Distributed Admission Control and Quality Control for Multicast Network

by

QIHUA YANG

A dissertation submitted to the Graduate Faculty in Engineering in partial fulfillment of the requirement for the degree of Doctor of Philosophy, The City University of New York

2012

This manuscript has been read and accepted for the Graduate Faculty in Engineering in satisfaction of the dissertation requirement for the degree of Doctor of Philosophy.

Date	Professor **Tarek N. Saadawi** Chair of Examining Committee
Date	Dean **Mumtaz Kassir** Executive Officer

Professor **Umit Uyar**
The City University of New York

Professor **Yi Sun**
The City University of New York

Professor **Jizhong Xiao**
The City University of New York

Dr. **Mariusz Fecko**
Telcordia Technologies

Supervision Committee

The City University of New York

Abstract

Distributed Admission Control and Quality Control for Multicast Network

Qihua Yang

Advosor: Professor Tarek N. Saadawi

The evolution of network communication techniques has reached a point where more and more applications, such as video conference and streaming audio require a guaranteed quality service. A lot of researchers propose different admission control schemes or resource schedule schemes to limit or reasonable use the network resource. Most of them focus on the unicast network while few consider the feature of multicast network. Multicast streaming use the different transmitting mode and has different network resource access approach.

The motivation of this thesis is to propose the network resource measurement algorithms regarding to the multicast characters and design a multicast admission control and quality control scheme. This system control the receiver to join the group by segment path measurement algorithm, choose the best branch point by single link capacity measurement algorithm and switch the group for the receivers by measurement and SVR mixed prediction algorithm in order to ensure the streaming quality.

We present a segment path capacity measurement by injecting a few different sized probing packets. We probe the branch tree instead of the full path to improve the efficiency and reduce the disturbance of cross traffic. The segment capacity together with end-to-end queuing delay decides whether the new receiver is accepted to join the group or not.

We introduce the single link capacity measurement by using the probing packet train pair. It is the further development of segment probing algorithm. This algorithm can be used to resource management, topology control and the branch point selection. We use the link capacity distribution to select the best branch point to graft the multicast receiver to the multicast tree.

Next we design a SVR and measurement based network condition prediction algorithm. For multicast streaming mechanism, in order to ensure the quality service of receiver, the same source normally creates multiple groups, which provide different streaming quality. The receiver can switch group based on the real time network condition. Our scheme injects the probing packet to measurement end-to-end queuing delay. The delay and buffer information are collected as training set of SVR algorithm to predict the network condition in near future. It can trigger the receiver to switch the group before it suffers severe packet loss or delay.

We also implement the multicast admission control system. The system supports multi group switching for receivers. The receiver measures the network by the algorithms we mentioned above to decide which group to join. In real time, the receiver monitors the network and switches the group in order to maintain the quality of streaming service. The system supports almost all kinds of video streaming format and has good performance.

Acknowledgements

First of all, I would like to thank my major advisor Professor Tarek Saadawi for his guidance and encouragement through my Ph.D study. Professor Saadawi is kind, patient and extremely responsible. I would not have done without his support.

I would also like to thank the other members of my thesis committee, Professor Yi Sun, Professor Jizhong Xiao, Professor Umit Uyar and Dr. Mariusz Fecko for taking time to review my dissertation and make valuable comments on my dissertation. At the same time, I am grateful to Dean Mumtaz Kassir for providing me with advice and guidance through my study.

I want to thank my wife, Qian Liu. It is her love, encouragement and sacrifice that enable me to focus on my research. I want to thank my parents for their love and support of my study aboard. I would like to thank my elder sister. She always gives me support whenever I need. Her study and life experience in the United States helped me learn many things in an easy way.

Contents

List of Figures

List of Tables

Chapter 1
Introduction

1.1 Challenge and Research Area

Currently, many applications, such as video conferencing or streaming audio require a guaranteed quality of service to work properly. Ensuring minimum quality of service to traffic flows and groups of flows become an important challenge for network performance. Under such conditions, admission control algorithms are utilized to ensure that admittance of a new flow into a resource constrained network does not violate the service requirements expected by a network to the already admitted flows, and to achieve high network utilization. Those algorithms can be based on different requirements, such as link delay, minimum bandwidth, and maximum packet loss suffered by flow.

Admission control has been extensively studied in unicast networks to control and avoid congestion, packet loss or video jitter. Most of those algorithms focus on single path because in unicast networks one sender transmits packet to a single receiver. However, admission control to multicast should be paid more attention because more and more real time applications require the transmission of flows from a sender to multiple receivers. These applications have higher quality requirements. At the same time, you cannot forecast when and where the new receiver will join or leave the same group. Those factors make admission control in multicasting network a difficult task.

In a network, multicast is an efficient way for one-to-many and many-to-many communications. Each multicast group owns a set of members, and every member can be a sender or receiver. Sender in a multicast group delivers packets to all receivers of the group. Multicast only duplicates packets in router from where packets are delivered to different receiver. This method saves network resources.

Unfortunately, there is no effective and efficient multicasting admission control algorithm until now. [1] - [4] are the current available multicasting admission control mechanisms. All these algorithms are based on probing mechanism, which is extended from unicast admission control, and more suitable to single path than multipath conditions.

1.2 Multicast vs. Unicast

Multicast and unicast are two main communication methods in packet forwarded network. Unicast is a one to one communication. The sender injects one copy of the traffic into the network. Only one receiver can receive it. If there are multiple receivers all want to receive the traffic from the same source, the sender has to send out multiple copies of the traffic. The number of copies depends on the number of the receivers, as shown in Figure 1.1. In extreme situation, when thousands of receivers try to receive the same traffic from the same source at the same time, transmitting thousands of copies for the same traffic bring huge work load to the sender. It requires the sender has powerful processing ability. Moreover thousands of traffic copy will use up networking resource. That could bring congestion into the network and force other traffic to suffer packet loss, delay and jitter.

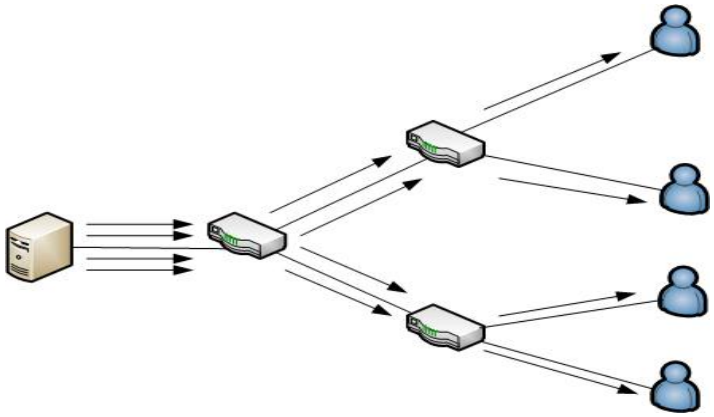


Figure 1.1 Unicast network traffic

Multicast is a different communication approach comparing with unicast. Multicast is one to multiple. It fixes the problem we discussed above. When multiple receivers all want to receive the traffic from the same source, they join the same group. The sender just injects one traffic copy into the network, as shown in Figure 1.2. Routers inside the network duplicate and forward traffic for each receiver. No matter what the number of receivers is, the sender only injects one copy into the network. And for any link inside the network, there is only one traffic copy travel across it. For multicast network, the multicast protocol creates a multicast tree for the group. Each receiver that tends to receive the group traffic will send join message toward the source. Finally the receiver graft itself to the multicast tree as a tree branch. It is the biggest different thing between multicast and unicast. This feature is also the foundation of my proposed scheme. While most of other researches focus on unicast network traffic, we dig into the multicast structure and develop the multicast admission control and quality control system.

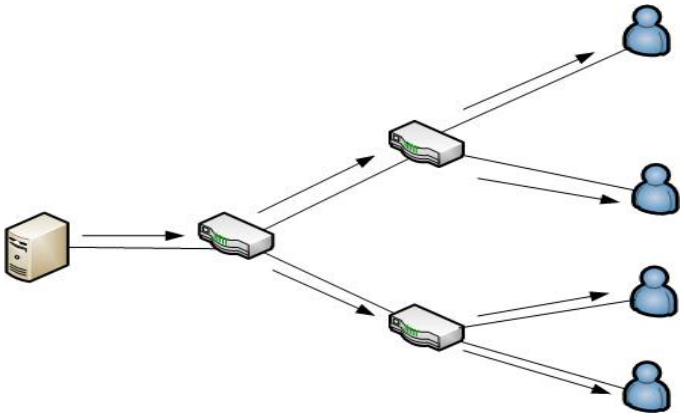


Figure 1.2 Multicast network traffic

1.3 Classify the Admission Control Algorithm

Referring to Figure 1.3, Admission control algorithms can be classified into two main categories. The first category is parameters based admission control techniques, References [5], [6] and [7]. The second category is measurement based admission control techniques, References [8], [9], [10] and [11]. Parameters based methods usually apply mathematical models to describe input traffic, analyze and predict network conditions, examples include Markov chain methods and central limit theorem to analyze queuing delay. Accuracy of these methods depends on the reliability of the assumed models.

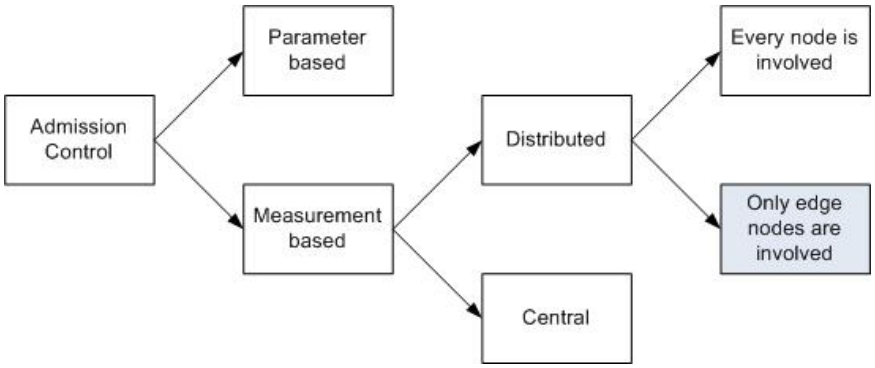


Figure 1.3 Classification of admission control policy

Measurement based methods reduce the error probability of decisions and since it's based on real time measurements, such as link delay, available bandwidth or packet loss, it can adjust measurement results with the dynamics of network resource usage, thus leading to increase in the network utilization.

Measurement based admission control techniques in turn can be classified into two sub-categories. The first sub-category is centralized admission control, References [12] and [13], which requires a central agent to measure and collect network information. This method is easy to implement and manage. However, since the admission control mechanism totally depends on that central agent, it can itself become a bottleneck in terms of performance. In addition link failures or congestion can lead to service unavailability despite the agency working properly and the network resources are available. The second sub-category is the distributed admission control techniques, References [14] and [15], where there is no single agent in the network. Multiple nodes or even every node may get involved into measurement and the decision making process. Based on the scope of nodes that are involved into admission control, we also can divide the second sub-category into two groups. In the first group, admission decision is made by every node inside the network. All nodes along the path measure and calculate its available resources. New receivers are admitted only to the path that has enough resources. This method avoids central agent management, and nodes do not need to keep state information for each flow. There is overhead due to the interaction and control between every node in the network. In addition, it is not suitable for some networks where the edge nodes do not have visibility to the network structure and cannot control intermediate nodes. The second group is also based on a distributed approach, but only edge nodes are involved, References [16] and [17]. Ingress or egress routers measure network parameters by sending probe packets.

Most current research on multicasting admission control techniques focuses on measurement-based admission control. These schemes are mostly deduced from unicast policies that use regular probing method to measure the network paths, References [18] - [21]. However, they don't consider particular features of multicasting traffic. Full path probing increases measurement error probability and wastes network resources.

The proposed scheme belongs to edge-based distributed admission control. Edge router injects probe packets to measure network resource, and measurements do not depend on the participation of core routers.

1.4 Thesis Contribution

The aim of this thesis is to design a distributed admission control and quality control system in multicasting environment. Our system considers the characters of multicast network that is ignored by most other researches and achieves very high quality performance for multicasting traffic. The system mainly provides QoS/QoE to the multicast service by three stages. First stage is to control the request of new receiver before they join the group. Second stage is to help the new receiver chose a better way to join the group. Third stage is after the new receiver joined the group, to predict the network condition and adjust the traffic sending rate in order to avoid congestion.

In this thesis, we present the admission control and quality control system. The system is composed of two components. First is admission control logic. Before the receiver joins the group, we use two novel probing measurement methods to estimate the network condition:

- Segment path capacity measurement for multicast network

We compare the difference between multicast and unicast. Based on the features of multicast network, we design the segment path capacity measurement to replace the commonly used full path capacity measurement by using different size probing packets. The probing policy is easy to implementation and only edge nodes involved. We demonstrate through simulation that segment path measurement has higher efficiency and more robust than full path capacity measurement.

- End-to-end queuing delay measurement and network condition prediction

  We propose a novel end-to-end queuing delay measurement method in this section. The proposed method never need time synchronization. It doesn't involve core router. This method only requires the network to support Diffserv. it only injects a few probing packets into the network. The dispersion between the probing packets can be used to calculate end-to-end queuing delay. We filter the measurement results to be data samples. Those samples are used to train the dataset for later prediction. The system utilize SVR algorithm to calculate the network condition of near future and update the training set. This process is repeated until the connection is close. The prediction to the network condition reduces the packet loss and jitter. It provides the user a better feeling during steaming quality adjusting.

Second is multicast tree optimization module. The receiver has multiple paths to graft itself to the multicast tree. In order to aid the receiver to choose the best path, we collect any link capacity by presenting the single link capacity measurement technique.

- Single link capacity measurement technique

  The single link capacity is also based on probing packets. We use multiple probing packet train to measure any single link capacity along the path. This is critical factor for

managing network topology. And for multicast network, the single link capacity information also helps the receiver to check the tree branch capacity and choose a better branch point to graft itself to the multicast tree.

We measure the link capacity or path capacity by injecting probing packets into the network. That gives you information about the network resource distribution. Based on this information, you make a wise decision whether to join the multicast group or not. And you can also choose a better way to join the multicast group. That is, graft your multicast branch to a better branch point. These kinds of quality control are called pre-quality control.

- Implementation of multi-streaming system

  This is an application can be used in real world. The system can create multi groups that provide different quality streaming for the same source. Receiver probes the network path and joins certain group based on its network condition. The probing algorithm is discussed in chapter 2. After the receiver joins the group, it monitors the network condition. It switches the group to ensure the receiver be free from congestion. The implementation has two modules to run on the two sides of network. We set up a test bed to test the system performance. The receiver can join the proper groups and can display the streaming stable even it switches among groups.

1.5 Organization of the Dissertation

The reminder of this thesis is organized as follows:

Chapter 2 introduces a segment path capacity measurement method by using different sized probing packets.

Chapter 3 extends the probing packet measurement method to measure a single link path capacity.

Chapter 4 introduces a measurement and SVR mixed prediction scheme to infer the network condition in near future.

Chapter 5 implements the admission control system, configure and test in real test-bed environment.

Chapter 6 concludes contributions of this thesis and lists a few future researches.

Chapter 2
Segment Path Capacity Measurement in Multicast Environment

In this chapter, we present the design, implementation and evaluation of a novel and effective capacity measurement algorithm as a major part of our admission control system. This measurement algorithm is based on end-to-end probing, never involving the routers along the path. In section 2.1 we first describe the regular packet forwarding features when packet is switching inside the network. In section 2.2, based on this knowledge, we present the segment path capacity measurement algorithm. For the multicast network, all receivers that belong to a certain group have to graft to the multicast tree as a tree branch. Based on the multicast feature, the new receiver will not add any additional traffic to old multicast tree. It only introduces traffic to the new tree branch. Therefore our proposed scheme only measures the segment of the path instead of the full path. That ensures the measurement precision and efficiency by dramatically reducing the influence of cross traffic. In section 2.3 we implement the admission control model that action is triggered by our packet probing results. In section 2.4 we compare the proposed algorithm with full path measurement for three aspects:

- measurement error rate under different cross traffic rate
- probing time vs. measurement accuracy for different probing algorithm
- the impact of probing packet size on measurement accuracy

In section 2.5 we conclude our probing scheme and discuss the future work.

2.1 Introduction

Network resource measurement is a challenge for packet switched network. Currently, numerous applications, such as video conferencing and streaming audio, require a guaranteed Quality of Service (QoS) to work properly. Network bandwidth is one kind of network resources. The concept of bandwidth is central to digital communications, specifically to packet switched networks, as it relates to the amount of data that a link or network path can deliver per unit of time. the path capacity is the maximum IP layer rat the path can transfer from the source to receiver. In other words, the path capacity establishes an upper bound to the throughput a receiver can expect to get from the network. For example, C_i is the capacity of the ith link, N is the number of links along the path. The path capacity is $C = \min_{i=1,\dots,N} C_i$. It is always a limited resource. All applications trying to transmit data through network cannot exceed the available capacity of network path. Exceeding the network bandwidth limits the application's performance. For example, for the multimedia streaming application, the receiver will suffer frame loss or suffer unendurable delay because of the limited path capacity. In this chapter we develop the capacity measurement scheme for multicast network. That scheme utilizes the character of multicast traffic to improve the measurement efficiency and precision.

2.1.1 Network Models and Definitions

Most current networks are packet switched networks that are based on store and forward switching. This kind of network decides how to transmit each packet independently of other packets. Store and forward means the router or switch cannot send out the first bit of packet until it receives the last bit of packet. The other kind of network is cut-through switching, which bring

less delay than store and forward switching. However store and forwarding is much easier to implement.

Link capacity is the maximum data rate that the receiver can transmit across the link. Link capacity is different to **path capacity**, which is the minimum link capacity along the network path.

Bottleneck link capacity actually is same to the path capacity. It is the smallest link capacity along the path. Network congestion often occurs over bottleneck link. Therefore, for congestion avoid algorithm, reasonably scheduling the resource usage over bottleneck link is most important.

Available bandwidth is the link bandwidth that is not used by other traffic. It also can be said the maximum amount of data the receiver can receive during a time interval.

2.1.2 Multicast Network Features

Compared with unicast traffic, the key benefit of multicast traffic is reduction of traffic load when multiple receivers require the same multimedia streaming from the same source. For multicasting network, only one copy of streaming is injected into the network whatever the number of receivers while the number of streaming is highly related to the number of receivers in unicast network. The different streaming behavior between multicast and unicast leads the different network resource measurement modes. While most current bandwidth measurement approaches focus on the unicast environment, our scheme specially considers the difference between multicast and unicast, is especially suitable to multicasting network.

We consider the multicasting network depicted in Figure 2.1. Routers inside this domain are classified into core router (CR) and edge router (ER). Every router runs multicasting protocol. In

the multicasting path $(L_1, L_2, \dots, L_n)$, L_1 is the link from where ingress router ER_1 injects multicast traffic into the network. L_n is the link that connects the network to egress router ER_2.Core routers $CR_j, j = 1,2 \dots, n-1$ switch packet from link L_j to L_{j+1}along the path. We assume the source always has more than one receiver in order to make sure there is at least a segment that is shared by receivers along the path. We assume multicast packet is copied at CR_i, therefore, multicasting path is divided into two parts. $(L_1, L_2, \dots, L_i)$ is the upper path segment that is common to some receivers. $(L_{i+1}, L_{i+2}, \dots, L_n)$ is the lower path segment that connects to the new receiver. Traffic is injected from ER_1, forwarded along the path until it reaches CR_i. In CR_i traffic is duplicated and one copy is sent to the lower path segment towards the new receiver.

In this network, the goal of the admission control mechanism upon the issuance of new group join request is for the receiver to probe the lower segment to ensure QoS for its potential multicasting flow while guaranteeing QoS for existing group members. The admission control mechanism at the receiver determines whether to admit a new join request or not based on the probing results. As shown in the Figure 2.1, there is an existing multicasting tree where receivers 1 and 2 are already on that multicasting group. Now, when a new receiver issues a join request, the multicasting tree can be expanded by the addition of the path segment $(L_{i+1}, L_{i+2}, \dots, L_n)$, referred to as the lower path segment. We assume that the flow rate of the existing multicast group is r, before the new receiver joins the group, and thus the multicast group traffic rate of link $(L_1, L_2, \dots, L_i)$ is r, and the multicast group traffic rate of link $(L_{i+1}, L_{i+2}, \dots, L_n)$ is zero. When a new receiver joins the multicast group, the multicast group traffic rate of link $(L_1, L_2, \dots, L_i)$ is still r, but multicast group traffic rate of link $(L_{i+1}, L_{i+2}, \dots, L_n)$ increase from zero to r. Therefore, if link $(L_{i+1}, L_{i+2}, \dots, L_n)$ is to support the multicast group traffic QoS, it's

essential now to estimate the lower path segment resources to achieve the needed QoS over that lower path segment. The proposed scheme focuses mainly on probing this path segment $(L_{i+1}, L_{i+2}, \dots, L_n)$.

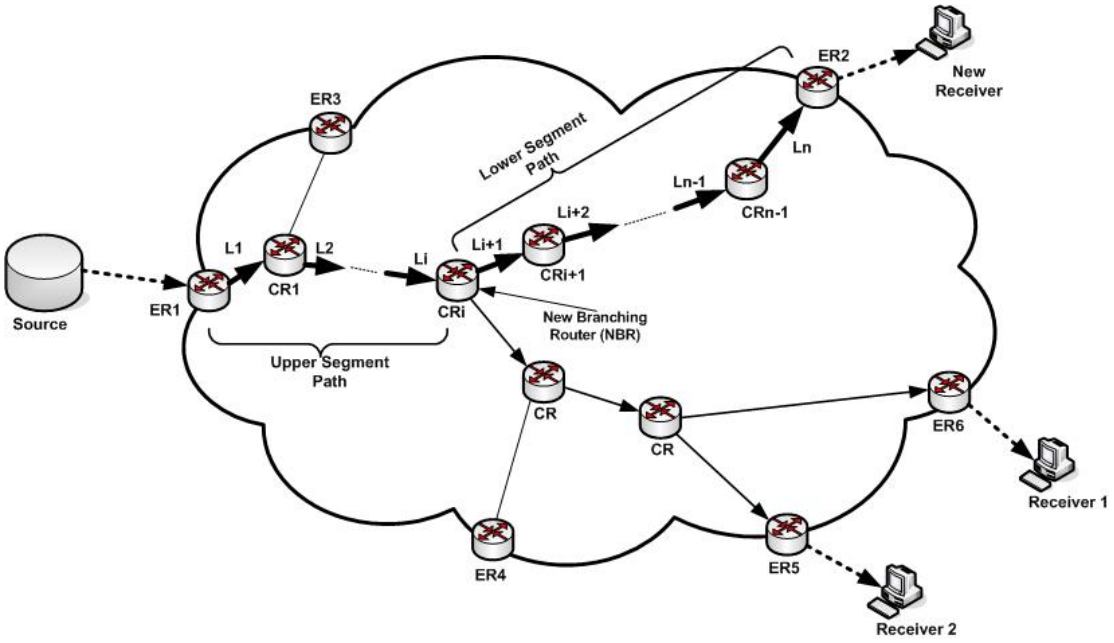


Figure 2.1 The receiver joins a multicasting tree

2.2 Related Works

The network capacity that represents the maximum possible transfer rate that is allowed, is a key metric for network design, resource management and admission control. Moreover, the network capacity can significantly affect communication quality for the end users. In past years, many methods have been proposed for the estimation of the network capacity. We can classify capacity measurement into two categories.

One category is *passive* measurement [22] - [25] where nodes inside the network cooperate with each other to monitor and measure data traffic. In [25], Katti uses equally spaced mode gaps

in TCP flows packet inter-arrival time distributions to detect multiple bottleneck capacities in their relative order. This scheme does not inject any probing traffic into the network. In [24], Katabi determines bottleneck link based on the observation that aggregated arrival trace from flows that share a bottleneck has very different statistics from those that do not share a bottleneck. References [22], [23] and [25] attempt to infer capacity information from packet trace of a TCP connection.

Another category is *active* probing, where probing packets are injected into network to measure network resource, such as [26] - [34]. While passive measurement is simple and easy to implement, active measurement is more accurate and efficient. Active measurements do not depend on cooperation from intervening routers. A vast majority of current capacity measurement techniques employ active approaches to estimate network capacity. Reference [26] is one of the earliest researches that use traditional probing packet pair to measure bottleneck capacity. References [27] and [35] extend packet pair technique by using three back-to-back probing packets and packet trains to measure network path capacity. This method helps to filter distorted probing packets. In [32] and [33], Lai is the first to use different sized packet probing technique to measure link capacity, but this method requires some information of all preceding link capacities. Reference [29] extends this technique to measure segment path capacity. Reference [31] combines delay as well as dispersion measurement of packet pairs to filter out samples distorted by cross traffic to improve measurement accuracy. In [28] and [34], the key element of the technique is to measure the round trip time from the source to each hop of the path as a function of packet size. By controlling the TTL field of the IP header to force probing packet to expire and is dropped by the router. The router return ICMP messages back to the source. The source uses the ICMP packets to measure the round trip time to that hop. By

measuring the ICMP response packet to inference can be made about the link characteristics.

Our scheme belongs to the category of active probing schemes that measure network capacity by injecting probing packets. Available bandwidth is more accurate to reflect the network condition. However we measure the network capacity instead of available bandwidth because the available bandwidth measurement suffers long probing time and is not stable. Group switching scheme compensates the disadvantage of capacity. The key contribution of the proposed scheme is that it utilizes the tree structure of multicasting network, skips the upper segment path to measure the lower segment path capacity by measuring the dispersions between different sized probing packets while current algorithms only measure full path bottleneck link capacity. Our scheme is tightly suitable for multicast environment and performs much better efficient than regular full path measurement. Our scheme is also much more robust to the disturbance of cross traffic.

2.3 Probing Technique

2.3.1 Regular Packet Pair Dispersion

References [5] and [8] i packet pair model to measure bottleneck link bandwidth. For a path composed of links $L_1, L_2, \cdots, L_n$ with bandwidths $C_1, C_2, \cdots, C_n$, respectively, if packet length is L, then the transmission delay is $T_i = \frac{L}{C_i}\ \forall i \leq n$, Packet pair model injects back-to-back packet pair PP into the path. $L(P)$ is the size of packet P. The transmitting time of packet P in link 1 is $T_1 = \frac{L(P)}{C_1}$. It is also the dispersion time Δ_1 between the two packets at the sender side, $\Delta_1 = T_1$ since the two packets are injected back to back. Depending on the next link bandwidth along the path and assuming no cross traffic at node i, we have three cases to consider;

1) Bandwidth of link i is larger than bandwidth of link $i-1$, $C_i \geq C_{i-1}$, as shown in Figure 2.2 (a), the packet transmitting time at link i is $T_i = \frac{L(P)}{C_i} \leq \Delta_{i-1}$, hence $\Delta_i = \Delta_{i-1}$

2) Bandwidth of link i is less than bandwidth of link $i-1$, $C_i < C_{i-1}$, Figure 2.2 (b), and the packet transmission time at the link i is $T_i = \frac{L(P)}{C_i} < \Delta_{i-1} - T_{i-1}$, hence $\Delta_i = \Delta_{i-1}$

3) Bandwidth of link i is less than bandwidth of link $i-1$, $C_i < C_{i-1}$ and the packet transmission time at the link i is $T_i = \frac{L(P)}{C_i} \geq \Delta_{i-1} - T_{i-1}$, Figure 2.2 (c). The second packet will queue behind the first packet before the first packet is sent into link i from link $i-1$, hence $\Delta_i = \frac{L(P)}{C_i}$.

