ABSTRACT

ENHANCING QOS PROVISIONING AND GRANULARITY IN NEXT GENERATION INTERNET

by
Jie Yang

Next Generation IP technology has the potential to prevail, both in the access and in the core networks, as we are moving towards a multi-service, multimedia and high-speed networking environment. Many new applications, including the multimedia applications, have been developed and deployed, and demand Quality of Service (QoS) support from the Internet, in addition to the current best effort service. Therefore, QoS provisioning techniques in the Internet to guarantee some specific QoS parameters are more a requirement than a desire. Due to the large amount of data flows and bandwidth demand, as well as the various QoS requirements, scalability and fine granularity in QoS provisioning are required. In this dissertation, the end-to-end QoS provisioning mechanisms are mainly studied, in order to provide scalable services with fine granularity to the users, so that both users and network service providers can achieve more benefits from the QoS provisioned in the network.

To provide the end-to-end QoS guarantee, single-node QoS provisioning schemes have to be deployed at each router, and therefore, in this dissertation, such schemes are studied prior to the study of the end-to-end QoS provisioning mechanisms. Specifically, the effective sharing of the output bandwidth among the large amount of data flows is studied, so that fairness in the bandwidth allocation among the flows can be achieved in a scalable fashion. A dual-rate grouping architecture is proposed in this dissertation, in which the granularity in rate allocation can be enhanced, while the scalability of the one-rate grouping architecture is still maintained. It is demonstrated that the dual-rate grouping architecture approximates the ideal per-flow based PFQ architecture better than the one-rate grouping architecture, and provides better immunity capability.

On the end-to-end QoS provisioning, a new Endpoint Admission Control scheme for Diffserv networks, referred to as Explicit Endpoint Admission Control (EEAC), is proposed, in which the admission control decision is made by the end hosts based on the end-to-end performance of the network. A novel concept, namely the service vector, is introduced, by which an end host can choose different services at different routers along its data path. Thus, the proposed service provisioning paradigm decouples the end-to-end QoS provisioning from the service provisioning at each router, and the end-to-end QoS granularity in the Diffserv networks can be enhanced, while the implementation complexity of the Diffserv model is maintained. Furthermore, several aspects of the implementation of the EEAC and service vector paradigm, referred to as EEAC-SV, in the Diffserv architecture are also investigated. The performance analysis and simulation results demonstrate that the proposed EEAC-SV scheme, not only increases the benefit to the service users, but also enhances the benefit to the network service provider in terms of network resource utilization. The study also indicates that the proposed EEAC-SV scheme can provide a compatible and friendly networking environment to the conventional TCP flows, and the scheme can be deployed in the current Internet in an incremental and gradual fashion.

ENHANCING QOS PROVISIONING AND GRANULARITY IN NEXT GENERATION INTERNET

by
Jie Yang

A Dissertation
Submitted to the Faculty of
New Jersey Institute of Technology
in Partial Fulfillment of the Requirements for the Degree of
Doctor of Philosophy in Electrical Engineering

Department of Electrical and Computer Engineering

May 2004

APPROVAL PAGE

ENHANCING QOS PROVISIONING AND GRANULARITY IN NEXT GENERATION INTERNET

Jie Yang

Dr. Symeon Papavassiliou, Dissertation Advisor — Date
Assistant Professor of Electrical and Computer Engineering, New Jersey Institute of Technology

Dr. Nirwan Ansari, Committee Member — Date
Professor of Electrical and Computer Engineering, New Jersey Institute of Technology

Dr. Roberto Rojas-Cessa, Committee Member — Date
Assistant Professor of Electrical and Computer Engineering, New Jersey Institute of Technology

Dr. Mostafa Hashem Sherif, Committee Member — Date
Distinguished Member of Technical Staff, AT&T Labs

Dr. Meng-Chu Zhou, Committee Member — Date
Professor of Electrical and Computer Engineering, New Jersey Institute of Technology

BIOGRAPHICAL SKETCH

Author: Jie Yang

Degree: Doctor of Philosophy

Date: May 2004

Undergraduate and Graduate Education:

- Master of Science in Communication and Information Systems,
  Xidian University, China, 1999
- Bachelor of Science in Information Engineering,
  Xidian University, China, 1996

Major: Electrical Engineering

Presentations and Publications:

J. Yang, J. Ye, S. Papavassiliou, and N. Ansari,
"A Flexible and Distributed Architecture for Adaptive End-to-End QoS Provisioning in Next Generation Networks," accepted by *IEEE Journal on Selected Areas in Communications*, 2004.

J. Yang, J. Ye, S. Papavassiliou, and N. Ansari,
"Decoupling End-to-End QoS Provisioning from Service Provisioning at Routers in the Diffserv Network Model," submitted to *IEEE Globecom 2004*, 2004.

J. Zhu, J. Yang, and S. Papavassiliou,
"Quality-driven Information Processing and Gathering in Distributed Sensor Networks," in preparation for journal publication, 2004.

J. Yang, J. Ye, and S. Papavassiliou,
"Enhancing End-to-End QoS Granularity in Diffserv Networks via Service Vector and Explicit Endpoint Admission Control," *IEE proceedings on Communications*, vol. 151, no. 1, pp. 77-81, February 2004.

J. Yang, J. Ye, and S. Papavassiliou,
"A New Differentiated Service Model Paradigm via Explicit Endpoint Admission Control," *Proc. of ISCC2003*, vol. 1, pp. 299-304, June 2003.

D. Wei, J. Yang, N. Ansari, and S. Papavassiliou,
"Guaranteeing Service Rates for Cell-based Schedulers with a Grouping Architecture," *IEE Proceedings on Communications*, vol. 150, no. 1, pp. 1-5, February 2003.

D. Wei, J. Yang, N. Ansari, and S. Papavassiliou,
"Cell-based Schedulers With Dual-rate Grouping," *IEICE Transactions on Communications*, vol. E86B, no. 2, pp. 637-645, February 2003.

S. Papavassiliou and J. Yang,
"Admission Control in Wired Networks", *Encyclopedia of Telecommunications*, Wiley, January 2003.

D. Wei, J. Yang, N. Ansari, and S. Papavassiliou,
"Implementing the Dual-rate Grouping Scheme in Cell-based Schedulers," *Proc. of IEEE Global Communications Conference (GLOBECOM 2002),* vol. 3, pp. 17-21, November 2002.

J. Hou and J. Yang and S. Papavassiliou,
" Integration of Pricing with Call Admission Control to Meet QoS Requirements in Cellular Networks," *IEEE Trans. On Parallel and Distributed Systems*, vol. 13, no. 9, pp. 898-910, September 2002.

J. Yang and S. Papavassiliou,
"Secure and Reliable Data Delivery in Ad Hoc Wireless Networks via Traffic Dispersion," *Proc. of 3rd Annual Information Assurance Workshop*, June 2002.

J. Yang, D. Wei, S. Papavassiliou and N. Ansari,
"Improving Service Rate Granularity by Dual-rate Session Grouping in Cell-based Schedulers", *Proc. of IEEE Globecom 2001*, vol. 4, pp. 2425-2429, November 2001.

J. Hou, J. Yang and S. Papavassiliou,
"Integration of Pricing with Call Admission Control for Wireless Networks", *Proc. IEEE Vehicular Technology Conference (VTC2001)*, pp. 1344-1348, October 2001.

J. Yang, N. Uzun and S. Papavassiliou,
"The Architecture Design for a Terabit IP Switch Router", *Proc. of IEEE workshop on High Performance Switch and Router*, pp. 358-362, October 2001.

J. Yang, and S. Papavassiliou,
"Improving Network Security by Multipath Traffic Dispersion," *Proc. of IEEE Milcom 2001*, vol.1, pp. 34-38, October 2001.

J. Yang, N. Uzun and S. Papavassiliou,
"T^2MPS: Architecture of a Terabit Multicast Packet Switch Emulating an OQ Switch," *Proc. of CISS 2001*, vol. 2, pp. 798-803, March 2001.

To my parents and family

ACKNOWLEDGMENT

I would like to express my sincere gratitude to my advisor, Professor Symeon Papavassiliou, for his invaluable suggestions, continuous support and dedicated help. I deeply appreciate his advice, guidance, and the academic atmosphere and freedom he brought to our research group, and treasure the opportunities he created for me to improve my research and professional skills. Sometimes I feel he is like a big brother in the sense of the consideration, care, and help he gave. I was fortunate to have had him as my advisor.

I am also very grateful to Professor Nirwan Ansari, who offered his help on my research throughout these years. I was his "uncounted" student and a large portion of my work was co-authored with him. I feel fortunate and will never forget his support and contributions to my academic development.

My sincere appreciation extends to Dr. Necdet Uzun (Cisco Systems), who led me into research during my first year at NJIT and after that, offered me as much as he could to help and support me. I also acknowledge the valuable comments and discussion with my committee members: Dr. Mostafa Hashem Sherif (AT&T Labs), Professor Roberto Rojas-Cessa, and Professor Meng-Chu Zhou. Especially I would like to give thanks to Dr. Sherif for his suggestions and helpful reviews on my dissertation, and for the academic insight and motivation he gave me. I would like to thank Professor Rojas-Cessa and Professor Zhou for their encouragement and contributions to my professional development and for their interest and suggestions for this work.

Throughout the years, I have enjoyed working with my fellow researchers: Dong Wei, Jiongkuan Hou, Jian Ye, and Jin Zhu, from whom I received a lot of sincere help and unselfish support. We have worked together on many interesting research topics and achieved fruitful results that we can be proud of. I also appreciate their friendship and that of Feihong Chen, Litao Gang, Surong Zeng, Sheng Xu, Shuangshuang Dai, Jun Jiang, Chengzhou Li and his wife, Yu Zhang, Mizhou Tan and her husband, Jun Li, as well as

many other friends on a long list for the help, care and support I received. I will never forget the time we spent together these years.

Finally, I express my special gratitude to my parents, for their dedicated and endless love. They supported me with their best in every aspect. Without them, I would not have achieved anything that I achieved today. In my eyes, they are the greatest parents that one could have. I wish them the very best from the bottom of my heart. I would also thank my brother, Yuan Wang, for his faith in me and for taking care of our parents in China. My English is always not good enough to express the gratitude and love I feel toward them.

TABLE OF CONTENTS

Chapter		Page
1	INTRODUCTION	1
	1.1 QoS Provisioning in the Internet	2
	1.2 Scalability and Granularity in QoS Provisioning	4
	1.3 Dissertation Objective and Outline	6
2	IMPROVING SERVICE RATE GRANULARITY IN CELL-BASED SCHEDULERS	8
	2.1 Introduction	8
	2.1.1 Related Work	8
	2.1.2 Objectives and Outline	10
	2.2 Dual-rate Grouping Architecture	12
	2.3 Realization of Dual-rate Grouping Architecture	15
	2.4 Performance Study	18
	2.4.1 Effect of Backlogged Flow Length on Performance	18
	2.4.2 Simulation Results	19
	2.5 Conclusion	28
3	THE FRAMEWORK OF EXPLICIT ENDPOINT ADMISSION CONTROL AND SERVICE VECTOR	35
	3.1 Introduction	35
	3.2 Motivations and Objectives	36
	3.3 Related Work	39
	3.4 Framework of the Explicit Endpoint Admission Control and Service Vector Paradigm	42
	3.4.1 The Architecture and Operation of EEAC	42
	3.4.2 Optimization Models of The User and Network Sides	44
	3.4.3 Discussions on the User Optimization Model	46
	3.5 Performance Evaluation of Different Solutions	47

TABLE OF CONTENTS
(Continued)

Chapter **Page**

3.5.1 Network Topology and Assumptions 49

3.5.2 Simulation Results and Discussion 51

3.6 Conclusion 55

4 IMPLEMENTATION OF EEAC AND SERVICE VECTOR IN DIFFSERV NETWORKS 56

4.1 Introduction 56

4.2 Related Work 57

4.3 The End-to-End Service Provisioning Architecture 58

4.3.1 Implementation of EEAC-SV via RSVP 58

4.3.2 The Probing Packet Marking Scheme 59

4.4 Properties and Advantages of the Service Provisioning Architecture 65

4.5 Performance Evaluation and Discussion 69

4.5.1 Impact of K on the Performance of the EEAC-SV Scheme 71

4.5.2 Discussions on Probing Overhead and Probing Period 74

4.5.3 Experiment 1: Heterogeneous Traffic Burstiness 78

4.5.4 Experiment 2 – Flows with Heterogeneous QoS Requirements . . . 79

4.5.5 Deployment Studies 83

4.6 Conclusion 87

5 CONCLUSIONS AND FUTURE WORK 92

5.1 Summary of Contributions 92

5.2 Future Work 94

REFERENCES 97

LIST OF TABLES

Table		Page
2.1	Notations of Chapter 2	34
3.1	Notations of Chapter 3	38
3.2	The summary of bandwidth occupied by the cross traffic	50
4.1	Notations of Chapter 4	90
4.2	The performance bounds under different congestion levels for the EF, AF and BE services at a core router	91
4.3	The performance bounds under different congestion levels for the EF, AF and BE services at a core router in the EEAC-SV scheme	91
4.4	The overhead for delivering each type of data flows by the EEAC-SV when $T = 0.5s$	91

LIST OF FIGURES

Figure		Page
2.1	The operational model of the IP Switch Router.	11
2.2	The cell-based scheduler architecture.	12
2.3	The relationship between the guaranteed service rate and the number of cells in a backlogged flow.	19
2.4	Bandwidth allocation of S_1 in Experiment 1.	21
2.5	Bandwidth allocation of S_2 in Experiment 1.	22
2.6	Queueing delay of S_1 in Experiment 1.	23
2.7	Queueing delay of S_2 in Experiment 1.	24
2.8	Bandwidth allocation of S_1 in Experiment 2.	25
2.9	Bandwidth allocation of S_2 in Experiment 2.	26
2.10	Queueing delay of S_1 in Experiment 2.	26
2.11	Bandwidth allocation of S_1 in Experiment 3.	29
2.12	Bandwidth allocation of S_2 in Experiment 3.	30
2.13	Queueing delay of S_1 in Experiment 3.	31
2.14	Queueing delay of S_2 in Experiment 3.	32
3.1	EEAC procedure in a Diffserv network.	43
3.2	An example of the non-elastic utility function.	48
3.3	The simulated network topology.	50
3.4	The average user cost per packet with different requested load.	52
3.5	The request dropping probability with different requested load.	53
3.6	The average end-to-end delay with different requested load.	54
4.1	The pseudo-code for the process of the data packet or probing acknowledgement packet arrival in a router.	65
4.2	The pseudo-code for the process of the probing packet arrival in a router.	66
4.3	The simulated network topology.	70
4.4	The request dropping probability with different K for flow type 1.	73

LIST OF FIGURES
(Continued)

Figure **Page**

4.5 The packet dropping probability with different K for flow type 1. 74

4.6 The probing overhead with different T under the requested load 1.0. 75

4.7 The end-to-end packet dropping probability with different requested load when $T = 0.5s$. 77

4.8 The end-to-end packet dropping probability with different T when the requested load is 1.0. 78

4.9 The end-to-end delay of flow type 1 and 2 with different requested load and traffic burstiness. 79

4.10 The request dropping probability of flow type 1 and 2 with different requested load and traffic burstiness. 80

4.11 The average end-to-end delay of flow type 1 and 2 under different operation schemes. 82

4.12 The request dropping probability of flow type 1 and 2 under different operation schemes. 83

4.13 The throughput of TCP flows under the requested load of 1.0. 85

4.14 The throughput of type 1 flows under the requested load of 1.0. 86

4.15 The request dropping probability for the type 1 flows. 88

CHAPTER 1

INTRODUCTION

Internet has experienced an exponential growth since the 90s of last century [1]. With the increasing popularity of the Internet applications and services, the networking environment and architecture is also evolving towards a new generation, which will demonstrate significantly different features from today's Internet:

- The Internet Protocol (IP) may become the vehicle for the integration of data and conventional telecommunication services. The simplicity of IP makes it economical to provide conventional telecommunication services, such as voice, over IP networks. The third and fourth generation of wireless networks are also required to provide seamless support of data services over the IP infrastructure.

- The access to the Internet services has expanded from wired networks to wireless networks, and from fixed networks to mobile networks. The terminals that support Internet access include not only desktops and notebooks, but also PDAs, mobile phones, etc. The location that provides Internet access is not limited to home or office, where fixed network infrastructure exists, but it is extended to cars, trains, or even airplanes where mobile and wireless network infrastructure are available.

- Higher data rates and capacity at both core and access networks. In the backbone, the data rate is increasing dramatically from OC-3 (155Mbps) to OC-192 (10G bps). In the access network, the data rate of Fast Ethernet has reached 100Mbps, ten times as fast as the earlier Ethernet (10M bps), and is envisioned to reach 1G bps; while the wireless access speed has increased from 11 Mbps to 56 Mbps in Wireless LAN (WLAN). Moreover, the DSL or Cable modems are becoming more widely deployed to upgrade the home Internet access from 56K modem to around 1M bps.

- New applications, including the multimedia applications (e.g. video conference, on-line broadcast) and real-time communications (e.g. Voice over IP) have been developed and deployed in the Internet. These applications may consume more network resources, such as bandwidth and memory buffers, than conventional Internet applications, such as e-mails and web browsing.

Unlike the conventional Internet applications, such as email and web page browsing, for which only the best effort service in the Internet is required, the new features of the next generation Internet now demand *Quality of Service* (QoS) support in the network infrastructure. QoS is referred to as the capability by which a network or a network equipment provides guarantees of certain performance to a data flow or network traffic, measured through various parameters at different layers. In this dissertation, the QoS is studied at the network layer of the OSI reference model and the Internet layer of the TCP/IP reference model [2], whose parameters include average delay and delay jitter of a packet, bandwidth and goodput of a data connection, packet loss probability during the data transfer, etc.

1.1 QoS Provisioning in the Internet

At the network layer, the QoS provisioning techniques can be classified as the mechanisms and strategies that aim to guarantee the QoS within a single node, and the mechanisms and strategies that aim to guarantee the QoS end-to-end or edge-to-edge within a network domain [3]. The single-node QoS provisioning techniques include packet forwarding techniques and resource management techniques. Fast packet forwarding techniques, which is one of the prerequisites for QoS support, are needed to ensure that sufficient bandwidth can be provided to accommodate the traffic in the Internet. The breakthrough in link speed at both core and access networks has laid ground for fast forwarding of packets and for providing QoS in the Internet. Correspondingly higher link speed also

requires routers to work at much higher data rate with high efficiency to provide: 1. fast packet delivery without blocking; 2. high capacity to accommodate large amount of data.

In addition to the high data rate, resource (output bandwidth, internal buffers, etc.) allocation and management schemes need to be implemented in a high capacity router. This is due to the fact that the increase of resources, such as bandwidth and buffers, cannot be unlimited while the demand will never stop, as new services and applications keep emerging and the number of users keep increasing. Furthermore, the bursty nature of the Internet traffic makes it neither economical nor efficient, to always attempt to meet the resource requirements of a data flow at its peak rate; while if the resources are not allocated according to the peak rate, more sophisticated resource allocation schemes are needed to guarantee the QoS. Specifically, resource allocation and management schemes may include buffer and queue management schemes, traffic shaping, policing and scheduling schemes, etc.

Although the single-node QoS provisioning mechanisms can help improve the overall offered QoS in the network, from the user point of view, it is the end-to-end performance that is of importance. The end-to-end or edge-to-edge QoS provisioning provides the capability for a network to guarantee the QoS in an end-to-end or edge-to-edge fashion. It consists of the techniques that coordinate the single-node QoS provisioning schemes on an end-to-end or edge-to-edge basis. This dissertation will not differentiate between the concepts of "end-to-end" and of "edge-to-edge", and will discuss "end-to-end" only. However, the results can be easily extended and applied to the "edge-to-edge" case.

Traditionally, end-to-end QoS cannot be guaranteed in the Internet and the service provided is referred to as "best effort". To guarantee the end-to-end QoS performance, new service models have to be introduced which provide not only the conventional connectionless best effort service in the Internet, but also services which are beyond best effort and able to guarantee the required QoS of a connection in an end-to-end fashion.

Currently there are two service models proposed for end-to-end QoS provisioning in the Internet: Intserv [4] and Diffserv [5]. The Intserv model aims to provide network services to each individual user data flow. Resource reservation and allocation are performed for each flow, which is similar to providing network services to the virtual connections in Asynchronous Transfer Mode (ATM) networks. A signaling protocol, namely, the resource ReSerVation Protocol (RSVP) [6] was proposed so that admission control and resource allocation can be performed in a dynamic manner and the state of each flow can be refreshed in the Intserv routers. The Intserv model requires per-flow management in the network, while the router needs to keep information about the state of each flow.

On the other hand, only a number of services are provided to users in the Diffserv model, based on the Service Level Agreement (SLA) between the user and the service provider. Traffic flows will be marked by the host or leaf router and classified, metered, shaped, and possibly re-marked at the ingress router where the flows will be aggregated according to the service class and forwarded to the core routers. At core routers, the aggregated flows are serviced according to the Per-Hop-Behavior (PHB) associated with their service classes. Thus the core router does not keep the state of each individual flow.

In addition to the architecture of the service models, the end-to-end QoS provisioning also includes the general QoS policing and management schemes. These schemes may include admission control policy and mechanism, QoS routing policy and mechanism, QoS pricing policy and mechanism, QoS deployment and other end-to-end QoS monitoring, control and administration mechanisms.

1.2 Scalability and Granularity in QoS Provisioning

With the high data rate supported in the core and access networks, both the single-node and end-to-end QoS provisioning techniques often face the trade-offs between the QoS granularity and solution scalability. In general, fine QoS granularity is desired by both users and service providers since it can provide fair resource allocation, efficient resource

utilization, and more robust services. However, scalable QoS provisioning schemes may only support coarse QoS granularity, while solutions to provide fine QoS granularity are usually either not scalable to accommodate the large amount of flows in a router, or too prohibitive to be implemented.

For example, the Generalized Processor Sharing (GPS) algorithm and its packet-based version, *Packet-by-packet* Generalized Processor Sharing (PGPS) [7] are ideal QoS provisioning techniques for a single node to guarantee the bandwidth allocation and delay bound of each individual data flow, i.e., they can provide the finest QoS granularity in terms of bandwidth allocation. However, they require per-flow based packet queueing and scheduling. Considering the high data rate and hence the huge amount of data flows coexisting in a network node (router), it may be prohibitive to realize an ideal per-flow based PGPS scheduler in a router. Trade-offs have to be made between the scalability of the implementation and the instantaneous bandwidth allocation accuracy, i.e., the QoS provisioning granularity, as compared to the ideal scheme. Implementations of scheduling algorithms with coarse QoS granularity will degrade the bandwidth and delay guarantee capability of the scheduling algorithm, affect negatively the fairness in bandwidth allocation, and reduce its service provisioning robustness in the sense that the conforming flows will be negatively affected by ill-behaving flows and cannot get their desired service performance.

Among the end-to-end QoS provisioning schemes, similar problems exist. For example, in the Intserv model, per-flow based resource reservation and allocation have to be performed end-to-end so that each flow's end-to-end QoS requirements can be supported in the network. However, by performing per-flow based resource management, the Intserv model suffers from the scalability problem because each router has to keep the state information for each flow and reserve resource for its QoS requirements. The Diffserv model is considered as a scalable solution in that individual flows are aggregated at the edge routers and provisioned with QoS based on the flows' class in the core networks.

Since the number of service classes is negligible as compared to the number of individual flows in the Intserv model, and the core routers will not differentiate individual flows but only the service class of the aggregated traffic, the core routers do not suffer from the scalability problem. However, in this model, only coarse end-to-end QoS granularity can be supported. Flows of the same service class along the same path in the network will receive the same end-to-end QoS, even if they have quite different QoS requirements. Thus, in this model, the network resource utilization may be reduced since users may obtain better services than what they desired, while users may achieve less net benefit from the services if service-based pricing is applied.

1.3 Dissertation Objective and Outline

Since coarse QoS granularity, provisioned either at a single node or end-to-end, will degrade the benefits that can be achieved by both the users and network service providers, in this dissertation, some QoS provisioning techniques, which can provide scalable solutions with fine QoS granularity, are studied. Although the emphasis is placed on the end-to-end QoS provisioning schemes, single-node QoS provisioning schemes that need to be deployed at each router in the network will also be discussed.

The remaining of this dissertation is organized as follows.

In Chapter 2, a novel and scalable traffic scheduling architecture is introduced and described to provide fine rate-allocation granularity in a router. The proposed scheduling architecture can be applied to both performing per-flow based scheduling at routers in the Intserv model or edge routers in the Diffserv model, and performing bandwidth allocation among different service classes at core routers in the Diffserv model.

Since Diffserv is a more scalable service model in the next generation Internet, the emphasis from Chapter 3 and on is placed on the end-to-end QoS provisioning strategies in the Diffserv model. The objective is to enhance the end-to-end QoS granularity provisioned in the Diffserv network while maintaining its scalability feature.

In Chapter 3, a novel end-to-end QoS provisioning strategy, referred to as Explicit Endpoint Admission Control (EEAC), is proposed and its effect on the user benefit is studied, while in Chapter 4 the implementation of the proposed strategy, referred to as EEAC-SV, in the Diffserv networks is described. The simulation results demonstrate that the proposed scheme can enhance the end-to-end QoS granularity provisioned in the Diffserv network and maintain the scalability feature of Diffserv networks. The proposed model and strategy can also provide a friendly networking environment for the conventional TCP flows in the Diffserv network, and can be deployed incrementally and gradually based on today's Internet QoS infrastructure. Finally Chapter 5 concludes the dissertation and discusses directions for future research.

CHAPTER 2

IMPROVING SERVICE RATE GRANULARITY IN CELL-BASED SCHEDULERS

2.1 Introduction

The Integrated and Differentiated Service models in the Internet [1] require that packet switches support various service classes and provide connections to large amount of flows over a single resource-shared physical infrastructure. Therefore, the scheduler design is critical to the single-node QoS that a packet switch is able to provide in terms of bandwidth allocation and delay bound guarantee. For this reason, packet scheduling algorithms have been intensively studied in recent years.

It is well known that the Generalized Processor Sharing (GPS) algorithm and its packet-based version, *Packet-by-packet* Generalized Processor Sharing (PGPS) [7] are idealized algorithms which are able to: a) guarantee delay bound to leaky-bucket constrain-ed flows; b) provide real-time fair allocation of bandwidth among backlogged flows, regardless of the behavior of the flows; and c) isolate well-behaving flows from the disadvantageous effects of other ill-behaving flows. However, GPS and PGPS are difficult to be implemented in packet schedulers due to their complexities. In the literature, a lot of scheduling algorithms for packet switching that approximate GPS and PGPS have been proposed such as WFQ [8], EFQ [9], H-PFQ [10] *etc.*, which are generally referred to as Packet Fair Queueing (PFQ) algorithms [11]. The objective of these algorithms is to find a balance between the implementation complexity (i.e., scalability) and the performance approximation to the idealized GPS (i.e., service granularity).

2.1.1 Related Work

Generally, the implementation complexity of PFQ algorithms is determined by the following factors [12]: 1) the calculation of the virtual system time; 2) sorting the service order of all flows; 3) the management of another priority queue to regulate packets, if

those algorithms with the "smallest eligible virtual finish time first," such as WF^2Q [13] or WF^2Q+ [10], are adopted as the service disciplines.

The PGPS [7],[14] and Weighted Fair Queuing [8] algorithms use the virtual system time defined by the fluid GPS model, which need to track all backlogged flows, and hence the worst case complexity is $O(N)$, where N is the number of flows. Some other PFQ algorithms, with virtual system time complexity $O(1)$ [15] and $O(\log N)$ [10] [16], have been developed.

The sorting complexity of most algorithms is $O(\log N)$. S. Suri, et al. [17], proposed to use the van Emde Boas data structure, which has the complexity of $O(\log \log N)$.

