PATHMON, A Methodology for Determining Available Bandwidth over an Unknown Network

D. Kiwior, J. Kingston and A. Spratt
The MITRE Corporation
202 Burlington Road
Bedford, MA 01730-1420

***Abstract*-Interest in measuring available bandwidth has increased in response to the need for fast, accurate, non-intrusive estimation of current network conditions. Several methods have been proposed in recent years with varying degrees of success. This paper presents the details of the algorithm and implementation of PathMon, an active probe method which requires no prior knowledge of the network and no management control over the network to provide an end-to-end measurement of available bandwidth. In comparison to other available bandwidth methods, experiments show that PathMon is more accurate, and converges more quickly with less overhead.**

I. Introduction

In recent years, interest in estimating the available bandwidth of a network path has been motivated by many network applications including QoS, overlay networks, adaptive and grid applications. QoS functions of admission control [1] and validation of service level agreements need a fast, minimally complex method to evaluate network performance [2, 3]. Routing decisions to support QoS and overlay networks are optimized by knowledge of resource availability in the network. Adaptive applications need to know current network conditions in order to modify their output and maximize network use [4]. For optimal performance, applications for grid computing and distributed computing must consider available bandwidth when selecting hardware resources [5, 6]. Since available bandwidth is a dynamic quantity, these applications require a fast, accurate measurement of available bandwidth. In addition, the measurement must be lightweight and non-intrusive to prevent additional network load. To address these needs, a variety of available bandwidth estimation tools, including pathload [7], pathChirp [8], and Initial Gap Increasing (IGI) [9] have been developed.

This paper presents PathMon, a new, more efficient method of estimating available bandwidth. Similarly to the tools listed above, PathMon uses a packet train of probe packets. In contrast to these tools, PathMon uses a single train with a simple statistical evaluation that eliminates insignificant data.

In comparison to the existing tools listed above, PathMon has been show in experiments to converge to an estimate of available bandwidth with less than 12% relative error. The PathMon algorithm also requires less overhead than the other methods, with 41 Kbytes of traffic and an average latency of 0.19 seconds. PathMon satisfies the need for a computationally simple method to provide a fast, accurate available bandwidth measurement that can be used for critical short-term decisions.

II. Related Work

The term bandwidth has traditionally referred to a static measure of capacity, the maximum amount of data that can be transmitted over a link or path. Available bandwidth is a more difficult quantity to measure since it is a dynamic quantity, the amount of traffic that can be transmitted over a link or path, given current traffic conditions. For an end-to-end path composed of multiple links, the path's bandwidth is limited by the link with the least capacity, referred to as the bottleneck or narrow link [7]. While the narrow link provides an upper bound of the available bandwidth of the path, it may not correspond to the tight link, i.e. the link that has the least available bandwidth under current conditions.

Available bandwidth has been successfully measured with packet train probes. As the probe packets pass through congested links, they are delayed by cross traffic. In theory, when the probe train's transmission rate exceeds available bandwidth, a queue builds up in the network devices. This results in increased interpacket intervals at the receiver in comparison to the interpacket intervals at the source. In reality, due to bursty traffic and network devices, queuing delays may be seen even without congestion and may not increase monotonically in the presence of congestion. Analysis of the interpacket intervals at the receiver must take this queuing variation into account to identify the true point of congestion, from which available bandwidth can be calculated.

Pathload sends multiple trains (fleets), each train consisting of 1200 same-size packets, with the same interpacket time gaps. After each fleet is sent, statistical evaluation of the interpacket time gaps at the destination compared to the interpacket time gaps at the sender determines whether the trend of the interpacket time gaps in that fleet has been increasing, non-increasing, or undeterminable. The transmission rate for the next fleet is adjusted based on the trend of the current fleet until the results converge. Pathload's use of a large number of packets allows statistical analysis to

eliminate insignificant queuing delays but increases the network load.

Several tools that avoid the high costs of multiple fleets have followed pathload. The method proposed in [10] transmits a single packet train of ICMP probe packets with decreasing time delays between each packet so that each packet requires a higher bandwidth than the previous. The algorithm presented in [10] compares the sending curve (transmission time of the probe packet vs. packet number) to the receiving curve (reception time of the probe acknowledgement vs. packet number). Using trend lines to compensate for fluctuations in the receiving curve, the point where the receiving curve diverges from the sending curve marks the congestion point, from which available bandwidth can be calculated. The pathChirp and Initial Gap Increasing (IGI) algorithms incorporate a statistical evaluation of the destination interpacket gaps to determine the true point of divergence. In addition, pathChirp and IGI use UDP probe packets rather than ICMP packets, which may be dropped or rate-limited at network devices. PathChirp measures the destination's interarrival times and identifies regions where the time intervals show an increasing trend for an extended period of time. The regions are statistically evaluated to determine the available bandwidth. IGI assumes that the bottleneck bandwidth of the path is known and further assumes that the bottleneck link is also the tight link, with ambiguous results when this is not the case. The IGI algorithm first measures competing traffic by calculating the difference between source interpacket gaps and destination interpacket gaps. Available bandwidth is then calculated by subtracting competing traffic from the bottleneck bandwidth.

