NETBAND : New Adoptive Technique for Estimating of available Bandwidth using Active probing in a Network between enD systems

G.C. BhanuPrakash[1], Srinivas B.C[2], Rajeswari Seshadri ,K.V.Ramakrishnan
[1]Research Scholar, Dr. MGR University, Chennai.
Asst. Prof., Department of Computer Science and Engineering,
[2] Sr. Lecturer Department of Computer Science and Engineering
Sir M Visvesvaraya Institute of Technology- Bangalore ,INDIA

Abstract
Traffic Measurement in this decade is becoming more complex due to the variety of traffic, complexities in high-speed network and its components. In a packet network the term “Bandwidth” or “throughput” often characterizes the amount of data that transfer per unit of time. Available Bandwidth estimation is of the interest to several user wishing to optimize and for better utilization of resource and planning. In general, the estimation measure relates to capacity, Available Bandwidth (ABWt), bulk transfer capacity etc., for ‘End to End’ systems. Even though accurate Available Bandwidth estimation is of the importance however, more often than not, is an approximate idea of the bandwidth limitations at any given point of time surface the need for traffic engineering and capacity planning which can be achieved in simpler methods. In this paper we have proposed a new technique, called NetBand (*Network Available Bandwidth Estimation*). This technique uses self induced congestion technique for estimating ABWt.

KEY WORDS
Network path, Mean delay, Queuing delay, Available Bandwidth, Relative error.

1. Introduction

Available bandwidth estimation is useful for route selection in a networks, QoS verification, and traffic engineering. Knowing available bandwidth also helps in provide information to network applications on how to control their outgoing traffic and evenly share the network bandwidth. Recent years, there has been lot of research work being carried out in estimation of available bandwidth. With the increase in usage of Internet, allocation of bandwidth by the ISP providers has become increasingly important. Unfortunately, the bandwidth available is scarce and expensive. It is a scarce resource that needs to be utilized efficiently and effectively. Though the bandwidth can be increased to an extent, it is expensive service. So, there is a need to obtain a proper estimate of the available bandwidth, in order to effectively allocate it to the different applications. Many services like video conferencing, audio blocking, mobile networks [2] etc. rely on the basic network metrics of which available bandwidth plays a key role.

The estimate of the available bandwidth could also be used by the service providing applications to understand the state of the network and adjust their service accordingly.

2. Available Bandwidth (ABWt)

Measurement of the ABWt requires some communication to take place between the two hosts. The currently available schemes used for bandwidth estimation fall into two major categories; based on statistical cross-traffic modeling, or on self-induced congestion.
Statistical cross-traffic modeling provides a sufficiently accurate estimate of the ABWt but is basically designed for single-hop networks and is not found to be much useful in multi-hop scenarios. Delphi [16] is an example for statistical cross-traffic model. Self-induced congestion on the other hand relies on congesting the network. The basic concept of self-induced congestion is: If the instantaneous probing rate at which a particular packet is sent across the network exceeds the ABWt of the network, packets will be queued at intermediate routers and this would cause an increase in network delay in the delivery of the packets. If the instantaneous probing rate were below the ABWt, the packets would be transferred normally without facing any queuing delay. So, the probing rate at which the congestion begins can hence be safely treated as the ABWt. Such schemes can be termed as equally suited to both single-hop and multi-hop networks, since they rely on the delay encountered by the probing packets across the entire network.
In a network path a packet gets transmitted from one hop to another passing through a link. Each link has a defined capacity to accommodate. Available Bandwidth (ABWt) in a network path or a link is mainly dependent on the bottleneck link (minimum capacity). The ABWt of a link can be described as the unused capacity of the link for a given time interval and it depends on the traffic load, (including cross-traffic); hence, it is a time-varying metric.

Available Bandwidth of a link or a path between End to End system relates to un-utilized bandwidth at any given instant of time. Capacity of the link mainly depends on, technology being used in the transmission path, propagation delay of the media. The ABWt of a path

Manuscript received April 5, 2010
Manuscript revised April 20, 2010

mainly depends on the traffic load and the configuration of the devices in the mid path.
At any given instance of time the link will be in either of two states i.e., transmitting the packet or in the idle state. So, the utilization of the a link at any instance of time will be 1 or 0. To find the ABWt of any link or path we need to find the weighted average of instantaneous utilization of the link for a given set period of time. The average utilization $\hat{U}(t-\delta,t)$ for a time period $(t-\delta,t)$ can be given as

$$\hat{U}(t-\delta,t) = \frac{1}{\delta}\int_{t-\delta}^{t} u(x)dx \qquad (1)$$

Where $u(x)$ is the instantaneous ABWt of the link at time x. δ is referred to the time length in averaging time scale of the ABWt.

