Performance and Buffering Requirements of TCP Applications in Asymmetric Networks

Subir Varma
Hybrid Networks
6409 Guadalupe Mines Road
San Jose, CA 95120

Abstract - **Asymmetric networks are defined to be those in the forward (or downstream) and reverse (or upstream) link speeds may assume different values. These types of networks are becoming more prevalent due to the growing penetration of Hybrid Fiber Coax (HFC) and Asymmetric Digital Subscriber Loop (ADSL) systems, in the last mile or local loop. Consequently, it is important to understand and quantify the effect that asymmetry has on the performance of TCP applications. In this paper we study the effect of varying the upstream buffer size on TCP performance. We characterize the maximum achievable throughput as a function of the number of upstream buffers, and also compute the minimum downstream buffer size required to achieve full link throughput.**

I. Introduction

Asymmetric networks are defined to be those in the forward (or downstream) and reverse (or upstream) link speeds may assume different values. These types of networks are becoming more prevalent due to the growing penetration of Hybrid Fiber Coax (HFC) and Asymmetric Digital Subscriber Loop (ADSL) systems, in the last mile or local loop. Consequently, it is important to understand and quantify the effect that asymmetry has on the performance of TCP applications. An excellent start in this direction has been made by Lakshman et.al. [1], who carried out a detailed analysis of the effect of the downstream buffer size and random loss on TCP performance, as the asymmetry ratio between the downstream and upstream link speeds changes. This paper is a continuation of this work, and our main contributions are:

- *Quantification of the effect of the upstream buffer size on TCP performance*: For the case when the upstream link speed is smaller that the downstream link speed, we show that the upstream buffer size exerts a strong influence on TCP performance. In particular, we show that is the upstream buffer size is smaller than a certain threshold, then a downstream TCP application performance is the same as in a system with symmetric links (Fig. 4). For upstream buffer sizes larger than this threshold, the TCP throughput decreases monotonically with buffer size, until it reaches a stable minimum value, at a larger buffer size threshold. In their analysis [1], Lakshman et.al. overlooked this variation, since they assumed that the upstream buffers in an asymmetric system are always overflowing in steady state, which turns out not to be the case.

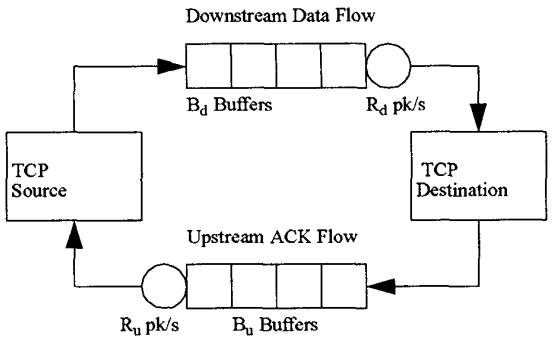

Fig. 1. System Model

- *Quantification of the downstream buffer size required to achieve full link throughput for asymmetric systems*: We extend the results in [1] on this subject, by explicitly considering the effect of the upstream buffer size on the downstream buffering requirements. Specifically we show that the rule of thumb which says that the number of downstream buffers should be at least equal to the delay bandwidth product in order to achieve full downstream link throughput, does not hold any longer in a asymmetric system. In fact the required number of downstream buffers in order to do is greater than the delay bandwidth product (Fig. 3), and the difference between the two numbers is proportional to the number of upstream buffers and the asymmetry ratio in the system.

The rest of this paper is organized as follows: Section II has the notation and definitions used in the rest of the paper, Sections III and IV provide conditions under which full TCP throughput is achieved in the downstream direction, as a function of the upstream buffer size, the asymmetry ratio and the delay-bandwidth product for the system. Section V analyzes the effect of the upstream buffer size on the throughput of TCP applications.

II. Notation

As in [1], we will consider the simplified system model in Fig. 1. Only the bottleneck links are explicitly modeled for both directions of data flow. For the purposes of the analysis in

0-7803-5417-6/99/$10.00 ©1999 IEEE.

this paper, we will assume that the TCP Reno Algorithm [2], is being for congestion control.

Define the following:

C_d (R_d). Capacity of the downstream channel, in bit/s (packets/s).

C_u (R_u). Capacity of the upstream channel, in bits/s (packets/s).

P_d. Packet size in the downstream direction, in bits.

