Bidimensional-Probe Multipath Congestion Control for Shared Bottleneck Fairness

Michio Honda

Graduate School of Media and Governance
Keio University
5322 Endo, Fujisawa Kanagawa 252-8520 JAPAN

Submitted in partial fulfillment of the requirements for the degree of Master of Media and Governance

Advisors:

Hideyuki Tokuda
Professor of Environmental Information, and Media and Governance, Keio University
Osamu Nakamura
Professor of Environmental Information, Keio University
Yoshifumi Nishida
Associate Professor of Graduate School of Media and Governance, Keio University

Summary

Bidimensional-Probe Multipath Congestion Control for Shared Bottleneck Fairness

Multi-homed hosts are becoming common and they have multiple paths between a source and a destination host. If transport protocols transmit data to multiple paths, they can improve available network capacity. Several such transport protocols have been proposed, however, they have a problem with fairness. When the transmissions along several paths share the same bottleneck router, an end-to-end multipath connection receives higher throughput than a competing regular TCP flow, because it executes congestion control per path with the same algorithm as TCP. We propose a congestion control scheme that addresses this problem based on the weighted Additive Increase Multiplicative Decrease (AIMD) algorithm. In our scheme, an end-to-end connection that uses flows along multiple paths can fairly compete with regular TCP-friendly flows at the shared bottleneck. In addition, in order to maximize utilization of different path characteristics, such as bandwidth capacity and RTT, our scheme probes the optimal proportion to apply the weight to flows on each path. Our simulation results show that a bundle of multiple flows based on our algorithm fairly competes with TCP flows at the same bottleneck.

Keywords: 1. Multipath, 2. congestion control, 3. fairness, 4. resource utilization, 5. transport protocols

Michio Honda Graduate School of Media and Governance

Keio University

Table of Contents

List of Figures

Chapter 1

Introduction

This chapter describes background of this research and the brief overview of the problem in existing proposals.

Efficient resource allocation is a key requirement for the future Internet. In 1999 [39], text or image-based web transactions comprised the majority of Internet traffic. In 2006, P2P traffic was 37%, and HTTP traffic containing video and audio (e.g., YouTube [40]) was 19% of the total Internet traffic [37]. These applications require high bandwidth allocations for a long time compared to text or image-based web transactions. TCP [31] traffic still comprises a major share of the total Internet traffic [16]. Based on this observation, we assume that high-speed and long-lived TCP connections are becoming more frequent. Such TCP connections remain in the congestion-avoidance phase [1], and hence compete against each other at shared bottlenecks. This means that users or applications require more network capacity.

Simultaneous multipath utilization is a promising way in which end hosts can increase available network capacity. As the market for networking technology is evolving, it is becoming more common that end hosts are equipped with multiple network interfaces (e.g., WLAN, GPRS and 3G). They have multiple links that are connected to the Internet simultaneously, which results in availability of multiple paths between a source and a destination end host. Such multi-homed hosts can use more available network capacity if they aggregate the available bandwidth of multiple paths.

We assume that transport protocols will have capabilities to utilize multiple paths simultaneously between a source and a destination host. Fig. 1.1 illustrates such a multipath communication. Because each host is equipped with two network interfaces, multiple paths exist between the two transport layer endpoints (EPs) on the hosts. There are actually four paths between these hosts, however Fig. 1.1 depicts only two paths, for simplicity. Two applications are communicating over a transport layer connection between these hosts. The most important concept of this architecture is the **multipath connection** that contains multiple **subflows**. A multipath connection is the entity over which applications communi-

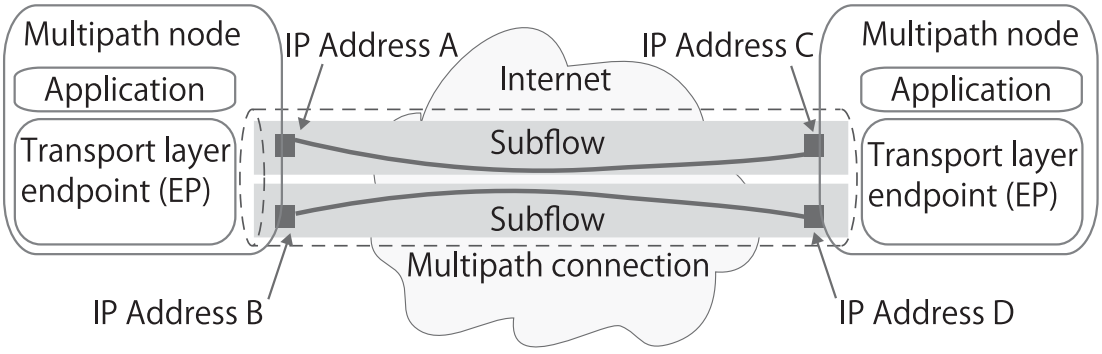

Figure 1.1: Architecture overview

cate. For example, a multipath connection looks like a TCP connection to the application and provides a reliable and ordered byte stream. The endpoint stripes user data across multiple distinct paths, using one subflow along each path. A different approach to achieve multipath communication might be possible, such as implementing a shim layer [26] between the network and the transport layers. However, shim layers can conceal the existence of multiple paths to the transport layer, and because the transport layer performs congestion control, it is important that it manages multiple paths directly.

Current connection-oriented transport protocols (e.g., TCP, SCTP [36] and DCCP [20]) transmit data only over a single path between a source and a destination hosts at any given time. Although SCTP supports multi-homing, standard SCTP does not transmit data over multiple paths simultaneously. However, several proposals extend transport protocols to simultaneously utilize multiple paths [14, 41, 6, 34, 15, 22]. These extensions can achieve higher throughput than the base protocols, because they independently perform TCP congestion control on each subflow for effective utilization of distinct paths.

Unfortunately, none of these extensions properly utilizes multiple paths because of the utilized congestion control mechanism. They crowd out competing TCP flows at a shared bottleneck, because each subflow independently performs congestion control with an algo-

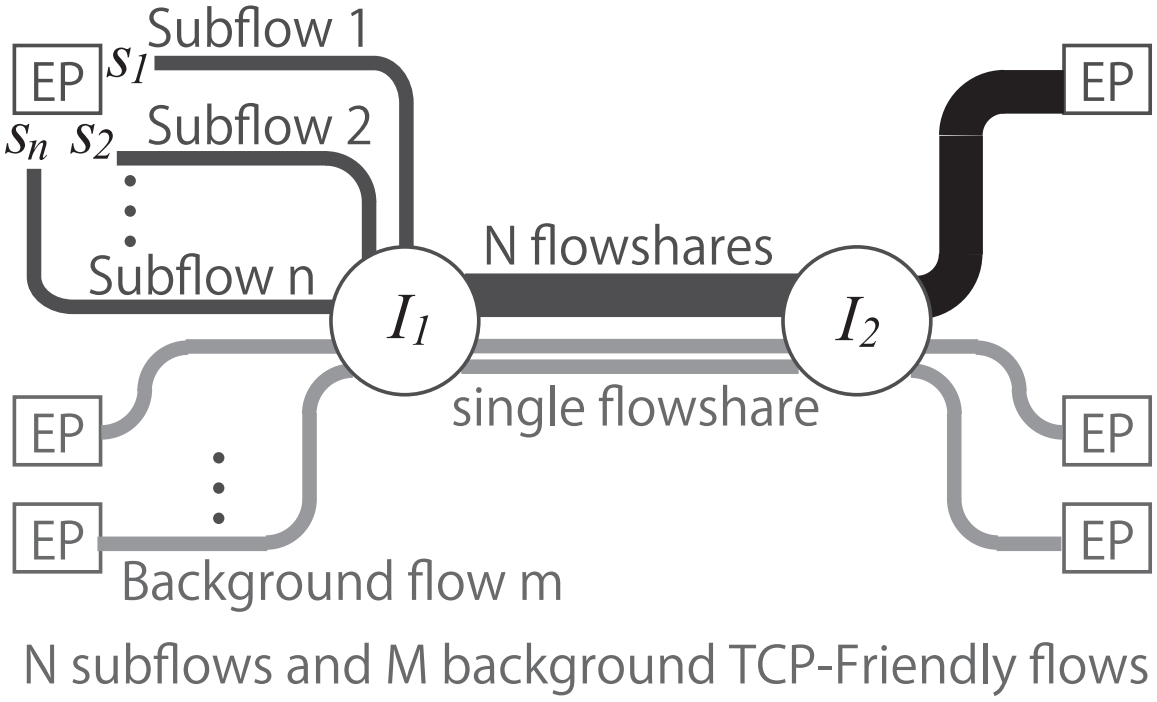

Figure 1.2: Unfair share at the shared bottleneck

rithm similar to TCP. When N subflows in a multipath connection compete against TCP flows at the same bottleneck, the multipath connection is approximately N times as aggressive as each of the TCP flows. In Fig. 1.2, one multipath connection that contains N subflows competes with M background TCP flows at a shared bottleneck between two intermediate nodes I_1 and I_2. While each of background flows receives a $1/(N+M)$ share of the bottleneck, a bundle of N subflows receives a $N/(N+M)$ share.

In Internet congestion control, a congestion-controlled flow between two transport endpoints (e.g., a single TCP connection) uses a single flowshare [29]. N flowshares receive throughput N times a single flowshare, which are called multiple flowshares [29]. Multiple flowshares are mainly utilized to achieve weighted proportional fairness [5] between aggregation points that bundle multiple flows transmitted by multiple users. For example, distributed-multimedia applications [28], Edge-to-Edge QoS [35] and wireless TCP proxies [4]

use aggregation points. Some applications leverage parallel TCP connections between the same hosts to obtain more bandwidth or avoid head-of-line blocking. Such use of multiple TCP connections is unfair to other traffic sharing the path. Congestion control of these connections should reduce their bandwidth consumption to that of a single flowshare, e.g., as with E-TCP [7] and CM [2]. Even if an endpoint utilizes multiple paths for transmission, the end-to-end connection is essentially a single connection in the communication primitive to applications. For example, it is represented as a reliable and ordered stream to the application. Consequently, the endpoint should use a combined single flowshare at the shared bottleneck for all subflows.

Sharing links among multiple end-to-end paths exist in a wide range of contexts in multi-homed environments. The most straightforward scenario is duplicate use of a source or a destination address, for example, when two subflows transmit data from different source addresses to the same destination address, or vice versa. This situation always occurs in multipath connections between a multi-homed host and a single-homed host. When both hosts are multi-homed, they can utilize multiple paths without duplicate use of same addresses. However, even if the endpoint uses only paths where both the source and destination addresses are different, some intermediate nodes can be shared.

ISPs can restrict or rate-limit the user accounts that transmit data excessively, if their end-to-end congestion control is insufficient due to a large number of parallel TCP connections or because it is unresponsive to congestion [9]. However, different subflows in a multipath connection can traverse different ISPs. It is difficult for one ISP to observe traffic inside another ISP. Because the entire multipath connection is not visible to an ISP, determining an appropriate restriction by an ISP is difficult. End-to-end congestion control is more important for multipath communication than that for single-path communication.

End-to-end multipath congestion control requires two properties, which are:

- **Fairness**

  TCP-friendliness is the most common fairness metric in the Internet. A multipath connection has the same communication primitive as a single TCP connection or the other end-to-end connection, as far as the application is concerned. Therefore, a multipath connection should be TCP-friendly at the shared bottleneck regardless of the number of its subflows.

- **Utilization**

  Distinct paths have different characteristics, such as RTT and spare bandwidth. In order to maximize the performance of the whole multipath connection, effective utilization of a shared bottleneck and spare bandwidth of distinct paths is desired.

In this paper, we investigate congestion control for multipath transport protocols. The main contribution of this paper is a new end-to-end congestion control scheme for multi-homed environment: Bidimensional-Probe Multipath Congestion Control (BMC). Using BMC, a bundle of subflows in a multipath connection fairly competes with background TCP-friendly flows at a shared bottleneck. In addition, BMC maximizes utilization of resources that along multiple paths with different characteristics. The remainder of this paper is organized as follows: Chapter 2 discusses the over-aggressiveness problem of per-subflow TCP congestion control. In Chapter 3, we propose a congestion control algorithm for multipath-enabled transport protocols. Chapter 4 discusses the challenges that arise when subflows traverse completely disjoint paths. In addition, we discuss the adaptability to weighted proportional fairness. Chapter 5 evaluates our algorithm through simulations. Chapter 6 describes related work, and the paper concludes with Section 7.

Chapter 2

Problem Analysis

This chapter details the problem of existing proposals that independently execute TCP congestion control per path, based on the increase rate of the sending rate, loss-event behavior and the overall throughput.

Independent congestion control per subflow is reasonable to maximize utilization of multiple distinct paths, because it can avoid subflows being affected by loss events on different subflows. TCP's congestion control algorithm is the most common congestion control algorithm on the Internet, hence many prior multipath transport schemes use it for each subflow, independently of one another. Since BMC is designed based on TCP's congestion control, we clarify the over-aggressiveness problem of these schemes by formulating the behavior.

The Additive Increase Multiplicative Decrease (AIMD) algorithm is based on TCP's standard congestion control in the congestion-avoidance phase. The AIMD algorithm additively increases the sending rate when packets are acknowledged without a packet loss within one RTT, and otherwise multiplicatively decreases the sending rate, by adjusting the congestion window. A general formulation of the AIMD algorithm refers to these as the "increase parameter" a and "decrease parameter" b. When the AIMD algorithm increases the sending rate, it increases the current window size by a packets. When the AIMD algorithm decreases the sending rate, it decreases the window size from W to $(1-b)W$. Hence, the sending rate or the window size varies in a sawtooth shape. The period beginning with a congestion window of $(1-b)W$ is called a *congestion epoch* [10]. Fig. 2.1 illustrates the relationship between the AIMD parameters, the window size, the congestion epoch and RTT. Since TCP increases window size by one MTU and otherwise decreases window size in half in steady-state, it uses 1 and 1/2 as a and b, respectively. In this paper, we denote the AIMD algorithm with parameters a and b as $AIMD(a, b)$.

