A Hidden Markov Model Approach to Available Bandwidth Estimation and Monitoring

Cesar D. Guerrero[1] and Miguel A. Labrador
University of South Florida
Department of Computer Science and Engineering
Tampa, Florida 33620
{cguerrer, labrador}@cse.usf.edu

***Abstract*—Available bandwidth estimation techniques are being used in network monitoring and management tools to provide information about the utilization of the network and verify the compliance of service level agreements. However, the use of these techniques in other applications and network environments is limited by the convergence time, accuracy, and amount of overhead that they introduce. In this paper, we propose a Hidden Markov Model-based technique to end-to-end available bandwidth estimation and monitoring that improves these performance metrics and therefore promises to expand the use of these techniques in other scenarios. The estimator, which has been implemented in a new tool called *Traceband*, is as accurate as Spruce and Pathload but considerably faster, and introduce far less overhead. In addition, when compared using bursty cross-traffic, Traceband is the only tool that accurately reacts to zero-traffic periods, which may be particularly useful for those applications that need to make decisions in real time.**

I. Introduction

Recently, the estimation of the available bandwidth (AB) of an end-to-end path has received considerable attention due to its applicability in several network applications. For example, AB estimation can be utilized in network management to provide information about current utilization of the network resources or to monitor and verify service level agreements. Transport layer protocols might also use AB information to change the transmission rate according to the amount of bandwidth available in the path, using the network resources efficiently while avoiding congestion.

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. Defining T as the *averaging timescale* of the available bandwidth [1], the average utilization for a sample during T, is given by

$$u_i(t, t+T) = \frac{1}{T}\int_t^{t+T} u_i(s)ds \qquad (1)$$

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

$$AB_i(t, t+T) = C_i[1 - u_i(t, t+T)] \qquad (2)$$

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

For an end-to-end path with H hops, the available bandwidth is given by the minimum non-utilized link in the path. That is, $AB(t, t+T) = min_{i=1..H} AB_i(t, t+T)$. 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 is called the *Probe Gap Model* (PGM) which bases the estimation on the gap dispersion between two consecutive probing packets at the receiver. That dispersion is used to estimate the amount of cross-traffic in the tight link during T which is subtracted from the Capacity to estimate the AB in the path. Examples of tools in this category are *Spruce* [2], *Delphi* [3], and *IGI* [4]. The second approach called *Probe Rate Model* (PRM) 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. *Pathload* [5], *TOPP* [6], and *Pathchirp* [7] are examples of tools utilizing this approach.

Although Pathload and Spruce have been recognized as the best performing tools in their respective estimation approaches, their applicability has been limited to network management tools where the accuracy, overhead, and convergence times are not as strict as in other applications or network environments. Otherwise, performance trade-offs must be made, as it is shown in [8]. Pathload is accurate but takes too long to provide an estimate and introduces considerable overhead. Spruce is faster and less intrusive than Pathload but, in general, is less accurate than Pathload. Further, neither tool is able to react fast enough to cross-traffic rapid changing conditions, which is specially important for those applications that need to make decisions in real time, such as available bandwidth-based transmission rate decisions of a transport layer protocol.

This paper introduces *Traceband*, a new available bandwidth estimation tool based on the probe gap model. Traceband utilizes a Hidden Markov Model approach to provide fast and accurate estimates using a low probing traffic rate, making it the first tool that improves most, if not all, performance metrics over Pathload and Spruce. Further, the tool is also able to quickly and accurately react to cross-traffic variations, like those present in links loaded with bursty traffic. These features not only make Traceband a better tool to perform available

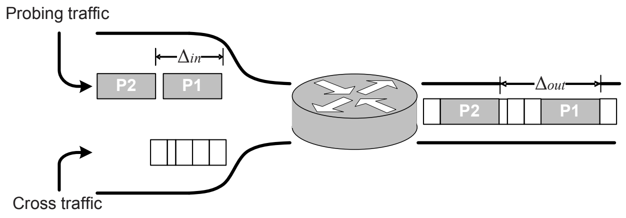


Fig. 1. Probe Gap Model (PGM) sampling.

