# ACTIVE QUEUE MANAGEMENT TECHNIQUES TO IMPROVE QUALITY OF SERVICE FOR REAL-TIME FLOWS IN THIRD GENERATION WIRELESS NETWORKS

by

**JIAN CHEN**

**B.E., Shanghai Jiao Tong University, China, 1995**

A THESIS SUBMITTED IN PARTIAL FULFILMENT OF

THE REQUIREMENTS FOR THE DEGREE

MASTER OF APPLIED SCIENCE

in

THE FACULTY OF GRADUATE STUDIES

Department of Electrical and Computer Engineering

We accept this thesis as conforming to the required standard

**THE UNIVERSITY OF BRITISH COLUMBIA**

**October 2002**

In presenting this thesis in partial fulfilment of the requirements for an advanced degree at the University of British Columbia, I agree that the Library shall make it freely available for reference and study. I further agree that permission for extensive copying of this thesis for scholarly purposes may be granted by the head of my department or by his or her representatives. It is understood that copying or publication of this thesis for financial gain shall not be allowed without my written permission.

Department of Electrical and Computer Engineering.

The University of British Columbia
Vancouver, Canada

Date Oct. 10th. 2002.

DE-6 (2/88)

# Abstract

In this thesis, we analyze the behavior of real-time flows with hard time deadline using link layer retransmission to overcome wireless errors, and propose two new mechanisms to improve these flows' end-to-end Quality-of-Service (QoS) over a wireless network.

Wireless channels have the characteristic that link quality varies according to propagation conditions. For real-time flows with hard time deadlines, link layer retransmissions over a wireless network, necessitated by fluctuations in link quality, may result in a high number of packets being dropped, as deadlines expire. The expired packets waste network resources and lead to long queuing delays for subsequent packets. After analyzing the characteristics of the Radio Link Control (RLC) layer of the General Packet Radio Service (GPRS)/Universal Mobile Telecommunication Services (UMTS) network, we have developed a new set of mechanisms to minimize expiration packet drops in favour of overflow-like packet drops. We propose to use active queue management to limit transmission queue length, hence queuing delay, thus eliminating expiration packet drops. This allows the buffer and wireless bandwidth, otherwise be wasted by expiring packets, to be released earlier for other packets. We apply this mechanism to the radio link control layer in the GPRS/UMTS wireless networks. The effectiveness of the proposed mechanism is verified by simulations.

Furthermore, we extend the similar idea from the radio link control layer to the whole GPRS/UMTS domain. We adapt the Early Regulation of Unresponsive Flows (ERUF) to the characteristics of wireless channels employing link layer retransmissions. We propose to regulate those congested flows at the DiffServ domain ingress edge nodes of the GPRS/

UMTS core network, and drop undeliverable packets earlier to release some shared network resources for other flows. We also present simulation results to show that this new Wireless Early Regulation of Unresponsive Flows (WERUF) scheme can significantly improve the overall end-to-end quality-of-service.

# Table of Contents

# List of Tables

# List of Figures

# Acknowledgments

I would like to extend my sincere gratitude to my supervisor, Dr. Victor Leung, who has been a constant source of inspiration, support and guidance. But not less, I would like to thank the helpful staff of the Department of Electrical and Computer Engineering at the University of British Columbia.

Motorola Canada Ltd. in Richmond and the Canadian Natural Sciences and Engineering Research Council sponsored this research project. I would also like to express my thanks to them.

# Chapter 1 Introduction

The Internet has seen phenomenal growth in the past few years. Recent research has begun to focus on the provision of Internet service in wireless networks, as it is widely expected to be the next growth area in the Internet. In order to transmit data packets over air interfaces, factors like inferior transmission links, insufficient spectrum and limited capacity should be considered. An error control mechanism for data link layers that supports stable performance under high error rate is especially needed for wireless data transmission.

The Internet protocol (IP) is not designed to offer quality guarantees to applications, as it only provides a best-effort data delivery service. However, the Internet Engineering Task Force (IETF) [1] has proposed Integrated Services (IntServ) [2] [3] [4], as well as the Differentiated Services (DiffServ) [5], to offer diverse service levels rather than just the single best-effort class. The IntServ model uses explicit resource reservation for every flow requesting Quality-of-Service (QoS). On the other hand, the DiffServ model relies on the prioritization of some flows over others.

The Universal Mobile Telecommunications System (UMTS) architecture [6], as defined by the 3rd Generation Partnership Project (3GPP), provides the building blocks for the vast deployment of the next generation mobile Internet. UMTS promises to support a wide range of applications with different QoS profiles. In the technical specification [7], there are four different QoS classes targeting conversational, streaming, interactive and background traffic. The first two classes are defined for real-time applications. Conversational class serves the traffic with the most stringent transfer delay and delay variation requirements, such as Voice over IP. The Stream class serves real-time traffic with relatively loose delay requirement, such as streaming video. The latter two QoS classes typically serve traffic without bounded transfer delay requirements. The most popular type of traffic is the Transmission Control Protocol (TCP) [8]. The promised QoS diversity is adequate to support future mobile Internet applications. Expected situations rely on the proper inter-working of UMTS and IP networks. DiffServ architecture is one of the mechanisms suggested for use in the UMTS core network (CN) to offer interoperability at the QoS level.

The provision of end-to-end QoS not only depends on the core network, but also on wireless links. Different resource management schemes must be deployed at the wireless-wireline interface nodes so as to cater to all requirements of QoS. Our proposed idea is one such scheme.

We focus our efforts on link layer retransmission schemes to overcome wireless errors, especially for real-time applications. Wireless links are error-prone channels. A lot of effort has been made to attack this problem. The two main classes of techniques proposed for this problem are Forward Error Correction (FEC), and link layer retransmission of lost packets in response to Automatic Repeat Request (ARQ) messages [9] [10]. They can either be separately implemented, or combined to get the best performance. For example, in a hybrid ARQ scheme [11], when the number of errors in a received packet is lower than the error-correcting capacity of the configured FEC code, this packet is accepted. Otherwise it is discarded and ARQ is triggered to request a retransmission.

As suggested by the 3GPP specification [7], for real-time applications with low and stringent transfer delay requirements, FEC is normally used as the only error correcting scheme because ARQ may be too time-consuming. Conversational class traffic falls into this category. For real-time traffic where transfer delay requirements are relatively lax, such as Streaming class applications for example, ARQ or the Hybrid ARQ can be used to reduce the bit error rate. We concentrate on the latter in our research work.

This chapter briefly introduces some background information, then presents the motivation and an overview of our concept. Further details are illustrated in following chapters.

## 1.1 Background Knowledge

### 1.1.1 General Architecture of GPRS Domain

General Packet Radio Service (GPRS) [12] is originally proposed for Global System for Mobile communication (GSM) data service. 3GPP adopts it as the common standard packet domain core networks for UMTS.

GPRS uses a packet-mode technique to transfer high-speed and low-speed data and signalling in an efficient manner. It is designed to support transfers from intermittent and bursty data through to occasional transmission of large volumes of data. It supports QoS and other features.

Figure 1.1 presents the interfaces and network nodes relating to the topics of this thesis.

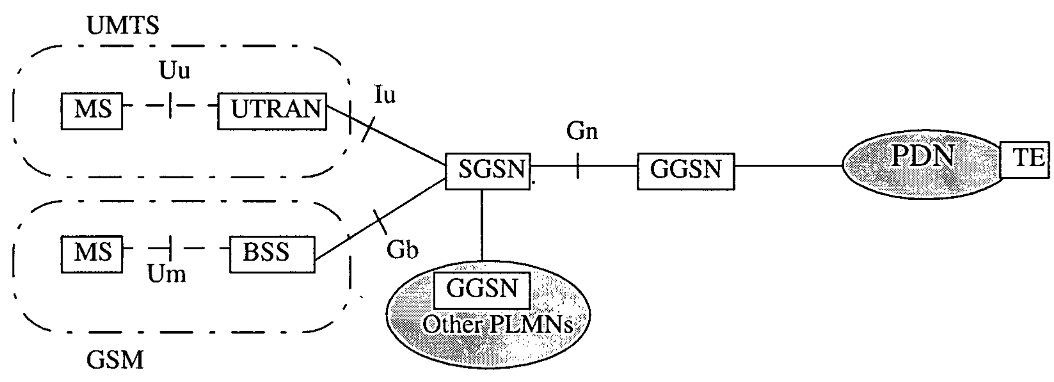


Figure 1.1 Overview of the GPRS packet domain architecture.

As we can see, the GPRS backbone can serve both UMTS and GSM access networks. GPRS introduces two new network nodes in the Public Land Mobile Network (PLMN). The Serving GPRS Support Node (SGSN) keeps track of the individual Mobile Station's (MS) location and performs security functions and access control. The SGSN is connected to the GSM base station system through the Gb interface and/or to the UMTS Radio Access Network through the Iu interface. The Gateway GSN (GGSN) provides interworking with external packet-switched networks, and is connected with SGSNs via an IP-based packet domain PLMN backbone network.

The meanings of other elements and interfaces are explained below.

Network nodes:

- MS - Mobile Station. In this thesis we use this term or the word "mobile" to refer to both the Mobile Terminal (MT) and Terminal Equipment (TE) at the end user terminal site.
- UTRAN - UMTS Terrestrial Radio Access Network. (UMTS only)
- BSS - Base Station System. (GSM only)
- PDN - Packet Data Network. We refer here to an external data network (Internet or others), either with QoS guarantees, or just best effort.
- TE - Terminal Equipment. We refer here to an Internet host or another terminal communicating with the MS.

Interfaces:

- Iu and Gb - The interfaces connecting SGSN with UTRAN in UMTS and BSS in GSM respectively.
- Gn - Interface between two GSNs within a PLMN.
- Uu and Um - The wireless interfaces between mobile stations (MS) and the GSM or UMTS fixed network part.

From Figure 1.1, we can see that an SGSN can connect to more than one GGSN even if they are in different PLMNs. Actually, each GGSN can also serve more than one SGSN. Other network elements and interfaces are specified in the technical documents. The figure does not show them because they are unrelated to our topic. The link layer retransmission scheme for overcoming wireless errors resides in the wireless interfaces Um and Uu.

### 1.1.2 Some Mechanisms for QoS

The mechanisms introduced below basically demonstrate the impetus behind our idea. In the following chapters they are discussed in detail.

Random Early Detection (RED) [13] is one of the most popularly used Active Queue Management (AQM) methods. It actively drops/marks packets in the aggregation of flows at core routers by a probability proportional to the average queue length. At first, this method targets the TCP/IP network, in which a single marked or dropped packet is sufficient to signal the presence of congestion to the transport-layer protocol. The emphasis on avoiding the global synchronization, which results when many connections reduce their windows at the same time, is particularly relevant in a network with TCP [8] where each connection goes through Slow-Start, reducing the window to one in response to a dropped packet. With the growing number of Internet applications relying on unresponsive protocols, with little or no end-to-end congestion control, RED is proposed for use in core routers in order to avoid unfairness and congestion collapses in the Internet [14] [15]. The general idea is similar to the TCP case, except with a new functionality to measure the unresponsive flows link share so that the "misbehaved" flows can be identified and regulated. The regulation may be implemented by differentially scheduling the packets from the identified flows, or by dropping the packets from the flows at the router.

The reference [16] proposed to extend such idea as Early Regulation of Unresponsive Flows (ERUF). The general concept deals with cases when unresponsive flows encounter network congestion at certain nodes over the end-to-end path, thus destining some packets to be dropped. The author in [16] argues that dropping packets at an earlier node (before reaching the shared core routers) over the same path frees shared network resources for other flows, with the assumption that network resources assigned to certain traffic aggregates are fixed. The author in [16] proposes a notification scheme to inform the core routers about the congestion status, and an effective regulation method at earlier nodes upon the signalling.

The regulation method relies on shaping elements in flow control. Such elements also exist in the DiffServ architecture [17]. There are many other functional elements in DiffServ. We introduce them in the next chapter. The main advantage of the DiffServ model is the minimal state information needed in the backbone. The aggregation of traffic performed in DiffServ architecture ensures high scalability.

## 1.2 Motivation

Because wireless links in PLMN are subject to noticeable variations in received signal strength due to local variations in terrain, buildings, and foliage, in order to ensure high reliability of data communications, it is necessary to resort to an error-control method to eliminate transmission errors caused by the channel noise. As we mentioned above, ARQ and FEC are the main methods used for this purpose.

Most real-time applications require hard time deadlines. If packets in a real-time flow can not reach their destinations before their deadlines, they become useless. When wireless links experience high bit error rate (BER), many packets accumulate at the link layer transmission buffer, or queue, residing at wireless-wireline interface nodes. Note that, in this thesis, we assume that packets in the transmission buffer form a first-in-first-out queue, except for some higher priority packets carrying control information. So the term "queue length" also refers to buffer occupancy in this thesis. If the queue length grows to a large enough value, the end-to-end latency is dominated by the queuing delay, which leads to many real-time packets being dropped due to deadline expiration. In this case, packet expiration caused by queuing delay becomes a critical factor degrading overall performance.

Such phenomena inspire the application of the AQM to the transmission buffer. Intuitively, packets in real-time flows expire within a certain probability relating to queue length. Longer queue lengths generate more queuing delay, and reduce the margin to the delivery deadline. Actively dropping packets with probabilities analogous to such "natural" dropping rates due to expiry may result in releasing network resources earlier, thereby reducing the queuing delay for following packets. With shorter queuing delays, the expiring probability of the remaining packets is largely reduced. If the active drop rate is kept lower than the "natural" one, we can guarantee better system performance.

## 1.3 Contributions

With this motivation, we set out to show the effectiveness of our idea both by analysis and by simulation. There are many aspects to be resolved in the implementation of such a new mechanism. They are presented one by one in the following chapters.

The main goals of our research work are:

- Based on the GPRS/UMTS models and an analysis of transmission queue behavior, mathematically express of the relation between transmission queue length and packets' natural drop rate due to expiry in real-time flows with hard deadlines. With the analysis results, demonstrate by simulation of the effectiveness of applying AQM to drop real-time packets earlier to release network resources in wireless access nodes.
- A further proposal and demonstration that the ERUF can also be applied to the transmission queue at wireless access nodes so that, not only can network resources at these nodes be freed earlier for other packets in the same flow, but also the network resources in the shared core network can be released earlier for other flows. The mechanism is realized by the DiffServ functional elements in GPRS/UMTS core network.

As far as we know, no publication has combined the link layer transmission queue with the AQM. Also no one has yet used link layer transmission buffer occupancy as an indicator to signal the GPRS/UMTS core network regarding the congestion status caused by wireless errors. The importance of applying appropriate buffer management schemes to the link layer transmission buffer has not been widely realized. We hope our work can attract more industrial and academic research interest to this area.

## 1.4 Outline of the Thesis

In Chapter 2, we introduce some previous work relating to end-to-end QoS in the GPRS/

UMTS wireless network. First, an overview of GPRS/UMTS network architecture is discussed. Second, we show the mechanisms for the provision of end-to-end QoS over GPRS/UMTS core networks. Third, we introduce the techniques used in GPRS/UMTS wireless access nodes catering to the end-to-end QoS requirement. Chapter 3 analyzes the behavior of link layer transmissions in a GPRS/UMTS specification, and derives the mathematical expression of the real-time packets expiration rate. This chapter also presents our idea of applying AQM to the link layer transmission queue, and the simulation results that verify its effectiveness. Chapter 4 extends the idea of ERUF by applying it to the link layer transmission queue in wireless access nodes and the GPRS/UMTS core network in combination so that the overall performance can be improved, and demonstrates its effectiveness by simulation. Chapter 5 gives our conclusion and future work.

# Chapter 2 Overview of Previous Work

In this chapter, we introduce some previous work relating to the provision of end-to-end QoS in the GPRS/UMTS network.

## 2.1 End-to-end QoS in GPRS/UMTS Architecture

3GPP defines four QoS classes: Conversational, Streaming, Interactive and Background. Each QoS class is specified by a QoS profile associated with the Packet Data Protocol (PDP) context.

UMTS definition divides the protocol stacks into a Control Plane and a User Plane. The User Plane is the layered protocol structure providing the user information transfer and the associated transfer control procedures such as flow control and error recovery. The Control Plane supplies support to the User Plane. We focus on the User Plane protocol stacks illustrated in Figure 2.1. The protocols relating to our research work are shaded in the figure, while others are

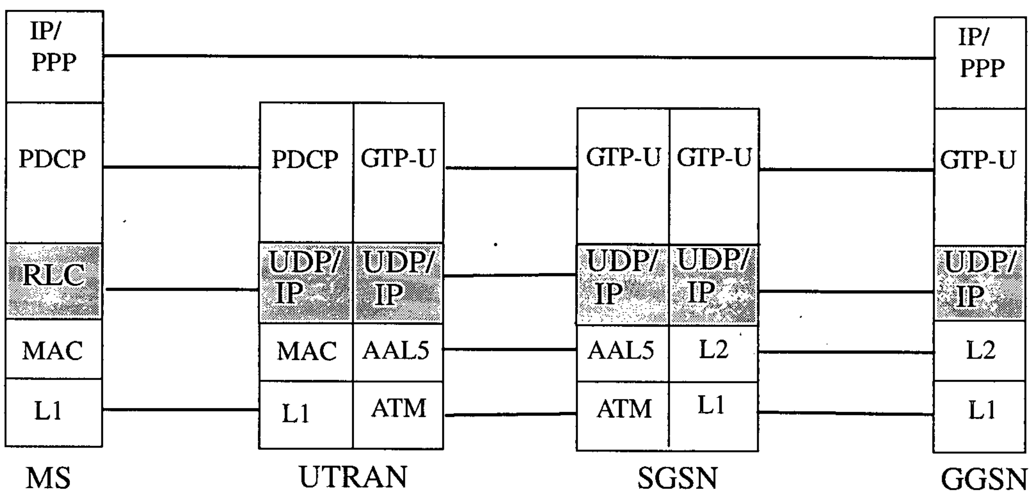


Figure 2.1 Protocol stacks of the UMTS user plane.

ignored in this thesis.

User Datagram Protocol/Internet Protocol (UDP/IP) [8] are the backbone network protocols used for routing user data and control signalling. The QoS management functions are also located at this layer. The Radio Link Control (RLC) [18] layer protocol provides logical link control over the radio interface. The link layer retransmission is one of its working modes.

In order to achieve the differentiation of the four classes' end-to-end QoS, both the core network and the radio access network must be involved. Compare Figure 2.2 with Figure 2.1. In

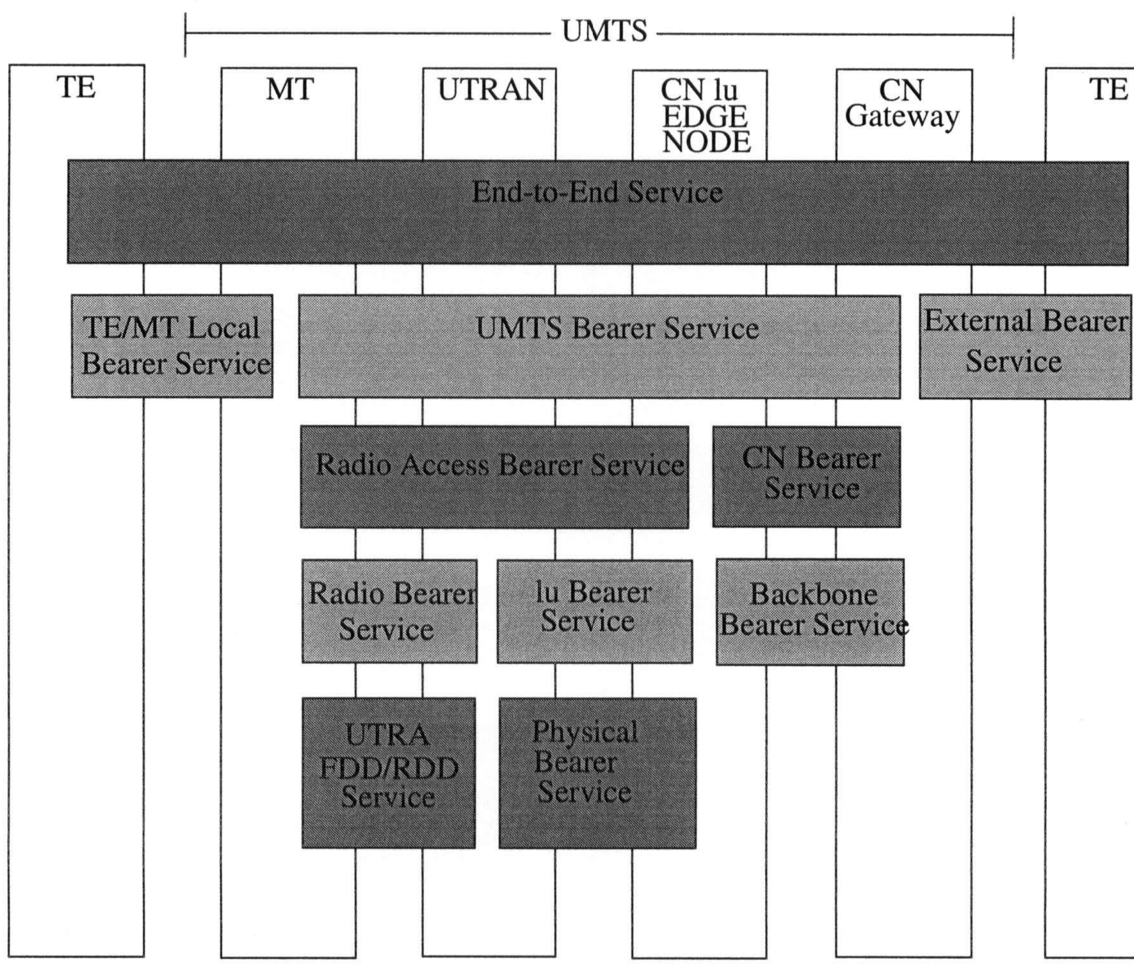


Figure 2.2 GPRS/UMTS QoS architecture.

Figure 2.2, the Terminal Equipment (TE) and Mobile Terminal (MT) form the Mobile Station (MS) in Figure 2.1. The CN Iu edge node and CN Gateway correspond to the SGSN and GGSN in Figure 2.1, respectively. In this thesis, we elaborate on the UMTS Bearer Service to improve the end-to-end QoS. The other bearer services under the UMTS Bearer Service provide the necessary lower layer functions.

## 2.2 GPRS/UMTS Core Network Mechanisms for QoS

The 3GPP technical specification [7] suggests the implementation of the IntServ or DiffServ architectures in the GPRS/UMTS core network to realize QoS management. We introduce these IETF mechanisms in this section, with emphasis on DiffServ and its mapping to the GPRS/UMTS domain. DiffServ architecture includes many functional elements. We emphasize Active Queue Management (AQM) at the droppers, because AQM is the most important buffer management scheme at the core routers, which we shall apply to the data link layer transmission buffer at wireless-wireline interface nodes in this thesis.

### 2.2.1 IETF QoS Architectures

The existing Internet service (namely, the best-effort service of IP) cannot satisfy the QoS requirements of emerging multimedia applications, primarily because of variable queuing delays and packet loss during network congestion. There has been a significant amount of work in the past decade on extending the Internet architecture and protocols to provide QoS support for multimedia applications. This has led to the development of a number of service models and mechanisms as suggested by IETF.