P_u. Packet size in the upstream direction, in bits.

B_d. Number of packet sized buffers in the downstream channel device

B_u. Number of packet sized buffers in the upstream channel device

T_d. Total downstream latency, in seconds.

T_u. Total upstream latency, in seconds.

F. FTP rate in the downstream direction, in (packets/s).

W. TCP Window Size in packets, for a downstream FTP.

W_{LIM} . Maximum window size for the TCP connection. This number cannot exceed 64 Kbytes.

W_{MAX}. The system packet capacity. An increase in window size beyond this value, causes a packet loss (see equation 4 for an expression for W_{MAX}).

Note that

$$R_d = \frac{C_d}{P_d} \quad \text{and} \quad R_u = \frac{C_u}{P_u} \qquad \text{(EQ 1)}$$

The total round trip latency in the system T, is given by

$$T = T_d + T_u + \frac{1}{R_d} + \frac{1}{R_u} \qquad \text{(EQ 2)}$$

Define a parameter k as follows:

$$k = \frac{R_d}{2R_u} \qquad \text{(EQ 3)}$$

Note that the definition of k is different from that used in [1], due to the fact that we explicitly take into account that most TCP implementations use the "Delayed ACK" policy, so that the number of ACKs is reduced by half in steady state. As pointed out in [1], systems with $k > 1$ have a fundamentally different behavior, as compared to systems with $k < 1$. This is due to the fact that $k > 1$ causes the rate of packet arrival into the upstream buffer to exceed the upstream link capacity, thus stressing this buffer. It also increases the burstiness of the downstream traffic.

III. Buffer Overflows

A. Objective

Our basic objective is have enough buffers in the system to achieve throughput equal to the link capacity. A necessary condition for achieving this objective is to have enough buffers in the system, so that there no buffer overflows, even with large window sizes. This is the criteria that is analyses in this section. However, the application of this criteria to situations in which the delay-bandwidth product is small can lead to an overestimation of the numbers of buffers needed. In such situations it is possible to achieve throughput equal to link capacity, even in the presence of buffer overflows, provided certain conditions hold. In Section IV we take this into account and provide sufficient conditions that guarantee full throughput in the presence of buffer overflows.

Buffer overflows in the downstream link interact with the TCP congestion control algorithm in ways that cause the system throughput to decrease. Hence one of our objectives in deciding upon the buffer sizes, is to have enough buffers to avoid this situation.

There are two conditions which can lead to buffer overflow:

- The total buffer carrying capacity of the system is less than the TCP window size. This scenario is analyzed in Section III B.
- The downstream link is subject to large packet bursts at link rate. This scenario is analyzed in Section III C.

We derive in-equalities (10), (11) and (12) which have to be satisfied in order to avoid these two scenarios.

B. Buffer Overflows Due To Insufficient System Packet Capacity

The system capacity is defined to be the maximum number of packets that can be in the system at the same time. These packets can be either waiting for service at a buffer, or in transit across a link. When the TCP window size exceeds the system capacity, then it leads to buffer overflows.

The total system capacity W_{MAX} is given by the following formulae:

$$W_{MAX} = R_d T + B_d + 2kB_u \quad \text{if } k > 1 \qquad \text{(EQ 4)}$$

$$W_{MAX} = R_d T + B_d \quad \text{if} \quad k < 1 \qquad \text{(EQ 5)}$$

Derivation of equation (4): Let W(D) and W(U) be the packet capacities of the downstream and upstream links respectively. Then by Little's Law,

$$W(D) = R_d\left(T_d + \frac{1}{R_d} + \frac{B_d}{R_d}\right) \qquad \text{(EQ 6)}$$

$$W(U) = 2 \times \frac{R_d}{2}\left(T_u + \frac{1}{R_u} + \frac{B_u}{R_u}\right) \qquad \text{(EQ 7)}$$

In order to derive (6), note that R_d is the maximum rate at which the downstream link can receive packets, and the expression in the brackets is the maximum delay that a packet can experience, assuming that all the downstream buffers are full. In order to derive (7), note that the rate of ACKs entering the upstream queue is $R_d/2$, and since each ACK represents two packets, we multiply this expression by two. Using the fact that $W_{MAX} = W(D) + W(U)$, we arrive at (4).

