IntServ compliant and that the access router has IntServ/DiffServ translation

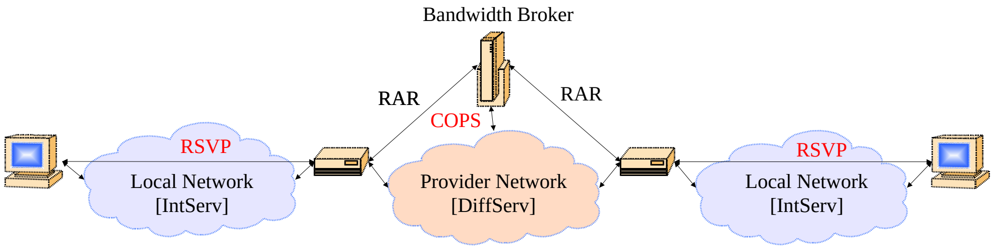


Figure 4.11: Provisioning of resources on the IntServ/DiffServ interoperability

capabilities. Translation of the IntServ to the DiffServ separates the control and data traffic into logically two different networks. Data traffic is passed as regular DiffServ traffic without any distinction, whereas control traffic is passed to the bandwidth brokers administering the DiffServ cloud, see Figure 4.11. Quality and performance metrics of the DiffServ core with the IntServ access are analyzed in [MMPV99].

Chapter 5

Implementation of the Differentiated Services

Implementation of the Differentiated Services requires functionalities that fulfill the tasks presumed by the conceptual model of the Differentiated Services. These functionalities are in short:

- Admission control
- Scheduling
- Queue space management
- Policy control

5.1 Admission Control

Differentiated Services is based on the assumption that there are no connection-oriented services and thus no admission control. However, usage of the admission control is left as an option. By now, it seems that RSVP and admission control are far away from the spirit of recommendations made by the IETF.

Admission control is related to the connection-oriented networking, see Section 2.1.1 and Section 3.4 for more information about the connection-oriented networking and the Integrated Services service model.

Admission control is a process of deciding whether a connection can be accepted to the network. This decision is based on the requirement that the new connection may not deteriorate the quality of the connections already in progress. A tightly coupled element with the admission control is the resource allocation that sets the reservation attributes to the actual forwarding path elements. In the Internet systems and protocols these two functions are usually thought to form one combined unit, see documentation of the RSVP [BZB$^+$97, Wro97b].

The driving force behind the RSVP is the fact that multimedia communication requires constant quality from the network and thus is best served if a connection is set between the end points. Regardless of whether the overall concept is RSVP or something else, the actual admission algorithm is usually left open for the implementations. A variety of different admission control algorithms for different packet network protocols (ATM, IP, DQDB) have been proposed in the literature [PE96, KS99b, RM98, LS99, Fra98, JSD97, JDSZ96, Flo96]. Some of these schemes require an explicit traffic model, some only require traffic parameters such as the peak and average rates and some rely on measurements taken out of the network.

However, as traffic profiles are hard to estimate, the ability of the admission control algorithm to secure the connections and the same time maintain high utilization in the network is somewhat diminished. If only a fraction of the traffic is guaranteed, conservative reservation of resources does not lower the overall utilization of the network.

5.2 Scheduling

In the Differentiated Services, the scheduler is the element which executes the inter-class resource division. Scheduling is the task of deciding the order of service for individual packets in different queues. Thus, scheduler executes the time priority based QoS differentiation policy of the network service provider. Scheduling is similar to the time sharing in the CPU of a computer system [SG94]. Following sections will briefly explain the difference between common scheduling algorithms and their relation to the Internet systems[1].

5.2.1 First come first served scheduling

First come first served (FCFS) is the prevalent scheduling scheme in the Internet routers. It utilizes only the information about arrival time of a packet in determining the scheduling order, see Figure 5.1. FCFS scheduling is aviable scheduling scheme for one queue and homogenous QoS requirements cases. If some differentiation is aimed for, either in the case of multiple queues or in the case of heterogeneous QoS requirements, the FCFS may not be a good choice. In the case of heterogeneous QoS requirements, the FCFS is not able to do delay differentiation, which makes the operation of real-time applications impossible. This is due to large variation in delay, caused by the other traffic. However, when more than one queue is used, the FCFS can be used within each queue and some other scheduling between the queues.

5.2.2 Priority scheduling

Priority scheduler serves packets based on their priority. All of the packets from the highest priority class are served before any packets are served from the lower priority classes. Packets with equal priority are served primarily on the FCFS order and secondarily, if they happen to arrive at the same time, based on some

[1] Complete presentation of scheduling algorithms and their differences can be found in references [Pin95, Zha95].

Class	Arrival Time	Service Time
C1	0	1
C2	0	3
C1	3	1
C3	3	3
C2	3	2
C1	5	1
C3	7	3

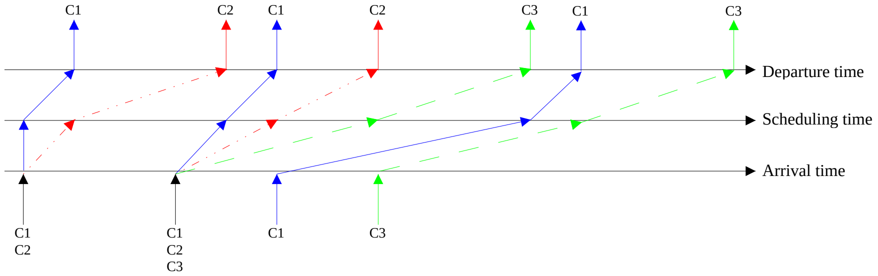


Figure 5.1: Gantt diagram for the FCFS scheduler

Class	Arrival Time	Service Time
C1	0	1
C2	0	3
C1	3	1
C3	3	3
C2	3	2
C1	5	1
C3	7	3

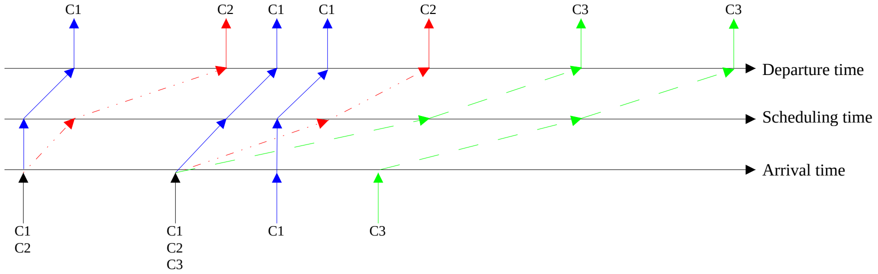


Figure 5.2: Gantt diagram for the priority scheduler

other criterion (for example shortest packets first). See Figure 5.2 for the Gantt diagram of the priority scheduler with two classes of the traffic.

Priority scheduling, if used carelessly, can potentially starve resources of the low priority traffic. Therefore, resource usage must be limited within each priority level through access policing.

In the simulations of this study priority scheduling is implemented as Figure 5.3 shows. There are three queues, one for each class. Queues are served in the order of EF, AF and BE. Due to the almost instant service for the highest priority class of the traffic, it is able to provide best jitter and delay characteristics for the real-time traffic.

Actual implementations of the priority scheduling algorithms vary a lot. Performance of the algorithms depends crucially on the implementation of the search algorithm used for the sorting of the packets for the service. Therefore, the results of this simulation study are not universally applicable. However, they give relative performance figures between different scheduling mechanisms.[2]

[2] Performance aspects of the different priority scheduling algorithms are evaluated in depth in the reference[RA97]. One should consult this reference about the limitations of the actual implementations of the priority scheduling.

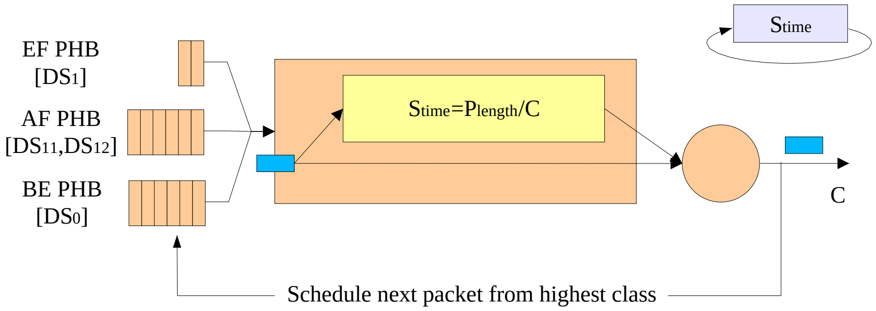


Figure 5.3: Implementation of the priority scheduling in this study

5.2.3 Processor Sharing scheduling

Processor sharing (PS) or Fair Queuing (FQ) [Nag85, Kes91] is an ideal scheduler in which the server resource is equally shared between all of the traffic classes. PS and other fair share schedulers [SV96, SSZ98, SZN97, BZ97, GVC97] are continuous time schedulers. This means that they use virtual finishing time in decision of which packet should be transmitted next. Virtual finishing time is the time when the packet would be sent out if the server capacity were equally divided among the served packets. In reality, packets are sent out as whole items, so they are scheduled in the order based on their virtual finishing time.

5.2.4 Weighted Fair Queuing scheduling

Weighted fair queuing (WFQ) [DKS89] is a modification of the fair queuing algorithm. It has a weight associated with the fairness criterion of a class. This makes it possible to offer a different amount of resources to the different classes. In the WFQ, resources that are left over by some class are divided between backlogged classes in the proportion of their weights. WFQ is still a continuous time scheduling algorithm and implementations of it are only approximations of the ideal case. Implementations are based on the packetized generalized processor sharing (PGPS) which is extensively studied in [PG94].

5.2.5 Round Robin scheduling

Round robin (RR) scheduling is designed for the time sharing systems, where a small slice of time is assigned for each transaction. These slices are then rotated over and over in order to accomplish the task, see Figure 5.4.

In a router this means that scheduling time is divided to each class so that each class is given service a small slice of time at the time (during single rotation). If this time slice (TS) is not equal between the classes, the scheduler is called weighted round robin (WRR) scheduler. Weight sets administrative constraints to the resource division during resource contention. Implementation of the WRR in this simulation study uses the half of the transmission time of a small packet (64 Bytes) on a given link as a rotation time.

Class	Arrival Time	Service Time
C1	0	1
C2	0	3
C1	3	1
C3	3	3
C2	3	2
C1	5	1
C3	7	3

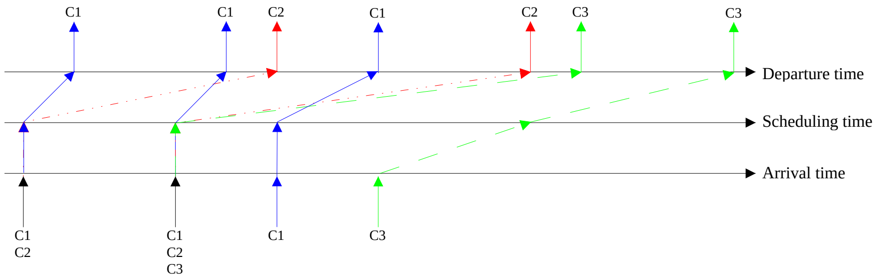


Figure 5.4: Gantt diagram for round robin scheduler

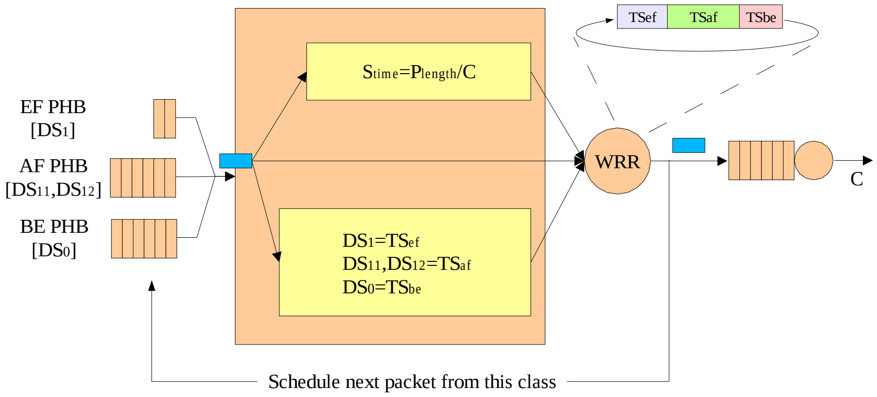


Figure 5.5: Implementation of the WRR in this study

$$TS = 0.5\dot{(}64Bytes/LinkSpeed) \tag{5.1}$$

This time slice is the maximum which a single class can get in one rotation, implying that every packet is served in multiple rotations. This study has neglected the effect of switching, which in the real systems will deteriorate the performance if the time slice is selected to be too small.[3]

Analysis of the fairness and implementation of RR schedulers are reviewed in [Hah91].

5.2.6 Class based queuing

Class based queuing (CBQ) [FJ95, FS98, Flo95] is a popular solution for the implementation of the scheduling of PHBs in the DiffServ architecture. Operation of the CBQ is based on two elements each of which has various sub-elements:

1. **Traffic Classes**, see Figure 5.6.

[3] If time slice equals a transmission time of a single bit and the effect of switching is neglected, RR scheduler is approximation of the PS scheduler and the WRR scheduler is approximation of the WFQ scheduler

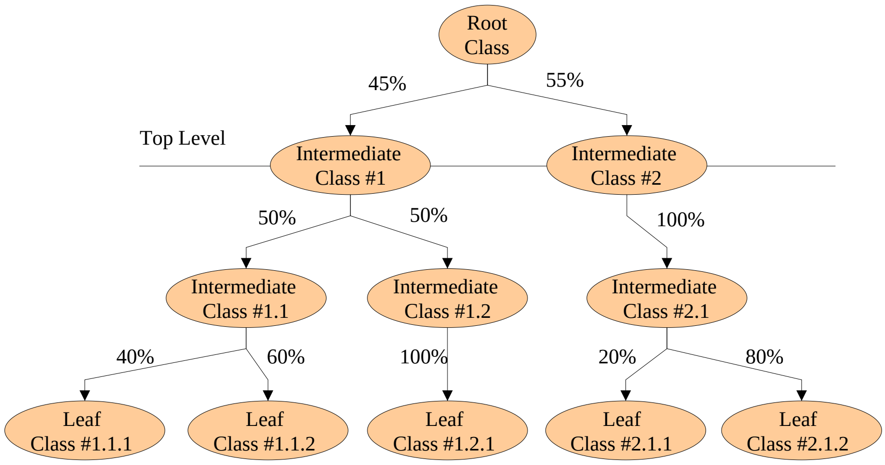


Figure 5.6: Structural tree representation of the CBQ scheduler with initial link share percentiles

(a) **Root Class** is the link resource. It contains all of the resources which are divided in the CBQ system.

(b) **Leaf Classes** are actual traffic classes which are served by the CBQ system.

(c) **Intermediate Classes** are middle layers in the resource partitioning. These classes are used to construct rules by which resources are divided to the leaf classes. They act as parents, from which resources are inherited or partitioned, for the leaf classes, and as leafs for the root and other higher layer intermediate classes.

2. **Schedulers**

(a) **General Scheduler** does all the work when all of the leaf classes retain enough resources to satisfy the traffic flows, which they serve. However, when one of the leaf classes gets congested, i.e. pathological queue starts to build, a link share scheduler gets activated.

(b) **Link Share Scheduler** does the work when at least one of the leaf classes is congested. Congestion this time does not mean that the link is congested. Link may have enough resources but initial resource division between the leaf classes has caused one or more classes to consume all their resources. On these occasions link share scheduler uses other rules to define a new policy for the resource division. This policy is based on the resource usage on the parent class and possibly on the other leaf classes.

Rules for the link sharing of the link share scheduler vary depending on the service which is offered on the network and on the structure of link sharing tree.

- **Ancestor only link sharing** offers to the unhappy class a possibility to increase capacity as long as there is a capacity left from its parents. This means that resources are divided from the root to the intermediate classes based on some aggregation rule (i.e. protocol, organization, service, and

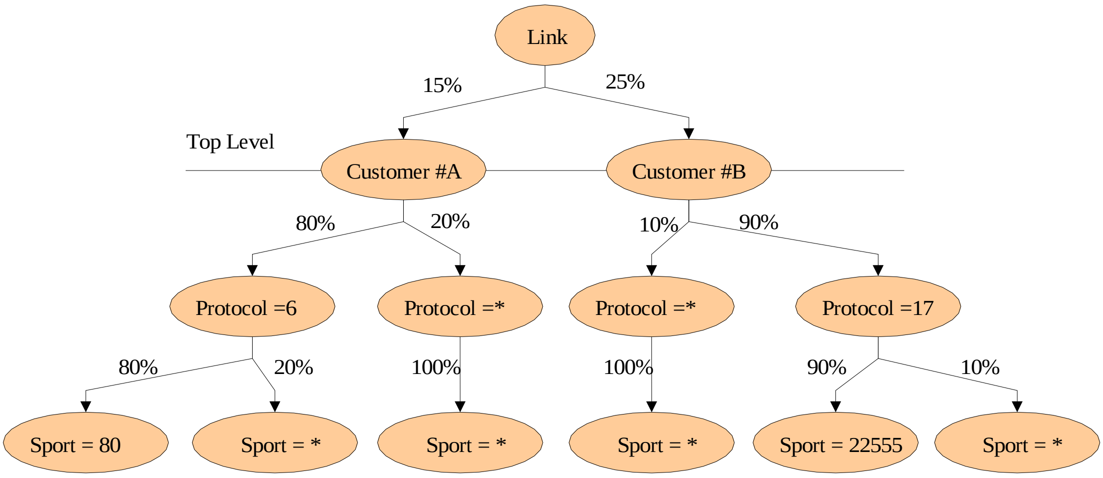


Figure 5.7: Link sharing rules for example service provisioning

PHB group). Intermediate classes divide their resources to a number of leaf classes, which are formed with similar aggregation rules from the parent class. Single leaf is allowed to use resources as long as there are resources available on the root level.

- **Top level link sharing** modifies the ancestor only link sharing by adding a parameter which reflects how many steps this 'borrowing' of capacity can go up in the tree, see Figure 5.6. In the link sharing structure, where there are many levels, a borrowing of bandwidth may be limited to some top level after which all of the branches have to be satisfied before borrowing can proceed towards the root.

To concretize the operational aspects of the CBQ, we will return to our example of a service provisioning scenario presented in Figure 4.8 of Section 4.2.

The link from the access router is shared between 'Customer A' and 'Customer B' so that 'Customer A' has bought 5Mbit/s and 'Customer B' has bought 8,5Mbit/s. Network provider has installed 34Mbit/s link to the access router. This gives for 'Customer A' 15% of link resources and 'Customer B' 25% of link resources. In the SLA of 'Customer A' network provider has agreed to do provider marking for traffic so that web traffic will have at least 3,2Mbit/s capacity all the time. Similarly the SLA of 'Customer B' contains agreement for Vocaltec phone to receive at least 7,5Mbit/s capacity. This can be done with CBQ based scheduling which follows the rules presented in Figure 5.7.

Implementation of the general scheduler in the CBQ can be freely chosen as long as it is capable of doing resource division between the leaf classes. However, many of the implementations use weighted round robin (WRR) and packet per packet round robin (PRR) due to their relatively low complexity in the computation compared to the virtual finishing time schedulers (PS and WFQ).

Implementation of the link share scheduler is conceptually simple, but algorithms used to implement resource division are far more complex than the general schedulers. This is due to flexibility and great variability in policies to be forced during the congestion. This does not leave room for an easy and optimized solution for the implementation.

5.3 Queue space management

The other task of queue management algorithms is to manage buffer space in a controlled manner. Buffers are used to accommodate *transient overloads*, contentions, in the links. In cases when the contention is a more permanent phenomenon, packets are lost due to the shortage of buffer capacity. Special algorithms to deal with these situations have been developed. They aim at providing a controlled operation with assured level of fairness - even in the case of unresponsive flows.

5.3.1 Random Early Detection

Random Early Detection (RED) is the de facto queue management algorithm in the Internet [BCC$^+$98]. RED was developed to reduce the global synchronization[4], maintain higher sustained throughput and to provide better fairness.

RED provides a queue management which aims at providing congestion avoidance by controlling the average queue size. This control is done through properly parameterized discard algorithm, see Figure 5.8. This algorithm is based on the average queue size, calculated by the exponentially weighted moving average (EWMA) algorithm, linearly increasing drop probability (see Figure 5.9) and random packet dropping.

5.3.2 RED In/Out

RED In/Out (RIO) is a twin algorithm implementation of the RED. One algorithm is used for the high priority traffic (*In*) and the other one for the low priority traffic (*Out*). High and low priorities are set by the co-operating conditioning function at the ingress of the network. As there are two algorithms running in parallel, there are more parameters to be set. Possible combinations are given in Figure 5.11 [5]. Calculation of the average queue size (*avg*) is separated into two cases:

1. Arriving *In* packet is counted in both classes
2. Arriving *Out* packet is counted only in the *Out* class

[4] Global synchronization is a result of concurrent loss of packets from many flows traveling through a congested router. TCP reacts to the loss of multiple packets by dropping the window size to one, thus effectively halting the communication. While this is done concurrently in many TCP clients, the load in the bottleneck falls down to a low level. The opening of the transmission window, though depending on many variables, tend to synchronize and lead the network to the oscillation between a very low and full utilization.

[5] See also [CF98] for a discussion of the parameters and their settings.

```
Initialization:
$avg = 0, \quad count = -1$

For each packet arrival:

  calculate the new $avg$:

    if $q > 0$
      $avg = (1 - w_q)avg + w_q q$

    else
      $m = f(time - q_{time})$
      $avg = (1 - w_q)^m avg$

  if $min_{th} \leq avg < max_{th}$

    increment $count$

    calculate probability $P_a$:
      $P_b = max_p(avg - min_{th})/(max_{th} - min_{th})$
      $P_a = P_b/(1 - count \cdot P_b)$

    with probability $P_a$:
      mark arriving packet
      $count = 0$

  else if $max_{th} \leq avg$
    mark arriving packet
    $count = 0$

  else
    $count = -1$

When the queue becomes empty:
  $q_{time} = time$
```

Figure 5.8: RED algorithm [FJ93]

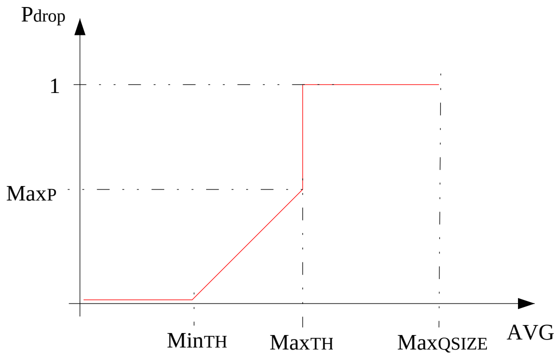


Figure 5.9: Drop behavior of the RED algorithm

```
For each packet arrival:

  if In packet
    calculate the new $avg_{in}$
    calculate the new $avg_{total}$

  else
    calculate the new $avg_{total}$

  if In packet

    if $min_{in} \leq avg_{in} < max_{in}$
      calculate probability $P_{in}$
      with probability $P_{in}$ :
        mark arriving packet

    else if $max_{in} \leq avg_{in}$
        mark arriving packet

  if Out packet

    if $min_{out} \leq avg_{total} < max_{out}$
      calculate probability $P_{out}$
      with probability $P_{out}$:
         mark arriving packet

    else if $max_{out} \leq avg_{total}$
        mark arriving packet
```

Figure 5.10: RIO algorithm [Fan98]

Based on this, dropping of packet is decided and executed as in the RED with the exception that there are two algorithms in parallel, one for the *In* and other for the *Out* packets (see Figure 5.10).

5.4 Conditioner

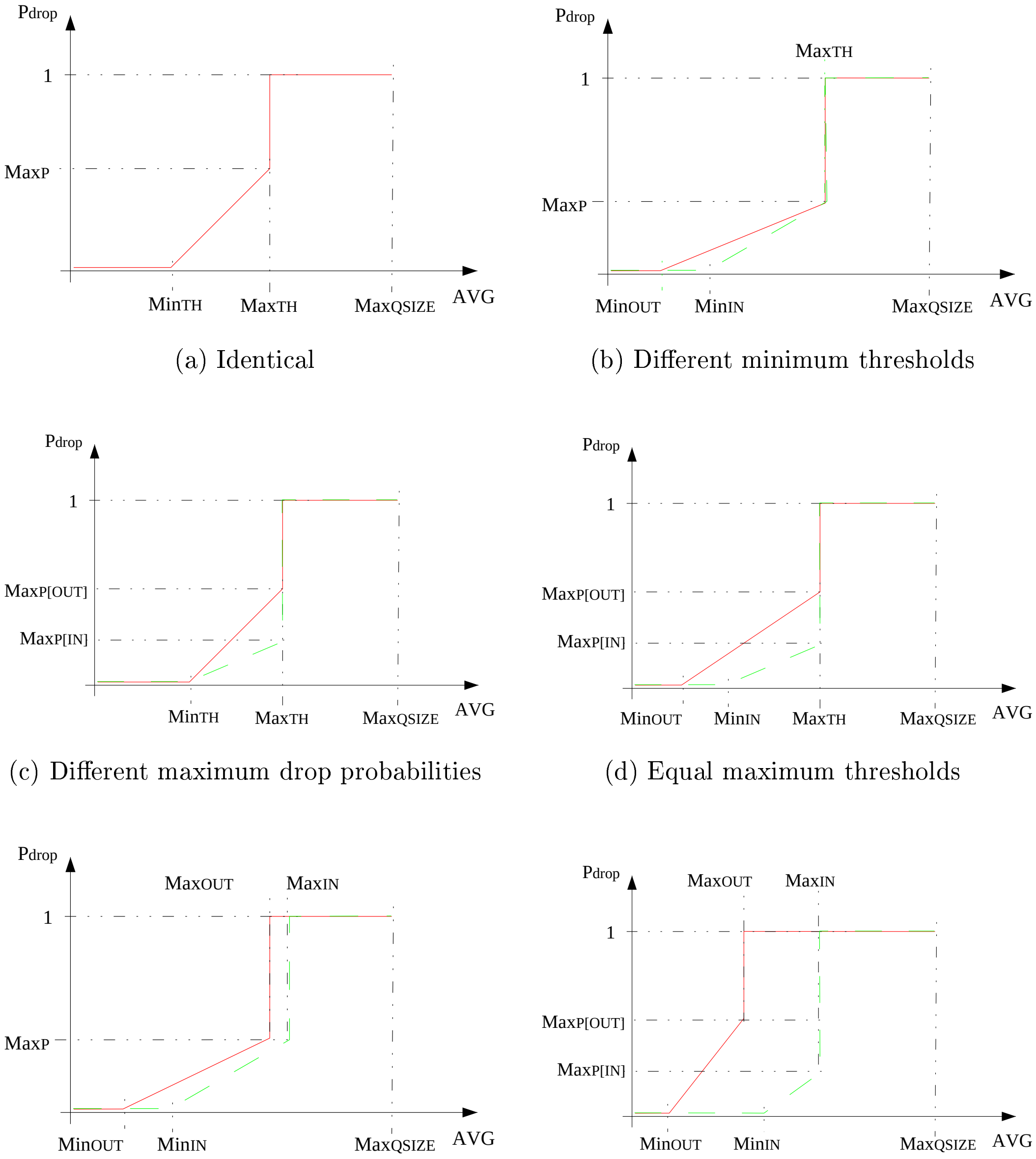
Conditioner is the first mechanism which an arriving packet encounters in the network. The task of the conditioner is to execute all the conditioning functions required for a particular packet and a user. There are many functionalities in the conditioner, but for this study main concerns are:

- Classification
- Metering
- Marking

Implementation and structure of these functions can differ a lot, but some aspects of these functions are general.

