End-to-end available bandwidth estimation using HybChirp

Wenzhen Chi and Tao Zheng

School of Information Science and Engineering,
Xiamen University,
Xiamen, 361005, China
and
ShenZhen Research Institute of Xiamen University,
ShenZhen, 518000, China
Email: chiwenzhen@gmail.com
Email: izhengtao@126.com

Yi Xie*

School of Information Science and Engineering,
Xiamen University,
Xiamen, 361005, China
and
Fujian Key Laboratory of Sensing and Computing for Smart City,
Xiamen University,
Xiamen, 361005, China
Email: csyxie@xmu.edu.cn
*Corresponding author

Zhongwen Li

The Computer College,
Chengdu University,
Chengdu, 610106, China
Email: lizw@cdu.edu.cn

Yijiang Chen

School of Information Science and Engineering,
Xiamen University,
Xiamen, 361005, China
Email: cyj@xmu.edu.cn

Abstract: The available bandwidth is a crucial metric of networks performance which is applied in congestion control, route selection, traffic analysis and QoS management. PathChirp is a well-known tool to estimate available bandwidth based on an approach of self-induced congestion, which exponentially increases the rate of probing packets in each packet train. But the inappropriate gaps of probing rates often result in a large deviation of estimated available bandwidth. In order to solve this problem, we propose HybChirp, an active probing tool which uses linearly spaced probing packets (lin-chirps) and exponentially spaced probing packets (exp-chirps) in combination. HybChirp improves PathChirp by adopting the hybrid of lin-chirps and exp-chirps, while inherits the advantages of PathChirp such as the short convergence time and low overhead. The NS-2 simulation results have shown that HybChirp outperforms PathChirp and Pathload in terms of accuracy, overhead, convergence time and adaptivity.

Keywords: available bandwidth estimation; PathChirp; linear probing; exponential probing; HybChirp.

Reference to this paper should be made as follows: Chi, W., Zheng, T., Xie, Y., Li, Z. and Chen, Y. (2016) 'End-to-end available bandwidth estimation using HybChirp', *Int. J. Computational Science and Engineering*, Vol. 12, No. 4, pp.360–369.

Biographical notes: Wenzhen Chi is a graduate student in Department of Computer Science, Xiamen University, China. He received his Bachelor degree from Xiamen University in 2010. His research fields are computer network, network modelling, simulation and analysis, and natural language processing.

Tao Zheng is a graduate student in Department of Computer Science, Xiamen University, China. He received his Bachelor degree from Hunan International Economics University in 2013. His research interests include network security, computer application technology, network analysis and performance evaluation.

Yi Xie is an Assistant Professor in Xiamen University, China. She received her BE and MS from Xian Jiaotong University, China, as well as her PhD from the Hong Kong Polytechnic University, Hong Kong SAR of China. Her current research interests include high performance communication, network protocol analysis, network security and modelling and simulation. She is a member of ACM and IEEE Computer Society.

Zhongwen Li is a Professor in Chengdu University, China. She received her MS and PhD from College of Computer Science and Engineering of UESTC in 1998 and 2001, respectively. In 2001, she worked in College of Communication Technology of UESTC as a postdoctoral research fellow. From 2003 to 2010, she worked in Department of Computer Science of Xiamen University. Her research interests include network security and networked computing.

Yijiang Chen is an Associate Professor in Xiamen University, China. He received his PhD from Xiamen University. His research mainly focuses on computer network, natural language processing, artificial intelligence, and text information extraction.

1 Introduction

Available bandwidth (AB) is one significant metric to evaluate the performance of network, which is the minimum AB among all links on an end-to-end path. The estimated AB plays an important role in network protocols of routing selection, admission control, congestion control, and QoS management (Kavitha and Sankaranarayanan, 2014). Therefore, many researchers paid attention to design an efficient and accurate tool to estimate the AB.

The tools of bandwidth estimation can be divided into two categories: active probing and passive measurement. Active probing tools mostly use packet trains for probing, such as RTT (Imai et al., 2013), Pathload (Jain and Dovrolis, 2002), PathChirp (Ribeiro et al., 2003), AProbing (Xie et al., 2014) and trains of packet pairs (TOPP) (Melander et al., 2000). A packet train is defined as a burst of packets sent from the same source to the same destination (Jain and Routhier, 1986). Passive measurement does not require any dedicated probing packets, but collects useful information by monitoring the packets in real communication, for example, cPEAB (Tursunova et al., 2010) and APBE (Park and Roh, 2010).

PathChirp is a well-known active probing tool for estimating AB using self-induced congestion. Self-induced congestion relies on an intuitive idea: if the probing bit-rate exceeds the actual AB, then some probing packets will be queued at a router and the transmission delay will be increased. Otherwise, the packets will not suffer from incremental queuing delay. PathChirp adopts an exponentially spaced packet train (called as a chirp) for probing, where the probing rates of chirps increase exponentially. The inter-arrival time of the probing packets decreases almost exponentially. Then the estimated bandwidth is proportional to the reciprocal of that time.

PathChirp results in a large deviation of estimated bandwidth because the probing bit-rate often changes sharply. Consider a network path whose actual AB is A and two probing packets are successively injected into this network with the rates of M and N respectively, where $M \leq A \leq N$. When the first packet arrives, its queuing delay is almost zero since $M \leq A$. When the second packet arrives, the receiver will detect an increase of queuing delay. The inter-arrival time is then used to estimate AB. The estimated bandwidth is accurate when A approximates to N. But the large deviation is introduced when A is closer to M.

This paper proposes HybChirp, an enhanced tool of PathChirp, to improve the accuracy of bandwidth estimation. HybChirp employs both linearly spaced packets (called as lin-chirp) and exponentially spaced packets (called as exp-chirp) for probing. Firstly, exp-chirps are used to find a reasonable range of AB, which is the same as PathChirp. Secondly, a fine tuning process using lin-chirps is operated within this range. In a lin-chirp, the difference of transmission rates between all successive packets are the same which can be adjusted small enough to satisfy the precision of bandwidth estimation. In this way, HybChirp estimates AB efficiently and accurately.