Derivation of equation (5): In this case the following equations hold

$$W(D) = R_d\left(T_d + \frac{1}{R_d} + \frac{B_d}{R_d}\right) \qquad \text{(EQ 8)}$$

$$W(U) = 2 \times \frac{R_d}{2}\left(T_u + \frac{1}{R_u}\right) \qquad \text{(EQ 9)}$$

Note that since $k < 1$, it follows that $R_d/2 < R_u$, so that there is no build-up of the upstream queue size in the upstream direction. Hence unlike for the case $k > 1$ in equation (7), W(U) does not have a dependency on B_u, the upstream buffer size. Once again by adding (8) and (9), we arrive at (5).

Equations (4) and (5) tie together the latency, the asymmetry factor, the buffer sizes and the downstream packet rate with W_{MAX} which is the maximum number of packets that the system can support. If W_{MAX} is less than 64 Kbytes, then a value of TCP window size W_{LIM} greater than W_{MAX} causes buffer overflows in the system. Hence objective in choosing the buffer sizes B_u and B_d is that they should be sufficiently large so that

$$R_d T + B_d + 2kB_u > 64 \text{ Kbytes}, \quad \text{if } k > 1 \qquad \text{(EQ 10)}$$

$$R_d T + B_d > 64 \text{ Kbytes}, \quad \text{if } k < 1 \qquad \text{(EQ 11)}$$

From the structure of equations (10) and (11), it follows that systems with $k > 1$ need a smaller number of downstream buffers in order to satisfy (10), because of the contribution from the upstream buffers to the RHS of the equation. However as we show in Section V, having too many upstream buffers reduces the maximum throughput that the system can support.

C. Buffer Overflows Due to Bursty Traffic

It may happen that the TCP window size is less than the system capacity, but the downstream link is subject to a large burst of packets at link rate. This situation is peculiar to systems with $k > 1$. It is caused when the number of upstream buffers is kept small to enhance the system throughput (the reason for doing so is discussed in Section V). This leads to overflow of ACKs in the upstream buffer, so that on the average only one out of k ACKs is able to enter the buffer. Due to the cumulative acknowledgment policy in TCP, this ACK carries the acknowledgment for 2k packets (since each ACK acknowledges 2 packets). When this ACK gets to the source, it leads to the release of a burst of size 2k into the downstream link. Hence in order to avoid buffer overflow, the following condition should be satisfied:

$$B_d > 2k \qquad \text{(EQ 12)}$$

IV. Criteria For Achieving Full Throughput In The Presence Of Buffer Overflows

The criteria given for the smallest buffer sizes needed to ensure that there are no packet losses in equations (10), (11) and (12), constitute a sufficient but not necessary condition for the resulting throughput to be equal to the link rate. It is possible for (10) and (11) to be violated, and still be able to achieve full link rate. In this section we explore this situation in greater detail, and derive equations that constitute necessary *and* sufficient conditions for the throughput to equal the link rate. In the process we obtain the smallest possible downstream buffer size needed to guarantee full link rate.

In the rest of this section we find it convenient to revert to the units of Kbytes for the buffer and window sizes.

A. The Case $k < 1$

If $k < 1$, then the following two inequalities constitute necessary and sufficient conditions for the downstream throughput to equal the line rate:

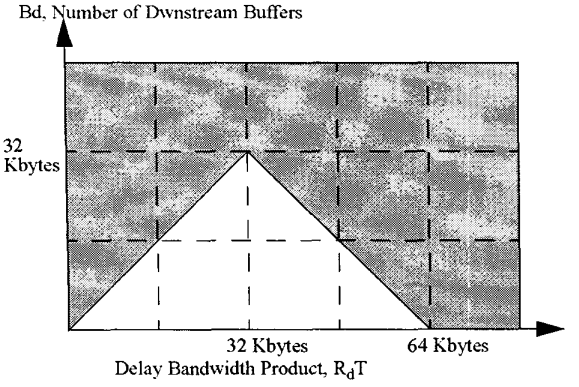

Fig. 2. Full Link Rate is achieved in the shaded region, for k < 1

$$B_d > R_dT \text{ if } R_dT \le 32 \text{ Kbytes} \quad \text{(EQ 13)}$$

$$B_d > 64 \text{ Kbytes} - R_dT \quad \text{(EQ 14)}$$
$$\text{if } R_dT > 32 \text{ Kbytes}$$

Note that inequality (14) is identical to (11). The region in which (13) and (14) are simultaneously satisfied, has been shaded in Fig. 2. From this result it follows that (11) (or (14)) provides a tight bound for the buffer size, only if the delay bandwidth product R_dT, exceeds 32 Kbytes. It ensures that the packet capacity of the system exceeds 64 Kbytes, so that there is no loss. An extrapolation of (11) for the case $R_dT < 32$ Kbytes, results in an overestimation of the amount of buffering required to attain link rate. In this range of the delay bandwidth product, it is possible to loose packets and still attain link rate, provided the inequality (13) is satisfied.

