

Experimental and Analytical Evaluation of Available Bandwidth Estimation Tools

Cesar D. Guerrero¹ and Miguel A. Labrador
University of South Florida
Department of Computer Science and Engineering
Tampa, Florida 33620
{cguerrer, labrador}@cse.usf.edu

Abstract

In this paper, we present a low cost and flexible testbed to evaluate the performance of available bandwidth estimation tools in a common and controlled environment. In addition, we include a model based on a network of M/M/1 queues to have an analytical reference to compare the experiments with. We then utilize the proposed testbed and model to evaluate the performance of Pathload, IGI, and Spruce, three well-known bandwidth estimation tools. Our main results indicate that Pathload is the most accurate tool but the slowest to converge. IGI, on the other hand, is the fastest tool but the least accurate. Spruce is the least intrusive tool with intermediate accuracy and convergence time.

1 Introduction

The available bandwidth of an end-to-end path is a time-varying metric related to the individual utilization of each link throughout the path. If we define τ as the *averaging timescale* of the available bandwidth [11], the average utilization for a sample of time τ , is given by

$$u_i(t, t + \tau) = \frac{1}{\tau} \int_t^{t+\tau} u_i(t) dt \quad (1)$$

where $0 \leq u_i(t, t + \tau) \leq 1$. For a link i with capacity C_i , the available bandwidth of the link in the interval $(t, t + \tau)$ can be defined as the average non-utilized capacity during the time τ . That is,

$$A_i(t, t + \tau) = C_i[1 - u_i(t, t + \tau)] \quad (2)$$

For an end-to-end path with H hops, the available bandwidth is given by the minimum non-utilized link in the path, as follows:

$$A(t, t + \tau) = \min_{i=1..H} A_i(t, t + \tau) \quad (3)$$

In the literature, the link with the minimum capacity is called the *narrow link* and the link with the minimum available bandwidth is called the *tight link*, which is considered the bottleneck of the path and the link that determines the end-to-end available bandwidth.

Two main available bandwidth estimation approaches have been reported. The first approach uses the *Probe Gap Model* (PGM), which bases the estimation on the gap dispersion between two consecutive probing packets at the receiver. Examples of tools in this category are *Delphi* [12], *Spruce* [16] and *IGI* [4]. The second approach is based on the idea of *induced congestion*, in which the turning point (available bandwidth) is determined by the variation in the probing packet rate from sender to receiver. *TOPP* [10], *Pathload* [8] and *Pathchirp* [13] are examples of tools utilizing this approach. In this paper, we selected Pathload, IGI and Spruce in our evaluation as they are important tools in each category.

Available bandwidth estimation tools have been mostly evaluated according to three performance metrics: *accuracy*, *overhead* and *convergence time*. That is, how accurate the tool estimate is compared with the real pre-known or analytical value, how much probe traffic the tool needs to inject into the network to be able to perform the estimation, and how much time it takes the tool to provide the estimate. In these evaluations, previous work utilized either simulation tools and/or real networks, such as links over the Internet. Although simulation tools offer a good alternative, the use of real networks is the preferred scenario. However, using real networks is very challenging and costly, as many resources need to be controlled and known in advance in order to make meaningful and valid experiments. In this pa-

¹Cesar D. Guerrero was supported in part by the Universidad Autonoma de Bucaramanga, Colombia.

per, we propose a low cost and flexible testbed to evaluate these tools and techniques over a real but fully controlled network. This testbed utilizes real networking equipment and specialized software that allows researchers to test these tools several times under different networking scenarios and conditions. In addition to the testbed, we propose an analytical model based on network of queues that can be utilized to obtain reference values in order to compare them with the experimental results. This end-to-end analytical model has not been used in the literature of available bandwidth estimation tools before.

As a demonstration of the capabilities of the testbed and the analytical model, we evaluate the performance of Pathload, IGI and Spruce using the performance metrics defined above. Our experience with the testbed and the analytical model indicates that they are in fact excellent mechanisms to evaluate available bandwidth estimation tools. As regards to the tools, we found that Pathload is the most accurate tool but the slowest to converge. IGI, on the other hand, is the fastest tool but the least accurate. Spruce is the least intrusive tool with intermediate accuracy and convergence time.

The remainder of the paper is organized as follow. Section 2 briefly presents the related work on current evaluations of available bandwidth estimation tools plus a brief background on Pathload, IGI and Spruce, since these are the tools to be evaluated. Sections 3 and 4 describe the testbed and the analytical model. Section 5 presents the results of the performance evaluation. Finally, Section 6 presents conclusions and future work.