Each method has disadvantages in satisfying the requirements of fast, accurate and non-intrusive. Pathload has high overhead in terms of convergence time and probing traffic. PathChirp's traffic overhead is less intrusive than pathload's but still substantial. In addition, pathChirp's computationally complex algorithm does not converge quickly. IGI's assumptions that bottleneck bandwidth is known and that the bottleneck bandwidth corresponds to the narrow link limits its usefulness.

III. PATHMON'S BANDWIDTH MEASUREMENT ALGORITHM

To measure available bandwidth we use an active measurement method where a series of time stamped UDP packets are injected into the network from the monitor application and received at the agent application. The intermediary network nodes between the monitor and agent applications are considered to be a "black box" for the purpose of bandwidth measurement. Thus the available bandwidth measurement is a measure of aggregate bandwidth available to the application from "end to end".

The PathMon algorithm measures one-way delay from the monitor application to the agent application. In general, if there is no congestion in the network, the interpacket intervals seen at the agent should be the same as those at the monitor. However, in a congested network each packet received at the agent will show increasing delay. By varying the bandwidth requirements of the packets sent, the algorithm uses this property to identify a bandwidth requirement that causes congestion in the network.

This model, however, is too simplistic for real networks where processing overhead and device characteristics can cause variations in delay. This "jitter" will necessarily lead to inaccuracy in the measurement. To counteract this inaccuracy our algorithm uses a two step measurement. The first step measures jitter and allows for statistical analysis of network delay. The second step calculates the delay measurement using cumulative packet delay intervals.

A. Jitter Measurement

During the jitter measurement stage, the monitor application sends a series of N_J equally spaced, same size packets. The rate at which this "train" of packets is sent should be equal to the lower bound of the bandwidth measurement range. The number of packets sent (N_J) is application dependent and should be large enough to provide a good statistical sample. However, if N_J is too large, the network's traffic load will increase as will the total measurement time. The agent application records the interarrival time gap of each of the N_J packets. This information is then used to calculate the mean interarrival jitter and the standard deviation.

B. Bandwidth Measurement

The available bandwidth is measured by sending N_B time stamped packets of equal size and decreasing time interval from the monitor application to the agent application. The decreasing interpacket interval corresponds to increasing instantaneous bandwidth requirement for each packet. This creates a train of packets of increasing bandwidth requirements evenly spaced between a lower and upper bound. The value of N_B is application specific and should be chosen such that it provides sufficient intermediary bandwidth steps to reach the desired resolution. The instantaneous bandwidth requirement (B) for a packet can be calculated from its size (s) divided by the time until the next packet is sent (t_{NP}), as

$$B = s / t_{NP}.$$

It should be noted that the monitor application could have also been implemented by varying the packet size and leaving the interpacket interval constant without changing the result.

The algorithm is based on identifying the point of divergence between the interpacket delays measured at the agent with those of the monitor. Fig. 1 illustrates the difficulty in determining the point of divergence when small delay variations result from context switches or queuing delays unrelated to congestion. To detect significant interpacket interval variations more easily, PathMon records all time values for packets in the train in terms of cumulative time, shown in Fig. 2.

