Estimating End-to-End Available Bandwidth for Cyber-Physical Applications in Hybrid Networks

Hui Zhou[1,2], Chunyang Ye[1,2(✉)], Yucong Duan[1,2], Qi Qi[1,2], and Yu Zhang[1,2]

[1] State Key Laboratory of Marine Resource Utilization in the South China Sea, Hainan University, Haikou, China
{zhouhui,cyye,duanyucong,qqi,yuzhang2015}@hainu.edu.cn
[2] College of Information Science and Technology, Hainan University, Renmin Road No. 58, Haidian District, Haikou, China

Abstract. Available bandwidth estimation is very important for network operators, users, and bandwidth-sensitive applications. In the last 20 years, various techniques and systems have been proposed to estimate end-to-end available bandwidth. They were mostly tested in simulation or inside small-scale networks, but they can't consistently offer satisfying accuracy over the Internet. An active probing method SOProbe is proposed, and it measures end-to-end available bandwidth from only the installed host. The key idea of SOProbe is to identify the rate range where the available bandwidth resides. To archives this, SOProbe sends probe packets at selected transmission rates, and tries to catch the relationship between probe packets and available bandwidth.

Keywords: Active probe · Available bandwidth · Network measurement

1 Introduction

Mobile devices like iPhone and Android phones have become hugely popular, and widely used. Mobile Internet devices quickly spread over the world, their processing ability, built-in cameras, sensors, and pervasive cellular connections make them ideal platforms for constructing comprehensive cyber-physical applications. A cyber-physical application is a computer system that receives and responses to data from outside stimuli, and makes decisions that also impact the physical world [14].

Traditional cyber-physical applications include flight avionics, electronic medical devices, power grid control systems, and anything that can be managed by a remote computer through cable or wireless connection [5]. Cyber-physical applications can impact the real world and react to events, they always require strict quality and performance assurance. Mobile devices are equipped with a variety of sensors (such as ambient light sensors, gyroscope, accelerometers, GPS, microphones, and cameras) that cyber-physical applications use to measure environmental digitally. The combination of data from sensors with wireless networks and the growing data process power motivates more and more cyber-physical applications.

M. Huang et al. (Eds.): WICON 2016, LNICST 214, pp. 197–206, 2018.
https://doi.org/10.1007/978-3-319-72998-5_21

Though a lot of work has been done to make the operating environment of mobile devices seem to be like general computer, they are indeed different. A step further, mobile device push more rigid demand on their communicating capacity. First of all, bandwidth is always very spare. Second, cyber-physical systems tend to generate small-size IP packets. Third, short roundtrip time is preferred, thus packet replication, corruption, loss, and retransmission would largely affect the usage.

For many large-scale cyber-physical applications, the design is divided into three steps. First, application runs on mobile devices, e.g., iPhone or Android-based smart phone, connect to WIFI or the Internet through cellular network of ISP. Second, data packets traverse from ISP to server nodes located closed to the mobile devices, these nodes are generally named access servers. Third, a large amount of nodes hosted together to perform data-intensive computation, as well as to visit vast stored data. For load-balance and data replication, there are always many backup nodes spread over different racks, server rooms, or even cities.

Consequently, the performance bottleneck often exist between the second and the third steps. It is often the case that when mobile devices in a city all connect to a network node in another rack, or another server room, or even another city. It becomes crucial for the access servers to be intelligent to choose suitable nodes for performing data-intensive tasks.

This paper presents SOProbe, an original light-weight method that can quickly and precisely measures the end-to-end available bandwidth from the installed nodes. The key point of SOProbe is to identify a very narrow rate range of available bandwidth. As verified by our theoretical method, the dispersion of a long packet array will be enlarged only if the initial transmission rate is higher than available-bw. Therefore, SOProbe sends packet arrays at different initial transmit rates, and catches per-link dispersion of the arrays to estimate the relation between available bandwidth and the initial transmit rates. The efficiency and accuracy of SOProbe have been tested in carefully designed simulations.

