Performance Evaluation of Biased Queue Management

Except where reference is made to the work of others, the work described in this thesis is my own or was done in collaboration with my advisory committee. This thesis does not include proprietary or classified information.

Xiaoming Li

Certificate of Approval:

Kai Chang
Professor
Department of Computer Science and Software Engineering

Saad Biaz, Chair
Assistant Professor
Department of Computer Science and Software Engineering

Cheryl Seals
Assistant Professor
Department of Computer Science and Software Engineering

Stephen L. McFarland
Acting Dean
Graduate School

Performance Evaluation of Biased Queue Management

Xiaoming Li

A Thesis

Submitted to

the Graduate Faculty of

Auburn University

in Partial Fulfillment of the

Requirements for the

Degree of

Master of Science

Auburn, Alabama
August 7, 2006

Performance Evaluation of Biased Queue Management

Xiaoming Li

Signature of Author

Date of Graduation

Vita

Xiaoming Li was born on Jan.12, 1982 in Hangzhou, a beautiful southern city in China. She graduated from Hangzhou Xuejun Middle School in 2000. Then, she attended Zhejiang University in Hangzhou, which is said to be the heaven in China. She got her Bachelor of Engineering Degree with honor from Chu kechen Honors College and Department of Computer Science, Zhejiang University in 2004. After that, she came to Auburn to continue her graduate study in Department of Computer Science and Software Engineering, Auburn University.

Thesis Abstract

Performance Evaluation of Biased Queue Management

Xiaoming Li

Master of Science, August 7, 2006
(B.S., Zhejiang University, 2004)

59 Typed Pages

Directed by Saad Biaz

Congestion is an important issue which researchers focus on in the Transmission Control Protocol (*TCP*) network environment. To keep the stability of the whole network, congestion control algorithms have been extensively studied. Queue management method employed by the routers is one of the important issues in the congestion control study.

Biased Queue Management (*BQM*) is a queue management method proposed in [1], and consists of an accurate packet loss discriminator and an implicit continuous congestion level measure. We can apply BQM in both wired and wireless circumstance. Adding the accurate packet loss discriminator into BQM, we can easily distinguish congestion losses and wireless random losses (*we refer to any loss unrelated to congestion as wireless loss*), and determine when to adjust the congestion window size to slow down the output. Furthermore, with continuous congestion level measure, BQM can relieve the congestion and improve the performance of queue method in TCP environment. Moreover, we will propose a new scheme (*Modified BQM*) for the accurate loss discriminator to improve BQM in some respects of congestion control in ad hoc network.

In order to show the improvement achieved by BQM and modified BQM, we will compare BQM with a popular queue management method, Droptail, in different aspects, such as throughput, packet loss rate and fairness. We observe the performance of BQM in both wired and wireless TCP network environment.

The comparison results indicate BQM has better throughput, higher fairness and lower packet loss rate than Droptail. BQM is a highly efficient queue management technique in congested TCP network environment, while modified BQM can get better throughput than normal BQM. Experiments are performed by Network Simulator (*NS*) from Lawrence Berkeley Labs with wireless extension from CMU.

Acknowledgments

I would like to express my sincere appreciation to my advisor Dr. Saad Biaz for his guidance and support throughout this work. With his great help, I can complete this work finally. Throughout my research and thesis writing period, he provided encouragement, sound advice, good teaching, and lots of good ideas.

I am also grateful for the contributions of Dr. Kai Chang and Dr. Cheryl Seals, both as members of my thesis committee and as great teachers. I learned a lot from them throughout the classes and their research attitude.

I would like to give my special thanks to my parents for their understanding, endless patience and encouragement.

Style manual or journal used Journal of Approximation Theory (together with the style known as "aums"). Bibliograpy follows van Leunen's *A Handbook for Scholars*.

Computer software used The document preparation package TEX (specifically LATEX) together with the departmental style-file `aums.sty`.

Table of Contents

List of Figures

Chapter 1

Introduction

When there are too many coming packets contending for the limited shared resources, such as the queue buffer in the router and the outgoing bandwidth, congestion may happen in the data communication. During the congestion, large amounts of packet experience delay or even be dropped due to the queue overflow. Severe congestion problems may result in degradation of the throughput and large packet loss rate. Congestion will also decrease efficiency and reliability of the whole network. Furthermore, if at very high traffic, performance collapses completely and almost no packet is delivered.

As a result, many congestion control methods are proposed to solve this problem and avoid the damage. Most of the congestion control algorithms are based on evaluating the network feedbacks to detect when and where congestion occurs, and take actions to adjust the output source, such as reduce the congestion window ($cwnd$). Various feedbacks are used in the congestion detection and analysis. However, there are mainly two categories: explicit feedback and implicit feedback. In explicit feedback algorithms, some signal packets are sent back from the congestion point to warn the source to slow down, while in the implicit feedback algorithms, the source deduces the congestion existence by observing the change of some network factors, such as delay, throughput difference and packet loss.

We will mainly compare two queue management methods in this report: Biased Queue Management (BQM) and Droptail. Furthermore, we will also propose a new

scheme *Modified BQM* that modified the normal BQM slightly in order to evaluate the congestion level in the network more exactly.

Droptail queuing method is by far the simplest approach to router queue management. The router accepts and forwards all the packets that arrive as long as its buffer space is available for the incoming packets. If a packet arrives and the queue is currently full, the incoming packet will be dropped. The sender eventually detects the packet lost and shrinks its sending window. Droptail is the most widely used queue management algorithm due to its simple implementation and relatively high efficiency. However, droptail has some weakness, such as the bad fairness sharing among TCP connections, and the throughput and link efficiency suffer severe degradation if congestion is getting worse.

Biased Queue Management (BQM) is an implicit feedback algorithm based on the packet losses, and proposed for improving the performance of TCP in both wired networks and wireless ad hoc networks. Accurate losses discriminator (ALD) and continuous congestion measure (CCM) are two important parts of BQM design.

In wired network, there is rarely random losses, and the losses are mainly due to congestion. The CCM based on BQM evaluates the congestion level in the network bottleneck with an implicit loss-based continuous congestion measure and adjusts the output window size respectively.

However, there are both congestion losses and random losses due to link failures in the Wireless Ad Hoc Networks, and the congestion status is not stable. Based on the accurate loss discriminator and implicit continuous congestion measure, the ALD based on BQM can distinguish the types of losses and respond differently according to each type of losses and the congestion level measures. As a result, BQM can be considered

as an efficient congestion control algorithm for the wireless ad hoc network. The ALD based on modified BQM is a bit different from the above. Modified ALD is able to respond a more exact congestion measure with distinguishing some "jump" losses.

In this report, we will compare the performance of BQM with Droptail in wired network, and also compare both BQM and Modified BQM with Droptail in wireless ad hoc network. We will evaluate their performance with throughput, fairness, and packet loss rate of the these mechanisms which we will give the definitions in this chapter. The following are some specific definitions of the Key words.

Throughput: The measure of how soon the receiver is able to get a certain amount of data. It is determined as the ratio of the total data received by the end to the connection time. Throughput is an important factor which directly impacts the network performance.

Fairness Index: The measure of whether each TCP connection gets a fair share. Fairness Index (*FI*)is computed as follows: let T1 ... Ti... and Tn be the throughput achieved by each of the N TCP connections. Fairness Index can be expressed as:

$$FI = \frac{(\sum T_i)^2}{N \sum T_i^2} \tag{1.1}$$

Fairness Index varies from 0 to 1, and the closer is FI to 1, higher is the fairness among the TCP connections.

Packet loss rate: Defined as a ratio of the number of lost packets to the number of total transmitted packets.

The rest of the report is organized as following. Chapter II is about the concept of wireless Ad hoc Network, the type of losses in Ad hoc Network, and the design issues of efficient congestion control techniques. In Chapter III, BQM and Modified BQM are described in detail with its rationale. Chapter IV will talk about the simulation environment. Chapter V will present the simulation results in the wired network in detail. Chapter VI is about the BQM performance in the wireless ad hoc network with specific analysis. Finally, Chapter VII concludes the total work and proposes the future research.

Chapter 2

Congestion Control in Wireless Ad hoc Networks

This chapter will introduce the concept of mobile and wireless ad hoc network, talk about the congestion control mechanism in ad hoc network, and give a short glance at Biased Queue management.

2.1 Mobile Ad Hoc Network

Mobile hosts and wireless networking hardware are widely used in the world, and extensive work has been done recently in integrating these elements into traditional networks such as the Internet[1]. In the next generation of wireless communication systems, there will be a need for the rapid deployment of independent mobile users. Oftentimes, mobile users will want to communicate in some situations in which no fixed wired infrastructure is available, either because it may not be economically practical, or physically possible to provide the necessary infrastructure, or because the expediency of the situation does not permit its installation. for example, students of a class may need to communicate with each other during a lecture, friends may would like to share files when waiting in the airport, or emergency rescue workers need to be quickly deployed after an earthquake. In such situations, a collection of mobile hosts with wireless network interfaces may form a temporary network without the aid of any established infrastructure or centralized administration. This type of wireless network is known as an ad hoc network.

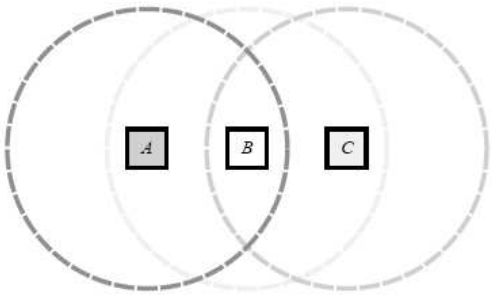

Figure 2.1: A sample ad hoc network of three mobile nodes

A mobile ad hoc network ($MANET$) is an autonomous system of independent mobile computers connected by wireless links. In such an environment, it may be necessary for one mobile host to enlist the aid of other hosts in forwarding a packet to its destination, due to the limited range of each mobile hosts wireless transmissions. Mobile computers may cooperate as hosts and/or routers temporarily from a network without any prior infrastructure or base-stations.

Since the nodes are mobile, the network topology may change rapidly and unpredictably over time. The network is decentralized, where all network activity including discovering the topology and delivering messages must be executed by the nodes themselves, i.e., routing functionality will be incorporated into mobile nodes. However, because of the unstable network topology, congestion control in mobile ad hoc network is different from normal wired network and need more consideration of some specific issues, such as type of losses.