The IntServ model is proposed as an extension to support real-time applications. The key here is to provide some control over end-to-end packet delays in order to meet real-time QoS. The fundamental assumption of the IntServ model is that resources (e.g., bandwidth and buffer) must be explicitly managed for each real-time flow. This requires routers to reserve resources in order to provide specific QoS for packet streams, or flows, which in turn requires flow-specific states in each router. Such mechanisms usually need to maintain states, manage buffers, and/or perform packet scheduling on a per flow basis. This complexity may prevent them from being cost-effectively implemented and widely deployed.

The DiffServ architecture is proposed in [5] to not only supply the QoS differentiation, but also to achieve scalability.

The Diffserv architecture is based on a simple model where traffic entering a network is classified and possibly conditioned at the boundaries of the network [19] [20], and assigned to different behavior aggregates (BAs), with each BA being identified by a single DiffServ codepoint (DSCP). Users request a specific performance level on a packet-by-packet basis, by marking the DiffServ field of each packet with a specific value. This value specifies the per-hop behavior (PHB) to be allotted to the packet within the provider's network. Within the core of the network, packets are forwarded according to the PHB associated with the DSCP.

Sophisticated classification, metering, marking, policing, and shaping operations only need to be implemented at network boundaries or hosts. Multi-field (MF) classifiers select packets based on the content of some arbitrary number of header fields, typically some combination of source address, destination address, DS field, protocol ID, source port and destination port. Markers are devices that perform marking. Shapers are devices that perform shaping, which refers to the process of delaying packets within a traffic stream to cause it to conform to some defined traffic profile. Metering is the process of measuring the temporal properties (e.g., rate) of a traffic stream selected by a classifier. Policing is the process of discarding packets (by a dropper) within a traffic stream in accordance with the state of a corresponding meter enforcing a traffic profile. Network resources are allocated to traffic streams by service provisioning policies which govern how traffic is marked and conditioned upon entry to a DiffServ-capable network, and how this traffic is forwarded within that network. A wide variety of services can be implemented on top of these building blocks. Such traffic control functions at hosts, or access or boundary routers are generically called traffic conditioning.

A salient feature of the DiffServ framework is its scalability, which allows it to be deployed in very large networks. This scalability is achieved by forcing much complexity out of the core of the network into boundary devices which process smaller volumes of traffic and fewer flows, and by offering services for aggregated traffic rather than on a per-microflow basis. That is, complex traffic classification functions are only implemented at network boundary nodes; inside the core network, PHBs are applied to aggregates of traffic which have been appropriately marked using the Diffserv field in the IP headers. PHBs are defined to permit a reasonably granular means of allocating buffer and bandwidth resources at each node among competing traffic streams. Per-application flow or per-user forwarding state need not be maintained within the core of the network.

In this thesis, we assume one of the IETF-suggested PHBs, Assured Forwarding (AF) [21], is implemented in the core network of the GPRS/UMTS domain. The Assured Forwarding (AF) PHB group constitutes a means for a provider DiffServ domain to offer different levels of forwarding assurances for IP packets received from a customer DiffServ domain. Four AF classes are defined, where each AF class in each DiffServ node is allocated a certain amount of forwarding resources (buffer space and bandwidth). IP packets that seek to use the services provided by the AF PHB group are assigned by the customer or the provider DiffServ domain into one or more of these AF classes according to the services that the customer has subscribed to. Within each AF class, IP packets are marked (again by the customer or the provider DiffServ domain) with one of three possible drop precedence values. In cases of congestion, the drop precedence of a packet determines the relative importance of the packet within the AF class. A congested DiffServ node tries to protect packets with a lower drop precedence value from being lost by preferably discarding packets with a higher drop precedence value.

In a DiffServ node, the level of forwarding assurance in an IP packet thus depends on (1) what amount of forwarding resources are allocated to the AF class that the packet belongs to, (2) what the current load of the AF class is, and, (3) in case of congestion within the class, what the drop precedence of the packet is.

In order for a user to receive DiffServ from Internet service provider (ISP), it must have a service-level agreement (SLA) with its ISP. An SLA basically specifies the service classes supported and the amount of traffic allowed in each class, respectively. Users can mark DiffServ fields of individual packets to indicate the desired service at hosts, or have them marked by the access or boundary routers. At the ingress of the ISP networks, packets are classified, policed, and possibly shaped. The classification, policing, and shaping rules used at the ingress routers are derived from the SLAs. When a packet enters one domain from another, its DS field may be remarked, as determined by the SLA between the two domains.

### 2.2.2 Mapping between GPRS/UMTS and DiffServ

Ref. [7] illustrates the QoS management functions in GPRS/UMTS. The traffic conditioning functions reside in the core network gateways and the UTRAN. It is very reasonable to treat the GPRS/UMTS network as a special DiffServ domain. In this domain, GGSN and UTRAN can be regarded as the DiffServ edge nodes. GGSN connects the GPRS network to the Internet through wired links, while UTRANs connect the GPRS network to mobile users through wireless links. Similarly, SGSN can be considered to be the core router. Figure 2.3 presents the mapping of

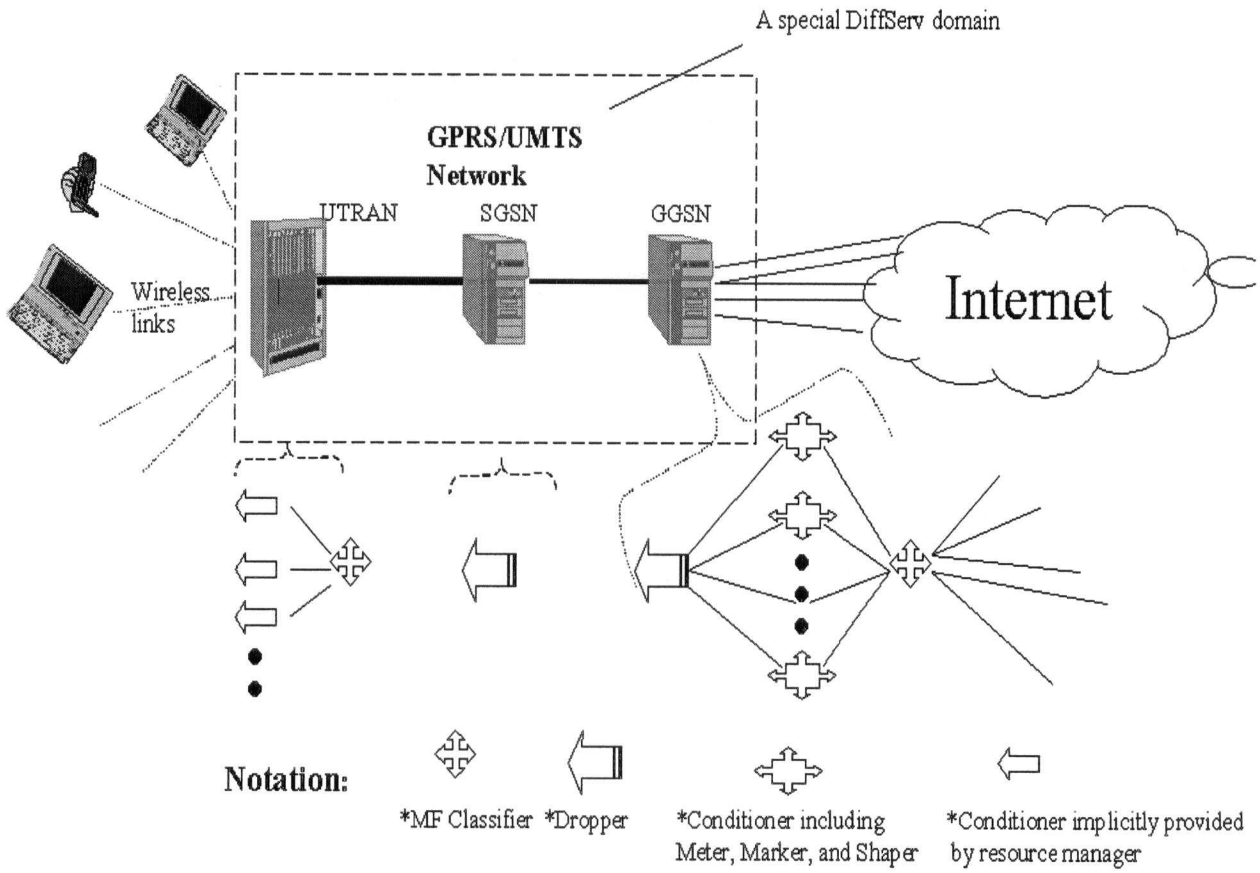


Figure 2.3 Mapping of the DiffServ domain and the GPRS/UMTS network.

these two architectures.

It is worth mentioning that the traffic conditioners at GGSN must be implemented explicitly, while at UTRAN they can be either implemented explicitly or realized by wireless link resource control functions. The traffic conditioners in GGSN include functional elements such as MF classifiers, meters, markers, shapers and droppers. Because SGSN is treated as the core router, only droppers are implemented.

The droppers are used for buffer management as well as differentiated QoS. An AF implementation must detect and respond to long-term congestion within each class by dropping packets, while handling short-term congestion (packet bursts) by queueing packets. This implies the presence of a smoothing or filtering function that monitors the instantaneous congestion level and computes a smoothed congestion level. The dropping algorithm uses this smoothed congestion level to determine when packets should be discarded. The most popularly used dropper scheme is the RED we introduce below.

### 2.2.3 Active Queue Management

Active Queue Management (AQM) was originally proposed to handle TCP-like flows. The TCP transport protocol detects congestion only after a packet has been dropped at the gateway. However, it is clearly undesirable to have large queues (possibly on the order of a delay-bandwidth product) that are full much of the time; this would significantly increase the average delay in the network. Therefore, with increasingly high-speed networks, it is increasingly important to have mechanisms that keep throughput high but average queue sizes low. The RED gateway is designed for a network where a single dropped packet is sufficient to signal the presence of congestion to the transport-layer protocol.

The RED gateway calculates the average queue size *avg* (in this thesis, the unit of queue length or buffer occupancy is always in packets), using a low-pass filter with an exponential weighted moving average,

$$avg = (1 - w_q)\, avg + w_q\, q \tag{2.1}$$

where the $w_q$ is the weight, which is normally 0.002 as [22] proposes, and $q$ is the current queue size.

The average queue size $avg$ is compared to two thresholds, a minimal threshold $min_{th}$ and a maximal threshold $max_{th}$. When the average queue size is less than the minimal threshold, no packets are marked. When the average queue size is greater than the maximal threshold, every arriving packet is marked. If marked packets are in fact dropped, or if all source nodes are cooperative, this ensures that the average queue size does not significantly exceed the maximum threshold. As $avg$ varies from $min_{th}$ to $max_{th}$, the packet-marking probability $p_a$ varies linearly from 0 to $max_p$,

$$\begin{aligned} p_b &= max_p\, (avg - min_{th}) / (max_{th} - min_{th}) \\ p_a &= p_b / (1 - count * p_b) \end{aligned} \tag{2.2}$$

where $count$ is the packets since the last marked packet.

Ref. [23] proposed Random Early Detection with IN-and-OUT (RIO). It is a simple but effective scheme to supply differentiated services. The general idea is to mark the packets that comply with the SLA as IN packets, otherwise as OUT packets, at ingress edge nodes. After that different sets of $max_p$, $min_p$ and $max_p$ are applied to the IN and OUT packet queues so as to generate different packet drop/mark precedences.

The general algorithm for the RED gateway is presented in Figure 2.4. We use the term "mark" here because the TCP flows can carry the congestion information back to the source, so their packets may not need to be dropped immediately.

**for each packet arrival**

    **calculate the average queue size** $avg$

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

        **calculate probability** $p_a$

        **with probability** $p_a$**:**

            **mark the arriving packet**

    **else if** $max_{th} \leq avg$

        **mark the arriving packet**

Figure 2.4 General algorithm for RED gateways.

The need for and advantages of limiting unresponsive best-effort traffic has been proven in [15]. Unresponsive flows are flows that do not use end-to-end congestion control, and in particular those that do not reduce their load on the network when subjected to packet drops/marks. Without knowledge of the congestion ahead, unresponsive traffic sources keep sending packets, whereas many data packets are actually undeliverable due to packet expiration and possible overflow. These undeliverable packets waste shared network resources. To solve the problem, the Early Regulation of Unresponsive Flows (ERUF) scheme has been proposed [16]. This concept dictates that when unresponsive flows encounter network congestion at certain nodes over the end-to-end path, and cause some packets to miss their delivery deadlines or to be dropped due to buffer overflow, dropping them at an earlier node frees the shared network resources for other flows. Source quench packets are generated by the RED algorithm, and sent from the congestion point to the earlier nodes. The earlier nodes regulate the bandwidth in a multiplicative-decrease/ additive-increase manner over a round trip time (RTT) estimate.

In order to get better performance, gentle-RED [24] was proposed for application to the high-speed network. The difference between the RED and gentle-RED is their marking rate variation curves, which are displayed in Figure 2.5. The $p_b$ are equal when the average queue

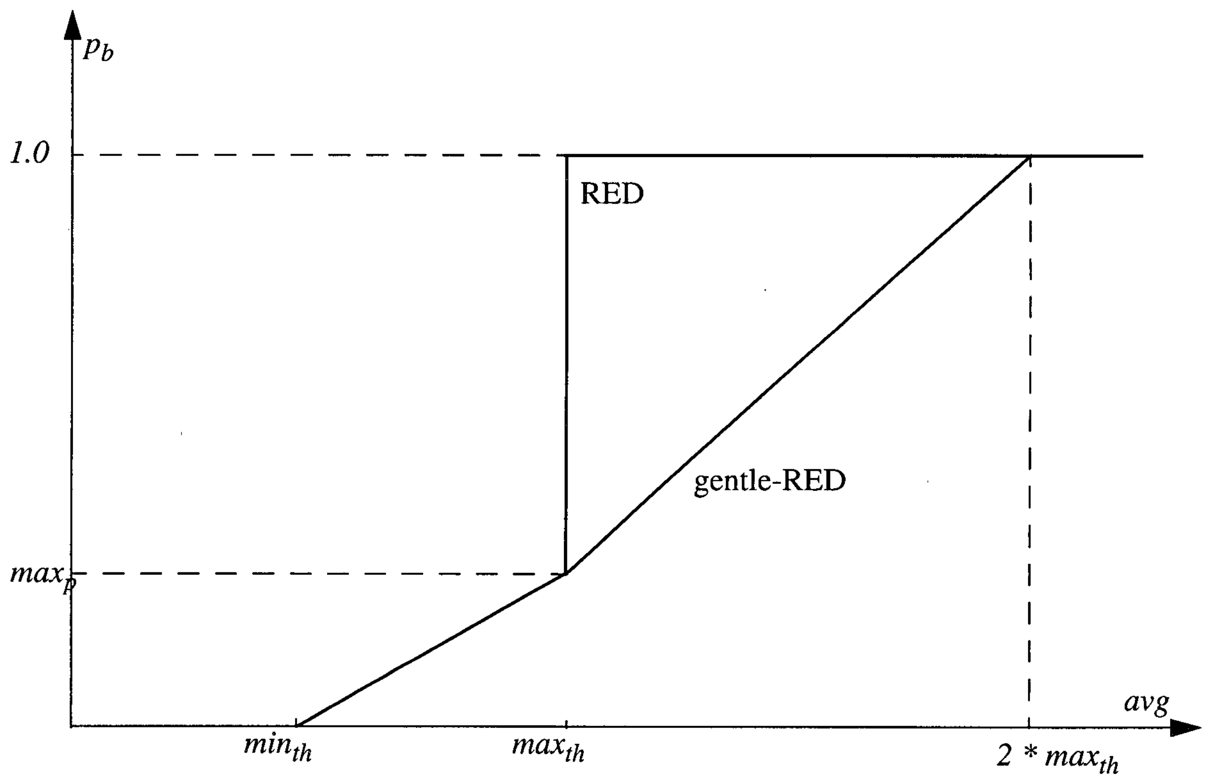


Figure 2.5 RED and Gentle-RED marking/dropping rate *vs.* average queue length.

length is less than $max_{th}$. Once the *avg* crosses $max_{th}$, the normal RED makes $P_b$ 1.0 immediately, while the gentle-RED continues to increase $p_b$ with a slope leading the marking rate curve to 1.0 at the point of $avg = 2 * max_{th}$. Again, the gentle-RED is originally proposed for TCP-like traffic, which is responsive to single packet drops/marks. In this thesis, we make necessary modifications to handle the unresponsive traffic.

## 2.3 Mechanisms at Wireless Access Nodes for QoS

In order to offer end-to-end QoS guarantees and differentiation to end users, not only the core network, but also the wireless access nodes of GPRS/UMTS must be taken into account. Much previous research work has been done on the mechanisms at the access nodes. FEC and link layer ARQ are used to attack the high BER of wireless links. Different scheduling and resource management schemes are proposed for the sake of provisioning the QoS, or utilizing resources (normally bandwidth and buffer) efficiently. In this section, we focus on the link layer ARQ and resource management, and present general knowledge of other mechanisms.

### 2.3.1 Radio Link Control (RLC) Layer in GPRS/UMTS

Wireless links are error-prone with relatively high and fluctuating bit-error rates (BERs). Different methods have been developed to mitigate transmission errors so as to deliver data packets to mobile users with guaranteed low BER. As we mentioned above, two main classes of techniques are FEC coding and link layer retransmissions of lost packets using ARQ protocols. For those real-time applications with low and stringent transfer delay requirements, FEC is normally used as the only error mitigation scheme, because ARQ may consume too much time. For real-time traffic where transfer delay requirements are relatively lax, for example, most applications in the Streaming class, ARQ or hybrid ARQ can be used. In our research, we assume the FEC code is fixed during the lifetime of a connection.

In the GPRS/UMTS networks, link layer retransmissions occur at the RLC layer residing at the wire-wireless interface nodes, that is, the Radio Network Controller (RNC) in UMTS networks. Our investigation focuses on the Radio Link Control (RLC) layer in Acknowledge Mode (AM) (Figure 2.6). It is actually a kind of sliding-window / multiple-reject ARQ. In the

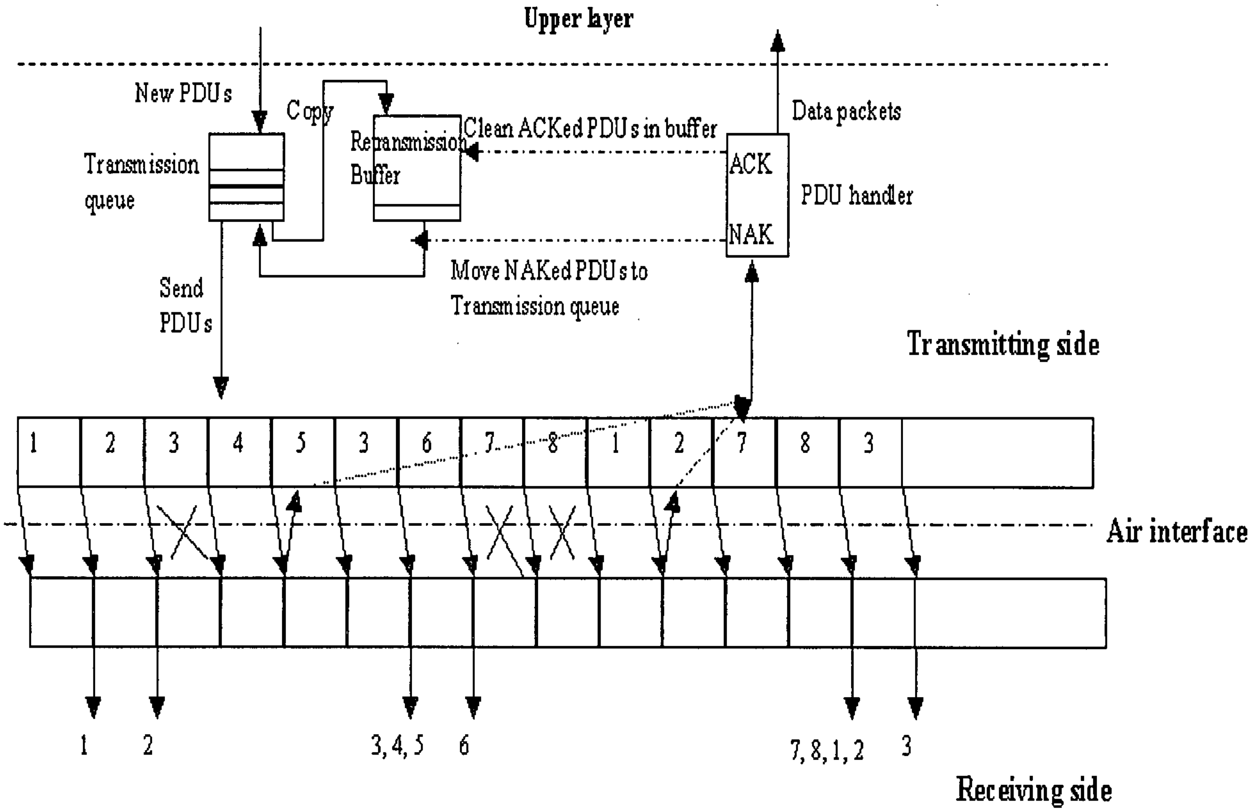


Figure 2.6 Acknowledged Mode in Radio Link Control layer (window size = 8).

model used in our research, we assume that the receiver must deliver the in-sequence packets to the higher layers.

RLC protocol does the segmentation and reassembly. After segmentation, the higher layer data segments are encapsulated into the RLC layer's Protocol Data Units (PDU). The RLC layer has a buffer holding newly arriving PDUs and outstanding PDUs which have been sent but not been Acknowledged (ACK). A PDU is deleted from this buffer if it is ACKed by a status report from the receiver, or transmitted more than a certain number of times. Senders can also actively set a "polling bit" in the PDU header to request a status report from receivers. This action can be triggered in different ways, for example, by a timer time-out, or using a window based rule, or when the last PDU remained in buffer, and so on. We carefully model the "Poll_PDU"-based and the timer-based trigger functions. The former is a value set by the system. Every time a sender sends out Poll_PDU PDUs (including newly transmitted and retransmitted PDUs), a polling bit is set on the next PDU's header. The latter is merely a timer that periodically triggers the polling.

The receiver reports on status by sending a report using Super-Fields (SUFI). There are several situations that can trigger the receiver to send a status report. For example, if the sender sends a polling request, a status report is triggered. Another situation is if the receiver detects one or more PDU missing. A third way is if the receiver transmits status reports periodically to the sender. A forth trigger occurs if not all PDUs requested for retransmission have been received before the Estimated PDU Counter (EPC) has expired. Finally, either peer entity can trigger the RESET function to reset the wireless connection when it detects an extremely abnormal situation on the wireless link, for example, a very long idle time or other phenomena, indicating high error rates.

If the transmission queue is full, an arriving Acknowledged Mode Data (AMD) PDU gets dropped. Otherwise, it joins the tail of the transmission queue waiting for its first transmission. Once it moves to the front, the RLC stores a copy of it in the retransmission buffer prior to passing it to the lower layer for transmission.

The receiver accepts the "good" PDUs and discards PDUs with more errors than the FEC code can correct. It generates status reports to request the sender to resend the lost PDUs that have not expired. RLC defines the multiple reject ARQ scheme, in which every status report not only contains NAKs, but also ACKs. Status reports can be sent as dedicated control PDUs or piggybacked in data PDUs over the return link. The control PDUs are put into the transmission queue with higher priority. In Figure 2.6, we assume these actions are incorporated in the "PDU handler" unit.