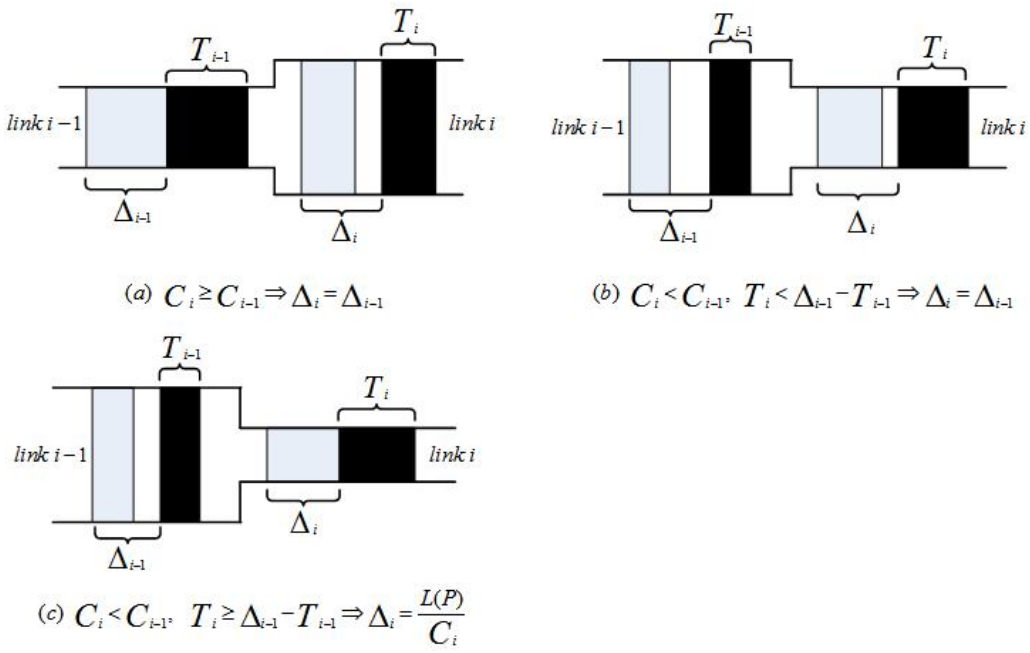


Figure 2.2 Packet pair is transmitted from link $\boldsymbol{i-1}$ to link $\boldsymbol{i}$

In both cases (a) and (b), we notice that the dispersion at link i, Δ_i, is the same value at link $i-1$, Δ_{i-1}, meaning that there's excess bandwidth at link i. In other words link i isn't a bottleneck link. On the other hand in case (c), we notice that the dispersion at link i, Δ_i, has now

a different value from that at link $i-1$, Δ_{i-1}, meaning that link i is a bottleneck link and caused a change in the dispersion between the two packets. Therefore when packet pair reaches destination n, the dispersion Δ_n reflects the total dispersion between the packet pair experienced as a result of the minimum bandwidth link. Thus measuring the arriving dispersion Δ_n which is the difference between the last bit of first packet arrival to the arrival time of the last bit of the second packet, leads to the determination of the bottleneck link bandwidth C_{min}. Note that this is under the assumption that there is no cross traffic along the path. Thus at the receiver side, $\Delta_n = \frac{L(P)}{C_{min}}$. Thus measuring the dispersion at the destination will lead to the calculation of the bottleneck link bandwidth C_{min} and eventually to the right admission control decision. The regular packet pair probing is normally used to measure the full path capacity for the receiver in a network.

2.3.2 Different Sized Packet Probing Technique

As discussed earlier, in packet pair probing, two same sized packets are injected into the network and by measuring the packet dispersion we are able to determine the bottleneck link bandwidth. However, in different packet probing techniques, the two packets that are injected are of different length (a large packet P_L followed by a small packet P_S) and this should help in estimating the path segment resources (in this case the lower segment in Figure 2.1). This technique is first introduced by [8] and extended by [29]. Lai and Baker in [29] design a train probing technique to measure any segment bandwidth. In a unicast environment, as large packet P_L always has longer transmission delay than small packet P_S over the same link., that's

$\frac{L(P_L)}{C} > \frac{L(P_S)}{C}$, where $L(P_L)$ is the length of P_L , $L(P_S)$ is the length of P_S and C is link bandwidth , and according to Reference [29], its *Lemma 2* states;

When packet pair is injected back to back into the path $L_1, L_2, \cdots, L_n$ with bandwidths $C_1, C_2, \cdots, C_n$. $P_L$and P_S have same destination at the end of link L_n. If $\forall i \leq n$, $\frac{L(P_L)}{L(P_S)} \geq \frac{C_{i+1}}{C_i}$, then $P_L P_S$will remain back to back along the path. This can be explained as follows and by referring to Figure 2.3.

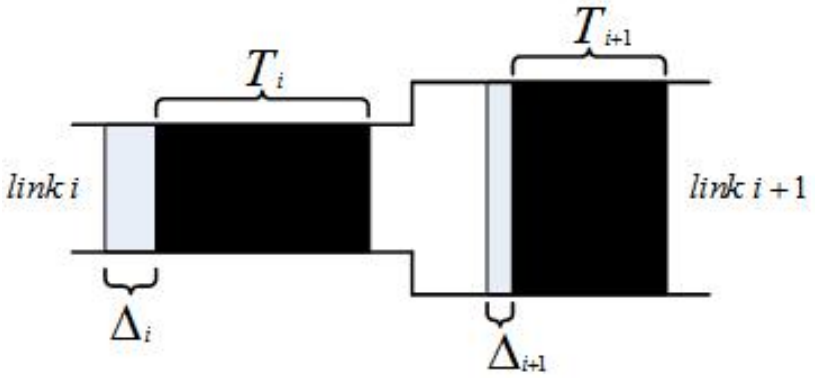


Figure 2.3 Packets are transmitted from link link $\boldsymbol{L_i}$ to link $\boldsymbol{L_{i+1}}$

We deduce another property from different sized packet probing technique to force same sized packet pair to stay back-to-back in link L_i. If $P_S P_S P_L$ are back-to-back and inserted into the path $L_1, L_2, \cdots, L_n$ with bandwidths $C_1, C_2, \cdots, C_n$, $T_{L,i} = \frac{L(P_L)}{C_i}$ is the transmission time of P_L over link L_i. $T_{S,i} = \frac{L(P_S)}{C_i}$ is the transmission time of P_S over link L_i. $\Delta_{max} = \frac{L(P_S)}{minC_j}, j \leq i$ is the largest dispersion between these two packets before they reach L_i. The condition of $P_S P_S P_L$ to stay back-to-back over link L_i is $T_{L,i} \geq \Delta_{max}$ and $(T_{L,i} - \Delta_{max}) + T_{L,i} \geq \Delta_{max}$, that is, $\frac{L(P_L)}{C_i} - \frac{L(P_S)}{minC_j} + \frac{L(P_S)}{C_i} \geq \frac{L(P_S)}{minC_j}$. $\frac{L(P_L)+L(P_S)}{2L(P_S)} \geq \frac{C_i}{minC_j}, j \leq i$.

Under this condition, a large sized packet is first injected in front of the two same sized packets to force them stay back-to-back till they reach the NBR. The NBR discards the large sized packet and continue to forward the two same sized packets along the lower path segment.

The goal in our technique is to measure bottleneck link bandwidth in the lower path segment by using the same sized two back-to-back packet injections. That means we have to make sure that these packets stay back-to-back when they travel along the upper path segment until they reach NBR. Implementing the different sized packet pair, we use a larger packet in front of the two small sized packets to keep these packets back-to-back from the source to the NBR (basically we are forcing the two packets to stay always back-to-back till they reach the NBR). Thus a large packet size is first injected in front of the two same sized packets to force them to stay back to back till they reach the NBR. The NBR discards the large packet size and continue to forward the other two small packets along the lower path segment. In summary, the reason for using different sized packet pair technique here is to force the probing packet pair to stay back-to-back till they reach the NBR. The different sized probing packets can be used to measure the segment path capacity for the receiver in multicast network.

2.3.3 Cross Traffic Effect

In this section, our goal is to analyze influence of cross traffic to probing packets in multicast environment. We show that different sized packet pair probing is more robust to cross traffic than packet pair probing.

Definition 1: Affected packet pair: when dispersion between packets in packet pair has been increased or reduced due to cross traffic, we say cross traffic affected the packet pair.

Definition 2: Affected different sized packet pair: when back to back probing packets dispersion has been increased because of cross traffic, we say cross traffic affected different sized packet pair.

We consider these two cases:

a) Cross traffic arrives in front of the first probing packet:

For packet pair probing, First probing packet will suffer additional delay, as shown in Figure 2.4 (a)

$$\Delta_i = f(x) = \begin{cases} T_i & if\ T_i \geq \Delta_{i-1} \\ T_i & if\ T_i + d_1 \geq \Delta_{i-1}, T_i \leq \Delta_{i-1}, d_1 \geq 0 \\ \Delta_{i-1} - d_1 & if\ T_i + d_1 < \Delta_{i-1}, d_1 \geq 0 \end{cases}$$

$T_i = \frac{L(P)}{C_i}$ is the transmission time of packet P over link L_i. And

d_1 is first probing packet queuing delay caused by cross traffic.

Without cross traffic, Δ_i should equal to Δ_{i-1}, however with cross traffic the value of Δ_i is now different. For example, when $T_i \geq \Delta_{i-1}$ means that link i has smaller bandwidth than link $i-1$. Queuing delay caused by cross traffic to first packet cannot bring any influence to the dispersion between the two packets. This is due to the fact that the second packet will be queued up back-to-back with first packet before first packet is transmitted to link i even if there was no cross traffic, $\Delta_i = T_i$. When $T_i \leq \Delta_{i-1}$, dispersion should stay unchanged if there is no queuing delay caused by cross traffic. Because of cross traffic's influence, $T_i + d_1 \geq \Delta_{i-1}$, the second packet is forced to queue up after first packet. Dispersion is going to be changed to $\Delta_i = T_i$. In this case dispersion will not show correct bottleneck link bandwidth anymore. When $T_i + d_1 < \Delta_{i-1}$, even though the second packet doesn't suffer queuing delay, dispersion is still changed. So cross traffic always change packet pair dispersion.

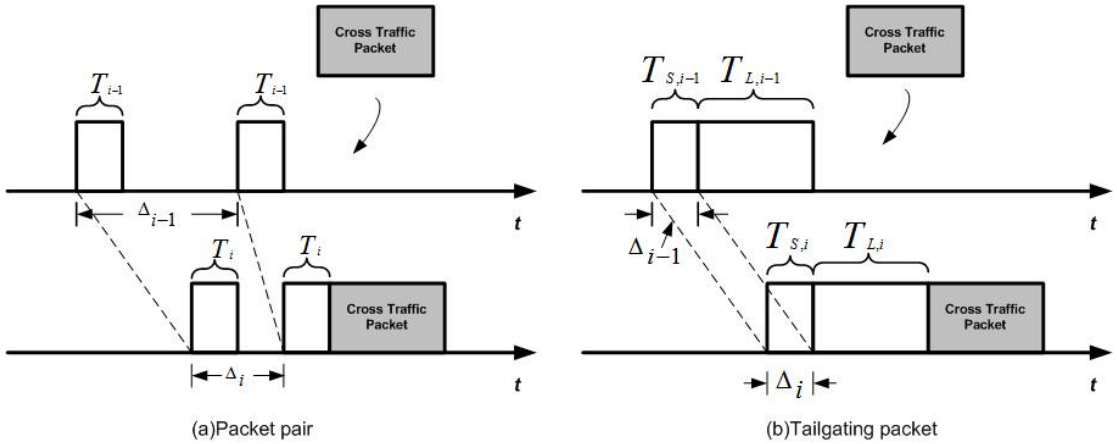


Figure 2.4 Cross traffic's influence to probing packets

With different sized packet pair, we already knew the property: $\Delta_{i-1} \leq \frac{L(P_L)}{C_i}$, it makes large and small packets stay back to back along the entire path.

$T_{L,i} = \frac{L(P_L)}{C_i}$ is the transmission time of P_L over L_i.

$T_{S,i} = \frac{L(P_S)}{C_i}$ is the transmission time of P_S over L_i

If there is no cross traffic, large packet suffer no queuing delay while small packet suffer $T_{L,i} - \Delta_{i-1}$ queuing delay. $\Delta_i = T_{S,i}$. Suppose cross traffic is inserted in front of large probing packet, it will introduce queuing delay (d_1) to large probing packet. To small probing packet, the queuing delay is $T_{L,i} + d_1 - \Delta_{i-1}$. The time of last bit of large packet leave node i is $T_L = T + d_1 + T_{L,i}$.

T: The last bit of large packet is inserted into queue.

The time of last bit of small packet leave node i is

$T_S = (T + \Delta_{i-1}) + (T_{L,i} + d_1 - \Delta_{i-1}) + T_{S,i} = T + T_{L,i} + d_1 + T_{S,i}$.

So $\Delta_i = T_s - T_L = T_{S,i}$, as showed in Figure 2.4 (b)

From the above calculation, cross traffic that is inserted in front of large packet has no effect on packet dispersion.

b) Cross traffic is injected between packet pair:

Since for packet pair probing dispersion between probing packets is $\Delta_{packet\ pair} = \frac{L(P)}{\min_{k<i} C_k}$, while for different sized packets probing dispersion between probing packets is $\Delta_{diff\ packet\ pair} = \frac{L(P_S)}{C_i}$.

Since in different sized packet probing technique, we choose the second probe packet with the smallest possible packet size while packet pair probing chooses regular size packet, we have $L(P) \geq L(P_S)$ and $\min_{k<i} C_k \leq C_i$ => $\Delta_{packet\ pair} \geq \Delta_{diff\ packet\ pair}$

In other words, packet pair dispersion is always larger than different sized packet pair dispersion. We assume cross traffic can be modeled as Poisson process with traffic arriving rate λ. So the probability of cross traffic is introduced into network at a given moment is $p = 1 - e^{-\lambda\tau}$, τ is the dispersion between probing packets. For packet pair probing, $\tau = \Delta_{packet\ pair}$, $p_1 = 1 - e^{-\lambda\Delta_{packet\ pair}}$. For different sized packet probing, $\tau = \Delta_{diff\ packet\ pair}$, hence $p_2 = 1 - e^{-\lambda\Delta_{diff\ packet\ pair}}$, so $p_1 \geq p_2$. Cross traffic has larger probability to affect packet pair probing than different sized packet probing. Then packet pair probing is more vulnerable than different sized packet probing.

We should also notice that introducing higher priority to the probe packets over the normal priority cross traffic will eliminate most of the normal priority cross traffic effect on our measurements. In our proposed scheme such higher priority to the probe packets is introduced.

In summary, packet pair probing is more vulnerable to cross traffic's influence than different sized packet probing. Thus replacing upper path segment packet pair probing with different sized probing reduces the probing measurements error.

2.3.4 Probing Packet Train Process

In this section, we describe in details the proposed multicast admission control mechanism. For a multicasting group, there is only one ingress router and multiple egress routers in the network. We assume at least one receiver has already joined the multicasting group, and a new receiver wants to join that multicasting group.

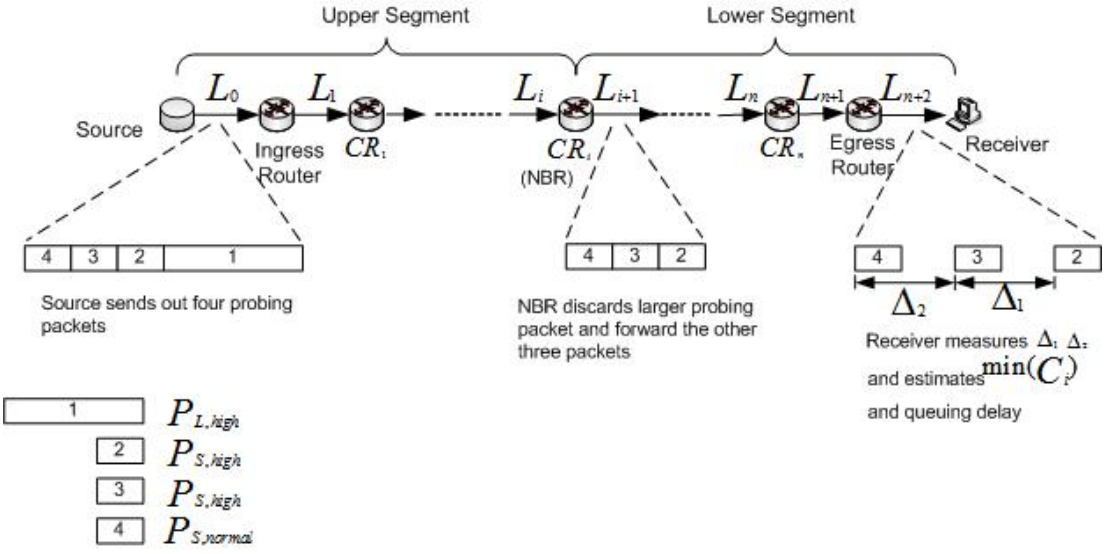


Figure 2.5 Probing packets train traveling from source to destination

2.3.4.1 Probing train structure

Figure 2.5 shows the path from the source to the new receiver. We first construct the probing packet train. Let $P_{S,normal}P_{S,high}P_{S,high}P_{L,high}$ be the probing packet train, with $P_{L,high}$ being the first packet to be injected into the path, and it is a large size probing packet with high priority.

$P_{S,high}$ is the small probing packet with high priority.

$P_{S,normal}$ is the small probing packet with normal priority.

Where $L(P)$ is the size of packet P and

$L(P_{L,high}) \gg L(P_{S,high}) = L(P_{S,normal})$.

These four packets are back to back injected into the network from the source side. Both $P_{S,high}$ and $P_{S,normal}$ are destined to the new receiver, while $P_{L,high}$ is destined to the NBR, i.e. CR_i (Figure 2.1). As shown earlier, since the flow whose rate is r over the existing multicast tree must travel over the upper path segment, and it must meet the QoS requirement, the flow rate must be less than the minimum link bandwidth in the upper path segment $\min_{k<i} C_k$, i.e. $r < \min_{k<i} C_k$.

In order to ensure the probing packet pair stay back-to-back while traveling in the upper path segment $(L_0, L_1, \dots, L_i)$ we need to satisfy the property $\frac{L(P_L)+L(P_S)}{2L(P_S)} \geq \frac{C_i}{minC_j}, j \leq i$, if packets pair satisfies $\frac{L(P_L)+L(P_S)}{2L(P_S)} \geq \frac{C_i}{r}, j \leq i$, they definitely satisfy $\frac{L(P_L)+L(P_S)}{2L(P_S)} \geq \frac{C_i}{minC_j}, j \leq i$. Hence we adjust the probe packet lengths to satisfy $\frac{L(P_L)+L(P_S)}{2L(P_S)} \geq \frac{C_i}{r}, j \leq i$ in order to insure that $P_{S,high}P_{S,high}P_{L,high}$ stay back-to-back until they reach NBR.

Also we make sure that $P_{S,high}$ and $P_{S,normal}$ are injected into the network back to back with different priorities. This allows to measure the queuing delay across the entire path $(L_0, L_1, \dots, L_i, L_{i+1} \dots, L_n, L_{n+1}, L_{n+2})$.

2.3.4.2 Probing process

A. Receiver-side procedure

When the potential new receiver sends a join message to the existing multicasting tree, the source will inject the probe packets $P_{S,normal}P_{S,high}P_{S,high}P_{L,high}$ along the upper path segment($L_0, L_1, \dots, L_i$) when it receive the request message that is initiated from the new receiver. At the same time, the new receiver starts admission control function to monitor the network for potential probing packets arriving from the source, as shown in Figure 2.6.

Admission control logic at the receiver side measures and records arriving time of all probing packets, assuming that the arriving time of first $P_{S,high}$ in the packet train is t_1, and the arriving time of second $P_{S,high}$ in the packet train is t_2, and arriving time of $P_{S,normal}$ is t_3.Then, $\Delta_1 = t_2 - t_1$ is dispersion between the two small equal size packets $P_{S,high}$. According to the packet pair probing technique, we calculate bottleneck link bandwidth of the lower path segment, $min_{segment}(C_i) = \frac{L(P_{S,high})}{\Delta_1}$. Clearly, in any real network, the influence of cross traffic cannot be ignored, it may decrease or increase the dispersion between probing packets as discussed earlier. So repeated measurements can identify dispersion distortion and reduce it to a minimum level. We measure and record each estimation. Then we compare these measurement results with a specified minimum required bandwidth threshold α.

The equation $\Delta_2 = t_3 - t_1$ is the queuing delay suffered by the normal priority probing packet $P_{S,normal}$ plus Δ_1. So queuing delay is $d_q = \Delta_2 - \Delta_1$. We compare it with a specified delay threshold β. We already know that queuing delay reflect network congestion condition. If queuing delay is much larger than threshold β, that means at least one link along the path suffer

congestion. Based on the measurements of queuing delay d_q and the estimating of $min(C_i)$, a decision is made whether to admit or reject as follows. If $min(C_i)$ is greater than the threshold bandwidth α and the estimated queuing delay is less than , then admit. Otherwise deny admission. Below Table 2.1 lists conditions of accepting or rejecting multicast admission request.

Table 2.1 Admission decision logic

Estimated bottleneck link bandwidth / Estimated queuing delya(dispersion)	$\min(C_i) < \alpha$	$\min(C_i) \geq \alpha$
$d_q \leq \beta$	reject	accept
$d_q > \beta$	reject	reject

Based on the decision of admission control logic, if new receiver's request is accepted, new receiver generates join message and forward it toward source.

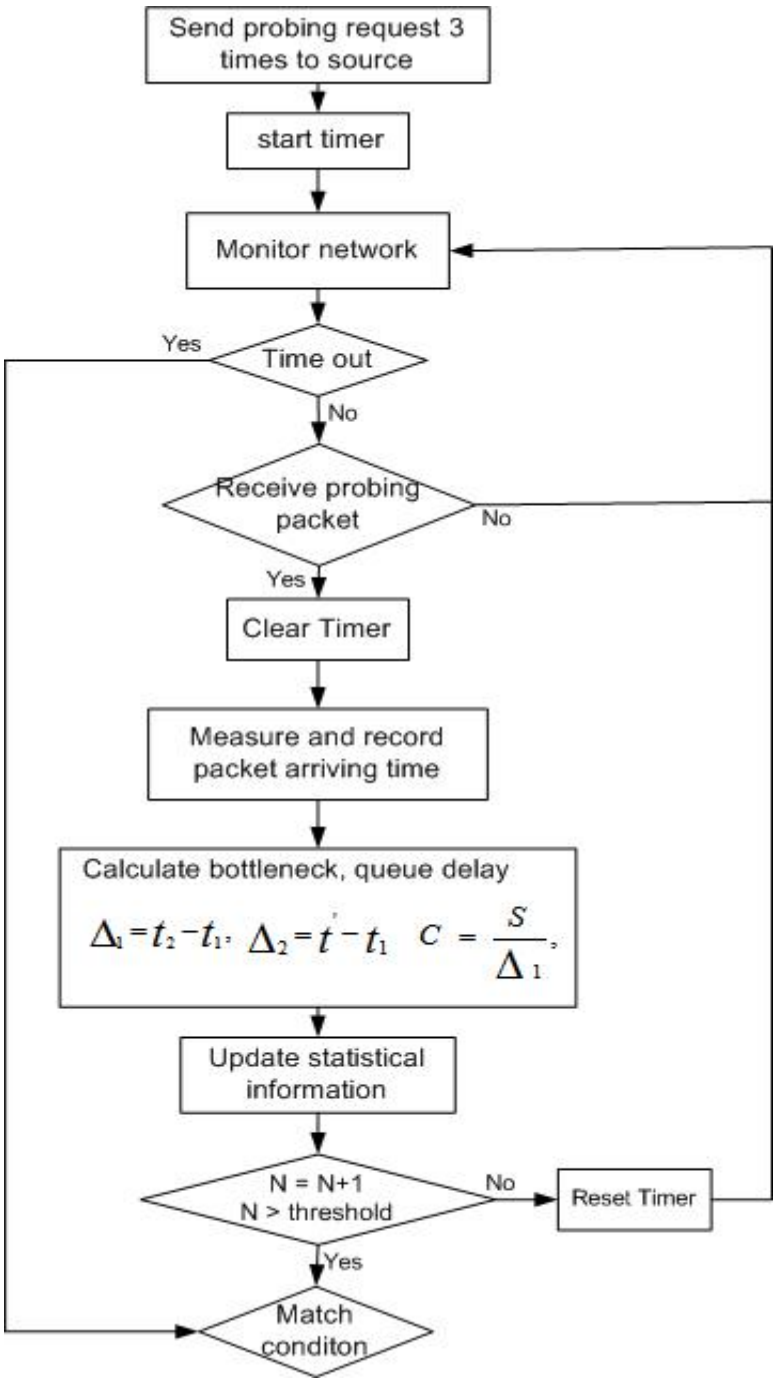


Figure 2.6 Receiver-side procedure

B. Sender-side procedure

When the source node receives the request message that is sent from potential new receiver, it generates the probing packet train as discussed earlier, $P_{S,normal}P_{S,high}P_{S,high}P_{L,high}$ along the multicasting tree and towards the potential new receiver with $\frac{L(P_L)+L(P_S)}{2L(P_S)} \geq \frac{C_i}{minC_j}, j \leq i$. The $P_{L,high}$ is discarded by NBR, while the destinations of other three probing packet are set to the potential new receiver.

The source node uses a simple and fast method that is used by some other papers to find out the location of NBR. The source node sends out a testing packet train to this group address. The TTL of train packet is $1,2,3\cdots,n$. n is the hops from potential new receiver to source node. When TTL is zero, the packet will be discarded by the router. At the same time the router send back an ICMP responding packet to the source node. If the source node receives ICMP responding packet pair having the same TTL value, the testing packets just go through a branching router. After testing, the ICMP responding packet pair with largest TTL reflects the branching router that is closest to potential new receiver. That is, this branching router is NBR. Using this way, we can infer the distance between NBR and source node. Therefore, we can set TTL of $P_{L,high}$ to make sure it is discarded by NBR.

The source node sends out probing trains along the packet tree toward new receiver.

2.4 Simulation and Results

In this section, we evaluate the performance of the proposed multicasting admission control technique using network simulation OPNET [36], and compare between the proposed scheme and full path probing based admission control scheme.

2.4.1 Simulation Environment

We consider the detailed network shown in Figure 2.7. In the shown network topology, CR_1, CR_2, and CR_3 are three core routers. ER_1, ER_2, ER_3, ER_4, ER_5, and ER_6 are edge routers. Source generates multicasting traffic for group members. Receiver_1 and Receiver_2 are two existing group members who receive multicast group traffic from the source. Node_1, Node_2, Node_3, and Node_4 introduce four different cross traffic into the network with destinations and sources are chosen as shown in Figure 2.7. The bottleneck link is chosen to be within the lower path segment. The small probing packet size is 40 bytes while the large probing packet size is 1500 bytes. The simulation time is always 30 minutes, and probing train is repeatedly sent 1000 times to make sure we get large enough samples to analyze.