In order to derive (13)-(14), note that if

$$\frac{W_{MAX}}{2} > R_dT \text{ then } \bar{F} = R_d . \quad \text{(EQ 15)}$$

The reason for this criteria is connected with the functioning of congestion control mechanism in TCP Reno. When there is a single packet loss at a window size of W_{MAX}, the window falls to half this value. After this event, if (15) holds, then the resulting throughput is still greater than R_d, so that the resulting average throughput still equals the link rate.

Substituting the expression (5) for W_{MAX}, we arrive at the equivalent condition

$$B_d > R_dT \quad \text{(EQ 16)}$$

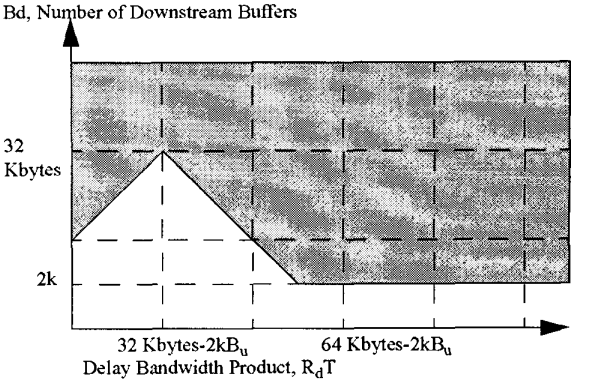

Fig. 3. Full link rate is achieved in the shaded region, for k > 1

Hence if (16) is satisfied, then it is possible to achieve link rate, even though the system packet capacity is less than 64 Kbytes. In order for these two conditions to occur simultaneously, the following set of inequalities hold true:

$$R_dT < B_d < 64 \text{ Kbytes} - R_dT, \quad \text{(EQ 17)}$$

i.e.,

$$R_dT < 32 \text{ Kbytes}. \quad \text{(EQ 18)}$$

which concludes the proof.

When $R_dT > 32$ Kbytes, then a value of $B_d = 64$ Kbytes - R_dT, which is less that R_dT, is sufficient to make the system packet capacity exceed 64 Kbytes, and thus attain full link rate by those means.

B. THE CASE K > 1

If k > 1, then the following two inequalities constitute necessary and sufficient conditions for the downstream throughput to equal the line rate:

$$B_d > max(R_dT + 2kB_u, 2k) \quad \text{(EQ 19)}$$
$$\text{if } R_dT \le 32 \text{ Kbytes} - 2kB_u$$

$$B_d > max(64 \text{ Kbytes} - R_dT - 2kB_u, 2k) \quad \text{(EQ 20)}$$
$$\text{if } R_dT > 32 \text{ Kbytes} - 2kB_u$$

Note that inequality (20) is identical to (10). The region in which (19) and (20) are simultaneously satisfied, has been shaded in Fig. 3. From this result it follows that (10) (or (20)) provides a tight bound for the buffer size, only if the delay bandwidth product R_dT, exceeds 32 Kbytes - $2kB_u$. It ensures

that the packet capacity of the system exceeds 64 Kbytes, so that there is no loss. An extrapolation of (10) for the case R_dT < 32 Kbytes - $2kB_u$, results in an overestimation of the amount of buffering required to attain link rate. In this range of the delay bandwidth product, it is possible to loose packets and still attain link rate, provided the inequality (19) is satisfied.