2 Related Work

Several papers have evaluated available bandwidth estimation tools using real networks. For example, in [15], Shriram *et al.* utilized a high-speed testbed to evaluate *Abing*, *Pathchirp*, *Pathload* and *Spruce*. They used passive monitors to verify the actual load level of the generated traffic and tested the tools using links of OC-48 and 1 Gbps capacities. The problem with these experiments is that they just provided a partial picture of the evaluation, as the researchers did not have the capability of working with links of different capacities. In [9], Lee *et al.* describe problems with some bandwidth estimation tools when used on the *Planetlab* infrastructure. Since the capacity of the links was unknown to the researchers, they had to use *Pathrate* [3] to measure the end-to-end capacity of the links. The problem is that the associated error incurred by *Pathrate* in the estimation of the link capacities introduces errors in the final estimation of the available bandwidth. In [2], Angrisani *et al.* evaluated *IGI*, *Iperf* and *Pathload* over a local area network and used MGEN [1] as a traffic generator. This evaluation suffers similar flexibility problems like the ones

found in [15]. Also, the authors did not measure the overhead generated by the tools.

There are several differences between our work and previous works. First, we used a very low cost and flexible testbed that allowed us to vary the bandwidth, delay, packet loss rate, queue size and amount of cross traffic in a controlled manner. Second, we included an analytical model to fairly compare the performance of the tools with a theoretical value. As far as we know, this is the first work that provides these theoretical references in an end-to-end manner. Finally, we developed routines to fully automate the testing process allowing us to perform several times the same experiment and obtain statistically valid average results and confidence intervals.

2.1 Pathload

Pathload [8] is a tool used to estimate the available bandwidth of an end-to-end path using the principle of Self-Loading Periodic Stream (SLoPS) [7]. SLoPS is based on the fact that the one way delay of a periodic packet stream increases when the rate of the probing traffic is higher than the available bandwidth in the path. Otherwise, there is no increase in the delay measured. A fleet of streams (of a fixed number of packets each) are sent at varying rates and the one way delay trend of each stream is then characterized at the receiver as either increasing or decreasing. When that delay is in a *grey region* (not clearly increasing nor decreasing), the methodology presents a variation range of the available bandwidth. *Pathload* sends periodic packet streams of UDP traffic and uses a TCP connection to send trend results back to the sender. Given a desired stream rate R , Pathload sets the packet inter-departure time T at $100 \mu s$ and calculates the necessary packet size L to satisfy $R = L/T$. If the L is less than 96 bytes, Pathload uses this minimum value and calculates T instead.

2.2 IGI

IGI [4] is a tool that works using the *Probe Gap Model*. Under this model, the amount of cross traffic is estimated because it is a function of the amount of traffic inserted between a packet pair. IGI finds an initial probing gap value so that a probing packet train interacts with the cross traffic in a non empty narrow link queue, which is called by the authors the *Joint Queuing Region* (JQR). In that region, there is a proportional relation between the gap when probing packets leave the queue (output gap) and the cross traffic. When the initial gap is increased and equal to the output gap, the available bandwidth on the narrow link is equal to the average rate of the packet train. After that point, called the turning point, the narrow link will be overflowed by the probing packets. It is shown by the authors that in the case

of multiple hops and significant cross traffic following the tight link, the accuracy of IGI suffers. A similar situation is found when the tight link is not the narrow link.

2.3 Spruce

Spruce [16] also uses the *Probe Gap Model*. It sends a Poisson sample of 1500B UDP pairs of packets with an intra-pair gap equal to the narrow link transmission time of a 1500B packet. That guarantees that the second packet arrives to the narrow link queue before the first packet leaves that queue. Using the dispersion of the probe packets measured at the receiver, Spruce calculates the average rate of the traffic that arrives to the queue between the two packets. The available bandwidth is determined by subtracting that cross traffic rate from the capacity in the bottleneck link. Spruce estimation requires a previous calculation of the tight link capacity.

3 Testbed Description

In order to experimentally evaluate available bandwidth estimation tools, we setup the network testbed shown in Figure 1. This is a fully controlled environment with parameterizable links in terms of capacities, packet loss rates, queues sizes and propagation delays. We can also control the amount and statistical distribution of the cross traffic inserted to each link. The testbed utilizes low cost PCs and freely available open source software. Python scripts are used to set up the link capacities and then automatically perform experiments and collect results.

The testbed has three main components: client, intermediate routers and server. The client and server are Linux-based machines that host the available bandwidth estimation tools under investigation. They have a recompiled 2.4.20 – 8 kernel to run at 500 Hz clock granularity. The more granularity the higher the clock ticks frequency and the lesser lack of accuracy in probing packets time stamping. A 500 Hz kernel generates a kernel tick interrupt once every 2 ms. Given that in our experiments the rate at which probe packets are sent from client to server is in most of the cases less than 500 packets/s, our testbed is not constrained by clock granularity. Intermediate routers are implemented by four 5.4–RELEASE-FreeBSD machines emulating a multi-hop network path. These machines are loaded with a packet shaper called DummyNet [14] and a traffic generator called MGEN [1]. With DummyNet the capacity of each link can be changed from 0 to unlimited (or limited by the physical capacity). It also allows to introduce packet losses and delays to emulate lossy and long links, such as wireless links and satellite channels, respectively. Different queue sizes can also be established if so desired. The traffic generator MGEN allows to choose the rate, packet size and the

statistical distribution of the cross traffic introduced on a per link basis.