2.2 Apply BQM in Ad Hoc Network

Despite advances in wireless technology, wireless links are still not as reliable as wired links. Moreover, node mobility makes it difficult to ensure a stable connectivity: as nodes moving, topology dynamically changes, breaking existing routes and creating new potential routes. As a result, in mobile ad hoc networks, frequent link failures and other random losses unrelated to congestion may disqualify packet loss as a reliable congestion indicator and require ad hoc congestion control algorithms that must accurately diagnose the cause of a loss and appropriately react to each type of losses.

TCP connections over mobile ad hoc networks usually have poor performance because of the lack of a mechanism to distinguish congestion losses from random wireless losses or link failures. Unable to accurately diagnose random wireless losses or link failures, TCP does not react appropriately to such kind of losses.

As a result, the key obstacles to improve TCP or design new congestion control techniques over mobile and/or wireless networks are the lack of an efficient mechanism to accurately diagnose the real cause of losses and the binary nature of current network's feedback. However, Biased Queue Management (BQM) is a novel approach proposed for solving this problem.

Biased Queue Management (BQM) marks each packet with different discard priority in order to de-randomize congestion losses. With the de-randomize the congestion losses mechanism, we can create an accurate loss discriminator to diagnose the congestion losses and wireless random losses with appropriately response to each type of failure. If the losses are due to the congestion, the packets are supposed to be dropped according to the discard priority, that means the packets with lower priorities will be dropped first.

In contrast, if a packet is dropped with higher discard priority, while the lower discard priorities packets all arrive at the receiver, it is very possible that the loss is the wireless random loss.

Furthermore, Biased Queue Management (BQM) can create an implicit packet loss based continuous congestion measure that provides an efficient congestion control algorithm. Implicit continuous congestion measure means the bottleneck router will not feedback to the source with any explicit information, and the source will measure the overload imposing on the network continuously. As a result, the source will know the congestion level in the network at this moment and adjust the congestion window relatively.

Chapter 3

Biased Queue Management

Biased Queue Management (*BQM*) is based on the fact that TCP senders mark all packets to be sent with different discard priorities, and the routers will always drop packets with the lowest discard priority first. *Biased* here is between the packets of the same flow, not between the flows. BQM mechanism is the key to the design of an *accurate loss discriminator* (ALD), and an implicit *continuous congestion measure* (CCM). In section 3.1, we will present the choice of pattern of discard priorities. We will talk about the design of an ALD based on BQM in Section 3.2, 3.3 and 3.4 with some different research and design issues. Similarly, Section 3.5 is respectively dedicated to CCM.

3.1 Discard Priority

TCP senders mark packets by using a mapping function from the packet sequence number to a set of sequence numbers [0, k-1], where k is the number of different discard priorities. With Biased Queue Management, packets with discard priority 0 are the packets to be dropped first when congestion occurs. When packets with discard priority 0 are exhausted from the queue and congestion persists, BQM starts to drop packets with discard priority 1. As congestion persists, the discard priority of dropped packets keeps increasing. The mapping function should be carefully chosen in order to avoid the successive packets getting close discard priorities.

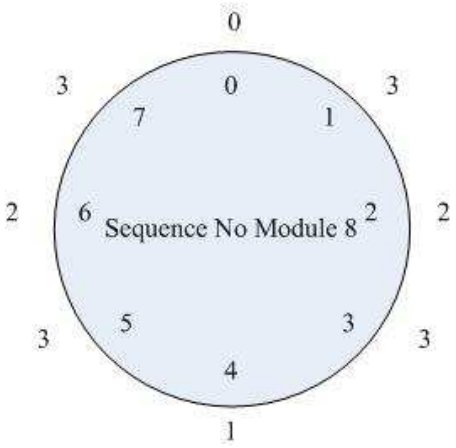

Figure 3.1: A candidate mapping pattern 8

For example, if the number of different discard priorities is 4 (k=4), that is from 0 to 3, we should choose a mapping function to map each packet into the appropriate discard priority [0, 3] in the fair distribution. Figure 3.1 is a candidate pattern that modulo the packet sequence number i with 8, and then maps the result to the discard priorities outside the circle corresponding to the number *(i%8)* placed inside the circle.

This pattern is modulo 8 because it repeats itself every batch of eight packets. We make sure that each batch contains only one packet with discard priority 0. The first reason for this is we want to make it quite unlikely that a packet with discard priority 0 is dropped randomly due to any phenomenon other than congestion. The second reason is that we want to increase the chances that packets be dropped from different flows in order to reduce the packet loss rate and improve the fairness index. Furthermore, we also choose to have only one packet with discard priority 1 in order to stagger the losses. We do not want the packets are dropped in groups in order to avoid timeouts, and staggered losses will often be recovered through fast retransmit and fast recovery.

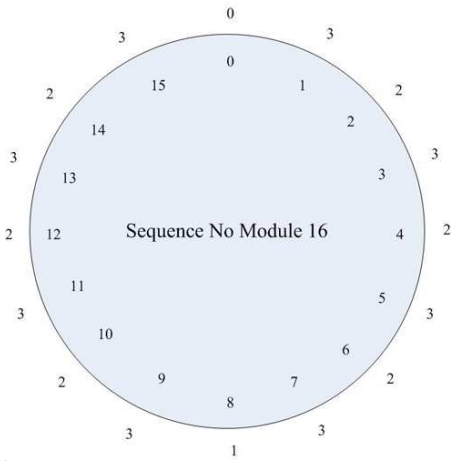

Figure 3.2: A candidate mapping pattern 16

Figure 3.2 presents the pattern with batches of 16 packets. The number of different discard priorities is also 4 (k=4), from 0 to 3, and this pattern modulo the packet sequence number *i)* with 16.

The reason for choosing such a mapping mechanism is the same as the pattern of modulo 8, that we want to increase the possibilities that the packets would be dropped from different flows. We will investigate the impact of the discard priority setting pattern and its size on the performance of BQM in the simulations.

3.2 Probability Model of ALD

TCP receiver considers the pattern of losses between the next expected packet P_next and the packet $P_highest$ with the highest sequence number received so far. Let W (W = *highest -next+1*) be the number of packets in the ordered set P_next, P_next + 1,...,$P_highest$. Suppose there are r losses among the W packets. We may consider the phenomenon of packet loss as a random sampling of r elements among a population of

W elements. The probability of drawing x packets with discard priority 0 in a random sample of size r among a population of W packets follows a hypergeometric distribution:

$$P(X = x) = \frac{\binom{\lfloor \frac{W}{k} \rfloor}{x}\binom{W - \lfloor \frac{W}{k} \rfloor}{r-x}}{\binom{W}{r}} \tag{3.1}$$

The quantity $\lfloor \frac{W}{k} \rfloor$ denotes the number of packets with discard priority 0 within the W packets. If the pattern of the r losses appears to be the result of a random sampling, then we can conclude that losses are wireless random losses. Randomness of sampling here is tested as following: if a loss pattern has a very low probability to occur, we can consider that losses are due to congestion safely. If a burst of 3 or more packets are detected, we unconditionally diagnose them as congestion losses.

As an example, suppose that we use 8 discard priorities (k=8) and that W=24. So one packet out of 8 has discard priority 0, and 3 packets of 24 have discard priority 0. There are two extreme cases: 1. when 3 packets with discard priority 0 are the only missing packets within the 24 packets(W=24). Since $P(X = 3) = \frac{1}{2024}$, it is quite unlikely that the sampling was random, so the sampling can be concluded as biased. If the sampling is probably not random, then the three losses are probably not wireless. 2. When 3 lost packets are not losses with discard priority 0. The probability of these losses are wireless is equal to 0.65, and it is a strong indication of wireless random losses. As a result, we can create an accurate loss discriminator based on these issues.

3.3 Rationale of Loss Distinguishing function

From these observations, a very simple function is developed to "summarize" the pattern of losses.

$$F(x, r, k) = 1 - \lfloor k * \frac{x}{r} \rfloor \tag{3.2}$$

Where x is the number of lost packets marked with the lowest discard priority 0 and r is the number of losses within the ordered set $P_n xt$, ..., $P_h i$. The function $F(x, r, k)$ is built upon the following rationale: if losses are wireless(random), it is expected to have $\frac{x}{r} \approx \frac{1}{k}$, otherwise, if the losses are uniformly distributed, and $F(x, r, k)$will be equal to 0. Here $\frac{x}{r}$ represents the proportion of lost packets with the lowest discard priority 0 over the total number of losses (r). If the losses are due to congestion, it is expected that the proportion $\frac{x}{r}$ of packets with the lowest discard priority 0 be higher than $\frac{1}{k}$, making $F(x, r, k)$ negative. Smaller is $F(x, r, k)$, higher is the likelihood of congestion losses. Note that 1 - $k \leq F(x, r, k) \leq 1$. The function $F(x, r, k)$ does not capture all the information that could be exacted from the pattern of losses. However, the function $F(x, r, k)$ yields a simple, robust, and accurate discriminator. Whenever r losses occur with x packets with discard priority 0, they are diagnosed as following: if $F(x, r, k) \geq 0$, then the losses are diagnosed as wireless losses, otherwise ($F(x, r, k)$ ¡ 0), they are diagnosed as congestion losses.

