Improving TCP Performance in Residential Broadband Networks: A Simple and Deployable Approach

Maxim Podlesny*
University of Waterloo
Waterloo, Canada
mpodlesn@cs.uwaterloo.ca

Carey Williamson
University of Calgary
Calgary, Canada
carey@cpsc.ucalgary.ca

ABSTRACT

ADSL and cable connections are the prevalent technologies available from Internet Service Providers (ISPs) for residential Internet access. Asymmetric access technologies such as these offer high download capacity, but moderate upload capacity. When the Transmission Control Protocol (TCP) is used on such access networks, performance degradation can occur. In particular, sharing a bottleneck link with different upstream and downstream capacities among competing TCP flows in opposite directions can degrade the throughput of the higher speed link. Despite many research efforts to solve this problem in the past, there is no solution that is both highly effective and easily deployable in residential networks. In this paper, we propose an Asymmetric Queueing (AQ) mechanism that enables full utilization of the bottleneck access link in residential networks with asymmetric capacities. The extensive simulation evaluation of our design shows its effectiveness and robustness in a variety of network conditions. Furthermore, our solution is easy to deploy and configure in residential networks.

Categories and Subject Descriptors

C.2.2 [**Computer-Communication Networks**]: Network Protocols; C.2.5 [**Computer-Communication Networks**]: Local and Wide-Area Networks

General Terms

Performance

Keywords

TCP, asymmetric links, residential networks

1. INTRODUCTION

The tremendous growth of the Internet has led to significant improvements of data transmission rates. In earlier times, residential users accessed the Internet through dial-up connections with speeds of 56 Kbps. Today, users are widely subscribed to broadband connections, which include Asymmetric Digital Subscriber Line (ADSL) [1] and cable [2] for accessing the Internet. As of June 2008, about 60% of high-speed lines were designed to serve residential end users in the USA [3]. Cable accounted for 46.7% of these lines, and ADSL for 34.1%, which means that these two technologies accounted for over 80% of residential broadband Internet access.

Since residential users typically download much more Internet content than they upload, the download link is usually provisioned with higher capacity than the upload link. For example, typical rates for Internet access are a download capacity of 6 Mbps, and an upload capacity of 756 Kbps, which is highly asymmetric. The Transmission Control Protocol (TCP) [13] is widely used as the transport layer for reliable data delivery on the Internet. However, TCP can become inefficient in scenarios when TCP flows in opposite directions share a bottleneck link with asymmetric upload and download rates [15]. Thus, a typical residential network may experience TCP throughput degradation if there is TCP traffic flowing concurrently in both directions. However, a residential user paying for a particular upload/download speed to a Internet Service Provider (ISP) is interested in full utilization of the uplink and downlink.

Figure 1 demonstrates the TCP throughput degradation, as observed in an *ns-2* simulation experiment. In this scenario, two TCP NewReno flows in opposite directions are traversing an access link, for which the upload and download link capacities are 1 Mbps and 10 Mbps, respectively. The results show that the upload link utilization is 97%, while the download link is utilized only 26%. Moreover, the throughput on the download link is unstable and unpredictable with time, ranging from almost zero up to the full link capacity.

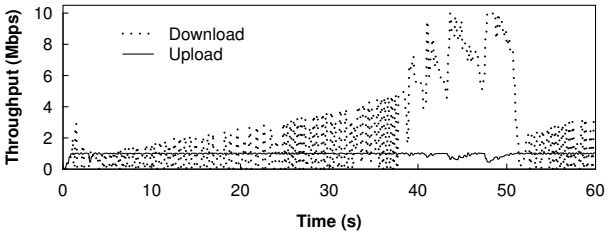

Figure 1: TCP Throughput in an asymmetric network.

Several researchers [16, 23] have explored the dynamics of two-way TCP traffic over a common bottleneck link, and Kalampoukas *et al.* [15] extended this analysis to asymmetric links. They explained that the degradation of the TCP download throughput was caused by *ACK compression*: the phenomenon in which download ACKs arrive in clumps, after being queued behind data packets on the uplink. Recently, Heusse *et al.* [11] revisited the fundamental reasons of TCP performance degradation in asymmetric links. They showed that ACK compression is not the fundamental cause. Instead, the TCP throughput degrades because of a *pendulum effect*, in which the bottleneck asymmetric link is busy only in one direction at a time. This observation undercuts many of the previously proposed solutions in the literature, which assume that the TCP throughput degradation in asymmetric links is due to ACK compression [15]. Other proposed solutions are not easy to config-

*This work was done when the author was the part of the University of Calgary

ure [5], or require changes of TCP stack [19] on one or both ends of a TCP connection, which may be unwieldly or unlikely, especially for home Internet users.

In this paper, we study the problem of how to achieve full resource utilization in both directions on asymmetric access links in residential networks. We propose a new mechanism called Asymmetric Queueing (AQ). The main idea is to serve TCP data and TCP ACK traffic through different queues, relying on specific control rules that we derive mathematically. The extensive simulations demonstrate the effectiveness and robustness of our approach in a wide range of network topology and traffic conditions. The main advantages of our solution are:

- full utilization of network access link in both directions
- easy incremental deployment in residential access networks
- simple and robust configuration process.