In general to calculate the averaging ABWt in a single hop with C_1 as the capacity of the single hop and u_1 be the average utilization of the hop at a given interval of time, then the ABWt A_1 can be interpreted as utilization factor of capacity.

$$A_1 = C_1(1-u_1) \qquad (2)$$

Further extending it to m number of hops in the path between End to End System, we can define ABWt for m hops as

$$A = \min_{1..m}(A_i) \qquad (3)$$

As ABWt changes over time, it is required to measure it quickly. This is very important to those applications which mainly depend on ABWt for their transmission rates. Till the time of any up-gradation in the network devices the capacity never changes, where as the ABWt which is not constant requires to be measured frequently.

The link with the minimum ABWt is called the tight-link of the path. Fig 1 shows a pipe network model where each pipe represents a different link. The capacity of each link is proportional to the pipe width.

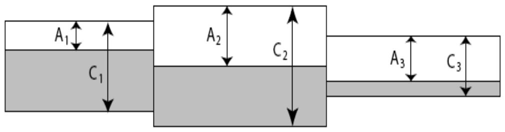

Fig 1: Three consecutive links with their respective capacity and ABWt.

The gray area refers to the used part of the link, so the white area is the ABWt. In this example, the link with the lowest capacity is the rightmost, whereas the link with the lowest ABWt is the leftmost. It may be noted that the tight-link and the narrow-link may not be the same [17].

3. Related work

Numerous network measurement papers published during last decade, mainly focusing on performance evaluation. This can be classified into three generations. Each generation builds on the measurement tools and experiences of the previous ones.
Network measurement techniques can be classified into two categories: passive and active measurement. Passive measurement tools use the trace history of existing data transmission. This technique is potentially very efficient and accurate; the scope of this technique is limited to network paths that have recently carried the user traffic. Active measurement, on the other hand, can explore the entire network by ending a stream of packets. The basic idea is that the sender sends a stream of packets from source to destination. By measuring the changes in the packet spacing, the receiver can estimate the bandwidth properties of the network path.
Self-induced congestion relies on congesting the network. The basic concept of self-induced congestion is: if the transmission rate of the packets is greater than the available bandwidth, then packets will be queued at some intermediate router and this would cause an increase in delay in the delivery of the packets. If the transmission rate of packets is below the available bandwidth, then the packets would be transferred normally with only the propagation delay and without any queuing delay. So, the rate of transmission of packets at which the congestion begins can hence be safely treated as the available bandwidth. Such schemes can be termed as equally suited to both single-hop and multi-hop networks, since they rely on the delay encountered by the probing packets across the entire network.
There are several bandwidth measuring techniques available but most of them measure capacity rather than available bandwidth. Some like Pathchirp[11], tailgating technique[4], however measure per-hop capacity. While on the other hand bprobe[5], nettimer[15] and pathrate[3] measure end-to-end capacity. Another method is Pathload [9], which uses the concept of self-induced congestion. Pathload employs long constant bit rate (CBR) packet trains and adaptively varies the rates of successive packet trains in an effort to converge to the available bandwidth range. The most recent technique to estimate the available bandwidth is Pathchirp [13]. It is also based on the concept of self-induced congestion. It features an exponential flight pattern of probes called a chirp. Packet chirps offer several significant advantages over current probing schemes that are based on packet pairs or packet trains. By rapidly increasing the probing rate within each chirp, Pathchirp obtains a rich set of information from which it dynamically estimates the available bandwidth.

4. NetBand

In this section, a novel ABWt estimation technique named as NetBand is discussed. NetBand is based on self-induced congestion technique. It estimates ABWt along a network path using a number of packets in a stream, each sent at different rates. It utilizes relatively less time to estimate ABWt when compared to other techniques discussed earlier. When packets are probed into a network, some packets whose rates are greater than the bottleneck link capacity would experience a delay. Such packets are considered as "representative packets". NetBand uses these representative packets to compute ABWt, instead of all the probe packets. As only a few packets are analyzed and utilized for estimation of ABWT, this concept becomes the unique proposition of NetBand.