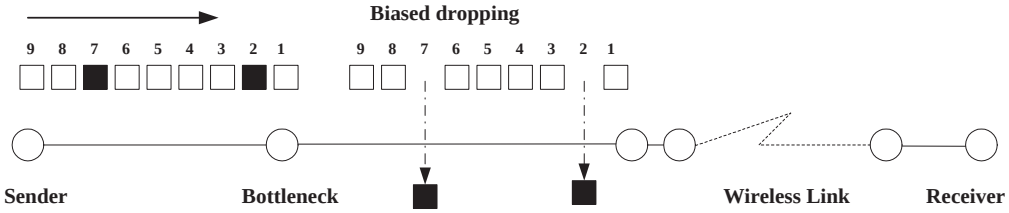

Figure 3.3: Congestion: Biased Dropping

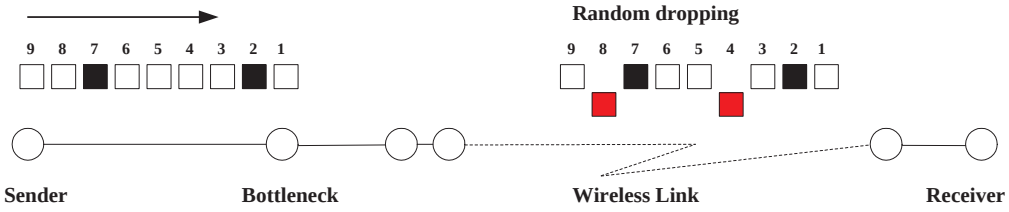

Figure 3.4: Wireless Losses: Random Dropping

3.4 Accurate Loss Discriminator (*ALD*)

We try to create an accurate loss discriminator to distinguish the congestion loss and wireless loss, and with such discriminator, we can highly improve the network performance and promote the efficiency. BQM provides us a way to create such an accurate loss discriminator. With BQM, we can easily diagnose the packet losses due to congestion according to their discard priorities. Packet losses with the lowest discard priority are high probability to be congestion losses, while if no congestion exists, packet losses with higher discard priorities have high probability to be wireless random losses.

Let's think about the scenario, a sender and a receiver with a path that contains a bottleneck link implementing BQM and a wireless link. The sender sends 9 packets with sequence number from 1 to 9 (the number below the packets in Figure 3.3 and

Figure3.4). The sender assigns discard priorities from 0 to 3 to the packets (discard priorities are above the packets in Figure 3.3 and Figure 3.4). Packets 2 and 7 (*black rectangles*) are assigned discard priority 0. If congestion happens, BQM must first drop packets with the lowest discard priority before dropping any other packet. In this case, BQM will unlikely drop other packets other than the black packets marked with the lowest discard priority 0, the packets numbered 2 and 7. If other packets are dropped, it is high probability not due to the congestion, refer to Section 3.2.

Our discriminator is based on the rationale (refer to Section 3.3), and created on the receiver. The receiver will always record the number of packets with lowest discard priority 0 among the packet losses. After diagnosing the losses with the function $F(x, r, k)$, we can respond differently to different kinds of losses. If the result of the function is larger than or equal to 0, it is very possible that it is a wireless random loss, otherwise, if the result of the function is lower than 0, we will consider these losses as congestion losses. We record the highest discard priority within theses losses and send back to the sender as the congestion measure. This algorithm can distinguish congestion losses effectively from wireless losses and improve the network congestion greatly.

3.5 Continuous Congestion Measure (*CCM*)

When congestion occurs, the bottleneck router will discard packets with the lowest discard priority 0, and if the congestion persists, the router continues to drop packets with discard priority 1, and so on. After receiving duplicate acknowledgements which notify the packet losses, the source will evaluate the congestion level by the duplicate acks. Packet loss with discard priority 0 indicates light congestion, while packet loss with discard priority 1 indicates relatively severe congestion. The source adjusts the

Table 3.1: Congestion Window Adjust

Discard Priority	Max Overload	α	β
0	W/8	7/8	1/8
1	W/4	3/4	1/4
2	W/2	1/2	1/2
3	≈W	1/4	1

congestion window size based on the different congestion level resulting from the packet losses. With BQM, the source can respond appropriately to different congestion levels.

In the modulo 8 pattern (Figure 3.1), suppose that a TCP connection has W packets in flight and losses occur. If only packets with discard priority 0 are dropped, which represent at most $\frac{1}{8}$ of the total number of packets, we can conclude that the TCP connection is imposing over the network an overload of at most $\frac{1}{8}$W.

Table 3.1 summarizes the maximum overload corresponding to the discard priorities of dropped packets. The first column is the highest discard priority of dropped packets within a window of W packets. The second column is the maximum overload imposed by the TCP connection. α represents the ratio of the decreased window to the original window, and β represents the increase coefficient of the congestion window after every successful ack of a full window.

In congestion avoidance phase[6], the congestion window size W is increased by 1 (β=1) when a window of W packets is successfully acknowledged, and W is multiplicational decreased by half(W=αW,with α=$\frac{1}{2}$) when duplicate acknowledgements received that means congestion occurs. Table 1 presents the values α and β to be used with BQM based on the discard priority of lost packets. For normal TCP, the congestion window is reduced to $\frac{W}{2}$(α=$\frac{1}{2}$) and it takes $\frac{W}{2}$ round trip times(RTTs) to reach back the previous congestion window size W before the loss. With BQM, if only packets with discard

priority 0 are dropped, then the overload is at most $\frac{1}{8}$ of the current load. Under these conditions, it is safe to decrease the congestion window to $\frac{7}{8}$W. Thus, congestion window W with BQM is drastically reduced(α=$\frac{7}{8}$) than with normal TCP. However, it will take W RTTs to reach back the initial congestion window W (β=$\frac{1}{8}$)[1]. We will investigate BQM by specific simulations in Chapter V.

With BQM, we can improve TCP responsiveness to losses and accurately "guess" the missing packets within a window. Recall the retransmission of Normal TCP, that when an out of order packet reaches a receiver, the receiver immediately sends back a duplicate acknowledgement (*dupack*) for the expected packet. TCP sender will trigger *fast retransmit* only when 3 dupacks are received to ensure that the packet is indeed lost and not simply out of order. With BQM, we argue based on a simple probability argument that if a sender receives ONE dupack for a packet with discard priority 0, then there is no need for waiting for three dupacks to trigger *fast retransmit*.

When multiple losses occur within the same window, it is difficult to identify the missing packets because of the cumulative TCP acknowledgement rule. We suppose that, with BQM, TCP sender can retransmit packets with discard priority 0 because these packets are most likely to be dropped in such case of congestion.

However, past papers typically advocate that TCP should not halve the congestion window size or adjust the slow start threshold when a packet loss is known to be due to wireless error. We advocate in the following that, even with a perfect loss discriminator, TCP should somewhat throttle its sending rate for wireless losses. TCP should decrease the congestion window size because a wireless loss signals to some extent a temporary decrease of the link layer goodput on the wireless link. Higher the wireless packet loss

[1]W RTTs to recover results from: *(full window W- αW)*RTT/β = (W-$\frac{7}{8}$W)*RTT/$\frac{1}{8}$ = W RTT*

rate, lower will be the goodput of a wireless link. This decrease may lead to queue build up and congestion drops. Therefore, TCP should at least decrease by one packet of its congestion window size for each wireless loss detected.

3.6 Improvement of BQM (*Modified BQM*)

Although BQM is a strong loss discriminator in both Ad hoc network and wired network, we still have some issues to discuss. Usually, when congestion happens, we record the highest discard priority within the losses and send back to the sender as the congestion measure. However, if there is some "jump" losses during these losses, it will be difficult to get the correct congestion measure. For example, if most of the losses are packets with discard priority 0, only 1 or 2 losses with discard priority 3, after diagnosing with the function, we can make the correct decision: the losses due to congestion, but also the wrong decision: the congestion measure is 3. Thus, we can realize the fault of BQM with "jump" losses: some high discard priority losses may due to wireless, but the discriminator consider them as congestion losses, and set the congestion measure incorrectly. Incorrect congestion measure will make the sender shrink its congestion window to an inappropriate size, and advise the throughput.

Some modification to the accurate loss discriminator is proposed based on a simple rationale: there is a gap among the "jump" losses. The router only drop packets with lowest discard priority, thus, if losses with discard priority 3 occur, there must be losses with discard priority 0,1,2 existing, otherwise, there will be a gap. For example, if most of losses with discard priority 0, only 1 loss with discard priority 3, we can discover the gap 1, 2. In such condition, we will suspect the loss with discard priority 3 is due to wireless not due to congestion, although the discriminator shows congestion losses exist

in the network. As a result, we can not simply set the congestion measure to the highest discard priority 3 among these losses, but need to traverse the whole losses, and find the highest discard priority before the gap. In this case, the highest discard priority before the gap is 0, so we should set the congestion to 0, not 3.

This modification can avoid the incorrect congestion measure with "jump" losses and useless congestion window size adjustment with "jump" losses. Modified BQM can get better performance than normal BQM.

Chapter 4

Simulation Environment

We use the Network Simulator (NS) from Lawrence Berkeley Labs to compare the performance of BQM and Droptail as our proposed schemes. NS is an extensible simulation engine built using C++ and Tcl/Tk which can simulate various flavors of TCP available today for wired networks. With the wireless extension from CMU, we can also do complex simulations in the wireless ad hoc network. We modified the original TCP-Newreno, TCP-Sink and Droptail to create BQM-Sender, BQM-Receiver, and BQM-Queue for the purpose of our simulations. Section 4.1 is about building BQM mechanism. We will present the simulation environment of wired network in Section 4.2 and wireless ad hoc network in Section 4.3.

4.1 Build BQM and Modified BQM

As NS is an extensible simulation engine, we could modify the existing TCP mechanisms to build BQM. We should create the BQM mechanism as a queue management in the network that always drops the packets with lowest discard priority. Thus, we need to modify TCP-sink to create a BQM receiver that can detect the packet losses, distinguish the congestion losses from wireless random losses, and finally send dupacks with continuous congestion measure to the TCP source. Dupacks here refers to the duplicate acknowledgement to the source which is sent by the receiver when a packet loss occurs. If there are several packets losses, the receiver will traverse all the losses, search for the packet loss with the highest discard priority, recognize the congestion measure in

the network at this moment and finally return the congestion measure in the dupacks to the BQM sender. Furthermore, we also need to add an accurate loss discriminator on the BQM receiver, that can distinguish the type of packet losses to improve the system throughput (refer to 3.3). Most of BQM and Modified BQM are same, so we only need to make a few modification on BQM to create Modified BQM. The modification to deal with "jump" losses cases is also applied on the receiver.

We should also modify TCP-Newreno to create the BQM sender which can mark packets with different discard priorities based on the chosen pattern (refer to Section 3.1). When receiving acknowledgements from the BQM receiver, the BQM sender can correctly respond to different congestion level and adjust the congestion window size appropriately.