The sender keeps sending PDUs in its transmission queue until it receives a status report or reaches the sending window boundary. Once a status report arrives, the RLC takes two actions according to the ACKs and NAKs received: delete ACKed PDUs from the retransmission buffer, and copy NAKed PDUs from the retransmission buffer to the transmission queue with higher priority than the new (first time transmission) data PDUs. This retransmitted PDU gets dropped when its number of transmissions reaches a pre-configured maximal value, which is called MaxDAT in the 3GPP document [18].

### 2.3.2 Other Mechanisms at Wireless Access Nodes

The wireless-wireline interface nodes may deploy a number of resource management mechanisms, including traffic scheduling, call admission control (CAC) / bandwidth reservation, and so on. These mechanisms are not defined in the 3GPP documents, instead, they are implementation dependent.

CAC and bandwidth reservation are used to address performance degradation due to mobile handoff. When a mobile changes cells, the new cell may not guarantee supplying enough network resources to keep the same QoS as the old cell, so the mobile may experience performance degradation during the handoff period [25] [26]. CAC and bandwidth reservation seek to estimate the amount of resources needed by the incoming mobiles, and make admission decisions to reduce the performance degradation or handoff dropping rates. Another kind of bandwidth reservation occurs when multiple users try to get access to the same media.

Packet scheduling at the wireless-wireline interface nodes is important for QoS provision. Much research work has been done in this area, for example, the Weighted Fair Queuing (WFQ) [27] variances such as Effort-Limited Fair (ELF) scheduling [28] and the Idealized Wireless Fair-Queuing (IWFQ)[29], the extended Earliest-Due-Date (EDD) / Delay-Earliest-Due-Date (Delay-EDD) for wireless [30], and so on. From the viewpoint of implementation, the EDD mechanisms need to check every packet's time stamp, thus it is hard to realize in reality. The WFQ variances guarantee delay bounds on weight tightly coupled to a preserved service rate, so it is relatively easier to implement. However, to determine a weight matching a specific delay requirement is still quite complex, because a number of factors such as propagation delay, queuing delay, packet arrival rate, and so on, have to be counted for.

In order to simplify our research model, we assume Media Access Control (MAC) protocols work perfectly to eliminate the negative impact of all of the above mentioned competition for resources. We make the assumption that both the assigned bandwidth (or the service data rate) of a real-time flow and the time consumed by the MAC layer are constant during the lifetime of a connection. The impact of the above factors on our proposed mechanisms can be further investigated in the future.

The buffer management issue at wireless-wireline interface nodes has not attracted much research interest. So far, all researchers assume that, at the link layer, drop-tail is used as the default queue management scheme. Under the drop-tail assumption, [31] [32] [33] estimate the effects of throughput and the buffer overflow dropping process on buffer capacity when the link layer ARQ is employed, but they neither consider the delay of packets, nor target the buffer management issue from the viewpoint of active queue management.

Our analysis in the next chapter shows that if a real-time flow with a hard time deadline uses link layer ARQ, when the transmission queue length grows too long due to retransmissions caused by wireless error, the queuing delay dominates the other delays, causing a great deal of expired packet drops.

Our proposal is that AQM not only be applied to traffic aggregates at the GPRS/UMTS core routers, but also to the link layer buffer at the wireless access nodes, namely, the DiffServ egress edge nodes, to control the queuing delay caused by wireless errors.

In the following chapters, we analyze the behavior of the link layer ARQ, and derive the necessary mathematical expression of the packet expiration rate on queue length with the link layer ARQ. We also prove that introducing AQM into the link layer transmission buffer significantly reduce packet dropping due to expiry, and improves the end-to-end QoS.

# Chapter 3 Applying AQM to Link Layer Buffers for Real-time Traffic

In Chapter 2, we observed that the possible causes of a real-time flow PDU drop are (1), the number of transmissions exceeding MaxDAT, (2), transmission queue overflow, and (3), packet AMD PDU expiry.

The first point is fully determined by the transient wireless bit error rate (BER), which is not a focus of this thesis. Points two and three are also caused by wireless link quality degradation, but they can be eliminated by appropriately setting the transmission queue capacity. If we set the capacity to a small enough value, expirations disappear, but overflow is very likely to occur. If the queue capacity is set to a large enough value, the situation reverses. This gives us a chance to improve the end-to-end performance of real-time flows.

In the above cases causing packet drops, we believe the dropping of real-time packets through overflow has more advantages than by expiration. With overflow, a newly arriving AMD PDU does not have a chance to consume buffer space and wireless bandwidth at the interface node. Dropping a packet this way does not affect the transfer of other packets in the same flow and other flows sharing the wireless channel. In contrast, such is not the case for expiration drops, as the packets have to be sent, possibly several times, until they are received - before expiration dropping occurs. We propose to use active queue management, such as Random Early Detection (RED), to drop those packets that are likely to expire at a much earlier point in the end-to-end connection, using an overflow-like mechanism. Because the packets are dropped before being enqueued to the buffer, they need not consume any network resources. However, if the packets are dropped at a rate higher than the natural dropping rate due to expiry, the system's performance degrades unnecessarily. In order to avoid this problem, we need an analytic expression of the natural drop rate due to expiry, and to adjust our active drop rate to be lower than but close to it.

## 3.1 Analysis of Real-time Flows over RLC with AM

We assume a real-time flow with effective service data rate $R$. Each link layer packet with fixed size $l$ takes $l/R$ seconds for one transmission. In an extreme case, such as where the wireless link is clean enough to transmit all PDUs successfully the first time, the upper bound of the transmission queue length that guarantees expiration packet drop is approximated thus,

$$L_{max} = \left\lceil \frac{(T - t1 - t2) \cdot R}{l} \right\rceil \tag{3.1}$$

where $T$ is the maximum allowable transfer delay of the real-time application, $t1$ is the latency caused by the Internet, and $t2$ is the sum of the one-way wireless propagation delay and processing/scheduling delays at both ends. When a new packet arrives at the transmission queue, if the observed queue length is longer than $L_{max}$, then even if the wireless link is clean, this packet is still dropped due to expiration. A queue capacity larger than $L_{max}$ is clearly not useful.

If the wireless link is so noisy that every PDU has to be transmitted $N$ times before being accepted or dropped, where $N$ is the maximum allowable number of transmission, every packet approximately spends

$$t3 = N \cdot \frac{l}{R} \tag{3.2}$$

in the transmission queue waiting for its transmissions to complete, during which following packets have to wait.

On the other hand, in this situation every packet also has to spend approximately

$$t4 = (N-1) \cdot t2 \cdot 2 \tag{3.3}$$

time in the retransmission buffer waiting for an ACK or NAK to come from the receiver, during which time it does not have any impact on the transmission of other packets. The factor 2 takes into account round trip delays. We assume that the delay $t2$ is exactly the same on both forward and backward paths. Thus, the lower bound of the queue length that guarantees no packet expiration is

$$Lmin = \left\lfloor \frac{T - t1 - t3 - t4}{t3} \right\rfloor \tag{3.4}$$

If the observed queue length falls into the range of $[L_{min}, L_{max}]$, there is a certain probability that a newly arrived PDU will be dropped due to expiration. Intuitively, we expect that a longer queue causes a larger queuing delay, and hence a higher probability of packet expiration. We propose to actively drop some packets at the RLC transmission queue in accordance with the above observation, so that buffer length is limited to avoid packet expirations, and bandwidth can be released earlier to improve utilization.

The relationship between the average packet delay and the average queue length has been analyzed in [33] [34]. Here, we derive the relationship between the expiration probability of real-time packets and queue length.

Assuming the wireless link has a long-term average packet loss rate of $P_B$ due to transmission errors, we can approximate the wireless link with an error-free data link, one that has a variable data rate ranging between $[R/N, R]$, with an average data rate of $\rho = (1 - P_B)R$. Note that this approximation does not change (3.1) - (3.4) above. The transmission time for each packet is thus independent of each others, with a mean $m = l/\rho$ and variance $\sigma^2$ upper bounded by $\{max(l/\rho - l/R, l \cdot N/R - l/\rho)\}^2$.

The independence here is extremely important for further analysis. We assume the expired real-time flow packets are dropped only at the application layer of mobile terminals, which is opposite to EDD or Delay-EDD as we mentioned in the previous chapter. Because the RLC and the lower layers do not retain any timing information regarding real-time applications, they are unable to determine whether a packet has expired or not. In fact, timing information is normally stored in upper layer packets that have been encapsulated into the RLC layer PDUs. Even the RLC layer manages to retrieve timing information (which would break through the protocol layering rules), it is very unlikely that every RLC layer PDU's delay time is recorded for its expiry status to be determined. When packets are ***never*** dropped at the RLC or the lower layers in the interface nodes due to expiry, the transmission time of any packet never affects the transmission time of other packets, so we can consider their transmission time to be independent of each other.

If a newly arrived packet observes $n$ packets in the queue, $L_{min} < n < L_{max}$, its queuing delay becomes a new random variable $V$, which is the sum of the previous $n$ independent random variables. When $L_{min}$ is large enough, this sum of $n$ packet transmission times forms a Gaussian distribution according to the central-limit theorem, with mean $M = m \times n$ and variance $\sigma'^2 = n \times \sigma^2$.

Now with the Gaussian distributed random variable $V$, we can analyze the relationship between the expiration drop rate $p$ and the observed queue length $n$. When a packet arrives at time $\tau_1$, its deadline is computed as $\tau_2 = \tau_1 + (T - t1)$. Since $V$ is a Gaussian variable, the probability of expiration $p$ is

$$p = \Pr\{\tau_1 + v > \tau_2\} = Q\left(\frac{\tau_2 - M}{\sigma'}\right) \tag{3.5}$$

where Q(.) is the well-known Q function giving the tail area of a Gaussian distribution, as shown by the shaded area in Figure 3.1. In Figure 3.1, we draw the Gaussian probability density

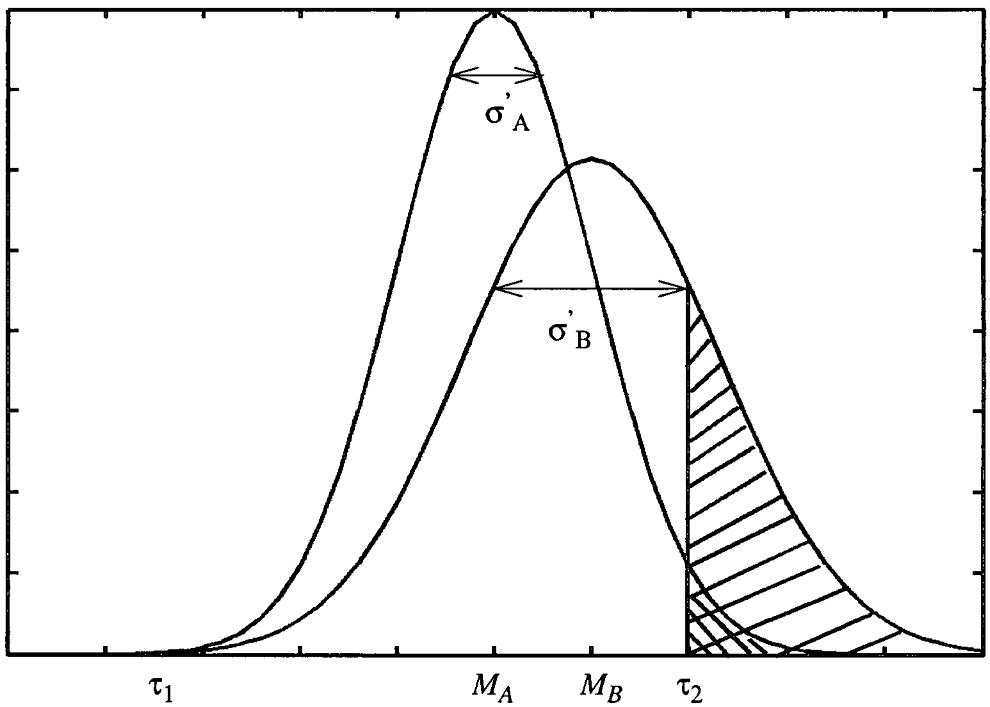


Figure 3.1 Gaussian distribution with means $M_A$ and $M_B$, variances $\sigma'_A$ and $\sigma'_B$, where $M_A < M_B$ and $\sigma'_A < \sigma'_B$.

functions for $n = A$ and $n = B$, where $A < B$. When the value of $n$ is $A$, both the corresponding mean and variance ($M$ and $\sigma'$) are small, so the shaded area is also small. When $n = B$, its $M$ and $\sigma'$ increase, thus the shaded area increases as well. This result matches the intuitive expectation earlier.

Formula (3.5) is extremely important for the scheme we propose in this thesis. It is the natural drop rate caused by expiration. Our active drop rate must not be higher than its value in a wide range of queue lengths, but must come as close to it as possible.

From (3.5) we can get the Gaussian curve I in Figure 3.2, which is the relationship

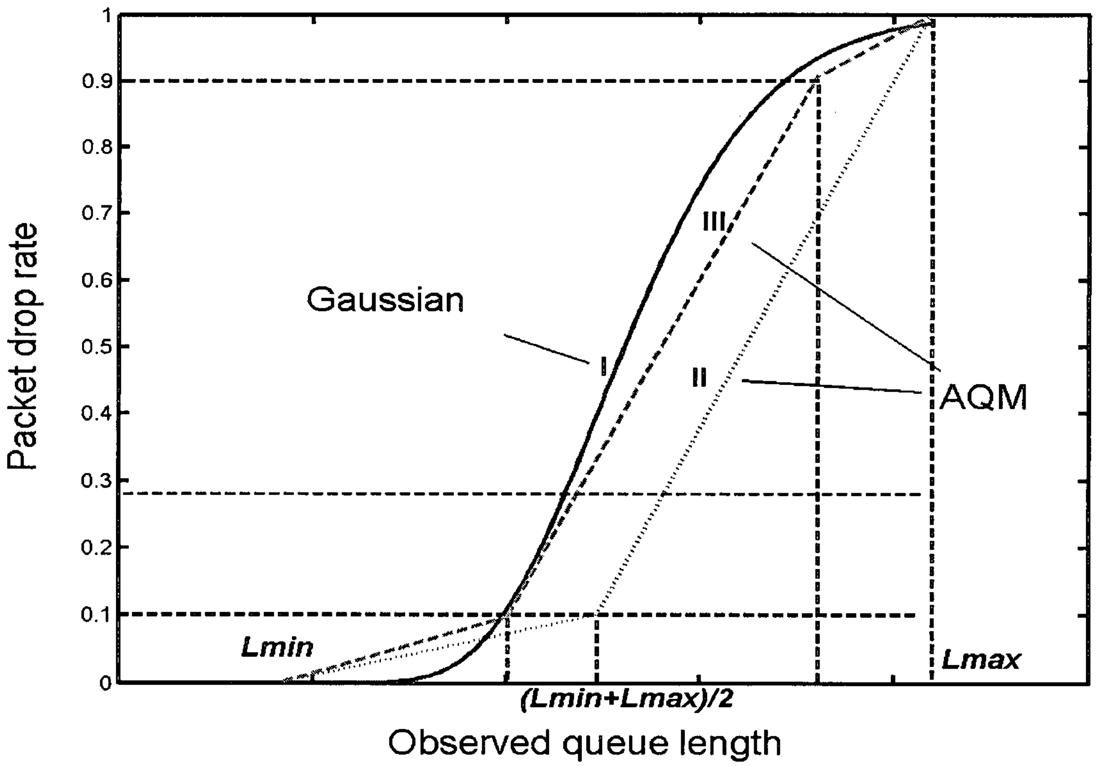


Figure 3.2 Gaussian expiration drop rate and AQM approximations.

between the expiration drop probability $p$ and the observed queue length $n$. To obtain this result, we fix the long-term packet error rate at $P_B = 15\%$, and queue capacity to $L_{max}$. To smooth the changes in $p$, the observed queue length $n$ here is actually made to be the calculated average queue size using a low-pass filter with an exponential weighted moving average $w_q$ [13].

Note that the above result is an approximation as the possible loss of some dedicated control PDUs for status reports has been ignored. These losses introduce some extra queuing delay that causes the actual Gaussian curve I to be slight higher than shown in Figure 3.2. Because the dedicated control PDUs are much smaller than the data PDUs in length, for simplicity we ignore this additional delay.

To actively drop packets with the exact probability obtained above is difficult to realize, due to computing complexity. Instead, we propose to use a simpler scheme similar to gentle-RED to approximate the expiration rate. In Figure 3.2 we present two piece-wise, linear AQM functions, II and III, to approximate the Gaussian curve. We set these two approximation curves to be lower than the Gaussian one in most of the range of the observed queue length, so as to guarantee that the number of packets actively dropped is less than the natural expiration drops.

Clearly, the better the approximation is, the better the performance should be. Therefore function III yields the best result as shown in the next section. However, in reality, it is hard to measure the Gaussian curve because of varying factors. We consider the two-slope, piece-wise, linear function II, defined in (3.6), to be the easiest to implement, as it can be uniquely fixed by $L_{max}$ and $L_{min}$, while the other parameters are chosen somewhat arbitrarily. We can see that this simple scheme works well enough in controlling queue length and improving system performance.

$$p = \begin{cases} max_p \cdot \dfrac{(n - L_{min})}{\left(\dfrac{L_{min} + L_{max}}{2} - L_{min}\right)} & \text{if} \quad L_{min} < n < \dfrac{L_{min} + L}{2} \\ (1 - max_p) \cdot \dfrac{\left(n - \dfrac{L_{min} + L_{max}}{2}\right)}{\left(L_{max} - \dfrac{L_{min} + L_{max}}{2}\right)} & \text{if} \quad n \geq \dfrac{L_{min} + L_{max}}{2} \\ 0 & \text{if} \ n \leq L_{min} \end{cases} \quad (3.6)$$

## 3.2 Simulation Results

The simplified system simulation model shown in Figure 3.3 has been built and exercised

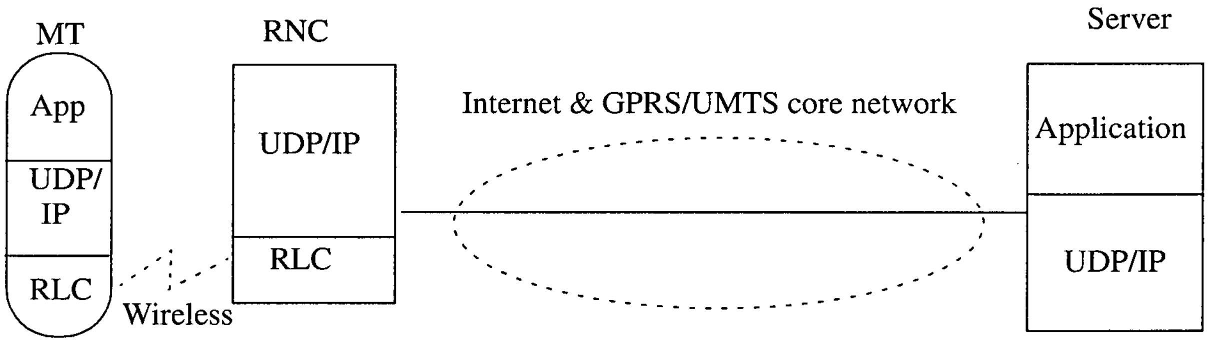


Figure 3.3 Simulation model for AQM in the link layer buffer.

in the OPNET 8.1 environment. The RLC layer protocols shown above are carefully modeled. We simulate the behavior of link layer ARQ with polling functions and other conditions that can trigger the generation of status report, introduced in Section 2.3.1. For the status report, we modeled the LIST Super-Fields (SUFI) [18] to carry the NAK and ACK information.

The wireless channel model in OPNET 8.1 takes into account multiple access interference, background noise and multipath fading, thus, it forms a more comprehensive error model than the Gilbert-Elliott model [35] [36]. To focus the performance evaluations on the management of the downlink transmission queue, we make a simplifying assumption that the wireless uplink and the wireline Internet links are error-free. We assume that the arrivals of real-time packets follow the Poisson distribution, that packets have a constant length, and that the unit of queue length in the figures given below are the number of packets. Thus, the traffic load varies in direct proportion to the average inter-arrival time of real-time packets.

### 3.2.1 Two-slope Piece-wise Linear AQM Function II

In this section, we take the two-slope piece-wise linear approximation AQM function II as an illustrative example of the effects of the scheme. The parameters for the wireless link and real-time application are given in Table 3.1.

Table 3.1 Parameters for Simulation of AQM in RLC layer.

| Parameters | Value |
|---|---|
| Effective wireless data rate $R$ | 1024 K bps |
| Max. number of transmissions *MaxDAT* | 5 |
| $L_{min}$ | 40 packets (AMD PDUs) |
| $L_{max}$ | 210 packets (AMD PDUs) |
| Maximum allowable end-to-end delay $T$ | 250ms |
| Propagation delay in Internet & GPRS/UMTS core network | 50ms |
| Bandwidth of Internet link | 2048 K bps |
| Average long-term packet error rate $P_B$ | 15% |

To get the values of the RED parameters such as $w_q$ and $max_p$, we set them as suggested in [23], $w_q = 0.002$ and $max_p$=0.1.