5.4.1 Classification

Classification of packets can be based on a number of criteria. The behavior aggregate (BA) classifier is a simple field recognition filter comparing information



(a) Identical

(b) Different minimum thresholds

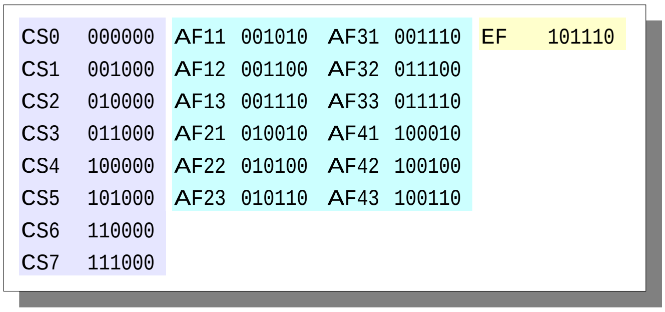
(c) Different maximum drop probabilities

(d) Equal maximum thresholds

(e) Equal maximum drop probabilities

(f) Differential

Figure 5.11: Drop behavior of the RIO algorithm



CS0	000000	AF11	001010	AF31	001110	EF	101110
CS1	001000	AF12	001100	AF32	011100		
CS2	010000	AF13	001110	AF33	011110		
CS3	011000	AF21	010010	AF41	100010		
CS4	100000	AF22	010100	AF42	100100		
CS5	101000	AF23	010110	AF43	100110		
CS6	110000						
CS7	111000						

Table 5.1: Standardized DSCP values

in the DSCP-field with the standardized values, see Table 5.1.

Multi-field (MF) classifier is used to achieve more detailed resolutions, based on information carried in the other fields of the IP header. A common implementation is to combine IP address, transport protocol and transport protocol port. With this flow identifier classification is done with variable granularity.

5.4.2 Metering

Metering in the DiffServ is a discrete value process by which the traffic is classified into some category based on its temporal behavior. This classification result is used in the marking for three purposes:

1. Marking
2. Shaping
3. Dropping

There are a number of estimator algorithms, which can be used in the metering, however, one is above the others: Token Bucket estimator. It has a long history in the packet networks and due to a relative simple implementation, it has gained a de facto status as a rate estimator. Following items represent three estimator algorithms which are relatively low in complexity.

1. **Average rate estimator**: rate is measured as in the RED algorithm, see Figure 5.8, using the exponentially weighted moving average (EWMA) filter to produce a weighted estimate of the average rate. The estimator is updated upon each arrival of a packet. This estimator, due to the single parameter tuning (i.e. memory of the estimator), is not responsive on many time scales, rather it is tuned to function on some time scale, which may not represent characteristics of the communication. This is clearly visible from Figure 5.12, where the average rate estimate and original traffic stream are plotted. Memory of the estimator is adjusted with update factor (w_q in the EWMA algorithm in Figure 5.8). Update factor adjusts the weight of the measured value compared to the previous value of the estimator. With a low update factor, it requires a number of packets to change the estimator

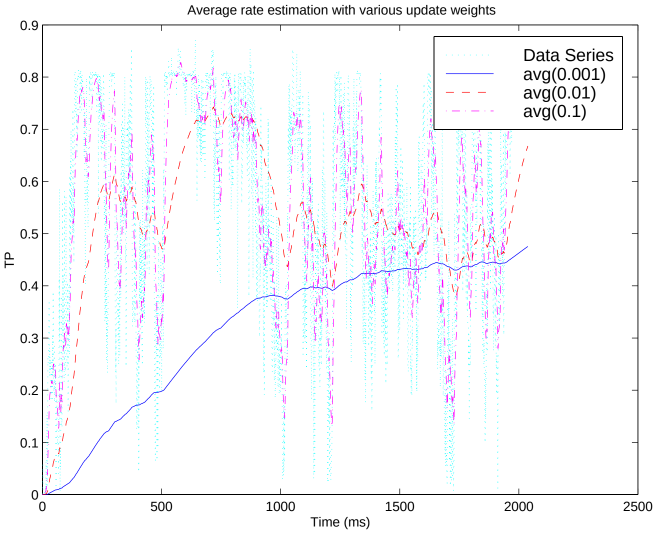
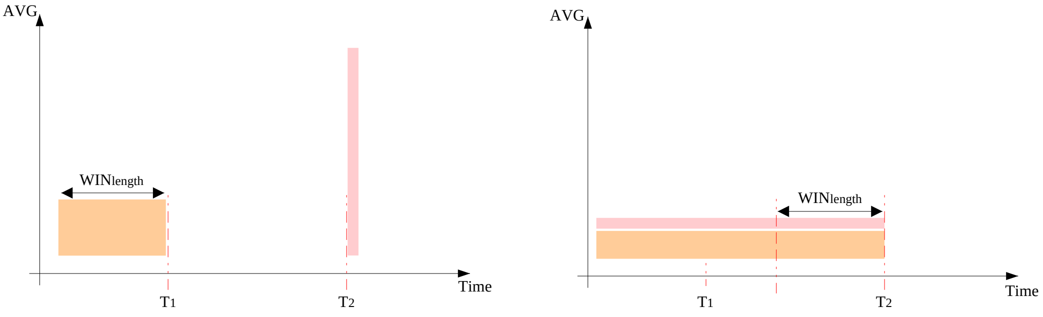


Figure 5.12: Average rate estimate using different weights



(a) Initial condition

(b) Condition after update

Figure 5.13: Operation of the TSW estimator

value significantly. A number associated with the label *avg* in Figure 5.12 represents the update factor used in the construction of the estimator value.

The update factor should be based on the measurement period and average rate agreed in the SLA. For example, with the average rate of 128kbit/s and measurement period of 1s the update factor should be of the order of 0.05. However, if the client is of the on/off type which produces high speed burst with long idle times, the estimator does not take idle times into account (there are no packets arriving for the estimator to update the estimator value).

2. **Time-sliding window estimator (TSW estimator)**: is a modification of the average rate estimator. TSW uses a time-based windowing of the transmission rate to produce measurement points for averaging. This eliminates the effect of idle times on the estimate. This modification causes the TSW estimator to have memory which is independent of arrival rate of the connection. See Figure 5.13 for an operational example and Figure 5.14 for the structural representation of the algorithm.

3. **Token bucket rate estimator**: is the most popular estimator in the cur-

```
Initially:
    $Win_{length} = C$
    $Avg_{rate} = Target\ \ rate$
    $T_{front} = 0$

Upon each packet arrival:
    $Bytes_{TSW} = Avg_{rate} \cdot Win_{length}$
    $New_{bytes} = Bytes_{TSW} + packet_{size}$
    $Avg_{rate} = New_{bytes}/(T_{now} - T_{front} + Win_{length})$
    $T_{front} = T_{now}$
```

Figure 5.14: TSW estimator algorithm [Fan98]

```
Initially:
    $Number\ \ of\ \ tokens = S$

Upon each packet arrival:
    $Increment = Token\ \ size \cdot R \cdot (T_{now} - T_{last\ \ arrival})$
    $Decrement = Packet\ \ length$

    $Conformance = Number\ \ of\ \ tokens + Increment - Decrement$
    if $Conformance >= 0$
      then $Number\ \ of\ \ tokens = min(S, Conformance)$
    else $Number\ \ of\ \ tokens = Number\ \ of\ \ tokens + Increment$
```

Figure 5.15: Token bucket algorithm

rent packet networks. Token bucket has, in general, integrated estimator and marking action, which is executed as a one algorithm. However, it is possible to extract the estimator part and the conformance checker (marking) part from the algorithm.

Token bucket rate estimator takes account of the idle times as the TSW does by applying the time information of the last arrived packet. Token bucket estimator usually has two operative parameters:

(a) Size of the bucket (S)

(b) Token generation rate (R)

Conceptually the algorithm operates in the way described in Figure 5.15. Initially bucket is full of tokens. Each arriving packet causes evaluation of the conformance. If the bucket has enough tokens[6] for the packet then the packet is within the profile, otherwise the packet is out of profile.

Token buckets can be cascaded to produce conformance information of many levels. This allows more information for marking purposes and, perhaps, better dynamics. Operation of the token bucket is stable and predictable. This estimator allows variance in the transmission rate within limits of the token bucket size.[7]

[6] Evaluation is done on bytes. Each token represents some number of bytes. A packet is in profile if bucket carries enough tokens to cover the size of the packet

[7] Evaluation of the token bucket estimator as a rate shaper, a rate regulator and a rate estimator can be found from references [Lel89, SLCG89, Lee94, TT99]

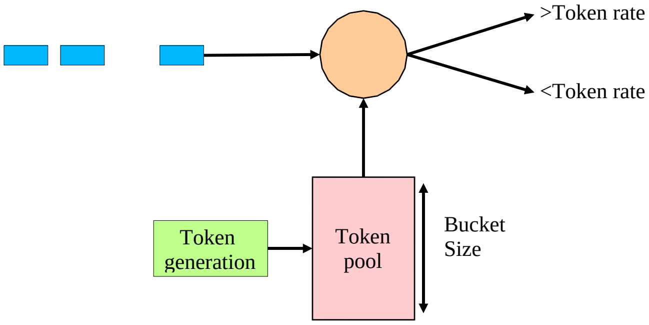


Figure 5.16: Token bucket estimator

5.4.3 Marking

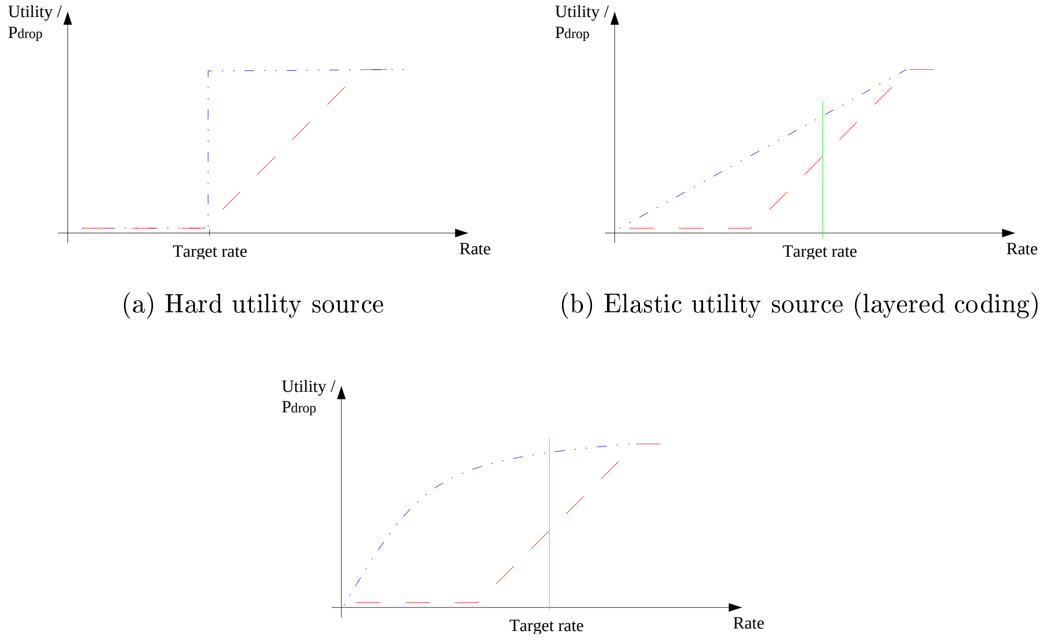
Marking is the process of actually assigning the service class to the packet. This process can be simple quantized table lookup, based on the estimated rate, or it may contain higher intelligence.

Marking has direct impact on the packet discarding at the congested router. Packets which are marked for lower importance have higher probability to be dropped than packets which are marked for higher importance.

There are many different ways to implement marking. Most of the implementations have some higher intelligence which aims with co-operative metering process to provide a fair treatment to the clients which use different applications and protocols. Following is a short summary of some markers, which can be combined with meters presented in Section 5.4.2.

1. **Linear probability marker**: marks packets exceeding the contracted rate to a lower priority with linearly increasing probability. For bursty connections, like TCP, this means that some of the packets from the bursts are marked and some are not. For the CBR connections, due to stochastic tagging, this causes problems through fuzzy operation in the conformance limit. While the TCP receives a smooth operation, oscillating below and above the conformance limit, UDP based CBR connections can send packets with continuous non-conformance without significant probability of marking the packet as out of profile. See Figure 5.17 for illustrative explanation about differences.

2. **Time-sliding window marker (TSW marker)**: is a modification of the linear probability marker. TSW tracks the TCP oscillations letting the TCP to reach 1.33 times the target rate, after which packets are marked to a lower priority. This way the mean rate should, in an ideal situation, be close to the target rate. This, however, is not necessarily optimal for the TCP, due to burst of lost packets during a single RTT. A burst of lost packets causes the TCP to move to the *slow start* state. For the UDP clients this allows a window of 0.33 times the target rate of excess capacity, as with linear probability marker. See Figure 5.18 for illustrative explanation of the TSW marker with different application types.

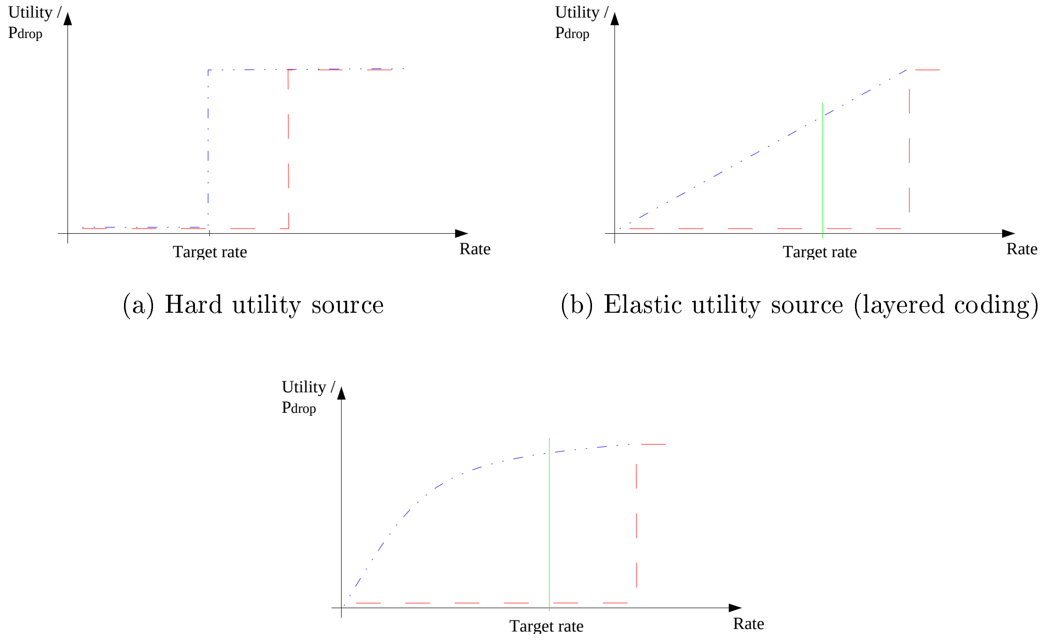


(a) Hard utility source

(b) Elastic utility source (layered coding)

(c) Elastic utility source (data transfer)

Figure 5.17: Marking and utility function with the linear probability marking

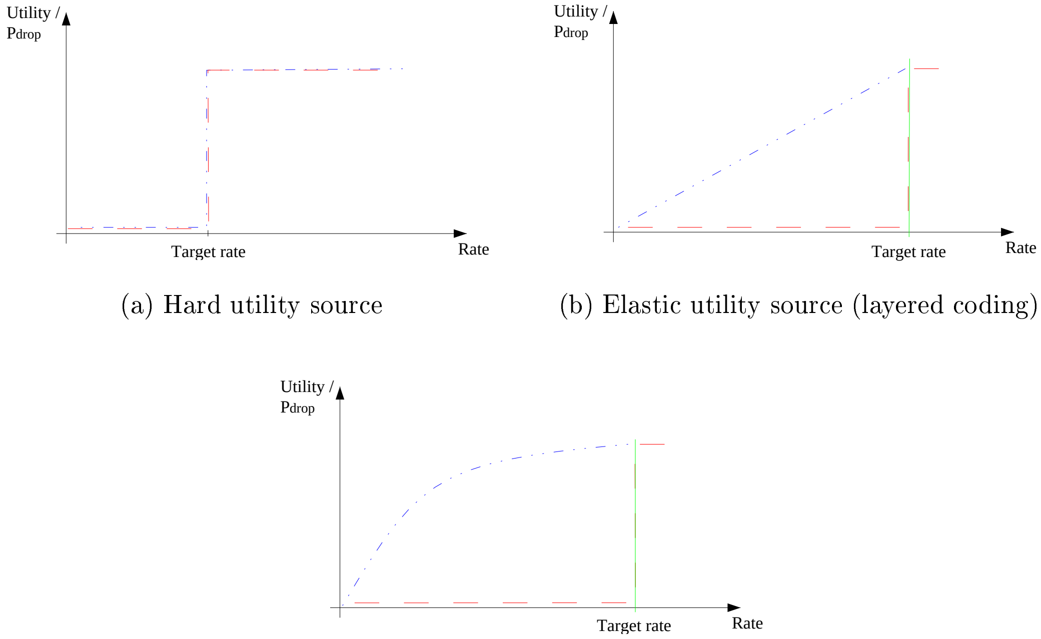


(a) Hard utility source

(b) Elastic utility source (layered coding)

(c) Elastic utility source (data transfer)

Figure 5.18: Marking and utility function with the TSW marking



(a) Hard utility source

(b) Elastic utility source (layered coding)

(c) Elastic utility source (data transfer)

Figure 5.19: Marking and utility function with the two color marking

3. **Two-color marker**: is a simple marker, which marks a packet not conforming to the target rate to a lower priority. This marker is normally seen with the token bucket meter, which produces information about whether or not a packet conforms to the traffic profile. See Figure 5.19 for illustrative explanation about the two color marker with different application types.

5.4.4 Problems with traffic conditioning

There are a number of difficulties and trade-offs in conditioning which relate to the nature of the Internet traffic. Most of these trade-offs are caused by two different types of network traffic: TCP and UDP.

1. **Network bias against closed loop control**

   Traffic in the Internet can roughly be divided into two classes:

   (a) Open loop, application controlled traffic (UDP as the transmission protocol)

   (b) Closed loop, source controlled traffic (TCP as the transmission protocol)

   This co-existence of the open loop and the closed loop traffic produces difficulties as the open loop traffic is able to dominate over the closed loop traffic in a congested network. The reason for this is that TCP control algorithm is designed to maximize throughput in time varying environment. It uses

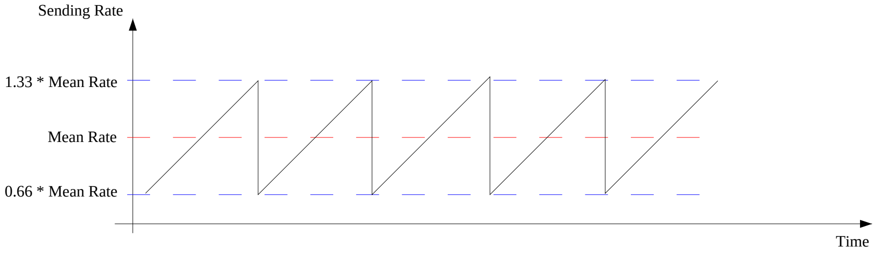


Figure 5.20: Ideal case of the TCP rate oscillation in the congestion avoidance state

acknowledgements of sent packets as a signal to increase or to decrease sending rate. Sending rate is controlled through the transmission window which defines the maximum amount of information that can be transmitted to the network without receiving acknowledgement for a successful delivery. The continuous process of comparing sent packets and received acknowledgements produces an uninterrupted probe mechanism. Probing allows sending rate to go up on steps of one packet per window and down by half a window if packet loss is experienced.

If the network has no or mild congestion, the TCP is operating in the *congestion avoidance* state. In this state, transmission window is oscillating between the maximum allowable window size[8] and half of the maximum window size. This oscillation, if averaged over the long time scale, produces a mean rate of 66% of the maximum rate.

In the case of multiple packet losses, TCP reacts by dropping the window to one and starting the *slow start* algorithm. This means that the client effectively ceases to transmit information to the network as it is not aware of the state of the network.

2. **Network bias against long RTT**

   An extra challenge for the metering comes from the network bias against long RTT with the TCP.

   As the TCP window algorithm is based on the acknowledgments of sent packets and indication of the congestion is based on missing acknowledgements, an estimate for the arrival time of the acknowledgment is required. This arrival time, which is the RTT, triggers the window control. For a small RTT the operation of the TCP source is more aggressive and the source receives more capacity than a similar source with a long RTT, see Figure 5.21. Taking this into account, when deciding metering result or marking action, is crucial if fairness among different flows and users is targeted.

3. **Network bias against high sending rates**

   The effect of the maximum rate is similar to the effect of the RTT. If the target rate, or guaranteed rate, of a connection is higher than the target rate of another connection, the connection with higher rate will suffer more from

[8] Flow control receives the advertised window from the receiver. Advertised window reflects the maximum amount of information which receiver is capable of handling at the moment.

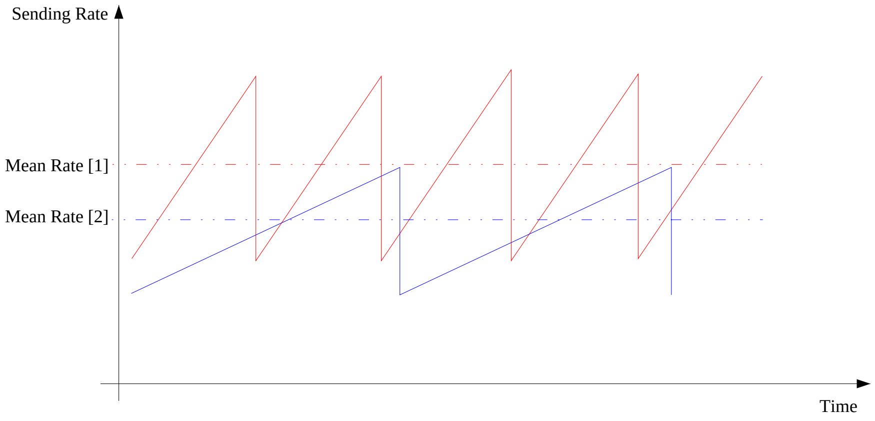


Figure 5.21: Effect of the RTT to the TCP rate oscillation

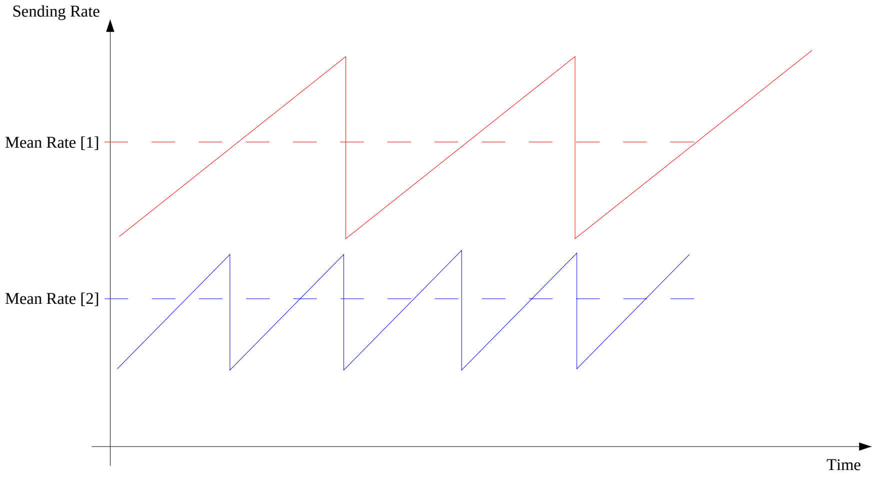


Figure 5.22: Effect of the target rate to the TCP rate oscillation, note that sources have similar RTT (slope of curves is same)

the action taken by the rate estimator or marker. This is a result from the fact that a packet loss triggers the window decrease process which shrinks the congestion window to half of what is was before the loss occurred. With a high-speed connection this decrease is bigger than with a low speed connection, see Figure 5.22.

As this marking is coupled with metering, applications which use TCP will regularly have packets marked for a lower importance. To allow users of the TCP to achieve good performance packet markings should occur as wide apart from each other as possible. This gives a higher probability for the TCP to stay in the *congestion avoidance* even if packets are lost in the network.

Chapter 6

Simulation of the Differentiated Services

6.1 Previous simulation studies

DiffServ has been previously studied by simulations e.g. in the following articles [JNP99, CLG99, IN98, CF98, Fan98, KS99a].

6.1.1 Simulation of the EF PHB

[JNP99] reports simulation results of the jitter variation in the EF PHB[1] implemented with the priority queuing (PQ) or weighted round robin (WRR). [JNP99] compares also the delivered service of the EF PHB based virtual leased line (VLL) with the conventional leased line system, where customer is served by dedicated resources.

The formulation of the problem statement is the following: *What is the jitter behavior of the WRR implementation of the EF PHB, when the WRR queue weights and number of queues is varied?*

PQ is the simplest implementation of the EF. With PQ, the EF-marked microflows are queued with a higher priority than the rest of the traffic. Due to the operation of priority queues, EF microflows will see little or no queues if load of the EF PHB group is kept reasonable. However, the PQ is not a good solution if fairness of all traffic flows is considered. This is due to the possible starvation of resources, which the high priority traffic causes to the low priority traffic.

WRR provides a tool for dividing the resources in a predictable manner to a number of flows/classes. WRR yields the worst possible jitter due to the partitioning of the scheduler service to the resource quantums, which are assigned to the different traffic classes. EF traffic gets its share of resources based on the weight that is set to the EF queue in the scheduler. The volume of the EF traffic should be small and consequently is the resource quantum for the EF queue small. Because of this, there will be queues on the path of the EF microflow which cause jitter to the traffic.

[1] More information about the EF PHB in Section 4.2.2.1

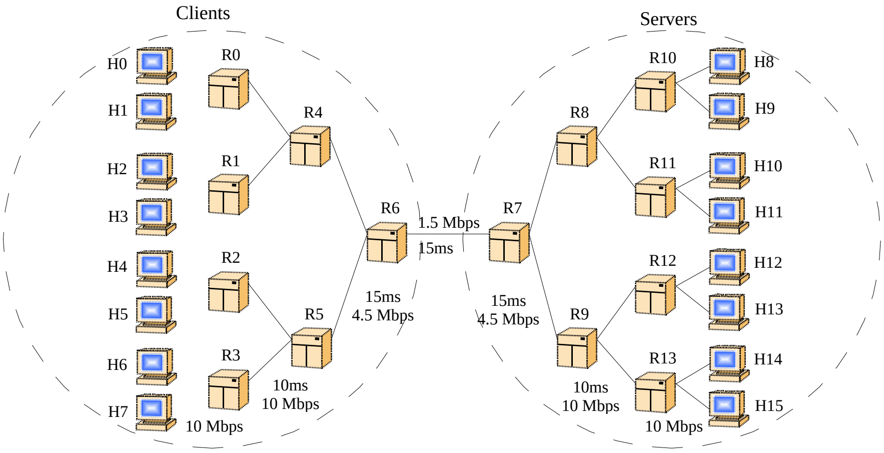


Figure 6.1: Expedited Forwarding simulation environment, simulation is done using the ns-2

6.1.1.1 Simulation model

The simulations are based on the ns-2 simulator with CBQ modules, which are modified to offer the PQ and the WRR based EF service. The simulated network is presented in Figure 6.1. In the evaluation of the jitter, the source model, which is used to produce EF traffic, is a constant bit rate (CBR) source with +/- 10% variation in the interpacket times. Packet size of the EF sources is constant, either 160 bytes or 1500 bytes. Total amount of the EF traffic is set to the maximum of 30% of the bottleneck link and the rest of the capacity is filled with HTTP and FTP traffic. In the evaluation of leased line emulation the EF traffic is also produced as a mixture of CBR (UDP) and HTTP/FTP (TCP) traffic. This situation resembles the current usage of leased lines in corporate interconnections.