There are two models for mathematically analyzing the behavior of the AIMD algorithm. One is the stochastic model introduced in [25, 30], and the other is the deterministic model introduced in [9, 8]. In this paper, we investigate the AIMD algorithm using the deterministic model. The stochastic model would give a more accurate model, however, the deterministic model is more convenient to formulate AIMD-based multipath congestion control

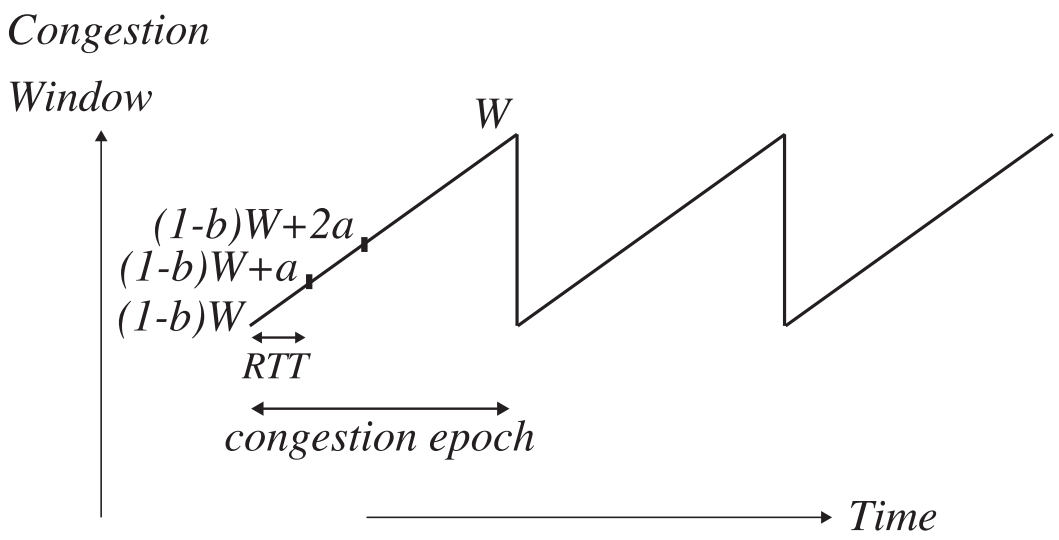

Figure 2.1: Behavior of the AIMD algorithm

with various a and b.

2.1 Increase and Decrease Behavior

The first component that affects the aggressiveness of the AIMD algorithm is the increase behavior of a bundle of subflows. We compare the increase behavior of a bundle of N subflows that each follow $AIMD(1, 1/2)$, and that of a single flow following $AIMD(1, 1/2)$ based on the situation in Fig. 1.2. In the prior proposals, since each subflow independently increases the sending rate by one packet per RTT, the sum of increase parameters of N subflows is N. TCP's AIMD, $AIMD(1, 1/2)$ increases the sending rate by one packet per RTT, thus $AIMD(1, 1/2)$ increases the sending rate $1/RTT$ packets per second. Hence, the increase rate of N subflows, δ_N packets per second can be determined by:

$$\delta_N = \sum_{n=1}^{N} \frac{1}{R_n} \tag{2.1}$$

where R_n is the RTT of subflow$_n$. Based on this equation, when the number of subflows increases, aggregate sending rate of the multipath connection increases rapidly, unless the

RTT of the subflows is much smaller than that of the competing background flows.

The other component that affects the aggressiveness of the AIMD algorithm is response to a packet loss. When the sending rate is T, $AIMD(1, 1/2)$ decreases it to $T/2$ in response to a single packet loss. However, a bundle of N subflows does not decrease the sending rate by half, because each subflow adjusts its sending rate individually. Only the subflow that experiences the loss event decreases the window size or the sending rate in response to a single packet loss. In Fig. 1.2, we denote the sending rate of subflow$_n$ as T_n, hence the sum of sending rate of N subflows is $\sum_{n=1}^{N} T_n$. When a packet is lost on subflow$_i$ $(1 \leq i \leq n)$, the decreased sending rate of N subflows T_N is:

$$T_N = \sum_{n=1}^{N} T_n - T_i/2 \tag{2.2}$$

When we assume uniformly distributed RTTs, each subflow behaves stochastically similar at the bottleneck. Then, the sending rate of N subflows after a packet loss decreases from T to $(N-1)T + (T/2)$.

Based on Equation (2.1) and (2.2), a multipath connection that obeys per-subflow TCP congestion control increases its sending rate more rapidly, and decreases its sending rate less conservatively than a single TCP connection does. This behavior is the reason for the over-aggressiveness, and it is similar to AIMD variants that achieve weighted proportional fairness, such as MulTCP [5, 21].

2.2 Throughput of Per-Subflow TCP Congestion Control

Under the deterministic model, the sending rate T, the number of packets transmitted in one second in $AIMD(a, b)$, is given by $(2-b)W/2RTT$ [10], where W is the window size at

the end of a congestion epoch. Hence, the throughput of $AIMD(1, 1/2)$, T_W is given by:

$$T_W = \frac{3}{4RTT}W \tag{2.3}$$

Based on (2.3), when each subflow follows $AIMD(1, 1/2)$, the total throughput of N subflows, T_{WN} is:

$$T_{WN} = \sum_{n=0}^{N} \frac{3}{4R_n}W_n \tag{2.4}$$

where R_n is the RTT of subflow$_n$. W_n is the congestion window size of subflow$_n$ at the end of the congestion epoch. When each subflow has the same RTT, W in (2.3) and W_n in (2.4) become approximately equal. Thus, the bundle of subflows receives throughput in proportion to the number of subflows at the bottleneck without the RTT bias of the AIMD algorithm [23]. Therefore, although per-subflow TCP congestion control meets the utilization property due to the loss-event behavior, it does not satisfy the fairness property.

Chapter 3

Designing Bidimensional-Probe Multipath Congestion Control

This chapter investigate the design space for multipath congestion control. Then we propose a new congestion control scheme, which is twofold; *Aggressiveness Manager* and *Proportion Manager*.

In order to satisfy the fairness and utilization properties for multipath connections, we consider two possibilities. The first one is congestion window sharing between subflows in the same multipath connection. This approach is implemented in E-TCP [7], CM [2] and MPAT [35] for aggregate congestion control of multiple TCP connections along the same path. The aggregated congestion window is increased by one packet when packets are acknowledged without a packet loss of any connections sharing it in an RTT. It is reduced in half by a packet loss on any connection sharing the aggregated congestion window. In order to apply this approach to multipath connections, we could allocate the window space to subflows in proportion to the spare bandwidth along each path. However, the congestion window is the allowed amount of data to be transmitted in an RTT, which is adjusted with the ACK-clocking interval. Hence, subflows with different RTTs cannot share the same congestion window, because they experience events to increase or decrease the window size with different interval. In addition, since packet loss on any subflow reduces the shared congestion window, behavior of a subflow is affected by that of other subflows. This leads to performance degradation of the whole multipath connection. Therefore, this approach is insufficient not only in terms of fairness, but also in terms of utilization.

The second approach is to apply weight to individual congestion control of each subflow so that a bundle of subflows can have the same aggressiveness as one TCP flow. In this approach, each subflow independently performs congestion control, adjusting its own congestion window. Hence, each subflow is not affected by the loss events on the other subflows. In addition, it can work if the RTTs of the subflows are different, because subflows do not need to share one congestion window. For this reason, we design our congestion control scheme based on this approach. In order to achieve fairness and utilization properties, two components of BMC, the *Aggressiveness Manager* and the *Proportion Manager* are introduced. The aggressiveness manager maintains fairness at the shared bottleneck. The proportion

manager effectively utilizes spare bandwidth of distinct paths, which maximizes the chances of full utilization of the shared bottleneck.

3.1 Aggressiveness Manager

The aggressiveness management maintains a constant aggressiveness for the overall multipath connection. It is critical for a bundle of subflows to compete fairly with background TCP-friendly flows which share the same bottleneck. Each of the individual subflows must be less aggressive than a TCP-friendly flow. In addition, the bundle of subflows must be as aggressive as a single TCP-friendly flow. We achieve this by using the weighted AIMD algorithm for each subflow, which receives throughput in proportion to the weight compared to TCP. The aggressiveness manager applies a weight parameter to each subflow, at the same time maintaining the sum of the weight parameters so that a bundle of subflows is as aggressive as one TCP flow.

We define the weight of standard TCP connection as 1, and denote the weight of a subflow$_n$ as D_n ($0 < D_n \leq 1$). A subflow with a weight parameter D_n receives D_n times as much throughput as a competing TCP flow. Hence, when a bundle of N subflows receives the same throughput as a TCP flow, the following equilibrium is satisfied:

$$\sum_{n=1}^{N} D_n = 1 \tag{3.1}$$

The aggressiveness manager maintains this equilibrium across the subflows in a multipath connection.

The aggressiveness manager applies the AIMD parameters to each subflow so that an individual subflow achieves proportional throughput with regard to the weight parameter. We investigate the relationship between the AIMD parameters and the weight parameter that achieves proportional throughput to the weight compared to a TCP flow. The throughput

of the AIMD algorithm is given as a function of the additive increase and multiplicative decrease parameters: a and b, round-trip time R and packet loss rate p [10]:

$$T = \frac{\sqrt{2-b}\sqrt{a}}{\sqrt{2b}R\sqrt{p}} \tag{3.2}$$

The throughput of $AIMD(1, 1/2)$, T_t is given by applying $a = 1$ and $b = 1/2$ into Equation (3.2):

$$T_t = \frac{1}{R}\sqrt{\frac{3}{2p}} \tag{3.3}$$

When we want D times $(0 < D \leq 1)$ of $AIMD(1, 1/2)$ throughput for an individual subflow, the desired throughput T_d is:

$$T_d = \frac{D}{R}\sqrt{\frac{3}{2p}} \tag{3.4}$$

In order to acquire the AIMD parameters a and b that achieves throughput in Equation (3.4), we investigate the equilibrium between Equation (3.4) and the response function in Equation (3.2):

$$\frac{D}{R}\sqrt{\frac{3}{2p}} = \frac{\sqrt{2-b}\sqrt{a}}{\sqrt{2b}R\sqrt{p}} \tag{3.5}$$

Hence, the relationship between a, b and D can be determined by the following equation.

$$a = \frac{3b}{2-b}D^2 \tag{3.6}$$

This means that $AIMD(1/2, 4/5)$ receives half the throughput compared to $AIMD(1, 1/2)$. $AIMD(3/4, 8/13)$ receives 3/4 of the throughput of $AIMD(1, 1/2)$. The aggressiveness manager applies a parameter pair satisfying Equation (3.6) to each subflow based on the weight parameter. The sum of the weight parameters of the subflows satisfies Equation (3.1). Then, a bundle of these subflows fairly competes with $AIMD(1, 1/2)$ flows that share the same bottleneck link. When the weight parameter D is determined, there are infinite combinations

of a and b. On the other hand, since the congestion window size is decreased from W to $(1-b)W$, we have to avoid b exceeding 1. Hence, the aggressiveness manager applies D^2 to a. Applying $a = D^2$ into Equation (3.6), b is always 1/2. For example, suppose if an endpoint has two subflows and their weight parameters are $D_1 = 2/3$ and $D_2 = 1/3$ with satisfying Equation (3.1). In this case, these subflows follow $AIMD(4/9, 1/2)$ and $AIMD(1/9, 1/2)$, respectively.

3.2 Proportion Manager

The proportion manager optimizes the combination of weight parameters for the subflows to effectively utilize spare bandwidth of distinct paths, which maximizes the chances of full utilization of the shared bottleneck. The aggressiveness manager maintains a bundle of subflows in the multipath connection as aggressive as a TCP flow at the shared bottleneck. However, it assumes every subflow is constrained at the shared bottleneck. If some subflows are constrained by disjoint links due to the limitation of the spare bandwidth, the multipath connection cannot achieve equal throughput with that of a TCP flow that competes at the shared bottleneck. This means that utilization of the shared bottleneck is not maximized. Therefore, we have to apply the weight to subflows so that the desired throughput of the subflow does not exceed the spare bandwidth of disjoint links.

Fig. 3.1 illustrates the situation that a subflow is constrained by the spare bandwidth of a disjoint link. Fig. 3.2 illustrates the optimized situation, which maximizes the utilization of the shared bottleneck with the adjustment of the weight parameters. In Fig. 3.1 and 3.2, the capacity of the shared link between I_1 and I_2 is 30 Mbps, and two subflows and one standard TCP flow exist there. In Fig. 3.1, the weight parameter of both the subflow$_1$ and the subflow$_2$ is 1/2. Hence, each of them should share 7.5 Mbps throughput at the shared link so that a bundle of them receives the equal throughput with the competing TCP flow.

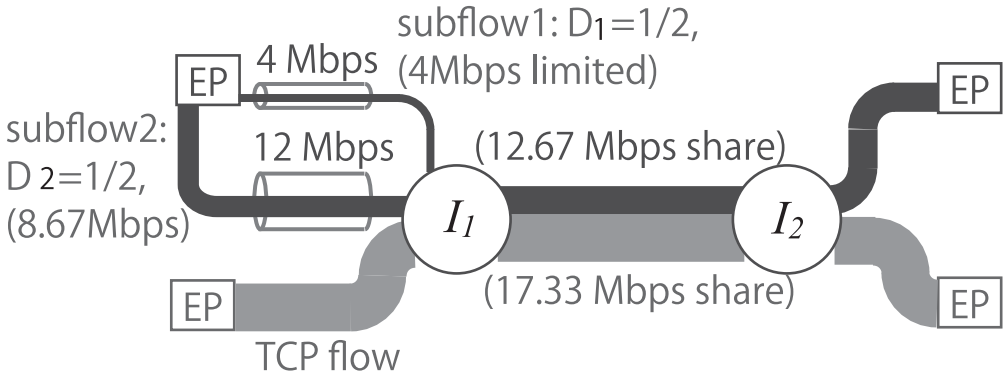

Figure 3.1: Ineffective utilization of disjoint link and shared link

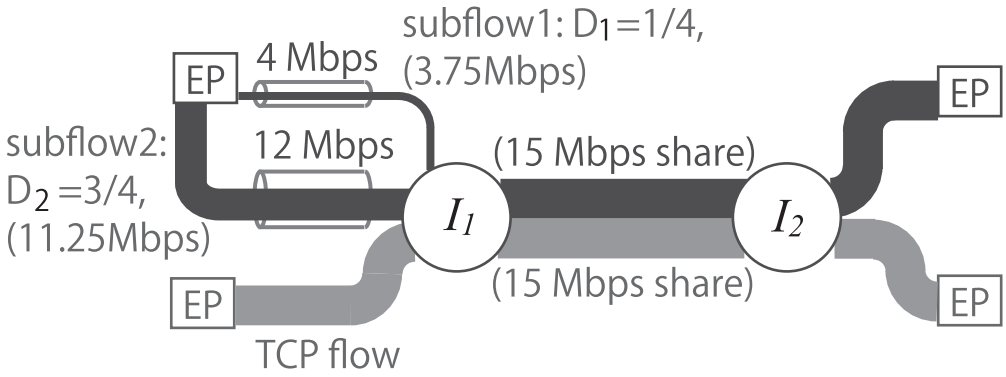

Figure 3.2: Effective utilization of disjoint link and shared link

On the other hand, the spare bandwidth of the disjoint link that the subflow$_1$ traverses is 4 Mbps. Since the maximum throughput of this subflow is limited to 4 Mbps, this subflow cannot utilize the shared link ideally. The capacity of the disjoint link that the subflow$_2$ traverses is 12 Mbps. The subflow$_2$ receives 8.67 Mbps $(((30-4)/3)\times 1)$ throughput at the shared link, because the weight is 1/2, which receives 1/2 times as much throughput as the standard TCP flow. As the result, the bundle of these subflows receives 12.67 Mbps throughput at the shared link, which is less than that of the TCP flow. In Fig. 3.2, the weight parameters of the subflow$_1$ and the subflow$_2$ are 1/4 and 3/4, respectively. Hence, the ideal throughput at the shared link is 3.75 Mbps $((30/8)\times 1)$ on the subflow$_1$ and 11.25 Mbps $((30/8)\times 3)$ on the subflow$_2$. In this case, the bundle of the subflows can receive as

much throughput as the competing TCP flow at the shared link, because capacity of their disjoint link is larger than their ideal throughput. Therefore, we apply weight to subflows so that their ideal throughput does not exceed the spare bandwidth of disjoint links.