Stephens *et al.* [11], [18] proposed a grouping architecture for schedulers in which flows with same rate are grouped together and the scheduler only supports a fixed number of rates. The implementation of the sorting complexity is then reduced from processing each flow in the scheduler to processing each service rate group supported by the scheduler. For example, if WF^2Q+ is employed, the sorting complexity is reduced from $O(\log N)$ to $O(\log M)$, where N is the number of flows and M is the number of service rate groups. The scalability of the PFQ algorithm family is thus improved. However, in such a grouping architecture, the rate granularity of a scheduler becomes coarse, which limits the fairness among the flows. For example, in a scheduler which supports exponentially distributed rates with 1Mbps, 2Mbps, 4Mbps,...., *etc.*, when a flow only requires a rate of 1.25Mbps, the typical solution is to provide it with the rate of 2Mbps. This solution not only makes the flow user take advantage of the service provider but also is unfair to other flows because such a flow will be able to consume more resources than it is supposed to and thus other flows may not get sufficient resources. Coarse granularity of the rate allocation in a router caused by the grouping architecture will also degrade the immunity capability of PFQ algorithms, which isolates ill-behaving flows from negatively affecting the conforming flows.

Moreover, the grouping architecture will also introduce the trade-off between the implementation complexity of the admission control policy and the link utilization. In general, the implementation complexity of the admission control policy for the ideal PFQ scheduler is $O(1)$, while the worst-case complexity of the admission control for the grouping architecture is $O(\log N)$ to achieve the same link utilization. Otherwise, if the admission control algorithm with the implementation complexity of $O(1)$ is applied in the grouping architecture, the link utilization will be lowered as compared to the ideal PFQ scheduler, and the level of under-utilization will be decided by the rate granularity provided in the grouping architecture. Interested readers may refer to [19] for further discussions on the admission control in the grouping architecture.

2.1.2 Objectives and Outline

In this chapter a scheme is proposed to enhance the rate granularity of the grouping architecture, in which when a flow requires a rate between two rates supported by the scheduler, it is split into two sub-flows which are enqueued into the corresponding rate groups respectively. The ratio of the entire flow in each of the sub-flow queues will ensure the average guaranteed service rate is equal to the rate that the flow requires. This scheme is referred to as *dual-rate grouping*. With dual-rate grouping, the current grouping architecture do not need to be changed and the rate granularity can be improved without increasing the number of rate groups, i.e., without increasing the sorting complexity of the scheduler. As long as there is sufficient memory to represent the ratio of a flow in a sub-flow queue, any rate a user requires can be provided by using the M rate groups, which maintains the sorting complexity as $O(\log M)$. This scheme is studied under the context that the length of each packet is fixed, referred to as a *cell*, in an IP switch router.

The operational model of the IP switch router is shown in Figure 2.1. In this model, when an IP packet arrives at the IP switch router, at the inbound side, a table look-up will be performed to determine how to deal with the packet. Variable-length IP packets will be segmented into fixed length "cells" at the inbound side to utilize the fixed-length

packet (ATM) switch architecture and reassemble them into packets at the outbound side before they are forwarded to the next hop. It should be noted that what Figure 2.1 provides is the general operation model for the proposed IP switch router. Slight differences may exist when it is applied at different networking environments. For example, reassembling may not be necessary if it works as an ingress edge router at an ATM backbone network, while segmentation may not be needed when it works as an egress edge router in an ATM backbone network. The proposed scheduler can be applied at the inbound side to schedule individual flows to their outbound ports and at the outbound side to schedule flows to access the output link. The detailed description and performance of such an IP switch router architecture can be found in [20].

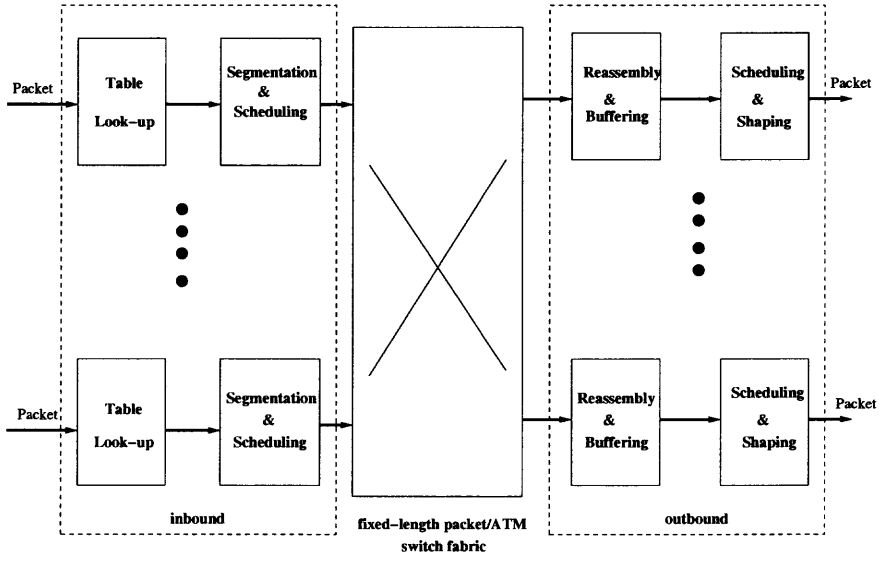

Figure 2.1 The operational model of the IP Switch Router.

The rest of this chapter is organized as follows. In section 2.2, the principle of the scheme and its associated performance properties are discussed. In section 2.3, the issues of implementation are explained, which include the rate error that will be induced during implementation to make this scheme easily realized, and mechanisms to maintain cell sequence of a flow while avoiding sorting in a rate group. The performance study and

simulation results are presented in section 2.4, while section 2.5 contains some concluding remarks.

2.2 Dual-rate Grouping Architecture

Similar to [11], the architecture of a cell-based scheduler with grouping architecture is shown in Figure 2.2, in which flows with same service rate requirement are grouped together into one *rate group*, while each flow in the rate group stores cells of that flow in a First In First Out (FIFO) fashion, which is therefore referred to as a *flow queue*. The scheduler will only support a limited number of rate groups and their flow queues. Although any type of PFQ algorithms can be applied in this scheduler architecture, in this chapter, WF^2Q+ [10] will be applied as the scheduling algorithm in the scheduler architecture, for the reason that WF^2Q+ algorithm has lower implementation complexity among the PFQ algorithms that most accurately approximate the ideal GPS algorithm.

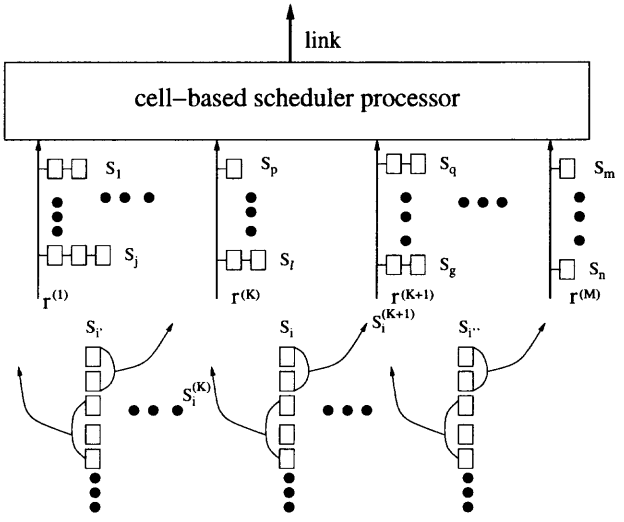

Figure 2.2 The cell-based scheduler architecture.

In each rate group, only the cell with the smallest virtual start time is placed into the scheduler processor, which in turn decides which cell should be served according to the packet selection policy of the corresponding PFQ algorithm. In Table 2.1, a list of most of the notations used throughout this chapter is provided. Without loss of generality, it is assumed that in Figure 2.2 the following condition holds: $r^{(1)} < r^{(2)} \cdots < r^{(M)}$. When a flow S_i requires a rate r_i that is between rate $r^{(K)}$ and $r^{(K+1)}$, i.e., $r^{(K)} < r_i \leq r^{(K+1)}$, where $r^{(K)}$ and $r^{(K+1)}$ are two consecutive rate groups supported by the scheduler, the conventional scheme adds S_i to rate group $(K+1)$ so that $\hat{r}_i$, the service rate guaranteed to S_i, will be $\hat{r}_i = r^{(K+1)} \geq r_i$ [19]. Such a scheme is referred to as *one-rate grouping architecture.*

Definition 2.1: For flow S_i, the relative error, e_i, between the required service rate r_i and the guaranteed service rate $\hat{r}_i$, is defined as

$$e_i = \left| \frac{\hat{r}_i - r_i}{r_i} \right| \tag{2.1}$$

It is easy to deduce that with one-rate grouping scheme, one will have

$$e_i \leq \gamma - 1 \tag{2.2}$$

where $\gamma = r^{(K+1)}/r^{(K)}$, $\gamma > 1$ and a rate r_i in the range $(r^{(K)}, r^{(K+1)}]$ is requested.

Based on such a cell-based, one-rate grouping scheduler architecture, the proposed *dual-rate grouping architecture* scheme works as follows: when S_i requests a rate r_i such that $r^{(K)} < r_i < r^{(K+1)}$, one can split S_i into two sub-flows, $S_i^{(K)}$ and $S_i^{(K+1)}$, which associate with rate groups K and $K+1$ respectively, and it is required that the average guaranteed service rate, $\hat{r}_i$, of the entire flow S_i, is equal to r_i. The ratio $x_i^{(K+1)}$ of cells in $S_i^{(K+1)}$ to the entire flow S_i is given by

$$x_i^{(K+1)} = \frac{r^{(K+1)}(r_i - r^{(K)})}{r_i(r^{(K+1)} - r^{(K)})} \tag{2.3}$$

where $r^{(1)} \leq r^{(K)} < r^{(K+1)} \leq r^{(M)}$ and flow i is leaky-bucket constrained by (L_i, r_i) [7].

Correspondingly,

$$x_i^{(K)} = 1 - x_i^{(K+1)} = \frac{r^{(K)}(r^{(K+1)} - r_i)}{r_i(r^{(K+1)} - r^{(K)})} \tag{2.4}$$

The ratio is achieved from that the service time for cells in $S_i^{(K)}$ is bounded by $\frac{lL_i x_i^{(K)}}{r^{(K)}}$. Correspondingly, the service time for cells in $S_i^{(K+1)}$ is bounded by $\frac{lL_i x_i^{(K+1)}}{r^{(K+1)}}$. Therefore, the average service rate, $\hat{r}_i$, guaranteed to S_i, is given by

$$\hat{r}_i = \frac{lL_i}{\frac{lL_i x_i^{(K)}}{r^{(K)}} + \frac{lL_i x_i^{(K+1)}}{r^{(K+1)}}} = \frac{1}{\frac{(1-x_i^{(K+1)})}{r^{(K)}} + \frac{x_i^{(K+1)}}{r^{(K+1)}}} \tag{2.5}$$

Since it is required that $\hat{r}_i = r_i$, then (2.3) and (2.4) can be achieved.

It should be noted that in the previous discussion the concepts of the guaranteed service rate $\hat{r}_i$ and the received service rate $\bar{r}_i$ are not differentiated. In general, $\hat{r}_i$ corresponds to the normalized weight in the PFQ algorithm while $\bar{r}_i$ is the bandwidth and service rate that a flow actually receives. One can have $\hat{r}_i \leq \bar{r}_i$, i.e. $\hat{r}_i$ is the guaranteed service rate to a flow, when the following condition is satisfied [19]:

$$\sum_{i=1}^{N} \hat{r}_i \leq C \tag{2.6}$$

where N is the number of flows in the scheduler and C is the output capacity.

From Eq. (2.3), it can be directly obtained that $x_i^{(K+1)}$ has the following features:

i) $0 < x_i^{(K+1)} \leq 1$;

ii) $x_i^{(K+1)}$ can be represented by

$$x_i^{(K+1)} = \frac{n_i}{d_i} \tag{2.7}$$

where $n_i > 0$, $d_i > 0$ and they are both integers.

It is easy to follow that with the dual-rate grouping architecture, the flow's average queueing delay D_i will be bounded by $L_i l/\hat{r}_i$. In the ideal case in which the ratio $x_i^{(K+1)}$ can be achieved, D_i is equal to the delay bound required by S_i and there is no relative error between the required service rate r_i and the guaranteed service rate $\hat{r}_i$.

2.3 Realization of Dual-rate Grouping Architecture

When the scheduler is implemented, a processing table has to be set up for each flow, which stores such information as the flow's current virtual start time s_i, the required service rate r_i, *etc.* To implement the dual-rate grouping, the representation of $x_i^{(K+1)}$, has to be implemented in the processing table. This can be achieved through a label for n_i and a counter c_i which resets at the value of d_i. Initially c_i is set to 0 and the flow queue is located in the rate group $K+1$ each time the flow is activated and backlogged. When a new cell of S_i departs from the flow queue, c_i is incremented by 1. If the flow is continuously backlogged and n_i cells have left the scheduler, the flow queue is moved to the tail of rate group K. When d_i cells have been departed, c_i is reset to 0 while flow is moved back to the tail of rate group $K+1$. The remaining operations of the scheduler, upon cell arrival and departure are the same as in the one-rate grouping architecture. Therefore, the implementation complexity remains the same as that in the one-rate grouping architecture. It should also be noted that the packet selection policy performed in the scheduler processor is not affected by the dual-rate grouping architecture.

For different $r^{(K)}$, r_i and $r^{(K+1)}$, different values of $x_i^{(K+1)}$ are required. For example, when $r^{(K)}$=1 Mbps, $r^{(K+1)}$=2 Mbps, if $r_i = 1.6$ Mbps, $x_i^{(K+1)} = \frac{3}{4}$, which can be implemented by four bits: two bits for n_i and two bits for c_i. In general, $\lceil \log_2 n_i \rceil$ bits for n_i, and $\lceil \log_2 d_i \rceil$ bits for c_i are needed, while $\lceil \log_2 n_i \rceil \leq \lceil \log_2 d_i \rceil$.

Since r_i can not be known *a priori*, the design of the dual-rate grouping architecture can be simplified as follows. When n_i and c_i are implemented, a fixed number of bits, b, are allocated in the processing table for the representation of n_i and c_i, respectively. To simplify the operation of the dual-rate grouping and save space for the processing table, the representation of d_i will not be implemented in the processing table. c_i will only be reset when it overflows and rolls over back to 0. Therefore, the possible values of $x_i^{(K+1)}$ are: $\frac{1}{2^b}, \frac{2}{2^b}, \cdots, \frac{2^b-1}{2^b}, 1$. From Eq.(2.5), it can be seen that each $x_i^{(K+1)}$ corresponds to a

specific $\hat{r}_i$. Therefore, the rate granularity is improved by adding $2^b - 1$ additional rates between the rate group $r^{(K)}$ and $r^{(K+1)}$.

For a user who requests a rate r_i that requires a $x_i^{(K+1)}$ not supported in the processing table , i.e. $\frac{n_i}{d_i} \neq \frac{j}{2^b}$, where $j = 1, 2, \cdots, 2^b - 1, 2^b$, a rate $\hat{r}_i > r_i$ is guaranteed to the user, which is also the closest rate to r_i and can be supported by the processing table. In this case, there is a relative error e_i between the service rate requested and guaranteed. In the following theorem, the upper bound of e_i is provided as follows.

Theorem 1 *Suppose* $r^{(K+1)} = \gamma r^{(K)}$, $\gamma > 1$. *If* $x_i^{(K+1)}$ *can only be represented in the processing table by the discrete values:* $x_i^{(K+1)} = \frac{j}{2^b}$, *where* $j = 1, 2, \cdots, 2^b - 1, 2^b$, *then* e_i *is bounded by*

$$e_i \leq \frac{\gamma - 1}{2^b} \tag{2.8}$$

Proof: Suppose that $\hat{r}_1$ and $\hat{r}_2$ are two consecutive guaranteed service rates in the processing table of the dual-rate grouping architecture and $r^{(K)} < \hat{r}_1 < \hat{r}_2 \leq r^{(K+1)}$. If $\hat{r}_1$ requires a ratio $x_1^{(K+1)}$ and $\hat{r}_2$ requires a ratio $x_2^{(K+1)}$, from Eq. (2.5), it can be derived that $x_1^{(K+1)} < x_2^{(K+1)}$, since Eq. (2.5) demonstrates that $\hat{r}_i$ is a monotonic increasing function of $x_i^{(K+1)}$. Let $x_1^{(K+1)} = \frac{n_1}{d_1}$ and $x_2^{(K+1)} = \frac{n_2}{d_2}$. From the implementation, $d_1 = d_2 = 2^b$, and $n_2 = n_1 + 1$. The e_i is maximized when a user requires a rate r_i slightly larger than $\hat{r}_1$, and is then provided with a rate $\hat{r}_2$. Therefore, from Eqs. (2.1) and (2.5), the following can be obtained.

$$e_i \leq \frac{\hat{r}_2}{\hat{r}_1} - 1 = \frac{\gamma - (\gamma-1)x_1^{(K+1)}}{\gamma - (\gamma-1)x_2^{(K+1)}} - 1 = \frac{(\gamma-1)/d_2}{\gamma - (\gamma-1)n_2/d_2} \leq \frac{\gamma-1}{d_2} = \frac{\gamma-1}{2^b}$$

Although from relation (2.8), it can be seen that e_i can be significantly reduced by using a large b, it requires long backlogged flow for the guaranteed rate to converge to the required value, which can be seen in subsection 2.4.1. Therefore, the rate granularity can benefit from large b only when the backlogged flow is long enough.

With one-rate grouping architecture, a flow queue is maintained in First In First Out (FIFO) manner. In the processing table, only the virtual start time at the head of the flow

queue needs to be stored. Each time a cell is fetched into the scheduler processor or a new flow comes in, the flow will be moved or appended to the tail of the rate group queue and a new virtual start time will be stored for that flow. It is proven in [11] that with this implementation, in a rate group queue, a flow with the smallest virtual start time will be served first by the processor without performing the costly sorting. Moreover, the sequence of cells in each flow will be maintained, which is important to virtual connections. In dual-rate grouping architecture, in a rate group the sequence between different flows and sub-flows can be maintained in the same way as in the one-rate grouping architecture.

A property of the PFQ algorithm is that the virtual clock $v(t)$, the virtual start time $s_i(t)$ and the virtual finish time $f_i(t)$ are monotonic increasing functions of time t. Moreover, when two cells k and $k+1$ arrive sequentially, the virtual start time has the following relationship [7]:

$$s_i^{k+1} = \max(v(t), s_i^k + \frac{l}{r_i^k}) \tag{2.9}$$

However, even in the one-rate grouping architecture, there is a complication in the implementation. Since the number of bits to represent the timestamp is finite, when the timestamp of virtual system time $v(t)$ is rolled over, the computation of the virtual start time for a new arrival flow may give wrong result. For this reason, in one-rate grouping architecture, when S_i is not backlogged as cell $k+1$ arrives, s_i^{k+1} will be assigned a value in the range $[v(t), v(t) + (l/r_i^k)]$. Moreover, in each rate group, sorting among different flows should be avoided. Therefore, the newly backlogged flow will be assigned a virtual start time at least as large as that of the tail of the backlogged flows in the rate group but still in the range $[v(t), v(t) + (l/r_i^k)]$ [18].

In the dual-rate grouping architecture, there are two additional complications regarding the overflow of the timestamp and sorting in the rate group. The first one is when cell $k+1$ arrives, cell k is not backlogged and cell $k+1$ enters a rate group different from that of cell k. The desired value of s_i^{k+1} is given by $\max(s_i^k + (l/r_i^k), v(t))$. However,

to guarantee the delay bound property of the rate group that cell $k+1$ will enter [11], s_i^{k+1} should be in the range $[s_T, s_H + (l/r_i^{k+1})]$, where s_H and s_T are the virtual start time of the head and tail of the backlogged flows in the rate group that cell $k+1$ will enter. Therefore s_i^{k+1} will be assigned a value given by $\max\{s_T, \min[s_H + (l/r_i^{k+1}), \max(s_i^k + (l/r_i^k), v(t))]\}$.

The second complication is that when cell $k+1$ arrives, cell k is backlogged, and cell $k+1$ will be in a different rate group after cell k departs from the flow. From Eq. (2.9), s_i^{k+1} should be given by $s_i^k + (l/r_i^k)$. But to avoid sorting and to satisfy the delay bound property in the new rate group, s_i^{k+1} should also be in the range $[s_T, s_H + (l/r_i^{k+1})]$, therefore, is actually given by $\max\{\min[s_i^k + (l/r_i^k), s_H + (l/r_i^{k+1})], s_T\}$.

2.4 Performance Study

2.4.1 Effect of Backlogged Flow Length on Performance

Note that the scheduler is based on cells. When the dual-rate grouping architecture is implemented, the number of cells in the backlogged flow will affect the performance of the scheme. For example, when the ratio $x_i^{(K+1)}$ is implemented by putting first n_i cells into rate group $K+1$ for every d_i continuously backlogged cells, if the length of the backlogged flow is smaller than n_i, the flow is guaranteed a service rate $r^{(K+1)}$. The performance converges to the average value given by(2.5) when the number of backlogged cells $k \to \infty$. Therefore, the larger d_i is, is the longer backlogged flow needed to get the desired performance by the dual-rate grouping architecture. This effect is shown by the following example. Suppose a user requires a rate of 3 Mbps, while the scheduler only has rate groups of 2 Mbps and 4 Mbps. By applying the dual-rate grouping architecture, $x_i^{(K+1)} = 2/3$ and $x_i^{(K)} = 1/3$. To implement this scheme, for every 3 continuously backlogged cells, first 2 cells are put to the rate group of 4 Mbps, and the last one cell to the rate group of 2 Mbps. If the backlogged flow is only 2-cell long, it is guaranteed a service rate of 4 Mbps, higher than it requires. However, the guaranteed service rate will

converge to 3 Mbps when the number of cells in the backlogged flow increases, as shown in Figure 2.3.

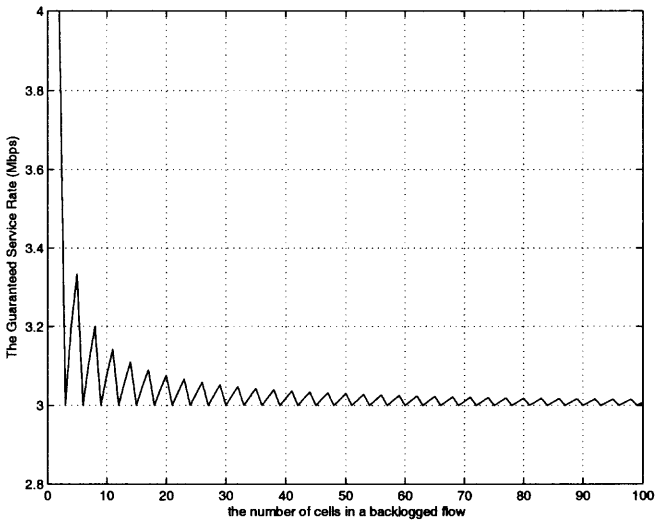

Figure 2.3 The relationship between the guaranteed service rate and the number of cells in a backlogged flow.

2.4.2 Simulation Results

In this subsection, the performance of the dual-rate grouping architecture is evaluated through modeling and simulation by OPNET. Specifically, four architectures are implement-ed and the corresponding performance are compared: WF^{2}Q+ with per-flow based architecture, WF^{2}Q+ with the one-rate grouping architecture, WF^{2}Q+ with the ideal dual-rate grouping architecture which can represent $x_i^{(K+1)}$ accurately by using any bit length to represent n_i and d_i of $x_i^{(K+1)}$, and WF^{2}Q+ with the simplified dual-rate grouping architecture which has only limited bits to represent n_i and d_i of $x_i^{(K+1)}$. In the simplified dual-rate grouping architecture, three bits are used to represent n_i and the counter c_i (i.e. d_i) respectively. Throughout the experiments, the output link capacity is $R = 8000$ cells/second. In all of the grouping architectures, including the one-rate grouping architecture and both of the dual-rate grouping architectures, $\gamma = 2$ is selected between neighboring rate groups. It should be noted that, in general, γ can be any

value larger than 1. The ratio of two is selected for simplicity in representation and implementation.

Experiment 1

The per-flow based WF^{2}Q+ algorithm can achieve fairness in instantaneous bandwidth allocation and the finest rate granularity. However, the one-rate grouping architecture may degrade the fairness due to its rate granularity. In the following Experiment 1, the improvement of rate granularity and fairness in bandwidth allocation in the dual-rate grouping architectures is presented and discussed. In the experiment, there are three flows S_1, S_2 and S_3 in the schedulers, each of which is leaky-bucket constrained by its required service rate and the token size of 1024 cells. Let r_1=2000 cells/second, which is 0.25 of the total output capacity, and $r_2 = r_3$=3000 cells/second, which is 0.375 of the total output capacity.

From Eqs.(2.3) and (2.7) it can be obtained that $n_1 = 1$, $d_1 = 1$; $n_2 = n_3 = 2$, and $d_2 = d_3 = 3$ for the ideal dual-rate grouping architecture. For the simplified dual-rate grouping architecture, $n_1 = d_1 = 1$, $n_2 = n_3 = 6$ and $d_2 = d_3 = 8$ due to the three-bit counter. The corresponding simulation results regarding the instantaneous bandwidth allocation to flow S_1 and S_2 are shown in Figures 2.4 and 2.5, respectively. It should be noted that in Experiment 1, since S_2 and S_3 are equivalent, only the performance of S_2 is observed. In the ideal per-flow based scheduler, when all flows are backlogged (for example in time interval (5s, 10s)), the normalized bandwidths received by each flow are $\bar{r}_1 = 0.25$ and $\bar{r}_2 = 0.375$ respectively, because the normalized guaranteed service rates of the three flows are $\hat{r}_1 = 0.25$ and $\hat{r}_2 = \hat{r}_3 = 0.375$ respectively.

With the one-rate grouping architecture, as can be seen in Figures 2.4 and 2.5, the received bandwidths of S_1 and S_2 are $\bar{r}_1 = 0.20$ and $\bar{r}_2 = 0.40$, respectively, because the guaranteed rates of the flows are $\hat{r}_1 = 0.25$ and $\hat{r}_2 = \hat{r}_3 = 0.50$. This result demonstrates that one-rate grouping architecture degrades the fairness of bandwidth allocation that can be achieved by the scheduling algorithm. It can also be noted that

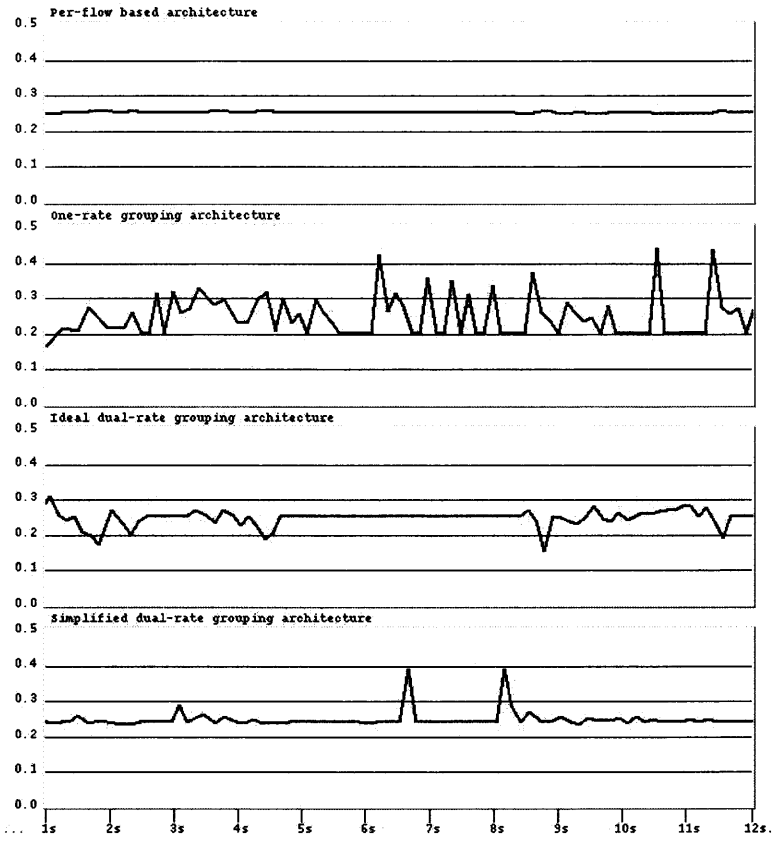

Figure 2.4 Bandwidth allocation of S_1 in Experiment 1.

since S_2 is over-provisioned with the bandwidth and S_1 is under-provisioned, the queue of S_2 is emptied frequently. S_1 can receive more bandwidth when the queue of S_2 is emptied. Therefore, the received bandwidth of each flow oscillates significantly in the one-rate grouping architecture.