This paper is organized as follows. Section 2 introduces the background. Section 3 presents the measurement methodology in details, and Sect. 4 sets up simulations to verify the accuracy and efficiency of SOProbe. Finally, Sect. 5 concludes the paper.

2 Related Work

Specifically, the available-bandwidth (available-bw) is defined as the maximum data rate that an end-to-end path can provide to a packet flow, without reducing the rate of the cross-traffic in the path [8]. Quite a few active probe techniques have been proposed. The first technique that tried to estimate available-bw was Cprobe [2]. Cprobe sends short arrays of ICMP echo packets to target host as close as possible, and calculates the obtained throughput from the head and the tail ICMP replies. The basic idea is that the time distance of a lengthy packet array is inversely proportional to available-bw.

However, Dovrolis proved that what the packet arrays catch is the asymptotic dispersion rate (ADR), instead of available-bw [8]. After Cprobe, quite a few methods that relied on both endpoints of a communication path were proposed. Delphi [12], for

example, estimate the volume of background traffic in network paths. However, since Delphi interprets queueing delays of all links as that of the bottleneck, it is not applicable in the presence of hidden bottleneck problem [16]. Because the initial transmit rate is adjusted according to the measurement result of previous probes, SOProbe is able to handle this problem when its bandwidth resolution is smaller than the bandwidth difference of the actual and fake bottleneck links.

A representational end-to-end available-bw estimation tool, i.e. Pathload, was introduced in [7], and was explained in [3]. Pathload doesn't simply report a number; instead, it outputs a rate range where available-bw would reside. In order to reduce the rate range, Pathload transmits packet arrays at different rates and watches their transmit delay at the other end of the path. Parallelly, SOProbe manipulates a rate range by probing a network path with packet arrays sent at selected data rates. But SOProbe is only installed on the source node, and it pay much more attention to the reverse path and router queuing effect.

A lot of techniques have also been presented to measure bandwidth with only the source node. However, these techniques mainly concentrated on measuring capacity, i.e. the physical communication rate of a link or a path, instead of available-bw. For example, pathchar [4], and the tailgating technique [1] measure per-hop link capacity. While Bprobe, nettimer, pathrate, and the PBM techniques catch path capacity [16]. In Parallel, Prasad demonstrated that layer-2 store-and-forward devices can strongly influence these methods [11]. But the layer-2 devices won't affect the accuracy of SOProbe since it relies on dispersion technique, instead of estimating delay.

Hu addressed the problem of bottleneck location and presented Pathneck [6]. Pathneck is based on the founding that background data interleave with probe packet arrays, thus changing the transmission time of packet array. By catching per-hop array transmission time, the location of bottleneck, and the narrowed rate of available-bw, can be inferred. The TTL configuration of packet array in SOProbe is similar to the recursive packet train adopted by Pathneck. However, SOProbe provides more accuracy since it can precisely control the inter-packet gap during estimation.

3 SOProbe Measurement Methodology

SOProbe is a single-end method and it uses the link congestion effect [16] in an iterative probing manner. SOProbe is built on the observation that, if a packet array is inserted into a network path at data rate r_p that is higher than available-bw, communication time must be enlarged by some links along the path; if r_p is not higher than available-bw, the communication time keep unchanged. As a result, by catching whether the path expands the packet arrays injected at selected rates, the available-bw can be computed.