The basic concept of the algorithm that probes optimal combination of weight parameters is to analyze the cause of the throughput that is achieved by one combination of weight parameters. For this purpose, the proportion manager performs the following two steps repeatedly.

1. Measure throughput of each subflow during a certain period, J seconds
2. Change the combination of weight parameters for subflows

The rationale for this behavior is that each subflow should receive throughput in proportion to the other subflows based on the weight and RTT, if subflows are constrained by a shared bottleneck, according to Equation (3.4). When the received throughput is less than that, the subflow can be constrained by the spare bandwidth of a disjoint link. This situation affects the loss-event rate p in Equation (3.4). If we reduce the weight of that subflow and increase the weight of another subflow, we can pass over the spare bandwidth limitation of the disjoint link. Therefore, we can increase the total throughput of the multipath connection.

In order to realize this behavior, the proportion manager measures the throughput during J seconds on each subflow. Subsequently, the proportion manager chooses two subflows, which are a subflow of which the weight should be reduced, and another one of which the weight should be increased. The proportion manager selects them based on a value which has deducted the effect of the weight and RTT from the measured throughput. The proportion manager reduces the weight of a subflow of which that value is the smallest. At the same time, the proportion manager increases the weight of a subflow of which that value is the largest. We denote the value which has deducted the effect of the weight and RTT as T_{wr}.

Based on Equation (3.4), which is calculated by following:

$$T_{wr} = \frac{R}{D} T_J \tag{3.7}$$

where T_J is the throughput of a subflow during J second measurement. D and R are the weight parameter and RTT of that subflow, respectively. The proportion manager decreases the weight of a subflow that has achieved minimum T_{wr}, at the same time increases the weight of a subflow that has achieved maximum T_{wr}.

The changing factor of the weight of subflows is important for the quick convergence to the optimal proportion and the stability. If we drastically reduce the weight of a subflow that achieves higher throughput, the throughput after the reduction is seriously reduced. In addition, if we drastically increase the weight of a subflow that weighs very low, it leads to performance degradation of the other subflows that has weighed higher. Thus, the proportion manager maintains subflows with larger weight or smaller weight more conservatively about increasing or decreasing the weight. This allows the subflows with larger weight parameters to maintain aggressiveness, hence the higher throughput of them are maintained. In addition, it allows the subflows with lower weight parameters to maintain conservativeness, hence it can avoid effect to the higher-throughput subflows.

In order to implement such behavior, when the selected subflow to reduce the weight has the more outstanding weight, we reduce the weight more conservatively. We denote the weight parameters before and after the reduction as D_{cur}^{dec} and D_{new}^{dec}. Then, the D_{new}^{dec} is calculated by the following equation:

$$D_{new}^{dec} = D_{cur}^{dec} - (D_{cur}^{dec} - (D_{cur}^{dec})^2)K \tag{3.8}$$

where K is a fixed parameter that affects the changing factor ($0 < K < 4$). For example, when $D_{cur}^{dec} = 1/2$ and $K = 1$, the D_{new}^{dec} is $1/4$. When $D_{cur}^{dec} = 4/5$ and $K = 1$, the D_{new}^{dec}

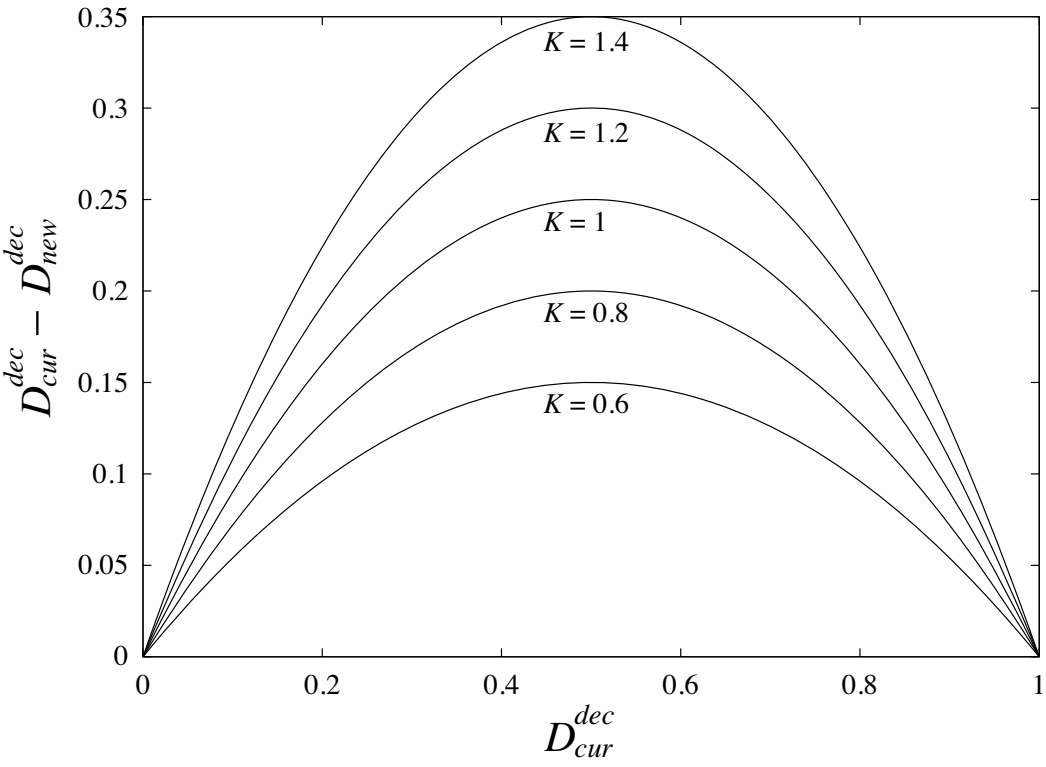

Figure 3.3: Difference between D_{cur}^{dec} and D_{new}^{dec}

is 16/25. Fig. 3.3 shows the relationship between D_{cur}^{dec} and the amount of the weight to be reduced. This means that the reduction of the weight is less when the original weight is more outstanding. When the original weight is close to 1/2, the reduction is larger in proportion to K.

Based on the reduction of the weight of a subflow, the proportion manager increases the weight of another subflow. We denote the weight parameters before and after increasing as D_{cur}^{inc} and D_{new}^{inc}. In order to maintain the multipath connection as aggressive as a single TCP connection based on Equation (3.1), D_{new}^{inc} is calculated by the following:

$$D_{new}^{inc} = D_{cur}^{inc} + (D_{cur}^{dec} - D_{new}^{dec}) \tag{3.9}$$

Subsequently, the proportion manager restarts throughput measurement of subflows for

J seconds. When each subflow receives proportional throughput based on the weight and RTT in Equation (3.7), we consider that the shared bottleneck is ideally utilized, otherwise every subflow is constrained by their disjoint links.

J and K are fixed parameters that can be tuned by the system administrator or applications. J has to be set to contain at least one congestion epoch of all subflows. One congestion epoch consists of $(bW/a) + 1$, where W is the window size at the end of congestion epoch. a and b are increase and decrease parameters of the AIMD algorithm, respectively. Hence, when the endpoint detects loss event on a subflow, it can calculate the congestion epoch of that subflow based on the maximum window size. The larger J is used, the more exact average throughput is measured. However, the convergence speed to the optimal proportion becomes slower in proportion to J. For example, when we double J, it doubles the convergence speed. Larger K enables the convergence speed to the optimal proportion to become rapidly, however, the stability during the convergence state goes down.

3.3 Requirements for Implementation

Transport protocols that obey BMC have to implement a weight parameter per subflow and fixed parameters J and K, in addition to the general parameters required to implement the standard AIMD algorithm. In addition, the throughput measurement feature has to be implemented per subflow. Since J and K are fixed parameters, they will be implemented as *sysctl* parameters in UNIX-based operating systems. They might also be tuned through the socket interface, such as *setsockopt()*. In addition, the initial weight parameters can be set as a policy by the system administrator or by applications. For example, an interface-type-based policy is one possibility, which specifies a larger initial weight for subflows that use a higher-speed network interface.

In order to implement BMC more easily, the transport protocol should implement se-

quence numbers and feedback per subflow. Per-subflow sequence numbers are useful to implement per-subflow RTT measurements and loss-event detection. It also makes per-subflow throughput measurement easy. In this case, T_J in Equation (3.7) can be acquired through the acknowledged number at the beginning and the end of the measurement. Therefore, reliable multipath transport protocols will implement both per-subflow sequence numbers and per-connection sequence numbers. Per-connection sequence numbers are used so that the receiver endpoint reassembles data from different subflows in the multipath connection. Since pTCP [14] and AMS [34] implement per-subflow and per-connection sequence numbers, they can be extended to implement BMC easily.

Chapter 4

Discussion

This chapter discusses our scheme at the view point of two situations. First, we discuss how our scheme works on the disjoint bottlenecks. Second, we discuss how our scheme achieves weighted proportional fairness on the shared bottleneck.

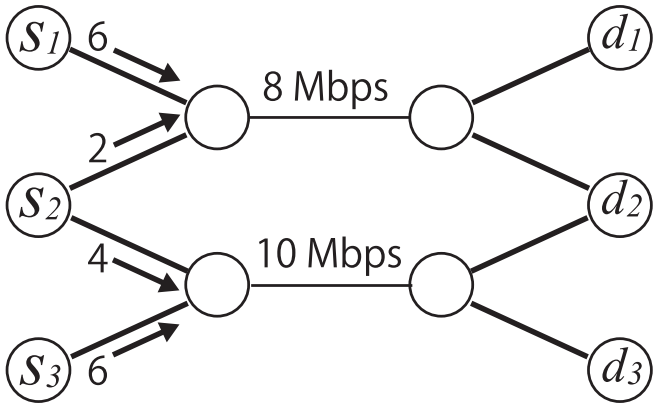

Figure 4.1: Resource pooling

4.1 Resource Pooling

BMC can achieve not only fairness at the shared bottleneck, but also resource pooling based on proposed principles [17, 19] along disjoint bottlenecks. Fig. 4.1 illustrates the resource allocation achieved by such principles. s_1, s_2 and s_3 are communicating with d_1, d_2 and d_3, respectively. There are two paths between s_2 and d_2. The sum of the available throughput of the two bottlenecks is 18 Mbps, hence each source should send at 6 Mbps to share it equally. If each source utilizes only a single path, a fair resource allocation is not achieved. On the other hand, if s_2 performs per-subflow TCP's congestion control, s_1, s_2 and s_3 receive 4, 9 and 5 Mbps of throughput, respectively. Hence, per-subflow TCP's congestion control is inappropriate not only for the shared bottleneck, but also for the resource pooling.

Our BMC achieves fair resource sharing based on the principle in Fig. 4.1. Our proportion manager converges to make subflows achieve throughput in proportion to the weight parameter and RTT based on Equation (3.7). In Fig. 4.1, we denote the weight parameters for subflows from s_2 on 8 Mbps and 10 Mbps bottlenecks as D_1 and D_2, respectively. We define RTT of each path as 1. In Fig. 4.1, since s_2 receives 2 Mbps at 8 Mbps bottleneck,

and 4 Mbps at 10 Mbps bottleneck, both flows transmitted by s_2 has $T_{wr} = 6$. Hence, the weight parameters for these flows are $D_1 = 1/3$ and $D_2 = 2/3$, respectively. If these weight parameters are different, an equal resource allocation is not achieved. For example, if both of these flows have weight parameter $1/2$, T_{wr} in Equation (3.7) for each flow is $3/4$ and $3/5$, respectively. Hence, our proportion manager reduces D_1 and increases D_2, according to the algorithm. If $D_1 = 1/4$ and $D_2 = 3/4$, T_{wr} for D_1 is 6.4 and T_{wr} for D_2 is 5.71. In this case, throughput that is received by s_2 is also less than that achieved by $D_1 = 1/3$ and $D_2 = 2/3$, which is 5.89. Our proportion manager reduces D_2 and increases D_1.

4.2 Weighted Proportional Fairness

Weighted proportional fairness is achieved through congestion control that uses multiple flowshares [5]. Multiple flowshares are utilized based on the number of flows in the aggregated flow or based on the cost in the cost-fairness criterion [3]. Since BMC is based on the AIMD algorithm, it can achieve multiple flowshares with small tuning. For example, we can achieve P flowshares ($P > 1$) by changing right-hand side of Equation (3.1) to P. It will be also optimized by the mechanism of PA-MulTCP [21] that improves fairness of MulTCP [5]. Multiple flowshares of BMC might be useful when multiple paths are available between aggregation points that bundle up multiple flows transmitted by multiple users.

In another scenario, the use of multiple wireless interfaces might be considered as the user pays a multiplicative cost because of battery consumption or subscription costs to multiple ISPs. In this case, multiple flowshares based on the number of wireless interfaces might be reasonable. Even in these cases, per-subflow TCP congestion control is insufficient. The number of wireless interfaces or the number of ISPs which the user pays do not directly correspond to the number of subflows. Multiple subflows can traverse the same access link due to multiple destinations or selective intermediate routers that are introduced in [12, 13].

Therefore, even if cost fairness is promoted, the weight that is applied to the multipath connection should not be determined by the number of subflows.

Chapter 5

Evaluation

This chapter evaluates our scheme with simulation results, which show how our scheme competes with TCP flows at the same bottleneck.

In this section, we evaluate fairness of BMC with the ns-2 network simulator [27]. We substitute TCP connections with the weighted AIMD algorithm for subflows of the multipath connection. This eliminates the effect of receiver-buffer blocking and handling packets that are received out-of-order, which could influence the behavior of the multipath transport protocols in the experiment. This is a deliberate decision, as the paper focuses on congestion control for multipath transport protocols. Protocol performance caused by the receiver-buffer blocking or out-of-ordered packet handling is separate area.

First, for the most fundamental experiment, we observe the throughput of the weighted AIMD algorithm of which the weight parameter is less than one to emulate the behavior of individual subflows. In this experiment, we make AIMD flows with the same weight parameter compete with standard TCP flows at the same bottleneck. Second, we observe the throughput of the bundle of the weighted AIMD flows with different weight parameters, which the sum is 1. This experiment is conducted to confirm that a bundle of the weighted AIMD flows receives equal throughput with a standard TCP flow, when the sum of the weight parameters is 1.