A Python client-server application is used to automate the experiments in the testbed. With this application, users can connect to the testbed remotely and set and run experiments. For example, the application allows users to select the bandwidth estimation tool to be evaluated, the link bandwidths, the position of the tight link, the type and rate of the cross traffic and the number of experiments per scenario (to allow statistical significance in the results). The application reads testing files placed in a particular folder and writes the results in another folder after the experiments are finished.

4 Analytical model

In 1957, Jackson [5, 6] stated a theorem to analyze network of queues. We utilize Jackson’s model to analytically estimate the available bandwidth corresponding to each queue. For that, we established the network of queues shown in Figure 2, which actually mimics the proposed network testbed explained before. The system consists of eight M/M/1 queues representing the network interface cards where the tight link will be set and evaluated (queues 1, 3, 5, and 7), where the cross traffic will be routed outside the system (queues 2, 4, and 6), and where output traffic of the system will be received (queue 8).

In Jackson’s model, if i is the number of nodes in the system ($i=1,2,\dots,K$), the theorem assumes that node i contains n_i queues (servers). Also, items arrive from outside the system or from other nodes to node i at a Poisson rate and are served in turn at an exponential service rate. Once served at a node, an item goes (instantaneously) to node j ($j= 1, 2, \dots, K$) with probability θ_{ij} , or out of the system. From these assumptions, in the steady state, the average arrival rate to node j (λ_j) is given by Equation 4, where θ_{ij} represents the routing probability of going from node i to node j and γ_j is the external traffic entering queue j .

$$\lambda_j = \gamma_j + \sum \lambda_i \theta_{ij} \quad (4)$$

The underlying stochastic process of the system is defined by

$$X = \{X_t^i : t \in R^+, i \in [1..8]\} \quad (5)$$

where X_t^i is the number of packets in the queue and server i at time t . It is worth noticing that the probing packet traffic rate γ_0 calculated from experimentation is utilized in Equation 4 to calculate the input rates. This is the reason why different theoretical values are obtained for each tool in the same experiment.

The value of θ_{ij} corresponds to the routing matrix on each queue, which is different from the transition probabil-

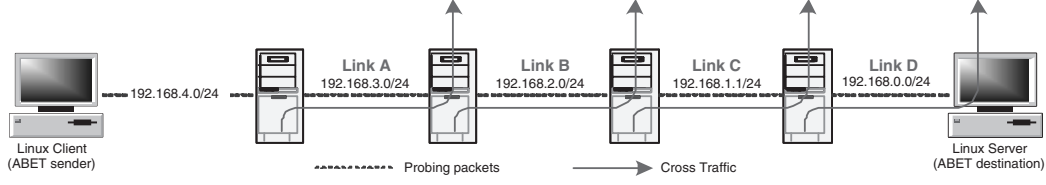


Figure 1. Testbed to evaluate available bandwidth estimation tools.

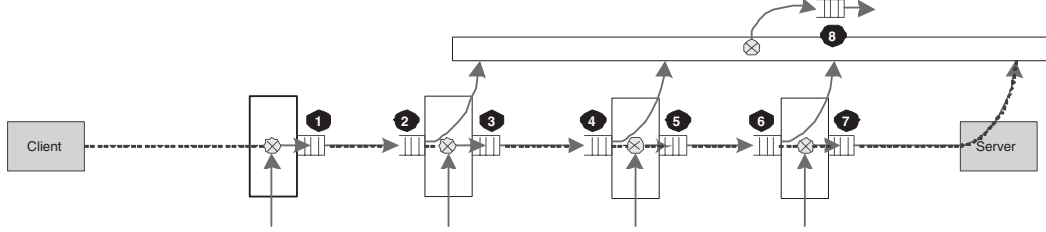


Figure 2. Network of queues for the testbed shown in Figure 1.

ity matrix of the underlying Markov model. In our case, the routing matrix θ_{ij} is given by:

$$\theta_{ij} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{\gamma_0}{\lambda_2} & 0 & 0 & 0 & 0 & 1 - \frac{\gamma_0}{\lambda_2} \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{\gamma_0}{\lambda_4} & 0 & 0 & 1 - \frac{\gamma_0}{\lambda_4} \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \frac{\gamma_0}{\lambda_6} & 1 - \frac{\gamma_0}{\lambda_6} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

Given all the M/M/1 queues input rates, we can estimate the available bandwidth corresponding to each queue as the non utilized capacity of the system as follows:

$$A_i = 1 - \left(\frac{\lambda_i}{\mu_i} \right) = 1 - \rho_i \quad (6)$$

where ρ_i is the calculated utilization of each queue.