Weights in the WRR scheduler are based on the service-to-arrival (SA) ratio. SA-ratio expresses the relative amount of resources which are dedicated to the EF class on the bottleneck link compared to resources, which it would need to pass traffic without congestion.

SA-ratio is thus defined as:

$$\text{SA-ratio} = \frac{(\text{WRR weight}) \times (\text{output link bandwidth})}{(\text{Arrival rate of EF packets to the queue})} \tag{6.1}$$

6.1.1.2 Results from simulation

Simulation results show that with a relative small SA-ratio (1.06) and small number of EF microflows the amount of jitter is larger. With large, 1500 bytes, packets jitter is in the range of a half of the packet time and with small, 160 bytes, packets jitter is in the range of 5 to 7 times the packet time. By increasing the SA-ratio to 1.5, jitter of small packets can also be pushed down to the range of the packet time. Variation of the number of queues seems to have little effect on the jitter.

This suggests that the SA-ratio should be of the order of 1.5 and the load of the EF microflows should be kept well in the range of 30% of the link capacity.

Conformance of the EF to the 'leased line' approximation shows that as the share of the EF traffic is increased the emulation is improved. This is somewhat expected result because statistical multiplexing smoothes the TCP ramping effect out as the number of flows within the class increases.

6.1.2 Simulation of the AF PHB

[CW97, CF98, Fan98, IN98] reports simulation results of the AF PHB[2] implemented with a 2-level RED queue (RIO). The effect of the round trip delay and nonresponsive flows to the target rate of the AF connections is examined through simulation of the bulk-data TCP connections in [CW97, CF98, Fan98] and FTP connections with AF and BE assignment in [IN98].

Formulation of the problem in these cases is common: *Are TCP sources able to achieve their target rate despite of highly variable round trip times and possibly interfering nonresponsive connections (UDP).*

6.1.2.1 Simulation model

The simulations are based on the ns-2 simulator. Topology of the simulations, presented in Figure 6.2, is the same in all the cases with the exception of the number of sources. The difference between simulations in [IN98] and [CF98, Fan98] is in the implementation of the conditioner[3]. Conditioner compares the transmission rate of the source to the subscribed target rate. This subscription can be based on different parameters (peak rate, mean rate, burst size etc.). Conditioner assigns a tag for each packet to indicate whether the packet is *in* or the packet is *out* of the profile.

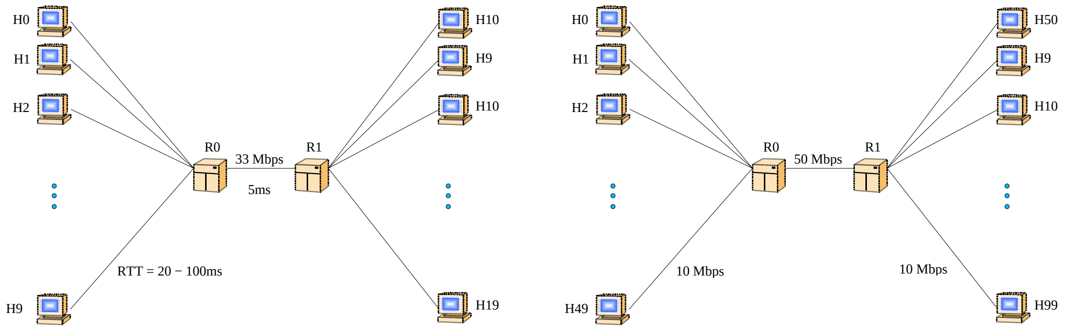
In [IN98] two options for the conditioner are examined:

- **Average rate estimator with linear probability marker**, which marks packets exceeding the conformed rate with linearly increasing probability. For bursty connections like TCP, this means that some of the packets from the bursts are marked and some are not. For the CBR connections, this allows continuous non-conformance without significant probability of marking the packet as out of the profile.

- **Token bucket rate estimator with two color marker**, which marks packets based on the conformance algorithm. Token bucket, as it has two parameters, bucket size and rate, allows bursts of the size of the bucket. Token bucket does not allow sustained excess rate for the CBR connections as the other conditioner does. Token bucket, however, allows natural deviation in the transmission rate with the use of buffered tokens.

[2] More information about the AF PHB in Section 4.2.2.2

[3] References [CF98, Fan98, IN98] refer the conditioner as profiler.



(a) Environment on [CF98, Fan98]

(b) Environment on [IN98]

Figure 6.2: Assured Forwarding simulation environment, simulations are made on ns-2

Token bucket is argued to perform more predictably with different types of sources and to allow stricter policing of the sources. This is apparent since the algorithm has no probabilities associated in the marking of packets.

In [CF98, Fan98] the time-sliding window (TSW) algorithm is presented to do conditioning. TSW is a windowed rate estimation algorithm where the window equals the expected round trip time of the connection. TSW tracks TCP oscillations letting TCP to reach 1.33 times the target rate, after which packets are marked as *out* of profile. This way the mean rate is balanced close to the target rate.

6.1.2.2 Results from the simulation

The simulation proves that AF tends to diminish the effect of the RTT on the achieved rate. Connections with small RTT try to grab more bandwidth leading to increased number of the *out* packets. These *out* packets are discarded with a higher probability, leading the TCP to reduce the rate. This phenomenon increases as the number of the AF connections increases (direct consequence from the higher utilization in the *in* class).

The effect of the conditioner is directly visible from the results of [IN98] and [CF98]. In simulations using the token bucket conditioner, connections achieve their target rates only if their target rate is small or moderate compared to the link speed. High bit rate connections have higher window values and during congestion need more time to increase window to the size required for the target profile. This gives an opportunity for the small and moderate bit rate connections to grab more capacity. TSW, however, is consistent with the target rates throughout the simulations. The rate oscillates +/- 15% from the target rate in every simulation, with the exception of the case where a nonresponsive UDP flow with zero bps reservation is applied. The nonresponsive UDP flow is able to grab 2.5 Mbps of the transmission capacity, even without reservation, by forcing the TCP connections to the slow start.

In the case of a mixture of AF and BE connections in [IN98] the excess capacity is shared by the TCP. AF connections, having higher rate, tend to have lower share due to the bigger drop in size of the congestion window after packet loss. Therefore, complete fairness with the BE and AF connections cannot be achieved. However, this is the case with all TCP connections with different windows.

6.2 Goals of this study

The purpose of this simulation study is to examine following questions:

- Jitter behavior of different implementation aspects of the EF PHB. This is done in order to be able to relate present results to previous studies presented in Section 6.1.1.
- Effect of the RTT on how well the target rate of the connections in the AF PHB is achieved. This is done in order to be able to relate present results to previous studies presented in Section 6.1.2.
- Isolation of the traffic classes, when there are EF and AF connections with BE background traffic in the network. This is studied through probe sources in each class. Probing is done to calculate delay distribution and loss probabilities.

6.3 Simulation model

6.3.1 Environment

Simulations were done by using the BONeS simulator. BONeS is a commercial, object oriented, simulation environment with capabilities of block based description language, state machine language and C++. Simulation model can be build with a heterogeneous combination of modules written in different languages. BONeS was chosen for the reasons of familiarity rather than performance or suitability issues.

6.3.2 Topology

The network topology for simulation, see Figure 6.3, was chosen to have close similarity to the topology in the simulation of [JNP99]. However, some changes were made to have more independent traffic sources.

The same topology is used throughout this study. Each simulation has different allocation of connections to the different classes. These allocations are shown in Figures associated to the particular simulation. The sources of the simulation model are bi-directional and the TCP sources have full implementation of the TCP (i.e. they have also connection setup and tear down modeled).

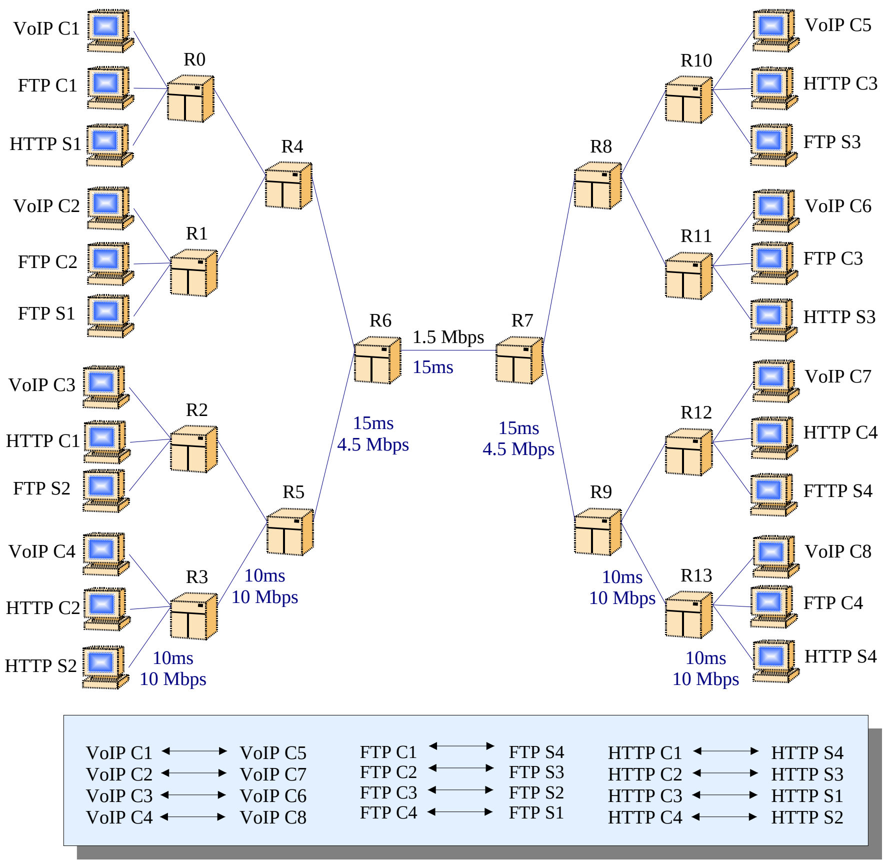


Figure 6.3: Network topology used in the simulations of this study

6.3.3 Source Models

6.3.3.1 VoIP

Call level model for the VoIP-client is based on the connection arrivals and holding times. It is known from the measurements and dimensioning rules of the telephone networks that calls arrive as a Poisson process and they have exponential holding times [KR86, Rah91]. The holding time distribution has mean of 180 s in the today's networks, during business hours. Mean arrival rate is varied to produce certain load levels.

$$\text{Load} = \frac{\text{(Mean holding time)}}{\text{(Mean inter-arrival time)}} \tag{6.2}$$

Packet level model for the VoIP-client is based on the results in [LPY98, LIP99]. These references present results from traffic measurements and analysis of various application types, also VoIP. Analyses in these references concentrate on the packet length and inter-arrival times. VoIP-clients are usually constructed based on the APPLICATION/RTP/UDP/IP protocol structure. Three different client models were derived from the analyses. Also aggregated model to represent the PBX or the VoIP trunk line was constructed. This aggregation model is multiplication of a single source. In this aggregation model, each source is independent of the others and the type of a client is a uniformly distributed random variable.

First client model is based on the analysis of the Network Voice Terminal (NeVoT) VoIP-client. NeVoT can be configured to use various coding and framing mechanisms. The analyzed client used G.711 coding with 20 ms framing. This combination produces 160 bytes of information for each frame. Adding the protocol structure gives respectively 160/172/180/200 byte packets. Because we did not use silence detection, NeVoT generated packets constantly, even when there was no speech signal to be transmitted.

The second and third client models are based on the analysis of the Selsius IP PBX and its clients. Selsius, which is meant to operate in LAN and WAN environment, has two rate limit classes, 1Mbps and 56kbps. These rate limits represent the maximum capacity, which can be used to convey the telephone conversation. In either case Selsius uses 30 ms framing and silence detection, during which *no* packets are sent to the network. In the 1Mbps limit class, telephones use G.711 coding to produce information. This generates packets of 240/252/260/280 bytes. In the 56kbps-limit class, telephones use G.723 coding which compresses the information and produces packets of 24/32/44/64 bytes.

Packet Level model of the VoIP follows these client models with the exception that the silence detection was not modeled.

6.3.3.2 HTTP

The model for the HTTP-client and HTTP-server is based on the traffic measurements and analysis in [Mah97, KkA97, Nie98]. These references present analyses of the HTTP-traffic and two models that fit to these results. The model which was developed in ETSI mobile working group [Nie98] and by Mah [Mah97] aimed to model HTTP conversation on various levels of activity.

- **Session arrival process:** Session arrival process models service calls from the users. Session arrivals are independent of each other and they follow loosely the laws of telephone call attempts. For this reason, it can be modeled as a Poisson process. Mean arrival rate of calls is varied to produce desired load level of the HTTP traffic.

- **Page requests per session:** Page requests per session models locality of the HTTP-session. Usually a relative small number of documents are retrieved from the single server before moving to a new server. Number of requests is expressed in the ETSI-model [Nie98] as geometrically distributed random variable with the mean of 5 pages in a session. Mah, in [Mah97], presents measurement results, which follow geometric distribution with reasonable accuracy, as seen in Figure 6.4.

- **Reading time between page requests:** Reading time between page requests is the time between two consecutive page requests. This time is used for reading the document or some other activity between requests. This time is also modeled in [Nie98] as a geometrically distributed random variable with the mean of 12 s. In the measurements of [Mah97] much longer tail was observed than with the geometric distribution, see Figure 6.5. In

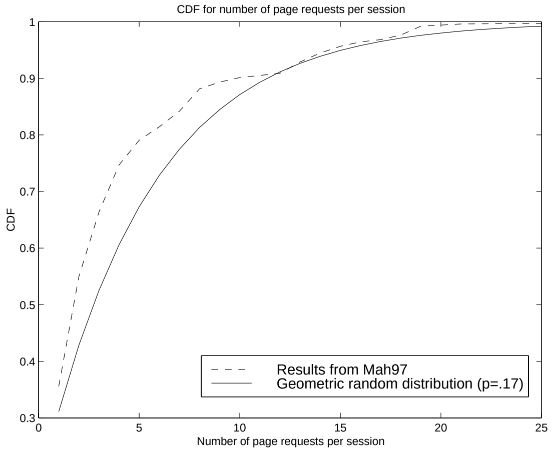


Figure 6.4: Cumulative distribution functions for 'number of page requests per session'

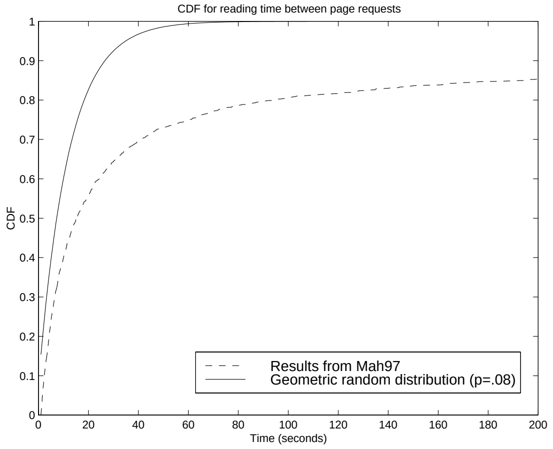


Figure 6.5: Cumulative distribution functions for 'time between two consecutive pages'

the article, Mah explains the long tail with the notion that people do not use service similarly during holidays (Thanksgiving Day was in the middle of the measurement period of [Mah97]).

- **Number of objects in page:** Number of objects in the page, with the object size, model the structure of a single web page. Commonly web page is a mixture of pictures and text. Every picture is a separate object whereas text in the page is treated as a single object. Usually single page contains a relatively small number of objects but there are exceptions to this rule with pages of tens or even hundreds of objects. In the results of [Mah97], this also follows reasonably well the geometric distribution with the mean of 3 objects in the page, see Figure 6.6.
- **Object size:** Normally, as connection speeds are small in the Internet, objects are tried to be kept as small as possible. Results of [Mah97] follow the geometric distribution up to 80% point after which the geometric distribution decays too fast compared to the long tail of measured distribution, see Figure 6.7.
- **Request length:** Request length, in Figure 6.8, is considerably smaller

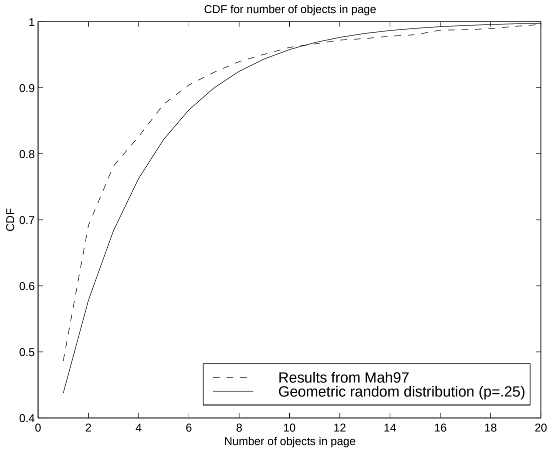


Figure 6.6: Cumulative distribution functions for 'number of objects in the page'

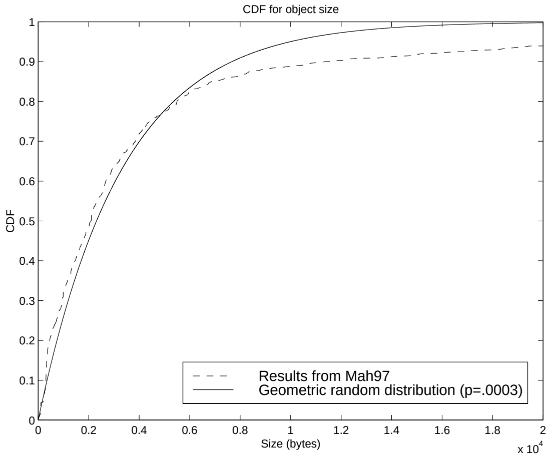


Figure 6.7: Cumulative distribution functions for 'size of responded objects'

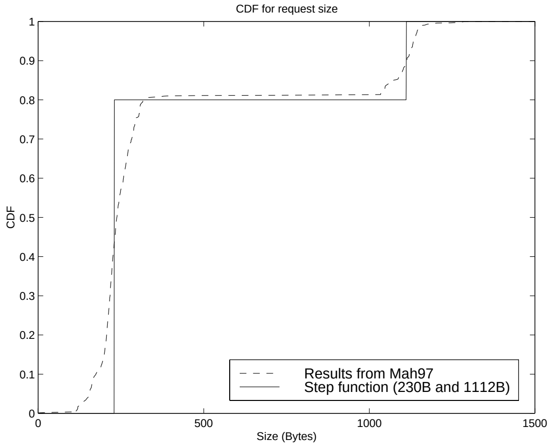


Figure 6.8: Cumulative distribution functions for 'size of requests'

than response in the HTTP. Requests typically fit in a single IP-packet as they usually have rather fixed size. In measurements of [Mah97], this was found to have two typical sizes, 250 B in normal requests and 1 KB in the case of HTML forms.

This work does not attempt to have the most refined model for each client type, rather it tries to make the client model as simple as possible yet still retaining general characteristics of the application.

For these reasons the following distributions and parameters were selected:

- **Session arrival process** is modeled as a Poisson process with the mean inter-arrival time of 5 seconds.
- **Page request per session** is a geometrically distributed random variable with the mean of 5 requests per session.
- **Reading time between page requests** is modeled as a geometrically distributed random variable the with mean 12 seconds and 1 second resolution.
- **Number of objects in page** is modeled as a geometrically distributed random variable with the mean 3 objects in page.
- **Object size** is modeled with geometric distribution having the mean value of 3.3 kB.
- **Request length** is modeled as a constant size of 256 bytes.

The client model is shown as a flow chart in Figure 6.9.

6.3.3.3 FTP

FTP is an important application in the Internet. It seems that the HTTP is slowly pushing it out. However, there are still many occasions when the FTP is

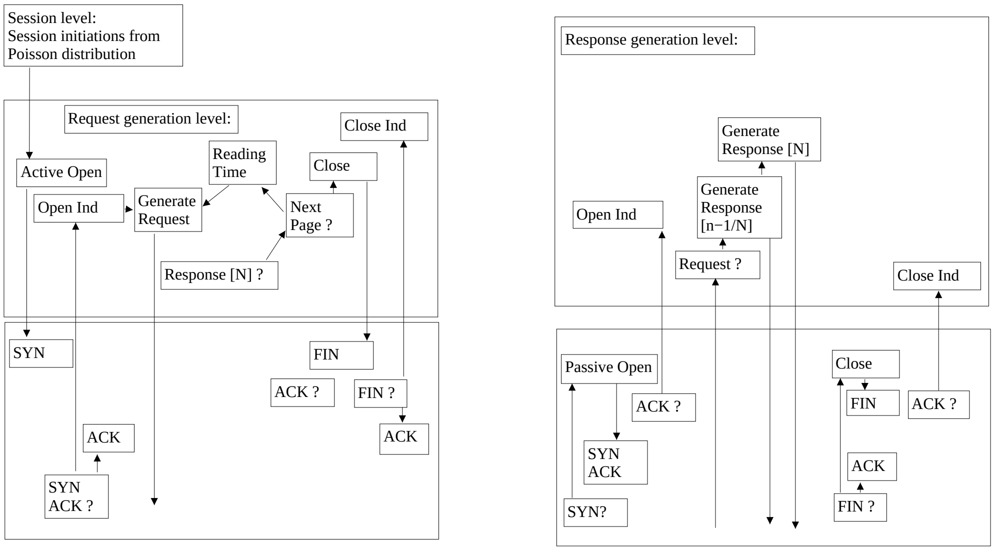


Figure 6.9: HTTP connection structure

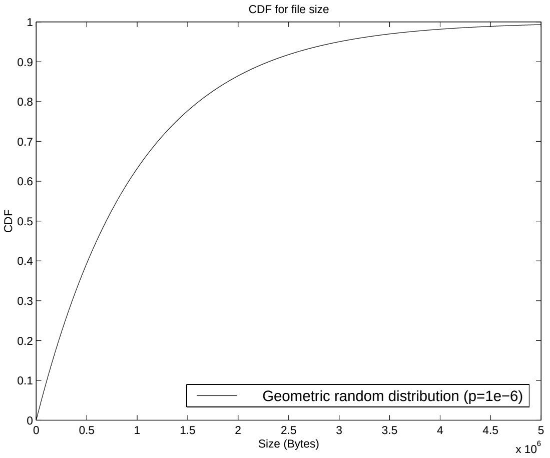


Figure 6.10: Size of FTP-files

used for the sake of reliability, which it has over the HTTP in large file transfers. The FTP can be modeled in a similar way as the HTTP. The difference is that there is no reading time between requests, no multiple objects in a single request and the larger size of responded objects. For some simulations studies FTP is used for the background traffic process. Usually in these studies, the file size is set to be infinite. Infinite FTP is a greedy application, which tries to grasp all of the resources.

- **Session arrival process** is modeled as a Poisson process with the mean inter-arrival time of 30 seconds
- **File request per session** is a geometrically distributed random variable with the mean of 5 requests per session.
- **File size** is modeled by geometric distribution with the mean 1000 kB, see Figure 6.10.
- **Request length** is modeled as a constant size of 256 bytes.

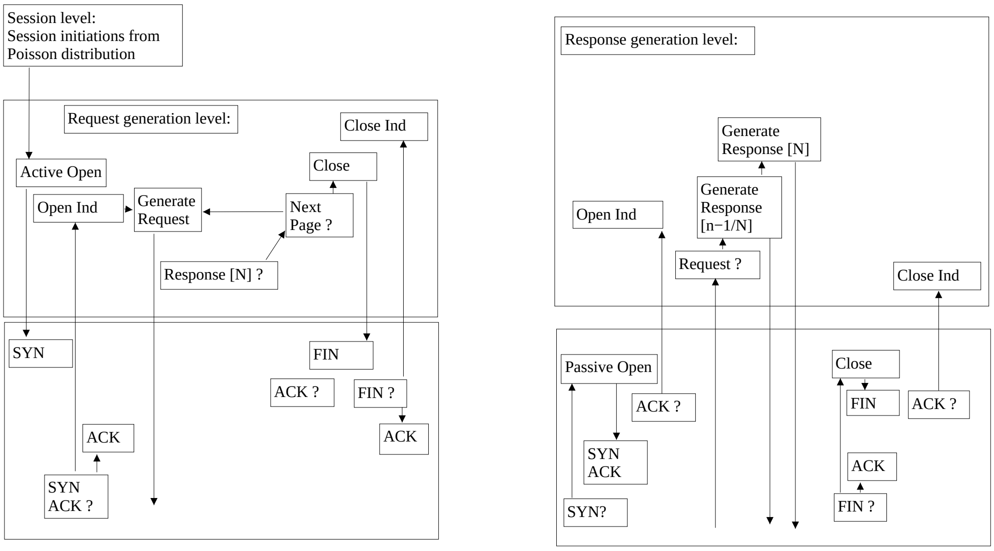


Figure 6.11: FTP connection structure

THe client model for the FTP is shown as a flow chart in Figure 6.11.

6.3.4 Per hop behaviors

This work concentrates on the Expedited Forwarding (EF), Assured Forwarding (AF) and default (BE) Per Hop Behavior (PHB) groups. Implementation of the PHBs is based on the Random Early Detection (RED) queues. Each PHB group has its own queue, which is serviced by the scheduler. The scheduler has three modes:

- Processor Sharing (PS)
- Weighted Round Robin (WRR)
- Priority Scheduling

PS and WRR schedulers serve first packet from each PHB group, PS with equal share of capacity for each group and WRR with weights assigned to each group. Weights are implemented as Time Slice (TS) attributes, which control the processing time which each group gets in a single cycle. When processing time equals service time, a packet is transmitted forward and next packet from that group enters the service.

6.3.4.1 Expedited forwarding

Expedited forwarding (EF) PHB is based on a small queue, tightly policed and correctly provisioned service. Scheduler is either priority or WRR scheduler.

6.3.4.2 Assured forwarding

Assured forwarding is implemented as two level RED queue - RIO. Packets are either *in* or *out* of their target profiles. Target profile consists of the mean rate together with the bucket size for bursts sent at the maximum rate of the link. The RIO queue, which has two RED algorithms running in parallel in a single queue, has two options for operation:

- **Connected:** RED for the *in* class sees number of *in* packets in a queue and RED for the *out* class sees the total number of packets in a queue.
- **Disconnected:** RED for the *in* class sees the number of *in* packets in a queue and RED for the *out* class sees the number of *out* packets in a queue.

Selection of the RED parameters can also be used as a tool for quality differentiation.