5.1 A Comparison of the Weighted AIMD Flows and TCP Flows

Fig. 5.1 illustrates the simulation setup. In this simulation, the weighted AIMD flows and standard TCP flows compete at the 60 Mbps bottleneck link with RED [11] queue management. The delay at the bottleneck link is set to 20 ms. Each sender is connected to a 100 Mbps link with 2 ms delay in front of the bottleneck. Each receiver is connected to a 100 Mbps link at the other side of the bottleneck link. The delay of the receiver link is set to the value of the index of the flow divided by 3. This means that nth standard TCP flow and weighted AIMD flow have $n/3$ ms delay at the receiver links. The TCP version is Reno in

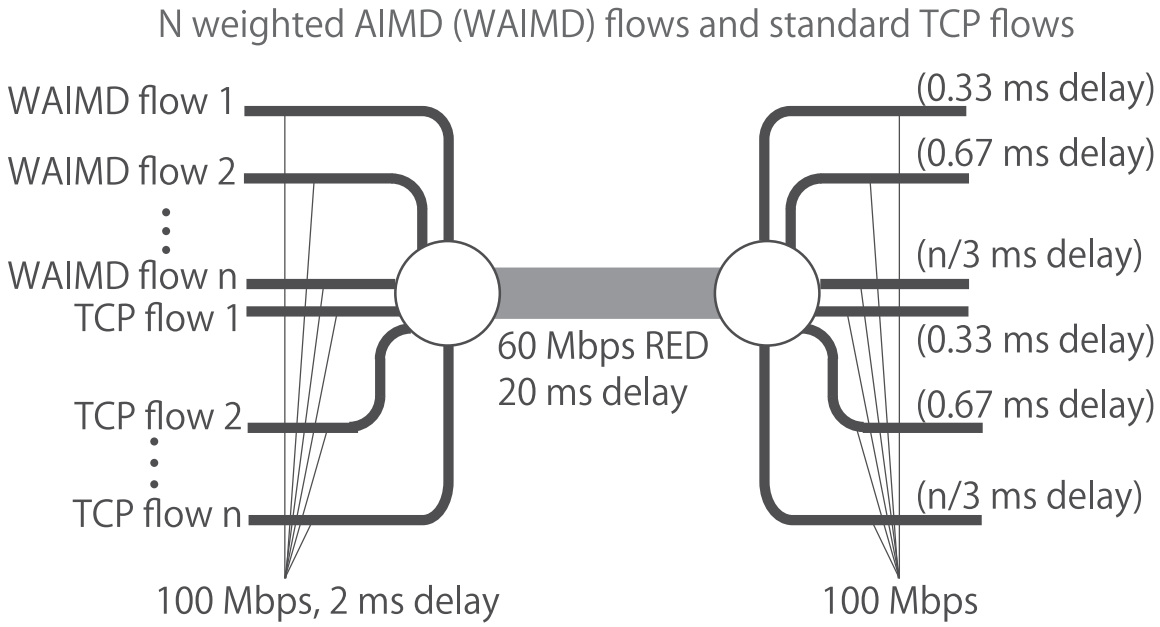

Figure 5.1: Simulation setup

both weighted AIMD and standard TCP flows. In the simulation, the weighted AIMD flows and the same number of standard TCP flows compete at the bottleneck.

Fig. 5.2 to Fig. 5.9 shows the simulation result that AIMD flows with the same weight parameter and standard TCP flows compete at the bottleneck. They plot throughput of each flow during 100 second simulation. The throughput is normalized so that value one equals to the bottleneck bandwidth capacity divided by the number of flows. The horizontal axis is the number of standard TCP flows and the weighted AIMD flows. The vertical axis is the normalized throughput of these flows. The dashed and solid lines show the average throughput of standard TCP flows and the weighted AIMD flows, respectively. We perform this simulation for the weight parameter 4/5, 3/4, 2/3, 1/2, 1/3, 1/4 and 1/5. According to the design of the aggressiveness manager, when the weight is D, D^2 and 1/2 are applied to the increase and the decrease parameters of the AIMD algorithm, respectively. For example, the AIMD flows with the weight parameter 2/3 follow $AIMD(4/9, 1/2)$. The number of the

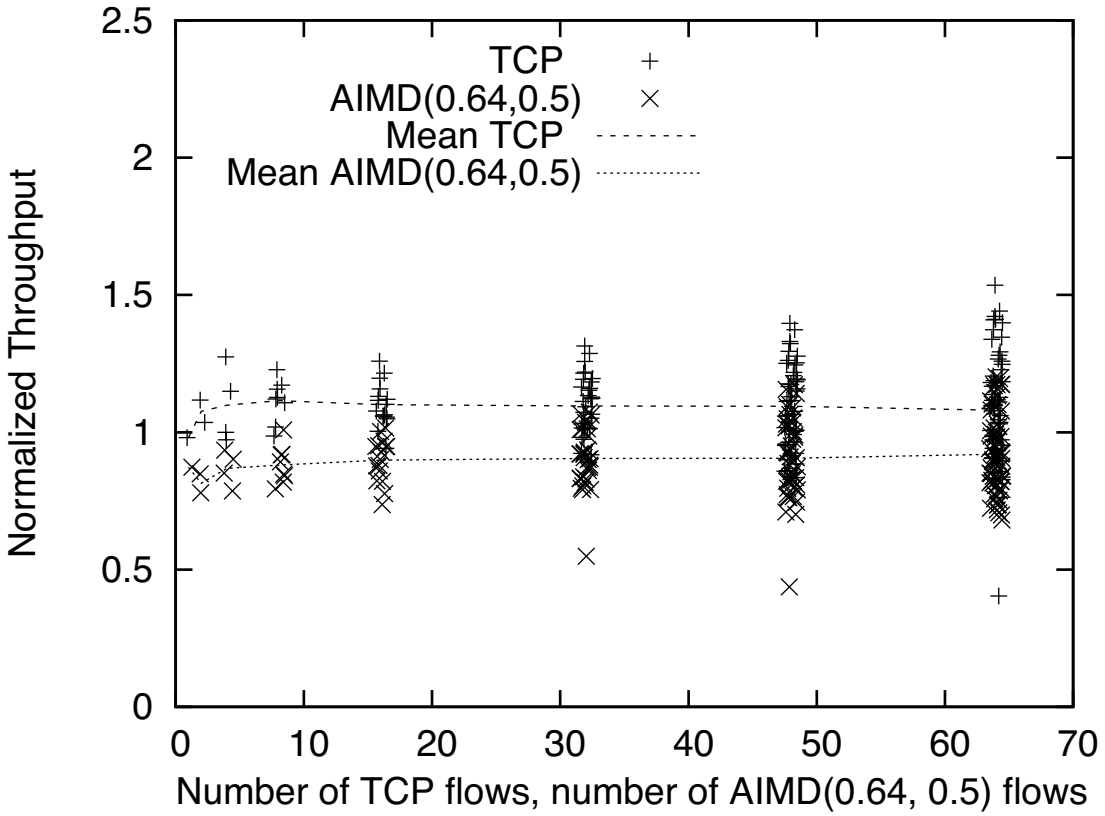

Figure 5.2: AIMD(16/25, 1/2) competing with TCP

weighted AIMD flows and standard TCP flows is 1, 2, 4, 8, 16, 32, 48 and 64. Fig. 5.10 summarizes the simulation results of Fig. 5.2 to Fig.5.9. In Fig. 5.10, the horizontal axis is the number of the weighted AIMD flows and standard TCP flows, and the vertical axis is the ratio of the throughput of weighted AIMD flows to standard TCP flows.

In various weight parameters and the number of the weighted AIMD and standard TCP flows, the weighted AIMD flows receive the throughput approximately in proportion to the weight parameter. When the number of flows is larger, hence the loss-event rate is higher, the weighted AIMD flows is a little more aggressive compared to the weight parameters. The possible reason is retransmission timeouts (RTOs) on the weighted AIMD flows. When the loss-event rate is higher, probability of RTO expiration becomes higher. Fig. 5.11 shows

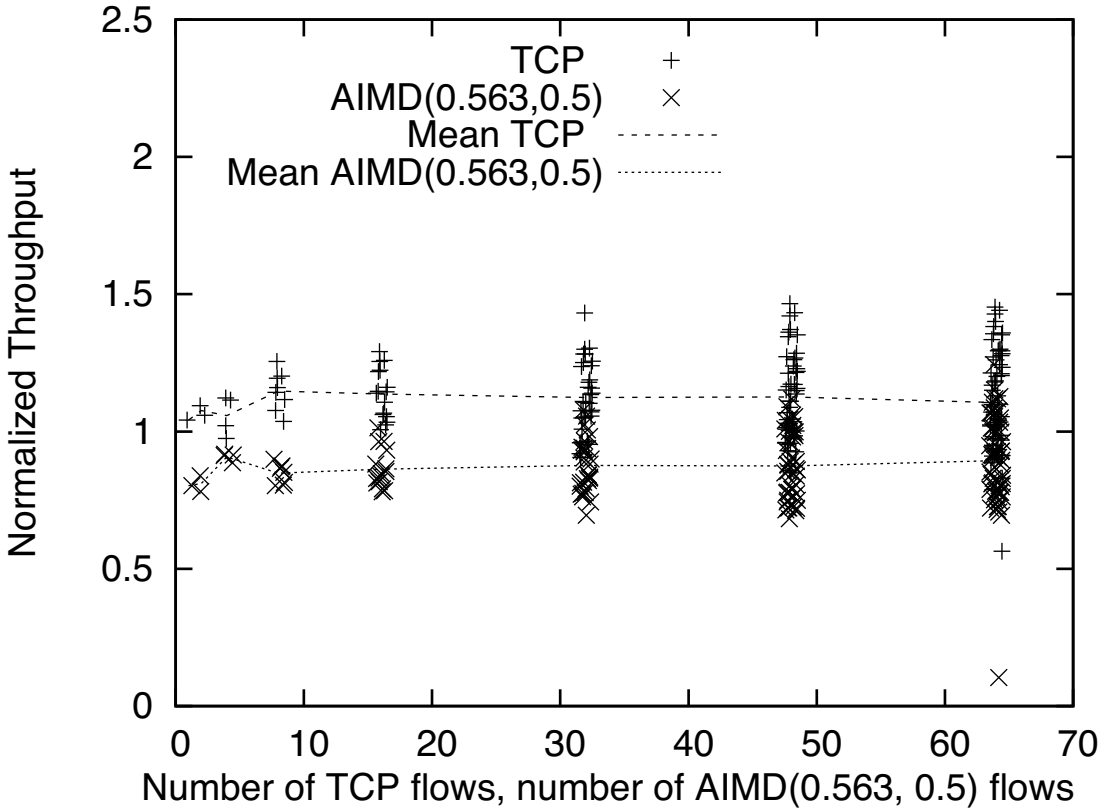

Figure 5.3: AIMD(9/16, 1/2) competing with TCP

the loss-event rate of each weight parameter and the number of flows. We do not change the behavior of the weighted AIMD flows on the retransmission timeout, such as timeout value and the slow-start behavior, because we focus on steady state of standard TCP and the weighted AIMD flows. Hence, duration for the timeout and the behavior in the slow-start phase could lead to these results. In addition, when the number of the weighted AIMD flows and standard TCP flows are 1, 2 and 4, the throughput is a little off the ideal throughput proportion. We consider that the extremely low loss-event rate affects this behavior. When the number of the weighted AIMD flows and standard TCP flows are 1, 2 and 4, the loss-event rate is 0.126 - 0.133 %, 0.126 - 0.135 % and 0.143 - 0.170 %, respectively. The number of loss events on each flow could affect this result, because when the loss-event rate is lower,

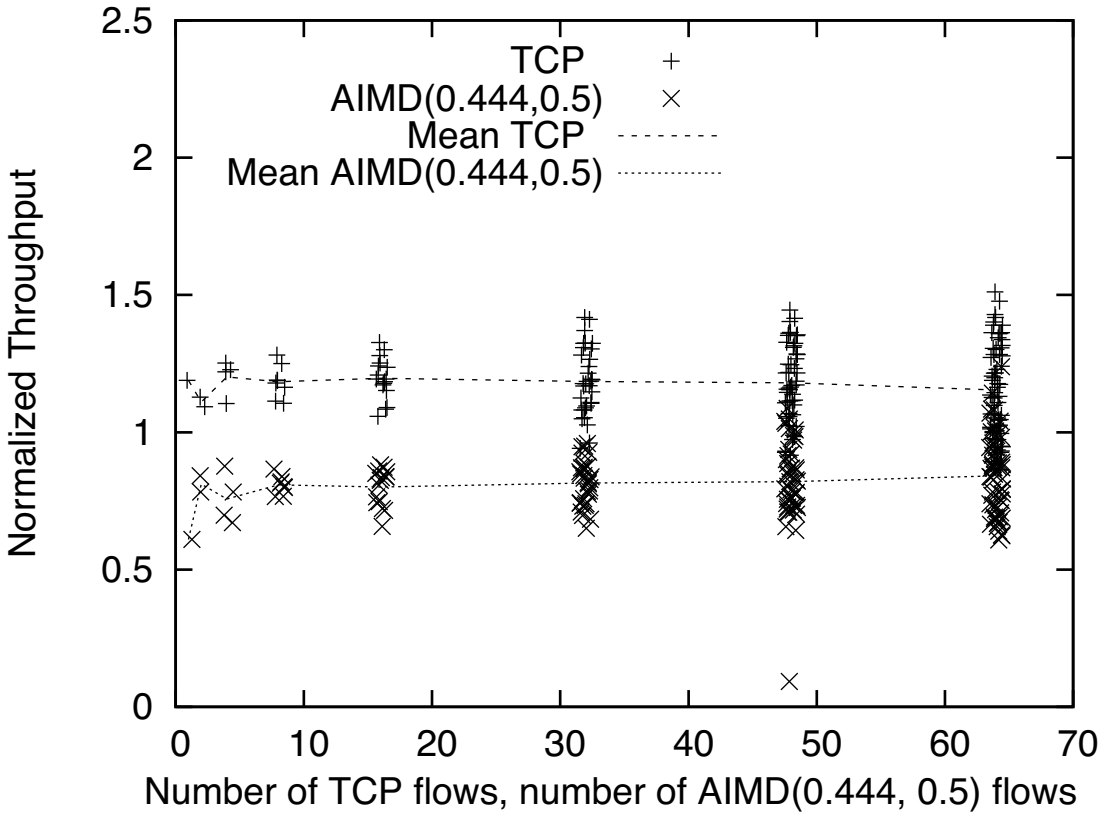

Figure 5.4: AIMD(4/9, 1/2) competing with TCP

the influence to the throughput caused by one loss event is larger.