We have evaluated HybChirp in NS-2 which is a discrete event simulator in computer networks. The simulation results have indicated that HybChirp achieve better performance compared with PathChirp and Pathload in terms of accuracy, overhead and convergence time.

The rest of this paper is organised as following. In Section 2, we introduce the related work on AB estimation and the basic technique of PathChirp. Section 3 illustrates the design of HybChirp and Section 4 evaluates HybChirp

by comparing it with PathChirp and Pathload. Finally, we draw a conclusion in Section 5.

2 Related work

2.1 Existed methods of AB estimation

Active probing method launches end-to-end probing packets into a network, and then estimates the AB by analysing the records of probing packets and their responding packets (Guerrero and Labrador, 2010). Active probing is flexible and convenient, but its additional probing traffic may influence the real traffic of network. Passive measurement monitors the communication of packets in the sender/receiver over a given period, and then calculates AB by the traffic information. Obviously, passive measurement has little effect on real network traffic, but requires abundant packet samples. If the packet samples are not enough, the accuracy of passive measurement will be influenced. Our study focuses on the methods of active probing.

In the early stage, AB is calculated by dividing the total amount of probing packets by the transmission period. For example, in Cprobe (Carter and Crovella, 1996), a dedicated server sends a short stream of probing packets and the receiver records the arrival time of the first packet and the arrival time of the last packet. The AB equals to the total length of all probing packets divided by the period between two records of arrival time. Cprobe-like methods, such as Pathchar (Downey, 2000) and Pipechar (Jin et al., 2001), are simple but result in high cost due to abundant probing packets.

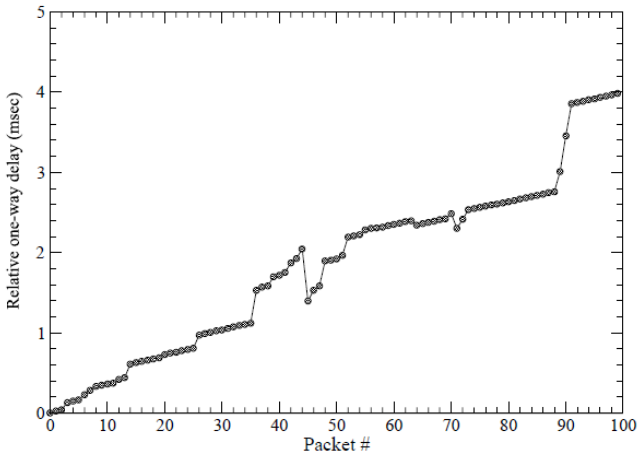
Some researchers use probing packets in an efficient way. The technique of packet pair/train dispersion (PPTD) (Jacobson, 1998) based on a probe gap model is studied. This technique relies on an assumption that the link with the minimum AB must be the link with the minimum capacity in an end-to-end path. PPTD estimates AB by analysing the time gaps between successive probing packets at the receiver, such as IGI (Hu and Steenkiste, 2006). TOPP is an extension to the PPTD technique, which gets rid of the unrealistic assumption by detecting the bottleneck link first. A comparatively new technique uses well-separated probing packet pairs with different spacing. It estimates AB from the time-averaged spacing of packets at the receiver, for example, Nettimer (Lai and Baker, 2001) and Pathrate (Dovrolis et al., 2001). However, the accuracy of estimated bandwidth is sensitive to the selection of spacing between packet pairs. And all routers involved in the end-to-end path are assumed to employ FIFO queuing rule, which is not applicable in most networks.

The newly methods of self-loading periodic streams (SLoPS) are based on the observation of the queuing delay of successive periodic probing packets. The delay increases when the rate of probing packets is higher than the AB on an end-to-end path. That is, the queuing delay of a packet stream shows an increasing trend when the rate of probing packet is higher than the AB. The widely known implementations of SLoPS are Pathload and PathChirp.

Pathload is an iterative algorithm which uses multiple constant bit-rate streams for producing a single estimation. The basic idea of Pathload is that the one-way delay of a periodic packet stream shows the increasing trend when the stream rate (SR) is larger than the AB. Denote a_i as the arrival time of the i^{th} packet and t_i is the timestamp of transmission. The relative one-way delay of each packet is calculated as $D_i = a_i - t_i$. When $SR > AB$, the stream creates a short-term overload in a bottleneck link. The queueing delay of packet i at the bottleneck link is expected to increase. Pathload firstly transmits constant bit-rate streams with a low SR and then increases the SR step by step, while the trend of packet delay D_i is closely monitored. When $SR > AB$, the relative one-way delay $\{D_1, D_2, \ldots, D_k\}$ of the stream packets are expected to have an increasing trend and then the current SR is considered to be AB, shown in Figure 1 (Jain and Dovrolis, 2002).

Compared with Pathload, PathChirp decreases the number of probing packets while obtaining an acceptable result of estimated bandwidth. The design of PathChirp is introduced in detail in Subsection 2.2. This paper proposes a new method HybChirp to improve the accuracy of PathChirp.

Figure 1 One-way delay variations when SR > AB



2.2 Introduction of PathChirp

In PathChirp, many chirps of probing packets are sent from a sender to a receiver. A chirp consists of N exponentially spaced packets, and all packets have the same size of P bytes. The sender transmits the k^{th} packet at the time t_k, $k = 1, \ldots, N$. And the transmission rate of the k^{th} packet is shown in equation (1), where the interspacing time $\delta_k = t_{k+1} - t_k$, $k = 1, \ldots, N-1$.

$$R_k = P / \delta_k . \tag{1}$$

A statistical analysis of arrival packets is conducted to estimate AB at the receiver. Figure 2 (Ribeiro et al., 2003) presents the queuing delay situation of N packets in a chirp, where one dot represents a packet. The vertical axis presents the queuing delay of packets and the horizontal axis

presents the time when the packets are transmitted. The queuing delay for the k^{th} packet is denoted as q_k. An excursion starts with the i^{th} packet if $q_i \le q_{i-1}$ and $q_i < q_{i+1}$, and stops with the j^{th} packet if $q_j = 0$ or equation (2) is satisfied, in this case q_j decreases by a factor F from the maximum queuing delay experienced during this interval. For example, the first excursion is from the third packet to the seventh packet. Additionally, the length of an excursion shall be longer than a threshold L. If R_k is less than the actual AB, the first few excursions produced by bursts will return to zero (we call them closed excursions). Otherwise, the excursion ends with increasing queuing delay, like the second excursion in Figure 2. Suppose the starting point of the last excursion is packet θ with the rate of R_θ. Note that the last excursion is not closed.