4.1 Overview

NetBand is a novel technique proposed in this work, for estimation of ABWt. It is primarily based on identifying the representative packets that experience more delays in the probe packets and utilizing them for estimating ABWt.

NetBand estimates ABWt along a network path by sending a number of packets in a stream, typically 10 such streams, where each packet in a stream is sent at different rates. They are sent at the same probing range [L, H], (where L is the low probing rate and H is the High probing rate) which constitutes an iteration. ABWt is measured after each iteration and the efficiency of the measured ABWt is calculated using the equation:

$$E = \frac{A_i - A_{i-1}}{A_i} \tag{4}$$

Where, A_i represents average Instant ABWt in the ith iteration. The iterations are carried for 50 minutes or up to an efficiency of 0.0001. Based on the estimation of ABWt in each of the iterations, the next low and high probing range is computed. As the number of iterations increases, the probing range becomes closer to ABWt.

NetBand uses the concept of One Way Delay (OWDs) of packets sent from a source to a destination to determine the point of congestion in the network. The self-induced congestion technique states that ABWt in a network path is the rate of the packet at which the congestion begins. Identifying the packet and its corresponding rate is the primary objective of this work. If the packet rates are lower than ABWt, then the packets would not be dropped, or in other words, it experiences less delay. However, when the packet rate exceeds ABWt, the subsequent packets start experiencing more delay. When the rate of a packet is more than the capacity, packet drop occurs. The subsequent packets in the stream would experience a high queuing delay, which is also the point at which congestion starts occurring in the corresponding link of the network.

Multiple FIFO queues are explicitly permitted in this method so that a path is modeled as a series of store and forward nodes, each with its own constant service rate.
In this technique, a number of packets with a sequence number are transmitted from a designated sender to a designated receiver. Extensive statistical analysis is carried out at the receiver. The receiver sends back the value of instant ABWt to the sender as a feedback, to adjust the next transmission rates.

To generate a stream of packet at different rates, a spread factor γ is used. The Inter Arrival Packet(IAT) spacing between the two successive packets is calculated using the equation:

$$IAT_n = \frac{P}{R_{low} * \gamma^{n-1}} \tag{5}$$

Where IAT_n is the nth inter arrival time, P is the packet size in Bytes; γ is the spread factor of 5% of the previous rate (1.05).

Space between the successive packets j and j-1 is denoted as IAT_j. The packet rate R_j can be calculated using the equation:

$$R_j = R_{low} * \gamma^{j-1} \tag{6}$$

4.2 Representative Packets

The representative packets are computed based on the average ratio of the received packet and the sent packet rate and is calculated using equation:

$$Avg^i_{Stream} = \frac{\sum_{j=1}^{n} \frac{R_j^{Rcv}}{R_j^{Snd}}}{n} \tag{7}$$

Where R_j^{Rcv} and R_j^{Snd} are the received rate and sending rate of the packet j, n is the number of packets in the stream and d_j is the delay experienced by jth packet.
If the calculated average ratio is above 0.95, then it can be concluded that the packet has experienced less delay. Whereas if the calculated average ratio is less than 0.95, then the packet has experienced more delay. Those packets whose average ratio is less than 0.95 are

considered as representative packets for computation of ABWt. As the number of packets generated is more and all the packets in a stream do not contribute in the calculation of ABWt, representative packets are filtered out. Therefore, calculating ABWt using this method yields faster results.

The representative packets are identified using the equation:

$$RPk = \begin{cases} (d_{j-1}, j-1) \text{ if } (d_j < Avg^{i}_{Stream}) \\ d_{j-1} \text{ discarded otherwise} \end{cases} \quad (8)$$

Where RP is an array of Representative Packets with the packet number, RP^k is the kth representative packet, d_j is the delay, and Avg^{i}_{Stream} is the average ratio of the ith stream. From the equation 5.21, it is apparent that:

- If the jth packet delay is less than the average ratio of the ith stream or equivalent to 0.95, then the packet is considered as the representative packet.
- If the jth packet delay is greater than the average of the ratio of the ith stream or equivalent to 0.95, then the packet is discarded as its delay does not contribute for ABWt computation.