The path is a sequence of store-and-forward links [16] that transfer packets from source host R_0 to destination host R_n through a set of intermediate routers $R_1, R_2 \ldots R_{n-1}$. Link L_i is the data connection from R_i to R_{i+1}. Since, at any time, L_i is either idle or forwarding packets at its capacity rate (maximum speed) C_i, its available-bw is defined as the spare bandwidth over a time interval τ:

$$A_i(t, t+\tau) = \frac{1}{\tau}\int_t^{t+\tau} \left(C_i - \lambda_i(t)\right)dt \tag{1}$$

In (1), $A_i(t, t+\tau)$ is the available-bw of L_i in the interval $(t, t+\tau)$, and λ_i is the rate of cross-traffic in L_i. Particularly, the end-to-end available-bw is the path available-bw, and is also the minimum link available-bw of the path:

$$A(t, t+\tau) = \underset{i=0\ldots n-1}{Min}\left\{A_i(t, t+\tau)\right\} \tag{2}$$

Here, τ is the time taken by an estimation instance. The link that maintains the minimum link available-bw is the bottleneck link. From a specific instant, the source node sends out an array of N packets to another node along a network path at data rate r_p. Every packet is of size S and is equally spaced. Figure 1 shows the array as it traverses on L_i. Δ_i is the transmission time between the first and the last packets, the per-packet dispersion (PPD) in L_i is

$$p_i = \frac{\Delta_i}{N(i)-1}(1 \le i \le n) \tag{3}$$

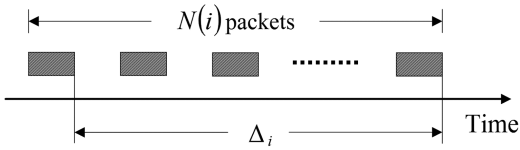

Fig. 1. A probing packet train on L_i.

Here, $N(i)$ is the number of packets that the array remains when it transmit on L_i. Then we have

$$\Delta p_i = p_i - p_{i-1}\,(1 < i \le n) \tag{4}$$

If $n = 1$, the path consists of one link. The path always keeps packet dispersion because PPD on the first link is also the PPD on the last link.

We focus on path effect when $n > 1$. Note that the source node can't send packets at rate higher than A_i, i.e. $r_p \le A_1$. As a result, $r_c^1 + r_p \le r_c^1 + A_1 = C_1$. Consequently, L_1 transmits all data packets without delay. This means that r_c^1 maintains unchanged since background traffic on L_1 is not affected by the probe packet array. Additionally, the background data on L_2 comes from L_1 and somewhere else; r_c^2 is unaffected because r_c^1 doesn't change and relative path features are regarded as fixed during a measurement instance.

It needs more attention to study how the packet array enters L_2 from L_1. In Average, in a p_1 period, one probe packet moves into L_2. Meanwhile, the amount of cross data packets entering L_2 is $X_2 = r_c^2 \cdot p_1$. As a result, the amount of packets that L_2 receives in one p_1 period is $S + X_2$. Since $r_p \le A$, we have $r_p \le A_2$ and

$$S + X_2 = \left(r_p + r_c^2\right) \cdot p_1 \le \left(A_2 + r_c^2\right) \cdot p_1 = C_2 \cdot p_1 \tag{5}$$

Therefore, L_2 is able to transmit all the packets from R_1 without the need of queue. So the rate of packet array remains r_p as the array enters L_2, and $p_2 = p_1$, i.e. $\Delta p_2 = 0$. Meanwhile, the background traffic on both L_2 and L_3 isn't influenced by the probe packet array. Inductively, this can be proved in the subsequent links. Suppose $n > 2$, $\Delta p_i = 0$ for $i = 2, 3 \cdots k$ $(2 \le k \le n-1)$. The rate of probe array on L_k is r_p; r_c^k and r_c^{k+1} are not affected by the probe array.

When the probe packet array moves in L_{k+1} from L_k, only one probe packet moves into L_{k+1} in a p_k period. In average, the amount of background traffic goes into L_{k+1} from R_k in that p_k period is $X_{k+1} = r_c^{i+1} \cdot p_k$. As a result, the total network amount that L_{k+1} receives during p_k period is $S + X_{k+1}$. By definition, $r_p \le A \le A_{k+1}$.

$$S + X_{k+1} = \left(r_p + r_c^{k+1}\right) \cdot p_k \le \left(A_{k+1} + r_c^{k+1}\right) \cdot p_k = C_{k+1} \cdot p_k \tag{6}$$

Therefore, all income packets can be delivered out by L_{k+1} without being queued, $\Delta p_{k+1} = 0$ and the rate of probe array on L_{k+1} is still r_p. In addition, r_c^{k+1} and r_c^{k+2} remain unchanged. Finally, we conclude that $\forall i \in (1, n]$, $\Delta p_i = 0$.