Figure 3.4 ~ Figure 3.7 come from the two-slope, piece-wise, linear AQM function II in (3.6) with 90% offered load. In this thesis, the term "offered load" refers to the ratio of the traffic generating rate of a specific traffic source, and the effective service rate of the corresponding wireless link. From Figure 3.4 we can see the relationship between the wireless BER, throughput,

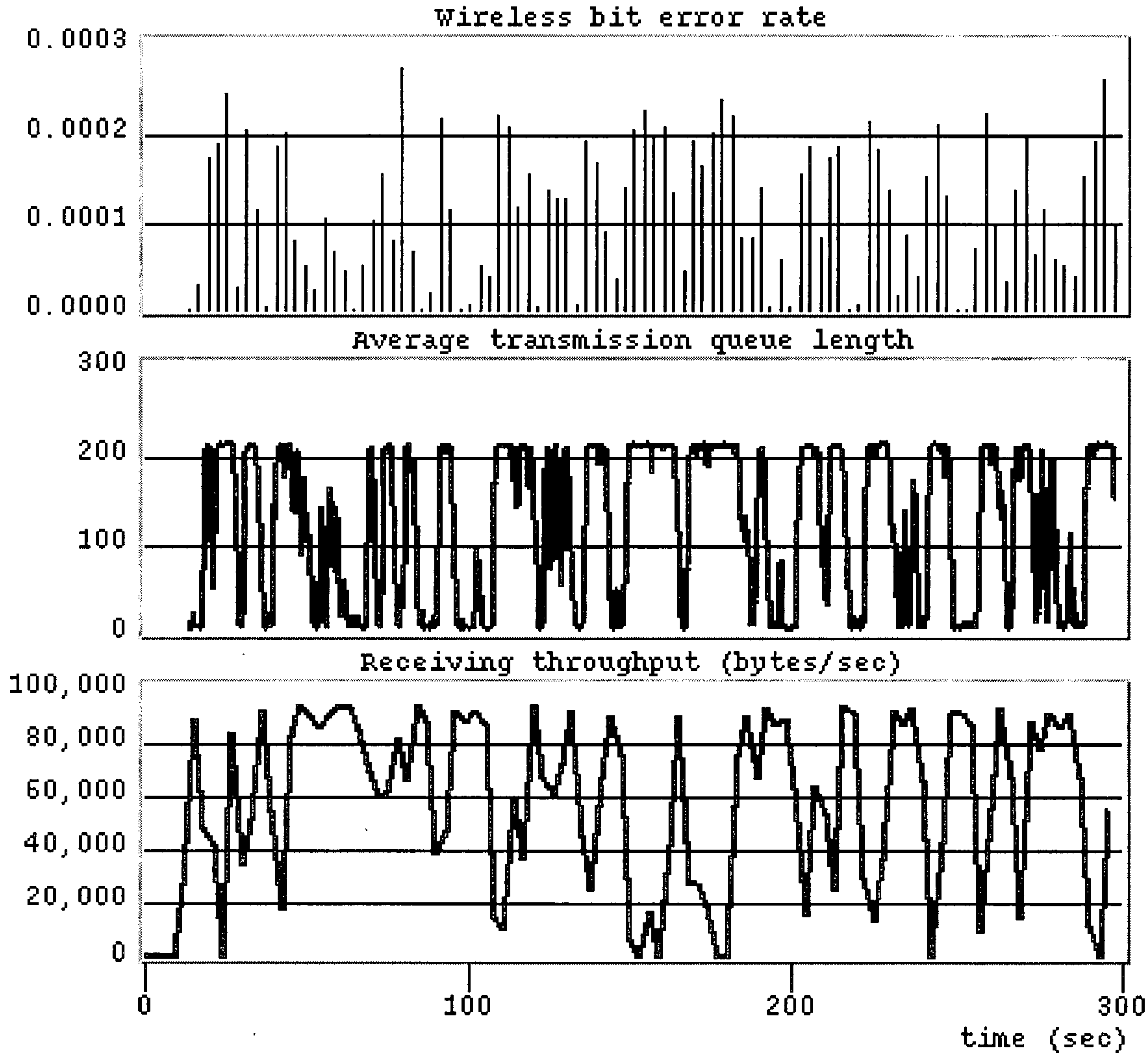


Figure 3.4 BER, transmission queue length, and throughput without AQM.

and the average transmission queue length (in packets) as functions of time, without AQM. When the queue length becomes large with a high wireless bit error rate, the corresponding throughput drops sharply. We must note that wireless error does not directly cause packet loss because of the link layer ARQ recovery mechanism. However, the high wireless bit error rate causes frequent retransmission at the link layer and large queuing delays, leading to expiration drops. Figure 3.5

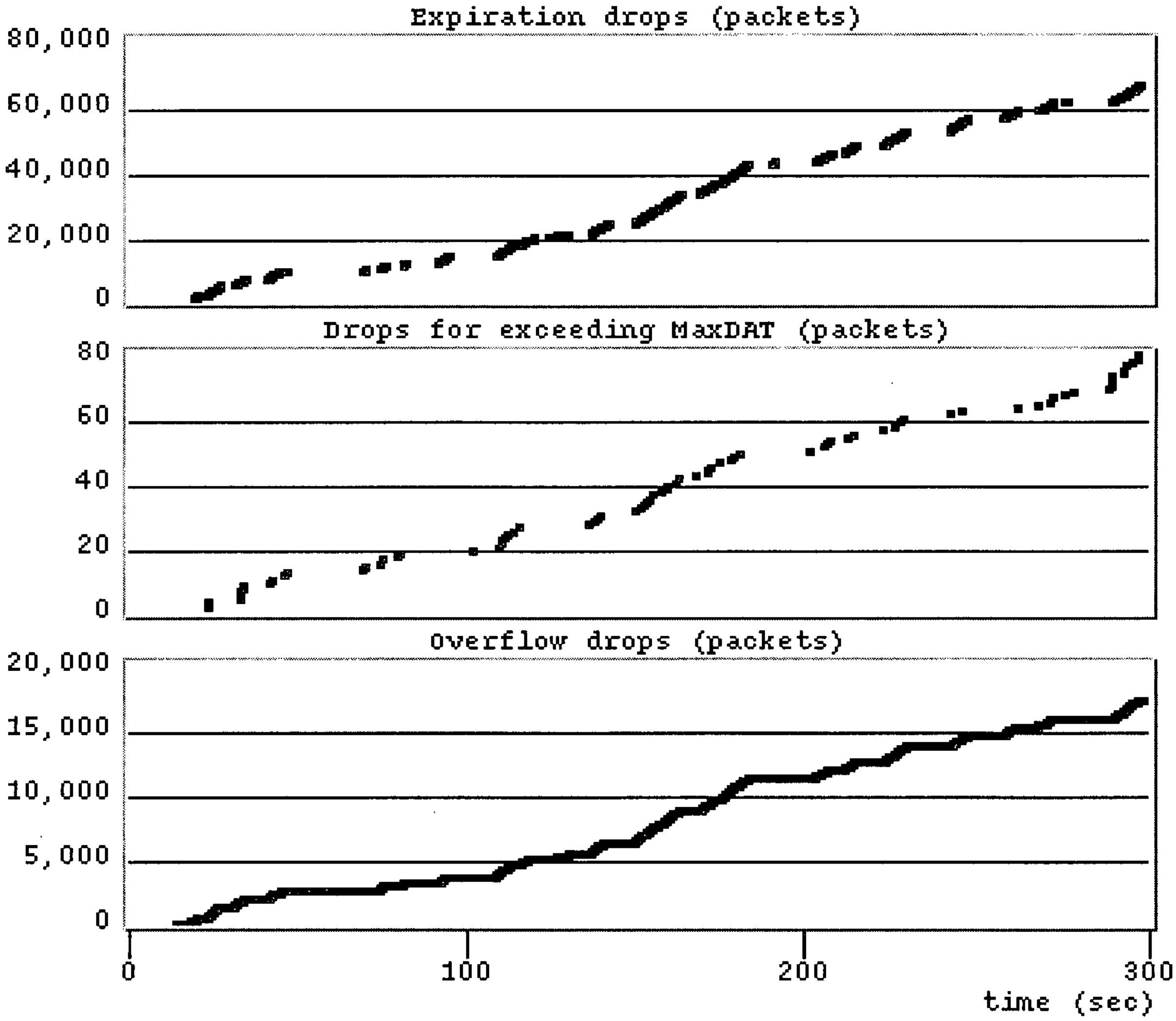


Figure 3.5 Different packet drops without AQM.

shows this clearly.

We can see that there are three types of packet drops when no AQM is applied: expiration, transmission time exceeding maximum value (*MaxDAT*), and overflow. Among them, the expiration packet drops account for the largest share. When the queue length or wireless BER increases, the number of all three types of packet drops increases as well, but expiration drops are the most harmful to system performance, due to their large share of total packet drops. Figure 3.5 shows how serious the problem of expiration can become when the system load is high.

Applying the AQM scheme in (3.6) to constrain the queue length changes the situation dramatically. Figure 3.6 shows the relationship between the BER, queue length (in packets), and

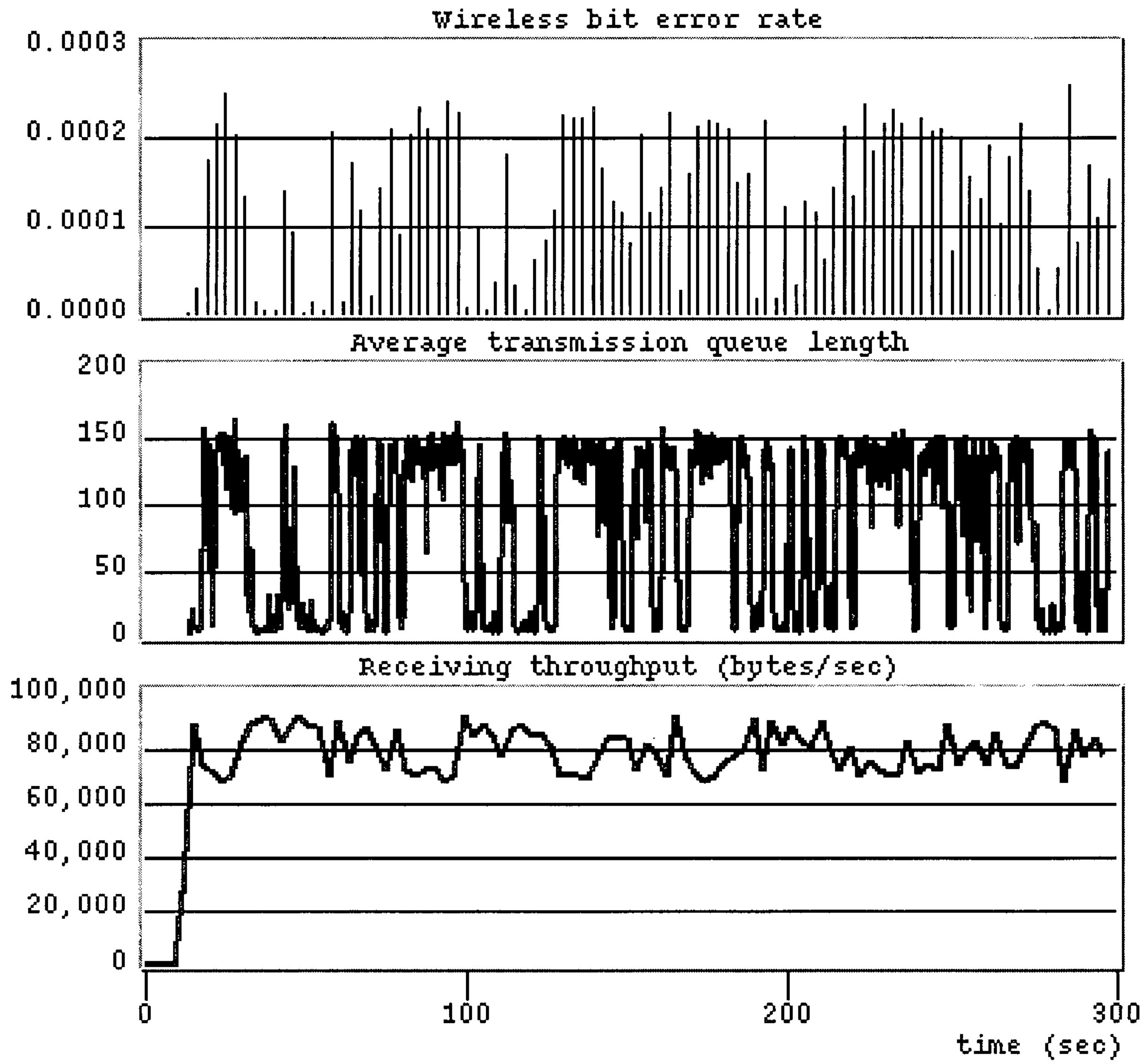


Figure 3.6 BER, transmission queue length, and throughput with AQM.

throughput over time, with AQM function II. We can see that the average queue length is effectively controlled near the middle point of $(L_{min} + L_{max}) / 2$, and that the throughput curve does not fluctuate much on the time scale, which is a better situation than in Figure 3.4. The simulations also show that the maximum queue length increases with the BER and traffic arrival rate, presented later.

Figure 3.7 shows the reasons behind the improvements. We can see that one of the main

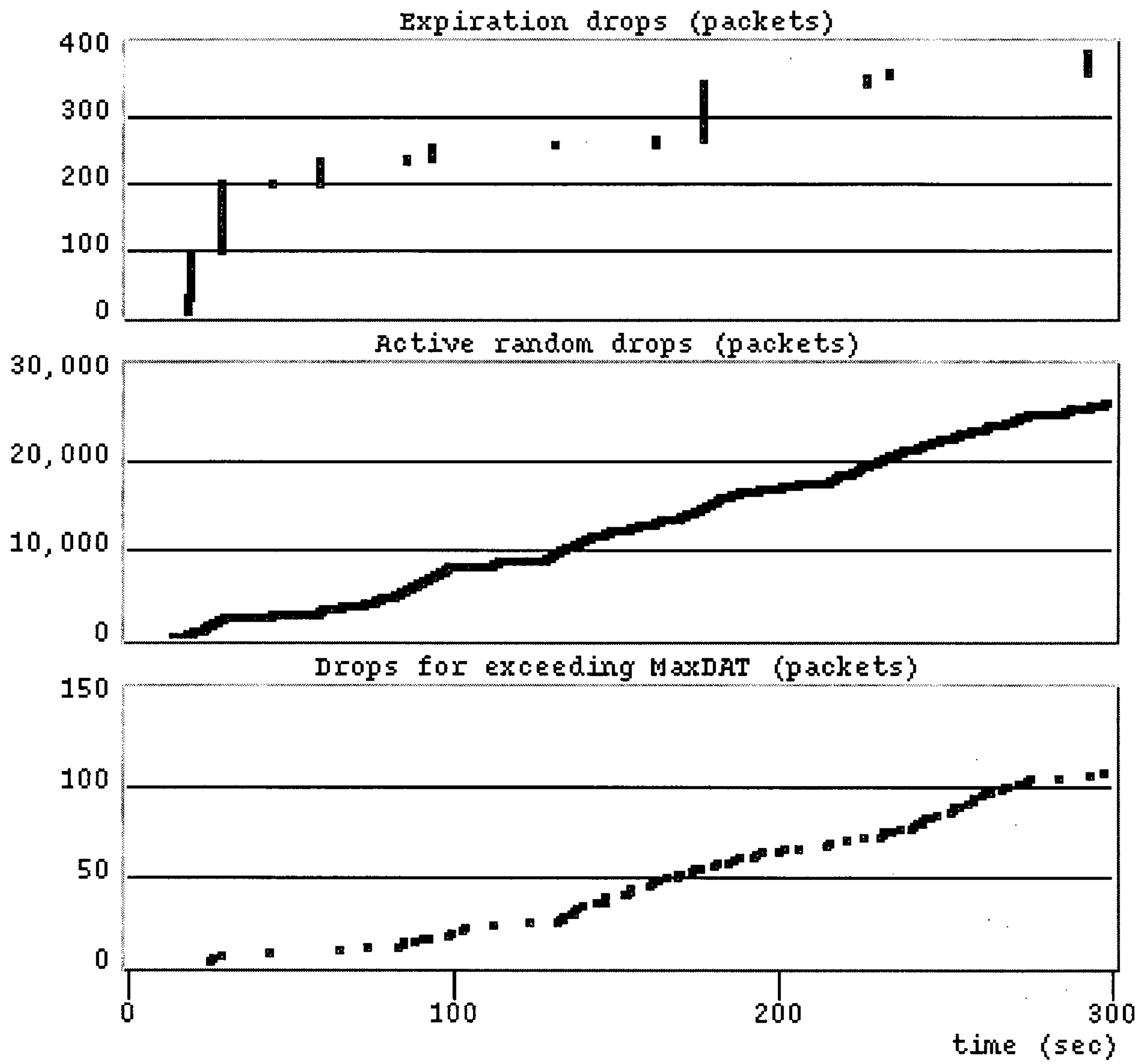


Figure 3.7 Different packet drops with AQM.

differences between Figure 3.5 and Figure 3.7 is that the types of packet drops have been changed. With AQM, all overflow and most expiration drops are eliminated, replaced by a new kind of packet drop, active random drops. The figure shows that the amount of active random drops is much less than of expiration drops in Figure 3.5, but still dominates other types of packet drops. This demonstrates the effectiveness of AQM in improving system throughput, without degrading the overall packet drop rate. Note that the latter condition is an important criteria in maintaining the quality of service for real-time applications, as mentioned above.

### 3.2.2 Comparison of Different AQM Functions with Varying Traffic Load

In this section, we present the effect of the two AQM functions II and III with different traffic loads. We fix all other parameters as in Table 3.1, and vary the traffic load from 50% to 100%.

The AQM function III has been presented in Figure 3.2. We set the two turning points corresponding to the drop rate 0.1 and 0.9 in the AQM function III as 100 and 180, with the parameters listed in Table 3.1. Figure 3.8 and Figure 3.9 show the throughput and end-to-end delay of these dropping functions.

Under heavy traffic (traffic load exceeding 70%), the AQM schemes yield great improvements to the overall throughput and end-to-end delay. As expected, the AQM function III gives better performance than function II, thanks to better approximation of the Gaussian curve.

Figure 3.10 shows the moving averaged queue length of the above three dropping schemes with varying traffic loads. Here, the average queue length increases with the traffic load when the BER is constant.

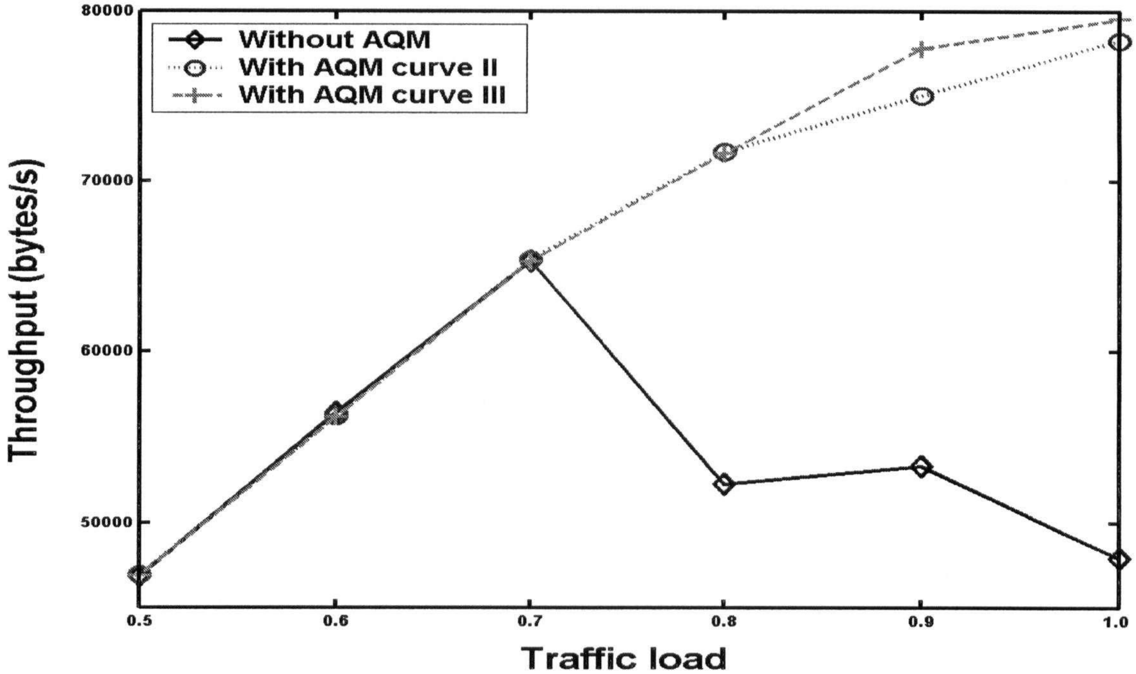


Figure 3.8 Throughput vs. traffic load with three schemes.

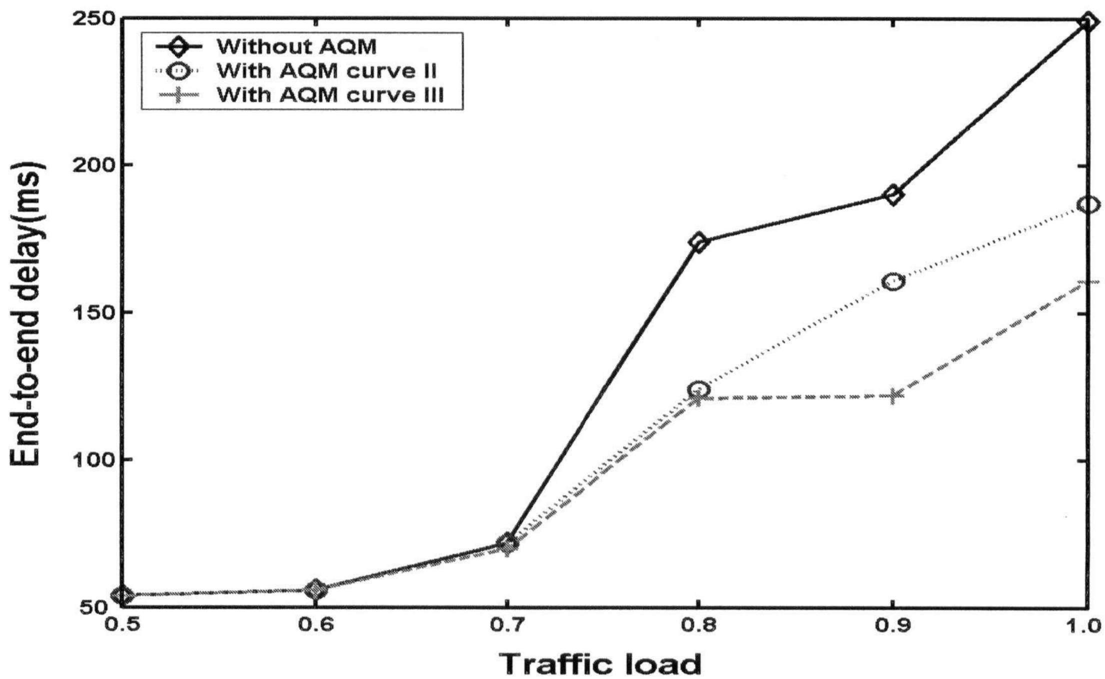


Figure 3.9 End-to-end delay vs. traffic load with three schemes.

Figure 3.11, Figure 3.12 and Figure 3.13 show different types of packet loss with varying

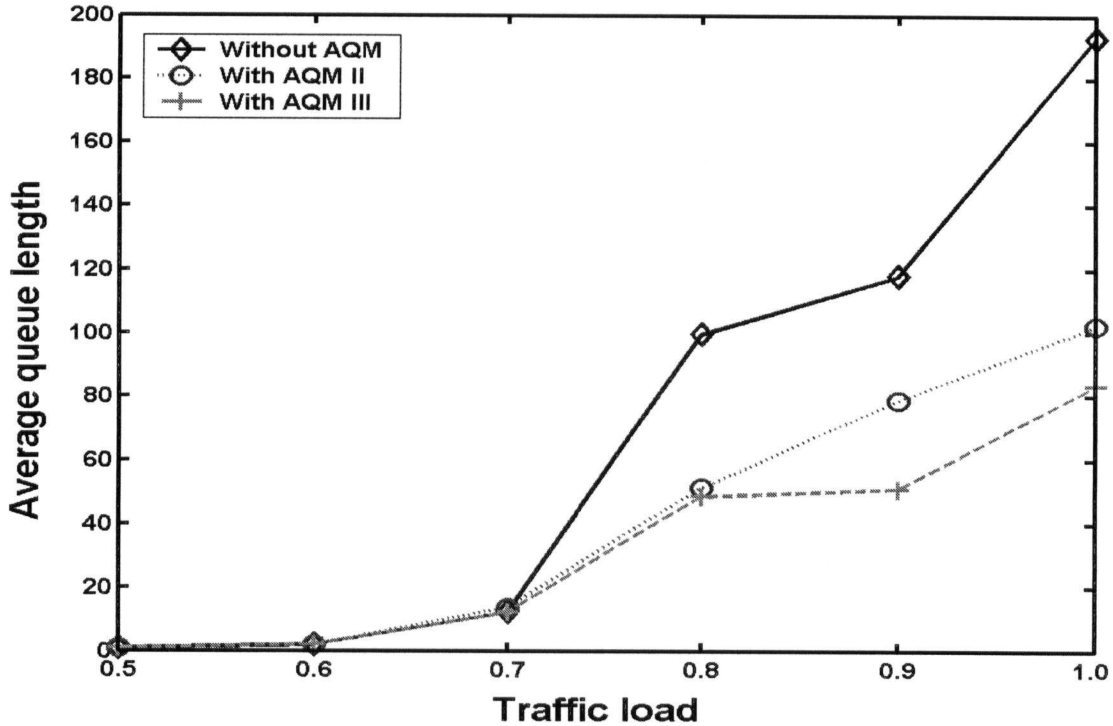


Figure 3.10 Average transmission queue length with three schemes.

traffic loads, without AQM, and with different types of AQM functions (II and III). Owing to the rule that the active packet drops rate is less than the Gaussian drop rate, the total numbers of drops with AQM functions II and III are much less than that without AQM.