5.2 A Comparison of Differently Weighted AIMD Flows and TCP Flows

Fig. 5.12 to Fig. 5.17 plots the normalized throughput of the weighted AIMD flows and standard TCP flows. In Fig. 5.12 to Fig. 5.15, when the half of the weighted AIMD flows have the weight parameter D, the other weighed AIMD flows have the weight parameter $1 - D$. The simulation setup is same as the Sec. 5.1. Hence, N TCP flows, $N/2$ AIMD(D^2, 1/2) flows and $N/2$ AIMD($(1 - D)^2, 1/2$) flows compete at the 60 Mbps bottleneck. In Fig. 5.12 to Fig. 5.15, we use the combination of the weight parameters 3/5 and 2/5, 2/3

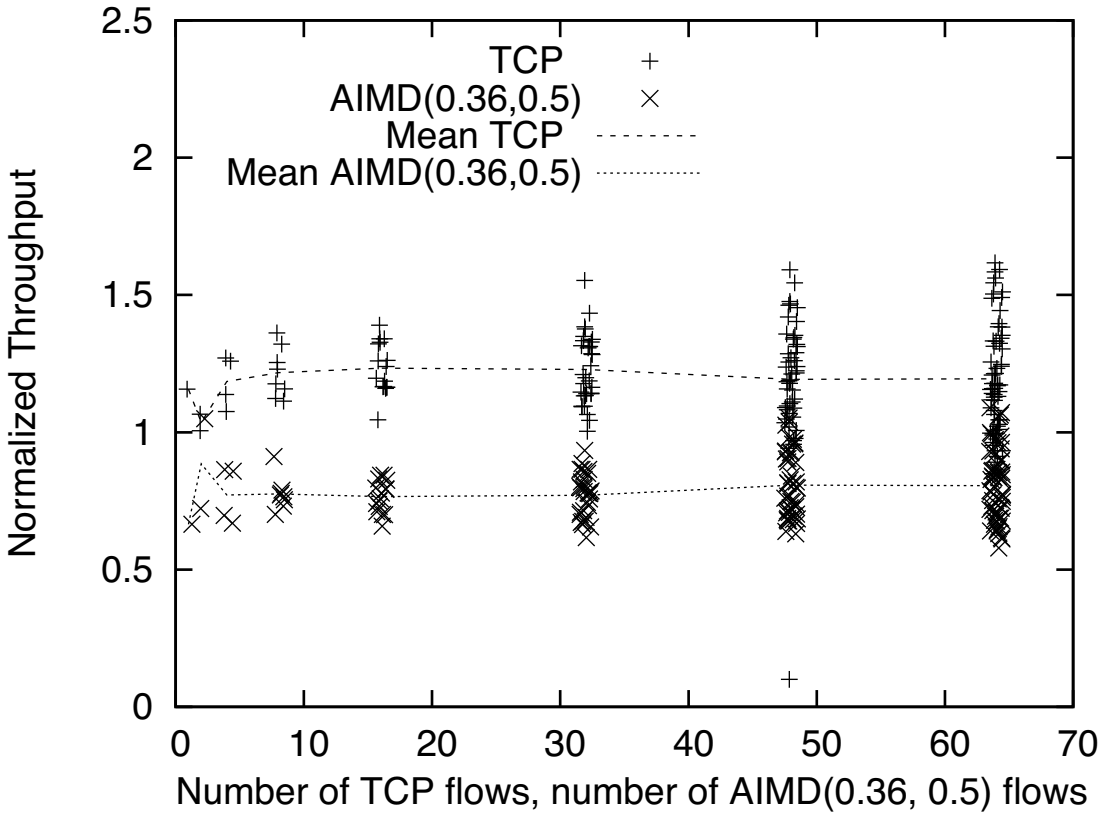

Figure 5.5: AIMD(9/25, 1/2) competing with TCP

and 1/3, 3/4 and 1/4, 4/5 and 1/5. The number of the weighted AIMD flows and standard TCP flows is 2, 4, 8, 16, 32, 48 and 64.

In Fig. 5.16, $N/4$ weighted AIMD flows with the weight parameter 3/18, $N/4$ weighted AIMD flows with the weight parameter 4/18, $N/4$ weighted AIMD flows with the weighted parameter 5/18 and $N/4$ weighted AIMD flows with the weight parameter 5/18 compete with N standard TCP flows in the simulation setup of Fig. 5.1. In Fig. 5.17, $N/4$ weighted AIMD flows with the weight parameter 1/10, and $3N/4$ weighted AIMD flows with the weight parameter 3/10 compete with N standard TCP flows at the 60 Mbps bottleneck.

In Fig. 5.12 to 5.15, when the mean throughput of all weighted AIMD flows is half of standard TCP flows, it means that the bundle of the two weighted AIMD flows achieves

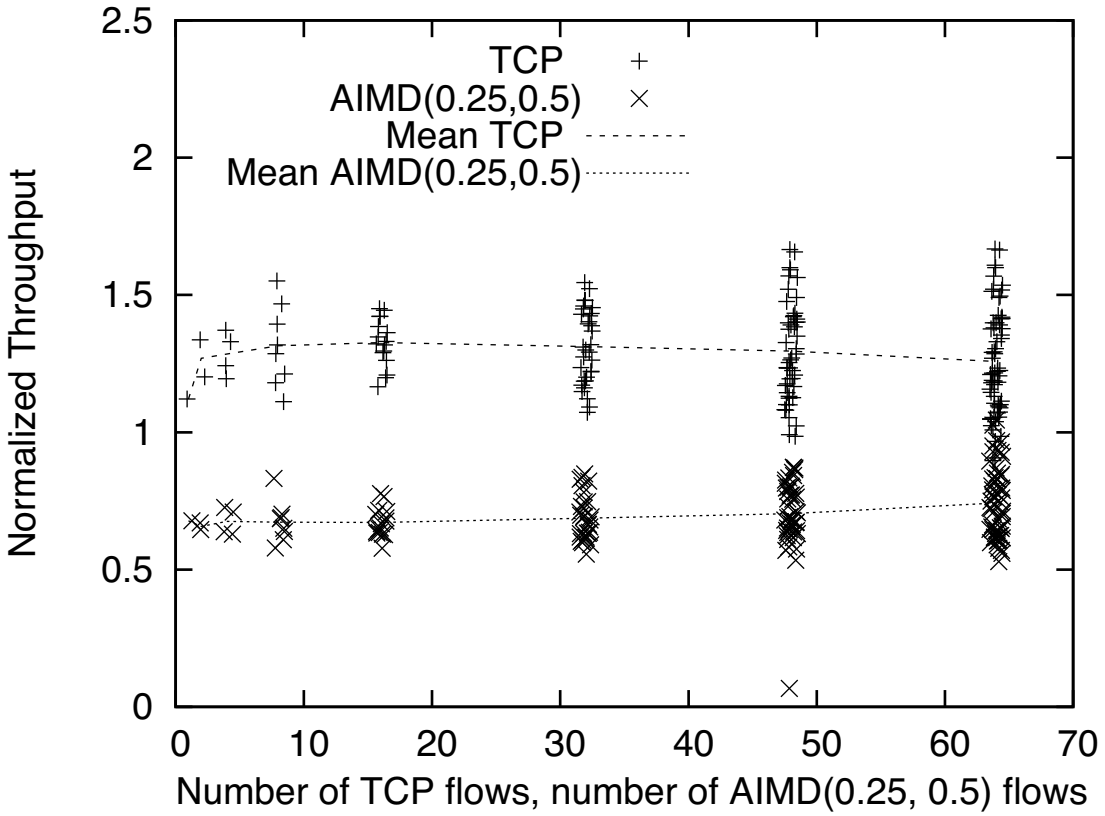

Figure 5.6: AIMD(1/4, 1/2) competing with TCP

equal throughput with that of TCP. In Fig. 5.16 and 5.17, when the mean throughput of all weighted AIMD flows is quarter of standard TCP flows, it means that the bundle of the four weighted AIMD flows achieves equal throughput with that of TCP. Fig. 5.18 summarizes the results of Fig. 5.12 to Fig. 5.17. This shows the throughput ratio of weighted AIMD flows to the standard TCP flows, in each combination of the weight parameters and the number of the weighted AIMD and standard TCP flows. The throughput ratio is per weighted AIMD flow, hence 0.5 or 0.25 is the ideal ratio.

The average throughput of the weighted AIMD flows with different weight parameters is approximately 1/2 or 1/4 times standard TCP throughput, regardless of the combination of the weighted parameters. Hence, a bundle of the weighted AIMD flows with different

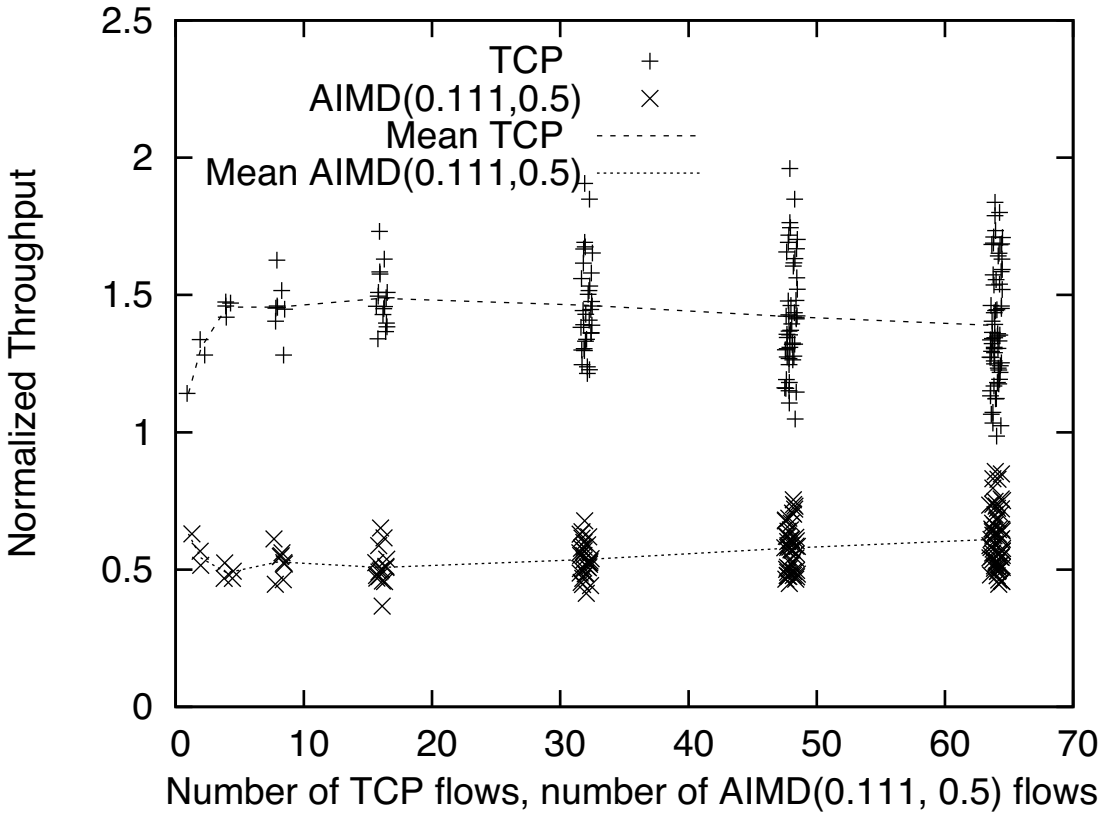

Figure 5.7: AIMD(1/9, 1/2) competing with TCP

weight parameters of which the sum is 1 receives approximately equal throughput with that of standard TCP flows. Similarly to the former simulations, the average throughput of the weighted AIMD flows becomes higher when there are more number of flows. It could be due to the behavior on RTO expirations and the slow-start phase. When the number of the weighted AIMD flows and standard TCP flows is 64, the difference of throughput proportion between ideal one and actual one is approximately 20 %. In addition, when the weight parameters are 2/3 and 1/3, and the number of flows is 4 in Fig. 5.18, average throughput of the weighted AIMD flows is approximately 25 % higher than the ideal average throughput. This could be affected by the extremely lower loss-event rate, similarly to the previous experiments.