The key difference of our solution from previous ones is the ease of deployment in home networks.

The rest of the paper is structured as follows. In Section 2, we present our model and the theoretical analysis of the problem. Section 3 describes the details of the proposed solution. Section 4 provides the simulation results from the performance evaluation of our scheme. Section 5 discusses the practical applicability of our solution, and its robustness to misconfiguration. Section 6 summarizes prior related work. Finally, Section 7 concludes the paper.

2. MODEL AND ANALYTICAL FOUNDATION

In this section, we describe our network model and assumptions, and derive the analytical basis of the proposed solution.

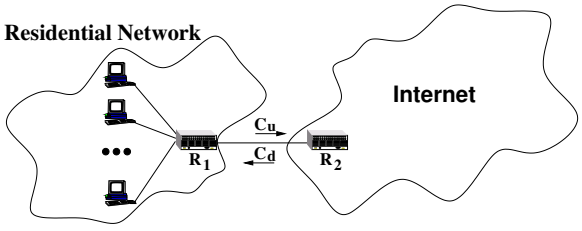

Figure 2: The dumbbell network topology model.

We consider a simple dumbbell network topology with asymmetric bottleneck links, as shown in Figure 2. The residential network is connected to the Internet through upload link R_1R_2 and download link R_2R_1, with respective capacities C_u and C_d, where $C_u < C_d$. We assume that these links constitute the bottleneck [8]. Thus, we implicitly assume that customers use higher-capacity wired access to their residential gateways. The TCP data packet size is S_D, and the size of a TCP ACK packet is S_A. Let us assume that the sending rates of TCP data and TCP ACK packets in the upload direction, from R_1 to R_2, are R_D and R_A, respectively. Let us also assume that the sending rates of TCP data and TCP ACK packets in the download direction, from R_2 to R_1, are R'_D and R'_A, respectively. Since we desire full utilization of both upload and download links, we require that:

$$R_D + R_A = C_u \tag{1}$$

$$R'_D + R'_A = C_d \tag{2}$$

Assuming that each TCP data packet is acknowledged by a TCP ACK packet, we may calculate R_A and R'_A as follows:

$$R_A = R'_D \frac{S_A}{S_D} \tag{3}$$

$$R'_A = R_D \frac{S_A}{S_D} \tag{4}$$

Substituting Equations (3) and (4) into Equations (1) and (2), we have that:

$$R_A = \frac{C_d S_D - C_u S_A}{S_D^2 - S_A^2} S_A \tag{5}$$

$$R_D = \frac{C_u S_D - C_d S_A}{S_D^2 - S_A^2} S_D \tag{6}$$

Thus, the upload link should be shared between TCP data and TCP ACK packets according to the following relationship:

$$\frac{R_D}{R_A} = \frac{r_c - r_s}{(1 - r_c r_s) r_s} \tag{7}$$

where

$$r_c = \frac{C_u}{C_d}, \quad r_s = \frac{S_A}{S_D} \tag{8}$$

This is the fundamental principle underlying our approach. In the next section, we discuss further details about our solution.

3. SOLUTION DETAILS

To implement our solution, we propose to use two different queues, the D queue for serving data traffic and the A queue for serving TCP ACK packets. The queues are served so that the ratio of the service rate of the D queue and the A queue is governed by Equation (7). The buffer at the upload link is shared in a similar fashion (i.e., $\frac{B_D}{B_A} = \frac{R_D}{R_A}$), where B_D is the buffer space allocated for the D queue, and B_A is the buffer space allocated for the A queue. The router classifies incoming traffic based on packet size. In particular, if the packet size exceeds 40 bytes, then the packet is served through the D queue, otherwise through the A queue. The scheduling algorithm is work-conserving, and enforces the ratio of the service rates of the queues according to Equation (7) only if both the D queue and the A queue are occupied (i.e., non-empty). The AQ mechanism is deployed only at the upload link.

Up to now, we have assumed that there are only two types of traffic in the network: TCP ACK and TCP data. However, there may also be non-TCP traffic, like UDP for VoIP and video streams. To accommodate this more general traffic scenario, we use the following approach. We assume that the upload and download rates for non-TCP traffic are R_u^U and R_d^U, respectively. Then the remaining capacity available for TCP uploading and downloading is $C_u - R_u^U$ and $C_d - R_d^U$, respectively. The A queue and the D queue are served as follows:

$$\frac{R_D}{R_A} = \frac{r_c - r_s}{(1 - r_c r_s) r_s} \tag{9}$$

where

$$r_c = \frac{C_u - R_u^U}{C_d - R_d^U}, \quad r_s = \frac{S_A}{S_D} \tag{10}$$

The volume of non-TCP traffic in each direction can be measured by the AQ router. The non-TCP traffic is served through the D queue. We periodically update the ratio between serving rates of the D queue and the A queue using the most recently measured values

of R_u^U and R_d^U. The recalculation period is T. According to [14], $75-90\%$ of all Internet flows have an RTT less than 200 ms. Thus, to avoid undue oscillations in the end-to-end control, we set $T = 200\ ms$. We assume that the rates of UDP traffic in both directions do not exceed the corresponding link capacities. Thus, r_c in our scheme is always bigger than 0. If the rate of UDP traffic exceeds the corresponding link capacity, e.g., $R_u^U > C_u$, then TCP traffic is completely stifled by UDP traffic. Note that this would hold for both DropTail and our scheme.