As a queue management method, BQM also should have a queue scheduling function that always drops the packets with lowest discard priority when the queue buffer exceeds the threshold. We can modify the original queue method Droptail to achieve this goal. When there is not enough buffer room, the new queue method will go through the arriving packets, and drop the first packet with lowest discard priority.

let's describe the whole process as following: The sender outputs large amount of packets and cause congestion in the bottleneck router. The queue buffer in the router is full and packets with discard priority 0 are dropped first. However, if congestion persists, the packets with discard priority 1 will be dropped. The receiver traverses the arrival packets to detect the sequence number of the lost packets and computes the congestion level by analyzing the lost packets' discard priorities. The fact is the current congestion level will be the highest priority of the lost packets. Then the receiver will send back

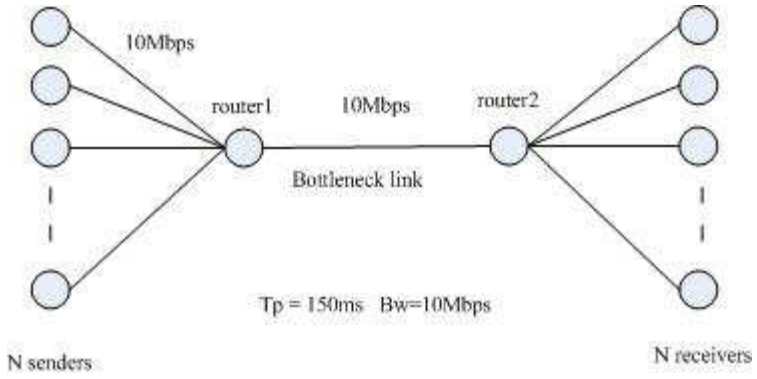

Figure 4.1: Simulation Topology of Wired Network

the congestion level in the dupacks. After getting the congestion level from dupacks, the source is able to adjust their congestion window size according to Table 3.1.

We should also notice some bad situations, for example, the permanent dead packets, which may happen to the packets with lower discard priorities. If the packets with discard priority 0 are dropped and the congestion persists, then when the resent packets arrive at the router, it has high probability that the resent packets with discard priority 0 are dropped again. As a result, every time the sender gets a dupack, it will shrink its congestion window, and the throughput is limited. So we still need to add some mechanisms into BQM that can accumulate the packet resend times and increase the discard priority to avoid permanent dead packets.

4.2 Simulation Model in Wired Network

First, we would like to compare the performance of BQM and Normal TCP (Drop-tail) in wired network. A simple network topology is chosen to make it easier to understand the wired congestion network environment. As shown in Figure 4.1, there are

n TCP connections in the network. n is a variable parameter that means how many connections share the bottleneck link. We choose n within 2, 4, 8, 16, and 32 in the simulation. The larger is the number of TCP connections, the worse is the congestion in the bottleneck. The bottleneck link's bandwidth is 10Mbps and 6Mbps (Simulation Result of Bw 6Mbps is in the Appendix A), and the n connections' output is all 10Mbps. The data transfer time (Tp) is 150ms. When the simulation starts, the congestion will occur at the bottleneck route, and we will assign BQM and Droptail separately on the bottleneck route as the queue management method.

In order to get the accurate results, we use a random mechanism to choose the experiment start time for per run per connection, which means each connection will start randomly in each run according to a start time list. The start time is random selected from 0 to 180secs. The start time list is recorded in a file, so Droptail and BQM are able to have the same simulation environment by reading the start time file, which can ensure the experiment's reliability. We get the average results of 25 runs for each simulation, and each run lasts 300s. TCP connections of Droptail have the original TCP-Newreno as their senders, TCP-Sink as their receivers, and Droptail as their queue method.

We write a C++ drive engine to collect the data from trace files of the simulation, analyze the data to get directly information, and finally output the graphs which is drawn by Matlab.

4.3 Simulation Model in Wireless Ad hoc Network

Different from BQM in wired network, BQM applied in wireless ad hoc network does not only respond to the congestion measure, but also can diagnose the type of losses.

Table 4.1: Expected Throughput

Hops	Throughput (Kbps)
1	477.060608
2	238.631058
3	129.470490
4	96.574333
5	75.598198
6	57.554642
7	48.755135
8	43.459724
9	37.747978
10	34.591873

In order to evaluate the performance of BQM, we need to introduce a concept, called expected throughput.

4.3.1 Expected throughput

To gauge the impact of changes on TCP performance, expected throughput is used as the upper bound on TCP throughput. The TCP throughput measure obtained by simulation is then compared with the expected throughput[13].

We obtained the expected throughput as follows[13]. We first simulate a static(fixed) network of n nodes that formed a linear chain containing $n-1$ wireless hops (similar to the sting topology in). The nodes used the 802.11 MAC protocol for medium access. Then a one way TCP data transfer was performed between these nodes at the ends of the linear chain, and the TCP throughput was measured between these nodes.

Table 4.1 presents the measured TCP throughput as a function of the number of hops, averaged over ten runs. Observe that the throughput decreases rapidly when the number of hops is increased from 1, and then stabilizes once the number of hops

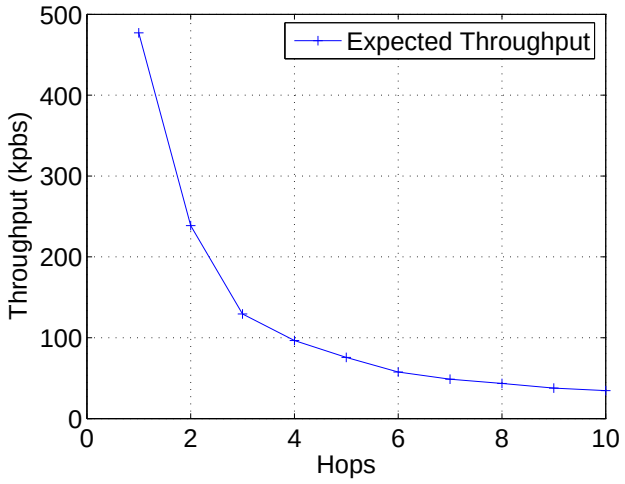

Figure 4.2: Throughput of different hops

becomes large. Figure 4.2 shows the trend. Our objective is to use these measurements to determine the expected throughput.

The expected throughput is a function of the mobility pattern. For instance, if two nodes are always adjacent and move together, the expected throughput for the TCP connection between them would be identical to that for 1 hop in Figure 4.2 . On the other hand, if the two nodes are always in different partitions of the network, the expected throughput is 0. In general, to calculate the expected throughput, let t_i be the duration for which the shortest path from the sender to receiver contains i hops. Let T_i denote the throughput obtained over a linear chain using i hops. When the two nodes are partitioned, we consider that the number of hops i is ∞ and $T_\infty = 0$. The expected throughput is then calculated as:

$$expectedthroughput = \frac{\sum_{i=1}^{\infty} t_i * T_i}{\sum_{i=1}^{\infty} t_i} \tag{4.1}$$

$\sum_{i=1}^{\infty} t_i$ is equal to the duration for which the TCP connection is in existence. The measured throughput may never become equal to the expected throughput.

4.3.2 BQM and Modified BQM Simulation Model

In order to simulate the wireless Ad hoc network environment, we create different numbers of mobile nodes first, which are supposed to have random movement in a previously decided area. The random movement of each node is generated by the command "setdest". TCP connections are randomly created between these mobile nodes based on a random seed which is also provided previously. The random seed should be generated from the time. Every mobile node is possible to be a sender of a TCP connection, while to be the receiver of another TCP connection at the same time. Moreover, every mobile node is supposed to work as the router for transmitting the flows of TCP connections in the wireless ad hoc network (refer to Section 2.1). Congestion may happen when the number of TCP connections increases and a lot of packet losses will occur due to the severe congestion during the simulation.

We will compare the performance of BQM , Modified BQM and Normal TCP in several aspects. First, we compare the performance of BQM, Modified BQM and Droptail with variable number of mobile nodes and constant number of TCP connections in the same area. The number of TCP connections is 24, and the number of mobile nodes is varying between 10, 15, 20, 25, and 30. Then, we will compare the performance of BQM,

Modified BQM, and Droptail with variable number of TCP connections and constant number of mobile nodes in the same area. The number of mobile nodes is set to be 20, and the number of TCP connections varies among 8, 16, 24, 32, and 40. The more TCP connections exist in the area, the heavier the congestion will be. The simulation area is 800 X 1600, the node moving speed is 5 and the bandwidth is 11Mbps. We will evaluate the throughput, fairness, and the improvement ratio.

In order to collect and analyze the data from simulation, we use the 'trace' command to record the traffic flows of TCP connections in the ad hoc network during the whole simulation. All information is saved in a trace file, and we can get some useful information by analyzing the trace file. We will modify the sender to record the number of hops for each successful packet transmission in a hop trace file. With the hop trace file, we are able to calculate the expected throughput for each TCP connection by equation 4.1. We will get the average of the average of each run and all the TCP connections.

The simulation time is 100sec, and as in the wired network, we select the start time for each TCP connection randomly. Before each run, we will first generate a start time list for the comparison simulation. In that way, we can ensure BQM and Droptail work in the exactly same environment. We get the average data of 10 runs for each simulation. In comparison, the Droptail sender will apply TCP-Newreno, and the receiver will apply TCP-sink.

Chapter 5

Simulation Results of Wired Network and Analysis

This Chapter is about the performance comparison of BQM and Droptail in the wired network. Pattern 8 is used(k=4), and the discard priorities are from 0 to 3.

In figure 5.1, the horizontal-axis shows the number of connections sharing the bottleneck link, while the vertical-axis denotes the fairness index of the connections. Fairness index is achieved by equation (2.1) with each connection's throughput.

Figure 5.1 shows the variation of the fairness. Extreme good fairness index is 1, which means the bandwidth is divided equally among these connections. In figure 5.1, if the number of connections is 2, that means there is almost no competition in the bandwidth. If the fairness index is close to 1, Droptail and BQM may have the same performance. However, when the number of connections increasing, congestion occurs, the fairness among the connections goes worse. Figure 5.1 shows that BQM performs much better than Droptail in the fairness, when severe congestion persists. If the number of connections is 8, BQM gets 8.5% better performance than Droptail in the fairness. If the congestion gets worse, 32 connections competing for the bottleneck link, BQM will get 40.0% better performance than Droptail.