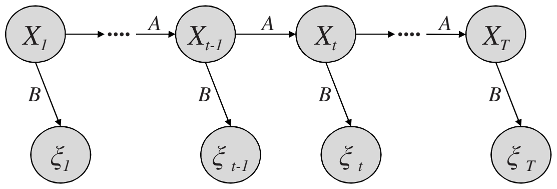


Fig. 2. Hidden Markov Model.

bandwidth estimations for network management tools (no trade-offs are necessary) but also extend the possibility of applying these techniques in other applications and scenarios. As an additional contribution, the paper also introduces the use of a modified version of the moving average technique presented in [9] that can be used in all tools to provide smoother estimations.

The reminder of the paper is organized as follows. Section II explains the Hidden Markov Model and the methods used to obtain observations from the network. Section III presents Traceband implementation details. Section IV compares the performance of Traceband with Pathload and Spruce by using synthetic generated traffic. Finally, Section V concludes the paper.

II. Estimation Model

The available bandwidth in an end-to-end path can be modeled by N states each one representing certain level of availability. For example, in a five-state representation, the AB could be in one of Low (L), Medium Low (ML), Medium (M), Medium High (MH), and High (H) states. That is, it could be located in any spare utilization range from [0,0.2), [0.2, 0.4), [0.4,0.6), [0.6,0.8), or [0.8,1]. Since the average timescale T in Equation 2 is very small (microseconds), it is assumed that transitions from one state to another during that period go no farther than one state apart. Therefore, a one-step transition Markov chain can be utilized to estimate the probability of being in a particular state, or AB range. This assumption is experimentally verified in this paper using Poisson and bursty cross traffic. Future work will verify if it holds in the case of Internet trace-driven traffic and synthetic self-similar cross traffic.

However, AB states can not be directly observed (hidden) since end-to-end estimators do not have information about bandwidth consumption in intermediate routers through the network path. Rather, AB estimators sample the network path with probe packets that convey packet dispersion information instead, which can be used by a Hidden Markov Model to infer the non-observable state.

A. Probing Sampling Method

In order to estimate the average available bandwidth during the period T, the network is sampled using the Probe Gap Model, as shown in Figure 1. Assuming a single tight link, a probing packet pair enters the router with a Δ_{in} separation. Because of interaction with cross-traffic in the router's output queue, a variation or dispersion is observed when the packet pair leaves the link. This variation has a strong correlation with the amount of cross-traffic in the queue during the sampling period. Defining $\epsilon = (\Delta_{out} - \Delta_{in})/\Delta_{in}$ as the relative time dispersion observed, then the available bandwidth can be estimated by:

$$AB = C_t \times (1 - \epsilon) = C_t \times \left(1 - \frac{\Delta_{out} - \Delta_{in}}{\Delta_{in}}\right) \quad (3)$$

Similar to other PGM tools, the value of the tight link capacity (C_t) can be calculated by using well-known and accurate tools, like Pathrate [10].

B. Hidden Markov Model (HMM)

Figure 2 represents the tight link available bandwidth transitions from time $t = 1$ to time $t = T$. This is a Hidden Markov Model (HMM) with discrete hidden states X representing the available bandwidth levels (ranges) and discrete observation variables ξ representing probing packet pair dispersions. A particular observation has associated a probability B to be generated by a particular hidden state. Transition between states are governed by probabilities specified in the transition probability matrix A. This model, which is refined with every new observation, is used to determine the most probable state sequence $(X_1, X_2, \cdots, X_T)$ responsible for what has been observed during T.