There are mathematical constraints on the degree of asymmetry that our approach can handle. In particular, since $\frac{R_D}{R_A}$ should be positive, we require that $r_c - r_s > 0$, and thus a limiting constraint of our solution is:

$$\frac{C_u}{C_d} > \frac{S_A}{S_D} \tag{11}$$

As an example, consider Ethernet, for which S_A = 40 bytes and S_D = 1500 bytes. To apply our mechanism, $\frac{C_u}{C_d}$ should be no smaller than 0.027. Fortunately, typical values for $\frac{C_u}{C_d}$ for DSL and cable are 0.07 to 0.1 (or larger), for at least 97% of hosts [8]. Thus our solution is widely applicable to residential networks, provided at least one host uses wired access to its home gateway. If all customers of a home network use wireless access like WiFi, then the bottleneck for them may be the wireless channel, and we cannot attain full utilization of the asymmetric access link. If at least one host is connected to the home gateway through Ethernet, then the asymmetric access link is the bottleneck at least for that host. It makes the full utilization of the access link feasible, which is realized by our scheme. We plan to extend our approach for wireless residential networks as future work.

There are four parameters for configuring our solution: C_u, C_d, S_A, and S_D. The first two parameters are determined by the Internet Service Provider (ISP), while the latter two parameters reflect Internet traffic characteristics. According to [12], the packet size distribution of IP traffic is bimodal with major modes at around 40 bytes corresponding to TCP ACKs, and 1500 bytes corresponding to the Maximum Transmission Unit (MTU) for Ethernet. Therefore, assuming that a TCP ACK packet does not contain options, we set S_A to 40 bytes and S_D to 1500 bytes. Thus, our approach is simple to configure for residential network users. Since the serving rates R_D and R_A do not depend on the packet sizes of non-TCP traffic, their distribution does not affect our design.

Let us now look at how our scheme affects the latency of packets at the upload link buffer. The maximum queueing delay under DropTail at the upload link buffer is:

$$d_{DT} = \frac{B_D + B_A}{R_D + R_A} \tag{12}$$

The maximum queueing delay of data packets under the AQ at the upload link buffer is:

$$d_{AQ} = \frac{B_D}{R_D} \tag{13}$$

Since:

$$R_A = kR_D, \quad k = \frac{(1 - r_c r_s) r_s}{r_c - r_s} \tag{14}$$

$$B_A = \frac{B_D R_A}{R_D} \tag{15}$$

using Equation (13), we have that:

$$B_A = d_{AQ} R_A \tag{16}$$

Substituting Equations (13), (14), and (16) into Equation (12), we have that:

$$d_{DT} = \frac{d_{AQ} R_D + d_{AQ} k R_D}{R_D + k R_D} = d_{AQ} \tag{17}$$

Thus, we have shown that our scheme does not worsen latency comparing to DropTail. Addressing the problem of real-time traffic is beyond the scope of this paper.

We do not consider the case of packets carrying both data and ACKs, which is out of the scope of the original problem formulation. However, we assume that the majority of the traffic in home residential networks is FTP-like, which means that packets do not carry both data and ACKs.

Our solution is assumed to be deployed on the upload link only. It means that the occupancy of the buffer at the download link by the ACK packets of the upload traffic cannot be controlled. Therefore, our design is not applicable for transport protocols using delay-based congestion control like TCP Vegas [7]. However, the AQ can be used for any loss-based TCP version.

The potential concern that may arise is that keeping all ACKs in the reverse direction is not necessary optimal, and, therefore, our solution is not optimal either in terms of maximizing the sum of data packets in both directions. However, we believe that maximizing that sum has a challenging stability issue in the context of TCP. Our design does not have that issue and is effective in a wide variety of network topologies and traffic patterns. We think that robustness and configuration simplicity of the scheme is more valuable than its optimality for home Internet users, who simply want robust and predictable performance.

4. SIMULATION EVALUATION

In this section, we evaluate the performance of our solution through simulations using version 2.29 of the *ns-2* network simulator [18]. We use the topology shown in Figure 2. The download and upload link capacities are 10 Mbps and 1 Mbps, respectively, and their buffer sizes are both set to 50 packets. The access link has 100 Mbps capacity and a buffer size of 1000 packets. The propagation delays for the bottleneck link and home access link are 5 ms and 1 ms, respectively. We randomize the propagation RTTs of the flows uniformly between 20 ms and 300 ms. There are 5 long-lived flows in the upload direction, and 10 long-lived flows in the download direction, which is more realistic than the simple two-flow scenario in Figure 1. All the flows use TCP NewReno [9]. The data segment size is 1460 bytes. Each experiment lasts 60 s, with 5 replications for each of the considered parameter settings.

