The Pennsylvania State University

The Graduate School

Department of Computer Science and Engineering

POWER-AWARE AND QOS-AWARE RESOURCE MANAGEMENT IN WIRELESS NETWORKS

A Thesis in

Computer Science and Engineering

by

Hao Zhu

Submitted in Partial Fulfillment
of the Requirements
for the Degree of

Doctor of Philosophy

August 2004

We approve the thesis of Hao Zhu.

Date of Signature

Guohong Cao
Associate Professor of Computer Science and Engineering
Thesis Adviser
Chair of Committee

Chita R. Das
Professor of Computer Science and Engineering

Thomas F. LaPorta
Professor of Computer Science and Engineering

George Kesidis
Associate Professor of Computer Science and Engineering

Aylin Yener
Assistant Professor of Electrical Engineering

Raj Acharya
Professor of Computer Science and Engineering
Head of the Department of Computer Science and Engineering

Abstract

With the advent of third generation wireless infrastructures and the rapid growth of wireless communication technology, people with battery powered mobile devices will have high-speed wireless data access at any time any place. However, due to the unique characteristics of wireless networks such as dynamic channel condition and limited energy supply, we cannot directly apply the existing resource management schemes designed for wired networks to wireless systems. Therefore, new techniques are needed for efficient resource management in wireless networks.

In this thesis, we have designed and evaluated new scheduling schemes and medium access control (MAC) protocols that focus on power efficiency, QoS provision, fairness and bandwidth utilization in wireless networks. We first concentrate on prolonging the battery life of the mobile terminal by reducing the power consumption of the wireless network interface. Each mobile terminal uses its proxies to buffer data so that the wireless network interface can sleep for a long time period and tolerate the state transition delay. The base station applies our proposed scheduling schemes to service each flow in a power efficient way without loss of QoS requirements, and alleviates the impact of channel errors on QoS provision.

We then design the absence compensation fair queuing (ACFQ) model to improve service differentiation under bursty data traffic in wireless networks. To achieve good service differentiation without loss of QoS provision for each flow, an absence compensation model and an error compensation model are integrated into ACFQ. By the

well-controlled absence compensations, ACFQ can provide much better service differentiation than the legacy fair queuing models (e.g. the weighted fair queuing) under bursty data traffic.

Finally, we propose the relay-enabled MAC protocols in the context of IEEE 802.11 based wireless networks. With the physical layer multi-rate capability, data can be transmitted at different rates according to the channel condition. As a result, when the transmission rate between the sender and the receiver is low, while high rates are available between the sender and the relay node and between the relay node and receiver, data can be delivered much faster through relay. We design protocols that enable the relay mechanism in the point coordination function and the distributed coordination function respectively. Many optimizations are used to reduce control overhead and deal with channel utilization issues.

Throughout our study, we have paid considerable effort to deal with design difficulties introduced by limited power supply, dynamic channel conditions, effect of interference or capture, and stringent QoS requirements. We hope that such effort has allowed practical solutions for achieving high-speed data access and providing good QoS in wireless networks. Meanwhile, we theoretically analyze our schemes and extensively evaluate the performance via simulations to give a detailed understanding of the characteristics of our schemes. We hope the results have a long, valuable life.

Table of Contents

List of Tables

List of Figures

Acknowledgments

I would like to show my sincerest gratitude to my advisor, Professor Guohong Cao for giving me copious amounts of insightful guidance, constant encouragement, constructive criticism, and expertise on every subject that arose throughout all these years. His enthusiasm and dedication to his students are truly inspiring; it is my very privilege to have been one of them. From Professor Cao, I have learned how excellent networking research is done: rigorous, energetic, open to new ideas, and dedicated to research. I greatly appreciate and enjoy the collaboration with him. I would also like to thank Professors Chita Das, Thomas LaPorta , George Kesidis and Aylin Yener for being on my advisory committee and for all their help and insightful comments along the way.

I am very thankful for the many friends and officemates I have had at Penn State. Special thanks to Wensheng Zhang, Guilin Wang, Liangzhong Yin, Guangyu Chen, Jing Zhao, Xiaonan Lu, Ping Zhang, Jianyong Zhang, Lin Li, Hui Song, Allen Mathias, Xing Gao, Guilin Chen, Xidong Deng, Yingqi Xu, and Heesok Kim for many stimulating discussions and warm friendship. I would also like to thank many other friends, especially Jianrui Wang, Yizhong Qu, and our tennis team; my years at Penn State would not have been the same without them. A special thank is given to Bryan Spang, his wife Stacy and his four lovely children, with whom I spent every Christmas here.

Finally, I want to thank my wife, Xiaoli Liang, for her love, support, encouragement, sacrifice, sense of humor, and for being a wonderful wife. I could not have accomplished this without her. Very special thanks to my parents and parents-in-law for their selfless love and support, and to my two elder brothers for their encouragement and love.

Chapter 1

Introduction

With the advent of third generation wireless infrastructures and the rapid growth of wireless communication technology, people with battery powered mobile devices will have high-speed wireless data access at any time any place. However, due to the unique characteristics of wireless networks such as dynamic channel condition and limited energy supply, we cannot directly apply the existing resource management schemes designed for wired networks to wireless systems. Therefore, new techniques are needed for efficient resource management in wireless networks.

1.1 The Growth of Wireless Networks

The first generation (1G) wide-area wireless networks, such as the advanced mobile phone systems (AMPS) and the total access communication services (TACS) [23, 41], first became commercially available in the early 1980s. The main mobile service provided by 1G radio system are analog and circuit-switched voice services. With the development of digital signal processing and transmission technologies, the second-generation (2G) wireless networks (e.g. GSM and IS-95 [15, 23, 41]) began to emerge in early 1990s and evolved to replace 1G wireless networks. Compared to 1G, 2G wireless networks have a number of significant advantages. For example, digital technologies increased

channel capacity and radio spectrum utilization, enhanced voice quality and reduced power consumption of mobile terminals.

As mobile data services exponentially grow, 2G wireless networks have been enhanced to what are commonly referred to as 2.5G wireless networks, which can support higher channel capacity and per-user data rates. For example, GSM has been enhanced to general packet radio services (GPRS). By allocating more than one time slot in parallel to a user, the maximum theoretical date rate of GPRS can reach 171.2 Kbps. Later, with the advanced modulation and channel coding techniques, the enhanced data rates for GSM (EDGE) systems have been proposed to achieve data rates up to 384 Kbps [1].

To provide higher data rates than 2.5G, in the late 1990s, standardization efforts for the third-generation (3G) wireless networks (e.g. UMTS [1], CDMA2000 [2]) began. Compared to 2/2.5G wireless systems, 3G wireless networks significantly increases the channel capacity and per-user data rates. 3G systems can support data rates up to 144Kbps to users moving at vehicular speeds, up to 384 Kbps to users moving at pedestrian speeds, and up to 2 Mbps to stationary users. Further, 3G systems are designed to support a broader range of IP-based mobile services than 2G systems. Seamless integration between 3G wireless networks and the Internet enables mobile users to access the abundant data services in the Internet. Another important advantage of 3G systems is better interoperability than 2G systems to support roaming among different service providers, different radio technologies and different countries.

Another important evolution path of wireless networks is the wireless local area networks (WLANs) [15, 30]. With the advantage of low cost and high data rate, the

[1] Due to the high speed, some people also refer EDGE as a 3G system.

WLAN market is exploding. The most popular WLANs are based on IEEE 802.11 standards [30]. IEEE 802.11b WLANs operate in the 2.4 GHz band at speed up to 11 Mbps, and IEEE 802.11a WLANs operate in 5.0 GHz band and can support data rate as high as 54 Mbps [15]. Recently, IEEE 802.11g whose performance is the same as IEEE 802.11a is becoming to replace IEEE 802.11b [20]. In order to support real-time applications in WLANs, IEEE 802.11e standard [26] has been proposed to support real-time traffic in WLANs. With the dramatic development of 3G and WLANs, there are lots of efforts on the integration of 3G networks and WLANs to build a cost-effective next generation wireless network [46, 67].

1.2 Challenges

The exponential increase of bandwidth demands, and the stringent power limitation of mobile terminals (MTs) have made it a big challenge to perform efficient resource management in wireless networks. Below we describe four major challenges:

1. Part of the difficulty in resource management in wireless networks is due to the time-varying channel condition. Generally, the signal observed at the receiver is related to the wavelength of the signal, the background noise, the interference from other ongoing transmissions, the spatial locations of the sender and receiver, the reflecting objects, and the orientation of the antenna of the sender and receiver [20, 54]. The superposition of these factors could enhance or dramatically degrade the channel condition. Thus, if the impact of channel errors is not carefully managed,

the utilization of the radio spectrum may be significantly decreased due to a large number of retransmissions.

2. Another difficulty comes from the limited power supply of small-size and low-cost MTs. Most MTs are powered by battery, but the rate at which battery performance improves is fairly slow [50]. Aside from major breakthroughs, it is doubtful that significant improvement can be expected in the foreseeable future. As a result, it is a challenge to design new techniques so that the MTs can perform the same functions and provide the same services, while minimizing their overall power consumption.

3. The third difficulty is how to provide good QoS (e.g. fairness, throughput) in wireless networks considering dynamic channel condition and bursty nature of the data traffic. There exists a tradeoff between the system throughput and fairness. Providing a strictly equal service (in bits) to each flow could significantly decrease the system throughput, because many time slots would be wasted by serving flows with poor channel condition. This problem becomes more challenging under bursty data traffic. By following the fluid fair queuing [17, 71], existing fair queuing service models cannot provide good service differentiation under bursty data traffic.

4. The fourth difficulty comes from how to efficiently utilize the multi-rate capability in wireless networks. With physical layer multi-rate capability, the transmission rate can be adapted according to the channel condition. When the link quality is good, data can be delivered faster at a higher rate. However, when the link quality between the sender and the receiver is poor and has a low rate, the system

throughput would be significantly reduced, since it would take much longer to transmit the data. In many cases, due to interference, mobility, path loss and so on, the channel condition between the sender and the receiver may often be poor, which would cause severe performance degradation. Therefore, to improve system performance, it is not enough to only exploit the multi-rate capability according to the channel condition of the direct link.

1.3 Thesis Overview

In this thesis, we design and evaluate new scheduling algorithms and IEEE 802.11 medium access control (MAC) protocols to address: 1) how to reduce the power consumption of the wireless network interface (WNI) of the MT without losing QoS provision to each flow; 2) how to improve service differentiation under bursty data traffic in wireless networks; 3) how to further utilize the physical layer multi-rate capability and improve the system performance when the direct link quality between the sender and the receiver is poor. For the first topic, we study two novel scheduling algorithms with different emphases. For wireless systems without rate adaptation capability, we propose a *priority-based bulk scheduling* service model to reduce the power consumption of the WNI without losing QoS provision. For the wireless systems with rate adaptation capability, we design a *rate-based bulk scheduling* service model to reduce the power consumption of the WNI with the emphases on error resiliency. For the second issue, we propose an *absence compensation fair queuing* service model to improve the service differentiation under

bursty data traffic and, still maintain QoS provision for each flow. For the third issue, we propose relay-enabled MAC protocols to further exploit the multi-rate capability of the IEEE 802.11 based wireless networks. When the direct link between the sender and the receiver has low quality and low rate, data can be delivered much faster through a two-hop high-speed relay.

1.3.1 The Power-Aware and QoS-Aware Service Model

Since most mobile terminals (laptop computers, PDAs, hand-held computers, etc.) are powered by battery, many researchers work on techniques to reduce the power consumption of these mobile devices. As the wireless network interface (WNI) accounts for a large part of the power consumed by the mobile terminal (MT), it is important to carefully design communication protocols to reduce the power consumption of the WNI. In particular, we need to design new scheduling schemes with good power-efficiency. With the existing rate-based scheduling algorithms [71], it is difficult to when and how long WNI should enter sleep since each MT does not know the scheduling pattern due to lack of information of all flows in the system.

We propose a new scheduling scheme, called *priority-based bulk scheduling* (PBS) to utilize the client buffer to save power and provide QoS. Under PBS, the scheduler keeps track of the amount of prefetched (buffered) data for each flow. The flow with sufficient buffered data will be suspended until the buffered data runs out, whereas the flow with insufficient buffered data will be served based on its priority, which is determined by its delay requirement. With buffered data, the WNI can

sleep long enough to offset the impact of its state transition delay. By suspending some flows, other active flows sharing the channel can obtain more bandwidth and take less time to fill the buffer.

PBS is designed for some systems that do not use rate adaptation techniques to deal with channel errors (e.g. CDMA [24]). On the other hand, PBS does not provide error resiliency for flows with severe channel conditions. In parallel with the PBS scheme, we have designed another scheme called *rate-based bulk scheduling* (RBS) for the systems with rate adaptation capability, and focus on providing good QoS in lossy wireless environments. Compared with PBS, RBS applies a different scheduling algorithm to let each flow have more buffered data before being suspended to combat channel errors. To tolerate severe channel errors, a novel adaptive scheme is used to control the sleeping time of the WNI based on the channel condition.

Through analysis, we prove that both PBS and RBS can provide delay guarantees and serve each flow more power efficiently than conventional rate-based fair queuing models. Experimental results show that PBS achieves excellent QoS provision for each flow and significantly reduces the power consumption of mobile terminals.

1.3.2 The Absence Compensation Fair Queuing Model

Some researchers [32] have found that, under bursty data traffic, when the network utilization is not very high, weighted fair queuing (WFQ) only provides little service differentiation to the flows. The WFQ model emulates *fluid fair queuing* (FFQ) [17]

which does not compensate the service loss of a flow due to absence; that is, after the low-weight flows take extra service from the absent high-weight flows, these high-weight flows cannot reclaim the service loss when they become backlogged again.

To achieve better service differentiation with QoS provision, we propose a new service model called *absence compensation fair queuing* (ACFQ). Fundamentally different from other wireless fair queuing models, the ACFQ scheduler accounts for the service loss or gain due to both channel errors and absence. The flow with service gain will relinquish part of its service to another flow with service loss. We design an absence compensation model and an error compensation model separately, and smoothly integrate these two models together in ACFQ. We provide analytical properties of ACFQ, and evaluate its performance by simulations. Simulation results show that ACFQ can provide much better service differentiation and higher system throughput than WFQ, and outperforms strict priority queuing (SPQ) in terms of QoS provision.

1.3.3 The Relay-enabled IEEE 802.11 MAC Protocols

It is well known that IEEE 802.11 provides a physical layer multi-rate capability, which means that data can be transmitted at a number of rates according to the channel condition. New MAC protocols are required to exploit this capability. Previous work only considers how to increase the bandwidth utilization of the direct link between the sender and the receiver under good channel conditions.

Thus, it is still unclear how to further utilize the multi-rate capability when the direct link quality is poor.

We propose new MAC protocols to further exploit the multi-rate capability in IEEE 802.11 based wireless networks. When the direct link between the sender and the receiver can only support a low transmission rate, but there exists a relay node such that both the links from the sender to the relay node and from the relay node to the receiver can support high transmission rates, the impending packet should be delivered from the sender to the receiver by two-hop high speed transmission via the relay node. We supplement the relay mechanism to the standard point coordination function (PCF) and the distributed coordination function (DCF) respectively, and enhance them to deal with issues such as bandwidth utilization, dynamic channel condition, and variable transmission range. Experimental results show that our protocols can significantly reduce packet delay, improve the system throughput and reduce the impact of channel errors on fairness.

1.4 Contributions

The contributions of this work can be viewed from three aspects:

(a) We have designed and evaluated power-efficient scheduling algorithms to reduce the power consumption of the WNI. In particular, we have proposed service models, power-efficient scheduling algorithms and the corresponding communication protocols. We have addressed how to deal with the impacts of

state transition delay and dynamic channel condition on our service models, and how to support different types of applications in our models.

(b) Besides power efficiency, we have also designed and validated the absence compensation fair service model to improve service differentiation under bursty data traffic in wireless networks. We have proposed several novel techniques to quantify the service loss due to absence, to let each flow gracefully relinquish its service gain and to smoothly integrate the error compensation model and the absence compensation model. With carefully designed algorithms, our scheme significantly improves service differentiation and maintains QoS for each flow.

(c) In the context of MAC protocols, we have proposed novel relay-enabled MAC protocols to further exploit the multi-rate capability of IEEE 802.11 PHY layer. We have elegantly extended the standard IEEE 802.11 MAC protocols with MAC layer relay mechanisms, so that the sender, the relay node and the receiver can efficiently coordinate together and agree on which rate to use and whether to use relay. Several optimization techniques have been proposed to reduce the control overhead, improve the bandwidth utilization, and deal with the impacts of variable transmission range and dynamic channel condition.

1.5 Thesis outline

The rest of this thesis is organized as follows. In Chapter 2, we present the priority-based bulk scheduling scheme and the rate-based bulk scheduling. We show their

effectiveness through both analysis and simulations. We describe the absence compensation fair queuing service model in Chapter 3, and explain how to improve service differentiation under bursty data traffic. In Chapter 4, we describe the relay-enabled PCF protocol and the relay-enabled DCF protocol, and validate their performance gain through extensive simulations. Finally, Chapter 5 concludes the thesis and discusses our future work.

Chapter 2

Power-Aware and QoS-Aware Service Models

2.1 Background

To facilitate ubiquitous connectivity with small-size and low-cost mobile terminals (MTs), reducing power consumption of each MT becomes an important concern. Since the battery performance improvement is fairly slow [50] and it is doubtful that significant improvement can be expected in the foreseeable future, instead of trying to improve the amount of energy that can be packed into a power source, we can carefully design communication protocols so that the MTs can perform the same functions and provide the same services while minimizing their overall power consumption.

Understanding the power characteristics of the wireless network interface (WNI) used in MTs is important for designing power efficient communication protocols. A typical WNI may exist in *active* or *sleep* state. In the active state, the WNI may be in the transmit, receive and idle modes. Many studies [13, 39, 61, 69] show that the power consumed in the active state is similar, which is significantly higher than the power consumed in the sleep state. As a result, most of the work on power management concentrates on putting the WNI into sleep when it is idle. Most of previous works [61, 34, 36, 51, 60, 74] focus on reducing the power consumption. These solutions can be applied to most data applications which do not have QoS requirements, but may not be enough for streaming applications (e.g. audio/video-on-demand) due to lack of

QoS provisions. As a result, *how to achieve power saving without violating QoS* is a big challenge for supporting streaming applications on wireless networks.

2.2 Motivations

In order to provide guaranteed service over a shared link, several rate-based service disciplines have been proposed [71]. The principle of these service models is to provide each flow with a guaranteed data rate without being affected by other mis-behaving flows sharing the link. A scheduling algorithm can be classified as *work-conserving* or *non-work-conserving*. In the work-conserving scheduling, a server is never idle when there is a packet to send. In the non-work-conserving scheduling, a packet is not served until it is eligible [71], even though the server is idle at that time.

When applying the existing scheduling algorithms to wireless networks, power issues should be considered. As explained in Section 2.1, putting the WNI into sleep is the most widely used method to save power. When work-conserving service discipline is used, it is difficult to put the WNI into sleep. The reason is as follows. When an MT shares the link with other MTs, if a work-conserving guaranteed service model is applied, the service sequence of the link depends on the scheduling pattern of all flows. The scheduling pattern is related to the number of backlogged flows and the deadlines of the head-of-line packets of these flows. Unfortunately, the MT does not know the following service sequence due to lack of global information regarding the scheduling pattern of all flows in the system. Thus, the WNI has to stay in active since it does not know when the next packet will arrive. This may cause the WNI to waste a lot of power. For example, as shown in Figure 2.1, suppose three flows share a link of capacity C, and

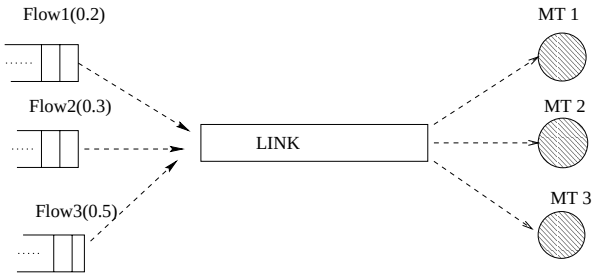


Fig. 2.1. An example of work-conserving link sharing

each flow has an MT as its receiver. Suppose all flows want to send B bits and their data rates are: $0.2C, 0.3C$ and $0.5C$ respectively. Under the work-conserving service model, if B is quite large, MT_1's active time is approximately $5B/C$, but MT_1's effective receive time is B/C. It is easy to see that almost 80% of the power has been wasted.

Non-work-conserving Virtual Clock (NVC): Non-work-conserving scheduling can be used to save power. The basic idea is to let the WNI enter sleep when it is not used. One simple approach is to let the BS and the MT mutually agree on a scheduling pattern. When the BS sends a packet to the MT, the scheduler piggybacks the information about the eligible time for the next packet to be transmitted. Note that the eligible time can be calculated based on the flow's data rate. The scheduler works in a *non-work-conserving* [71] manner since it will not serve the flow before the eligible time even though the channel is idle. Thus, the MT can enter sleep for a while until the eligible time of the next packet. This simple approach is referred to as the *Non-work-conserving Virtual Clock* (NVC) scheduling. Although NVC can save a large amount of power in theory, it may not be possible in practice. This is due to the *state transition delay* of each WNI. As reported in [59], the transition time from active to sleep and back to active is on the order

of tens of milliseconds. Suppose $T_{off \rightarrow on} = 10ms$, if the packet interval time, denoted by T_{intvl}, of the flow is less than $10ms$, we have $T_{on \rightarrow off} + T_{off \rightarrow on} > T_{intvl}$. Thus, the NVC scheme cannot put the WNI into sleep without violating the QoS requirement of the flow[1]. Even when $T_{on \rightarrow off} + T_{off \rightarrow on} < T_{intvl}$, not too much power can be saved unless $T_{on \rightarrow off} + T_{off \rightarrow on} \ll T_{intvl}$. From Figure 2.2, we can see that $\frac{T_{on \rightarrow off} + T_{off \rightarrow on}}{T_{intvl}}$ of

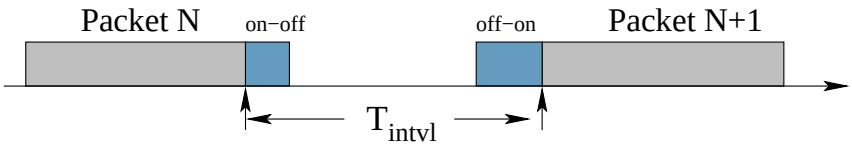


Fig. 2.2. The relationship between state transition and the packet interval time

the total packet interval time will be used for state transition. Since this transition also costs a lot of power [65], the NVC scheduling algorithm cannot save too much power unless $T_{on \rightarrow off} + T_{off \rightarrow on} \ll T_{intvl}$.

Bulk Scheduling (BKS): Based on the above reasoning, another approach, called *Bulk Scheduling (BKS)*, can be used to further reduce power. With this service policy, the channel is divided into bulk slots. A flow which needs to be served wakes up at the beginning of a bulk slot. The scheduler randomly selects a flow to serve at that time, and the selected flow will be served until the end of the bulk slot. Meanwhile, other losing flows enter sleep and wake up again to wait for the service at the beginning of the

[1] In this case, in order to provide QoS, the WNI has to stay in active and the scheduler serves the flow in a work-conserving manner.

next bulk slot. Figure 2.3 shows one example of bulk scheduling. In this example, three

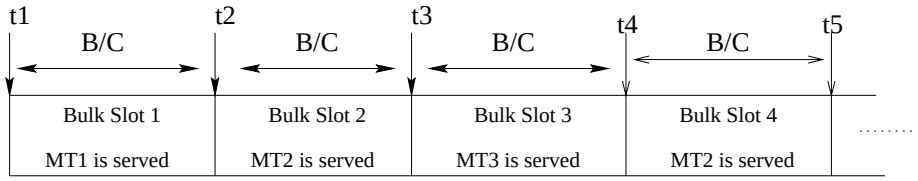


Fig. 2.3. An example of bulk scheduling

flows share the link and the receivers are MT_1, MT_2, and MT_3 respectively. Each bulk slot is equal to B/C, where B is the number of bits and C is the capacity of the link. For MT_3, it wakes up at t_1 and enters sleep since it found that the scheduler has selected MT_1 to serve. The same procedure happens at t_2. At t_3, it wakes up again and starts to receive data. Suppose MT_3 wants to transmit $k * B$ bits. If B is large enough so that the power used for state transitions between idle and active is negligible, the active time for MT_3 is only $k * B/C$, which allows MT_3 to stay in sleep for the maximum amount of time. Thus, if the bulk slot is significantly larger than $T_{on \to off} + T_{off \to on}$, the power consumption of state transition can be neglected, and bulk scheduling is an optimal service model in terms of power efficiency. However, bulk scheduling cannot provide QoS when multiple flows request data at the same time. For example, at t_1, suppose MT_3 and MT_1 will miss their deadline if waiting for another B/C time slot, scheduling MT_1 to serve will force MT_3 to miss its deadline.

In order to address the drawbacks of the NVC approach and the BKS approach, we propose new service models considering QoS provision and power efficiency. The basic idea of PBS is to let the MT buffer as much data as possible without affecting the QoS requirement of other flows. Relying on the buffered data, the MT can put its WNI into sleep and wake up only when the prefetched data is not enough to satisfy its QoS requirement. In this way, the WNI can power off for many time slots. Furthermore, as the MT enters sleep, other MTs can share the channel with fewer MTs (ideally no other MTs). Thus, MTs only spend a very small amount of time in the active mode to prefetch enough data, and then saves power.

2.3 The Priority-based Bulk Scheduling Service Model

In this section, we introduce a new scheduling scheme, called *priority-based bulk scheduling* (PBS) to utilize the client buffer to save power and provide QoS. Under PBS, the scheduler keeps track of the amount of prefetched (buffered) data for each flow. The flow with sufficient buffered data will be suspended until the buffered data runs out, whereas the flow with insufficient buffered data will be served based on its priority, which is determined by its delay requirement. With buffered data, the WNI can sleep long enough to offset the impact of the state transition delay. By suspending some flows, other active flows sharing the channel can obtain more bandwidth and take less time to fill the buffer. We also extend the service model to consider channel errors and applications that do not have strict delay requirements.

2.3.1 System Model

The geographical area is divided into cells in a wireless network. Inside each cell, the base station (BS) communicates with the mobile terminals (MTs) through uplink and downlink channels. Both uplink and downlink channels are divided into time slots. For uplink, we assume a channel can be either randomly accessed by MTs or granted through downlink (e.g., by reservation). Location-dependent errors may happen due to channel fading, interference, etc [42, 45, 47, 64]. Similar to many existing works [42, 43, 45, 47], we assume that the BS has a mechanism to predict the channel condition. For simplicity, we assume the modulation techniques and coding techniques are fixed, and power adaptive techniques [63] are used to deal with channel errors. Since a WNI spends significantly higher amount of power (from 10 to 100 times [13, 39, 61, 69]) during active compared to sleep, we use the accumulated WNI sleep time to measure the power efficiency of different schemes. In order to accurately measure the power consumption, we use the following notations.

- $T_{off \rightarrow on}$: time spent to transit from sleep to active.
- $T_{on \rightarrow off}$: time spent to transit from active to sleep.

As reported in [65], the power consumed during the state transition from active to sleep, or from sleep to active is similar to that in the active mode, which is much larger than the power consumed in the sleep mode.

2.3.2 The PBS in Detail

The PBS service model has two parts: a scheduler at the BS side and a proxy at the MT side. The scheduler is used to control the channel access among multiple MTs. The proxy is used to coordinate with the scheduler at the MT side. We describe the algorithm used by the scheduler, and show how the proxy works. Then, we present solutions to deal with channel errors, calculate the computation complexity and prove some properties of the service model.

2.3.2.1 The PBS Scheduler

In PBS, each packet of a flow is assigned a deadline. The scheduler orders the transmission of packets according to their deadlines. If a flow's aggregated service goes beyond the minimum service required to maintain the QoS[2], it will be removed from the scheduling region until it needs more data to maintain the QoS. As a result, the flow is suspended periodically and the data are transmitted in the form of several runs of packets.

Service accounting: Without loss of generality, for each flow, we assume that the first served packet after entering the scheduling region is indexed by 1. Each packet is assigned a deadline according to the packet length and the flow's data rate. Formally,

[2]At this time, the MT of the flow has enough prefetched data. For simplicity, we say the flow has got enough *ahead-service*.

the deadlines (in seconds) are computed as follows:

$$\begin{aligned} d_i^1 &= e_i \\ d_i^j &= d_i^{j-1} + \frac{l_i^{j-1}}{r_i} \end{aligned} \tag{2.1}$$

where p_i^j is the j^{th} packet of flow f_i, e_i is the eligible time of f_i, d_i^j is the deadline of packet p_i^j, l_i^j is the length of p_i^j and r_i is the data rate of f_i (in bps). Suppose, p_i^n is the head-of-line packet of f_i at time t, the ahead-service (in seconds) of f_i, denoted by $ahead_i$, is computed as follows:

$$ahead_i(t) = max(d_i^n - t, 0) + I(i) * \frac{l_i^n}{r_i} \tag{2.2}$$

where I(i) returns 1 if f_i is being served, otherwise, it returns 0. Since $ahead_i$ is used to trace the amount of prefetched data at the MT side, it cannot be negative at any time.

Flow state management: At the BS side, each flow has two scheduling states: *idle* and *active*. The transitions between scheduling states of f_i (denoted by $state_i$) are controlled by the scheduler. For the purpose of flow control, there is an upper limit of ahead-service for each flow f_i, denoted by $Maxserv_i$. When $ahead_i > Maxserv_i$, the scheduler stops serving it and changes $state_i$ to idle. Suppose the scheduler provides enough ahead-service to f_i; i.e., $ahead_i \geq \phi$, ϕ is a system parameter to represent the lower bound for ahead-service. If there exists another flow, say f_j, which have not got enough ahead-service (i.e.,$ahead_j < \phi$), the scheduler lets f_i yield the channel to other

flows by changing $state_i$ to be idle. Whenever $state_i$ is changed to idle, the eligible time of f_i is set to be the current time plus $ahead_i$. At the same time, the scheduler updates the value of the eligible time to indicate when f_i will be moved back in scheduling region again. Whenever f_i is changed to idle, the scheduler will not serve it until the eligible time expires. After the eligible time expires, its state will be changed to active again (not shown in Figure 2.4).

When $state_i$ is set to idle, the scheduler should notify MT_i which can shutdown its WNI. We assume that the BS can mark the packet transmitted to the MT that will power off its WNI after receiving the marked packet. Since the MT's address is equal to the destination address of the packet, the BS can simply use one bit in the packet header to represent whether the packet is marked or not. When the MT receives the marked packet, it replies an ACK and puts its WNI into sleep. When the BS receives the ACK, it knows that the WNI has been powered off, and suspends the related flow.

The scheduler: The PBS scheduler works as follows. When n ($n \geq 1$) flows are active, the scheduler selects one flow as the *primary flow*, and the other $n-1$ flows are *secondary flows*. At any time, the scheduler exclusively serves the primary flow provided that the deadlines of the secondary flows will not be violated. If the deadline of the secondary will be violated, the scheduler has to serve the secondary flow in order to meet the QoS requirement of the flow. In other words, the scheduler serves the primary flow in a work-conserving manner, whereas each secondary flow is served in a non-work-conserving way. As $\frac{(\phi - ahead_i)r_i}{C - \sum_{j \in \mathcal{A}} r_j}$ (See Property 2 in Section 2.3.3) is an approximation of how fast a flow f_i can have enough ahead-service as the primary flow, the scheduler always selects the

Notations:
$\mathcal{A}$**:** the set of flows in active state
D_i**:** the deadline of the head-of-line packet of f_i
t: the current time

schedule()

```
1  begin:
2  if (A == NULL)
3      { idle in the time slot; goto begin; }
4  if (no primary flow)
5    select the primary flow f_i according to Eq. (2.3)
6  if ( t ≥ arg min{D_j} ∧ j ≠ i )
                j∈A
7      i = j; /* the deadline of the secondary flow f_j
       will be violated, so serve f_j */
8  p=f_i.deque(); /* get the packet to be transmitted */
9  if (ahead_i > Maxserv_i ∨ (ahead_i ≥ φ∧
       ∃j(f_j ∈ A ∧ ahead_j < φ))
10      mark(p);
11   send(p);
12   if (the transmission is successful)
13      { D_i = D_i + p.length/r_i;
14        if (p is marked)
15            {e_i = t + ahead_i; state_i = idle; } }
16   goto begin
```

Fig. 2.4. The PBS Algorithm

flow that can take the shortest time to get enough ahead-service as the primary flow. Formally, at time t, the primary flow (f_{prim}) is selected as follows:

$$f_{prim} = \arg\min_{i \in \mathcal{A}} \{ \frac{(\phi - ahead_i(t))r_i}{C - \sum_{j \in \mathcal{A}} r_j} \} \tag{2.3}$$

where $\mathcal{A}$ is the set of flows in active state and C is the channel capacity. The principle behind Eq (2.3) is similar to the shortest job first policy, which can minimize the average waiting time. Thus, the average time for each flow to get enough ahead-service is also minimized under PBS.

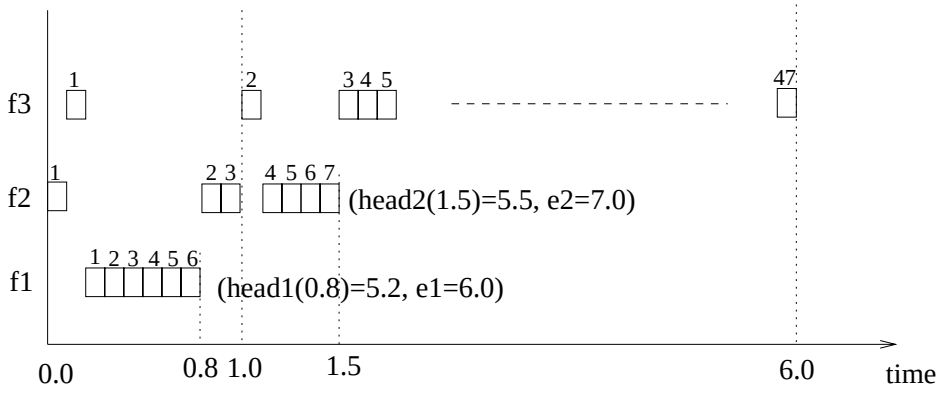


Fig. 2.5. An illustration of the PBS scheme

Figure 2.5 shows how the PBS scheme works. There are three backlogged flows (f_1, f_2, f_3) in the system. Each flow has 1Kbps data rate and unlimited $Maxserv$. Suppose $\phi = 5.0$ seconds, $C = 10$ Kbps, and all packets have the same packet length of 1 Kb. At time 0.0, the eligible time of all flows $(f_1 ... f_3)$ expires, and the ahead-service

of each flow is 0. Suppose f_1 is selected as the primary flow at time 0.0. To meet the deadlines of f_2 and f_3, the scheduler serves p_2^1 and p_3^1 first. From time 0.2 to 0.8, without violating the deadlines of f_2 and f_3, which are equal to 1.0, $p_1^1 ... p_1^6$ are served back-to-back. At time 0.8, according to Eq (2.2), $ahead_1$ is $d_1^6 - 0.8 = 5.2$, which is greater than ϕ. Thus, f_1 is suspended and its eligible time is set to 6.0. At time 0.8, f_2 becomes the primary flow. Follow the same procedure, f_2 is suspended from time 1.5 to 7.0. After time 1.5, f_3 is the only active flow, and the scheduler serves f_3 until time 6.0 when the eligible time of f_1 expires.