It is apparent that when traffic is light, the queue length is not long enough to trigger the AQM (see Figure 3.10), and the expiration packet drop rate is small enough not to damage system performance. Therefore the performances in both scenarios, with and without AQM, are similar. With an increasing traffic load, both the expiration drop rate and queue length grow. When the traffic load exceeds 70%, the advantages of our proposed AQM schemes become quite obvious. Under such conditions, without AQM, the wireless link is saturated by an ever increasing number of retransmissions, resulting in the reduction of throughput as traffic load increases, as shown in Figure 3.8. As the transmission queue length approaches $L_{max}$, the expiration drop rate becomes

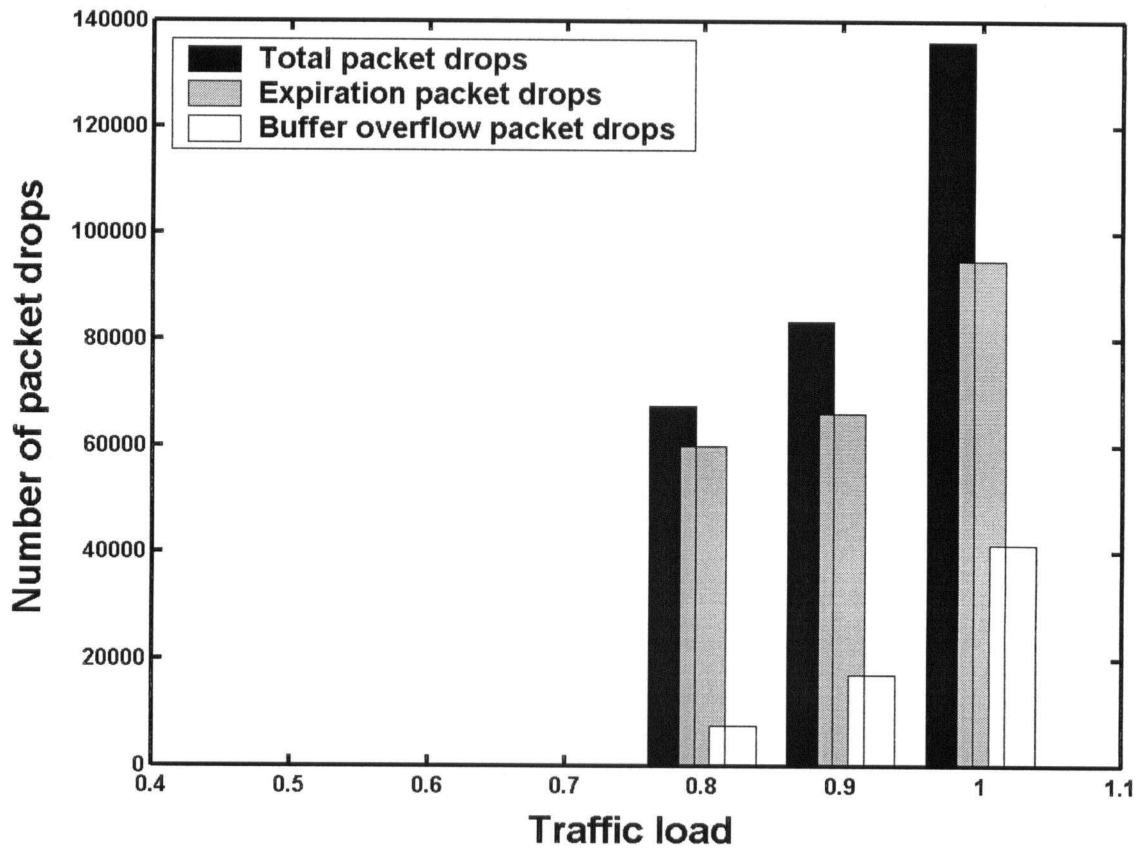


Figure 3.11 Different packet drops vs. traffic load without AQM.

the dominant (Figure 3.11) and most harmful factor to system performance, in both throughput and end-to-end delay. With AQM, we can control queue length and limit it to a lower value, thus eliminating expiration packet drops. Since the AQM releases resources that would otherwise be used ineffectively by expired packets, it is also able to mitigate throughput saturation, as shown in Figure 3.8, where the throughput curve for the AQM continues to increase up to a traffic load of 100%.

From these figures we can also see that the better the AQM functions approximate the Gaussian curve, the better they can constrain queue length, and the better performance becomes. When we compare Figure 3.12 to Figure 3.13, we see that the AQM function III does a better job of eliminating expiration packet drops, and it outperforms the AQM function II. In Figure 3.12

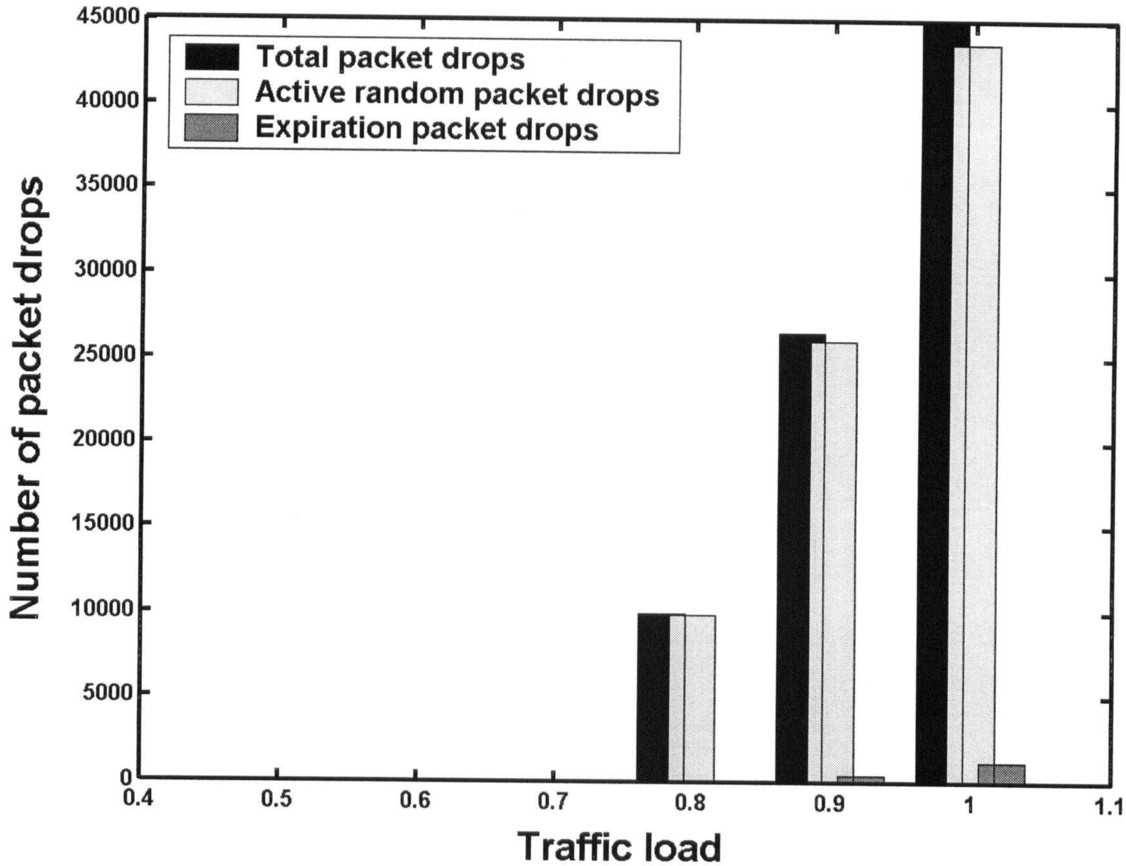


Figure 3.12 Different packet drops vs. traffic load with AQM function II.

some expiration packet drops still remain, while in Figure 3.13, none do. This complete conversion from expiration drops to active random drops is critical to improving the end-to-end Quality-of-Service in this scenario.

As we can see from Figure 3.2, the AQM function II can be uniquely fixed by $L_{min}$ and $L_{max}$, while the AQM function III lacks a middle point from which to determine the turning points, and requires more accurate information regarding the factors effecting the Gaussian curve for approximation. If we have all the necessary information to derive the Gaussian curve, we can easily determine an AQM function with a good approximation. Any AQM function can be actually chosen to approximate the Gaussian curve, if the computing capacity is powerful enough.

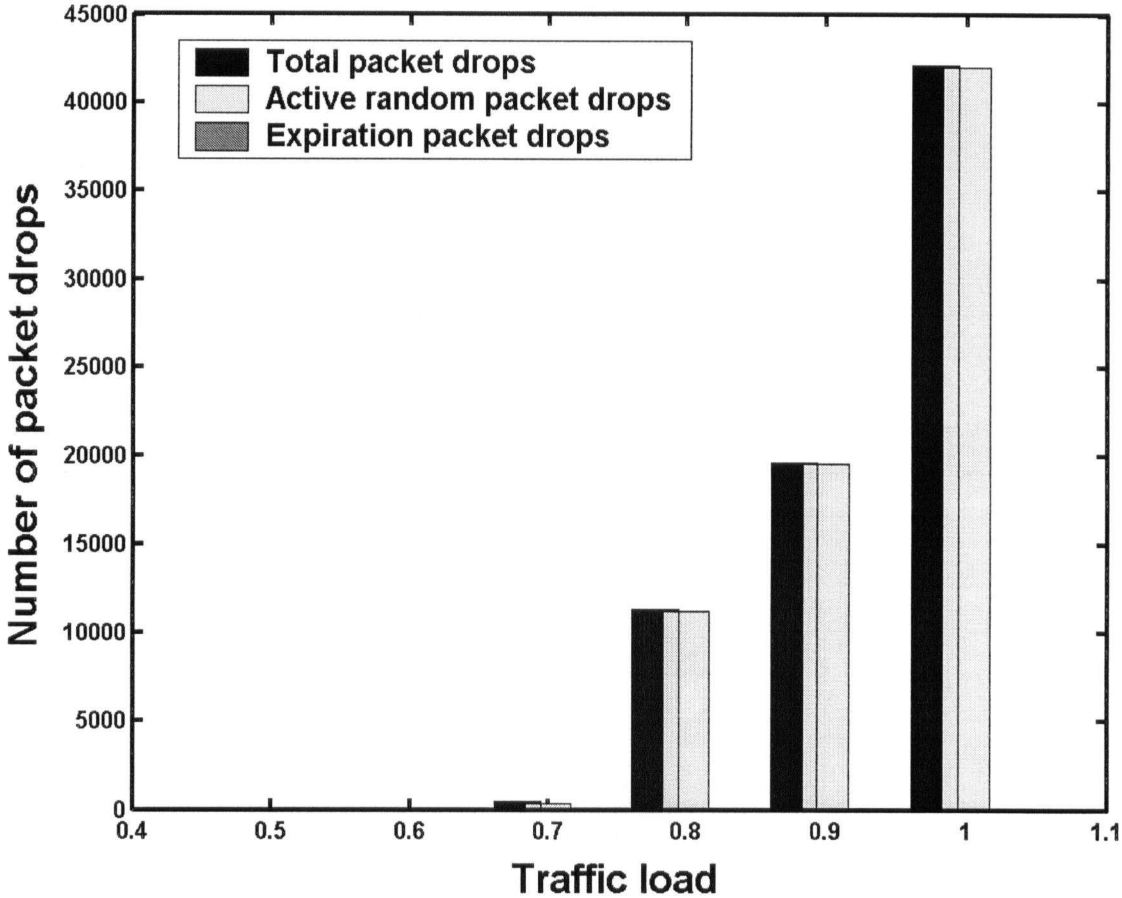


Figure 3.13 Different packet drops vs. traffic load with AQM function III.

One of the parameters most difficult to determine, but deeply effecting the Gaussian drop rate, is the long-run wireless packet error rate $P_B$. In the next section we show results using AQM functions to approximate different Gaussian curves caused by different BERs.

### 3.2.3 AQM Functions with Different Error Rates

Different wireless packet error rates in the long term (during transmission periods) lead to different Gaussian drop rates, as illustrated in Figure 3.14 below.

The Gaussian curves vary with different long-term packet error rates in a trend where the larger $P_B$ moves the Gaussian curves closer to the $L_{min}$. Understandably, the larger $P_B$ leads to

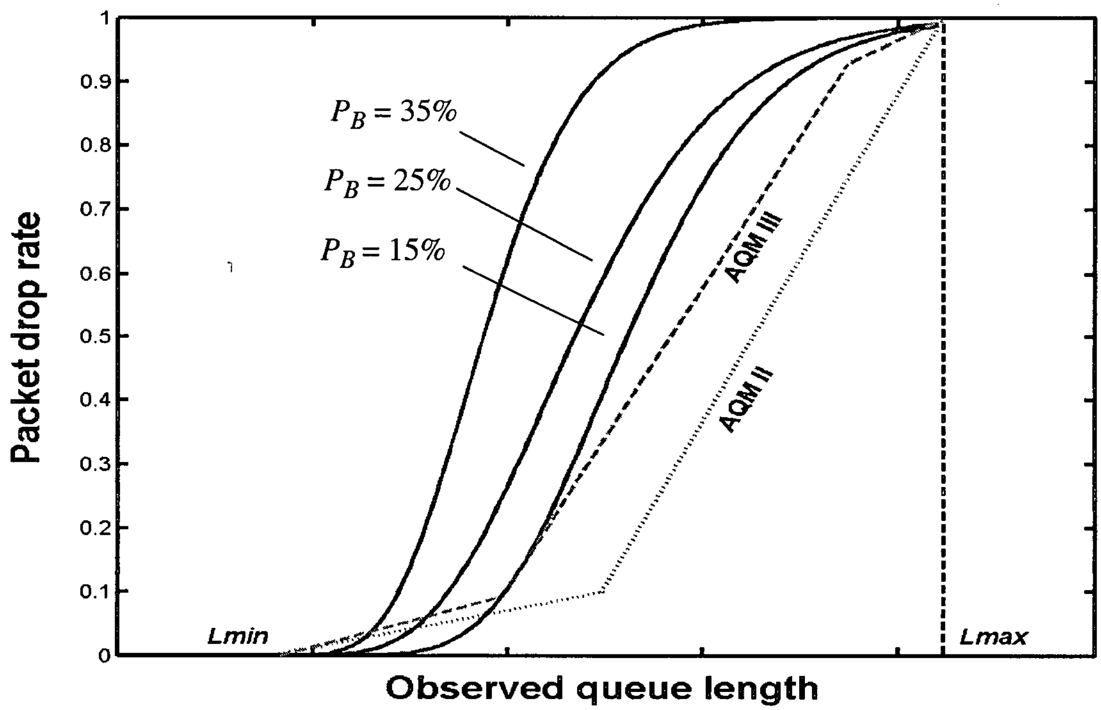


Figure 3.14 Packet drop rate with different long-run packet error rates.

more retransmission and longer queue length, thus causing a higher probability of real-time packet expiration. The wisest way to choose an AQM function is to sample the error rate for a short period at the beginning of the transmission, and use the sampled values to determine an initial Gaussian curve, and then derive the appropriate AQM function to approximate it. Because such short periods may not generate reliable enough sample values, the Gaussian curve and the AQM function may need to be further adjusted during the following transmission period. The basic rule, however, that the AQM drop rate be not higher than the Gaussian drop rate in most queue length ranges, must be adhered to.

Every AQM function may perform well within a certain queue length range, even if it does not accurately approach to the Gaussian curve. Below we show the effect of the AQM functions II and III in Figure 3.2 with different error rates.

Again, we use the parameters listed in Table 3.1. The packet error rate $P_B$ ranges from 5% to 35%. The traffic load is constant at 80%. The system performance of the AQM functions and the non-AQM (Gaussian) cases with a varying long-term packet error rate of $P_B$ is presented in Figure 3.15 and Figure 3.16.

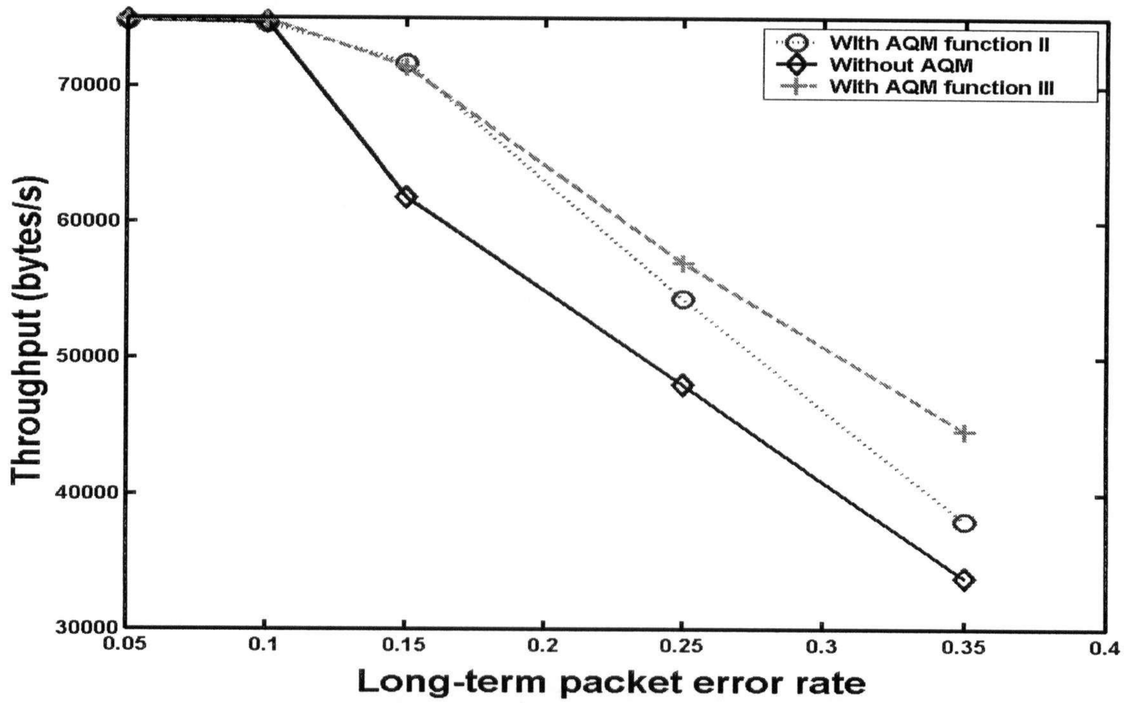


Figure 3.15 Throughput vs. long-term packet error rates $P_B$.

When the long-term packet error rate $P_B$ varies within a certain range (around 15%), both AQM functions achieve very good performance. If the error rate grows to higher values, as in Figure 3.14, the Gaussian curve moves closer to $L_{min}$. Here, the effect of the AQM function II gradually weakens because the gap between the Gaussian curves widens, while the AQM function III still performs well because it is closer to the Gaussian curves. When $P_B$ is small, the average queue length rarely grows to a large enough value, thus the AQM rarely takes effect, and the results with and without AQM are similar. This shows the importance of accurately measuring the

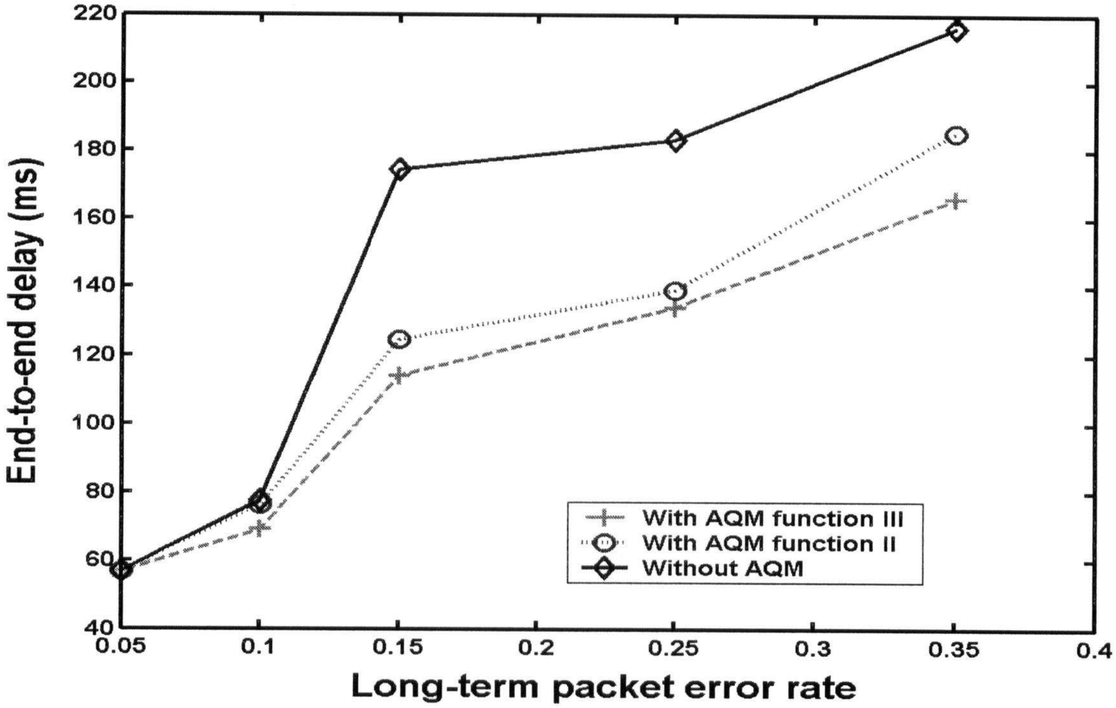


Figure 3.16 End-to-end delay vs. long-term packet error rates $P_B$.

Gaussian curve and carefully choosing a good AQM function. Also, in order to get the best AQM function, the long-term packet error rate must be measured before the AQM function is chosen, and further adjustments may be needed during the following transmission period.

In this simulation scenario, the RESET function of the RLC layer in GPRS/UMTS is not considered. If $P_B$ grows to a high value, the wireless link is very likely to be reset by this function.

## 3.3 Summary

In this chapter, we analyze the behavior of the 3GPP GPRS/UMTS RLC layer's transmission queue in Acknowledged Mode, and derive a novel expression for the relationship between the average queue length and drop rate due to expiry of real-time flows with hard time deadlines. We prove that this drop rate can be approximated as a Gaussian curve on the average queue length. With this approximation, we propose to actively drop those packets that are likely to expire, according to probability, following piece-wise, linear functions before they are enqueued. This action guarantees no QoS degradation by making the active drop rate to be lower than the Gaussian drop rate in every case, and on this basis, improves overall performance when traffic load or the wireless error rate is high.

Through simulations, we prove the effectiveness of the mechanism. We also prove that the better the active drop rate curve is at approximating the Gaussian curve, the more the improvement.

# Chapter 4 Applying WERUF to RLC Queue

We not only apply the idea of dropping real-time packets earlier at the wireless-wireline interface nodes, but also extend it to the whole GPRS/UMTS domain in this chapter.