6.4 Simulation results

6.4.1 General parameters for simulations

There are many parameters to be set for the simulation of different applications in the Internet. The following are the parameters, which were set to the different devices and protocols:

1. Clients
    - Application layer - Parameter:
        - 'Mean intersession time':
            - HTTP client: 5 seconds
            - FTP client: 30 seconds
    - TCP layer - Parameter:
        - 'Receiver window size': 9180 bytes
        - 'Maximum segment size': 1500 bytes
2. Routers
    - 'Buffer size':
        - EF: 10 packets
        - AF and BE: 100 packets
    - 'RED parameters':
        - EF: $min_{th} : 0.45$, $max_{th} : 0.8$, $p_{max} : 0.05$
        - AF: $min_{th}(in) : 0.45$, $max_{th}(in) : 0.8$, $p_{max}(in) : 0.05$, $min_{th}(out) : 0.25$, $max_{th}(out) : 0.8$, $p_{max}(out) : 0.1$
        - BE: $min_{th} : 0.25$, $max_{th} : 0.8$, $p_{max} : 0.1$

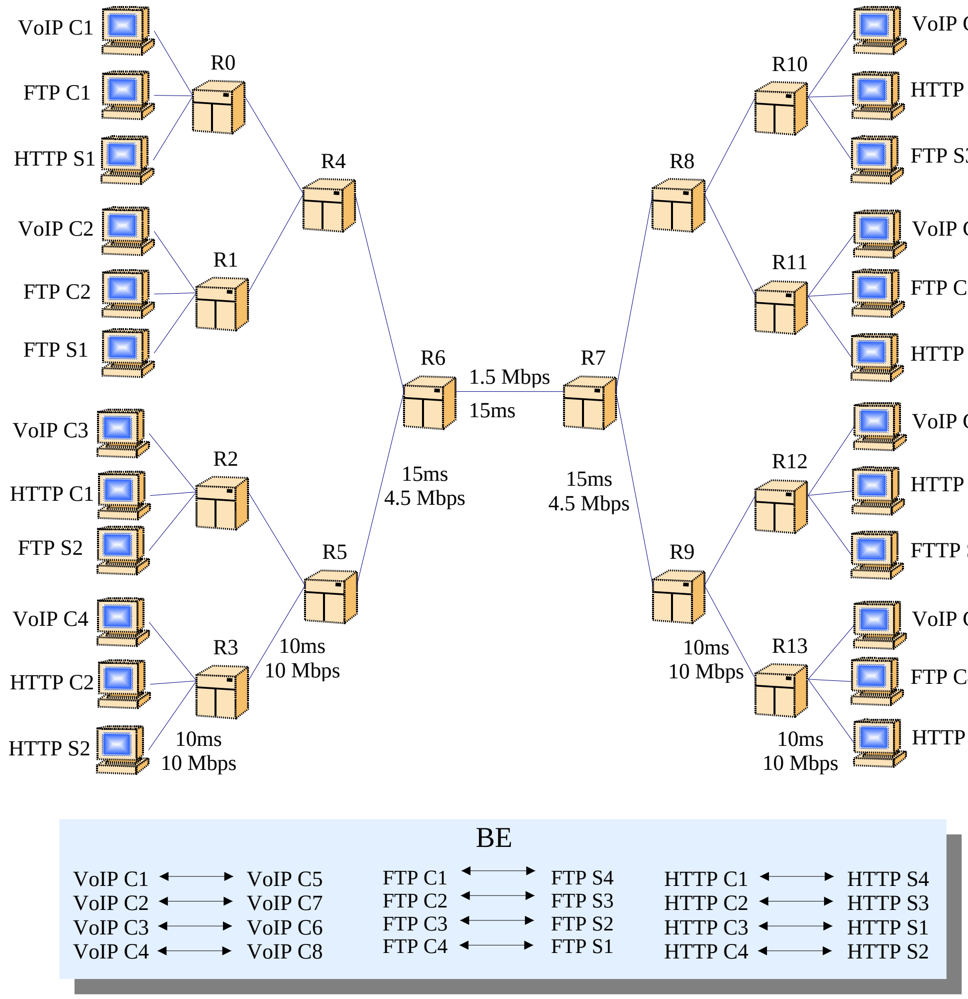


Figure 6.12: Network topology and connection pairs in the BE simulation

6.4.2 BE simulations

This section presents simulation results of a conventional best effort network. The simulated network topology and connection pairs are presented in Figure 6.12.

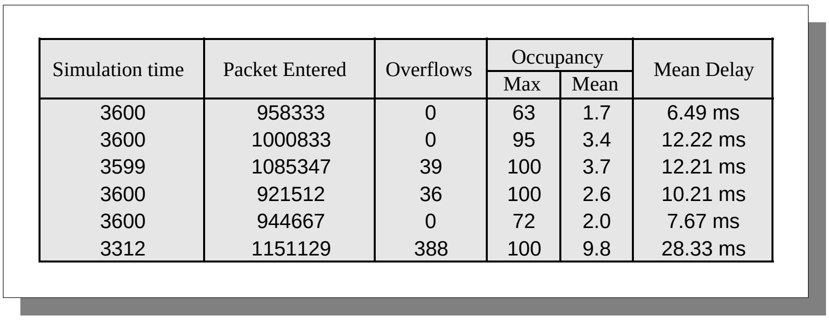
Simulations were run with two different configuration options:

1. **Similar RTT times** - each client/server pair has similar RTT on the connection path.
2. **Dissimilar RTT times** - the link between routers R7 and R8 has 10 times higher (150ms) delay than the link between routers R7 and R9 (15ms).

Both cases were reproduced six times to provide statistical confidence of the results. Iterations were independent processes with different seeds.

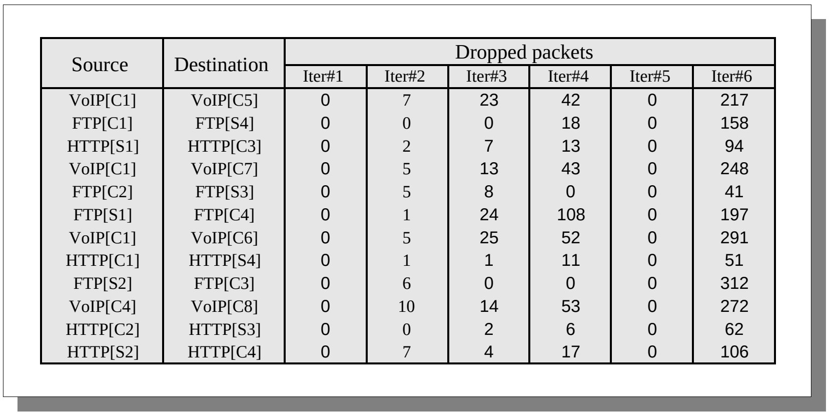
6.4.2.1 BE with similar RTT

The case where each connection has the 'same' RTT is only theoretical. In the real world each router and link has its own characteristics that depend on the



Simulation time	Packet Entered	Overflows	Occupancy		Mean Delay
			Max	Mean	
3600	958333	0	63	1.7	6.49 ms
3600	1000833	0	95	3.4	12.22 ms
3599	1085347	39	100	3.7	12.21 ms
3600	921512	36	100	2.6	10.21 ms
3600	944667	0	72	2.0	7.67 ms
3312	1151129	388	100	9.8	28.33 ms

Table 6.1: General buffer statistics from the router R6, CASE: BE with similar RTTs



Source	Destination	Dropped packets					
		Iter#1	Iter#2	Iter#3	Iter#4	Iter#5	Iter#6
VoIP[C1]	VoIP[C5]	0	7	23	42	0	217
FTP[C1]	FTP[S4]	0	0	0	18	0	158
HTTP[S1]	HTTP[C3]	0	2	7	13	0	94
VoIP[C1]	VoIP[C7]	0	5	13	43	0	248
FTP[C2]	FTP[S3]	0	5	8	0	0	41
FTP[S1]	FTP[C4]	0	1	24	108	0	197
VoIP[C1]	VoIP[C6]	0	5	25	52	0	291
HTTP[C1]	HTTP[S4]	0	1	1	11	0	51
FTP[S2]	FTP[C3]	0	6	0	0	0	312
VoIP[C4]	VoIP[C8]	0	10	14	53	0	272
HTTP[C2]	HTTP[S3]	0	0	2	6	0	62
HTTP[S2]	HTTP[C4]	0	7	4	17	0	106

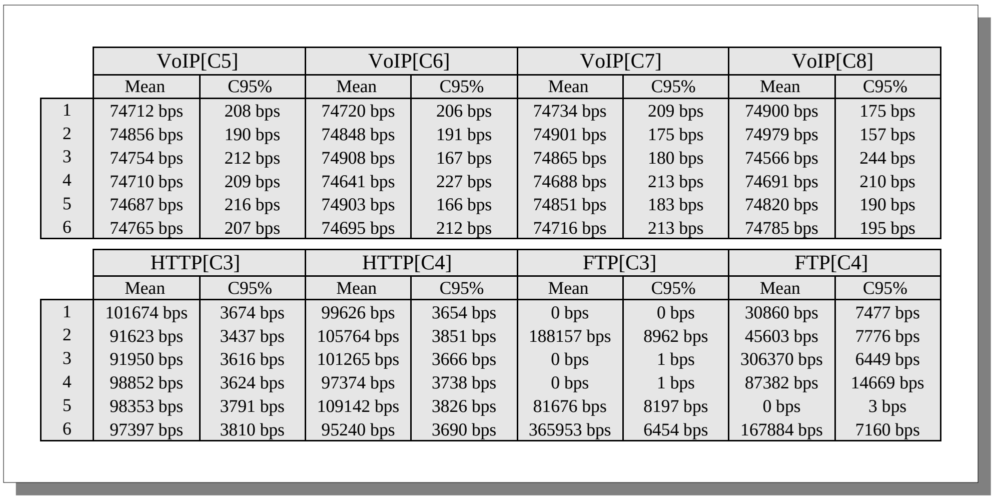
Table 6.2: RED statistics from the router R6, CASE: BE with similar RTTs

HW/SW capabilities, link type and distance. In addition, offered load has big effect to the overall delay. However, simulations allow every router to have equal capabilities and link characteristics to be easily adjusted. Offered load is the only time varying function which cannot be directly controlled.

Table 6.1 provides general statistics from the buffer of the bottleneck link between R6 and R7. Simulation time was set to be 3600 s but in some occasions, like in iteration 6, simulation was aborted due to an anomaly in the TCP (duplicate packet which was not handled correctly).

Table 6.2 shows numerical results of packet drops in the RED control of the same buffer. Results show that the VoIP clients lose more packets in RED than the TCP clients do. This is expected as they are not able to control their sending rate at the times of congestion. However, also FTP connections exhibit a large number of packet drops in some iterations.

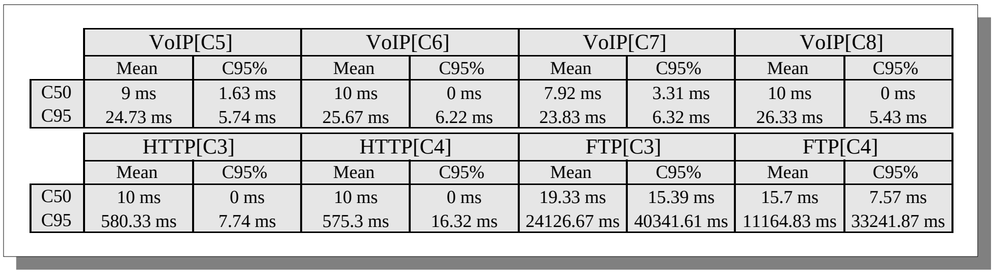
Throughput results of different clients are presented in Table 6.3. VoIP clients, which use UDP, are able to make the most of resources. They are on the average only 500 bps behind from their information generation rate (75200 bps), i.e. they suffer a packet loss of 1% in the network. An interesting behavior, which is directly observable form the throughput results, is the high variability in the performance between two FTP clients. There is no constructional reason for this, as the clients are identical and experience, in general, similar behavior in the network. One possible cause for this behavior is reported in the book of Huston [Hus00]:



	VoIP[C5]		VoIP[C6]		VoIP[C7]		VoIP[C8]	
	Mean	C95%	Mean	C95%	Mean	C95%	Mean	C95%
1	74712 bps	208 bps	74720 bps	206 bps	74734 bps	209 bps	74900 bps	175 bps
2	74856 bps	190 bps	74848 bps	191 bps	74901 bps	175 bps	74979 bps	157 bps
3	74754 bps	212 bps	74908 bps	167 bps	74865 bps	180 bps	74566 bps	244 bps
4	74710 bps	209 bps	74641 bps	227 bps	74688 bps	213 bps	74691 bps	210 bps
5	74687 bps	216 bps	74903 bps	166 bps	74851 bps	183 bps	74820 bps	190 bps
6	74765 bps	207 bps	74695 bps	212 bps	74716 bps	213 bps	74785 bps	195 bps

	HTTP[C3]		HTTP[C4]		FTP[C3]		FTP[C4]	
	Mean	C95%	Mean	C95%	Mean	C95%	Mean	C95%
1	101674 bps	3674 bps	99626 bps	3654 bps	0 bps	0 bps	30860 bps	7477 bps
2	91623 bps	3437 bps	105764 bps	3851 bps	188157 bps	8962 bps	45603 bps	7776 bps
3	91950 bps	3616 bps	101265 bps	3666 bps	0 bps	1 bps	306370 bps	6449 bps
4	98852 bps	3624 bps	97374 bps	3738 bps	0 bps	1 bps	87382 bps	14669 bps
5	98353 bps	3791 bps	109142 bps	3826 bps	81676 bps	8197 bps	0 bps	3 bps
6	97397 bps	3810 bps	95240 bps	3690 bps	365953 bps	6454 bps	167884 bps	7160 bps

Table 6.3: Per client throughput results, CASE: BE with similar RTTs



	VoIP[C5]		VoIP[C6]		VoIP[C7]		VoIP[C8]	
	Mean	C95%	Mean	C95%	Mean	C95%	Mean	C95%
C50	9 ms	1.63 ms	10 ms	0 ms	7.92 ms	3.31 ms	10 ms	0 ms
C95	24.73 ms	5.74 ms	25.67 ms	6.22 ms	23.83 ms	6.32 ms	26.33 ms	5.43 ms

	HTTP[C3]		HTTP[C4]		FTP[C3]		FTP[C4]	
	Mean	C95%	Mean	C95%	Mean	C95%	Mean	C95%
C50	10 ms	0 ms	10 ms	0 ms	19.33 ms	15.39 ms	15.7 ms	7.57 ms
C95	580.33 ms	7.74 ms	575.3 ms	16.32 ms	24126.67 ms	40341.61 ms	11164.83 ms	33241.87 ms

Table 6.4: Per client jitter results, CASE: BE with similar RTTs

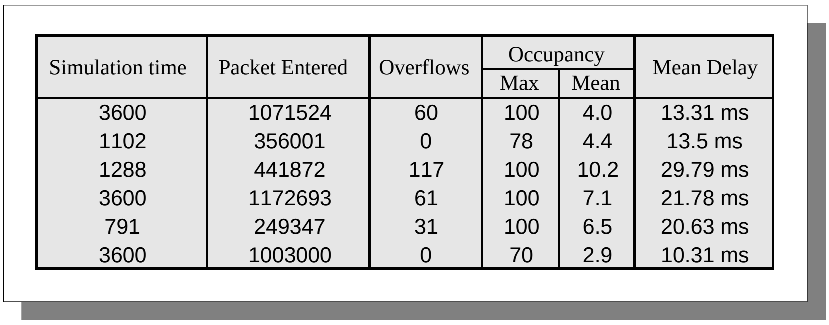
> *"Short TCP connection, like HTTP object transfer, operates on slow start mode where it effectively doubles its sending rate on each RTT. This causes overload to the network and forces long live TCP connections to time out their connections."*

This could be the reason as there is a new HTTP session in every second in the network. Moreover, as these sessions start from the slow start phase they will cause instantaneous overload to the network.

Table 6.4 present jitter results of the simulation. Results show that median jitter between clients is on the same level, but 95% percentile jitter values vary a lot between applications using TCP and UDP. UDP based VoIP has much lower jitter than HTTP and FTP which use TCP. Ratio of jitter is roughly 20 to 1.

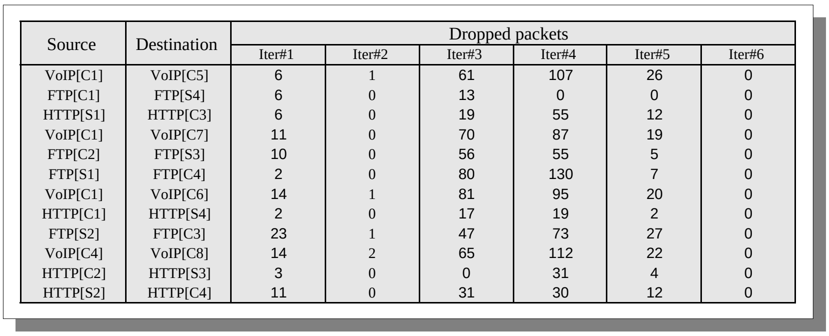
6.4.2.2 BE with dissimilar RTT

This section deals with the results from a simulation where 10 times larger delay was used on the link between routers R7 and R8. This makes the fixed part of RTT for clients VoIP[C5], VoIP[C6], FTP[C3] and HTTP[C3] to be three times what it is with the rest of the clients. Table 6.5 provides general statistics from the buffer of the bottleneck link between R6 and R7. Simulation time was again set to be 3600 s but the same anomaly in the TCP caused simulations to abort earlier in iterations 1, 3 and 5.



Simulation time	Packet Entered	Overflows	Occupancy		Mean Delay
			Max	Mean	
3600	1071524	60	100	4.0	13.31 ms
1102	356001	0	78	4.4	13.5 ms
1288	441872	117	100	10.2	29.79 ms
3600	1172693	61	100	7.1	21.78 ms
791	249347	31	100	6.5	20.63 ms
3600	1003000	0	70	2.9	10.31 ms

Table 6.5: General buffer statistics from router R6, CASE: BE with dissimilar RTTs



Source	Destination	Dropped packets					
		Iter#1	Iter#2	Iter#3	Iter#4	Iter#5	Iter#6
VoIP[C1]	VoIP[C5]	6	1	61	107	26	0
FTP[C1]	FTP[S4]	6	0	13	0	0	0
HTTP[S1]	HTTP[C3]	6	0	19	55	12	0
VoIP[C1]	VoIP[C7]	11	0	70	87	19	0
FTP[C2]	FTP[S3]	10	0	56	55	5	0
FTP[S1]	FTP[C4]	2	0	80	130	7	0
VoIP[C1]	VoIP[C6]	14	1	81	95	20	0
HTTP[C1]	HTTP[S4]	2	0	17	19	2	0
FTP[S2]	FTP[C3]	23	1	47	73	27	0
VoIP[C4]	VoIP[C8]	14	2	65	112	22	0
HTTP[C2]	HTTP[S3]	3	0	0	31	4	0
HTTP[S2]	HTTP[C4]	11	0	31	30	12	0

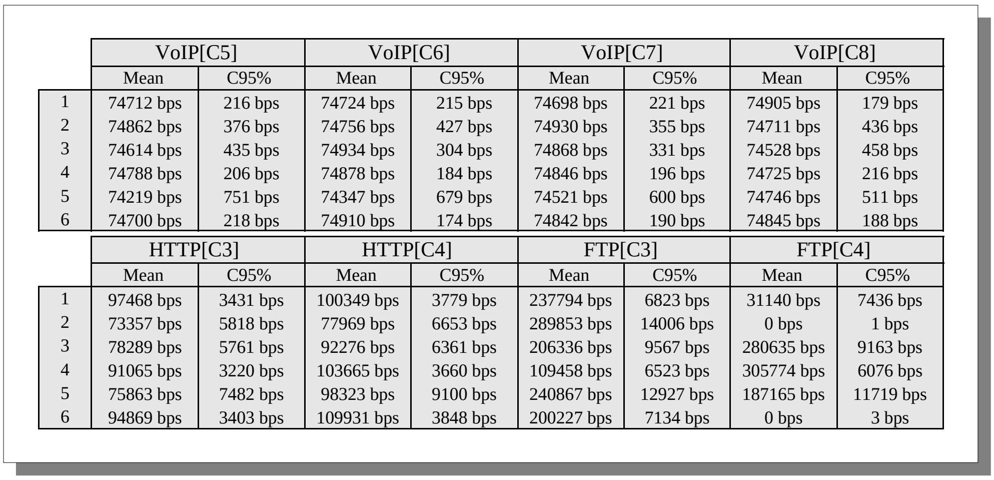
Table 6.6: RED statistics from router R6, CASE: BE with dissimilar RTTs

Table 6.6 shows numerical results of packet drops in the RED control of the same buffer. The results show that VoIP clients lose more packets in RED than TCP clients do. This is expected as they are not able to control their sending rate at the times of congestion. However, also FTP connections exhibit a large number of packet drops in some iterations. In addition, connections with shorter RTT have higher packet loss due to their more aggressive window control.

Throughput results, in Table 6.7, confirm the expected behavior of the TCP. Connections with shorter RTT reclaim more bandwidth than connections with longer RTT. This is due to rate of window opening[4], which is slower with long RTT connections. This is very well observable from results of HTTP[C3] and HTTP[C4] that are not affected by the problem of HTTP/FTP interference, explained in the previous section. Throughput difference between these two clients is roughly 1.15 while the fixed RTT difference between them is 2.6. However, as explained earlier this is only the fixed component of the RTT. To receive overall RTT, one has to add queuing delays, which are on the average 40 ms in these simulations.

Table 6.8 presents jitter results of the simulations. The results show a similar behavior as in the case with similar RTTs; median jitter is about the same but 95% percentile values have the same difference between VoIP and others. However, there is also a difference in jitter between clients in low and high delay paths. This is also as expected since the number of packets in congested queue is predominantly occupied by clients using the low delay path (this holds only for TCP clients). In general, this makes the experienced jitter lower. With VoIP,

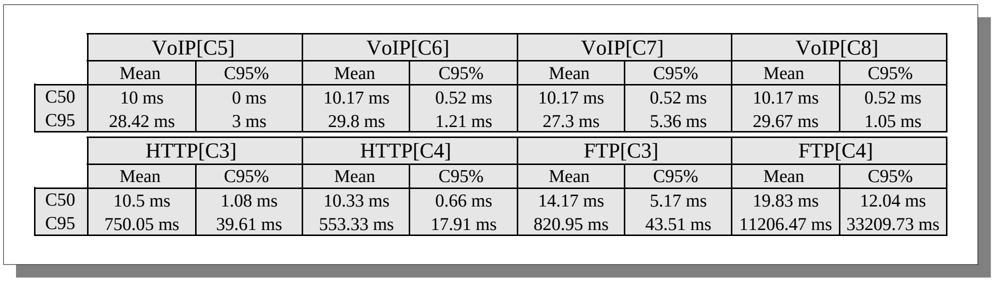
[4] Effect of RTT to TCP operation was explained in Section 5.4.4.



	VoIP[C5]		VoIP[C6]		VoIP[C7]		VoIP[C8]	
	Mean	C95%	Mean	C95%	Mean	C95%	Mean	C95%
1	74712 bps	216 bps	74724 bps	215 bps	74698 bps	221 bps	74905 bps	179 bps
2	74862 bps	376 bps	74756 bps	427 bps	74930 bps	355 bps	74711 bps	436 bps
3	74614 bps	435 bps	74934 bps	304 bps	74868 bps	331 bps	74528 bps	458 bps
4	74788 bps	206 bps	74878 bps	184 bps	74846 bps	196 bps	74725 bps	216 bps
5	74219 bps	751 bps	74347 bps	679 bps	74521 bps	600 bps	74746 bps	511 bps
6	74700 bps	218 bps	74910 bps	174 bps	74842 bps	190 bps	74845 bps	188 bps

	HTTP[C3]		HTTP[C4]		FTP[C3]		FTP[C4]	
	Mean	C95%	Mean	C95%	Mean	C95%	Mean	C95%
1	97468 bps	3431 bps	100349 bps	3779 bps	237794 bps	6823 bps	31140 bps	7436 bps
2	73357 bps	5818 bps	77969 bps	6653 bps	289853 bps	14006 bps	0 bps	1 bps
3	78289 bps	5761 bps	92276 bps	6361 bps	206336 bps	9567 bps	280635 bps	9163 bps
4	91065 bps	3220 bps	103665 bps	3660 bps	109458 bps	6523 bps	305774 bps	6076 bps
5	75863 bps	7482 bps	98323 bps	9100 bps	240867 bps	12927 bps	187165 bps	11719 bps
6	94869 bps	3403 bps	109931 bps	3848 bps	200227 bps	7134 bps	0 bps	3 bps

Table 6.7: Per client throughput results, CASE: BE with dissimilar RTTs



	VoIP[C5]		VoIP[C6]		VoIP[C7]		VoIP[C8]	
	Mean	C95%	Mean	C95%	Mean	C95%	Mean	C95%
C50	10 ms	0 ms	10.17 ms	0.52 ms	10.17 ms	0.52 ms	10.17 ms	0.52 ms
C95	28.42 ms	3 ms	29.8 ms	1.21 ms	27.3 ms	5.36 ms	29.67 ms	1.05 ms

	HTTP[C3]		HTTP[C4]		FTP[C3]		FTP[C4]	
	Mean	C95%	Mean	C95%	Mean	C95%	Mean	C95%
C50	10.5 ms	1.08 ms	10.33 ms	0.66 ms	14.17 ms	5.17 ms	19.83 ms	12.04 ms
C95	750.05 ms	39.61 ms	553.33 ms	17.91 ms	820.95 ms	43.51 ms	11206.47 ms	33209.73 ms

Table 6.8: Per client jitter results, CASE: BE with dissimilar RTTs

there is no rate control and the difference in jitter is therefore lower.

6.4.2.3 Conclusions from BE simulations

The Best Effort simulations provide a reference case to which different PHB scenarios can be compared. Results of the BE simulation were as expected. UDP sources dominate over the TCP sources in the bottleneck link, causing TCP clients to have lower throughput. The effect of the RTT on the throughput was also as expected. Shorter RTT allows the TCP to have more aggressive window control, which leads to a higher throughput.