The fairness is improved because all packets are marked with different discard priorities, and the router drops packets with the lowest priorities, that are distributed equally in the flows. As a result, BQM is a good queue management mechanism to share resource fairly among the connections, and can get much better performance than Droptail in the fairness.

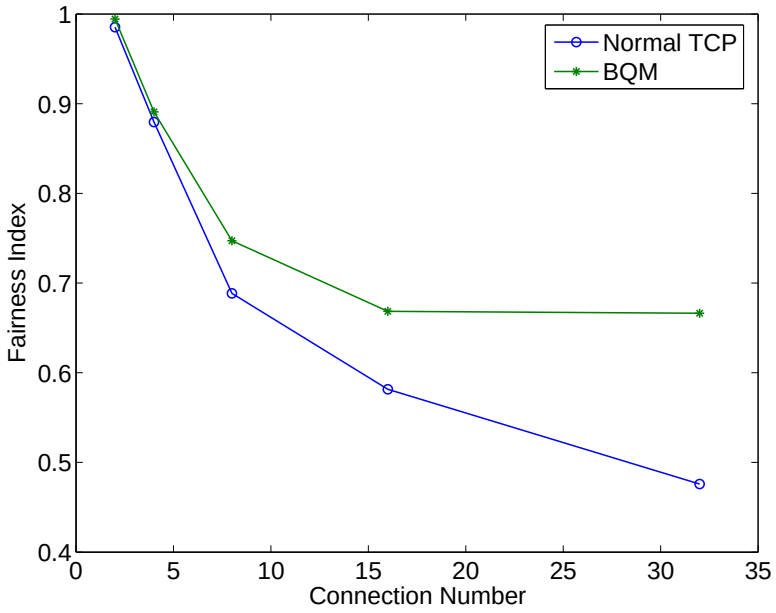

Figure 5.1: Fairness of all TCP connections in the Wired Network

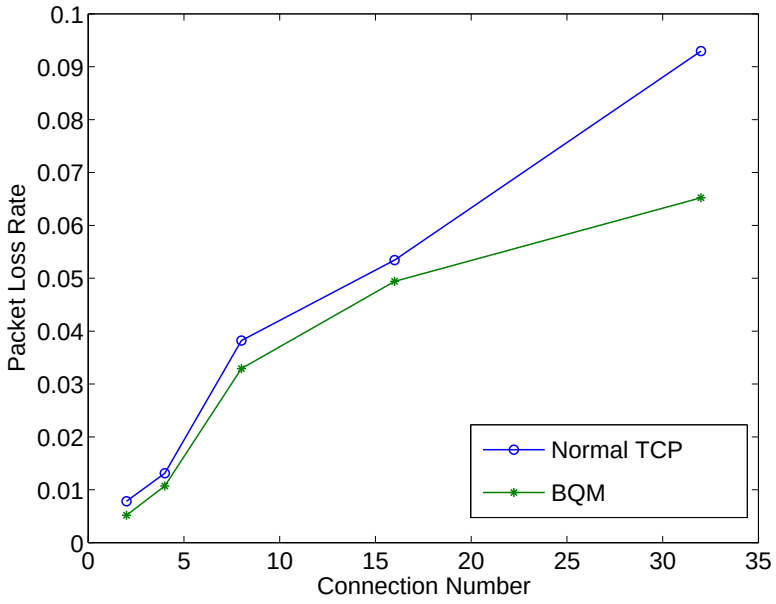

Figure 5.2: Packet loss rate of all TCP connections in the Wired Network

Figure 5.2 shows the packet loss rates of the two queue management techniques when in different congestion conditions. The vertical-axis denotes the packet loss rate of the network, which is the ratio of packets received to packets sent. Unless the sender gets the acknowledgement from the receiver, the packet can not be considered as received.

Extreme good packet loss rate is 0, which means no packet is dropped during the communication. In figure 2, when there are only 2 connections, the packet loss rate is close to 0, and the two queue management techniques perform almost the same. When more connections join in the competition, and congestion becomes worse, BQM performs a better packet loss rate than Droptail. When 8 connections exist, BQM gets 13.8% better performance than Droptail. However, if 32 connections exist, BQM will get 29.8% better performance than Droptail.

The reason for the low packet loss rate is the router drops packets in the ascending order of discard priorities, which can avoid the drop of a whole window. BQM is a low packet loss rate method which sparsely distributes losses to improve the network efficiency and save energy.

In figure 5.3, the horizontal-axis shows the number of connections sharing the bottleneck link, while the vertical-axis denotes the total throughput of the connections in the network.

Figure 5.3 shows BQM also gets better throughput than Droptail. The throughput here is the sum of each connection's throughput, and we get the average throughput from 25 runs. The throughput also shows the usage of the link in the bottleneck. If there are a few connections in the network, and even no congestion occurs, BQM and Droptail perform almost the same in the throughput. When connection number is 8, the

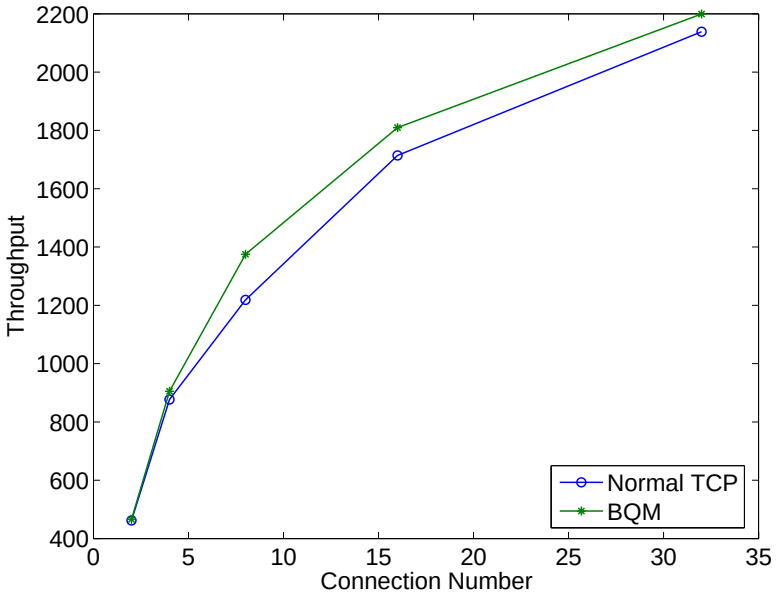

Figure 5.3: Throughput of all TCP connections in the Wired Network

throughput of Droptail is improved 12.9% by BQM. However, if the connection number continues increasing, then the performance of BQM is about 10.9%.

The appendix A includes more simulation results in different parameters, such as bandwidth, Tp, and efficiency.

Chapter 6

Simulation Results of Wireless Ad hoc Network and Analysis

We observe the performance of BQM, Modified BQM and Droptail in the wireless Ad Hoc network. We evaluate the throughput, fairness and throughput improvement ratio in variant number of mobile nodes, and variant connection number among these mobile nodes. The simulations are running based on both pattern 8 (packet sequence number modulo 8) and pattern 16 (packet sequence number modulo 16).

6.1 Performance Analysis of Pattern 8

Figure 6.1 shows the performance of TCP connections when the area size is varying in two dimension. We increase the area size in one dimension, while keeping the other dimension one time larger than the increasing one, that means if one side of the area is X, the other side should be 2X (area = X x 2X). From figure 6.1, we can get more information about the throughput of the TCP connections in the two-dimensional variant area. In our experiments, the number of mobile nodes is varying from 10 to 30, and the number of TCP connections is varying from 8 to 40, so we choose 20 as the number of mobile nodes and 24 as the number of TCP connections to test the ad hoc area size.

In figure 6.1, the horizontal-axis shows the area size, while the vertical-axis denotes the throughput of these connections in the area. We can notice from this figure, when X varies from 400 to 3200, 2X should be from 800 to 6400. The simulation results show that, when area size equals to 1600x3200, the system get the best performance of throughput. As the area size keeps increasing, the throughput decrease dramatically.

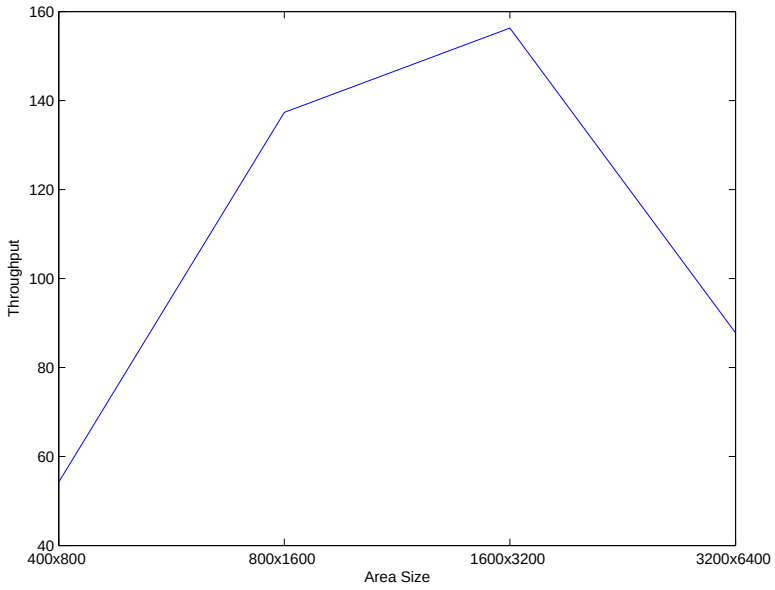

Figure 6.1: Throughput of 20 Mobile Nodes with different area size

However, when we varied the number of mobile nodes in area 1600x3200, we can notice if there are fewer nodes, for example, 5 nodes or 10 nodes in this area, the traffic is nearly 0. This is because area size of 1600x3200 is appropriate for 20 mobile nodes, but is too large for fewer nodes. On the other hand, we can test the traffic with different number of mobile nodes from 10 to 30 when area size is 800x1600. As a result, we may consider 800x1600 as the simulation area size for our experiments.

This simulation demonstrates the area size should be appropriate to the number of random mobile nodes. The area size should not be too small or too large. If the size is too small, all nodes are moving in a very limited range, it is not helpful for BQM to schedule the queue management, and improve the performance. If the size is too large, it is difficult to create reliable TCP connections among the mobile nodes. As a result, we choose 8R x 16R as our testing area for variant number of mobile nodes.