In the ideal dual-rate grouping architecture, bandwidths received by each flow approximate the per-flow based scheduler, since S_2 and S_3 are placed into two different service groups alternately (normalized rate groups of 0.25 and 0.5, respectively), and the guaranteed service rate they receive is the required 0.375. From these results, it can be seen that not only the flows get their fair share of bandwidth, but also the oscillations in bandwidth allocation have been reduced.

In the simplified dual-rate grouping architecture, since S_1 gets the guaranteed rate of $\hat{r}_1 = 0.25$ and S_2 and S_3 each get the guaranteed rate of $\hat{r}_2 = \hat{r}_3 = 0.4$ according to Eq.

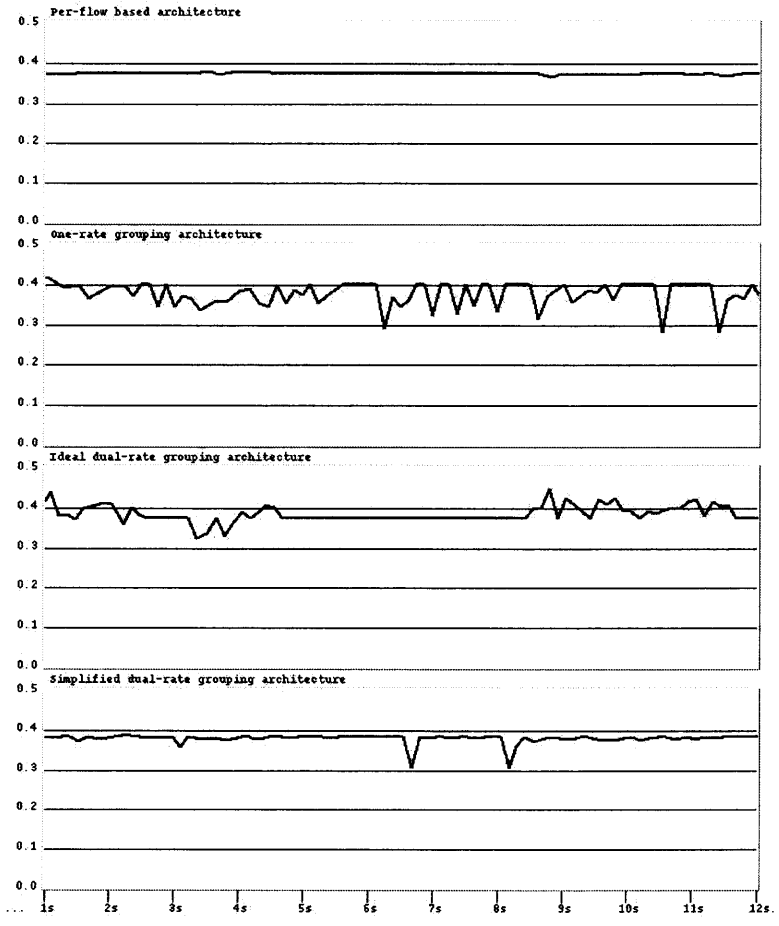

Figure 2.5 Bandwidth allocation of S_2 in Experiment 1.

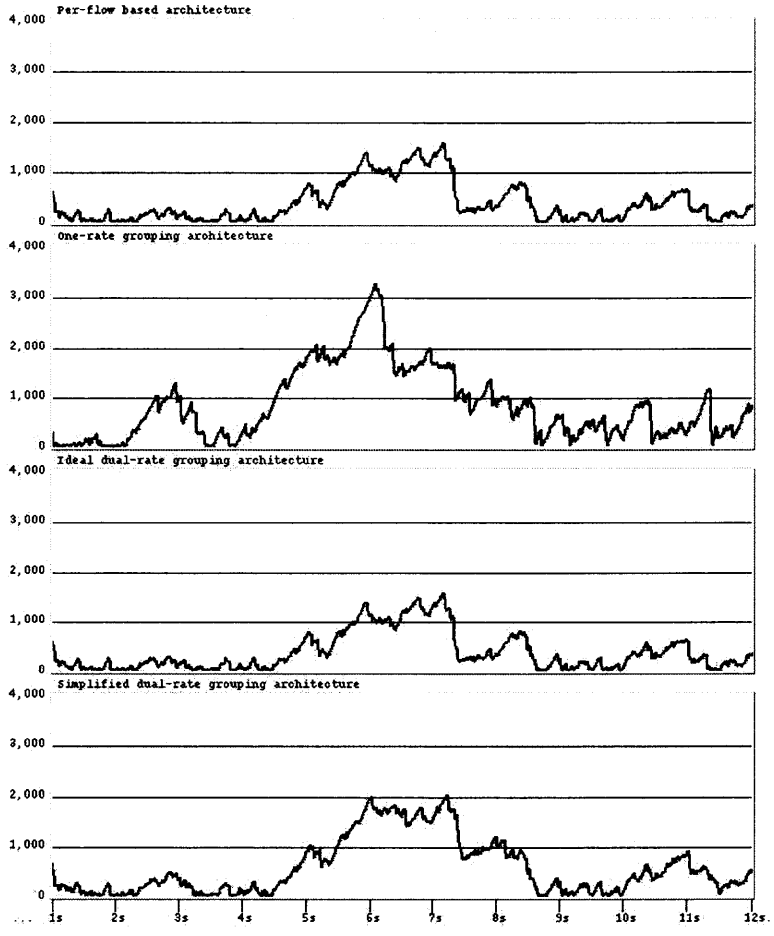

Figure 2.6 Queueing delay of S_1 in Experiment 1.

(2.5), from the WF^{2}Q+ algorithm, theoretically $\bar{r}_1 = 0.238$, and $\bar{r}_2 = \bar{r}_3 = 0.381$, which are confirmed in the simulation. Therefore, the simplified dual-rate grouping architecture can also provide better fairness in the bandwidth allocation to each flow than the one-rate grouping architecture.

As shown in Figures 2.6 and 2.7, the dual-rate grouping architectures also approximate the per-flow based WF^{2}Q+ better than the one-rate grouping architecture in terms of delay. Especially for the ideal dual-rate grouping architecture, the curves of delay of S_1 and S_2 are identical with those of the per-flow based architecture. This is attributed to the fact that the ideal dual-rate grouping scheme can provide finer rate granularity and allocate more accurate bandwidth to all flows than the one-rate grouping

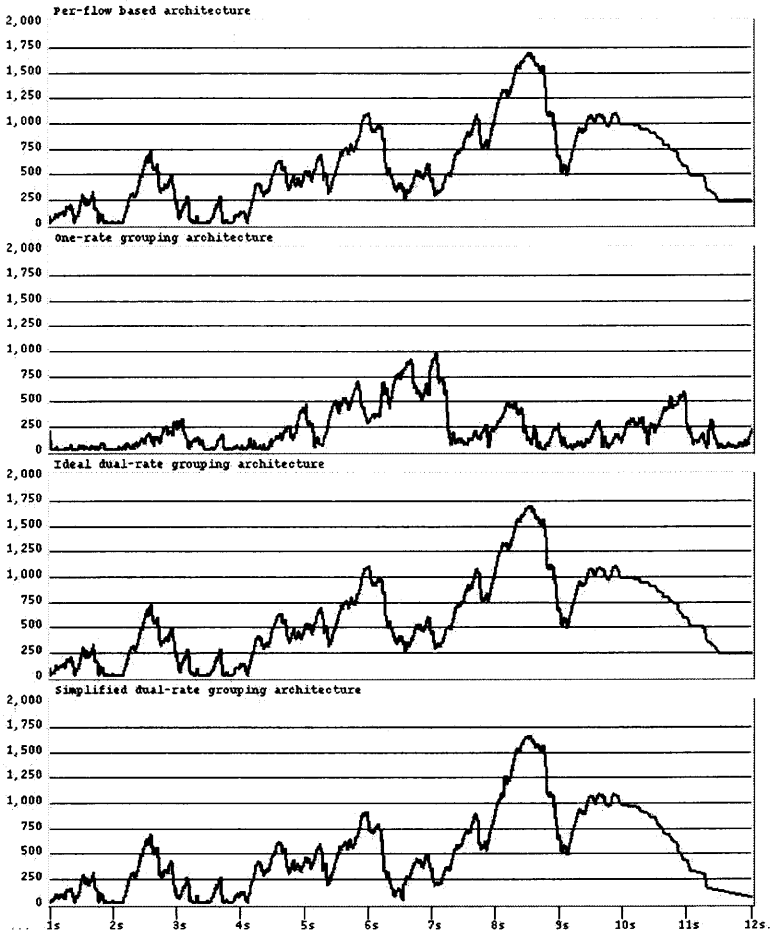

Figure 2.7 Queueing delay of S_2 in Experiment 1.

architecture. Although there is rate allocation error caused by the representation of d_i in the simplified dual-rate grouping architecture, the delay curves of the simplified dual-rate grouping architecture are also close to the ideal per-flow based architecture. Note that, in Figure 2.6, the delay of S_1 with the one-rate grouping architecture is larger than that of the per-flow based architecture, because S_1 in the one-rate grouping architecture receives less bandwidth than it requires and is guaranteed in the ideal per-flow based architecture; while in Figure 2.7 the delay of S_2 with the one-rate grouping architecture is smaller than that of the per-flow based scheduler, because S_2 receives more bandwidth than it should receive when it is backlogged. In all these figures, the unit of delay is time slot.

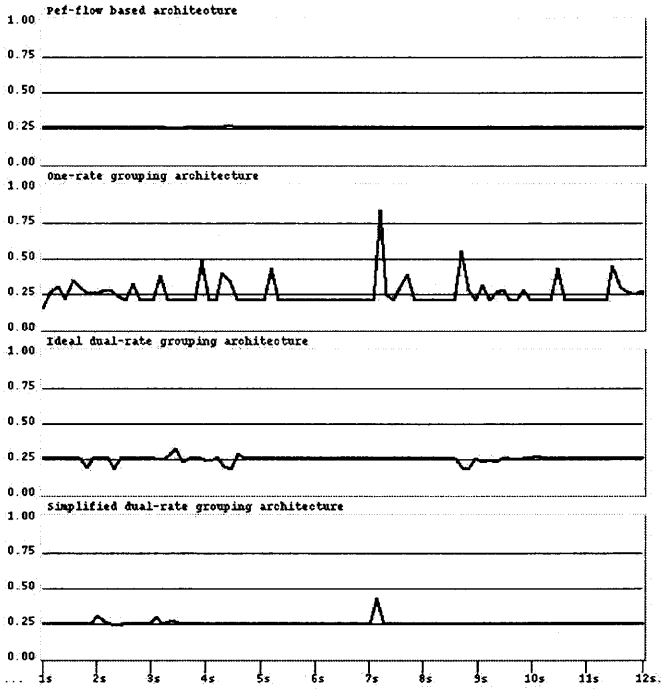

Figure 2.8 Bandwidth allocation of S_1 in Experiment 2.

Experiment 2

In reality, per-flow based leaky-bucket shaping may not be implemented in high-speed routers due to the implementation complexity. However, the ideal per-flow based WF^{2}Q+ scheduler can isolate those ill-behaving flows from negatively affecting the conforming flows. This feature is referred to as the immunity capability of the scheduler. In Experiment 2, the immunity capability of the WF^{2}Q+ algorithm are demonstrated with different implementation architectures. In this experiment, it is assumed that S_2 is ill-behaving with arrival rate 4000 cells/second, although its reserved service rate is only 3000 cells/second. Traffic characteristics of flows 1 and 3 remain the same as in Experiment 1.

As shown in Figures 2.8 and 2.9, with per-flow based architecture, the received bandwidths for S_1 and S_2 are 0.25 and 0.375, respectively, implying that this scheme possesses the capability to protect well-behaving flows from the adverse impact of ill-behaving flows. With the one-rate grouping architecture, the guaranteed bandwidth

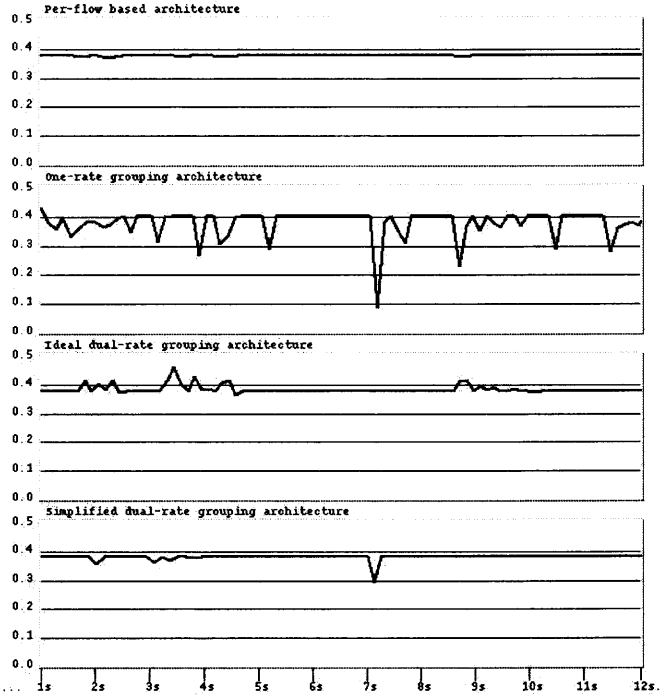

Figure 2.9 Bandwidth allocation of S_2 in Experiment 2.

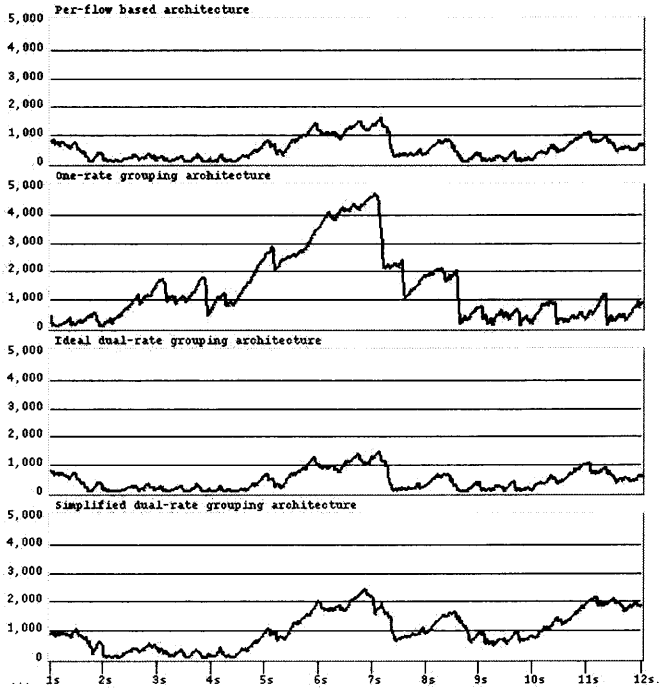

Figure 2.10 Queueing delay of S_1 in Experiment 2.

of well-behaving S_1 is adversely affected by the ill-behaving S_2, which oscillates more dramatically as compared to Experiment 1 and receives less bandwidth (0.2) than it should (0.25), due to the reason that the ill-behaving S_2 receives more bandwidth (0.4) than it should (0.375), as shown in Figure 2.9. As a result, the delay of S_1 in the one-rate grouping architecture also becomes longer as compared to the ideal per-flow based architecture, as shown in Figure 2.10.

On the other hand, the received bandwidth and delay of S_1 approximate those of the per-flow based architecture very well in the ideal and simplified dual-rate grouping architectures. Moreover, with the ideal dual-rate grouping architecture, S_2 receives the bandwidth of 0.375, which is S_2's reserved bandwidth. Therefore, as shown Figure 2.10, in the ideal dual-rate grouping architecture, the delay bound of the conforming flow S_1 can be the same as in the per-flow based architecture.

From experiment 2, it can be seen that the per-flow based WF^{2}Q+ scheme possesses the ability to protect the well-behaving flows from being affected by the ill-behaving flows and has the ability to regulate those ill-behaving flows. Although it can be noted that the other three architectures also have the immunity capability, the immunity capability can be degraded when the one-rate grouping architecture is employed. In this experiment, it is demonstrated that the ideal and simplified dual-rate grouping architectures can alleviate the adverse affects of those ill-behaving flows, and hence significantly improve the immunity capability as compared to the one-rate grouping architecture.

Experiment 3

It should be noted that the more the number of rate groups, i.e., the larger M, the better performance in terms of approximating per-flow based scheduler. The reason is that, with more rate groups, more service rates can be supported, and thus finer rate granularity can be achieved, while the oscillation in bandwidth provisioning is reduced. It should be noted that it is possible to use only two rate groups, one with the maximum service rate and the other with the minimum service rate, to achieve any required service rate by pumping cells

into these two rate groups alternately. This is an extreme case of the dual-rate grouping architecture, with $M = 2$. The performance of such a scheduler and that of a scheduler with more service rate groups are compared in the following experiment 3.

The parameters used throughout experiment 3 are as follows: R=8000 cells/s, $r_1 =$ 2000cells/s, $r_2 = r_3$=3000 cells/s. Consider the following three schedulers. Scheduler 1: per-flow based WF^{2}Q+ scheduler; Scheduler 2: scheduler with the ideal dual-rate grouping architecture in which the minimum service rate is 7.8125 cells/s, the maximum service rate is 8000 cells/s, and there are totally 11 service rate groups with $\gamma = 2$. S_1 always stays in group 9 (the service rate is 2000 cells/s), and flow 2 and 3 are switched between group 9 and 10 to achieve the average service rate of 3000 cells/s; Scheduler 3: scheduler with the ideal dual-rate grouping architecture in which there are only two rate groups, i.e., the minimum rate 7.8125 cells/s and the maximum rate 8000 cells/s.

As shown in Figures 2.11 and 2.12, the bandwidth guaranteed to S_1 and S_2 oscillate much more dramatically by using scheduler 3 as compared to using the scheduler 1 and 2. The reason is that when flows are put into the maximum rate group, their bandwidths are provisioned much more than they require, and when they are put into the minimum rate group, their bandwidths are provisioned much less than they require. As a result, the delay of cells in scheduler 3 changes dramatically, and the maximum delay of each flow becomes larger as compared to that in the scheduler 1 and 2, which are shown in Figure 2.13 and Figure 2.14, respectively. Therefore, more rate groups, which can provide finer rate granularity without the help of the dual-rate grouping, can also provide better performance in the dual-rate grouping architecture.

2.5 Conclusion

In this chapter, a dual-rate grouping scheme was proposed to improve the service rate granularity of the cell-based scheduler with grouping architecture. Its principle and performance properties were discussed, which indicates that any service rate that a user requires can be guaranteed by using the dual-rate grouping strategy, while the complexity

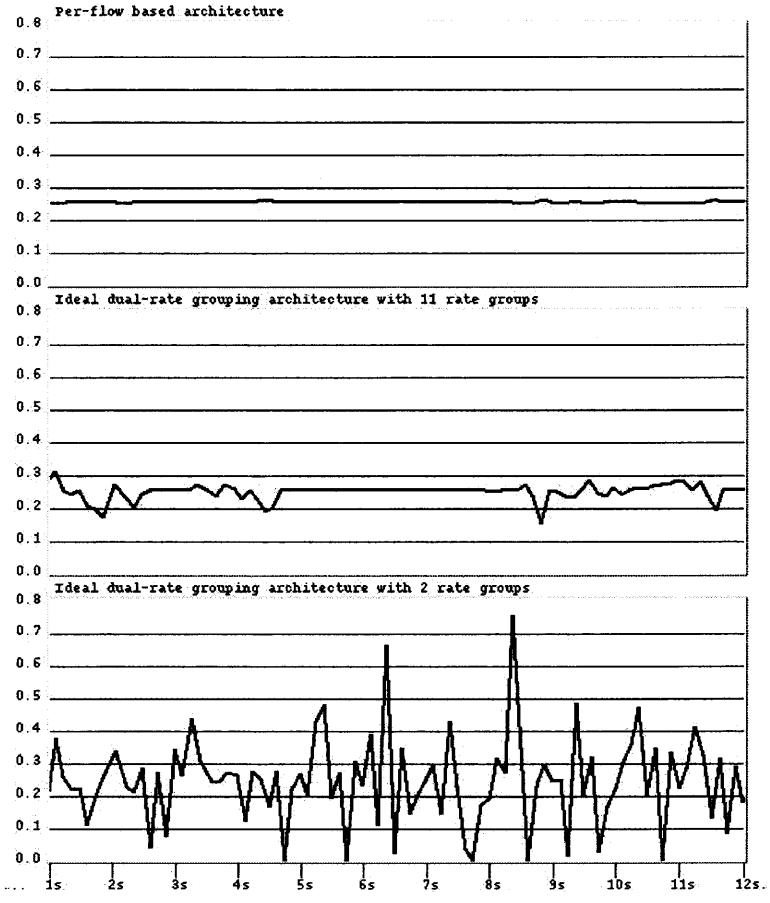

Figure 2.11 Bandwidth allocation of S_1 in Experiment 3.

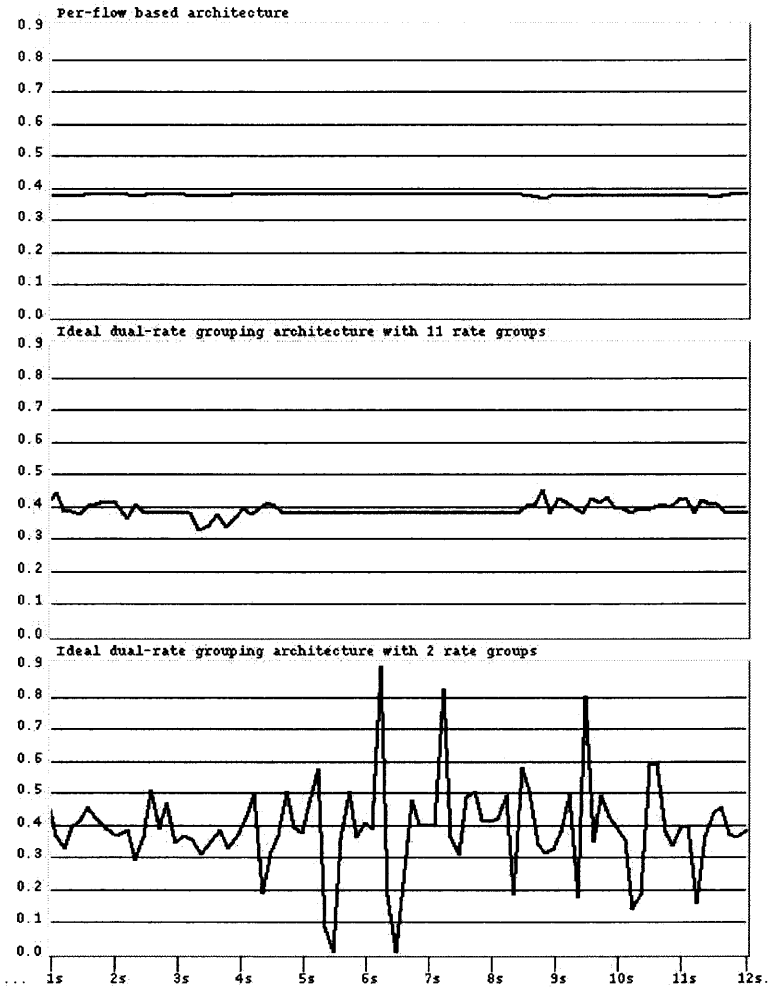

Figure 2.12 Bandwidth allocation of S_2 in Experiment 3.

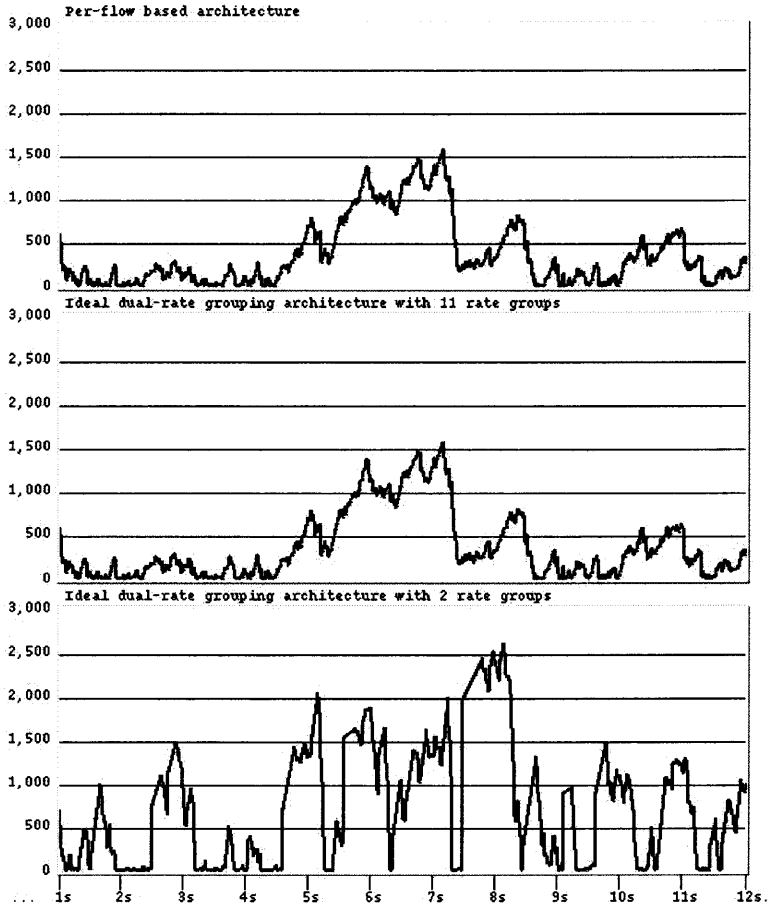

Figure 2.13 Queueing delay of S_1 in Experiment 3.