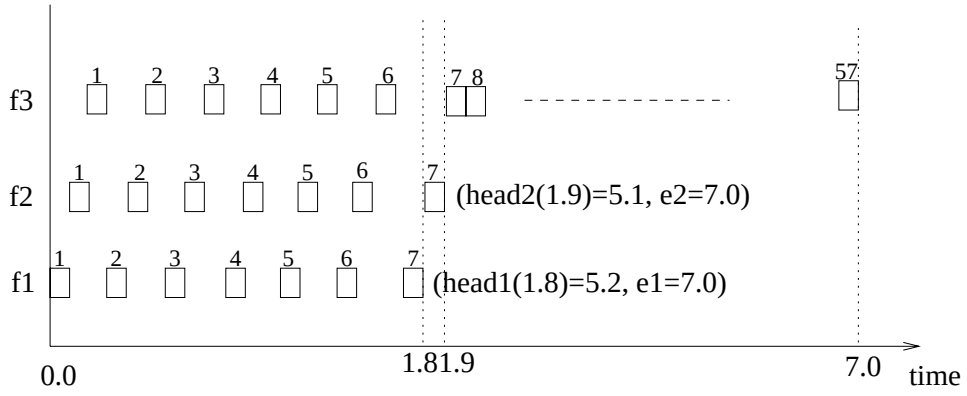


Fig. 2.6. An illustration of the WFQ scheme

To demonstrate the power efficiency of PBS, we compare the time period of each flow to get enough ahead-service under WFQ [49] and PBS. As shown in Figure 2.6, under WFQ, f_1 needs 1.8 seconds to get enough ahead-service, and f_2 needs 1.9 seconds. Comparing to the correspondent time periods in Figure 2.5 (e.g., f_1 needs 0.8 second),

we can see that, on average, the time period of each flow to get enough ahead-service under PBS is much smaller than that under WFQ.

Application-aware extensions: Compared with streaming applications, other applications (e.g. FTP, WWW) may not have stringent delay requirements. Suppose a non-streaming flow, say f_k, requires data and the channel utilization is high. If f_k is admitted into the scheduling region, from Eq (2.3), we can see that the time spent by the primary flow to get enough ahead-service will be increased, and the secondary flows have to wait longer before being the primary flow. Thus, all active flows spend more time in the active state, and the power consumption is also increased. Since f_k does not have strict delay requirement, it is better to postpone the serving time of f_k for a specified time period, denoted by $yield_k$. We assign an integer *relax factor* (σ_i) to each flow f_i, which is bounded by σ_i^{max}. For the flow with strict delay requirement, the upper bound is simply set to 0. For f_i with $\sigma_i > 0$, when its deadline expires, the scheduler decides whether to serve it or not according to the current system utilization, which can be measured by the number of active flows. If the current system utilization is greater than a threshold, denoted by μ_{thresh}, the scheduler lets f_i yield the channel for a period of $yield_i$. Otherwise, f_i is served. To avoid starvation, an adaptive scheme is used to manage σ_j as follows.

- When the scheduler decides to let f_j yield the channel, σ_j is decreased by one;
- When f_j leaves the channel with the ahead-service greater than $\phi + yield_j$, σ_j is increased by one. The service loss due to yielding is compensated by decreasing the ahead-service by $yield_i$.

By using the adaptive scheme, the maximum delay for f_i to be admitted into the scheduling region is $\sigma_i * yield_i$.

2.3.2.2 The PBS Proxy

A proxy is associated with the receiver of each flow. The proxy downloads data from the BS, monitors the amount of prefetched data, and manages the operation modes of the WNI. The proxy coordinates with the server and decides whether the WNI should enter sleep. Similar to the scheduler, the proxy can calculate the ahead-service of each flow according to the flow's data rate, packet length and the arrival time of each packet. If the proxy finds that the packet is marked, the WNI will be shutdown for a time period equal to the calculated ahead-service. In this way, the control overhead can be reduced since the scheduler does not need to tell the length of the sleep period. If multiple applications are supported, the WNI is shut down only when *all proxies* running on the MT have requested to do so. However, the WNI wakes up if *any proxy* needs it at any time. Since the MT knows all proxies running on it, this can be easily implemented.

2.3.2.3 Dealing with Channel Errors

The wireless communication channel is error prone, and the error is location-dependent and bursty. If channel errors exist, the probability of a successful transmission becomes very low, and hence, bandwidth and power may be wasted during re-transmissions. Similar to [8, 45, 47], we deal with channel errors by swapping time slots from flows suffering channel errors to flows which have good channel conditions. However, we focus on minimizing the influence of channel errors on QoS and power

consumption of each flow under the PBS service model. When a flow, say f_i, has bad channel condition, if $ahead_i \geq \phi$, the BS stops serving f_i and notices f_i with a marked null packet. After receiving the null packet, the proxy of f_i realizes the bad channel condition and shuts down the WNI. If $ahead_i < \phi$, it may not worth to power off the WNI since the actual sleep time could be too short. Thus, the scheduler stops serving f_i for a pre-specified short period, which is called *backoff period*, and the proxy of f_i lets the WNI stay in active. After the backoff period, the scheduler will resume serving f_i. If f_i still suffers from channel errors, the same backoff procedure will be applied to f_i again.

Since most channel errors are bursty, after leaving the channel for some time, the flow may get good channel state when being served. After a flow leaves the channel, the total number of flows sharing the channel decreases and the allocated data rate of each remaining flow increases. As a result, these active MTs spend less time in active, and then reduce the power consumption.

2.3.2.4 Computation complexity of PBS

It only takes one division and one addition to get the ahead-service (in seconds) at the MT side. At the BS side, the operation consists of the selection of the primary flow at the cost of $O(log(n))$, and the updates for ahead-service per-flow at the cost of $O(n)$. Thus, PBS is computationally feasible in many wireless networks that have a moderate number of flows per BS. MTs can easily get the data rate of the flow, ϕ, Maxserv_i, the backoff period and $yield_i$ (for non-streaming applications) during session initialization. Since the performance (in terms of QoS and power efficiency) of PBS is not sensitive to

the selection of ϕ even when the system is heavily loaded (see Section 2.3.4.4), it does not require precise information about the length of the state transition delay. The only requirement is that ϕ is much larger than the delay.

2.3.3 Analysis of PBS

In this section, we present some important properties of the PBS scheme. We prove that PBS can achieve power efficiency and QoS provision for all flows in the system. For simplicity, we assume that the channel, with a channel capacity of C, is error-free, and each flow has the same data rate r. The proofs are shown in the Appendix.

PROPERTY 1. *(Delay Guarantee) If Q is the set of backlogged flows in the system, and $|Q|r < C$, the PBS scheduler will serve each packet p_i^j according to:*

$$s_i^j - d_i^j \leq (|Q| - 1)\frac{L^{max}}{C} \tag{2.4}$$

where s_i^j is the time when the server starts serving p_i^j and L^{max} is the maximum packet length.

Proof: The set Q can be partitioned into two subsets: the set of active flows, denoted by $\mathcal{A}$, and the set of idle flows denoted by $\mathcal{I}$. At any time t, flow f_i has two cases:

- **case 1:** $f_i \in \mathcal{I}$. Suppose p_i^j is the last packet served before f_i was suspended. Since $d_i^j > t$ (otherwise $f_i \in \mathcal{A}$) and $s_i^j \leq t$ (since f_i was suspended before t), the property holds.

- **case 2:** $f_i \in \mathcal{A}$. Suppose the head-of-line packet of f_i is p_i^j at time t and $d_i^j = t$. Since the definition of the deadline of each packet is the same as the *start-time*

first fair queuing (SFQ), and f_i has the highest priority under PBS provided that $d_i^j \leq t$, with similar arguments to Theorem 4 of [25] and $\mathcal{A} \subseteq Q$, we get:

$$\begin{aligned} s_i^j - d_i^j &\leq \sum_{k \in \mathcal{A} \wedge k \neq i} \frac{L^{max}}{C} \\ &\leq \frac{(|Q|-1)L^{max}}{C} \end{aligned}$$

□

PROPERTY 2. *(Power Efficiency) Suppose Q is the set of backlogged flows in the system and $|Q|r < C$. Consider the process that each flow prefetches the ahead-service greater than or equal to ϕ and leaves the channel. If $\phi r \gg L_{max}$, the average active time under PBS and under weighted fair queuing (WFQ), denoted by $\bar{T}_{act,PBS}$ and $\bar{T}_{act,WFQ}$ respectively, follows:*

$$\frac{\bar{T}_{act,WFQ}}{\bar{T}_{act,PBS}} > \frac{\frac{\phi r - L^{max}}{C/|Q| - r}}{\sum_{i=1}^{|Q|} \frac{(|Q|-i+1)(\phi r + (|Q|-i+1)L^{max})}{|Q|(C-(|Q|-i+1)r)}} \tag{2.5}$$

Proof: If each flow is served under *generalized processor sharing* (GPS) [49], the actual data rate of each flow is equal to $\frac{C}{|Q|}$. Then, flow f_i's aggregated service (in bits) during a time period of Δt is equal to $\frac{C}{|Q|}\Delta t$. Now, let's consider the situation that the scheduler applies the WFQ scheme and each flow is served with the granularity of packet.

Since all the flows are continuously backlogged and have the same data rate, the WFQ scheduler behaves similar to the worst-case fair weighted fair queuing (WF^2Q) [7]

scheduler. Following Theorem 1 of [7], the relationship between the amount of f_i's aggregated service under GPS and WFQ during the interval Δt, denoted by $W_{i,WFQ}(\Delta t)$ and $W_{i,GPS}(\Delta t)$ respectively, follows $W_{i,WFQ}(\Delta t) - W_{i,GPS}(\Delta t) < L^{max}$. Hence, we have

$$W_{i,WFQ}(\Delta t) < \frac{C}{|Q|}\Delta t + L^{max} \tag{2.6}$$

Without loss of generality, suppose flow f_1 is the first flow that leaves the channel, and f_1 spends T_1 to get the ahead-service equal or greater than ϕ. T_1 needs to satisfy $W_{i,WFQ}(T_1) - rT_1 \geq \phi r$. With Ineq (2.6), we get

$$T_1 > \frac{\phi r - L^{max}}{C/|Q| - r} \tag{2.7}$$

Since f_1 is the first flow that gets enough ahead-service, according to Ineq (2.7), we have:

$$\bar{T}_{act,WFQ} > \frac{\phi r - L^{max}}{C/|Q| - r} \tag{2.8}$$

Under PBS, at any time, set Q can be partitioned into set $\mathcal{A}$ and $\mathcal{I}$. The flows in $\mathcal{I}$ have got enough ahead-service and left the channel, while flows in $\mathcal{A}$ are served. Since the secondary flows in $\mathcal{A}$ are served in non-work-conserving manner, similar to the proof of Lemma 1 in [25], the aggregated service of secondary flow f_j during any time interval Δt, denoted by $W^s_{j,PBS}(\Delta t)$, follows:

$$r\Delta t - L^{max} \leq W^s_{j,WFQ}(\Delta t) \leq r\Delta t + L^{max} \tag{2.9}$$

Suppose the primary flow f_i takes $\Delta T_{i,PBS}$ to get ahead-service greater than or equal to ϕ. Similar to the proof of Theorem 2 in [25], the aggregated service of f_i during $\Delta T_{i,PBS}$, denoted by $W^p_{i,PBS}(\Delta T_{i,PBS})$, follows:

$$\begin{aligned} & W^p_{i,PBS}(\Delta T_{i,PBS}) \\ & \qquad \geq \quad \Delta T_{i,PBS}(C - |\mathcal{A}|r) - \textstyle\sum_{j \in \mathcal{A} \wedge j \neq i} L^{max} \\ & \qquad \geq \quad \Delta T_{i,PBS}(C - |\mathcal{A}|r) - (|\mathcal{A}| - 1)L^{max} \end{aligned} \tag{2.10}$$

Since f_i leaves the channel whenever it gets the ahead-service greater than or equal to ϕ, $\Delta T_{i,PBS}$ satisfies: $W^p_{i,PBS}(\Delta T_{i,PBS}) - \phi r \leq \phi r + L^{max}$. Combined with Ineq (2.10), we get

$$\Delta T_{i,PBS} \leq \frac{\phi r + |\mathcal{A}|L^{max}}{C - |\mathcal{A}|r} \tag{2.11}$$

Without loss of generality, suppose the order of primary flows during the whole process is: $f_1, f_2 ... f_{|Q|}$. Due to $\phi r \gg L^{max}$ and Ineq (2.9), each flow cannot have enough ahead-service when it is a secondary flow. As a result, the time spent by f_i to leave the system, denoted by $T_{i,PBS}$, follows:

$$T_{i,PBS} = \sum_{k=1}^{i} \Delta T_{k,PBS} \tag{2.12}$$

During the time period when f_i is the primary flow, $|\mathcal{A}| = |Q| - i + 1$. Hence, with Ineq (2.11) and Eq (2.12), we get:

$$\bar{T}_{act,PBS} \leq \sum_{i=1}^{|Q|} \frac{(|Q| - i + 1)(\phi r + (|Q| - i + 1)L^{max})}{|Q|(C - (|Q| - i + 1)r)} \tag{2.13}$$

With Ineq (2.8) and Ineq (2.13), it is easy to find out that the property holds. □

Different fair queuing model may have different fairness property [71]. However, since we assume that each flow is continuously backlogged and has the same date rate, with the results of [68], Ineq (2.6) still holds for other rate-based fair queuing models. Thus, Property 2 is also valid for other rate-based fair queuing models. With Property 2, we can calculate the *lower bound* ratio of the average active time under WFQ to that under PBS. We give an example to show some numerical results of the ratio as a function of $|Q|$. Suppose $C = 400Kbps$, $r = 50Kbps$, $\phi = 500ms$, and $L^{max} = 1000bits$. As shown in Figure 2.7, when $|Q|$ increases from 2 to 6, the ratio lower bound increases from 1.26 to 2.14. This shows that PBS is more power efficient than WFQ.

2.3.4 Performance Evaluations

2.3.4.1 The Experimental Setup

We evaluate the performance of the PBS service model through case studies consisting of *Audio-on-Demand (AoD)* and WWW services. The AoD streaming service is evaluated through trace-driven simulation. We downloaded a demo MP3 audio streams from [48] as the trace source. Since the MP3 streams are based on variable bit rate (VBR), we extracted information of every frame, i.e. the frame size, the frame bit rate,

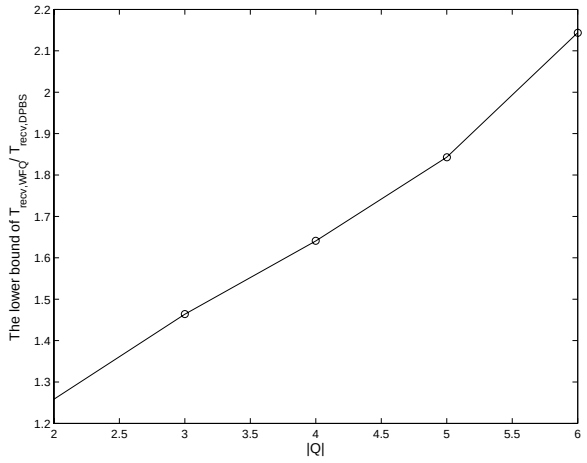


Fig. 2.7. The numerical results of the lower bound of $\frac{\bar{T}_{act,WFQ}}{T_{act,PBS}}$

and the frame sample rate, to get the bit rate of each packet. There is an interval between successive songs, which is distributed randomly between $[0.0, 2.0]$. The WWW service is emulated by a simple ON/OFF traffic model that mimics the web user access behavior. An ON period will start when a new Web page is requested. Once the run of data has finished, there is an OFF period during which the user studies the information just downloaded. We assume the total page size of an ON period is exponentially distributed with the mean of 12 KB, and the length of an OFF period is exponentially distributed with the mean of 2.0 seconds.

The capacity of the wireless channel is assumed to be 384 Kbps, and is based on TDMA. Each time slot is $2.5ms$, which can be used to transmit 112 Bytes of data (not including the header). Each MT represents a user having an AoD or WWW flow. Each flow is allocated a data rate of 56 Kbps. The simulation time is 100 seconds. For the WNI, we assume $T_{on \to off} = 5ms$ and $T_{off \to on} = 10ms$. The PBS server sets

$\phi = 80ms$ and $\mu_{thresh} = 50\%$. In addition, $yield_i$ and backoff period of flow f_i are set to be 100 ms and 15 ms respectively. The relax factor is set to be 2 for WWW flows, and the $Maxserv$ is set to be $2000ms$ for AoD flows.

Channel errors are modeled by a two state Markov chain, which has two states: good and bad. The probability from a good state to a bad state is 0.05, and from a bad state to a good state is 0.08. When the channel is in good state, packets are transmitted correctly. When the channel is in bad state, packet transmissions will fail.

We evaluate the performance of the proposed service model based on the following factors: the amount of prefetched data, the QoS and the time spent in active, sleep and state transition. We use the total *noticeable interrupt time* (NIT) to measure the quality of the playback audio. Since tiny interrupts are not noticeable by human being, only continuous interrupts which are greater than $20ms$ are counted as NITs. This is less than the normal delay requirement for telephone voice service since we are dealing with music here. In the simulations, the total NIT is equal to the aggregated noticeable interrupt time intervals. Since WWW service does not have strict delay requirement, its QoS is measured by the *throughput*, which is equal to the total amount of data (in bytes) transmitted. To evaluate the time spent in each operation mode, t_{on}, t_{off}, t_{switch} are used to denote the total time spent in active, sleep and state transition respectively.

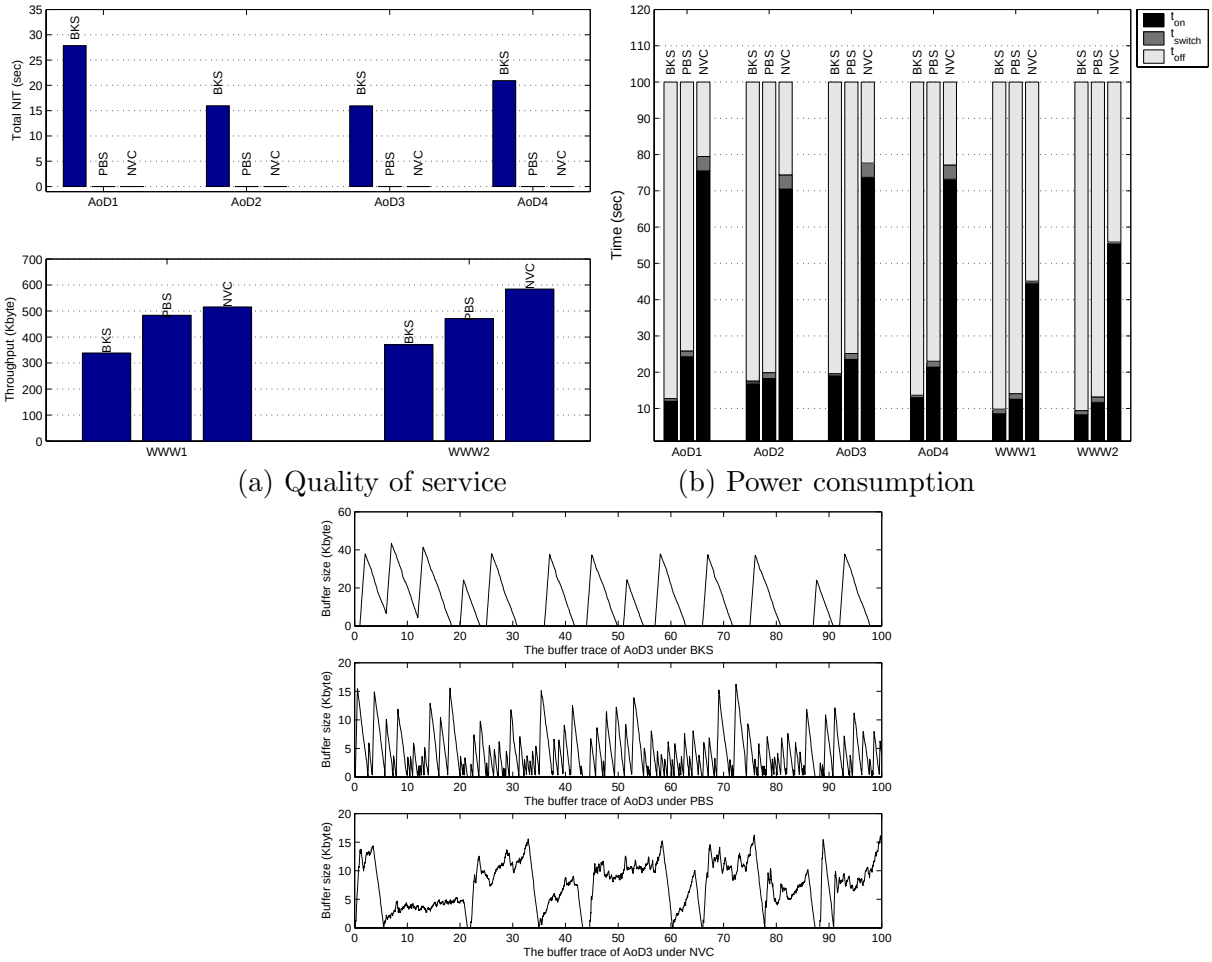
We compare the performance of the PBS scheduling with the bulk scheduling (BKS) and the Non-work-conserving Virtual Clock (NVC) scheduling scheme. We choose the bulk slot time to be 1.0 second for BKS. There is a buffer on the MT side in the BKS approach and the NVC approach. In BKS, the MT stays in sleep after being served until the prefetched data run out. In NVC, the MT with an AoD flow goes to sleep when the

buffer is full and wakes up when the buffer is near empty. The evaluation considers two scenarios. In Scenario 1, the channel is error free. In Scenario 2, an error-prone channel is considered. Finally, we evaluate the impacts of ϕ.

2.3.4.2 Scenario 1: Error-free Channel

In this scenario, we show the fundamental difference among PBS, BKS, and NVC. The evaluation includes four AoD flows and two WWW flows. From Figure 2.8 (a), we can see that the playback quality of the PBS approach and the NVC approach are perfect since their total NITs are 0. Compared with PBS and NVC, the BKS approach provides poor QoS since the total NIT of each AoD flow is greater than 15 seconds, and the throughput of each WWW flow is much less than that under PBS and NVC. Since there are six flows sharing the channel, it is highly possible that more than one flow request data at the beginning of a bulk slot. The BKS scheduler only randomly selects one winning flow to serve. As a result, other losing flows cannot be served during the bulk slot, and their delay requirements are violated.

As shown in Figure 2.8 (a), flows WWW1 and WWW2 have higher throughput in NVC than PBS, since PBS lets WWW1 and WWW2 yield the channel when the system's utilization is greater than 50%. On the other hand, as shown in Figure 2.8 (b), the power consumption of each WNI under NVC is significantly higher than that under PBS and BKS. Since the data rate is quite high (i.e., 56 Kbps on average), many of the inter-packet arrival periods are less than $T_{on \rightarrow off} + T_{off \rightarrow on}$. As a result, the WNI cannot always go into sleep in the NVC approach. Even though the WNI can enter sleep, the actual sleep period is reduced by $T_{on \rightarrow off} + T_{off \rightarrow on}$.



(a) Quality of service (b) Power consumption

(c) Buffer trace of AoD3

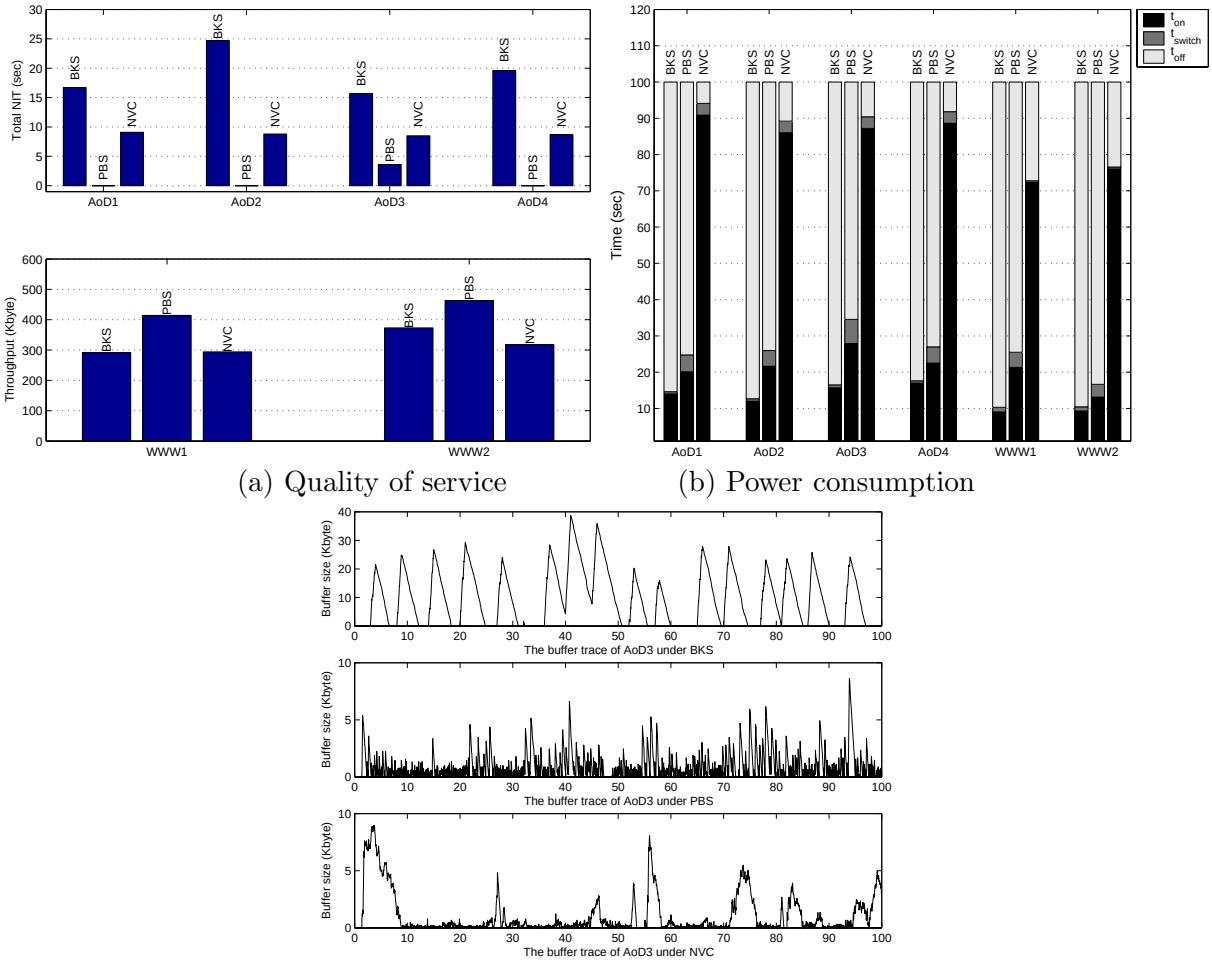
Fig. 2.8. Performance comparisons among BKS, PBS, and NVC without channel errors

Figure 2.8 (c) shows how these three approaches work through the buffer trace of flow AoD3. When the buffer size goes up, the WNI receives data in the active mode. During the time period when the buffer size goes down until the buffer size is near zero, the WNI stays in sleep. The slope of the buffer increment indicates how fast the flow fills the buffer. For BKS, since the flow is always served alone during a bulk slot, its actual data rate is equal to the channel capacity, which is the ideal power efficiency. From Figure 2.8 (c), we can see that the slope of the buffer increment under PBS is almost equal to that under BKS, which shows that the power efficiency under PBS is good. Under PBS, at any time, the scheduler serves the primary flow alone provided that the deadlines of the secondary flows will not be violated. As a result, the actual data rate of the primary flow may be quite high when there are few secondary flows. On the other hand, by suspending the primary flow that gets enough ahead-service, the new selected primary flow competes the channel with less flows and gets higher data rate.

Compared with PBS and BKS, the slope under NVC is much less than that under BKS during some time period; e.g., during the time period of $(53.0, 57.0)$. Since the data rate of each flow is quite high, the receiver's WNI has very few chances to go to sleep under NVC. At this time, the scheduler serves the flows in a work-conserving manner. Since a large number of flows share the channel, the actual data rate of each flow is small, which reduces the power efficiency.

2.3.4.3 Scenario 2: Error-Prone Channel

In this subsection, we evaluate the impact of channel errors on three scheduling approaches. We introduce channel errors in this scenario and assume that the channel



(a) Quality of service (b) Power consumption

(c) Buffer trace of AoD3

Fig. 2.9. Performance comparisons among BKS, PBS, and NVC with channel errors

errors are bursty and location-dependent. We assume only AoD3 and WWW1 experience channel errors, while other users still have error-free channels.

As shown in Figure 2.9 (a), when channel errors exist, the playback quality of the BKS approach and the NVC approach is much worse than the PBS approach. The poor playback quality of the BKS approach has been explained in Scenario 1, and the reason is still valid for this example. Since the NVC approach does not consider how to deal with bursty and location-dependent channel errors, flows that have good channel conditions are affected by the two flows with channel errors. As a result, the playback quality of each flow is severely degraded. In contrast, during each time slot, the PBS scheduler lets the flow suffering from channel errors yield the channel to other flows with clean channel, so that flows with clean channel can still have good QoS.

As for AoD3, which suffers from channel errors, its playback quality is not as good as AoD1 and AoD2 which don't have channel errors, but it is still much better than that under BKS and NVC. In the PBS approach, when AoD3 has good channel, it prefetches data. With the prefetched data, when the channel condition is bad, AoD3 can leave the channel for a while. Since channel errors are bursty, this mechanism can offset the impact of channel errors. However, if channel errors last for a long time, the QoS may be violated; e.g., total NIT of AoD3 becomes 3.6 seconds.

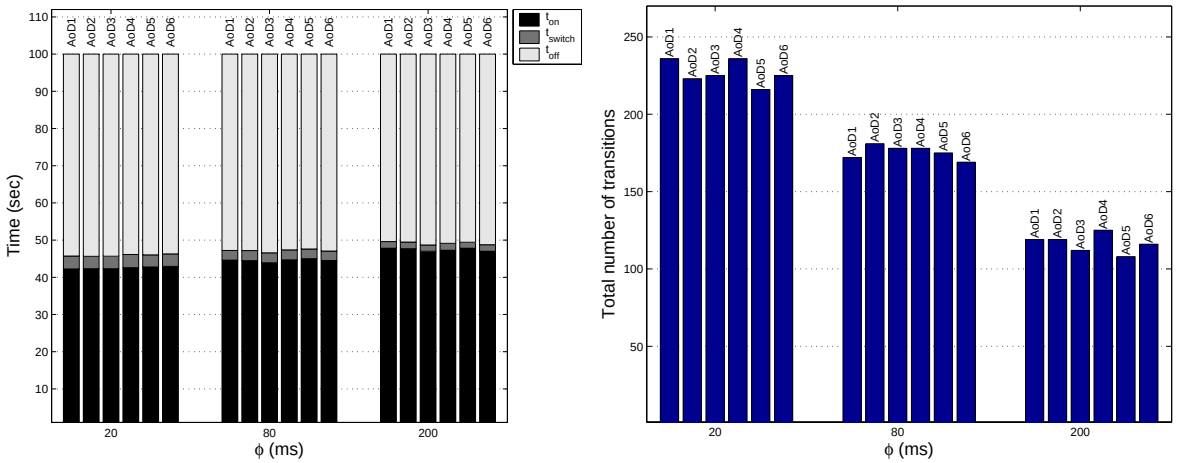
As shown in Figure 2.9 (b), t_{on} and t_{switch} of each flow under PBS is greater than that in scenario 1, since the flows with channel errors are more likely to stay in the scheduling region due to lack of enough ahead-service. According to the state management scheme of PBS, other flows are suspended when their ahead-service is equal to or a little bit more than ϕ. Since ϕ is $80ms$, the receiver's WNI of the flows cannot

sleep for a long time, and the number of state transitions increases. This also explains why t_{switch} increases for flows with channel errors, and can be verified by the buffer trace of AoD3 in Figure 2.16 (c). As we can see, under PBS, the average amount of data in the buffer is less than that in Figure 2.15 (c).

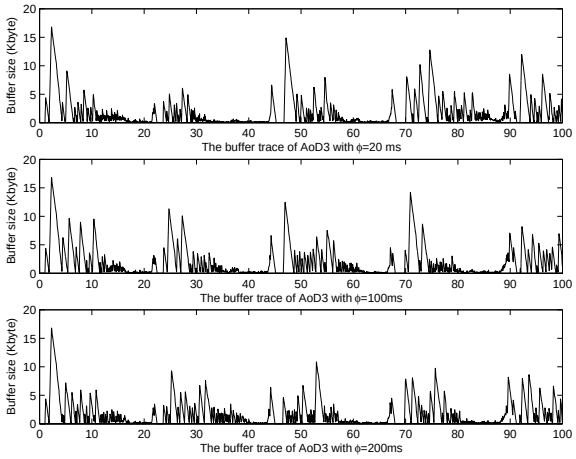
2.3.4.4 The Impacts of ϕ

As ϕ increases, the number of transitions between sleep and active drops. However, if ϕ is too large, the time spent in getting enough ahead-service will be long. When the workload of the system is not heavy, in most cases, the number of flows with insufficient ahead-service is small. As a result, the actual data rate of the flows is high and then the time spent in getting enough ahead-service is very short. Thus, the PBS is not sensitive to ϕ when the workload is small. Since the average rate of each AoD flow is 56Kbps, the maximum number of AoD flows can reach six if the average flow rate is used for admission control. In this scenario, we increase the workload to six AoD flows, and we assume that the channel is error free. We evaluate the performance when ϕ is $20ms$, $80ms$ and $200ms$ respectively. All the NITs of the flows are 0, and we do not show them due to the limited space.

The simulation results are shown in Figure 2.10. When ϕ increases, the time to get enough ahead-service increases and t_{on} becomes longer. As ϕ becomes smaller, the number of state transitions increases, and t_{switch} becomes larger. However, the sum of the active time and the state transition time under different values of ϕ does not have too much difference. If the power consumed during state transition is similar to that in



(a) Power consumption (b) Total number of transitions



(c) Buffer trace of AoD3

Fig. 2.10. The impacts of ϕ on PBS

the active state, the performance of PBS is not sensitive to ϕ. However, if the power consumption of the state transition is much higher, ϕ cannot be too small.