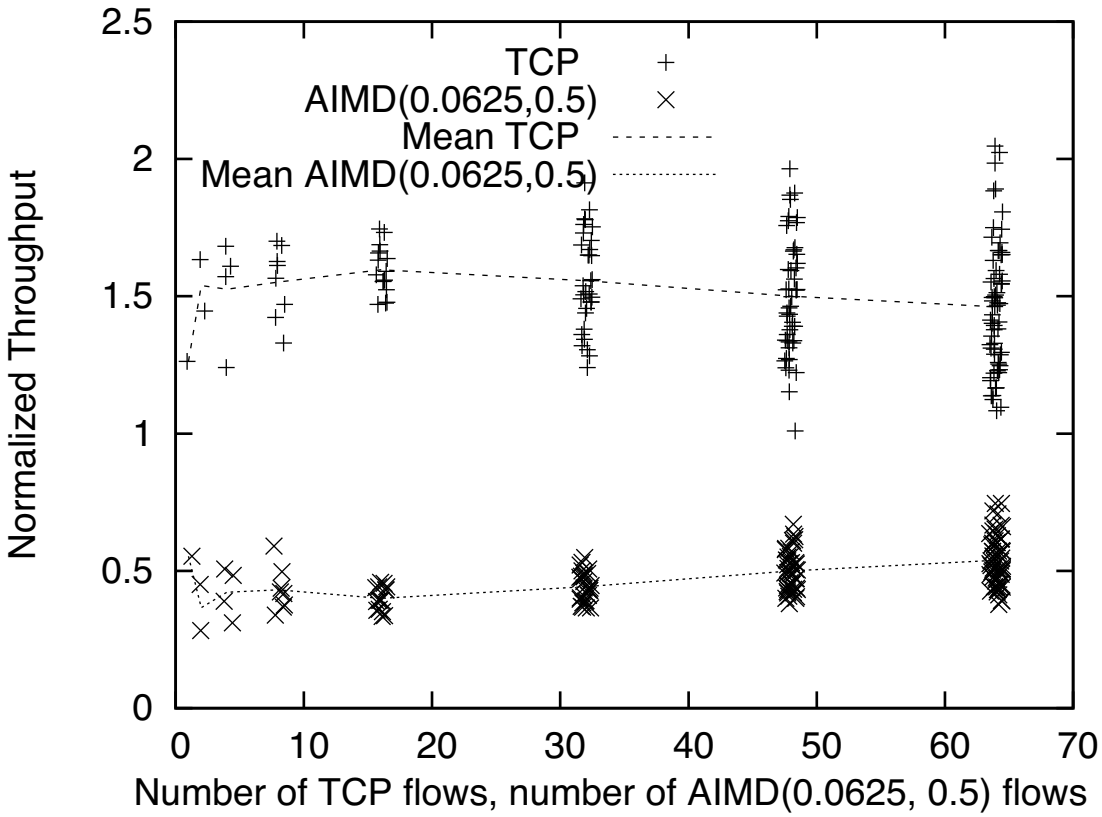

Figure 5.8: AIMD(1/16, 1/2) competing with TCP

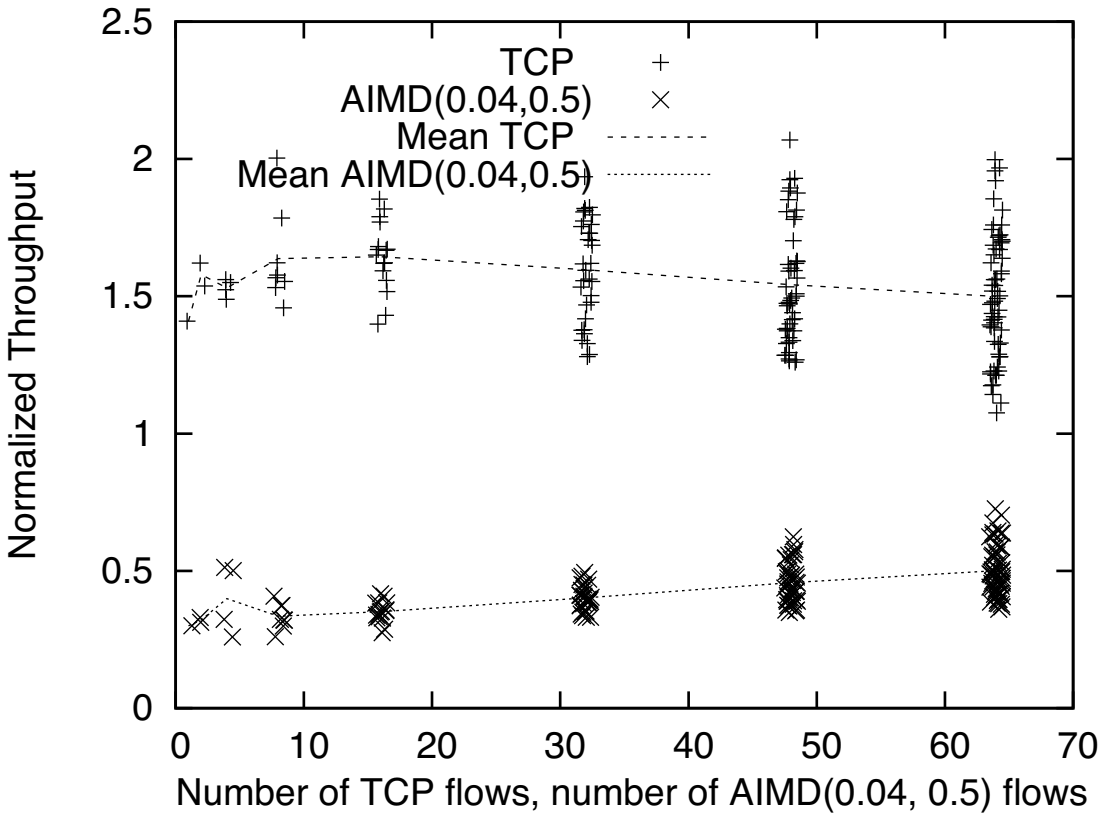

Figure 5.9: AIMD(1/25, 1/2) competing with TCP

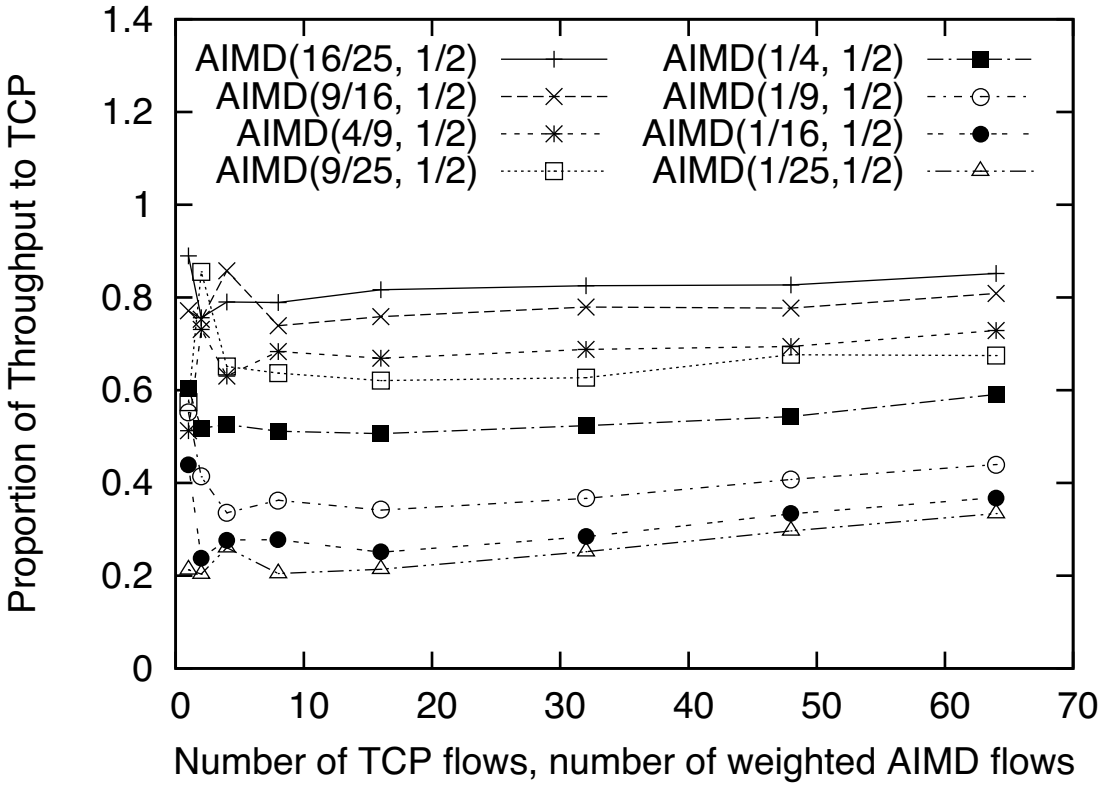

Figure 5.10: Throughput ratio of the weighted AIMD flows to TCP flows

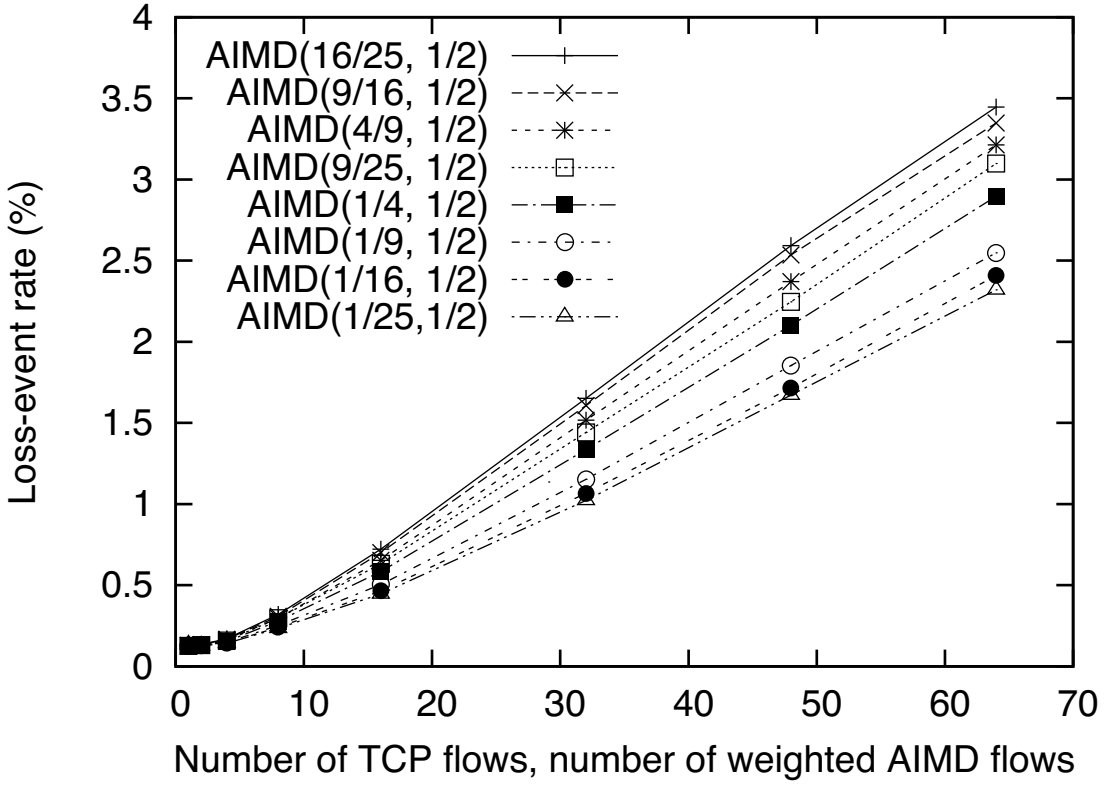

Figure 5.11: Loss-event rate of weighted AIMD flows in Fig. 5.2 - Fig. 5.9

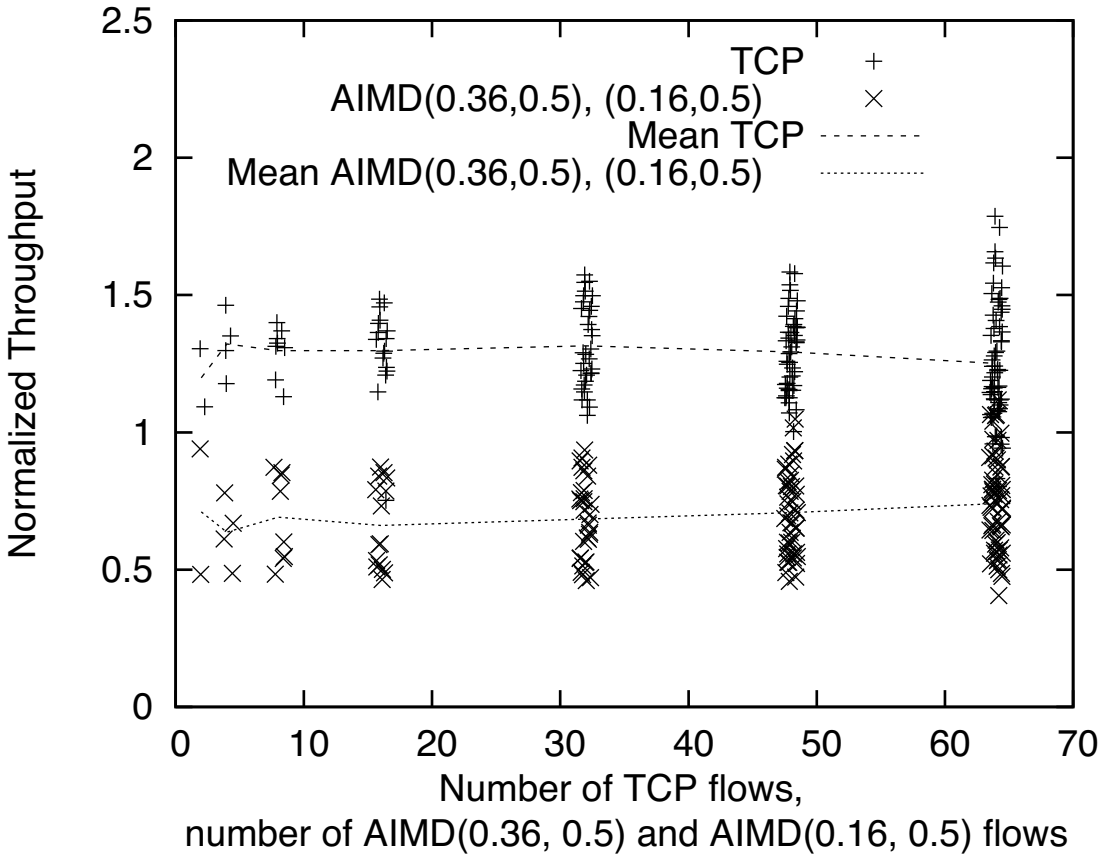

Figure 5.12: AIMD(9/25, 1/2) and AIMD(4/25, 1/2) with TCP

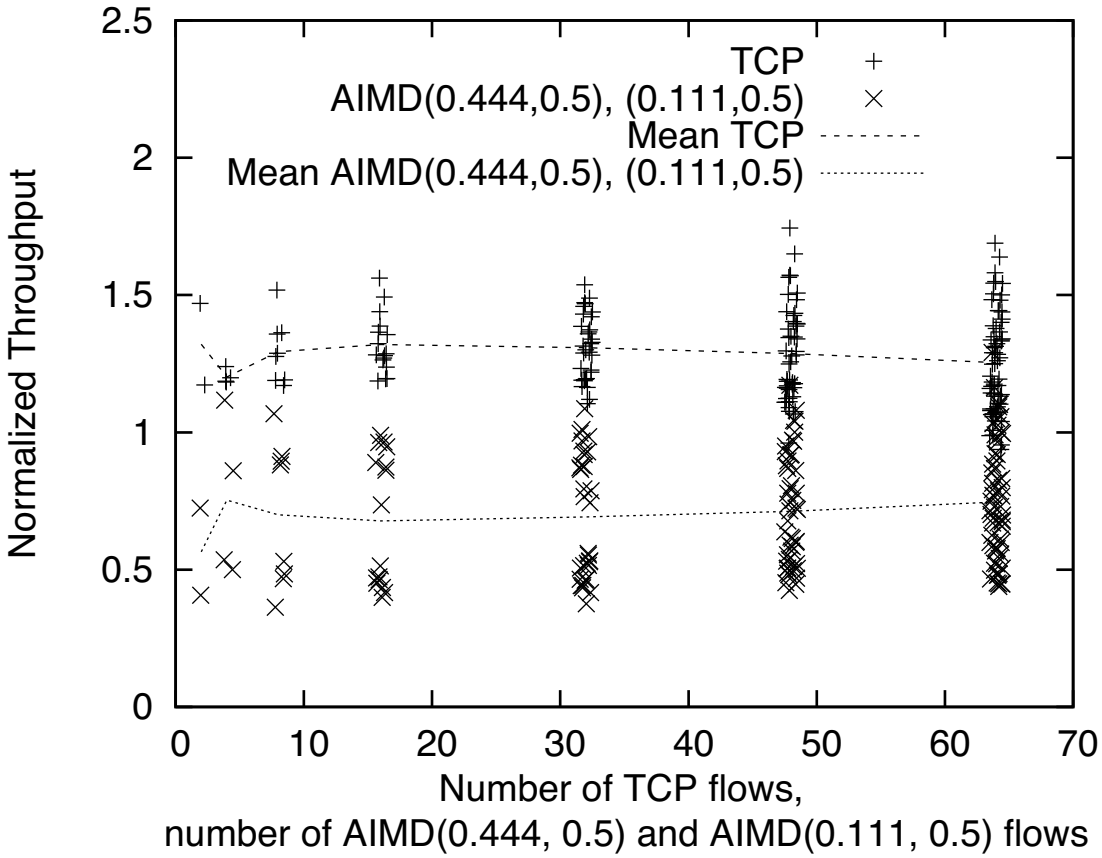

Figure 5.13: AIMD(4/9, 1/2) and AIMD(1/9, 1/2) with TCP

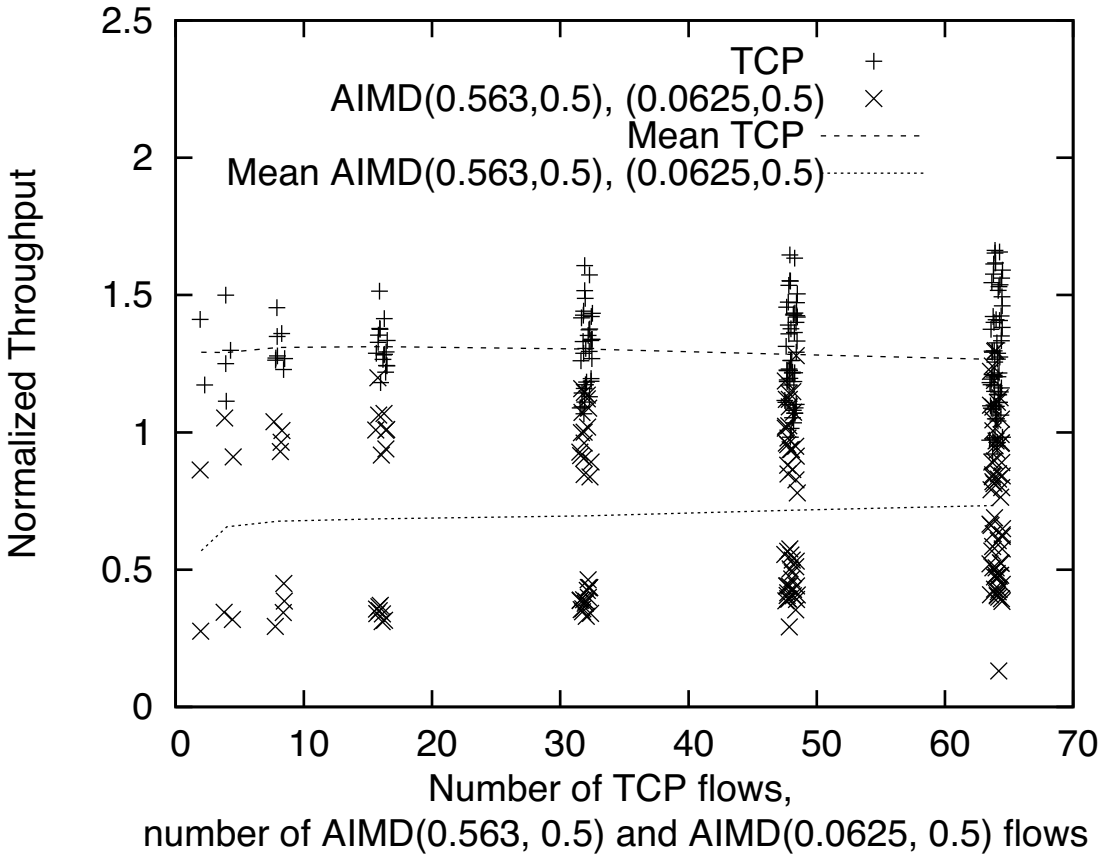

Figure 5.14: AIMD(9/16, 1/2) and AIMD(1/16, 1/2) with TCP