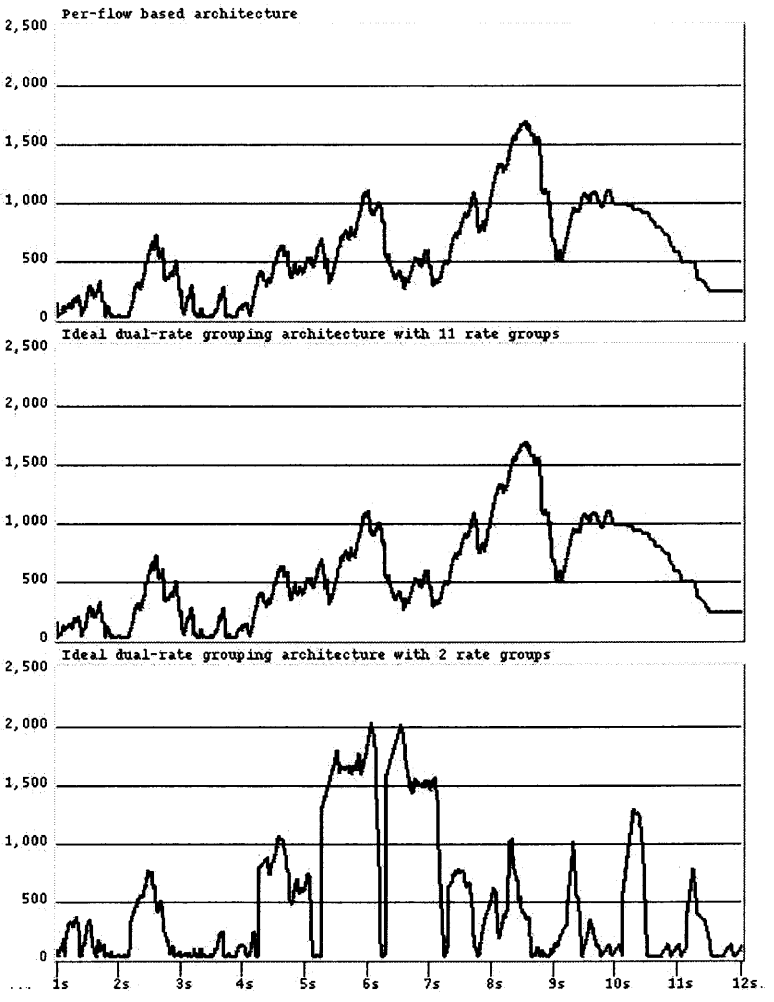

Figure 2.14 Queueing delay of S_2 in Experiment 3.

of the applied PFQ algorithms and the architecture of the scheduler are not affected by the scheme. The implementation issues of the architecture are presented as well. The relationship of the length of the backlogged flow with the guaranteed service rate is studied through a numerical example, while the fairness feature of the dual-rate grouping architecture is presented through simulation. Furthermore, the immunity capability of the different architectures was studied and compared, which demonstrated that the one-rate grouping architecture may degrade the immunity capability that can be achieved by the per-flow based scheduling algorithm due to its coarse rate granularity, while the proposed dual-rate grouping architecture can better approximate the performance of the per-flow based architecture by providing finer rate granularity. It can be also seen from the simulation study, that provisioning finer rate granularity through increasing the number of rate groups can not only enhance the performance of one-rate grouping architecture, but also enhance the performance of the dual-rate grouping architecture, therefore better approximating the performance of the ideal per-flow based scheduling algorithm. These performance evaluation studies demonstrated that the proposed dual-rate grouping architecture can provide finer rate granularity and approximate the PFQs better than the original one-rate grouping architecture in terms of bandwidth allocation, delay, and immunity capability. It should also be noted that one of the most important advantages of the proposed scheme is that its implementation complexity remains the same as the one-rate grouping architecture.

The proposed dual-rate grouping architecture can be used as one of the single-node QoS provisioning mechanisms to enhance the rate granularity provisioned by a router. In the following chapters, the strategies to enhance the QoS granularity in an end-to-end fashion are discussed, which can be applied in today's Diffserv networks.

M	the number of rate groups supported by a scheduler
l	the length of a cell
L_i	the bucket size (in terms of number of cells) of S_i
$r^{(K)}$	the guaranteed rate of the rate group K
S_i	flow i
$S_i^{(K)}$	the sub-flow of S_i entering rate group K
$x_i^{(K)}$	the ratio of cells in $S_i^{(K)}$ to the entire flow S_i
n_i	the numerator of $x_i^{(K+1)}$
d_i	the denominator of $x_i^{(K+1)}$
c_i	the counter to implement d_i
s_i	virtual start time of S_i
f_i	virtual finish time of S_i
f_i^k	virtual finish time of cell k, S_i
r_i	the service rate that S_i requires
r_i^k	the guaranteed service rate of cell k in S_i
$\bar{r}_i$	the service rate that S_i actually receives
$\hat{r}_i$	the service rate that is guaranteed to S_i
e_i	the relative error between the service rates that are guaranteed to S_i and S_i requires
D_i	the average delay of S_i

Table 2.1 Notations of Chapter 2

CHAPTER 3

THE FRAMEWORK OF EXPLICIT ENDPOINT ADMISSION CONTROL AND SERVICE VECTOR

3.1 Introduction

In today's Internet, the Internet Protocol (IP) becomes the vehicle for delivering various types of data flows and services, including current and emerging real-time and multimedia applications, such as voice, image and video streams. However, the current IP infrastructure lacks the support of Quality of Service (QoS), as required by many real-time and multimedia applications, thus significantly hampering its further development.

It is well known that there are two fundamental frameworks proposed to provide end-to-end QoS in the Internet: Intserv [4] and Diffserv model [5]. The Intserv model, which aims to provide "hard" end-to-end QoS guarantees [21] to each individual data flow, i.e., to provide fine QoS granularity to data flows, requires per-flow based resource allocation and service provisioning, and thus suffers from the scalability problem due to the huge amount of data flows that may coexist in today's high-speed core routers. The proposed Diffserv model simplifies the design of core routers by aggregating individual flows at edge routers and provisioning only a number of services to the aggregated data flows at each core router. However, in this model, it is difficult to identify each individual flow's QoS requirements at core routers, and to contrive efficient resource allocation mechanisms to guarantee the end-to-end QoS of each individual data flow.

Currently some Internet equipment vendors statically map the data flows to a specific service class according to their QoS requirements, and make resource reservation in Expedited Forwarding (EF) service for flows with "hard" QoS requirements [3]. Thus, although the Diffserv model is a scalable service provisioning model as compared to the Intserv model, it can only provide a coarse QoS granularity, in which flows only have limited options in choosing the service class along the data path, resulting in similar end-to-end delay, loss probability, etc., even though they may actually have quite different

QoS requirements. For a Diffserv network with n services at each router, the QoS granularity provided in the network will be $O(n)$ while by static service mapping the granularity that a data flow can actually obtain is $O(1)$. The coarse QoS granularity provided in such a QoS mapping and provisioning paradigm affects not only the network resource utilization but the user benefit from the network service as well, especially when service-based pricing schemes are introduced into the network service provisioning. It can be seen that the main challenge in the end-to-end QoS provisioning in the Internet is to find a scheme that balances the implementation complexity in the network and the service granularity to the individual flows.

3.2 Motivations and Objectives

In this chapter of the dissertation, the concept of decoupling the end-to-end QoS provisioning from the service provisioning at each core router is introduced to enhance the QoS granularity in the Diffserv networks, which enables a flow to choose different service classes and their associated Per-Hop Behaviors (PHBs) at different routers. Thus, the definition of services and their PHBs may remain the same at each router as in the conventional Diffserv model, while if the path contains m intermediate routers with n available services at each hop, the granularity of the end-to-end QoS provisioned to a data flow by the network can be as fine as $O(n^m)$. However, there are still some issues in the implementation of such an approach which are associated with the place and methodology of determining the most appropriate services for each data flow. In the current service model, there might be three choices available: the edge routers, the core routers along the data path, and the bandwidth brokers. Since each core or edge router may only have local information on the resource utilization of each service, it is difficult for them to evaluate the end-to-end performance of a data flow and choose the services in an end-to-end fashion. On the other hand, if the corresponding services are determined by one or several bandwidth brokers in the network, the computational overhead of the optimization

procedure in finding the services for each flow is prohibitive due to the possible huge amount of requests.

Recently a new approach of admission control, referred to as *distributed admission control* or *endpoint admission control* (EAC), has been studied in the literature ([22],[23], etc.), which provides a possible solution to detecting the resource availability of different services and determining the service that the flow will use at a router. In EAC, the end host sends out probing packets and evaluates the end-to-end performance that a flow may experience. If the probing result indicates that the current congestion level is low and the QoS provided by the Diffserv network can be accepted, the host will decide to use the service; otherwise it gives up the connection request. Since it is the host that actually makes the admission decision, the routers do not need to make admission control decision for each individual flow. Thus, the scalability feature of the Diffserv model can be maintained in EAC schemes. Moreover, before the flow is admitted, the end-to-end performance has been evaluated, thus the end user can be assured that the flow's end-to-end QoS requirements can be guaranteed in the Diffserv network. It should be noted that in the following the terms of "user", "end user", "end host" and "host" will be used interchangeably, and the terms of "router" and "node" will be used interchangeably.

However, the EAC schemes assume that the data flow will use the same service along the data path, and therefore, they cannot be applied conveniently to indicate the performance of each service at each router along the data path, but only the end-to-end performance of a specific service. In this chapter, the EAC concept is further developed and a novel EAC scheme, referred to as Explicit Endpoint Admission Control (EEAC), is proposed, with the introduction of a new concept, namely the *service vector*. The EEAC and service vector scheme allows a flow to detect and use different services at different routers along its data path, and provides an ideal vehicle to support the new Diffserv model paradigm of decoupling the end-to-end QoS provisioning from the service provisioning at each router in order to enhance the end-to-end QoS granularity in the Diffserv network.

In this chapter, the framework of the EEAC and service vector scheme is introduced and the user benefit that can be achieved by the EEAC and service vector scheme is described; while in the next chapter the implementation details and the performance properties of the proposed scheme in the Diffserv network, and the benefit that can be achieved by the network service provider will be discussed. In Table 3.1, a list of most of the notations used throughout this chapter is provided.

m	the number of routers that a data flow passes through from the source to the destination
n	the number of services provided at each router
S	the set of available service classes at each router
S_j	service class j in S
s	the service vector of a flow
s_i	the service that a flow chooses at router i
R	the end-to-end performance that a flow may get
$U(R)$	the utility function of a flow
C	the user cost function
C_i	the user cost function at router i
p	the pricing policy set by the service provider
G	the net benefit that can be achieved by a user
A_i	the revenue function of router i

Table 3.1 Notations of Chapter 3

The rest of this chapter is organized as follows. In section 3.3 some endpoint admission control schemes proposed in the literature are reviewed, by summarizing their features and revealing the design issues associated with them. In section 3.4, the framework of the EEAC scheme and the concept of the service vector are introduced and described, while section 3.5 contains the performance evaluation and comparison

of different EAC schemes including the proposed EEAC scheme. Finally, the chapter concludes in section 3.6.

3.3 Related Work

In general, in order to implement and offer some degree of QoS, admission control mechanisms have to be implemented in the Diffserv networks as well. Currently, there are basically two admission control schemes for Diffserv networks: static Service Level Agreement (SLA) which may not be efficient; and dynamic admission control, which however may not be scalable, because per-flow requests have to be processed. Moreover, neither of these admission control schemes can enhance the QoS granularity provided in a Diffserv network.

EAC schemes have been extensively studied in the literature to address the scalability issue of the dynamic admission control in the Diffserv networks. Generally, there are two parties involved in EAC with the Diffserv network: user side and network side. The basic functionality of the network side is to provide differentiated services to users. The end user will carry out the task of admission control with or without the help of the Diffserv network. There are two phases of Endpoint Admission Control: probing phase and data transfer phase. In the probing phase, the host would map the QoS requirements to a network service and start a probing process to obtain information about its performance. Then the host determines whether or not to admit the flow into the network. If the flow is admitted, the data transfer phase starts. Since the core node does not need to make admission control decision for each individual flow, the scalability feature of the Diffserv model can be maintained in EAC schemes.

At the probing phase there are two design options for an EAC scheme: the network can be unaware of the probing process or can cooperate with the probing process. Different design options of the probing schemes without the network awareness in the literature were discussed in [22], in which probing packet trains are sent to the network to detect the network service performance. The advantage of using probing packets without awareness

of the network is its deployment simplicity. However, probing by real traffic can only obtain a realization of the random process that characterizes the QoS performance along the data path. The test models for probing packet trains or probing packet pair [23] [24] demonstrate that long probing packet train may be needed to achieve accurate probing results regarding the end-to-end QoS performance, especially in terms of packet loss probability [22]. Therefore, the probing process without the cooperation of the network may result in incorrect admission decision, or introduce a long admission delay and increase the network traffic overhead [22], which may even make endpoint admission control impractical [25].

To reduce the probing overhead and enhance the probing accuracy, the performance measurement can be conducted by each router and conveyed to the end hosts by marking or dropping the probing packet [26]. The end host will start sending data only when the probing packet is neither dropped nor marked. Since the measurement is performed on the aggregated traffic, the result should be more accurate compared to the measurement obtained through the individual probing flows [27], and it is not necessary to send a large sequence of probing packets to the network. Although classified as an EAC scheme, it should be noted that in this type of EAC schemes, the admission decision is actually made by the router through the marking or dropping of probing packets, rather than by the end hosts. Therefore, the admission decision may not be customized according to the user requirements. All these schemes assume that the data flow will use the same service along the data path, and therefore, they cannot be applied conveniently to indicate the performance of each service at each router along the data path, but only the end-to-end performance of a specific service.

Current EAC schemes also bring up the problem of resource abuse. Some users with loose QoS requirements may abuse network resources by choosing the services that are supposed to provide stringent QoS, and therefore users that really require stringent QoS bounds may not get satisfactory QoS. The resource abuse problem may be partially solved

by the integration of pricing and admission control. Actually provisioning of different services with different pricing is one of the objectives for the deployment of Diffserv model [5]. In the literature, user benefit optimization models that introduce pricing schemes into the network service provisioning have been studied to enhance the user benefits from the network services [21], [28]. For applications with adaptive QoS requirements, i.e., applications that can adjust their QoS requirements according to the network congestion, pricing schemes can be used to provide incentives to users to adjust their data rate so that network service congestion can be avoided and users can maximize their benefits [21]. The results presented in [21] demonstrate that changing service price according to network congestion enhances both the network performance and the perceived user benefit compared to the flat-rate pricing.

It is proven in [28] that when throughput (which can be represented by flow data rates and packet dropping probability) is used as the QoS parameter, priority-based differentiated service provisioning can reach max-min fairness among flows in the network. However, to reach the user benefit maximization for each of the K flows in the network, the network may need to provide as many as K priorities, i.e., the service granularity may be as fine as $O(K)$. In the current Diffserv network architecture, the service granularity is $O(n)$, where n is the number of service classes in the network. Usually, $n << K$ and the user benefit maximization may not be achieved. On the other hand, in the proposed EEAC and service vector scheme, different service vectors may lead to different end-to-end service priorities and performance in the network. As mentioned before, if a data flow passes through m routers, the end-to-end service granularity can be as fine as $O(n^m)$. Thus, it is easier and more feasible to achieve the user benefit maximization. Moreover, since adjusting the data rate sent to the network as adopted in [21] can enhance the user benefits, it can also be incorporated into the proposed scheme and the user benefit maximization model.

3.4 Framework of the Explicit Endpoint Admission Control and Service Vector Paradigm

In this section, the use of Explicit Endpoint Admission Control (EEAC) is introduced and described, in which the QoS performance measured at each node is explicitly attached to the probing packets, so that the state of the network can be conveyed to the end hosts explicitly from the network routers. End hosts can therefore use the information attached in the probing packets to numerically evaluate the end-to-end performance and adopt the proper strategies in utilizing the network services. Thus, the network provider may convey its preference to the user by attaching QoS and pricing information and the user may maximize his benefit by using the information provided by the network.

3.4.1 The Architecture and Operation of EEAC

The proposed EEAC architecture also consists of two parties: network side and user side; and contains two stages: the probing and admission control stage, and the data forwarding stage. During the probing and admission control stage, at the user side, a host sends an explicit request message (i.e., a network probing packet) along the path to the destination host and waits for an acknowledgement packet back. When an acknowledgement packet is received within a pre-defined timeout interval, the end-to-end performance can be evaluated according to the information contained in the acknowledgement packet. The probing packet and acknowledgement packet may get lost during the transmission along the path. The destination may also drop the source request by not responding to the probing. In either case, at the source host, a timeout will occur. If the end-to-end performance is not satisfactory, or a timeout occurs, the source host may give up the connection effort, or retry after an exponential back off. At the network side, the router along the path attaches the QoS related information of each service class to the probing packet.

In the data forwarding stage of EEAC, the use of a new service model paradigm is proposed, which is implemented through a service vector selected during the probing phase according to the information in the probing packet, and attached to each data packet by the end hosts. Intermediate routers will use the service vector information to provide corresponding service to the data packets. Suppose a flow is going from its source to the destination via m intermediate routers. At each router, the set of available service classes is $S = (S_0, S_1, ..., S_{n-1})$. A flow may choose service s_i at router i ($s_i \in (S_0, S_1, ..., S_{n-1})$) which may be different from service s_j it chooses at router j. So a service vector $s = (s_0, s_1, ..., s_{m-1})$ is defined to represent the services that are chosen along the path from the source to the destination. Although the granularity of QoS provided at each router does not increase, hence, the scalability of the Diffserv network is maintained, the end-to-end QoS granularity that a user may get can be as fine as $O(n^m)$. A successful EEAC procedure is shown in Figure 3.1.

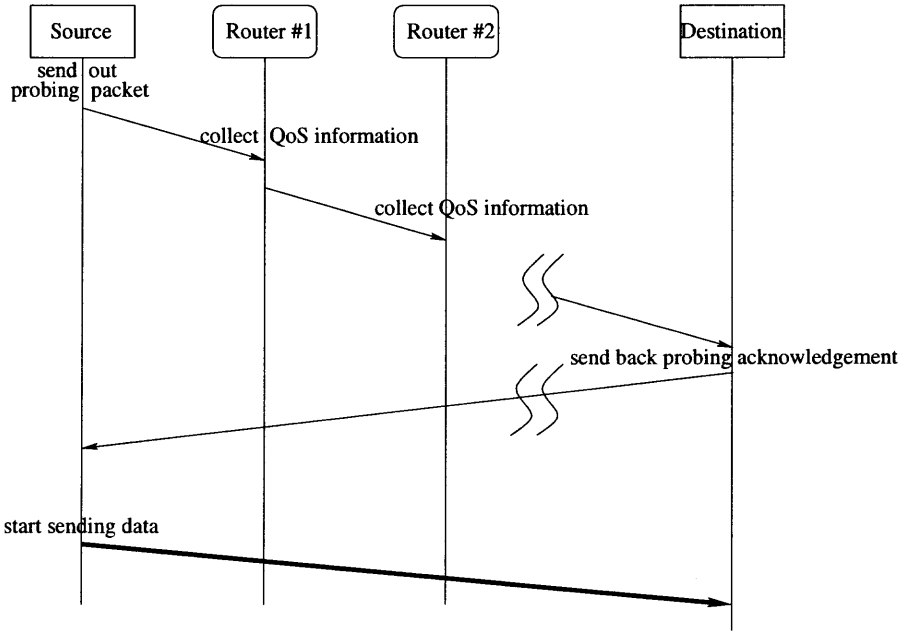

Figure 3.1 EEAC procedure in a Diffserv network.

3.4.2 Optimization Models of The User and Network Sides

In the following, the mathematical models are discussed, which can be used in the proposed service model paradigm to optimize the network and user benefits respectively. The models are based on the following assumptions, which are realistic for a large-scale, large-capacity Diffserv network:

Assumption 1: The performance of a certain service class in a router may not fluctuate significantly in time scale of seconds and minutes. Therefore, the end-to-end performance of a flow during the probing phase and after the probing phase does not change considerably.

Assumption 2: Compared to the aggregated traffic in core routers, a certain traffic flow has small impact on the congestion level of a core router. Therefore, even though a long sequence of probing packets are not sent at the rate of the data flow to the network, the router can still estimate the performance that the future flow will get.

In the following, assume that a user data flow may get an end-to-end QoS performance represented by a vector $R = (R_0, R_1, ..., R_{K-1})$, which leads to its utility represented by a utility function $U(R)$. A utility function describes users' level of satisfaction with the perceived QoS; the higher the utility, the more satisfied the users. It characterizes how sensitive users are to the changes in QoS [29]. To achieve a certain utility, the user pays for a cost represented by a user cost function C, which is determined by a set of pricing policies $p = (p_0, p_1, ..., p_{l-1})$ and the service vector s the flow chooses. Thus the end host behavior in selecting the appropriate network services can be modeled and represented by the following optimization:

$$G = \max_{s}(U(R) - C(s,p)) \tag{3.1}$$

$$s.t.\ R(s) \in \Re \tag{3.2}$$

$$and\ U(R) - C(s,p) \geq \delta \tag{3.3}$$

where $\Re$ is the vector space that meets the end-to-end QoS requirements of the data flow, s is the service vector that a flow may choose and δ is the minimum net benefit that the user can accept. Note that R is a function of service vector s and can be further decomposed into the performance that the flow experiences at each router, which can be represented by $(R^0, R^1, ..., R^{m-1})^T$, where $R^j = (R_0^j, R_1^j, ..., R_{K-1}^j)^T$ $(j = 0, ..., m-1)$ is the performance vector that the flow gets at router j. Since the service set that each router can provide is S, R^j is a function of s_j, where $s_j \in S$.

The user optimization model can be applied at the probing and admission control stage in EEAC for the end hosts to determine the service vector that the data flow will use. If an optimized service vector s is found, the EEAC procedure proceeds to the data forwarding stage. According to *Assumption 1*, the route during the probing and data forwarding stages is assumed to remain the same, and each vector can thus be read by the proper router. If s cannot be found, the end host may give up the request and the flow would not be admitted into the network. Thus, the interaction among users is modeled as a non-cooperative game since each user is trying to optimize its own benefits.

At the network side, each router i maximizes its own revenue function $A_i(N^i, p)$ under the constraint that it provides service differentiation, where N^i is the number of flows at router i, and p is its pricing policies. The maximization at the network side can be formulated as follows:

$$\max A_i(N^i, p) \tag{3.4}$$

$$s.t.\ R^i(S_j, N_j) \in \Re^i(S_j),\ j = 0, 1, ..., n-1 \tag{3.5}$$

$$and\ N^i = \sum_{j=0}^{n-1} N_j \tag{3.6}$$

where $R^i(S_j, N_j)$ is the performance vector of service S_j when there are N_j users of it, and $\Re^i(S_j)$ is the vector space of service S_j that meets the service differentiation requirements

at router i. It should be noted that N^i is determined by the user optimization model (3.1)-(3.3), which are associated with the service vector s and the pricing policy vector p.

3.4.3 Discussions on the User Optimization Model

It can be noted that the current end-to-end service provisioning mechanisms in Diffserv networks could be easily included in the optimization model (3.1)-(3.3). Therefore, in the following consider three types of service vectors in solving the relations (3.1)-(3.3):

Type 1: Static Service Mapping. This type of solution corresponds to the scheme in which a data flow is statically mapped to a certain class of service. Therefore, s is a constant vector with $s_i = s_j$, $(i, j = 0, 1, ...m-1)$. The solution is simplified to determine whether relations (3.2) and (3.3) can be satisfied. There is no maximization process of the relation (3.1) and the resulting G, denoted by G_{type1}, is a determined value when (3.2) and (3.3) are satisfied. The end-to-end QoS granularity is $O(1)$.

Type 2: Dynamic Service Mapping. In this case although $s_i = s_j$, $(i, j = 0, 1, ...m-1)$ along the path, the flow is not statically mapped to a service class but dynamically mapped to an available "best service" determined by the solution to relations (3.1)-(3.3). Since there are n classes of service, the solution can be simplified to finding the maximum value of (3.1) among the n possible solutions with the constraint that (3.2) and (3.3) are satisfied. The end-to-end QoS granularity determined by this type of solution is $O(n)$.

Type 3: Combination of Service Classes via the service vector. In this solution, the service vector is used to choose different services at different nodes. Relations (3.1)-(3.3) are considered jointly to find the best available service vector. Apparently, this is an optimization problem which is generally NP-hard. However in most practical cases it is only needed to find some feasible sub-optimal solutions. Throughout this study, without loss of generality in the applicability of the proposed method, Genetic Algorithms (GA) are used to find the corresponding type 3 service vector. GAs have already been used successfully in the literature to solve various network resource allocation and optimization problems (e.g., [30]). Although the granularity of QoS provided at each router does not

increase, and hence the scalability of the Diffserv network is maintained, the end-to-end QoS granularity that a user may get can be as fine as $O(n^m)$ in this type of solution.

It can be seen that the type 3 solution can provide the finest end-to-end QoS granularity, while the type 1 solution can provide the most coarse QoS granularity. It should also be noted that the solution space of type 1 is a subset of the solution space of type 2, which in turn is a subset of the solution space of type 3. Therefore, when all three types of solutions can be found under certain network state, one can have $G_{type1} \leq G_{type2} \leq G_{type3}$. Furthermore it should be noted that a feasible type 3 solution may exist and be found, even if a feasible type 1 and/or type 2 solution cannot be found. Overall the type 3 solution may result in the highest network utilization and type 1 solution results in the lowest network utilization.

3.5 Performance Evaluation of Different Solutions

In order to demonstrate the performance improvements that can be achieved by EEAC scheme with the service vector of type 3 , along with its efficiency and robustness, as compared to the solutions of type 1 and 2, in the following, an experiment is presented to obtain the corresponding numerical results by applying different endpoint admission control schemes with the three types of solutions. The experiment is built in a Diffserv networking environment and the non-adaptive applications are studied, which usually have non-elastic utility functions (i.e. if an upper bound of the QoS requirement is guaranteed, the QoS is considered satisfactory and the utility is a constant value. Otherwise, the utility is zero. The application does not care if better QoS than the required bound is provided. The interested reader may refer to [29] for a more detailed discussion regarding the utility functions). For simplicity but without loss of generality, the QoS performance considered in the example is the average end-to-end delay. Thus, the non-elastic utility function is as shown in Figure 3.2. Assume that the pricing policy at each router is to charge each packet of the flow according to its assigned service class. Then at the user side the optimization

can be simplified as follows:

$$\min_{s} \sum_{i=0}^{m-1} C_i(s_i) \tag{3.7}$$

$$s.t. \sum_{i=0}^{m-1} delay(s_i) \leq Delay \tag{3.8}$$

where *Delay* represents the user's end-to-end QoS requirement in terms of average end-to-end delay and $delay(s_i)$ represents the average delay at router i that the flow may experience if the flow uses service s_i. At the network side, if pricing is only associated with the service class, relations (3.4)-(3.6) will lead to the conclusion that the router should accommodate as many flows as possible at each service class, with the constraint that the service differentiation is guaranteed.

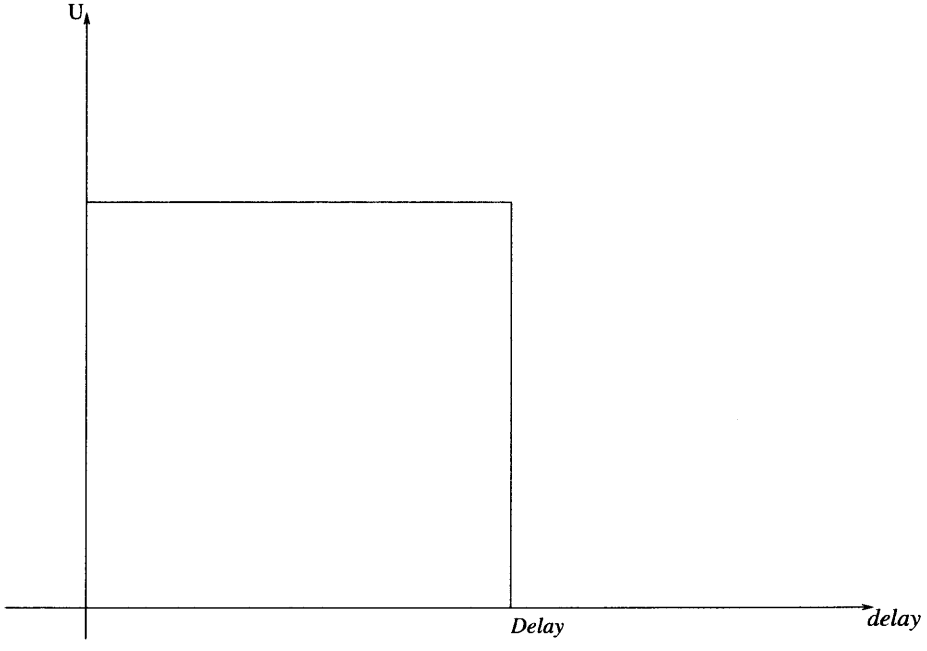

Figure 3.2 An example of the non-elastic utility function.