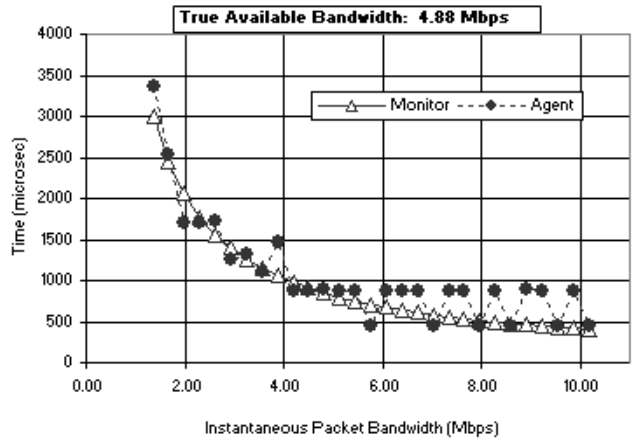

Figure 1. Interpacket Delay Intervals

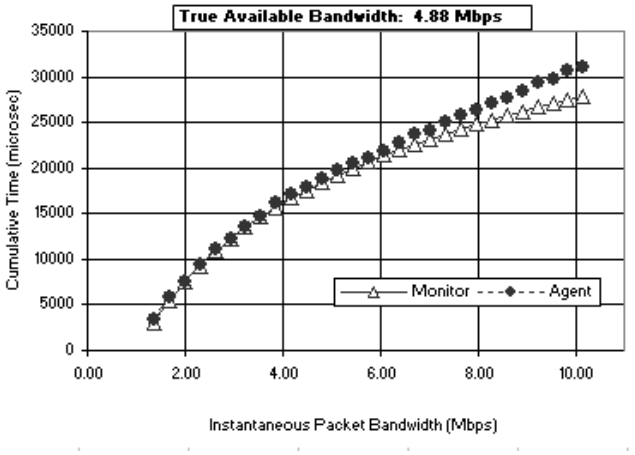

Figure 2. Cumulative Interpacket Delay Intervals

C. *Measurement Results Analysis*

The first step in analyzing the results is to check the endpoints to make sure they are correct. Specifically the lower bound endpoints must not be congested, while the upper bound endpoints must be congested. If both of these conditions are not met then the actual available bandwidth point falls outside the measurement range.

If the lower bound endpoints are congested, then the actual available bandwidth point is below the lower bound of the measurement and can not be determined from the measurement data. To check this, the algorithm examines the timestamps for the packets that were sent in the jitter measurement stage. If the delay of these packets is increasing over the whole measurement, then the lower bound is congested.

To check the upper bound end points, we use the jitter statistics that were calculated in the jitter measurement stage. The algorithm first checks that the delay for the last packet in the train is greater than the average jitter plus 2 standard deviations. If this is true the last M packets of the train are checked to make sure the delay is increasing over their range. If these conditions aren't met then the actual available bandwidth point is greater than the upper bound of the measurement.

If all the end point conditions are met, then the available bandwidth point lies within the measurement range. If not, the measurement should be repeated with a new set of boundary conditions. Once the end point conditions are met, the algorithm identifies the congestion point by starting at the upper bound endpoint and traversing backwards over the timestamp information for each packet in the train comparing the measured delay to the measured jitter statistics. The congestion point corresponds to the packet that has a time difference greater than the average jitter but is preceded by a packet with a time difference less than the average jitter. The instantaneous bandwidth requirement of the packet preceding the congestion point is the available bandwidth.

IV. Comparison to Other Available Bandwidth Tools

Lab experiments to compare the performance of PathMon to existing tools, including pathload, pathChirp and IGI, were carried out in an isolated subnet shown in Fig. 3. Cross traffic was generated by a SmartBits 600 so that the true available bandwidth was known. Each tool's accuracy and efficiency in terms of time required and number of bytes used was evaluated. In addition, the accuracy of each method was evaluated by calculating the average error of the measurements relative to the true available bandwidth.

PathMon was configured with the jitter train consisting of 10 packets and the measurement train consisting of 30 packets. The size of each packet was 1 KB and the measurement range was 1 MB to 15 MB.

The metrics recorded for each experiment include overhead bandwidth, the amount of traffic in kilobytes used by the tool in determining the estimate; latency, the time in seconds for the tool to report an available bandwidth estimate; and the measured available bandwidth. Each tool was tested by injecting different amounts of SmartBits 600 traffic into the shared link, thus creating known available bandwidths of 2.44, 4.88, 7.26, 8.78, and 9.77 Mbps. Results of the experiments are presented in Fig. 4 with overhead bandwidth shown in Fig. 4(a) and average latencies in Fig. 4(b). Accuracy of the tools in terms of the average relative error is shown in Fig. 4(c).