The reasons for and advantages to limiting unresponsive best-effort traffic have been proven in [15] [16]. The idea behind ERUF is that when unresponsive flows encounter network congestion at certain nodes over the end-to-end path, causing some packets to miss their delivery deadlines or overflow, dropping them at an earlier node frees shared network resources for other flows. We apply a similar idea in GPRS/UMTS wireless networks. In these networks, although the shaping functions at edge (both wireless and fixed sites) nodes limits the volume of unresponsive flows, the problem of these flows unnecessarily using up valuable network resources is exacerbated by the costliness of wireless bandwidth. Furthermore, propagation-induced variability of transmission quality over wireless links causes congestion at the wire-wireless interface nodes, that is, RNCs in UMTS networks. We adapt the ERUF scheme to address these problems. The new scheme is called Wireless Early Regulation of Unresponsive Flows (WERUF). Ref. [14] also proposes an algorithm to identify flows that are not TCP-friendly for regulating. In our case, however, there is no need for identification because the link layer transmission queue length is the best indicator of the flows to be regulated.

## 4.1 Early Regulation in GPRS/UMTS Network

Figure 4.1 demonstrates the general idea of WERUF. Leaky Bucket 1 (LB1) and Leaky Bucket (LB2) can be realized by the DiffServ traffic conditioners and the wireless channel resource management functions, respectively. When packets accumulate at RNC, due to RLC retransmission caused by wireless error, some signals are sent back to the DiffServ edge node,

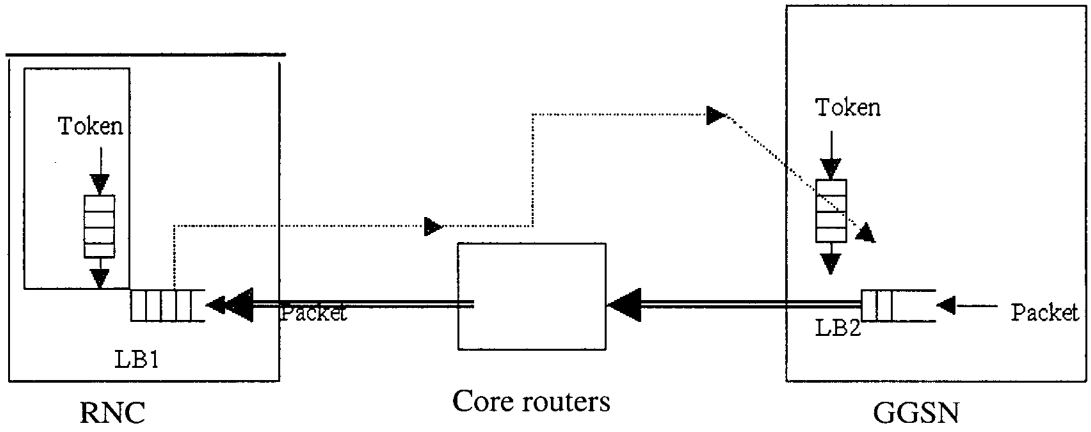


Figure 4.1 The general concept behind Wireless Early Regulation of Unresponsive Flows (WERUF).

namely, GGSN in the GPRS domain, to notify it to shrink the token rate of the corresponding traffic shaper (the token bucket LB2). With this operation, the shared network resources in the forwarding nodes (or core routers) such as SGSNs are released for other flows. Because the packets accumulating at RNC may expire, as presented in discussed last chapter, we intend to generate the congestion signal using a lower probability and drop these packets at the GGSN. In this way, we actually convert the expiration packet drops to overflow-like packet drops at the DiffServ ingress edge node, and at the same time, guarantee performance degradation will not occur at the constrained flows. It is worth noting that the packets in RNC are the AMD PDUs, while the packets in GGSN are the IP packets. They are handled by different layers.

Again, we need to analyze the behavior of the RLC layer retransmission queue length as stated in previous chapter. Equations (3.1) ~ (3.5) still apply, except the meaning of some variables are changed. In this chapter we are concerned with finding an algorithm that caters to the QoS requirement of the UMTS bearer service, and making improvements.

Focusing on the GPRS/UMTS domain, we assume the latency caused by the Internet, external to the GPRS/UMTS domain, is a fixed value. The variable $T$ in (3.1) and (3.4) becomes the amount of transfer delay over the UMTS bearer service. According to [7], $T < 250$ ms. In the simulation we use 250 ms for simplicity. As a result, $L_{min}$ and $L_{max}$ in these equations becomes the minimal and maximal transmission queue length that can cater to the transfer delay requirement of the UMTS bearer service instead of the end-to-end delay requirement of real-time applications. We also assume that no PDU is dropped in the RLC layer due to expiry, so that the independence of each PDU's transmission time is still maintained. Another change is that the meaning of $t1$ in these equations becomes the latency caused by the GPRS/UMTS core network, excluding the Internet. The equations still hold true, despite the changes listed.

Further, we use OPNET 8.1 to model the DiffServ functional elements in the GPRS/UMTS routers, as shown in Figure 2.3. In GGSN, we build the Multi-field classifier, the traffic shaper, the multiplexer and the dropper. In SGSN we build only a dropper. These droppers are RIO droppers handling IN and OUT packets. They strictly follow the IETF document [21] to complete the Assured Forwarding (AF) architecture accurately.

One of the base of this scheme is that every flow must be independent, except at the core routers. Therefore we assume the Complete Partitioning (CP) [37] policy is used at the GGSN for each flow shaped by the Leaky Bucket. At RNC, if no explicit traffic conditioner is implemented and the shaping relies on wireless resource management, some assumptions must be clarified. If these flows are multiplexed at the same transmission buffer, our scheme can either apply to cases where each flow goes through different RNCs, served by the same core network router, or the CP policy is also used at the RNC for every flow. In order to separate the flows, a Multi-field Classifier resides at the RNC (Figure 2.3). Whether there are one or more RNCs in the simulation, the network topology figures below show only one RNC for simplicity.

Source quenches employ Internet Control Message Protocol (ICMP) [38] packets, sent to the GGSN to adjust the shaper's token rate for the flow (here we consider the per-flow shaping only). These ICMP packets are generated according to the AQM functions described in last chapter. In the GGSN shapers, the token rate adjustment is made by multiplicative-decreases/additive-increases. When an ICMP packet is received, and the token rate has not been cut within the current estimated end-to-end round trip time (RTT), the new token rate is calculated by

$$new\ rate = min\{guaranteed\ token\ rate,\ current\ token\ rate/2\} \tag{4.1}$$

where the guaranteed data rate is pre-configured by the system [7]. This cut-by-RTT scheme protects the regulator from being rapidly driven down by a sequence of source quenches before the source has had a chance to respond to the congestion.

The end-to-end RTT is difficult to measure for real-time traffic, due to the lack of return link acknowledgments. [14] proposes to use twice the propagation delay on the next link towards the congested router to approximate the RTT, but this proves unfeasible due to the extra delay caused by link layer retransmissions. With "lossy" wireless links, the queuing delay dominates the propagation delay. When this type of estimate is made at the DiffServ ingress edge node, GGSN, there is no easy way to predict the queuing delay at the egress edge node, like RNC. To resolve this problem, we consider that the 3GPP-defined maximum transfer delay in the UMTS bearer service, *T*, to be a reasonable estimate for real-time applications for several reasons. First, in most cases, the latency introduced by the external Internet is much less than the latency in the GPRS/UMTS domain with noisy wireless links. We can assume the delay in the GPRS/UMTS domain represents the total trip time. Second, if the one trip delay in the GPRS/UMTS network for all real-time applications is uniformly distributed, the mean value of this delay is likely to be located in the middle of [0, 250 ms]. Therefore using 250 ms to approximate the round trip delay is quite reasonable.

The token rate recovery mechanism proposed in [16] consists of the bandwidth of the rate-reduced shaper associated with a flow being increased by one average packet size for every RTT estimate that passes, without the arrival of a source quench for the flow until the token rate recovers to the original value. This scheme recovers the token rate too slowly, however, and does not work well in wireless networks. For congestion caused by wireless errors, which normally has a much smaller time scale than congestion periods in wired networks, the edge-node token rate should preferably not stay at a low level for an extended period of time. Otherwise, if the wireless link has recovered but the token rate has not, the throughput degrades unnecessarily. Our recovery rule is that after a token rate reduction, the token rate increases by one average packet size every time a new downlink packet arrives at the token bucket, until the token rate has recovered to the initial value. Our simulation results prove the effectiveness of this mechanism.

As mentioned before, we use a complete partitioning (CP) policy to manage the buffer of the GGSN for each token bucket. The buffer size is set to a small value so new packets start dropping earlier, after the token rate is reduced.

## 4.2 Simulation Results

We use the same OPNET 8.1 RLC models as in last chapter with the new models of GSNs. Some parameters commonly used in all below simulations are listed in Table 4.1. All of the following simulation scenarios are with some background traffic in the core network. ICMP packets are generated by the RNC with probabilities determined by the AQM function II. The

Table 4.1 Common Parameters for Simulation of WERUF.

| Parameters | Value |
|---|---|
| Max. number of transmissions MaxDAT | 3 |
| Packet error rate in noisy link | 15% |
| Packet error rate in clean link | 0.2% |
| $L_{min}$ | 30 packets (AMD PDUs) |
| $L_{max}$ | 180 packets (AMD PDUs) |
| Propagation delay in the GPRS/UMTS core network | 1.0 (ms) |
| Wireless effective service data rate for each real-time flow | 1 M bps |
| Wireless effective service data rate for each TCP flow | 384 K bps |

token rate adjustment and RTT estimate methods at the GGSN, upon receiving an ICMP packet, are exactly the same as described in section 4.1. The term "offered load" in scenarios given below refers to the offered load for every real-time flow, for example, a 90% offered load means every real-time flow has a 90% offered traffic load, no matter what its wireless link condition. We also let the wireless links be independent of each other. The wireless errors are caused by a separated interference source.

### 4.2.1 Real-time Flow and TCP Flows

Real-time flows and TCP flows normally have different timing requirements. In the following two scenarios, we apply the WERUF to real-time flows only because TCP flows do not have any expired packet drops.

#### 4.2.1.1 One Real-time Flow in a Noisy Link and Multiple TCP Flows in Clean Links with Equal RTT

In this scenario, M1, M2, M3 and M4 are downloading traffic from H1, H2, H3 and H4, respectively (Figure 4.2). The M1~H1 pair is transferring real-time traffic, experiencing high

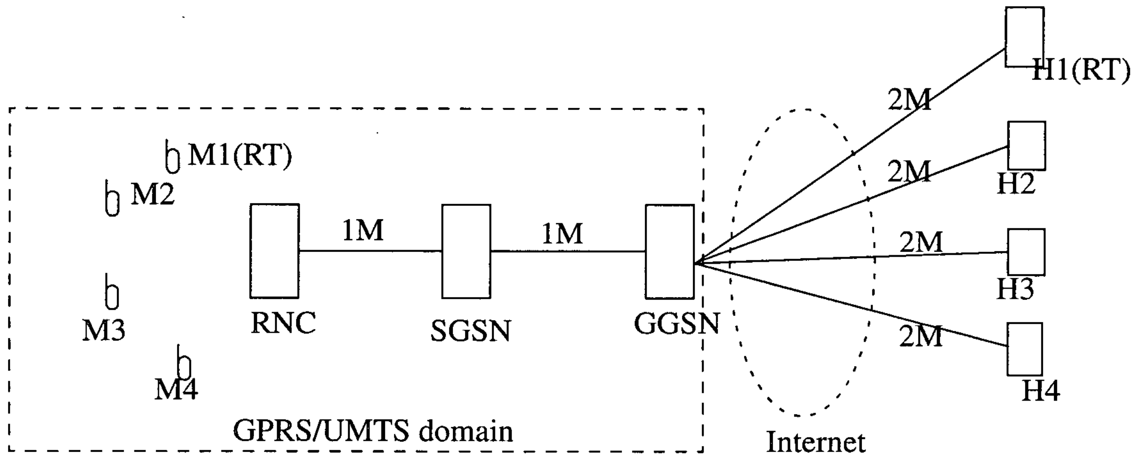


Figure 4.2 Network topology of one real-time plus three TCP flows.

BER in the wireless link, and has a maximal token rate of up to 1 M bps, as well as a 256 K bps guaranteed rate. The other pairs are running long-lived FTP applications, and have maximal token rates of up to 384 K bps with clean wireless links, but do not have guaranteed rates. The total core network bandwidth is limited to 1 M bps. The external Internet introduces a 30 ms delay to all hosts. We set the real-time packets with a higher priority over TCP packets in the core network to cater to their different delay requirements. In this network topology, the SGSN acts as the core router in the DiffServ domain.

Figures 4.3 and 4.4 show that after applying WERUF, the throughput of the real-time flow in the noisy wireless link improves significantly, when its offered load increases of more than 60%. Further, the end-to-end transfer delay reduces to less than 65%. Figure 4.5 shows the improvement to TCP flows. When the real-time flow's offered load is higher than 60%, the improvement for TCP flows also increases significantly, due to the network resources in the core network being released by the real-time flow.

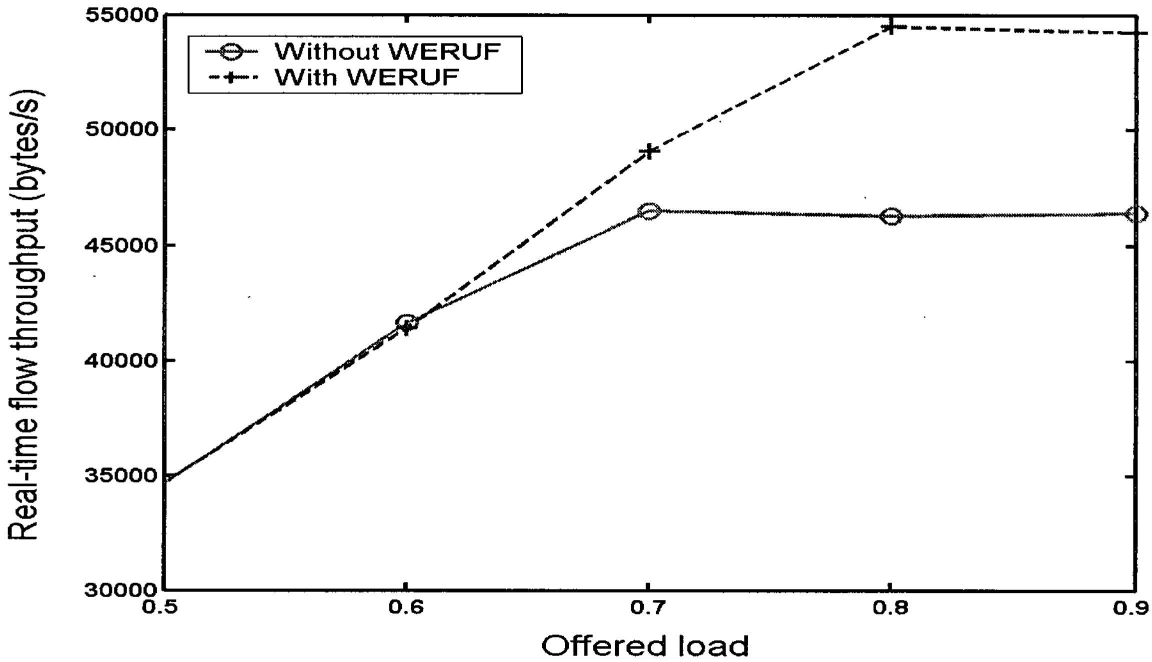


Figure 4.3 Real-time flow throughput vs. offered load.

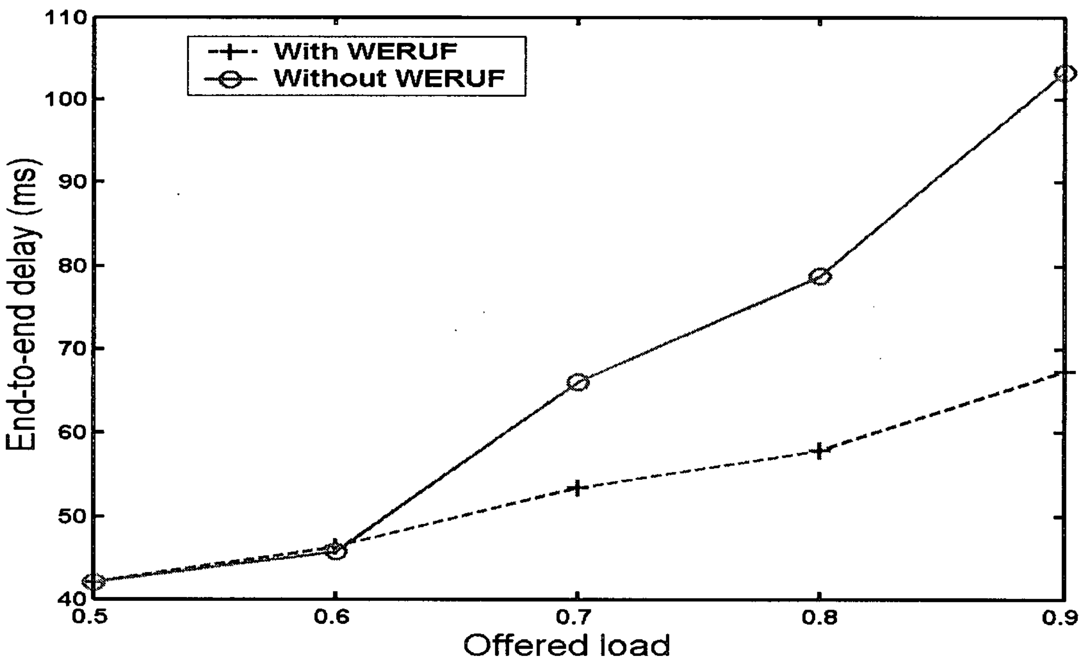


Figure 4.4 End-to-end delay of the real-time flow vs. offered load.

We can see that improvements only occur when the real-time traffic flow is heavy. In this case, the noisy wireless link is close to saturation and many real-time PDUs accumulate in the

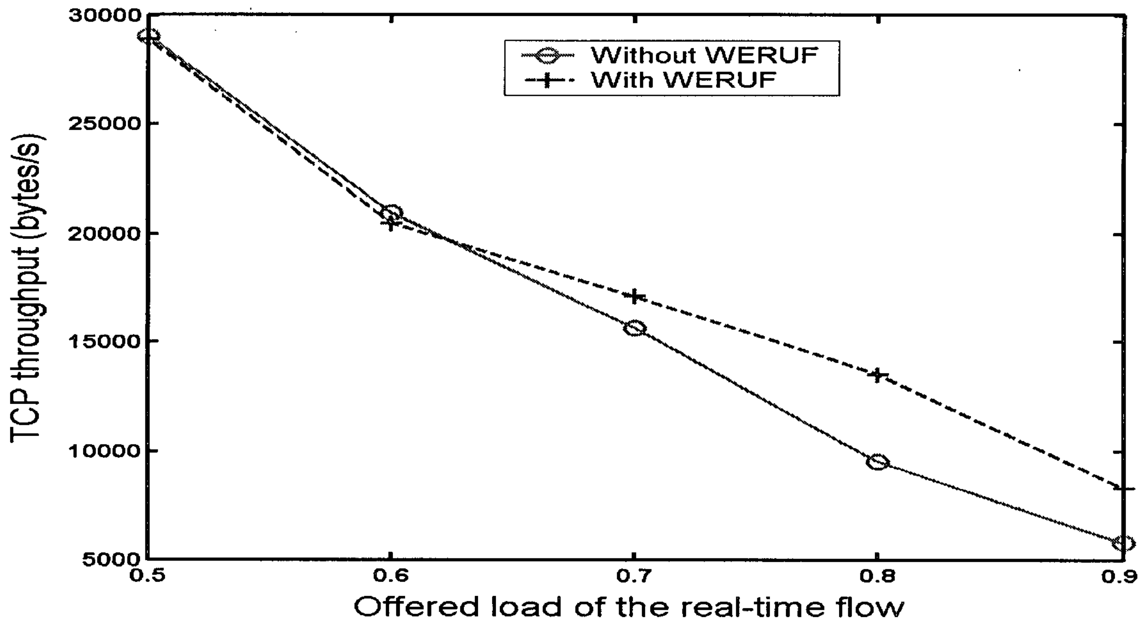


Figure 4.5 Total TCP throughput vs. real-time traffic offered load

RNC, the average transmission queue length is long enough to trigger the WERUF. On the other hand, there is much competition for the shared bandwidth/buffer in the core network, thus the saved resources due to WERUF can make a positive impact on overall performance.

Next, we investigate what causes to the improvements. In our simulation, the real-time packet drops due to transmission time exceeding MaxDAT are almost equal in both cases, so they are not presented in the figures. Figure 4.6 shows that without WERUF, most packets are dropped due to expiration, but with WERUF applied, packets drop due to overflow in the GGSN only, while none drop due to expiry (Figure 4.7). More importantly, the total packet drops in Figure 4.7 is less than the total packet drops in Figure 4.6. This results from our strategy of making the active packet drop rate (actually the ICMP packets' generating rates following the AQM function II here) less than the natural packet drop rate. The fewer expired drops results in more efficient resource utilization, discussed in Chapter 3.

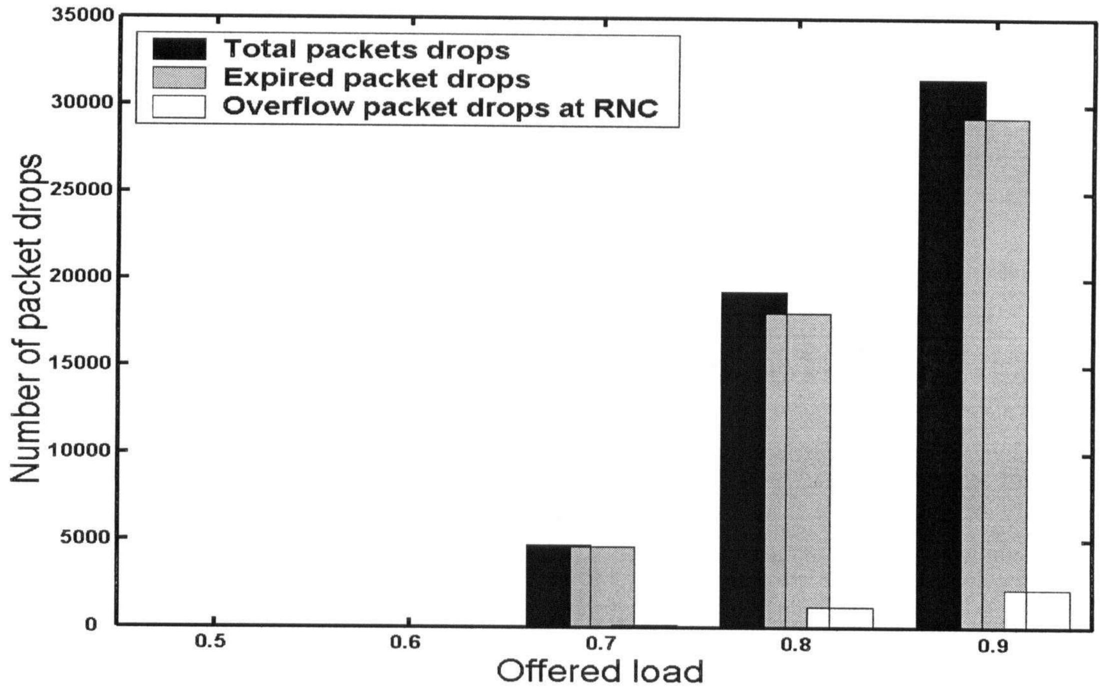


Figure 4.6 Real-time flow packet drops vs. offered load without WERUF.