4.3 Algorithm for Computing Representative Packets

The following algorithm is used to compute representative packets in a stream.

// Samples the data and returns the number of samples

```
sampledata(samp,num_interarrival)
{
    while(current<=num_interarrival) {
         prev=current-1;
    while(current<=num_interarrival &&
           !s[current])
    current++;
    if(current-prev>1){
       if(prev==0 ||
           current<num_interarrival){
       RP[j].samp_data=qing_delay[current];
       RP[j++].index=current;
      }
      else {
     RP[j].samp_data=qing_delay[current-1];
     RP[j++].index=current-1;
    }
   }
   current++;
}
```

4.4 Computation of ABWt

In this technique, the Constant Bit Rate (CBR) cross-traffic that never exerts any kind of burst in the traffic. Therefore, queuing delays between two successive packets in the packet stream does not increase drastically. The Fig. 2 illustrates a graph showing the queuing delays of representative packets. The graph is the signature of a particular stream of packets.

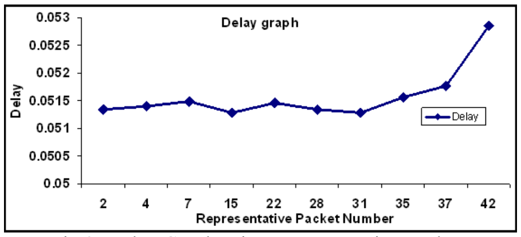

Fig 2: Delay Graph using Representative Packets

As only the representative packets are used to plot the graph, it can be observed that there is a considerable rise and drop between two consecutive packets. As the first few packets' transmission rate is less than the bottleneck link speed, they experience least queuing delay. However, towards the end of the stream, the packets usually have increasing queuing delay as the packet rates are greater than the bottleneck link capacity. ABWt corresponds to the point where the delay starts increasing.

The slope values are calculated for all the representative packets. Initially, when computing the slope value, few representative packets would have a value as zero. When the slope value turns non-zero, the corresponding representative packet's sending rate is considered as instant ABWt. In addition, the weighted average of the packet is also computed to get an accurate ABWt value. The value computed corresponds to, one iteration.

As mentioned earlier, the above process is repeated for 16 streams, with the same low and high rate. The average ABWt of all these streams is sent to the sender as a feedback to modify the low and high rates for the next iteration. The following sections provide different algorithms used in the implementation of NetBand.

4.5 Algorithm for Computing ABWt

```
compute_ABWt(int number_of_interarrival){
   s[0]=0;
// Used for identifying Representative
// Packets
    for(i=0;i<=num_interarrival;i++)
       ratio[i]=rcv_rates[i]/rates[i];
    sampFactor=
       avg(ratio,0,num_interarrival -1);
    val=(sampFactor<0.9?9.5,sampFactor);
    for(i=1;i<=num_interarrival;i++)
      s[i-1]=(ratio[i]>=val?0:1);
// last packet sign is set to 0
    s[i-1]=1;
// Sample the delay from the set of data
    j = sampledate(samp,num_interarrival)
   // Finding minium delay
   ground_rcvr_interval=min(RP,j);
   IAB_W=compute_instant_bw(j, RP);
   for(i=1;i<num;i++) {
     sum+=
        IAB_W*(snd_time[i]-snd_time[i-1]);
     sum_iat+=snd_time[i]-snd_time[i-1];
   }
    return(sum/sum_iat); }
```

4.6 Algorithm for Instant ABWt of a stream using Representative Packets

// Compute Instant bandwidths of a stream

```
compute_instant_bw(no_of_samples,*data)
{
  for(j=1;j<=no_of_samples;j++)
    slops[j]=
      fabs(data[j].samp_data - data[j-
                      1].samp_data);
    for(p=0;p<no_of_samples;p++)  {
  // ground_rcvr_interval is Minium Delay
  // FLUCTUATE = 0.000075
    var=data[p].samp_data -
                ground_rcvr_interval;
    if(var>=-FLUCTUATE && var<=FLUCTUATE)
        slops[p]=0;
    }
    for(j=no_of_samples-1;j>=0;j--)
        if(slops[j]==0) break;
    instant_bw=rates[data[j].index];
    return instant_bw;
}
```

4.7 Rate Adjustment Algorithm

The rate adjustment algorithm was implemented to match the offered network load with the available resources, by modifying the rate at which the packets are sent into the network.