6.4.3 EF+BE Simulations

This section presents simulation results of the case where EF PHB is applied to the VoIP connections and the rest of the traffic is relayed by conventional best effort service. Questions which we want to answer at this point are:

1. *What is the differentiation factor which can be achieved by using EF PHB for a part of the traffic*

2. *What is the effect of selected scheduling to this differentiation*

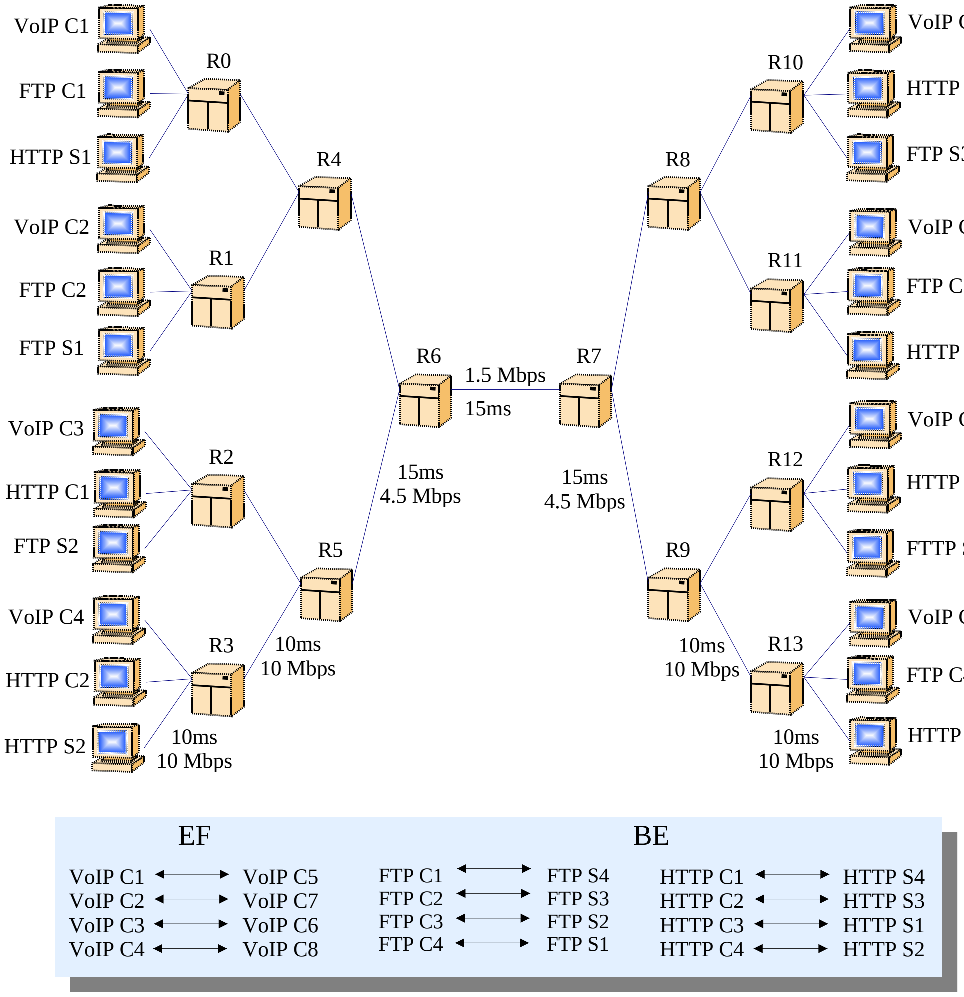


Figure 6.13: Network topology and connection pairs in the EF+BE simulation

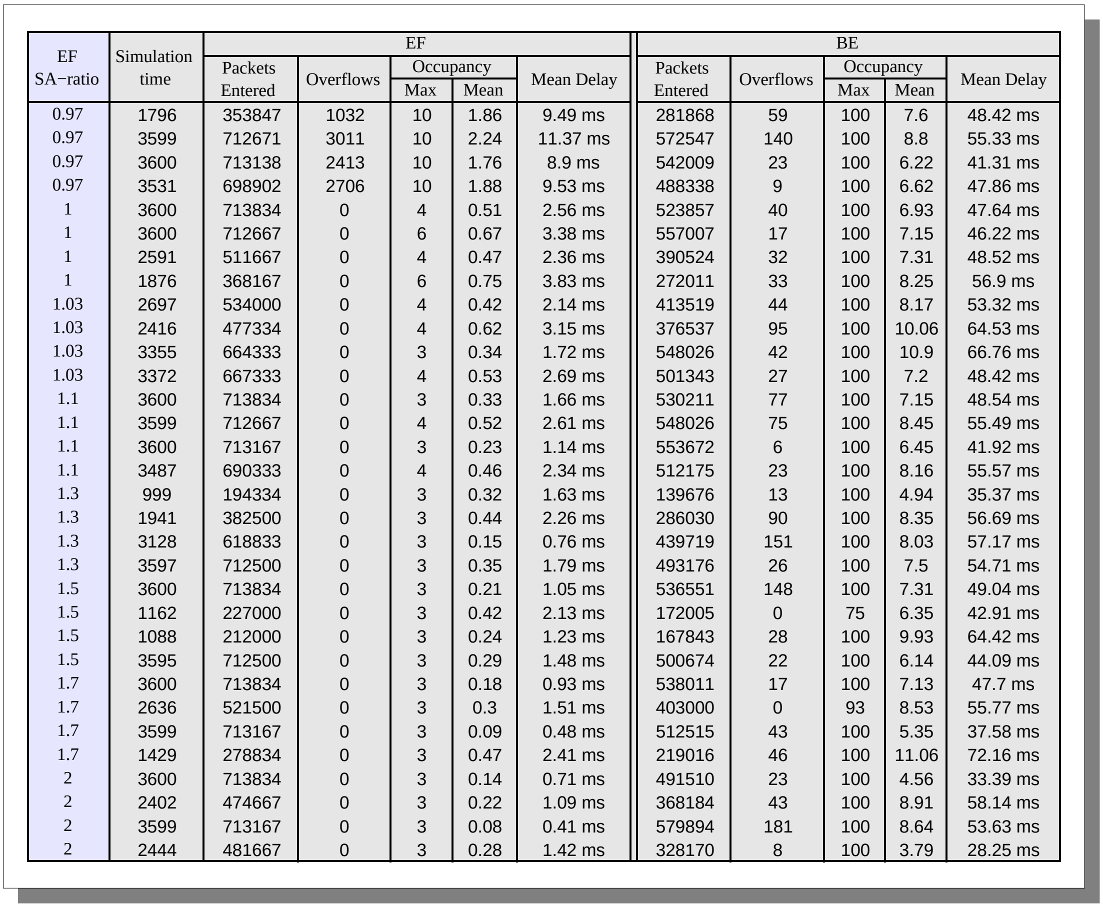
Network topology and connection pairs are presented in Figure 6.13. Simulations were executed with two different configuration options:

1. **Priority scheduling**
2. **Weighted Round Robin scheduling**, service to arrival (SA) ration of the EF class is varied from 0.97 to 2 times of the arrival rate to the EF class on the bottleneck link (link between routers R6 and R7).

Simulations were executed with similar RTT's, because delay has no effect on the posed questions.

6.4.3.1 EF+BE with WRR scheduling

Table 6.9 shows general statistics from the buffers in the router R6, when WRR scheduling was used. Quality separation on the link was made by adjusting the scheduler weights. Service to arrival ratio expresses the EF class provisioning in relation to the subscription of the class. The first set of iterations shows results of underprovisioned network (SA=0.97). Underprovisioning causes the



EF SA−ratio	Simulation time	EF					BE				
		Packets Entered	Overflows	Occupancy		Mean Delay	Packets Entered	Overflows	Occupancy		Mean Delay
				Max	Mean				Max	Mean	
0.97	1796	353847	1032	10	1.86	9.49 ms	281868	59	100	7.6	48.42 ms
0.97	3599	712671	3011	10	2.24	11.37 ms	572547	140	100	8.8	55.33 ms
0.97	3600	713138	2413	10	1.76	8.9 ms	542009	23	100	6.22	41.31 ms
0.97	3531	698902	2706	10	1.88	9.53 ms	488338	9	100	6.62	47.86 ms
1	3600	713834	0	4	0.51	2.56 ms	523857	40	100	6.93	47.64 ms
1	3600	712667	0	6	0.67	3.38 ms	557007	17	100	7.15	46.22 ms
1	2591	511667	0	4	0.47	2.36 ms	390524	32	100	7.31	48.52 ms
1	1876	368167	0	6	0.75	3.83 ms	272011	33	100	8.25	56.9 ms
1.03	2697	534000	0	4	0.42	2.14 ms	413519	44	100	8.17	53.32 ms
1.03	2416	477334	0	4	0.62	3.15 ms	376537	95	100	10.06	64.53 ms
1.03	3355	664333	0	3	0.34	1.72 ms	548026	42	100	10.9	66.76 ms
1.03	3372	667333	0	4	0.53	2.69 ms	501343	27	100	7.2	48.42 ms
1.1	3600	713834	0	3	0.33	1.66 ms	530211	77	100	7.15	48.54 ms
1.1	3599	712667	0	4	0.52	2.61 ms	548026	75	100	8.45	55.49 ms
1.1	3600	713167	0	3	0.23	1.14 ms	553672	6	100	6.45	41.92 ms
1.1	3487	690333	0	4	0.46	2.34 ms	512175	23	100	8.16	55.57 ms
1.3	999	194334	0	3	0.32	1.63 ms	139676	13	100	4.94	35.37 ms
1.3	1941	382500	0	3	0.44	2.26 ms	286030	90	100	8.35	56.69 ms
1.3	3128	618833	0	3	0.15	0.76 ms	439719	151	100	8.03	57.17 ms
1.3	3597	712500	0	3	0.35	1.79 ms	493176	26	100	7.5	54.71 ms
1.5	3600	713834	0	3	0.21	1.05 ms	536551	148	100	7.31	49.04 ms
1.5	1162	227000	0	3	0.42	2.13 ms	172005	0	75	6.35	42.91 ms
1.5	1088	212000	0	3	0.24	1.23 ms	167843	28	100	9.93	64.42 ms
1.5	3595	712500	0	3	0.29	1.48 ms	500674	22	100	6.14	44.09 ms
1.7	3600	713834	0	3	0.18	0.93 ms	538011	17	100	7.13	47.7 ms
1.7	2636	521500	0	3	0.3	1.51 ms	403000	0	93	8.53	55.77 ms
1.7	3599	713167	0	3	0.09	0.48 ms	512515	43	100	5.35	37.58 ms
1.7	1429	278834	0	3	0.47	2.41 ms	219016	46	100	11.06	72.16 ms
2	3600	713834	0	3	0.14	0.71 ms	491510	23	100	4.56	33.39 ms
2	2402	474667	0	3	0.22	1.09 ms	368184	43	100	8.91	58.14 ms
2	3599	713167	0	3	0.08	0.41 ms	579894	181	100	8.64	53.63 ms
2	2444	481667	0	3	0.28	1.42 ms	328170	8	100	3.79	28.25 ms

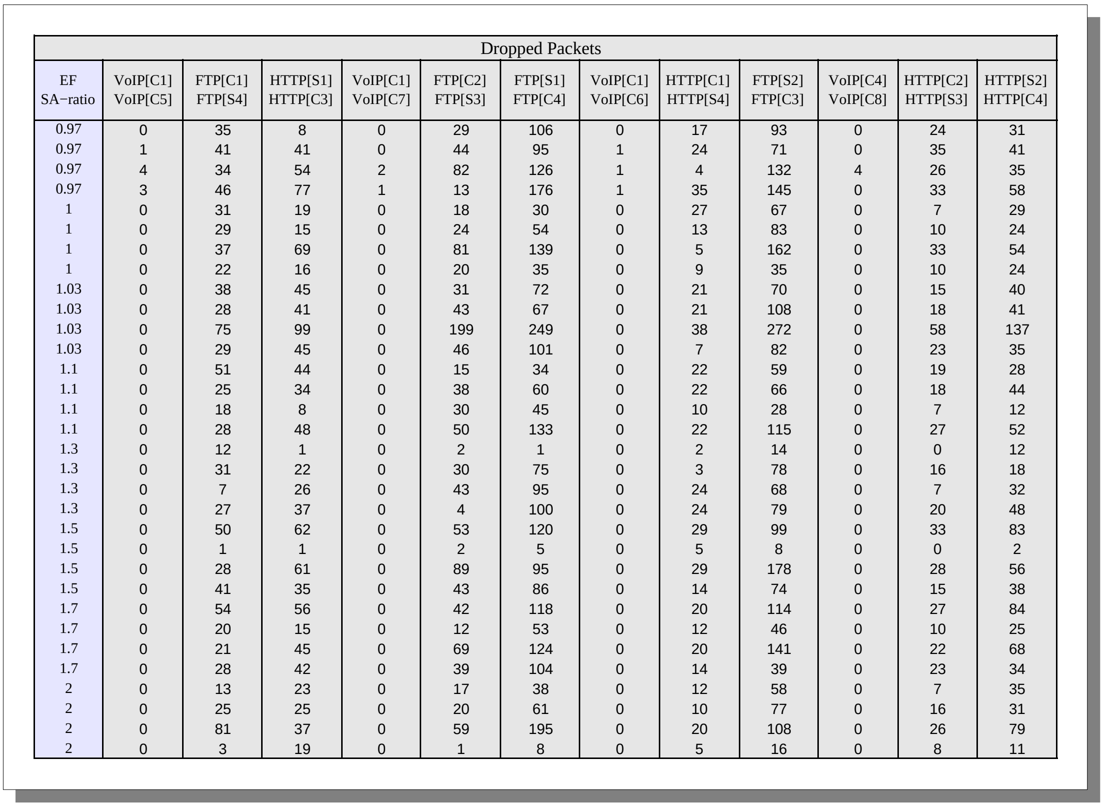
Table 6.9: General buffer statistics from the router R6, CASE: EF with WRR scheduling

queue of the EF class to build up to a level where packets are dropped due to overflow. Following iterations were provisioned to have an equal or greater capacity compared to the subscription at the class.

Table 6.10 gives drop statistics of the RED control from the same buffer. These results show that VoIP clients do not lose packets in the RED except in the underprovisioned case.

Figure 6.14 and Table 6.11 present throughput results of the simulations. Figure 6.14 presents combined results, i.e. all results from identical clients are combined. Confidence levels are calculated from independent replications of simulation (viz. four iterations per SA-ratio). VoIP connections clearly receive resources which they need and are therefore satisfied with the service. Variation in transmission rate is due to call arrivals and departures (which are random processes). Throughput of the FTP clients is now much better than it was in the case of best effort simulation. The reason for this must be that the UDP traffic (VoIP) has now a separate queue and leaves more room for the TCP traffic.

Figures 6.15 and 6.16 present jitter results of the simulations. Results show that median jitter is 10 times higher in the BE class than in the EF class. The SA-ratio does not have an effect on the median jitter. 95% percentile values have still a 10-fold difference between classes. However, in this case the EF class has a decreasing jitter and jitter variance as a function of the SA-ratio. If low jitter



Dropped Packets												
EF SA−ratio	VoIP[C1] VoIP[C5]	FTP[C1] FTP[S4]	HTTP[S1] HTTP[C3]	VoIP[C1] VoIP[C7]	FTP[C2] FTP[S3]	FTP[S1] FTP[C4]	VoIP[C1] VoIP[C6]	HTTP[C1] HTTP[S4]	FTP[S2] FTP[C3]	VoIP[C4] VoIP[C8]	HTTP[C2] HTTP[S3]	HTTP[S2] HTTP[C4]
0.97	0	35	8	0	29	106	0	17	93	0	24	31
0.97	1	41	41	0	44	95	1	24	71	0	35	41
0.97	4	34	54	2	82	126	1	4	132	4	26	35
0.97	3	46	77	1	13	176	1	35	145	0	33	58
1	0	31	19	0	18	30	0	27	67	0	7	29
1	0	29	15	0	24	54	0	13	83	0	10	24
1	0	37	69	0	81	139	0	5	162	0	33	54
1	0	22	16	0	20	35	0	9	35	0	10	24
1.03	0	38	45	0	31	72	0	21	70	0	15	40
1.03	0	28	41	0	43	67	0	21	108	0	18	41
1.03	0	75	99	0	199	249	0	38	272	0	58	137
1.03	0	29	45	0	46	101	0	7	82	0	23	35
1.1	0	51	44	0	15	34	0	22	59	0	19	28
1.1	0	25	34	0	38	60	0	22	66	0	18	44
1.1	0	18	8	0	30	45	0	10	28	0	7	12
1.1	0	28	48	0	50	133	0	22	115	0	27	52
1.3	0	12	1	0	2	1	0	2	14	0	0	12
1.3	0	31	22	0	30	75	0	3	78	0	16	18
1.3	0	7	26	0	43	95	0	24	68	0	7	32
1.3	0	27	37	0	4	100	0	24	79	0	20	48
1.5	0	50	62	0	53	120	0	29	99	0	33	83
1.5	0	1	1	0	2	5	0	5	8	0	0	2
1.5	0	28	61	0	89	95	0	29	178	0	28	56
1.5	0	41	35	0	43	86	0	14	74	0	15	38
1.7	0	54	56	0	42	118	0	20	114	0	27	84
1.7	0	20	15	0	12	53	0	12	46	0	10	25
1.7	0	21	45	0	69	124	0	20	141	0	22	68
1.7	0	28	42	0	39	104	0	14	39	0	23	34
2	0	13	23	0	17	38	0	12	58	0	7	35
2	0	25	25	0	20	61	0	10	77	0	16	31
2	0	81	37	0	59	195	0	20	108	0	26	79
2	0	3	19	0	1	8	0	5	16	0	8	11

Table 6.10: RED statistics from the router R6, CASE: EF with WRR scheduling

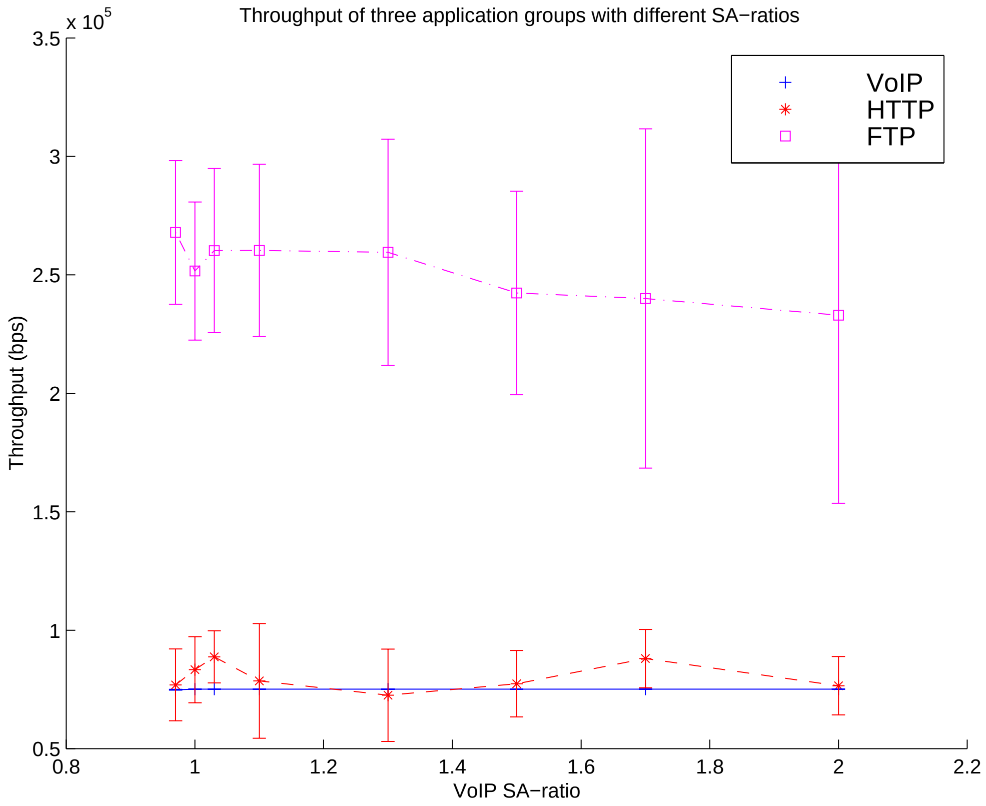
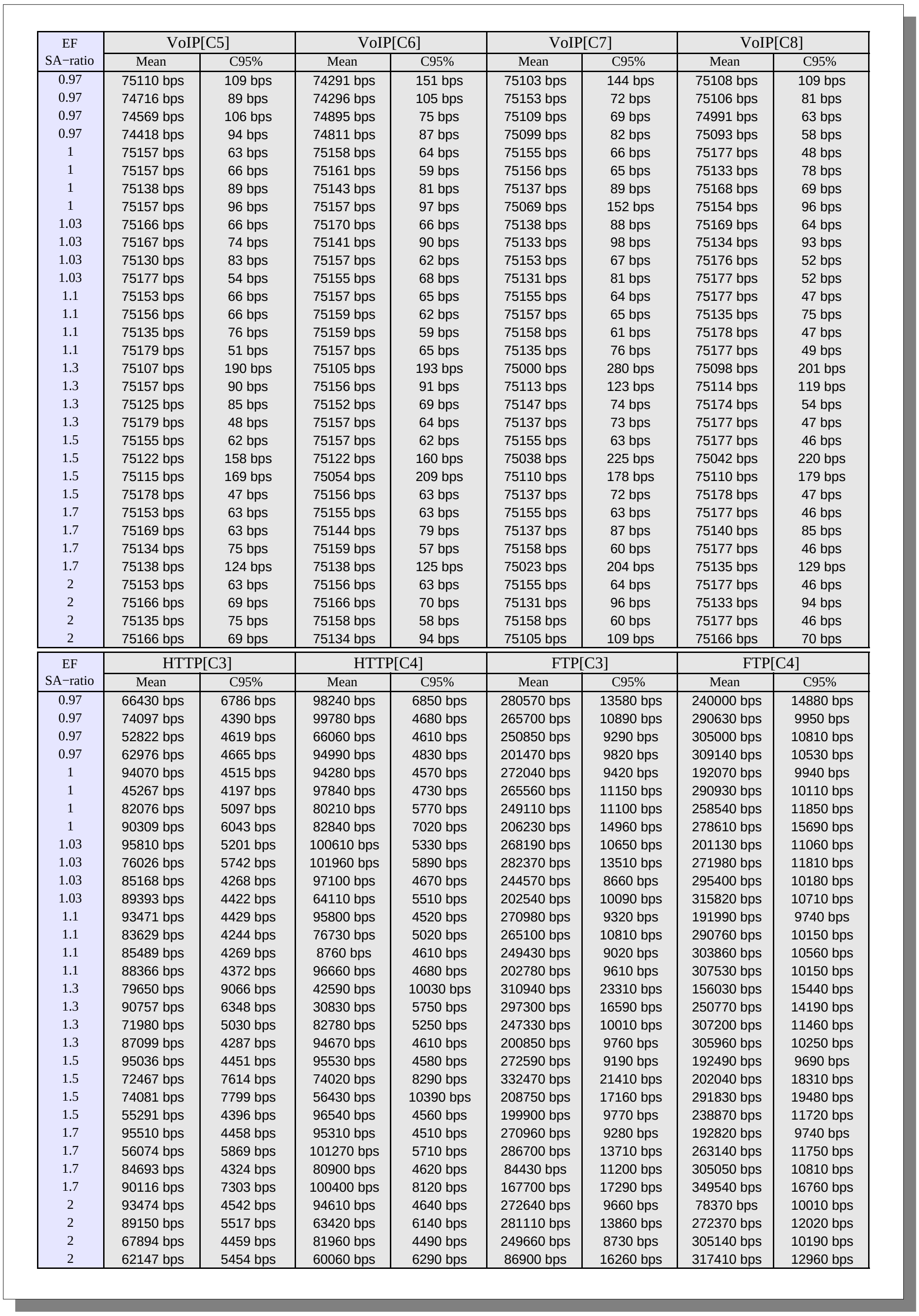


Figure 6.14: Mean throughput with 95% confidence intervals, Case: EF with WRR scheduling



EF	VoIP[C5]		VoIP[C6]		VoIP[C7]		VoIP[C8]	
SA−ratio	Mean	C95%	Mean	C95%	Mean	C95%	Mean	C95%
0.97	75110 bps	109 bps	74291 bps	151 bps	75103 bps	144 bps	75108 bps	109 bps
0.97	74716 bps	89 bps	74296 bps	105 bps	75153 bps	72 bps	75106 bps	81 bps
0.97	74569 bps	106 bps	74895 bps	75 bps	75109 bps	69 bps	74991 bps	63 bps
0.97	74418 bps	94 bps	74811 bps	87 bps	75099 bps	82 bps	75093 bps	58 bps
1	75157 bps	63 bps	75158 bps	64 bps	75155 bps	66 bps	75177 bps	48 bps
1	75157 bps	66 bps	75161 bps	59 bps	75156 bps	65 bps	75133 bps	78 bps
1	75138 bps	89 bps	75143 bps	81 bps	75137 bps	89 bps	75168 bps	69 bps
1	75157 bps	96 bps	75157 bps	97 bps	75069 bps	152 bps	75154 bps	96 bps
1.03	75166 bps	66 bps	75170 bps	66 bps	75138 bps	88 bps	75169 bps	64 bps
1.03	75167 bps	74 bps	75141 bps	90 bps	75133 bps	98 bps	75134 bps	93 bps
1.03	75130 bps	83 bps	75157 bps	62 bps	75153 bps	67 bps	75176 bps	52 bps
1.03	75177 bps	54 bps	75155 bps	68 bps	75131 bps	81 bps	75177 bps	52 bps
1.1	75153 bps	66 bps	75157 bps	65 bps	75155 bps	64 bps	75177 bps	47 bps
1.1	75156 bps	66 bps	75159 bps	62 bps	75157 bps	65 bps	75135 bps	75 bps
1.1	75135 bps	76 bps	75159 bps	59 bps	75158 bps	61 bps	75178 bps	47 bps
1.1	75179 bps	51 bps	75157 bps	65 bps	75135 bps	76 bps	75177 bps	49 bps
1.3	75107 bps	190 bps	75105 bps	193 bps	75000 bps	280 bps	75098 bps	201 bps
1.3	75157 bps	90 bps	75156 bps	91 bps	75113 bps	123 bps	75114 bps	119 bps
1.3	75125 bps	85 bps	75152 bps	69 bps	75147 bps	74 bps	75174 bps	54 bps
1.3	75179 bps	48 bps	75157 bps	64 bps	75137 bps	73 bps	75177 bps	47 bps
1.5	75155 bps	62 bps	75157 bps	62 bps	75155 bps	63 bps	75177 bps	46 bps
1.5	75122 bps	158 bps	75122 bps	160 bps	75038 bps	225 bps	75042 bps	220 bps
1.5	75115 bps	169 bps	75054 bps	209 bps	75110 bps	178 bps	75110 bps	179 bps
1.5	75178 bps	47 bps	75156 bps	63 bps	75137 bps	72 bps	75178 bps	47 bps
1.7	75153 bps	63 bps	75155 bps	63 bps	75155 bps	63 bps	75177 bps	46 bps
1.7	75169 bps	63 bps	75144 bps	79 bps	75137 bps	87 bps	75140 bps	85 bps
1.7	75134 bps	75 bps	75159 bps	57 bps	75158 bps	60 bps	75177 bps	46 bps
1.7	75138 bps	124 bps	75138 bps	125 bps	75023 bps	204 bps	75135 bps	129 bps
2	75153 bps	63 bps	75156 bps	63 bps	75155 bps	64 bps	75177 bps	46 bps
2	75166 bps	69 bps	75166 bps	70 bps	75131 bps	96 bps	75133 bps	94 bps
2	75135 bps	75 bps	75158 bps	58 bps	75158 bps	60 bps	75177 bps	46 bps
2	75166 bps	69 bps	75134 bps	94 bps	75105 bps	109 bps	75166 bps	70 bps