In order to derive (19)-(20), we assert that in addition to (15), the following condition is required to achieve full link throughput.

$$\frac{W_{MAX}}{2} < B_d \text{ then } \bar{F} = R_d \,. \qquad \text{(EQ 21)}$$

Note that as a result of (15) and (21), inequality (16) is automatically satisfied for k > 1. We provide the following justification for (21): There is a significant difference between the window size evolutions in TCP Reno for the case k > 1 and the case k < 1. As pointed out in [1], in the case k > 1, a single packet drop may be followed by a TCP re-transmission timer time-out, so that the window re-initializes to one segment at the beginning of each cycle, while for the case k < 1, the window only drops to half its maximum value after each packet loss. This difference is caused due to the fact that the Fast Recovery scheme in TCP Reno was designed for symmetric networks, and does not work very well for slow upstream links. It results in a burst of $W_{MAX}/2$ packets sent into the network when the ACK for a re-transmitted packet arrives. Thus if $B_d < W_{MAX}/2$, then it results in multiple packet drops which usually results in a time-out, while if (21) is satisfied, then the Fast Recovery phase does not cause any additional packet drops.

Substituting expression (4) for W_{MAX}, we arrive at the equivalent condition

$$B_d > R_dT + 2kB_u \qquad \text{(EQ 22)}$$

Hence if (22) is satisfied, then it is possible to achieve link rate, even though the system packet capacity is less than 64 Kbytes. In order for these two events to hold simultaneously, the following set of inequalities hold true:

$$R_dT + 2kB_u < B_d < 64Kbytes - R_dT - 2kB_u, \qquad \text{(EQ 23)}$$

i.e.,

$$R_dT < 32 \text{ Kbytes} - 2kB_u \,. \qquad \text{(EQ 24)}$$

which concludes the proof.

When R_dT > 32 Kbytes - $2kB_u$, then a value of B_d = 64 Kbytes - R_dT - $2kB_u$, which is less that R_dT + $2kB_u$, is sufficient to make the system packet capacity exceed 64 Kbytes, and thus attain full link rate by those means.

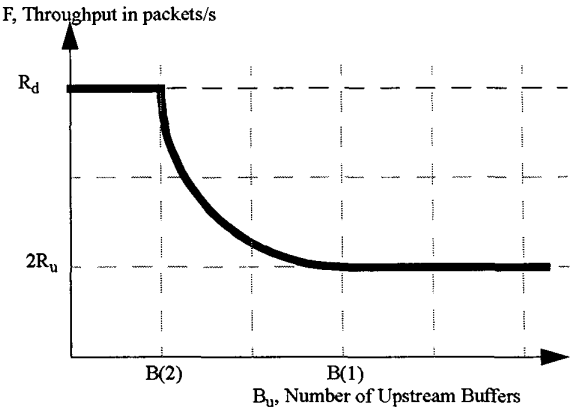

Fig. 4. Variation of downstream throughput as a function of the number of upstream buffers, for k > 1

V. Variation Of Throughput With Number Of Upstream Buffers

The criteria developed in Section IV tell us the minimum downstream buffer size required to achieve throughput equal to the link rate.One of the assumptions made in that analysis, is that in steady state, in the absence of downstream buffer overflows, the throughput equals R_d. However for the case k > 1, this assumption is not satisfied. In this section we investigate this situation in greater detail, and provide deatiled information on the variation of the throughput as a function of various system parameters. We will show that systems with k > 1 exhibit an interesting variation in performance, as the number of upstream buffers is changed (Fig. 4). There exist two numbers B(1) and B(2) (see equations (29) and (30) for a definition), such that:

- If the number of upstream buffers exceeds B(1), then the system throughput cannot exceed $2R_u$.
- As the number of upstream buffers is reduced below B(1), the system throughput gradually increases until it reaches the maximum of R_d at a buffer size of B(2).

This result shows that in order to attain full link rate, the number of upstream buffers should lie below the threshold B(2). Hence the analysis in Section IV is valid for systems with k > 1, only if B_u < B(2).

Define B'_d and B'_u as the steady state queue lengths at the downstream and upstream links respectively. The calculations in Sections V. A and V. B are based on the application of Little's Law to the system,

$$W_{LIM} = FT + \frac{FB'_d}{R_d} + \frac{FB'_u}{R_u} \quad \text{(EQ 25)}$$

or equivalently,

$$F = \frac{W_{LIM}}{T + \frac{B'_u}{R_u} + \frac{B'_d}{R_d}} \quad \text{(EQ 26)}$$

where F is the steady state throughput and W_{LIM} is the TCP window size.