$$q_j - q_i = \frac{\max_{i \le k \le j}[q_k - q_i]}{F}. \tag{2}$$

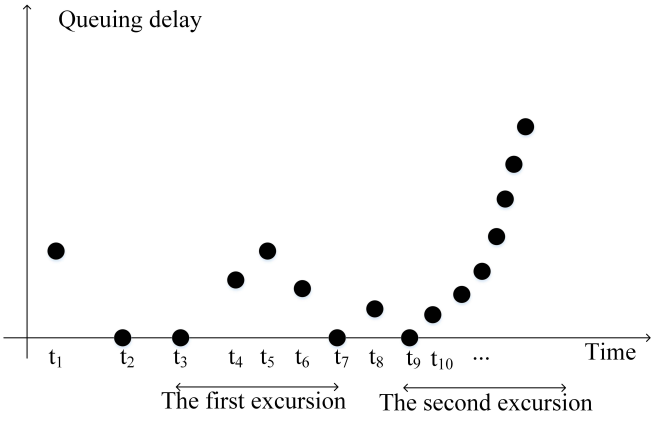
For a chirp of N packets, the bandwidth calculated by each packet is denoted as E_k, shown as equation (3). Then the estimated AB calculated by the chirp is denoted as $\hat{A}$, shown in equation (4).

$$E_k = \begin{cases} R_k, & k < \theta \text{ and } q_k \le q_{k+1} \\ R_\theta, & \text{else.} \end{cases} \tag{3}$$

$$\hat{A} = \frac{\sum_{k=1}^{N-1} E_k \delta_k}{\sum_{k=1}^{N-1} \delta_k}. \tag{4}$$

If the transmission rate of probing packets changes within the range of [G_1, G_2] Mbps, Pathchirp will use approximately $\log(G_1) - \log(G_2)$ packets for probing. That is, PathChirp estimates AB with high speed and low cost, because probing packets are exponentially spaced. But this feature also introduces a big deviation in bandwidth estimation. In order to improve the accuracy of PathChirp, we propose HybChirp which adopts linearly spaced packets (lin-chirp) and exponentially spaced packets (exp-chirp) together.

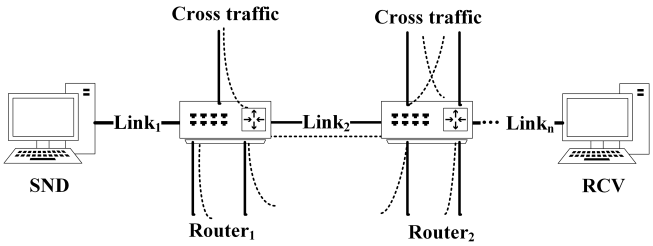
Figure 2 A typical situation of chirp's queuing delay



3 Design of HybChirp

In this paper, HybChirp is designed to estimate the AB on the path between node SND and node RCV, shown in Figure 3. Firstly, SND transmits a packet chirp to RCV with an initial rate. RCV receives the chirp and replies a value of estimated bandwidth. Secondly, SND sends a new chirp with the updated rate according to the probing strategy of chirps, in which lin-chirp and exp-chirp are selected according the feedback from RCV. Therefore, RCV keeps recording the delay of probing packets and calculates the current AB for each chirp, $\hat{A}$. The algorithm of bandwidth estimation is designed according to different probing strategies of chirps. The AB is finally calculated by the cooperation of SND and RCV.

Figure 3 Network topology



Next, the probing strategy of chirps at SND and the new algorithm of bandwidth estimation are described in detail. The parameters and notations of PathChirp are inherited.

3.1 Probing strategy

According to the experiments of PathChirp, exp-chirps can speed up the search of AB. HybChirp inherits its advantage, which firstly sends the exp-chirps of probing packets to locate a potential range of AB. Secondly, the lin-chirps of probing packets are used to obtain a more precise estimated value within this range. Therefore, the probing strategy of HybChirp uses exp-chirps and lin-chirps interchangeably. Figure 4 presents the probing strategy of HybChirp in SND, which includes three parts: the initial part on the top, the exp-chirp part on the left and the lin-chirp part on the right.

The initial part is similar with PathChirp. SND transmits an exp-chirp with a rate within [R_l, R_h], where R_l is the lower limit of probing rates and Rh is the upper limit. The actual AB shall be within this range. The rates of probing packets in one exp-chirp grows in a geometric progression, R_l, γR_l, $\gamma^2 R_l$, …, $\gamma^{N-1} R_l$, where γ is a spread factor. RCV calculates the estimated bandwidth of $\hat{A}$ using the algorithm in Subsection 3.2, and then replies $\hat{A}$ to SND.

After sending one chirp, SND waits for the estimated value $\hat{A}$ from RCV and accordingly adjusts the values of R_l and R_h for sending the next exp-chirp. When $\hat{A} \le \alpha R_l$, $\hat{A}$ approaches R_l and the rates of probing packets are too high. Then R_h will be reset as $R_l * m$ and R_l will be decreased by dividing m. When $\hat{A} \ge \beta R_h$, $\hat{A}$ approaches R_h. Then R_l will be reset as R_h / m and R_h will be increased by multiplying m. Here, α, β and m are tuning parameters, $0 < \beta < 1 < \alpha$ and

$m = 4$ typically. Otherwise, the counter exp_{cnt} increases by 1. The algorithm reaches a stable state when R_l and R_h remain unchanged continuously for c_1 times. That is, the AB shall be within a good range of $[R_l^*, R_h^*]$, where $R_h^* = \hat{A}$ and $R_l^* = \hat{A} / \gamma$.