EF	HTTP[C3]		HTTP[C4]		FTP[C3]		FTP[C4]	
SA−ratio	Mean	C95%	Mean	C95%	Mean	C95%	Mean	C95%
0.97	66430 bps	6786 bps	98240 bps	6850 bps	280570 bps	13580 bps	240000 bps	14880 bps
0.97	74097 bps	4390 bps	99780 bps	4680 bps	265700 bps	10890 bps	290630 bps	9950 bps
0.97	52822 bps	4619 bps	66060 bps	4610 bps	250850 bps	9290 bps	305000 bps	10810 bps
0.97	62976 bps	4665 bps	94990 bps	4830 bps	201470 bps	9820 bps	309140 bps	10530 bps
1	94070 bps	4515 bps	94280 bps	4570 bps	272040 bps	9420 bps	192070 bps	9940 bps
1	45267 bps	4197 bps	97840 bps	4730 bps	265560 bps	11150 bps	290930 bps	10110 bps
1	82076 bps	5097 bps	80210 bps	5770 bps	249110 bps	11100 bps	258540 bps	11850 bps
1	90309 bps	6043 bps	82840 bps	7020 bps	206230 bps	14960 bps	278610 bps	15690 bps
1.03	95810 bps	5201 bps	100610 bps	5330 bps	268190 bps	10650 bps	201130 bps	11060 bps
1.03	76026 bps	5742 bps	101960 bps	5890 bps	282370 bps	13510 bps	271980 bps	11810 bps
1.03	85168 bps	4268 bps	97100 bps	4670 bps	244570 bps	8660 bps	295400 bps	10180 bps
1.03	89393 bps	4422 bps	64110 bps	5510 bps	202540 bps	10090 bps	315820 bps	10710 bps
1.1	93471 bps	4429 bps	95800 bps	4520 bps	270980 bps	9320 bps	191990 bps	9740 bps
1.1	83629 bps	4244 bps	76730 bps	5020 bps	265100 bps	10810 bps	290760 bps	10150 bps
1.1	85489 bps	4269 bps	8760 bps	4610 bps	249430 bps	9020 bps	303860 bps	10560 bps
1.1	88366 bps	4372 bps	96660 bps	4680 bps	202780 bps	9610 bps	307530 bps	10150 bps
1.3	79650 bps	9066 bps	42590 bps	10030 bps	310940 bps	23310 bps	156030 bps	15440 bps
1.3	90757 bps	6348 bps	30830 bps	5750 bps	297300 bps	16590 bps	250770 bps	14190 bps
1.3	71980 bps	5030 bps	82780 bps	5250 bps	247330 bps	10010 bps	307200 bps	11460 bps
1.3	87099 bps	4287 bps	94670 bps	4610 bps	200850 bps	9760 bps	305960 bps	10250 bps
1.5	95036 bps	4451 bps	95530 bps	4580 bps	272590 bps	9190 bps	192490 bps	9690 bps
1.5	72467 bps	7614 bps	74020 bps	8290 bps	332470 bps	21410 bps	202040 bps	18310 bps
1.5	74081 bps	7799 bps	56430 bps	10390 bps	208750 bps	17160 bps	291830 bps	19480 bps
1.5	55291 bps	4396 bps	96540 bps	4560 bps	199900 bps	9770 bps	238870 bps	11720 bps
1.7	95510 bps	4458 bps	95310 bps	4510 bps	270960 bps	9280 bps	192820 bps	9740 bps
1.7	56074 bps	5869 bps	101270 bps	5710 bps	286700 bps	13710 bps	263140 bps	11750 bps
1.7	84693 bps	4324 bps	80900 bps	4620 bps	84430 bps	11200 bps	305050 bps	10810 bps
1.7	90116 bps	7303 bps	100400 bps	8120 bps	167700 bps	17290 bps	349540 bps	16760 bps
2	93474 bps	4542 bps	94610 bps	4640 bps	272640 bps	9660 bps	78370 bps	10010 bps
2	89150 bps	5517 bps	63420 bps	6140 bps	281110 bps	13860 bps	272370 bps	12020 bps
2	67894 bps	4459 bps	81960 bps	4490 bps	249660 bps	8730 bps	305140 bps	10190 bps
2	62147 bps	5454 bps	60060 bps	6290 bps	86900 bps	16260 bps	317410 bps	12960 bps

Table 6.11: Per client throughput results, Case: EF with WRR scheduling

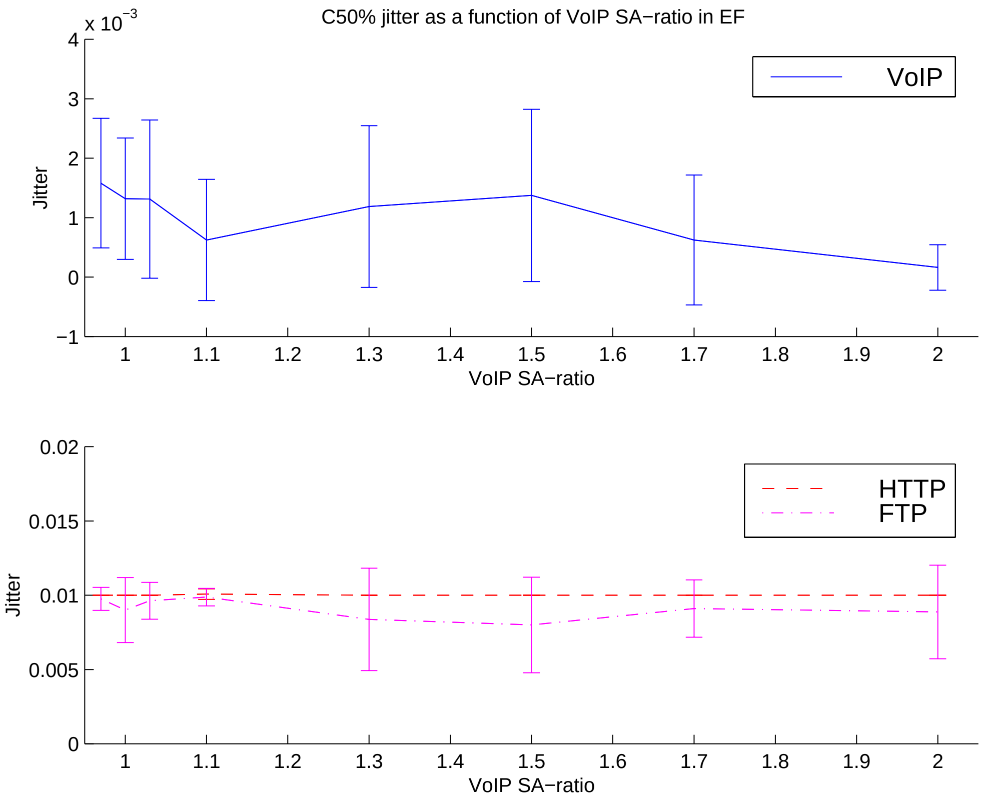


Figure 6.15: Median jitter with 95% confidence intervals, Case: EF with WRR scheduling'

and jitter variance are wanted a SA-ratio of 1.7 or higher should be used. Jitter of the TCP clients is in same order of magnitude irrespective of resource share given to BE class.

6.4.3.2 EF+BE with priority scheduling

Table 6.12 provides general statistics from the buffer of the bottleneck link in the router R6. Scheduling between classes is now priority based. Table 6.13 shows packet droppings of the RED control from the same buffer. Results show that VoIP clients do not lose packets in the RED in any occasion. This is what one would expect when they have hard priority over other traffic in the network. Simulation times were short due to complications in the TCP control in this environment.

Throughput results in Table 6.14 show that VoIP receives all the resources it needs for communication. Rest of the capacity, which is left to the BE class, is not fairly shared. This is again an indication of interference between TCP flows with short and long RTTs. Other reason for this problem might be that hard priority uses resources so aggressively that there are moments when no packets from the BE class are served. This should not, however, be the case at this time, because EF class has only 20% utilization on the bottleneck link.

Jitter, in Table 6.15, is consistently small in the EF class. This is what one would expect when the EF traffic has priority over other traffic.

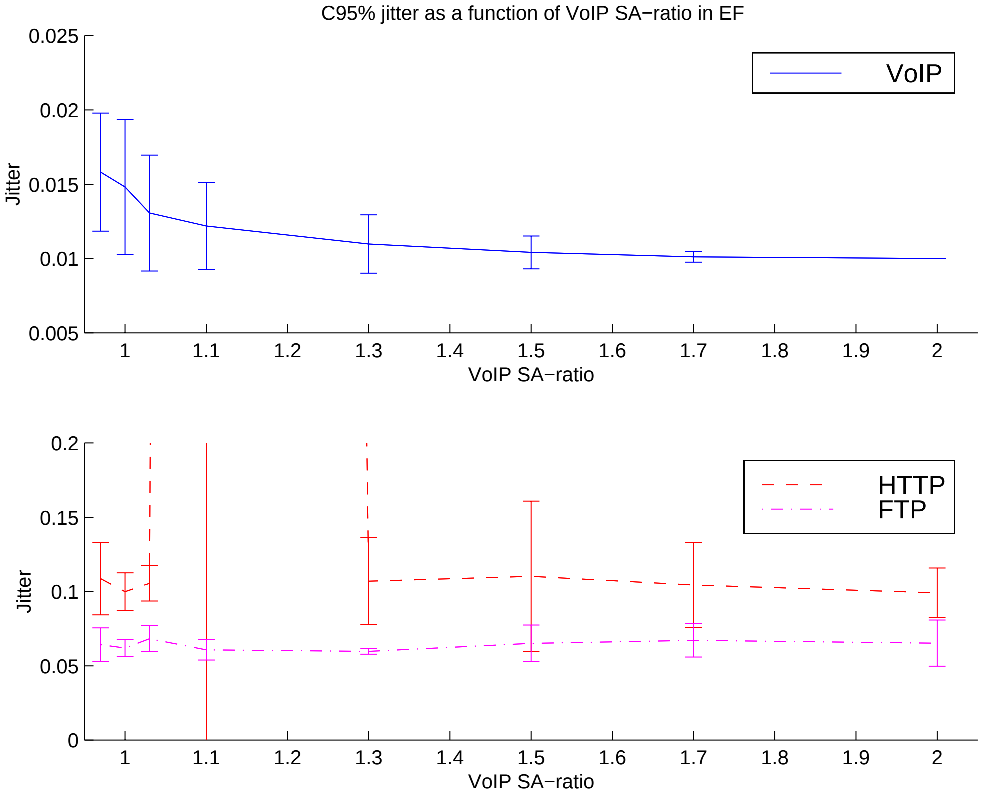
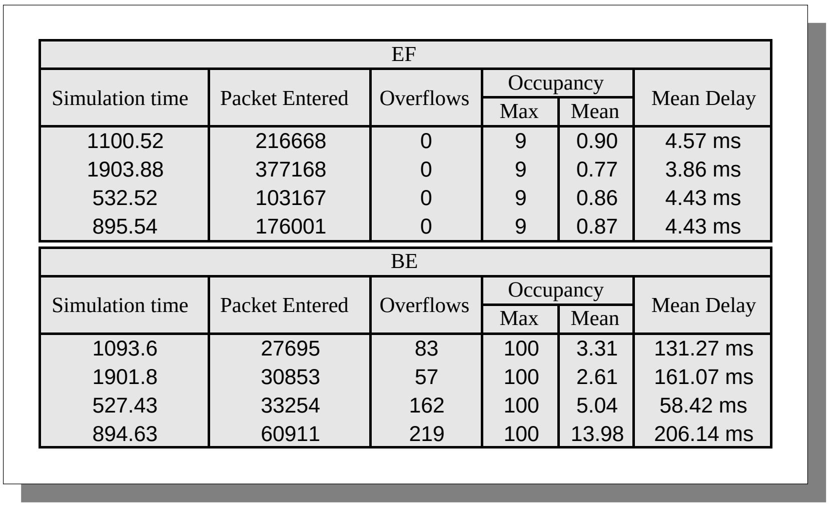


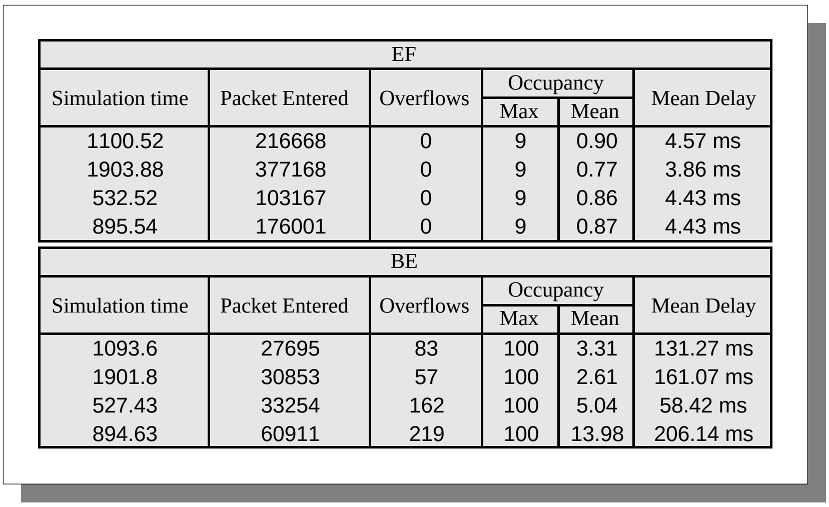
Figure 6.16: 95% percentile jitter with 95% confidence intervals, Case: EF with WRR scheduling



EF					
Simulation time	Packet Entered	Overflows	Occupancy		Mean Delay
			Max	Mean	
1100.52	216668	0	9	0.90	4.57 ms
1903.88	377168	0	9	0.77	3.86 ms
532.52	103167	0	9	0.86	4.43 ms
895.54	176001	0	9	0.87	4.43 ms

BE					
Simulation time	Packet Entered	Overflows	Occupancy		Mean Delay
			Max	Mean	
1093.6	27695	83	100	3.31	131.27 ms
1901.8	30853	57	100	2.61	161.07 ms
527.43	33254	162	100	5.04	58.42 ms
894.63	60911	219	100	13.98	206.14 ms

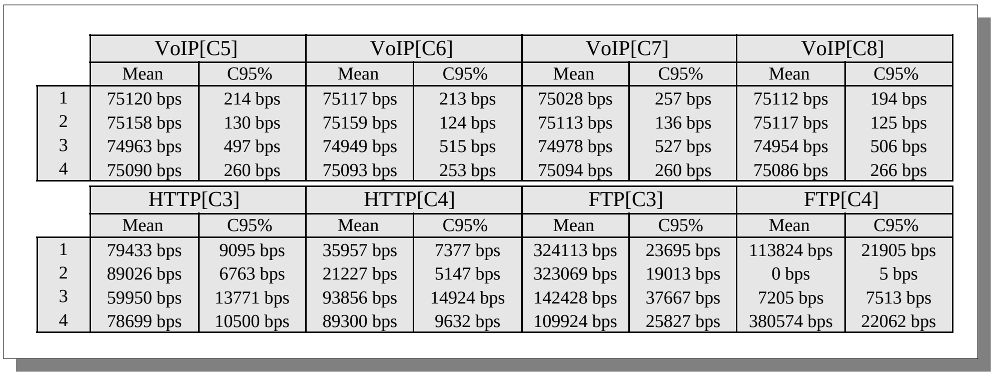
Table 6.12: General statistics from router R6 CASE: EF with priority scheduling



EF					
Simulation time	Packet Entered	Overflows	Occupancy		Mean Delay
			Max	Mean	
1100.52	216668	0	9	0.90	4.57 ms
1903.88	377168	0	9	0.77	3.86 ms
532.52	103167	0	9	0.86	4.43 ms
895.54	176001	0	9	0.87	4.43 ms

BE					
Simulation time	Packet Entered	Overflows	Occupancy		Mean Delay
			Max	Mean	
1093.6	27695	83	100	3.31	131.27 ms
1901.8	30853	57	100	2.61	161.07 ms
527.43	33254	162	100	5.04	58.42 ms
894.63	60911	219	100	13.98	206.14 ms

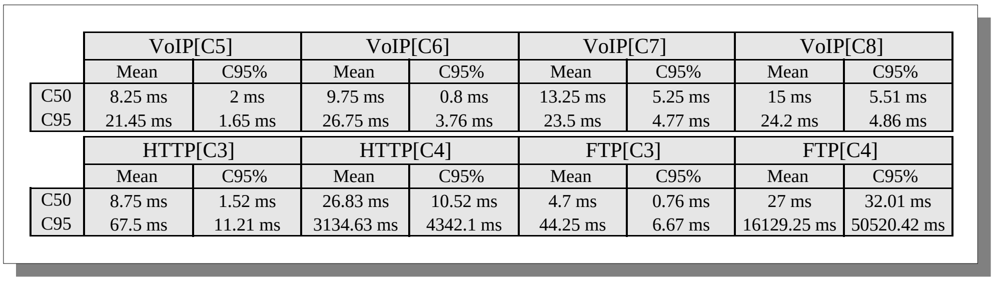
Table 6.13: RED statistics from router R6 CASE: EF with priority scheduling



	VoIP[C5]		VoIP[C6]		VoIP[C7]		VoIP[C8]	
	Mean	C95%	Mean	C95%	Mean	C95%	Mean	C95%
1	75120 bps	214 bps	75117 bps	213 bps	75028 bps	257 bps	75112 bps	194 bps
2	75158 bps	130 bps	75159 bps	124 bps	75113 bps	136 bps	75117 bps	125 bps
3	74963 bps	497 bps	74949 bps	515 bps	74978 bps	527 bps	74954 bps	506 bps
4	75090 bps	260 bps	75093 bps	253 bps	75094 bps	260 bps	75086 bps	266 bps

	HTTP[C3]		HTTP[C4]		FTP[C3]		FTP[C4]	
	Mean	C95%	Mean	C95%	Mean	C95%	Mean	C95%
1	79433 bps	9095 bps	35957 bps	7377 bps	324113 bps	23695 bps	113824 bps	21905 bps
2	89026 bps	6763 bps	21227 bps	5147 bps	323069 bps	19013 bps	0 bps	5 bps
3	59950 bps	13771 bps	93856 bps	14924 bps	142428 bps	37667 bps	7205 bps	7513 bps
4	78699 bps	10500 bps	89300 bps	9632 bps	109924 bps	25827 bps	380574 bps	22062 bps

Table 6.14: Per client throughput results, Case: EF with priority scheduling



	VoIP[C5]		VoIP[C6]		VoIP[C7]		VoIP[C8]	
	Mean	C95%	Mean	C95%	Mean	C95%	Mean	C95%
C50	8.25 ms	2 ms	9.75 ms	0.8 ms	13.25 ms	5.25 ms	15 ms	5.51 ms
C95	21.45 ms	1.65 ms	26.75 ms	3.76 ms	23.5 ms	4.77 ms	24.2 ms	4.86 ms

	HTTP[C3]		HTTP[C4]		FTP[C3]		FTP[C4]	
	Mean	C95%	Mean	C95%	Mean	C95%	Mean	C95%
C50	8.75 ms	1.52 ms	26.83 ms	10.52 ms	4.7 ms	0.76 ms	27 ms	32.01 ms
C95	67.5 ms	11.21 ms	3134.63 ms	4342.1 ms	44.25 ms	6.67 ms	16129.25 ms	50520.42 ms

Table 6.15: Per client jitter results, Case: EF with priority scheduling

6.4.3.3 Conclusions from EF simulations

Expedited forwarding (EF) provided the quality differentiation what was expected from it. Both priority and WRR scheduling provided the isolation which was expected. The difference between the WRR and priority scheduling was in the controllability of resource usage. WRR provides means to offer some predetermined amount of resources to each class, irrespective of its requirements. This helps in providing fixed quality levels during congestion.

6.4.4 AF simulations

This section presents simulation results of a network with the AF PHB. Simulated network topology and connection pairs are presented in Figure 6.17. AF allows both time and space priority to be used in quality differentiation. Our simulations concentrated on the space priority. Space priority is controlled with parallel RED algorithms, one for each space priority class. In our simulations, AF has two classes: *In* and *Out*. The class of a packet is decided based on the ratio of the packet generation rate to the subscribed (target) rate.

Questions which we want to answer at this point are:

1. *What is the differentiation factor which we can achieve by using AF PHB and different target rates.*
2. *What is the effect of different RTTs to the achievement of target rate*

Therefore, simulations were run with two different configuration sets:

1. **Similar RTT times** - each client/server pair has similar RTT on the connection path.
2. **Dissimilar RTT times** - the link between routers R7 and R8 has 10 times higher (150 ms) delay than the link between routers R7 and R9 (15 ms).

Both cases were reproduced six times to provide statistical confidence of the results. Iterations were independent processes with different seeds.

6.4.4.1 AF with similar RTT

Table 6.16 provides general statistics from the buffer of the bottleneck link between routers R6 and R7. Simulation time was set to be 3600 s but in some occasions, like in iterations 2, 3 and 4, simulation was aborted due to an anomaly in the TCP.

Throughput results in Table 6.17 show that with the AF FTP clients are able to open their communication processes. The effect of the HTTP and VoIP clients is diminished and the overall performance is good. However, it should be noted that rate variation between VoIP clients is even higher than with BE. This is due

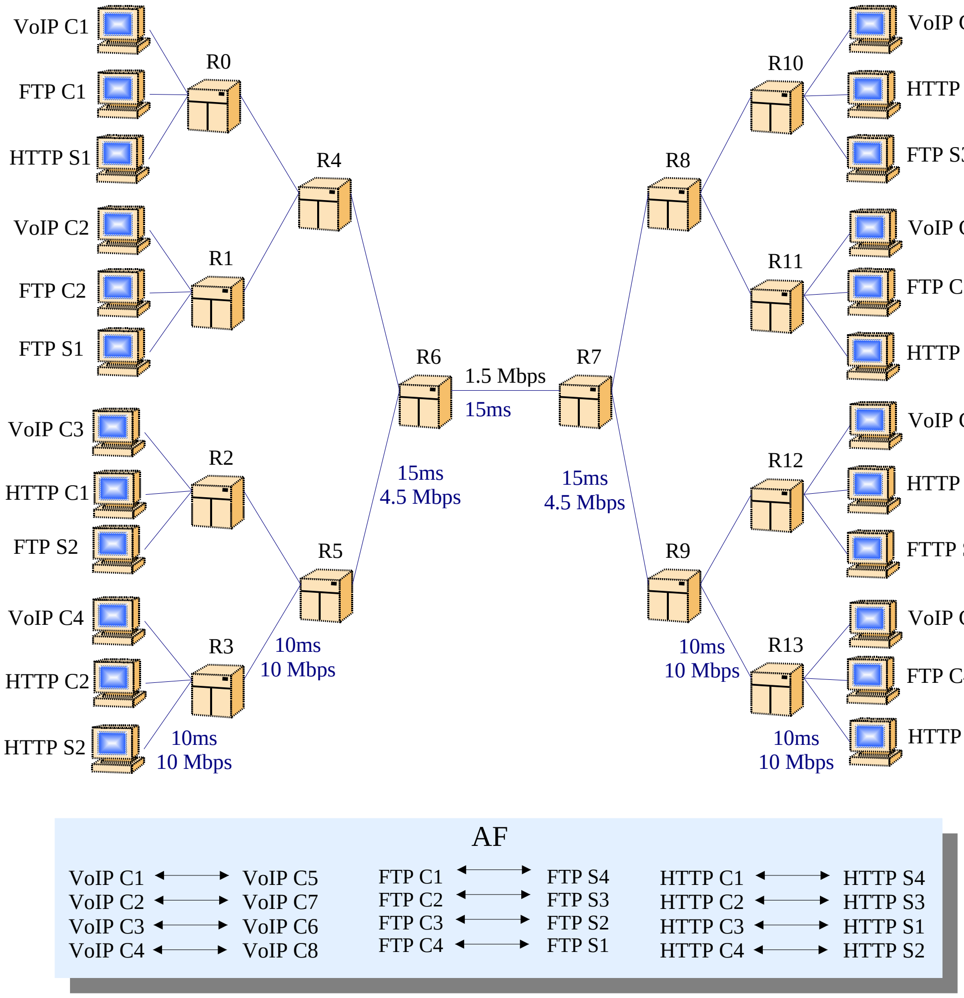
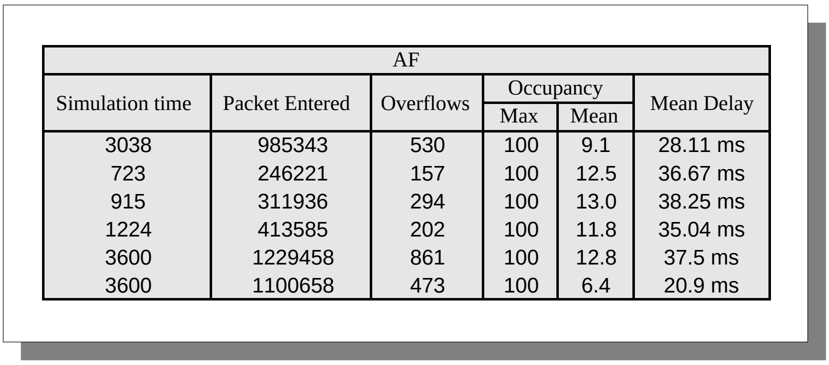
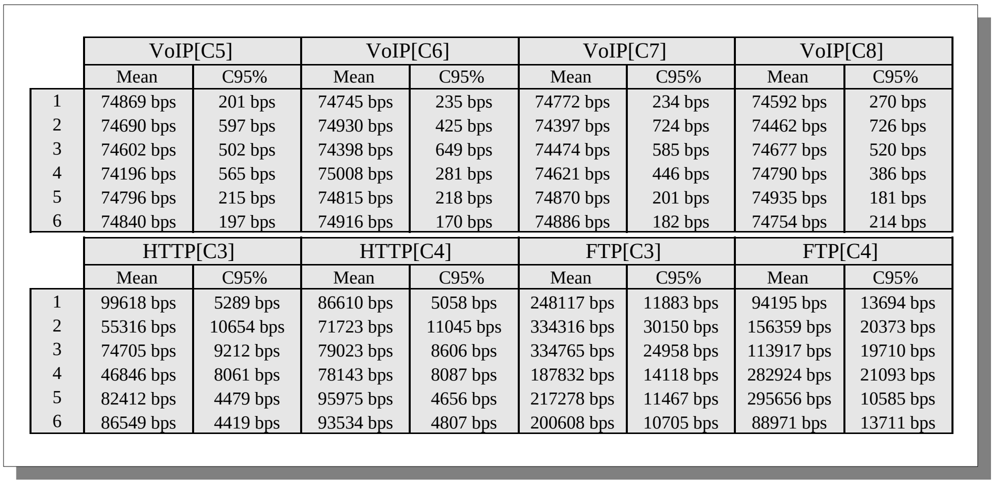


Figure 6.17: Network topology and connection pairs in the AF simulation



AF					
Simulation time	Packet Entered	Overflows	Occupancy		Mean Delay
			Max	Mean	
3038	985343	530	100	9.1	28.11 ms
723	246221	157	100	12.5	36.67 ms
915	311936	294	100	13.0	38.25 ms
1224	413585	202	100	11.8	35.04 ms
3600	1229458	861	100	12.8	37.5 ms
3600	1100658	473	100	6.4	20.9 ms

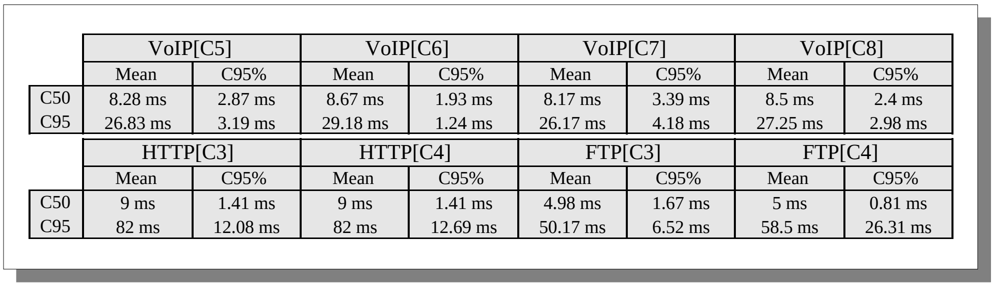
Table 6.16: General buffer statistics from the router R6, CASE: AF with similar RTTs



	VoIP[C5]		VoIP[C6]		VoIP[C7]		VoIP[C8]	
	Mean	C95%	Mean	C95%	Mean	C95%	Mean	C95%
1	74869 bps	201 bps	74745 bps	235 bps	74772 bps	234 bps	74592 bps	270 bps
2	74690 bps	597 bps	74930 bps	425 bps	74397 bps	724 bps	74462 bps	726 bps
3	74602 bps	502 bps	74398 bps	649 bps	74474 bps	585 bps	74677 bps	520 bps
4	74196 bps	565 bps	75008 bps	281 bps	74621 bps	446 bps	74790 bps	386 bps
5	74796 bps	215 bps	74815 bps	218 bps	74870 bps	201 bps	74935 bps	181 bps
6	74840 bps	197 bps	74916 bps	170 bps	74886 bps	182 bps	74754 bps	214 bps

	HTTP[C3]		HTTP[C4]		FTP[C3]		FTP[C4]	
	Mean	C95%	Mean	C95%	Mean	C95%	Mean	C95%
1	99618 bps	5289 bps	86610 bps	5058 bps	248117 bps	11883 bps	94195 bps	13694 bps
2	55316 bps	10654 bps	71723 bps	11045 bps	334316 bps	30150 bps	156359 bps	20373 bps
3	74705 bps	9212 bps	79023 bps	8606 bps	334765 bps	24958 bps	113917 bps	19710 bps
4	46846 bps	8061 bps	78143 bps	8087 bps	187832 bps	14118 bps	282924 bps	21093 bps
5	82412 bps	4479 bps	95975 bps	4656 bps	217278 bps	11467 bps	295656 bps	10585 bps
6	86549 bps	4419 bps	93534 bps	4807 bps	200608 bps	10705 bps	88971 bps	13711 bps

Table 6.17: Per client throughput results, CASE: AF with similar RTTs



	VoIP[C5]		VoIP[C6]		VoIP[C7]		VoIP[C8]	
	Mean	C95%	Mean	C95%	Mean	C95%	Mean	C95%
C50	8.28 ms	2.87 ms	8.67 ms	1.93 ms	8.17 ms	3.39 ms	8.5 ms	2.4 ms
C95	26.83 ms	3.19 ms	29.18 ms	1.24 ms	26.17 ms	4.18 ms	27.25 ms	2.98 ms

	HTTP[C3]		HTTP[C4]		FTP[C3]		FTP[C4]	
	Mean	C95%	Mean	C95%	Mean	C95%	Mean	C95%
C50	9 ms	1.41 ms	9 ms	1.41 ms	4.98 ms	1.67 ms	5 ms	0.81 ms
C95	82 ms	12.08 ms	82 ms	12.69 ms	50.17 ms	6.52 ms	58.5 ms	26.31 ms

Table 6.18: Jitter results, CASE: AF with similar RTTs

to the higher load in the class, which causes delays in scheduling of packets. This generates bursts to the communication.