Moreover, each router has to attach its performance measurement to the probing packet. There are many factors that could make the performance information inaccurate for the QoS evaluation by the hosts. For example, the current probing processes may

bring new flows into the network and current flows may leave the network as well, which will cause the fluctuation of traffic load in the network; while traffic in the system may have different burst that also affects the measurement results. These factors often have a complicated correlation and in general modeling these factors could be a very complex process. In the following experiment the implementation details of the performance measurement will not be discussed, but simply a predictor based on the Weiner process [31] is used to perform measurement and estimation of the performance of each service class in a router. This is motivated by the fact that Weiner process (also known as Brownian motion) describes a random process that is affected by a large number of independent or weakly dependent factors, each with a relatively small impact, which matches well our assumption 1 and 2 for a Diffserv network.

3.5.1 Network Topology and Assumptions

The network topology under consideration is depicted in Figure 3.3. In this experiment, the performance of the non-adaptive real time VBR streams from node A to node E is studied, each of which is modeled by an on-off model with average rate of 12.8 kb/s and peak rate 25.6 kb/s. The length of each packet is 500 bytes. The average on and off period is 0.1s respectively. Suppose the end-to-end delay requirement of each flow is 500ms, which is non-elastic. The objective of this experiment is to demonstrate the performance difference between different types of solutions in a Diffserv network when the traffic load at each router is different.

In the network, three services are provided: class 0, class 1 and class 2, which are implemented through rate-limited priority scheduling [22]. The class 0, class 1 and class 2 services can occupy up to 25%, 25% and 50% of the output link bandwidth respectively. The Random Early Detection (RED) scheme is implemented for class 1 service in case of network congestion. It can be noted that the class 0 service can be considered as an implementation of Expedited Forwarding (EF) service, the class 1 service can be considered as an implementation of Assured Forwarding service with only one

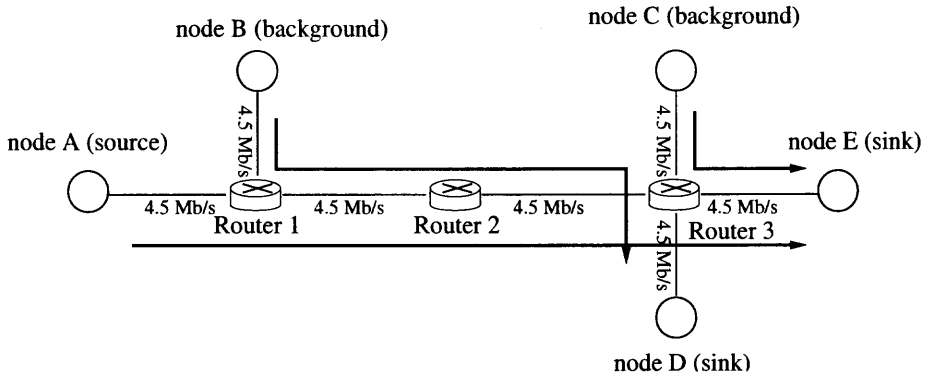

Figure 3.3 The simulated network topology.

Source	Destination	Class 0 bandwidth	Class 1 bandwidth	Class 2 bandwidth
Subnet B	Subnet D	15%	5 %	20%
Subnet C	Subnet E	5%	15 %	30%

Table 3.2 The summary of bandwidth occupied by the cross traffic

drop precedence, which does not harm the general results and conclusions [21], and the class 2 service can be considered as an implementation of the best effort service. At each router, each user is charged for every packet sending out, and the price of each packet is differentiated according to the service the packet requires. The unit price of the packet requiring the class 0 service is normalized as 1 unit and the unit prices of the class 1 and 2 are assumed to be 0.5 and 0.4 respectively. As shown in Figure 3.3, the traffic distribution in this network is as follows. Traffic from node A only goes to node E while cross traffic from node B only goes to D and traffic from C only goes to E. In the network the class 0 traffic from node B to D consumes 15% of the output bandwidth of node B, while the class 1 traffic consumes 5% of the output bandwidth and the class 2 traffic consumes 20% of the bandwidth. The class 0 traffic from node C to E needs 5% of the output bandwidth of node C; the class 1 needs 15%, and the class 2 needs 30%. The cross traffic is summarized in Table 3.2 and it makes the load of corresponding services at each node unbalanced.

3.5.2 Simulation Results and Discussion

The following results are obtained by modeling and simulation performed by OPNET. The received performance of the flows from node A to E is obtained and evaluated under the simulation scenario described above for three EAC schemes:

Scheme 1: Conventional EAC Scheme (EAC-CS). In this scheme, users map their QoS statically to a certain class of service before the probing process, i.e., the type 1 solution is obtained through an EAC scheme in the literature to provide individual flows the end-to-end QoS. The flows from node A to E always use service class 0 since real-time streams are assumed for these flows. Before entering the network, the end user (host) sends out a sequence of probing packets at average data rate and measures the QoS performances at the destination. The probing packets use the same service as the data packets, i.e. class 0 service. If the measurement results can meet the QoS requirements, the end user starts using the network service; otherwise it gives up the request.

Scheme 2: EEAC with Single Class of service scheme (EEAC-SC). In this case EEAC is applied to obtain the type 2 solution.

Scheme 3: EEAC with Combination of Service Classes scheme (EEAC-CSC). In this scheme, EEAC is applied to obtain the type 3 solution and the service vector is used to represent different services at different nodes.

Figure 3.4 compares the average user cost per received packet at node E from node A as a function of the requested load under different EAC schemes. The requested load is defined as the ratio of the aggregated bandwidth requirement from the source node to its output link rate, i.e., 4.5 Mbps. The requested load varies while the load of cross traffic remains unchanged in the simulation. For the EAC-CS scheme, since there is only static price for each service class, a user has no choice but pay for 3 units (1 unit at each router) for each data packet it sends to the network. The EEAC schemes attempt to find type 2 and 3 solutions. Therefore, they can lower the user cost of a packet. Since EEAC-SC obtains the type 2 solution, which is a sub-optimal solution of the objective function, it results in

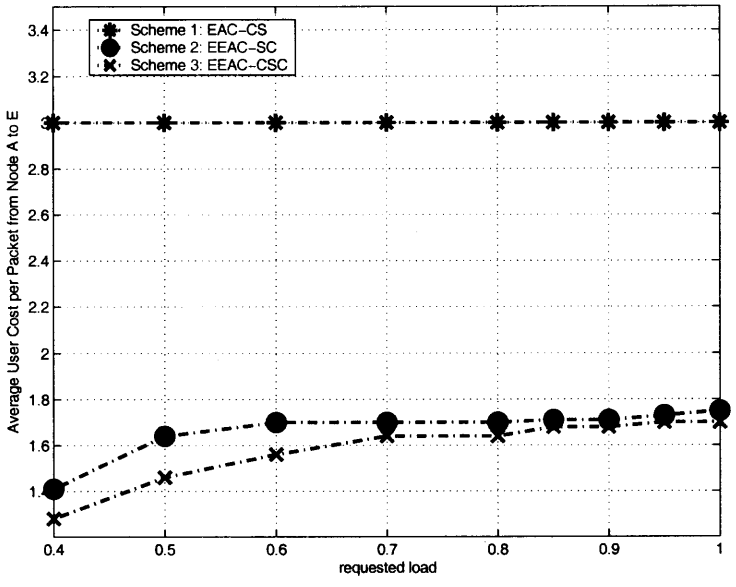

Figure 3.4 The average user cost per packet with different requested load.

higher user cost than the EEAC-CSC scheme. This demonstrates that the type 3 solution can increase the user net benefits.

In Figure 3.5 the request dropping probability from node A to E for the three different schemes is presented. The request dropping probability is defined as the ratio of the number of probing results that do not meet the end-to-end QoS requirements to the total number of probing requests. It is assumed that once the probing results indicate that the end-to-end QoS requirements cannot be met, the end hosts will drop the request. As demonstrated in this figure, type 2 and 3 solutions achieve lower request dropping probability as compared to the type 1 solution. This is due to the fact that in schemes EEAC-SC (scheme 2) and EEAC-CSC (scheme 3) when one class of service is congested along a path, the flow can choose other non-congested services while in the EAC-CS scheme (scheme 1) the request will be aborted. The EEAC-CSC scheme has lower request dropping probability than the EEAC-SC scheme because the load of service classes is unbalanced along the path, and the EEAC-CSC scheme provides more service options and

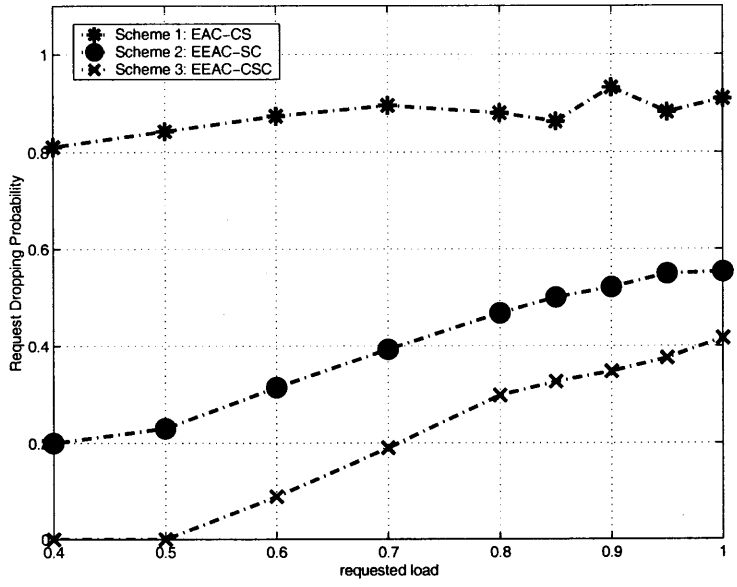

Figure 3.5 The request dropping probability with different requested load.

larger solution space to help users achieve the desired end-to-end QoS. It should be noted that the lower request dropping probability also implies higher network utilization.

Figure 3.6 demonstrates that all the three schemes can satisfy the non-elastic end-to-end delay requirement. The EEAC-CSC scheme results in longer end-to-end delay than the EEAC-SC scheme because the EEAC-CSC scheme attempts to utilize every possible combination of service classes and makes every service class more loaded than in the EEAC-SC scheme. Therefore, in each service class, a packet may experience longer delay in the EEAC-CSC scheme. For the same reason, the delay in both EEAC-CSC and EEAC-SC schemes is longer than that of the EAC-CS scheme. In the EAC-CS scheme, when the class 0 service becomes congested as the requested load increases, the average end-to-end delay of the data packets is always around 170ms, determined by the service rate and buffer length of the class 0 service at each router. This implies that the end-to-end delay bound of class 0 service is not associated with individual user's end-to-end QoS requirements, but only with the router's congestion level and service policy. On the other hand, in the EEAC-CSC and EEAC-SC schemes, although the average end-to-end

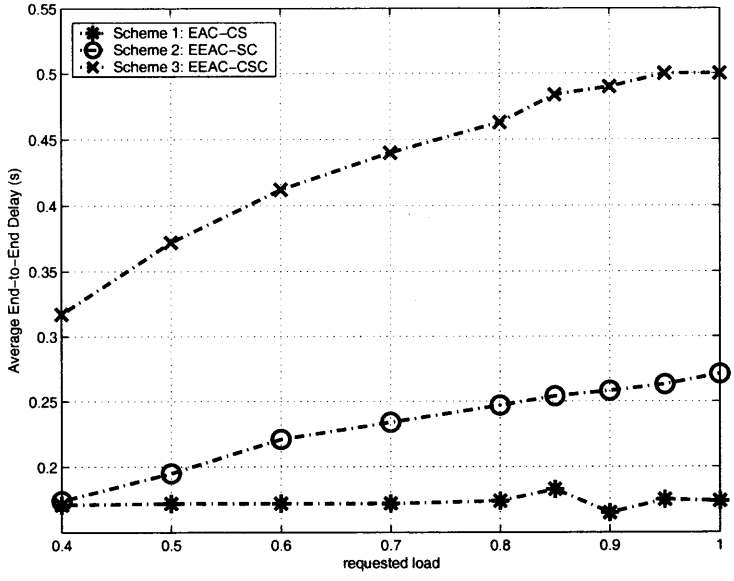

Figure 3.6 The average end-to-end delay with different requested load.

delay and the request dropping probability are increasing with the requested load, the performance requirements are met with the actual delay values close to the end-to-end delay requirement, which implies that the network provides "customized" service to the user while maintaining high resource utilization. Therefore Figure 3.6 demonstrates that in the conventional differentiated service model, only coarse QoS granularity can be provided while the proposed EEAC and service vector schemes can provide finer QoS granularity.

The delay performance of EAC-CS also reveals the effect of the inaccuracy of probing results [22] in the conventional EAC schemes, which becomes even more significant when the requested load approaches to 1, resulting in the increase of the delay at load of 0.85 and decrease of the delay at load 0.9. The corresponding effect is also noted in Figure 3.5 where the request dropping probability of the EAC-CS scheme fluctuates accordingly.

3.6 Conclusion

In this chapter, a new service paradigm was proposed, which decouples the end-to-end QoS provisioning from the service provisioning at each core router, via the framework of Explicit Endpoint Admission Control (EEAC) and the use of service vector. The objective of this new service paradigm is to enhance the QoS granularity in the Diffserv networks. Unlike conventional EAC schemes, the proposed EEAC scheme requires the explicit support of the endpoint admission control from the network, which should provide corresponding performance information to the end hosts. By performing EEAC and utilizing the service vectors during the data transfer, the end hosts can obtain fine QoS granularity and thus enhance their net benefits from the network service, while the scalability of service provisioning in the Diffserv networks can be maintained. It is demonstrated through the discussion and simulation results presented in this chapter, that the solution of the user optimization model implemented through the framework of EEAC and service vector, can provide users the finest end-to-end QoS granularity and users can achieve the most net benefit from the network services as compared to other schemes. In the next chapter, the implementation details of the EEAC and service vector scheme in the Diffserv networks, are discussed. Furthermore, it is demonstrated that the service providers can accommodate more users and enhance their resource utilization by supporting the proposed service paradigm via the EEAC and service vector scheme.

CHAPTER 4

IMPLEMENTATION OF EEAC AND SERVICE VECTOR IN DIFFSERV NETWORKS

4.1 Introduction

Among current Internet QoS models, the Intserv model suffers from the scalability problem due to the huge amount of data flows that may coexist in today's high-speed core routers, while the Diffserv model is difficult to identify each individual flow's QoS requirements at core routers, and to contrive efficient resource allocation mechanisms to guarantee the end-to-end QoS of each individual data flow. Various alternatives have been proposed in order to exploit the benefits of both the Intserv and Diffserv while avoiding their drawbacks. As argued before, the main challenge here is to find the balance between the implementation simplicity in the network and the service guarantee of the individual flows.

In the previous chapter, the concept of decoupling the end-to-end QoS provisioning from the service provisioning at each router was introduced to enhance the QoS granularity. The proposed scheme enables a flow to choose different service classes and their associated Per-Hop Behaviors (PHBs) at different routers. Thus, the scalability feature of the core routers can remain the same as in the Diffserv networks, while if the path contains m intermediate routers with n available services at each hop, the granularity of the end-to-end QoS provisioned to a data flow by the network can be as fine as $O(n^m)$. It was also demonstrated that users can achieve higher net benefits from the network services by adopting such an approach as compared to other solutions. The framework of the Explicit Endpoint Admission Control (EEAC) and service vector scheme was proposed, which provides an ideal vehicle for the support and implementation of the proposed QoS provisioning paradigm.

However, how the EEAC and the service vector scheme can be incorporated (referred to as the EEAC with service vector scheme, EEAC-SV) and deployed in the

current Diffserv network architecture, and how the routers can efficiently estimate and report their service performance in the probing packet remain challenging research issues.

In this chapter, the service provisioning architecture and algorithm are further develop-ed and examined at both the user and network sides, to implement the EEAC-SV scheme, so that the proposed idea of decoupling the end-to-end QoS provisioning from the service provisioning at each router can be efficiently deployed based on the current Diffserv network paradigm. It is demonstrated that by decoupling the end-to-end QoS provisioning from the service provisioning at each router, not only can the user achieve more benefits from the network service, but also the network service provider may can enhance their benefits in terms of higher network resource utilization as compared to various service provisioning mechanisms in the Diffserv networks. In Table 4.1, a list of most of the notations used throughout this chapter is provided.

The rest of the chapter is organized as follows. In section 4.2, some related work in the literature is discussed, while in section 4.3 the architecture for decoupling the end-to-end QoS provisioning from the service provisioning at each router is studied and analyzed. The advantages of the proposed approach over different service mapping and provisioning techniques in Diffserv networks are discussed in section 4.4, while the performance of the proposed scheme is evaluated in section 4.5. The concluding remarks are provided in section 4.6.

4.2 Related Work

In [32], the operation of Intserv over Diffserv model was introduced. In this model, the admission control and resource allocation procedures are adopted from those in the Intserv model so that sufficient resources can be reserved to satisfy the data flows' QoS requirements, while the data flows are served in the network domain in a Diffserv fashion, i.e., data flows are aggregated and provided only with a limited number of services. This approach increases the implementation complexity of Diffserv networks by requiring per-flow based admission control and resource allocation (either at the routers or at some

intelligent agents such as "bandwidth brokers") to guarantee each individual flow's QoS requirements, while maintaining the simplicity of Diffserv service provisioning (i.e., traffic marking, buffering, shaping, and scheduling). However, this service paradigm cannot enhance the QoS granularity provisioned in the Diffserv network infrastructure; therefore, how to map each individual flow's end-to-end QoS requirements from the Intserv model to the Diffserv model still remains a challenging issue.

The ECN [33] scheme allows the router to mark the data packets to indicate the network congestion. However, this scheme assumes that the data flow will use the same service along the data path, and therefore, it cannot be applied conveniently to indicate the performance of each service at each router along the data path, but only the end-to-end performance of a specific service. Although the similar approach of packet marking is followed, in the proposed service provisioning paradigm (i.e., the EEAC-SV scheme), each router can only mark its "own" bits in the probing packet based on the measurement performed by the router, by which the performance of each service at each router can be represented in the probing packets.

4.3 The End-to-End Service Provisioning Architecture

4.3.1 Implementation of EEAC-SV via RSVP

The EEAC-SV operation procedure includes two stages at the user side (the end hosts): the probing and admission control stage, and the data forwarding stage. Detailed descriptions of the operations of the EEAC-SV procedure can be found in the previous chapter. Although the routers in the EEAC scheme are assumed to be capable of providing an "explicit report" of the service performance and availability, thus increasing the complexity of the core router design due to the fact that the core routers may need to support per-flow request (the probing packets), the increased complexity is similar to that of the Intserv operations over the Diffserv network which may require the core routers supporting RSVP PATH and RESV messages [32]. In fact, the EEAC-SV scheme can be

implemented by extending the RSVP such that the RSVP PATH message serves as the probing packet and the RESV message serves as the acknowledgement packet. However, in the EEAC-SV scheme, it is not necessary for the core routers to keep the state of the RSVP PATH and RESV pair as in the conventional RSVP model, thus simplifying the core router design and operations as compared to the Intserv operations over Diffserv networks.

As explained before, suppose a flow is going from its source to the destination via m intermediate routers. At each router, the set of services that can be used by the flow is $S = (S_0, S_1, ..., S_{n-1})$. A flow may choose service s_i at router i ($s_i \in S$) which may be different from service s_j it chooses at router j. Thus, a service vector is defined as $s = (s_0, s_1, ...s_{m-1})$. If a service vector s is found by the end hosts, the EEAC-SV procedure proceeds to the data forwarding stage and each service vector may be represented by a group of labels attached to the data packets. Like RSVP, probing should be performed periodically in the EEAC-SV scheme, so that the flow can adjust its service vector to the dynamics of the network, while the route during the probing and data forwarding stages is assumed to remain the same, and each vector can thus be read by the proper router.

4.3.2 The Probing Packet Marking Scheme

Since each router in the ideal EEAC model is required to report the performance of each of its service classes to the probing packet so that the end hosts can obtain the service vector that maximizes their benefits, there are two issues associated with the implementation of the reporting scheme at each router: 1) how to evaluate and predict the availability of each service class, so that the incoming data flow will receive the performance as described in the probing packets, and 2) how to represent the performance of each service class in the probing packet.

Service Provisioning in Routers

There are many parameters that can be used to represent a flow's QoS requirements including bandwidth, delay bound, and loss probability, etc. Since in general the capacity

of the core network is much larger than the data rate of an individual data flow, i.e., the bandwidth requirement of a data flow is negligible as compared to the bandwidth of the core networks, the end-to-end delay bound and packet loss probability are adopted as the descriptors for describing a flow's end-to-end QoS requirements. Correspondingly, the QoS parameters used to describe the service provisioning characteristics at each router are local delay bound and packet loss probability.

In the Diffserv model, the individual flows are classified and aggregated at the edge routers while packets are queued and serviced according to their classes at the core routers. Packets with the same class at the router will be served in a First In First Out (FIFO) manner. Although it may be difficult to guarantee the end-to-end performance for a data flow in a conventional Diffserv network due to the lack of resource reservation, it is reasonable to assume that performance bounds can be provided by each service class at each router. In the literature, Weighted Fair Queueing (WFQ) [7] is recognized as a scheduling policy that can guarantee the delay bound and fair resource allocation of each service class. Without loss of generality, it is assumed that WFQ is used as the scheduling algorithm between the different service classes at each router, although other schedulers are equally applicable.

It is assumed that three classes of service are provisioned in an edge or core router: Expedited Forwarding service (EF), Assured Forwarding service (AF), and Best Effort service (BE), i.e., $S = \{EF, AF, BE\}$. For simplicity, the AF class is only implemented with one dropping precedence in this dissertation , which does not compromise the general conclusions. It is also assumed that probing packets use the EF service. Let us denote by R the output link capacity and the maximum buffer length of each service class is L_{EF}, L_{AF}, and L_{BE}, respectively. Without loss of generality, it is assumed that the buffer length at different routers for the same service class is the same, while the buffer length of different service classes at each router may be different from each other. The corresponding delay bound can be represented by $D(S_j) = \frac{L_{S_j}}{R_{S_j}} + \frac{L}{R}$ where $S_j \in \{EF, AF, BE\}$, L is the

maximum packet length, while R_{S_j} is the guaranteed service rate of S_j [7]. To implement the EF, AF and BE service differentiation, it is guaranteed that $D(EF) < D(AF) < D(BE)$. For the EF and AF services, it is assumed that some random packet dropping schemes such as Core-Stateless Queueing [34] or Random Early Detection (RED) [35] are applied so that the dropping probability at a router has the following feature:

$$P_{S_j}(t) = \begin{cases} 1 & \bar{L}_{S_j}(t) \geq L_{S_j} \\ \leq p_{S_j}^{max} & L_{S_j}^{min} \leq \bar{L}_{S_j}(t) < L_{S_j} \\ 0 & 0 \leq \bar{L}_{S_j}(t) < L_{S_j}^{min} \end{cases} \tag{4.1}$$

where $S_j \in \{EF, AF\}$, $\bar{L}_{S_j}(t)$ is the average length of the queue at time t, $L_{S_j}^{min}$ is the threshold above which the dropping algorithm starts dropping packets at a probability determined by the arrival rate [34] or buffer occupancy [35], and $p_{S_j}^{max}$ is the maximum packet dropping probability of service S_j at a router. For the BE service, the drop-tail is simply used and the packet dropping probability is given by:

$$P_{BE}(t) \approx P[\bar{L}_{BE}(t) > L_{BE}] \tag{4.2}$$

where $\bar{L}_{BE}(t)$ is the average queue length of the BE service at time t. It is also assumed that $p_{EF}^{max} < p_{AF}^{max} < P[\bar{L}_{BE}(t) > L_{BE}]$. The values of $\bar{L}_{S_j}(t)$, $S_j \in \{EF, AF, BE\}$, are updated upon the arrival of a packet of service S_j and computed by using the exponential moving average [35], as follows:

$$\bar{L}_{S_j}(t) = (1 - e^{-\tau_{S_j}(t)/K}) L_{S_j}(t) + e^{-\tau_{S_j}(t)/K} \bar{L}_{S_j,old}(t) \tag{4.3}$$

where $\tau_{S_j}(t)$ is the interval between the current time t and the arrival time of the previous received packet of service S_j, and $\bar{L}_{S_j,old}(t)$ is the most recently updated average queue length before t, while K is a constant as described in [34], and $L_{S_j}(t)$ is the queue occupancy at time t.

The Probing Packet Marking Algorithm

In the EEAC-SV scheme, if a router can accurately predict the performance of its service classes and report them in the probing packet, each flow will receive the guaranteed QoS performance it desires with the help of the service vector. Although the QoS granularity can be greatly enhanced using this approach, the estimation of the performance of each service cannot be accurate, while attaching the estimated performance data to the probing packet increases the implementation complexity at the router. At the same time, the packet size grows with the number of routers that the probing packet traverses to the destination.

To alleviate these drawbacks, first discretize the performance of each service class into several congestion levels. In a probing packet, each router that the packet traverses is allocated several bits to represent the congestion levels of its service classes. A router can indicate its congestion level of a service class by marking these bits. As a result, one can use $\lceil \log_2 Y_{S_j} \rceil$ bits to represent Y_{S_j} congestion levels for service class S_j. For n service classes, one need $(n\lceil \log_2 Y_{S_j} \rceil)$ bits in a probing packet to represent the congestion levels at a router. If the data path consists of m routers, the probing packet needs $(mn\lceil \log_2 Y_{S_j} \rceil)$ bits to represent all the congestion levels along the data path. For example, based on the service provisioning assumptions in this chapter, for three congestion levels at each service class in a router, six bits are required to represent these congestion levels in a probing packet. If the path consists of three routers, the total length of the probing bits will be 18.

The service provider provides users with the performance bounds at each congestion level of a service class in a router as *a priori* knowledge. When the end hosts receive the probing packet with the marked bits to indicate the congestion level of a service class at a router, they can obtain the performance bounds of the service classes at each router that the probing packet traverses. As an example, let $(d_{S_j,i}, p_{S_j,i}), (S_j \in \{EF, AF, BE\}, i = 0, 1, 2)$ represent the delay and packet dropping probability bounds at each congestion level i of service S_j provisioned by a router. The service performance information that can

be used as *a priori* knowledge provided to the end hosts is summarized in Table 4.2. In fact, each of the congestion levels corresponds to a unique buffer occupancy, represented by $B_{S_j,q}, (q = 0, ..., Y_{S_j} - 1)$. For the EF and AF services, $B_{S_j,0} = L_{S_j}^{\min}, B_{S_j,1} = L_{S_j}$, $(S_j \in \{EF, AF\})$, and $B_{S_j,2} = \infty$ is used to indicate that no more QoS-required flows can be admitted into the service class at the router. For the BE service, $B_{BE,0} = L_{BE}^{min}$ indicates that no packet will be dropped at this level, and $B_{BE,1} = B_{BE,2} = \infty$ represents that the packets may be dropped at the tail of the buffer, and no more QoS-required flows should use the BE service at this router. With *a priori* knowledge of the performance bounds of a service class at the end hosts, and the probing packet marking at each router, the complexity of the router operation and the length of probing packets can be significantly reduced.