It is worth noticing that we assumed a Poisson distribution for the probing traffic generated by the tools. Although this assumption might not be true, the results obtained indicate that this was not a bad assumption. We attribute this to the low overhead introduced by the tool compared with the amount of cross traffic, which is generated using the exponential distribution. Future work will study the exact distribution of this traffic and consider using a G/M/1 network of queues in case of a different distribution than Poisson.

5 Performance Evaluation

Using the testbed and the Jackson's model described before, we evaluate Pathload, IGI and Spruce according to

TOOL	Packet Size	Packets/stream
Pathload	Variable. Minimum: 96B	100
IGI	500B	60 to 256
Spruce	1500B	100

Table 1. Tool parameters.

their convergence time, overhead and accuracy. The convergence time in the case of Pathload and IGI is provided directly by the tool. In the case of Spruce, the convergence time is calculated by the difference of times before and after running the tool. The overhead is given by the ratio between the traffic generated by the tool and the capacity of the tight link. In other words, it represents the percentage of the tight link capacity utilized by the tool. The accuracy is calculated comparing the available bandwidth estimation given by the tool with the expected value from the mathematical model using *Matlab*. We plotted the accuracy as the ratio of the estimated value and the capacity of the tight link. We also include plots showing the relative error on the estimation β , which is calculated using Equation 7.

$$\beta = \frac{m_A - \mu_A}{\mu_A} \quad (7)$$

where m_A is the value calculated from experimentation and μ_A is the value from the analytical model. The main parameters in the evaluated tools are given in Table 1

For each tool evaluated, we defined 28 different scenarios each one corresponding to variations in the capacity of the tight link from 1 Mbps to 9 Mbps at 1Mbps intervals, and from 10 Mbps to 100 Mbps at 5 Mbps intervals. For each scenario, we considered the situation where the links were completely empty of cross traffic and loaded at 25, 50

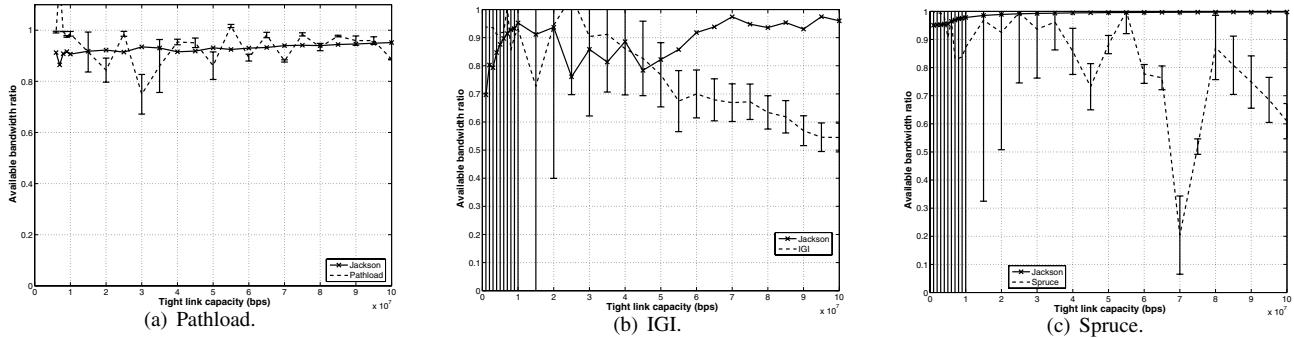


Figure 3. Accuracy estimation with 0% cross traffic.

and 75 percent of the capacity. We used the MGEN utility to generate Poisson processes with mean rates equal to the given desired amount of cross traffic. It is worth noticing that the cross traffic was generated on a node by node basis, i.e. the traffic generated at node i loaded its output queue and the link from node i to node $i+1$, but it did not load the output queue of nodes $j \neq i$. In this manner no traffic correlations from node to node are included and all nodes are completely independent.

For the experimental results, each point in the graphs is the average of running each experiment 35 times. That allowed us in the case of IGI and Spruce, to calculate and plot a 95% confidence interval. In the case of Pathload, we plot the average range given by the tool. Therefore, we performed a total of 11760 experiments: 3 tools, 28 capacity variations, 4 cross traffic loads and 35 samples.

For the analytical results, γ_0 in Equation 4 is the probing traffic rate generated by each tool. That value is the result of dividing the amount of probing packet bytes calculated with *tcpdump* by the convergence time of the tool. Again, this is the reason why the Jackson model behaves differently with each experiment and with each tool. It is worth noticing that most studies have not considered this traffic in their evaluations. We plugged in the input rate from Equation 4 and the output rate (link capacity) in Equation 6 to determine the tight link utilization.