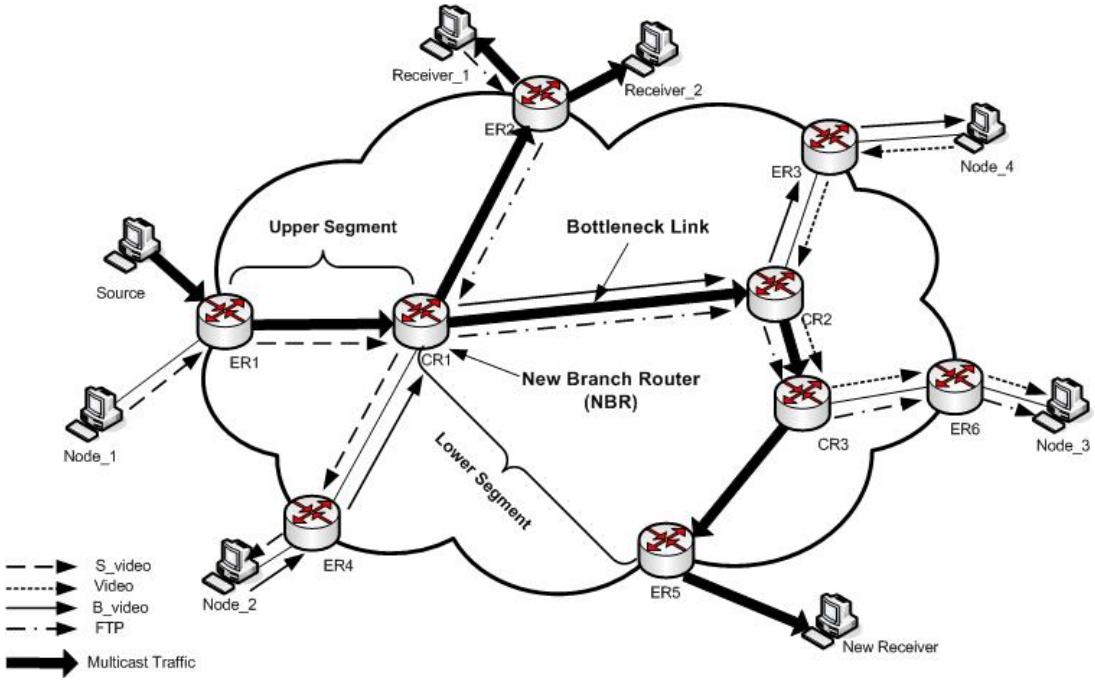


Figure 2.7 The simulated multicasting network topology

First we consider the case that cross traffic rates are below bottleneck link bandwidth. The parameters of the traffic flows are listed in Table 2.2. The link that connects CR1 and CR2 is the bottleneck link. Bandwidth of link CR1-CR2 is 55Kbps while bandwidths of other links are 1,544Kbps.

Table 2.2 Traffic parameters

Flow Name	Packet Size (bytes)	Transmission Rate (bits/sec)
B_video	2000	41600
FTP	2000	560K
S_video	2500 bytes	460K
Video	80 bytes	518

2.4.2 Accuracy of Probing Measurements

Here we adjust the cross traffic sending rate to observe its influence on probing packets dispersion measurements. In this section, the process of probing and measurements is conducted repeatedly, and the cumulative distribution function (CDF) for the estimated bottleneck link bandwidth is obtained. We compare the estimation precision between the proposed probing policy and traditional full path probing policy. The proposed probing policy uses the different sized packet to probe the network while the full path probing policy uses the regular packet pair to probe the network.

First, we consider the case when there is only cross traffic (S_video) in upper path segment of the network. S_video traffic is upper path segment cross traffic. No cross traffic is forwarded through lower path segment. Parameters of S_video are listed in the Table 2.3 below:

Table 2.3 Traffic parameters for S_video flows

Scenarios	Packet Size(bytes)	Interarrival Time(sec)
Scenario_1	2500	0.4
Scenario_2	2500	0.2
Scenario_3	2500	0.025

Figure 2.8 - Figure 2.10 show the Cumulated Distribution Function (CDF) of the estimated bottleneck link bandwidth, F(x) for both the full path probing measurements and path segment probing measurements. In Figure 2.8, when cross traffic rate is low, we observe that cross traffic has little influence to both full path probing and path segment probing. When we increase cross traffic payload, as shown in Figure2.9 and Figure 2.10, full path probing is affected more by cross traffic than path segment probing. As discussed earlier in section 2.2, cross traffic effect on different sized probing packet train is much lower than traditional packet pair. This proposed scheme obviously has a better performance than full probing when used to measure bottleneck link bandwidth.

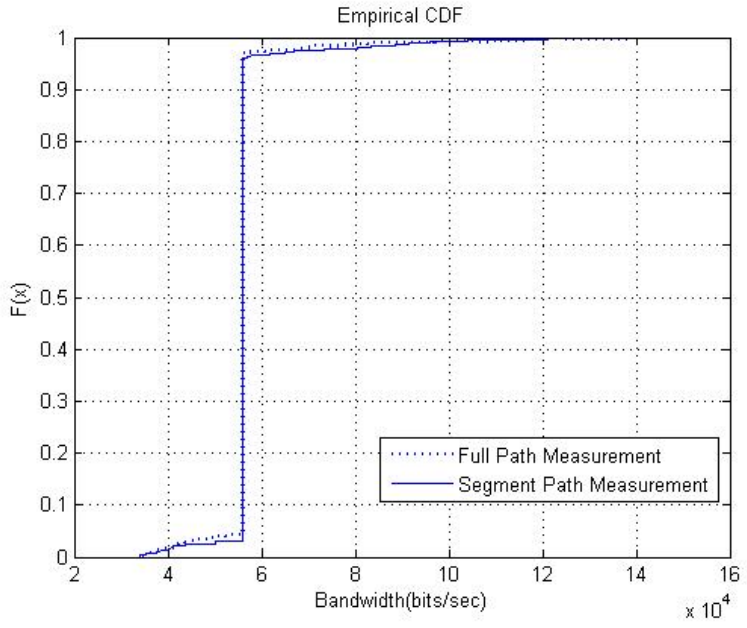


Figure 2.8 Scenarios_1 (cross traffic interarival time is 0.4 sec)

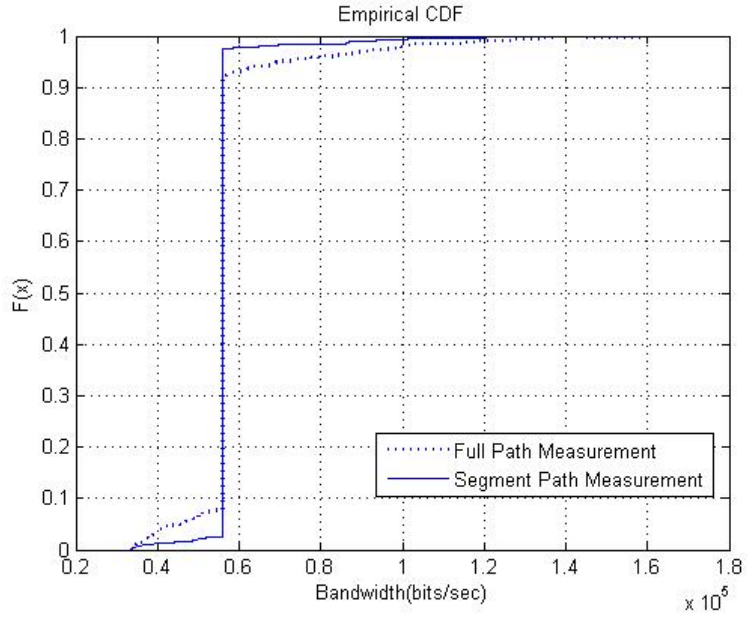


Figure 2.9 Scenarios_2 (cross traffic interarrival time is 0.2 sec)

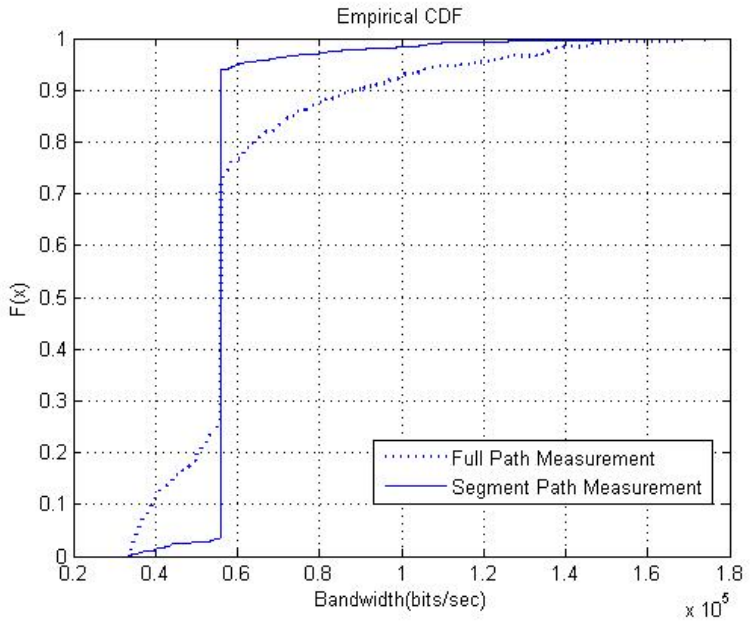


Figure 2.10 Scenarios_3 (cross traffic interarrival time is 0.025 sec)

Here, we introduce cross traffic (video traffic) to the lower path segment. Table 2.4 is the parameters of video traffic:

Table 2.4 Traffic parameters for video

Scenarios	Packet Size(bytes)	Interarrival Time(sec)
Scenario_4	2000	0.6
Scenario_5	2000	0.14
Scenario_6	2000	0.02

Figure 2.11 - Figure 2.13 show the CDF of the bottleneck link bandwidth when cross traffic goes through the lower path segment. When cross traffic sending rate is increased, measurement precision decreases for both full path measurement and path segment measurement. The curves show they suffer similar cross traffic influence. In other words, when cross traffic is flowing

through the lower path segment, the proposed scheme has similar performance to full path probing.

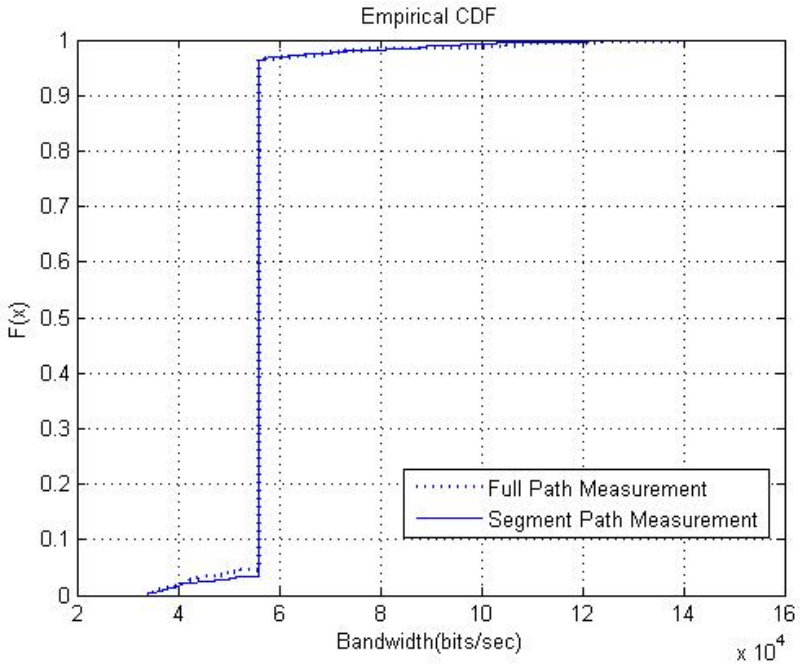


Figure 2.11 Scenarios_4 (cross traffic interarival time is 0.6 sec)

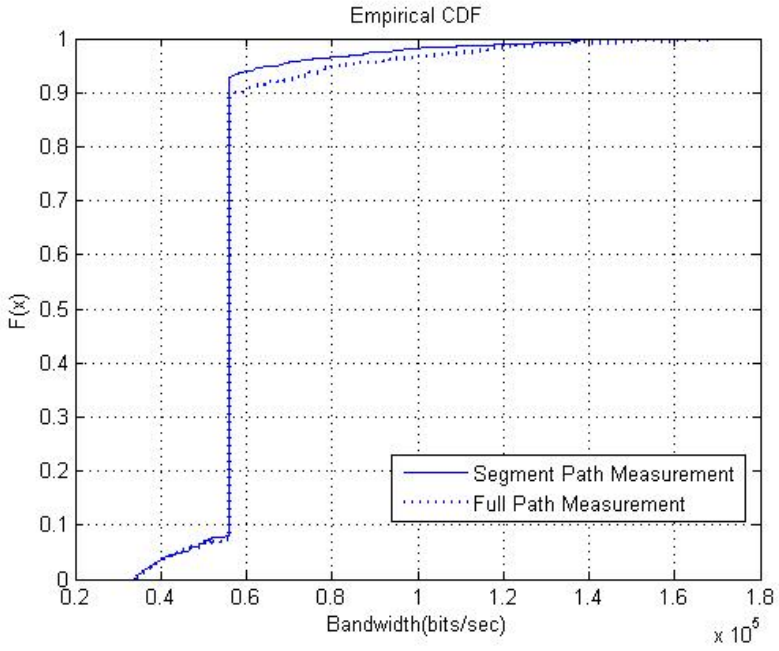


Figure 2.12 Scenarios_5 (cross traffic interarrival time is 0.14 sec)

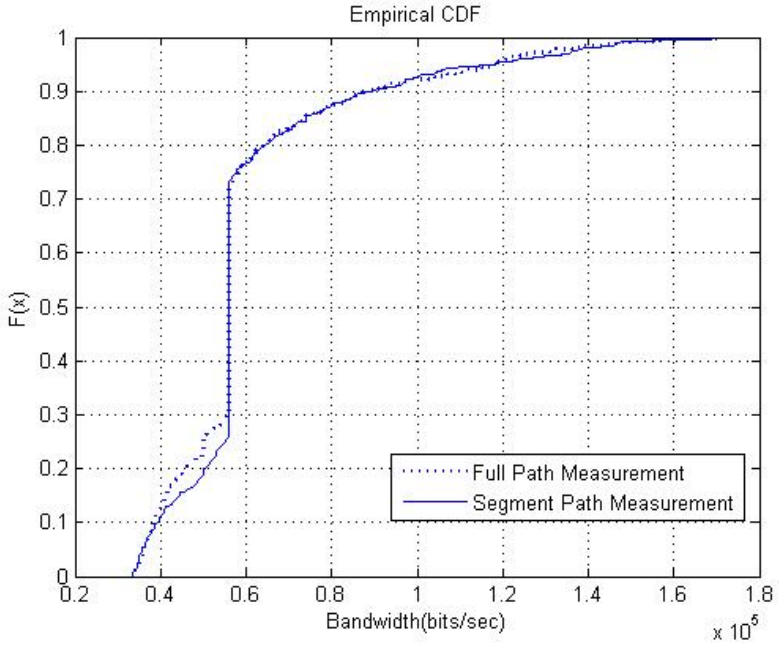


Figure 2.13 Scenarios_6 (cross traffic interarrival time is 0.02 sec)

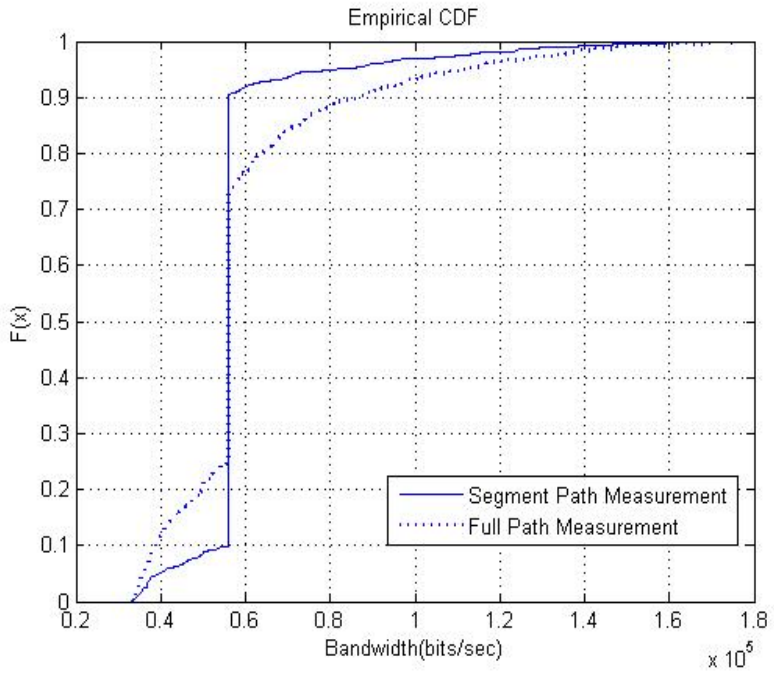
Figure 2.14 presents the curves as cross traffic go through the entire network. As shown in Figure 2.7. There are four different cross traffic in the network. S_video flows through upper

segment. B_video and FTP go through the full path from upper segment to lower segment, and video flows through the lower segment. Table 2.5 lists the parameters of those cross traffic.

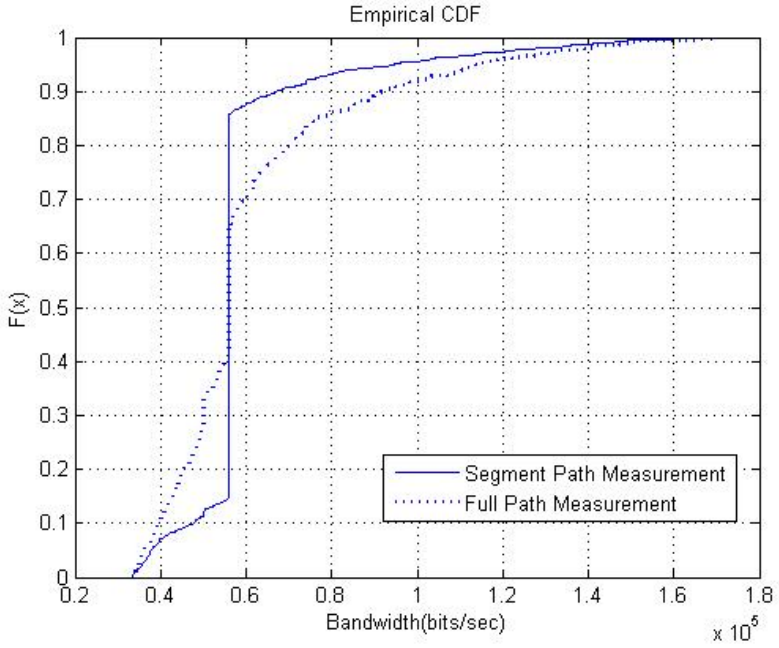
Table 2.5 Traffic parameters for these four cross traffic

Scenario Name	Traffic Name	Packet Size(bytes)	Interarrival Time(sec)
Scenario_7	S_video	2500	0.15
	B_video	2000	0.10
	FTP	2000	0.3
	Video	80	0.15
Scenario_8	S_video	2500	0.1
	B_video	2000	0.05
	FTP	2000	0.3
	Video	80	0.09

As shown in Figure 2.14, our path segment measurement scheme has much better performance than full path measurement.



(a) Scenario_7



(b) Scenario_8

Figure 2.14 Mix cross traffic along the whole path

Figure 2.15 shows the measurement time that need for infer the correct path capacity under different cross traffic condition. It is obviously that the measurement logic need more time to get reasonable result under large cross traffic load. At the same time, the measurement time of segment path measurement is always less than the time of full path measurement needs under the same cross traffic load. When cross traffic load is over 30Kbits/second, the time of full path measurement increase sharply. That is also proved in last section.

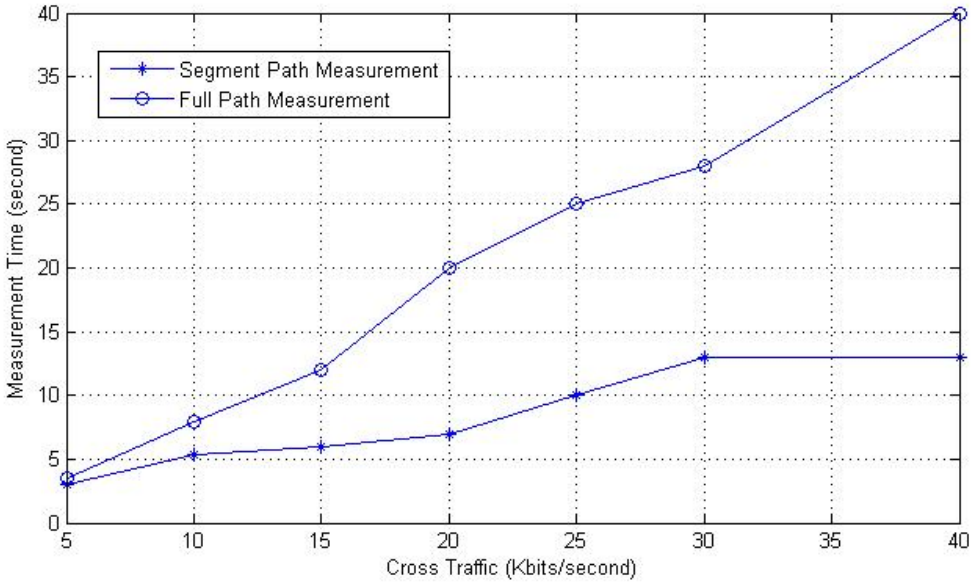


Figure 2.15 Cross traffic rate vs. measurement time

2.4.3 Measurement Accuracy for Different Path Capacity

In previous section, we compare the full path measurement with segment path measurement for the same link capacity under different cross traffic rate. Those measurement results show the segment measurement has better performance than the full path measurement. In this section, we compare the measurement error rate for the different link capacity under the same cross traffic condition.

We introduce cross traffic into the network with 10kbits/second sending rate and 40kbits/second. We modify the network path capacity with different value. Figure 2.16 and Figure 2.17 show the measurement error rate of full path measurement is always higher than the measurement result of segment path measurement no matter what the cross traffic rate. At the same time, the measurement error rate decrease with the increasing of the link capacity. Under the same cross traffic condition, the measurement precision for the large capacity link is much better than the small capacity link. We discussed in section 2.2, the dispersion between probing packets is related to the link capacity. Narrow link capacity tends to extend the dispersion between probing packets. Therefore the packet pair dispersion for 56k path capacity is larger than the packet pair dispersion for 2M path capacity. Large dispersion suffers large probability to be disturbed by cross traffic.

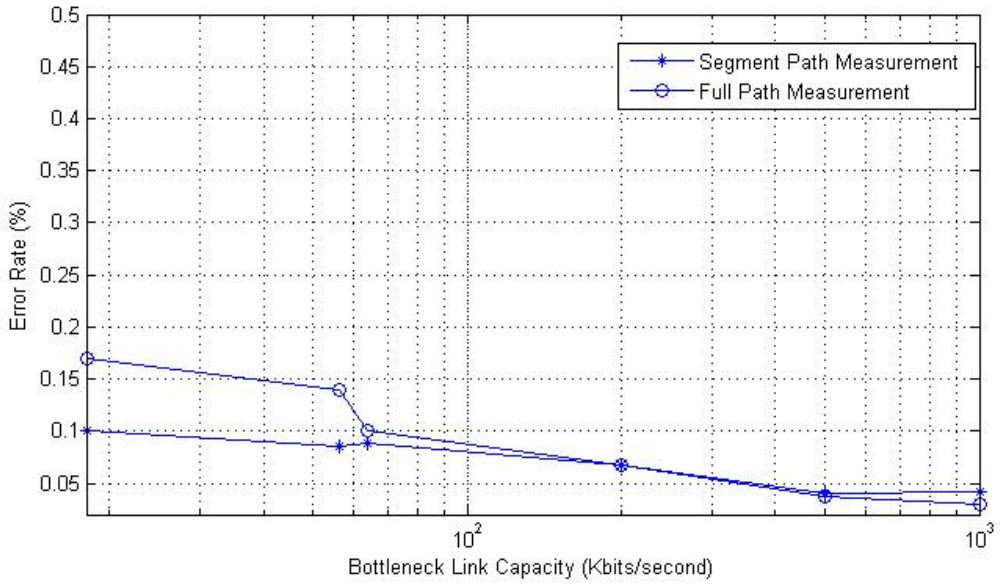


Figure 2.16 Error rate vs. link capacity when cross traffic rate is low

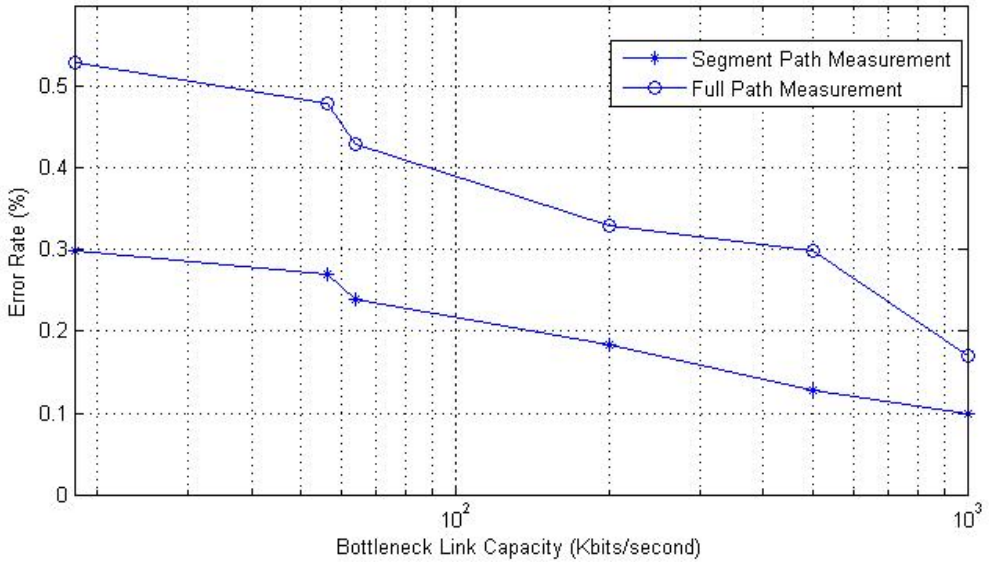


Figure 2.17 Error rate vs. link capacity when cross traffic rate is high

2.5 Conclusion

We have presented a path segment measurements based multicasting admission control scheme. In our scheme, admission decision is made for each potential receiver at edge routers. Our most important contribution is that we depend on measuring the bandwidth of a path segment instead of the full path. This proves to improve measurement precision, and avoids invalid measurement, especially under interference of cross traffic. At the same time, we develop a delay measurement scheme based on priority packet pair probing.

Chapter 3
Single Link Capacity Measurement Technique

The measurement technique is the key important part for admission control logic. Many researches have been done on how to measure delay, available bandwidth and packet loss because these factors reflect the network condition more directly. Few researches focus on the single link capacity measurement. In last chapter, we discussed the segment path capacity measurement approach in multicasting network. It is clear that replacing the path capacity measurement with segment path capacity measurement is more efficient and more robust from the disturbance of cross traffic because segment path capacity skips the common path measurement that is shared by the same group receivers. In this chapter, we further dig into the measurement technique in multicast network. Based on the segment path measurement we discussed in the last chapter, we improve the packet train probing method to measure any single link capacity along the probing path. Single link capacity is a valuable factor for networking analysis. Collection of all network link capacity gives you the distribution of network processing ability. It provides a way to rationally assign the network resource and schedule the traffic to avoid potential network congestion caused by overload the bottleneck link. Some methods never consider the network resource distribution. They inject their traffic into the network without any

policy. Their policy is to dynamically adjust the traffic load to fit the network condition in real time. Those kinds of schemes are called post-adjustment. These schemes suffer a problem that deteriorates the network temporary before the sender adjusts the traffic load. However if the sender know the network topology, at least the link capacity distribution, it can adjust the sending rate or choose a better path that has higher path capacity to send out the traffic. The Link capacity information also can be used for network resource management and network topology control. In multicast network, network link capacity information allows us to optimize multicast tree structure. The potential new receiver grafts the new tree branch to the existed multicast tree in order to receive the group streaming. Normally a receiver can find multiple paths to graft itself to the multicast tree. The location of graft point decides the performance of the multicast tree. With the assistance of link capacity information along the branch, the receiver has the ability to avoid certain narrow link and graft to the multicast tree through a link with high capacity.