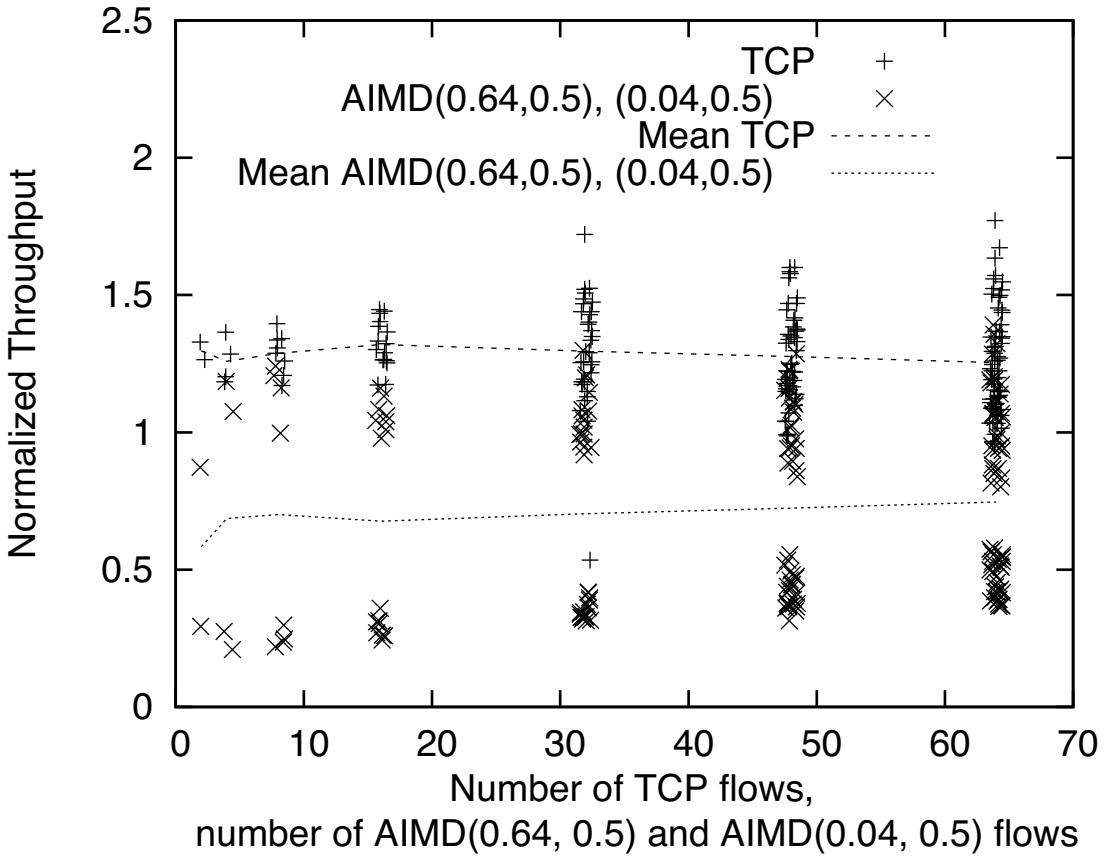

Figure 5.15: AIMD(16/25, 1/2) and AIMD(1/25, 1/2) with TCP

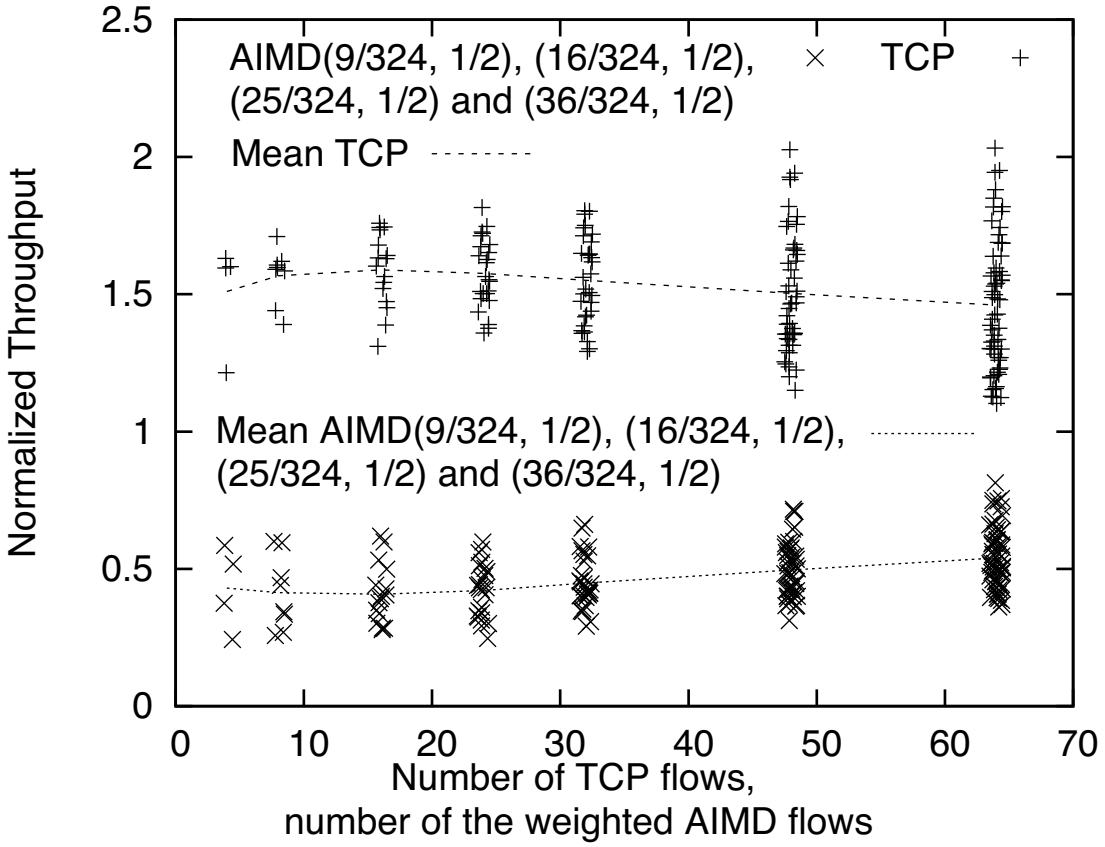

Figure 5.16: AIMD(9/324, 1/2), (16/324, 1/2), (25/324, 1/2) and (36/324, 1/2) with TCP

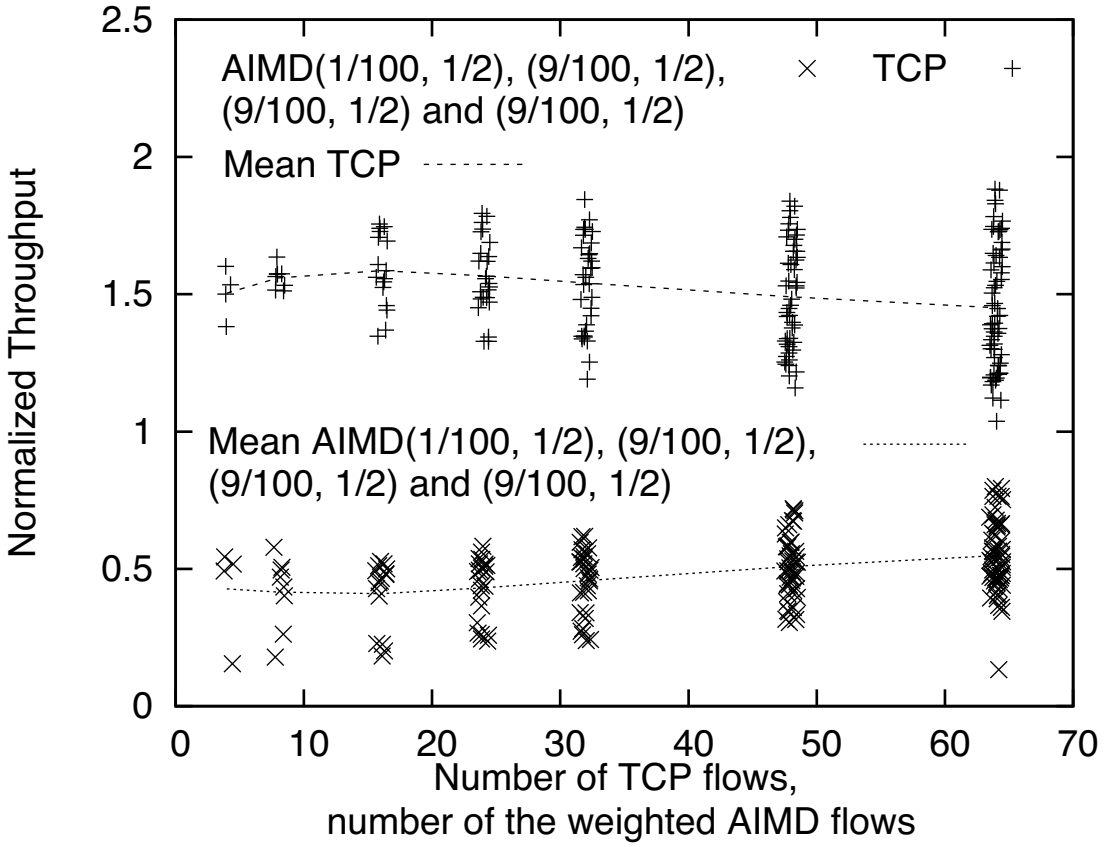

Figure 5.17: AIMD(1/100, 1/2), (9/100, 1/2), (9/100, 1/2) and (9/100, 1/2) with TCP

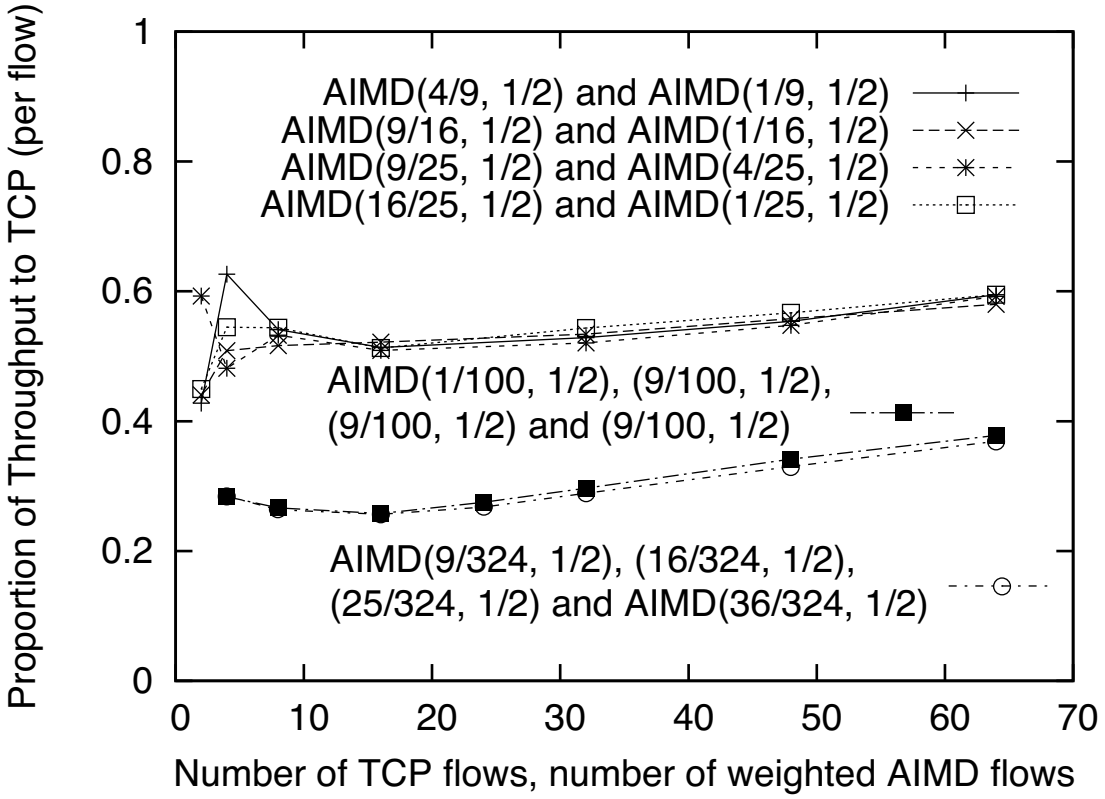

Figure 5.18: Throughput ratio of the weighted AIMD flows with different parameters

Chapter 6

Related Work

This chapter describes prior and related work of this research. Transport protocols for multipath utilization, integrated congestion control for multiple flows are discussed in this chapter.