5.1 Accuracy

Figures 3, 4, 5 and 6 compare the accuracy of *Pathload*, *IGI* and *Spruce* when the tight link is loaded with 0%, 25%, 50% and 75% of cross traffic. Pathload provides the best approximation to the analytical value obtained from the Jackson’s model. We do not show some Pathload measurements because the tool had convergence problems in low capacity links. However, as shown in Figure 7(a), when the tool converges, regardless of the amount of cross traffic and tight link capacity, it has a relative error of less than 20%. In most cases the tool overestimates the available bandwidth. It is

well known that Pathload is one of the most accurate bandwidth estimation tools [15]. These results testing Pathload, validate our testbed and the analytical model.

In the case of IGI, this tool has shown some accuracy problems. For example, in [4], the authors of IGI show that the tool has an error of less than 20% in scenarios with low *RTT* values. Our experiments also verify this conclusion although our results also indicate very high variability. Figure 7(b) shows that when the cross traffic is high, the accuracy of the tool is very low. This is also mentioned by the authors of IGI when they tested links with long *RTT*s. However, in contrast to the IGI paper, Pathload is still accurate in our experiments with highly loaded links. Spruce, on the other hand, shows a relative error smaller than 30% in most scenarios, which also verifies the results presented by the authors in [16]. As in the case of IGI, Spruce also presents problems when estimating over high capacity links with high traffic loads. Its estimation variance is also high over low capacity links. It is worth noticing that IGI and Spruce belong to the same *Probe Gap Model* category.

5.2 Overhead

Figure 8 shows the overhead ratio of the tools for each cross traffic load. The overhead of Pathload does not exceed 10% of the tight link capacity. Pathload introduces more probe traffic when the cross traffic decreases. This is completely expected as it works based on the principle of induced congestion, so the emptier the channel the higher the amount of probe traffic that the tool needs to inject.

In Figure 8(b) we can observe that IGI has low overhead over high congested links. This is because IGI finds several packet trains in the *Joint Queuing Region* and does not need to send additional packets to determine the turning point. There are, however, some scenarios where the average overhead grows up to 30% or more, such as those points in Figure 8(b) where the capacity of the tight link is 25 Mbps and the cross traffic is 25 or 50% of the capacity.

There are some interesting results. For instance, Spruce

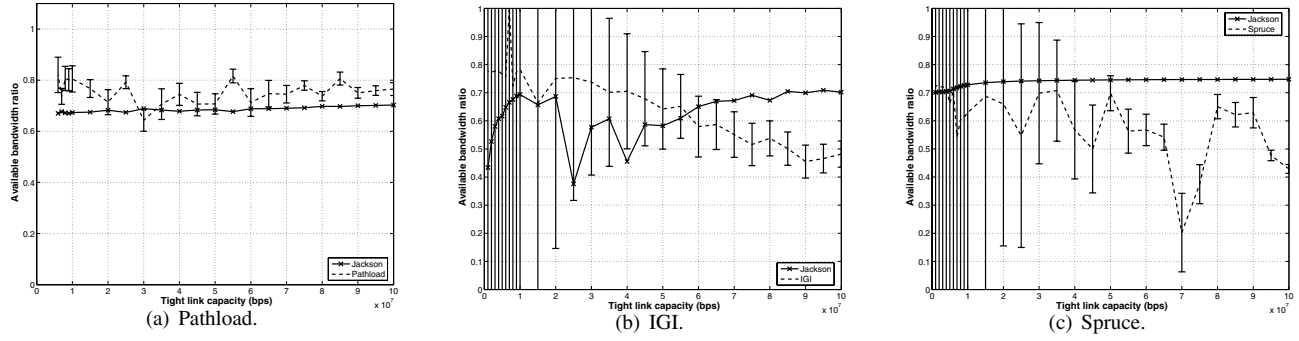


Figure 4. Accuracy estimation with 25% cross traffic.