Because available-bw is the minimum link available-bw of the path and $r_p > A$, there is at least one link whose available-bw is lower than r_p. Precisely, a link is named *the first narrow link* if its available-bw is first less than r_p. Evidently, L_1 can't be the first narrow link since $r_p \le A_1$.

Assume that L_2 is the first narrow link, then $r_p > A_2$. As shown above, when the probe packet array moves from L_1 to L_2, the total amount of traffic that enters L_2 in a p_1 period is $S + X_2$, so

$$S + X_2 = \left(r_p + r_c^2\right) \cdot p_1 > \left(A_2 + r_c^2\right) \cdot p_1 = C_2 \cdot p_1 \tag{7}$$

This means that L_2 must take more time to deliver out the packets it receives in p_1. Consequently, internal queue is built up and the transmission time of probe array is enlarged. In average, L_2 expands the PPD to be

$$p_2 = \frac{S + X_2}{C_2} > p_1 \tag{8}$$

Therefore, $\Delta p_2 = p_2 - p_1 > 0$.

SOProbe uses binary search algorithm to identify the rate range. Initially, $R_{\min} = 0$ and $R_{\max} = A_1$. Since $r_p \le A_1$, A_1 is the maximum probe data rate that the source node can achieve. Suppose that the n^{th} packet array is transmitted at rate $r_p(n)$, and the first probe rate $r_p(1) = A_1/2$.

If $r_p(n) > A$, then $R_{\max} = r_p(n)$
If $r_p(n) \le A$, then $R_{\min} = r_p(n)$

$$r_p(n+1) = \frac{(R_{\min} + R_{\max})}{2}$$

When $R_{\max} - R_{\min} \leq \delta$, the process finishes. Here, δ is the bandwidth resolution that indicates how small the estimated rate range could be. This process eventually converges to a rate range $[R_{\min}, R_{\max}]$ after probe the target path with $\log_2(A_1/\delta)$ different data rates.

To understand the relation between available-bw and probe data rate, the source node should be able to obtain PPD of its probe packet array. According to the ICMP protocol that is widely used by network infrastructure, we design a probe packet array that would trigger intermediate router replies. In particular, N IP packets of size S are equally spaced by ΔP, and are sent to the destination at rate r_p. Obviously, $\Delta P = p_1$. S and ΔP are carefully selected, and

$$r_p = \frac{S}{\Delta P} \tag{9}$$

Additionally, the time-to-live (TTL) parameters of probe packets are $\{1, 2 \cdots (n-1), n, n, (n-1) \cdots 2, 1\}$. Here, n is the number of links and $N \geq 2{\cdot}n$. Setting TTL parameters in this way makes each intermediate router along the path response two ICMP packets back to the source. When the array moves to the first router (R_1), its first and last packets expire because their TTL parameters are 1. Consequently, these two packets are dropped and R_1 sends two ICMP packets back to the source [10]. The other probe packets of the train are forwarded to R_2 after their TTL are decremented by 1. Every subsequent router along the path repeats the above process (Fig. 2).

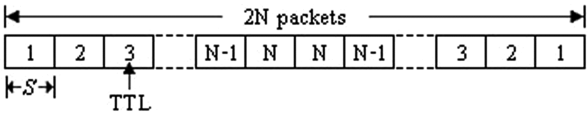

Fig. 2. Structure of the probing packet train.