The chapter is organized as follows: firstly, we propose a novel way to measure the any single link capacity by injecting a serial probing packet train and discuss the critical conditions to ensure the probing validity. Secondly, we describe the probing process and apply the single link measurement to determine the new branch point that grafts the potential receiver to the existed multicast tree. Thirdly, the simulation results for our proposed scheme are presented.

3.1 Single Link Capacity Measurement and Properties

We inject the packet train into the network, as we mentioned in last chapter. However the packets inside the train have different size and follow the different switching behavior. The basic idea of this scheme is to inject a pair of probing packet trains (each packet train is composed of

only three packets, as will be shown later) from the source node into the network. The middle probing packets will be discarded by the intermediate routers that connect the two ends of the target link while other probing packets continue to be relayed to the receiver node, as shown in Figure 3.1. The inter-arrival times of probe packets at that intermediate router will be preserved to the receiver. Measuring the dispersion at the destination will lead to the calculation of the target link capacity based on the theory that will be shown below.

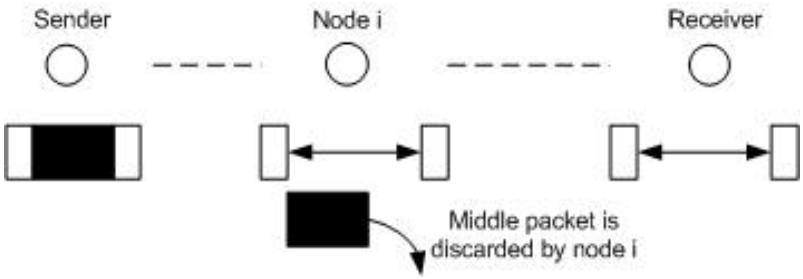


Figure 3.1 Single packet train along the path

This technique in effect requires only the interaction between the source and the receiver. The algorithm is thus independent of the core network and is edge-based. As our scheme does not require special functionalities at the intermediate routers, other than enabling packet prioritization typical in DiffServ-type networks, it is thus suitable for networks where edge node does not have visibility or access to the core of the network.

3.1.1 The Condition for Different Sized Packets to Stay Back-to-back

When two packets that are injected back-to-back are of different length (a large sized packet P_L followed by a small sized packet P_S), the large sized packet $P_L$always has longer transmission

time than small sized packet P_S over the same link, that's $\frac{L(P_L)}{C} > \frac{L(P_S)}{C}$, where $L(P_L)$ is the length of P_L, $L(P_S)$ is the length of P_S and C is the link bandwidth. If the two packets $P_S P_L$ are inserted back-to-back into the path $L_1, L_2, \cdots, L_n$ with bandwidths $C_1, C_2, \cdots, C_n$, $T_{L,i} = \frac{L(P_L)}{C_i}$ is the transmission time of P_L over link L_i. The dispersion at a given point is defined as the time duration from the time when the last bit of the first packet arrives to the time when the last bit of the second packet arrives, as shown in Figure 3.2. Note that $\Delta_{max} = \frac{L(P_S)}{minC_j}, j \leq i$ is the largest dispersion between these two packets before they reach L_i. Thus in order to force these two packets stay back-to-back in link L_i, the transmission time of the first packet P_L must be larger than the maximum dispersion between P_L and P_S, i.e. $T_{L,i} \geq \Delta_{max}$, that is $\frac{L(P_L)}{C_i} \geq \frac{L(P_S)}{minC_j}, j \leq i$ =>

$$\frac{L(P_L)}{L(P_S)} \geq \frac{C_i}{minC_j}, j \leq i \qquad (3.1)$$

In other words, if equation (3.1) is met, we guarantee that the two packets $P_S P_L$ stay back-to-back till they reach node i.

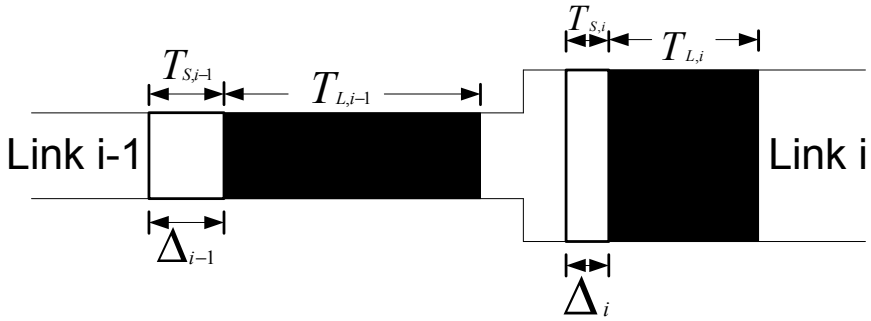


Figure 3.2 $\boldsymbol{P_S P_L}$ are forwarded from $\boldsymbol{L_{i-1}}$ to $\boldsymbol{L_i}$

3.1.2 The Condition for Different Sized Packets to Stay Apart

When two different sized packets (a small sized packet P_S followed by a large sized packet P_L) are injected back-to-back into the network, the dispersion between these two packets may increase as they travel through the path since the small sized packet is always transmitted faster than the larger sized packet along the same link. However if packets travel over a link with large capacity to a link with small capacity, it is possible for the large sized packet to queue behind the small sized packet. In this section, we determine the condition to ensure packets separate when they go through from link $i-1$ to link i.

If the two packets $P_L P_S$ are back-to-back inserted into the path $L_1, L_2, \cdots, L_n$ with bandwidths $C_1, C_2, \cdots, C_n$, $T_{L,i} = \frac{L(P_L)}{C_i}$ is the transmission time of P_L over link L_i. $T_{S,i} = \frac{L(P_S)}{C_i}$ is the transmission time of P_S over link L_i. Δ_i is the dispersion of these two packets when they go through from link L_{i-1} to link L_i, $i \le n$, as shown in Figure 3.3. It is easy to see that

$$\Delta_i = \begin{cases} T_{L,1} & i = 1 \\ T_{L,i} & T_{S,i} > \Delta_{i-1} \text{ and } i > 1 \\ \Delta_{i-1} + (T_{L,i} - T_{S,i}) & T_{S,i} \le \Delta_{i-1} \text{ and } i > 1 \end{cases} \quad (3.2)$$

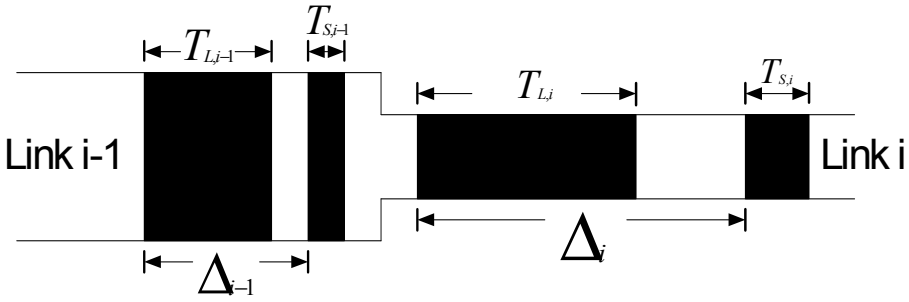


Figure 3.3 $\boldsymbol{P_L P_S}$ are forwarded from $\boldsymbol{L_{i-1}}$ to $\boldsymbol{L_i}$

Observed from Figure 3.3, $\Delta_i > T_{L,i}$ is the condition that P_L and P_S stay apart when they are

forwarded from link L_{i-1} to link L_i. From equation (3.2), it is easy to derive that $\Delta_i > T_{L,i}$ if and only if $\Delta_i = \Delta_{i-1} + (T_{L,i} - T_{S,i})$.

If P_L and P_S stay apart from link L_2 to link L_i, it has to satisfy $\Delta_k > T_{L,k}, \forall k \le i$, that is $\Delta_k = \Delta_1 + \sum_{j=2}^{k}(T_{L,j} - T_{S,j}) > T_{L,k}, k \ge 2, \forall k \le i$, ($\Delta_1$ is the dispersion when $P_L P_S$ go through link L_1, $\Delta_1 = \frac{L(P_L)}{C_1}$).

$$\Delta_1 + \sum_{j=2}^{k}(T_{L,j} - T_{S,j}) > T_{L,k}, k \ge 2, \forall k \le i$$

$$\Delta_1 + \sum_{j=2}^{k-1}(T_{L,j} - T_{S,j}) + (T_{L,k} - T_{S,k}) > T_{L,k}, k \ge 3, \forall k \le i$$

$$\Delta_1 + \sum_{j=2}^{k-1}(T_{L,j} - T_{S,j}) > T_{S,k}, k \ge 3, \forall k \le i$$

$$\frac{L(P_L)}{C_1} + (L(P_L) - L(P_S)) \sum_{j=2}^{k-1} \frac{1}{C_j} > \frac{L(P_S)}{C_k}, \qquad k \ge 3, \forall k \le i$$

$$\frac{L(P_L)}{L(P_S)} > \frac{\frac{1}{C_k} + \sum_{j=2}^{k-1} \frac{1}{C_j}}{\frac{1}{C_1} + \sum_{j=2}^{k-1} \frac{1}{C_j}}, k \ge 3, \forall k \le i \quad (3.3)$$

Equation (3.3) is the sufficient condition for $P_L P_S$ to stay apart when they travel along the path $L_1, L_2, \cdots, L_i$. C_{min} is the bottleneck link capacity from L_1 to L_k.

$$\frac{\frac{1}{C_k} + \sum_{j=2}^{k-1} \frac{1}{C_{min}}}{\frac{1}{C_1}} > \frac{\frac{1}{C_k} + \sum_{j=2}^{k-1} \frac{1}{C_j}}{\frac{1}{C_1} + \sum_{j=2}^{k-1} \frac{1}{C_j}} \quad (3.4)$$

$$\frac{\frac{1}{C_k}+(k-2)\frac{1}{C_{min}}}{\frac{1}{C_1}} > \frac{\frac{1}{C_k}+\sum_{j=2}^{k-1}\frac{1}{C_j}}{\frac{1}{C_1}} \quad (3.5)$$

From (3), (4) and (5)

$$\frac{L(P_L)}{L(P_S)} > \frac{\frac{1}{C_k}+(k-2)\frac{1}{C_{min}}}{\frac{1}{C_1}}, k \geq 3, \forall k \leq i \quad (3.6)$$

If $\frac{L(P_L)}{L(P_S)}$ satisfies (3.6), then it satisfies (3.3)

The condition to ensure P_L never queue behind P_S from link L_2 to link L_i is

$$\frac{\frac{1}{C_k}+\frac{(k-2)}{C_{min}}}{\frac{1}{C_1}} < \frac{\frac{1}{C_{min}}+\frac{(k-2)}{C_{min}}}{\frac{1}{C_1}} = \frac{C_1(k-1)}{C_{min}} \quad (3.7)$$

From (3.6), (3.7), if

$$\frac{L(P_L)}{L(P_S)} > \frac{C_1(k-1)}{C_{min}}, k \geq 3, \forall k \leq i \quad (3.8)$$

Then the two packets stay apart from each other, hence

$$\frac{L(P_L)}{L(P_S)} > \frac{\frac{1}{C_k}+\sum_{j=2}^{k-1}\frac{1}{C_j}}{\frac{1}{C_1}+\sum_{j=2}^{k-1}\frac{1}{C_j}}, k \geq 3, \forall k \leq i$$

In summary, when packet train satisfies equation (3.8), the back-to-back injected packets $P_L P_S$ will have time dispersion when go through the entire path.

3.1.3 Dispersion Requirement between Packet Trains

The proposed scheme uses a pair of packet probing trains to measure the single link capacity,

as shown in Figure 3.4. If $P_{S1}P_{L1}P_{S1}$ and $P_{S2}P_{L2}P_{S2}$ are two probing trains that are inserted into the path $L_1, L_2, \cdots, L_n$ with bandwidths $C_1, C_2, \cdots, C_n$, $T_{j,i} = \frac{L(P_j)}{C_i}$ is the transmission time of P_j over link L_i, $L(P_{S1}) = L(P_{S2})$, $L(P_{L1}) = L(P_{L2})$.

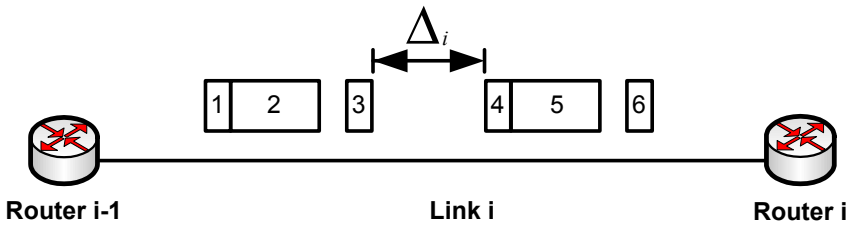


Figure 3.4 Dispersion between two packet trains

In order to make sure the first probing train does not affect the second probing train, Δ_1 (the initial dispersion between two probing trains) should be large enough to make sure the first P_{S2} of packet train 2 never queue behind the second P_{S1} of packet train 1 when packet trains travel through the network, that is, $\Delta_i > 0, \forall i \leq n$.

Suppose the packet trains satisfy equations (3.1) and (3.8) in *Section 3.1* as we have shown earlier. Dispersion between two packet probing trains is Δ_i in link L_i.

$$\Delta_i = \Delta_{i-1} - \left(T_{S1,i} + T_{L1,i}\right) - T_{S2,i}, i \geq 2$$

$$T_{S1,i} = T_{S2,i}$$

So $\Delta_i = \Delta_{i-1} - T_{L1,i}, i \geq 2$

$$\Delta_i = (\Delta_{i-2} - T_{L1,i-1}) - T_{L1,i}, i \geq 2$$

$$\Delta_i = \Delta_1 - \sum_{j=2}^{i} T_{L1,j}, i \geq 2$$

In order to ensure that the two packet trains don't affect each other, the condition is $\Delta_i > 0$, that's

$$\Delta_i = \Delta_1 - \sum_{j=2}^{i} T_{L1,j} > 0, i \geq 2$$

$$\Delta_1 > \sum_{j=2}^{i} T_{L1,j} = L(P_{L1}) \sum_{j=2}^{i} \frac{1}{C_j}, i \geq 2 \quad (3.9)$$

$$\sum_{j=2}^{i} \frac{1}{C_j} \leq \sum_{j=2}^{i} \frac{1}{C_{min}} = \frac{i-1}{C_{min}} \quad (3.10)$$

From (3.9) and (3.10), if $\Delta_1 > L(P_{L1}) \frac{i-1}{C_{min}}$, then

$$\Delta_1 > L(P_{L1}) \sum_{j=2}^{i} \frac{1}{C_j}$$

$$\Delta_1 > L(P_{L1}) \frac{i-1}{C_{min}}, \qquad (3.11)$$

If initial dispersion of probing packet trains satisfies (3.11), the first packet of the second train will not be affected by the last packet of the first train. This is the fundamental condition for the pair of packet trains to measure dispersion correctly.

3.1.4 Single Link Capacity Measurement

In this section, we show the technique that uses packet trains to measure the single link capacity. We suppose that the first probing packet train $P_{S1}P_{L1}P_{S1}$ is injected back-to-back into the path $L_1, L_2, \cdots, L_n$, $L(P_{S1}) < L(P_{L1})$ are of different lengths (a small sized packet P_{S1} followed by a large sized packet P_{L1} and a small sized packet P_{S1}), suppose $L(P_{L1})$ and $L(P_{S1})$ satisfy the equation (3.1) and (3.8) in *Section 3.1.3*. $\Delta_{L1S1,i}$ is the dispersion between P_{L1} *and* P_{S1}, $\Delta_{S1L1,i}$ is the dispersion between P_{S1} *and* P_{L1} and $\Delta_{S1S1,i}$ is the dispersion between

$P_{S1}\ and\ P_{S1}$ when the packet train go through link L_i. The destination of P_{S1} is the receiver node n while the destination of P_{L1} is node i that connects link $L_i, i \leq n$. Thus, when the first packet train travels through link $L_i, i \leq n$

$$\Delta_{L1S1,i} = \Delta_{L1S1,i-1} + \left(T_{L1,i} - T_{S1,i}\right)$$

As discussed in Section II *(C)*

$$\Delta_{L1S1,i} = \Delta_1 + \sum_{j=2}^{i} (T_{L1,j} - T_{S1,j}), 2 \leq i \leq n \quad (3.12)$$

$$\Delta_{S1L1,i} = T_{S1,i} \quad (3.13)$$

From (3.12), (3.13)

$$\Delta_{S1S1,i} = \Delta_{L1S1,i} + \Delta_{S1L1,i} = \Delta_1 + \sum_{j=2}^{i} T_{L1,j}, 2 \leq i \leq n$$

Large sized packet P_{L1} is discarded by intermediate node *i* while two other packets P_{S1} continue to be transmitted toward destination node *n*.

$$\Delta_{S1S1,i} = \Delta_1 + \sum_{j=2}^{i} T_{L1,j} \quad (3.14)$$

As we discussed earlier in section 3.1.3, the dispersion between the two same sized packets is $\Delta_{S1S1,j} = \max\left\{\frac{P_S}{C_j}, \Delta_{S1S1,j-1}\right\}, i \leq j \leq n$ when they travel through the path $L_i, \cdots, L_n$ with initial dispersion $\Delta_{S1S1,i}$. Since, $\Delta_{SS,i} \geq \frac{P_S}{C_j}, \forall j \geq i$, the dispersion between $P_{S1}P_{S1}$ will be $\Delta_{S1S1,i}$ until they arrive at receiver side.

$$\Delta_{S1S1,n} = \Delta_{S1S1,\mathrm{i}} = \Delta_1 + \sum_{j=2}^{i} T_{L1,j} \quad (3.15)$$

If the second probing train $P_{S2}P_{L2}P_{S2}$ is injected back-to-back into path $L_1, L_2, \cdots, L_n$.

$L(P_{S2}) = L(P_{S1})$, $L(P_{L2}) = L(P_{L1})$. The destination of P_{S2} is the receiver node n while the destination of P_{L2} is node $i-1$. $\Delta_{S2S2,i}$ is the dispersion between $P_{S2}\ and\ P_{S2}$ when packet train go through link L_i. Hence,

$$\Delta_{S2S2,i-1} = \Delta_1 + \sum_{j=2}^{i-1} T_{L2,j}$$

$$\Delta_{S2S2,n} = \Delta_{S2S2,i-1} = \Delta_1 + \sum_{j=2}^{i-1} T_{L2,j} \qquad (3.16)$$

From (3.15), (3.16)

$$\Delta_{S1S1,n} - \Delta_{S2S2,n} = \Delta_1 + \sum_{j=2}^{i} T_{L1,j} - \left(\Delta_1 + \sum_{j=2}^{i-1} T_{L2,j} \right) = T_{L1,i}$$

$$\Delta_{S1S1,n} - \Delta_{S2S2,n} = T_{L1,i} = \frac{L(P_{L1})}{C_i}$$

Thus,

$$C_i = \frac{L(P_{L1})}{\Delta_{S1S1,n} - \Delta_{S2S2,n}} \qquad (3.17)$$

In summary and based on the above calculations, the sender node injects a pair of packet probing trains with different middle probing packet destinations. By simply measuring and recording packet dispersions, the receiver node can estimate every single link capacity. This method does not need cooperation of intermediate nodes.

3.2 Active Probing Process

In this section, we describe in details the proposed active probing measurement mechanism to estimate every single link capacity along a given end-to-end path. We assume the given

network path is $L_1, L_2, \cdots, L_n$ with bandwidths $C_1, C_2, \cdots, C_n$. The probing trains are injected from sender *node 0* and received at receiver *node n*. Figure 3.5 shows the transmitting process of packet trains along the path from the sender node to the receiver node.

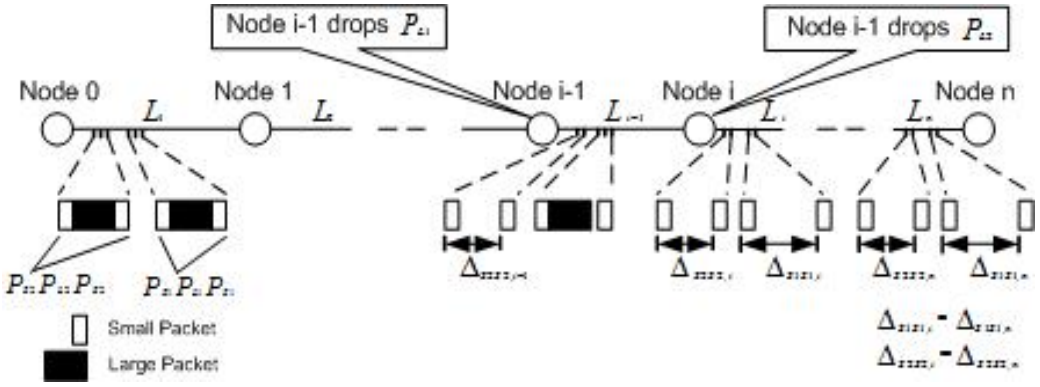


Figure 3.5 Packet train transmitting process

We proceed by first demonstrating the probing train structure.

3.2.1 Probing Train Structure

The probing packet train is composed of three packets; one small packet P_S followed by one large packet P_L, followed by another small packet P_S. Let $P_{S1}P_{L1}P_{S1}$ be the first probing packet train.

These three packets are back-to-back injected into the network from the source side. Both P_S packets are destined to the receiver *node n*, while the destination of P_L is *node i* or *node* $i-1$ if we are going to measure capacity of link L_i. In order to ensure that P_S *and* P_L stay back-to-back in every link and P_L*and* P_S never stay back-to-back, we have to adjust packet sizes to satisfy equations (3.1) and (3.8) in *Section 3.1,* as discussed earlier.

$$\frac{L(P_L)}{L(P_S)} \geq \frac{C_i}{C_{min}}, j \leq i$$

And

$$\frac{L(P_L)}{L(P_S)} > \frac{C_1(k-1)}{C_{min}}, k \geq 3, \forall k \leq i$$

3.2.2 Sender-side Process

When the sender node receives the request message from receiver node, it starts the probing process as follows:

Step 1:

After the sender node receives probing request message from receiver, it records the target link number and starts to monitor the network for possible probing packet pair sent from the receiver node.

Step 2:

The sender node measures and records probing packets sent from receiver node and using traditional method (the packet pair property mentioned in section 3.1) to calculate bottleneck link capacity $C_{min} = \frac{L(P)}{\Delta}$.

Step 3:

Based on C_{min} and target link number *i,* the sender node generates the first packet train $P_{S1}P_{L1}P_{S1}$. The destination of P_S is receiver *node n* while the destination of $P_L$is *node i* that connects the target link L_i. The sender injects this packet train into the path toward the receiver *node n*.

Step 4:

The sender generates the second probing packet train $P_{S2}P_{L2}P_{S2}$. The destination of P_{S2} is the receiver node while the destination of P_{L2} is *node i-1 which* connects the target link L_i.

In order to make sure the second packet train is not affected by the first packet train, the sender node has to wait $L(P_{L2})\frac{n-1}{C_{min}}$ seconds (we discussed this sending dispersion in *section 3.1*) before injecting the second packet train into the network. After $L(P_{L2})\frac{n-1}{C_{min}}$ seconds, the sender node sends out packet train $P_{S2}P_{L2}P_{S2}$ along the path $L_1, L_2, \cdots, L_n$.

The sender side procedure is shown below:

```
Probing process()
{
  Packet = receive();
  Packet header information = readhead();
  If (sequence number legal)
  {
      Record dispersion ;
      If (dispersion > 0)
      {
          While (n < train number)
          {
                  Calculate bottleneck link capacity;
                  Create packet header: destination address;
                  Generate probing packet, encapsulate with header;
                  Calculate sending dispersion;
          }
```

}

Send out the encapsulated packets with calculated dispersion;

}

}

3.2.3 Receiver-side Process

The receiver node initiates link capacity probing by generating a probing request message (this request includes the link number of the link it wants to measure) followed by a couple of traditional packet pair toward the sender node. At the same time, the receiver node starts to monitor the network for potential probing packets arriving from the sender node.

The receiver node measures and records arriving time of all probing packets received from sender node, assuming that the arriving time of the first P_{S1} in the first packet train is t_1, the arriving time of the second P_{S1} in the first packet train is t_2, the arriving time of the first P_{S2} in the second packet train is t_1' and the arriving time of the second P_{S2} in the second packet train is t_2' (P_{L1} and P_{L2} are dropped by intermediate nodes because of the expiration of packet TTL. We configure the TTL field of IP header to the hop number of the target link).

Then, $\Delta = t_2 - t_1$ is dispersion between the two small sized packet P_{S1}. $\Delta' = t_2' - t_1'$ is the dispersion between the two small sized packets P_{S2}. According to the packet train probing technique we discussed in Section 3.1, we can calculate the capacity of link L_i using these measured packet dispersions.

$$C_i = \frac{L(P_{L2})}{\Delta - \Delta'}$$

Clearly, in any real network, the influence of cross traffic cannot be ignored, it may decrease or increase the dispersion between probing packets as discussed earlier. So repeated

measurements can identify dispersion distortion and reduce it to a minimum level. We measure and record each estimation value. Then we calculate link capacity based on these records using statistical averaging methods.

The receiver node can repeat this process to measure any link capacity along the path from the receiver to the sender. These link capacities are important for admission control schemes and resource management, and provide much help to network topology control.

3.3 Simulation and Analysis

In this section, we evaluate the performance of the proposed active probing measurement technique using network simulation OPNET.

We set up a network path *L* that connects sender and receiver as shown in Figure 3.6. Router_1 to Router_8 are intermediate routers. Except the target link, all other link capacities are randomly configured to illustrate various scenarios. We set the target link capacity is 56 Kbps. Cross traffic is modeled by a set of Poisson distribution. By changing parameters of cross traffic and the cross traffic path, we simulate different network scenarios. All experiments in this section are run for a duration of 30 minutes. Note that in each experiment, we repeatedly send probing trains in order to collect enough data for analysis.

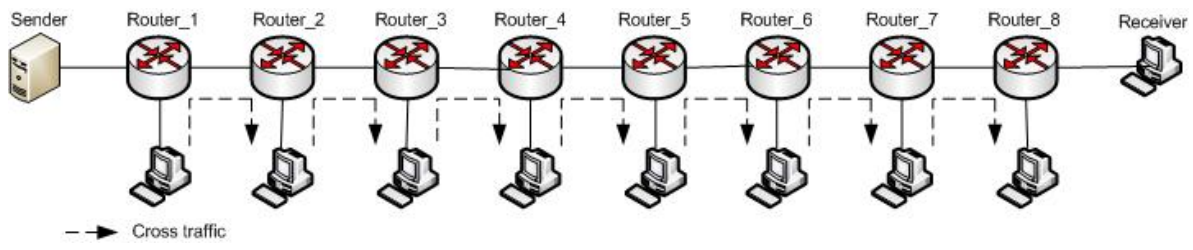


Figure 3.6 Network topology

3.3.1 Impact of Cross Traffic

Let us first investigate the impact of cross traffic to the probing packet train. The target link is link_4 which connect router_3 and router_4.

Figure 3.7 shows the histograms of measuring results when the path is lightly loaded (cross traffic is Poisson traffic with packet size of 500 bytes, average Poisson inter-arrival time is $1/\lambda = 0.36$ travelling through from router_1 to router_8). The proposed scheme exactly estimates the target link capacity with the probing train measurements. Cross traffic has little impact on the measurement results.