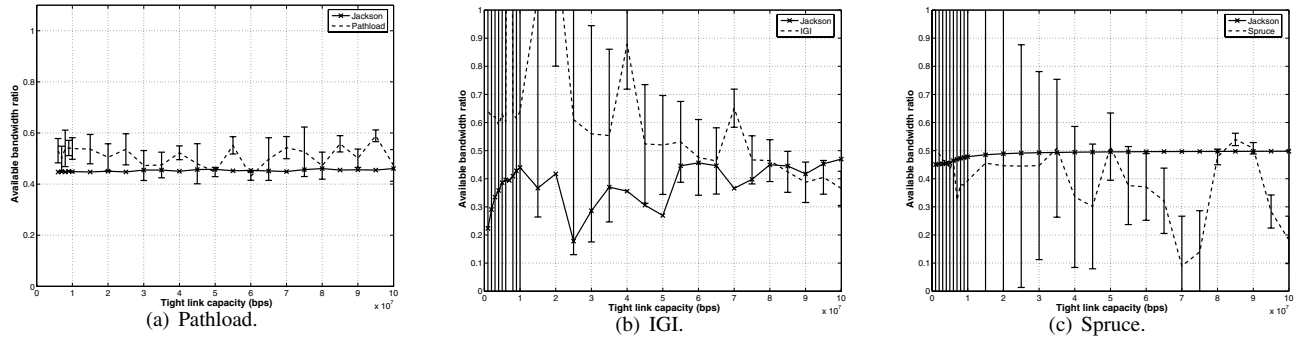


Figure 5. Accuracy estimation with 50% cross traffic.

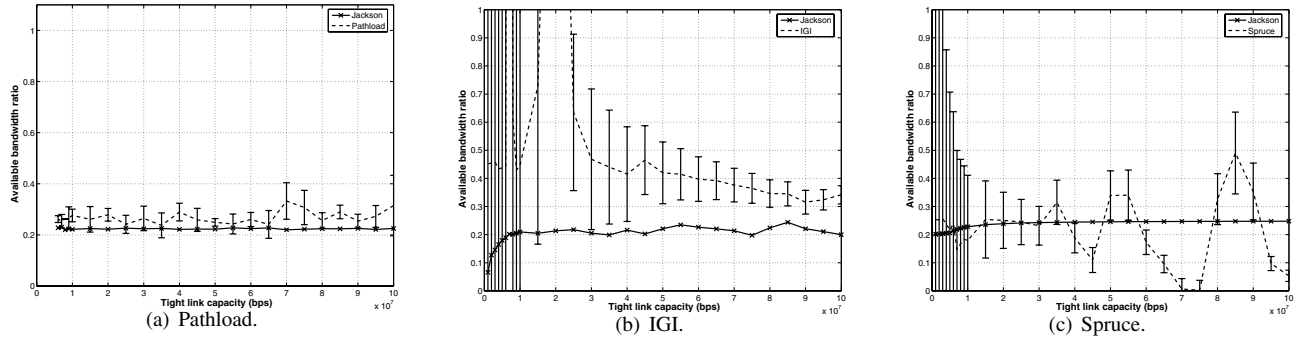


Figure 6. Accuracy estimation with 75% cross traffic.

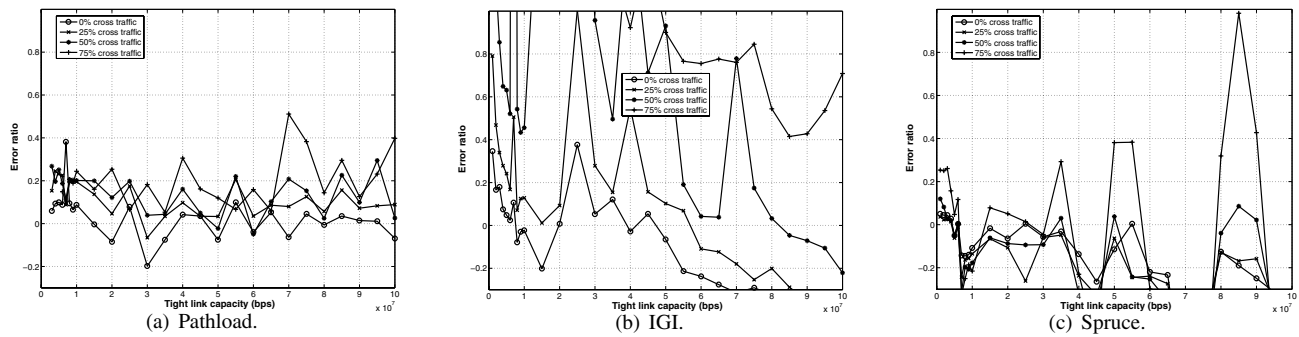


Figure 7. Tool relative error for a cross traffic of 0%, 25%, 50% and 75% of the capacity.