As defined in [11], the Hidden Markov Model consists of the following five elements:

1) **Number of states in the model** (N). The larger this number, the better the accuracy but the longer the time needed to provide an estimation. The set of states is defined by $S = S_1, S_2, \ldots, S_N$ where the available bandwidth level grows from S_1 to S_N (from low to high). The state at time t is denoted by X_t.
2) **Number of distinct observation symbols per state** (M). These are all the possible outcomes of a state. That is, the set of symbols corresponding to observed dispersions from the probing sampling method. Although this number can be changed in the model, in the default implementation of Traceband this number is set to ten symbols denoted by $V = \upsilon_1, \upsilon_2, \ldots, \upsilon_{10}$. These symbols are decimal numbers from 1 to 10 grouping continuous observed values of ϵ in the ranges [0,0.1), [0.1,0.2), ..., and [0.9,1]. Every single observation is converted to a discrete symbol by:

$$\xi_t = \lceil M \times |1 - \epsilon_t| \rceil \quad (4)$$

Equation 3 provides the relation between states and observations symbols.

3) **State transition probability matrix** (A). $A = [a_{ij}]$ where $a_{ij} = P(X_{t+1} = S_j | X_t = S_i)$, $1 \leq i, j \leq N$. Since only one-step transitions between states are possible, the number of unknown elements in the matrix is reduced to the three main diagonals:

$$A = \begin{bmatrix} a_{1,1} & a_{1,2} & 0 & \cdots & 0 \\ a_{2,1} & a_{2,2} & a_{2,3} & 0 & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & 0 & a_{N-1,N-2} & a_{N-1,N-1} & a_{N-1,N} \\ 0 & \cdots & 0 & a_{N,N-1} & a_{N,N} \end{bmatrix}$$

4) **Observation probabilities** (B). This is a set of probabilities that indicates how likely is that at time t an observation symbol ξ_t is generated by each state from the set S. More specifically, $B = [b_j(m)]$ where $b_j(m) = P(\xi_t = v_m | X_t = S_j)$ for $1 \leq m \leq M$ and $1 \leq j \leq N$:

$$\begin{aligned} b_{S_1} &= [P(\xi_1/S_1), \cdots, P(\xi_M/S_1)] \\ b_{S_2} &= [P(\xi_1/S_2), \cdots, P(\xi_M/S_2)] \\ &\vdots \\ b_{S_N} &= [P(\xi_1/S_N), \cdots, P(\xi_M/S_N)] \end{aligned}$$

It is expected that small values of ξ are the result of a highly loaded network and therefore more likely generated by a low order state (one indicating low available bandwidth) and conversely. Based on this, probability values are assigned as shown below. Note that 0.4 and 0.25 are high probability values assigned to more likely states:

$$\begin{aligned} b_{S_1} &= [0.40,\ 0.25,\ 0.15, \cdots,\ 0.01,\ 0.01] \\ b_{S_2} &= [0.15,\ 0.40,\ 0.15, \cdots,\ 0.02,\ 0.01] \\ &\vdots \\ b_{S_N} &= [0.01,\ 0.01,\ 0.02, \cdots,\ 0.25,\ 0.40] \end{aligned}$$

5) **Initial state probabilities** (Π). This is a vector with the probabilities that each state is the first in the state sequence that generated the observations. $\Pi = [\pi_i]$ where $\pi_i = P(X_1 = S_i)$ for $1 \leq i \leq N$.

The last three probabilities are usually denoted as λ=(A,B,Π) to indicate the complete parameter set of the model. Table I summarizes all the variables used in the estimation model.

C. Parameter Estimation

Given an observation sequence $O = \xi_1, \xi_2, \ldots, \xi_T$, that is, a set of samples from the network during T, it is desired to estimate the model λ that most likely generated that sequence, i.e. the model λ for which the $P(O|\lambda)$ is maximized. This is done by using the Baum-Welch algorithm [12]. For implementation purposes, a modified version of the Baum-Welch algorithm written in C by Tapas Kanungo [13] is used. The algorithm runs as follows:

Variable	Description
ϵ	Relative time dispersion
ξ_t	Observation symbol at time t
C_t	Tight link capacity
Δ_{in}	Packet pair separation before the tight link
Δ_{out}	Packet pair separation after the tight link
N	Number of states representing AB levels
S	Set of states (low to high): $S = S_1, S_2, \ldots, S_N$
M	Number of distinct observation outcomes
V	Set of observations: $V = v_1, v_2, \ldots, v_M$
A	State transition probability matrix
B	Observation probabilities
Π	Initial state probabilities
T	Number of observations
O	Observation sequence: $O = \xi_1, \xi_2, \ldots, \xi_T$

TABLE I
MODEL VARIABLES.