Based on the instant ABWt value, different events (as described below) are designed to modify the next transmission rates. Each of these event indicators and along with the instant ABWt value, are passed as the acknowledgement to the sender. These events are indicated using a numerical value from 1 to 6. The sender checks the acknowledgment received and based on the event indicator, modifies the low and high rate of the window size. Adjusting the window size can efficiently control the traffic, queuing limit, and the buffer requirements at the network nodes. This results in a significant improvement in sending rate and minimizes packets loss.

The following are the events and the corresponding actions to be taken by the sender.

Event 1: When the $R_{low} > R_{high}$ then the R_{low} and R_{high} values are swapped.

Event 2: When the last packet is not received in the expected time, then R_{low} is set to the R_{inst}, value and R_{high} is reduced to 50% of the previous high rate.

Event 3: When a majority of streams indicate the instant ABWt rate as R_{low}, then R_{low} is reduced to 25% and R_{high} is reduced to 25% of its previous rates, respectively

Event 4: When a majority of packets get dropped both R_{low} and R_{high} are reduced to 50% of their previous rates.

Event 5: When the instant ABWt is greater than the current R_{high}, then R_{low} is reduced to 25% of the R_{inst} and R_{high} is increased by 50% of R_{inst}.

Event 6: When the instant ABWt is greater than the current R_{low}, then R_{low} is reduced to 75% of the R_{high} and R_{high} is increased by 50% of R_{high}.

Where R_{low}, R_{high} and R_{inst} are low, high, and instant ABWt rates, respectively, for certain iteration.

5. Single-Hop Scenario

The simulations were carried using NS2.27 simulators. In this paper we propose results of both single hop and the multiple hop scenarios. In single hop scenario the link's bandwidth is set to both low and high bandwidth ranges fed with CBR cross traffic with a single queue. This technique was tested for both NetBand and Pathchip.

In these experiments we have varied several parameters by keeping few parameters constant. The parameters considered are, the Agent probe packet size, CBR cross traffic packet size, and the interval between the two packets. The experiment is carried out by keeping

EXP_1: The experiment was conducted for with agent packet sizes varied from 200Bytes to 1400Bytes, with the CBR cross-traffic of 0.6Mbps and the bottleneck link capacity of 2Mbps.

EXP_2: The experiment was carried out for CBR cross-traffic ranging from 0.1Mbps to 1.3Mbps, the agent packet size and bottleneck link capacity set to a constant value of 1000Bytes and 2.4Mbps, respectively.

EXP_3: The experiment was carried out for varied bottleneck link capacities ranging from 1.3Mbps to 2.5Mbps, the agent packet size and the CBR cross-traffic were set to constant values of 1000Bytes and 0.8Mbps, respectively.

The experiment was carried for both the techniques NetBand and Pathchirp [13] keeping the same parameters. The time taken for estimating the available bandwidth in NetBand is much lesser than the time taken for computing the available bandwidth using Pathchirp.

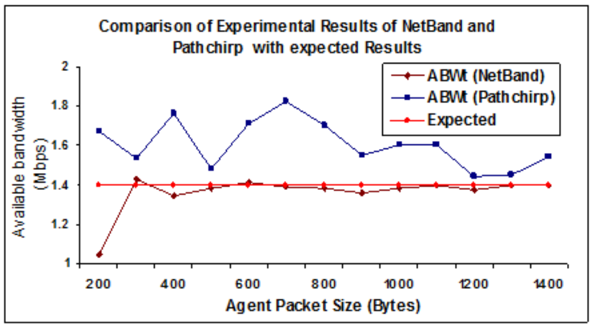

Fig 3a: Comparison of ABWt between NetBand and Pathchirp estimations in EXP_1.

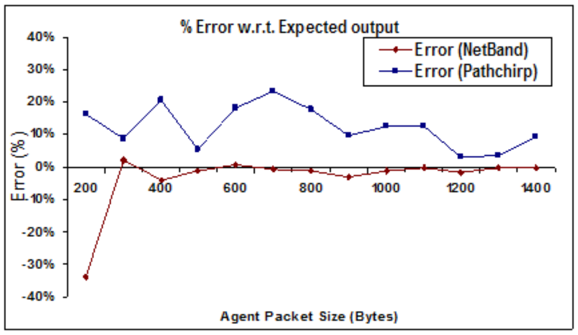

Fig 3b: Comparison of Relative error between NetBand and Pathchirp estimations in EXP_1.