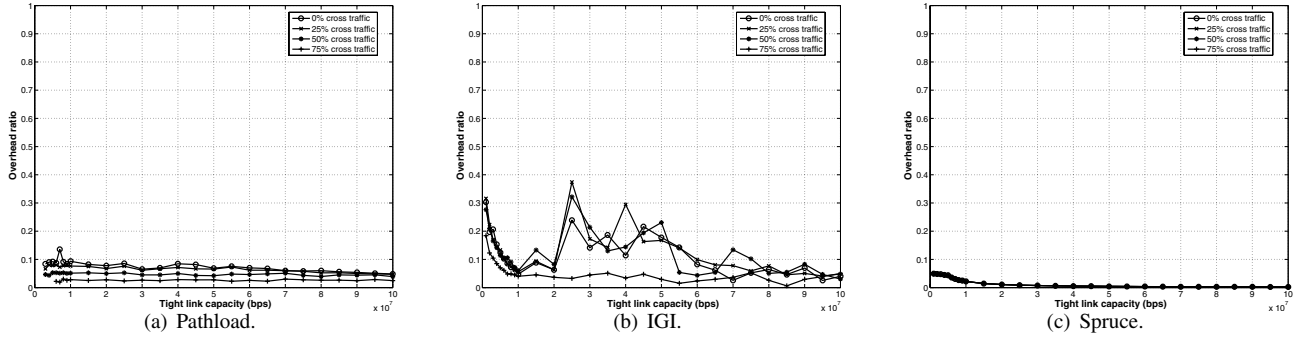


Figure 8. Tool overhead for a cross traffic of 0%, 25%, 50% and 75% of the capacity.

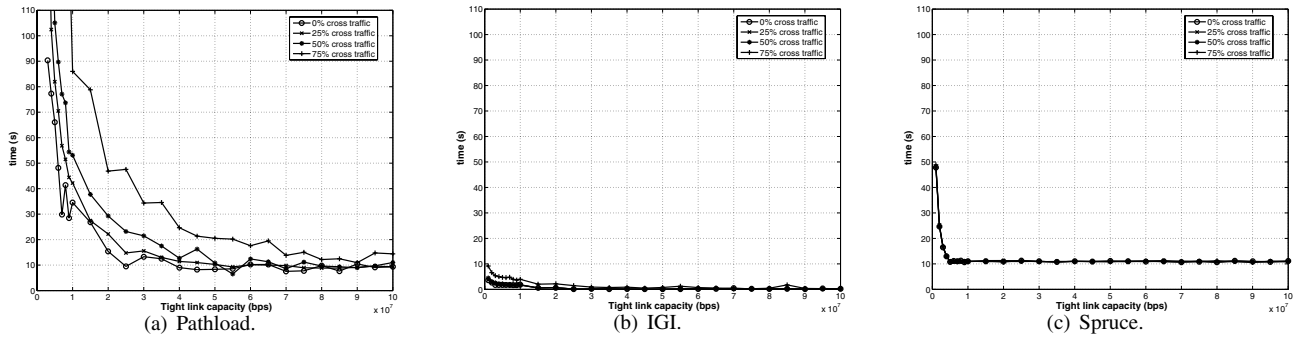


Figure 9. Convergence time for a cross traffic of 0%, 25%, 50% and 75% of the capacity.

overhead is low and constant regardless of the amount of cross traffic. This is explained by the Poisson sampling method utilized by the tool. Another interesting observation is the increase of traffic overhead when the tools based on the *Probe Gap Model* operate in low bandwidth scenarios. In this case, IGI needs to send more probing packets to find the correct probing gap value and Spruce achieves only a small inter-pair gap in the Poisson sampling process, which results in a quite more intrusive sample. This is also reflected in the high accuracy variation shown by these two tools in low link capacities.

In the best case, when the network is highly loaded, the tools need to inject around 3% probe traffic of the narrow link capacity to perform the estimations. Although 3% sounds like a low value, in reality it may be a big number. For instance, in the case of a 50 Mbps narrow link, the tools would occupy 1.5 Mbps. This amount of overhead could limit the utilization of the tools in certain environments, such as wireless networks where link bandwidth is a scarce resource.

5.3 Convergence Time

Figure 9 depicts the convergence time in seconds when the cross traffic varies from 0% to 75% of the narrow link.

From the figure, it can be seen that Pathload needs less time to converge in the case of 0% cross traffic than in the 75% case. Topdump traces provide the explanation for this behavior. When the network is slightly loaded, the tool is able to send probe traffic more frequently and obtain feedback about each sample faster. As a result, it injects more traffic and converges faster. When the network is highly loaded, the tool needs to increase the gap between probe packets and the gap between trains, which reduces the amount of probe traffic. However, the tool has more problems finding the estimation, which traduces into longer convergence times. Pathload can take more than 100 seconds to provide the estimation in some cases. This long convergence time may limit the applicability of Pathload in certain applications or may provide erroneous estimations in those environments with fast changing traffic patterns.

As it is shown in Figure 9(b), IGI needs considerably less amount of time to converge than Pathload. Spruce convergence time is directly associated to the amount of probing packets sent to the network, which is constant regardless the amount of cross traffic.

Regardless of the amount of cross traffic, the evaluated tools have more problems converging when the narrow link is a low capacity link. In the case of IGI and Spruce, their behavior can be explained by the difficulty of the tools to

set the appropriate gap, which implies more measurements and more delay in the estimation. In the case of Pathload, the smaller the available bandwidth, the higher the number of iterations the tool needs to perform to detect the gray region. That is also the reason why in some of these points the tool was not able to converge.

