An Improved Packet Pair Method for Filtering Estimation Noise and Fast Convergence in Measuring Bottleneck Bandwidth

Han-seung Yoo, Ju-wook Jang

Department of Electronic Engineering, Sogang University

C. P. O. Box 1142, Seoul, 100-611, Korea

wamozartkr@hotmail.com, jjang@ccs.sogang.ac.kr

Abstract

A new scheme is proposed to speed up a known bandwidth measuring method which employs potential bandwidth for filtering out noises (in estimation) from time compression caused by a packet queuing ahead of two probe packets. Instead of incrementing the potential bandwidth by a fixed amount as in the original method we increase the potential bandwidth exponentially for faster convergence. To retain its filtering capability as well as its agility to adapt to new bottleneck bandwidth, each trial potential bandwidth(PB) is adjusted using MAX and MIN as upper bound and lower bound. An experiment using known bandwidths shows 45-89% improvement in convergence time.

Keywords: *Internet, path, Packet Pair, Estimation, measuring bandwidth*

This work was supported by the Korea Science and Engineering Foundation under Grant #97-0102-02-01-3.

1. Introduction

Many applications on Internet would benefit from accurate bottleneck bandwidth measurement. For example, network clients can choose a server from a server pool based on the bottleneck bandwidth of the path leading to the server.

Packet pair method[1] uses the observation that the interval between two packets spread out after passing through the bottleneck link. The spacing and size of the packets are used to estimate the bottleneck bandwidth.

One main problem with the packet pair method is how to filter out noise from time expansion or time compression caused by a third packet queuing in between or ahead of two measure packets. A well known method to solve this problem is to use a histogram. However it is hard to determine appropriate bin size without a priori knowledge of the sample distribution. Lai and Baker[2] filters out the noise by using the potential bandwidth which denotes the bandwidth at which the pair of measuring packets are sent. Samples with higher bandwidth than the potential bandwidth(PB) are considered as time compressed and hence filtered out. One problem with the method(called PBF hereafter) is that it may take many round trips for PBF to arrive at actual bandwidth since it increments the PB by a fixed amount at a time. If the potential bandwidth is incremented by m kbps, then it takes more than B/m round trips. Since bottleneck bandwidth may change rapidly as many applications join and leave the network in random fashion, the agility to converge to a new bottleneck bandwidth is very important for online utilization of the estimation.

In this paper, this problem is addressed to achieve faster convergence. Instead of just incrementing the potential bandwidth by a fixed amount as in the original method we increase or decrease the potential bandwidth exponentially for faster convergence. Once PB gets near actual bandwidth, in order to refine measurements, the amount of increment is adjusted(narrowed). The proposed algorithm also provides a mechanism for fast detection of any (positive or negative) change in the actual bandwidth by adjusting the range in which PB is allowed to reside. Experiments using known bandwidths shows 45-89% improvement in convergence time.

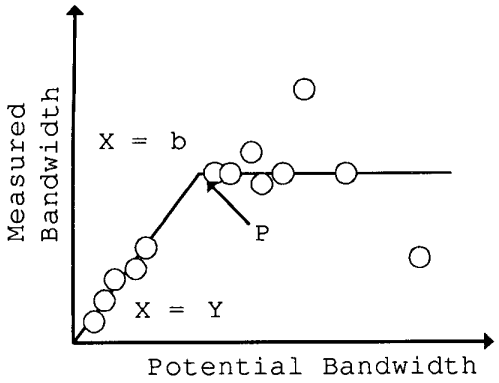

Figure 1. Circles represent measured bandwidth samples. All samples over line x=y are considered time compressed and filtered out. Initially measured bandwidth grows linearly with potential bandwidth, but begins to saturate at y=b. The b is considered as the bottleneck bandwidth

2. The Packet Pair Method using PBF

Lai and Baker[2] filters out the noise by using the potential bandwidth which denotes the bandwidth at which the pair of measuring packets are sent. Samples with higher bandwidth than the potential bandwidth(PB) are considered as time compressed and hence filtered out. By plotting the measured bandwidth(MB) with ascending values of potential bandwidth as in Figure 1, one can determine the "knee" at which measured bandwidth begins to saturate notwithstanding the increasing potential bandwidth. All samples over line x=y in Figure 1 are considered time compressed and filtered out. Initially measured bandwidth grows linearly with potential bandwidth, but begins to saturate at y=b. The b is considered as the "bottleneck bandwidth". The relative error of each sample from x=y line or y = b is normalized to prevent samples with large error from dominating the calculation.

One problem with the method is its slowness to arrive at actual bandwidth.