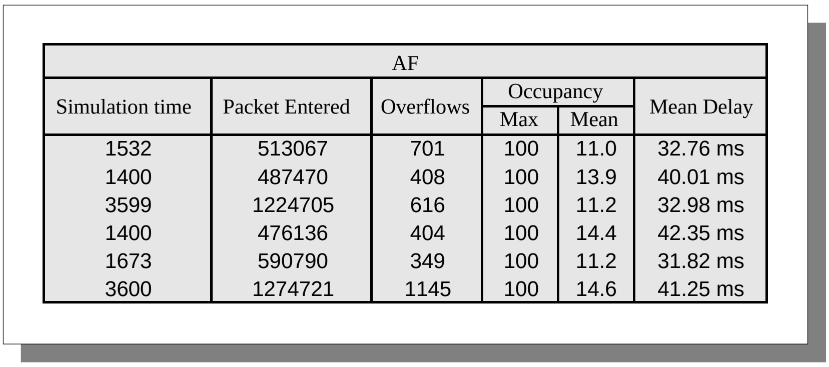
Table 6.18 presents jitter results of the simulation. Again median jitter between clients is of the same order and the 95% percentile jitter has 2-3 fold difference between the UDP and the TCP clients. This is, however, relatively good result compared to the BE case where the difference was 20 fold. Jitter of the UDP clients is still of the same magnitude as it was in the EF simulations.

6.4.4.2 AF with dissimilar RTT

Table 6.19 provides general statistics from the dissimilar RTT simulations. Most of the simulations were aborted due to TCP anomalies.

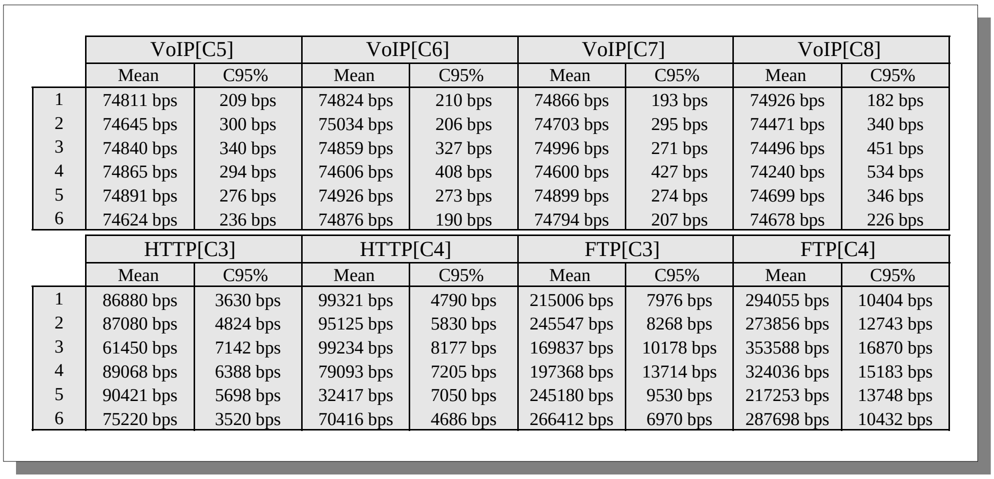
Throughput results in Table 6.20 show the effect of the RTT on the throughput of TCP connections. Clients HTTP[C3] and FTP[C3] have 2.6 times higher RTT than clients HTTP[C4] and FTP[C4] (based on the link delay metrics). In throughput, difference is not so big only 1.5 times. This is expected since adding queuing delays to the fixed parts of RTT lowers the difference in delay to a half of what it would be in the absolute case.

Table 6.21 presents jitter results of the simulations. Results show similar behavior as in the case with similar RTTs; median jitter is about the same but 95% percentile values have same difference between VoIP and other sources.



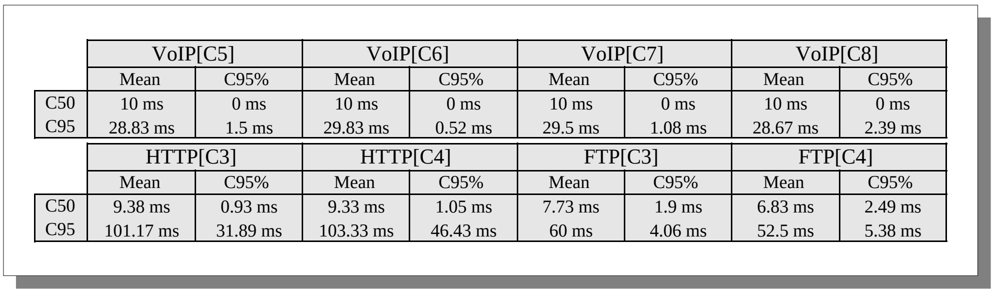
AF					
Simulation time	Packet Entered	Overflows	Occupancy		Mean Delay
			Max	Mean	
1532	513067	701	100	11.0	32.76 ms
1400	487470	408	100	13.9	40.01 ms
3599	1224705	616	100	11.2	32.98 ms
1400	476136	404	100	14.4	42.35 ms
1673	590790	349	100	11.2	31.82 ms
3600	1274721	1145	100	14.6	41.25 ms

Table 6.19: General buffer statistics from the router R6, CASE: AF with dissimilar RTTs



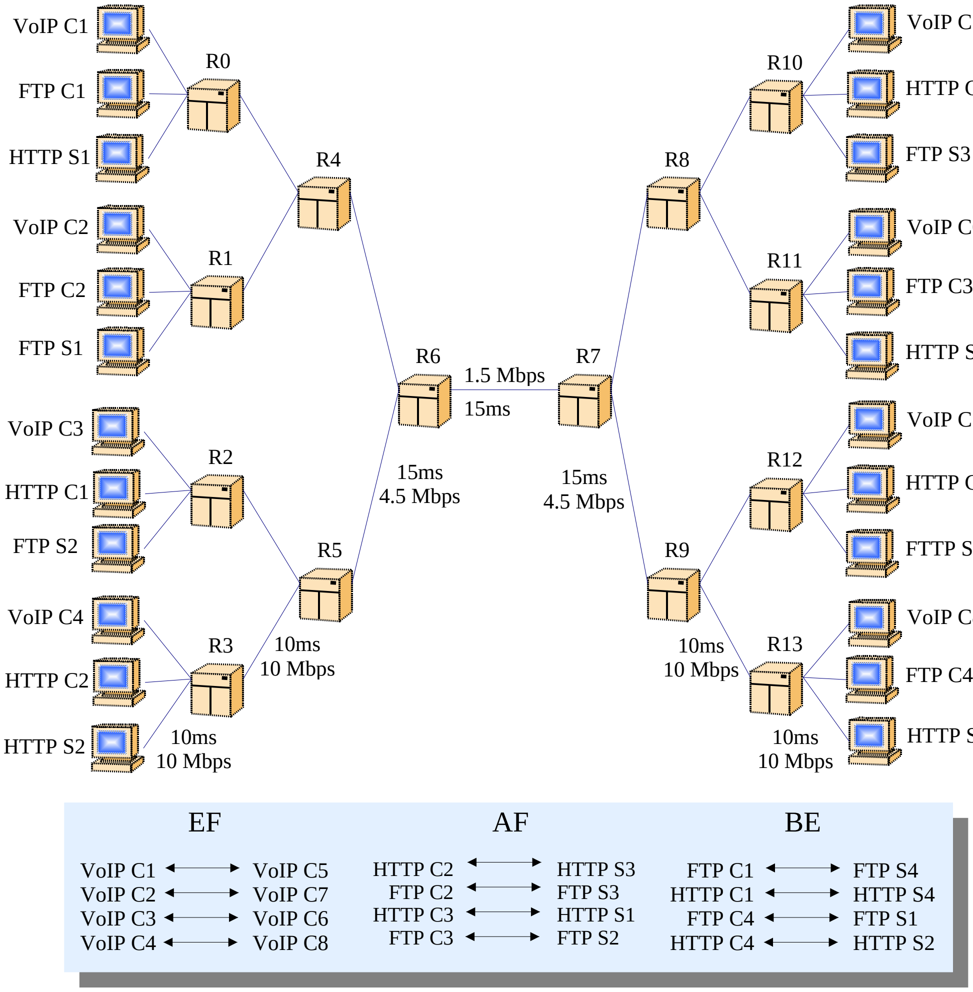
	VoIP[C5]		VoIP[C6]		VoIP[C7]		VoIP[C8]	
	Mean	C95%	Mean	C95%	Mean	C95%	Mean	C95%
1	74811 bps	209 bps	74824 bps	210 bps	74866 bps	193 bps	74926 bps	182 bps
2	74645 bps	300 bps	75034 bps	206 bps	74703 bps	295 bps	74471 bps	340 bps
3	74840 bps	340 bps	74859 bps	327 bps	74996 bps	271 bps	74496 bps	451 bps
4	74865 bps	294 bps	74606 bps	408 bps	74600 bps	427 bps	74240 bps	534 bps
5	74891 bps	276 bps	74926 bps	273 bps	74899 bps	274 bps	74699 bps	346 bps
6	74624 bps	236 bps	74876 bps	190 bps	74794 bps	207 bps	74678 bps	226 bps
	HTTP[C3]		HTTP[C4]		FTP[C3]		FTP[C4]	
	Mean	C95%	Mean	C95%	Mean	C95%	Mean	C95%
1	86880 bps	3630 bps	99321 bps	4790 bps	215006 bps	7976 bps	294055 bps	10404 bps
2	87080 bps	4824 bps	95125 bps	5830 bps	245547 bps	8268 bps	273856 bps	12743 bps
3	61450 bps	7142 bps	99234 bps	8177 bps	169837 bps	10178 bps	353588 bps	16870 bps
4	89068 bps	6388 bps	79093 bps	7205 bps	197368 bps	13714 bps	324036 bps	15183 bps
5	90421 bps	5698 bps	32417 bps	7050 bps	245180 bps	9530 bps	217253 bps	13748 bps
6	75220 bps	3520 bps	70416 bps	4686 bps	266412 bps	6970 bps	287698 bps	10432 bps

Table 6.20: Per client goodput results, CASE: AF with dissimilar RTTs



	VoIP[C5]		VoIP[C6]		VoIP[C7]		VoIP[C8]	
	Mean	C95%	Mean	C95%	Mean	C95%	Mean	C95%
C50	10 ms	0 ms	10 ms	0 ms	10 ms	0 ms	10 ms	0 ms
C95	28.83 ms	1.5 ms	29.83 ms	0.52 ms	29.5 ms	1.08 ms	28.67 ms	2.39 ms
	HTTP[C3]		HTTP[C4]		FTP[C3]		FTP[C4]	
	Mean	C95%	Mean	C95%	Mean	C95%	Mean	C95%
C50	9.38 ms	0.93 ms	9.33 ms	1.05 ms	7.73 ms	1.9 ms	6.83 ms	2.49 ms
C95	101.17 ms	31.89 ms	103.33 ms	46.43 ms	60 ms	4.06 ms	52.5 ms	5.38 ms

Table 6.21: Jitter results, CASE: AF with dissimilar RTTs



EF		AF		BE	
VoIP C1	VoIP C5	HTTP C2	HTTP S3	FTP C1	FTP S4
VoIP C2	VoIP C7	FTP C2	FTP S3	HTTP C1	HTTP S4
VoIP C3	VoIP C6	HTTP C3	HTTP S1	FTP C4	FTP S1
VoIP C4	VoIP C8	FTP C3	FTP S2	HTTP C4	HTTP S2

Figure 6.18: Network topology and connection pairs in the EF+AF+BE simulation

6.4.4.3 Conclusions from AF simulations

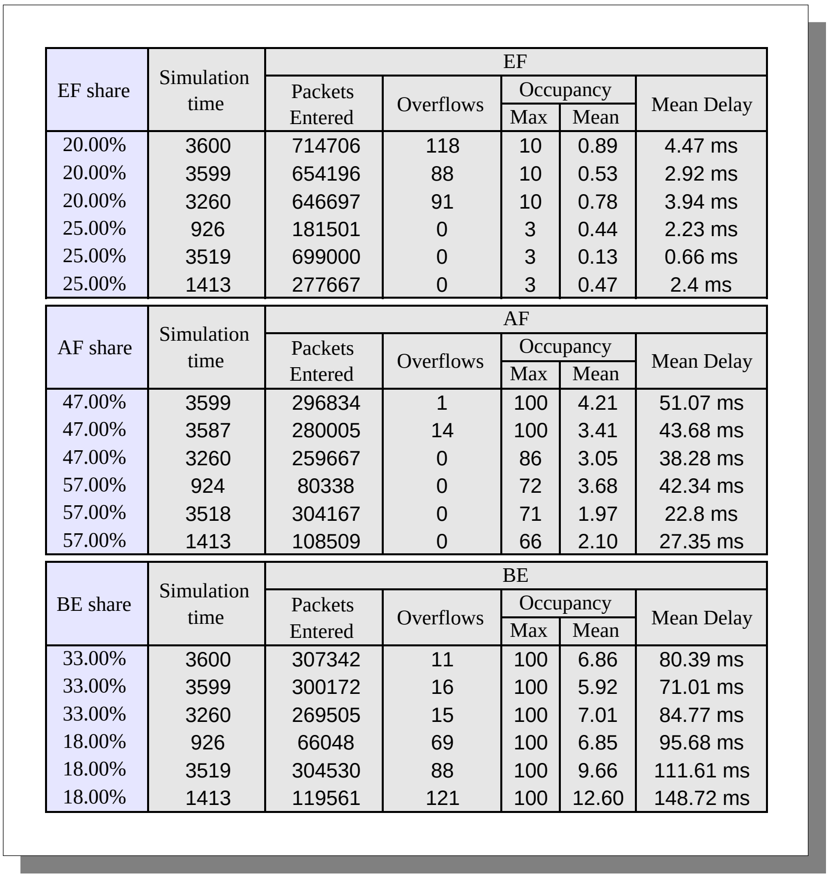
AF behaved as was expected. Connections on the high delay path were not able to retrieve as much resources as connections on the low delay path. However, controllability and jitter statistics of connections show that the service could be used to offer some level of performance guarantees - at least it performs much better than the conventional BE.

6.4.5 EF+AF+BE simulations

This section presents simulation results of a case where EF PHB is applied to the VoIP connections, AF PHB is applied to the connections HTTP[C3] and FTP[C3], and BE PHB is applied to the connections HTTP[C3] and FTP[C3]. Network topology and connection pairs are presented in Figure 6.18.

Simulations were run with two different configuration options:

1. **Priority scheduling**



EF share	Simulation time	EF				
		Packets Entered	Overflows	Occupancy		Mean Delay
				Max	Mean	
20.00%	3600	714706	118	10	0.89	4.47 ms
20.00%	3599	654196	88	10	0.53	2.92 ms
20.00%	3260	646697	91	10	0.78	3.94 ms
25.00%	926	181501	0	3	0.44	2.23 ms
25.00%	3519	699000	0	3	0.13	0.66 ms
25.00%	1413	277667	0	3	0.47	2.4 ms

AF share	Simulation time	AF				
		Packets Entered	Overflows	Occupancy		Mean Delay
				Max	Mean	
47.00%	3599	296834	1	100	4.21	51.07 ms
47.00%	3587	280005	14	100	3.41	43.68 ms
47.00%	3260	259667	0	86	3.05	38.28 ms
57.00%	924	80338	0	72	3.68	42.34 ms
57.00%	3518	304167	0	71	1.97	22.8 ms
57.00%	1413	108509	0	66	2.10	27.35 ms

BE share	Simulation time	BE				
		Packets Entered	Overflows	Occupancy		Mean Delay
				Max	Mean	
33.00%	3600	307342	11	100	6.86	80.39 ms
33.00%	3599	300172	16	100	5.92	71.01 ms
33.00%	3260	269505	15	100	7.01	84.77 ms
18.00%	926	66048	69	100	6.85	95.68 ms
18.00%	3519	304530	88	100	9.66	111.61 ms
18.00%	1413	119561	121	100	12.60	148.72 ms

Table 6.22: General statistics of the router R6, CASE: EF+AF+BE and WRR scheduling

2. **Weighted Round Robin scheduling**, two sets of simulations were run with following resource provisioning rules:

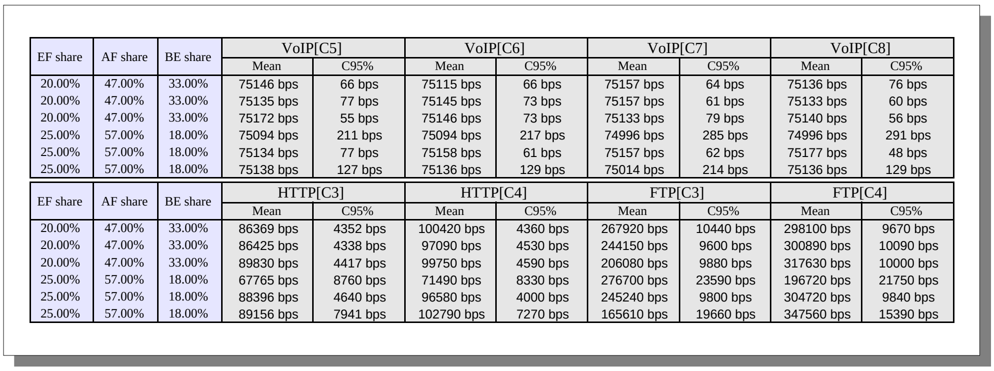
   (a) EF class 20%, AF class 47% and BE class 33%

   (b) EF class 25%, AF class 57% and BE class 18%

6.4.5.1 EF+AF+BE with WRR scheduling

General statistics in Table 6.22 show that packets are lost in every class in the first set of simulations. This is due to provisioning, in particular, 20% provisioning for the EF class causes continuous contention. Small buffer in the EF class is not able to buffer the contending packets. With 25% provisioning, EF class is able to pass all the traffic without excessive buffering and delays. Number of overflows in the BE class is, however, high in the second set of provisioning. This is due to the small amount of resources which are dedicated to this class. Changes in the AF class are not notable, which shows flexibility that the AF PHB has with respect to provisioning.

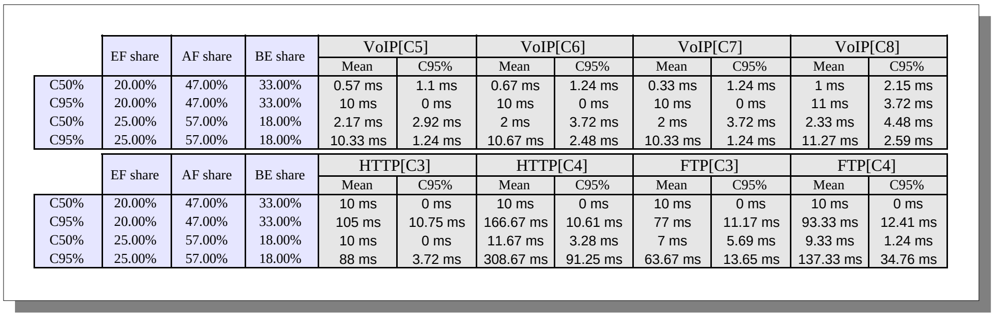
Throughput results in Table 6.23 show an interesting thing. The throughput of VoIP clients is lower in the second set of provisioning (while their relative share of scheduler capacity is increased from 20% to 25%). This may be due to even higher increase of provisioned resources in the AF class. Jitter results in Table 6.24 show



EF share	AF share	BE share	VoIP[C5]		VoIP[C6]		VoIP[C7]		VoIP[C8]	
			Mean	C95%	Mean	C95%	Mean	C95%	Mean	C95%
20.00%	47.00%	33.00%	75146 bps	66 bps	75115 bps	66 bps	75157 bps	64 bps	75136 bps	76 bps
20.00%	47.00%	33.00%	75135 bps	77 bps	75145 bps	73 bps	75157 bps	61 bps	75133 bps	60 bps
20.00%	47.00%	33.00%	75172 bps	55 bps	75146 bps	73 bps	75133 bps	79 bps	75140 bps	56 bps
25.00%	57.00%	18.00%	75094 bps	211 bps	75094 bps	217 bps	74996 bps	285 bps	74996 bps	291 bps
25.00%	57.00%	18.00%	75134 bps	77 bps	75158 bps	61 bps	75157 bps	62 bps	75177 bps	48 bps
25.00%	57.00%	18.00%	75138 bps	127 bps	75136 bps	129 bps	75014 bps	214 bps	75136 bps	129 bps

EF share	AF share	BE share	HTTP[C3]		HTTP[C4]		FTP[C3]		FTP[C4]	
			Mean	C95%	Mean	C95%	Mean	C95%	Mean	C95%
20.00%	47.00%	33.00%	86369 bps	4352 bps	100420 bps	4360 bps	267920 bps	10440 bps	298100 bps	9670 bps
20.00%	47.00%	33.00%	86425 bps	4338 bps	97090 bps	4530 bps	244150 bps	9600 bps	300890 bps	10090 bps
20.00%	47.00%	33.00%	89830 bps	4417 bps	99750 bps	4590 bps	206080 bps	9880 bps	317630 bps	10000 bps
25.00%	57.00%	18.00%	67765 bps	8760 bps	71490 bps	8330 bps	276700 bps	23590 bps	196720 bps	21750 bps
25.00%	57.00%	18.00%	88396 bps	4640 bps	96580 bps	4000 bps	245240 bps	9800 bps	304720 bps	9840 bps
25.00%	57.00%	18.00%	89156 bps	7941 bps	102790 bps	7270 bps	165610 bps	19660 bps	347560 bps	15390 bps

Table 6.23: Per client goodput results, CASE: EF+AF+BE and WRR scheduling



	EF share	AF share	BE share	VoIP[C5]		VoIP[C6]		VoIP[C7]		VoIP[C8]	
				Mean	C95%	Mean	C95%	Mean	C95%	Mean	C95%
C50%	20.00%	47.00%	33.00%	0.57 ms	1.1 ms	0.67 ms	1.24 ms	0.33 ms	1.24 ms	1 ms	2.15 ms
C95%	20.00%	47.00%	33.00%	10 ms	0 ms	10 ms	0 ms	10 ms	0 ms	11 ms	3.72 ms
C50%	25.00%	57.00%	18.00%	2.17 ms	2.92 ms	2 ms	3.72 ms	2 ms	3.72 ms	2.33 ms	4.48 ms
C95%	25.00%	57.00%	18.00%	10.33 ms	1.24 ms	10.67 ms	2.48 ms	10.33 ms	1.24 ms	11.27 ms	2.59 ms

	EF share	AF share	BE share	HTTP[C3]		HTTP[C4]		FTP[C3]		FTP[C4]	
				Mean	C95%	Mean	C95%	Mean	C95%	Mean	C95%
C50%	20.00%	47.00%	33.00%	10 ms	0 ms	10 ms	0 ms	10 ms	0 ms	10 ms	0 ms
C95%	20.00%	47.00%	33.00%	105 ms	10.75 ms	166.67 ms	10.61 ms	77 ms	11.17 ms	93.33 ms	12.41 ms
C50%	25.00%	57.00%	18.00%	10 ms	0 ms	11.67 ms	3.28 ms	7 ms	5.69 ms	9.33 ms	1.24 ms
C95%	25.00%	57.00%	18.00%	88 ms	3.72 ms	308.67 ms	91.25 ms	63.67 ms	13.65 ms	137.33 ms	34.76 ms

Table 6.24: Per client jitter results, CASE: EF+AF+BE and WRR scheduling

also the same thing. The jitter of VoIP connections increases while their share of resources in the scheduler also increases. However, with an AF client there is clear reduction in the jitter. In this light, it seems that AF is the winner in this provisioning change. Throughput changes are not notable but jitter is only a half of what it is with the connections in the BE class.