Since the total transition time is very close for different ϕ in Figure (2.10) (a), we use Figure (2.10) (b) to show the number of transitions for different ϕ. As can be seen, the number of transitions when $\phi = 200ms$ is about half of that when $\phi = 20ms$. When ϕ decreases, for a given number of flows, the time spent in getting ahead-service increases as ϕ decreases. Thus, each flow takes a small amount of time to receive data. However, a small ϕ prevents the WNI from staying sleep for a long time period, which makes the WNI switch from on to off, and off to on more frequently. As a result, t_{switch} increases. Due to similar reason, t_{switch} decreases as ϕ increases. From Figure 2.10 (c), we can also see that the behavior of the WNI is different with different ϕ.

2.4 The Rate-based Bulk Scheduling Service Model

Since the PBS service model is designed for systems that do not use rate adaptation techniques to deal with channel errors, we design another service model, which is based on the *rate-based bulk scheduling* (RBS) algorithm, to support power efficient streaming service in wireless systems with rate adaptation capability. The RBS service model applies algorithms focusing on how to deal with the impact of variable transmission rate and how to achieve good error resiliency in lossy wireless environments.

2.4.1 System Model

The network architecture is almost the same as we described in Section 2.3.1. The difference is that we assume the system applies rate-adaptive technique which selects

the most suitable modulation and coding schemes according to the channel condition. Based on the channel condition, like some existing systems (e.g. EDGE [64] and HDR [6]), the system selects the most suitable modulation and coding scheme to transmit the impending packet. For each time slot, there is a set of possible transmission rates $\{0, C^1, C^2, ..., C^M\}$.

2.4.2 The RBS in Detail

The basic idea of RBS is to let the MT buffer as much data as possible without affecting the QoS requirement of other flows. Relying on the buffered data, the MT can put its WNI into sleep and wake up only when the prefetched data is not enough to satisfy its QoS requirement. To provide QoS to each flow, with the ease in implementation, the *start-time first fair queuing* (SFQ) [25] is used to give each flow a fair link sharing. Meanwhile, the scheduler records the ahead service of each flow. When a flow has enough ahead-service with regard to its QoS requirements, the scheduler suspends serving the flow so that the related WNI can sleep and wake up only when the buffered ahead-service is not enough to satisfy the QoS requirements.

The RBS service model has two parts: a scheduler at the BS side and a proxy at the MT side. The scheduler is used to control the channel access among multiple flows. The proxy is used to coordinate with the scheduler and manage the operating state of the WNI based on the amount of buffered data. Next, we only describe the details of the scheduler and the proxy of RBS.

2.4.2.1 Balancing Power Efficiency and Fairness

For wireless channels with multi-rate capability, the amount of bits transmitted per time slot depends on the respective modulation and coding scheme [64]. As a result, there exists a tradeoff between system throughput and fairness among flows. Several schemes have been proposed to opportunistically exploit the system throughput with the constraint of long-term temporal or throughput fairness [43, 42]. Due to the stringent delay requirement of streaming applications, long-term fairness is not enough for QoS provision, since it is possible for a flow with channel errors to be starved for a long time (say, a few seconds) [42]. However, simply providing short-term throughput fairness to each flow may significantly degrade the system throughput [43, 42, 45, 47, 58], and reduce the actual data rate of each flow. As a result, the WNI of the MT may stay longer in active and consume much more power to receive a certain amount of data.

Based on the above reasoning, we need to balance the tradeoff between power efficiency and short-term fairness among flows by modifying SFQ to provide *temporal fairness* to each flow. In each time slot, the system exploits the highest possible transmission rate for the serving flow according to the channel condition. Flows with high transmission rate can get high power efficiency, while flows with low transmission rate can avoid being starved for a long time. Similar to SFQ, each packet has two tags: the start tag and the finish tag. The only difference between our scheme and SFQ is that we use different rules to calculate the finish tag of each packet. Suppose the jth packet of f_i, denoted by p_i^j, is transmitted at the rate of $X_i \in \{C^1, C^2, ..., C^M\}$, where C^M

is the highest available transmission rate. The start tag S_i^j and finish tag F_i^j of p_i^j are defined by:

$$S_i^j = max\{V(t), F_i^{j-1}\},\ F_i^j = S_i^j + \frac{C^M l_i^j}{X_i r_i} \tag{2.14}$$

where $V(t)$ is the virtual time of the system and equal to the start time of the packet in service[3], l_i^j (in bits) is the length of p_i^j and r_i (in bps) is the data rate of f_i. The packets are served in increasing order of the associated start tag. Compared to SFQ, the normalized service of p_i^j is scaled by $\frac{C^M}{X_i}$, which means that the amount of data transmitted per time slot is accounted as if each flow were served in error-free channel. In this way, the time slots can be fairly allocated in proportion to the data rate of each flow.

2.4.2.2 Service Accounting

With RBS, an active flow with enough ahead-service may be suspended by the scheduler for a time period. In order to calculate the ahead-service, each flow f_i is associated with a service counter W_i (in seconds). Suppose f_i is served during the time period of $[t_1, t_2]$. The service counted for f_i during $[t_1, t_2]$, denoted by $W_i(t_1, t_2)$, can be calculated by:

$$W_i(t_1, t_2) = \sum_{j=k}^{l} \frac{l_i^j}{r_i} \tag{2.15}$$

[3]If the server is idle at t, $V(t)$ is set to the maximum finish tag of the packet that has been served by t.

Notations:
$\mathcal{A}$**:** the set of flows in active
t**:** the current time
S_i, F_i**:** the start tag and finish tag of the head-of-line packet of f_i

```
1  begin:
2  if (A == NULL)
3      { idle in the time slot; goto begin; }
4  f_i = arg min_{f_j ∈ A} {S_j};
5  p=f_i.deque(); /* get the packet to be transmitted */
6  if (ahead_i > Maxserv_i ∨ (ahead_i ≥ φ ∧ ∃j(f_j ∈ A ∧ ahead_j < φ))
7      mark(p);
8  send (p);
9  if (transmission is successful)
10     { if (p is marked)
11         { state_i = idle; e_i = t + ahead_i; }
12     else
13         update S_i and F_i according to Eq (2.14); }
14 goto begin;
```

Fig. 2.11. The RBS algorithm at the BS

where p_i^k and p_i^l is the first and last packet of f_i served or being served during $[t_1, t_2]$. In practice, t_1 can be the time when the flow starts the session. Eq (2.15) alone cannot precisely keep track the amount of buffered service (in seconds) at the client side. Since the buffer size is no less than zero, we also need to take into account the time elapsed when the client's buffer is empty. As a result, the obtained ahead-service (in seconds) of f_i during $[t_1, t_2]$, denoted by $ahead_i(t_1, t_2)$, is calculated by:

$$ahead_i(t_1, t_2) = W_i(t_1, t_2) - (t_2 - t_1) + \int_{t_1}^{t_2} \mathbf{I}(W_i(t_1, s) + t_1 < s)ds \tag{2.16}$$

where $\mathbf{I}(x)$ is 1 if x is true; otherwise it is 0. The integration part is used to compute the time elapsed when the buffer is empty during $[t_1, t_2]$.

2.4.2.3 Dealing with Channel Errors

Sometimes, the channel condition may be too bad to be tolerated by using modulation and coding schemes. At this time, the probability of a successful transmission becomes low, and bandwidth and power may be wasted during re-transmissions. To deal with channel errors, time slots are swapped from flows suffering channel error to flows with good channel conditions. In particular, when f_i has channel errors:

- **If** $ahead_i \geq \phi$, the scheduler suspends f_i and sends a marked null packet to MT_i. When MT_i receives the packet, the proxy shuts down the WNI.
- **Else if** $ahead_i < \phi$, it may not worth to power off the WNI since the actual sleep time could be too short. Since the time period of some channel errors may be short (especially in the case of fast fading), it is better to postpone serving f_i a little

bit. Thus, the RBS scheduler stops serving f_i for a pre-specified period, called the *backoff period*, during which the WNI of f_i stays in active. After the backoff period, the scheduler tries to resume serving f_i. If f_i still suffers from channel errors, the same backoff procedure is applied to f_i again.

For the idle flows in the system, simply turning on the WNI when the buffered data runs out may not be enough to tolerate channel errors. For example, as shown in

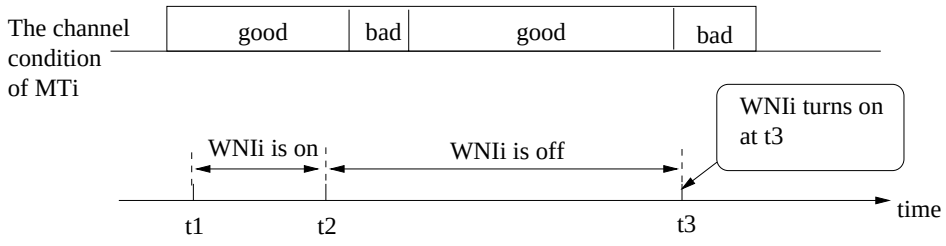


Fig. 2.12. An example of the impact of channel errors

Figure 2.12, suppose the WNI of MT_i is active from t_1 to t_2, and is turned off at t_2 after it has buffered enough data. At t_3, when the buffer is near empty, the WNI wakes up to receive more data to fill the buffer. At the same time, since the channel condition is continuously bad, most data packets addressed to MT_i will be corrupted, and the QoS requirements cannot be met. To alleviate the impact of channel errors on QoS, it should be better to let the WNI wake up earlier than t_3 so that MT_i could have got data under good channel conditions. It is easy to see that this approach may increase the power consumption since the sleep time is reduced. To balance the tradeoff between

error-resilience and power conservation, we design a novel adaptive scheme to adjust the sleeping time of the WNI according to the channel condition.

At any time t, a probabilistic function is used to evaluate the QoS provision of a flow f_i, which is defined as follows:

$$Pr(W_i(t_1, t) - (t - t_1) < \delta_i) \leq \mathcal{P}_i \tag{2.17}$$

where δ_i is the threshold of service deficiency of f_i and it is less than zero (e.g. -20 ms[4]), and $\mathcal{P}_i$ is the target probability. For a fixed $\mathcal{P}_i$, a smaller $|\delta_i|$ provides more stringent delay requirement. For each flow f_i, it has a control parameter θ_i (in second) indicating how early the WNI should be turned on before the buffered data is depleted. The value of θ_i is adjusted as follows: At time t, the observation function of f_i, denoted by $\mathcal{O}_i$, is defined as:

$$\mathcal{O}_i = \frac{\int_{t_1}^{t} I(W_i(t_1, s) + t_1 < s + \delta_i) ds}{t - t_1} \tag{2.18}$$

In time slot k, θ_i is updated as follows:

$$\theta_i(k+1) = \begin{cases} min\{\theta_i(k) + (\mathcal{P}_i - \mathcal{O}_i)\delta_i, \theta_{max}\}, & \mathcal{O}_i > \mathcal{P}_i \\ \theta_i(k), & \mathcal{O}_i = \mathcal{P}_i \\ max\{\theta_i(k) + (\mathcal{P}_i - \mathcal{O}_i)\delta_i, 0\}, & \mathcal{O}_i < \mathcal{P}_i \end{cases} \tag{2.19}$$

[4]The value of δ_i can be set based on the maximum delay requirement of the application.

where θ_{max} is a pre-specified system parameter and less than ϕ. From Eq (2.19), we can see that θ_i is bounded by $[0, \theta_{max}]$. When f_i is suspended with $ahead_i$ at time t, the eligible time e_i is adjusted to $e_i = t + ahead_i - \theta_i$.

2.4.2.4 The RBS Proxy

The proxy of RBS is almost the same as that of PBS. The only difference is that: if the adaptive scheme for error-resilience is applied, both the BS and the proxy need to run the same algorithm. Otherwise, θ_i is always zero. When the proxy of f_i finds that the received packet is marked, the WNI will sleep for a time period of $max\{0,$ $buffer_i - \theta_i - transition\ delay\ \}$.

2.4.3 Analysis of RBS

In this section, we prove that RBS can provide the delay guarantee and achieve power efficiency for all flows in the system. For simplicity, we assume the channel is error-free, and the $Maxserv$ of each flow is infinite. Since the channel is error-free, the modified SFQ is the same as the original one. We first present the delay guarantee of RBS. The *expected arrival time* of packet p_i^j served by a RBS server is defined as:

$$EAT(p_i^j) = EAT(p_i^k) + \sum_{m=k}^{j-1} \frac{l_i^m}{r_i}, \quad j > k \geq 0 \tag{2.20}$$

where p_i^k is the last packet served before flow f_i is activated.

THEOREM 1. *If Q is the set of all backlogged flows in service by a RBS server, and $\sum_{n \in Q} r_n \leq C$, the RBS scheduler will serve each packet p_i^j conforming to:*

$$d_i^j - EAT(p_i^j) \leq \frac{|Q|L^{max}}{C} \tag{2.21}$$

where d_i^j is the departure time of p_i^j and L^{max} is the maximum packet length.

Proof: The set Q can be partitioned into two subsets: the set of the active flows, denoted by $\mathcal{A}$, and the set of idle flows denoted by $\mathcal{I}$. At any time t, flow f_i has two cases:

- **case 1:** $f_i \in \mathcal{I}$. Suppose p_i^j is the last packet served before f_i was suspended. Since $EAT(p_i^j) > t$ (otherwise $f_i \in \mathcal{A}$) and $d_i^j \leq t$, the theorem holds.

- **case 2:** $f_i \in \mathcal{A}$. Suppose the last time f_i was activated is t_a. Since f_i is served with SFQ during the interval $[t_a, t]$, with the same arguments of Theorem 4 in [25] and $\mathcal{A} \subseteq Q$, we get:

$$\begin{aligned} d_i^j - EAT(p_i^j) &\leq \sum_{k \in \mathcal{A} \wedge k \neq i} \frac{L^{max}}{C} + \frac{L^{max}}{C} \\ &\leq \frac{|Q|L^{max}}{C} \end{aligned}$$

□

LEMMA 1. *Suppose Q is the set of backlogged flows being served by a SFQ server during $[t_1, t_2]$, $\forall (n \in Q) ahead_n(t_1) = 0$ and $\sum_{n \in Q} r_n < C$, the following property holds for f_i:*

$$\begin{aligned} &(\frac{C - \sum_{n \in Q} r_n}{\sum_{n \in Q} r_n})(t_2 - t_1) - \frac{|Q|L^{max}}{\sum_{n \in Q} r_n} \leq ahead_i(t_1, t_2) < \\ &\qquad (\frac{C - \sum_{n \in Q} r_n}{\sum_{n \in Q} r_n})(t_2 - t_1) + (\frac{1}{r_i} + \frac{|Q| - 1}{C})L^{max} \end{aligned} \tag{2.22}$$

Proof: By adopting the concept of the *rate controller* in [7], the actual data rate of f_i, denoted by $\hat{r_i}$, is equal to $\hat{r_i} = \frac{r_i}{\sum_{n \in Q} r_n} C$. With the concept of the *latency-rate server* [62] and Theorem 1, under SFQ, the aggregated service (in bits) of f_i during $[t_1, t_2]$, denoted by $S_i(t_1, t_2)$, follows:

$$S_i(t_1, t_2) \geq \hat{r}_i(t_2 - t_1) - \frac{\hat{r}_i |Q| L^{max}}{C} \tag{2.23}$$

Meanwhile, since the SFQ scheduler serves the packet with the minimum *start tag*, which must be eligible to be served in the corresponding GPS system, with Theorem 1 of [7], we get:

$$S_i(t_1, t_2) \leq \hat{r}_i(t_2 - t_1) + (1 - \frac{\hat{r}_i}{C}) L^{max} \tag{2.24}$$

Since $S_i(t_1, t_2) = W_i(t_1, t_2) * r_i$ and $\hat{r_i} = \frac{C}{\sum_{n \in Q} r_n} r_i$, combining Eq (2.23) and Eq (2.24), we have:

$$\frac{C}{\sum_{n \in Q} r_n}(t_2 - t_1) - \frac{|Q| L^{max}}{\sum_{n \in Q} r_n} \leq W_i(t_1, t_2) < \frac{C}{\sum_{n \in Q} r_n}(t_2 - t_1) + (\frac{1}{r_i} - \frac{1}{C}) L^{max} \tag{2.25}$$

With Eq (2.23), we get:

$$0 \leq \int_{t_1}^{t_2} \mathbf{I}(W_i(t_1, s) - t_1 < s) ds \leq \frac{|Q| L^{max}}{C} \tag{2.26}$$

With Eq(2.16), Eq (2.25) and Eq (2.26), by simple conversions, the Lemma holds. □

LEMMA 2. *Suppose Q is the set of backlogged flows being served by a RBS server from t_1, $\forall(i \in Q) ahead_i(t_1) = 0$, and $\sum_{n\in Q} r_n < C$. If ϕ satisfies $\forall(i \in Q), \phi > \frac{\sum_{n\in Q} r_n(\frac{1}{r_i}+\frac{|Q|-1}{C})L^{max}}{C-\sum_{n\in Q} r_n}$, there are at most $|Q|-1$ flows sharing the channel together after $t_1 + \frac{\sum_{n\in Q} r_n\phi}{C-\sum_{n\in Q} r_n}$.*

proof: Since $Maxserv$ is infinite for each flow, according to the RBS scheme, the RBS server is always busy. Thus, during any time interval $[t_1, t]$, $C(t-t_1) = \sum_{i\in Q} S_i(t_1, t)$ holds. Since

$$\forall(i \in Q), S_i(t_1,t) \leq W_i(t_1,t)r_i + \int_{t_1}^{t} \mathbf{I}(W_i(t_1,s) - t_1 < s)ds = (ahead_i(t_1,t) + (t-t_1))r_i$$

we get:

$$\sum_{n\in Q} ahead_n(t_1,t)r_n \geq (C - \sum_{n\in Q} r_n)(t-t_1) \tag{2.27}$$

Suppose at $t^*(t^* > t_1)$, f_i is the *first* flow that gets the ahead-service of at least ϕ, with Lemma 1 and Eq (2.27), we get

$$t^* > t_1 + \frac{\phi - (\frac{1}{r_i} + \frac{|Q|-1}{C})L^{max}}{\frac{C-\sum_{n\in Q} r_n}{\sum_{n\in Q} r_n}} \tag{2.28}$$

Now, we prove the Lemma by contradiction. Suppose at time $\tilde{t}$ $(\tilde{t} > t^*)$, all flows are served together again. With RBS, the ahead-service of each flow must be less than

ϕ, we have:

$$\tilde{t} < t_1 + \frac{\phi}{\frac{C}{\sum_{n\in Q} r_n} - 1} \tag{2.29}$$

Since $\phi > \frac{\sum_{n\in Q} r_n(\frac{1}{r_i}+\frac{|Q|-1}{C})L^{max}}{C-\sum_{n\in Q} r_n}$ and $e_i \geq t^* + \phi$, with Eq (2.28) and Eq (2.29), we have $\tilde{t} - e_i < 0$, which contradicts the assumption that all flows are served at $\tilde{t}$. □

LEMMA 3. *Suppose a backlogged flow f_i is served by a RBS server. During each period when f_i is in active state, the total amount of transmitted data (in bits) is no less than $\frac{\phi r_i C}{C-r_i}$.*

Proof: Suppose flow f_i receives B_k^i bits during the active period of Δt_i. With RBS, f_i is suspended only after it has the ahead-service of at least ϕ. Then we have

$$B_k^i \geq (\Delta t_i + \phi - \int_{t_{a,i}^k}^{t_{a,i}^k+\Delta t_i} \mathbf{I}(W_i(t_{a,i}^k, s) - t_{a,i}^k < s)ds)r_i \tag{2.30}$$

where $t_{a,i}^k$ is the time when f_i starts to transmit B_k^i. As explained in Lemma 1, $\int_{t_{a,i}^k}^{t_{a,i}^k+\Delta t_i} \mathbf{I}(W_i(t_{a,i}^k, s) - t_{a,i}^k < s)ds$ is bounded by $\frac{|Q|L^{max}}{C}$, we can see that B_k decreases as Δt_1 decreases. Since Δt_i reaches the minimum value when f_i is served alone during Δt_i, we have ${\Delta t_i}^{min} = \frac{\phi r_i}{C-r_i}$. Since ${\Delta t_i}^{min}$ is achieved when f_i is served alone, we have $B_k^i \geq (\frac{\phi r_i}{C-r_i} + \phi)r_i = \frac{\phi r_i C}{C-r_i}$. □

THEOREM 2. *Suppose Q is the set of backlogged flows being served by a RBS server from t_1, $\forall(i \in Q) ahead_i(t_1) = 0$, and $\sum_{n\in Q} r_n < C$. Assume the following two conditions hold*

to transmit B bits ($B \gg \phi C$):

$$(1)\forall(i \in Q), \phi > \frac{\sum_{n\in Q} r_n(\frac{1}{r_i} + \frac{|Q|-1}{C})L^{max}}{C - \sum_{n\in Q} r_n} \tag{2.31}$$

$$(2)\forall(i \in Q), (\frac{\phi r_{min}}{C - r_i} - \frac{(|Q|-1)L^{max}}{C} - \frac{\sum_{n\in Q} r_n(\frac{1}{r_i} - \frac{1}{C})L^{max}}{C})P_a > E_s \tag{2.32}$$

where P_a (in Watt) is the power consumption in active, E_s (in Joule) is the energy used by a state transition and $r_{min} = min_{n\in Q}\{r_n\}$. Then, RBS is more power efficient than SFQ.

Proof: As shown in Figure 2.13, for a flow, say f_i, served by a RBS server, the total

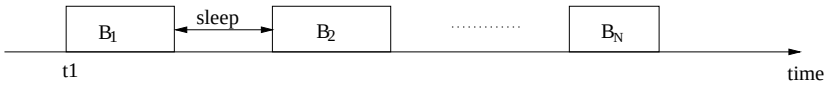


Fig. 2.13. An illustration of data transmission under RBS

data of B is transmitted in the form of several runs of data (i.e., $\sum_{k=1}^{N} B_k = B, N > 2$). With condition 1 and Lemma 2, f_i can be served with the actual data rate of at least $\frac{r_i}{\sum_{n\in Q} r_n - r_{min}}C$ during the transmission of $B_k(k = 2...N)$. With Lemma 3, we have $B_k \geq \frac{\phi r_i C}{C - r_i}$.[5]

Now, let's compare the power consumption used by the WNI for receiving B data between RBS and SFQ. With Eq (2.23), we have the time spent by f_i to transmit B

[5]It is possible that B_N does not have this property. In this case, without loss of correctness, we discard B_N by setting $N = N - 1$.

data in SFQ, denoted by $T_i^{SFQ}(B)$, follows:

$$T_i^{SFQ} \geq \frac{B - (1 - \frac{r_i}{C})L^{max}}{\frac{r_i}{\sum_{n \in Q} r_n} C} \tag{2.33}$$

Thus, the energy (in Joule) used by the WNI of f_i for receiving B data in SFQ, denoted by $E_i^{SFQ}(B)$, follows: $E_i^{SFQ}(B) \geq \frac{B-(1-\frac{r_i}{C})L^{max}}{\frac{r_i}{\sum_{n \in Q} r_n} C} P_a$. With RBS, according to Eq (2.23), condition 1 and Lemma 2, the maximum time to transmit B_k data is less than or equal to $\frac{(\sum_{n \in Q} r_n - r_{min})B_k}{r_i C} + \frac{(|Q|-1)L^{max}}{C} (k = 2...N)$. The power consumed by the WNI for receiving B_1 data in RBS is less than or equal to that in SFQ. Since $B = \sum_{i=1}^{N} B_i$, the power consumption difference between SFQ and RBS follows:

$$\begin{aligned}
E_i^{SFQ}(B) - E_i^{RBS}(B) \geq & \quad \sum_{k=2}^{N} (\frac{\sum_{n \in Q} r_n B_k}{r_i C} P_a - E_i^{RBS}(B_k) - E_s) - \frac{\sum_{n \in Q} r_n (\frac{1}{r_i} - \frac{1}{C}) L^{max}}{C} P_a \\
\geq & \quad \sum_{k=2}^{N} ((\frac{\sum_{n \in Q} r_n \phi}{C - r_i} - \frac{(\sum_{n \in Q} r_n - r_{min})\phi}{C - r_i} - \frac{(|Q|-1)L^{max}}{C}) P_a - E_s) \\
& - \frac{(\sum_{n \in Q} r_n)(\frac{1}{r_i} - \frac{1}{C}) L^{max}}{C} P_a \\
\geq & \quad \sum_{k=2}^{N} ((\frac{\phi r_{min}}{C - r_i} - \frac{(|Q|-1)L^{max}}{C}) P_a - E_s) - \frac{(\sum_{n \in Q} r_n)(\frac{1}{r_i} - \frac{1}{C}) L^{max}}{C} P_a
\end{aligned}$$

With condition 2, we have $E_i^{SFQ}(B) > E_i^{RBS}(B)$. □

Theorem 2 gives the sufficient conditions to guarantee that RBS is more power efficient than SFQ. In terms of power consumption, referring to the calculation of $T_i^{SFQ}(B)$, we can see that the power consumptions of other rate-based service models [68, 71] are similar to SFQ. Thus, the RBS model is also more power efficient than other rate-based service models.

2.4.4 Performance Evaluations

2.4.4.1 The Experimental Setup and Parameters

We evaluate the performance of RBS through a case study called *Audio-on-Demand (AoD)*. The simulation setup is almost the same as that of PBS. The main difference is the channel error model. Based on the results of [66], when the channel

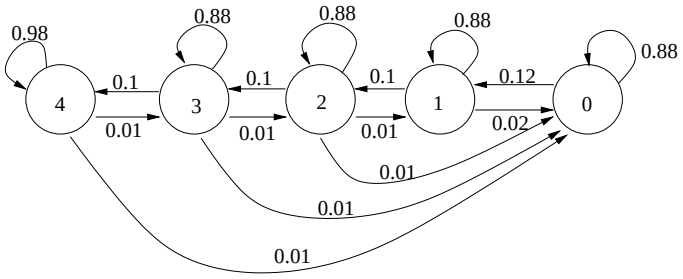


Fig. 2.14. The error-prone channel model

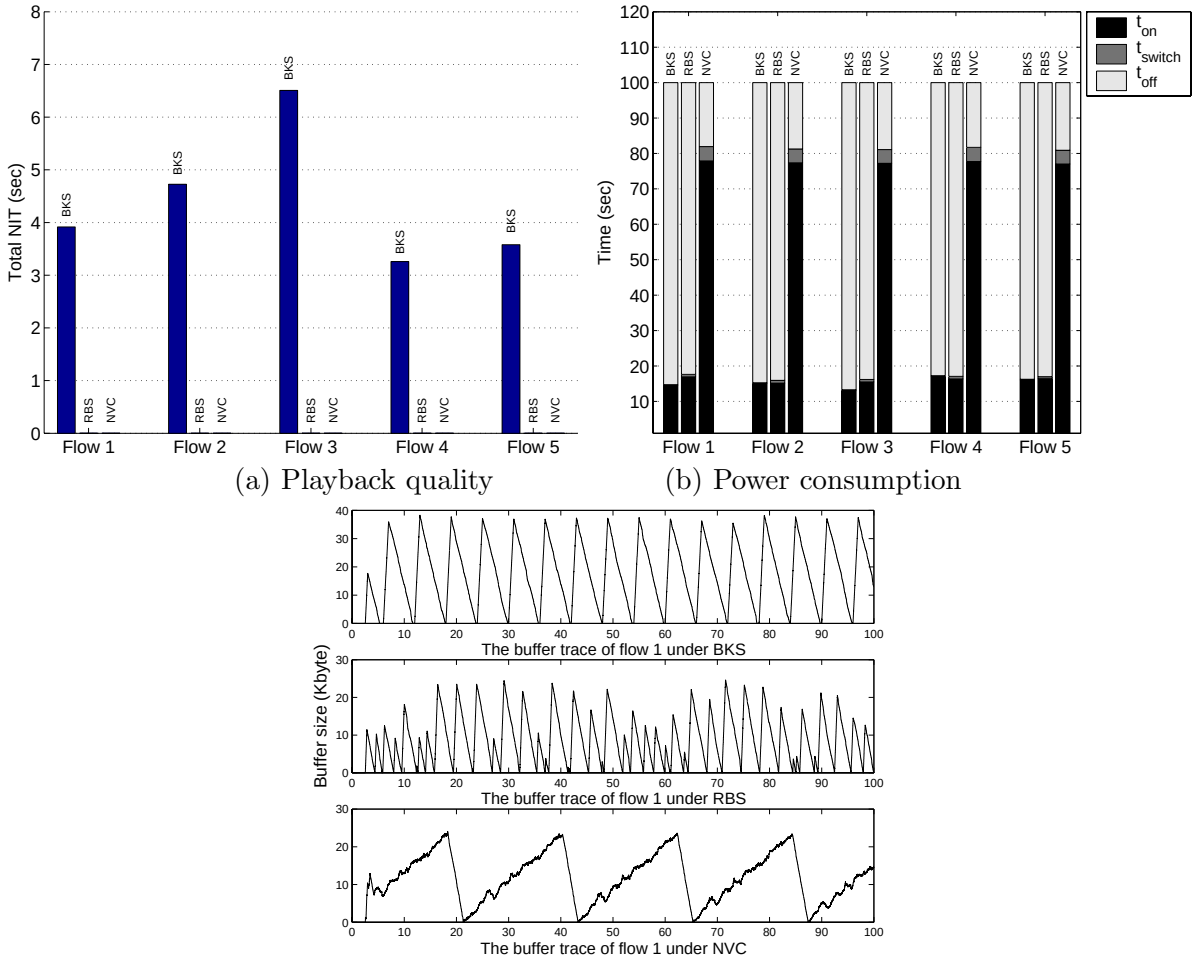
is error-prone, a five-state Markov chain is used to emulate the process of channel conditions with fast fading. As shown in Figure 2.14, the marked line or curve shows the transition probability from one state to another. The transmission rate in state 4 is equal to the capacity of the AoD channel which is assumed to be $384Kbps$, and is zero when the channel condition is in state 0. The channel is accessed based on TDMA. Following the standard of EDGE [64], each time slot is $2.5ms$, which can be used to transmit 112, 74, 56, 44 bytes of data (not including the header) in state 4, 3, 2, 1 respectively.

Similar to RBS, flows with channel errors in NVC also yield the channel, and backoff for a backoff period, which is set to be $15ms$ for both RBS and NVC.

2.4.4.2 Scenario 1: Error-Free Channel

In this scenario, five identical AoD flows in an error-free channel are used to show the fundamental differences among RBS, BKS, and NVC. From Figure 2.15 (a), we can see that the playback quality of RBS and NVC are perfect since their total NITs are 0. However, BKS violates the delay requirement a lot. For example, the total NITs of all flows are greater than 3 seconds. This can be explained as follows. Since there are five flows sharing the channel, it is highly possible that more than one flows request data at the beginning of a bulk slot. Since BKS only randomly selects one winning flow to serve, other losing flows cannot be served during the bulk slot. Because the bulk slot is 1.0 second, all the losing flows are starved for 1.0 second.

Although the playback quality under NVC is perfect, from Figure 2.15 (b), we can see that the power consumption of the WNI under NVC is significantly larger than that under RBS. This is due to the fact that the data rate of the flow is quite high and many of the inter-packet arrival periods are less than $T_{on\rightarrow off} + T_{off\rightarrow on}$. As a result, the WNI cannot frequently go to sleep in the NVC approach. Even though the WNI can enter sleep, the actual sleep period is significantly reduced by $T_{on\rightarrow off} + T_{off\rightarrow on}$. For example, if the data rate is $56Kbps$ and the maximum payload of a packet is 112 Bytes, the inter-packet arrival time is $16ms$. Then, the actual sleep period is only $1ms$ and it is not worth to power off the WNI. Note that no data will be received during state transition.



(a) Playback quality (b) Power consumption

(c) Buffer trace

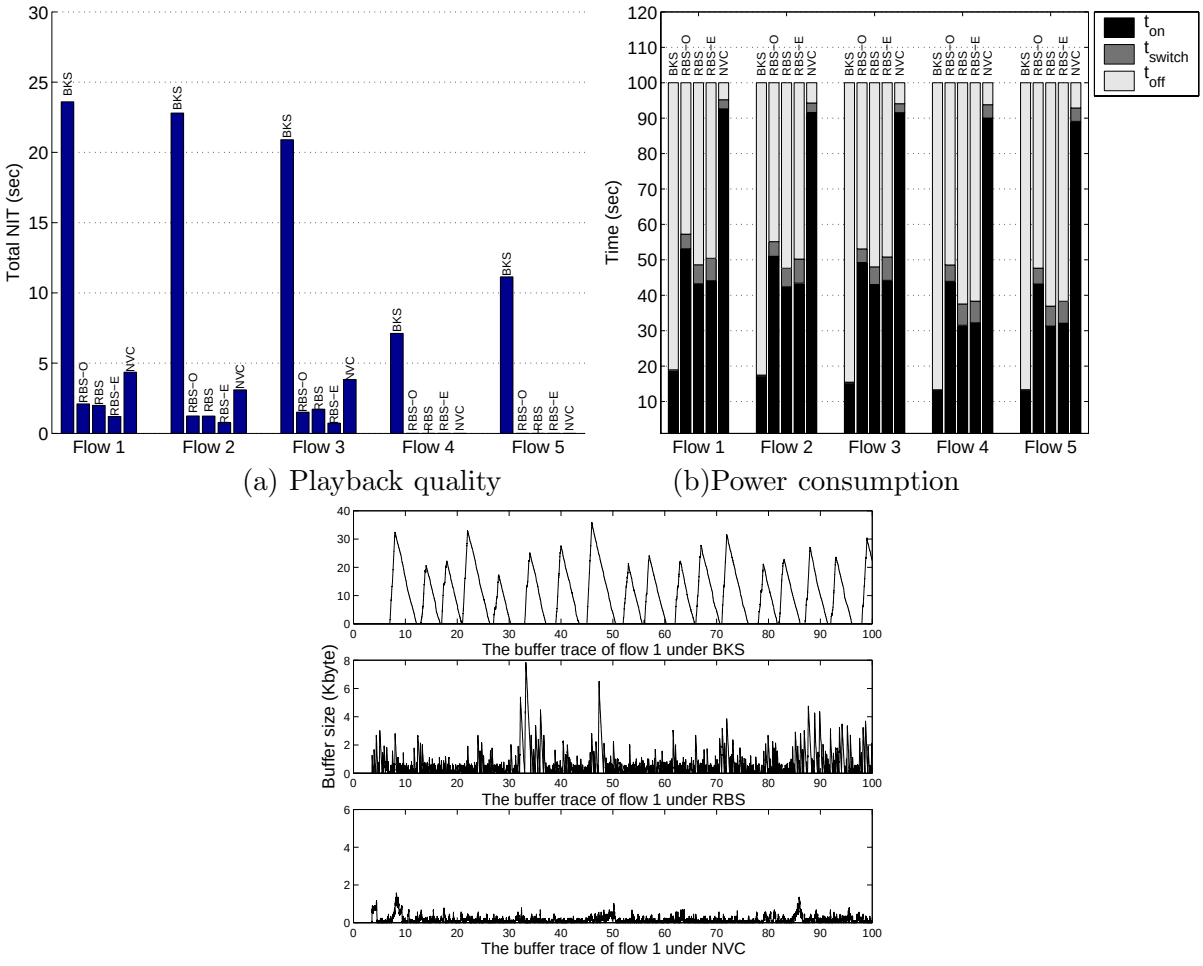
Fig. 2.15. Performance comparisons among BKS, RBS, and NVC under in error-free channel

Figure 2.15 (c) shows how these three approaches work through the buffer trace. When the buffer size goes up, the WNI receives data in the active mode. The slope of buffer increment indicates how fast the flow fills the buffer. For BKS, since the flow is alway served alone during a bulk slot, its actual data rate is equal to the channel capacity. During many time periods, the slope in NVC is much less than that in RBS, which means that the flow takes more time to get a certain amount of data in NVC than in RBS. This is because the RBS scheduler suspends flows with enough ahead-service so that the remaining active flows can have much higher actual data rate. In NVC, since the inter-packet arrival period is too short, the flow has to keep active and compete with other flows for the channel. As a result, the actual data rate of the flow is not as high as that in RBS.