The following Lemma is used in proving the results in Sections V. A and V. B:

Lemma 1:

$$\text{If } \frac{W_{LIM}}{T} < x, \text{ then F} < \text{x}. \quad \text{(EQ 27)}$$

$$\text{If } \frac{W_{LIM}}{T} \geq R_d, \text{ then either B'}_u > 0 \text{ or B'}_d > 0. \quad \text{(EQ 28)}$$

Proof: Inequality (27) follows from (26). Equation (28) follows from (26) and the fact that F cannot be larger than R_d.

We will use the following notation for the buffer thresholds that will be used in this section:

$$B(1) = \frac{W_{LIM} - 2R_u T}{2} \quad \text{(EQ 29)}$$

$$B(2) = \frac{W_{LIM} - R_d T}{2k} \quad \text{(EQ 30)}$$

A. THE CASE K > 1

(1a): If

$$W_{LIM} < 2R_u T, \quad \text{(EQ 31)}$$

then

$$B'_d = B'_u = 0 \ . \quad \text{(EQ 32)}$$

and

$$F = \frac{W_{LIM}}{T} < 2R_u \ . \quad \text{(EQ 33)}$$

Proof: From (31) and Lemma 1 it follows that $F < 2R_u$, which leads to $B'_u = 0$. Since $k > 1$, it follows that $F < R_d$, so that $B'_d = 0$. Substituting these values for the buffer sizes into (26), we obtain (33).

This result tells us that if the window size is very small, then there is no steady state queue build-up in either the upstream or downstream buffers. The rate at which ACKs fill the upstream buffer is smaller than the upstream link speed, which leads to this situation.

(1b): If

$$W_{LIM} > 2R_u T \text{ and } B_u > B(1) \quad \text{(EQ 34)}$$

then

$$B'_u = B(1) \text{ and } B'_d = 0 \quad \text{(EQ 35)}$$

and

$$F = 2R_u \ . \quad \text{(EQ 36)}$$

Proof: Note that since the upstream buffer is in steady state, its input and output rates must balance each other, from which it follows that $F = 2R_u$. Since $k > 1$, it follows that $F < R_d$, so that $B'_d = 0$. Substituting these values into (25), we obtain that $B'_u = B(1)$.

This result tells us that as the window size increases beyond $2R_uT$, a steady state backlog develops over the upstream queue, whose size is given by B(1). As long as the number of upstream buffers exceeds B(1), the system will stay in this state of low throughput, even if the window size is increased beyond R_dT.

(1c): If

$$R_d T > W_{LIM} > 2R_u T \text{ and } B_u < B(1) \quad \text{(EQ 37)}$$

then

$$B'_u = B_u \text{ and } B'_d = 0 \quad \text{(EQ 38)}$$

and

$$2R_u < F = \frac{W_{LIM}}{T + \frac{B_u}{R_u}} < R_d \qquad \text{(EQ 39)}$$

Proof: Since $W_{LIM} < R_dT$, it follows that $B'_d = 0$. If the number of upstream buffers is reduced below the threshold B(1), then this buffer overflows and ACKs begin to get dropped. This also leads to the situation in which $B'_u = B_u$. Substituting these values of buffer occupancy into (25), we obtain an equation for the resulting throughput F in (39).

Note that, by virtue of the fact that the upstream buffer is now dropping ACKs, each ACK that makes it back to the source is now ACKing k downstream packets. This allows the resulting throughput to exceed $2R_u$.

(1d): If

$$W_{LIM} > R_dT \text{ and } B(2) < B_u \leq B(1) \qquad \text{(EQ 40)}$$

then

$$B'_u = B_u \text{ and } B'_d = 0 \qquad \text{(EQ 41)}$$

and

$$2R_u < F = \frac{W_{LIM}}{T + \frac{B_u}{R_u}} < R_d \qquad \text{(EQ 42)}$$

Proof: In the previous section, since $W_{LIM} < R_dT$, it followed that $B(2) < 0$. However if $W_{LIM} > R_dT$ then $B(2) > 0$, and the result in this section tells us that even under this condition, as long as $B_u > B(2)$, equations (38) and (39) continue to hold. In order to prove this, note that in the expression for F in (39), as B_u is decreased, F increases. But as long as $B_u > B(2)$, F remains less than R_d. Thus B'_d stays at 0 and the result follows.