Next, SND adopts the lin-chirps of probing packets to refine the estimated AB within $[R_l^*, R_h^*]$. In a lin-chirp, the rates of probing packets increases step by step, i.e., R_l^*, $R_l^* + \Delta$, $R_l^* + 2\Delta$, ..., R_h^*, where the step Δ is a function of $\hat{A}$ as shown in equation (5). Intuitively, lin-chirps change the rates of probing packets slower than exp-chirps, which increase the accuracy of bandwidth estimation. If $R_l + n\Delta \le \hat{A} \le R_h - n\Delta$, SND will send the next lin-chirp and increase the counter lin_{cnt} by 1, where n is a tuning factor. Otherwise, SND will adjust R_l and R_h according to Figure 4 and switch to use exp-chirps. If $\hat{A} < R_l + n\Delta$, R_l and R_h will be decreased. If $\hat{A} > R_h - n\Delta$, R_l and R_h will be increased. And the retry of exp-chirps can find a better range of estimated bandwidth. Finally, the probing of SND terminates when $lin_{cnt} = c_2$. The final estimated bandwidth can be simply the latest value of $\hat{A}$ in lin-chirps.

$$\Delta = \begin{cases} 0.1\ \text{Mbps}, & 0 \le \hat{A} < 1\ \text{Mbps} \\ 0.5\ \text{Mbps}, & 1\ \text{Mbps} \le \hat{A} < 10\ \text{Mbps} \\ 1\ \text{Mbps}, & 10\ \text{Mbps} \le \hat{A} < 300\ \text{Mbps} \\ 10\ \text{Mbps}, & 300\ \text{Mbps} \le \hat{A} < 1\ \text{Gbps} \\ 50\ \text{Mbps}, & 1\ \text{Gbps} \le \hat{A} \end{cases} \quad (5)$$

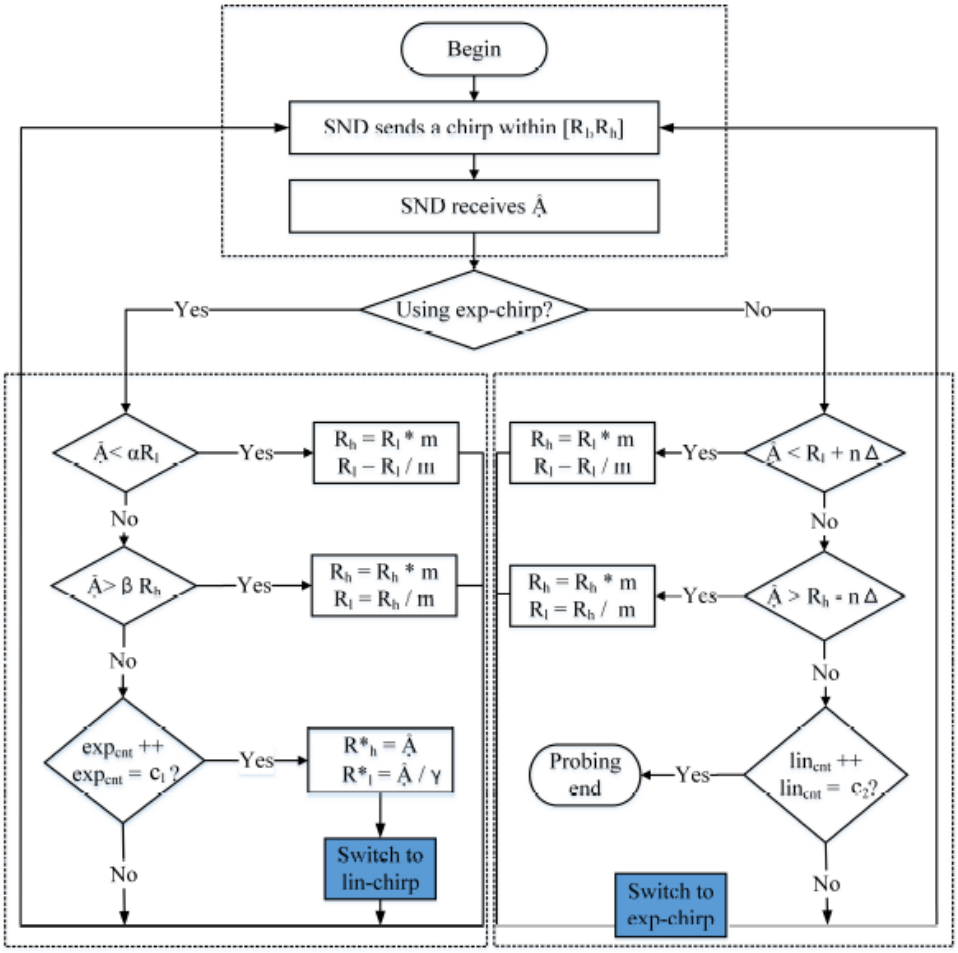
Therefore, exp-chirps and lin-chirps are combined to obtain the AB on an end-to-end path.

3.2 Algorithm in RCV

In Pathchirp, all excursions are used to calculate $\hat{A}$. But the probing packets in closed excursions provide few information because their rates do not exceed the actual AB. Therefore, HybChirp only focus on the last excursion of each chirp and the estimated bandwidth is calculated by formula (6) in SND, where the starting point of the last excursion is packet θ with the probing rate of R_θ. When RCV receives an exp-chirp, the last R_θ is simply taken as $\hat{A}$. When RCV receives a lin-chirp, we take the average $(R_\theta + R_{\theta-1}) / 2$ as $\hat{A}$.

$$\hat{A} = \begin{cases} R_\theta, & \text{in a exp-chirp} \\ (R_\theta + R_{\theta-1}) / 2, & \text{in a lin-chirp} \end{cases} \quad (6)$$

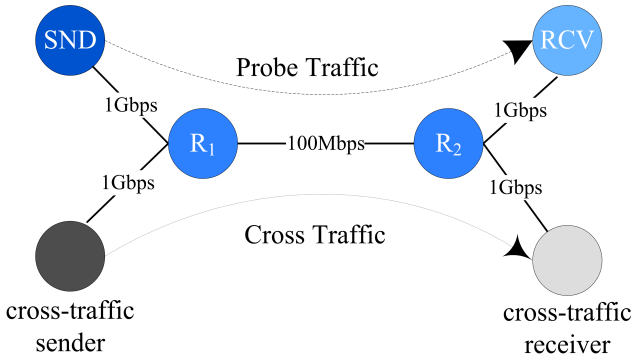
Figure 4 Flow chart of HybChirp's probing strat (see online version for colours)



4 Evaluation in NS-2

HybChirp is evaluated by the NS-2 (version 2.27) simulator in a simple network topology which is also used in related works (Angrisani et al., 2006; Ali et al., 2006), shown in Figure 5. The link between routers R1 and R2 has the capacity of C (Mbps), where $C = 100$. Therefore, HybChirp is carried out at SND and RCV and the end-to-end path is SND-R1-R2-RCV. Other two hosts, the cross-traffic sender and the cross-traffic receiver, control the cross-traffic with the throughput of C_T (Mbps), then the actual end-to-end AB is $A = C - C_T$. Here, three types of cross traffic are considered: constant bit rate (CBR), Poisson and Pareto.