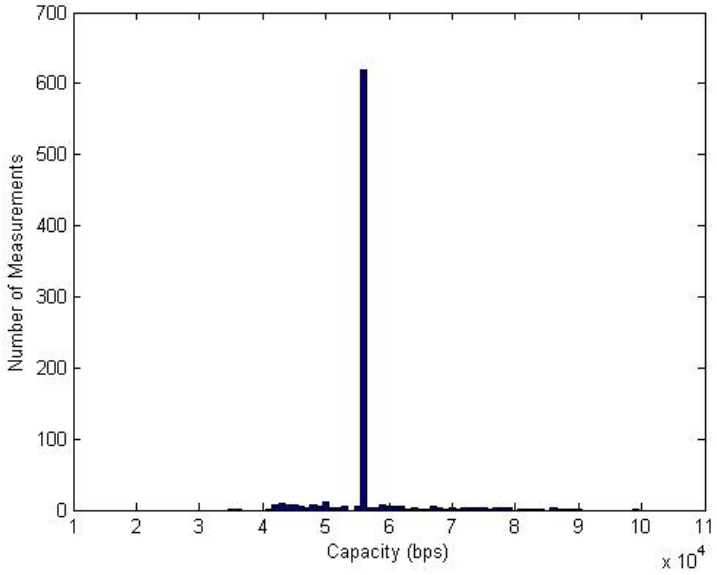


Figure 3.7 Measurement results under light cross traffic load

When the path payload is increased, cross traffic packets will bring certain influence to the probing train measurements. We use two ways to study the effect of cross traffic load, one is to increase cross traffic packet size and the other is to increase the average packet arrival rate. In case 1, we set cross traffic packet size to 1500 bytes. In case 2, we set cross traffic average inter-

arrival time $1/\lambda = 0.12$. Both of the two cases increase cross traffic load by a factor of three. Figure 3.8 shows the histograms of the measured results. We observe that the probing packet train measurements are more sensitive to the cross traffic arrival rate increase than the cross traffic packet size increase. Our observations are consistent since for the same cross traffic load, cross traffic with small packet size has higher arrival rate. Higher arrival rate allows cross traffic packet insertion between the probing packets train leading to the distortion in the dispersion measurements. That is, probing packet measurement is sensitive to small sized cross traffic. At the same time, from Figure 3.8, we observe that the large sized cross traffic packet tends to over-estimate the capacity measurement result while small sized cross traffic tends to under-estimate the measurement results.

When the path is heavily loaded (in case 1, we set cross traffic packet size to 9000 bytes while in case 2, we set cross traffic inter-arrival time $1/\lambda = 0.02$), Figure 3.9 shows that cross traffic brings serious effect to the measurement results. We observe that there is no obvious difference between enlarging packet size and increasing arrival rate since large packets will be fragmented into small packets in IP layer. Therefore, cross traffic in both cases will bring similar effect on the measurement results.

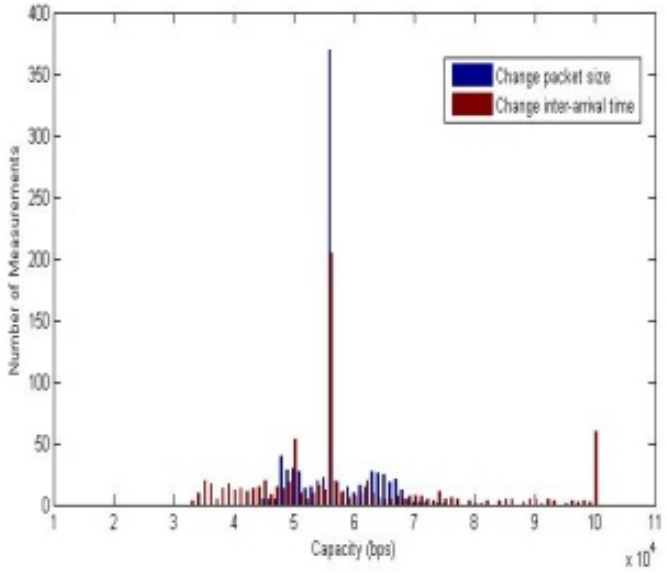


Figure 3.8 Influence of cross traffic with increasing rate and packet size

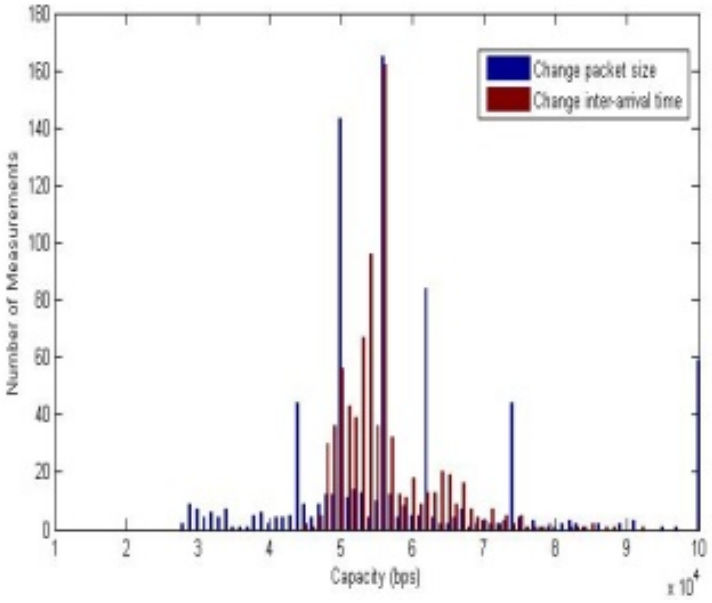


Figure 3.9 Influence of heavy cross traffic

3.3.2 Impact of Target Link Location

The proposed scheme can measure any single link capacity along the path. Now we investigate the influence of target link location on the measurement precision. We set up a new

network scenario that injects same cross traffic to every intermediate router. The cross traffic packets exit the path one hop after they enter the path. This ensures that the target link suffers the same network condition regardless of its location along the path.

Figure 3.10 and Figure 3.11 show the measurement results when the target link is the second link and the seventh link along the path. The cross traffic load is light (packet size is 1000 bytes, inter-arrival time is $1/\lambda = 0.08$). We observe that under the light cross traffic, probing trains can accurately measure both the link capacities for target link_2 and link_7.

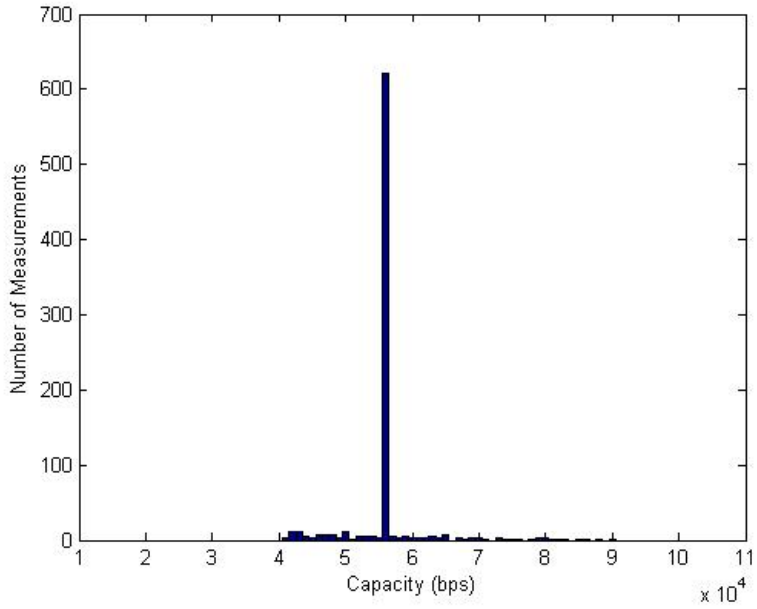


Figure 3.10 Target link is Link_2 with light cross traffic

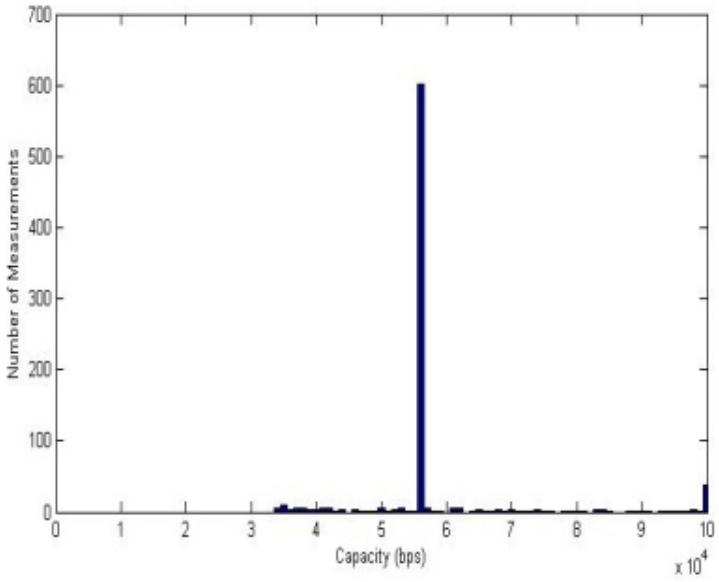


Figure 3.11 Target link is Link_7 with light cross traffic

When we increase the cross traffic load (packet size is 8000 bytes, inter-arrival time is $1/\lambda = 0.04$). In Figure 3.12 and Figure 3.13, we observe that the cross traffic has more effect on the measurement of the second link than the measurement of the seventh link. For the same cross traffic load, when probing measuring the second link capacity, the large sized probing packets are dropped at router_1 and router_2, while dispersions between small sized probing packets will be retained from router 2 till the end of the path in theory. However if the target link is the seventh link, large sized probing packets will be dropped at router_6 and router_7. The dispersion between small sized probing packets will be enlarged hop by hop until they arrive at router_7. That is, when the target link is one of the last few links, the small sized probing packets dispersion in receiver side is much larger than the packet dispersion when target link is one of the first few links. Large dispersion is easily to be affected by cross traffic. That is probing measurements to far target link is more vulnerable to cross traffic's influence than near target link.

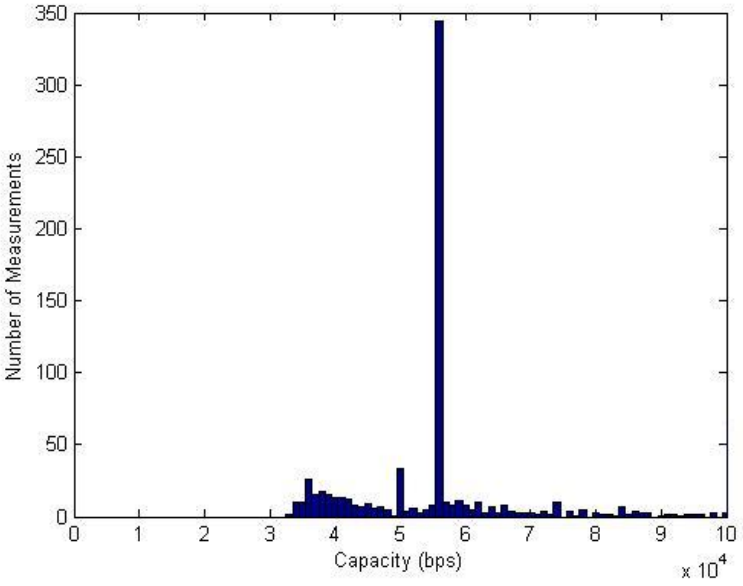


Figure 3.12 Target link is Link_2 with heavy cross traffic

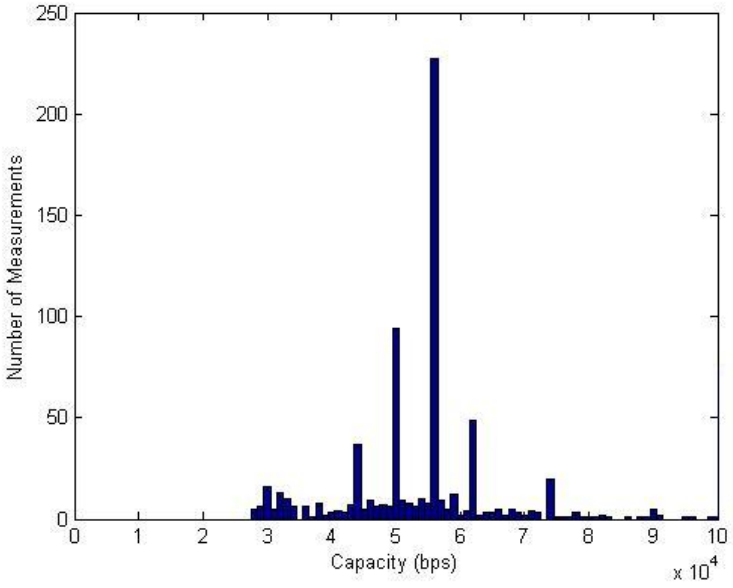


Figure 3.13 Target link is Link_7 with heavy cross traffic

In other words, under the same network condition, the proposed scheme suffers less influence of cross traffic when the target link is close to the sender.

3.3.3 Impact of the Probing Packet Size

In this section, we discuss the impact of the probing packet size. As mentioned earlier in section 3.1, the probing packet train should satisfies equation (3.1) and equation (3.8) in order to ensure that the probing packet train helps in measuring the link capacity accurately. Assuming that the probing packet trains satisfy equations (3.1) and (3.8), we create two scenarios. Note that in both scenarios, we set the small probing packet size to 46 bytes. Cross traffic is Poisson traffic with packet size 1000 bytes and inter-arrival time is $1/\lambda = 0.6$.

Figure 3.14 and Figure 3.15 shown the different measurement results when we change the large probing packet size. When we set a smaller value to the large probing packet size, the queuing time of the second small sized probing packet suffers is short. Dispersion between the two small probing packets is not easily distorted by cross traffic packet as shown in Figure 3.14.

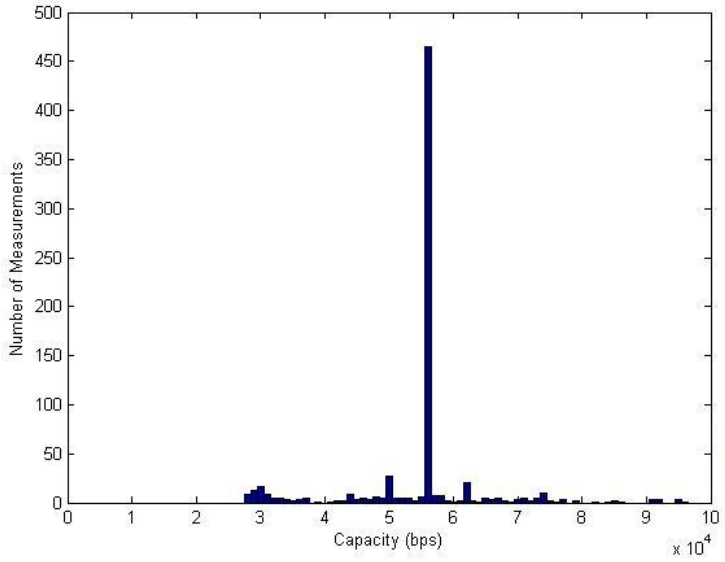


Figure 3.14 Large probing packet with small value

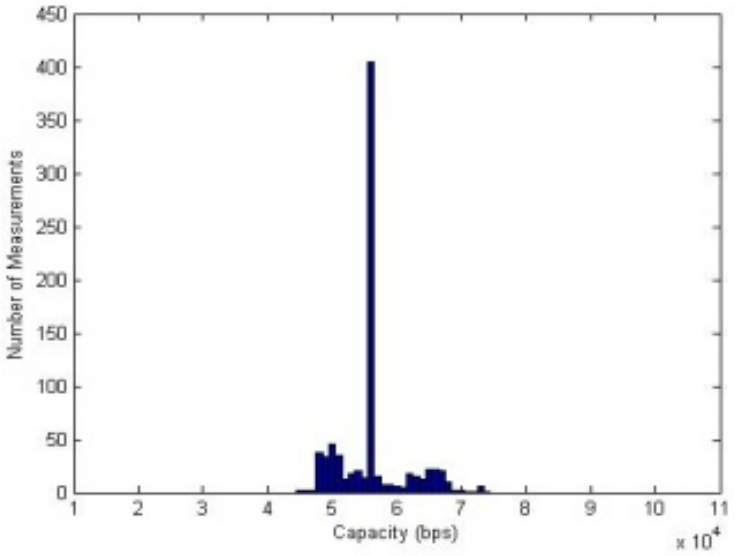


Figure 3.15 Large probing packet with large value

When we set the large sized probing packet with a large value, the queuing delay of the second small sized probing packet will increase. Hence the dispersion between two small sized probing packets will be extended. Thus, cross traffic will have more probability to affect the dispersion between the small sized probe packets. Figure 3.15 shows the measurement errors increase when compared with those of Figure 3.14.

In summary, measurement precision is sensitive to probing packet size. Under the assumption in previous section, the large probing packet with a larger value is vulnerable to the cross traffic effect.

3.4 Conclusion

In this chapter, we proposed a novel scheme for single link capacity measurement in multicast environment. This scheme uses a pair of packet probing trains to measure and estimate single link capacity. Our scheme is an end-to-end probing technique, simple and does not require

cooperation of intermediate routers or time synchronization. Unlike existing schemes that need to know all earlier link capacities, our method only have to know the bottleneck link capacity. Measuring the single link capacity is quite useful in network topology measurement and network resource management.

Future research directions will focus on how to reduce influence of cross traffic by using statistical methods and apply the proposed scheme to network topology measurements.

Chapter 4
SVR based Multimedia Quality Control in Multicast Network

Multimedia quality control has become an important research field especially with the increased QoS requirements. Another kind of quality control is post-quality control, which adjust the streaming sending rate in real time to fit the current available bandwidth in order to avoid network congestion and perform quality control.

In this chapter we propose a novel measurement and machine learning mixed approach for real-time multimedia quality control for multicast networks. This proposed scheme belongs to post-quality control. Most of current adaptive multimedia quality control schemes are based on measurements of available network resources (whether active or passive measurements) or the use of network traffic models and they are reactive in nature. The proposed scheme is based on statistical learning techniques, such as Support Vector Regression (SVR), mixed with network measurements to predict packet loss rate in the near future and thus is a pro-active scheme. The prediction scheme triggers the rate control mechanism to pre-switch the group that provides different streaming rate before network conditions start deteriorating.

At the same time, we also propose a measurement based end-to-end queuing delay approach. This approach only involves the end nodes and Diffserv supported network. The major idea is calculate the dispersion between probing packet with different priority. In Diffserv network,

different priority packet will be injected into different queue, suffer different queuing delay. This feature allows us to infer the end-to-end queuing delay from the network edge.

We use packet dispersion technique to measure end-to-end queuing delay, and combined with video display buffer information to train the acquired dataset for prediction in real time.

4.1 Introduction

Multimedia quality control is a challenge for packet switched network. Currently, numerous applications, such as video conference and video streaming require a guaranteed Quality of Service (QoS) to work properly. However, due to the dynamic and unpredictable features of network traffic, it is hard to reserve enough available network resource that satisfies the multimedia QoS requirements. Therefore real-time quality control schemes attempt to adjust multimedia sending rate in real time to meet current available network resource in order to avoid severe frame loss or intolerant frame delay. The core of quality control is to monitor the network path conditions by using active or passive measurement methods in order to detect the fluctuation of the available network resources. The quality control mechanism is then trigged to adjust the multimedia sending rate. Most of prior research on this problem is divided into two categories. One category is formula-based approach. The other category is measurement-based approach. Formula-based approach applies mathematical models to describe traffic, analyzes and predicts network conditions, examples include Markov chain methods and central limit theorem. Accuracy of these methods depends on the reliability of the assumed models.

Measurement-based methods depend on active or passive network measurements to gather path resource information, such as path delay, packet loss and available bandwidth. Based on these measurements, the control mechanism may send a feedback control message asking the

source to lower its sending rate in order to adapt to the changing network conditions. Clearly these techniques are reactive in nature and they attempt to adjust the sending rate after the network conditions have changed. With this latency in reaction, the received multimedia traffic has already suffered degradation.

Support vector regression (SVR) is a powerful machine learning technique that allows computer to evolve behaviors based on training data. It collects multiple inputs (linear or nonlinear) as training examples and generates the output prediction. It is widely used in various other fields such as finance, vehicular traffic and weather forecast. However, little research has focused on SVR applications to network status prediction. Our innovative approach here proposes to apply SVR algorithm and develops a measurement and a machine learning mixed approach to predict the network condition status in the near future. The prediction is sensitive to changes in the network condition and makes sure multimedia quality can be adjusted to fit the available network resource before the network condition worsens. This scheme brings end customer a better experience by smoothing the rate adjusting process and avoids the possible severe congestion. We also have developed a simple method based on packet dispersion techniques to measure end-to-end queuing delay. This measurement scheme is accurate, lightweight, and doesn't require time synchronization. These measurements, combined with user buffer information are applied as multiple training inputs for the SVR prediction algorithm.

The remainder of this chapter is organized as follows. Section 4.2 summarizes the previous work. Section 4.3 describes the basic of SVR algorithm. Section 4.4 discusses the end-to-end queuing delay measurement, the training and prediction processes. In section 4.5, we evaluate the performance of this proposed scheme using network simulation. Section 4.6 concludes the work.

4.2 Related Works

Support Vector Regression (SVR) algorithm is a novel scheme of machine learning. It is an extension to Support Vector Machine (SVM) that is first proposed by Vapnik of the AT&T Bell Laboratories [37] and [38]. In [39] Smola and Scholkopf discussed the basic principles of SVR. They introduced the basic theory, system model, and outlined some advantages and challenges. SVR has been proposed in various areas. In the financial data prediction, Cao [40] studied the use of SVR for predicting five specific time series sources. In environmental parameter estimation, Lu [41] proposed the use of SVR to estimate forest air quality parameters. However, SVR prediction technique has not been widely used in network related estimation problems. Reference [42] however uses SVM for prediction of round trip latency to random network destinations.

Liang and Sun proposed an adaptive method for modifying the kernel function during the training of an SVR. Using a Gaussian RBF kernel, the authors proposed to modify the kernel function based on the method of information geometry. Using a conformal mapping, a new method was introduced which improves the precision of forecasting. An optimal partition algorithm (OPA) was used to modify the kernel, making the kernel data dependent. Results using S&P 500 time series data as well as Shanghai Stock Exchange (CISSE) data were presented and compared to an unmodified SVR.

SVR algorithm periodically trains input examples that are normally measured by other tools. A large number of researches have focused on network resources and parameters measurements, such as network delay [43], capacity [44], and available bandwidth [45]. These measurements

can be used to apply various admission control techniques, schedule network resource, load balance, etc.

In [46], Katti uses equally spaced mode gaps in TCP flows packet inter-arrival time distributions to detect multiple bottleneck capacities in their relative order. This scheme does not inject any probing traffic into the network. In [47], Katabi determines bottleneck link based on the observation that aggregated arrival trace from flows that share a bottleneck has very different statistics from those that do not share a bottleneck.

References [48] and [49] extend packet pair techniques by using three back-to-back probing packets and packet trains to measure network path capacity. This method helps to filter distorted probing packets. Reference [50] is a multimedia player based on rate control by using packet pair to measure path queuing delay.

Our scheme extends SVR to network prediction based on network feature inputs collected from our novel network packet probing measurement technique.

4.3 Support Vector Regression

Support Vector Regression (SVR) is a special implementation of the support vector machine for prediction. The basic idea is to train a regression function $f(x)$ that maps input features to output $\hat{y}_i$ to ensure it has no more than ε error from the obtained targets y_i and as flat as possible. $f(x)$ is then used to predict the future y_{i+1}. SVR has two advantages when compared to other neural network regressors. First, SVR estimates the regression using a set of linear functions or linear functions in high-dimensional feature space by finding the map from low-dimensional input space to high-dimensional feature output. Second, SVR is based on the structural risk minimization principle, where the risk is measured using certain loss function. The features in

the implementation of SVR are quadratic programming and kernel functions. The parameters are calculated by solving the quadratic programming.

Consider the set of training data $T = \{(x_1, y_1), (x_2, y_2), \cdots, (x_l, y_l)\}$ where $x_i \in R^n$ denote the input space of the example. n is the dimension of the input space. l is the total number of training dataset. $y_i \in R$ is the corresponding target value for $i = 1, 2, \cdots, l$. Then the following linear function is used to make regression in the feature space.

$$f(x) = \langle w, \varphi(x) \rangle + b \qquad (4.1)$$

Where $w \in R^n$ is the weight vector, $b \in R$ is the bias. $\varphi(x)$ is the function that maps the input feature from nonlinear low dimensional feature space to the high dimensional feature space in order to perform a linear regression in the high dimensional feature space, as shown in Figure 4.1.

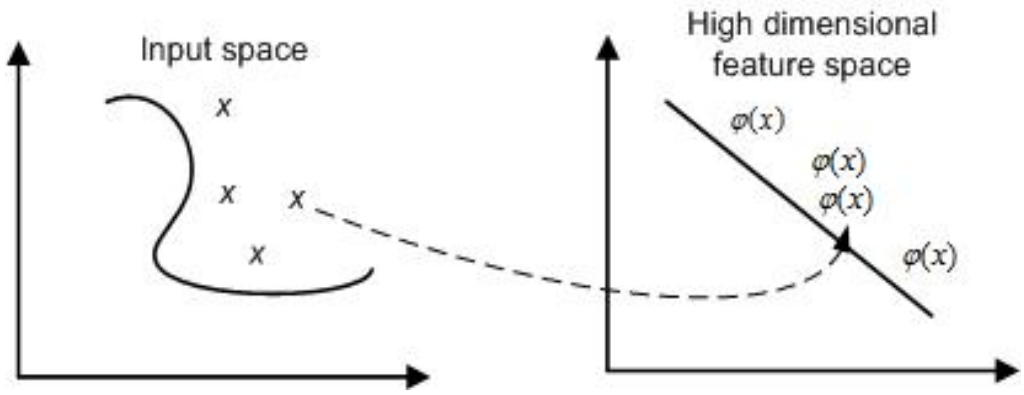


Figure 4.1 The map function

From equation (4.1), the objective is clearly to estimate the optimal values of weight *w* and bias *b* so that *f(x)* is close to real output *y* for all training data. It can be achieved by minimizing the regression risk function:

$$minmize\ R(f) = \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{l} L(y_i, f(x_i)) \qquad (4.2)$$

where

$$L\big(y_i, f(x_i)\big) = \begin{cases} |y_i - f(x_i)| - \varepsilon, & if\ \ |y_i - f(x_i)| \geq \varepsilon \\ 0, & otherwise \end{cases}$$

Minimize $\|w\|^2$ is a way to make the function as flat as possible.

$L\big(y_i, f(x_i)\big)$ is ε-insensitive loss function [51], which measures the absolute error between prediction and real value as shown in Figure 4.2. Besides ε-insensitive loss function, there are other loss functions available such as Laplacian loss function and Gaussian loss function. ε is the tube size, which refer to the precision of the function. All data inside this tube do not contribute to the regression function. Other points are used for learning and they are called support vectors. A large ε allows most data falls inside the tube. Therefore few support vectors can be used for learning. The result is the predicting precision decline. On the other side, a small ε allows too many data to be support vectors. The parameters updating based on those support vectors is time-consuming. *C* is the regularization constant that determines the tradeoff between the flatness and the accuracy. C determines the penalties to estimation error. Large C assigns high penalties to errors while small C assigns fewer penalties to errors.