2.4.4.3 Scenario 2: Error-Prone Channel

In this subsection, we evaluate the impacts of channel errors. To evaluate the effect of temporal fairness, we compare RBS and the *RBS-O* scheme, which applies the original SFQ [25] and provides short-term throughput fairness to each active flow. For RBS, we also study the performance of the error-resilient enhancement (see the adaptive scheme in Section 2.4.2.3), and denote it as *RBS-E*. We introduce channel errors in this scenario. Specifically, we assume flow 1-3 have channel error and the channel of flows 4 and 5 is error-free. For each flow, $\mathcal{P}_i$, δ_i and θ_{max} set to be 0.01, $-10ms$ and $40ms$ respectively.

As shown in Figure 2.16 (a), the playback quality of each flow in BKS is much worse than that in RBS. The poor playback quality of BKS has been explained in Scenario



(a) Playback quality (b)Power consumption

(c) Buffer trace

Fig. 2.16. Performance comparisons among BKS, RBS, RBS-O, RBS-E, and NVC in error-prone channel

1, and the reason is still valid for this example. Comparing RBS and RBS-O in Figure 2.16 (a) and (b), we can see that, for each flow, RBS saves much more power than RBS-O without loss of playback quality. This verifies that RBS can achieve a good balance between power efficiency and fairness.

With the backoff mechanism for flows with channel error, the playback quality of flow 4 and 5 in RBS and NVC are not affected by channel errors. However, NVC results in much higher power consumption than RBS. Moreover, the playback qualities of flow 1-3 in NVC are much worse than those in RBS. For NVC, as explained in Scenario 1, there are alway a large number of flows sharing the channel and each flow has low data rate. Meanwhile, the system throughput is degraded due to channel errors. As a result, the actual data rate of the flow could be too low to provide QoS sometimes. In RBS, at a time, the number of flows sharing the channel could be much smaller than in NVC. As a result, on the average, each flow can buffer data faster than in NVC. With more buffered data, the playback quality of each flow in RBS is better than that in NVC.

From Figure 2.16 (b), we can see that the power consumptions in BKS, RBS and NVC become larger than those in Scenario 1. This is mainly caused by the decreased system throughput due to channel errors. Comparing BKS and RBS, we can see that the difference of the power consumption is greater than that in the error-free channel. For example, t_{switch} of each flow in RBS increases. Since the actual data rate of each flow decreases, on average, the amount of buffered data is reduced. This can be easily verified by comparing the amount of buffered data in Figure 2.15 (c) and Figure 2.16 (c).

To deal with channel error, we use the error-resilient enhancement to control how long the WNI should sleep based on the channel condition. From Figure 2.16 (a), we can see that RBS-E provides better playback quality than RBS. On average, the total NIT of each flow with channel errors can be reduced by 50%. By turning on the WNI earlier, the flow could have more chance to buffer more data to tolerate long period of error. As shown in Figure 2.16 (b), t_{on} in RBS-E is almost the same as that in RBS, but t_{switch} in RBS-E is more than that in RBS. When the WNI wakes up before the buffered data runs out, its sleep time is reduced, which increases the number of state transitions. Since we use the adaptive scheme to carefully adjust the time to power on the WNI according to the channel condition, the power consumption of the WNI in RBS-E is slightly higher than that in RBS.

2.5 Related Work

Power management in wireless networks has received considerable attention. IEEE 802.11 [30] supports power saving mode in which the WNI only needs to be active periodically. In a wireless LAN, the WNI in sleep mode only wakes up periodically to check for possible incoming packets from the BS. The BS transmits a *beacon frame* after a regular *beacon interval*. In each beacon frame, a *traffic indication map* (TIM) contains information about which WNI has buffered incoming packets. If the WNI finds that it has incoming packets, it should stay active to receive the packets. However, this mechanism does not work well for streaming applications since the continuously arriving packets forces TIMs to always report new data, keeping the WNI stay active.

Yuan *et al* [70] proposed a sender-buffering approach for video sensors. With the sender-buffering approach, each sensor buffers the encoded frames and transmits them in bursts. As a result, it can prolong the WNI sleeping time but slow down the CPU speed. Meanwhile, Chandra *et al* [12] proposed a proxy-buffering approach to support power-efficient multimedia playback in wireless LANs. The local proxy shapes the traffic from the access point to the client in bursts and informs the client of the next packet arrival time using a special control packet. Compared to the RBS scheme, these two schemes may incur a long delay to a flow because their scheduling algorithms cannot provide guaranteed service to each flow. Similar to the BKS scheme, when multiple MTs share the channel at the same time, they cannot support streaming applications in wireless networks without significantly degrading playback quality.

Zhang *et al* [72] proposed a frame-based scheduling scheme, called the scheduled contention free burst (S-CFB), to achieve energy efficient data transmission with delay guarantees. With the predefined schedule, WNIs can be turned off when they are not in use. However, if the message length varies during each transmission burst [72], bandwidth may be wasted since the burst is non-preemptable. Moreover, when considering the impact of channel errors, it becomes more difficult to balance the tradeoff between the length of the transmission burst and the bandwidth utilization. Different from S-CFB, RBS serves each flow on the granularity of packet, which is more flexible in terms of variable packet length and dynamic packet arrival pattern. Also, RBS provides mechanisms to deal with channel errors.

An energy conserving medium access control (EC-MAC) scheme for wireless and mobile ATM networks was proposed in [14]. EC-MAC was designed for supporting

multimedia traffic and providing QoS for wireless ATM networks. The authors proposed a priority frame-based round robin scheduling scheme considering dynamic reservation update and error compensation. Power saving is achieved by allocating contiguous time slots for each flow. Therefore, in each frame, the WNI only needs to be active during its data phase. However, EC-MAC does not consider the state transition delay. If the frame length is too short, the WNI may not be able to go to sleep due to the transition delay. To achieve high power efficiency, the frame length should be significantly increased, which may increase the queuing delay. Compared with EC-MAC, the RBS scheme considers the issue of state transition delay, and allocates bandwidth on the granularity of per packet rather than per frame, which is flexible to handle the problems of variable packet lengths and dynamic packet arrival patterns.

Prabhakar *et al* [51] studied power conservation with regard to scheduling. They show that the power consumption can be significantly reduced by lowing the transmission power and transmitting the packet over a long period of time. Based on this motivation, the Lazy Packet Scheduling (LPS) approach is proposed to reduce the transmission rate for every packet without violating the deadline of each packet. The approach has been proven to be power optimal. The main difference between LPS and RBS is that: LPS focuses on reducing the power consumed by the WNI of the sender by changing the transmission rate, whereas RBS focuses on reducing the power consumption of the WNI of the receiver (i.e. the MTs) by powering off the WNI.

Fitzek and Reisslein [19] proposed a prefetching protocol to support high performance streaming applications over wireless links. Parts of the ongoing media streams are prefetched into buffers in clients according to a *join-the-shortest-queue* (JSQ) policy.

The JSQ dynamically allocates more bandwidth to the clients with small buffered data while allocating less bandwidth to the clients with large prefetched reserves. With the buffered data, the clients can have smooth playback quality during the periods of adverse transmission conditions on the wireless links. The basic idea of PBS is similar to JSQ. However, PBS focuses on the aspect of power conservation of WNIs, while JSQ deals with channel errors of wireless links. Furthermore, PBS also considers how to achieve power efficient communication with regard to scheduling.

Chapter 3

The Absence Compensation Fair Queuing

3.1 Background and Motivation

To carefully manage the limited bandwidth of wireless links, one widely used bandwidth management approach is to apply the wireline fair queuing schemes (e.g., weighted fair queuing (WFQ) [17, 71]) to the wireless environment considering the characteristics of wireless channels such as time-varying channel conditions. Fair queuing serves flows in proportion to their pre-specified weights, and isolates the mis-behaving flows. Most recent studies in fair queuing are motivated by *fluid fair queuing* (FFQ). FFQ guarantees that for any time period $[t_1, t_2]$, any two *backlogged* flows f_i and f_j have the same amount of normalized service, i.e., $W_i(t_1, t_2)/r_i = W_j(t_1, t_2)/r_j$, where $W_i(t_1, t_2)$ is the service (in bits) received by f_i and r_i is the weight of f_i. However, since network schedulers serve flows at the granularity of packet, and the service is non-preemptive, the service of each flow cannot be counted in bits. Therefore, all the existing packetized fair queuing models [7, 17, 21, 25, 49] try to emulate FFQ in the unit of packet and have a bounded $|W_i(t_1, t_2)/r_i - W_j(t_1, t_2)/r_j|$. In the long run, these service models behave similarly in term of providing an equal amount of normalized service to each flow.

As studied in [32], under bursty data traffic, when the network utilization is not very high, WFQ only provides little service differentiation to the flows. The reason is as follows. When a flow is *absent*; i.e., the flow does not have any data in its queue,

the WFQ scheduler distributes the service that belongs to the absent flow (if it were backlogged) to all backlogged flows in the system. As a result, backlogged low-weight flows could get many extra services from the absent high-weight flows and then their actual data rate could be quite high. Since the WFQ model emulates (FFQ) [17] which does not compensate the service loss of a flow due to absence, after the low-weight flows take extra service from the absent high-weight flows, these high-weight flows cannot reclaim the service loss when they become backlogged again. Thus, under bursty data traffic, WFQ can only provide good service differentiation when the network utilization is very high, in which case almost all flows are backlogged.

This phenomenon reflects the gap between network-level fairness and application-level fairness under bursty data traffic. Since most of the current wireless fair queuing schemes [7, 17, 21, 25, 49] follow the principle of FFQ, they also suffer from the same problem. One simple solution to increase service differentiation under bursty data traffic is to use the *strict priority queuing* (SPQ) model [18]. Under SPQ, high-weight flows are granted the exclusive priority over flows with low weights. Therefore, low-weight flows cannot be served whenever there are some backlogged high-weight flows in the system. It is easy to see that SPQ can achieve the maximum service differentiation between high-weight and low-weight flows. However, low-weight flows do not have any QoS guarantee under SPQ since they may be starved if there are backlogged high-weight flows. As a result, we need to design new service model to *achieve better service differentiation without loss of QoS provision.*

3.2 The System Model

We only consider the wireless part of the communication system, where mobile users communicate with wireless base stations directly. The wireless link is accessed in TDMA and is managed by the base station. The packet scheduling algorithms discussed in this paper allocate time slots to packets of users within the coverage area of the base station. We assume the majority of the traffic is from the base stations to the mobile users, and the sole congestion point of the system is the downlink. Similar to many existing works [44, 45], we assume that the base station has a way to obtain the channel condition. Based on the information of channel conditions, the system (e.g., EDGE [64]) selects the most suitable modulation and coding scheme to transmit the impending packet. For each time slot, there is a set of possible transmission rates $\{0, C^1, C^2, ..., C^M\}$. We assume the service provider uses the scheduler to implement the *Olympic service model* [18], which assigns different service weights to different service classes (i.e., the 'Gold', 'Silver', and 'Bronze').

3.3 The Absence Compensation Fair Queuing (ACFQ) Service Model

3.3.1 Overview of ACFQ

In order to improve service differentiation and provide QoS, the ACFQ service model is designed with the following objectives:

1. The model should be backward compatible to the existing fair service models in terms of QoS provision. Therefore, we design the new service model based on the wireline WFQ considering the characteristics of wireless networks.

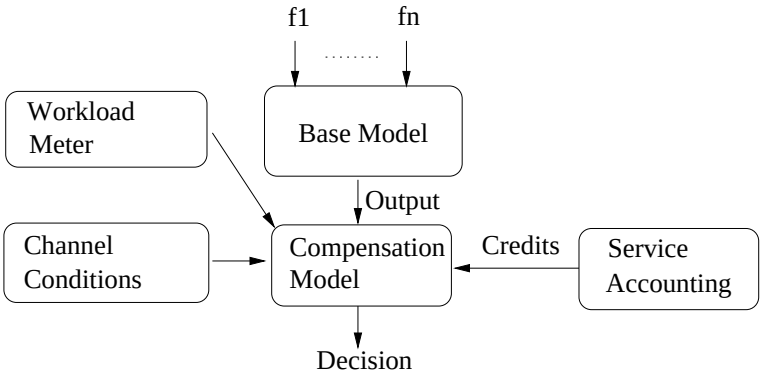


Fig. 3.1. The architecture of the ACFQ model

2. The model should consider service losses due to absence and channel errors. The absence compensation model targets at improving service differentiation under bursty data traffic, whereas the error compensation model is used to exploit channel conditions to increase the system throughput with fairness constraints.

3. The model should be simple, elegant, and easy to implement. The extent of the action of compensations should be flexibly controlled by the network administrator.

Based on these objectives, as shown in Figure 3.1, the ACFQ model consists of three parts: the base model that is based on WFQ to provides QoS to each flow, the accounting mechanism that tracks the service gain/loss due to absence or channel errors, and the compensation model that improves the service differentiation and provides fair service by letting the flows with service gain relinquish part of their services to the flows with service loss. The channel condition of each flow is used by the compensation model for error compensations and the workload meter is used for controlling the extent of absence compensation according to system workload.

3.3.2 The Base Model

The base model is used for providing QoS for each flow and acting as the reference system to account for the service loss or gain. By comparing the amount of received service with that of the reference system, a flow can be in three status: *losing, gaining, normal*. A flow is *losing* if it has received less service than it would have received in the reference system, *gaining* if it has received more, and *normal* if it has received the same amount. We choose the wireline WFQ model as the base model to assign a weight r_i to each flow f_i. We assume that all weights are normalized based on the smallest weight so that $r_i \geq 1$. The j^{th} packet of f_i, denoted by p_i^j, is assigned a start tag $S(p_i^j)$, and a finish tag $F(p_i^j)$ according to:

$$S(p_i^j) = \begin{cases} V(a_i^j), & f_i \text{ is absent} \\ F(p_i^{j-1}), & f_i \text{ is backlogged} \end{cases} \tag{3.1}$$

$$F(p_i^j) = S(p_i^j) + \frac{l_i^j}{r_i} \tag{3.2}$$

where a_i^j is the arrival time of p_i^j and $V(t)$ is the virtual time of the system. The relationship between the virtual time and real time conforms to:

$$\frac{dV(t)}{dt} = \frac{C(t)}{\sum_{j \in B(t)} r_j}$$

where $C(t)$ is the channel capacity at time t and $B(t)$ is the set of flows that are backlogged at time t. The scheduler picks up the packet for service in increasing order of the associated finish tag.

3.3.3 The Accounting Mechanisms

In ACFQ, we need to establish the service account for each flow to keep track of the service loss or gain due to absence or channel errors. To decouple the service gain/loss due to absence and channel error, we establish the accounting mechanism for them separately.

3.3.3.1 Error Accounting

When the channel condition of the serving flow is poor, continually serving the flow will decrease the system throughput. In order to alleviate the impact of the poor channel condition, it is better to swap the time slot from the flow suffering channel errors to another flow with clean channel and compensate the former flow later. With ACFQ, each flow f_i is associated with an error credit (denoted by EC_i), which is bounded by $[-EC_{max}, EC_{max}]$. Since we assume that the channel has multi-rate capability, the amount of data transmitted in different time slot may be different. Suppose a time slot is swapped from f_i to f_j and f_j transmits the packet of b bits, EC_i is decreased by b and EC_j is increased by b.

3.3.3.2 Absence Accounting

One simple way to accumulate the service loss because of absence is to calculate the finish tag for each packet as:

$$F(p_i^j) = \frac{L_i^j}{r_i} + F(p_i^{j-1})$$

with $F(p_i^0) = 0$. With this scheme, whenever a packet p_i^j is served, its finish tag is updated to the total normalized service provided to f_i. Therefore, f_i's service loss due to absence can be expressed by the normalized service difference between f_i and the flow being served. As discussed in [21, 49, 71], if the scheduler always serves the packet with the minimum finish tag, a backlogged flow with large finish tag can be starved for a long time when some new flows join the system. One modification mentioned in [73] is to replace the aforementioned $F(p_i^j)$ by $max\{F(p_i^j), a_i^j\}$, where a_i^j is the arrival time of p_i^j. However, as discussed in [21], since the real time a_i^j is not a true representation of the work progress in the system upon arrival of p_i^j, this solution still cannot avoid blocking backlogged flows with large normalized service.

Our approach: We present a new approach to calculate the service gain/loss due to absence. In our approach, each flow in the system is assumed to be virtually backlogged, and the scheduler needs to insert a *virtual packet* to a flow *whenever* it actually becomes absent. To support this mechanism, each flow is required to *join* or *leave* the system explicitly so that the scheduler can stop tracking service gain/loss due to absence of the

flow. Like a real packet, the virtual packet is assigned a finish tag as follows:

$$F(p_i^j) = \frac{L_{vp}}{r_i} + F(p_i^{j-1}) \tag{3.3}$$

where L_{vp} is the virtual packet length that is pre-specified by the system. Under ACFQ, each flow has an absence credit AC_i bounded by $[-AC_{max}, AC_{max}]$. We can keep track of the service loss due to absence as follows. When the scheduler picks up a virtual packet of f_i, and swaps the time slot to serve the real packet of another flow f_j, AC_i is decreased by 1 and AC_j is increased by 1.

Three important things need to be mentioned here: First, at any time, there is at most one virtual packet in the queue of each flow, and it must stay at the head of the queue. Second, there is no need to actually allocate a space for the virtual packet in the queue, and the queue only needs to keep the finish tag of the virtual packet. Third, the use of virtual packets does not have any side effect on the service tags of real packets since Eq (3.1) implies that the virtual packet staying in the queue will be discarded if a real packet arrives at the queue.

3.3.4 Workload Meter

When the workload of bursty data traffic[1] is very low, the number of backlogged flows competing the channel is small at a time. Due to this light workload, even with SPQ, the extent of service differentiation would be small since each flow has a high bandwidth. As a result, besides the absence accounting, it is also important to have the

[1]For conciseness, we refer the workload to the workload of bursty data traffic in the follows of the paper.

information about the workload of the system. When the system is lightly loaded, we can adjust the extent of absence compensation more aggressively to achieve a similar service differentiation to that under SPQ (Note that SPQ achieves the maximum service differentiation).

One simple way to use the number of the active flows as the workload of the system. However, the number of active flows cannot reflect the workload accurately because the bandwidth demand of each flow could be difficult to estimate due to bursty traffic. As a result, it is difficult to determine the workload solely based on the number of active flows, and then adjust the extent of absence compensation accordingly. To have a better way to calculate the workload, we design a measurement-based module called *workload meter* to obtain system workload. The basic idea is that we can make use of virtual packet to derive the workload. When the workload is low, the virtual packets of an absent flow would be scheduled more often than when the workload is high. It means that the absent flow loses it service due to absence more quickly when the workload is low, in which case the scheduler should compensate the flow's service loss due to absence more aggressively to achieve a good service differentiation. Based

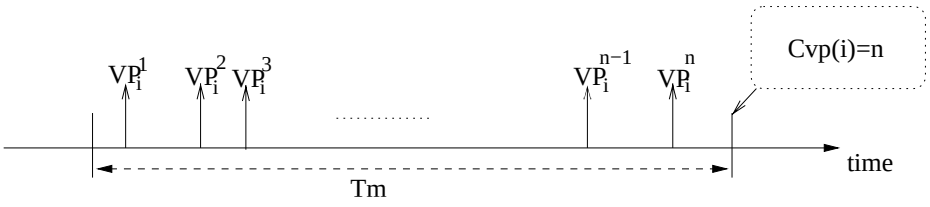


Fig. 3.2. An illustration of the workload meter

on this observation, the workload meter works as follows: as shown in Figure 3.2, f_i is associated with a counter, denoted by $C_{vp}(i)$, to record how many times a virtual packet has been picked by the scheduler over a predetermined measurement window T_m (in seconds). When a measurement window expires, the sample of workload, denoted by $SampleR_{vp}$, is calculated by:

$$SampleR_{vp} = \frac{\sum_{i \in \mathcal{K}} C_{vp}(i)}{|\mathcal{K}|}$$

where $\mathcal{K}$ is the set of active flows in the system. Meanwhile, the workload of the system, denoted by R_{vp}, is calculated as follows:

$$R_{vp} = (1 - x) * R_{vp} + x * SampleR_{vp}$$

where x is the smoothing factor and $SampleR_{vp}$ is the most recent measurement result. For all the flows with bursty data traffic, when the workload of the system is low, the virtual packet (if exists) of each flow would be served often and then $SampleR_{vp}$ is large. Otherwise, when the workload is high, the number of each virtual packet served during T_m decreases and then $SampleR_{vp}$ becomes small.

3.3.5 The Compensation Model

After knowing which flow has how many service loss or gain, the gaining flow should decide whether to use the allocated time slot or relinquish it to other losing flows. There are several options to relinquish service gains. One simple way is to relinquish all service gains, which means that the gaining flow f_i cannot transmit any data until

its credits become zero. However, if the credits are large, f_i may be starved. While bounding credits provides a partial solution, we present a more elegant solution. Our compensation model is based on the parameter which is called *relinquish probability*. When the scheduler selects a flow, it decides whether to let the flow relinquish the time slot or not according to the relinquish probability. We decouple the absence compensation and the error compensation by defining different ways to calculate the related relinquish probability.

Error Compensation: The goal of the error compensation model is to exploit the channel utilization to achieve high system throughput without loss of the long-term fairness of each flow. In order to efficiently exploit the channel utilization, the speed of the compensation should not be too fast. In other words, reducing the compensation speed may give the scheduler more opportunities to exploit the system throughput by serving each lagging flow at the time when it has good channel conditions. We use $\alpha_i = X_i/C$ to represent the channel condition of flow i , where X_i is the available transmission rate of f_i[2]. Under ACFQ, the channel condition and the error credit of each flow are the parameters of the error compensation model. The *error relinquish probability* of flow f_i, denoted by $rel_{err}(i)$ is defined as follows:

$$rel_{err}(i) = \begin{cases} max\{0, 0.5\frac{EC_i}{EC_{max}} + 0.625(1-\alpha_i)\}, \alpha_i > 0 \\ 1.0, \qquad\qquad\qquad \alpha_i = 0 \end{cases} \tag{3.4}$$

[2]Without loss of generality, we assume that α_i is no less than 0.2 when $\alpha_i > 0$ in this paper.

When the channel is clean (e.g. $\alpha = 1.0$), only the gaining flow has positive rel_{err}. On

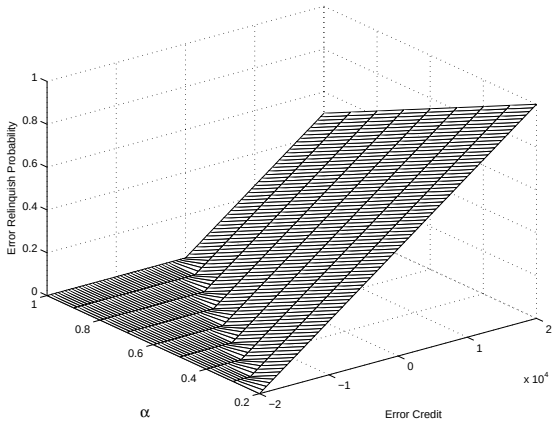


Fig. 3.3. The error relinquish probability as a function of the error credit and α

the other hand, when the channel quality is poor (e.g., $\alpha = 0.2$), the lagging flow may still relinquish the time slot, since serving the flow will significantly degrade the system throughput. However, the service delay is bounded since the error credit of the flow will eventually be sufficiently small to reduce the relinquish probability. Figure 3.3 gives the numerical results of rel_{err} as a function of the error credit and α, where EC_{max} is assumed to be 2×10^4 bits.

When a flow relinquishes the time slot, the time slot is used to serve another flow. Again, we need to consider the channel quality and the error credit of each flow to balance the tradeoff between system throughput and fairness. Under ACFQ, each flow

f_i is assigned an *error compensation value* val_i, which is defined by:

$$val_i = \begin{cases} -sign(EC_i)(\frac{EC_i}{EC_{max}})^2 + \alpha_i, & \alpha_i > 0 \\ -\infty, & \alpha_i = 0 \end{cases} \tag{3.5}$$

where $sign(x) = 1$ for $x \geq 0$, and $= -1$ for $x < 0$. Figure 3.4 shows an example of the

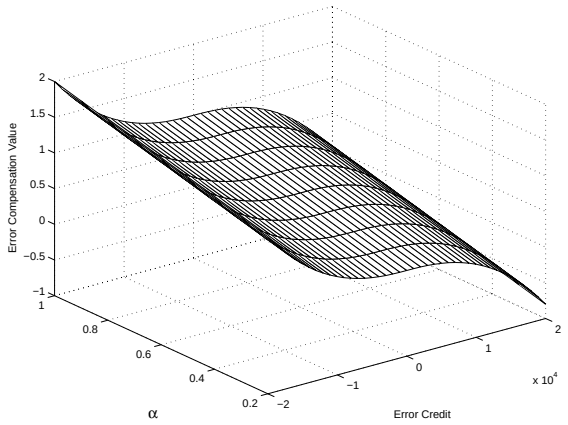


Fig. 3.4. The error compensation value as a function of the error credit and α

value as a function of error credit and α. Suppose flow f_i decides to relinquish a time slot, the candidate flow f_j to be compensated is selected as following:

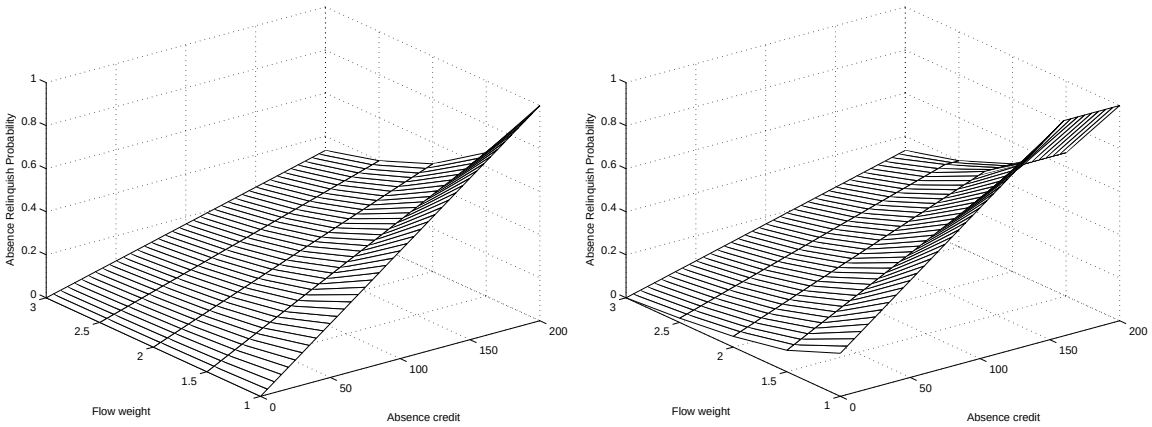
$$f_j = \underset{k \in \mathcal{B} \wedge val_k > val_i}{\arg\max} \{val_k\}; \tag{3.6}$$

where $\mathcal{B}$ is the set of backlogged flows. Note that if the candidate flow f_j does not exist, flow f_i will be served.

Absence Compensation: Suppose flow f_i is allocated a time slot, the scheduler first calculates the *absence relinquish probability* of f_i, denoted by $rel_{abs}(i)$, according to:

$$rel_{abs}(i) = \begin{cases} \frac{\min\{AC_i + R_{vp} * (r_{max} - r_i), AC_{max}\}}{AC_{max} * r_i}, & AC_i > 0 \\ 0, & AC_i \leq 0 \end{cases} \tag{3.7}$$

where r_{max} is the maximum weight. Unlike the calculation of error relinquish proba-



(a) $R_{vp} = 0$ (b) $R_{vp} = 20$

Fig. 3.5. The absence relinquish probability as a function of the absence credit and the flow weight

bility, we consider the absence credit, the weight of f_i and the workload to determine $rel_{abs}(i)$, since the goal of absence compensation is to improve the service differentiation. With the same amount of absence credit, high-weight gaining flows have smaller

relinquish probability than low-weight gaining flows. The factor $R_{vp} * (r_{max} - r_i)$ is used to force low-weight gaining flows relinquish their service gain faster when the workload is low. In Figure 3.5, with different R_{vp}, we give numerical results of the absence relinquish probability as a function of the absence credit and flow weight. As can be seen, the difference of the absence relinquish probability between high-weight flows and low-weight flows increases as R_{vp} increases. According to the obtained $rel_{abs}(i)$, the scheduler makes a decision as follows:

- **Case 1:** The head-of-line packet of f_i is a real packet. Depending on $rel_{abs}(i)$, it decides to allocate the time slot to serve f_i or let f_i relinquish the time slot to the other backlogged losing flows. If it decides to let f_i relinquish the time slot but no backlogged losing flow exists, the scheduler further examines whether to serve f_i or not according to $rel_{err}(i)$.

- **Case 2:** The head-of-line packet of f_i is a virtual packet, i.e., the queue of f_i is empty. Depending on $rel_{abs}(i)$, the time slot is allocated to a backlogged losing flow or the backlogged flow whose head-of-line packet has the minimum finish tag among all the real packets. If there is no backlogged flow, the time slot is wasted.

To select a backlogged flow for absence compensation, the scheduler picks up the candidate losing f_j with the minimum weighted absence credit as follows:

$$f_j = \underset{k \in \mathcal{B} \wedge AC_k < 0 \wedge \alpha_k > 0}{\arg\min} \{AC_k * r_k\} \tag{3.8}$$

With the same amount of credit, high-weight losing flows are compensated much faster than the low-weight losing flows, which is helpful to improve the service differentiation.

3.3.6 The ACFQ Scheduler

In the ACFQ service model, the scheduler serves the packets in the increasing order of their finish tags. After selecting flow f_i, the scheduler first decides if f_i should relinquish the time slot for absence compensation according to the rules of the absence compensation model. If f_i does not relinquish the time slot, the scheduler further checks the efficiency of serving f_i according to $rel_{err}(i)$ to balance the tradeoff between fairness and system throughput. There exists a tradeoff between the absence compensation and the error compensation. Suppose the time slot is swapped from f_i to f_j for absence compensation, and there is another backlogged flow f_k which has better channel condition than f_j. At this time, serving f_j favors the service differentiation, whereas the system throughput can be increased by serving f_k. Since we focuses on improving the service differentiation, we choose to serve f_j in this case. Thus, the scheduler does not further evaluate the channel condition of f_j (see line 7 in Figure 3.6).

When f_i is absent and the scheduler selects flow f_m that has the minimum finish tag among all the backlogged flows, the scheduler treats f_m as a new selected flow and examine whether to serve f_m or not according to the relinquish probabilities of f_m (see line 11 in Figure 3.6). Finally, it is possible that the time slot is swapped in the form of a chain; i.e., $f_i \rightarrow f_m \rightarrow f_j$. In this case, the service accounting should take place between f_i and f_j since f_j takes the service opportunity of f_i. Although the time slot is swapped from f_m to f_j because of the poor channel condition of f_m, since the service

Notations:
V**:** system virtual clock
F_i**:** the finish tag of the head-of-line packet of f_i
$\mathcal{K}$**:** the set of all flows
$\mathcal{B}$**:** the set of backlogged flows
j: the flow to be served
ac: the absence compensation indicator
ec: the error compensation indicator

schedule()
begin:
1. ac = **false**; ec = **false**; j = **NULL**; m = **NULL**;
   i = $\underset{k\in\mathcal{K}}{argmin}\{F_k\}$; /* initializations */
select:
2. **if** (f_i is backlogged)
3. **if** ($random(0,1) < rel_{abs}(i)$) { select f_j according to Eq (3.8);}
4. **if** (f_j not exists) $j = i$;
5. **else** ac= **true**;
   /* check whether to relinquish the slot due to error */
6. **if** ($ac == false \wedge (random(0,1) < rel_{err}(i))$ { select f_j according to Eq (3.6); }
7. **if** (f_j not exists) j=i;
8. **else** ec=**true**;
9. **else** /* f_i is absent */
10. **if** ($random(0,1) < rel_{abs}(i)$) { select f_j according to Eq (3.8); }
11. **if** (f_j exists) ac=**true**;
12. **else** $m = \arg\min_{k\in\mathcal{B}}\{F_k\}$;
13. **if** ($m \neq NULL$) { i=m; **goto select**; }
14. **if** (f_j not exists) {idle in the time slot; **goto begin**;}
15. $p = Q_j.deque()$;
16. **send**(p);
17. **if** ($m \neq NULL$) { i=m; ac = **true**; }
18. **if** ($ac == true$)
19. **if** (f_i is backlogged) $F_i = F_i + p.length/r_i$;
20. **else** $F_i = F_i + L_{vp}/r_i$;
21. $AC_i = AC_i - 1$; $AC_j = AC_j + 1$;
22. **else if** ($ec == true$)
23. { $F_i = F_i + p.length/r_i$;; $EC_i = EC_i - p.length$;$EC_j = EC_j + p.length$;
24. } **else if** ($Q_j.legnth > 0$)
25. update F_j according to Eq (3.2);
26. **else** $F_j = F_j + L_{vp}/r_j$;
27. **goto begin**;

Fig. 3.6. The ACFQ algorithm

opportunity of f_i is absence related, the service gain of f_j should be counted as the gain due to absence. This is why the absence indicator (ac) is forced to be true in line 20 in Figure 3.6.