Figure 2 shows an experiment illustrating the number of steps (round trips) for measured bandwidth (which is upper bounded by PBF to filter out time compression noise) to arrive at actual bandwidth which is set to 100kbps. It takes as many as 55 steps if we use an increment of 2kbps.

3. Proposed approaches

One immediate solution to the slow convergence in the PBF method is to exponentially increase or decrease the PB.

Figure 2 shows only 6 steps are needed with exponential growth of increment/step starting from as small as 1kbps.

The exponential growth finishes when PB exceeds MB. However, the gap between PB and MB may be larger than in the original PBF method as a result of exponential increase of PB. Since the ability of the PB to filter out the noise from time compression is degraded as the gap between the PB and MB grows, the exponential growth in increment will be more susceptible to time compression than the original PBF method.

To remedy this we suggest exponential decrease of PB after it grows above MB and exponential growth of PB when PB goes below MB again. The gap between PB and MB gets narrow as shown in Figure 3. This remedy, however, also has its drawbacks. Once PB gets equal to MB, it is hard to detect improved bottleneck bandwidth. Through these initial experiments, two conflicting goals are identified:

1) Filtering out noise from time compression (Small gap between PB and MB is favored)
2) Fast detection of change in bottleneck bandwidth (Large gap between PB and MB is favored)

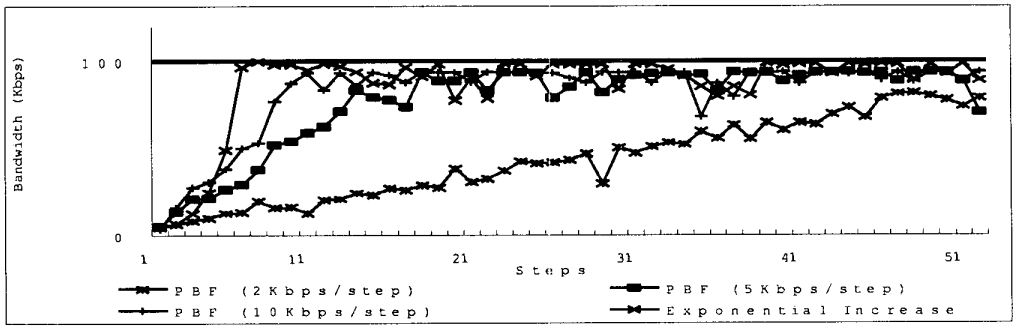

Figure 2. PBF method is slow in convergence to new bandwidth method

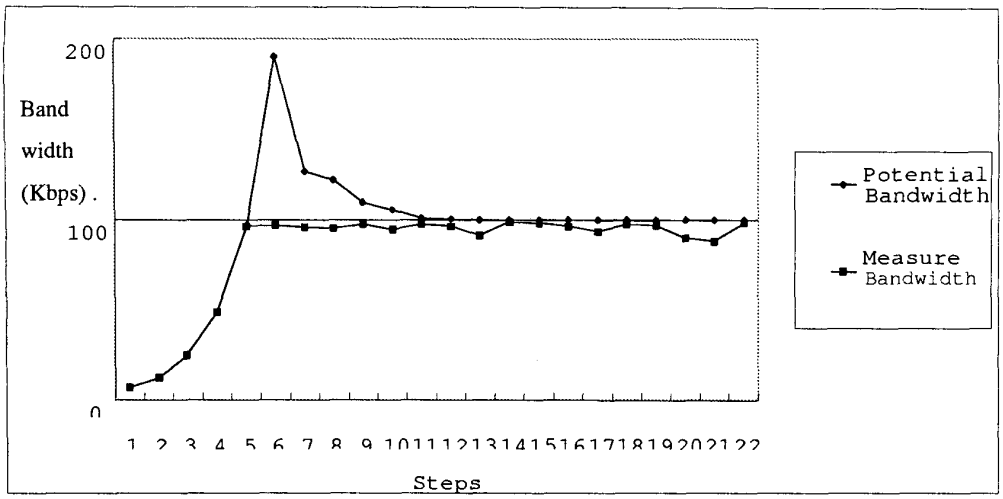

Figure 3. PB is exponentially increased/decreased as it passes MB

We tried two schemes to achieve these goals. Method 1 restarts the exponential growth of the PB after some fixed time to avoid situation where PB upper bounds MB in such a way that positive change in bottleneck bandwidth goes unnoticed. Method 2 restarts the exponential growth of PB when MB gets very close to PB or MB goes far below PB. In our experiment, it is observed that method 1 takes time to adapt to new improved bottleneck bandwidth. Method 2 is faster but it keeps PB high above MB, undermining its filtering capability.