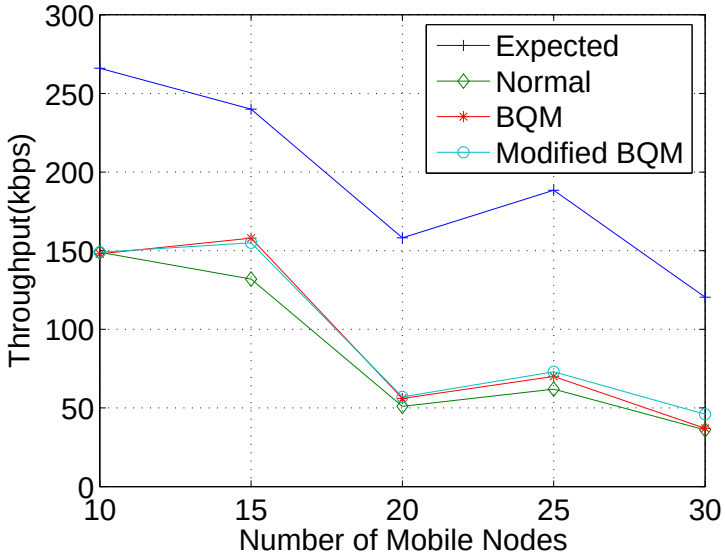

Figure 6.2: Throughput of variant number of Mobile Nodes

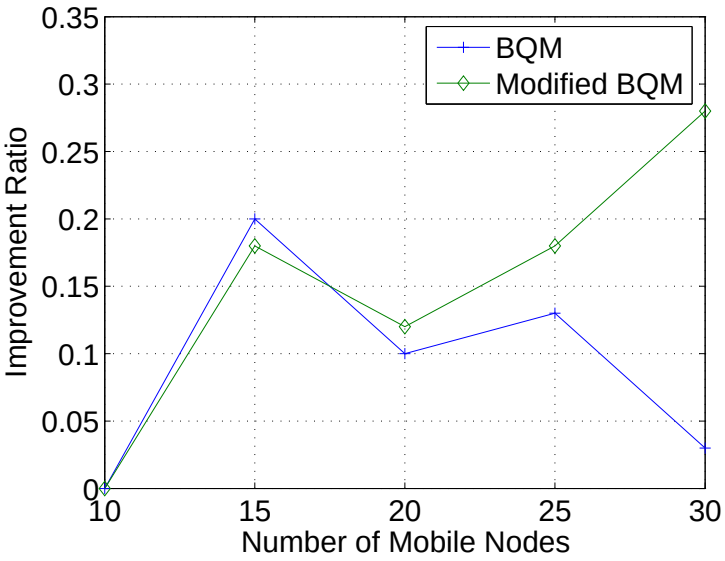

Figure 6.3: Improvement Ratio of variant number of Mobile Nodes

Figure 6.2, figure 6.3 are about the performance comparison of BQM, Modified BQM and Normal TCP (Droptail) when the number of mobile nodes varying from 10 to 30 in the 800 x 1600 area. The number of TCP connections is 24. Figure 6.2 shows the throughput of the TCP connections in this area. We can notice the throughput drops when there are more mobile nodes in the area. This is due to the number of hops between these mobile nodes is increasing. The curve on the top of the three is the expected throughput, as mentioned in section 4.2.1, usually, expected throughput is much higher than measured throughput.

Generally, BQM can get a better performance of throughput than Normal TCP, and Modified BQM is even better than BQM. Figure 6.3 is about the throughput improvement ratio, which shows BQM gets about 5% to 20% better performance than Normal TCP in throughput and also denotes Modified BQM can get about 5% to 28% better performance than Normal TCP. When there are fewer mobile nodes and a lot of competition among these mobile nodes, Modified BQM performs almost the same as BQM. However, as the number of mobile nodes increasing, Modified BQM can get much better performance than BQM.

BQM performs better than Normal TCP as the number of mobile nodes increasing, while congestion persists. This is because BQM can distinguish the congestion losses from random wireless losses, and adjust the congestion window size respectively. When a wireless random loss occurs, Normal TCP will recognize it as congestion loss and cut the congestion window to half. Although, no congestion exists, the output packets from the sender is limited, and the network system is in low efficiency. However, with BQM, the sender can react differently to random losses and congestion losses. For random losses, the sender will retransmit the packet. For congestion losses, the sender will not

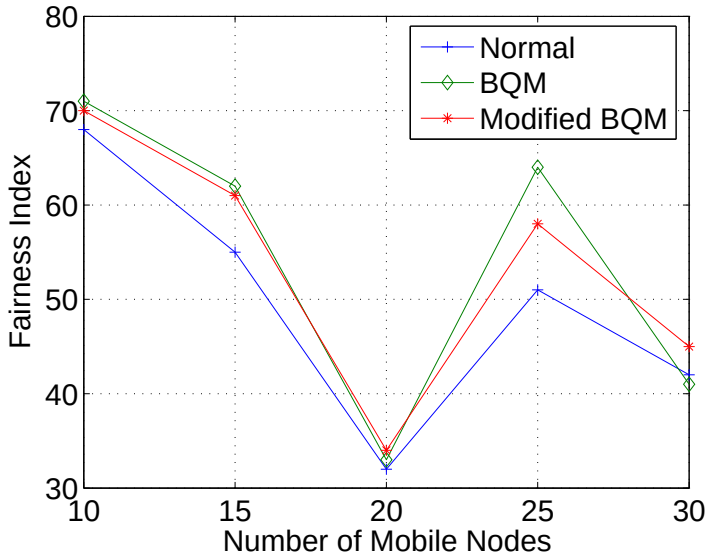

Figure 6.4: Fairness Index of variant number of Mobile Nodes

just simply set the window size to half, but adjust the congestion window size according to the congestion level. This will help the system to improve the system efficiency, and get better performance.

Figure 6.4 is about the fairness index of these TCP connections. We can notice BQM can get better fairness than Normal TCP, however, Modified BQM does not get better fairness than BQM. This is because our modification does not affect the flows. Modified BQM can get better performance than BQM is because modified BQM can tell "jump" losses, and get the correct congestion measure, which let the sender adjust the congestion window to the most appropriate size.

Figure 6.5, figure 6.6 are about the performance comparison of BQM, Modified BQM and Droptail when there are 20 mobile nodes moving randomly in the 800 x 1600 area and the number of TCP connections varies from 8 to 40. Figure 6.5 shows

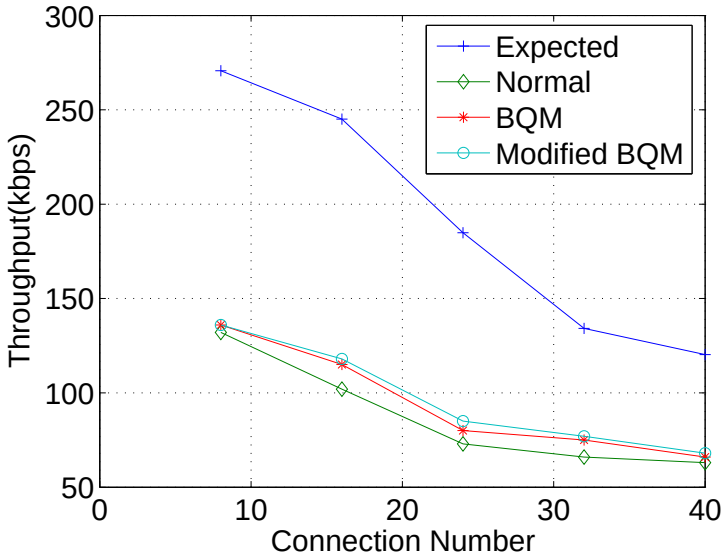

Figure 6.5: Throughput of 20 Mobile Nodes with different TCP Connection Number

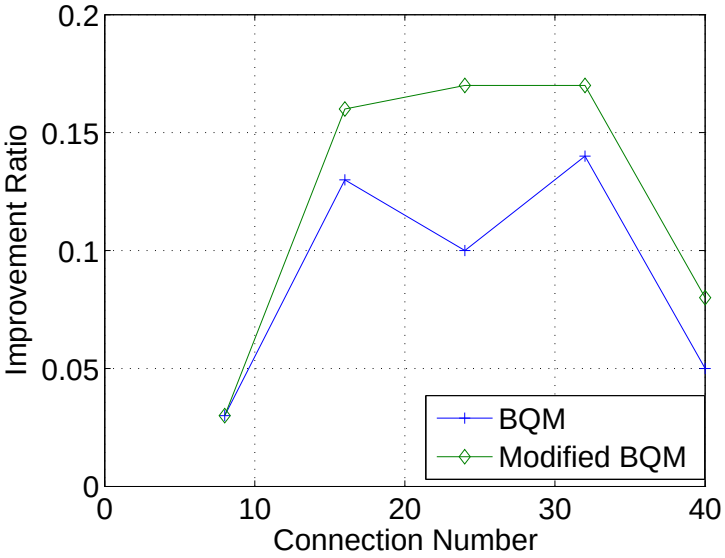

Figure 6.6: Improvement Ratio of 20 Mobile Nodes with different TCP Connection Number

the throughput of the TCP connections in this area. We can see the throughput is decreasing while the number of TCP connections is increasing. This phenomenon is due to the increasing competition between the TCP connections. When there is only 8 connections among these mobile nodes, only a few congestion losses will occur, so BQM, or Modified BQM, does not have obvious improvement to Normal TCP. However, when more TCP connections are added into the network, and the congestion keeps getting worse, BQM can get about 5% to 15% better performance of throughput than Normal TCP, and Modified BQM can get about 5% to 20% better performance than Normal TCP. If the congestion is too severe, that there are too many packet losses, all of the three can not get a good performance. Figure 6.6 shows the throughput improvement ratio. The curve shows the same information as figure 6.3, that Modified BQM can improve the throughput when congestion persists.

This simulation gives more specific information about the BQM performance when congestion getting worse. With the Accurate Loss Discriminator, BQM indeed improve the Normal TCP network, but if there is no congestion or the congestion is too heavy, we can not expect BQM to get great improvement.

The same as figure 6.4, figure 6.7 shows that the fairness is improved by BQM, but Modified BQM does not get better fairness than BQM.