The performance metric for average link utilization is calculated using all but the warmup period (first 5 seconds) of the experiment. For calculating instantaneous link utilization, we calculate and plot the results every 0.2 s interval. The primary performance metric is the average link utilization. We compare the results under our mechanism with the traditional DropTail queue in the same settings.

4.1 Illustration of the Behavior

To demonstrate the performance of our solution, we run the experiment under the same settings as in Figure 1, but with AQ on the upload link. The propagation RTT of the upload and download flows is 100 ms. Figure 3 shows steady TCP throughput in both directions for AQ. In particular, the average utilization of each of the upload and download links is 99%. For DropTail the TCP throughput is unstable in the download direction (see Figure 1).

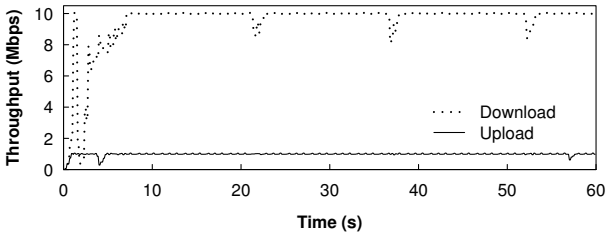

Figure 3: Example illustrating the behavior under AQ.

4.2 Effect of Download Capacity

To explore the effect of download capacity on our approach, we vary it between 1 Mbps and 10 Mbps, while keeping the upload capacity fixed at 1 Mbps. Thus we vary the asymmetry of the bottleneck link capacity. In addition, we add stochastic Web traffic to the network in the download direction. The intensity of the Web traffic is 1 flow per sec (fps), and flow sizes are Pareto distributed with a mean of 30 KB and a shape index 1.3. Figure 4 shows that AQ supports full utilization of both upload and download links for the whole range of asymmetry studied. On the other hand, DropTail is unable to do that: the download link utilization decreases by 20% for the download capacity 10 Mbps. Interestingly, even when the bottleneck link is symmetric (i.e., the download capacity is 1 Mbps), our approach has an 11% advantage over DropTail in upload link utilization.

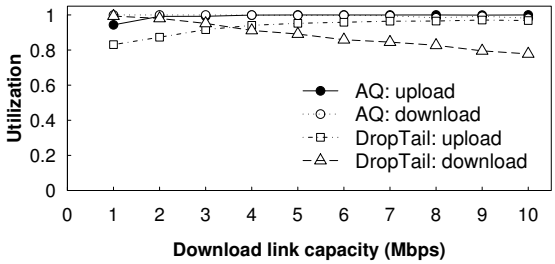

Figure 4: Effect of download capacity on link utilization.

4.3 Effect of Upload Capacity

We study how the upload capacity affects the performance by varying the upload capacity from 0.256 Mbps to 10 Mbps, while keeping the download capacity fixed at 10 Mbps. Thus we again vary the asymmetry of the bottleneck link. Figure 5 shows that our approach has an advantage over DropTail for the whole range of the varied parameter, except for the case of 0.256 Mbps. For that value of the upload capacity, the download link utilization is near zero, which reflects the limiting constraint of our design expressed by Equation (11). For other settings, the improvement of the download utilization for AQ is up to 20% compared to DropTail. For upload link capacities larger than 4 Mbps, there is a monotonic decrease of upload link utilization (down to 84%) under our scheme, which we explain as follows. Since the buffer size for the upload link is fixed, increasing the upload link capacity leads to decreased TCP throughput. If the link is symmetric (i.e., the upload capacity is 10 Mbps), then the AQ mechanism improves the upload link utilization by 13% compared to DropTail.

4.4 Effect of Packet Size

We investigate the effect of the packet sizes by varying the TCP data segment size between 100 bytes and 1460 bytes, while keeping

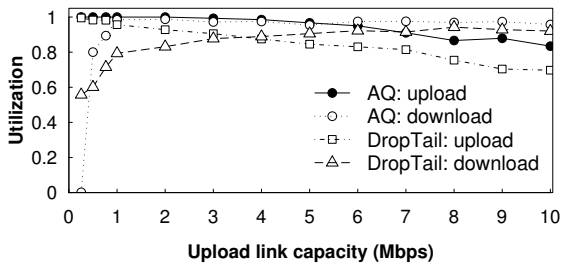

Figure 5: Effect of upload capacity on link utilization.

the size of an ACK packet fixed at 40 bytes. Figure 6 shows that our scheme provides higher utilization than DropTail when the TCP data segment size is larger than 360 bytes. The use of smaller TCP segment sizes leads to zero download link utilization in AQ, due to the limiting constraint of our solution expressed in Equation (11). For other settings, there is a decrease of download link utilization down to 85% for segment sizes smaller than 900 bytes, which we explain as follows. Since link buffers are expressed in packets, the decrease of a packet size implies a smaller size of the buffered TCP traffic, and therefore degraded download performance for TCP. The upload link is utilized fully since its capacity is one-tenth that of the download link.