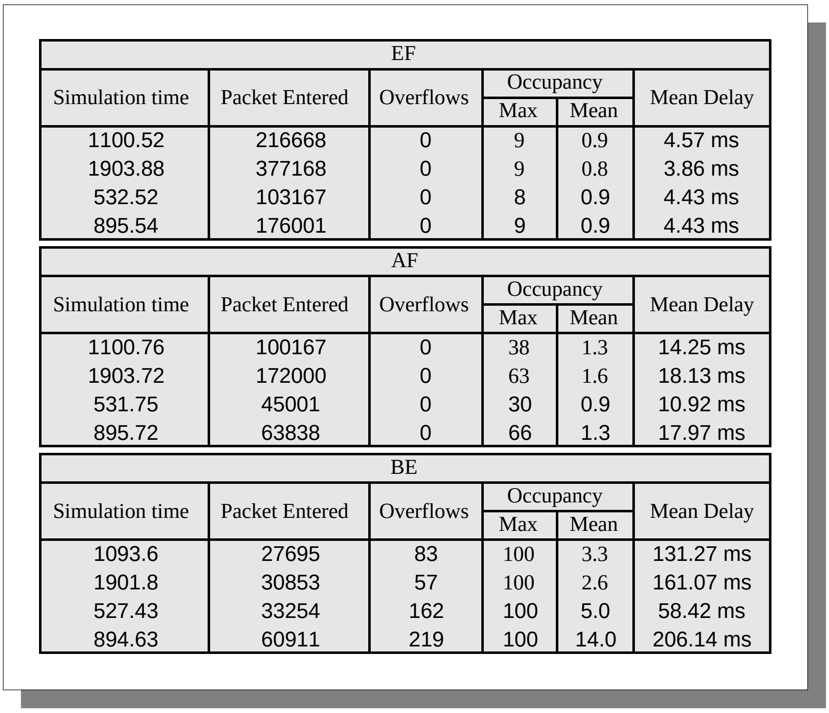
6.4.5.2 EF+AF+BE with Priority scheduling

General statistics in Table 6.25 show that there is not a single lost packet in the EF and the AF classes due to overflow. This is what should be expected when there is hard priority over the classes. Delay characteristics of the classes show clear differentiation. The EF class has a quarter of the delay of the AF class. In the same way the AF class has one tenth of the delay of the BE class.

Throughput results in Table 6.26 show that the VoIP clients get the resources they need. The AF class seems to have enough resources to fulfill demands of its clients. However, little or nothing is left for the BE class which effectively starves in many occasions. Jitter results in Table 6.27 show also the same thing. Jitter of the EF and AF connections are within the limits of reasonable operation but the BE class suffers from jitter of order of seconds.

6.4.5.3 Conclusions EF+AF+BE simulations

Service differentiation using the EF, AF and BE PHB seems to work as was expected. However, the WRR case showed that the provisioning of resources and

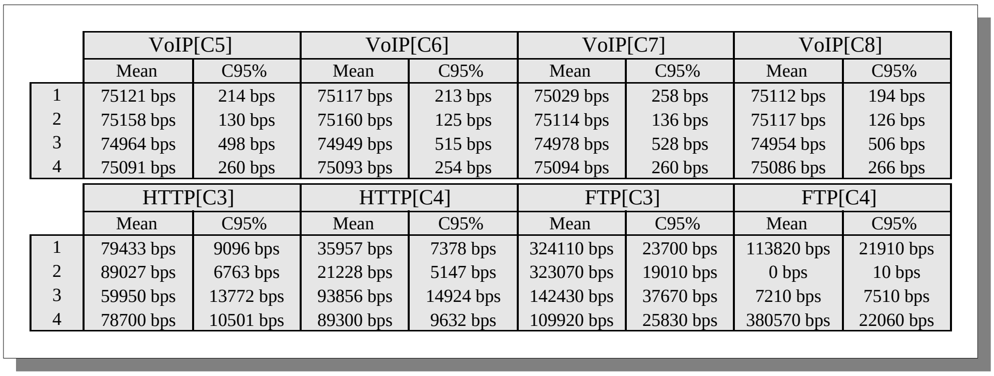


EF					
Simulation time	Packet Entered	Overflows	Occupancy		Mean Delay
			Max	Mean	
1100.52	216668	0	9	0.9	4.57 ms
1903.88	377168	0	9	0.8	3.86 ms
532.52	103167	0	8	0.9	4.43 ms
895.54	176001	0	9	0.9	4.43 ms

AF					
Simulation time	Packet Entered	Overflows	Occupancy		Mean Delay
			Max	Mean	
1100.76	100167	0	38	1.3	14.25 ms
1903.72	172000	0	63	1.6	18.13 ms
531.75	45001	0	30	0.9	10.92 ms
895.72	63838	0	66	1.3	17.97 ms

BE					
Simulation time	Packet Entered	Overflows	Occupancy		Mean Delay
			Max	Mean	
1093.6	27695	83	100	3.3	131.27 ms
1901.8	30853	57	100	2.6	161.07 ms
527.43	33254	162	100	5.0	58.42 ms
894.63	60911	219	100	14.0	206.14 ms

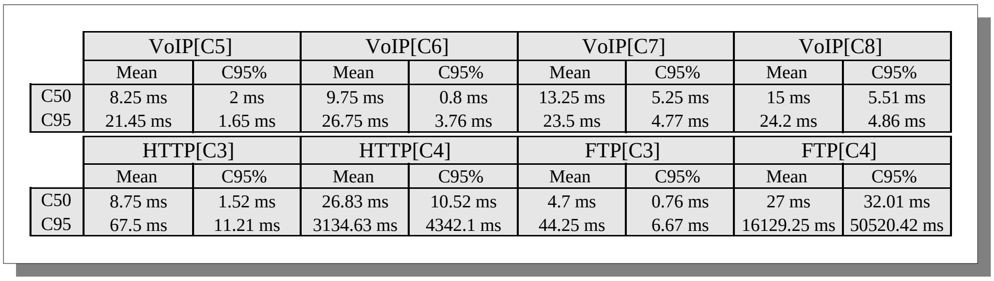
Table 6.25: General statistics of the router R6 CASE: EF+AF+BE with priority scheduling



	VoIP[C5]		VoIP[C6]		VoIP[C7]		VoIP[C8]	
	Mean	C95%	Mean	C95%	Mean	C95%	Mean	C95%
1	75121 bps	214 bps	75117 bps	213 bps	75029 bps	258 bps	75112 bps	194 bps
2	75158 bps	130 bps	75160 bps	125 bps	75114 bps	136 bps	75117 bps	126 bps
3	74964 bps	498 bps	74949 bps	515 bps	74978 bps	528 bps	74954 bps	506 bps
4	75091 bps	260 bps	75093 bps	254 bps	75094 bps	260 bps	75086 bps	266 bps

	HTTP[C3]		HTTP[C4]		FTP[C3]		FTP[C4]	
	Mean	C95%	Mean	C95%	Mean	C95%	Mean	C95%
1	79433 bps	9096 bps	35957 bps	7378 bps	324110 bps	23700 bps	113820 bps	21910 bps
2	89027 bps	6763 bps	21228 bps	5147 bps	323070 bps	19010 bps	0 bps	10 bps
3	59950 bps	13772 bps	93856 bps	14924 bps	142430 bps	37670 bps	7210 bps	7510 bps
4	78700 bps	10501 bps	89300 bps	9632 bps	109920 bps	25830 bps	380570 bps	22060 bps

Table 6.26: Per client throughput results, CASE: EF+AF+BE with priority scheduling



	VoIP[C5]		VoIP[C6]		VoIP[C7]		VoIP[C8]	
	Mean	C95%	Mean	C95%	Mean	C95%	Mean	C95%
C50	8.25 ms	2 ms	9.75 ms	0.8 ms	13.25 ms	5.25 ms	15 ms	5.51 ms
C95	21.45 ms	1.65 ms	26.75 ms	3.76 ms	23.5 ms	4.77 ms	24.2 ms	4.86 ms

	HTTP[C3]		HTTP[C4]		FTP[C3]		FTP[C4]	
	Mean	C95%	Mean	C95%	Mean	C95%	Mean	C95%
C50	8.75 ms	1.52 ms	26.83 ms	10.52 ms	4.7 ms	0.76 ms	27 ms	32.01 ms
C95	67.5 ms	11.21 ms	3134.63 ms	4342.1 ms	44.25 ms	6.67 ms	16129.25 ms	50520.42 ms

Table 6.27: Per client jitter results, CASE: EF+AF+BE with priority scheduling

the actual load of the class could bring up unpredicted results. Priority scheduling revealed its worst behavior - starvation of resources of the lowest classes. This was expected as there is little or no control over the operation of the AF class with regard to the sharing of the excess capacity. On many occasions AF connections were able to exploit more resources than BE connections, which were sending single packets spaced by the time-out intervals. These clients are not able to notice left over resources and open their transmission window.

6.5 Conclusions from the simulations

Differentiated Services provides a level of service differentiation which is entirely dependent on the service provider's success in predicting traffic loads of different classes. This is clearly visible from the results of the simulations. DiffServ offers multiple levels of best effort service, i.e. each class (PHB) is a best effort class of its own. This is due to the nature of the DiffServ - there is no intra-class isolation of traffic flows. These effects can be clearly seen by looking the throughput results of the best effort simulation (Table 6.3), EF+BE simulation (Table 6.11), AF simulation (Table 6.17) and EF+AF+BE simulation (Table 6.23). Throughputs of different applications (mostly the aggressive FTP) vary depending on the traffic mixture in a class and the number of classes. Best effort case clearly shows the problem which we have today in the Internet. Traffic streams, produced by different types of applications, are interfering with each other. This causes low overall utilization and low quality in the light of numerical analysis of received service. Applying the EF PHB clearly shows what we can achieve with it - nothing if we think residential users of the Internet, but a lot for corporate users. EF PHB clearly provides leased line emulation. Therefore, it is restricted to the service scenario which leased line emulation has - a point-to-point, long time scale, strictly policed and manually provisioned service. However, this is what many corporates are looking for; a VPN service over which data and telephone traffic could be transmitted. The rest of the traffic, which was still transmitted in the best effort service class, had difficulties, which were already observable in the first simulation. This is what to expect when only a fraction of the traffic is transmitted in a special class implemented by the EF PHB. AF PHB simulation seems more promising for a common residential user. The AF provides a service which has a bit of proportional sharing mechanisms implemented. In our simulations, there were only two priority levels, but even this small proportionality made it possible to operate all of the applications in the same network with reasonable quality. However, implementation of the parallel AF class for real-time traffic would have made it possible to offer time priority handling along with space priorities. This is something, which is promising for mass-market users. Last simulations with the EF, AF and BE PHBs showed again the problem which we are about to face when resources are partitioned into small fragments. It is not easy to provision the network in a way that the delivered service in each class is what one should expect. Radical changes, which were made with two different sets of simulations, caused only a small change in the delivered quality.

In general, one can say that the Differentiated Services is able to offer tools for flexible network engineering. Table 6.22 shows clearly that quality differentiation based on time priorities (Mean Delay) and space priorities (Mean Occupancy) can be achieved if a little effort is made. However, the delivered service to the customers depends entirely on the service models which are build upon the tools of the Differentiated Services.

Chapter 7

Conclusions

Internet has evolved from the experimental research network to an information pool which is used frequently by millions of users. This has set an enormous pressure to the development of the Internet environment. Internet Engineering Task Force has been active in bringing up new possibilities for Internet environment. This thesis has presented these ideas in a general level to provide a broad overview where we are today. The most recent hype, Differentiated Services, was analyzed thoroughly. Its spinal structure, general behavioral rules, and vital organs, implementations of the behavioral rules, were dissected into a pieces to have a look what makes the DiffServ work.

Differentiated Services is a broad formulation of operational aspects of the service environment. It does not exactly formulate the way a network device should work nor does it provide clear picture of services which could be delivered by it. Standardized features along some suggested amendments were presented in Chapter 4. Loose formulations in the DiffServ give a lot of freedom, and also responsibility, to the network provider. Simulations, in Section 6.4, were used to analyze the space which the service provider has in service provisioning. As expressed in the summary of Section 6.4, the level of quality separation, which DiffServ provides, is totally dependent on the accuracy of load level prediction in different classes. This is not promising if the system is based on the static provisioning and human based analyses of the traffic. Therefore, automated management systems for user and traffic differentiation are needed. Mobility of the user population affects the management platform with strong need to apply some centralized directory service to distribute SLA based quality information. This with a good service model and reasonable provisioning, could bring DiffServ to the level where it is able to do quality separation, which is satisfactory for the next two to five years. This is the time frame after which new packet based mobile networks will take off. This will push the mobility requirements to the level which may very well be out of the limits of DiffServ structure.

General questions which we need to answer before DiffServ based Internet services become successful are therefore mostly on management level. The first is the question of service models - what kind of network usage will be the best option for the majority of users and how they should be charged and accounted for. The second is the issue of service provisioning, how should services be provisioned in the network - should there be separate real-time and non-real-time classes, how about separation of TCP and UDP flows. There is really no consensus about this issue. It seems that people have strong opinions, but very little facts to back up their opinions. The third question is how to make the system dynamic, in order to allow users to change their traffic commitments and also to allow them to be mobile in the network. This relates to the requirements of packet based mobile networks but also to the changing nature of work - people do not sit behind the same desk every day. Simulation results presented in Section 6.4 are not able to give answers to these questions, as they relate to the issues in forwarding path quality and to the management of services. Therefore, a task for future work is to associate management models to the forwarding path simulations. This should give more insight to the problems discussed above.

Bibliography

[Alm92] P. Almquist. Type of service in the internet protocol suite. Technical Report RFC 1349, IETF, July 1992.

[Bak95] Fred Baker. Reguirements for ip version 4 routers. Technical Report RFC 1812, IETF, June 1995.

[BBB+99] Yoram Bernet, James Binder, Steven Blake, Mark Carlso, Srinivasan Keshavn, Elwyn Davies, Borje Ohlman, Dinesh Verma, Zheng Wang, and Walter Weiss. A framework for differentiated services. Technical Report draft-ietf-diffserv-framework-02.txt, IETF, February 1999.

[BBCW94] Roger Bohn, Hans-Werne Braun, Kimberly C. Claffy, and Stephen Wolff. Mitigating the coming internet crunch: Multiple service levels via precedence. Technical report, San Diego Supercomputer Center, and NSF, March 1994.

[BCC+98] 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 in the internet. Technical Report RFC 2309, IETF, April 1998.

[BO97] L. Berger and T. O'Malley. Rsvp extensions for ipsec data flows. Technical Report RFC 2207, IETF, September 1997.

[BV98] Meera Balakrishnan and Ramathan Venkateswaran. Qos and differentiated services in multiservice network environment. *Bell Labs Technical Journal*, pages 222–238, October-December 1998.

[BZ97] Jon C.R. Bennett and H. Zhang. Hierarchical packet fair queueing algorithms. *IEEE/ACM Transactions on Networking*, 5(5):675–689, October 1997.

[BZB+97] R Braden, Lixia Zhang, S Berson, S Herzog, and S Jamin. Resource reservation protocol (rsvp) - verions 1 functional specification. Technical Report RFC 2205, IETF, September 1997.

[CBP95] Kimberly Claffy, Hans-Werner Braun, and George Polyzos. A parameterizable methodology for internet traffic flow profiling. *IEEE Journal on Selected Areas in Communications*, 13(8):1481–1494, October 1995.

[CF98] David Clark and Wenjia Fang. Explicit allocation of best-effort packet delivery service. *IEEE/ACM Transactions on Networking*, 6(4):362–373, August 1998.

[Cla88] David Clark. The design philosophy of the darpa internet protocols. In *Proceedings of ACM SIGCOMM'88*, pages 16–19. ACM, August 1988.

[CLG99] Hungkei Chown and Alberot Leon-Garcia. A feedback control extension to differentiated services. Technical Report draft-chown-diffserv-fbctrl-00.ps, IETF, March 1999.

[CM97] Kimberly Claffy and Tracie Monk. What's next for internet data analysis? status and challenges facing the community. *Proceedings of the IEEE*, 85(10):1563–1571, October 1997.

[CSEZ93] Ron Cocchi, Scott Shenker, Deborah Estrin, and Lixia Zhang. pricing in computer networks: Motivation, formulation, and example. *IEEE/ACM Transactions on Networking*, 1(6):614–627, December 1993.

[CSZ92] David D. Clark, Scott Shenker, and Lixia Zhang. Supporting real-time applications in an integrated services packet network: Architecture and mechanism. In *Proceedings of SIGCOMM'92*. ACM, August 1992.

[CV96] Girish P. Chandranmenon and George Varghese. Trading packet headers for packet processing. *IEEE/ACM Transactions on Networking*, 4(2):141–151, 1996.

[CW97] D. Clark and J. Wroclawski. An approach to service allocation in the internet. Technical Report draft-clark-diff-svc-alloc-00.txt, IETF, July 1997.

[DKS89] A Demers, S Keshav, and Scott Shenker. Analysis and simulation of a fair queueing algorithm. In *Proceedings of ACM SIGCOMM'89*, pages 1–12. ACM, September 1989.

[Dov98] Constantinos Dovrolis. Class-based service differentiation. Technical Report draft-dovrolis-cbsd-00.txt, IETF, June 1998.

[Fan98] Wenjia Fang. The "expected capacity"framework: Simulation results. Technical report, Princeton University, January 1998.

[FJ93] Sally Floyd and Van Jacobson. Random early detection gateways for congestion avoidance. *IEEE/ACM Transactions on Networking*, 1(4):397–413, August 1993.

[FJ95] Sally Floyd and Van Jacobson. Link-sharing and resource management for packet networks. *IEEE/ACM Transactions on Networking*, 3(4):365–386, August 1995.

[Flo95] Sally Floyd. Notes on cbq and guaranteed service. July 1995.

[Flo96] Sally Floyd. Comments on measurement-based admissions control for controlled-load services. *Computer Communications Review*, July 1996.

[Fra98] Brichet Francois. Comparision of measurement-based admission control methods using bufferless multiplexing. May 1998.

[FS98] Sally Floyd and Michael Francis Speer. Experimental results for class-based queueing. November 1998.

[GB99] Manuel Günter and Torsten Braun. Evaluation of bandwidth broker signaling. In *Proceedings of ICNP'99*, pages 145–152. IEEE, 1999.

[Gro99] Dan Grossman. New terminology for diffserv. Technical Report draft-ietf-diffserv-new-terms-02.txt, IETF, November 1999.

[GVC97] Pawan Goyal, Harrick M. Vin, and Haichen Cheng. Start-time fair queueing: a scheduling algorithm for integrated services packet switching networks. *IEEE/ACM Transactions on Networking*, 5(5):690–704, October 1997.

[Hah91] Ellen Hahne. Round robin scheduling for fair flow control in data communication networks. *IEEE Journal on Selected Areas in Communications*, 9(7):1024–1039, September 1991.

[HBWW99] Juha Heinanen, Fred Baker, Walter Weiss, and John Wroclawski. Assured forwarding phb group. Technical Report RFC 2597, IETF, June 1999.

[HS88] John Henshall and Sandy Shaw. *OSI Explained.* Ellis Horwood, 1988.

[Hus00] Geoff Huston. *Internet Performance Survival Guide.* Wiley networking Council Series. John Wiley Sons, Inc., 2000.

[ILK98] Mika Ilvesmäki, Marko Luoma, and Raimo Kantola. Flow classification schemes in traffic-based multilayer ip switching - comparison between conventional and neural approach. *Computer Communications*, 21(13):1184–1194, 1998.

[IN98] J. Ibanez and K. Nichols. Preliminary simulation evaluation of an assured service. Technical Report draft-ibanez-diffserv-assured-eval-00.txt, IETF, August 1998.

[Jam96] Sugih Jamin. *A Measurement-based Admission Control Algorithm for Integrated Services Packet.* PhD thesis, University of Southern California, 1996.

[JDSZ96] Sugih Jamin, Peter B. Danzig, Scott J. Schenker, and Lixia Zhan. A measurement-based admission control algorithm for integrated services packet networks. *IEEE/ACM Transactions on Networking*, December 1996.

[JNP99] Van Jacobson, Kathleen Nichols, and Kedarnath Poduri. An expedited forwarding phb. Technical Report RFC 2598, IETF, June 1999.

[JR86] Raj Jain and Shawn Routhier. Packet trains — measurements and a new model for computer network traffic. *IEEE Journal on Selected Areas in Communications*, 4(6):986–995, September 1986.

[JSD97] Sugih Jamin, Scott J. Schenker, and Peter B. Danzig. Comparison of measurement-based admission control algorithms for controlled-load service. In *Proceedings of INFOCOM'97*. IEEE, April 1997.

[Kes91] Srinivasan Keshav. *Congestion Control in Computer Networks*. PhD thesis, University of California, Berkeley, August 19991.

[KH93] E. Krol and E. Hoffman. Fyi on "what is the internet?". Technical Report RFC 1462, IETF, May 1993.

[KkA97] Pär Karlsson and Åke Arvidsson. The characteristics of www traffic and the relevance to atm. Technical Report COST257 TD(97)21, University of Karlskrona/Ronneby, 1997.

[KR86] Kalevi Kilkki and Kauko Rahko. Teleliikenteen mallintaminen. *Sähkö*, (59):48–50, 1986.

[KR99] Kalevi Kilkki and Jussi Ruutu. Interoperability phb group. Technical Report draft-kilkki-diffserv-interoperability-00.txt, IETF, October 1999.

[KS99a] Stefan Köhler and Uwe Schäfer. Performance comparison of different class-and-drop treatment of data and acknowledgements in diffserv ip networks. Technical Report 237, University of Würzburg, Institute of Computer Science, August 1999.

[KS99b] Edward W. Knightly and Ness B. Shroff. Admission control for statistical qos: Theory and practice. *IEEE Network*, 13(2):20–29, March/April 1999.

[Lee94] Daniel C. Lee. Effects of leaky bucket parameters on the average quwuwing delay: Worst case analysis. In *Proceedings of INFOCOMM'94*, volume 2, pages 482–489. IEEE, 1994.

[Lel89] Will E. Leland. Window-based congestion management in broadband atm networks: The performance of three access-control policies. In *Proceedings of GLOBECOMM'89*, pages 1794–1800. IEEE, November 1989.

[LIP99] Marko Luoma, Mika Ilvesmäki, and Markus Peuhkuri. Use of packet length and inter-arrival time for quality differentiation in differentiated services. Technical Report COST 263 TD 1999/4/, Helsinki University of Technology, October 1999.

[LPY98] Marko Luoma, Markus Peuhkuri, and Tomi Yletyinen. Quality of service for ip voice services - is it necessary? In Raif Onvural, Seyhan Civandlar, Paul Doolan, and James Luciani, editors, *Internet Routing and Quality of Service*, volume 3529 of *SPIE Proceedings series*, pages 242–253. SPIE, SPIE, November 1998.

[LRK98] Mika Loukola, Jussi Ruutu, and Kalevi Kilkki. Dynamic rt/nrt phb group. Technical Report draft-loukola-dynamic-00.txt, IETF, November 1998.

[LS99] S.K. Lee and J.S. Song. An integrated admission control based on measurements in atm networks. *Computer Communications*, 22:140–1446, 1999.

[Mah97] Bruce Mah. An empirical model of http network traffic. In *Proceedings of INFOCOM'97*, volume 2, pages 592–600. IEEE, April 1997.

[MMPV99] G. Mamais, M. Markaki, G. Politis, and I. S. Venieris. Efficient buffer management and scheduling in a combined intserv and diffserv architecture: a performance study. In *Proceedings of ICATM'99*, pages 236–242. IEEE, 1999.

[Nag85] John Nagle. On packet switches with infinite storage. Technical Report RFC 970, IETF, December 1985.

[Nie98] Tapani Nieminen. Www-traffic modeling. Technical report, Helsinki University of Technology, May 1998.

[PE96] Harry G. Perros and Khaled M. Elsayed. Call admission control schemes: A review. *IEEE Communications Magazine*, pages 82–91, November 1996.

[PG94] Abhay Parekh and Robert Gallager. A generalized processor sharing approach to flow control in integrated services networks: the multiple node case. *IEEE/ACM Transactions on Networking*, 2(2):137–150, April 1994.

[Pin95] Michael Pinedo. *Scheduling; Theory, Algorithms and Systems*. International series in Industrial and Systems Engineering. Prentice-Hall, 1995.

[Pos80] John Postel. User datagram protocol. Technical Report RFC 768, IETF, August 1980.

[Pos81a] John Postel. Internet protocol. Technical Report RFC 791, IETF, September 1981.

[Pos81b] John Postel. Transmission control protocol. Technical Report RFC 793, IETF, September 1981.

[RA97] Robert Rönngren and Rassul Ayani. A comparative study of parallel and sequential priority queue algorithms. *ACM Transactions on Modelling and Computer Simulation*, 7(2):157–209, April 1997.

[Rah91] Kauko Rahko. Measurements for control and modelling teletraffic. In A. Jensen and V.B. Iversen, editors, *Period of Change, ITC-13*. IAC, Elsevier Science Publishers B.V., 1991.

[RK99] Jussi Ruutu and Kalevi Kilkki. More about sima. URL:http://www-nrc.nokia.com/sima/, March 1999.

[RM98] Jim Roberts and Laurent Massoulie. Bandwidth sharing and admission control for elastic traffic. In *Proceedings of ITC specialist seminar*. ITC, October 1998.

[SCFJ96] H. Schulzrinne, S. Casner, R. Frederick, and Van Jacobson. Rtp: A transport protocol for real-time applications. Technical Report RFC 1889, IETF, January 1996.

[SG94] Abraham Silberschatz and Peter Galvin. *Operating system concepts*. Addison-Wesley, 1994.

[She95] Scott Shenker. Fundamental design issues for the future internet. *IEEE Journal on Selected Areas in Communications*, 13(7):1176–1188, September 1995.

[SLCG89] Moshe Sidi, Wen-Zu Liu, Israel Cidon, and Inder Gopal. Congestion control through input rate regulation. In *Proceedings of GLOBECOMM'89*, pages 1764–1768. IEEE, November 1989. Also in IEEE Transactions on Communications V(41), N(3), March 1993.

[SPG97] Scott Shenker, C Partridge, and Roch Guerin. Specification of guaranteed quality of service. Technical Report RFC 2212, IETF, September 1997.

[SSZ98] Ion Stoica, Scott Shenker, and Hui Zhang. Core-stateless fair queueing: Achieving approximately fair bandwidth allocations in high speed networks. In *Proceedings of SIGCOMM'98*, pages 118–130. ACM, 1998.

[Ste97] W Stevens. Tcp slow start, congestion avoidance, fast retransmit, and fast recovery algorithms. Technical Report RFC 2001, IETF, January 1997.

[SV96] Dimitrios Stiliadis and Anujan Varma. Design and analysis of frame-based fair queueing: A new traffic scheduling algorithm for packet switched networks. In *Proceedings of SIGMETRICS'96*. ACM, 1996.

[SZN97] Ion Stoica, Hui Zhang, and T. S. Eugene Ng. A hierarchical fair service curve algorithm for link-sharing, real-time and priority service. In *Proceedings of SIGCOMM'97*. ACM, 1997.

[THD$^+$99] Benjamin Teitelbaum, Susan Hares, Larry Dunn, Robert Neilson, Vishy Narayan, and Francis Reichmeyer. Internet2 qbone: Building a testbed for differentiated services. *IEEE Network*, 13(5):8–16, September 1999.

[TT99] Pugi P. Tang and Tsung-Yuan C. Tai. Network traffic characterization using token bucket model. In *Proceedings of INFOCOMM'99*, volume 1, pages 51–62. IEEE, 1999.

[VFJ$^+$00] B. Vandalore, S. Fahmy, R. Jain, R. Goyal, and M. Goyal. General weighted fairness and its support in explicit rate switch algorithms. *Computer Communications*, 23:149–161, 2000.

[WGC$^+$95] Ian Wakeman, Atanu Ghosh, Jon Crowcroft, Van Jacobson, and Sally Floyd. Implementing real time packet forwarding policies using streams. In *Proceedings Usenix Technical Conference 1995*, pages 71–82. USENIX, January 1995.

[Wro97a] John Wroclawski. Specification of the controlled-load network element service. Technical Report RFC 2211, IETF, September 1997.

[Wro97b] John Wroclawski. The use of rsvp with ietf integrated services. Technical Report RFC 2210, IETF, September 1997.

[Zha95] Hui Zhang. Service disciplines for guaranteed performance service in packet-switching networks. *Proceedings of IEEE*, 83(10):1374–1396, October 1995.