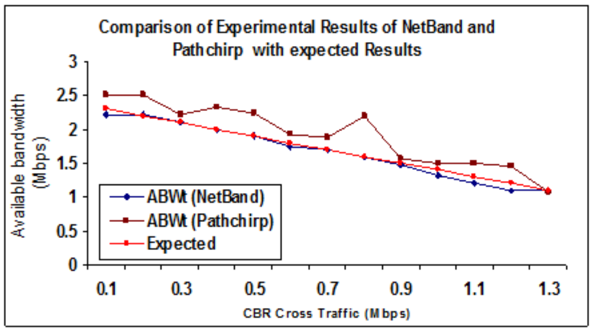

Fig 3a: Comparison of ABWt between NetBand and Pathchirp estimations in EXP_2.

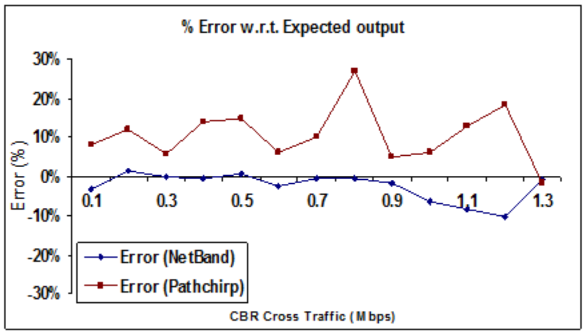

Fig 3b: Comparison of Relative error between NetBand and Pathchirp estimations in EXP_2.

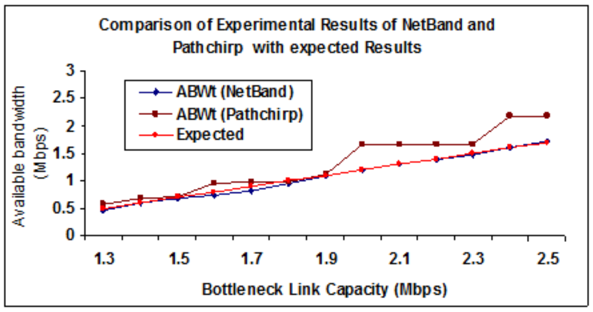

Fig 3a: Comparison of ABWt between NetBand and Pathchirp estimations in EXP_3.

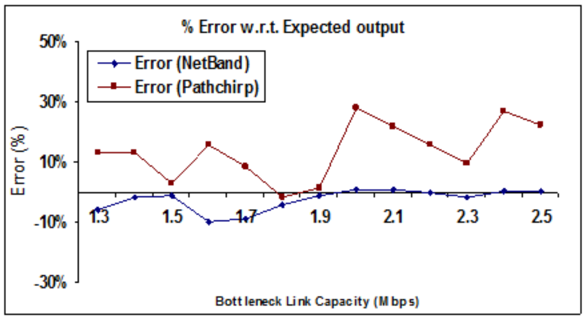

Fig 3b: Comparison of Relative error between NetBand and Pathchirp estimations in EXP_3.

Fig 3a shows the ABWt values measured using both NetBand and Pathchirp with respect to expected value. The relative errors in the estimated values with that of the expected values are shown in the Fig 3 (b). The Fig 3(a) clearly indicates that ABWt values estimated using NetBand are much nearer to the expected values, whereas the estimated values of Pathchirp are higher. The error percentage is within the 10% in case of Pathchirp for agent packet sizes of 300Bytes, 500Bytes and above 1100Bytes. For the rest of the packets the error percentage is above 10%. This clearly indicates Pathchirp does not provide consistent results.