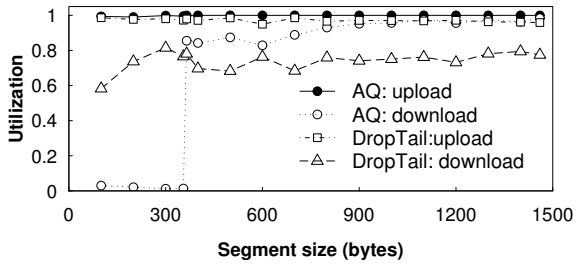

Figure 6: Effect of packet size on link utilization.

4.5 Effect of the Buffer Size

To explore the impact of the buffer size of the upload link, we vary the buffer size between 10 packets and 100 packets, while keeping other parameter settings at their defaults (1 Mbps and 10 Mbps). The results (not shown here) confirm that the link utilizations are higher for AQ than for DropTail, for all the tested values of the buffer size. In particular, our approach has up to 20% higher download link utilization. As expected, increasing the buffer size improves the download link utilization. For buffer sizes larger than 50 packets (the default setting), both the upload and download links are fully utilized. For DropTail, on the other hand, increasing the buffer size improves the upload link utilization, while deteriorating the download link utilization, consistent with the results in [11].

Figure 7 presents a more detailed breakdown of the AQ results. We observe that as the uplink buffer size is increased, the ratio between data and ACK usage of the uplink converges to 73% utilization of the uplink by data, and 27% by ACKs, which is a consequence of the control rules in our design. We attribute such a convergence behavior to the fact that TCP ACK packets on the uplink experience higher loss when there is a small buffer size allocated for them on the uplink. Increasing the buffer size leads to the saturation of the bandwidth allocated to upload TCP ACK traffic. Moreover, the utilization of the download TCP data traffic mono-

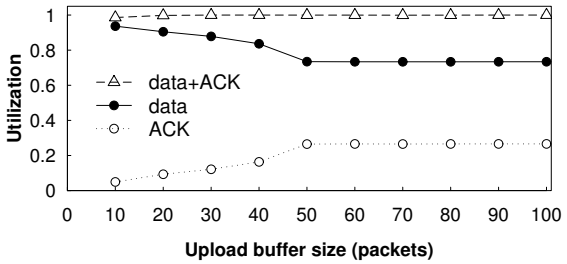

Figure 7: Effect of uplink buffer size on upload link utilization by TCP data and TCP ACK traffic for AQ.

tonically increases with the growth of the uplink buffer size.

4.6 Effect of Number of Flows

To explore how our solution scales with the number of flows, in the first part of our study we fix the number of upload flows to 3 and vary the number of download flows between 1 and 21. We define the scaling factor as the ratio between the download and upload flow populations. The results reported in Figure 8 show the total download link utilization monotonically increases from 39% to 93%, and from 69% to 99% under the DropTail and our scheme respectively. In addition, while the upload link is fully utilized for the whole range of the varied parameter under AQ, it is between 92% and 98% under DropTail.

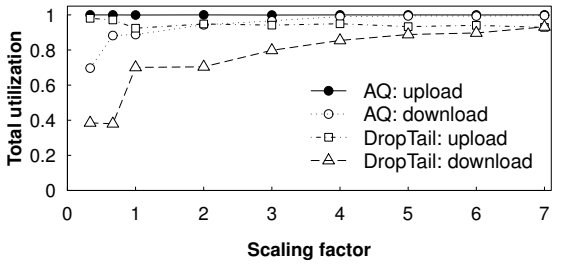

Figure 8: Effect of download population on link utilization.

In the second part, we set the number of download flows to 10, and change the number of upload flows between 1 and 20. The results reveal that the upload and download links are fully utilized for the whole range of the varied parameter under AQ. The total download link utilization decreases monotonically from 90% to 58%, and the total upload link utilization is between 86% and 100% under DropTail.

4.7 Effect of Sudden Changes

We investigate how our design reacts to dynamic changes in network traffic by adding competing UDP traffic in both directions. Figure 9(a) shows the sending rate of UDP traffic. For example, during the interval from 60s to 80s, the UDP upload rate is 0.5 Mbps and the UDP download rate is 3 Mbps. Figures 9(b) and 9(c) show that the upload throughput of TCP flows adapts well to the changes in upload capacity, and the upload link is fully utilized under either DropTail (b) or AQ (c). The only difference is during the very first and very last interval, during which there is no UDP upload traffic in the network. During these time periods, the AQ approach uses the upload link in a more effective and stable fashion. For the download link, Figures 9(d) and 9(e) show that DropTail provides unstable TCP throughput, while our approach adjusts well to the changing download capacity.