Figure 5 The network topology in NS-2 (see online version for colours)



We also compare HybChirp[1] with two well-known tools of bandwidth estimation: PathChirp[2] and Pathload[3]. PathChirp uses its default configurations (Ribeiro et al., 2003), i.e., $\gamma = 1.2$, $P = 1{,}000$ bytes. Pathload sets the bandwidth resolution as 10 Mbps, and other parameters are the same as the paper (Jain and Dovrolis, 2002). HybChirp inherits PathChirp's parameters and its probing parameters are set as follows: $\alpha = 1.33$, $\beta = 0.75$, $m = 4$, $n = 3$, $c_1 = c_2 = 3$. These three tools are compared in terms of accuracy, convergence time, overhead and adaptivity (as applicable). This paper improves the precision of performance metrics by averaging the results of ten times repeated simulations.

- Error metric of ϵ presents the measurement error of estimated bandwidth, where A is the actual AB and $\hat{A}$ is the estimated bandwidth. If the result of bandwidth estimated tool is a range of $[R_l^*, R_h^*]$, we will simply it by $\hat{A} = (R_l^* + R_h^*)/2$. The lower ϵ, the more accurate result.

$$\epsilon = \frac{|\hat{A} - A|}{A}. \tag{7}$$

- Convergence time is the total period cost in a complete estimation of bandwidth. Less convergence time means a higher speed in bandwidth estimation.
- Overhead is the total amount of probing packets used in a complete estimation of bandwidth. Reducing overhead can decrease the impact of probing on the network.
- Adaptivity, presents the real-time performance of a bandwidth estimation tool. A standard deviation σ is defined to reflect the differences between the estimated values and the actual values during a simulation period, where K estimated values are considered. A low value of σ means that the bandwidth estimation tool well follows the trend of actual AB.

$$\sigma \sqrt{\frac{1}{K}\sum_{i=1}^{K}\left(\frac{\hat{A}_i - A_i}{A_i}\right)^2}. \tag{8}$$

4.1 CBR cross-traffic

Firstly, HybChirp, PathChirp and Pathload are compared in the presence of CBR cross-traffic with different rates of 10, 30, 50, 70 and 90 Mbps.

HybChirp outperforms PathChirp and Pathload in terms of accuracy because its measurement errors are lowest under all cases of CBR cross-traffic, shown in Table 1. For example, in Figure 6(a), when $C_T = 70$ Mpbs and $A = 30$ Mpbs (shown in the solid line), the estimated bandwidth of HybChirp is 28.6 Mbps, while the estimated bandwidth of PathChirp is 25.7 Mbps and the range of estimated bandwidth of Pathload is [20, 42] Mbps.

Table 1 ϵ of three tools under CBR cross-traffic

C_T (Mbps)	10	30	50	70	90
HybChirp	8.0%	4.6%	5.1%	2.5%	5.3%
PathChirp	20.5%	14.1%	15.7%	8.7%	10.9%
Pathload	63.1%	3.3%	6.0%	1.4%	4.4%

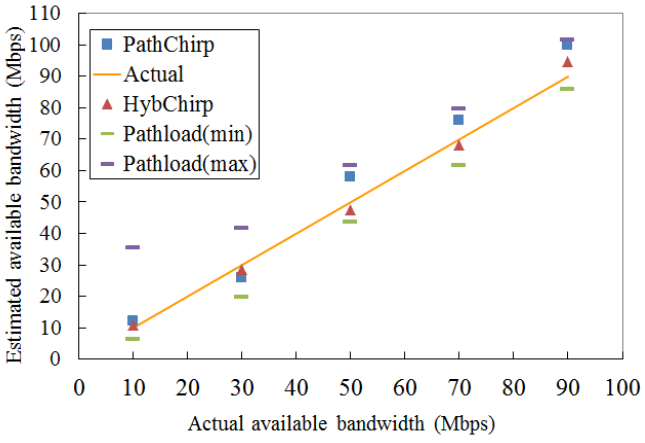
Figure 6(b) compares the convergence time of three tools. The average of convergence time in HybChirp is shortest, around 1 second. PathChirp uses a little longer time, 3 seconds in average. However, Pathload costs about 20 seconds to finish one simulation. Apparently, HybChirp speeds up the bandwidth estimation since that exp-chirps and lin-chirps are combined efficiently.

Figure 6(c) compares the overhead of three tools used in one simulation. It is clear that the amount of traffic injected by Pathload is much more than HybChirp and PathChirp. For example, when $C_T = 10$ Mpbs and A = 90 Mpbs, HybChirp costs probing packets of 0.26 M bytes, PathChirp costs almost double, while Pathload costs as high as 13 Mbytes. In short, HybChirp shows the best performance in terms of overhead under CBR cross-traffic.