6. Conclusions and Future Work

In this paper we present a low cost and flexible testbed to evaluate available bandwidth estimation tools. In addition, we include an analytical model based on Jackson's networks that provides the available bandwidth of an end to end path and considers not only the cross traffic in the network but also the probing traffic generated by the experimental estimation tool. We utilize the testbed and the analytical model to evaluate the accuracy, overhead and convergence time of Pathload, IGI and Spruce. We performed a large number of experiments and validated the correct operation of the testbed, the analytical model and the tools under consideration.

Regarding the performance evaluation of the tools, we found that Pathload is the most accurate tool but the slowest to converge. IGI, on the other hand, is the fastest tool but the least accurate. Spruce is the least intrusive tool with intermediate accuracy and convergence time.

Future work is under way in two directions. First, we want to make the analytical model more accurate. We are looking at the distribution of the probe traffic to verify that it is exponentially distributed. If it is not, we will also look at the distribution of the cross traffic and probe traffic together to see if the sum of the traffic is exponentially distributed. Depending of the results, we will expand the model to G/M/1 type of queues. Second, we are in the process of incorporating more tools into the testbed and perform experiments in higher bandwidth links and with RTT variations.

7. Acknowledgments

The authors want to acknowledge the programming work to automate the tests and data collection tasks performed by Shannon Osmond, Jorge Ortiz and Andi Ocasio, REU students from USF and the University of Puerto Rico Mayaguez supported by the National Science Foundation under grant No. 0453463.

References

- [1] *MGEN: Multi-Generator Toolset* [Online]Available: <http://tang.itd.nrl.navy.mil/5522/mgen/mgen4.html>.
- [2] L. Angrisani, S. D'Antonio, M. Vadursi, and G. Ventre. Performance Comparison of Different Techniques for Available

- Bandwidth Measurement in Packet Switched Networks. In *Proceedings of VECIMS*, July 2003.
- [3] C. Dovrolis, P. Ramanathan, and D. Moore. What do Packet Dispersion Techniques Measure? In *Proceedings of IEEE Infocom*, April 2001.
- [4] N. Hu and P. Steenkiste. Evaluation and Characterization of Available Bandwidth Probing Techniques. *IEEE Journal on Selected Areas in Communication*, 21:879–894, 2003.
- [5] J. R. Jackson. Networks of Waiting Lines. *Operations Research*, 5:518–521, 1957.
- [6] J. R. Jackson. Job Shop Like Queuing Systems. *Management Sciences*, 10:131–142, 1964.
- [7] M. Jain and C. Dovrolis. End-to-end Available Bandwidth: Measurement Methodology, Dynamics, and Relation With TCP Throughput. In *Proceedings of ACM SIGCOMM*, August 2002.
- [8] M. Jain and C. Dovrolis. Pathload: A Measurement Tool for End-to-end Available Bandwidth. In *Proceedings of 3rd Passive and Active Measurements Workshop*, March 2002.
- [9] S. Lee, P. Sharma, S. Banerjee, S. Basu, and R. Fonseca. Measuring Bandwidth between PlanetLab Nodes. In *Proceedings of 6th Passive and Active Measurements Workshop*, March 2005.
- [10] B. Melander, M. Bjrkman, and P. Gunningberg. A New End-to-End Probing and Analysis Method for Estimating Bandwidth Bottlenecks. In *Proc. of IEEE Globecom*, 2000.
- [11] R. Prasad, C. Dovrolis, M. Murray, and K. Claffy. Bandwidth Estimation: Metrics, Measurement Techniques, and Tools. *IEEE Network*, pages 27–35, 2003.
- [12] V. Ribeiro, M. Coates, R. Riedi, S. Sarvotham, B. Hendricks, and R. Baraniuk. Multifractal Cross-Traffic Estimation. In *Proceedings of ITC Conference on IP Traffic, Modeling and Management*, September 2002.
- [13] V. Ribeiro, R. Riedi, R. Baraniuk, J. Navratil, and L. Cottrell. pathChirp: Efficient Available Bandwidth Estimation for Network Paths. In *Proceedings of 4th Passive and Active Measurements Workshop*, April 2003.
- [14] L. Rizzo. Dummynet: A simple approach to the evaluation of network protocols. *Computer Communication Review*, 27(1), 1997.
- [15] A. Shriram, M. Murray, Y. Hyun, N. Brownlee, A. Broido, M. Fomenkov, and K. Claffy. Comparison of Public End-to-End Bandwidth Estimation Tools on High-Speed Links. In *Proceedings of 6th Passive and Active Measurements Workshop*, March 2005.
- [16] J. Strauss, D. Katabi, and F. Kaashoek. A measurement study of available bandwidth estimation tools. In *Proceedings of the 3rd ACM SIGCOMM Conference on Internet Measurement*, 2003.