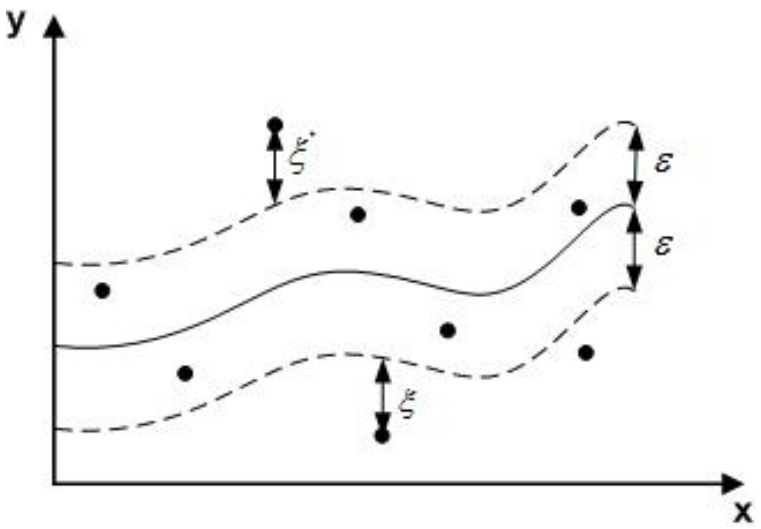


Figure 4.2 $\boldsymbol{\varepsilon}$-insensitive tube for Support Vector Regression

In order to estimate parameters *w* and *b*, errors have to be accepted to make the problem feasible. ξ_i, ξ_i^* are introduced to cope with infeasible constraints of the optimization problem. Equation (4.2) is transformed to (4.3)

$$minimize\ R(f) = \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{l}(\xi_i + \xi_i^*) \qquad (4.3)$$

Subject to

$$\begin{cases} y_i - \langle w, \varphi(x_i)\rangle - b \le \varepsilon + \xi_i \\ \langle w, \varphi(x_i)\rangle + b - y_i \le \varepsilon + \xi_i^* \\ \xi_i, \xi_i^* \ge 0 \end{cases}$$

Introducing Lagrange multipliers α_i, α_i^* transforms the problem to the dual optimization problem. They can be obtained by

$$maximize\ -\frac{1}{2}\sum_{i,j=1}^{l}(\alpha_i + \alpha_i^*)\left(\alpha_j - \alpha_j^*\right)\langle\varphi(x_i), \varphi\left(x_j\right)\rangle - \varepsilon\sum_{i=1}^{l}(\alpha_i + \alpha_i^*)$$

$$+\sum_{i=1}^{l} y_i(\alpha_i - \alpha_i^*) \qquad (4.4)$$

Subject to $\sum_{i=1}^{l}(\alpha_i - \alpha_i^*) = 0$ and $\alpha_i, \alpha_i^* \in [0, C]$.

So the weight *w* can be written as

$$w = \sum_{i=1}^{l}(\alpha_i - \alpha_i^*)\,\varphi(x_i)$$

Thus, the equation (4.1) can be transformed into

$$f(x) = \sum_{i=1}^{l}(\alpha_i - \alpha_i^*)\,\langle\varphi(x_i), \varphi(x)\rangle + b \qquad (4.5)$$

$\varphi(x_i)$ maps the input data from low dimensional space to high dimensional space, as we discussed before. Therefore, the more dimension $\varphi(x_i)$ has, the more dimension *w* becomes. However, dual optimization never uses the feature $\varphi(x_i)$ explicitly. $\langle\varphi(x_i), \varphi(x_j)\rangle = k(x_i, x_j)$ is defined as the kernel function, which enable the low dimensional space input data to be used in high dimensional space without explicitly calculating the map function $\varphi(x)$. Kernel function has to satisfy Mercer's condition [51]. For instance, Gaussian kernel is commonly used as the kernel function:

$$k(x_i, x_j) = e^{(-\frac{(x_i - x_j)^2}{2\sigma^2})}$$

We are going to use Gaussian kernel function as the default kernel function.

Equation (4.5) can be transformed into:

$$f(x) = \sum_{i=1}^{l}(\alpha_i - \alpha_i^*)k(x_i, x_j) + b \qquad (4.6)$$

The bias *b* can be computed by following the Karush-Kuhn-Tucker (KKT) condition [52]. The product of Lagrange multipliers and the constraints from (4.3) equals to zero at the solution points.

$$\alpha_i(\varepsilon + \xi_i - y_i + \langle w, \varphi(x_i)\rangle + b) = 0$$

$$\alpha_i^*(\varepsilon + \xi_i + y_i - \langle w, \varphi(x_i)\rangle - b) = 0$$

and

$$(C - \alpha_i)\xi_i = 0$$

$$(C - \alpha_i^*)\xi_i^* = 0$$

so

$$b = y_i - \langle w, \varphi(x_i)\rangle - \varepsilon$$

$$b = y_i - \langle w, \varphi(x_i)\rangle + \varepsilon$$

for $\alpha_i \in (0, C)$ and $\alpha_i^* \in (0, C)$

After computing the parameters we need, we bring them back to equation (4.6), the function *f(x)* is available to predict future output based on current network features.

4.3 The End-to-end Queuing Delay Measurement & Prediction Processes

4.3.1 End-to-end Queuing Delay Measurement

End-to-end queuing delay is the time packet stay in the queue along the path. In theory, queuing delay and packet loss are linear relationship if no other influence. Even though many other factors could also affect packet loss, packet queuing delay still partly reflects the network condition. Therefore in our scheme end-to-end queuing delay is one of the important features used to establish training set and estimate our evaluation. We apply a priority based back-to-back packet pair to measure the end-to-end queuing delay. This scheme does not require time synchronization and easy to implement. The only requirement is the network is Diffserv enabled network and support priority queue. Active measurement feature allows us to training dataset whenever it's needed.

The basic idea of queuing delay measurement is based on the different network delay suffered between the high priority packets and normal packets when they travel along the network. Basically, we inject probing packets with different priority into the network. Packets with different priorities will suffer different network delays along the path. We evaluate the queuing delay based on the dispersion between packets. For example, and referring to Figure 4.3, packet P_{high} has the highest priority and as a result it bypasses all the packets queued ahead, say in node 1, while packet P_{normal} with the normal priority will stay in the queue, so the dispersion Δ_{delay} between P_{high} and P_{normal} $-\Delta$ represents the queuing delay at node 1. Δ is the second packet P_{normal}'s transmission time divided by the bottleneck link bandwidth, $\Delta = \frac{L(P_{normal})}{C_{min}}$, where $L(P_{normal})$ is the length of P_{normal}, C_{min} is bottleneck link bandwidth. This will be repeated at all the nodes along the path. We define,

t_{high}: The arriving time at destination for the last bit of packet P_{high}

t_{normal}: The arriving time at destination for the last bit of packet P_{normal}

Network delay is a concatenation of four parts, processing delay (d_p), transmission delay (d_t), propagation delay (d_{pg}) and queuing delay (d_q). The first three parts are fixed delays and their summation d_f is

$$d_f = d_p + d_t + d_{pg}$$

Queuing delay is variable, which changes with traffic intensity. Hence the end-to-end path delay D is given by

$$D = d_f + d_q$$

Monitoring queuing delay is an effective way to avoid network congestion that is brought by existing or new traffic. We assume there are two kinds of priority packets (high, normal), high

priority is for probing packet while normal priority is for general traffic flows (traffic is treated according to first come first serve policy). High priority control packet has the highest priority over all other traffic in the network. Basically it goes directly to the head of the line in any node and is transmitted immediately. High priority control packet, with the aid of the normal priority control packet, will help in determining the delay along the path and the segment.

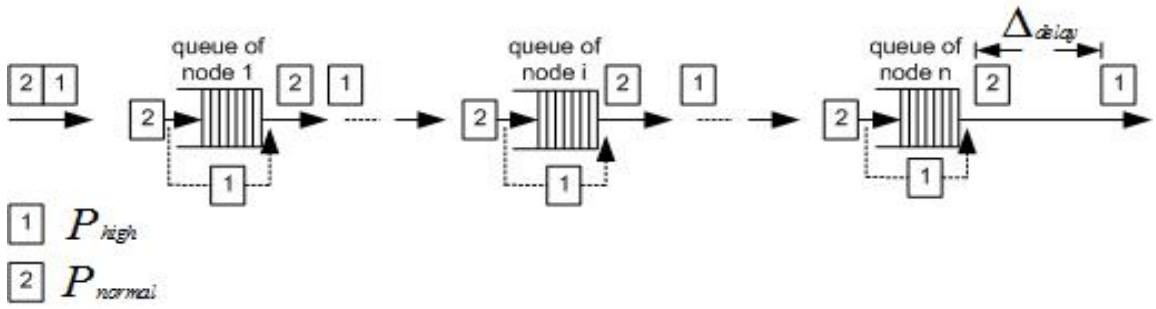


Figure 4.3 Priority based packet pair

Suppose the path from source to destination is $(L_1, L_2, \ldots, L_n)$. The packet pair $P_{normal}P_{high}$ is injected into link L_1. Both packets have the same length, that's, $L(P_{normal}) = L(P_{high})$, however they have different priority. P_{normal} is the packet with normal priority while P_{high} is the packet with high priority. Thus, the total path end-to-end delays D is given by the following equations,

$$d_f = d_{f,1} + d_{f,2} + \cdots + d_{f.n}$$

$$d_q = d_{q,1} + d_{q,2} + \cdots + d_{q,n}$$

$$D = d_f + d_q$$

Assume that the instant of time at which the last bit in the first packet P_{high} is sent is t and its arriving time at the destination is t_{high}, then

$$t_{high} = t + D \qquad (4.7)$$

P_{high} has highest priority over all other traffic in the network, hence P_{high} suffers no queuing delay. Equation 4.7 then becomes

$$t_{high} = t + d_f \qquad (4.8)$$

Similarly for the second packet P_{normal}. However the last bit of P_{normal} is sent after some dispersion time Δ_{intial} which equals to the P_{normal} packet transmission time $T_{normal,1}$ divided by link l_1 bandwidth, i.e. $\Delta_{initial} = T_{normal,1} = \frac{L(P_{normal})}{C_1}$. According to the earlier discussion in section 2.2, this dispersion $\Delta_{initial}$ will be extended to $\Delta_{C_{min}}$ when packet P_{normal} goes through bottleneck link whose bandwidth is C_{min}, $\Delta_{C_{min}} = \frac{L(P_{normal})}{C_{min}}$. Thus the arrival time instant of the last bit in the normal priority packet t_{normal} is given by

$$t_{normal} = \left(t + \Delta_{C_{min}}\right) + D \qquad (4.9)$$

From equations (4.8) and (4.9), dispersion Δ between P_{high} and P_{normal} is

$$\Delta = t_{normal} - t_{high} = t + \Delta_{C_{min}} + D - \left(t + d_f\right) = \Delta_{C_{min}} + d_q \qquad (4.10)$$

From equation (4.10), the end-to-end queuing delay d_q is given by $d_q = \Delta - \Delta_{C_{min}}$

Therefore, total end-to-end queuing delay experienced by packet P_{normal} is $t_{normal} - t_{high} - \Delta$. In other words, P_{high}, having the highest priority, bypasses all the queues along the path and its total end-to-end delay represents the accumulation of transmission times, propagation times and processing times along the path. On the other hand, P_{normal} with its normal priority waits in every queue along the path similar to every other data packet along the path.

4.3.2 Measurement and Prediction Process

4.3.2.1 End-to-end Delay Measurement

A. Receiver-side procedure

The receiver monitors the network in real time. When it starts to receive streaming from the source, it separates the probing packets from regular streaming. Based on the sequence number read from the packet header, the receiver distinguishes the probing packet pair and calculates the dispersion between the probing packets. The receiver also has to filter out those discontinuous probing packets in order to calculate the correct dispersion. The measurement results will be saved as data samples and send to SVR unit to predict the network condition in near future moment.

B. Sender-side Procedure

Sender-side procedure is simple. After the sender receive the request from the receiver, it starts the probing process by injecting probing packet pair with different priorities into the network along the path to the receiver. The priority level is decided by the streaming. In Diffserv enabled network, the traffic can be classified into 12 levels. The first probing packet of the packet pair has the highest priority while the second probing packet has the same priority level with the streaming traffic. After the packet priority is decided, the sender injects the packet pair in back-in-back into the network together with streaming traffic.

4.3.2.2 Prediction Process

The prediction process can be divided into two phases. Before the algorithm can be used to predict network condition, the training process starts the initialization of the predictor. A few data samples are chosen to generate SVR coefficients. And these coefficients are used to train the

remaining part of the training segment.

The second phase is to update the algorithm parameters. When the application is receiving the flow, the algorithm continues to collect the flow parameters measurement information. If the incoming streaming data set example is inside the ε-tube, that is to say it is not a support vector, the example doesn't influence the predictor parameters, and as a result no updates are made to the regression model, and this incoming data set is ignored. When the data example lies outside the ε-tube, the predictor parameters must be adjusted to update the regression model in order to reduce the future prediction error. This process continuously repeated.

4.3.3 Regression Parameters

In section 4.2, C is the regularization constant that determines the tradeoff between the flatness and the accuracy. When C is a constant, all training data has the same contribution to the accuracy of the predictor. However, in real time network environment, the most recent training data obviously are more important than the distant training data since recent data closely reflect the current network condition. As mentioned earlier, large C means the error between prediction and real value measured by the loss function is more important. Therefore, it is reasonable to give C a larger value for the recent training data than the distant training data. To achieve this goal, we configure C as a function of the training data location inside the training data set.

$$C_i = e^{\frac{i}{l}-1} \times C$$

i is the location of training data inside the data set ($i = l$ is the location of most recent incoming training data. $i = 1$ is the most distant training data). Therefore, for new incoming training data, the value of C_i will increase.

4.4 Experimental Results

In this section, we evaluate the performance of the proposed measurement and SVR mixed predictor using network simulation OPNET and compare it with that using historical mean predictor.

4.4.1 Effectiveness of Queuing Delay Measurement

We use the network topology as shown in figure 4.4. It is similar to the network topology in figure 2.7. In this network topology, CR_1, CR_2, and CR_3 are three core routers. ER_1, ER_2, ER_3, ER_4, ER_5, and ER_6 are edge routers. Source generates multicasting traffic for group members. Receiver_1 and Receiver_2 are two existing group members who receive multicast group traffic from the source. Node_1, Node_2, Node_3, and Node_4 introduce four different cross traffic into the network with destinations and sources are chosen as shown in Figure 4.4. The bottleneck link is chosen to be within the lower path segment. The small probing packet size is 40 bytes while the large probing packet size is 1500 bytes. The simulation time is always 30 minutes, and probing train is repeatedly sent 1000 times to make sure we get large enough samples to analyze.

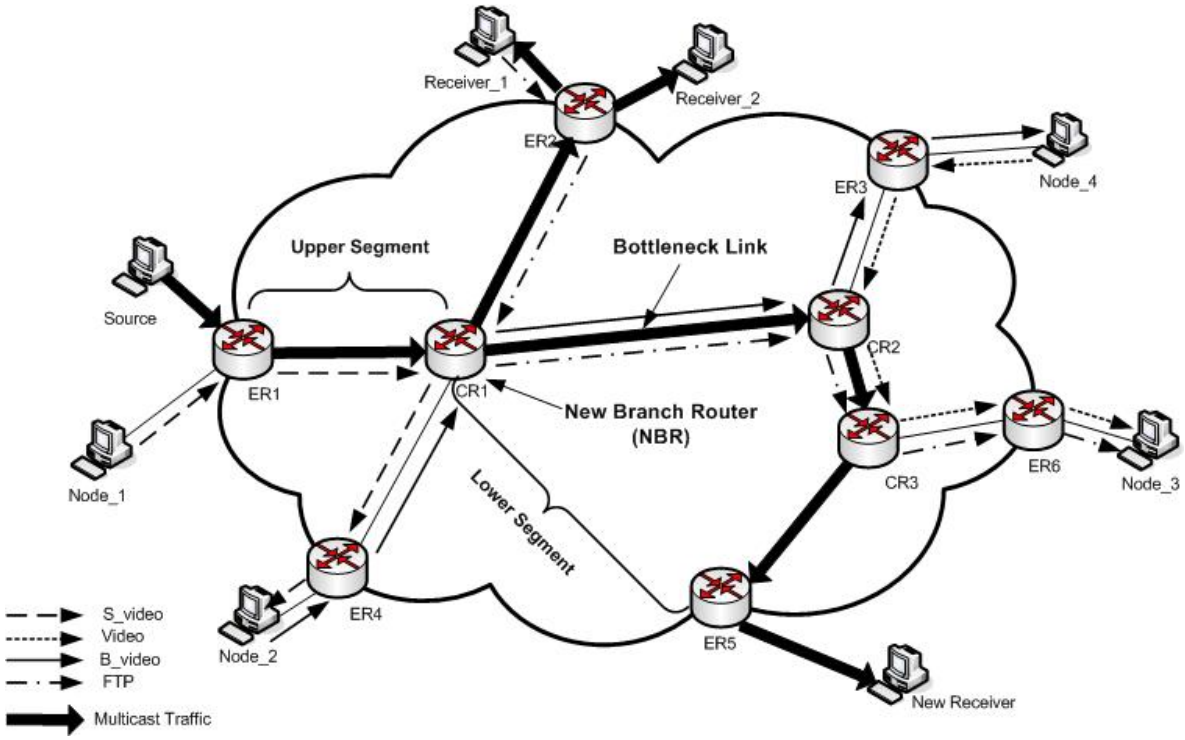


Figure 4.4 The simulated multicasting network topology

In this section, we probe and measure end-to-end queuing delay for each traffic path to show that our dispersion measurement exactly reflect path bandwidth usage condition. The process of probing and measurements is conducted repeatedly.

First we consider the case that cross traffic rates are below bottleneck link bandwidth. The parameters of the traffic flows are listed in Table 4.1. The link that connects CR1 and CR2 is the bottleneck link. Bandwidth of link CR1-CR2 is 55Kbps while bandwidths of other links are 1,544Kbps.

Table 4.1 Traffic parameters

Flow Name	Packet Size (bytes)	Transmission Rate (bits/sec)
B_video	2000	41600
FTP	2000	560K
S_video	2500 bytes	460K
Video	80 bytes	518

Figure 4.5 shows that the average end-to-end queuing delays of the four traffic paths are all in low level. Those results are in keep with the network condition we set in TABLE II.

When we add a new receiver for the group, multicast flow is introduced to the lower path segment where the bottleneck link is located. Overloading of the bottleneck link will result in delay increase sharply of those flows that go through the lower path segment, such as B_Video and FTP traffic. Other flows, such as S_video and Video that don't flow through lower path segment, will maintain their own traffic rate. Upper path segment network resource is free from the influence of new receiver. Figure 4.6 shows the average end-to-end queuing delays we measured using our probing dispersion method when new multicast receiver worse the network.

Our delay measurements exactly reflect queuing delay distribution and also reflect the congestion level in the network.

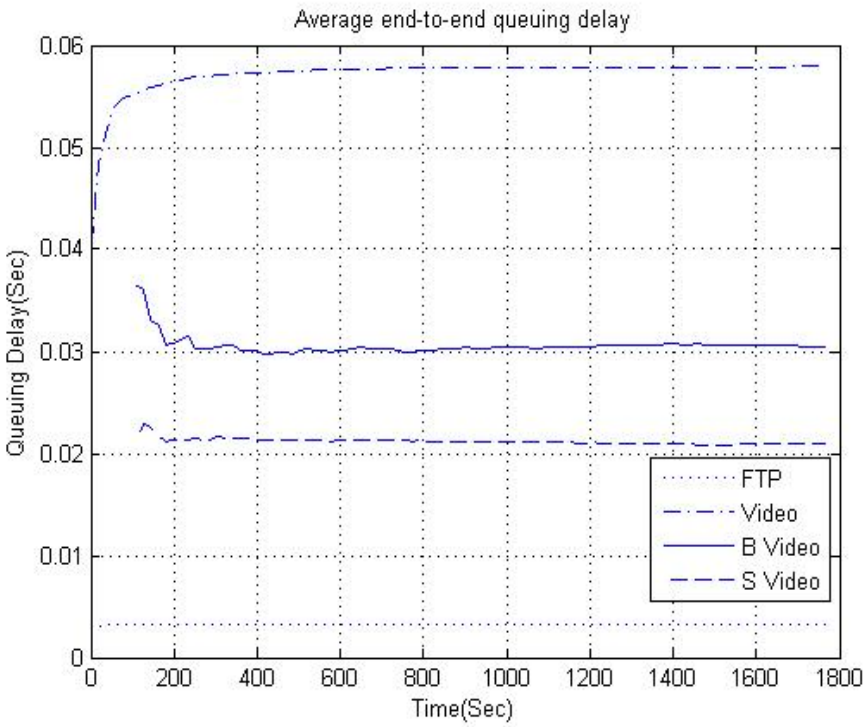


Figure 4.5 Ave end-to-end queuing delay-vs-simulation time (no new receiver)

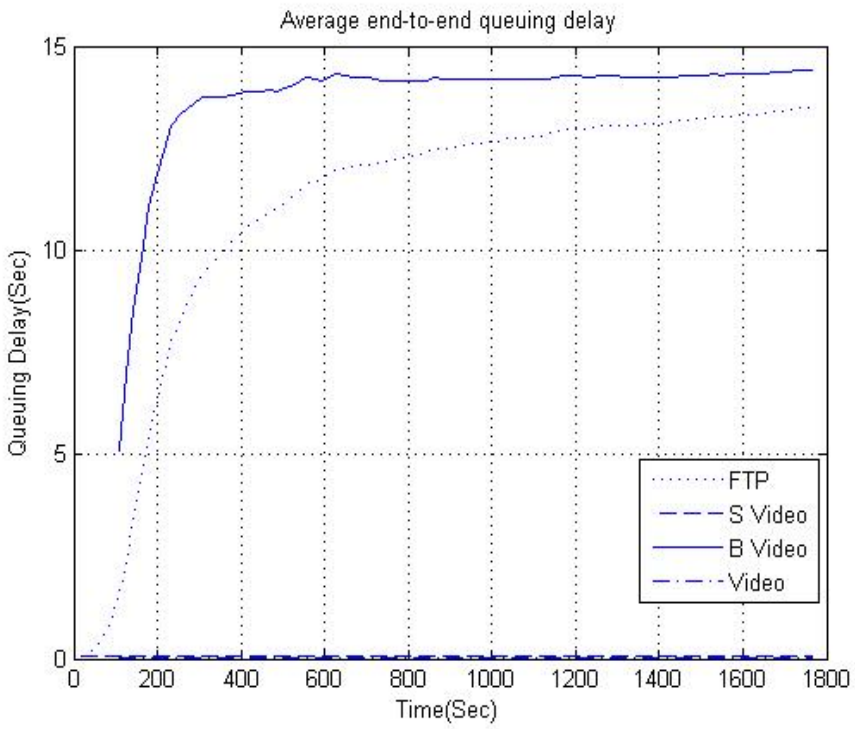


Figure 4.6 Ave end-to-end queuing delay-vs-simulation time (new receiver)

4.4.2 Experimental Environment for Predictor

We set up the network topology that connects a multimedia source and a receiver, as shown in Figure 4.7. Intermediate routers have been configured to support DSCP based QoS in order to make sure queuing delay can be measured by the receiver. The probing packets that are used for measuring queuing delay are injected into the network by the source side host and follow the same path as the multimedia stream toward the receiver. At the same time, the receiver joins the multicast group to receive the video from source side. For conveniently analyzing the received video stream, we choose a constant rate video source.

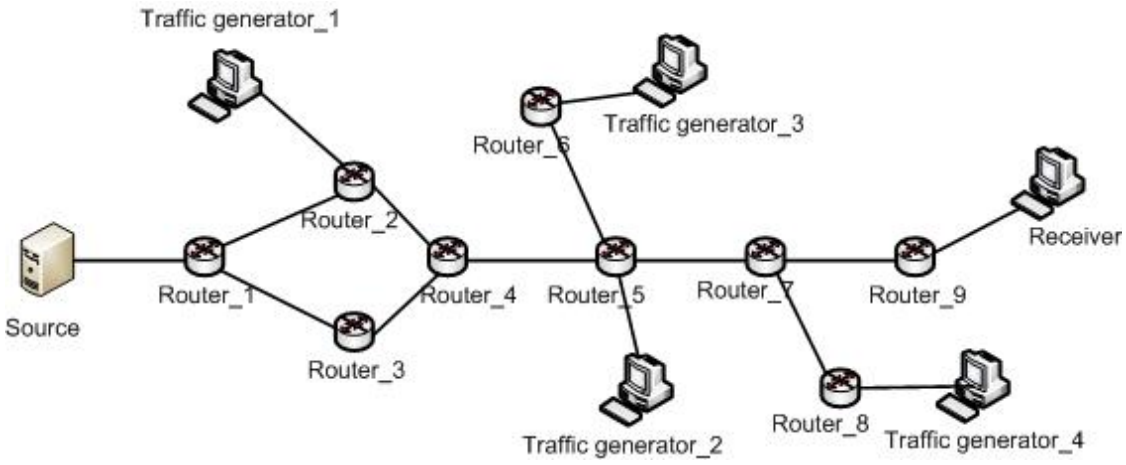


Figure 4.7 Simulation network

We define the packet loss rate as the ratio between the number of the lost packets and the number of transported packets during each interval.

$$lossrate(k) = \frac{P_{Loss}(k)}{P_{Loss}(k) + P_{received}(k)}$$

$P_{Loss}(k)$ is the number of packet loss. $P_{received}(k)$ is the number of received packets.

In order to analyze and predict the packet loss rate under different network environment, we generate background traffic by using four traffic generators to the network as shown in Figure 4. They can simulate three kind of traffic (video, audio and large file) with average load ranging from 40% to 100% of the bottleneck link (the link that connect Router_4 and Router_5). We adjust the background traffic sending rate to control the available network resource so as to generate different packet loss rate to aid in verifying the accuracy of the proposed predictor and compare it with historical mean predictor.

4.4.2.1 Stationary background traffic

The background traffic that is generated in stationary sending rate ensures the packet loss that the receiver suffers cannot fluctuate sharply. As shown in Figure 4.7, traffic generators inject 4 different flows in constant rate to occupy network available resource in order to force multicast group video drop packets in a stationary rate. Figure 4.8 and 4.9 show the packet loss rate prediction when background traffic rate is low and high respectively. As shown in Figure 5, the prediction curves of historical mean predictor and our SVR based predictor are all very close to the actual packet loss rate curve. Both methods can accurately predict the near future packet loss rate.

However, regardless of the packet loss rate, historical mean predictor needs to calculate the mean value from the dataset for every interval while SVR based predictor use an adaptive mechanism. When the packet loss rate changes slowly, the SVR based predictor doesn't need to update the training parameters since most data examples fall inside the ε-insensitive tube and accordingly are ignored because they do not contribute to the regression function. The training parameters are only updated when packet loss rate change exceeds the reasonable scope. The proposed predictor has higher computation efficiency than the historical mean predictor.