Let $0 \leq q_1 < q_2 \leq Y_{S_j} - 1$ represent the congestion levels from low to high and $B_{q_1} < B_{q_2}$. At the highest congestion level, $B_{S_j,Y_{S_j}-1} = \infty$ indicates that no more QoS-required flows can be admitted into class S_j. Let us denote by $\bar{r}_{S_j}(t)$ the average packet arrival rate of service S_j at time t. Assuming that the probing period is T for each flow, when a router receives a probing packet from flow k, the average buffer occupancy of service S_j is increased by $(\bar{r}_{S_j}(t) + \hat{r}^{(k)} - R_{S_j})T$ at the time when the next probing packet of flow k arrives, where $\hat{r}^{(k)}$ is the bandwidth requirement of the flow. Thus, during the interval of $[t, t + T)$, the buffer occupancy is estimated to be in the range of $[\bar{L}_{S_j}(t), \bar{L}_{S_j}(t) + (\bar{r}_{S_j}(t) + \hat{r}^{(k)} - R_{S_j})T)$, and the center point is used as the estimation of the average queue occupancy during the interval. Therefore, one can predict $\hat{B}_{S_j}(t)$ as follows:

$$\hat{B}_{S_j}(t) = \begin{cases} B_{S_j,q} & B_{S_j,q-1} \leq \bar{L}_{S_j}(t) + (\bar{r}_{S_j}(t) + \hat{r}^{(k)} - R_{S_j})T/2 < B_{S_j,q}, \\ & (0 < q < Y_{S_j} - 1) \\ B_{S_j,0} & \bar{L}_{S_j}(t) + (\bar{r}_{S_j}(t) + \hat{r}^{(k)} - R_{S_j})T/2 < B_{S_j,0}, (q = 0) \\ \infty & \bar{L}_{S_j}(t) + (\bar{r}_{S_j}(t) + \hat{r}^{(k)} - R_{S_j})T/2 \geq B_{S_j,Y_{S_j}-2}, (q = Y_{S_j} - 1) \end{cases} \tag{4.4}$$

where $\hat{B}_{S_j}(t)$ is the prediction of the congestion level in the future T time units (seconds) and $Y_{S_j} \geq 2$ is the number of congestion levels of service S_j. Then, the router marks its bits in the probing packet to indicate the congestion level of each service $S_j, S_j \in S$ according to $\hat{B}_{S_j}(t)$.

Upon receiving a probing acknowledgement, with *a priori* knowledge of the performance bounds of each service at a router, the end host can evaluate and determine the optimal solution that maximizes its benefit G using Relations (3.1)-(3.3). Note that although each probing packet may contain the bandwidth requirement of the data flow, assuming that a single data flow's impact on the network is negligible and thus $\hat{r}^{(k)}T << B_{S_j,0}$, the term $\hat{r}^{(k)}T$ in Eq. (4.4) can be ignored. As a result, the router does not need the flow information from the probing packet, thus further simplifying the router operation. Similar to how $\bar{L}_{S_j}(t)$ is computed by using Eq.(4.3), $\bar{r}_{S_j}(t)$ is obtained through the exponential moving average [34]:

$$\bar{r}_{S_j}(t) = (1 - e^{-\tau_{S_j}(t)/K})\frac{l_{S_j}(t)}{\tau_{S_j}(t)} + e^{-\tau_{S_j}(t)/K}\bar{r}_{S_j,old}(t) \tag{4.5}$$

where $l_{S_j}(t)$ represents the received packet length at time t and $\bar{r}_{S_j,old}(t)$ is the most recently updated average rate before t. At each router, both $\bar{L}_{S_j}$ and $\bar{r}_{S_j}$ are updated when a data packet of service class S_j is received. When a probing packet is received, all of the three $\bar{L}_{S_j}$ and $\bar{r}_{S_j}$ ($S_j \in \{EF, AF, BE\}$) are updated, in which $l_{S_j}(t)$ in Eq.(4.5) ($S_j \in \{AF, BE\}$) will be 0 and $l_{EF}(t)$ is the probing packet length, since it is assumed that the probing packet uses the EF service. The pseudo-codes for the arrival processes of a data packet and a probing packet are shown in Figure 4.1 and Figure 4.2, respectively.

Since $\bar{L}_{S_j}(t)$ and $\bar{r}_{S_j}(t)$ are based on the measurements performed by the router, they may reflect the performance of the service class better than measuring and modeling the performance of the probing packet train, which is only a small sample of the stochastic service process, while at the same time this approach significantly reduces the bandwidth overhead of probing as well.

```
On receiving packet p
if ((data packet) or (probing acknowledgement))
//we assume probing acknowledgement packets will use EF service and be serviced as a
// regular data packet of the EF service.
    i=classify (p);
    interval=crt_time-old_time[i];
    rate[i]=update_rate (p.length, interval,  rate[i]);
    //use Eq.(7)to update the arrival rate of the packets of service i ;
    old_time[i]=crt_time;
    //keep  time for next update of service i;
    drop_prob=calculate_drop(average_queue_length[i],rate[i], i );
    //use Eq. (3) to calculate the dropping probability of EF or AF service.
    //use drop-tail to drop packet of BE service
    if (drop_prob>uniform_distrib(0,1))
      packet_drop(p);
    else
      enqueue(i,p);
      queue_length[i]+=p.length;

  average_queue_length[i]=update_queue(queue_length[i],
                                       average_queue_length[i], interval );
  //use Eq.(5) to update the average queue length after dropping or queuing the packet.
```

Figure 4.1 The pseudo-code for the process of the data packet or probing acknowledgement packet arrival in a router.

4.4 Properties and Advantages of the Service Provisioning Architecture

The main features and advantages of decoupling the end-to-end QoS provisioning from the service provisioning at each router, implemented via the proposed distributed end-to-end QoS provisioning architecture and the EEAC-SV scheme, reside in the following two aspects: 1) it may provide the users with the flexibility to choose their most desirable QoS provisioned in the network; 2) it distributes the computational overhead between the end hosts and routers, thus maintaining the simplicity of the router and network architecture. Note that the static service mapping for a data flow is a subset solution of the service vector scheme in the user benefit maximization model (3.1)-(3.3). Therefore, from the user viewpoint, using the EEAC-SV scheme can enhance users' benefits.

In principle, it is possible that the available bandwidth and buffers can be shared among different classes and can be dynamically allocated to a service class according

```
On receiving packet p
if (probing packet)
 //probing packet will use the EF service and maybe dropped due to congestion.
    for (i∈{EF,AF,BE})
          interval=crt_time-old_time[i];
         if (i==EF)  //determine whether the probing packet will be dropped.
            rate[i]=update_rate (p.length, interval,  rate[i]);
            //use Eq.(7)to update the arrival rate of the packets of service i ;
             //since probing packets use EF service, it uses p.length to update rate.
            old_time[i]=crt_time;
            //keep time value for next update of EF service.
             drop_prob=calculate_drop(average_queue_length[i],rate[i], i );
              //use Eq. (3) to calculate the dropping probability of EF service.
               if (drop_prob>uniform_distrib(0,1))
                    packet_drop(p);
                else
                    queue_length[i]+=p.length;
        else
            rate[i]=update_rate (0,  interval,  rate[i]);
            //for services other than EF service, the received packet length is 0,
            //queue_length[i] is unchanged.
            // And since no data packet of service i is received, old_time[i] is not updated.
        average_queue_length[i]=update_queue(queue_length[i],
                                             average_queue_length[i], interval );
         //use Eq.(5) to update the average queue length after dropping or queuing the
         // packet;
        congestion_pattern[i]=estimate_congestion(average_queue_length[i], p.rate,
                                  rate[i], i);
         //use Eq.(6) estimate the congestion level
 if (packet_not_dropped (p)==TRUE)
    marking_probing_field (p, congestion_pattern);
    enqueue(EF, p);
    // We assume probing packets will use EF service.
```

Figure 4.2 The pseudo-code for the process of the probing packet arrival in a router.

to the users' demands. Although it is ideal to have the dynamic allocation schemes to share network resources among different classes at each node, the allocation algorithms are rather complex and there is no consensus on how the allocation should be performed due to the various flow characteristics and QoS requirements, as well as different policies the service providers may have.

In the trivial case, when the network load is light as compared to the bandwidth which is over-provisioned by the network service provider, the dynamic resource allocation schemes and the proposed decoupling mechanism may be equivalent from the network perspective. For example, if the AF service is idle and the EF service is congested, while the total traffic load to the router is light, allocating the idle buffer and bandwidth of

the AF service to the EF service is the same as reassigning the flows from the EF service to the AF service from the perspective of network resource utilization, if the reassigned flows will not congest the AF service queue (e.g., below congestion level 0). However, when both services have a large number of data flows to serve, the effects are not equivalent. Flows entering AF queues may encounter higher loss probability and delay than those entering the EF queues. Thus, if some buffers and bandwidth are dynamically allocated to the EF service from the AF service, these buffers and bandwidths may accommodate fewer flows than in the scheme in which flows use the service vector to choose the AF service at the router.

In the following, the trivial case, in which the traffic load to a router is light as compared to its capacity, is ignored. It can be demonstrated that the proposed service decoupling scheme (i.e., the EEAC-SV scheme) can reduce the request dropping probability and enhance the resource utilization in the network using the notion of "effective bandwidth" ([36],[37], etc.). The request dropping probability is defined as the probability that a flow will not use the network service due to the unsatisfactory end-to-end QoS.

Let the delay bound and packet loss probability of service S_j when the service queue is in the stable condition be $(d_{S_j,\max}, p_{S_j,\max})$. Thus, the effective bandwidth that flow k may consume at node i is $\hat{r}^k_{i,S_j}(\theta_i)$, where θ_i is a function of $(d_{S_j,\max}, p_{S_j,\max})$ and $\hat{r}^k_{i,S_j}(\theta_i)$ has the following properties [37]:

Property 1: $\hat{r}^k_{i,S_j}(\theta_i)$ is an increasing function of θ_i.

Property 2: θ_i is a decreasing function of $d_{S_j,\max}$ and $p_{S_j,\max}$.

Property 3: If the number of flows using service S_j at node i is K_{S_j}, the service can provide bounded delay and loss probability of $(d_{S_j,\max}, p_{S_j,\max})$, $S_j \in S$, and the service queue is in the stable condition, if and only if

$$\sum_{k=1}^{K_{S_j}} \hat{r}^k_{i,S_j}(\theta_i) \leq R_{S_j} \tag{4.6}$$

where R_{S_j} is the guaranteed service rate of class S_j, and $\sum_{S_j \in S} R_{S_j} \leq R$.

Without loss of generality, assume that service classes in the service set $S = (S_0, S_1, ..., S_{n-1})$ at each node can be sorted according to one or several of the performance metrics. In this chapter, assume that the services in S can be sorted, such that $k < l$ for service S_k and service S_l implies that $d_{S_k,r} \leq d_{S_l,q}$ and $p_{S_k,r} \leq p_{S_l,q}$, where $1 \leq r < Y_{S_k} - 1$ and $1 \leq q < Y_{S_l} - 1$ are the congestion levels of service class S_k and S_l, respectively. That is, service S_k can always provide better quality in terms of delay bound and packet loss probability than service S_l, when both of their queue occupancies are above the congestion level 0, under the assumption that these service queues are in the stable condition. Such a sorting can be represented by $S_k > S_l$ (i.e., service S_k has a higher priority than service S_l). Also assume that each flow of type k has a default service $S_D^k \in S$, which is the only service that is acquired and used along the data path by the flow in the static service mapping scheme, while in the EEAC-SV scheme the service vector $s = (s_0, ..., s_{m-1})$ must satisfy $s_i \leq S_D^k, (i = 0, 1, ..., m-1)$. Note that this condition imposes more constraints in finding the optimal service vector as compared to Relations (3.1)-(3.3). However, since the trivial case is ignored that some service is nearly idle (for example, at congestion level 0), according to the proposed service provisioning scheme, the added condition does not affect the service vector search results for the flows whose default service is the EF service, which is the case for most of today's real-time applications using the Intserv operations over Diffserv networks and requiring quantitative end-to-end QoS guarantees. Then, the following propositions can be proven.

Proposition 1 Assume a flow of type k traverses a single link, and the flow's end-to-end QoS requirement is (D_R, P_R), while the end-to-end QoS that the flow's default service S_D^k can provision is $(D_{S_D^k}, P_{S_D^k})$. If there exists a service $S_j, S_j < S_D^k$, which satisfies $D_{S_D^k} < D_{S_j} \leq D_R$, $P_{S_D^k} < P_{S_j} \leq P_R$, and the effective bandwidth of each individual flow is small enough as compared to the output link R, one have $P_{drop}(S_j) \leq P_{drop}(S_D^k)$, where

$P_{drop}(s)$ represents the request dropping probability using service vector s; in this single link case, $s = S_j$ and $s = S_D^k$, respectively.

Proof: This can be easily proven by using *Properties 1-3* above.

Proposition 2 Assume a flow of type k traverses a set of m nodes, each of which provides an output link with capacity R in the network. With the same assumptions as in *Proposition 1*, if there exists a service vector s, such that $D_{S_D^k} < D_s \leq D_R$ and $P_{S_D^k} < P_s \leq P_R$, then among the m nodes there exists a node i in which more flows can be admitted by using service vector s than using S_D^k, and thus $P_{drop}(s) \leq P_{drop}(S_D^k)$.

Proof: Assume two identical networks. Along the data path in network 1 the flows are using only the default service S_D^k, while in network 2 the flows are allowed to use the service vector s. Since $D_{S_D^k} < D_s \leq D_R$ and $P_{S^k} < P_s \leq P_R$, there must exist a node i at which the flow chooses service $s_i < S_D^k$. From *Proposition 1*, more flows can be admitted into node i of network 2. Therefore, $P_{drop}(s) \leq P_{drop}(S_D^k)$.

In the above proposition, P_{drop} is not limited to the flows that use the entire m nodes of the data path but also includes the flows using a subset of the m nodes and their attached links. As a result, lower request dropping probability may result in higher resource utilization in the network.

4.5 Performance Evaluation and Discussion

In the following, the performance of the proposed scheme is evaluated and compared with the corresponding performance results of Intserv operations over Diffserv networks through modeling and simulation using OPNET. The network topology used throughout the simulation study is shown in Figure 4.3, and is similar to the one used in the previous chapter (shown in Figure 3.3). The difference is that the network consists of three core routers connected by links of 9.0 Mbps to increase the capacity of the backbone network. Three services, EF, AF and BE service, are provisioned at each router. The normalized weight of EF, AF and BE services are 0.25, 0.25, and 0.5, respectively. At each router, let $L_{EF} = 40000$ bytes, $L_{AF} = 100000$ bytes and $L_{BE} = 200000$ bytes. For the EEAC-

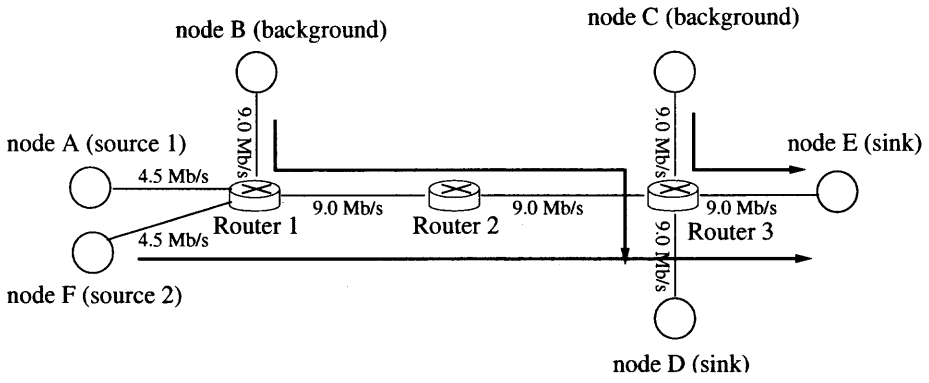

Figure 4.3 The simulated network topology.

SV scheme, the performance bounds of each congestion level for each service class is summarized in Table 4.3, which is determined by the given network and router capacity.

Throughout the simulation study, unless otherwise specified, an on-off traffic source to model some typical types of real-time flows is used, with an average rate of 12.8kb/s and the peak rate of 25.6kb/s. Each packet is 500 bytes long, while the average on and off period is 0.1s, respectively. Nodes A and F are homogeneous source nodes with maximum output data rate of 4.5 Mbps each, and two types of flows at node A and node F are defined according to the end-to-end QoS requirements in terms of delay. Specifically the end-to-end delay requirement of flow type 1 is assumed to be $D_{type1} \leq 500$ ms and the end-to-end delay requirement of flow type 2 is $D_{type2} \leq 750$ ms. It is further assumed that delay is the only QoS parameter of the two types of flows, and the closer (from the left hand side) the average end-to-end delay is to the delay bound, the more benefits that the users may obtain from the two types of flows due to the network pricing policy. An example of such an assumption can be found in the non-elastic real-time applications with service-based pricing, as described in the numerical example of section 3.5, whose utility function is shown in Figure 3.2. Node A and node F can generate both types of flows and among the traffic generated by node A or F, 50% of the flows are type 1 while the remainings are type 2. Data flows from node A and F go to node E only. The background crossing traffic consists of flows going from node B to node D and flows going from node

C to node E. The load of the background traffic in each service class with respect to the output bandwidth of the source nodes (node B and node C, 9.0 Mbps) is same as in Table 3.2. The length of probing packets is 50 bytes, which is sufficient enough for packet marking.

4.5.1 Impact of K on the Performance of the EEAC-SV Scheme

In Eq. (4.3) and (4.5) the exponential moving average was used to estimate the arrival data rate and buffer length of each service class, where $e^{-\tau_{S_j}(t)/K}$ is the exponential weight. It can be seen that when $\tau_{S_j}(t)$ is large, i.e., the interval between two packet arrivals is large, the moving average effect becomes small, while the values of $\bar{r}_{S_j}(t)$ and $\bar{L}_{S_j}(t)$ are mainly determined by the instantaneous measurement. When $\tau_{S_j}(t)$ is small, the values of $\bar{r}_{S_j}(t)$ and $\bar{L}_{S_j}(t)$ will also be affected by the effect of moving average and its weight parameter K. When K is small, $\bar{r}_{S_j}$ and $\bar{L}_{S_j}(t)$ can follow the traffic fluctuation in a timely manner. Thus, more flows maybe accepted by a service class at a node. However, in this case the transient changes in rate and buffer length cannot be filtered out, and the packet dropping probability may become large.

On the other hand, although larger K can provide stable network performance by filtering out those transient changes in rate and buffer length, it cannot provide responsive feedback regarding the changes of traffic arrivals. If K is so large that its response to the changes of data arrival rate and buffer length is too slow, it will also cause high packet dropping probability since the end hosts cannot obtain accurate estimation of the network performance. Therefore, the choice of K is constrained by the packet length, link speed, and router capacity, i.e., the buffer capacity of each service class, L_{S_j}.

For example, in the network of Figure 4.3, when the requested load is 1.0, τ_{S_j} may reach its minimum value $\tau_{S_j}^{\min} = l_{S_j}/R = 0.000444s$, where $l_{S_j} = 4000$ bits and $R = 9Mbps$. It can be seen that the moving average is a filter system with the unit sample

response function, $h(a)$ as

$$h(a) = (1 - e^{-\tau_{S_j}^{\min}/K})(e^{-\tau_{S_j}^{\min}/K})^a U(a) \tag{4.7}$$

where a is the number of packet arrivals, and determines the convergence time that is needed for the measurement result to converge to the actual data arrival rate and buffer occupancy. Assume that each arrival flow will use the available n services with equal probability $1/n$ (in the example, $n = 3$). Since $L_{EF} = 40000\ bytes = 80$ packets, $L_{AF} =$ 200 packets, and $L_{BE} = 400$ packets, to avoid packet dropping due to buffer overflow caused by the slow convergence of the moving average, the average convergence time needed for the moving average should be less than $(80+200+400)/3$ of the packet arrival intervals, i.e., $a < 226.7$. Let a_{stop} represent the convergence time, where $h(a_{stop}) = -10db$, due to the reason that $h(a)$ will have very small impact on the moving average when $a > a_{stop}$. Then it can be obtained that $a_{stop} = 226.7$, and as a result $K \leq 0.088$. The possible smallest K corresponds to the case that the moving average converges to the actual measurement immediately, i.e., $a_{stop} = 1$, which results in $K \geq 0.00039$. Therefore, K should be in the range of $0.00039 \leq K \leq 0.088$, when the packet size is 500 bytes, the link rate is 9 Mbps and the buffer size of each service queue is 40 kbytes, 100 kbytes and 200 kbytes, respectively.

In Figure 4.4, the request dropping probabilities as a function of K for flow type 1 are presented, when the requested load is 1.0. It can be seen that the request dropping probability increases as K increases, until it reaches a threshold (in this case $K \leq 0.1$). This happens because the moving average filters out more transient network state changes as K increases, and the end hosts correspondingly cannot obtain the responsive network performance. When K increases beyond a certain threshold (in this case $K > 0.1$), the request dropping probability starts to decrease, since the convergence speed of the moving average becomes slow, as compared to the network state changes.

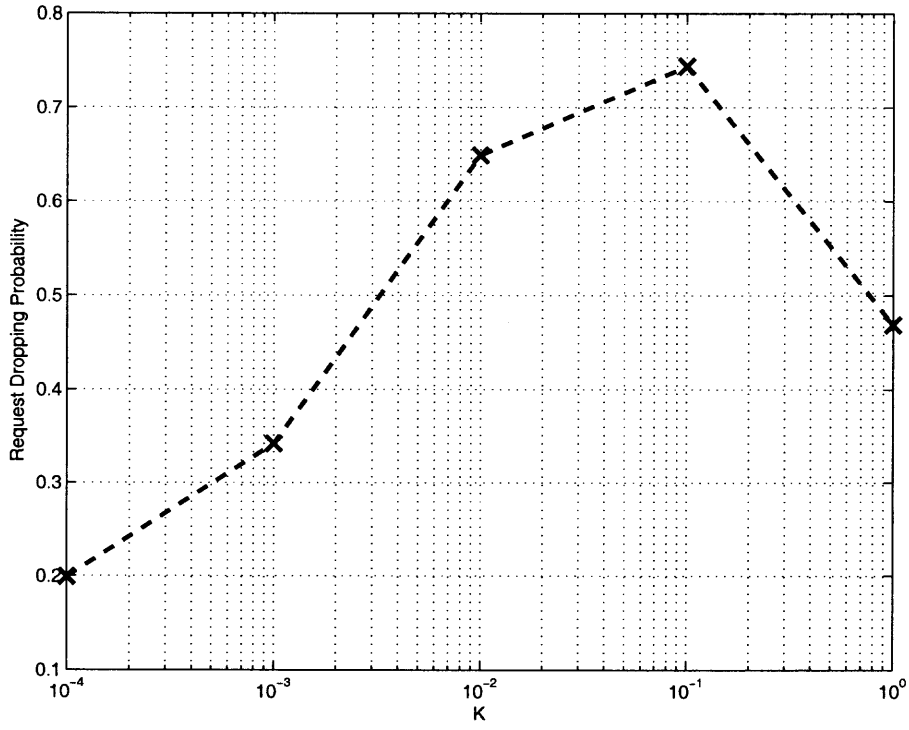

Figure 4.4 The request dropping probability with different K for flow type 1.

Although when K is small, the request dropping probability can be very low, the transient network state can result in high packet dropping probability, as shown in Figure 4.5. It can be seen that when $K \leq 0.1$, the end-to-end packet dropping probability decreases as K increases. When $K > 0.1$, the packet dropping probability starts increasing since the convergence speed for $\bar{r}_{S_j}(t)$ and $\bar{L}_{S_j}(t)$ is slow. Note that in Eq. (4.7), the minimum packet arrival interval $\tau_{S_j}^{\min}$ is used, which corresponds to the highest bandwidth utilization, 1.0. Therefore, as can be seen from the above numerical results, the derived upper bound of K is valid but conservative. The proper value of K should be chosen to obtain the desirable performance in terms of request dropping probability and packet dropping probability. In the following experiments, unless otherwise specified, $K = 0.01$ is used.

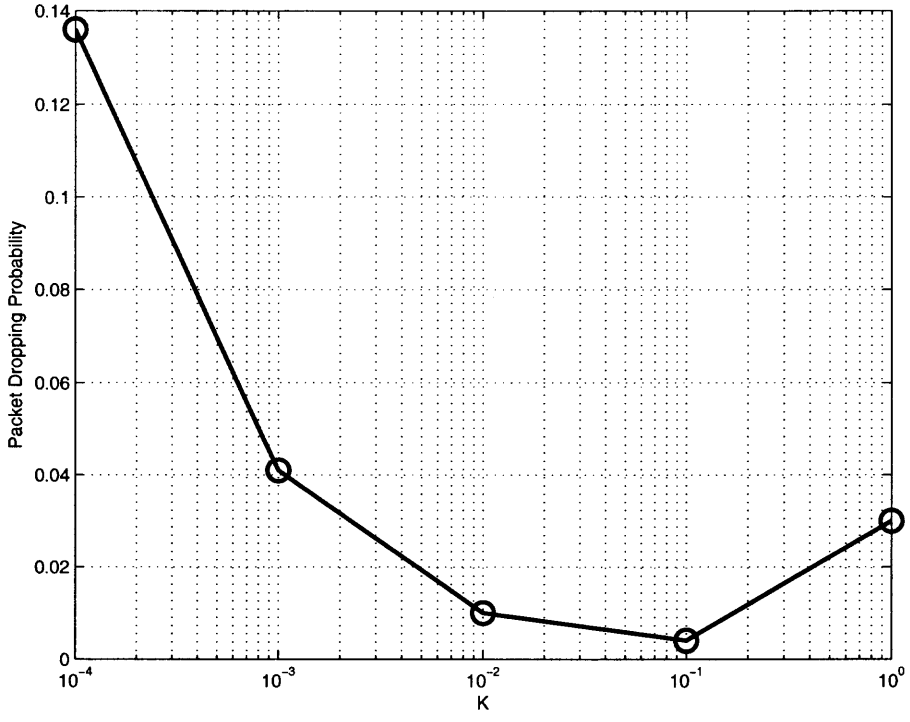

Figure 4.5 The packet dropping probability with different K for flow type 1.

4.5.2 Discussions on Probing Overhead and Probing Period

In general, probing overhead is defined as the ratio of the number of bytes received at the destination for probing, to the number of bytes received from the actual data of flows. If a flow k is admitted into the network, the overhead δ of this flow can be estimated by

$$\delta = \frac{l_{probing}}{l_{probing} + \hat{r}^{(k)}T} \tag{4.8}$$

If the probing period is $T = 0.5s$ with 50-byte probing packets ($l_{probing} = 50$), and the average data rate is $\hat{r}^{(k)} = 12.8kbps$ in the network, from Eq.(4.8) it can be obtained that the probing overhead is 5.9%. Generally, Eq.(4.8) only provides a lower bound estimation on the probing overhead, since it does not take into account the fact that a flow may drop its request, due to unsatisfactory QoS, and thus although it sends probing packets, it does not actually send data packets. For the network of Figure 4.3, the overhead of each type of data flows in the EEAC-SV scheme is shown in Table 4.4. It can be seen

that the actual overhead is slightly higher than the result of Eq.(4.8) and increases with the requested load, since more requests are dropped when the load increases. It can be also noted that the overhead for type 1 flows is higher (6.1%-6.9%) than that of the type 2 flows (6.0%-6.2%) due to the fact that more requests from type 1 flows will be dropped due to its more stringent QoS requirement. In Figure 4.6, the probing overhead for different probing periods under the requested load 1.0 for the two types of flows, is demonstrated.

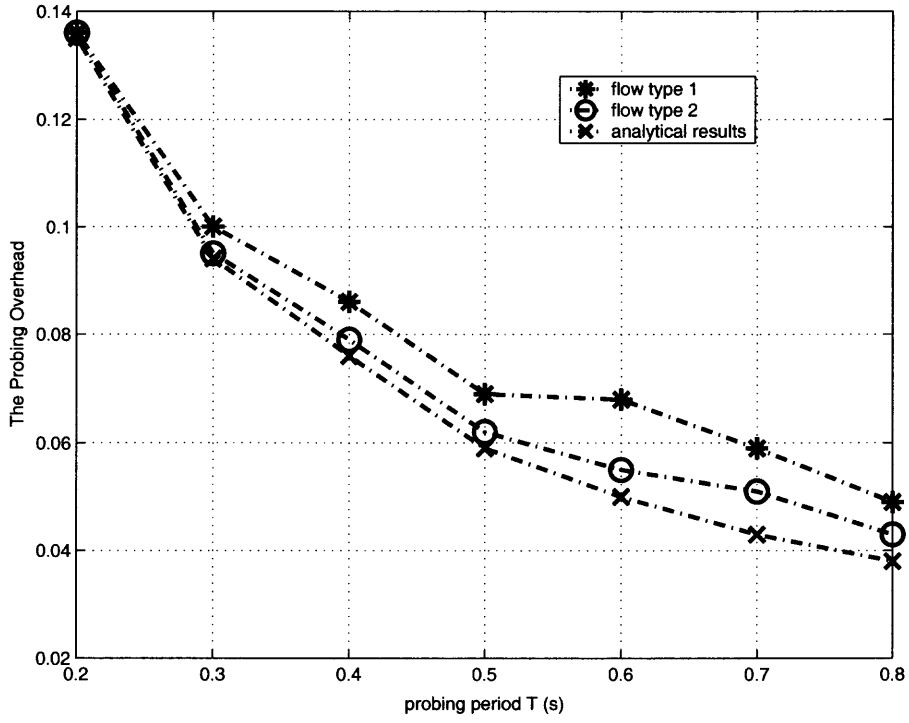

Figure 4.6 The probing overhead with different T under the requested load 1.0.

