Journal of Network and Computer Applications 47 (2015) 11–22

Contents lists available at ScienceDirect

Journal of Network and Computer Applications

journal homepage: www.elsevier.com/locate/jnca

On-demand scheduling for concurrent multipath transfer using the stream control transmission protocol ☆

T. Daniel Wallace [a,1], Khalim Amjad Meerja [b], Abdallah Shami [b]

[a] Auvik Networks Inc., Waterloo, Ontario, Canada
[b] Department of Electrical and Computer Engineering, The University of Western Ontario, London, Ontario, Canada

ARTICLE INFO

Article history:
Received 25 March 2014
Received in revised form
19 August 2014
Accepted 8 September 2014
Available online 28 September 2014

Keywords:
Multihoming
Stream control transmission protocol
Concurrent multipath transfer

ABSTRACT

Disparate path characteristics and a constrained receive buffer can frustrate the promised performance gains of concurrent multipath transfer (CMT). Known as the *receive buffer blocking* problem, out-of-order arrivals disrupt the sender's ability to transmit new packets while buffering delays ensue at the multihomed receiver. To mitigate this effect, earlier work used scheduling algorithms that assigned new packets to the receiver's destination address with the lowest delay. Unfortunately, this approach is not always successful; and in some cases, even a simpler method will achieve better results.

Our research suggests that congestion and flow control—standard elements of the stream control transmission protocol (SCTP)—counteract the scheduling process. Since congestion and flow control dictate when packets are actually transmitted to a destination address, making a scheduling decision prior to a transmission opportunity can be ineffective.

In this paper, we propose an on-demand scheduler (ODS); a scheduling approach for CMT that waits for a transmission opportunity before assigning a packet to one of the receiver's destination addresses. In some circumstances, however, ODS will enable one destination address to monopolize shared resources, such as the receive buffer (RBUF). To circumvent this issue we have also developed a new congestion window update policy for CMT. When compared to previous scheduling algorithms, ODS and our new update policy can significantly improve throughput for CMT under delay and bandwidth-based disparity.

1. Introduction

Known as *smartphones*, multihomed devices like the iPhone and BlackBerry can simultaneously connect to Wi-Fi and 4G LTE networks. Unfortunately, due to the architectural constraints of standard transport layer protocols like the transmission control protocol (TCP), an Internet application (e.g., a file transfer) can use only one access network at a time. Due to recent developments, however, concurrent multipath transfer (CMT) using the stream control transmission protocol (SCTP) (Wallace and Shami, 2010, 2012a,b, 2014; Stewart) can enable multihomed devices to exploit additional network resources for transport layer communications. A most recent literature on multihoming by other authors is available in Luo et al. (2014), Kim et al. (2014), Yao et al. (2014), Campista et al. (2014), Secci et al. (2014), Kuntz et al. (2013), and Ismail et al. (2013).

Although aggregating the resources of multiple network paths for a single transport layer session has been shown to increase throughput potential (Ye et al., 2004; Ahmed Abd et al., 2004; Iyengar et al., 2006), improvement is restricted to the assumption of a infinite (i.e., excessively large) receive buffer (RBUF); which is not a realistic scenario, especially for mobile devices with limited memory. When the RBUF is limited, aggregated performance diminishes; primarily due to *naive* round robin scheduling. Additionally, when multiple paths have disparate performance characteristics, such as round trip times (RTTs), throughput can reduce to the capacity of the slowest path (Iyengar et al., 2007). The problem exists, mainly, from a need to transfer data reliably and in sequential order to a destination with limited room for buffering. The literature has dubbed this as the *receive buffer blocking* problem.

In this paper we propose an on-demand scheduler (ODS) to address receive buffer blocking under delay-based disparity. ODS differs from previous scheduling algorithms in that it waits until congestion and flow control allow transmission before assigning a packet to a destination address. Furthermore, ODS requires a new congestion window (CWND) update policy to circumvent one

☆This paper has been published in part in the Proceedings of *The 8th International Wireless Communications and Mobile Computing Conference* (*IWCMC*), Cyprus, August 27–31, 2012.

E-mail addresses: kmeerja2@uwo.ca (K.A. Meerja), ashami2@uwo.ca (A. Shami).

[1] This work was carried out when T.D. Wallace was working at The University of Western Ontario.

http://dx.doi.org/10.1016/j.jnca.2014.09.008
1084-8045/

destination from monopolizing shared resources, like the RBUF. Contrary to SCTP's standard congestion avoidance mode, our new update policy confines CWND growth to the bandwidth potential of a network path.

The rest of the paper is organized as follows: Section 2 provides a brief literature review on receive buffer blocking, congestion window update policies, and buffer auto-sizing; Section 3 compares singlehoming to multihoming from the perspectives of network topology and transport layer architecture and the current scheduling approaches for CMT; the proposed algorithms for our new on-demand scheduler and CWND update policy are introduced in Section 4; Section 5 illustrates the performance of each scheduling algorithm; finally, Section 6 summarizes the work and suggests future direction.

2. Literature review

Related work is broken into three major categories: (1) receive buffer blocking, (2) congestion window update policies, and (3) transport layer buffer sizes.