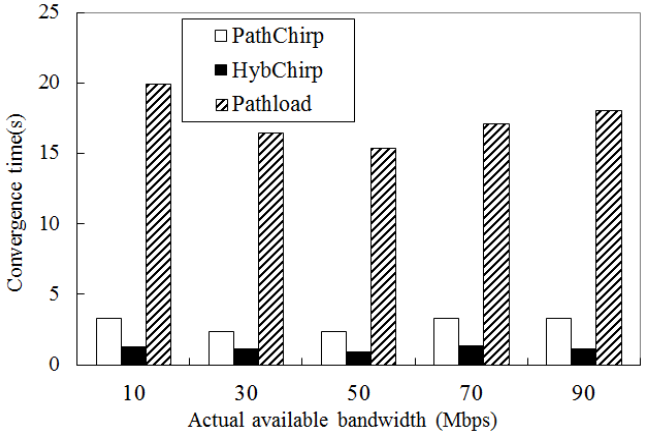
4.2 Poisson cross-traffic

This section studies the performance of three tools under Poisson cross-traffic. The rate of Poisson cross-traffic ranges from 40 Mbps to 60 Mbps with the average of 50 Mbps and the packet size is fixed as 1,500 bytes. Then, the actual AB changes dynamically during a simulation, which are calculated as $100 - C_T$.

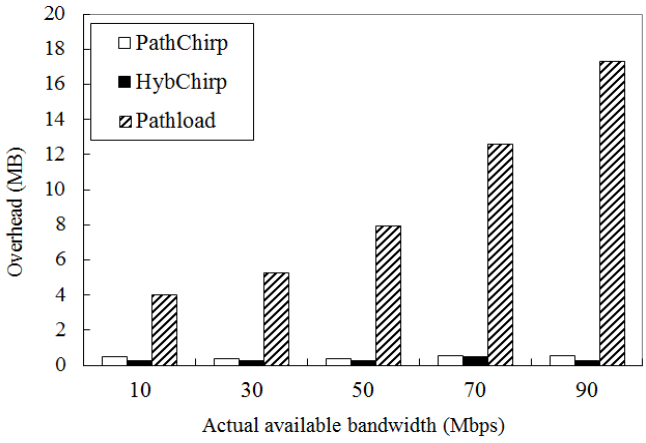
Figure 6 Comparison of HybChirp, PathChirp and Pathload under CBR cross-traffic, (a) the estimated values of bandwidth (b) the convergence time (c) the overhead (see online version for colours)



(a)



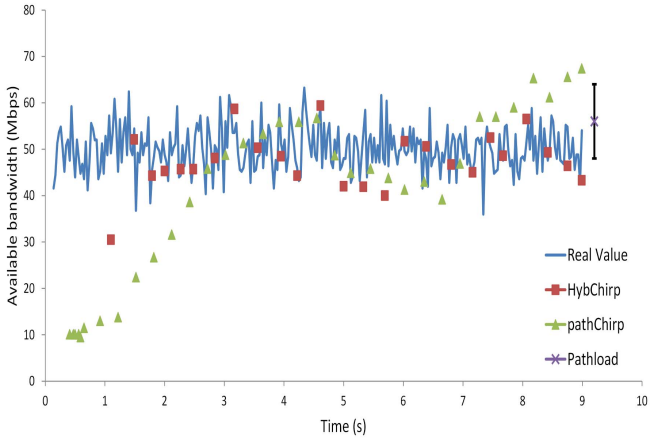
(b)



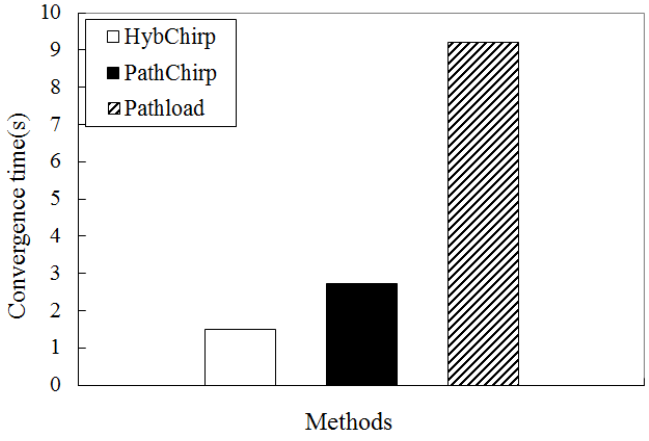
(c)

In Figure 7(a), the actual values of AB are shown as the blue curve. The red squares represent the estimated values of HybChirp, the green triangles represent the estimated values of PathChirp and the purple crosses represent the estimated bandwidth range of Pathload. It is obvious that HybChirp closely follows the rapid changes of actual AB. The big measurement error only occurs at few points, e.g., 30 Mbps is estimated at 1.1 s while the actual value is 48 Mbps. PathChirp displays less accurate than HybChirp. For example, the estimated values between 0.0 s–3.0 s of PathChirp have rather large deviations. Pathload only provides one estimated range of bandwidth without recording any change of AB. Moreover, Figure 7(b) and Figure 7(c) show that HybChirp costs less convergence time and overhead than other PathChirp and Pathload.

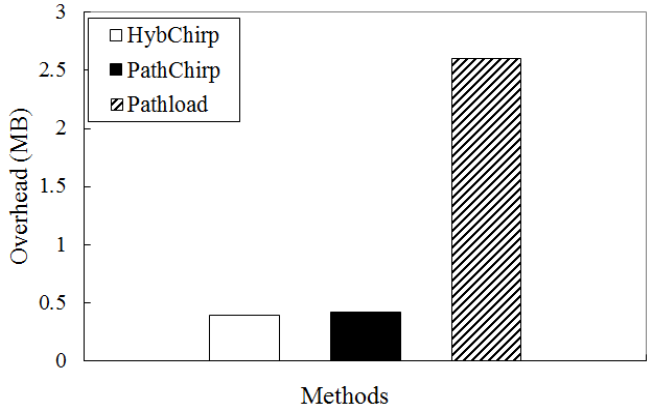
Figure 7 Comparison of HybChirp, PathChirp and Pathload under Poisson cross-traffic, (a) the estimated values of bandwidth (b) the convergence time (c) the overhead (see online version for colours)



(a)

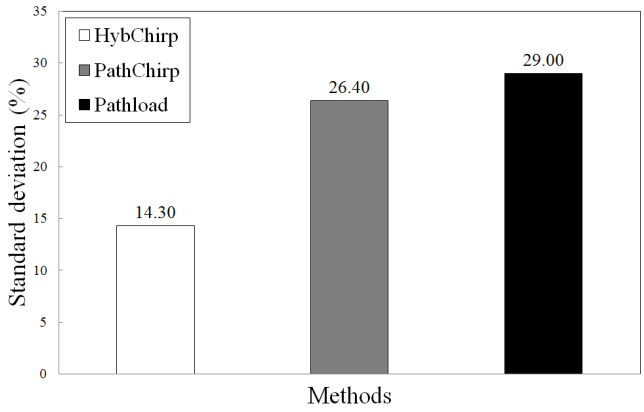


(b)



(c)

Figure 8 σ (%) of HybChirp, PathChirp and Pathload under Poisson cross-traffic



Shown in Figure 8, HybChirp has the lowest value of σ and does well in obtaining the real-time AB. Therefore, HybChirp and PathChirp can be used to estimate the AB online, and HybChirp has a higher precision and adaptivity.