As a result, each intermediate router sends back two ICMP packets. The source measure the interval between two ICMP packets from router R_i to estimate Δ_i, i.e. the dispersion of packet array in the incoming link of that router.

4 Simulation Verification

SOProbe have been tested in Network Simulator (NS) [9] - a widely used simulate platform. SOProbe was developed in the agent and application levels of NS. In real networks, it is the intermediate router that counts the TTL field of every incoming IP packet, decrements it and replies with ICMP error if the TTL expires. While in NS, it is the link that checks TTL and will drop the expired packets without any reply. To handle this problem, we implemented an application and attached it to every node except the

source node. This application sends 56-byte ICMP packet back to the source when it receives a probe packet. Therefore, the source directly sends packets to the ICMP application attached to a node if the source wants that node to send back ICMP replies.

Our experiment use a linear topology. In this topology, nodes 0 and 7 are the source and destination, nodes 1–6 are intermediate routers. C_4 (the X) is configured according to different use cases. All links are duplex; capacities of links are in the unit of bits per second. In addition, every link applies the drop-tail queueing principle. Table 1 lists parameters used in the experiment.

Table 1. Parameters used in the simulation

Parameter	Description
N	The number of packets in a train
S	The packet size
g	The PPD gauge parameter
δ	The bandwidth resolution
β	The loss gauge parameter, and $\beta = 100\%$
u_t	The utilization of a path

In experiments, one hop persistent (OHP) background traffic is applied to the path. Precisely, an OHP packet stream comes from four CBR sources, and will move out of the path after one link. Packets of each OHP traffic stream are carefully set as follows: 15% 576 bytes, 20% 1500 bytes, 50% 40 bytes and 15% randomly distributed between 40 and 1500 bytes, similar to the Internet packets measured in [13]. All link usage are equal. Here, $X = 10$ Mbps and the path is 40% utilized indicates that all links are 40% utilized, i.e. the background traffic on each link is {20, 32, 24, 4, 28, 8, 40 Mbps}. Precisely, there are two paths in Fig. 3: the forward path from node 0 to node 7, and the reverse one from node 7 back to node 0. What we are measuring is the forward path available-bw.

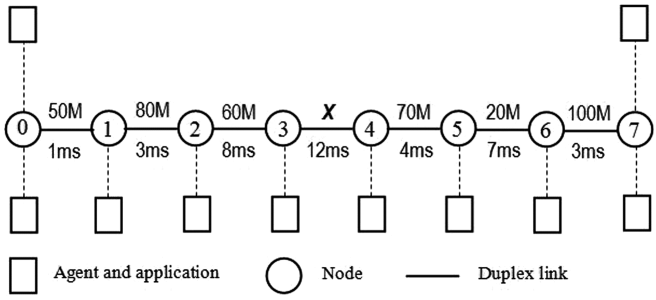

Fig. 3. Simulation topology.

We first demonstrate one SOProbe estimation process. Then we test SOProbe with bidirectional background traffic and different bottleneck link positions. The effect of

path features is also studied. After that, we record the convergence time that SOProbe process takes.

X = 10 Mbps, all forward links are 20% usage, then available-bw is 8 Mbps. Figure 4 demonstrates a SOProbe measurement process that finishes after 6 probes. The initial rate range of available-bw is [0, 50 Mbps], and r_p = 25 Mbps. SOProbe catches that $r_p > A$ because it detects PPD expansion. As a result, it shrink the rate range to [0, 25 Mbps] and generates the next probe packet array at 12.5 Mbps. PPD expansion is still there, so SOProbe decreases r_p to 6.25 Mbps and tries again. This time, there isn't a PPD expansion; SOProbe sets the rate range to [6.25, 12.5 Mbps] and probes at 9.375 Mbps. Particularly, at the 6th probe, the range is [7.813, 9.375 Mbps], and r_p = 8.594 Mbps. Once again, a PPD expansion is detected. After this, the range [7.813, 8.594 Mbps] is outputted because it meets the limit of bandwidth resolution (δ = 1 Mbps in this case).