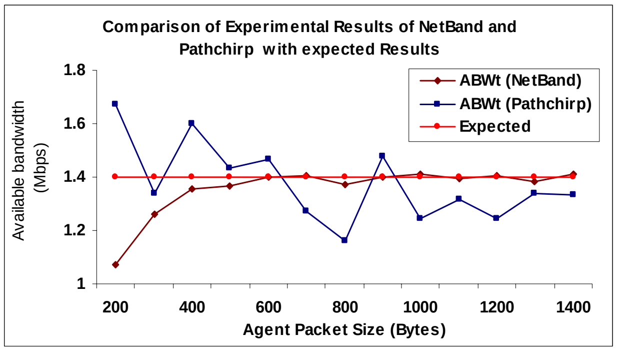

Fig 5a: Comparison of ABWt between NetBand and Pathchirp estimations in Multiple hop Scenario in EXP_4

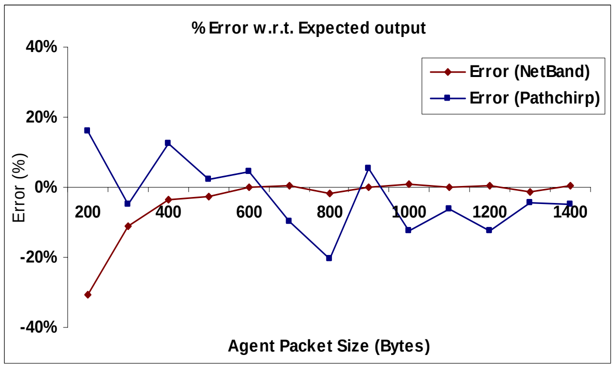

Fig 5b: Comparison of Relative error between NetBand and Pathchirp estimations in Multiple hop Scenario in EXP_4

Fig 3c shows the ABWt values measured using NetBand and Pathchirp with respect to expected value. The relative errors in the estimated values with that of the expected values are shown in the Fig 3d. The maximum error percentage of the estimated ABWt using NetBand is -10% over all CBR cross-traffic. The error percentage in estimated values of Pathchirp only agrees with that of NetBand when the CBR cross-traffic is 13Mbps. For the rest of the experiment, the error in the estimated values is greater than 10%.

Fig 3d shows the ABWt values measured using NetBand and Pathchirp with respect to the expected value. The relative errors in the estimated values with that of the expected values are shown in the Fig 3e. The error in the estimated value lies in the range of ±5% for the bottleneck link capacity of 1.7Mbps onwards. The error percentage is above 10% in the estimated ABWt value by Pathchirp, in most of the cases. These experiments clearly indicate that NetBand is a better method when compared to Pathchirp. NetBand works better with low bandwidth range in a single hop scenario. In all the three experiments the error in the estimated values are in the range of ±5%.

6. Multiple-Hop Scenario

The experiment was executed using CBR cross traffic which uses UDP connection. CBR traffic was introduced to congest the network. The Bottleneck link is fixed to 30MB between Node 4 and Node 5.

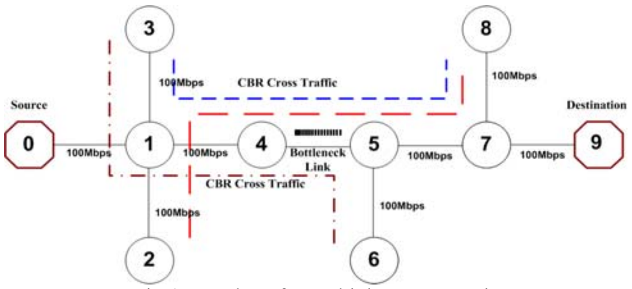

Fig 4: Topology for Multiple Hop scenario

All other links in the topology is fixed to 100MB to make sure the effect of delay in the packet will not influence much when the packets are in bottleneck link. Node 0 is designated as NetBand sender of and node 9 is designated as NetBand receiver. CBR traffic is introduced from node 2 to node 6 and node 6 to node 8 as defined in section 5.
EXP_4: The experiment was carried out for agent packet sizes of 200Bytes to 1400Bytes, with the CBR cross-traffic of 0.6Mbps, and the bottleneck link capacity of 2Mbps.

The Fig 5(a) shows the ABWt values measured using NetBand and Pathchirp with respect to the expected value. The relative errors in the estimated values with that of the expected values are shown in the Fig 5(b). The error in the experimental values of ABWt estimated using NetBand is almost in the range of ±10% except for 1.6Mbps bottleneck link capacity. The error in the estimated values of Pathchirp method is ±20%.