From Eq.(4.8) and Figure 4.6, it can be concluded that longer probing period T will reduce the probing overhead. However, it will increase the packet dropping probability due to the fact that the end hosts consider the probing period T to react to the performance changes at a router. From Eq. (4.6), for a specific service S_j at a node, if $\sum_{k=1}^{K_{S_j}} \hat{r}^k_{i,S_j}(\theta_i) \leq R_{S_j}$, the service queue is stable. When probing packets for flow $K_{S_j}+1$ arrives, if $\sum_{k=1}^{K_{S_j}+1} \hat{r}^k_{i,S_j}(\theta_i) \leq R_{S_j}$, the flow can be admitted to the service; otherwise, the probing packet should be marked so that the flow will not use S_j at this node. However,

due to the convergence time needed for the rate and buffer measurement, a flow may be admitted into the service by mistake and the service will be overloaded. In this case, it will take up to T time units before any flows changing services or dropping the request, during which the additional overloaded packets will be dropped. If the requested load, R_a, is used to approximate the probability that such an overload will occur, the packet dropping probability of service S_j, as a function of the requested load R_a, $\hat{P}_{S_j}(R_a)$, can be estimated by

$$\hat{P}_{S_j}(R_a) = \frac{\hat{r} \cdot T \cdot R_a}{(L_{S_j})} \tag{4.9}$$

where $\hat{r}$ represents the individual flow rate and it is assumed that $\hat{r} << R$, in which R is the link rate. Assuming that at each node the flow will choose the n available services with equal probability $1/n$, then at a node i, the packet dropping probability $\hat{P}_i(R_a)$ would be

$$\hat{P}_i(R_a) = \frac{1}{n} \sum_{j=0}^{n-1} \frac{\hat{r} \cdot T \cdot R_a}{(L_{S_j})} \tag{4.10}$$

Correspondingly, the end-to-end packet dropping probability $\hat{P}_{e2e}(R_a)$ is given by

$$\hat{P}_{e2e}(R_a) = 1 - \prod_{i=0}^{m-1} (1 - \hat{P}_i(R_a)) \tag{4.11}$$

It should be noted that in Eq.(4.9), the maximum individual flow rate may be used to estimate the worst case packet dropping probability, and assume all packets of the flow during T will be dropped. Furthermore, a flow's QoS requirements are not taken into account. Thus, Eq. (4.11) can be considered as the estimation for the upper bound of the end-to-end packet dropping probability, while flows with stringent QoS may have a much smaller packet dropping probability, because they may have stopped using the service before the service queue is unstable. In Figure 4.7, the end-to-end packet dropping probabilities for type 1 flows and type 2 flows, as a function of the requested load, are presented, for the case that $T = 0.5s$. The corresponding analytical results derived from Eq. (4.11) are also shown in this figure. It can be seen that in general the analytical results

are larger than the corresponding simulation results for the two types of flows, while flows of type 1 have much smaller packet dropping probability than the analytical bound. This happens because flows of type 1 have more stringent QoS requirement and will stop using the service before the network service starts getting congested.

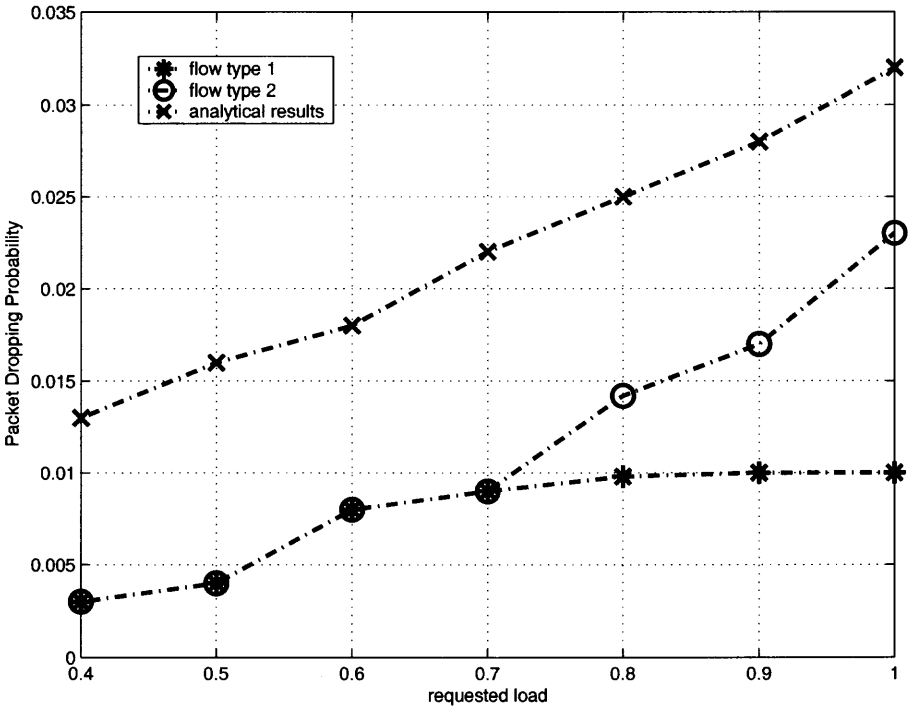

Figure 4.7 The end-to-end packet dropping probability with different requested load when $T = 0.5s$.

In Figure 4.8, the packet dropping probabilities of type 1 and type 2 flows obtained from simulation, and the corresponding analytical results, are shown as a function of different probing periods when the requested load is 1.0. Although noted from the previous results and discussions that increasing T can reduce the probing overhead, from Figure 4.8, it is observed that smaller T can detect the network state changes responsively, while larger T will increase the packet dropping probability correspondingly. Furthermore, in EAC schemes, the thrashing effect [22] exists, in which many flows send probing packets simultaneously and get similar probing results, resulting in the overload of the service at a router. The thrashing effect becomes significant, when the probing period increases. This

is because large probing period will make flows take longer time to detect the network state changes and take actions to avoid it. As a result, even more packets will be dropped than given by Eq. (4.9)-(4.11), when the probing period T increases. This can be seen from Figure 4.8, in which the analytical results become smaller than the simulation results when $T > 0.5$. In the following experiments, $T = 0.5s$ is considered.

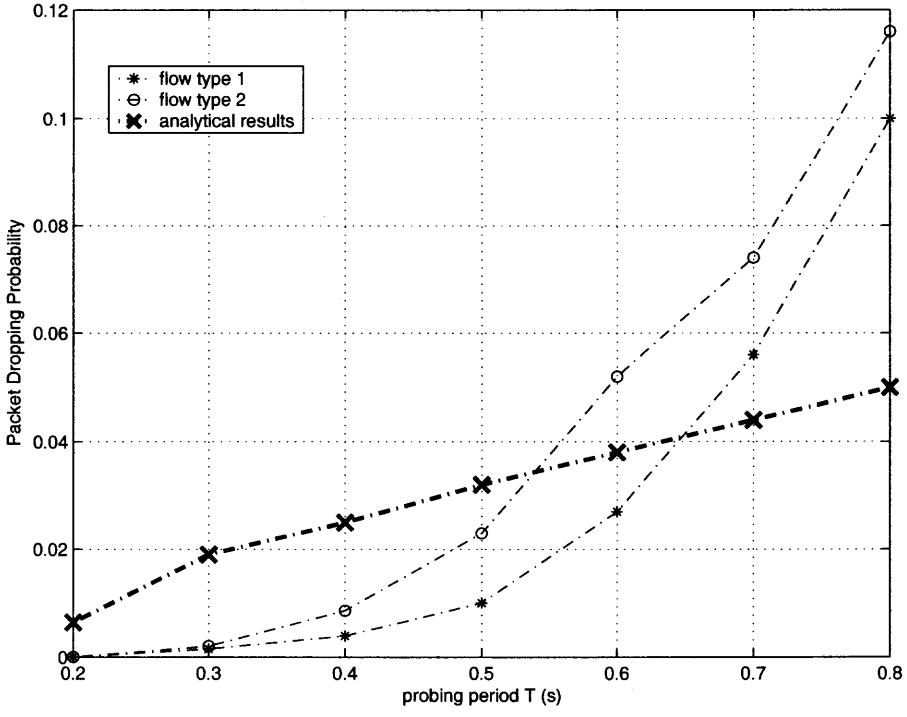

Figure 4.8 The end-to-end packet dropping probability with different T when the requested load is 1.0.

4.5.3 Experiment 1: Heterogeneous Traffic Burstiness

In the previous experiments, the emphasis was placed on gaining some insight about the behavior of the system parameters in the EEAC-SV scheme. Therefore, the same traffic burstiness was used for both the type 1 and 2 flows. However, in general, the traffic in the network is aggregated from different sources and applications which have different burstiness characteristics. In this experiment (Experiment 1), a CBR traffic of 12.8Kbps is used to replace the type 2 flow. The corresponding end-to-end delay and request dropping

probability are shown in Figure 4.9 and Figure 4.10, respectively. It can be seen that, compared to the performance of the homogeneous traffic types, the flow's performance in terms of end-to-end delay and request dropping probability presents similar trend, although the CBR traffic reduces the traffic burstiness and results in smaller end-to-end delay and request dropping probability in the heterogeneous traffic burstiness experiment. In the following experiments, unless otherwise specified, type 1 and type 2 flows will have same burstiness.

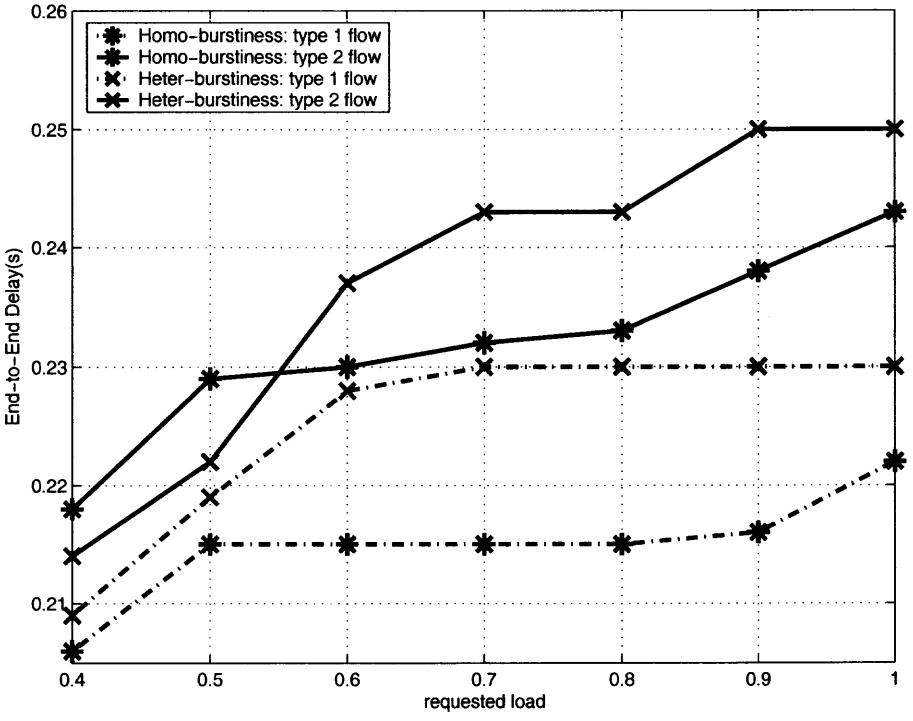

Figure 4.9 The end-to-end delay of flow type 1 and 2 with different requested load and traffic burstiness.

4.5.4 Experiment 2 – Flows with Heterogeneous QoS Requirements

Since the objective of the proposed scheme (EEAC-SV) is to provide fine end-to-end QoS granularity so that flows with different QoS requirements can achieve different performance, in the following experiment (Experiment 2), the performance of the EEAC-SV scheme is exhibited when heterogeneous QoS requirements of flows coexist

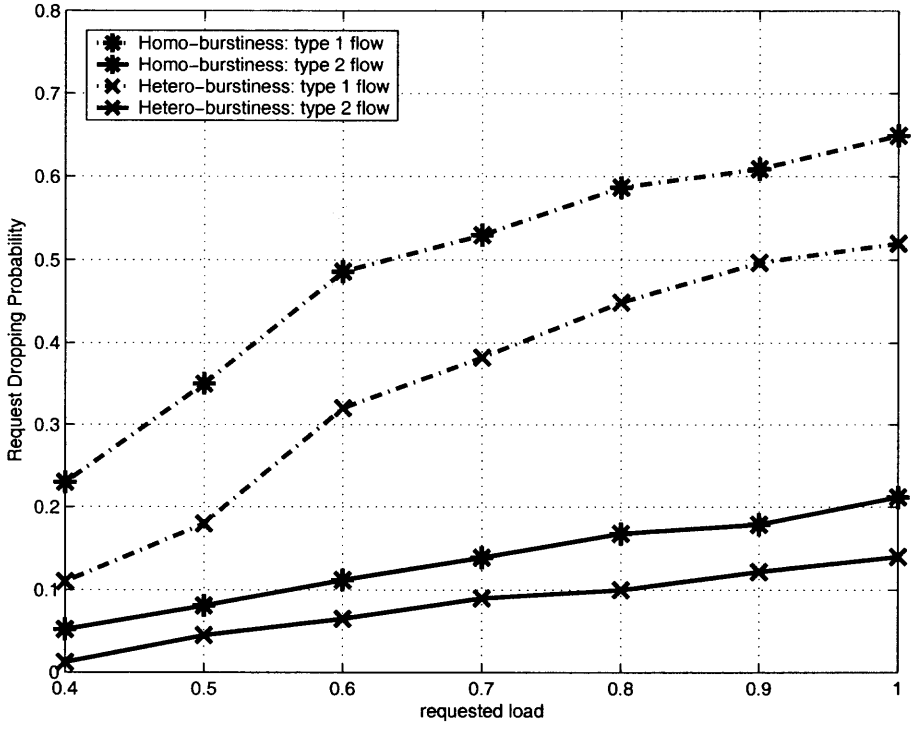

Figure 4.10 The request dropping probability of flow type 1 and 2 with different requested load and traffic burstiness.

in the network. Three QoS provisioning scenarios are considered and evaluated in this study: EEAC-SV, Intserv over Diffserv with static resource allocation, and Intserv over Diffserv with dynamic resource allocation.

- *EEAC with Service Vector (EEAC-SV)* scheme implements the end-to-end QoS provisioning architecture proposed in this dissertation. At each router, the user always tries to use the lowest priority service that can guarantee the end-to-end QoS bound.

- *Intserv over Diffserv with Static Resource Allocation (IDSRA)* scheme statically maps the flow to a specific service (e.g., the EF service) while at each node the guaranteed bandwidth (i.e., the normalized weight) of the service is kept constant and the buffer allocated to each service queue cannot be shared.

- *Intserv over Diffserv with Dynamic Resource Allocation (IDDRA)* scheme maps the flow to a service, while both the bandwidth and buffers of each service at each node can be shared among different service classes. For demonstration purposes, the EF service provisioning is mainly studied and compared with the EEAC-SV scheme. Therefore, it is assumed that the EF service can share the bandwidth and buffers from BE and AF services while the other two services cannot share bandwidth and buffers from the EF service. To avoid starvation of AF and BE services, the AF and BE services are guaranteed 50% of their original bandwidth and buffers. Moreover, the dynamic allocation of bandwidth and buffers are non-preemptive, i.e., only idle bandwidth and buffers can be allocated to the EF service when the EF service needs more resources. Note that the idle bandwidth is defined as $\max(0, R_{S_j} - \bar{r}_{S_j})$ where $S_j \in \{BE, AF\}$.

Moreover, for the IDSRA and IDDRA schemes, it is assumed that the routers can be aware of the flow's end-to-end QoS requirements and make admission control decisions accordingly, although they may map the flows with different QoS requirements to the same service provisioned in the network. It can be found that due to the performance bound at each router, the two types of flows have to be mapped to the EF service in both of the IDSRA and IDDRA schemes and the services provisioned at each router can be sorted as $EF > AF > BE$.

Figure 4.11 shows the average end-to-end delay of the two types of the flows with different requested load under the three scenarios. In the EEAC-SV scheme, different types of flows have different end-to-end delay due to their different requirements. However, in the IDSRA and IDDRA schemes, the average end-to-end delay of the two types of flows are very close due to the fact that the IDSRA and IDDRA schemes cannot provide service differentiation to these two types of flows, both of which are mapped to the EF service. The slight performance difference between the two types of flows in the IDSRA and IDDRA schemes is due to the fact that the admission control procedure may

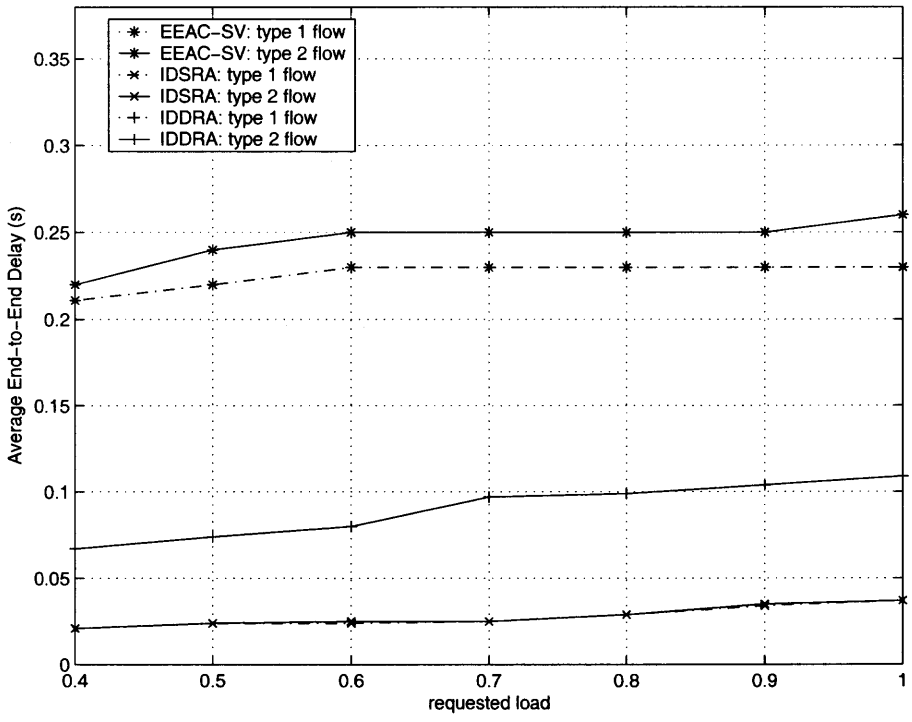

Figure 4.11 The average end-to-end delay of flow type 1 and 2 under different operation schemes.

make different admission decisions. The results presented in this figure demonstrate that the EEAC-SV scheme can provide finer QoS granularity than the IDSRA and IDDRA schemes.

The effects of the finer QoS provisioning capability of the EEAC-SV scheme are demonstrated in Figure 4.12 in which type 1 and type 2 flows received different request dropping probability in the EEAC-SV scheme, while the corresponding values are close under the IDSRA and IDDRA schemes. For the EEAC-SV scheme, the request dropping probability consists of the request dropping probability before a flow starts sending data, and the request dropping probability during the data forwarding stage when the end host finds that no feasible service vector can be found due to the dynamics of the network load. In general, the request dropping probabilities for the two types of flows in the IDSRA and IDDRA schemes are larger than those in the EEAC-SV scheme due to the fact that the finer end-to-end QoS granularity provided by the EEAC-SV scheme can fit the flows'

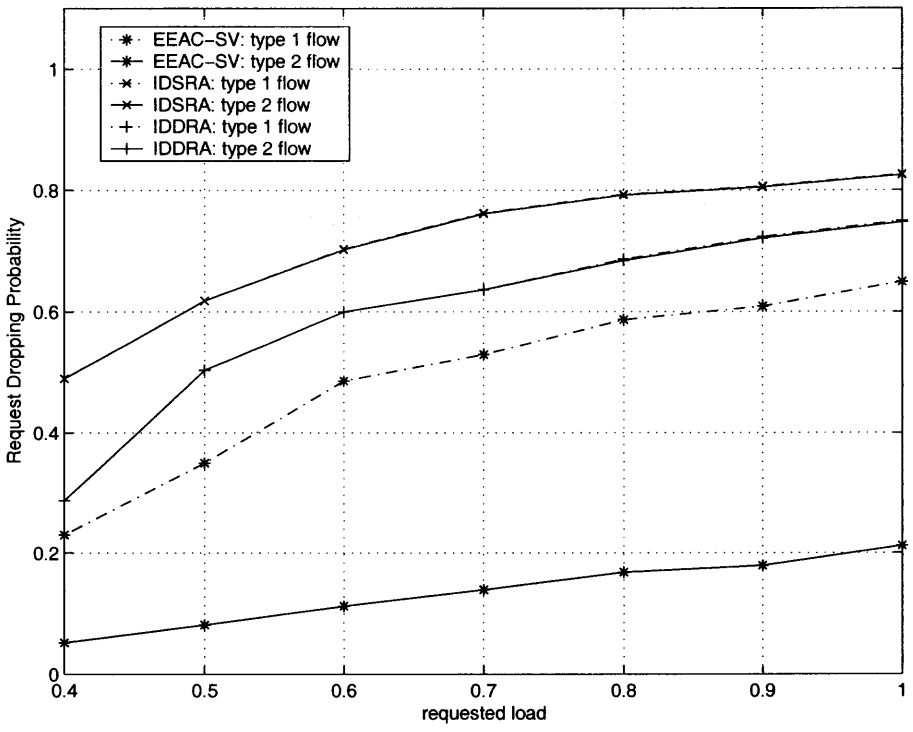

Figure 4.12 The request dropping probability of flow type 1 and 2 under different operation schemes.

requirements better, and as a result each flow consumes less resources; while the QoS granularity is not enhanced in the IDSRA and IDDRA schemes and each flow may get more service than it actually requires. Since EEAC-SV can provide lower request dropping probability for each type of the flows, it may result in higher network resource utilization.

4.5.5 Deployment Studies

Generally, new QoS provisioning models will not be deployed in the entire network immediately due to the cost and feasibility considerations. Therefore, the new provisioning model should be compatible to currently deployed schemes, in the sense that it can coexist with them in the network, and both new and conventional schemes will not be negatively affected by each other. It can be seen that the proposed EEAC-SV scheme can be compatible to QoS provisioning models which need RSVP as the signaling protocol.

Furthermore, two experiments are designed and the corresponding results are provided, to demonstrate: 1. the proposed EEAC-SV scheme can coexist with the conventional end-to-end congestion control mechanisms used by TCP flows and will share the network resources fairly with them (Experiment 3); 2. the EEAC-SV scheme can be deployed gradually and the data path used by an end host pair can consist of nodes that does not support EEAC-SV but only the conventional IDSRA scheme, while both the user and service provider can still achieve benefits by using the EEAC-SV scheme (Experiment 4). From these two experiments, one can conclude that EEAC-SV can be deployed in today's Internet gradually and incrementally.

Experiment 3–TCP friendly

In Experiment 3, it is demonstrated that the proposed architecture preserves a compatible and friendly networking environment for conventional TCP flows, which is an essential feature for a QoS provisioning scheme in today's Internet due to the reason that end-to-end congestion control mechanisms have been widely deployed through TCP [34]. It is expected that the proposed service provisioning architecture with the EEAC-SV scheme will be mainly used by flows with guaranteed QoS requirements, while these flows may not have end-to-end TCP-friendly congestion control mechanisms. Since the service vector mechanism will allow these flows to use the AF and BE services, which are conventionally reserved for TCP flows, it is important that the aggregated flows adopting the proposed service provisioning architecture will not affect the performance of TCP flows negatively when they use the AF or BE service. To demonstrate the TCP friendly feature of the proposed architecture, in this experiment, the type 2 flows are changed to TCP Reno flows that use the AF service only from the source (node A and F) to the destination (node E). When the requested load is 1.0 (i.e., the heaviest load), the throughput of each type of the flows from node A is shown in Figure 4.13 and Figure 4.14. It can be seen that in the EEAC-SV scheme, TCP flows and type 1 flows receive similar throughput and their performance is stable. Although in the IDSRA scheme the TCP flows have

larger throughput due to the static resource allocation to the AF service at each node, the throughput of type 1 flows is low due to their strict QoS requirements and lack of the resource they need, which is unfair when both types of flows are heavily requesting service from the network. In the IDDRA scheme, although the throughput of both types of flows is similar, their performance changes dramatically over the time, which is an undesirable feature of the network service. From this experiment, it can be seen that the proposed QoS provisioning architecture with the EEAC-SV scheme will provide a TCP friendly environment in the network for TCP flows. It should be noted that in the simulation there are not any end-to-end congestion control mechanisms to be assumed for the type 1 flows. However, the probing packet marking scheme and algorithm will control the aggregated load to each service so that these flows will not use up all the network resources and starve the TCP flows.

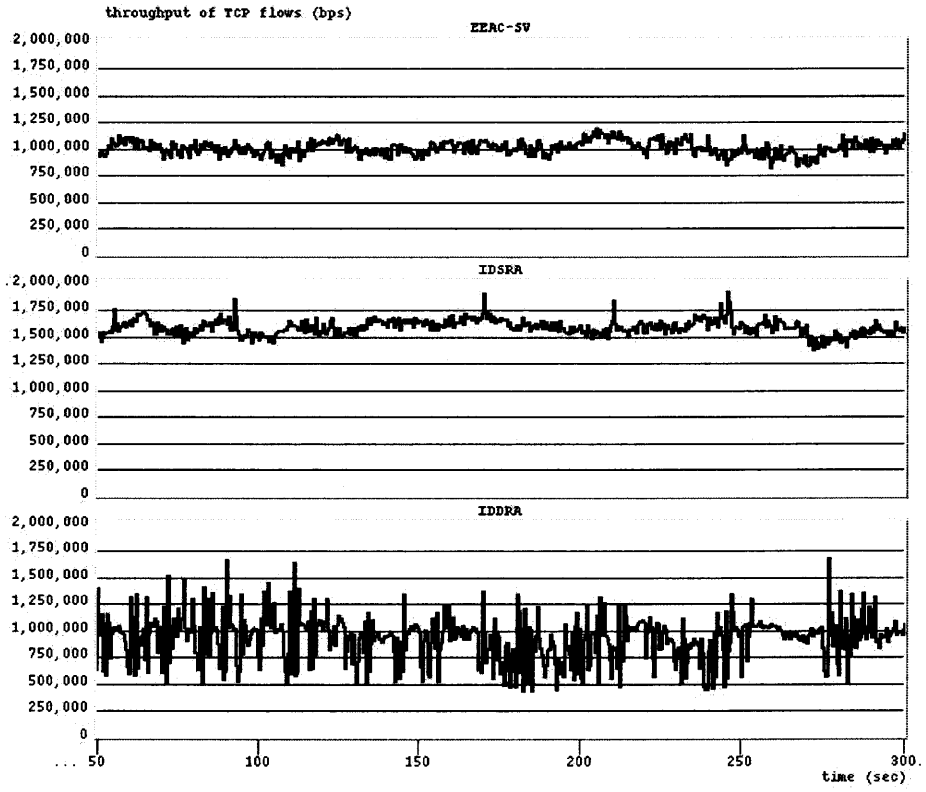

Figure 4.13 The throughput of TCP flows under the requested load of 1.0.