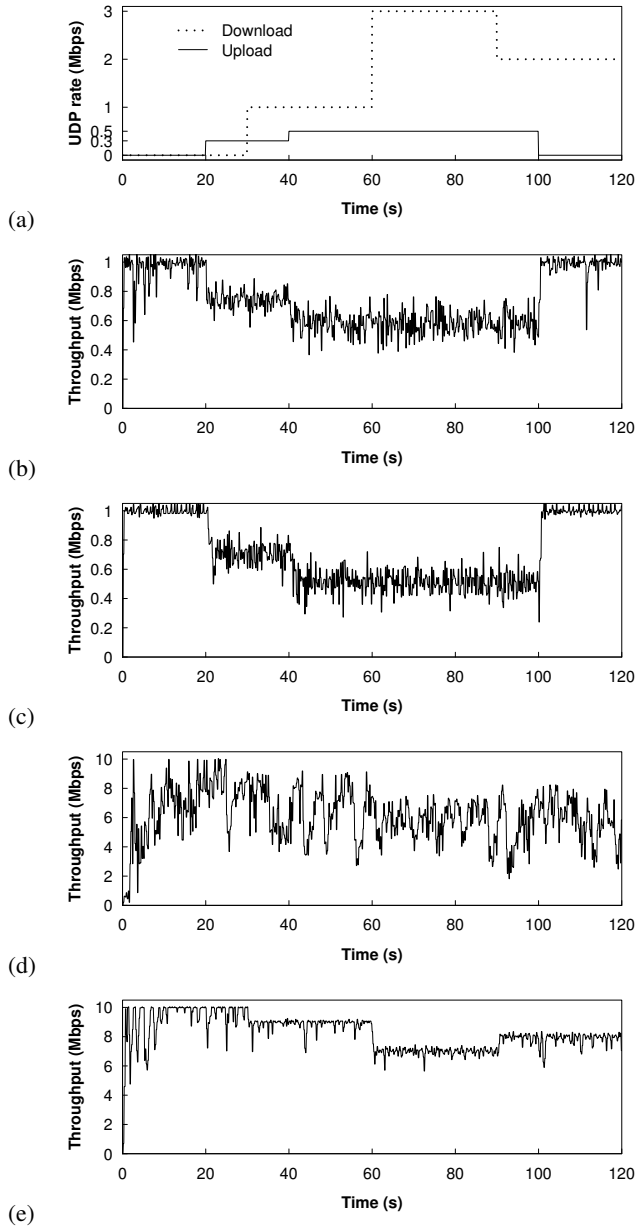

Figure 9: Effect of sudden changes in link capacities: (a) UDP rate; (b) average uplink utilization for DropTail; (c) average uplink utilization for AQ; (d) average downlink utilization for DropTail; (e) average downlink utilization for AQ.

4.8 Effect of UDP Traffic

To study how the overall rate of UDP traffic affects the performance of AQ, we first vary the rate of the download UDP traffic between 1 Mbps and 10 Mbps. The results (not shown here) indicate that our solution provides full utilization of both upload and download links for all UDP rates considered. By contrast, DropTail achieves comparable performance only for UDP rates between 7 Mbps and 9 Mbps. Next, we vary the rate of the upload UDP traffic between 0.1 Mbps and 1.0 Mbps. While DropTail provides upload and download link utilizations of 100% and 80%, respectively, the AQ approach provides full utilization of both the upload and download links.

5. DISCUSSION

In this section, we discuss the practical applicability of our solution, particularly in the case of possible (e.g., inadvertent or malicious) misconfiguration of the AQ router, for reasons beyond the control of users.

First, we consider the case of TCP versions in which not every TCP data packet is acknowledged by a TCP ACK packet. In particular, we consider TCP with delayed ACK [6], with the delayed ACK timeout set to 100 ms. We use the same network topology and traffic pattern as in Section 4.2 and vary the download capacity between 1 Mbps and 10 Mbps. Figure 10 shows the resulting utilization of the upload link by TCP data and TCP ACK packets for TCP NewReno with and without delayed ACKs. The throughput of TCP data flows with delayed ACKs is slightly better. Moreover, that difference becomes more evident (4% to 11%) as the download capacity is increased. However, the sum of the utilizations of uplink and downlink is the same for both versions of TCP.

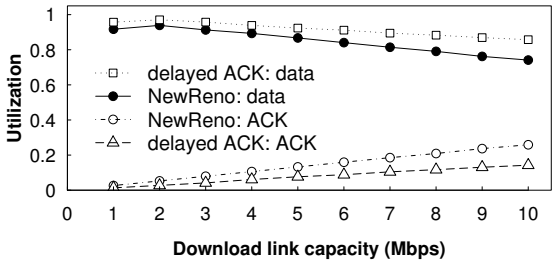

Figure 10: Average data utilization of upload link by TCP data and TCP ACKs for delayed ACKs.

We have analyzed the general scenario when not all the data packets are acknowledged. Due to space constraints, we do not provide the analysis. The results of the analysis show that our design performs well achieving its goal regardless of the number of data packets per ACK.