2.1. Receive buffer blocking

The performance of CMT, when constrained by a RBUF, was showcased in Iyengar et al. (2005, 2007). The authors demonstrated poor transfer times using a shared RBUF between high and low bandwidth paths. In fact, in some circumstances using only the higher bandwidth path provided better performance results. The problem was dubbed *receive buffer blocking*; described as an inefficiency causing a sender to pause transmission while cumulative packets remain unacknowledged. Attempts to mitigate this problem took the form in various retransmission policies, such as: (1) send to the same destination after a loss; (2) send to any destination with an open[2] CWND; (3) send to the destination with the largest CWND; (4) send to the destination with the largest slow-start threshold (SSTHRSH); and (5) send to the destination with the lowest loss rate. Although conclusions were vague, the authors felt that loss rate was of most importance when choosing a destination for retransmission. A final strategy suggests the combination of retransmission policies (2)–(4) when path characteristics are similar (Liu et al., 2008). For instance, when multiple CWNDs are equal, choose a destination based on SSTHRSH or loss rate; else if all variables are the same, make a random selection.

A similar blocking effect can also occur at the sender if acknowledged packets are unable to leave the send buffer (SBUF); even if packets are received successfully, they must remain in the SBUF until an acknowledgement guarantees ordered delivery. Keeping packets in the SBUF, even after they have been acknowledged, is a safe guard against receiver-side reneging.[3] Assuming a receiver never reneges, Yilmaz et al. (2010) proposed the *non-renegable* selective acknowledgement (NR-SACK) as a way of freeing up room for new transmissions. The new acknowledgement type simply tells the sender to remove acknowledged packets from the SBUF, regardless of reordering.

More recently, modelling techniques were applied to CMT in an attempt to avoid receive buffer blocking altogether. Yang et al. (2010a) modify the well known PFTK model (Padhye et al., 2000) to calculate a minimum RBUF size, such that receive buffer blocking cannot exist. The same authors, moreover, used a similar model in Yang et al. (2010b), to choose the configuration of destination addresses that will maximize throughput for a given RBUF size.

While the previous approaches can offer some relief, we think intelligent scheduling shows the most promise. Using performance characteristics (e.g., bandwidth and delay), an ideal scheduler should be able to approximate packet delivery times; thus minimizing reordering at the receiver. The first scheduling algorithm used for CMT, however, simply transmitted the next cumulative packet to the first destination with an open CWND (Iyengar et al., 2006), regardless of delivery time. As this method ignores bandwidth and delay, it has no effect on receive buffer blocking. Such an approach will be referred to as *naive*, since no intelligence is used in the scheduling process.

Contrary to the naive approach, Casetti et al. (2004) proposed the bandwidth aware scheduler (BAS) to approximate packet delivery times before transmission. BAS assigns newly created packets to virtual send queues corresponding to one of the receiver's destination addresses. To make its scheduling decision, BAS uses a destination's bandwidth estimate as well as a backlog of scheduled packets to predict delivery times. A new packet is then placed in the send queue of the destination with the earliest delivery time. Although BAS decides which packet is sent to which destination, it does not control when packets are transmitted. Unfortunately, maintaining an accurate backlog can be difficult, especially if packets are lost and need to be retransmitted. To simplify the matter, the same authors improved BAS in Fiore et al. (2007). The updated BAS assigns a new packet to a destination with the lowest *reception index* (see Section 3.4 for details).

2.2. Congestion window update policies

An extensive survey presenting twenty years worth of TCP implementations can be found in Afanasyev et al. (2010). The survey, moreover, explores a number of alternative congestion window update policies aimed at specific network problems, not uncommon to SCTP, nor CMT. Unfortunately, very few techniques from Afanasyev et al. (2010) suggest a solution to our particular problem that concerns CMT; i.e., how to prevent one destination address from monopolizing the shared RBUF?

For the most part, techniques for updating the CWND try to send as much information, as fast as possible, while still minimizing loss at the bottleneck link. For example, works from Xu et al. (2004), Kliazovich et al. (2008), and Ha et al. (2008) approach the *optimal CWND*[4] in a related manner, i.e., by rapidly increasing the CWND (similar to slow-start) until noticing a loss, then applying some algorithm so that new growth approximates a horizontal asymptote.

Unlike other research from this area, the work in Wang et al. (2005) points out an interesting relationship between achievable rate and the size of the CWND. For instance, if the network is uncongested, throughput will continue to rise, but flattens when congestion sets in. Taking advantage of this notion, the authors developed a scheme called persistent noncongestion detection (PNCD) to probe for additional bandwidth. PNCD calculates a *congestion boundary*, i.e., an estimate on network limitation, to benchmark throughput potential. PNCD will then either increase or decrease a counter, depending on whether sampled throughput is above or below this congestion boundary. Finally, every time the counter equals the CWND, PNCD assumes more bandwidth is available and increases the size of the current CWND.

[2] The term *open* means the size of the CWND is greater than the number of outstanding packets.

[3] The potential for a receiver to give an acknowledgement only to disregard it later.

[4] In this case, the optimal CWND is a maximum value that does not overflow the bottleneck.