It is worth to study the situation when f_i *leaves* the system with non-zero credit C_i, which can be AC_i or EC_i. In order to make sure that $\sum_{k\in\mathcal{K}} C_k = 0$ is always valid, we need to distribute C_i proportionally to other flows. For example, if f_i with $C_i > 0$ leaves the system, the credits of the losing flows should be increased proportionally. However, since credits are integer, we cannot achieve exact proportional distribution. Our solution is as follows. Suppose $\mathcal{L}$ denotes the set of the losing flows; i.e., $\mathcal{L} = \{l | l \in \mathcal{K} \wedge C_l < 0\}$. After a gaining flow f_i with $C_i > 0$ leaves the system, the credit of each flow in $\mathcal{L}$, denoted by f_j, is increased by $\lfloor \frac{C_j}{\sum_{l\in\mathcal{L}} C_l} C_i \rfloor$. Then, we get

$$\Delta = C_i - \sum_{j\in\mathcal{L}} \lfloor \frac{C_j}{\sum_{l\in\mathcal{L}} C_l} C_i \rfloor \tag{3.9}$$

If $\Delta > 0$, we randomly generate the subset of $\mathcal{L}$, denoted by $\mathcal{SC}$, with the cardinality of Δ. The credit of each flow in $\mathcal{SC}$ is increased by 1. As a result, the credit of f_j is increased by $\lfloor \frac{C_j}{\sum_{l\in\mathcal{L}} C_l} C_i \rfloor$ or $\lfloor \frac{C_j}{\sum_{k\in\mathcal{L}} C_k} C_i \rfloor + 1$. The correctness is based on the fact that $0 \leq C_i - \sum_{j\in\mathcal{L}} \lfloor \frac{C_j}{\sum_{l\in\mathcal{L}} C_l} C_i \rfloor < |\mathcal{L}|$ when $C_i > 0$. Similar approach can be applied when $C_i < 0$.

For the computational complexity of the ACFQ model, we observe that the selection of the candidate flow is at the cost of $O(log(n))$, which is feasible in many wireless networks that have a moderate number of flows per base station. The main computational overhead of ACFQ is divisions to compute the relinquish probabilities. According

to the scheduling scheme, to serve a packet, the scheduler needs to calculate the probability at most three times. For most current microprocessors (>1GHz), the cost is less than 150ns since one floating devision needs less than 50 cycles [18]. Thus, this delay would not be an issue for base stations.

3.4 Analysis of ACFQ

In this section, we analyze the properties of ACFQ. For simplicity, we assume all real packets have the same fixed size L_p and the channel is error-free.

LEMMA 4. *If f_i is continually absent over a time interval $[t_1, t_2]$ and at least one flow is backlogged over the time interval, its service credit decrement, denoted by ΔAC_i, follows:*

$$\Delta AC_i = \lfloor \frac{r_i(v_1 - v_2)}{L_{vp}} \rfloor \tag{3.10}$$

where $v_1 = v(t_1), v_2 = v(t_2)$.

Proof: If f_i is continuously absent and at least one flow is backlogged during $[v_1, v_2]$, f_i's aggregated service loss (in bits) is equal to: $r_i(v_2 - v_1)$. Since f_i only contains virtual packets during $[v_1, v_2]$, the accounted service loss is equal to $|\Delta AC_i L_{vp}|$. Considering ΔAC_i is a negative integer, and $r_i(v_1 - v_2) - L_{vp} < \Delta AC_i L_{vp} \leq r_i(v_1 - v_2)$, the Lemma holds. □

Lemma 4 shows the absence service account of an absent flow during a time interval $[t_1, t_2]$.

LEMMA 5. *Consider a gaining flow f_i over a time interval $[t_1, t_2]$. Suppose $AC_i(t_1) = AC_0$,, where $AC_i(t_1)$ is the current absence credit of f_i at t_1. Suppose $R_{vp}(t_1) = R_0$ where $R_{vp}(t_1)$ is the workload measurement at t_1, and $R_{vp}^{min} \leq R_{vp}(t) \leq R_{vp}^{max}$ for any $t \in [t_1, t_2]$. Assume there always exists another flow which can take the compensation slot whenever f_i relinquishes the slot during $[t_1, t_2]$. Then, for any $t \in [t_1, t_2]$, the expected value of credit $AC_i(t)$, denoted by $E(AC_i(t))$, is bounded by:*

$$(AC_0 + \frac{AC_{max} r_i R_0 (r_{max} - r_i)}{C}) exp(-\frac{C}{AC_{max} r_i}(t - t_1)) - \frac{AC_{max} R_{vp}^{max} (r_{max} - r_i)}{C}$$
$$< E(AC_i(t)) \leq \max\{0, (AC_0 + \frac{AC_{max} r_i R_0 (r_{max} - r_i)}{\phi_i C}) exp(-\frac{\phi_i C}{AC_{max} r_i}(t - t_1))$$
$$- \frac{AC_{max} r_i R_{vp}^{min} (r_{max} - r_i)}{\phi_i C}\}$$

where $\phi_i = \frac{r_i}{\sum_{j \in \mathcal{K}} r_j}$, $\mathcal{K}$ is the set of all flows, C is the link capacity.

Proof: At time $t \in [t_1, t_2]$, suppose f_i has the data rate $\hat{r_i}(t)$ (in bps) under the FFQ model. As described in the absence compensation model, the expected *actual* data rate of f_i, denoted by $E(\hat{r_i}(t))$, is adjusted as: $E(\hat{r_i}(t)) = \hat{r_i}(t)(1 - \min\{E(AC_i(t)) + R_{vp}(t)(r_{max} - r_i), AC_{max}\}/(AC_{max} r_i))$. Without loss of correctness, we assume that $E(AC_i(t)) + R_{vp}^{max}(r_{max} - r_i) < AC_{max}$. Thus, the expected rate of service compensation for f_i follows:

$$\frac{dE(AC_i(t))}{dt} = -\frac{E(AC_i(t) + R_{vp}(t)(r_{max} - r_i))\hat{r_i}(t)}{AC_{max} r_i}$$

Since $\phi_i C \leq \hat{r}_i(t) < C$, where $\phi_i = \frac{r_i}{\sum_{j \in \mathcal{K}} r_j}$, and $R_{vp}^{min} \leq R_{vp}(t) \leq R_{vp}^{max}$, we have:

$$-\frac{(E(AC_i(t)) + R_{vp}^{max}(t)(r_{max} - r_i)))C}{AC_{max} r_i} < \frac{dE(AC_i(t))}{dt}$$
$$\leq -\frac{(E(AC_i(t)) + R_{vp}^{min}(t)(r_{max} - r_i))\phi_i C}{AC_{max} r_i} \tag{3.11}$$

By solving the differential inequation in form of $\frac{dy}{dx} + ay > b$ where a,b are $\frac{C}{AC_{max} r_i}$ and $\frac{R_{vp}^{max}(r_{max} - r_i)C}{AC_{max} r_i}$, and by integrating the resulting function over $[t_1, t]$, we arrive at:

$$E(AC_i(t)) > (AC_0 + \frac{AC_{max} r_i R_0 (r_{max} - r_i)}{C}) exp(-\frac{C}{AC_{max} r_i}(t - t_1))$$
$$-\frac{AC_{max} R_{vp}^{max}(r_{max} - r_i)}{C} \tag{3.12}$$

Similarly, with the fact that $E(AC_i(t) \geq 0)$, we have:

$$E(AC_i(t)) \leq \quad \max\{0, (AC_0 + \frac{AC_{max} r_i R_0 (r_{max} - r_i)}{\phi_i C}) exp(-\frac{\phi_i C}{AC_{max} r_i}(t - t_1))$$
$$-\frac{AC_{max} r_i R_{vp}^{min}(r_{max} - r_i)}{\phi_i C}\} \tag{3.13}$$

Combining Eq (3.12) and Eq (3.13), we conclude the proof. □

Lemma 5 shows the graceful absence compensation of a leading flow.

THEOREM 3. *Consider a gaining flow f_i over a time interval $[t_1, t_2]$. Assume there always exists another flow which can take the compensation slot whenever f_i relinquishes the slot during $[t_1, t_2]$. Then, the expected finish time when f_i relinquishes all its service*

gains, denoted by $E(t^i_{fin})$, is bounded by:

$$t_1 + \frac{AC_{max} r_i}{C} ln \frac{AC_0 + \frac{AC_{max} r_i R_0 (r_{max} - r_i)}{C}}{1 + \frac{AC_{max} R^{max}_{vp} (r_{max} - r_i)}{C}} < E(t^i_{fin}) \leq$$

$$t_1 + \frac{AC_{max} r_i}{\phi_i C} ln \frac{AC_0 + \frac{AC_{max} r_i R_0 (r_{max} - r_i)}{\phi_i C}}{1 + \frac{AC_{max} r_i R^{min}_{vp} (r_{max} - r_i)}{\phi_i C}}$$

Proof: According to the calculation of the absence relinquish probability, $rel_{abs}(i)$ becomes zero when $AC_i \leq 0$. Since AC_i is a discrete variable, f_i does not have service gain when $AC_i < 1$. Therefore, by applying $E(AC_i) = 1$ to Lemma 5, we get:

$$(AC_0 + \frac{AC_{max} r_i R_0 (r_{max} - r_i)}{C}) exp(-\frac{C}{AC_{max} r_i}(t^i_{fin} - t_1)) - \frac{AC_{max} R^{max}_{vp} (r_{max} - r_i)}{C} < 1$$

$$\leq (AC_0 + \frac{AC_{max} r_i R_0 (r_{max} - r_i)}{\phi_i C}) exp(-\frac{\phi_i C}{AC_{max} r_i}(t - t_1)) - \frac{AC_{max} r_i R^{min}_{vp} (r_{max} - r_i)}{\phi_i C}$$

By solving the above inequations, we conclude the proof. □

Theorem 3 shows the expected relinquish speed for a backlogged gaining flow during a time interval $[t_1, t_2]$. As we can see, given AC_0, R_0, R^{max}_{vp} and R^{min}_{vp}, the relinquish speed is reversely proportional to the flow weight, which mean low-weight flows would relinquish their service gains faster than high-weight flows.

THEOREM 4. *Consider a gaining flow f_i over a time interval $[t_1, t_2]$. Assume its queue is continually backlogged, and there always exists another flow which can take the compensation slot whenever f_i relinquishes the slot during $[t_1, t_2]$. Then, its expected aggregated*

service, denoted by $E(W_i(t_1,t_2))$, is bounded by:

$$E(W_i(t_1,t_2)) > \phi_i C(t_2-t_1) - \phi_i|\mathcal{K}|L_p - L_p - (AC_0 -$$
$$(AC_0 + \frac{AC_{max}r_iR_0(r_{max}-r_i)}{C})exp(-\frac{C}{AC_{max}r_i}(t-t_1)) + \frac{AC_{max}R_{vp}^{max}(r_{max}-r_i)}{C})L_p$$

Proof: For a continuously backlogged gaining flow, the number of slots relinquished is equal to $AC_0 - AC_i(t_2)$. Hence, the expected relinquished service of f_i during $[t_1,t_2]$, denoted by $E(W_i^{rel}(t_1,t_2))$ is equal to $(AC_0 - E(AC_i(t_2)))L_p$. With Lemma 5, we have:

$$E(W_i^{rel}(t_1,t_2)) > (AC_0 - (AC_0 + \frac{AC_{max}r_iR_0(r_{max}-r_i)}{C})exp(-\frac{C}{AC_{max}r_i}(t-t_1)) - \frac{AC_{max}R_{vp}^{max}(r_{max}-r_i)}{C})L_p$$

Similar to the proof of Theorem 4.2 in [45], if no compensation occurs during $[t_1,t_2]$, the aggregated service of f_i during the period follows:

$$W_i(t_1,t_2) \geq \phi_i C(t_2-t_1) - \phi_i|\mathcal{K}|L_p - L_p \tag{3.14}$$

Actually, the expected aggregated service of f_i during $[t_1,t_2]$ is equal to $W_i(t_1,t_2) - E(W_i^{rel}(t_1,-t_2))$. With Ineq (3.14) and Ineq (3.14), we conclude the proof. □

Theorem 4 shows the expected throughput guarantee for a backlogged gaining flow during a time interval $[t_1,t_2]$.

THEOREM 5. *Consider a losing flow f_i over a time interval $[t_1,t_2]$.Assume that all the flows are continuously backlogged during $[t_1,t_2]$ and $\forall(k \in \mathcal{K})AC_k(t_1) = AC_k^0$. Its*

expected aggregated service is bounded by:

$$E(W_i(t_1,t_2)) \geq \phi_i C(t_2-t_1) - \phi_i|\mathcal{K}|L_p - L_p + SL_p \tag{3.15}$$

where S is calculated by:

$$\begin{aligned} S \;=\; & \underset{n=0,1,2\ldots}{arg\ max}\{\sum_{m=0}^{n}\sum_{l\in\mathcal{L}\wedge l\neq i}\lceil AC_i^m\frac{r_i}{r_l} - AC_l^m\rceil^+ + n \leq \\ & \sum_{j\in\mathcal{G}}(AC_j^0 - \max\{0,(AC_j^0 + \frac{AC_{max}r_iR_0(r_{max}-r_i)}{\phi_i C})exp(-\frac{\phi_i C}{AC_{max}r_i}(t-t_1)) \\ & -\frac{AC_{max}r_iR_{vp}^{min}(r_{max}-r_i)}{\phi_i C}\})\} \end{aligned} \tag{3.16}$$

where $\lceil x\rceil^+$ is defined as: $\max(0,\lceil x\rceil)$, $\mathcal{L}=\{l|l\in\mathcal{K}\wedge AC_l<0\}$, $\mathcal{G}=\{j|j\in\mathcal{K}\wedge AC_j>0\}$. For each flow f_l in $\mathcal{L}$, AC_l^m is defined as:

$$AC_l^m = \begin{cases} AC_l^m = AC_l^{m-1} + \lceil AC_l^m\frac{r_i}{r_l} - AC_l^{m-1}\rceil^+, m>0\wedge l\neq i \\ AC_l^m = AC_k^{m-1}+1,\ \ m>0\wedge l=i \end{cases} \tag{3.17}$$

Proof: With Lemma 5, we can see that the expected total number of slots relinquished by all the gaining flows is no less than: $\sum_{j\in\mathcal{G}}(AC_j^0 - \max\{0,(AC_j^0+\frac{AC_{max}r_iR_0(r_{max}-r_i)}{\phi_i C})$ $exp(-\frac{\phi_i C}{AC_{max}r_i}(t-t_1)) - \frac{AC_{max}r_iR_{vp}^{min}(r_{max}-r_i)}{\phi_i C}\})$. According to Eq (3.8), at time t_1, before the losing flow f_i can be compensated with a time slot, the number of relinquished time slots that have passed is: $\sum_{l\in\mathcal{L}\wedge l\neq i}\lceil AC_{i,0}\frac{r_i}{r_l} - AC_{l,0}\rceil^+$. Following the same reasoning, after f_i gets its m^{th} relinquished time slot, the absence credit of each losing flow

f_l can be computed as:

$$AC_l^m = \begin{cases} AC_l^{m-1} + 1 & l = i \\ AC_l^{m-1} + \lceil AC_i^{m-1} \frac{r_i}{r_j} - AC_l^{m-1} \rceil^+ & l \neq i \end{cases}$$

Hence, the total number of relinquished slots that f_i can get during $[t_1, t_2]$ can be computed by Eq (3.16). Since f_i gets two sources of service: the allocated service according to r_i and the compensated service from the gaining flows, with Ineq (3.14) and Eq (3.16), we conclude the proof. □

Theorem 5 shows the expected throughput guarantee for a backlogged losing flow during a time period $[t_1, t_2]$.

3.5 Performance Evaluation

In this section, we evaluate the performance of the proposed ACFQ and demonstrate its effectiveness by comparing to strict priority queuing (SPQ) [18] and weighted fair queuing (WFQ) [17]. The simulation is based on ns-2 [27]. Following the results of [66], when the channel is error-prone, a five-state Markov chain is used to emulate the process of channel conditions with fast fading. As shown in Figure 3.7, the marked line

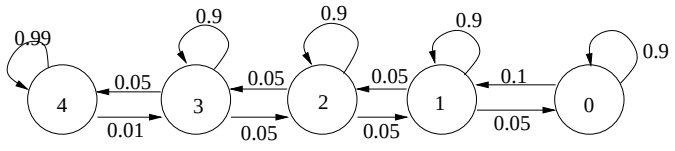


Fig. 3.7. The error-prone channel model

or curve shows the transition probability from one state to another. The wireless link is based on TDMA. The transmission rate in state 4 is equal to the channel capacity which is assumed to be $384Kbps$, and is zero when the channel condition is in state 0. Following the standard of EDGE [64], each time slot is $2.5ms$, which can be used to transmit 112, 74, 56, 44 bytes of data (not including the header) in state 4, 3, 2, 1 respectively.

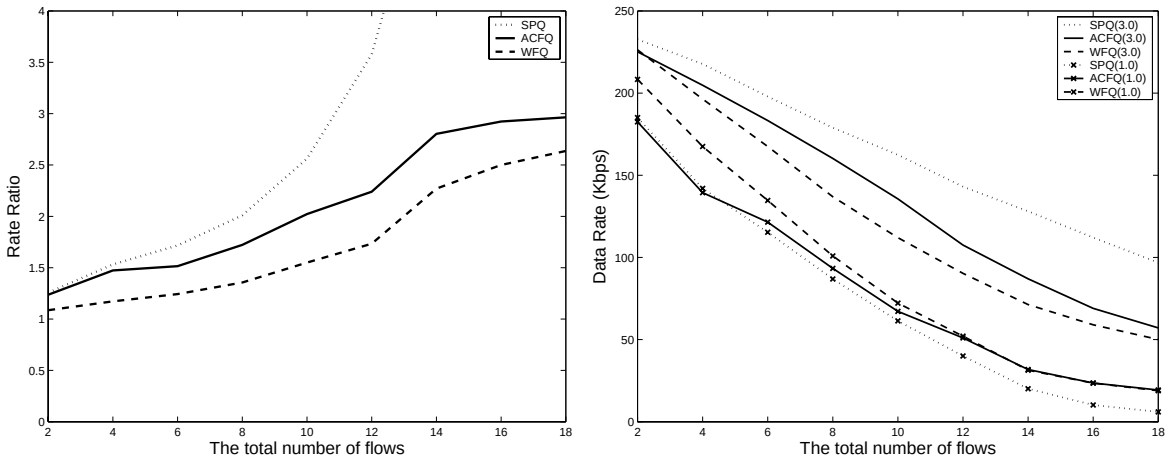
Component	Model	PDF	Parameters
File Sizes - Body	Lognomal	$p(x) = \frac{1}{x\sigma\sqrt[2]{2\pi}} e^{-(ln(x)-\mu)^2/2\sigma^2}$	$\mu = 7.881; \sigma = 1.339$
File Sizes - Tail	Pareto	$p(x) = \alpha k^{\alpha} x^{-(\alpha+1)}$	$k = 34102; \alpha = 1.177;$ $max = 100Kbytes$
Embedded References	Pareto	$p(x) = \alpha k^{\alpha} x^{-(\alpha+1)}$	$k = 2; \alpha = 1.245;$ $max = 30$
OFF Times	Pareto	$p(x) = \alpha k^{\alpha} x^{-(\alpha+1)}$	$k = 1; \alpha = 1.4;$ $max = 300sec$

Table 3.1. Distributions and parameters of the traffic model

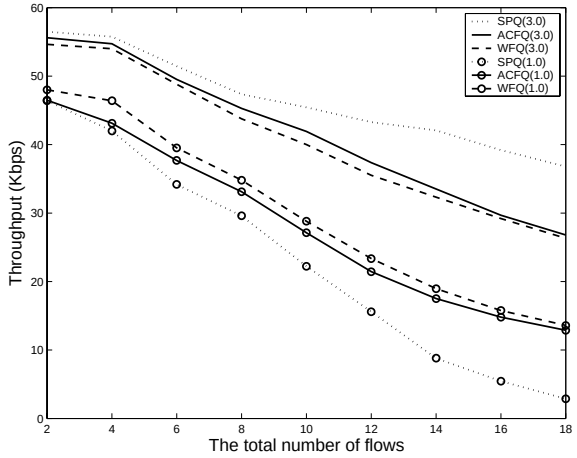
Since we focus on improving service differentiation under bursty data traffic, Web browsing is used as a case study. In order to reduce the overhead of TCP hand-shaking and slow start, we assume that all connections are persistent [37]. By setting different seeds, we generate different traffic trace for each flow. Similar to [32], the parameters of the data traffic model are shown in table 3.1. In most scenarios, for simplicity, we assume that there are two kinds of flows in the system: high-weight flows with weight of 3.0 and low-weight flows with weight of 1.0, where the number of high-weight flows

and low-weight flows are equal. We further study the the case in which more than two service classes are provided. The total simulation time is 1000 seconds. We set the default AC_{max} and EC_{max} to be 200 and 20 KB respectively. We set the default virtual packet length (L_{vp}) to 1000 bytes, and further evaluate its impacts in Section 3.5.3. The window size of the workload meter T_m is $100ms$ and the smoothing factor x is set to be 0.8.

We evaluate the system performance with the following three metrics: the *average data rate*, the *average throughput*, and the *rate ratio*. The data rate of a file is calculated by dividing its file size with the time used to deliver the entire file, which shows how fast the file is delivered. The throughput is obtained by dividing the total amount of data (in bits) delivered by the simulation time. Since there may be multiple flows with the same weight and each flow may have different data rate, we use the *average data rate* to measure the application-level QoS. Similarly we use the *average throughput* instead of throughput for each flow. For simplicity, we will remove *average* from these terms in the follows of the paper. The *rate ratio* is the ratio of the data rate of the high-weight flows to that of the low-weight flows, and is used as the metrics of the service differentiation. The evaluation considers five scenarios. In the first scenario, the channel is error-free. In the second scenario, the performance is evaluated in the error-prone channel. In the third scenario, we examine the impacts of virtual packet length on the performance of ACFQ.



(a) Rate ratio (b) Data rate



(c) Throughput

Fig. 3.8. Performance comparisons under three service models

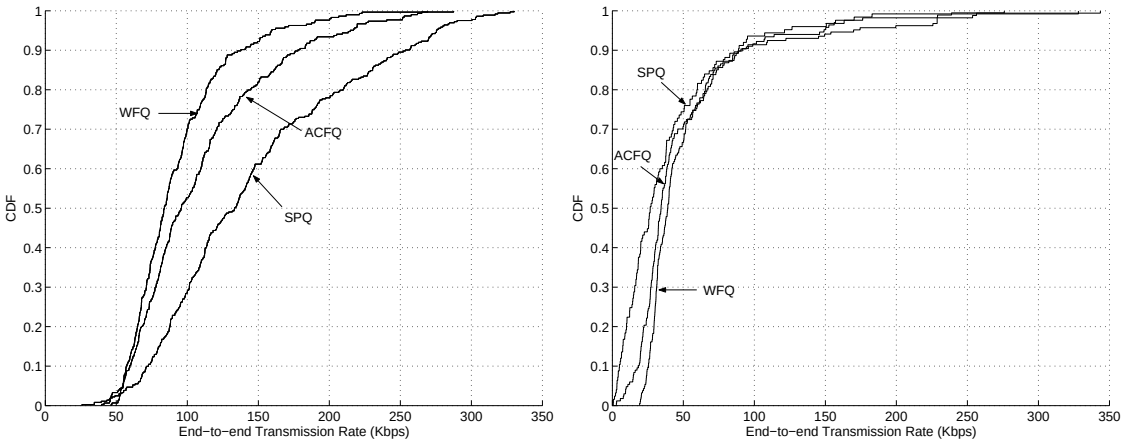
3.5.1 Scenario 1: Error-free Channel

We evaluate the performance of these three service models under different workload, which is represented by the total number of flows, denoted by $N(2 \leq N \leq 18)$. As shown in Figure 3.8 (a), the rate ratio under SPQ is always higher than that under WFQ and ACFQ. As N is larger than 12, the rate ratio of SPQ becomes much larger than 3.0, which is the ideal rate ratio because the weight ratio between the high-weight flows and the low-weight flows is 3.0. It indicates that the high-weight flows get too much service that is not proportional to their weights. This is due to the fact that SPQ gives high-weight flows exclusive priority over the low-weight flows, and cannot provide QoS for the low-weight flows when the traffic of the high-weight flows is heavy.

From Figure 3.8 (a), we can see that the rate ratio of ACFQ is much higher than that of WFQ, especially when N is larger than 6, because ACFQ performs service compensation due to absence, but WFQ does not. In addition, the high-weight flows under ACFQ get preferential treatment under the compensation model. Compared to the low-weight flows, the high-weight gaining flows have slower relinquish rate and the service loss of the high-weight flows can be quickly compensated. As a result, the high-weight flows can transmit their packets at higher speed. Since the workload of the system is also considered as a factor in the absence compensation model, when N is small (i.e. $N = 2, 4, 6$), the data ratio of ACFQ is quite close to that of SPQ. Since SPQ achieves the maximum service differentiation, ACFQ provides a nearly optimal service differentiation when system is lightly loaded. As the workload increase, the low-weight gaining flows relinquish the service gain due to absence less aggressively, which can be shown by Figure

3.8(b). As can be seen, when N increases, the data rate of low-weight flows under ACFQ becomes close to that under WFQ.

From Figure 3.8 (b) and Figure 3.8 (c), we can see that the data rate and the throughput of the high-weight flows under ACFQ are higher than that under WFQ in most cases. When the workload is very heavy (e.g. $N = 18$), the progress of the virtual clock becomes slower (*i.e.,* for the same interval $[t_1, t_2]$, $v(t_2) - v(t_1)$ is smaller). According to Lemma 1, the absence credit of an absent flow decreases slower. Correspondingly, service compensations occur less often and the total number of time slots that has to be relinquished reduces. This explains why the data rate difference of the high-weight flows between ACFQ and WFQ when $N = 18$ is smaller than that when $N = 10$.



(a) high-weight flows (b) low-weight flows

Fig. 3.9. The CDF of the transmission rate ($N = 12$)

In order to show the difference of these three models clearly, we draw the cumulative distribution function (CDF) of the data rate when $N = 12$. As shown in Figure 3.9, more than 70% of the packets in the high-weight flows are transmitted at a data rate of less than 100 Kbps under WFQ. By compensating the service loss due to absence, the percentage is reduced to be less than 52% under ACFQ, which means that more than 28% of the packets are transmitted faster. Due to exclusive priority, only 30% of the packets in high-weight flows are transmitted less than 100Kbps under SPQ. On the other hand, the CDF of the data rate for low-weight flows under ACFQ is quite similar to that under WFQ. Compared to the high-weight flows, the low-weight gaining flows have higher relinquish rate and the low-weight losing flows have less opportunities to be compensated. Even though the low-weight flows are also compensated for their service loss, they cannot get as much benefit as the high-weight flows. As shown in Figure 3.8 (b), the data rate difference of low-weight flows between ACFQ and WFQ is very small. This also shows that, in the long run, the action of absence compensations does not incur much negative impact on the QoS to the low-weight flows. Since ACFQ uses WFQ as the base model, each flow still has QoS provision. This explains why the rate ratio under ACFQ is bounded by 3.0 even when the workload is very heavy as shown in Figure 3.8 (a).

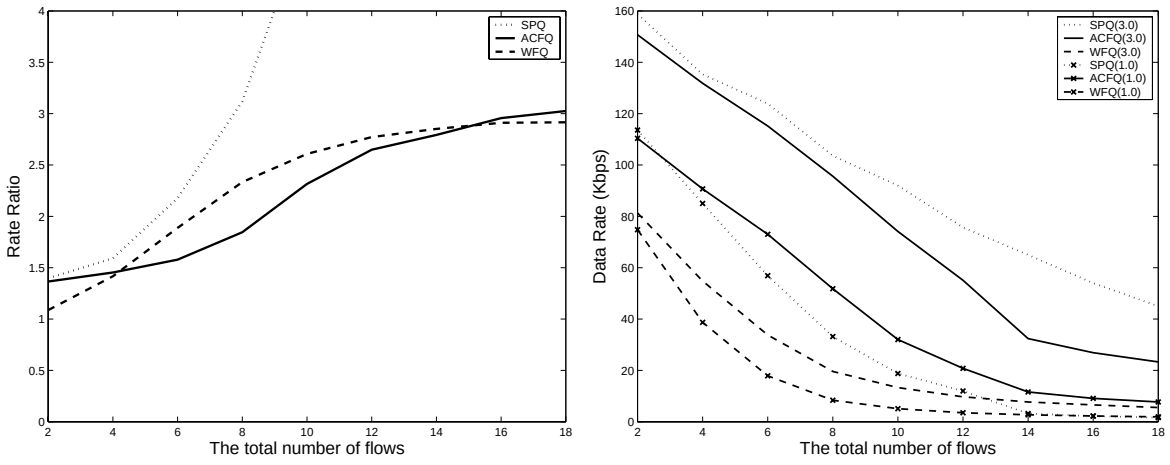
In summary, SPQ can provide maximum service differentiation but cannot provide QoS to the low-weight flows. In contrast, the rate ratio under ACFQ and WFQ is bounded, which shows that ACFQ and WFQ can provide QoS for each flow. Compared with WFQ, ACFQ can significantly improve the service differentiation under bursty data traffic by compensating service loss due to absence.

3.5.2 Scenario 2: Error-prone Channel

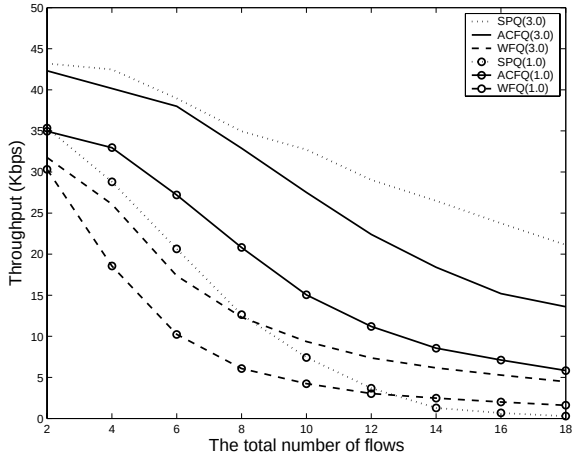
In this scenario, we compare the performance of these service models when the channel is error-prone. We assume that each flow has time-varying channel condition. As shown in Figure 3.10 (a), as N increases, the rate ratio of SPQ grows out of the bound (=3.0) since the data rate of the low-weight flows drops much faster than that of the high-weight flows. The reason has been explained in Scenario 1 and is still valid here. From Figure 3.10 (c), we can see that the system throughput under SPQ is higher than that under WFQ. Since SPQ serves the flows with the same weight in the round robin way, each flow with the same weight can only be served once in each round. This can alleviate the impact of channel errors since the scheduler will skip over the flow when the flow cannot transmit data due to channel errors.

As shown in Figure 3.10 (b) and (c), the performance of WFQ is the worst when channel is error-prone. This is due to the fact that WFQ only tries to achieve the short-term throughput fairness so that the scheduler will continuously serve the flow until the normalized throughput of the flow is no less than that of other flows. As a result, the flow with poor channel condition will take much longer time to get a certain amount of normalized service than the flow with good channel condition. Since we assume that all flows have error-prone channels, the data rates of high-weight flows and low-weight flows drop very fast as N increases. However, WFQ still provides fair service to each flow. From Figure 3.10 (a), the data ratio of WFQ is always bounded by 3.0.

Compared to SPQ and WFQ, ACFQ can improve service differentiation without losing too much system throughput. As shown in Figure 3.10 (a), ACFQ still has similar



(a) Rate ratio (b) Data rate



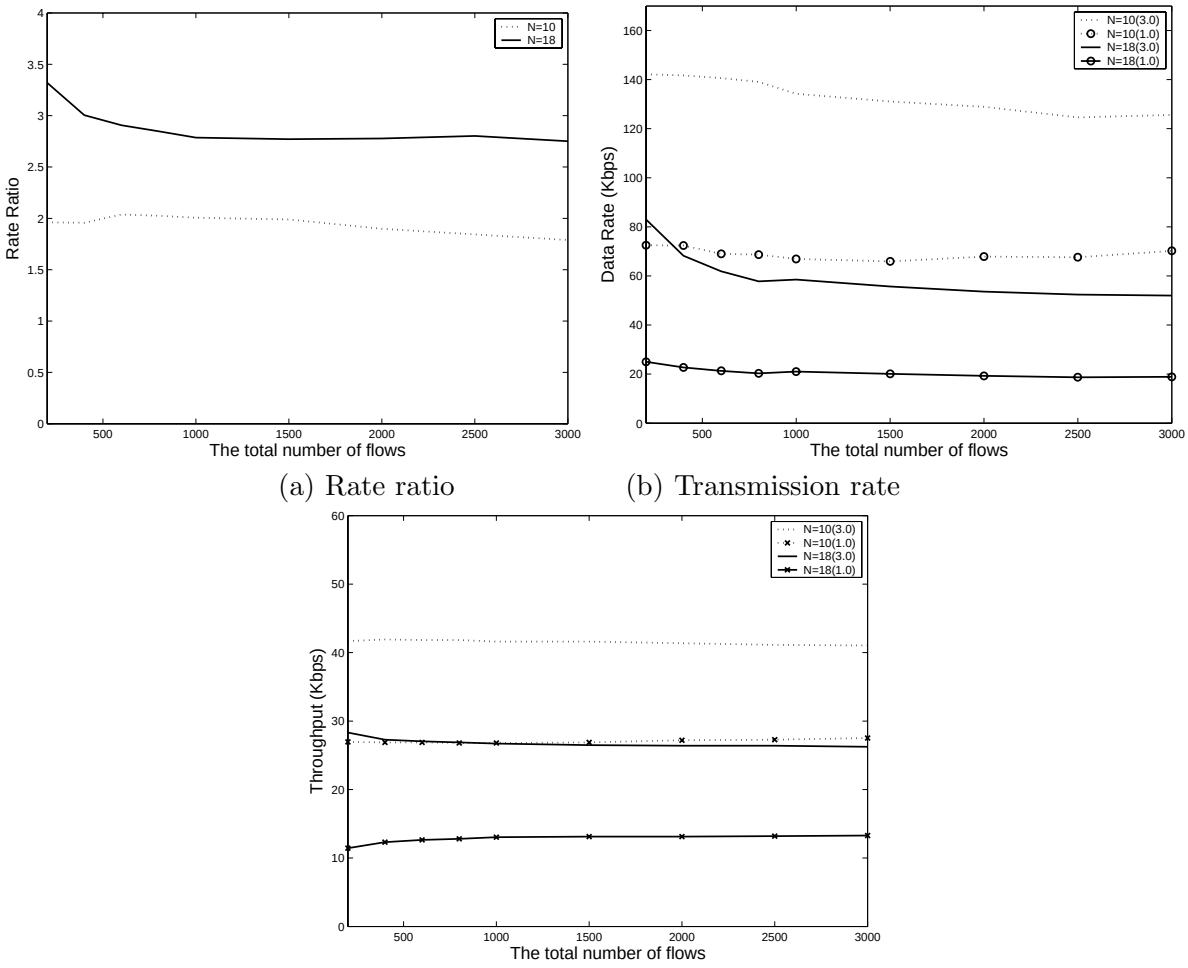
(c) Throughput

Fig. 3.10. The performance comparisons with channel errors

rate ratio as in Scenario 1. This shows that the absence compensation model works well in the error-prone channel. From Figure 3.10 (c), we can see that, when $N \geq 4$, the sum of the throughput of each flow under ACFQ is much higher than that under WFQ, which proves the effectiveness of the error compensation model of ACFQ. With the fairness constraint, the ACFQ scheduler tries to serve the flow which has the best channel condition so that the system throughput can be increased.