Second, we consider the general case of a misconfigured AQ router. We use the same topology and traffic scenario as in Section 4.3, and vary the upload capacity between 0.5 Mbps and 4 Mbps. We assume that the download capacity C_d is correctly configured to 10 Mbps, while the value of C_u is always set to 2 Mbps, regardless of the actual upload link capacity. The results indicate that the upload link is always fully utilized. However, Figure 11 shows that the download link is fully utilized only if the upload capacity matches or exceeds 2 Mbps (the misconfigured setting). We can explain this result as follows. If the upload capacity is less than 2 Mbps, then the ratio of upload and download link capacities is below the value of $r_c = 0.2$ used for configuring the AQ router. Since $\frac{R_A}{R_D}$ is a decreasing function of r_c, we allocate a lower service rate for the A queue than required for AQ to fully utilize the uplink and downlink. For upload capacities exceeding 2 Mbps, the misconfiguration is no longer a problem.

6. RELATED WORK

The problem of TCP throughput degradation in asymmetric links has attracted significant research attention. In general, there are two main approaches for tackling the problem: host-based and network-based.

The first approach assumes modifications at end hosts (senders and receivers). Ming-Chit *et al.* [19] proposed Acknowledgement based on Cwnd Estimation (ACE), the main idea of which is to vary

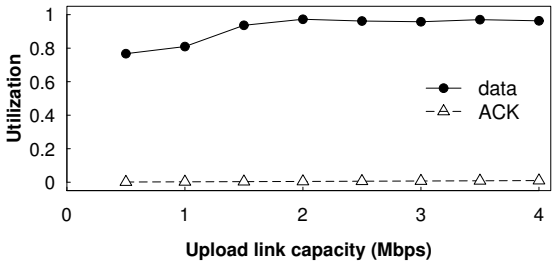

Figure 11: The value of upload capacity C_d is misconfigured. Average utilization of the download link by TCP data and TCP ACKs.

the number of data packets acknowledged by an ACK according to the congestion window. Ack Reconstruction [5] used with Ack Filtering (AF) aims at reconstructing an ACK stream at a data sender after its filtering by AF on the uplink. However, the technical skills required for modification of a TCP source may be beyond those of typical residential users.

In the second approach, new scheduling and queueing techniques are used instead of DropTail. Louati *et al.* [17] proposed Adaptive Class-based Queueing (ACQ), which requires one queue for data and one queue for ACKs, and uses the gradient projection method to determine a good configuration. While ACQ maximizes the sum of utilizations of uplink and downlink by data packets only, our design maximizes the sum of the utilizations of uplink and downlink, accounting for both data and ACK traffic. Importantly, while configuring ACQ is not a straightforward process, and may be challenging depending on the network topology and traffic pattern, configuring our scheme is simple and does not depend on the network topology or traffic pattern.

Connection-level bandwidth allocation [16] strives to provide a guaranteed minimum utilization of the download link by data packets while providing the maximum possible throughput for upload connection under this constraint. In contrast, we try to maximize the total utilization of both upload and download links. The other difference is that while our scheme does not depend on the window size of the download connection, the solution in [16] requires knowing that parameter to estimate the efficiency of download and upload connections. Thus, configuring the scheme in [16] can be a challenge for a home user.

ACK Filtering (AF) [5] is a link-layer mechanism for reducing the number of ACKs sent to the upload direction. This approach is effective, but can have subtle impacts on TCP throughput, depending on how the TCP version interprets the ACKs received. ACK Congestion Control (ACC) [5] assumes the use of Random Early Detection (RED) [10] for both data and ACK packets at the uplink, essentially applying congestion control to ACK packets. The problem with ACC is that configuring RED properly is always a challenging task. Moreover, AF and ACC can cause sender burstiness, a slowdown in window growth, and deterioration of the effectiveness of TCP fast retransmission [5]. Park *et al.* [21] proposed ACK-First Variable-size Queueing (AFVQ), which serves traffic through three different queues (TCP data, TCP ACKs, and non-TCP packets), and dynamically adjusts the size of the buffer allocated for TCP ACKs depending on the size of TCP data queue. However, AFVQ requires deployment at both uplink and downlink.

In [4], the authors recommended upstream link scheduling as a solution to the asymmetry problem. In particular, they consider per-flow queueing, like Fair Queueing (FQ) [20] and ACK-first

scheduling, which prioritize ACK packets over data packets. However, per-flow queueing requires per-flow state, which may not be an appropriate solution for resource-constrained residential gateways. In addition, Shekar *et al.* [22] showed that using the ACK-first mechanism can lead to unacceptably low throughput of data flows in the upload direction.

Detailed performance comparison of our approach to prior existing approaches remains for future work.

7. CONCLUSION

In this paper, we considered the problem of TCP performance degradation in asymmetric access links of residential networks. We proposed Asymmetric Queueing (AQ) as a mechanism for maximizing the utilization of upload and download links in such an environment. The key idea in AQ is to serve TCP data and ACK traffic through two different queues. Deployment is only required on the upload bottleneck link.

We evaluated our approach using extensive *ns-2* simulations. The simulation results show that our design is effective in a wide range of network and traffic conditions, improving the download and upload link utilization of asymmetric links by up to 22% and 20% respectively. We also have investigated the robustness of our approach to possible misconfiguration of the AQ router, and demonstrated that our solution is quite robust in these cases. The primary advantage of our approach, which distinguishes it from previously proposed schemes, is the ease of configuration for deployment in home networks.