6.2 Performance Analysis of Pattern 16

According to the simulations in pattern 8 of discard priorities, we can evaluate the performance of BQM, Modified BQM and Normal TCP by using pattern 16 (Figure 3.2) of discard priorities in area 800 x 1600. Figure 6.8 and Figure 6.9is about the throughput and the fairness index in 800 x 1600 area when the number of mobile nodes is varying

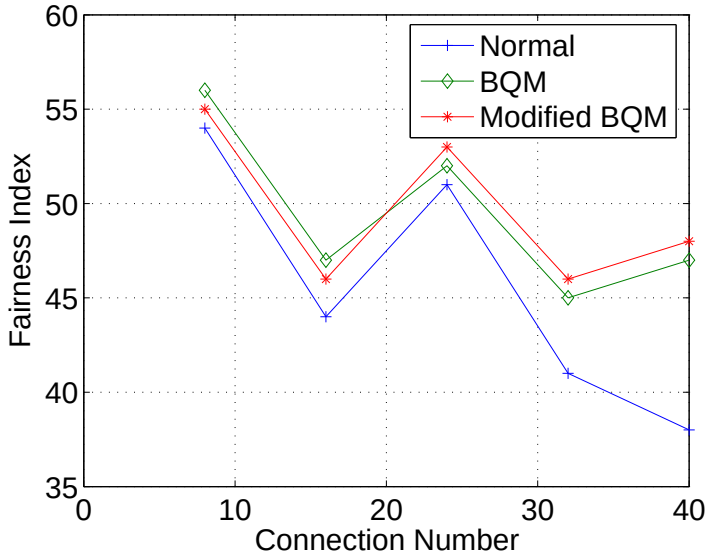

Figure 6.7: Fairness Index of 20 Mobile Nodes with different TCP Connection Number

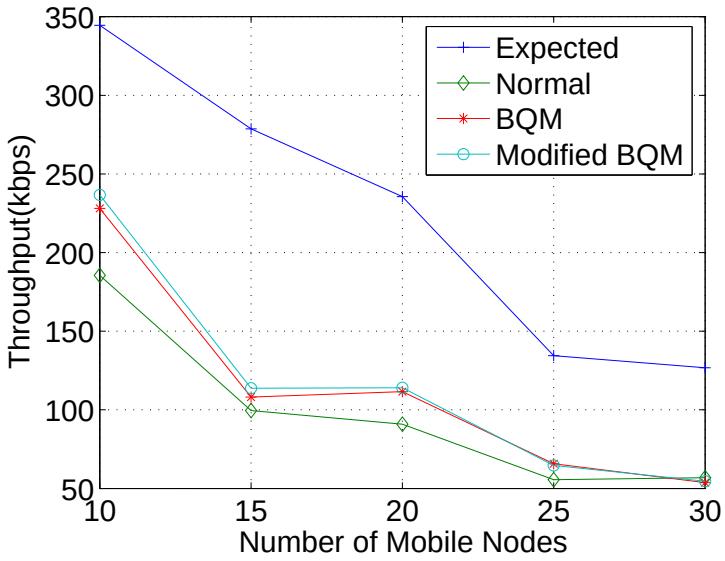

Figure 6.8: Throughput of variant number of Mobile Nodes with Pattern 16

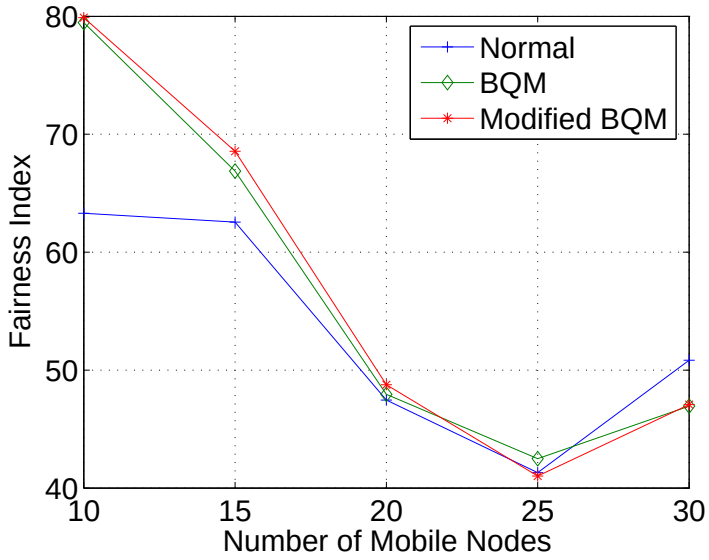

Figure 6.9: Fairness Index of variant number of Mobile Nodes with Pattern 16

from 10 to 30, and the number of TCP connections is 24. The horizontal-axis shows the number of mobile nodes. We can get the information from figure 6.8 that BQM can get about 5% to 20% improvement of the throughput, and Modified BQM can get about 5% to 25% improvement of the throughput. From figure 6.9, we can notice BQM performs a better fairness than Normal TCP and Modified BQM also works the same as BQM.

Figure 6.10 and Figure 6.11is about the throughput and the fairness index in 800 x 1600 area when the number of TCP connections is varying from 8 to 40, and the number of mobile nodes is 20. We can get the information from figure 6.10 that BQM can get about 5% to 10% improvement of the throughput, and Modified BQM can get about 5% to 12% improvement of the throughput. From figure 6.11, we can notice BQM performs a better fairness than Normal TCP and Modified BQM also works the same as BQM.

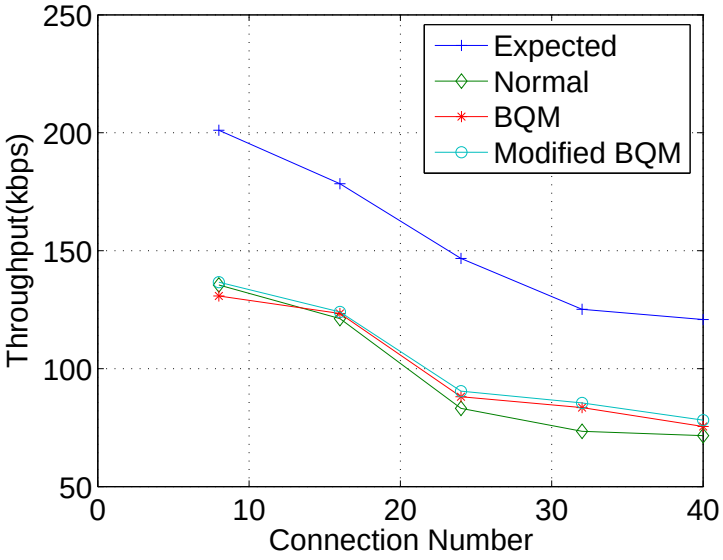

Figure 6.10: Throughput of variant number of Connections with Pattern 16

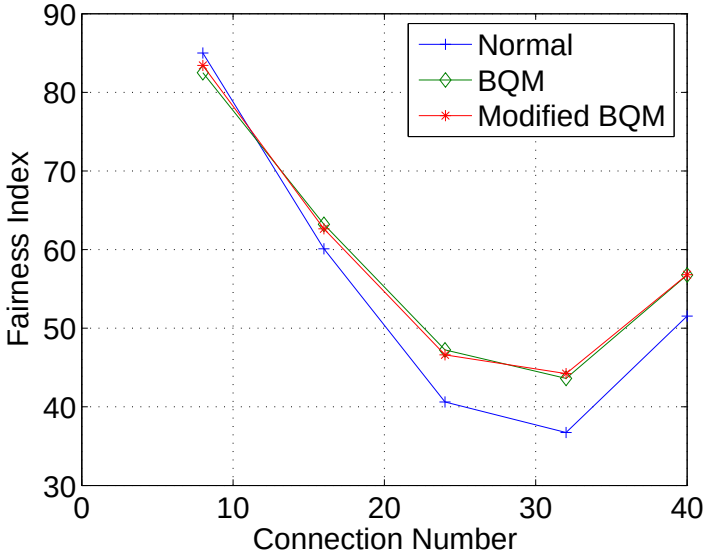

Figure 6.11: Fairness Index of variant number of Connections with Pattern 16

Chapter 7

Conclusion and Future Work

This report presents a study of a new queue management algorithm Biased Queue Management (BQM) and also proposes some modification to BQM. By comparing BQM, Modified BQM with another widely used queue management mechanism Droptail in several aspects, we are able to figure out some characteristics of BQM. We design several experiments to simulate the these queue management techniques in both wired network and Wireless ad hoc network. In wired network, we analyze the throughput, packet loss rate, and fairness of the two queue management methods, and draw the conclusion that BQM performs better than Normal TCP. In wireless ad hoc network, we compare the throughput and fairness index of BQM, Modified BQM and Normal TCP with variant area size, variant number of mobile nodes and variant TCP connection number.

By the comparison, we show that, in wired network, BQM enforces the fairness among flows, improve the packet loss rate by distributing the losses among the flows and get better throughput in the traffic; in wireless ad hoc network, BQM can get better TCP performance of throughput and fairness index than Normal TCP. Furthermore, Modified BQM can even get better performance of throughput than BQM, but no better performance of fairness index. However, the improvement is related to the simulation area size, number of mobile nodes and number of TCP connections. As a result, we should choose the appropriate queue management method according to the number of mobile nodes in different network environment.

Mobile ad hoc networks are built on a predicate of cooperation and trust. However, a malicious node may well attempt to disrupt the network. We are aware that the biased queue management will facilitate the disruption of the network. A malicious node could bombard the network with packets bearing the highest discard priority. We plan to investigate stateless techniques to defeat these greedy and malicious nodes. A possible approach is to use queue management techniques such as CHOKE. Another approach is to associate a shadow price based on the discard priority of packets.

We can adapt CCM and ALD to heterogeneous internet. While the biased queue management may well be adopted over some mobile ad hoc networks, it is unrealistic to hope for its wide deployment over the internet. We investigate other queue management schemes that can substitute BQM to enable the design of an accurate loss discriminator (ALD) and a continuous congestion measure (CCM).

Bibliography

[1] David B. Johnson and David A. Maltz, "Dynamic Source Routing in Ad Hoc Wireless Networks" *book Mobile Computing, edited by Tomasz Imielinski and Hank Korth, Kluwer Academic Publishers*, July, 1996.