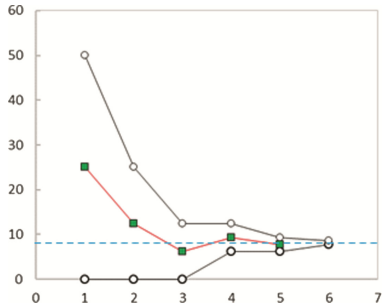

Fig. 4. A SOProbe measurement process.

X = 10 Mbps, both forward and reverse paths are 20%, 40%, 60%, and 80% utilized, respectively. Figure 5 reports the rate ranges after performing 50 SOProbe measurements for the designed 16 cases. In the following simulations, SOProbe are executed 50 times for each setting. In particular, "FP $u_t = x\%$" means the forward path is $x\%$ utilized. In the same way, "RP $u_t = x\%$" denotes that every link of the reverse path is $x\%$ utilized.

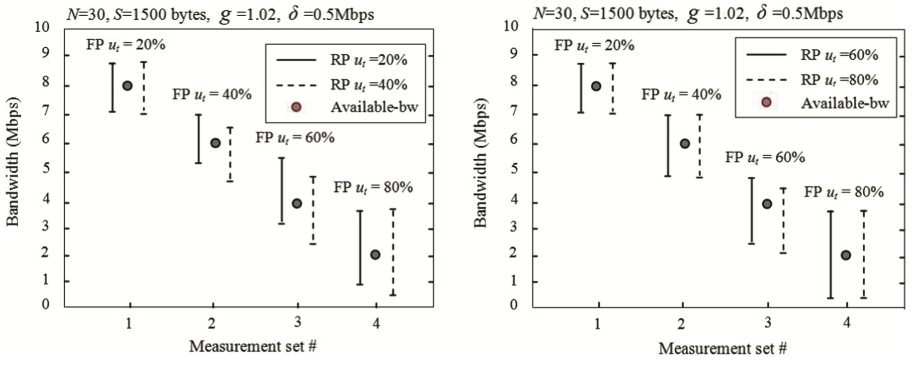

Fig. 5. Results under different loaded conditions, link utilization conditions.

The first founding is that background traffic in the forward path mainly determines the measurement accuracy. When the forward path was no more than 40% utilized, SOProbe reports rate ranges where available-bw nicely resides. As the forward path became more and more congested, SOProbe lost the accuracy slowly. Though the outputted range still includes the available-bw when forward path was 80% utilized, the range fluctuates and cover a comparably bigger region.

Another observation is that background traffic in reverse path influences the measurement accuracy lightly. When the forward path is underutilized ($u_t = 20\%$), the estimations are stable no matter how high is the traffic load in the reverse path. When forward path is 40% or 60% utilized, background traffic in the reverse path would affect the measurement. Finally, as the forward path comes to 80% utilized, background traffic in forward path dominates the measurement, and the influence of the reverse path traffic decreased.

The simulation reveals that a lengthy packet array of large network packets would interleave with the existing background traffic. Compared with probe packets (1500 bytes), ICMP packets (56 bytes) returned by nodes 1–7 are affected by background traffic slightly. SOProbe isn't sensitive to the background traffic in reverse path. Meanwhile, SOProbe can accurately estimate the available-bw when the forward path is not much loaded.

In the above experiments, the time taken by one SOProbe measurement highly depends on N and δ. The longer the packet array is, the more time that transmission requires. A smaller resolution parameter δ often leads to more probe tries. Every packet array is transmitted at a selected rate, different packet size would only result in the space in between packets. Analogously, parameter g would affect the estimation accuracy, but it isn't relative to the number of probes that one estimation takes.

Table 2 shows that, one single SOProbe estimation instance uses nearly 2–3 s. In particular, all high estimation time occurred only when the forward path is overloaded. According to a research founding, the Internet path properties do not change much on the scale of hours [15], SOProbe can measure available-bw in a timely manner.