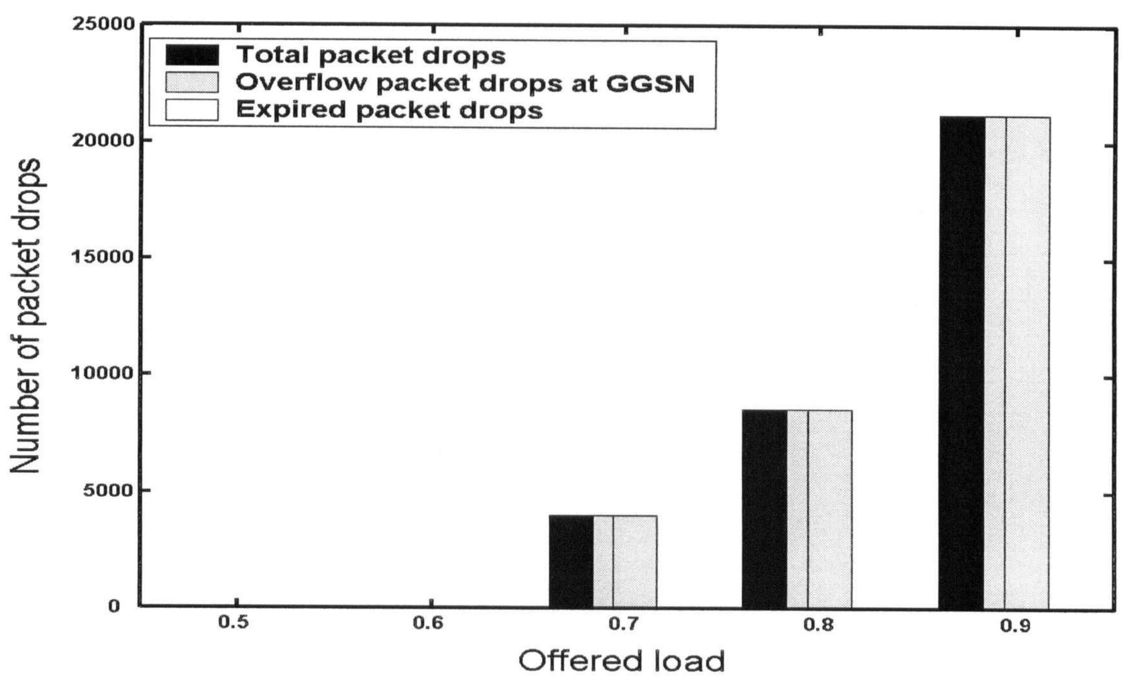


Figure 4.7 Real-time flow packet drops vs. offered load with WERUF.

This effect is similar to the packet drops described in the last chapter, except we also improve the performance of low priority TCP flows by early regulation. The undeliverable packets are dropped at the GGSN because of the overflow caused by a smaller token rate and limited buffer space. This mechanism has a drop-tail effect, thus, it is likely to drop packets continuously, contrary to the AQM schemes discussed in Chapter 3. Different real-time applications may favour different dropping disciplines. This topic would be interesting for future research.

#### 4.2.1.2 One Real-time Flow in a Noisy Link and Multiple TCP Flows in Clean Links with Different RTT

In this scenario, the Internet latency of the real-time flow (M1~H1 pair) is 30 ms, while the latencies from H2, H3,..., H6 to GGSN are 5 ms, 20 ms, 80 ms, 160 ms and 250 ms, respectively (Figure 4.8). We set the real-time traffic load to be constant at 80%. Again, the real-time

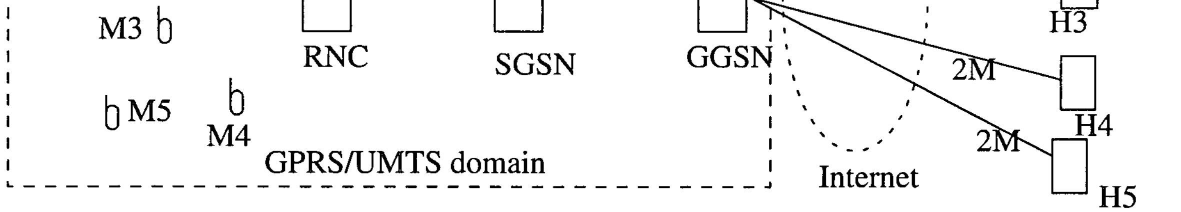


Figure 4.8 Network topology of one real-time and five TCP flows with different Internet latencies.

flow has a higher priority in the core network, but suffers from a noisy wireless link, while TCP flows with lower priority in the core network are sent over clean wireless links. Their token rates in GGSN and wireless data rates in RNC are the same as in section 4.2.1.1. The total core network bandwidth is 2M bps.

The simulations yield throughputs of 43357 bytes/s and 51719 bytes/s for the real-time flow without WERUF and with WERUF, respectively - an improvement of better than 19% for WERUF. The total packet drop rate with WERUF is also less than the packet expiry rate without WERUF. TCP throughput with different Internet latencies are shown in Figure 4.9. Here the

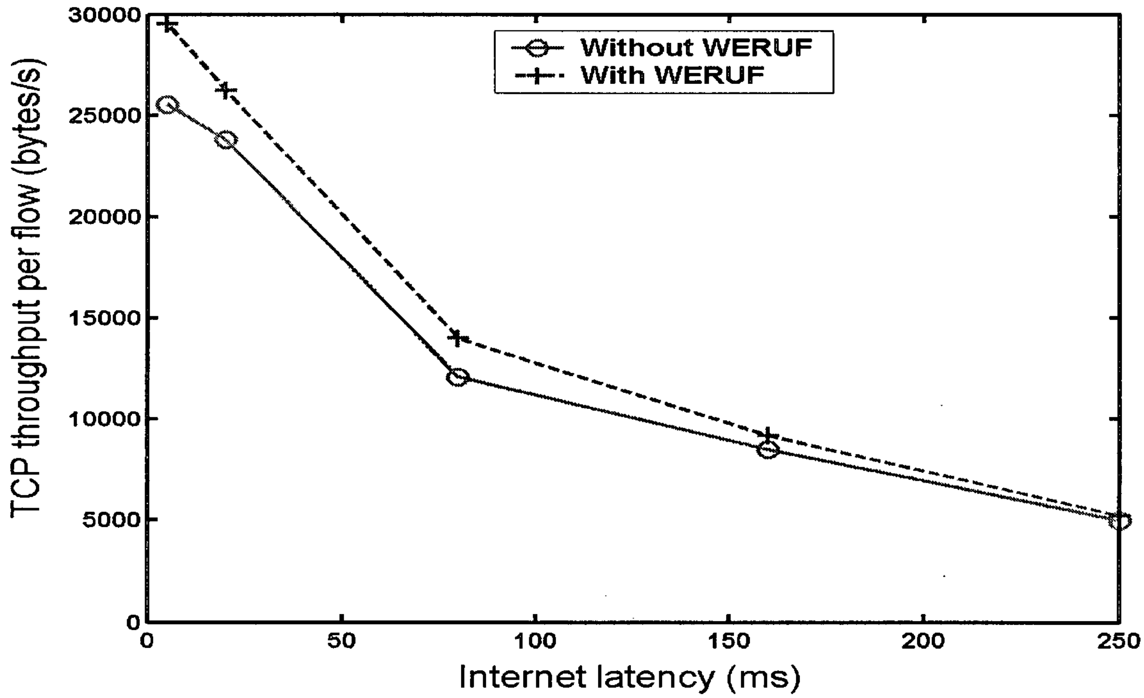


Figure 4.9 TCP throughput per flow vs. Internet latency.

improvement favors TCP flows with smaller Internet latency. Because TCP uses ACKs to "clock out" new segments [8], flows with shorter RTTs occupy more core network bandwidth given up by the real-time flow under WERUF.

### 4.2.2 Real-time Flows Over Different Link Conditions

These scenarios show cases when multiple real-time flows compete for limited core network bandwidth. WERUF is applied to their transmission queues at RNCs.

#### 4.2.2.1 Two Real-time Flows in Different Links

In this scenario, only two real-time flows are investigated (Figure 4.10). These two real-

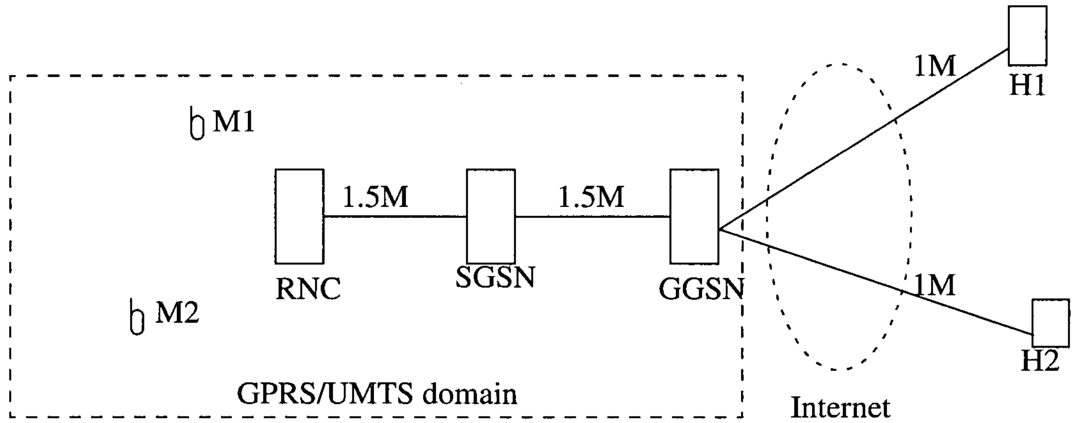


Figure 4.10 Network topology of two real-time flows.

time flows both get a 1 M bps token rate in the GGSN, and the same Internet latency of 30 ms. The flow in the noisy link gets a higher priority than the one in the clean link, that is, their packets in the core network are handled by different drop precedences. The purpose of this setting is to more easily show the effects of WERUF. When these two flows have the same drop precedence when competing for core network resources, most packets are dropped by the RIO dropper in the core network, while the surviving packets arriving the RNC are not dense enough to accumulate a long enough queue to trigger the WERUF.

The Internet latency is 30ms for both flows. The other parameters are the same as in Table

4.1.

From Figures 4.11 to 4.14, we can see that WERUF not only significantly improves the low drop precedence real-time flow performance in the noisy link, but also the performance of the high drop precedence flow in the clean link, under heavy load condition.

From Figures 4.11 and 4.13, we can see that without WERUF, when the traffic load

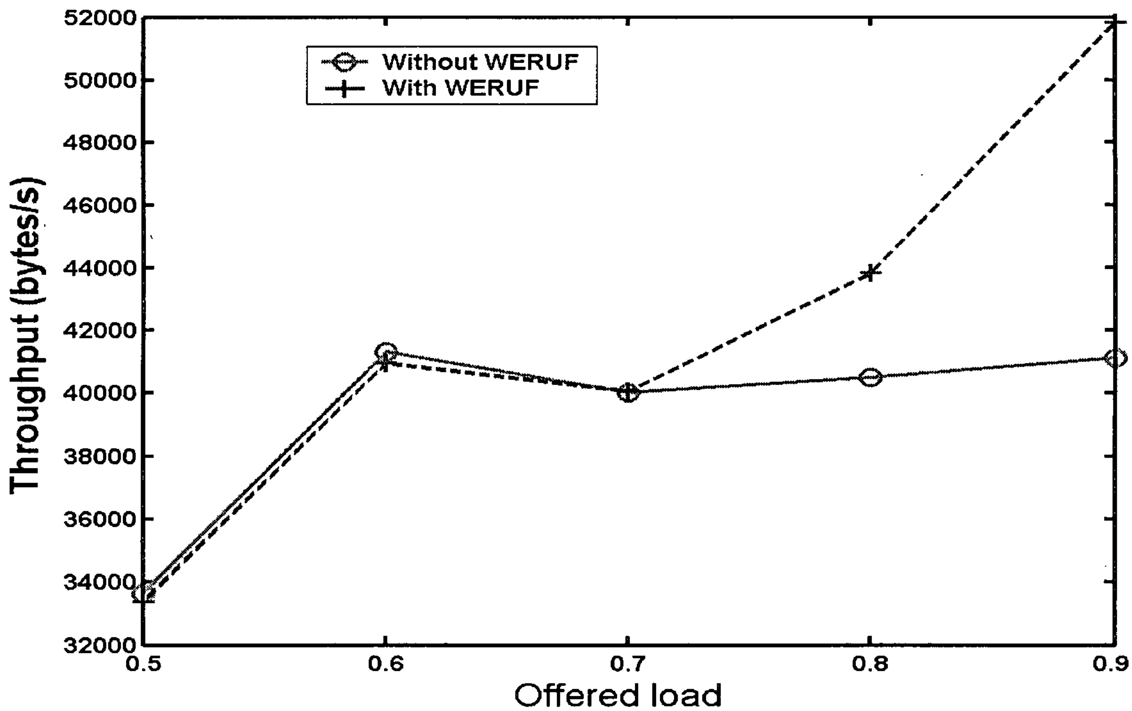


Figure 4.11 Throughput of the low drop precedence real-time flow in a noisy link vs. offered load.

becomes heavy, the low drop precedence flow in a noisy link approaches to saturation, that is, from 70% to 90%, the throughput hardly increases, while the end-to-end delay grows. But with WERUF, its throughput continues to increase with the offered load, and the incremental end-to-end delay is less than that without WERUF.

Without WERUF, the performance of the high drop precedence flow in a clean link

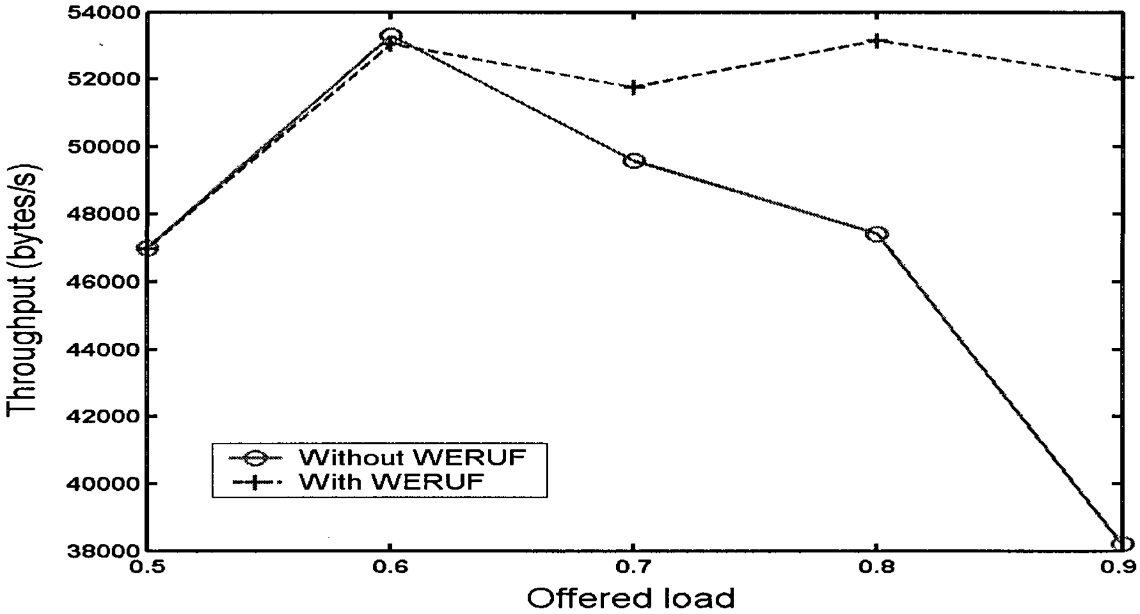


Figure 4.12 Throughput of the high drop precedence real-time flow in a clean link vs. offered load.

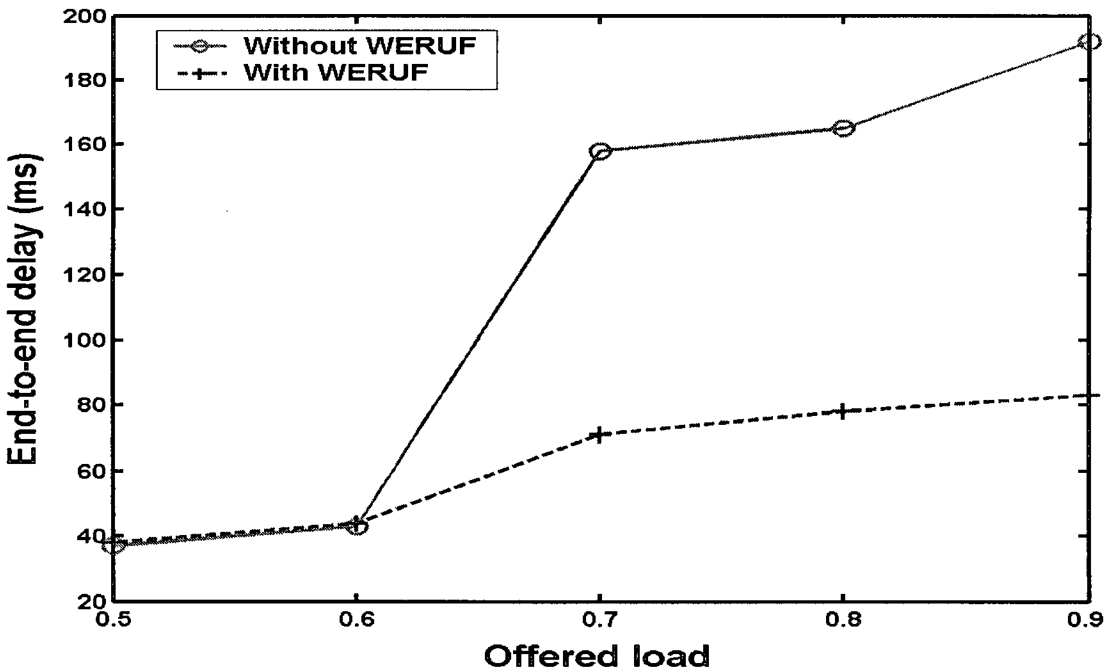


Figure 4.13 End-to-end delay of the low drop precedence real-time flow in a noisy link vs. offered load.

degrades when the total traffic load increases (Figure 4.12 and Figure 4.14), because packets of

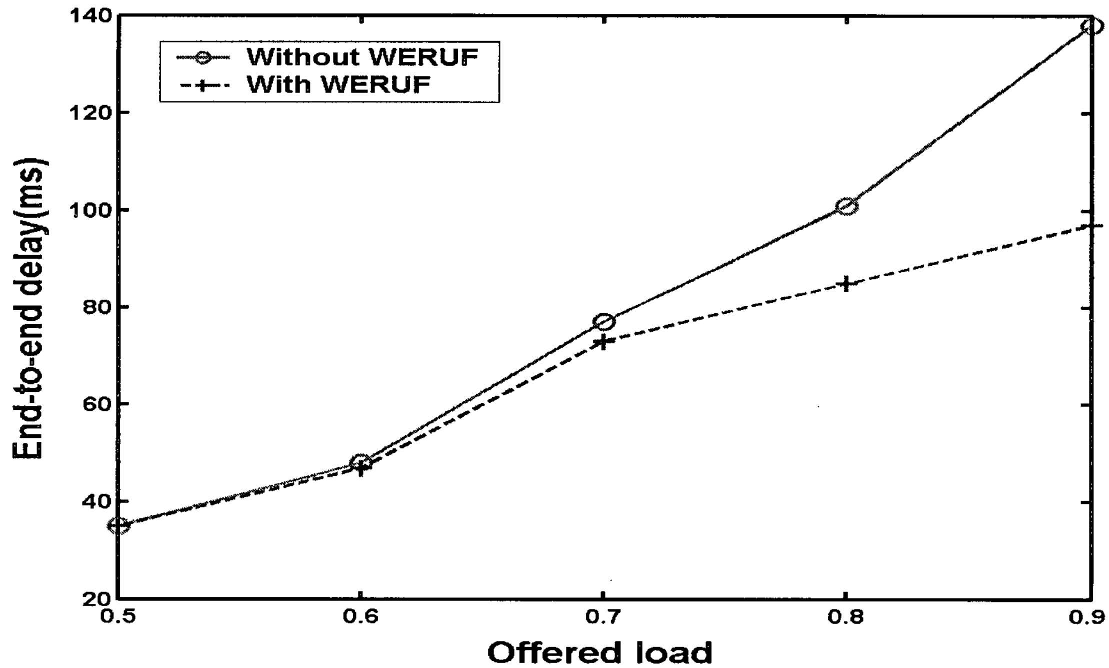


Figure 4.14 End-to-end delay of the high drop precedence real-time flow in a clean link vs. offered load.

the low drop precedence flow uses up shared resources of the core network. Many of these packets are actually undeliverable due to expiration, thus network resources are wasted. With WERUF, these undeliverable packets are dropped earlier at GGSN, network resources are utilized more efficiently, and the performance of this high drop precedence flow significantly improves.

Figure 4.15 and Figure 4.16 show the packet drops of the low drop precedence flow in the noisy link. The total number of packet drops in these two figures does not include those drops made due to transmission times exceeding MaxDAT, which are roughly equal in both cases for the same $P_B$ and offered load.

Obviously, WERUF converts the expired packet drops and overflow at the RNC to the overflow packet drops at the GGSN. Earlier packet drops prevent undeliverable packets from

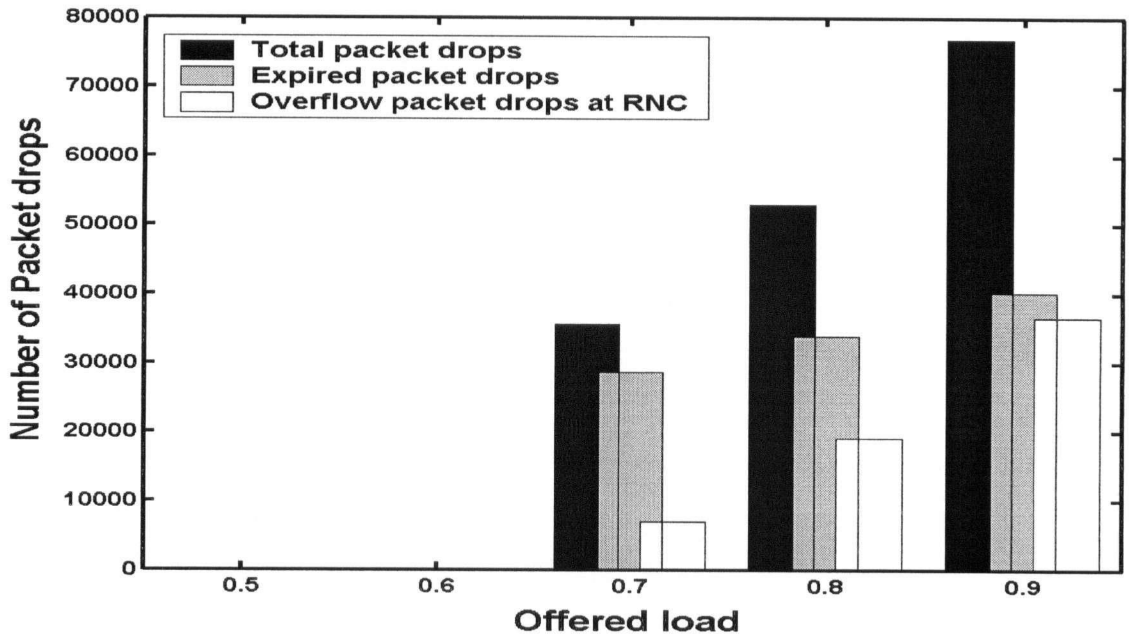


Figure 4.15 Packet drops of the low drop precedence real-time flow in a noisy link vs. offered load without WERUF.