1) Set the initial model λ_0 with a randomly generated matrix A_0 (one-step transition) and vector Π_0. Matrix B_0 is initialized as explained in the previous section.
2) Calculate a new $\bar{\lambda} = (\bar{A}, B_0, \bar{\Pi})$ based on λ_0 and the observation sequence O.
3) if $logP(O/\bar{\lambda}) - logP(O/\lambda_0) < 0.001$ then stop else $\lambda_0 \leftarrow \bar{\lambda}$ and go to step 2.

Notice that $\bar{B} = B_0$ all the time since as explained before, it is expected that small variations of ϵ correspond to a low loaded link and conversely.

D. State Sequence Estimation

Once an approximation to the available bandwidth model λ is available, the Viterbi algorithm [14] can be applied to find the state sequence $(X_1, X_2, \cdots, X_T)$ that maximizes the likelihood of $P(X_1, X_2, \cdots, X_T | O, \lambda)$. The algorithm selects the most likely path from a particular state to all possible paths and does the same for each state. The final most likely path represents the levels of AB that the probing sampling packets have observed during the sampling time. As defined in Equation 1, the final estimation is based on the average utilization observed during T. Therefore, the AB is calculated as the average state in the sequence.

III. TRACEBAND

Traceband[2] is a client-server tool written in ANSI C that uses the described Hidden Markov representation of the available bandwidth dynamics to provide fast, continuous, and accurate AB estimates. The Traceband client runs in cycles of ten estimations. In the first estimation the tool sends 50 UDP packet pairs 1498 bytes long. The nine remaining estimations are performed with 30 packet pairs each. This reduction is possible thanks to the HMM, which is able to learn the AB dynamics with an initial sample and keep the model updated with samples of reduced size. It was found from experimentation that re-learning every ten estimations was enough to maintain good accuracy with low overhead.

Traceband utilizes different values for the intra-gap and inter-gap times of packet pairs. The intra-gap refers to the

[2]On line available at http://www.cse.usf.edu/~guerrerc/traceband/soft.htm

time between the two packets of each packet pair. The intra-gap or Δ_{in} is specified at the sender and is set equal to the transmission time of a single probe packet in the tight link. In that way, the packet pair will be able to capture cross-traffic in the queue, if any. The inter-gap refers to the time between pairs of probe packets, i.e. the time between the second packet of probe pair $i-1$ and the first packet of probe pair i. Similar to Spruce [2], Traceband performs a Poisson sampling process of the available bandwidth of the path by using exponentially distributed inter-gap times. In order to keep the overhead controlled and low, the mean inter-gap time value is calculated so that the maximum overhead introduced by the tool is 5% or less of the tight link capacity.

At the receiver side, the tool server application timestamps each received probing packet at the kernel level. This is performed by setting the `SO_TIMESTAMP` option in the socket. Packets are numbered to determine which packets are in the same pair and calculate the correct relative time dispersion (ϵ) between them. By applying Equation 4, the corresponding observation symbol for the HMM is determined for each packet pair.

The HMM module in Traceband reads the values of N, M and B from a file to compute the model λ based on the 50 (or less) observations calculated at the receiver. The model is used to determine the most likely sequence of states that generated the observations. For every new estimation, the initial model λ_0 is the output of the previous estimation. The sequence of states is then averaged and multiplied by the tight link capacity to provide a final AB estimation.

Traceband includes an optional moving average post processing technique to smooth the AB estimations. This technique, similar to the one proposed in [9] to filter out abrupt changes in the received signal strength of wireless devices, calculates the mean and the standard deviation of five continuous estimations to generate a single estimation with a 95% confidence interval using the t-student distribution. If the next single estimation lies above or below the upper and lower limits of the calculated interval, that estimation is considered a "peak" (a very rare sample) and is changed to the interval upper or lower limit value. Then a new confidence interval is calculated with the last five estimations (a window of five estimations is continuously shifted once every time). The smoothed estimation is therefore the result of averaging the five last measurements after adjusting those out of the confident interval limits. It is worth noticing that this technique is general and could be applied and incorporated into any other available bandwidth estimation tool.