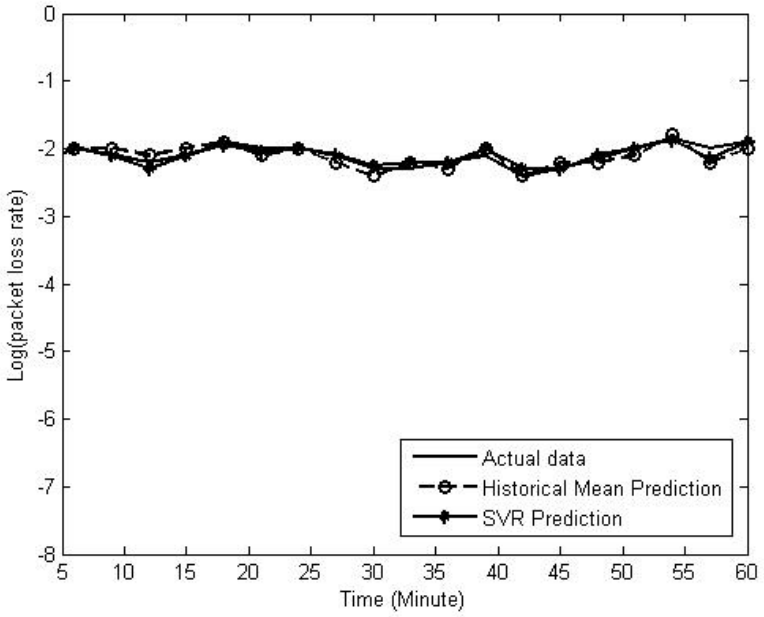


Figure 4.8 Packet loss rate is high, but keeps in a stationary level

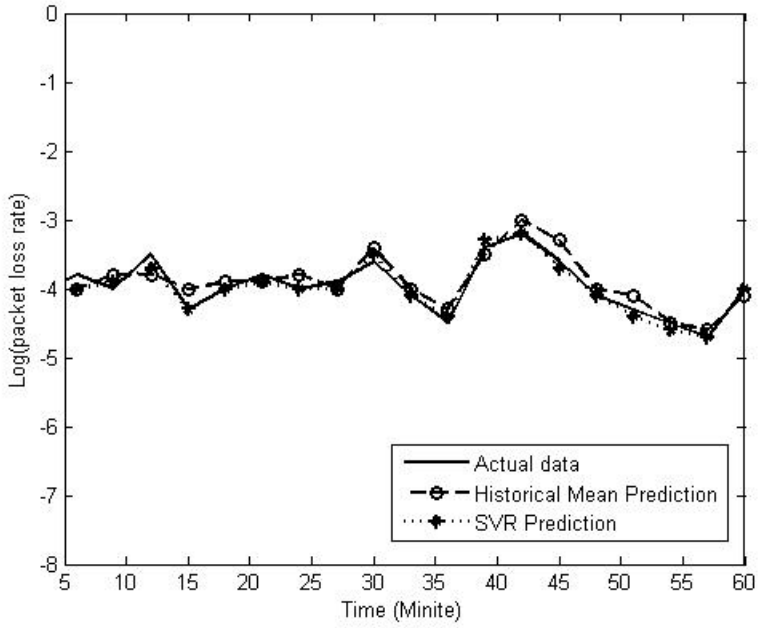


Figure 4.9 Packet loss rate is low and stable

4.4.2.2 Fluctuate Background Traffic

In this section, we check whether the predictor can work accurately when the packet loss rate change frequently and sharply. In Figure 4.7, four workstations generate four flows that start at different times and have different sending rates..

Figure 4.10 and 4.11 show the prediction curves of two methods for dynamic packet loss rate. As expected, under the fast changing background network traffic scenario, the historical mean predictor suffers a large prediction error and is slow in reflecting the changes in packet loss rate. On the other hand, the proposed SVR predictor tightly follows up the changes in packet loss rate within a reasonable error scope. In Figure 4.11, where the packet loss rate changes sharply, the historical mean predictor makes large errors in prediction, while the proposed predictor effectively tracks the changes and accurately predicts with small error margin,. Frequent change of packet loss rate allows more data examples fall outside the ε-insensitive tube. They are chooses as support vector to update the training parameters. The updating of the parameters make sure regression function still can reflect the correct relationship between input and output. In short, when compared with historical mean predictor, the proposed predictor is suitable to various network environments. It is robust to stationary or non stationary background traffic effect.

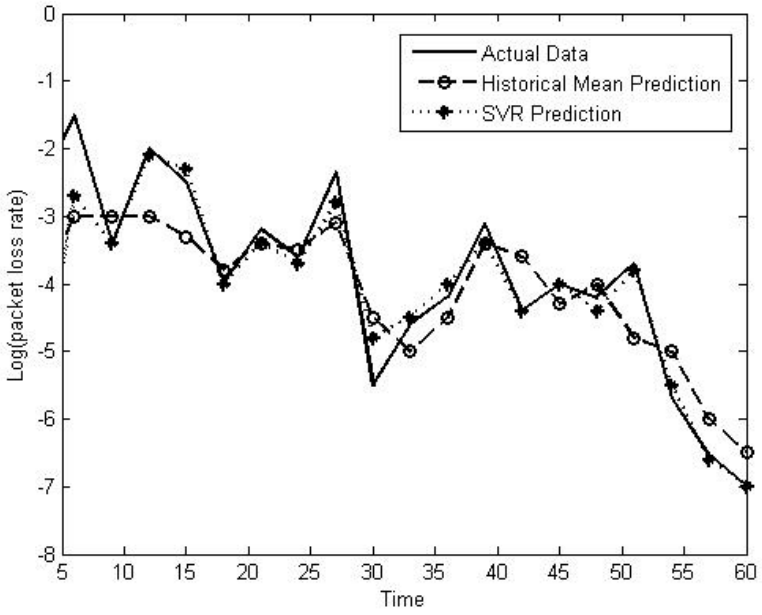


Figure 4.10 Packet loss rate is non-stationary

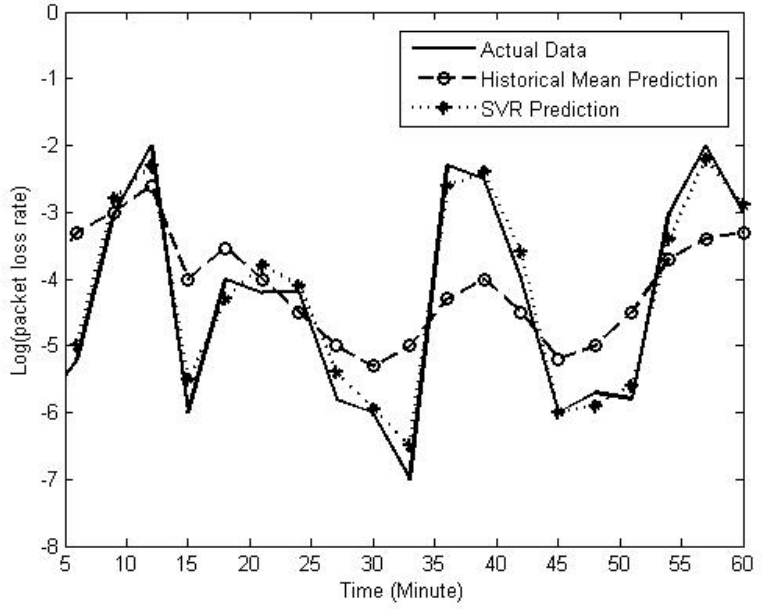


Figure 4.11 Packet loss rate change sharply

4.5 Conclusion

Support vector regression has been proved their success in many fields such as financial industry and environmental industry. However, little work has been done for network condition prediction. We combine SVR algorithm with active probing measurement to predict packet loss rate in the near future. It is very useful to provide QoE for real-time multimedia by predicting and pre-adjusting multimedia sending rate. Further analysis is needed to improve the computing efficiency for real-time prediction.

Chapter 5
Implementation of Multicast Admission Control and Testbed

5.1 Introduction

As we discussed in chapter 2 – chapter 4, multicast is the main steaming transmitting method. However, currently, admission control prototypes have focused on admission control for *unicast* traffic with little or no consideration for multicast traffic. Therefore, we address the challenge of systematically admitting/denying/preempting multicast flows to promote QoS in the context of a mixed unicast/multicast traffic environment. The primary goal of this chapter is to design a multicast admission control system, which works with existing multicast routing protocols. This solution will function in inter-domain routing environments operating with different unicast routing (e.g., OSPF and BGP) and multicast routing (e.g., PIM-SM and PIM-DM) protocols. In addition, this system will be designed to work with existing QoS mechanisms such as Differentiated Services (DiffServ).

5.2 PIM-SM Protocol Description

PIM-SM is a type of sparse mode multicast protocol. It uses the “pull mode” for multicast forwarding, and is suitable for large- and medium-sized networks with sparsely and widely distributed multicast group members.

The basic implementation of PIM-SM is as follows:

- PIM-SM assumes that no hosts need to receive multicast data. In the PIM-SM mode, routers must specifically request a particular multicast stream before the data is forwarded to them. The core task for PIM-SM to implement multicast forwarding is to build and maintain rendezvous point trees (RPTs). An RPT is rooted at a router in the PIM domain as the common node, or rendezvous point (RP), through which the multicast data travels along the RPT and reaches the receivers.
- When a receiver is interested in the multicast data addressed to a specific multicast group, the router connected to this receiver sends a join message to the RP corresponding to that multicast group. The path along which the message goes hop by hop to the RP forms a branch of the RPT.
- When a multicast source sends a multicast packet to a multicast group, the router directly connected with the multicast source first registers the multicast source with the RP by sending a register message to the RP by unicast. The arrival of this message at the RP triggers the establishment of an SPT. Then, the multicast source sends subsequent multicast packets along the SPT to the RP. Upon reaching the RP, the multicast packet is duplicated and delivered to the receivers along the RPT.

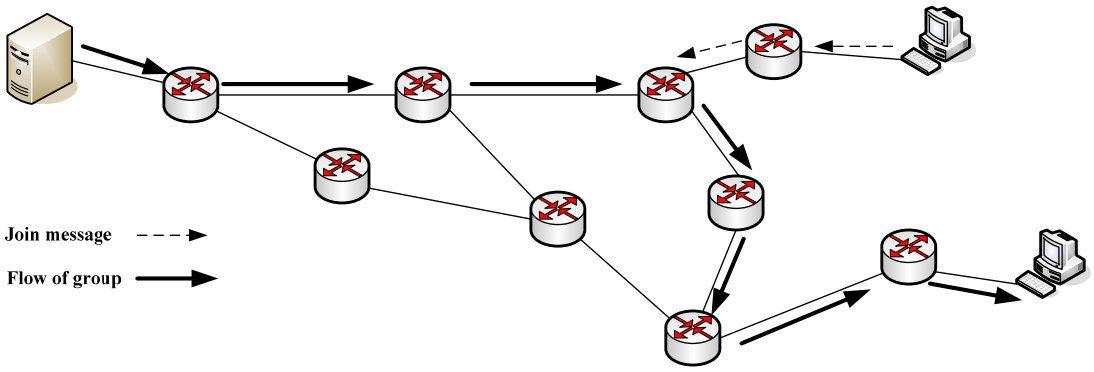


Figure 5.1 Before the new receiver join the group

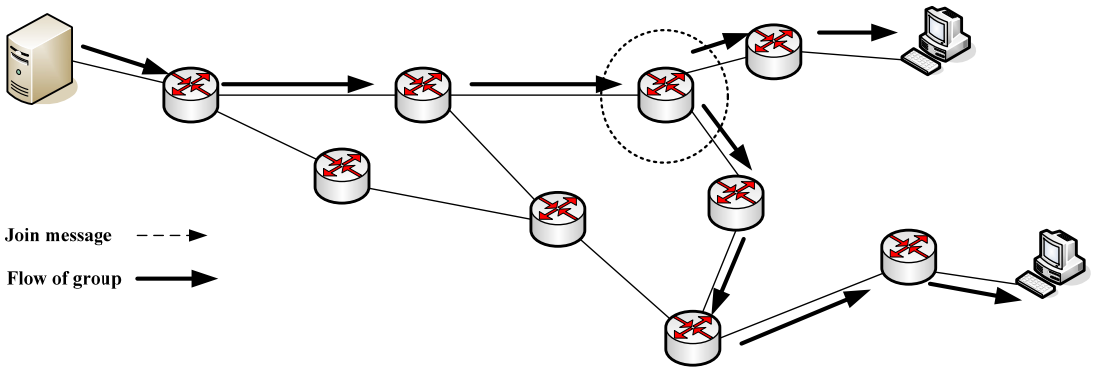


Figure 5.2 After the new receiver join the group

The main working mechanism of PIM-SM is summarized as follows:

- Neighbor discovery
- RPT building
- Multicast source registration

5.2.1 Neighbor Discovery

In a PIM domain, a PIM router discovers PIM neighbors, maintains PIM neighboring relationships with other routers, and builds and maintains SPTs by periodically multicasting hello messages to all other PIM routers (224.0.0.13).

5.2.2 RPT Establishment

When a receiver joins a multicast group G, it uses an IGMP message to inform the directly connected DR.

Upon getting the receiver information, the DR sends a join message, which is hop by hop forwarded to the RP corresponding to the multicast group.

The routers along the path from the DR to the RP form an RPT branch. Each router on this branch generates a (*, G) entry in its forwarding table. The * means any multicast source. The RP is the root, while the DRs are the leaves, of the RPT.

The multicast data addressed to the multicast group G flows through the RP, reaches the corresponding DR along the established RPT, and finally is delivered to the receiver.

When a receiver is no longer interested in the multicast data addressed to a multicast group G, the directly connected DR sends a prune message, which goes hop by hop along the RPT to the RP. Upon receiving the prune message, the upstream node deletes its link with this downstream node from the outgoing interface list and checks whether it itself has receivers for that multicast group. If not, the router continues to forward the prune message to its upstream router.

5.2.3 Multicast Source Registration

The purpose of multicast source registration is to inform the RP about the existence of the multicast source.

When the multicast source S sends the first multicast packet to a multicast group G, the DR directly connected with the multicast source, upon receiving the multicast packet, encapsulates the packet in a PIM register message, and sends the message to the corresponding RP by unicast. When the RP receives the register message, it extracts the multicast packet from the register message and forwards the multicast packet down the RPT, and sends an (S, G) join message hop by hop toward the multicast source. Thus, the routers along the path from the RP to the multicast source constitute an SPT branch. Each router on this branch generates an (S, G) entry in its forwarding table. The multicast source is the root, while the RP is the leaf, of the SPT.

The subsequent multicast data from the multicast source travels along the established SPT to the RP, and then the RP forwards the data along the RPT to the receivers. When the multicast traffic arrives at the RP along the SPT, the RP sends a register-stop message to the source-side DR by unicast to stop the source registration process.

5.3 Multicast Multi-Streaming

An important aspect of ensuring the QoS of receivers within a multicast session is to ensure that receivers can get the media rate and quality that is compatible with their needs and capabilities, as well as the capacities of the path leading to it. This is referred to as the fairness of the multicast scheme. The fairness problem results from the fact that multicast communication trades economy of the bandwidth with granularity of control. Distributing video using individual feedback-controlled point-to-point streams results in high bandwidth utilization but the

granularity of control is high as communication parameters can be negotiated with each receiver. In contrast, using a single multicast stream has good bandwidth economy but with very low granularity of control.

Multi-stream Multicast is a multicast policy that a limited number of streams of the original content are sent over different multicast groups. In the context of video multicasting, different multicast groups are used to deliver different versions of the original video, e.g., low-, medium-, and high-quality versions. This scheme is also referred to as Stream Replication. The different streams are made possible by encoding the original video using encoders that are set to provide streams with different rate. In the case of stored media, a different stream can also be created by transcoding an existing stream. Thus in the proposed scheme, a limited number of multicast groups are created, e.g., three groups, that carry the same video but each is targeted at receivers with different capabilities and/or needs. In order to accommodate the need of the receivers as much as possible, the stream sources can adjusts the rate of the streams, within prescribed non-overlapping ranges, based on feedback from the receivers. In addition, receivers are allowed to move between the different groups based on their needs. Each multicast group is assigned a different multicast group address. The availability of multicast streams/groups can be known to recipients by either pre-planning the groups/streams or by dynamically announcing the stream.
Due to non-uniform characteristics of multicast receivers, we deploy *receiver-based admission control*. In an attempt to implement admission control without scalability problems, the receivers decide which group to Join, as well as switch between different groups based on the quality of the received media as well as the needs of the receivers. The source of the stream periodically multicasts “feedback request packets” to the receivers to collect the state of their video reception, and accordingly it adapts the stream rate. If the source of the stream cannot reduce the stream

rate, the receiver can decide to switch to leave the current group and join another group that delivers the same video with lower rate, if available. On the other side, it is possible that the quality of the received stream goes below the expected quality. Again in this case, the receiver first notifies the source, via responses to feedback requests, hoping that the source can increase the media rate. If the source cannot increase its current rate, due to others receivers in the group, the receiver can decide to switch to join another group with higher rate, if available.

Based on feedback about the network condition collected from the receivers, the group source can adapt the group stream rate through feedback from the receivers regarding the quality of the received stream. This feedback can be retrieved by allowing the source to poll the receivers within the group. The source polling and the receiver's feedback should be done in resource-friendly manner. One possible implementation will merge the feedbacks from similar receivers on a red network which are receiving the same stream before sending it to the source.

For example, in the multi-streaming multicast network, the receivers respond to feedback requests from the source, which are an indicative of the quality of the received video. This enables the source to adapt the stream rate to match the quality of the receivers within the session. The receivers combine the video quality metrics as well as statistics collected by the proposed system about the status of the network in terms of available bandwidth, delay, and lossed to respond to the source feedback requests. One possible response scheme can be in the form of three congestion levels, Unloaded, Loaded, and Congested. Based on the received feedback from all the receivers within the group, the source can decide to increase or decrease the stream rate within its prescribed range.

To reduce the disruption of service, when a receiver decides to switch to a new stream, we propose that the receiver joins the new group before it leaves the original group. As soon as it

has been admitted successfully to the new group, and it switches to the new stream, the receiver can leave the original group. This will also help the receiver to decide when to switch between the original and new stream. Most existing adaptation protocols ignore the detailed operation for stream or layer switching. However, this is nontrivial given that a video stream generally has a complex syntax, and its content is highly dependent.

5.4 The Architecture of the Proposed System

The following section presents a receiver-based admission control architecture to address the performance objectives identified here for the FBN environment. The proposed admission control system will address these objectives in an efficient, scalable manner and function reliably in the context of MLS, irrespective of the network routing implementation.

At the high level, the architecture of the proposed admission control system performs the following primary functionalities:

1. Estimate (or predict) network performance in terms of throughput and delay
2. Evaluate receiver-based multicast group membership requests based on multicast session QoS needs, session priority and network performance
3. Admit multicast group memberships and associate multicast sessions whose QoS needs can and should be satisfied and assign an appropriate DiffServ Code Point (DSCP) or Type of Service (ToS) setting for the multicast session packets
4. Deny multicast group-memberships/sessions whose QoS needs cannot be satisfied.

5. Creating multiple streams with different qualities (i.e., encoding rates) and assigning each stream to separate multicast groups. Receivers are allowed to join and switch between the multicast groups, based on their available network resources as well as on their need.

The key features of this architecture are as follows:

- As a receiver-based architecture applying IGMP messages to trigger the pruning/grafting of a multicast tree, it is naturally compatible with tree-based multicast routing, in general, and, in particular, with the widely available Protocol Independent Multicast (PIM) routing protocols.
- It is an edge-based admission control architecture and, as such, its functionality is largely independent of the type of the network and independent of whether it comprises a single routing/radio domain (or autonomous system) or multiple radio/routing domains.

5.4.1 Implementation Details

The core functionality of the admission control system is, of course, to receive multicast group join requests and respond with a decision (e.g., ACCEPT, DENY, ACCEPT WITH PREEMPTION) as to whether the request has been granted. As summarized in Chapter 2, this functionality is performed by the Admission Decision Logic with coordinated support from the remaining McastAC modules.

Prior to presenting the admission decision logic design, however, there are a number of terms unique to multicast admission control that merit description. Table 5.1 below represents a glossary of some of these terms that are used in the of specification admission control procedures, herein.

Table 5.1 Glossary of admission control terms

Term	**Definition**
Node	Routers inside the network and end-hosts
Candidate group member	A host that is eligible for consideration to join a particular multicast group
Accepted group member	host that has been approved by the admission control system to join a particular multicast group
Denied group requestor	A host whose request to join a particular multicast group has been denied
Threshold Performance	The network performance levels that (ideally) are satisfied prior to the admission control system accepting a new group member (e.g., > throughput-threshold, < delay-threshold, < loss-ratio-threshold, < congestion-threshold)
Current group member nodes	at least one accepted group member for a particular multicast group

A variety of events can potentially result in one or more admission decision procedures being invoked. Table 5.2 below identifies some of the key events/scenarios that can trigger admission decision processing.

Table 5.2 Admission Decision Logic Triggers

Trigger	**Description**	**Triggered Action**
Multicast group join request	A request from a local (same platform-enclave) candidate group member to join a particular multicast group	Accept/Deny/Preempt decision logic for the join request
Multicast group member departure	Notification (either explicit or implicit) from a local accepted group member it is leaving a particular multicast group	Check for whether recently denied join requests should be allowed or promoted
Stream rate increase	A increase in the multicast stream rate as requested by the receiver	Check whether increase in stream rate should be accommodated
Stream rate decrease	A decrease in the multicast stream rate is specified by the receiver	Check whether the new stream rate is so small it should be preempted
Switch up the group	A group member switch to a group that provides higher quality streaming	Check whether the new group permit the switching behavior
Switch down the group	A group member switch to a group that provides lower quality streaming	Check whether the new group permit the switching behavior

The Figure 5.3 shows the application runs on the source side.

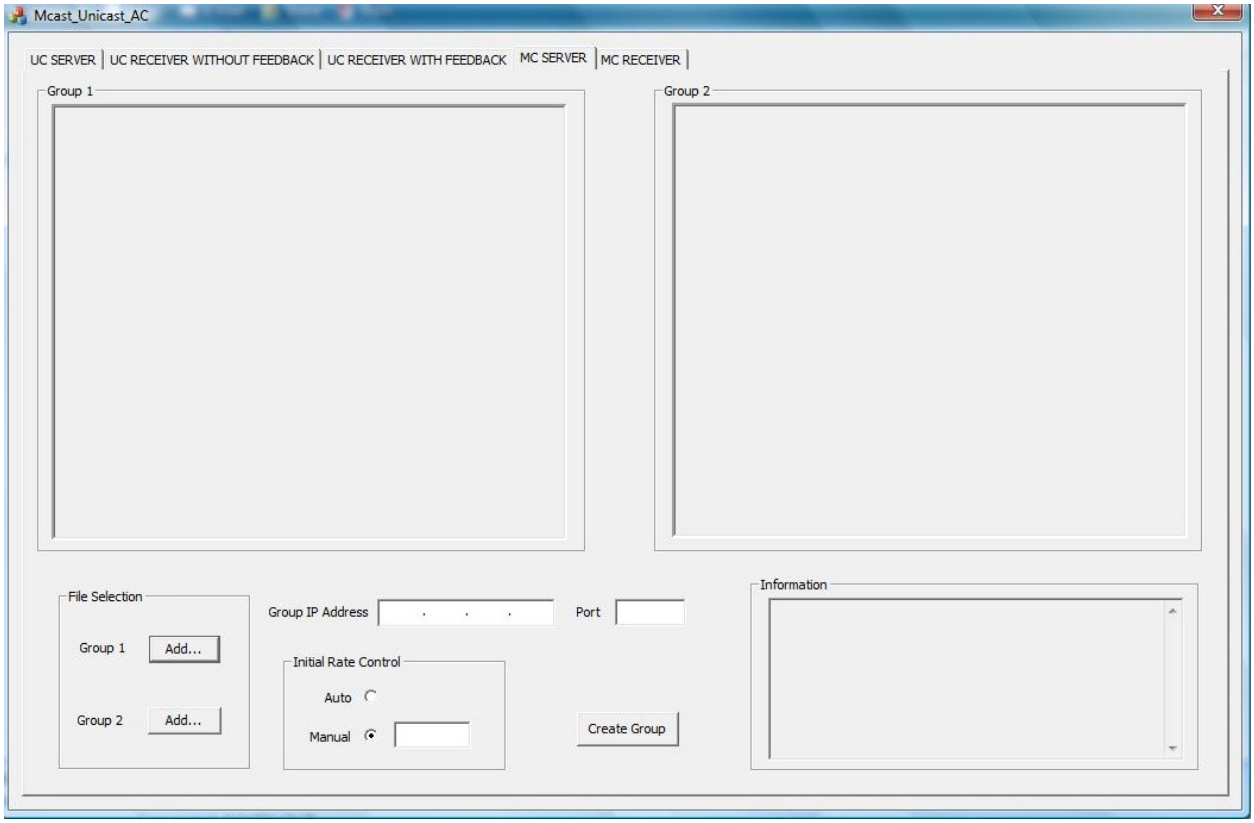


Figure 5.3 Source side application

The source side application can create two groups that provide different quality streaming. The streaming sending rate can be configured according to requirements. After we decide the sending rate for each group and configure the multicast address. For multicast network, multicast address is the only identification. The application create two groups that identified by the multicast address. Before any receiver joins the group, the source monitors the network and registers itself to the rendezvous point inside the multicast network for potential receiver.

When the source receives the join request from the receiver, the probing module is triggered to probing the network path by injecting the probing packet train. At the same time the source monitors the network to wait join message or other join request from other receiver. Once the

receiver joins the group successfully, the source will forward the streaming to the receiver along the multicast tree.

The source actually keeps monitoring the network during the display. When the application receives the request from the receiver to increase or decrease the streaming rate, it will check its condition to trigger the rate control module or send back message to the receiver for switching group.

The Figure 5.4 shows the application runs on the receiver side.

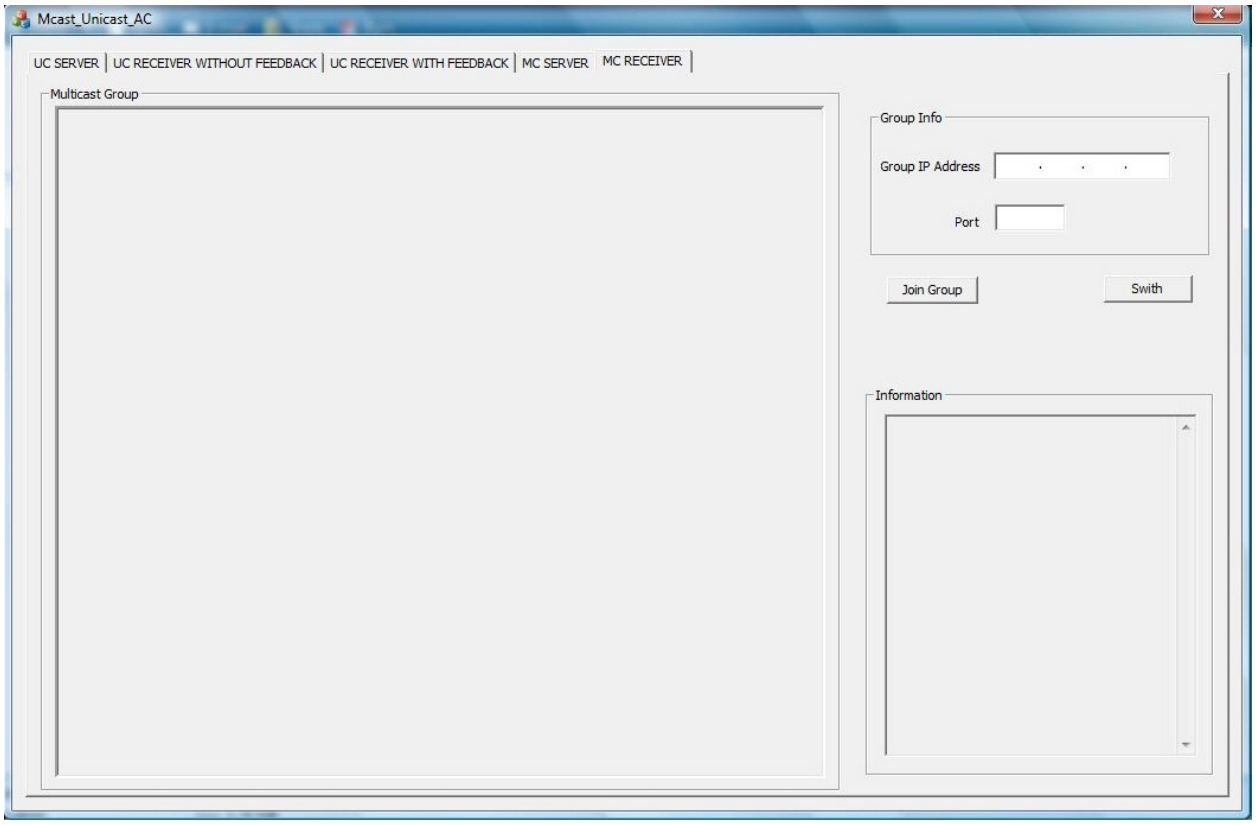


Figure 5.4 Receiver side application

The receiver application runs at the end user side. When the end user tends to join the group, it has to configure the multicast address that is the identification of the interested group. For this

application, we just can configure the multicast address. Next step we are going modify the application to support domain name.