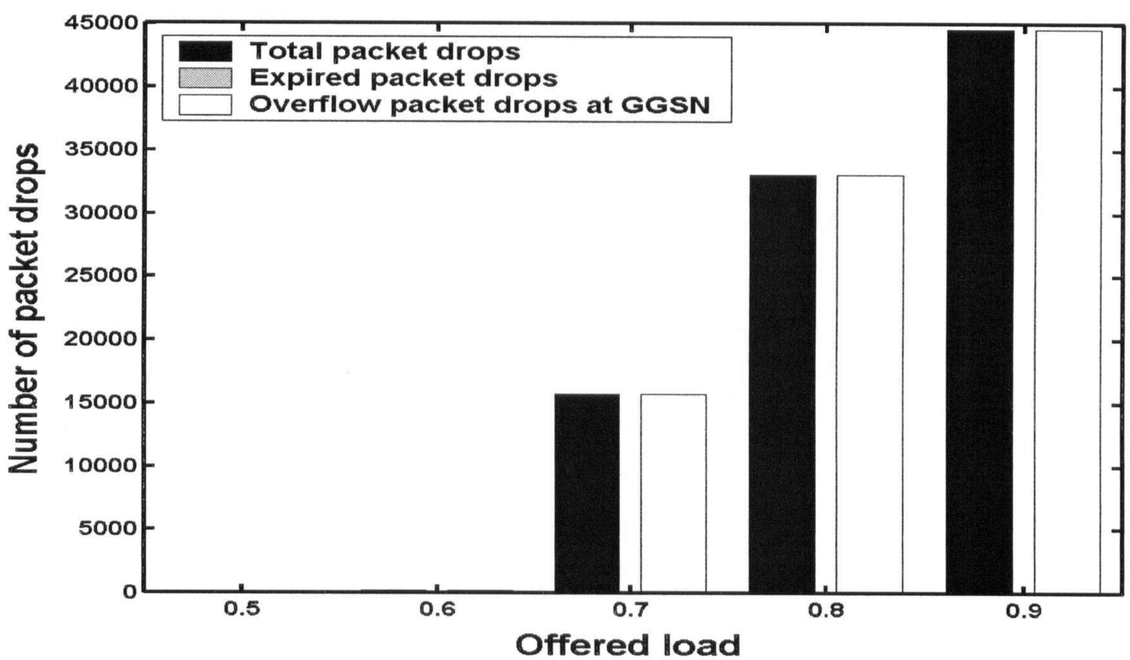


Figure 4.16 Packet drops of the low drop precedence real-time flow in a noisy link vs. offered load with WERUF.

occupying the limited core network resources, and improve overall performance. Another phenomenon worth mentioning is that for this flow (with low drop precedence, and over the noisy link) the total packet drops in WERUF are less than without WERUF, which guarantees no observable QoS loss due to the introduction of WERUF from end users' viewpoint.

#### 4.2.2.2 Multiple Real-time Flows in Different Wireless Links

The network topology of this scenario is illustrated in Figure 4.17 below. Five real-time

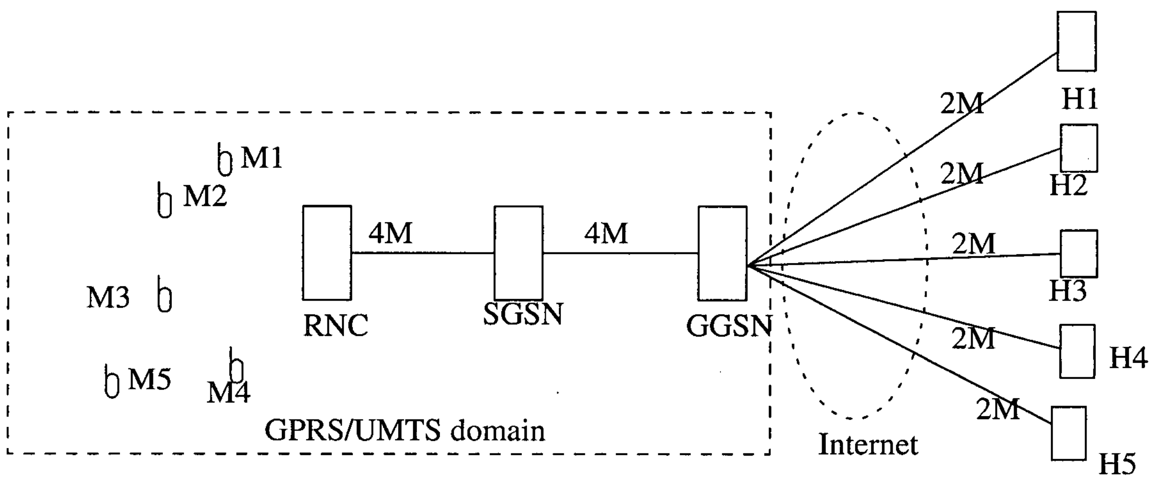


Figure 4.17 Network topology of multiple real-time flows in noisy and clean links.

flows come from five Internet sources, with the same Internet latency of 30 ms. The maximum token rate of each of them is 1M bps, and the guaranteed token rate is 256K bps. The other parameters are listed in Table 4.1.

We set the offered load of these flows to 90%, and vary the number of noisy links from 1 to 4. As in the last scenario, in the core network we assign a low drop precedence to those flows experiencing noisy wireless links, and assign a high drop precedence to flows experiencing clean links. The total throughput of flows in noisy and clean links is measured. The result is demonstrated in Figure 4.18.

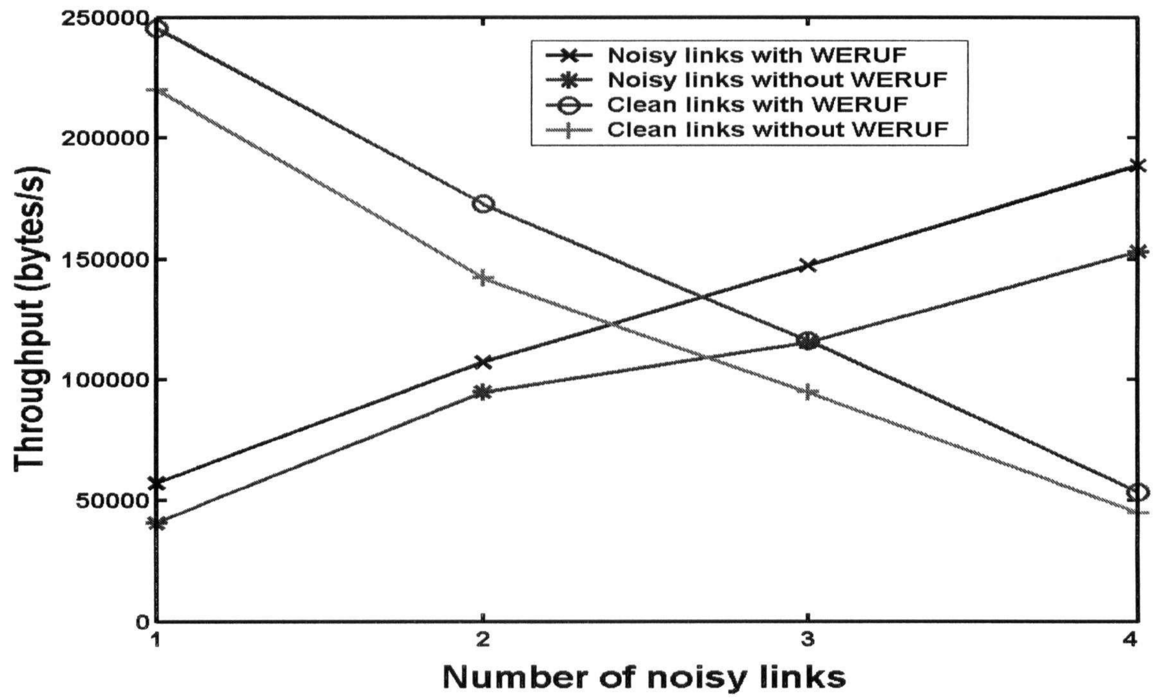


Figure 4.18 Total throughput in noisy and clean links vs. number of noisy links

In this figure, the number of noisy links increases from left to right (1 to 4). Clearly, WERUF improves the performance not only in noisy links but also in clean ones.

The trend of the improvement is interesting. The improvement of clean link flows gradually shrinks with the growth of the number of noisy links. As the number of flows over clean links decreases, the traffic load on these flows becomes too light to fully utilize core network resources made available by constrained flows over noisy links. On the other hand, the improvement of noisy link flows grows with the number of noisy links, because early regulation is triggered in more and more flows' transmission queues, showing a greater and greater positive effect.

## 4.3 Summary

In this chapter, we extend the idea of active packet drops discussed in the last chapter from the wireless-wireline interface nodes to the whole GPRS/UMTS domain by Wireless Early Regulation of Unresponsive Flows (WERUF). We utilize DiffServ functional elements at the edge nodes to restrict real-time flows with hard time deadlines, so as to release shared core network resources to other flows. These specific real-time flows are identified by the transmission queue length at the wireless-wireline interface nodes with the probability generated by the piece-wise linear AQM functions. We also propose some important rules for the rate adjustment at the edge nodes to make the scheme work efficiently. In this way the congestion caused by wireless error at the interface nodes can be propagated to the edge nodes.

Simulations are excised in a wide spectrum of scenarios. The results prove the WERUF can not only significantly improve the QoS of congested real-time flows experiencing high traffic load and wireless error rates, but can also improve the QoS of other flows when shared core network resources are scarce.

# Chapter 5 Conclusions and Future Work

In this thesis, we have analyzed the relationship between the real-time expired packet drop rate and the transmission queue length, when link layer retransmissions are used to recover packet losses over the wireless link. We propose a novel active queue management method for the transmission queue that actively drops potentially expiring packets before they are enqueued, in order to free network resources and control transmission queue lengths effectively. Using the RLC layer, defined in UMTS as specified by the 3GPP as an example, we have presented simulation results to show that the proposed AQM method is effective in reducing queuing delays and eliminating expiration packet drops. This improves overall system performance by increasing throughput and reducing end-to-end delay. Analysis and simulations have been made to prove the effectiveness of this scheme.

In a similar vein, we extend source quenches to DiffServ ingress edge nodes, namely the Gateway GPRS Support Node (GGSN) in the GPRS/UMTS domain, to prevent undeliverable packets from consuming shared network resources in core networks, thus utilizing network resources more efficiently and improving overall system performance. We have proposed a new mechanism, Wireless Early Regulation of Unresponsive Flows (WERUF) in this thesis, which converts the undeliverable expired packet drops to overflow packet drops that can be applied to packet ingress nodes, freeing the network resources for other flows. This mechanism is a variation of active queue management. Simulation results have been presented to prove that this scheme can efficiently improve the end-to-end QoS of GPRS/UMTS networks. It achieves its highest efficiency when the real-time traffic load is heavy and the core network resources are limited. We believe the mechanism can be applied to any wireless communication systems with link layer retransmissions and shaped traffic.

The drawbacks of these mechanisms include needed extra computing power and uplink bandwidth for the source quenches. We minimize computing complexity by choosing simple active queue management functions at the wireless-wireline interface node, and using a simple token rate operation at the edge nodes. However, the impact of such actions on computing power is still unknown and needs further investigation. We believe the extra uplink bandwidth consumed by the source quenches has a minor effect, because in GPRS/UMTS, the uplink traffic is normally slight, thus the core network bandwidth of the uplink is redundant in most cases. The source quenches can also be treated as flow control signals, therefore, it is possible to implement them in control plane protocols, which can be guaranteed by the system [12].

While this paper does not distinguished between different types of packets that may exist in real-time flows, there may be a need for differentiation to be made, because in many real-time applications, some types of packets carry more important information than others. In these cases, Random Early Detection with IN-and-OUT (RIO) [25] can be used as a basis for our AQM method, at wireless-wireline interface nodes, to implement differentiation. Furthermore, the best way to apply AQM effectively to a mix of different real-time applications remains to be investigated.

As we mentioned in the previous chapter, the WERUF drops packets in a drop-tail mode. The side-effects of this dropping discipline to some real-time applications may need further analysis. Again, for those packets carrying more important information than others, some effective mechanisms will be developed to avoid inappropriate drops.

Another question that may prove interesting to research is what impact the resource management schemes have on our proposed mechanism. Different bandwidth reservation methods used by scheduling, CAC and MAC eventually affect resources assigned to a single flow, which results in varying service rates and times. In this thesis we have ignored these impacts for the sake of clarity so that we can focus on the presentation of our idea. Further analysis is needed, however, to show their effect.

In this thesis, we have successfully presented the problem of an absence of buffer management schemes at the link layer transmission queue, and a novel solution to the problem. We hope our work attracts attention to this area to resolve these important problems.

# Bibliography

[1] The Internet Engineering Task Force, http://www.ietf.org.

[2] J. Wroclawski, "The Use of RSVP with IETF Integrated Services.", RFC 2210, http://www.ietf.org.

[3] J. Wroclawski, "Specification of the Controlled-Load Network Element Service.", RFC 2211, http://www.ietf.org.

[4] S. Shenker, C. Partridge and R. Guerin, "Specification of Guaranteed Quality of Service.", RFC 2212, http://www.ietf.org.

[5] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang and W. Weiss, " An Architecture for Differentiated Services", RFC 2475, http://www.ietf.org.

[6] 3G TS 22.100 V3.7.0 (2001-10), "UMTS phase 1 Release 99", http://www.3gpp.org.

[7] 3G TS 23.107 v5.3.0 (2002-01), "QoS Concept and Architecture" , http://www.3gpp.org.

[8] W. R. Stevens: TCP/IP Illustrated, Vol. 1: The Protocols. Addison-Wesley, 1994.

[9] J. B. Cain and D. N. McGregor, "A Recommended Error Control Architecture for ATM Networks with Wireless Links", IEEE Journal on Selected Areas in Communications, vol. 15, pp. 16-28, Jan. 1997.

[10] N. Guo and S. D. Morgera, "Frequency-Hopped ARQ for Wireless Data Services", IEEE Journal on Selected Areas in Communications, vol. 12, pp. 1324 -1337, Oct. 1994.

[11] J. J. Spilker: Digital communications by satellite. Prentice-Hall, c1977.

[12] 3GPP TS 23.060 V3.10.0 (2002-01), “General Packet Radio Service (GPRS); Service description; Stage 2 (Release 1999)”, http://www.3gpp.org.

[13] S. Floyd and V. Jacobson, “Random Early Detection Gateways for Congestion Avoidance”, IEEE/ACM Transactions on Networking, vol. 1, pp. 397-413, Aug. 1993.

[14] S. Floyd and K. Fall, “Router Mechanisms to Support End-to-end Congestion Control”, http://www.icir.org/floyd/papers/collapse.ps.

[15] S. Floyd and K. Fall, “Promoting the Use of End-to-end Congestion Control in the Internet”, IEEE/ACM Trans. on Networking, vol. 7 no. 4 , pp. 458 -472, Aug. 1999.

[16] A. Acharya and A. Rangarajan, “ERUF: Early Regulation of Unresponsive Best-effort Traffic”, in Proc. IEEE ICNP’99, Oct. 1999.

[17] K. Nichols, V. Jacobson, and L. Zhang, "A Two-bit Differentiated Services Architecture for the Internet", Internet Draft, ftp://ftp.ee.lbl.gov/papers/dsarch.pdf.

[18] 3GPP TS 25.322 V4.2.0 (2001-09), “RLC Protocol Specification (Release 4)”, http://www.3gpp.org.

[19] K. Bala, I. Cidon and K. Sohraby, “Congestion Control for High Speed Packet Switched Networks”, in Proc. INFOCOM’90, vol. 2, pp. 520-526, San Francisco, Apr. 1990.

[20] I. Stoica, S. Shenker and H. Zhang, "Core-Stateless Fair Queueing: Achieving Approximately Fair Bandwidth Allocations in High Speed Networks", in Proc. ACM SIGCOMM'98, Vancouver, Sep. 1998.

[21] J. Heinanen, F. Baker, W. Weiss and J. Wroclawski, " Assured Forwarding PHB Group.", RFC 2597, http://www.ietf.org.

[22] S. Floyd, "Discussion of Setting Parameters", http://www.icir.org/floyd/REDparameters.txt.

[23] D. D. Clark, and W. Fang, "Explicit Allocation of Best-Effort Packet Delivery Service", IEEE/ACM Transactions on Networking, vol. 6, pp. 362 -373, Aug. 1998.

[24] V. Rosolen, O. Bonaventure and G. Leduc, "A RED Discard Strategy for ATM Networks and its Performance Evaluation with TCP/IP Traffic", ACM Computer Communication Review, vol. 29, no. 3, Jul. 1999.

[25] C. C. Chao and W. Chen, "Connection admission control for mobile multiple-class personal communications networks", IEEE Journal on Selected Areas in Communications, vol. 15, pp. 1618-1626, Oct. 1997.

[26] M. Naghshineh and M. Schwartz, "Distributed call admission control in mobile/wireless networks", IEEE Journal on Selected Areas in Communications, vol. 14, pp. 711-717, May 1996.

[27] 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, Jun. 1993.

[28] D. A. Eckhardt and P. Steenkiste, "Effort-limited fair (ELF) scheduling for wireless networks", in Proc. INFOCOM'00, vol. 3, pp. 1097-1106, Israel, Mar. 2000.

[29] S. Lu, V. Bharghavan and R. Srikant, "Fair scheduling in wireless packet networks", IEEE/ ACM Transactions on Networking, vol. 7, pp. 473-489, Aug. 1999.

[30] S. L. Tsao, "Extending earliest-due-date scheduling algorithms for wireless networks with location-dependent errors", in Vehicular Technology Conference, vol. 1, pp. 223-228, Boston, Sep. 2000.

[31] M. Zorzi, "Packet dropping statistics of a data-link protocol for wireless local communications", in IEEE 6th International Conference on Universal Personal Communications Record, vol. 2, pp. 536 - 540, San Diego, Oct. 1997.

[32] Y. Uooyeol, P. Seongsoo and P.S. Min, "Performance analysis of data transmission in W-CDMA system", in IEEE 54th Vehicular Technology Conference, vol. 3, pp. 1584 - 1588, Atlantic City, Oct. 2001.

[33] R. Fantacci, "Queuing Analysis of the Selective Repeat Automatic Repeat Request Protocol Wireless Packet Networks", IEEE Transactions on Vehicular Technology, vol. 45, pp. 258 - 264. May 1996.

[34] E. Chan and X. Hong, "Analytical Model for an Assured Forwarding Differentiated Service over Wireless Links", IEE Proc. on Communications, vol. 148, pp. 19 -23 Feb. 2001.

[35] E. N. Gilbert, "Capacity of a Burst-noise Channel", Bell System Technical Journal. vol. 39, pp. 1253-1265, Sep. 1960.

[36] E. O. Elliott, "Estimates of Error Rates for Codes on Burst-noise Channels", Bell System Journal, vol. 42, pp: 1977-1997, Sep. 1963.

[37] S. Bhagwat, D. Tipper, K. Balakrishnan and A. Mahapatra. "Comparative Evaluation of Output Buffer Management Schemes in ATM Networks", in Proc. IEEE ICC'94, New Orleans, May, 1994.

[38] J. Postel, "Internet Control Message Protocol", RFC 777, http://www.ietf.org.

# Appendix A. List of Abbreviations and Acronyms

| | |
|---|---|
| **3GPP**: | 3rd Generation Partnership Project |
| **ACK**: | ACKnowledge |
| **AF**: | Assured Forwarding |
| **AM**: | Acknowledge Mode |
| **AMD**: | Acknowledged Mode Data |
| **AQM**: | Active Queue Management |
| **ARQ**: | Automatic Repeat Request |
| **BA**: | Behavior Aggregates |
| **BER**: | Bit Error Rate |
| **BSS**: | Base Station System |
| **CAC**: | Call Admission Control |
| **CN**: | Core Network |
| **CP**: | Complete Partitioning |
| **CS**: | Complete Sharing |
| **DiffServ**: | Differentiated Services |

**DSCP**: DiffServ CodePoint

**EDD:** Earliest-Due-Date

**ELF:** Effort-Limited Fair

**EPC**: Estimated PDU Counter

**ERUF**: Early Regulation of Unresponsive Flows

**FEC**: Forward Error Correction

**GGSN**: Gateway GPRS Support Node

**GPRS**: General Packet Radio Service

**GSM**: Global System for Mobile

**ICMP**: Internet Control Message Protocol

**ISP**: Internet Service Provider

**IP** : Internet Protocol

**IETF**: Internet Engineering Task Force

**IWFQ:** Idealized Wireless Fair-Queuing

**IntServ**: Integrated Services

**LB**: Leaky Bucket

| | |
|---|---|
| **MAC:** | Media Access Control |
| **MF:** | Multi-Field |
| **MS:** | Mobile Station |
| **MT:** | Mobile Terminal |
| **NAK:** | Negative Acknowledge |
| **PDN:** | Packet Data Network. |
| **PDP:** | Packet Data Protocol |
| **PDU:** | Protocol Data Unit |
| **PHB:** | Per-Hop Behavior |
| **PLMN:** | Public Land Mobile Network |
| **QoS:** | Quality of Service |
| **RED:** | Random Early Detection |
| **RIO:** | Random Early Detection with IN-and-OUT |
| **RLC:** | Radio Link Control |
| **RNC:** | Radio Network Controller |
| **SGSN:** | Serving GPRS Support Node |

| | |
|---|---|
| **SLA**: | Service Level Agreement |
| **SUFI:** | Super-Fields |
| **TCP**: | Transmission Control Protocol |
| **TE**: | Terminal Equipment |
| **UMTS**: | Universal Mobile Telecommunications System |
| **UTRAN**: | UMTS Terrestrial Radio Access Network. |
| **WERUF**: | Wireless Early Regulation of Unresponsive Flows |
| **WFQ:** | Weighted Fair Queuing |

# Appendix B. Simulation models in OPNET 8.1

OPNET provides a comprehensive development environment supporting the modeling of communication networks and distributed systems. It is a discrete-event driven simulator. There are three modeling domains (Table B.1) in OPNET: Network, Node and Process modeling environments. These three modeling domains span all the hierarchical levels of a model.

Table B.1 OPNET modeling domains.

| Domain | Editor | Modeling Focus |
|---|---|---|
| Network | Project | Network topology described in terms of subnetworks, nodes, links, and geographical context. |
| Node | Node | Node internal architecture described in terms of functional elements and data flow between them. |
| Process | Process | Behavior of processes (protocols, algorithms, applications), specified using finite state machines and extended high-level language. |

The topology of a system consists of both an inventory of its devices and the communications links between them. The network domain refers to these devices as nodes, and has several types of links that can be defined to connect them together for the purpose of sending information between them. Groups of nodes and links can be used to form subnetworks, and subnetworks can in turn contain lower-level subnetworks to form unlimited hierarchies. Some system topologies can be entirely represented in the node domain by interconnecting the node level building blocks, called modules. In most cases the network domain is used to represent high-level system topology because of its direct support for unlimited depth hierarchies, sophisticated communication links, and measured physical layout.