IV. Performance Evaluation

The performance of Traceband is evaluated and compared with Pathload and Spruce, which are used without any modification of their default parameters. The evaluation is performed using the testbed shown in Figure 3. This is a fully controlled environment with a 10 Mbps capacity tight link. Cross-traffic is generated from the host called *US* to the host called *China* and the estimation is performed from *Sender* to *Receiver*. The Multi-Generator MGEN [15] is used as traffic generator; it allows to send cross-traffic at different rates and with different probability distributions. A computer using *tcpdump* sniffs the output link in the router and records a trace with the joined cross and probing traffic. This trace is used to calculate and plot the average utilization of the link every 1/10 seconds.

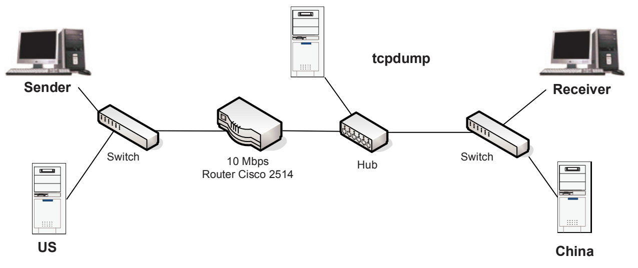


Fig. 3. Testbed to evaluate available bandwidth estimation tools.

The performance metrics used in the evaluation are *accuracy*, *overhead*, and *estimation rate*. The accuracy metric compares the estimation provided by the tool with the real average value obtained from the *tcpdump* trace, during the tool estimation period. In this paper, the accuracy is given by the relative error according to Equation 5, where m_{AB} is the value given by the tool and μ_{AB} is the real AB value from the trace.

$$error = \left|\frac{m_{AB}-\mu_{AB}}{\mu_{AB}}\right| \times 100\% \tag{5}$$

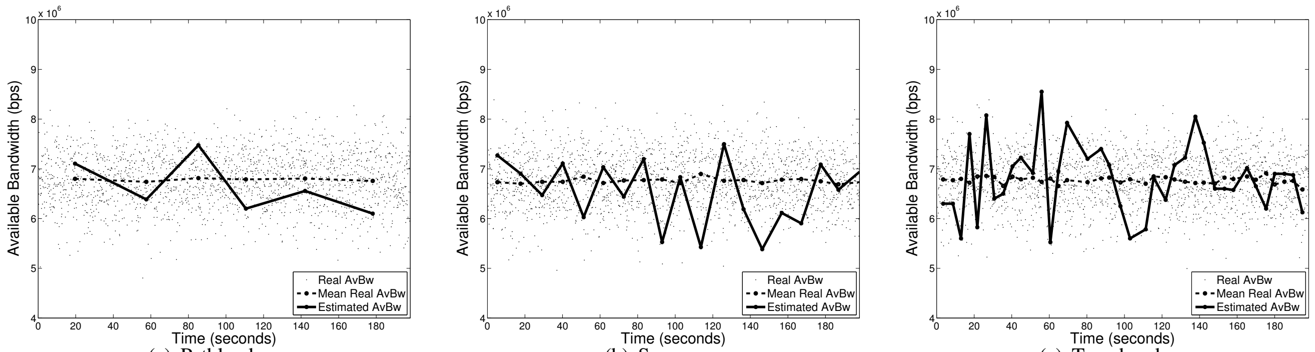
The overhead is related to the amount of probe packets that the tool needs to inject into the network in order to perform the estimation. Although the overhead is usually expressed in *bps*, in this article it is defined as the percentage of tool traffic rate (tool traffic divided by the tool running time) with respect to the total capacity of the tight link.

Finally, the estimation rate shows how often the tool is able to provide an estimate. This rate is given in estimations per minute. The higher this value the better the convergence time of the tool. Pathload and Traceband directly report the estimation time. Spruce estimation time was recorded using a script to calculate the difference of times before and after running the tool.