3.5.3 Scenario 3: Impact of the Virtual Packet Length

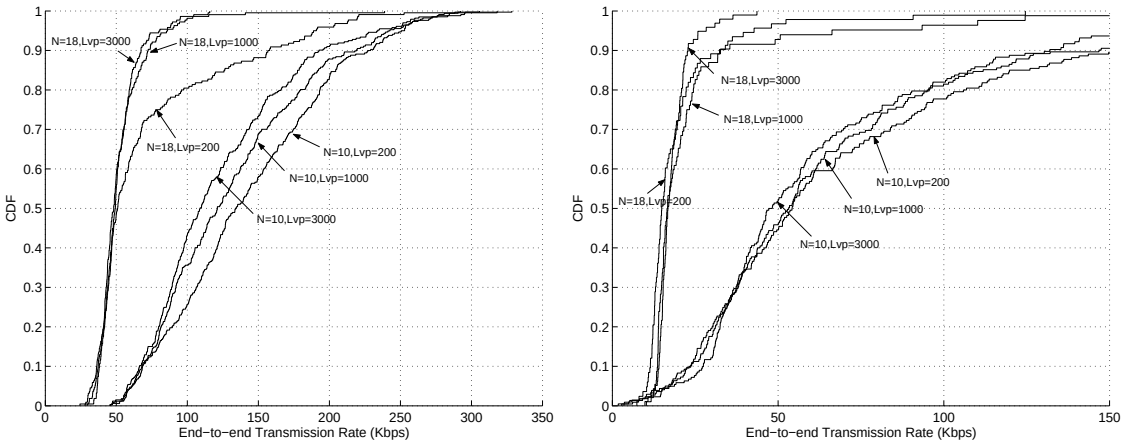
In this scenario, we investigate the impacts of the virtual packet length (L_{vp}) on the performance of ACFQ. We consider two cases: $N = 10$ and $N = 18$, and assume the channel is error-free. As shown in Figure 3.11 (a), in both cases, the rate ratio slightly drops when L_{vp} increases. However, as shown in Figure 3.11 (b), the date rate of high-weight flows and low-weight flows (lines $N = 10(3.0)$ and $N = 10(1.0)$) increases when L_{vp} drops. For example, when L_{vp} drops from 1000 to 200, the data rate of high-weight flows increases from 135 Kbps to 143 Kbps and the rate of low-weight flows increases from 67 Kbps to 73 Kbps. During the same virtual time interval, when L_{vp} decreases, according to Lemma 1, the absence credit increases. As a result, both high-weight and low-weight flows with service loss can get more chances for service compensations. In particular, because the workload is not heavy ($N = 10$), there are not many high-weight flows competing the service compensation with low-weight flows. Therefore, low-weight flow can be compensated most of the time, and has relatively high data rate. This can be verified from Figure 3.12 (b). As can be seen, when $L_{vp} = 3000$, less than 15% of the packets in low-weight flows are transmitted at a rate of more than 100 Kbps under



(a) Rate ratio (b) Transmission rate

(c) Throughput

Fig. 3.11. Performance of ACFQ with different virtual packet length



(a) high-weight flows (b) low-weight flows

Fig. 3.12. The CDF of the transmission rate

WFQ. When L_{vp} is reduced to 200, more than 22% of the packets are transmitted at the rate of more than 100 Kbps.

Things are different when the workload is heavy ($N = 18$). As shown in Figure 3.11 (a), the rate ratio increases out of bound as L_{vp} decreases. For example, when L_{vp} drops from 2000 to 200, the rate ratio increases from 2.9 to 3.3. This can be explained by Figure 3.11 (b). As can be seen from lines ($N = 18(3.0)$) and ($N = 18(1.0)$), when L_{vp} drops from 2000 to 200, the data rate of high-weight flows increases from 56 Kbps to 83 Kbps, but the data rate of low-weight flows is almost unchanged. This can be further explained by Figure 3.12. In Figure 3.12 (b), when $N = 18$, changing the virtual packet length almost does not affect the data rate of low-weight flows. However, as shown in Figure 3.12 (a), when $L_{vp} = 3000$, more than 99% of the packets in high-weight flows are transmitted at the rate of less than 100 Kbps; when L_{vp} drops to 200, this percentage is reduced to be less than 82%. This can be explained as follows. When L_{vp}

decreases, according to Lemma 1, the absence credit increases. As a result, the number of relinquished time slots for absence compensation increases. However, in contrast to the case when $N = 10$, the low-weight flow is competed with many high-weight flows for service compensations, and cannot get compensations fast enough to increase its transmission rate . On the other hand, most of the increased service compensations (through reduced L_{vp}) are used to compensate high-weight flows. As a result, the data rate of high-weight flows increases when L_{vp} decreases.

3.6 Related Work

In [32], Jiang *et. al.* studied the problem of providing multiple service classes for bursty data traffic in cellular networks. Assuming the channel is error-free, they studied the performance of WFQ in providing multiple service classes for typical Internet users in a cellular data network. They found that WFQ can provide differentiated services only to a limited extent due to the burstiness of the data traffic. They also investigated several factors such as propagation delay, persistent TCP connections, and distribution of users, and found that these factors only have little impact on the service differentiation under WFQ. By increasing the weight assigned to high-weight flows, the service differentiation can be improved. However, high-weight flows get too much service when the system is heavily loaded. Our work was stimulated by their work, and we proposed the ACFQ service model to improve service differentiation for bursty data traffic.

In order to provide fairness to the flows with service loss due to channel errors, a channel-condition independent fair model [47] is proposed. Under this model, each leading flow reserves a minimal fraction of service so that the degradation of service for

leading flows is graceful. In [45], a different compensation model was proposed. The leading flow achieves graceful service degradation by dynamically adjusting the amount of compensation service based on the service credit. ACFQ uses some similar ideas to [45, 47]. However, ACFQ focuses on improving the service differentiation under bursty traffic with QoS provision. It is different from [45, 47] in the following aspects. First, [45, 47] are based on a two-state channel model, whereas ACFQ considers the channel with the multi-rate capability. Second, ACFQ considers not only the error compensation, but also the the absence compensation. The absence compensation model of ACFQ provides preferential treatment to high-weight flows; that is, with the same amount of credit, high-weight gaining flows have smaller relinquish probability, and high-weight losing flows have higher priority to be compensated.

In [18], Dovrolis *el al.* proposed proportional differentiation service models which guarantee the ratio of packet delay difference between classes within the Differentiated Service architecture. They proposed three models to achieve relative QoS among classes: the proportional average delay scheduling, the waiting time priority scheduling, and the hybrid model, which combines the other two. The ACFQ acts somewhat similar to their hybrid model, but our work is different from theirs from the following aspects: First, ACFQ is based on WFQ and targets at improving the service differentiation with absolute QoS provision (in terms of throughput, delay, and long-term fairness) for each flow. Second, ACFQ also considers the time-varying channel condition, which is an important issue in wireless networks.

Chapter 4

The Relay-Enabled IEEE 802.11 MAC Protocols

4.1 Background

IEEE 802.11 has physical-layer multi-rate capability [30], which means data can be transmitted at a number of rates according to the channel condition. For example, when the signal-to-noise ratio (SNR) is high, i.e., error detection and recovery is not that important [28], an aggressive and efficient modulation scheme can be applied to increase the rate. When the SNR is low, a conservative and redundant modulation scheme should be applied to reduce bit error rate. In practice, IEEE 802.11b supports transmission rates of 1, 2, 5.5, and 11 Mbps, and IEEE 802.11a supports data rates of 6, 9, 12, 18, ..., 54 Mbps [28, 55].

With the physical layer multi-rate capability, new MAC layer mechanisms are required to exploit this capability. Two different MAC mechanisms are supported by the IEEE 802.11 standard [30]: one is called *distributed coordination function* (DCF), which is based on carrier-sense multiple access with collision avoidance; the other is called *point coordination function* (PCF), which is based on polling. DCF is the basic MAC mechanism whereas PCF is built on top of DCF and provides contention-free media access.

4.1.1 The IEEE 802.11 PCF Protocol

In PCF, the access point (AP) acts in the role of the point coordinator, and it controls the medium access in a poll-and-response manner. The period during which PCF operates is called the *contention-free period* (CFP). Before the CFP begins, the AP operates under DCF, but it makes use of the *priority inter-frame space* (PIFS) to seize the medium, and then sends out a beacon packet containing the duration of the CFP. Once the AP has announced the start of a CFP, it may start transmitting data to the stations, or sending contention-free poll frames to the stations. During a CFP, a station can only transmit the impending data after being polled by the AP. If the polled station has data to send, it transmits one data packet. Otherwise, it will response the AP with a NULL packet. When the AP sends a packet to a station, it expects to hear a packet within a *short inter-frame space* (SIFS)[53]. If no packet is heard within a SIFS, the AP will reclaim the medium and send its next poll packet after a PIFS period. When the CFP ends, the AP sends a contention-free end packet. As operating under the PCF, the AP maintains a polling list for selecting stations that are eligible to be polled. Each station entering the coverage area of the WLAN needs to perform an association procedure [30] so that it can be added to the polling list by the AP.

4.1.2 The IEEE 802.11 DCF Protocol

IEEE 802.11 DCF is based on the request-to-send (RTS) and clear-to-send (CTS) mechanism. In particular, after a transmitting node senses an idle channel for a time period of a *distributed inter-frame space* (DIFS), it backs off for a time period which is chosen uniformly from the range of 0 to its contention window size. After each successful

data transmission, the window size is set to CW_{min}, which denotes the pre-specified minimum contention window. After the backoff timer expires, the node sends a RTS to the receiver. If the receiver successfully receives the RTS, it replies a CTS after a time period of *short inter-frame space* (SIFS). When the sender receives the CTS, it transmits the impending packet. For the purpose of reliability, the receiver needs to reply an ACK after it receives the packet correctly. Any other node overhearing either the RTS or the CTS extracts the information contained in the packet and updates its *network allocation vector* (NAV), which contains the time period reserved for data transmissions. Then, the node defers its transmission until its NAV expires. For each transmission failure, which may be caused by collisions or channel errors, a binary exponential backoff is applied to double the backoff window, and the window size is bounded by the maximum contention window (denoted by CW_{max}).

4.2 Motivations

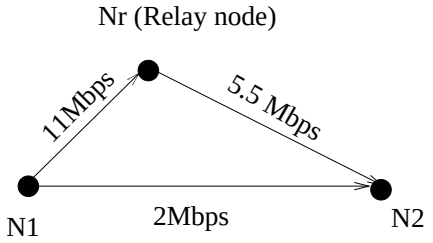


Fig. 4.1. The advantage of using the relay node

Since the channel condition varies with time and it is location dependent [54], the multi-rate capability can be further exploited by enabling MAC layer multi-hop transmission (mainly two-hop in our work). For example, as shown in Figure 4.1, suppose N_1 needs to send data to N_2, and the channel of $N_1 \rightarrow N_2$ only supports a transmission rate of 2 Mbps. At the same time, the channel conditions of $N_1 \rightarrow N_r$ and $N_r \rightarrow N_2$ are much better, and they can support data rates of 11 Mbps and 5.5 Mbps respectively. With a packet length of L, if the data can be transmitted along $N_1 \rightarrow N_r \rightarrow N_2$ at the MAC layer, the transmission delay is approximately $(\frac{1}{11} + \frac{1}{5.5})L$. Thus, the actual transmission rate is approximately equal to $\frac{5.5 \times 11}{5.5+11} = 3.7 Mbps$, which is much larger than 2 Mbps, when the packet is transmitted along $N_1 \rightarrow N_2$. Considering the implementation complexity, we focus on two-hop MAC layer relay, which is sufficient in most cases.

To use relay, the sender and the receiver must be within the transmission range of each other. Otherwise, network layer packet forwarding is required. Two important things need to be mentioned: first, the MAC layer relay differs from network layer packet forwarding in that packets relayed at the MAC layer do not have queuing delays. Specifically, when N_r has many packets in the queue, the packet relayed by N_r would experience a long queuing delay. second, the MAC layer relay does not affect the bandwidth allocation of the relay node. During the time period of the data transmission between the sender and the receiver, if the relay node does not relay the data packet, it still cannot access the medium, since its NAV has been set so that it would defer medium access until the transmission between the sender and the receiver finishes. This

property is helpful to apply some rewarding schemes [11, 40, 56] to incentive each node cooperatively relay packets for other nodes.

There may be doubts on whether the relay mechanism will work since the channel conditions of $N_1 \rightarrow N_2$ and $N_2 \rightarrow N_r$ may be unstable, and then the actual transmission rate that can be achieved with relay could be lower than that with direct transmission. Fortunately, as stated in [55], when the node does not move very fast, i.e., less than 20 m/s, the coherence intervals [54, 55][1], are on the order of multiple packet transmission times. In most cases, since mobile nodes move fairly slow (say less than 5 m/s) in ad hoc networks, it is feasible to exploit relay opportunities for each packet transmission (if there exists a suitable relay node) so that the performance of the system can be significantly improved.

4.3 System Model

When a WLAN operates with PCF, the system is a typical basic service set (BSS) where all mobile stations are in the coverage area of the *access point* (AP). Each node, which refers to either a station or the AP, is equipped with a WLAN card that enables the node to communicate with each other. The AP relies on the PCF protocol to control the media access within the BSS and grants access to each station by polling. When the system is under DCF, nodes may form an multi-hop ad hoc network without an pre-installed infrastructure. We consider a wireless network based on IEEE 802.11b that can support transmission rates of 1, 2, 5.5 and 11 Mbps. The wireless medium is shared

[1]The coherence interval is the average time interval during which the channel conditions are correlated.

among multiple contending mobile nodes, i.e., a single physical channel is available for wireless transmission.

The physical layer uses direct-sequence spread spectrum (DSSS). Following the specification of 802.11 [30], the direct peer-to-peer communication is enabled as long as two nodes are within the transmission range of each other. According to the channel condition, a packet could be transmitted at different transmission rates. We assume that only data packets can be transmitted at different transmission rates, but control packets (e.g. poll, beacon, NULL, request-to-send, clear-to-send, ACK) are transmitted with the base rate which is 2 Mbps in this thesis. For simplicity, we assume that each node transmits its packets using a constant transmission power level. The wireless channel between the sender and the receiver is assumed to be almost symmetric. In this thesis, we will not consider security issues and the motivation for relay. Many existing techniques [11, 29, 46] can be used to address security issues and the motivation for relay. Although most WLANs allow roaming or hand-off, for simplicity, we do not consider this in the work.

Based on the distance, the sensing power and the modulation scheme, a node can be in different range of the sender: the *transmission range* and the *carrier sensing range*.

- *transmission range:* within this range, the node can receive and correctly decode the packet.

- *carrier sensing range:* within this range, the node can sense the signal but cannot decode the packet.

4.4 The Relay-enabled PCF Protocol

4.4.1 Collecting the Channel Condition

The AP is designed to perform the rate adaption for the mobile stations. As a result, the AP must be able to estimate the channel condition from the packet sender to the packet receiver. To achieve this goal, each station performs receiver-based channel condition measurement, and the AP collects the channel condition measurements from each station. In particular, each station is required to have a receive table, denoted by *recv_table*, to store the channel condition measurements of each flow. Each entry of the recv_table is a tuple: $< sender\_id, rate >$ and the content of each entry is determined as follows. When a station, say S_i, overhears a packet (by listening promiscuously), it measures the SNR when receiving the packet, and selects the proper transmission rate based on the measured SNR. It also takes the MAC layer header of the packet and uses the source address as the sender_id. If the sender_id does not exist in the recv_table, a new entry is added. If the selected rate is different from the old one, the station needs to update the corresponding rate field. If there is any change in recv_table, the station should notify the AP by sending a *rate notification* (RN), which has a structure similar to that of the the recv_table. In practice, a station only sends the notification when it sends out a data packet or an ACK. For example, when S_i receives a data packet or is polled but no data to send, it checks if any entry in the recv_table has been changed since the last time of sending the RN. If there is any change, S_i sends a RN piggybacked with the ACK or the NULL packet to the AP. Since the ACK or the NULL packet is

transmitted at the rate of 2 Mbps, the AP can successfully extracts the RN from the received packet.

The rate notification tunneling: When S_i is polled and has data packet to send, the situation becomes more complex due to different transmission rates. It is possible that the data packets are transmitted at a high rate (e.g., 11 Mbps) from S_i, whereas the channel quality of $S_i \rightarrow AP$ can only support a low rate (e.g., 2 Mbps). Hence, the AP may not be able to overhear the data packet from S_i, and cannot extract the piggybacked RN_i correctly. Without the up-to-date channel condition from S_i, the AP may under-estimate or over-estimates the channel capacity of the links related to S_i, and the system performance will be degraded. To deal with this problem, we use the following solution: Suppose station S_i sends a data packet to another station S_j, it piggybacks its RN_i (if exists) with the packet. When S_j receives the packet, it checks the transmission rate as follows.

- **If** the transmission rate is equal to the lowest data rate (e.g., 2 Mbps in this paper) or there is no piggybacked rate notification, it replies an ACK piggybacked with RN_j (if exists) if there is any change in its recv_table.
- **If** the transmission rate is greater than the lowest rate, and RN_i has been piggybacked with the data packet, S_j sends S_i an ACK which piggybacks both RN_i and RN_j (if exists).

This process has been illustrated in Figure 4.2. In Figure 4.2, the dotted line pointing to the AP means that the AP can successfully overhear the packet. We can see that if RN_i is transmitted at a high data rate, it will be tunneled to the AP through

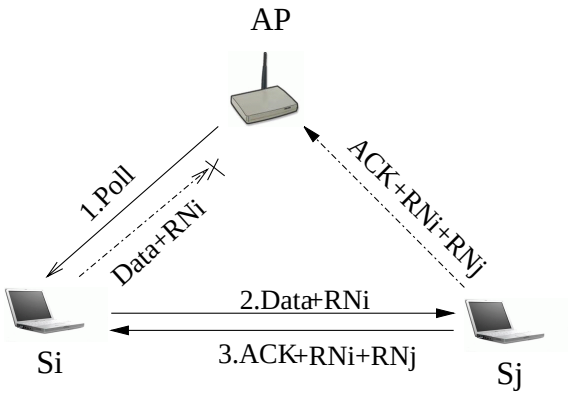


Fig. 4.2. An example of the rate notification tunneling

the ACK from S_j. Since the ACK is transmitted at the rate of 2 Mbps, the AP can overheard (by listening promiscuously,) and extract RN_i from the ACK.

The sender-initiated NACK mechanism: S_j may not be able to receive the packet from S_i correctly under some situations. Since the AP may use out-of-date channel condition to estimate the data transmission rate of $S_i \rightarrow S_j$, it is possible that the AP asks S_i to send the data packet to S_j with a high data rate, which is higher than what the channel can support. As a result, S_j cannot extract the piggybacked RN_i, and cannot tunnel it to the AP. To address this problem, we propose a new acknowledgment mechanism as follows:

- **If** S_i sends the data packet to S_j and cannot hear the ACK within a SIFS, which means the transmission failed (otherwise, S_j would reply the ACK), S_i generates a NACK and sends out the NACK after a SIFS. If it also has RN_i to send, the RN_i is piggybacked with the NACK.

- **If** the AP does not hear an ACK after the transmission of the data packet within a SIFS, it will send another poll provided that it does not hear any packet within a PIFS.

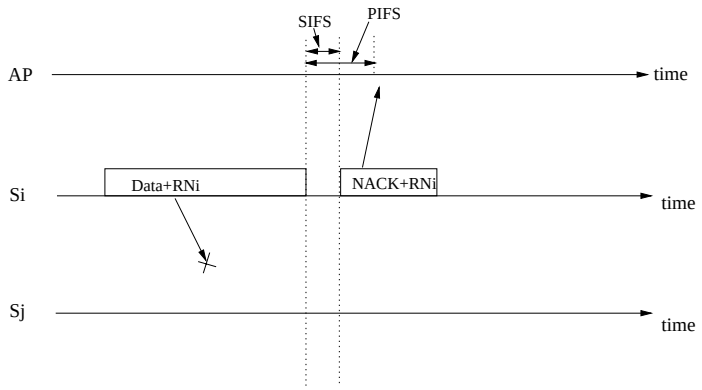


Fig. 4.3. An illustration of the sender-initiated NACK mechanism

This process has been illustrated in Figure 4.3. With the rate notification and the sender-initiated NACK mechanisms, RN_i can always be delivered to the AP in time if all transmissions are successful at the lowest rate. When the transmission failure is detected, there are two options: First, S_i can reduce the rate of $S_i \rightarrow S_j$ to the next lower rate and the AP polls it again for retransmission; Second, the AP just skips over S_i and polls the next station in the polling list. In this paper, we select the first one.

4.4.2 The Rate Adaption and Relay Mechanism

In this section, we describe how the AP finds the proper data rate and how to achieve the MAC layer relay mechanism.

The rate selections: Before the AP polls a station S_i, it needs to determine the transmission rate for S_i according to the collected channel conditions from other stations. With the estimated channel conditions related to S_i, the AP finds the proper rate for all possible receivers of S_i because it does not know to which node S_i will send. Suppose the AP stores the collected information into a rate table. where an entry, denoted by $\hat{r}_{ij}$, is the estimated transmission rate from node i to node j (A node refers to a station or the AP). The AP can search the rate table and find out the set of potential receivers of S_i, denoted by R_i, according to:

$$\mathcal{R}_i = \{\forall j \in BSS | \hat{r}_{ij} \neq NULL\} \tag{4.1}$$

where BSS represents all nodes in the basic service set. However, $\hat{r}_{ij} \neq NULL$ only implies that node j is within the transmission range of node i, which does not necessarily mean that node j will be the receiver of node i. Since finding and sending all possible transmission rates of the nodes in $\mathcal{R}_i$ may incur unnecessary computation and transmission overheads. In order to get a more accurate receiver set of each station, the AP relies on a learning process as follows: The AP maintains a flow table, denoted by FT, which contains the address pair of the sender and receiver of each flow that has been active in the system. When the AP overhears a data packet, and the address of the sender and receiver has not been registered in the FT, the AP adds the new address pair into the

FT. With the FT, $\mathcal{R}i$ can be represented by:

$$\mathcal{R}_i = \{\forall j \in BSS | \hat{r}_{ij} \neq NULL \wedge < i, j > \in FT\} \tag{4.2}$$

With $\mathcal{R}_i$, before the AP polls S_i, it looks up the rate table and finds the proper transmission rates for S_i. t Since the way of transmission can be one-hop or two-hop, the data rate set of S_i, denoted by DR_i, is computed according to:

$$DR_i = \{max(\hat{r}_{ij}, max(\frac{\hat{r}_{ik} \times \hat{r}_{kj}}{\hat{r}_{ik} + \hat{r}_{kj}})) | \forall j \in \mathcal{R}_i \wedge \forall k \in BSS \wedge \hat{r}_{ik}, \hat{r}_{kj} \neq NULL\} \tag{4.3}$$

Intuitively, for each flow whose sender is S_i, the AP selects the highest transmission rate among all possible options the transmission (one-hop or two-hop). The computational cost of the selections is at $O(n^2)$ where n is the number of the nodes in the BSS. Since n is a moderate number per BSS, the cost is affordable to APs.

The relay mechanism: After selecting the transmission rates for S_i, the AP needs to notify S_i of what rate to use and the way of the transmission. This data transmission notification can be piggybacked with the poll message. If the data rate is achieved by one-hop transmission, the relay ID and the second hop rate will not be needed. If the channel conditions do not change very frequently, the content of two consecutive DR_is does not have too much difference. In order to reduce the transmission overhead, the AP only sends the difference between the current DR_i and the previously computed DR_i. Meanwhile, each station also has a transmit table which caches the received data transmission notifications. When S_i extracts the data transmission notification from the

poll message, it updates its transmit table. Before it sends a data packet, it checks if there is an entry in the table whose destination ID matches the address of the packet receiver. If an entry is found, S_i can get the correspondent transmission mode (one hop or two hop) and data rate. Otherwise, it transmits the data packet at the lowest data transmission rate.

According to the standard of IEEE 802.11, each data packet only has two addresses: one for the sender and one for the receiver. In order to support two-hop MAC layer transmission, the address of the relay node needs to be added into the MAC layer header. With three different addresses, the relay mechanism is designed as follows. If a station receives a poll packet and finds out that the packet to send needs a two-hop transmission, it switches the addresses of the receiver and the relay node so that the packet is transmitted to the relay node first. Meanwhile, it sets the subtype of the packet to a pre-specified value, which indicates that the packet should be delivered in the MAC layer relay manner. When the relay node receives the packet, it checks the subtype of the packet and finds out that the packet needs to be transmitted with two-hop. Then, the relay node sets its address as the source address saves the address of the original sender in the header and sends the packet to the original receiver. If the packet is correctly transmitted, the receiver replies an ACK directly to the original sender (Note that the address of the original sender is stored in the packet header). If the packet is corrupted in either hop, the sender of that hop is able to know the error by overhearing if there is any packet transmitted within a SIFS, and reacts similarly to one-hop transmissions. To achieve high data rate, no ACK is transmitted from the relay node (S_2) to the sender,

and no ACK is transmitted from the receiver to the relay node. The acknowledgment is achieved explicitly by overhearing.

4.5 The Performance Evaluations

4.5.1 The Propagation Model

When the wireless channel is assumed to be stable, we use the propagation model in ns-2 [27], which combines the Friis free space propagation model and the two-ray ground propagation model [54]. Basically, when the sender and the receiver are close, the Friis free space model is applied so that the path loss exponent is 2. Otherwise, the two-ray ground propagation model and the path loss exponent becomes 4.

When there is multi-path fading or relative movement between the sender and receiver, the channel condition between them may change frequently. The frequency of this change depends on the relative speed of the mobile node with respect to its surroundings. We use the Ricean fading model [54] to simulate the fading channel conditions. The Ricean distribution is given by:

$$p(r) = \frac{r}{\alpha^2} e^{-(\frac{r}{2\alpha^2}+K)} I_0(2Kr) \tag{4.4}$$

where K is the distribution parameter representing the line-of-sight component of the received signal, α^2 is the variance of the background noise, r is the received power, and $I_0(.)$ is the modified Bessel function of the first kind and zero order [54].

When a node receives or overhears a packet, it determines whether the packet is corrupted according to the packet length, the SNR and the corresponding bit error rate (BER). With the BER of BPSK given by [38] and the approximately BER performance

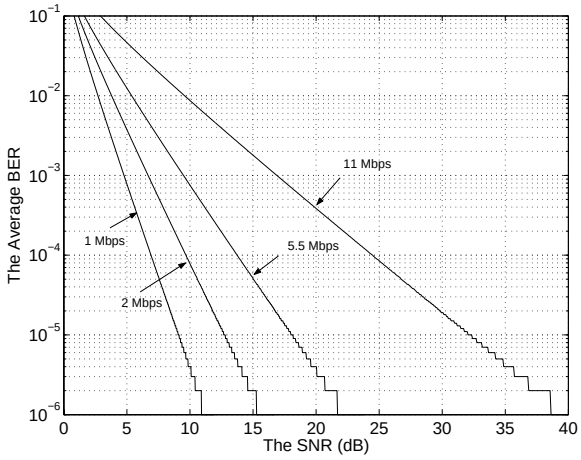


Fig. 4.4. The BERs under different transmission rates

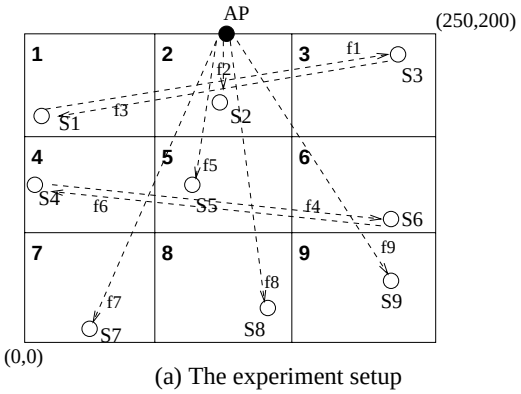
using different modulation techniques in [3], we have the BERs at different transmission rates shown in Figure 4.4. The probability that p can be successfully received, denoted by P_{succ}, is calculated by:

$$P_{succ} = (1 - BER(\gamma))^L \tag{4.5}$$

where $BER(\gamma)$ is the BER with the SNR of γ, and L is the packet length.

4.5.2 The Simulation Setup

Our simulation is based on ns-2 extension [52, 27]. The experiment setup is shown in Figure 4.5(a). As shown in Figure 4.5(a), one AP and nine stations exist in a 250× 200 flat area. The coordinates of the nodes are listed in Figure 4.5(b), and the distance between any two nodes can be calculated. We assume the stations are evenly distributed in the area. We divide the area into 9 subareas indexed from 1 to 9, and the mobile station with the same index resides in the correspondent subarea. Each station has a flow, starting either from another station or from the AP. If the connection is from one



(a) The experiment setup

ID	AP	S1	S2	S3	S4	S5	S6	S7	S8	S9
x	125	13	112	223	9	96	224	35	96	237
y	200	139	163	192	105	92	70	11	36	59

(b) The coordinates of each node

Fig. 4.5. The experiment setup

station to another, the flow number is the same as the index of the sender (e.g., f_1 represents the connection from $S_1 \rightarrow S_3$.) If the connection is between the AP to a station, the flow number is the same as the index of the receiver (e.g., f_2 represents the connection from $AP \rightarrow S_2$.) Similar to [55], the distance threshold for 11Mbps, 5.5Mbps, and 2Mbps are 100m, 200m, and 250m respectively. For simplicity, all flows in the system are assumed to be always backlogged. The packet length is set to be 1000 bytes and the simulation time is assumed to be 100 seconds.

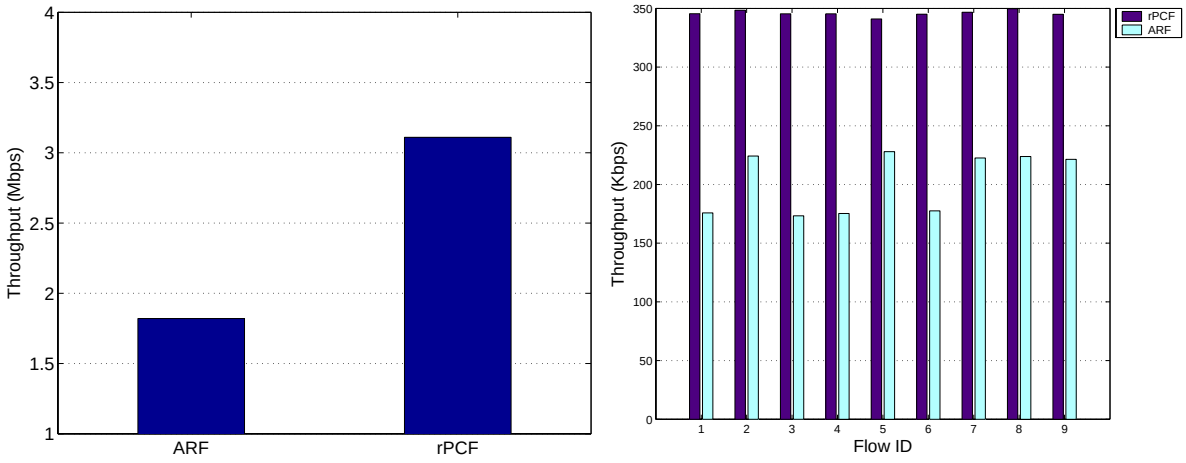
We compare rPCF with PCF-based ARF [35], which is a sender-based rate adaption scheme. The protocol works as follows. If ACKs for two consecutive data packets are lost, the sender reduces the transmission rate to the next lower data rate and starts a timer. If ten consecutive ACKs are received, the transmission rate is raised to the next

higher data rate and the timer is cancelled. If the timer expires, the transmission rate is raised if an ACK is received for the next packet; otherwise, the rate is lowered again and the timer is restarted. Since the simulation results of [28] show that the throughput of ARF does not change much when the timeout is larger than 60 ms, we set the timeout value to be 80 ms in our simulations.

4.5.3 Simulation Results

We evaluate the impacts of different factors on the system performance, which is measured by the system throughput. We also evaluate the system overhead of the proposed solution.

4.5.3.1 Performance under Stable Links



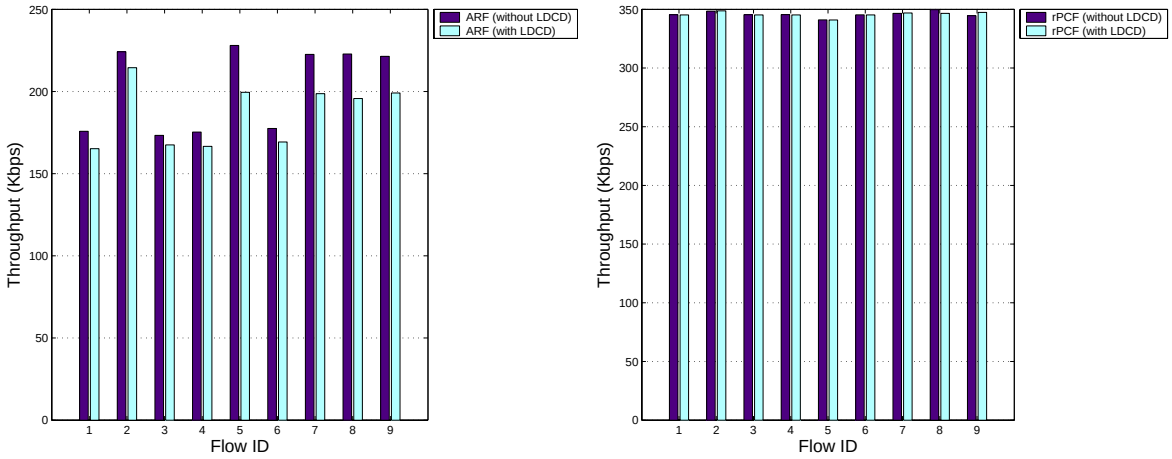
(a) System Throughput (b) Throughput of each flow

Fig. 4.6. Performance comparisons between rPCF and ARF

In this subsection, we assume the channel conditions are stable, and the received power is calculated only by the Friis and 2-ray ground propagation models. Further, all stations are assumed to be static. In this way, we can maximize the performance of different schemes. The simulation results are shown in Figure 4.6. From Figure 4.6(a), we can see that rPCF improves the system throughput by more than 70%. The performance gain is mainly due to the MAC layer relay. For example, the transmission rate for f_1 is limited to 2Mbps under ARF due to the long distance between S_1 and S_3. Under rPCF, the packets of f_1 can be relayed by the AP. Because the data rates of $S_1 \rightarrow AP$ and $AP \rightarrow S_3$ are 5.5Mbps and 11.0Mbps respectively, the actual data rate of f_1 is approximately 3.7Mbps. In addition, let us take f_1 as the example, ARF raises the transmission rate of f_1 to 5.5Mbps after S_1 receives the ACKs for ten consecutive packets. Then, this data rate raising will cause f_1's next packet to fail since the distance between S_1 and S_3 is too far to support 5.5Mbps. However, ARF reduces the transmission rate of f_1 back to 2.0Mbps only when S_1 misses two ACKs for two consecutive data packets. This leads to the waste of bandwidth due to retransmissions.