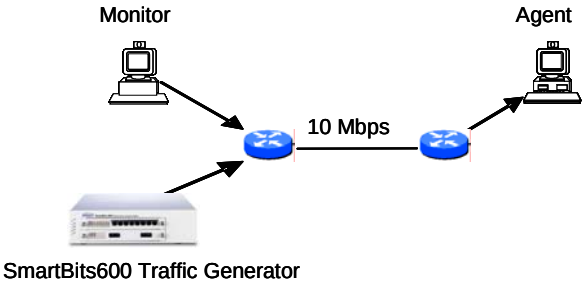

Figure 3. Testbed Configuration

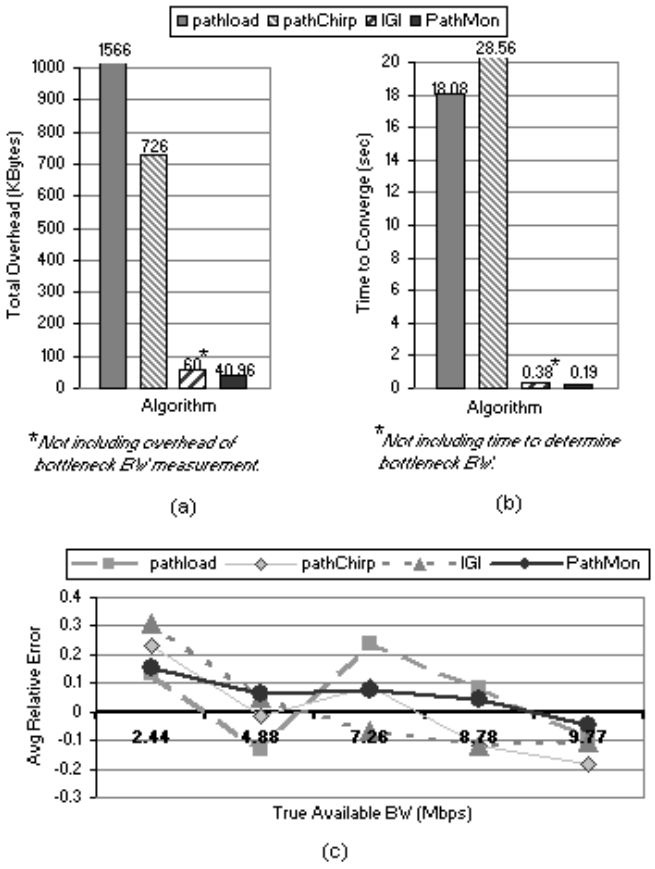

Figure 4. Comparison of Available Bandwidth Measurement Tools

V. Results and Conclusions

PathMon was clearly shown to be accurate and efficient in reporting path characteristics. It requires less than 41 kilobytes of probe traffic and less than .25 second to report available bandwidth. All tested measurement tools demonstrated accuracy within 30% of the true available bandwidth with least accuracy in measurements of the 2.44 Mbps true available bandwidth, where relatively small variations result in a significant relative error. With the exception of the 2.44 Mbps bandwidth, PathMon's measurements were within 8% of the true available bandwidth for the range of bandwidths tested. Of the other tools, IGI's performance was closest to PathMon's with accuracy within 12% (without the 2.44 Mbps bandwidth) of the true available bandwidth, minimal overhead and a short convergence time. However, due to the nature of the IGI software, the values measured for overhead and convergence of the IGI algorithm do not include the packet overhead and time required to determine the bottleneck bandwidth, which the IGI algorithm assumes to be known. Furthermore, IGI presumes that the bottleneck bandwidth corresponds to the tight link, i.e. the link in a path with the least available bandwidth. PathMon makes no such assumptions and does not require a priori knowledge of the network. When compared to other probe train algorithms, PathMon requires considerably less traffic, resolves the measurement in the least time and provides accurate measurement across a range of bandwidths.

References

[1] L. Breslau, E. W. Knightly, S. Shenker, I. Stoica, and H. Zhang, *"Endpoint Admission Control: Architectural Issues and Performance,"* In Proceedings of the ACM SIGCOMM, 2000.

[2] V. Elek, G. Karlsson, and R. Ronngren, *"Admission Control Based on End-to-End Measurements,"* In Proceedings of the IEEE INFOCOM 2000.

[3] V. Firou, J. Le Boudec, D. Towsley, and Z. Zhang, "*Theories and Models for Internet Quality of Service*", In Proceedings of the IEEE, Special Issue on Internet Technology, August 2002.

[4] M. Stemm, R.Katz, and S. Seshan, "*A Network Measurement Architecture for Adaptive Applications*," In Proceedings of the IEEE INFOCOM 2000.

[5] C. A. Lee, J. Stepanek, C. Kesselman, R. Wolski, and I. Foster, "*A Network Performance Tool for Grid Environments*," Supercomputing '99, 1999.

[6] B. B. Lowekamp, "*Combining Active and Passive Network Measurements to Build Scalable Monitoring Systems on the Grid*," ACM Performance Evaluation Review, 30(4):19-26, March 2003.

[7] M. Jain and C. Dovrolis, "*Pathload: A Measurement Tool for End-to-end Available Bandwidth*," In Proceedings of the 3rd Passive and Active Measurements Workshop, March 2002, Fort Collins CO.

[8] V. Ribeiro, R. Riedi, R. Baraniuk, J. Navratil, and L. Cottrell, "*pathChirp: Efficient Available Bandwidth Estimation for Network Paths*," In Proceedings of the 4th Passive and Active Measurements Workshop, April 2003, La Jolla, CA.

[9] N. Hu and P. Steenkiste, "*Evaluation and Characterization of Available Bandwidth Probing Techniques*," IEEE Journal on Selected Areas in Communications, August 2003.

[10] J. He, Y. Lu, C. Chow, and T. Chujo, "*Available Bandwidth Measurement, Implementation, and Experimentation*," ICC 2002.