This observation leads us to method 3. While keeping PB high above MB enables quick detection of improved bottleneck bandwidth, it is also necessary to reduce PB for reduced bottleneck bandwidth in order to retain PB's filtering capability. To achieve two contradictory goals, we devised the following algorithm which we call method 3. Four variables, MAX, MIN, PB and MB, are used to determine trial potential bandwidth(PB). The basic idea behind this algorithm is to guide PB using MAX and MIN as the upper bound and the lower bound. The MAX, MIN are adjusted as PB or MB touches them and PB is set to new MAX(See Figure 4):

1) IF MB touches MAX, PB is doubled for exponential growth of PB or fast detection of bandwidth improvement
2) IF MB touches MIN or gets close to MIN, MAX is set to MIN and MIN is halved and PB is set to new MAX to better filter out time compression noises.
3) IF MB is in upper half of the range bounded by MAX and MIN, the range is halved upward MAX by setting MIN to (MAX+MIN)/2
4) IF MB is in lower half of the range bounded by MAX and MIN, the range is halved downward by setting MAX to (MAX+MIN)/2
5) After adjusting MAX or MIN as above, PB is set to MAX

MAX and MIN represent the upper and lower bound for trial potential bandwidth(PB). The difference between MAX and MIN is used to determine the increment(or decrement) in PB. We illustrate how in our algorithm PB grows fast to arrive at a known bandwidth at start and adapts to any subsequent change. Initially, MAX, MIN and PB are all set to 1 kbps (One can change this initial starting bandwidth to suit his or her networking environment). A pair of packets spaced by 1 kbps is sent and received to calculate the MB. If the MB is larger than the PB, then it is considered time compression and discarded. If it is equal or very close to the PB, MIN is set to MAX. Then MAX is doubled and PB is set to the new MAX. In this way, the MAX, MIN and PB are increased exponentially. At some point, MB goes below PB indicating saturation. The MB is compared against (MIN+MAX)/2. If MB is between (MIN+MAX)/2 and MAX, MIN is set to (MIN+MAX)/2 and PB is set to MAX. If MB is between (MIN+MAX)/2 and MIN, MAX is set to (MIN+MAX)/2 and PB is set to MAX. As a result, the range for PB is halved to improve the filtering capability. If MB is equal or lower than MIN, it is very likely that actual bottleneck bandwidth has dropped. To converge to new

bottleneck bandwidth, MAX is set to MIN, MIN is halved and PB is set to MAX.

```
MB : Measured Bandwidth
PB : Potential Bandwidth
MAX : Maximum of the Range
MIN : Minimum of the Range

MAX, PB = a
MIN = 0

Do
        If MB > PB Then
        Error "Time Compression"
        Else If (MB - PB) < b Then
                MIN = MAX
                MAX = 2 * MAX
                PB = MAX
        Else If MB < MIN Then
                MAX = MIN
                MIN = MAX / 2
                PB = MAX
        Else
IF MB > (MAX + MIN) / 2 Then
        MIN = (MAX + MIN) / 2
   Else
        MAX = (MAX + MIN) / 2
        PB = MAX
   Endif
Endif

End do
```

Figure 4. The 3rd method

4. Experimental results

Three Pentium-II PC running Linux is used. Two PCs are configured as sender and receiver, respectively. One PC simulates a bottleneck link with two routers at both ends.

It is programmed to throttle the packet as dictated by predetermined bottleneck change curve(broken line in Figure 5).

Figure 5 compares the three proposed methods against the original PBF method(we implemented the method based on its algorithmic description). Actual bandwidth experiences quantum improvement around step 10 and method 2 takes 6 more steps to catch up the change. Method 2 is much faster at step 10 and close to method 1 at step 36. One fact hidden from the figure is that during steps 30-39 both method 1 and method 2 keeps PB high above MB, which undermines its filtering capability when bottleneck bandwidth is reduced. Some time compression noise might not be filtered out, causing fluctuation in estimated bandwidth curve. Method 3 reduces this problem by exponentially decreasing PB on quantum drop of MB to close the gap. It is also fastest in catching up the improved bandwidth. Figure 6 plots the MAX and the MIN which guides the PB. MAX(hence PB) is maintained slightly above actual bandwidth except step 19 and 27 which vindicates its filtering capability. Notice also actual bandwidth resides between MAX and MIN almost of the duration. 45% ~ 89% improvement in convergence time is observed in an experiment with bottleneck bandwidth curve starting from 50kbps and changing to 90kbps, 20kbps and 65kbps.