The tools were evaluated as if they were performing a continuous network monitoring task during a period of 200 seconds. In the case of Pathload and Spruce, it was necessary to run the tools in a loop. In the case of Traceband, the tool has an option to set the estimation period. For every experiment, the output of the tool was redirected to a log file that was processed to extract information about the time, amount, and values of the estimations. The tight link was loaded with 3 Mbps (30% of its capacity) with Poisson and bursty cross-traffic.

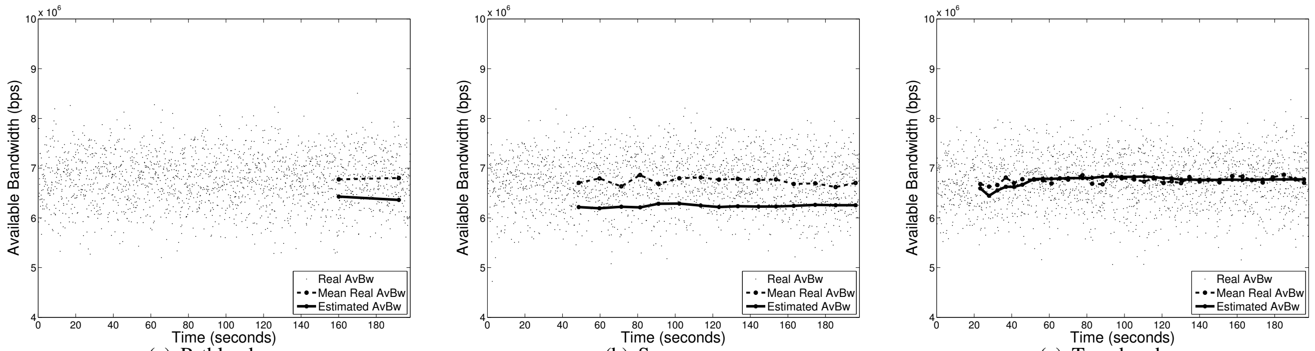
A. Poisson cross-traffic experiments

Figure 4 shows the tools estimations when the tight link is loaded with an average of 3 Mbps Poisson cross-traffic. The mean value for the real available bandwidth is calculated as the average of all real AB values observed between two estimations of each tool. This is done in that way since the tools also provide an average over the estimation period. For



(a) Pathload. (b) Spruce. (c) Traceband.

Fig. 4. Available Bandwidth Estimation for a 10 Mbps tigth link with 30% of Poisson cross-traffic.



(a) Pathload. (b) Spruce. (c) Traceband.

Fig. 5. Moving average post processing to experiments in Figure 4.

Tool	Estimation Error	Estimations/min	Overhead
Pathload	$6.71\% \pm 1.17\%$	1.754 ± 0.066	$6.57\% \pm 0.20\%$
Spruce	$7.77\% \pm 0.98\%$	5.579 ± 0.059	$1.41\% \pm 0.02\%$
Traceband	$8.83\% \pm 0.43\%$	11.645 ± 0.132	$1.96\% \pm 0.03\%$

TABLE II
PERFORMANCE EVALUATION FOR 30% POISSON CROSS-TRAFFIC WITH A 95% CONFIDENCE INTERVAL.

Tool	Estimation Error
Pathload	$4.88\% \pm 2.13\%$
Spruce	$3.84\% \pm 1.92\%$
Traceband	$2.93\% \pm 1.42\%$