4.3 Pareto cross-traffic

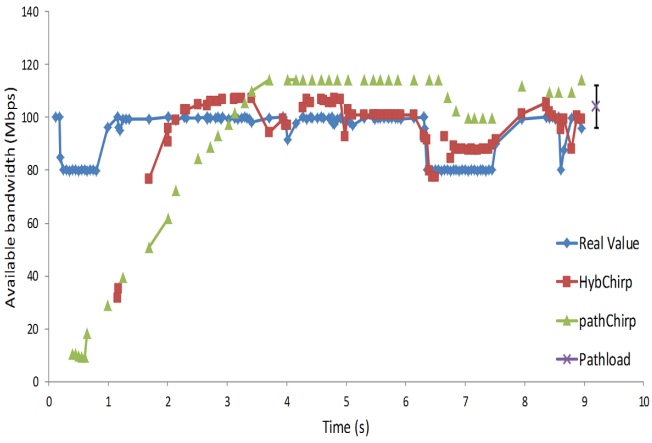
This section considers another type of cross-traffic: Pareto. The parameters of Pareto cross-traffic are as follows: burst time = 5 ms, idle time = 400 ms, packetSize = 1,500 bytes, average rate = 20 Mbps and shape = 1.1. The simulation results are shown in Figure 9 with the period of 10 seconds. The actual ABs on the vertical axis are calculated as the total capacity 100 Mpbs minus Pareto cross-traffic's throughput C_T.

Figure 9(a) shows that HybChirp has best performance during the bust time or idle time of Pareto cross-traffic. In the beginning, HybChirp and PathChirp have lower estimated values of bandwidth than the actual value. At around 2.5 s, HybChirp shows higher accuracy in bandwidth estimation compared with PathChirp. HybChirp costs the least time in bandwidth estimation under Paroto cross-traffic, shown as Figure 9(b). Pathload costs much time in bandwidth estimation because it obtains the estimated range at 9.2 s. HybChirp may send more probing packets to ensure high accuracy than PathChirp. For example, in Figure 9(c), HybChirp shows a little higher overhead than PathChirp.

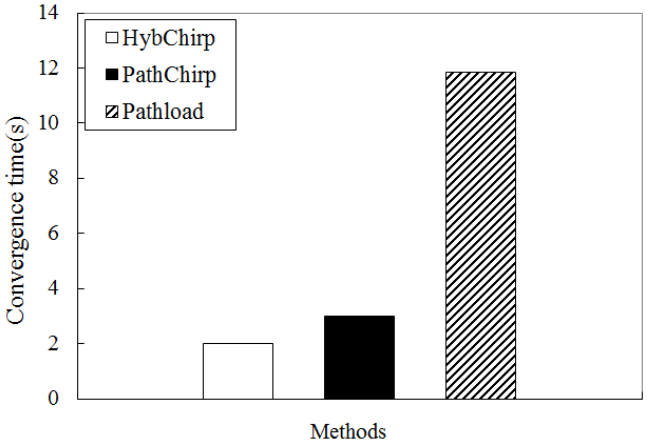
Figure 10 shows that the value of σ in HybChirp is lowest. It means that HybChirp responds well to immediate changes in network environment. We shall note that Pathload only provides one range including two values of estimated bandwidth. Compared with dozens of real-time results estimated by HybChirp and PathChirp, Pathload exposes its disadvantages: long convergence time and ossified measurement.

In summary, the abundant simulations indicate that HybChirp outperforms PathChirp and Pathload in terms of accuracy, convergence time, overhead and is adaptive under different cross-traffic.

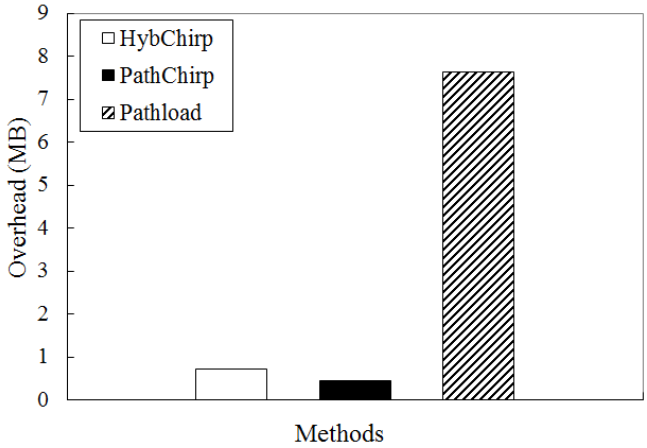
Figure 9 Comparison of HybChirp, PathChirp and Pathload under Pareto cross-traffic, (a) the estimated values of bandwidth (b) the convergence time (c) the overhead (see online version for colours)



(a)

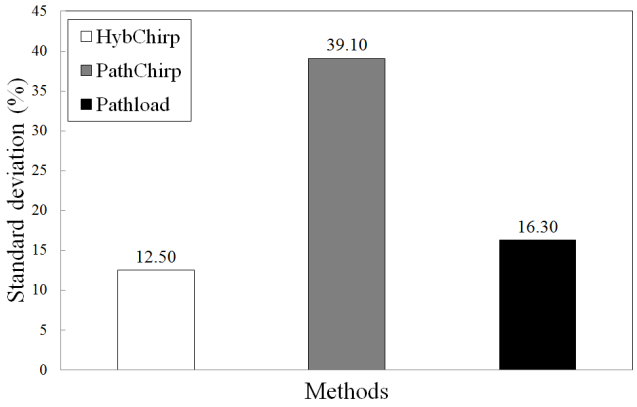


(b)



(c)

Figure 10 σ (%) of HybChirp, PathChirp and Pathload under Pareto cross-traffic



5 Conclusions

In this paper, we propose an efficient tool HybChirp to estimate the end-to-end AB. By inheriting the exp-chirps in PathChirp, HybChirp proposes lin-chirps to improve the accuracy of bandwidth estimation. HybChirp employs exp-chirps to locate a suitable range of the AB and uses lin-chirps to obtain a precise AB. Using the simulations in NS-2, we have shown that HybChirp can obtain more accurate estimated bandwidth quickly with lower overhead of probing packets, compared with PathChirp and Pathload. It is also shown that HybChirp is good at adaptivity. This is an advantage to estimate the AB in real-time which rapidly reflects the changes of network environment.