The experiment was carried for the entire scenario as described in single hop. The theoretically expected and estimated ABWt values of NetBand and Pathchirp are plotted in the graph; NetBand results are more of realistic. The experiment was further tried with several scenarios to confirm the behavior of the NetBand in the critical situation like packet drop occurs in a link, congestion in the network. The results were also compared with Pathchirp. Estimation of next iterations low rate and high rate were computed much faster than Pathchirp.

7. Conclusion

After extensive testing on a simulator and followed by detailed analysis of the obtained data it is concluded that the performance of the proposed method is fairly on the expected lines. NetBand is an active probing tool which uses self-induced congestion technique to dynamically estimate the available bandwidth along an end-to-end network path. Simulation experiments and comparison with existing techniques show that the available bandwidth estimations using NetBand are accurate. Moreover NetBand takes much set of values for estimating the available bandwidth Pathchirp. The results obtained from our research are encouraging.

NetBand uses only the Representative packets to estimate the available bandwidth. The tool needs to be tested over the internet traffic. The current algorithm of NetBand for available bandwidth estimation mainly uses information about whether delays are increasing or decreasing. In future work we will investigate algorithms that more fully exploit the rich information contained in the in each iteration.

Acknowledgement

The author acknowledges MGR University for the encouragement and the permission to publish this paper based on the research work carried out by the author towards his Ph.d work. The author would like to thank the principal of Sir MVIT, Dr.M.S.Indira for her constant encouragement and would like to thank Prof. Dilip K. Sen, Head of Department, CSE/ISE for his invaluable guidance and suggestions from time to time.

References :

[1] Alok Shriram, Margaret Murray, Young Hyun, Nevil Brownlee, Andre Broido, Marina Fomenkov1 and kc claffy, Comparison of Public End-to-End Bandwidth Estimation Tools on High-Speed Links.

[2] C. Dovrolis, P. Ramanathan, and D. Moore, “What do packet dispersion techniques measure?,” in Proc. Conf. Computer Communication, Apr. 2001, pp. 905–914.

[3] Cao Le Thanh Man, Go Hasegawa and Masayuki Murata, A Merged Inline Measurement Method for Capacity and Available Bandwidth

[4] I. Sivarajah, D.W. Armintage and N.M. Allinson, History-based bottleneck bandwidth estimation technique.

[5] Jacob Strauss, Dina Katabi, Frans Kaashoek, A Measurement Study of Available Estimation Tools

[6] Jiri Navratil and R.Les.Cottrell, ABwE : A Practical Approach to Available Bandwidth Estimation

[7] Kazumine Matoba, Student Member, Shingo Ata, and Masayuki Murata, Improving Bandwidth Estimation for Internet Links by Statistical Methods

[8] Manish Jain, Constantinos Dovrolis, An Estimation Methodology for End-to-End Available Bandwidth

[9] Manish Jain, Constantinos Dovrolis, End-to-End Available Bandwidth: Measurement Methodology, Dynamics, and Relation with TCP Throughput

[10] Ningning Hu, Evaluation and Characterization of Available Bandwidth Probing Techniques Ningning Hu,Peter Steenkiste ,” Evaluation and Characterization of Available Bandwidth probing Techniques “ IEEE Journal on selected area in communication Vol 31 No 6 Aug 2003

[11] Ravi Prasad, M. Murray, C. Dovrolis, K.Claffy, Bandwidth Estimation: metrics, measurement techniques, and tools.

[12] Seung Yeob Nam, Sunggon Kim, and Dan Keun Sung, Minimal Backlogging Method for Estimation of Available Bandwidth.

[13] Vinay J. Ribeiro, Rudolf H. Riedi, Richard G.Baraniuk, Jiri Navratil, Les Cottrell Pathchirp: Efficient Available Bandwidth Estimation for Network Paths.SLAC/SCS-Network Monitoring.

[14] Xiliang Liu, A stochastic analysis of end-to-end available bandwidth estimation

[15] Y Labit, POwezarski, N. Larrieum, Evaluation of Active Measurement Tools for Bandwidth Estimation in Real Environment.

[16] V. Ribeiro, M. Coates, R. Riedi, S. Sarvotham, B. Hendricks, and R. Baraniuk. Multifractal Cross-Traffic Estimation. In Proceedings ITC Specialist Seminar on IP Traffic Measurement, Modeling, and Management, Sept. 2000.