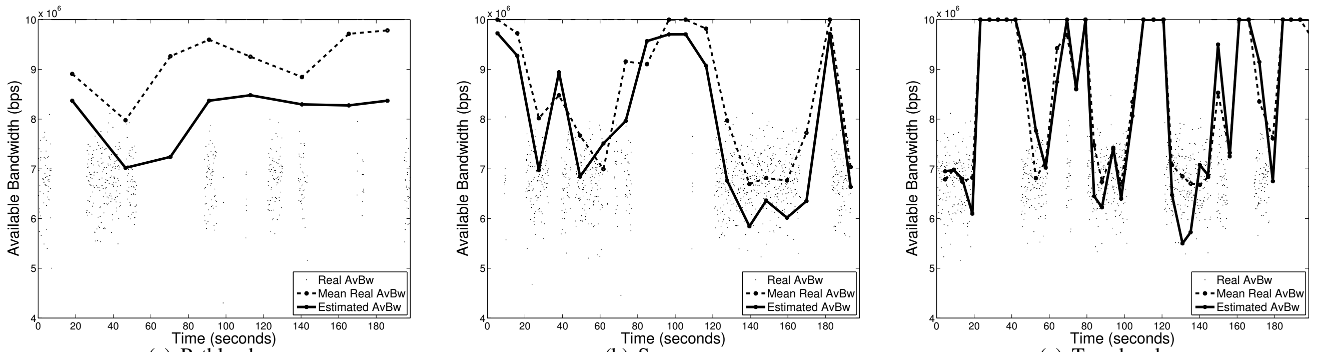
TABLE III
ESTIMATION ERROR AFTER APPLYING MOVING AVERAGE TO EXPERIMENT RESULTS IN TABLE II.

comparison purposes, Pathload single points were calculated as the mid point of the range reported by the tool.

¿From Figure 4, it can be seen that Pathload makes 1.86 estimations per minute, inserts 6.86% of the path capacity as tool overhead, and presents an average estimation error of 6.92%. Spruce, performs 5.49 estimations per minute, inserts 1.42% of the path capacity as tool overhead, and has an average estimation error of 8.54%. Finally, Traceband performs an average of 11.42 estimations per minute, inserts 1.90% of the path capacity as tool overhead, and presents an average estimation error of 8.40%. It is worth noticing that Traceband introduces far less overhead than Spruce and Pathload; the overhead percentages are similar because of the higher number of estimations that Traceband performs during the same time period. In order to have statistically significant results, each experiment was performed five times. For each time, the average value of each metric was calculated. Using these averages and the t-student distribution, a 95% confidence interval was calculated for each tool on each performance metric. Results are shown in Table II.

In this scenario, the three tools under evaluation showed estimation errors below 10%, which according to evaluations performed by other authors like in [16] can be considered as of high accuracy. Compared with Spruce, Traceband has shown to perform twice the number of estimations per minute with similar total overhead. Pathload has shown to be more than three times more intrusive and more than six times slower than Traceband.

The optional moving average described before and available in Traceband was also evaluated. For comparison purposes, it was also applied to the results of Pathload and Spruce shown in Figure 4. From Figure 5, it can be observed that since Traceband estimations are more symmetric over the mean value than Pathload's and Spruce's estimations, after applying the moving average technique, the tool shows the best accuracy and the lowest variability. It is worth noticing that given the small estimation rate of Pathload, using this filtering algorithm the tool is not able to perform the first estimate before 150 seconds. As before, for the set of five experiments, a 95% confidence interval was calculated. The



(a) Pathload. (b) Spruce. (c) Traceband.

Fig. 6. Available Bandwidth Estimation for a 10 Mbps tight link with 30% of Burst cross-traffic.

overhead and estimation rate are the same as in Table II but the estimation error results are shown in Table III.

B. Bursty cross-traffic experiments

Figure 6 shows the results of running the tools when the network is loaded with 3 Mbps of bursty cross-traffic. The length of the bursts and the burst interarrival times are both exponentially distributed with averages of 5 and 10 seconds, respectively. In this case, Pathload makes 2.45 estimations per minute, inserts 7.65% of the path capacity as tool overhead, and presents an average estimation error of 12.06%. Spruce performs an average of 5.46 estimations per minute, inserts 1.34% of the path capacity as tool overhead, and has an average estimation error of 8.21%. Traceband performs an average of 11.86 estimations per minute, inserts 1.98% of the path capacity as tool overhead, and presents an average estimation error of 4.12%. As in the Poisson cross-traffic case, the amount of overhead introduced by Traceband is consoderably lower than Pathload and Spruce. The percentages are similar because of the higher number of estimations performed by Traceband. As it can be observed from Figure 6, since Traceband performs more estimations per minute, the tool is able to accurately react to periods where the tight link has no cross-traffic. Further, during those empty periods, the HMM provides 100% accuracy setting the estimations to the state representing the highest availability. As far as the authors' knowledge is concerned, this is the first tool capable of reacting in such a fast and accurate manner.