5. Conclusions

An improved scheme to measure the bottleneck bandwidth using a pair of packets with potential bandwidth for an end-to-end path on Internet is proposed. It shows greater agility in adapting to change in bottleneck bandwidth than the original method[2] while retaining the capability of filtering out noise from time compression comparable to the original method. While the former goal favors large gap between PB and MB, the latter calls for small gap. We satisfied two contradictory goals by introducing two variables, MAX and MIN which guides adaptive update of PB. Experiment with known bandwidth shows 45% ~ 89% improvement in convergence time

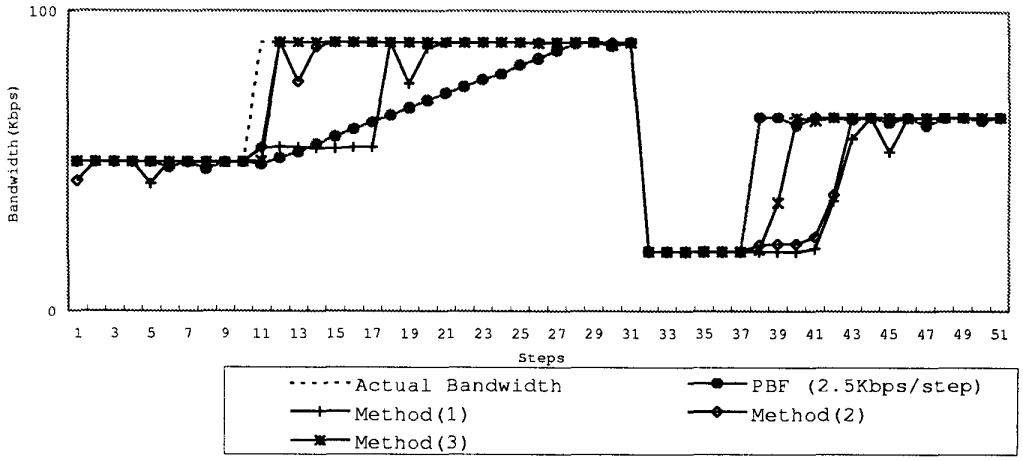

Figure 5. Comparison of proposed methods against the PBF method[2]

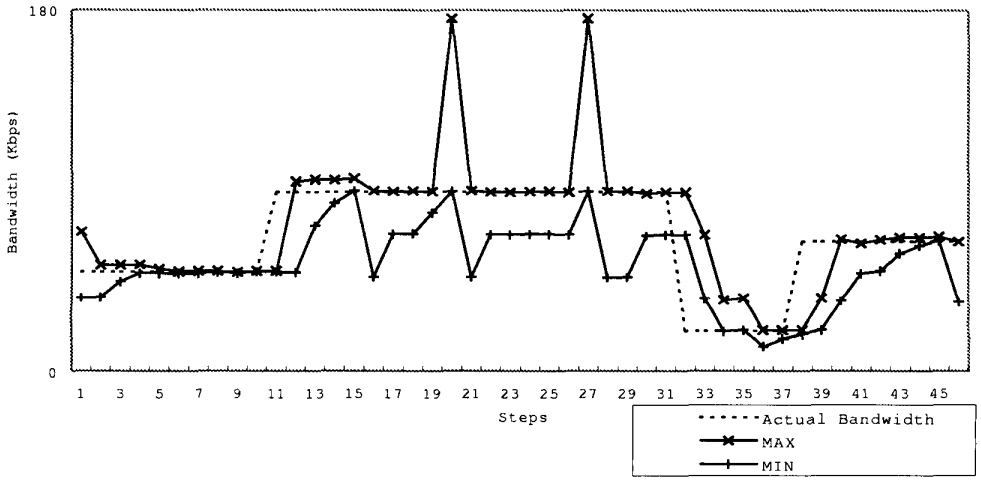

Figure 6. MAX, MIN are used to guide PB

6. References

[1] Jean-Chrysostome Bolot, "End-to-end packet delay and loss behavior in the Internet," Proc. of SIGCOMM pp. 289-298, 1993.

[2] Kevin Lai and Mary Baker, "Measuring bandwidth," Proc. Of INFOCOM, pp. 1999

[3] Vern Paxson, "End-to-end Internet packet dynamics," Proc. Of SIGCOMM, 1997

[4] Srinivasan Keshav, "A control-theoretic approach to flow control," Proc. Of SIGCOMM, 1991

[5] Robert L. Carter, Mark E. Crovella, "Measuring bottleneck link speed in packet-switched networks," Technical report BU-CS-96-007, Boston University, 1996

[6] Vern Paxson and Sally Floyd, "Why we dont know how to simulate the Internet?," In Proc. Of Winter Simulation Conference, 1997

[7] Srinivasan Seshan, Mark Stemm and Randy Katz, "Spand: Shared passive network performance discovery," In Proc. Of USENIX Symposium on Internet Technologies and Systems, 1997