4.5.3.2 Impacts of Location Dependent Channel Degradation

The wireless channel condition can be affected by several factors. One of them is the surrounding environment, and the channel quality can be significantly degraded when the surrounding interference or noise is high. As a result, the location dependent channel degradation (LDCD) commonly exists in wireless networks. To evaluate its impacts, we assume that the channel condition of $AP \rightarrow S_5$ is poor, and it can only support a data rate of 2Mbps. All stations are assumed to be static. The simulation results are



(a) Throughput difference of ARF (b) Throughput difference of rPCF

Fig. 4.7. Impacts on throughput

shown in Figure 4.7. As shown in Figure 4.7, the system throughput of ARF is reduced. However, the throughput is not affected under rPCF. Under rPCF, after S_5 sensed the poor channel condition between the AP and itself, it notifies the AP of such change. When the AP needs to send a data packet to S_5, it can find that the direct link between them can only support 2Mbps, but a higher data rate can be achieved if the packet is relayed by S_2. Since the rates of $AP \rightarrow S_2$ and $S_2 \rightarrow S_5$ are both 11.0Mbps, with the relay of S_2, the actual transmission is approximately 5.5Mbps, which is approximately the same as the rate can be support if the channel condition of $AP \rightarrow S_5$ is good.

4.5.3.3 Impacts of the Line-of-sight parameter K

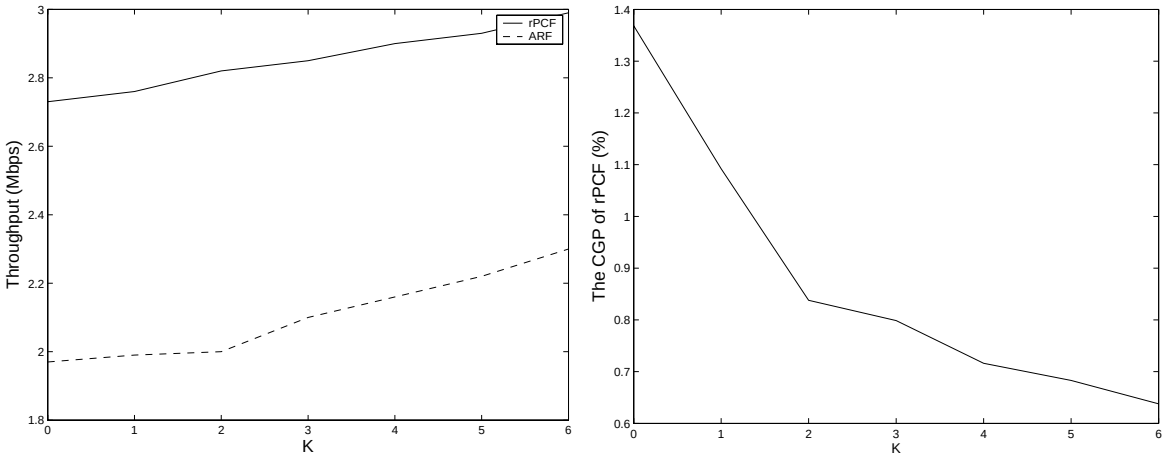
In the previous two subsections, we assume the channel condition is stable. Since the channel condition is stable and the stations do not move, the AP and the mobile stations only send a very limited number of rate notifications (RNs) and data transmission

Type	Size (bits)
RN	52
one-hop TN	52
two-hop TN	104
NACK	304

Table 4.1. The control overhead of rPCF

notifications (TNs), and the overhead of rPCF is negligible. Since the wireless channel condition may change frequently, the overhead of rPCF, which includes transmitting RNs, TNs, and NACKs, should be evaluated. To evaluate this overhead, we change the line-of-sight parameter K. A large K means a better channel quality while a small K means a poor channel quality. The sizes of RN, TN, and NACKs are listed in Table 4.1. We collect the generated RNs, TNs and NACKs, and add them together to get the control overhead in terms of bits. Then, using these control overhead (in bits) divided by the simulation time, we get the control overhead in terms of data rate (bps). Since ARF does not have any control overhead, we introduce a new metric called *cost gain percentage* (CGP) to evaluate the control overhead of rPCF fairly. The CGP (in percent) is the ratio of the control overhead (bps) to the throughput gain of rPCF (compared to ARF).

The simulation results are shown in Figure 4.8. From Figure 4.8(a), we can see that the system throughput of both rPCF and ARF increase as K increases. As K increases, the impact of fast fading becomes smaller, and the channel condition becomes better. As a result, the system throughput improves. From the figure, we can also see that the impact of K on the throughput of ARF is higher than that on rPCF. This is due



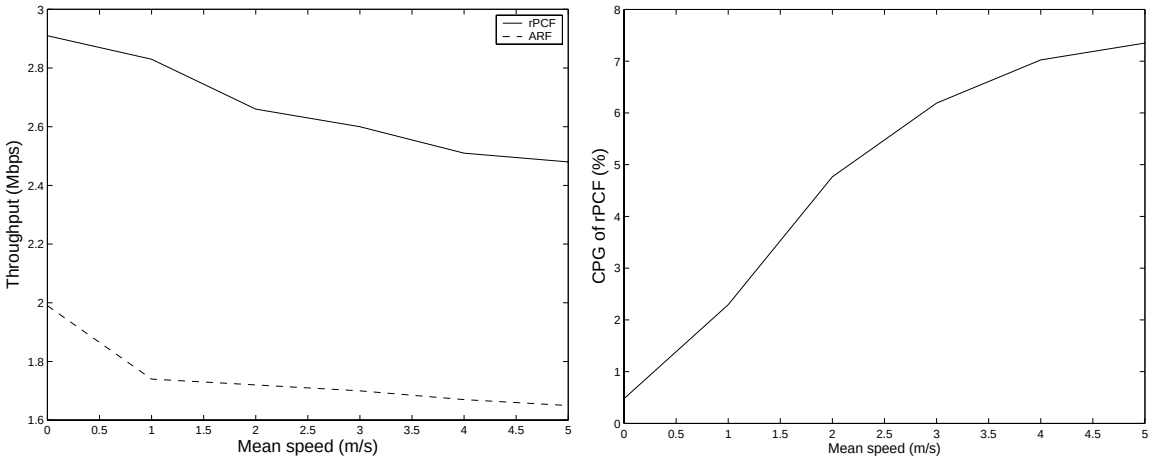
(a) Throughput of ARF and rPCF (b) The CGP of rPCF

Fig. 4.8. Impacts of the line-of-sight parameter K

to the fact that the number of mis-adaptations increases under ARF when the channel condition becomes unstable. However, under rPCF, each station frequently reports the channel conditions to the AP, so that the AP can perform rate adaption and notify correspondent stations in time. As a result, stations can adjust the transmission rates in time and reduce chance of using wrong data rates. From the results shown in Figure 4.8(b), we can see that the control overhead is very small. For example, the CGP is only about 1.3% when $K = 0$, and drops to 0.6% when $K = 6$.

4.5.3.4 Impacts of Mobility

Mobility affects the channel condition in two ways. First, it changes the node's location which may affect the value of K and the strength of the received signal power. Second, due to Doppler shift in frequency of the received signal, it may reduce the channel coherence time period, which is the interval during which the channel condition



(a) Throughput of ARF and rPCF (b) The CGP of rPCF

Fig. 4.9. Impacts of mobility

keeps stable.In this subsection, we evaluate the impact of mobility on the performance of rPCF. Each station is assumed to move randomly within the correspondent subarea with the same speed, and the mean speed changes from 0 to 5 m/s. Similar to [55], the value of K is fixed to be 5. The simulation results are shown in Figure 4.9. As shown in Figure 4.9 (a), the system throughput of rPCF and ARF drops as the the mean moving speed increases. As the moving speed increases, the channel condition changes frequently. When the channel condition becomes unstable (e.g. when the mean speed is 5 m/s), it is difficult for ARF to adjust the transmission rate to a high rate, and hence most of the data are transmitted at a low rate of 2Mbps. For rPCF, the increased control overhead such as NACKs, RNs and TNs also reduces the system throughput as the moving speed increases. As shown in Figure 4.9 (b), the control overhead is still very small. For example, when the mean speed is 5 m/s, the CGP is only about 7.7%.

4.6 The Relay-enabled DCF Protocol

4.6.1 The Basic Protocol

4.6.1.1 The Service Advertisement

Similar to most existing work [28, 55], we apply receiver-initiated channel condition measurement and let the receiver notify the sender of the transmission rate via CTS. With rDCF, each node promiscuously listens to all ongoing RTS and CTS packets. By extracting the piggybacked transmission rate in the CTS, a node knows the channel condition between the sender and the receiver of the impending data packet. Meanwhile, it can measure the channel quality between the sender (or the receiver) and itself by sensing the signal strength of RTS or CTS packets. Since CTS packets do not have the MAC address of the packet sender, a node needs to infer the sender of the CTS according to the semantic of CTS. In particular, suppose N_r overhears a RTS from N_i to N_j. If it overhears a CTS addressed to N_i after a SIFS, N_r can infer that the sender of the CTS is N_j.

For a given flow between a pair of sender and receiver, with the measured channel quality, if a node finds that the packets can be transmitted faster with the MAC layer relay, it adds the identity (e.g. MAC address) of the sender and the receiver into its willing list. In order to reduce the control overhead, we can limit the length of the willing list (i.e. 10 entries). Periodically, each node advertises its willing list to its one-hop neighbors. Some schemes such as [5] can be used to improve the reliability of the broadcast. Once a node, say N_i, receives a willing list from N_r, and finds that $N_i \rightarrow N_j$ is in the list, it adds N_r into its relay table (Note that it is possible that there are

more than one relay node available for $N_i \rightarrow N_j$). As an optimization, the number of redundant service advertisements for a given flow can be reduced as follows: Before sending the advertisement, if N_r has overheard more than m advertisements containing $N_i \rightarrow N_j$ from other nodes, it knows that there are at least m other nodes have claimed to be the relay node for $N_i \rightarrow N_j$, and then deletes $N_i \rightarrow N_j$ from the willing list. In this thesis, we set the value of m to be 3.

4.6.1.2 The Triangular Handshake

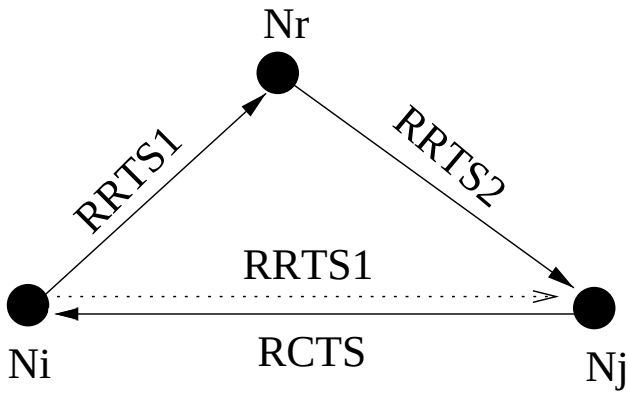


Fig. 4.10. An illustration of the triangular handshake

In the standard DCF protocol, the RTS/CTS handshake is required for each unicast packet transmission in order to prevent collisions. In [28, 55], this handshake is further utilized to probe the channel condition on the per-packet basis. Following these principles and considering backward compatible to standard DCF, we modify DCF and refer this new protocol as the basic protocol of rDCF. As shown in Figure 4.10, where

the dashed line pointed to N_2 means N_2 can overhear the packet, when a node N_i has a packet for N_j, it first searches the relay table using N_i as index. If N_i cannot find a relay node, the standard DCF is applied. Otherwise, N_i picks a relay node N_r and starts to coordinate the communication with N_r and N_j. Specifically, N_i sends a new packet, called *relay RTS* (RRTS1), to N_r. When N_r receives the RRTS1, it generates another relay RTS (RRTS2) and sends it to N_j. By sensing the signal strength of RRTS1 and RRTS2, N_r and N_j individually determines the achievable transmission rate of $N_i \rightarrow N_r$, $N_i \rightarrow N_j$ and $N_r \rightarrow N_j$, denoted by R_1, R_{dir} and R_2 respectively, where R_1 is piggybacked in RRTS2. After receiving RRTS2, based on R_1, R_{dir} and R_2, the receiver replies CTS which piggybacks R_{dir} if the packet cannot be transmitted faster with relay. Otherwise, N_j replies a *relay CTS* (RCTS), which piggybacks R_1 and R_2, to N_i.

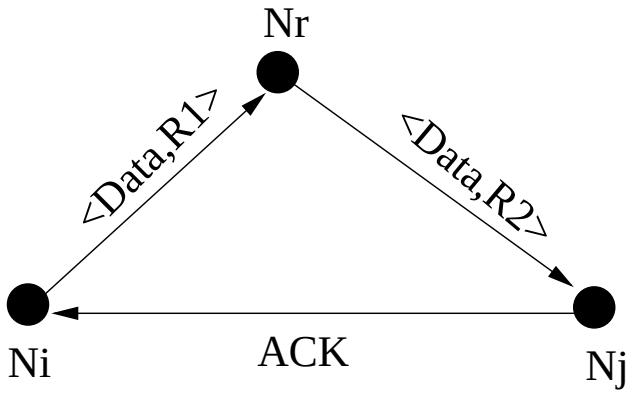
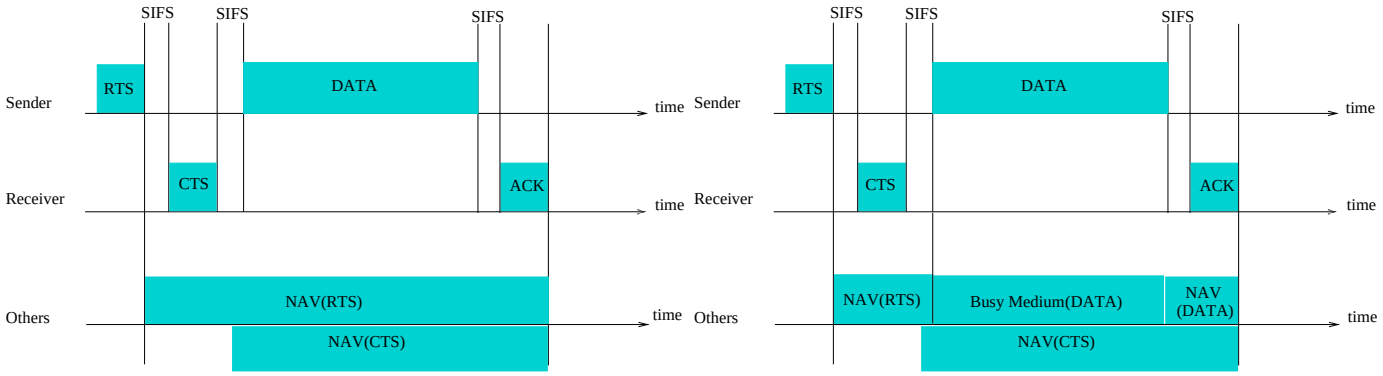


Fig. 4.11. An illustration of the MAC layer relay

If N_i receives a CTS, it sends the data packet directly to N_j with the transmission rate of $R_{i\rightarrow j}$. If N_i receives a RCTS, as shown in Figure 4.11, it sends the data packet to N_r with the transmission rate of R_1. After N_r receives the packet, it relays the packet to N_j with the transmission rate of R_2 after a SIFS. If the packet is correctly received by N_j, N_j replies an ACK to N_i. If the transmission fails, the sender can detect the failure with a time out mechanism similar to the standard DCF [30].

4.6.2 Enhancements of rDCF

The basic protocol of rDCF describes the basic mechanism to achieve relay-enabled DCF. However, considering the bandwidth utilization, the dynamical nature of wireless channels and the impact of multi-rate transmissions, we propose techniques to further improve the performance of rDCF.



(a) The standard scheme (b) The new scheme

Fig. 4.12. The comparison of two different carrier sensing schemes

4.6.2.1 Dealing with Multi-rate Transmission

With IEEE 802.11 DCF, carrier sensing is performed using physical carrier sensing as well as virtual carrier sensing. As shown in Figure 4.12 (a) (on the next page), when the data is transmitted with a fixed rate, the sender can easily calculate the duration of the packet transmission based on the packet length and the transmission rate. However, when the transmission rate can be adaptively changed, the sender cannot precisely calculate the length of the duration before sending the RTS, since it does not know the transmission rate of the impending packet in advance. In the solution of [28], the sender chooses a data rate based on some heuristic; i.e., the most recent rate that was successfully used for transmission. This solution is not good enough for rDCF since the sender needs to estimate the transmission rates for both hops of the relay, and it may be difficult to get a precise estimate.

Our approach: We designed a new carrier sensing scheme for rDCF, which is shown in Figure 4.12 (b). Instead of estimating the possible transmission rates and calculating the duration of the data transmission, the sender first calculates the duration of the RTS and CTS transmissions only[2]. The duration can be precisely calculated since all control packets (e.g. RTS, CTS, ACK, ...) are transmitted at the base rate, say 2 Mbps. After the sender receives CTS or RCTS, it calculates the durations of the packet and the ACK based on the piggybacked transmission rate(s). In this way, our scheme can guarantee that other nodes within the transmission range of the sender and the receiver would defer medium access for exactly the duration of the packet transmission. Compared to

[2]In case of relay, it needs to calculate the duration of RRTS1, RRTS2 and RCTS transmissions

the standard approach, our approach can achieve better bandwidth utilization in some situations. For example, suppose a CTS is lost at the sender due to collision or channel error, since the standard approach has longer duration piggybacked in the RTS than our approach, the neighbor nodes of the sender would defer for a longer time period in the standard DCF. Table 4.2 lists the duration for each packet used in rDCF. In the

Packet Type	The Duration
RTS	$CTS + \sigma + 2SIFS$
CTS	$Data(L, R_{dir}) + \sigma + 2SIFS$
RRTS1	$RRTS2 + RCTS + 2\sigma + 3SIFS$
RRTS2	$RCTS + DATA(L, R_1) + 2\sigma + 3SIFS$
RCTS	$DATA(L, R_1) + DATA(L, R_2) + 2\sigma + 3SIFS$
Data_{dir}	$ACK + \sigma + SIFS$
Data_1	$DATA(L, R_2) + ACK + 2\sigma + 2SIFS$

Table 4.2. The calculations of the duration in rDCF

table, σ is the maximum propagation delay, $DATA(L, r)$ means the packet with length of L is transmitted at the rate of r. Note that the calculation of each duration takes into account the transmission time of both PHY layer header and MAC layer header. $Data_{dir}$ refers to the data packet with direct transmission, and $Data_1$ is the data packet sent from the sender to the relay node, Other unlisted packets have a duration of 0.

Besides the impact on virtual carrier sensing, different transmission rates also result in different transmission ranges. For a given receiving power level, the packet transmitted with higher rate may have higher bit error rate. As shown in Figure 4.13, suppose N_i and N_j are far away from each other and the channel quality can only support 2 Mbps. Then N_j may not be able to decode a packet if N_i sends the packet at the

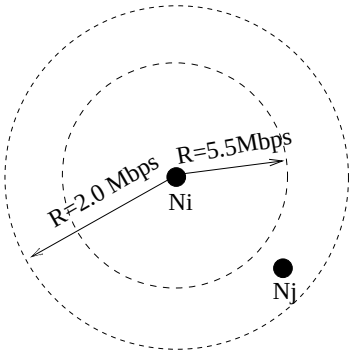


Fig. 4.13. An illustration of the different transmission ranges

rate of 5.5 Mbps. In this case, N_j is out of the transmission range of N_i. Based on this fact, when the sender sends data at high transmission rate, some one-hop neighbors may stay within its carrier sensing range but cannot extract the information of the duration piggybacked in the packet. To deal with such problems, we adopt the *reservation-sub-header* (RSH) in [28]. Specifically, a RSH is inserted preceding the data frame and is sent at the same or lower rate compared to RTS. Different from [28], as shown in Figure 4.14 and 4.15, our RSH does not need to include the MAC addresses of the sender and the receiver because the revised carrier sensing scheme would not incur any incorrect medium reservation of RTS. As a result, the overhead of our RSH is smaller than that in [28]. Since the RSH is transmitted at a low rate (1 Mbps in this paper), all one-hop neighbor nodes can extract the value in the RSH and update their NAV values accordingly.

4.6.2.2 Dealing with Dynamic Channel Condition

The channel condition may change frequently in wireless networks [54], which may have significant impacts on the performance of rDCF. In order to alleviate the impacts

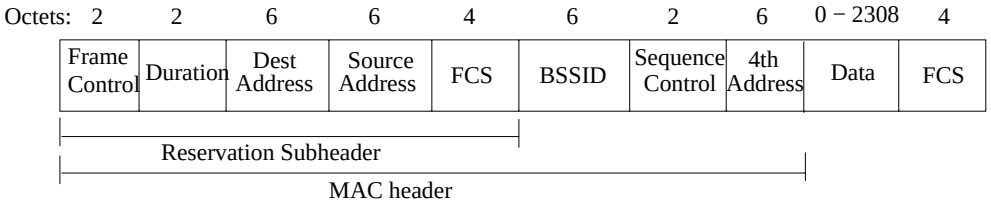


Fig. 4.14. Data packet frame format in [28]

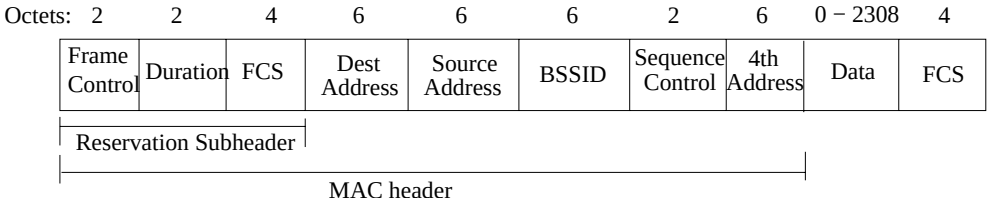


Fig. 4.15. Data packet frame format in our scheme

of dynamic channel conditions, it is desirable to adaptively decide when to perform relay according to the channel conditions.

We design a simple randomized algorithm as follows: Each relay node in the relay table of N_i is associated with a *credit* ranging in $[0.0, 1.0]$. To exploit successful relays, each time when N_i finds a relay node for the receiver N_j, N_i chooses the one with the largest credit. After selecting the relay node, N_i generates a random number in $[0.0, 1.0]$ and sends RRTS1 to the chosen relay node if the credit is greater than or equal to the random number. Otherwise, N_i applies DCF and sends RTS to N_j. When a node N_r successfully relays a packet for N_i, which is indicated by receiving the ACK, the credit of N_r is increased by 0.1. When a relay via N_r fails, the credit is decreased by 0.1. When N_i receives that willing list from N_r and finds itself in the list, the credit of N_r is enhanced by 0.5.

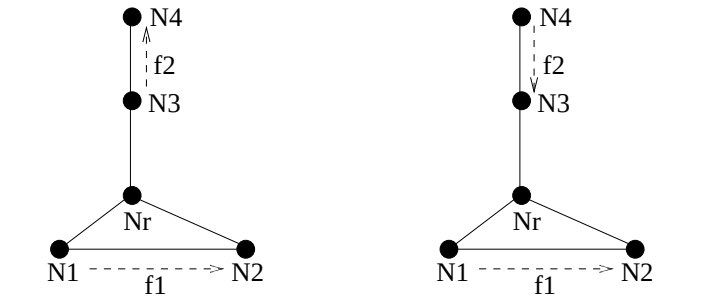
Some types of transmission failures can be detected and recovered quickly in rDCF to reduce the cost of failures. As shown in Figure 4.10, suppose N_i has a packet for N_j and finds the relay node N_r. We add two optimizations to the basic protocol as follows:

- If RRTS1 is lost, N_i can detect it if no packet is overheard after $SIFS + \sigma$ when the transmission of RRTS1 is finished. Then, it replies a CTS to N_i;
- If the data packet sent from N_i to N_r is lost, N_i can detect it if no packet is overheard after $SIFS + \sigma$. Then, N_i backoffs based on the binary exponential backoff protocol for re-transmission.

4.6.3 Impacts of Relay

In multi-hop ad hoc networks, the relay node may have some impacts on the system performance. In this section, we discuss some issues caused by relaying packets, and show that these impacts are very small in most cases through analysis.

4.6.3.1 The Impact on Spatial Reuse



(a) Exposed terminal (b) Hidden terminal

Fig. 4.16. An illustration of the impact of rDCF on spatial reuse

As packets being relayed, rDCF may have impacts on the spatial reuse of the network. As shown in Figure 4.16 (a) and (b), any pair of nodes connected by a solid line can hear each other. With standard DCF, f_1 and f_2 can simultaneously transmit data since they don't contend with each other for the medium. When N_r relays packets for flow f_1, N_3 has to defer its transmissions in order to avoid collisions, which may cause an exposed or hidden terminal problem [9, 10]. At a first glance, if N_r always relays packets for f_1, the performance of f_2 may be significantly affected. After looking into the carrier sensing mechanism of IEEE 802.11, we can see that the impact is quite small in most cases.

Suppose N_r relays a packet for f_1 at time t. For exposed terminal problem, there are two cases:

- **Case 1:** N_3 is in the transmission range of N_r at t, which means that it can extract the *duration* of the packet. N_3 can defer medium access for the period of one data transmission, and then start to contend for the medium again. As a result, in the long term, N_3 and N_1 have similar opportunities to access the channel.

- **Case 2:** N_3 is within the carrier sensing range of N_r so that it cannot extract the duration of the packet. In this case, N_3 resumes contending the medium only when the medium is idle for an extended inter-frame space (EIFS), which is equal to 364 μs [30]. As a result, N_3 may defer the medium access to sometime later than the time N_1 receives the ACK. Since the time of ACK+the post backoff[3]+$DATA_{1\rightarrow r}$

[3]After receiving the ACK, the sender is required to backoff for a random period between 0 and CW_{min}

is greater than EIFS, we can see that N_3 would not be starved and can eventually obtain the medium access.

When N_3 transmits a packet to N_4, N_r sets its NAV to be either the period of the data transmission from N_3 to N_4 or EIFS (when a collision happens). When N_1 sends packets to N_r at this time, N_r will not send RRTS2 to N_2 since its NAV has not expired. In this case, the receiver applies the optimization technique in Section 4.6.2.2 and the impending packet of N_1 is served with DCF.

For the hidden terminal problem, the impact of relay could be greater since the sender of f_2 will double its current contention window size and backoff again. However, similar to the exposed terminal problem, since N_3 does not always senses busy medium, this impact would not significantly affect the performance of f_2.

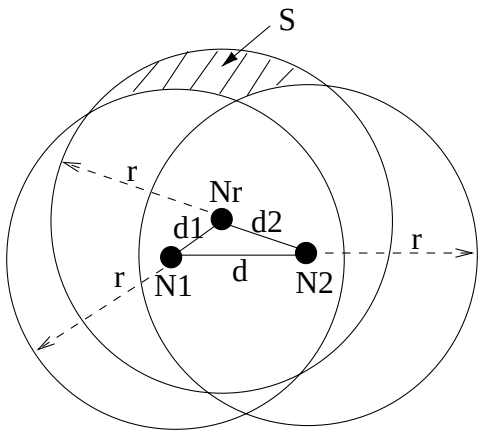


Fig. 4.17. An illustration of the extended sensing area

We also analyze the extended sensing area caused by N_r. As shown in Figure 4.17, the extended sensing area S is N_r's sensing area which does not overlap with the sensing areas of N_1 and N_2. It is not difficult to see that, for a given distance (d) between N_1 and N_2, the size of S increases as $d_1 + d_2$ increases. To meet the criteria of relay, $d_1 + d_2 \leq D_{5.5} + D_{11}$ should hold, where $D_{5.5}$ and D_{11} are the transmission range of 5.5 Mbps and 11 Mbps respectively. By setting d_1 and d_2 to be $D_{5.5}$ and D_{11} respectively, we can calculate the upper bound of the size of S.

We give some numerical results of the upper bound of increased sensing area as a function of d. Following ns-2 [27], we set r, $D_{5.5}$ and D_{11} to be 550 m, 200 m and 100 respectively. d changes from 210 m to 250 m. The numerical results are shown in Table 4.3. As can be seen, compared to the total sensing area of the sender and the receiver, the increased sensing area is small.

d (meters)	210	220	230	240	250
Upper bound of increased sensing area (%)	11.5	10.5	9.2	8.2	7.2

Table 4.3. The impact of relay on the sensing area

4.6.3.2 The Impact of Hidden Relay

Based on the location of the relay node, some node may be able to hear from the sender, but unable to hear from the relay node. For example, as shown in Figure 4.18, N_4 can hear from N_1 but cannot hear from N_r. This may cause collisions at N_1 since

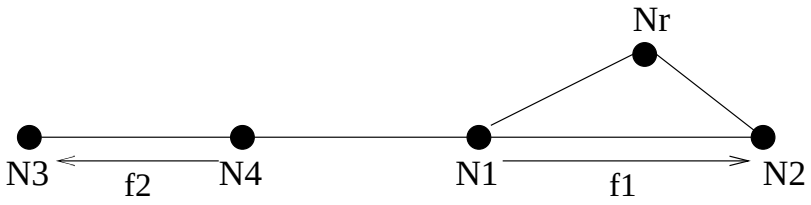


Fig. 4.18. An illustration of the impact of hidden relay node

N_4 may not defer medium access for the period of one data transmission when N_r relays a packet for f_1. In the following, we analyze this impact, and show that it is very small. Suppose N_1 sends a packet to N_r at time t, there are two cases:

- **Case 1:** N_4 can extract the duration from the packet, and defer medium access accordingly. Since the duration is equal to the time needed for relaying the data packet, N_4 would not contend for the medium before N_1 gets the ACK.

- **Case 2:** N_4 can extract the duration from the packet, and set its NAV to be EIFS. With DCF, EIFS can be used to guarantee that the sender can receive the ACK. However, it may not always hold in rDCF. Since EIFS could be smaller than $Data_{r\rightarrow 2} + ACK + DIFS$, N_4 may send a packet to N_5 before N_1 receives the ACK, since it cannot sense the signal sent by N_r and N_2. As a result, it is possible that the packet sent by N_4 collides with the ACK at N_1.

When Case 2 happens, N_1 needs to re-transmit the data packet. As stated in Section 4.6.2.2, N_1 also reduces the credit of N_r by 0.1 since the previous relay operation failed. Then, even if the flow rate of f_2 is high, the occurrence of Case 2 is bounded

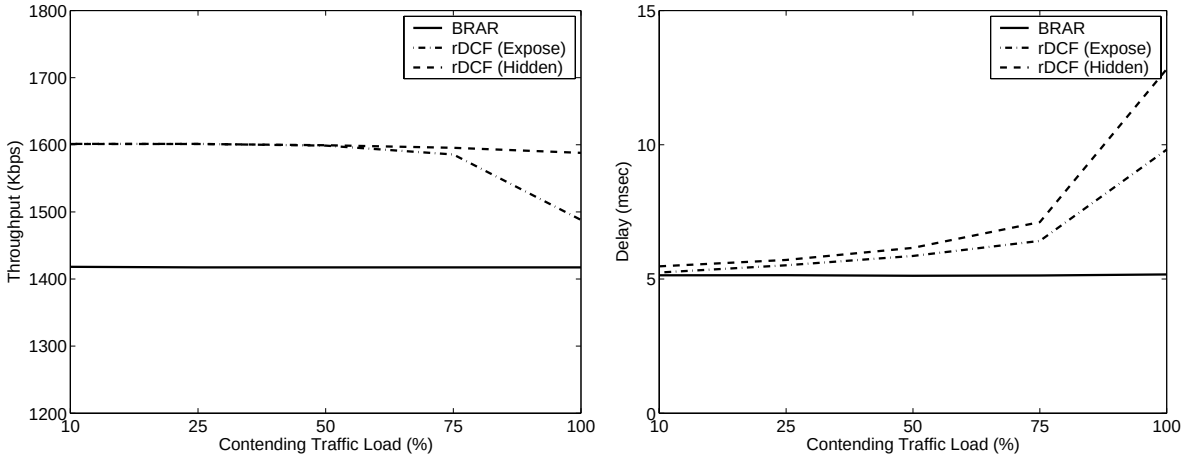
since the credit of N_r will eventually be small enough so that N_r would not be chosen for relay.

4.7 Performance Evaluation

4.7.1 The Simulation Setup

The propagation model and the distance threshold are the same as those used in rPCF. The mean period for service advertisements is 1.0 second. The packet length is set to be 1000 bytes and the simulation time is set to be 100 seconds.

We compare rDCF with the state-of-the-art protocol called *receiving based auto rate* (RBAR) protocol [28]. It has been shown that RBAR outperforms the standard DCF and the sender-based rate adaption protocol called auto rate fallback (ARF). The reason we do not compare rDCF with the opportunistic auto rate (OAR) protocol is that OAR degrades to RBAR when the link quality between the sender and the receiver is poor. The RBAR protocol works as follows. The receiver measures the channel quality based on the signal-to-noise ratio of the arriving RTS packet. Then, it sets the transmission rate for the data packet according to the highest feasible value allowed by the channel condition, and piggybacks the rate with the CTS packet. After receiving the CTS, the sender sends out the data packet with the piggybacked transmission rate.



(a) Throughput of Flow 1 (b) Delay of Flow 2

Fig. 4.19. The impact of rDCF on spatial reuse

4.7.2 Simulation Results

4.7.2.1 Impacts on Spatial Reuse

In this experiment, we evaluate the impacts of rDCF on the spatial reuse, and assume the channel condition is stable. The topologies used are Figure 4.16 (a) and (b), under which the performances are denoted as rDCF (Exposed) and rDCF (Hidden) respectively. The channel quality between the sender and the receiver of each flow can only support the rate of 2 Mbps. N_r and N_3 are within the sensing range of each other. The *contending traffic load* (CTL), which is the percentage of the maximum system throughput, of flow 1 (flow 2) increases as the aggregated traffic of the flows that are spatially close to flow 2 (flow 1) increases, and vice versa.