(1e): If

$$W_{LIM} > R_dT \text{ and } B_u \leq B(2) \qquad \text{(EQ 43)}$$

then

$$B'_u = B_u \text{ and } B'_d = W_{LIM} - R_dT - 2kB_u \qquad \text{(EQ 44)}$$

and

$$F = R_d \qquad \text{(EQ 45)}$$

Proof: If B_u decreases below B(2), then from (42) it follows that $F = R_d$. Under this condition, the steady state queue length in the downstream buffer also becomes greater than zero, and substituting for F and B'_u into (25), we obtain the expression for the steady state value of B'_d in (44).

B. The Case K < 1

(2a): If

$$W_{LIM} < R_dT \qquad \text{(EQ 46)}$$

then

$$B'_u = 0 \text{ and } B'_d = 0 \qquad \text{(EQ 47)}$$

and

$$F = \frac{W_{LIM}}{T} < R_d \qquad \text{(EQ 48)}$$

Proof: From (46) and Lemma 1 it follows that $F < R_d$, which leads to (47). Substituting these values for the buffer sizes into (25), we obtain (48).

(2b): If

$$W_{LIM} > R_dT \qquad \text{(EQ 49)}$$

then

$$B'_u = 0 \text{ and } B'_d = W_{LIM} - R_dT \qquad \text{(EQ 50)}$$

and

$$F = R_d . \qquad \text{(EQ 51)}$$

Proof: Note that even though $W_{LIM} > R_dT$, B'_u stays at 0, since $k < 1$ so that even the maximum rate into the upstream buffer is not sufficient to saturate it. Since the downstream buffer is in steady state, it follows that $F = R_d$ and substituting for F and B'_u into (25), we obtain the expression for B'_d in (50).

VI. Final Criteria

Combining the conditions in Sections III, IV and V, we arrive at following criteria for the minimum downstream buffer size required to achieve throughput equal to the full link capacity:

$$B_d \geq max(min(R_dT, 64KB - R_dT), 0) \qquad \text{(EQ 52)}$$
$$\text{if } k < 1$$

$$B_d \geq max(min(R_dT + 2kB_u, \quad \text{(EQ 53)}$$
$$64KB - R_dT - 2kB_u), 2k)$$
$$\text{and } B_u < B(2) \text{ if } k > 1$$

From Fig.2 and 3, note that a buffer size of 32 Kbytes is an upper bound to the expression on the RHS of (52) and (53), irrespective of the value of the delay-bandwidth product (provided 2k is less than 32 Kbytes for k > 1).

VII. CONCLUSION

In this paper, we were able to show that several aspects of TCP performance change significantly when operating in an asymmetric environment. Some well known "folk theorems" and rules of thumb no longer apply. In particular, the upstream buffer size and the asymmetry factor play a major role in determining the system performance and downstream buffer requirements, and this paper takes a first step in understanding and quantifying the effect of these variables.

Some of the directions in which this work can be extended, are:

- Extensions to other TCP flow control schemes: We only considered the TCP Reno flow control, which is currently the most common one on the Internet. Other schemes such as Tahoe and SACK need to be analyzed as well.
- Extension to channels with errors: We assumed that the channel was error free. More realistic channel models, such as those with iid or bursty errors need to be considered. Such an analysis would be especially relevant for the wireless broadband access case.
- Extension to shared upstream channels: This paper assumed that the upstream channel is dedicated, which is a good assumption for ADSL systems and for cable or wireless broadband access system with phone return. However for 2-way cable or wireless systems, the upstream channel is shared among multiple users, and the effect of this on TCP performance needs to be analyzed.

REFERENCES

1. T.V.Lakshman, U.Madhow and B. Suter, "Window based error recovery and flow control with a slow acknowledgment channel: A study of TCP/IP performance," *Proceedings of INFOCOM* (1997).
2. V. Jacobsen, "Berkeley TCP evolution from 4.3-tahoe to 4.3-reno," *Proc. of the 18th IETF*, (1990).
3. H. Balakrishnan, V.N. Padmanabhan and R.H. Katz, "The effects of asymmetry on TCP performance," *Proceedings of the 3rd ACM/IEEE Intl. Conf. on Mobile Computing and Networking, Budapest* (1997).
4. S. Varma, "The effect of ACK Suppression on TCP performance in asymmetric networks," *Hybrid Networks Internal Report*, (1996).