6.1 Multipath Transport Protocol

pTCP [14], mTCP [41], AMS [34], ChTCP [6], CMT [15] and R-MTP [22] propose transport protocols that simultaneously utilize multiple paths between end-to-end. pTCP, mTCP, AMS and ChTCP extend TCP. CMT extends SCTP. R-MTP is a new transport protocol.

pTCP, CMT and ChTCP purely perform TCP congestion control per subflow. Hence, whenever subflows traverse the same bottleneck, they take an unfair share. AMS uses paths where both source and destination addresses are different, to avoid over-aggressiveness at the shared bottleneck. However, the rationale that such paths are completely disjoint is not discussed. mTCP performs TCP congestion control for each subflow, however, mTCP implements a mechanism that detects shared bottlenecks [33] to avoid the use of multiple subflows on the same bottleneck. When mTCP detects a shared bottleneck, it suppresses subflows traversing the same bottleneck. However, when the spare bandwidth of one link on the remaining path is less than the TCP-friendly sending rate, mTCP cannot utilize the shared link effectively. In this case, equal bandwidth with each of background flows at the shared link cannot be achieved by only one subflow, as described in Sec. 3.2. Hence, the utilization property cannot be satisfied. In addition, according to mTCP, it takes maximum 15 seconds to detect the shared bottleneck. This response in the congestion collapse is later compared to that of TCP. Thus, this approach is not efficient. R-MTP implements rate control based on bandwidth estimation, which uses packet inter-arrival times and jitter to detect congestion. However, R-MTP does not discuss how each subflow and the bundle of subflows compete with TCP-friendly flows.

[17, 13] model a resource-allocation architecture enabling multipath routing based on the fluid-flow model. They at the same time propose a TCP extension based on Scalable TCP [18] for their network models. Unlike our approach, their models and TCP extensions assume

congestion feedback from routers. Hence, their congestion control is not achieved by end-node functionalities only. In addition, they do not analyze how their schemes work with existing TCP-friendly flows utilizing a single path.

6.2 Aggregate Congestion Control

[5, 21, 29, 35] introduce congestion control that receives multiple flowshares following weighted proportional fairness. The main concept is to emulate the behavior of multiple TCP connections between aggregate points that integrate flows transmitted by multiple users. They achieve multiple flowshares by assigning weight to the TCP's AIMD algorithm. Although their goal is multiple flowshares at the bottleneck on a single path, the concept that assigns weight to the AIMD algorithm is applied to the aggressiveness manager of BMC. While they emulate more than one TCP flows, BMC emulates a proportionally less aggressive TCP flow for each subflow.

E-TCP [7] aggregates congestion control of multiple TCP connections to avoid over-aggressiveness of applications that utilize parallel TCP connections on the same path. In this scheme, TCP connections between the same source-destination pair share one congestion window, which is affected by transmission of any connections sharing it. Thus, it results in a single flowshare regardless of the number of TCP connections between a source-destination pair. CM [2] integrates congestion control of all TCP and UDP flows that traverse the same path. Similarly to E-TCP, each integrated congestion control entity manages one congestion window, which results in a single flowshare of the bundle of flows at the bottleneck. BMC achieves a single flowshare at the shared bottleneck regardless of the number of subflows or paths in the multipath connection. Therefore, the concept that integrates congestion control of multiple flows for a single flowshare is similar to the fairness property of BMC. On the other hand, while E-TCP achieves it with congestion window sharing, BMC achieves it with

weighted congestion control for different characteristic paths.

Chapter 7

Conclusion and Future Work

This chapter concludes this paper, and describes our future work.

In this paper, we presented a congestion control scheme for multipath transport protocols. The key idea is bidimensional probe. It probes the available sending rate with the same aggressiveness as TCP. This results in fair bandwidth share with competing TCP-friendly flows at the shared bottleneck between multiple paths. Each of subflow is proportionally less aggressive than a TCP-friendly flow based on the weighted AIMD algorithm. The bundle of subflows is as aggressive as a TCP flow. BMC also probes the optimal combination of weight between subflows. It maximizes utilization of the shared bottleneck link, because it effectively utilizes spare bandwidth of distinct paths. Therefore, it maximizes the throughput of the whole multipath connection. Our simulation results showed that the weighted AIMD algorithm of which the weight is less than one can receive throughput approximately in proportion to standard TCP flows. In addition, they showed that a bundle of the weighted AIMD flows can receive approximately equal throughput with a standard TCP flow when the sum of the weight parameters is one.

Our future work explores the slow-start behavior and RTT bias of BMC, in addition to evaluation of the proportion manager. As shown in our simulation results, subflow behavior on the RTO expiration affects the fairness of BMC, especially when the loss-event rate is higher. In addition, the proportion manager currently determines the weight parameter of each subflow based on the value which has deducted the effect of RTT and the weight from the measured throughput. If the proportion manager allocates the larger weight for short-RTT subflows so that the desired throughput does not exceed the spare bandwidth of the disjoint links, BMC will achieve higher throughput reasonably. At the same time, investigating the principle of TCP friendliness in multipath congestion control is interesting, for example, how the multipath connection should receive RTT bias on multiple subflows. We are also designing a multipath transport protocol, because existing proposals are not practical to the current Internet. For example, none of these considers middlebox transparency such as NAT

boxes and firewalls.

According to [32], when the sending rates of flows are significantly different, they experiences different loss-event rates. The higher sending rate is, the lower loss-event rate becomes. Since each subflow in a multipath connection obeying BMC is less aggressive than TCP, it might give higher loss-event rate on subflows. Hence, a multipath connection might experience lower throughput than competing TCP flows at the shared bottleneck in some cases. We explore the impact to throughput of BMC with further experiments and analysis. In addition, in this paper, we investigated weighted AIMD-based design of BMC. On the other hand, it is interesting to use other congestion control variants (e.g., CTCP [38], Westwood [24]) for BMC. It will result in adaptation to more various scenarios, such as high-speed and long-distance networks and lossy wireless networks. We will investigate to weight these algorithms for TCP-friendly throughput management.

Acknowledgment

I gratefully acknowledge my adviser, Professor Hideyuki Tokuda, for his professional advice, guidance and encouragement. I would like to thank professors, associate professors and assistant professors in Keio University, Jun Murai, Osamu Nakamura, Hiroyuki Kusumoto, Kazunori Takashio, Noriyuki Shigechika, Rodney D. Van Meter Ⅲ, Keisuke Uehara, Jin Mitsugi and Jin Nakazawa. I particularly thank Dr. Yoshifumi Nishida, who helped design the algorithm, carefully reviewed the thesis and discussed about the presentation for a long period; Dr. Lars Eggert for discussions of the principle, his technical comments and helping to improve the quality of this thesis; Dr. Pasi Sarolahti for valuable discussions, his insightful comments and suggestions of future work. I am grateful to Dr. Hideaki Imaizumi and Dr. Jin Nakazawa and Dr. Ryuji Wakikawa for their invaluable encouragement. I also thank Vamsi Kambhampati, Elena Balandina, Stephen Strowes, Michiko Nitta, Jun-ichi Yura, Yusuke Kawakita, Shoko Mikawa, Tomohiro Ishihara, Masato Saito, Hitomi Takahashi, Hiroshi Sakakibara, Takuro Yonezawa, Noriatsu Kudo, Katsuhiro Horiba, Tsuyoshi Hisamatsu, Satomi Fujimaki, Masahiro Kozuka, Takashi Tomine, Hikoichiro Nakai, Mizuki Kawazoe, Masayoshi Mizutani, Yuki Oyabu, Takesi Matsuya, Yohei Kuga, Akira Kanai, Yuichi Nakamura, Yusuke Okumura, Masato Mori, Takahiro Nozawa, Kenji Yonekawa, Masashi Horiguchi and Kaori Kan for their daily support and encouragement. Finally, I acknowledge the member of Hide Tokuda laboratory, WIDE project and Future Internet Team at Nokia Research Center.

Bibliography

[1] M. Allman, V. Paxson, and W. Stevens. TCP Congestion Control. *RFC 2581*, Oct. 1999.

[2] H. Balakrishnan, H. Rahul, and S. Seshan. An integrated congestion management architecture for internet hosts. In *Proc. ACM SIGCOMM'99*, pages 175–187, Sep. 1999.

[3] B. Briscoe. Flow rate fairness: dismantling a religion. *ACM Computer Communication Review*, 37(2):63–74, 2007.

[4] R. Chakravorty, S. Katti, J. Crowcroft, and I. Pratt. Flow aggregation for enhanced tcp over wide-area wireless. In *Proc. IEEE INFOCOM 2003*, pages 1754–1764, Mar. 2003.

[5] J. Crowcroft and P. Oechslin. Differentiated end-to-end Internet services using a weighted proportional fair sharing TCP. *ACM Computer Communication Review*, 28(3):53–69, Jul 1998.

[6] Y. Dong, N. Pissinou, and J. Wang. Concurrency Handling in TCP. In *Proc. CNSR '07*, pages 255–262, May 2007.

[7] L. Eggert, J. Heidemann, and J. Touch. Effects of ensemble-TCP. *ACM Computer Communication Review*, 30(1):15–29, Jan. 2000.

[8] S. Floyd. Connections with multiple congested gateways in packet-switched networks part 1: one-way traffic. *ACM Computer Communication Review*, 21(5):30–47, 1991.

[9] S. Floyd and K. Fall. Promoting the use of end-to-end congestion control in the Internet. *IEEE/ACM Trans. on Networking*, 7(4):458–472, 1999.

[10] S. Floyd, M. Handley, and J. Padhye. A Comparison of Equation-Based and AIMD Congestion Control. URL http://www.icir.org/tfrc/, May 2000.

[11] S. Floyd and V. Jacobson. Random early detection gateways for congestion avoidance. *IEEE/ACM Trans. on Networking*, 1(4):397–413, 1993.

[12] K. Gummadi, H. Madhyastha, S. Gribble, H. Levy, and D. Wetherall. Improving the reliability of internet paths with one-hop source routing. In *Proc. USENIX OSDI 2004*, 6, Dec. 2004.

[13] H. Han, S. Shakkottai, C. V. Hollot, R. Srikant, and D. Towsley. Multi-path TCP: a joint congestion control and routing scheme to exploit path diversity in the internet. *IEEE/ACM Trans. on Networking*, 14(6):1260–1271, 2006.

[14] H.-Y. Hsieh and R. Sivakumar. A transport layer approach for achieving aggregate bandwidths on multi-homed mobile hosts. In *Proc. ACM MobiCom 2002*, pages 83–94, Sep. 2002.

[15] J. R. Iyengar, P. D. Amer, and R. Stewart. Concurrent multipath transfer using SCTP multihoming over independent end-to-end paths. *IEEE/ACM Trans. on Networking*, 14(5):951–964, 2006.

[16] W. John and S. Tafvelin. Analysis of internet backbone traffic and header anomalies observed. In *Proc. ACM IMC '07*, pages 111–116, Oct. 2007.

[17] F. P. Kelly and T. Voice. Stability of end-to-end algorithms for joint routing and rate control. *ACM Computer Communication Review*, 35(2):5–12, Apr. 2005.

[18] T. Kelly. Scalable TCP: improving performance in highspeed wide area networks. *ACM Computer Communication Review*, 33(2):83–91, Apr. 2003.

[19] P. Key, L. Massoulie, and D. Towsley. Combining Multipath Routing and Congestion Control for Robustness. In *Proc. CISS 2006*, pages 345–350, Mar. 2006.

[20] E. Kohler, M. Handley, and S. Floyd. Designing DCCP: congestion control without reliability. In *Proc. ACM SIGCOMM '06*, pages 27–38, 2006.

[21] F. Kuo and X. Fu. Probe-Aided MulTCP: an aggregate congestion control mechanism. *ACM Computer Communication Review*, 38(1):17–28, Jan 2008.

[22] L. Magalhaes and R. Kravets. Transport Level Mechanisms for Bandwidth Aggregation on Mobile Hosts. In *Proc. ICNP 2001*, page 0165, 2001.

[23] T. V. Lakshman and U. Madhow. The performance of tcp/ip for networks with high bandwidth-delay products and random loss. *IEEE/ACM Trans. on Networking*, 5(3):336–350, 1997.

[24] S. Mascolo, C. Casetti, M. Gerla, M. Y. Sanadidi, and R. Wang. TCP westwood: Bandwidth estimation for enhanced transport over wireless links. In *Proc. ACM MobiCom 2001*, pages 287–297, 2001.

[25] M. Mathis, J. Semke, J. Mahdavi, and T. Ott. The macroscopic behavior of the TCP congestion avoidance algorithm. *ACM Computer Communication Review*, 27(3):67–82, Jul 1997.

[26] E. Nordmark and M. Bagnulo. Shim6: Level 3 Multihoming Shim Protocol for IPv6. *Internet Draft*, Dec. 2008.

[27] The network simulator - ns-2. URL http://www.isi.edu/nsnam/ns/.

[28] D. Ott and K. Mayer-Patel. An open architecture for transport-level protocol coordination in distributed multimedia applications. *ACM Trans. Multimedia Comput. Commun. Appl.*, 3(3):17, 2007.

[29] D. Ott, T. Sparks, and K. Mayer-Patel. Aggregate congestion control for distributed multimedia applications. In *Proc. IEEE INFOCOM 2004*, pages 13–23, Mar. 2004.

[30] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose. Modeling TCP throughput: a simple model and its empirical validation. In *Proc. ACM SIGCOMM '98*, pages 303–314, Aug. 1998.

[31] J. Postel. Transmission Control Protocol. *RFC 793*, Sep. 1981.

[32] I. Rhee and L. Xu. Limitations of equation-based congestion control. *IEEE/ACM Trans. on Networking*, 15(4):852–865, 2007.

[33] D. Rubenstein, J. Kurose, and D. Towsley. Detecting shared congestion of flows via end-to-end measurement. *IEEE/ACM Trans. on Networking*, 10(3):381–395, 2002.

[34] S. Saito, Y. Tanaka, M. Kunishi, Y. Nishida, and F. Teraoka. AMS: An Adaptive TCP Bandwidth Aggregation Mechanism for Multi-homed Mobile Hosts. *IEICE Trans. on Information and Systems*, 89:2838–2847, 2006.

[35] M. Singh, P. Pradhan, and P. Francis. MPAT: aggregate TCP congestion management as a building block for Internet QoS. In *Proc. ICNP 2004*, pages 129–138, Oct. 2004.

[36] R. Stewart. Stream Control Transmission Protocol. *RFC 4960*, Sep. 2007.

[37] M. Sullivan. Surveys: Internet Traffic Touched by YouTube. URL http://www.lightreading.com/, Jan. 2007.

[38] K. Tan, J. Song, Q. Zhang, and M. Sridharan. A compound tcp approach for high-speed and long distance networks. In *Proc. IEEE INFOCOM 2006*, pages 1–12, Apr. 2006.

[39] K. Thompson, G. J. Miller, and R. Wilder. Wide-Area Internet Traffic Patterns and Characteristics. *IEEE Network*, pages 10–23, Nov. 1997.

[40] YouTube. http://www.youtube.com.

[41] M. Zhang, J. Lai, A. Krishnamurthy, L. Peterson, and R. Wang. A transport layer approach for improving end-to-end performance and robustness using redundant paths. In *Proc. USENIX 2004 Annual Technical Conference*, pages 99–112, 2004.