V. Conclusions

This paper introduces a Hidden Markov Model approach to end-to-end available bandwidth estimation. The new approach is implemented in a tool called *Traceband* that, compared with Pathload and Spruce, not only provides better performance results overall but also is able to react and accurately estimate the available bandwidth under abrupt changes in cross-traffic. Experimental results over Poisson and bursty traffic show that Traceband can provide more estimations per unit time with comparable accuracy to Pathload and Spruce with far less probe traffic. Traceband also includes an optional moving average technique that smooths out the estimations and improves its accuracy even further. With this result, it is expected that available bandwidth estimation tools be used in a larger number of applications and networking scenarios. Future work will test Traceband using real Internet traces with self-similar cross traffic.

References

[1] R. Prasad, C. Dovrolis, M. Murray, and K. Claffy, "Bandwidth estimation: metrics, measurement techniques, and tools," *IEEE Network*, vol. 17, no. 6, pp. 27–35, 2003.

[2] 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, pp. 39–44.

[3] V. Ribeiro, M. Coates, R. Riedi, S. Sarvotham, B. Hendricks, and R. Baraniuk, "Multifractal cross-traffic estimation," in *Proceedings of the ITC Conference on IP Traffic, Modeling and Management*, 2002.

[4] N. Hu and P. Steenkiste, "Evaluation and characterization of available bandwidth probing techniques," *IEEE Journal on Selected Areas in Communications*, vol. 21, no. 6, pp. 879–894, 2003.

[5] 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*, vol. 11, 2002, pp. 14–25.

[6] B. Melander, M. Bjorkman, and P. Gunningberg, "A new end-to-end probing and analysis method for estimating bandwidth bottlenecks," in *Proceedings of the IEEE Global Telecommunications Conference*, vol. 1, San Francisco, CA, USA, 2000, pp. 415–420.

[7] V. J. Ribeiro, R. H. Riedi, R. G. Baraniuk, J. Navratil, and L. Cottrell, "pathchirp: Efficient available bandwidth estimation for network paths," in *Proceedings of the 4th Passive and Active Measurements Workshop*, vol. 2, 2003.

[8] C. D. Guerrero and M. A. Labrador, "Experimental and analytical evaluation of available bandwidth estimation tools," in *Proceedings of the IEEE Local Computer Networks*, 2006, pp. 710–717.

[9] P. Wightman, D. Jabba, and M. A. Labrador, "An rssi-based filter for mobility control of mobile wireless ad hoc-based unmanned ground vehicles," in *Proceedings of SPIE*, 2008.

[10] C. Dovrolis, P. Ramanathan, and D. Moore, "Packet-dispersion techniques and a capacity-estimation methodology," *IEEE/ACM Transactions on Networking*, vol. 12, no. 6, pp. 963– 977, 2004.

[11] L. R. Rabiner, "A tutorial on hidden markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.

[12] L. E. Baum, T. Petrie, G. Soules, and N. Weiss, "A maximization technique occurring in the statistical analysis of probabilistic functions of markov chains," *The Annals of Mathematical Statistics*, vol. 41, no. 1, pp. 164–171, 1970.

[13] T. Kanungo, "Umdhmm: Hidden markov model toolkit," 1999. [Online]. Available: http://www.kanungo.com/software/software.html

[14] G. D. Forney Jr, "The viterbi algorithm," *Proceedings of the IEEE*, vol. 61, no. 3, pp. 268–278, 1973.

[15] B. Adamson and S. Gallavan, "Mgen," 1997. [Online]. Available: http://cs.itd.nrl.navy.mil/work/mgen/index.php

[16] 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 the 6th Passive and Active Measurements Workshop*, 2005, pp. 306–320.