Table 2. Convergence time of SOProbe

N	δ (Mbps)	Convergence time (min, mean, max) (seconds)
20	0.5	1.90, 2.02, 2.35
20	1.0	1.62, 1.77, 1.96
30	0.5	2.52, 2.83, 3.11
30	1.0	2.06, 2.40, 2.88

5 Conclusions

A novel active probe technique, SOProbe, was presented to estimate end-to-end available bandwidth from only the source node. With SOProbe, one can estimate available bandwidth from a single source, this enable smart devices and applications to estimate and select better network paths actively. The key idea of SOProbe and its working

process have been verified in NS simulation with carefully designed topology and comprehensive cross-traffic setting.

Acknowledgments. Our work is supported by National Natural Science Foundation of China under grant No. 61379047, 61562019, 61662019, 61662021, the Natural Science Foundation of Hainan Province under grant No. 20156223, the Major Science and Technology Project of Hainan Province under grant No. ZDKJ2016015.

References

1. Allman, M., Paxson, V.: On estimating end-to-end network path properties. In: Proceedings of ACM SIGCOMM, pp. 263–274 (1999)
2. Carter, R., Crovella, M.: Measuring bottleneck link speed in packet-switched networks. Perform. Eval. **27**(28), 297–318 (1996)
3. Dovrolis, C., Ramanathan, P., Moore, D.: Packet dispersion techniques and a capacity estimation methodology. IEEE/ACM Trans. Netw. **12**, 963–977 (2004)
4. Downey, A.: Using pathchar to estimate internet link characteristics. In: Proceedings of ACM SIGCOMM, pp. 222–233 (1999)
5. Govindan, R., Minei, I., Kallahalla, M., Koley, B., Vahdat, A.: Evolve or die: high-availability design principles drawn from Googles network infrastructure. In: Proceedings of ACM SIGCOMM (2016)
6. Hu, N., Li, L.: Locating internet bottlenecks: algorithms, measurements, and implications. In: Proceedings of ACM SIGCOMM, pp. 41–54 (2004)
7. Jain, M., Dovrolis, C.: Pathload: a measurement tool for end-to-end available bandwidth. In: Proceedings of Passive and Active Measurements Workshop, pp. 14–25 (2002)
8. Jain, M., Dovrolis, C.: End-to-end available bandwidth: measurement methodology, dynamics, and relation with TCP throughput. IEEE/ACM Trans. Netw. **11**(4), 537–549 (2003)
9. Network Simulator. http://www.isi.edu/nsnam/ns/
10. Paxson, V.: End-to-end routing behavior in the internet. IEEE/ACM Trans. Netw. **5**(5), 601–615 (1997)
11. Prasad, R., Dovrolis, C., Mah, B.: The effect of layer-2 store-and-forward devices on per-hop capacity estimation. In: Proceedings of IEEE INFOCOM, pp. 243–254 (2003)
12. Ribeiro, V., Coates, M., Baraniuk, R.: Multifractal cross-traffic estimation. In: Proceedings of ITC Specialist Seminar IP Traffic Measurement, Modeling, and Management (2000)
13. Thompson, K., Miller, G., Wilder, R.: Wide-area internet traffic patterns and characteristics. IEEE Netw. **11**, 10–23 (1997)
14. White, J., Clarke, S., Groba, C., Dougherty, B., Thompson, C., Schmidt, D.: R&D challenges and solutions for mobile cyber-physical applications and supporting internet services. J. Internet Serv. Appl. **1**(1), 45–56 (2010)
15. Zhang, Y., Duffield, N.: On the constancy of Internet path properties. In: Proceedings of ACM (2001)
16. Zhou, H.: Measuring available bandwidth for smart cyber-physical applications. Tsinghua Sci. Technol. **16**, 601–610 (2011)