In the near future, we will build a test bed to verify the efficiency of HybChirp. Moreover, we will apply Hybchirp into some internet applications and protocols that require accurate estimated bandwidth, and further evaluate its performance.

Acknowledgements

We acknowledge the support from Natural Science Foundation of Fujian Province of China (No. 2013J05101), Shenzhen City Special Fund for Strategic Emerging Industries (No. JCYJ20120830153030584), National Natural Science Foundation of China (No. 61379157), National Special Fund for Major Research Equipment and Instruments of China (No. 2011YQ03012417), the Scientific Research Fund of SiChuan Provincial Science and Technology Department (Nos. 2015GZ0333, 2014SZ0107), and Research Fund of Fujian Key Laboratory of Sensing and Computing for Smart City.

References

Ali, A.A., Michaut, F. and Lepage, F. (2006) 'End-to-end available bandwidth measurement tools: a comparative evaluation of performances', *Proceedings of the 4th International Workshop on Internet Performance, Simulation, Monitoring and Measurements IPS-MoMe 2006*, pp.1–14.

Angrisani, L., Dõantonio, S., Esposito, M. and Vardusi, M. (2006) 'Techniques for available bandwidth measurement in IP networks: a performance comparison', *Computer Networks*, Vol. 50, No. 3, pp.332–349.

Carter, R.L. and Crovella, M.E. (1996) 'Measuring bottleneck link speed in packet-switched networks', *Performance Evaluation*, Vols. 27–28, No. 4, pp.297–318.

Dovrolis, C., Ramanathan, P. and Moore, D. (2001) 'What do packet dispersion techniques measure?', *Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies*, Anchorage, AK, pp.905–914.

Downey, A.B. (2000) 'Using pathchar to estimate internet link characteristics?', *ACM SIGCOMM Computer Communication Review*, Vol. 29, No. 4, pp.241–250.

Guerrero, C.D. and Labrador, M.A. (2010) 'On the applicability of available bandwidth estimation techniques and tools', *Computer Communications*, Vol. 33, No. 1, pp.11–22.

Hu, N. and Steenkiste, P. (2006) 'Evaluation and characterization of available bandwidth probing techniques', *IEEE Journal on Selected Areas in Communications*, Vol. 21, No. 6, pp.879–894.

Imai, M., Sugizaki, Y. and Asatani, K. (2013) 'A new estimation method using RTT for available bandwidth of a bottleneck link', *International Conference on Information Networking*, Bangkok, pp.529–534.

Jacobson, V. (1998) 'Congestion avoidance and control', *ACM SIGCOMM Computer Communication Review*, Vol. 18, No. 4, pp.314–329.

Jain, M. and Dovrolis, C. (2002) 'Pathload: a measurement tool for end-to-end available bandwidth', *Proceedings of Passive and Active Measurements (PAM) Workshop*, pp.14–25.

Jain, R. and Routhier, S. (1986) 'Packet trains – measurements and a new model for computer network traffic', *IEEE Journal on Selected Areas in Communications*, Vol. 4, No. 6, pp.986–995.

Jin, G., Yang, G., Crowley, B. and Agarwal, D. (2001) 'Network characterization service (NCS)', *Proceedings of the 10th IEEE Symposium on High Performance Distributed Computing*, San Francisco, CA, pp.289–299.

Kavitha, G. and Sankaranarayanan, V. (2014) 'A novel resource selection framework to improve QoS in computational grid', *International Journal of Computational Science and Engineering*, Vol. 9, No. 1, pp.130–138.

Lai, K. and Baker, M. (2001) 'Nettimer: a tool for measuring bottleneck link bandwidth', *Proceedings of the USENIX Symposium on Internet Technologies and Systems*, p.11.

Melander, B., Bjorkman, M. and Gunningberg, P. (2000) 'A new end-to-end probing and analysis method for estimating band width bottlenecks', *IEEE Global Communications Conference*, San Francisco, CA, pp.415–420.

Park, H.Y. and Roh, B-H. (2010) 'Accurate passive bandwidth estimation (APBE) in IEEE 802.11 wireless LANs', *International Conference on Ubiquitous Information Technologies and Applications (CUTE)*, Sanya, pp.1–4.

Ribeiro, V.J., Riedi, R.H., Baraniuk, R.G., Navrati, J. and Cottrell, L. (2003) 'PathChirp: efficient available bandwidth estimation for network paths', *Proceedings of the Passive and Active Measurement Workshop*, San Diego, CA

Tursunova, S., Inoyatov, K. and Kim, Y.T. (2010) 'Cognitive passive estimation of available bandwidth (cPEAB) in overlapped IEEE 802.11 WiFi WLANs', *Proceedings of Network Operations and Management Symposium (NOMS)*, Osaka, pp.448–454.

Xie, Y., Zheng, T., Wang, Y.X. and Yuan, P.F. (2014) 'AProbing: estimating available bandwidth using ACK pair probing', *International Conference on Smart Computing Workshops*, pp.43–49.

Notes

1 The source code for HybChirp is available at https://github.com/TaoZheng/HybChirp.git.

2 The source code for PathChirp is available at https://github.com/TaoZheng/pathchirp-for-NS2.

3 The source code for Pathload is available at https://github.com/TaoZheng/Pathload-for-ns2.