After the end user configure the multicast address, the application will trigger a join request message and send it to the source application before the real join message is generated. The join request is used to trigger the probing process in source side. At the same time, the receiver application starts to monitor the network for possible probing packet it received. The probing module run as admission control logic that is responsible to measure the network condition and decide to join the group or not. When the admission control logic decides to join the group, it will generate the real join message and inject into the network toward to the source.

The receiver application supports another group switching function. After the receiver joins the group, the receiver still monitors the network. When the network condition cannot support current streaming rate with good quality, the switching module will trigger the group switch process. Switching the group from high quality group to the low quality group allows the streaming can be displayed smoothly. Otherwise, the receiver will suffer severe packet loss, delay or jitter. Switching the group permit the user to keep better feeling.

5.4.2 Experiments Environment

The network we set up to run the admission control system is composed of two Cisco 3560 routers, four servers and OPNET system in the loop(SITL) model. The Table 5.3 lists the hardware configurations. The Table 5.4 lists the software we use to develop and run the application.

Table 5.3 The hardware configurations

Node	Computer	OS	CPU	Network Card
Server	Laptop	Windows XP/ Ubuntu 9.10	Intel Core2 Duo	10/100Ethernet
SITL	Desktop	Windows XP	Intel Celeron	10/100Ethernet
Receiver 1	Laptop	Windows Vista/ Ubuntu 9.10	Intel Core2 Duo	10/100Ethernet
Receiver 2	Laptop	Windows Vista/ Ubuntu 9.10	Intel Core2 Duo	10/100Ethernet

Table 5.4 The software configurations

Operation System	Windows XP, Windows Vista, Ubuntu 9.10
Emulator	OPNET 15.0
Development Tool	Visual Studio 2005
Media Player	VLC 1.05

We use nodes inside the OPNET network generate cross traffic to control the network condition.

SITL is a kind of emulator that connect real network with simulation topology together to create complicated network. SITL utilize network card to capture and parse the network traffic from outside network and injected into the OPNET network topology. Finally the traffic is sent out by the network card to the real network again. SITL give us the change to build and test large and complicated network environment for real application.

5.4.3 Experiment Results

As Figure 5.5 shown, the source creates two groups with different streaming rate. Group 1 has much better quality than group 2. Those two groups are all ready to provide streaming for receivers.

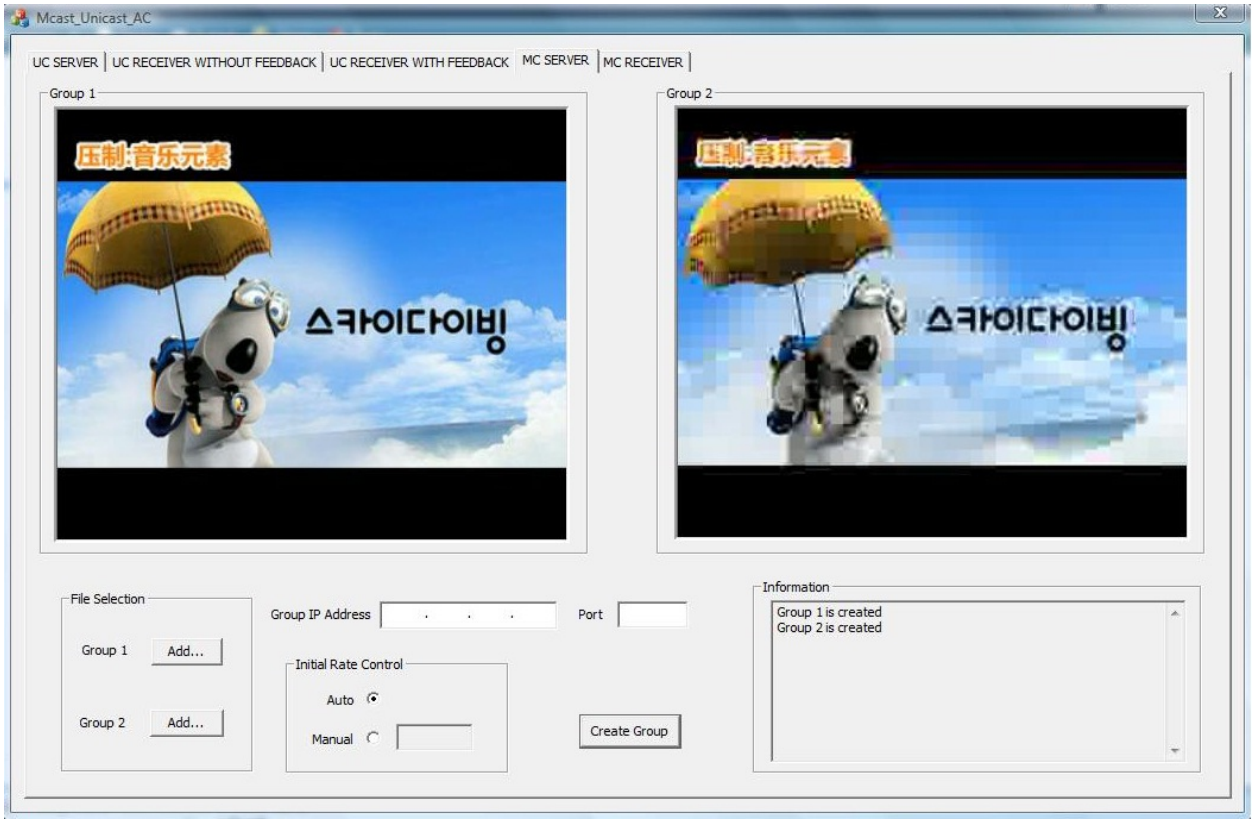


Figure 5.5 Source side application creates two groups

Figure 5.5 is the application run in one computer as the receiver. before we generate the cross traffic to congest the network, the admission control logic permit the receiver to join the group 1, which provide high quality streaming. The Figure 5.5 shows the quality is as good as the group1 streaming display in source side.

We generate cross traffic inside the OPNET network to congest the multicast tree branch, from the Figure 5.6, the receiver monitor this change of network condition, it switches to group 2 before it suffers severe packet loss. As Figure 5.6 shown, the receiver now join the group 2, the quality of streaming is not as good as the group 1. But the receiver suffers less packet loss or delay. The receiver can display the streaming smoothly.

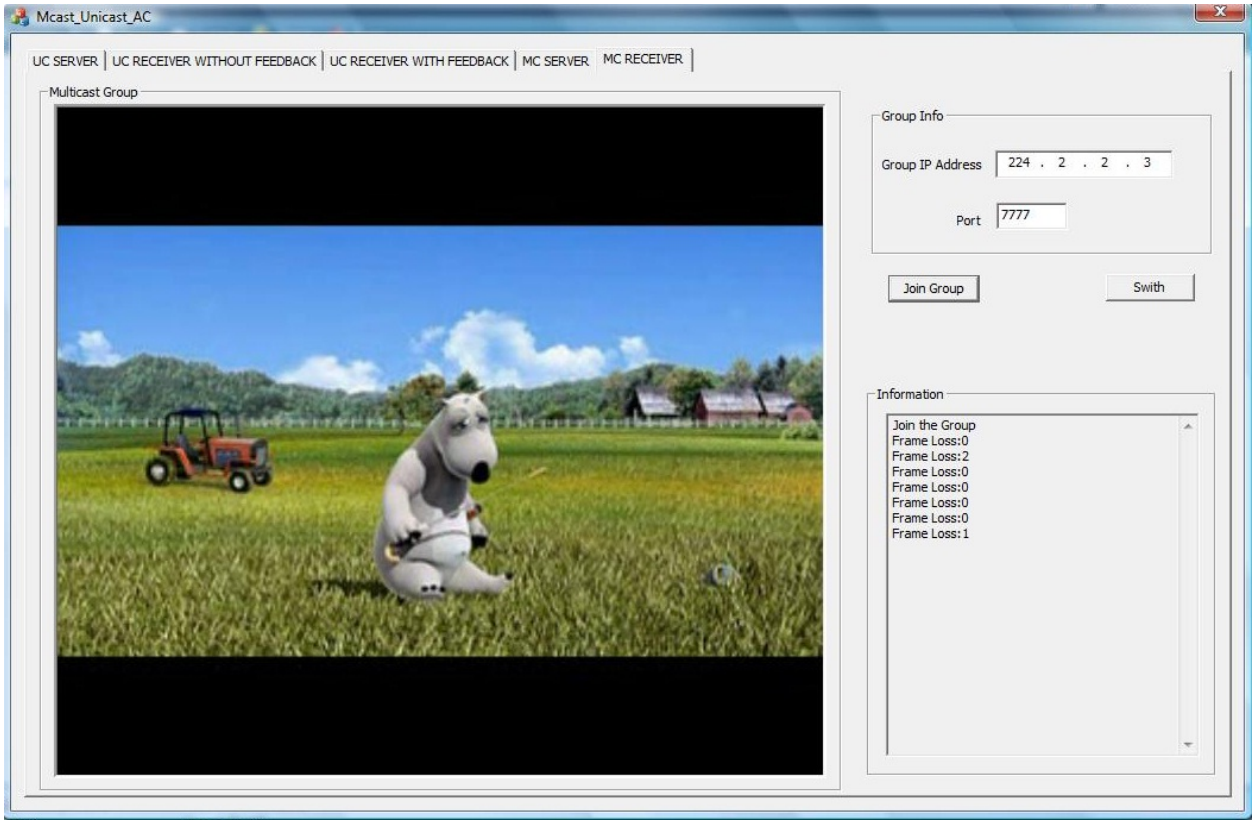


Figure 5.6 The receiver join the group 1

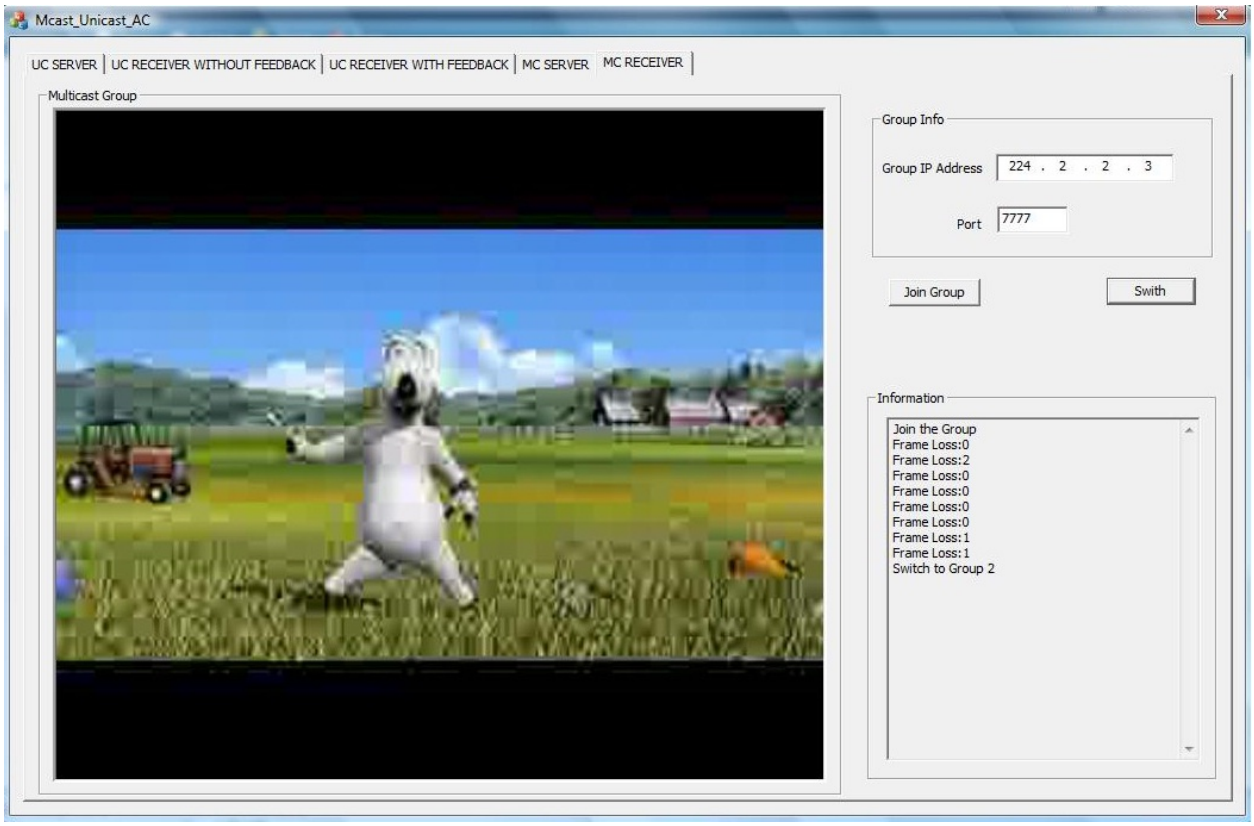


Figure 5.7 The receiver switched to group 2

Figure 5.8 compare the streaming quality before the network congestion and after the network congestion. The Figure 5.8 shows that network congestion doesn't bring severe influence to the streaming display. The receiver can still watch the streaming smoothly. The only different is the picture quality is worse than before.

When the network congestion occurs, the receiver switches the group from high quality group to the low quality group. Figure 5.9 show that after the network congestion is removed, the receiver that monitors the network in real time can detect the change of network condition. It will trigger the switching module to switch back to the high quality group. Our application can dynamically switch the group based on the network condition in order to provide the receiver best streaming.

Figure 5.8 Before and after the congestion

Figure 5.9 Congestion and after the congestion

5.5 Conclusion

In this chapter, we implement the admission control system to ensure the quality for receiver. This system includes the measurement probing algorithm we discussed in previous chapters by modified the regular multicasting protocol and embedded the multi-streaming mechanism to

dynamically switch groups to provide better streaming quality. This system support most current popular video and audio encoders and decoders. Therefore it can display most formatted streaming. The experiment proved it can be used for real quality control efficiently.

Chapter 6
Conclusion and Future Works

6.1 Summary

In this thesis, we present a comprehensive admission control and quality control for multicast network. We analyzed the problems faced by multicast network and compared with existed unicast solutions. Finally based on the character of multicast network, we present the novel packet probing measurement methods to measure multicast condition. Our multicast admission control and quality control system divides the receiver joining process into several stages. Pre-join stage is before the receiver joins the group, it runs as admission control by probing the network path. Next stage is branch point selections. Single link capacity measurement provides the receiver multiple choices to graft to the multicast tree. After the receiver joined the group, the system can dynamically switch among groups to ensure the streaming quality.

We first compare the difference between multicast network and unicast network. According the multicast tree structure and traffic behavior, we present a segment path capacity measurement approach based on different sized packet probing. This method utilizes the packet dispersion and packet life time to calculate the segment path capacity. This probing packet train can also measure end-to-end queuing delay by using the different priority. These two measurements together are inputs of admission control logic. This is the first stage of our system.

We then further explore the packet probing technique. We developed a single link capacity measurement algorithm by injecting probing packet train pair. A pair of probing packet train can measure a link capacity. We can use multiple probing packet train pair to measure any link capacity. Those link capacities provide the receiver the network resource distribution. The receiver can choose the best branch point to graft to the multicast tree.

The next part is machine learning and measurement mixed prediction algorithm. After the receiver joined the group, along with transmission of streaming traffic, the source inject a few probing packet into the network for receiver to monitor. The receiver measures the packet dispersion to infer the end-to-end delay. Those measurement results are used as training data for prediction logic together with user buffer information. The prediction logic predicts the near future network condition, triggers the group switching module to switch up or switch down the group in order to maintain the streaming quality under different network condition.

Finally we implemented the admission control and quality control system. This system includes probing packet measurement algorithm, single link capacity measurement algorithm, streaming rate control and group switching control. The system supports most of current popular video and audio encoder and decoder. It can real time display and stream most kind of video format. We set up the router, SITL and OPNET mixed test bed to run and test this admission control and quality control system. The experiments show the system performs efficiently and can satisfy most of user requirements.

6.2 Future Work

In the future, we should consider the different wireless network environment. The multicast protocol under wireless network is different to the multicast protocol under wired network. At

the same time, the traffic behavior in wireless network is also different. We should consider those differences to improve current probing algorithms.

Next for current system, the branch point selection is also an unsolved problem. The effective way to locate and select the correct branch point is also the work we should focus on.

The application we developed only supports two or three groups for the same source right now. The next goal is to allow the source to create groups based on the user's requirements.

Bibliography

[1] Zongkai Yang, Chunhui Le, Jianhua He, Chun Tung Chou, Wei Liu: An Enhanced Scalable Probe-Based Multicast Admission Control Scheme. IEICE Transactions 88-B(8): 3466-3470 (2005)

[2] Sudeept Bhatnagar, Badri Nath, Arup Acharya: Distributed Admission Control for Heterogeneous Multicast with Bandwidth Guarantees. IWQoS 2003: 115-136

[3] B. J. Vickers, C. Albuquerque, and T. Suda, Source-adaptive multi-layered multicast algorithms for real-time video distribution, *IEEE/ACM Transactions on Networking*, 8(6):720-733, December 2000.

[4] I. Más, V. Fodor, and G. Karlsson, "Probe-Based Admission Control for Multicast," in *Proc. ofInternational Workshop on Quality of Service (IWQoS) 2002*, May, 2002.

[5] Yu Cheng, W. Zhuang, "Effective bandwidth of multiclass Markovian traffic sources and admission control with dynamic buffer partitioning", IEEE Transaction on Communication, vol. 51, no. 9, pp. 1524–1535, Sep. 2003.

[6] Jinwoo Choe and Ness B. Shroff, "A Central-Limit-Theorem-Based Approach for Analyzing Queue Behavior in High-Speed Networks", IEEE/ACM Transaction on Networking, vol. 6, no. 5, October 1998

[7] S. Kim, J. Y. Lee, and D. K. Sung, "A shifted gamma distribution model for long-range dependent Internet traffic," IEEE Commun. Lett., vol. 7, no. 3, pp. 124–126, Mar. 2003.

[8] K. Lai and M. Baker, "Measuring link bandwidths using a deterministic model of packet delay", ACM SIGCOMM'00, Stockholm, Aug. 2000.

[9] I. Más, V. Fodor, and G. Karlsson, “Probe-Based Admission Control for Multicast”, in Proc. Of International Workshop on Quality of Service (IWQoS) 2002, May, 2002.

[10] Seung Yeob Nam, Sunggon Kim, and Dan Keun Sung, “Measurement-Based Admission Control at Edge Routers”, IEEE/ACM Transaction on Networking, vol. 16, no. 2, April 2008

[11] J. Qiu and E. Knightly, “Measurement-based admission control with aggregate traffic envelopes”, IEEE/ACM Transaction on Networking, April 2001.

[12] C. Bouras and K. Stamos, “An Adaptive Admission Control Algorithm for Bandwidth Brokers”, IEEE NCA 2004, Cambridge, MA, USA, pp. 243-250

[13] Z. Duan, Z.-L. Zhang, Y.T. Hou, and L. Gao, “A core stateless bandwidth broker architecture for scalable support of guaranteed services”, IEEE Transactions on Parallel and Distributed Systems, vol. 15, no. 2, pp.167-182, February 2004.

[14] Sudeept Bhatnagar and Badri Nath, “Distributed Admission Control to Support Guaranteed Services in Core-Stateless Networks”, IEEE INFOCOM 2003

[15] Weijia Jia, Dong Xuan, Wanqing Tu, Lidong Lin, and Wei Zhao, “Distributed Admission Control for Anycast Flows”, IEEE Transaction on Parallel and Distributed Systems, , vol. 15, no. 8, August 2004

[16] Coskun Cetinkaya, Vikram Kanodia, and Edward W. Knightly, “Scalable Services via Egress Admission Control”, IEEE Transaction on Multimedia, vol. 3, no. 1, March 2001

[17] Manish Jain and Constantinos Dovrolis, “End to End Available Bandwidth: Measurement Methodology, Dynamics, and Relation with TCP Throughput”, IEEE/ACM Transaction on Networking, vol.11, no.4, August 2003

[18] Sudeept Bhatnagar, Badri Nath and Arup Acharya: Distributed Admission Control for Heterogeneous Multicast with Bandwidth Guarantees. IWQoS 2003: 115-136

[19] I. Más, V. Fodor, and G. Karlsson, “Probe-Based Admission Control for Multicast”, in Proc. Of International Workshop on Quality of Service (IWQoS) 2002, May, 2002.

[20] B. J. Vickers, C. Albuquerque, and T. Suda, “Source-adaptive multi-layered multicast algorithms for real-time video distribution”, IEEE/ACM Transactions on Networking, 8(6):720-733, December 2000.

[21] Zongkai Yang, Chunhui Le, Jianhua He, Chun Tung Chou, Wei Liu, “An Enhanced Scalable Probe-Based Multicast Admission Control Scheme”, IEICE Transactions 88-B(8): 3466-3470 (2005)

[22] T. En-Najjary and G. Urvoy-Keller, “PPrate: A Passive Capacity Estimation Tool”, In IEEE e2emon, Vancouver, CANADA, 2006

[23] M. Fornasa, M. Maresca, P. Baglietto, N. Zingirian, “Passive Access Capacity Estimation for QoS Measurement”, Quality of Service, 2009. IWQoS. 17th International Workshop on, 2009

[24] Dina Katabi, C.B., “Inferring Congestion Sharing and Path Characteristics from Packet Inter-arrival Times”, MIT-LCS-TR828. 2001, MIT

[25] Sachin Katti, Dina Katabi, Charles Blake, Eddie Kohler, and Jacob Strauss, “MultiQ: Automated Detection of Multiple Bottleneck Capacities Along a Path”, In proceedings of the ACM SIGCOMM 04, Taormina, Sicily, ITALY, October 2004

[26] R. L. Carter and M. E. Crovella, “Measuring bottleneck Link Speed in Packet-Switched Networks”, Technical Report BU-CS-96-006, Boston University, 1996

[27] C. Dovrolis, P. Ramanathan, and D. Moore, “Packet-dispersion Techniques and a Capacity Estimation Methodology”, IEEE/ACM Trans. Networking, 2004

[28] A. Downey, “Using Pathchar to Estimate Internet Link Characteristics”, in ACM SIGCOMM’99, Boston, MA, Aug. 1999

[29] Khaled Harfoush, Azer Bestavros, and John Byers, “Measuring Capacity Bandwidth of Targeted Path Segments”, IEEE/ACM Transaction on Networking, vol. 17, no. 1, February 2009

[30] N. Hu, L. Li, and P. Steenkiste, “Locating internet Bottlenecks: Algorithms, Measurements and Implications”, in ACM SIGCOMM 2004, Portland, OR, Sep. 2004

[31] R. Kapoor, L. Chen, L. Lao, M.Gerla, and M. Sanadidi, “CapProbe: A simple and Accurate Capacity Estimation Technique”, in ACM SIGCOMM 2004, Portland, OR, Sep. 2004

[32] K. Lai and M. Baker, “Measuring Link Bandwidth Using a Deterministic Model of Packet Delay”, in ACM SIGCOMM’00, Stockholm, Aug. 2000

[33] K. Lai and M. Baker, “Nettimer: A Tool for Measuring Bottleneck Link Bandwidth”, in Proc, USITS’01, Mar. 2001

[34] B. Mah, “Pchar: A Tool for Measuring Internet Paths Characteristics”, http://www.employees.org/bmah/Software/pchar/, 2000

[35] Z. ZiXuan, B. Lee, C. Fu, and J. Song, “Packet Triplet: A novel Approach to Estimate Path Capacity”, IEEE Commun. Lett., vol. 9, no. 12, 2005

[36] Zhang Ming, Dou Haolei, and Chang Chunteng, “OPNET Modeler and Network Simulation”, monograph, Posts and Telecom Press, Beijing, 2007.

[37] Vladimir N. Vapnik, “The Nature of Statistical Learning Theory”, Springer, 1995.

[38] Vladimir N. Vapnik, “An overview of statistical learning theory”, IEEE Trans. Neural Networks, vol. 10, pp. 988-999, 1999

[39] A.J. Smola and B. Scholkopf, “A tutorial on Support Vector Regression”, Royal Holloway College, London, UK, NeuroCOLT Tech. Rep., 1998

[40] F. E. H. Tay and L. J. Cao, “Application of Support Vector Machines in Financial Time Series Forecasting”, Omega, vol. 29, pp. 309-317, 2001

[41] W. Lu, W. Wang, A. Y. T. Leung, S. M. Lo, “Air Pollutant Parameter Foresting using Support Vector Machines”, in Proc. 2002 Int. Joint Conf. on Neural Network, May 12-17 2002.

[42] R. Beverly, K. Sollins, and A. Berger. “SVM Learning of IP Address Structure for Latency Prediction”, In Proceedings of the SIGCOMM Workshop on Mining Network Data, Pisa, Italy, September 2006

[43] F. Baccelli, S. Machiraju, D. Veitch, J. Bolot., “On Optimal Probing for Delay and Loss Measurement”, ACM IMC, 2007.

[44] S. Suthaharan and S. Kumar, “Measuring available bandwidth: pathChirp’s chirp train structure remodeled,” ATNAC’2008, 7-10 Dec. 2008

[45] A. Cabellos-Aparicio, F. Garcia, and J. Domingo-Pascual, “A Novel Available Bandwidth Estimation and Tracking Algorithm,” in Proc. Network Operations and Management Symp. Work, Salvador da Bahia, Brazil, Apr. 2008.

[46] Sachin Katti, Dina Katabi, Charles Blake, Eddie Kohler, and Jacob Strauss, “MultiQ: Automated Detection of Multiple Bottleneck Capacities Along a Path”, the ACM SIGCOMM 04, ITALY, October 2004

[47] Dina Katabi, C.B., “Inferring Congestion Sharing and Path Characteristics from Packet Inter-arrival Times”, MIT-LCS-TR828. 2001, MIT

[48] C. Dovrolis, P. Ramanathan, and D. Moore, “Packet-dispersion Techniques and a Capacity Estimation Methodology”, IEEE/ACM Trans. Networking, 2004

[49] Z. ZiXuan, B. Lee, C. Fu, and J. Song, “Packet Triplet: A novel Approach to Estimate Path Capacity”, IEEE Communication Letter, vol. 9, no. 12, 2005

[50] Qihua Yang, Tarek Saadawi, Ahmed Abdelal, Mitesh Patel, Biao Jiang, “Multicast Multi-Streaming and Real-Time Stream Rate Control Demo”, IEEE Globecom conference, Demo Section, December 2010, Miami, FL

[51] B. Scholkopf, C. J. C. Burges and A. J. Smola, “Using Support Vector Machines for Time Series Prediction”, Cambridge, MA: MIT Press, 1999, pp. 242-253.

[52] J. Ma, T. James, and P. Simon, “Accurate on Line Support Vector Regression”, Neual Comput., vol. 15, pp. 2683-2703, 2003.