We first evaluate the impacts of CTL on the throughput of flow 1. We fix the rate of flow 1 to be 1600 Kbps. As shown in Figure 4.19 (a), when the CTL of flow 1 is not high (e.g. 50%), the throughput of flow 1 under rDCF is not affected and is much

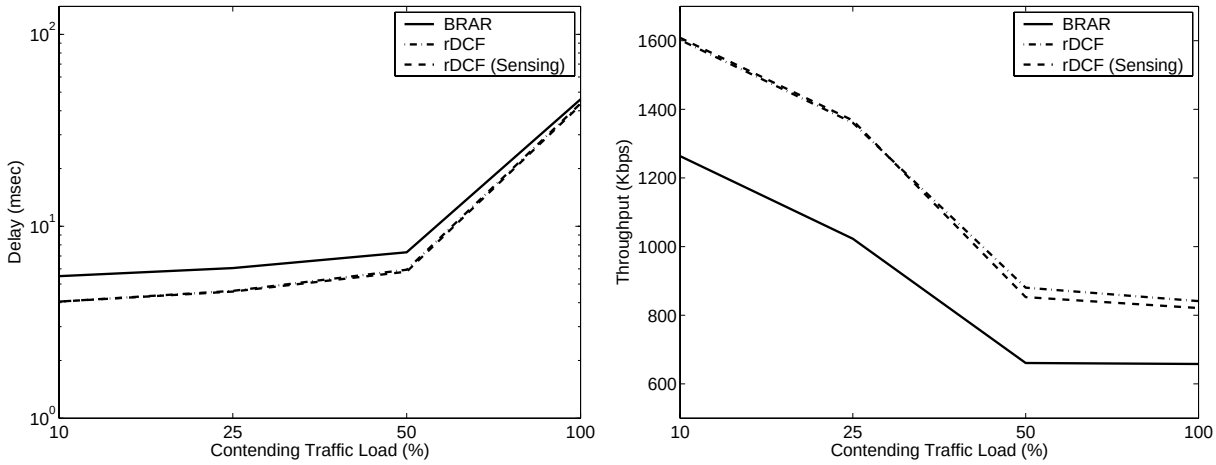
higher than that under RBAR. In case of rDCF (Expose), when the CTL of flow 1 is high (i.e. over 75%), the throughput of flow 1 decreases. As discussed in Section 4.6.3.1, since N_r frequently defers medium access for flow 2, many data packets are transmitted with direct transmissions. In case of rDCF (Hidden), the impact of flow 1's CTL is very small since N_3 only sends short packets (i.e. CTSs and ACKs).

We then evaluate the impacts of CTL on the delay of flow 2. The rate of flow 2 is fixed to be 160 Kbps (or 20 pkt/sec). As shown in Figure 4.19 (b), in case of both rDCF (Expose) and rDCF (Hidden), when the CTL of flow 2 is not high (i.e. less than 50%), its impact on flow 2's delay is quite small. When the CTL of flow 2 is very high (i.e. near 100%), the delay of flow 2 increases. The reason of the prolonged delay has been discussed in Section 4.6.3.1, and the result conforms our claim that flow 2 would not be starved.

4.7.2.2 The Impact of Hidden Relay

We study the impact of hidden relay in this section. The topology has been shown in Figure 4.18. We assume that the channel is stable. By default, in rDCF we assume N_4 can extract the duration of each data packet sent by N_1. rDCF (Sensing) denotes the situation that N_4 cannot extract the duration field. As stated in Section 4.6.3.2, the impact of hidden relay does not exist in the default rDCF, but it exists in rDCF (Sensing).

We evaluate the impact of CTL on the delay of flow 1. The rate of flow 1 is fixed to be 160 Kbps. As shown in Figure 4.20 (a), when the CTL of flow 1 is low, because of relay, the delay of flow 1 in rDCF and rDCF (Sensing) is much smaller than that under



(a) Delay of flow 1 (b) Throughput of flow 1

Fig. 4.20. The impact of hidden relay on rDCF

RBAR. As the CTL of flow 1 increases, the delay of flow 1 under rDCF and rDCF (Sensing) increases and becomes close to that under RBAR. Since N_4 and N_1 can hear from each other, they compete the medium access together. As a result, as the CTL of flow 1 increases, N_1 takes more time to contend the medium. From the figure, we can also see that the delay of flow 1 under rDCF (Sensing) is almost the same as that under rDCF, which shows that the impact of hidden relay on the delay of flow 1 is almost negligible.

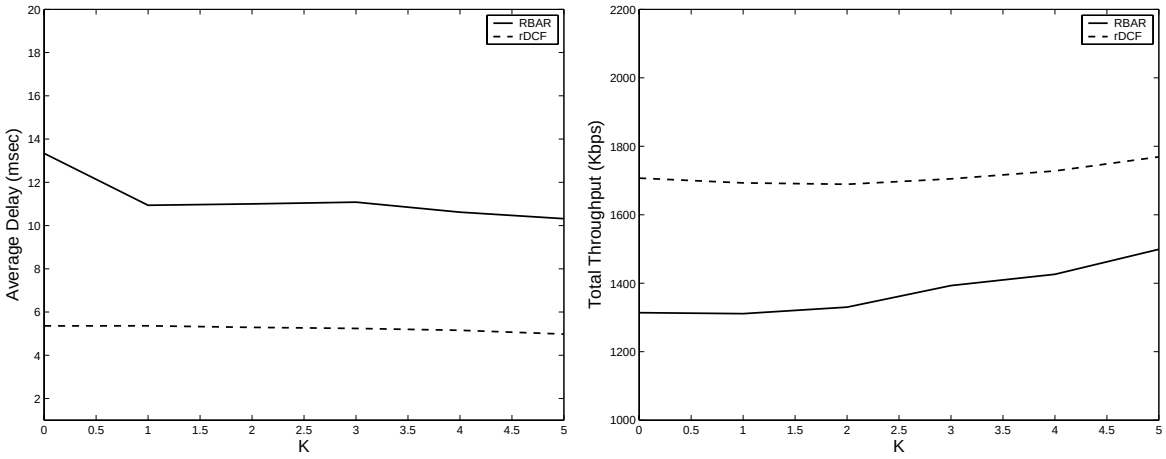
We then examine the impact of CTL on the throughput of flow 1. The rate of flow 1 is set be 1600 Kbps. As shown in Figure 4.20 (b), the throughput of flow 1 under rDCF and rDCF (Sensing) is constantly greater than that under RBAR. Only when the CTL of flow 1 is high (say more than 50%), we can see the difference between rDCF and rDCF (Sensing). As expected in Section 4.6.3.2, due to collisions caused by N_4, the throughput of flow 1 under rDCF-S is less than that under rDCF. However, we can see

that the throughput difference is small, which shows that the impact of hidden relay on the throughput of flow 1 is not a big issue.

4.7.2.3 Fully Connected Topology

We study the performance of rDCF in the fully connected topology in which all nodes can hear from each other. We put 20 nodes in the area (220m $\times$ 220m). Among them, 10 nodes act as either the sender or the receiver of the five flow. The long-term average channel condition between the sender and the receiver of each flow can only support 2 Mbps. The remaining 10 nodes are randomly distributed in the area. We use the Ricean propagation model to emulate the dynamic channel condition and evaluate the impacts of the line-of-sight parameter K and the mobility.

Impact of K: The channel condition could be quite dynamic due to various factors. One important factor is the line-of-sight parameter K. A large K means a good channel quality while a small K means a poor channel quality. We first set the rate of each flow to be 160 Kbps and evaluate the packet delay under rDCF and RBAR. As shown in Figure 4.21 (a), the delay under rDCF is much smaller than that under RBAR and the impact of K on rDCF is smaller than that on RBAR. We then evaluate the system throughput under rDCF and RBAR by letting all the flows be always backlogged. As shown in Figure 4.21 (b), under rDCF and RBAR, the system throughput increases as K increases, since the system-wide channel condition becomes better when K is larger. Compared to RBAR, rDCF can have much higher system throughput (at least 25% more). The performance gain is mainly due to the high transmission rate achieved by the MAC layer relay.



(a) Delay (b) Throughput

Fig. 4.21. The performance comparison between RBAR and rDCF under different K

After looking at the throughput of each flow, we found that the impact of channel errors on fairness can be significantly reduced by rDCF. Figure 4.22 shows the throughput of each flow when K=0. As can be seen, under RBAR, the throughput of flow 3 and flow 5 is much less than that of flow 1, flow 2 and flow 4. The reason is that the distances between the sender and the receiver of flow 3 and flow 5 are longer than those of other flows. As a result, the accumulated time period when the channel condition is poor becomes larger, which causes more packets of flow 3 and flow 5 being lost due to channel errors. Consequently, due to the binary exponentially backoff, the accumulated backoff time of flow 3 and flow 5 becomes more than other flows. However, as shown in the figure, this unfairness does not exist under rDCF. The reason is that most of packets of flow 3 and flow 5 can be delivered via relay, where both the channel conditions between the sender and the relay node and between the relay node and the receiver are

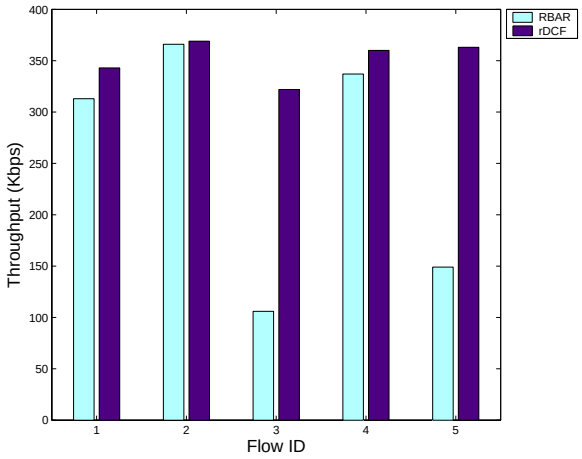
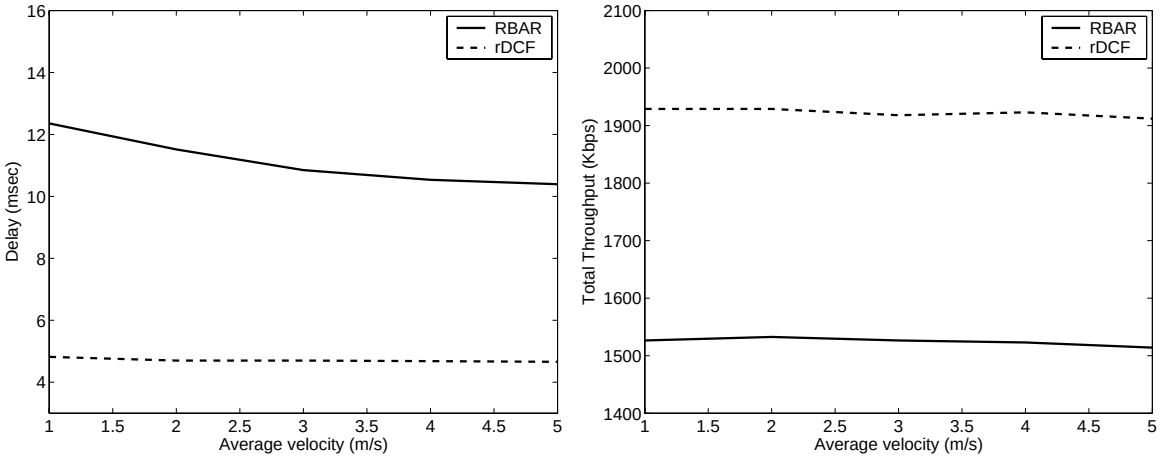


Fig. 4.22. The fairness comparison between RBAR and rDCF

more stable than the direct link. As a result, the number of transmission failures due to channel errors is significantly reduced by using relay.

Impact of Mobility: Mobility affects the channel condition in two ways. First, it changes the node's location which may affect the value of K and the strength of the received signal strength. Second, due to Doppler shift in frequency of the received signal, it may reduce the channel coherence time period. We evaluate the impact of mobility on the performance of rDCF. Each receiver of a flow moves to the sender until the distance is equal to 150m, and then moves back to its original position. Similar to [55], the value of K is fixed to be 5. As shown in Figure 4.23 (a), the delay under rDCF and RBAR decreases as the mean moving speed increases. This can be explained as follows: as the moving speed increases, the receiver may have more chances to move closer to the sender, which makes the average channel quality between the sender and the receiver better. With relay, rDCF outperforms RBAR because it can have higher transmission rate when the sender and the receiver are far away from each other. With



(a) Delay (b) Throughput

Fig. 4.23. The performance comparison between RBAR and rDCF under different velocities

the same reason, as shown in Figure 4.23 (b), rDCF obtains more benefit from mobility than RBAR.

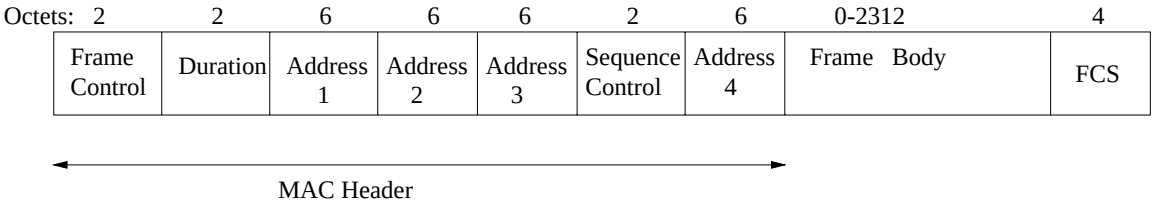
4.7.2.4 Multi-hop Topology

We evaluate the performance of rDCF under multi-hop topology in which 30 nodes are randomly distributed in a rectangular area of 1000m $\times$ 400m. All nodes are assumed to move around the area randomly and the mean moving speed is 3 m/s. The line-of-sight parameter K is set to be 5. We simulate a single flow in the system and the routing protocol is the dynamic source routing (DSR) [33]. The throughput of the flow is shown in Table 4.4. As can be seen, compared to RBAR, rDCF has significantly shorter delay since the impact of channel error has been largely relieved via relay. With the same reason, rDCF also achieves much better throughput than RBAR.

	Delay (msec)	Throughput (Kbps)
RBAR	92.1	270.3
rDCF	17.5	387.5

Table 4.4. The performance comparison under the multi-hop topology

4.8 Implementation Issues



(a) The general packet frame format

In this section, we describe how rPCF and rDCF can be incorporated into IEEE 802.11. The general IEEE 802.11 frame format is shown in Figure 4.8. As shown in Figure 4.24, the frame control field is used to carry frame identification information such as the type of the frame (e.g. control or data), the protocol version, and the power/order management information. The duration field contains the time needed for the packet transfer; the four address fields can be used to indicate the BSS identifier (BSSID), source address (SA), destination address (DA), transmitting station address, and receiving station address. These addresses may appear in different order in different type of frames. The sequence control is a sequence number that is used to detect duplicate frames. The FCS is the frame check sequence. The standard formats of Data and ACK frames are shown

Bits:	2	2	4	1	1	1	1	1	1	1	1
	Protocol Version	Type	Subtype	To DS	From DS	More Flag	Retry	Pwr Mgt	More Data	WEP	Order

(b) The frame control field format

Fig. 4.24. The general 802.11 MAC frame formats

In order to support rPCF and $rDCF$, some minor modifications to the standard 802.11 frames are required. Each data frame uses all four address fields in the order of SA, DA, BSSID, and RA (relay address). The first and second hop relay can be differentiated by the subtype value in the frame control field. For data type, which is indicated by type value of 10 (binary), subtype value can be selected from the reserved ones from 1000 to 1111. In order to identify the piggybacked transmission rates, we append an 8-bit *rate tag* to the frame. The tag is divided into two 4-bit fields, which can be used to represent two transmission rates. FOr rPCF, in order to collect and estimate the channel condition of its neighbors, each station needs to know the address of the packet sender of each packet transmitted. Thus, the ACK frame format has been changed to include the source address. Since many functions of DCF (e.g. RTS/CTS, packet fragmentation, beacon scanning) are implemented in firmware [31], these modifications can be easily done.

4.9 Related Work

Kamerman and Monteban [35] designed the auto rate fallback (ARF) protocol to utilize the multi-rate feature of IEEE 802.11. In ARF, the sender adapts the rate of each

data transmission based on the history of previous successful transmissions. Since ARF is a sender-initiated protocol, it does not work well when the channel condition becomes quite unstable. Holland *el al.* [28] proposed a receiver-based auto rate (RBAR) protocol. With the rate feedback by the receiver, RBAR can adapt the channel condition more promptly than ARF. Later, the opportunistic auto rate (OAR) scheme was proposed in [55]. OAR utilizes the fragment burst in IEEE 802.11 [30], which allows more than one packets to be transmitted when the sender is granted medium access. OAR outperforms RBAR only when the channel condition between the sender and the receiver can support a high transmission rate (say 11 Mbps). ARF, RBAR and OAR only consider the channel quality between the sender and the receiver. When the channel quality between the sender and the receiver is poor, the performance of these schemes would be significantly degraded.

The channel quality has been used as a metric for route selection in some routing protocols [4, 16, 22, 57]. A path with overall best channel condition is selected to improve the end-to-end throughput [4, 16, 57] or power efficiency [22]. However, compared to MAC layer relay, network layer relay has higher control overhead and may incur a long queuing delay. When the channel condition changes frequently, due to the slow response of the routing protocols, network layer relay cannot react quickly to exploit the opportunities to deliver data at a high transmission rate.

Recently, the concept of relay has been used in heterogeneous networks to improve system performance. *i*CAR [67] makes use of cellular and ad hoc techniques reduce call blocking probability. With the pre-deployed infrastructures, traffic can be diverted from heavily-loaded cells to lightly-loaded cells. UCAN [46] is another heterogeneous

network architecture which focuses on improve data throughput of a single cell without incurring additional operation costs. When the node is far away from the base station, its downlink data could be delivered much faster by multi-hop IEEE 802.11 relay. These two schemes do the relay at network layer. However, the performance of network layer relay forwarding may be significantly effected when the relay node is congested. Also, it would increase the workload of each relay node.

Chapter 5

Conclusions and Future Work

In this chapter, we summarize our contributions, and point out directions for future research.

5.1 Conclusions

In this thesis, several novel algorithms and protocols are proposed for power-aware and QoS-aware resource management in wireless networks. To reduce the power consumption without losing QoS provision, we have designed two novel scheduling schemes: the priority-based bulk scheduling (PBS) and the rate-based bulk scheduling (RBS), where PBS is designed for power-adaptive systems and RBS is for rate-adaptive systems. The PBS scheduler decides to serve or suspend a flow based on the amount of the prefetched data. Among the flows with insufficient prefetched data, one flow is selected as the primary flow and the others are secondary flows. The primary flow is exclusively served provided that the deadlines of the secondary flows will not be violated. The flow with enough buffered data will be suspended until the buffered data runs out. As some flows do not have strict delay requirement, the PBS scheduler may let these flows yield the channel for a while if the workload of the system is high. With prefetched data, the WNI can sleep long enough to offset the impact of the state transition delay. By suspending some flows, other active flows sharing the channel can obtain more bandwidth

and take less time to fill the buffer. RBS applies a different scheduling algorithm with the emphases on balancing power efficiency and error resiliency. The RBS scheduler provides temporal fairness to each active flow and the flow with insufficient buffered data will be served with the start-time first queuing service discipline, whereas the flow with enough buffered data will be suspended until the buffered data runs out. A novel adaptive scheme has been proposed to enhance the error-resilience of RBS by adjusting the sleep time of the WNI according to the channel condition of the flow. Analysis has been given to prove that the proposed service models have delay guarantee and are more power efficient than any other rate-based fair queuing service model. Extensive experiments are carried out to evaluate the proposed methodology. Compared to the non-work-conserving virtual clock (NVC) scheduling and the bulk slot (BKS) scheduling, the PBS service model can significantly reduce the power consumption and provide excellent QoS.

For the sake of providing fair service to flows with different service weights, the absence compensation fair queuing (ACFQ) model has been proposed so that the service differentiation under bursty data traffic can be significantly improved, while the fairness among all flows is still well maintained. ACFQ achieves QoS provision on the basis of fair queuing, and improves the service differentiation by using the virtual packet to quantify the amount of service loss/gain due to absence. After the losing flow is backlogged again, it can reclaim the service loss from other gaining flows. The extent of the compensation is gracefully controlled according to the service credit and the service weight of the flow. We also considered channel errors and extended the ACFQ model with a separate error compensation model. By exploiting the channel condition of each flow, the ACFQ

scheduler can opportunistically balance the tradeoff between the constraint of fairness and the system throughput. We showed the analytical properties of the ACFQ model, and used simulation to evaluate the performance of the model. Simulation results showed that ACFQ can significantly improve service differentiation without loss of QoS provision.

Considering the multi-rate capability of IEEE 802.11 WLANs, we have designed new protocols that support the MAC-layer relay mechanism for both point coordination function and distributed coordination function in WLANs and ad hoc networks. With these protocols, when the transmission rate of a direct link is low, the data can be transmitted much faster through a relay node provided that both links from the sender to the relay node and from the relay node to the receiver can support a high data rate. The basic relay-enabled MAC protocols have been proposed to help the sender, the relay node and the receiver coordinate to decide what data rate to use and whether to use a relay node. Regarding the bandwidth utilization, the dynamic nature of wireless channels and some variable transmission range, we also have proposed several techniques to enhance the performance of the basic protocol. Simulation results showed that our scheme outperforms the state-of-the-art protocols in terms of throughput and delay in various scenarios.

5.2 Future Work

In the context of my thesis work, our future work includes, but is not limited to:

- Designing an adaptive scheme for the PBS service model so that the WNI can wake up a little bit earlier when the channel condition is very dynamic. In this way, we can achieve a good balance between power efficiency and error-resiliency in PBS.

Furthermore, we can extend the PBS scheme and the RBS scheme to wireless ad hoc networks.

- Studying the performance of the ACFQ model under mixed data traffic (i.e., both bursty and continuous traffic) and analyze the impacts of the virtual packet length. It could be interesting to see whether there is a good way to adaptively adjust the virtual packet length according to different network conditions.

- Studying new approaches to further reduce the control overhead of the protocols and improve the performance of rate adaption. Since it takes some power for the relay node to process and forward the packet, power consumption should be an important consideration to refine the rPCF and rDCF protocols.

Beyond my current research, I plan to focus on some important and challenging issues in heterogeneous wireless network, because integrating a number of wireless techniques is a promising way to build the next generation wireless networks. In contrast to various proposed vertical handoff schemes, I plan to address issues on how to support efficient data services in heterogeneous environments. The proposed techniques would cover issues related to data prefetch, quality-of-service (QoS), fairness and medium access control protocols. With the aid of these techniques, we can achieve high bandwidth utilization and system throughput, seamless vertical handoff, good QoS provision, fairness, and robustness.

References

[1] 3GPP. http://www.3gpp.org.

[2] 3GPP2. http://www.3gpp2.org.

[3] M. S. Alouini and A. Goldsmith. Adaptive modulation over nakagami fading channels. *Wireless Personal Communications*, pages 119–143, May 2000.

[4] B. Awerbuch, D. Holmer, and H. Rubens. High throughput route selection in multirate ad hoc wireless netwo rks, wireless on-demand network systems. *Invited for for a possible publication on Kluwer MONET, Special Issue on "Internet Wireless Access: 802.11 and Beyond"*, Jan. 2004.

[5] L. Bao and J.J. Garcia-Luna-Aceves. Mac reliable broadcast in ad hoc networks. *IEEE MilCom 2001*, 2001.

[6] P. Bender, P. Black, M. Grob, R. Padovani, N. Nagabhushana, and A. Viterbi. Cdma/hdr: A bandwidth-efficient high-speed wireless data service for momadic users. *IEEE Communication Magazine*, July 2000.

[7] J. Bennett and H. Zhang. Wf2q: Worst-case fair weighted fair queueing. *IEEE INFOCOM'96*, March 1996.

[8] P. Bhagwat, P. Bhattacharya, A. Krishma, and S. Tripathi. Enhancing throughput over wireless lans using channel state dependent packet scheduling. *IEEE INFOCOM'96*, March 1996.

[9] V. Bharghavan, A. J. Demers, S. Shenker, and L. Zhang. A media access protocol for wireless lans. *ACM Sigcomm 1994*, pages 212–225, 1994.

[10] G. Bianchi. Performance analysis of the ieee 802.11 distributed coordination function. *IEEE Journal on Selected Areas in Communications*, pages 535–547, March 2000.

[11] L. Buttyan and J. P. Hubaux. Stimulating cooperation in self-organizing mobile ad hoc networks. *ACM/Kluwer Mobile Networks and Applications (MONET)*, Oct. 2003.

[12] S. Chandra and A. Vahdat. Application-specific network management for energy-aware streaming of popular multimedia formats. *The Proceedings of USENIX Annual Technical Conference*, 2002.

[13] B. Chen, K. Jamieson, H. Balakrishnan, and R. Morris. An energy-efficient coordination algorithm for topology maintaince in ad hoc wireless networks. *ACM Mobicom'01*, 2001.

[14] J. C. Chen, K. M. Sivalingam, P. Agrawal, and R. Acharya. Scheduling multimedia services in a low-power mac for wireless and mobile atm networks. *IEEE/ACM Transactions on Multimedia*, 1(2), June 1999.

[15] J.-C. Chen and T. Zhang. *IP-Based Next Generation Wireless Networks: Systems, Architectures and Protocols*. John Wiley & Sons, Inc. Hoboken, New Jersey, 2003.

[16] D. S. J. De Couto, D. Aguayo, J. Bicket, and R. Morris. A high-throughput path metric for multi-hop wireless routing. *ACM Mobicom 2003*, 2003.

[17] A. Demers, S. Keshav, and S. Shenkar. Analysis and simulation of a fair queuing algorithm. *ACM SIGCOMM'89*, Sept. 1989.

[18] C. Dovrolis, D. Stiliadis, and P. Ramanathan. Proportional differentiated services: Delay differentiation and packet scheduling. *ACM SIGCOMM'99*, pages 109–120, Sept 1999.

[19] F. H. Fitzek and M. Reisslein. A prefetching protocol for continuous media streaming in wireless environments. *IEEE Jounal on Selected Areas in Communications*, 19(10), October 2001.

[20] Jim Geier. Wireless lans, 2nd edition. *Sams Publishing, Indiana*, 2002.

[21] S. Golestani. A self-clocked fair queueing scheme for broadband applications. *IEEE INFOCOM'94*, June 1994.

[22] J. Gomez, A. T. Campbell, M. Naghshineh, and C. Bisdikian. Conserving transmission power in wireless ad hoc networks. *IEEE ICNP'01*, 2001.

[23] D. Goodman. *Wireless Personal Communication Systems*. Addison-Wesley Publishing Company, Reading, MA, 1997.

[24] D. Goodman and N. Mandayam. Power control for wireless data. *IEEE Personal Communications*, 7, April 2000.

[25] P. Goyal, H. Vin, and H. Cheng. Start-time fair queuing: A scheduling algorithm for integrated services packet switching networks. *ACM SIGCOMM'96*, August 1996.

[26] IEEE Work Group. Draft supplement to standard for telecommunications and information exchange between systems-lan/man specific requirements- part 11: Wireless lan medium access control (mac) and physical layer (phy) specifications: Medium access control (mac) enhancements for quality of serice (qos). *IEEE 802.11e Draft 3.1*, May 2002.

[27] VINT group. Ucb/lbnl/vint network simulator – ns (version 2). *http://mash.cs.berkeley.edu/ns*.

[28] G. Holland, N. Vaidya, and P. Bahl. A rate-adaptive mac protocol for multi-hop wireless networks. *ACM Mobicom 2001*, July 2001.

[29] Y. Hu, A. Perrig, and D. B. Johnson. Ariadne: A secure on-demand routing protocol for ad hoc networks. *ACM MobiComm 2003*, 2002.

[30] IEEE. Wireless lan medium access control (mac) and physical layer (phy) spec. *IEEE 802.11 standard*, 1999.

[31] Agere Systems Inc. Wavelan 802.11b chipset for standard form factors. *http://www.agere.com/client/docs/PB03025.pdf*.

[32] Z. Jiang, L. Chang, and N. Shankaranarayanan. Providing multiple service classes for bursty data traffic in cellular networks. *IEEE INFOCOM'00*, March 2000.

[33] D. Johnson and D. Maltz. Dynamic source routing in ad hoc wireless network. *Mobile Computing*, pages 153–181, 1996.

[34] E. Jung and N. Vaidya. A power control mac protocol for ad hoc networks. *ACM MobiCom'02*, 2002.

[35] A. Kamerman and L. Monteban. Wlan-ii: A high-performance wireless lan for the unlicecsed band. *Bell Labs Technical Journals*, summer 1997.

[36] R. Krashinsky and H. Balakrishnan. Minimizing energy for wireless web access with bounded shutdown. *ACM MobiCom'02*, 2002.

[37] J. Kurose and K. W. Rose. *Computer networking: A top-down approach featuring the Internet, 2nd Edition.* Addison-Wesley Publishers, 2002.

[38] UCLA Parallel Computing Lab. Glomosim. *http://pcl.cs.ucla.edu/projects/glomosim/.*

[39] Q. Li, J. Aslam, and D. Rus. Online power-aware routing in wireless ad-hoc networks. *ACM MobiCOM'01*, July 2001.

[40] H. Lin, M. Chatterjee, S. K. Das, and K. Basu. Arc: An integrated admission and rate control framework for cdma data networks based on non-cooperative games. *ACM Mobicom 2003*, 2003.

[41] Y.-B. Lin and I. Chlamtac. *Wireless and Mobile Network Architectures.* Wiley, New York, 2001.

[42] X. Liu, E. K. P. Chong, and N. B. Shroff. Transmission scheduling for efficient wireless utilization. *IEEE INFOCOM'01*, 2001.

[43] Y. Liu, S. Gruhl, and E. Knightly. Wcfq: an opportunistic wireless scheduler with statistical fairness bounds. *IEEE Transactions on Wireless Communication*, 2(5), September 2003.

[44] Y. Liu and E. Knightly. Opportunistic fair scheduling over multiple wireless channels. *IEEE INFOCOM'03*, 2003.

[45] S. Lu, T. Nandagopal, and V. Bharghavan. A wireless fair service algorithm for packet cellular networks. *MobiCom'98*, October 1998.

[46] H. Luo, R. Ramjee, P. Sinha, L. Li, and S. Lu. Ucan: A unified cellular and ad-hoc network architecture. *ACM MobiCom 2003*, 2003.

[47] T.S.E. Ng, I. Stoica, and H. Zhang. Packet fair queueing algorithms for wireless networks with location-dependent errors. *IEEE INFOCOM'98*, March 1998.

[48] MPEG Org. Mp3 test streams. *http://www.mpeg.org/MPEG/mp3.html*.

[49] A. Parekh and R. Gallager. A generalized processor sharing approach to flow control in integrated services networks: single node case. *IEEE/ACM Transactions on Networking*, 1(3):344–357, June 1993.

[50] R. Powers. Batteries for low power electronics. *Proceedings of the IEEE*, 83(4):687–693, April 1995.

[51] B. Prabhakar, E. Uysal-Biyikoglu, and A. El Gamal. Energy-efficient transmission over a wireless link via lazy packet scheduling. *IEEE INFOCOM'01*, March 2001.

[52] R. Punnoose, P. Nikitin, and D. Stancil. Efficient simulation of ricean fading within a packet simulator. *IEEE VTC 2000*, pages 764–767, 2000.

[53] D. Qiao, S. Choi, A. Soomro, and K. G Shin. Energy-efficient pcf operation of ieee 802.11a wireless lan. *IEEE INFOCOM'02*, June 2002.

[54] Theodore S. Rappaport. Wireless communications: Principle and practice. *Prentice Hall, New Jersey*, 1996.

[55] B. Sadeghi, V. Kanodia, A. Sabharwal, and E. Knightly. Opportunistic media access for multirate ad hoc networks. *ACM Mobicom 2002*, July 2001.

[56] N. B. Salem, L. Buttyan, J. P. Hubaux, and M. Jakobsson. A charging and rewarding scheme for packet forwarding in multi-hopo cellular networks. *ACM MobiHoc 2003*, 2003.

[57] Y. Seok, Jaewoo Park, and Yanghee Choi. Multi-rate aware routing protocol for mobile ad hoc networks. *IEEE VTC'03 Spring*, 2003.

[58] S. Shakkottai and R. Srikant. Threshold-based dynamic replication in large-scale video-on-demand systems. *Wireless Networks*, 8:13–26, 2002.

[59] T. Simunic and S. Boyd. Managing power consumption in networks on chips. *ACM DATE'02*, 2002.

[60] S. Sinha and C.S. Raghavendra. Pamas: Power aware multiple access protocol with signaling for ad hoc networks. *Computing Communication Review*, 28(3):5–26, 1998.

[61] M. Stemm and R. H. Katz. Measuring and reducing energy consumption of network interfaces in handheld devices. *IEICE Transactions on Communications*, E80-B(8), August 1997.

[62] D. Stiliadis and A. Varma. Latency-rate servers: a general model for analysis of traffic scheduling algorithms. *IEEE/ACM transactions on networking*, 6, Oct. 1998.

[63] V. Vanghi, A. Damnjanovic, and B. Vojcic. The cdma2000 system for mobile communications. *Prentice Hall, Upper Saddle River, NJ*, 2004.

[64] B. H. Walke. *Mobile Radio Networks: Networking, Protocols and Tranffic Performance, Second Edition*. John Wiley & Sons, Ltd, 2002.

[65] A. Wang, S. Cho, C. G. Sodini, and P. Chandrakasan. Energy efficient modulation and mac for asymmetric rf microsensor system. *International Symposium on Low Power Electronics and Design*, 2001.

[66] H. S. Wang and N. Moayeri. Finite-state markov channel-a useful model for radio communication channels. *IEEE Transactions on Vehicular Technology*, 44(1):163–171, Feb. 1995.

[67] H. Wu, C. Qiao, S. De, and O. Tonguz. Integrated cellular and ad hoc relaying systems: icar. *IEEE journal on selected areas in communications (JSAC)*, pages 2105–2115, Oct. 2001.

[68] J. Xu and R. J. Lipton. On fundamental tradeoff between delay bounds and computational complexity in packet scheduling algorithm. *ACM SIGCOMM'02*, August 2002.

[69] Y. Xu, J. Heidemann, and D. Estrin. Geography-informed energy conservation for ad hoc routing. *ACM MobiCom'01*, July 2001.

[70] W. Yuan and K. Nahrstedt. Buffering approach for energy saving in video sensors. *IEEE International Conference on Multimedia and Expo (ICME)*, 2003.

[71] H. Zhang. Service disciplines for guaranteed performance service in packet-switching networks. *Proceedings of the IEEE*, 83(10), October 1995.

[72] H. Y. Zhang, Y. Ge, C. J. Hou, and L. Sha. Energy-efficient real-time scheduling in ieee 802.11 wireless lans. *IEEE ICDCS'03*, 2003.

[73] L. Zhang. Virtual clock: a new traffic control algorithm for packet switching networks. *ACM SIGCOMM'90*, Sept. 1990.

[74] H. Zhu, G. Cao, G. Kesidis, and C. Das. An adaptive power-conserving service discipline for bluetooth. *IEEE ICC'02*, 2002.

Vita

Hao Zhu was born in Xian, China on November 12, 1974. In 1997 he received the B.E. degree in computer science and engineering from the Xian Jiaotong University . He received his M.S. degree in computer science from the Institute of Software, Chinese Academy of Sciences in 2000. He started to pursue his Ph.D degree in computer science and engineering at the Pennsylvania State University in 2000 and finished his Ph.D program in 2004. His research interests include resource management in wireless networks. Hao Zhu is a student member of IEEE.