As future work, we plan to implement our mechanism in a real network. Also, it will be interesting to apply our approach to wireless residential access networks.

Acknowledgement

Financial support for the work was provided in part by iCORE (Informatics Circle of Research Excellence) in the Province of Alberta, and by NSERC (Natural Science and Engineering Research Council) Discovery Accelerator RGPAS-380438 45185.

8. REFERENCES

[1] Asymmetric Digital Subscriber Line (ADSL) Tranceivers. ITU-T Recommendation G.992.1, June 1999.

[2] Cable Modem to Customer Premise Equipment Interface. http://www.cablelabs.com/specifications/CM-SP-CMCI-C01-081104.pdf. CableLabs, November 2008.

[3] http://www.internetworldstats.com, July 2009.

[4] BALAKRISHNAN, H., PADMANABHAN, V., FAIRHURST, G., AND SOORIYABANDARA, M. TCP Performance Implications of Network Path Asymmetry. RFC 3449, December 2002.

[5] BALAKRISHNAN, H., PADMANABHAN, V., AND KATZ, R. The Effects of Asymmetry on TCP Performance. *Mobile Networks and Applications 4*, 3 (October 1999), 219–241.

[6] BRADEN, R. Requirements for Internet Hosts - Communication Layers. RFC 1122, April 1989.

[7] BRAKMO, L., MALLEY, S., AND PETERSON, L. TCP Vegas: New Techniques for Congestion Detection and Avoidance. In *Proceedings ACM SIGCOMM 1994* (August 1994).

[8] DISCHINGER, M., HAEBERLEN, A., GUMMADI, K., AND SAROIU, S. Characterising Residential Broadband Networks. In *Proceedings ACM IMC 2007* (October 2007).

[9] FLOYD, S., AND HENDERSON, T. The NewReno Modification to TCP's Fast Recovery Algorithm. RFC 2582, April 1999.

[10] FLOYD, S., AND JACOBSON, V. Random Early Detection Gateways for Congestion Avoidance. *IEEE/ACM Transactions on Networking 1*, 4 (August 1993), 397–413.

[11] HEUSSE, M., MERRITT, S., BROWN, T., AND DUDA, A. Two-way TCP Connections: Old problem, New Insight. *ACM Computer Communication Review 41*, 2 (April 2011), 6–15.

[12] HURTIG, P., JOHN, W., AND BRUNSTRMNAN, A. Recent Trends in TCP Packet-Level Characteristics. In *International Conference on Networking and Services (ICNS)* (June 2011).

[13] JACOBSON, V. Congestion Avoidance and Control. In *Proceedings ACM SIGCOMM 1988* (August 1988).

[14] JIANG, H., AND DOVROLIS, C. Passive Estimation of TCP Round-Trip Times. *ACM Computer Communication Review 32*, 3 (July 2002), 75–88.

[15] KALAMPOUKAS, L., VARMA, A., AND RAMAKRISHNAN, K. Improving TCP Throughput over Two-Way Asymmetric Links: Analysis and Solutions. In *ACM SIGMETRICS* (June 1998).

[16] KALAMPOUKAS, L., VARMA, A., AND RAMAKRISHNAN, K. Two-Way TCP Traffic over Rate Controlled Channels: Effects and Analysis. *IEEE/ACM Transactions on Networking 6*, 6 (Decmber 1998), 729–743.

[17] LOUATI, F., BARAKAT, C., AND DABBOUS, W. Handling Two-Way TCP Traffic in Asymmetric Networks. In *Proceedings IEEE International Conference on High Speed Networks and Multimedia Communications (HSNMC) 2004* (July 2004).

[18] MCCANNE, S., AND FLOYD, S. *ns Network Simulator. http://www.isi.edu/nsnam/ns/.*

[19] MING-CHIT, I., JINSONG, D., AND WANG, W. Improving TCP Performance over Asymmetric Networks. *ACM Computer Communication Review 30*, 3 (July 2000), 45–54.

[20] PAREKH, A., AND GALLAGER, R. A Generalized Processor Sharing Approach to Flow Control in Integrated Services Networks: The Single-Node Case. *IEEE/ACM Transactions on Networking 1*, 3 (June 1993), 344–357.

[21] PARK, J., PARK, D., HONG, S., AND PARK, J. Preventing TCP Performance Interference on Asymmetric Links Using ACKs-first Variable-size Queuing. *Computer Communications 34*, 6 (May 2011), 730–742.

[22] SHEKHAR, D., QIN, H., KALYANARAMAN, S., AND KIDAMBI, K. Performance Optimization of TCP/IP over Asymmetric Wired and Wireless Links. Invited paper at European Wireless 2002, February 2002.

[23] ZHANG, L., SHENKER, S., AND CLARK, D. Observations on the Dynamics of a Congestion Control Algorithm: The Effects of Two-Way Traffic. *ACM Computer Communication Review 21*, 4 (September 1991), 133–147.