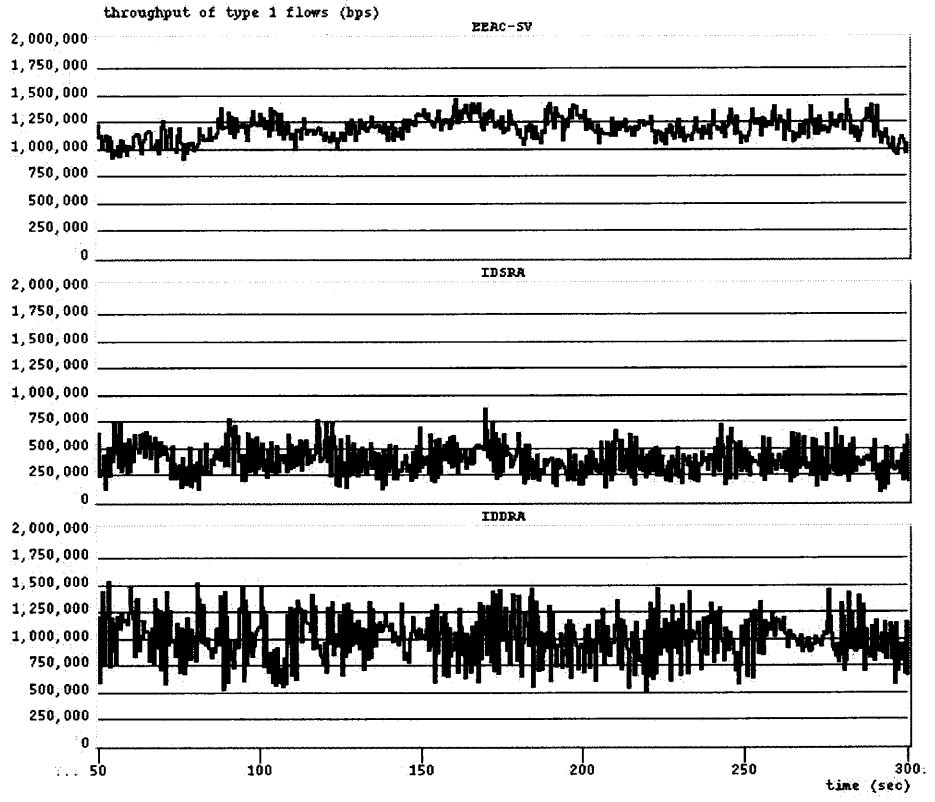

Figure 4.14 The throughput of type 1 flows under the requested load of 1.0.

Experiment 4: Hybrid Data Path

During the incremental deployment procedure, the source and destination pair may expand several network domains, among which some domains may not support the EEAC-SV scheme (i.e., a hybrid data path is formed). If each flow has a default service and each service provider can provide the default performance metrics of the services, the end hosts can use the default service for the flow and evaluate the corresponding performance by using the default metrics at routers that do not support the EEAC-SV scheme. In the following, an experiment (Experiment 4) is used to demonstrate that benefits can be achieved from the proposed EEAC-SV scheme on the hybrid data paths. Assume in Figure 4.3 that router 2 does not support EEAC-SV, but only supports the Intserv over Diffserv operations and the service mapping scheme. Also assume that it uses the same resource allocation algorithm as the routers described in the IDSRA scheme of Experiments 2 and

3. In Figure 4.15, the request dropping probabilities of flow type 1 in the EEAC-SV, EEAC-SV on hybrid data paths, IDSRA and IDDRA schemes are shown. It can be seen that when router 2 is using the IDSRA scheme on the hybrid data path and only static service mapping and resource allocation are supported, the request dropping probability is higher than the one when every router supports the EEAC-SV scheme. However, the request dropping probability is still lower than those probabilities on data paths with routers only deployed with the IDSRA or IDDRA schemes. Moreover, the EEAC-SV on hybrid data paths can still achieve better QoS differentiation capabilities as compared to the IDSRA or IDDRA schemes. This experiment demonstrates that both users and network service providers can start achieving benefits from the EEAC-SV scheme, even when the EEAC-SV scheme is only partly deployed in the network. Specifically, the deployment of EEAC-SV can start from the edge routers and be extended to core routers gradually. Thus, the EEAC-SV scheme can be deployed incrementally based on the current Diffserv architecture.

4.6 Conclusion

In this chapter, the EEAC-SV mechanism that enables the end hosts to choose different services at different nodes is described, and a probing packet marking mechanism to be incorporated into the EEAC-SV scheme in order to effectively estimate the service performance at each router and reduce the probing overhead, is proposed. The main feature of this new QoS provisioning paradigm is that it decouples the end-to-end QoS provisioning from the service provisioning at each router, thus enhancing the end-to-end QoS granularity in the Diffserv network while maintaining the simplicity of the service implementation at each router. It can be seen that the definitions and implementations of each service can remain the same at each core router as in the conventional Diffserv networks, and the deployment of the proposed architecture can be incremental.

By providing finer end-to-end QoS, the network can provide better services to the user, and enhance its resource utilization by reducing the request dropping probability.

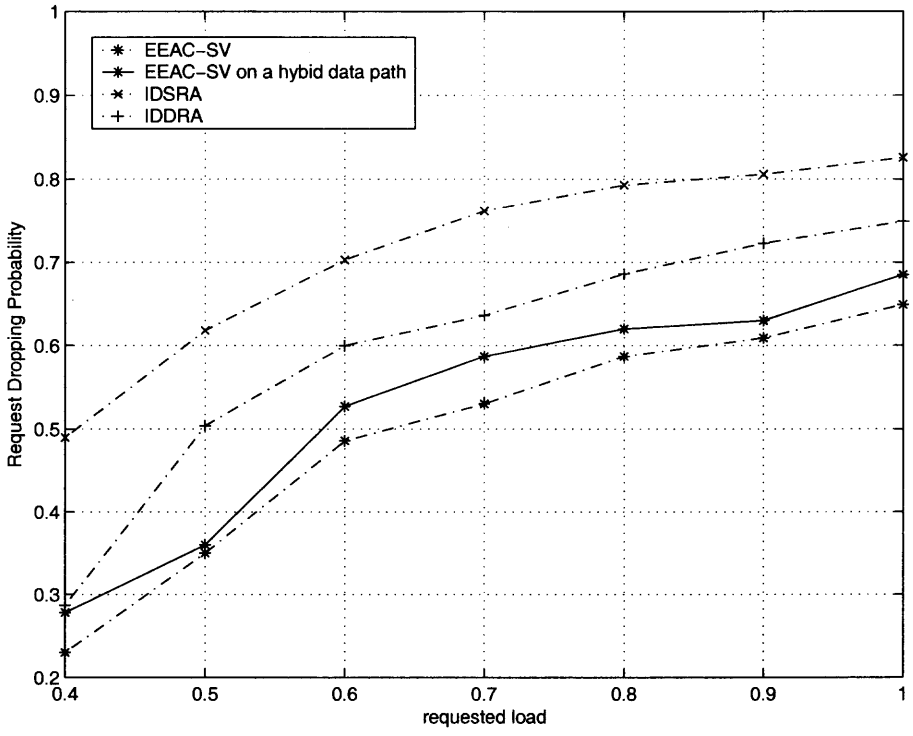

Figure 4.15 The request dropping probability for the type 1 flows.

The numerical results presented here, demonstrate that the proposed QoS provisioning architecture can have better service differentiation capability than the Intserv over Diffserv schemes via "customizing" the QoS offered to each user, reduce the request dropping probability, and provide a compatible and friendly networking environment for current TCP flows with end-to-end congestion control mechanisms.

The proposed EEAC-SV scheme have the following advantages over the conventional per-hop based, centralized or endpoint admission control schemes, and the other existing service mapping strategies in the Diffserv networks.

- Finer end-to-end QoS granularity in Diffserv networks: The EEAC-SV scheme enhances the granularity of the QoS space without increasing the number of service classes in Diffserv networks. Since the same end-to-end QoS may be achieved by more than one combination of service classes, users may flexibly choose the desired combinations of available services to achieve their QoS requirements. The proposed

service vectors enhance the flexibility for accommodating a wider range of QoS parameters as needed for emerging requirements and applications.

- QoS provisioning improvement: service vectors make it possible for a flow to acquire its desired QoS by using different services at different nodes, while the conventional approach of using the same service along the whole path may not be able to provide the same QoS performance. Such an approach, from the perspective of a user, may improve the net benefits from the network services, while from the network perspective, may result in higher network utilization.

- Distributed user based decisions and computation: it is desirable for users to adopt their own strategies in using the network service and optimize their benefit. The optimization of a user benefit may be relatively complicated and therefore, it is difficult for a centralized device at the network side to provide the optimal solution for each user. In the proposed framework, the optimization problems are solved in a distributed way at end hosts, and thus the computational load is balanced in the network.

- The measurement results are obtained from the measurement of the aggregated traffic flows at each node, which reflects the network condition more accurately.

- It is demonstrated that the EEAC-SV scheme can be deployed easily and incrementally in current Diffserv networks. The upgrade can start from the edge routers and be extended to core routers incrementally and gradually.

m	the number of routers that a data flow passes through from the source to the destination
n	the number of services provided at each router
S	the set of available service classes at each router
S_j	service class j in S
s	the service vector of a flow
s_i	the service that a flow chooses at router i
$D(S_j)$	the delay bound of service S_j at a router
L_{S_j}	the buffer length of service S_j at a router
R_{S_j}	the guaranteed service rate of service S_j at a router
L	the maximum packet length
R	the output link capacity
$P_{S_j}(t)$	the random packet dropping probability of service S_j at time t
$\bar{L}_{S_j}(t)$	the moving average queue length of service S_j at time t
$L_{S_j}^{\min}$	the buffer length threshold of service S_j above which the random dropping algorithm starts dropping packets
$p_{S_j}^{\max}$	the maximum dropping probability of service S_j determined by the random dropping algorithm at a router
τ_{S_j}	the interval between the current time t and the arrival time of the previous received packet of service S_j
K	the constant in the moving average
Y_{S_j}	the number of congestion levels of service S_j at a router
$\bar{r}_{S_j}(t)$	the average packet arrival rate of service S_j at time t
T	the probing period
$\hat{r}^{(k)}$	the bandwidth requirement of flow k

Table 4.1 Notations of Chapter 4

Congestion Level	0	1	2
EF	$(\frac{L_{EF}^{min}}{R_{EF}} + \frac{L}{R}, 0)$	$(\frac{L_{EF}}{R_{EF}} + \frac{L}{R}, p_{EF}^{max})$	$(\infty, 1)$
AF	$(\frac{L_{AF}^{min}}{R_{AF}} + \frac{L}{R}, 0)$	$(\frac{L_{AF}}{R_{AF}} + \frac{L}{R}, p_{AF}^{max})$	$(\infty, 1)$
BE	$(\frac{L_{BE}^{min}}{R_{BE}} + \frac{L}{R}, 0)$	$(\infty, 1)$	$(\infty, 1)$

Table 4.2 The performance bounds under different congestion levels for the EF, AF and BE services at a core router

Congestion Level	0	1	2
EF	$(0.057s, 0)$	$(0.128s, 0.01)$	$(\infty, 1)$
AF	$(0.142s, 0)$	$(0.320s, 0.02)$	$(\infty, 1)$
BE	$(0.142s, 0)$	$(\infty, 1)$	$(\infty, 1)$

Table 4.3 The performance bounds under different congestion levels for the EF, AF and BE services at a core router in the EEAC-SV scheme

Requested Load	0.4	0.5	0.6	0.7	0.8	0.9	1.0
Type 1 flow	6.1%	6.2%	6.3 %	6.5%	6.6%	6.9%	6.9%
Type 2 flow	6.0%	6.0%	6.2%	6.2%	6.2%	6.2%	6.2%

Table 4.4 The overhead for delivering each type of data flows by the EEAC-SV when $T = 0.5s$

CHAPTER 5

CONCLUSIONS AND FUTURE WORK

During the last decade, innovations concerning networking technologies, as well as advances in digital compression and transmission, have dramatically increased the capabilities and the efficiency of the existing and imminent access, metropolitan and core networks. Meanwhile, the Internet has evolved from a research network, targeting a limited audience of academic and military users, to a huge and commercially operated network. If the next generation of network technology is to operate beyond the levels of current networks, it will require a set of well-designed mechanisms for the provisioning of the various QoS. QoS provisioning in the Internet includes providing end-to-end performance guarantees such as bandwidth, delay, packet loss, security, etc. to individual data flows. In general, the QoS provisioning mechanisms can be categorized as single node and end-to-end mechanisms, which must work together to provide the desired end-to-end QoS to flows.

5.1 Summary of Contributions

Due to the high data rate supported in both the access and backbone network, the design of QoS provisioning mechanisms often face the trade-offs between the scalability and granularity in QoS provisioning. In this dissertation, both the single-node and end-to-end QoS provisioning mechanisms were studied, which can provide scalable services with fine granularity to the users, so that both users and network service providers can achieve more benefits from the QoS provisioned in the network.

Specifically, on the single node QoS provisioning mechanism, the problem of sharing the output bandwidth among the large amount of data flows was studied, so that fairness in the bandwidth allocation among the flows can be achieved in a scalable fashion. In the literature, the one-rate grouping architecture provides a scalable architecture to implement the PFQ algorithms which approximate the ideal fluid model to achieve

fairness. However, the one-rate grouping architecture can only provide coarse rate granularity, which may result in the low utilization of the resources, unfairness in resource allocation among different flows and poor immunity capability.

A dual-rate grouping architecture is proposed in this dissertation, in which granularity in rate allocation can be enhanced, while the scalability of the one-rate grouping architecture is still maintained. The proposed architecture reduces the sorting complexity from $O(\log N)$ in the per-flow based architecture, where N is the number of flows, to $O(\log M)$, where M is the number of rate groups in the grouping architecture; while the rate allocation granularity can be as fine as the per-flow based architecture. The corresponding results demonstrated that in the proposed scheme, the fairness can be achieved better than in the one-rate grouping architecture. Moreover, the dual-rate grouping architecture approximates the ideal per-flow based PFQ architecture better than the one-rate grouping architecture and provides better immunity capability.

On the end-to-end QoS provisioning techniques, the design and development of scalable QoS provisioning architectures were mainly studied, in order to enhance the QoS granularity provisioned in the Diffserv network, while maintaining its simplicity and scalability.

A new Endpoint Admission Control scheme for Diffserv networks, referred to as Explicit Endpoint Admission Control (EEAC) scheme, was proposed, in which the admission control decision is made by the end host based on the end-to-end performance of the network. Furthermore, a new concept, the service vector concept, was introduced, by which an end host can choose different services at different routers along its data path.

Thus, the proposed service provisioning paradigm decouples the end-to-end QoS provisioning from the service provisioning at each router. Following this paradigm, the end-to-end QoS granularity in the Diffserv networks can be enhanced. With the EEAC and service vector paradigm, the definition of each service and their PHBs at each router can remain the same and the implementation complexity of the Diffserv model is maintained.

If the data flow path contains m intermediate routers with n available services at each hop, the granularity of the end-to-end QoS provisioned to a data flow by the network can be as fine as $O(n^m)$, which is significantly finer than the static service mapping scheme in the Diffserv model which only provides service granularity of $O(1)$. The study on the user optimization model by using the EEAC and service vector scheme showed that it can enhance the user benefits from the services provisioned in the network.

The implementation of the EEAC and service vector paradigm, referred to as EEAC-SV, in the Diffserv architecture, was proposed, by using RSVP signaling messages as the probing packets. A packet marking algorithm at each router was described, so that the service performance can be represented efficiently in the probing packets, and the end hosts can evaluate the end-to-end performance effectively. The parameters of the moving average, probing period, and overhead are studied, while the trade-offs on the choices of different parameter values are presented.

The performance analysis and simulation results demonstrated that the proposed EEAC-SV scheme not only increases the benefit to the service users, but also enhances the benefits to the network service provider, by providing finer QoS granularity to the data flows in the network. It is demonstrated that the proposed EEAC-SV scheme can have better service differentiation capability, and enhance the network resource utilization by reducing the request dropping probability, as compared to other service provisioning operations in Diffserv networks. The numerical results also indicated that the proposed QoS provisioning architecture via EEAC-SV can provide a compatible and friendly networking environment to the widely deployed TCP flows, which use conventional end-to-end congestion control mechanisms, and therefore, the EEAC-SV scheme can be deployed in the current Internet in an incremental and gradual manner.

5.2 Future Work

In the end-to-end QoS provisioning architecture and methodologies presented in this dissertation, the emphasis was placed on the user optimization model and the

implementation schemes, while service-based pricing with constant unit price at each router, was assumed. At the same time in the literature, the study on the pricing schemes was also placed on the user side, under different networking context [21], [28], [38], etc. However, pricing schemes can be used as means for traffic management and congestion control. Through pricing, the network may convey its preference on the resource allocation, and send signals to the users providing incentives that may influence their behavior. For instance, each router may dynamically adjust its unit price according to the demand of its service, which will affect the decision of each user when choosing the service vector. Thus, besides the non-cooperative games between the users, as described by Eqs.(3.1)-(3.3), there exist cooperative or non-cooperative games between the network service providers and users, as well as among different routers, which need to be studied and investigated.

Furthermore, throughout this study only unicast flows were considered. However, given the emergence of new multimedia and real-time applications, the issue of multicast traffic support with QoS guarantees, becomes a challenging one. Since, it is envisioned that in the next generation Internet, multicast traffic will increase, the enhancement of the EEAC-SV scheme to provide enhanced QoS support to such traffic, needs to be considered and studied. Specific questions and issues that need to be addressed, are how the multicast probing can be implemented efficiently and effectively, and what is the achievable performance of EEAC-SV within the multicast context.

Moreover, the explosive growth of the Internet combined with the continued dramatic increase for all wireless services, lead to a growing need to access information while on the move or away from office or home. These trends, along with the convergence of communications, information, commerce and computing, are creating significant demand and opportunity for multimedia personal communication services. However, large-scale deployment of multimedia services over next generation mobile systems depends heavily on the offered QoS. The proposed EEAC-SV scheme may be extended to

wireless access networks (Wi-Fi, 3G/4G) as well, in which each handset may send probing packets not only to detect service availability at each router, but also to detect the wireless channel capacity, which is time-dependent. The probing can be combined with other signaling messages, and can help the mobile resource allocation process. Furthermore, in the wireless network, the entity that performs probing may also be the base station or access point for the down-link (forward) detection, to determine the optimal scheduling approach [39]. Extending the EEAC-SV mechanism into wireless Internet services, is an issue of high research and practical importance, that needs to be further investigated and analyzed.

Finally, as explained before, in the EEAC-SV scheme as well as in other EAC schemes, the decision of whether or not, and how the network service is utilized, is placed on the user side. This means that each user will send probing packets periodically to the network, which needs additional operations at the router to process the probing packets. Thus, a malicious user may issue the Denial of Service (DoS) attack [40] to the network, by sending large amount of probing packets to slow down and even disrupt the router services. Malicious users may also "hide" their actual data in the probing packets to avoid possible service charge by the network service provider. Therefore, the network security becomes an additional concern in the EEAC-SV scheme. Authentication Authorization Accounting (AAA) protocols [41] are needed to assure that only legitimate users will apply EEAC-SV schemes in a legitimate manner.

REFERENCES

[1] X. Xiao and L. M. Ni, "Internet QoS: A Big Picture," *IEEE network*, pp. 8–18, March/April 1999.

[2] A. S. Tanenbaum, "Computer Networks ," *Prentice Hall*, 1996.

[3] Cisco, *Internetworking Technology Handbook*, ch. 49. Cisco Press, February 2002.

[4] R. Braden, D. Clark, and S. Shenker, "Integrated Services in the Internet Architecture: an Overview," *RFC1633*, June 1994.

[5] S. Blake, D. Black, M. Calson, E. Davies, Z. Wang, and W. Weiss, "An Architecture for Differentiated Services," *RFC2475*, December 1998.

[6] R. Braden, L. Zhang, S. Berson, S. Herzog, and S. Jamin, "Resource ReSerVation Protocol (RSVP)–Version 1 Functional Specification," *RFC2205*, Septmember 1997.

[7] A. K. Parekh and R. G. Gallager, "A Generalized Processor Sharing Approach to Flow Control in Integrated Services Networks: The Single-Node Case," *IEEE/ACM Transactions On Networking*, vol. 1, pp. 344–357, June 1993.

[8] A. Demers, S. Keshav, and S. Shenker, "Analysis and Simulation of a Fair Queueing Algorithm," *J. Internetworking Res. Experience*, vol. 1, pp. 3–26, 1990.

[9] C. Wang, K. Long, X. Gong, and S. Cheng, "Effective Fairness Queueing Algorithms," *ICON'2000*, pp. 294–301, 2000.

[10] J. C. R. Bennett, D. C. Stephens, and H. Zhang, "Hierarchical Packet Fair Queueing Algorithms," *ACM-SIGCOMM'96*, pp. 143–156, 1996.

[11] D. C. Stephens, J. C. R. Bennett, and H. Zhang, "Implementing Scheduling Algorithms in High-speed Networks," *IEEE Journal on Selected Areas in Communications*, vol. 17, pp. 1145–1158, June 1999.

[12] D. Wei, J. Yang, N. Ansari, and S. Papavassiliou, "Cell-based Schedulers with Dual-rate Grouping," *IEICE Transactions on Communications*, vol. E86-B, pp. 637–645, February 2003.

[13] J. C. R. Bennett and H. Zhang, "WF^2Q: Worst-case Fair Weighted Fair Queuing," *Proceedings of IEEE INFOCOM'96*, pp. 120–128, March 1996.

[14] A. K. Parekh and R. G. Gallager, "A Generalized Processor Sharing Approach to Flow Control in Integrated Services Networks - the Multiple Node Case," *INFOCOM'93*, vol. 2, pp. 521–530, 1993.

[15] S. Golestani, "A Self-clocked Fair Queuing Scheme for Broadband Applications," *Proceedings of IEEE INFOCOM'94*, pp. 636–646, April 1994.

[16] D. Stiliadis and V. Varma, "Design and Analysis of Frame-based Fair Queuing: a New Traffic Scheduling Algorithm for Packet-switched Networks," *Proceedings of the ACM SIGMETRICS Conference on Measurement and Modeling of Computer Systems*, pp. 104 – 115, 1996.

[17] S. Suri, G. Varghese, and G. Chandranmenon, "Leap Forward Virtual Clock," *Proceedings of INFOCOM'97*, vol. 2, pp. 557 –565, April 1997.

[18] J. Bennett, D. C. Stephens, and H. Zhang, "High Speed, Scalable, and Accurate Implementation of Packet Fair Queueing Algorithms in ATM Networks," *IEEE ICNP'97*, pp. 7–14, 1997.

[19] D. Wei, J. Yang, N. Ansari, and S. Papavassiliou, ""Guaranteeing Service Rates for Cell-based Schedulers with a Grouping Architecture"," *IEE Proceedings on Communications*, vol. 150, pp. 1–5, February 2003.

[20] J. Yang, N. Uzun, and S. Papavassiliou, "The Architecture Design for a Terabit IP Switch Router," *Proc.of IEEE HPSR'01 Workshop*, 2001.

[21] X. Wang and H. Schulzrinne, "Pricing Network Resources for Adaptive Applications in a Differentiated Services Network," *IEEE INFOCOM'01*, pp. 943–952, 2001.

[22] L. Breslau, E. W. Knightly, S. Shenker, I. Stoica, and H. Zhang, "Endpoint Admission Control: Architectural Issues and Performance," *ACM SIGCOMM Computer Communication Review*, vol. 30, pp. 69–81, August 2000.

[23] F. P. Kelly, P. B. Key, and S. Zachary, "Distributed Admission Control," *IEEE Journal on Selected Areas in Communications*, vol. 18, pp. 2617–2628, December 2000.

[24] N. Hu and P. Steenkiste, "Evaluation and Characterization of Available Bandwidth Probing Techniques," *IEEE Journal on Selected Areas in Communications*, vol. 21, pp. 879–894, August 2003.

[25] T. Timotijevic and J. Schormans, "Bandwidth Overhead of Probe Technology Guaranteeing QoS in Packet Networks," *IEE Electronics Letters*, vol. 39, pp. 816–818, May 2003.

[26] G. Bianchi, A. Capone, and C. Petrioli, "Throughput Analysis of End-to-End Measurement-Based Admission Control in IP," *Proc. of IEEE INFOCOM'00*, pp. 1461–1470, 2000.

[27] G. Bianchi, N. Blefari-Melazzi, M. Femminella, and F. Pugini, "Admission Control over Assured Forwarding PHBs: a Way to Provide Service Accuracy in a Diffserv Framework," *IEEE Globecom'01*, pp. 25–29, November 2001.

[28] P. Marbach, "Priority Service and Max-Min Fairness," *IEEE INFOCOM'02*, 2002.

[29] J. Hou, J. Yang, and S. Papavassiliou, "Integration of Pricing with Call Admission Control to Meet QoS Requirements in Cellular Networks," *IEEE Trans. On Parallel and Distributed Systems*, vol. 13, pp. 898 – 910, September 2002.

[30] S. Papavassiliou, A. Puliafito, O. Tomarchio, and J. Ye, "Mobile Agent Based Approach for Efficient Network Management and Resource Allocation: Framework and Applications," *IEEE Journal on Selected Areas in Communications*, vol. 20, pp. 858–872, May 2002.

[31] T. Zhang, E. Berg, J. Chennikara, P. Agrawal, J.-C. Chen, and T. Kodama, "Local Predictive Resource Reservation for Handoff in Multimedia Wireless IP Networks," *IEEE Journal on Selected Areas in Communications*, vol. 19, pp. 1931–1941, October 2001.

[32] Y. Bernet, P. Ford, R. Yavatkar, F. Baker, L. Zhang, M. Speer, R. Braden, B. Davie, J. Wroclawski, and E. Felstaine, "A Framework for Integrated Services Operation over Diffserv Networks," *RFC2998*, November 2000.

[33] K. Ramakrishnan and S. Floyd, "A Proposal to Add Explicit Congestion Notification ECN to IP," *RFC2481*, 1999.

[34] I. Stoica, S. Shenker, and H. Zhang, "Core-Stateless Fair Queueing: A Scalable Architecture to Approximate Fair Bandwidth Allocations in High-Speed Networks," *IEEE/ACM Transactions on Networking*, vol. 11, pp. 33–46, February 2003.

[35] S. Floyd and V. Jacobson, "Random Early Detection For Congestion Avoidance," *IEEE/ACM Transactions on Networking*, vol. 1, pp. 397–413, July 1993.

[36] G. Kesidis, J. Walrand, and C.-S. Chang, "Effective Bandwidth for Multiclass Markov Fluids and Other ATM Sources," *IEEE/ACM Transactions on Networking*, vol. 1, pp. 424–428, August 1993.

[37] C. Chang, "Stability, Queue Length, and Delay of Deterministic and Stochastic Queueing Networks," *IEEE Transactions on Automatic Control*, vol. 39, pp. 913–931, May 1994.

[38] C. Courcoubetis, V. A. Siris, and G. D. Stamoulis, "Integration of Pricing and Flow Control for Available Bit Rate Services in ATM Networks," *Proc. of IEEE Globecom'96, London, UK*, 1996.

[39] X. Liu, E. Chong, and N. Shroff, "A Framework for Opportunistic Scheduling in Wireless Networks," *Computer Networks Journal (Elsevier)*, vol. 41, no. 4, pp. 451–474, 2003.

[40] R. Chang, "Defending Against Flooding-based Distributed Denial-of-Service Attacks: a Tutorial," *IEEE Communications Magazine*, vol. 40, pp. 42–51, October 2002.

[41] C. Rensing, M. Karsten, and B. Stiller, "AAA: a Survey and a Policy-based Architecture and Framework," *IEEE Network*, vol. 16, pp. 22–27, November-December 2002.