[2] Saad Biaz and Nitin Vaidya, ""De-randomizing" Congestion Losses to Improve TCP Performance over Wired-Wireless Networks" *Proc. of IEEE Global Telecommun. Conf.*

[3] S. Floyd and V. Jacobson, "Random early detection gateways for congestion avoidance", *IEEE/ACM Transactions on Networking,* vol. 1, pp 397-413, Aug, 1993.

[4] Hari Balakrishnan, Venkata Padmanabhan, Srinivasan Seshan, and Randy Katz. "Eectiveness of loss labeling in improving TCP performance in wired/wireless networks. " *In Proceedings of ICNP'2002: The 10th IEEE International Conference on Network Protocols,* , Paris, France, November, 2002.

[5] L. Brakmo and S. O'Malley, "TCP-Vegas : New techniques for congestion detection and avoidance,", *In ACM SIGCOMM'94,* pp. 24-35, OCT, 1994.

[6] V.Jacobson, "congestion Avoidance and Control", *Proc. SIGGCOMM,*pp. 314-329, Aug, 1998.

[7] R.Yavatkar and N.Bhagwat, "Improving End-to-End Performance of TCP over Mobile Internetworks" *Proc of Worjshop on Mobile Computing Systems and Applications*, Dec, 1994.

[8] H. balakrishnan, S. Seshan, E. Amir, R.H. Katz, "Improving TCP/IP Performance over Wireless Networks," *Proc. 1st ACM Conf. on Mobile Computing and Networking*, November, 1995.

[9] Sally Floyd, "TCP and Explicit Congestion Notification," *ACM Computer Communication Review*, vol. 24, No. 5, October, 1994.

[10] B. Bakshi, P. Krishna, N. Vaida, and D. Pradhan, "Improving performance of TCP over wireless networks," *in proceedings of 17th Int. Conf. on Distributed Computing Systems*. pp. 693-708, May, 1997.

[11] A.Bakre and B.Badrinath, "I-TCP: Indirect TCP for mobile hosts", *in Proc. 15th International Conf. on Distributed Computing System (ICDCS)*, pp. 152-159, May, 1995.

[12] H.Balakrihnan, V. Padmanabhan, S.Seshan, and R. katz, "A comparison of mechanisms for improving TCP performance over wireless links", in ACM SIGCOMM'96, pp. 152-159, Aug, 1996.

[13] Gavin Holland and Nitin Vaidya, "Analysis of TCP Performance over Mobile Ad hoc Networks", *in Proc. 15th International Conf. on Distributed Computing System (ICDCS)*, pp. 152-159, May, 1995.

[14] H.Balakrihnan and R.Katz, "Explicit loss notification and wireless web performance," in *IEEE Globecom Internet Mini-Conference, Sydey,* Oct, 1998.

[15] H.Balakrihnan, V.N. Padmanabhan and R.Katz, "The effect of asymmetry on TCP performance," in *Proceedings of the IEEE Mobicom'97*,(Budapest, Hungary), pp.77-89, September, 1997.

[16] J.Broch, D.A.Maltz, D.B.Johnson, Y.Hu, and J.Jetcheva, "A performance comparison of multi-hop wireless ad hoc network routing protocols," in *ACM/IEEE Int. Conf. on Mobile Comprting and Networking*, pp.85-97, Oct, 1998.

[17] J.Broch, D.B.Johnson, and D.A.Maltz, "The dynamic source routing protocol for mobile ad hoc networks", Internet-Draft of the IETF MANET Working Group, December, 1998.

[18] T.D.Dyer, R.V.Boppana, "A comparison of TCP performance over three routing protocols for mobile Ad hoc networks", ACM MOBIHOC, 2001.

[19] M.S.Corson and A. Tphremides, "A distributed routing algorithem for mobile wireless networks," *ACM J. Wireles Networks*, vol.1, pp. 61-81, 1995.

[20] B.Das, E.Sivakumar, and W.Bhargavan, "Routing in ad hoc networks usin a virtual backbone." manuscript, 1997.

[21] Sally. Floyd, Steve McCanne "Network Simulator," *LBNL public domain software.* Available via ftp from frp.ee.lbl.gov.

[22] ISI, "ns2: network simulator" http:
www.isi.edu/nsnam/ns

[23] V. Jacobson, "Modified TCP congestion avoidance algorithm," Apr 1990. mailing list, end2endinterst@isi.edu.

Appendices

This Chapter provides more information about the performance of BQM in Wired Network. Figure 1 is the throughput of TCP connections when Bandwidth is 6Mbps, and Tp is 150ms. Figure 2 is the fairness index among these TCP connections, and figure 3 is the packet loss rate of the network. The simulation environment is the same as Chapter 4.

We can notice from these figures: the performance of BQM is better than Droptail in the three aspects, throughput, fairness, and packet loss rate. This fact is another evidence that demonstrate BQM can improve the TCP performance over wired networks.

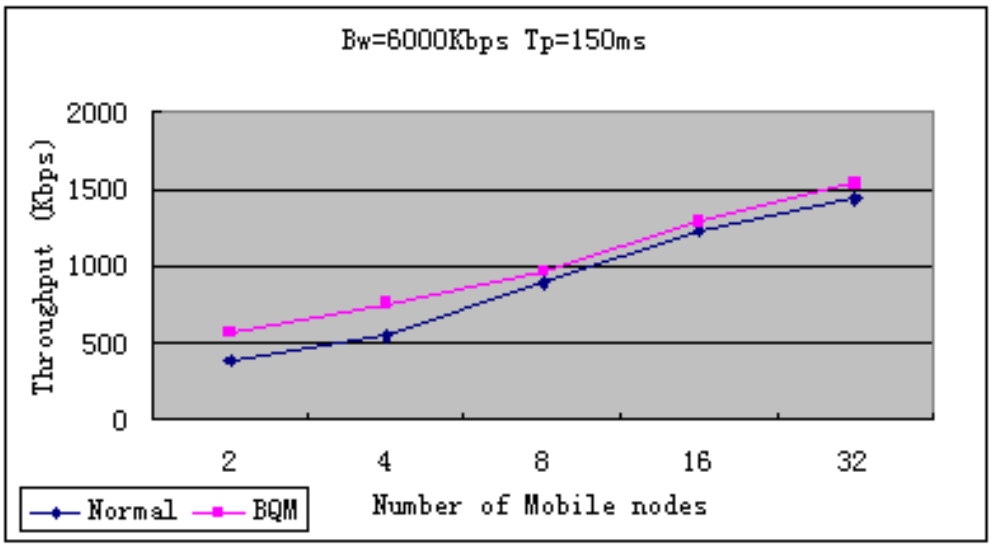

Figure 1: Throughput of all TCP connections in the Wired Network

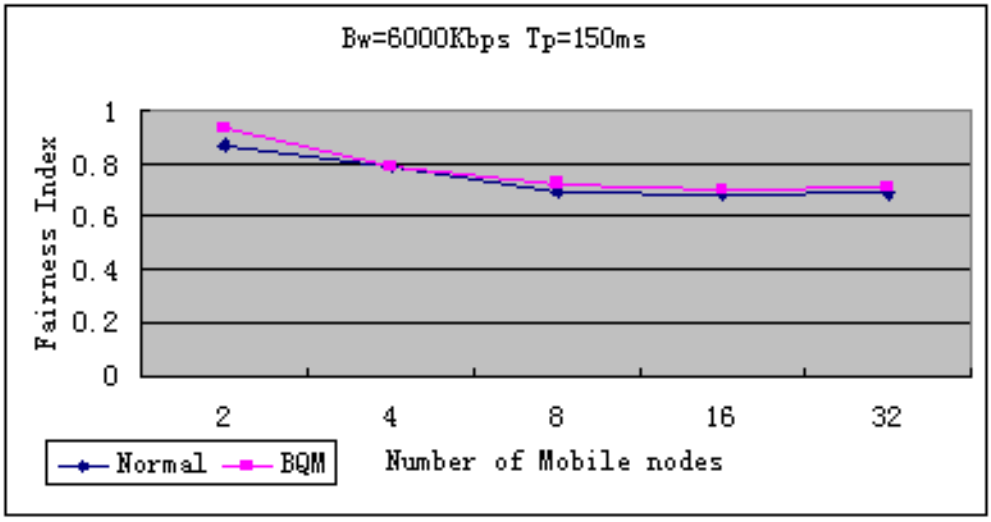

Figure 2: Fairness Index of all TCP connections in the Wired Network

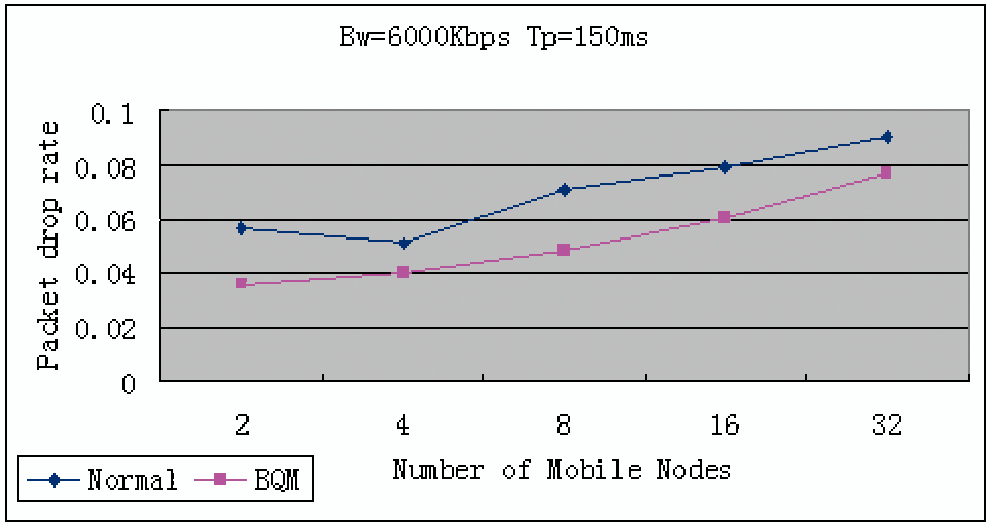

Figure 3: Packet loss rate of all TCP connections in the Wired Network