On Lower Estimating Internet Queuing Delay

Attila Csoma, László Toka, and András Gulyás

***Abstract*—With the advent of online social media, latency became a primary factor and large Content Delivery Network (CDN) providers optimize their networks for. While many ingredients (e.g. propagation delay, transmission delay, processing delay) of the end-to-end latency can be adequately characterized, capturing queuing delay seems to be a difficult task due to the complex and often unpredictable nature of Internet paths and the imperfect tools used for measurements. Dealing with the queuing delay is so challenging that most papers concerning end-to-end latency completely ignore it for "simplicity". In this paper we take this view to the extreme and try to interpret the cause of various delays between the same endpoints as measurement artifacts, such as continuous changes in end-to-end paths and the imperfection of measurement tools. We arrive to the conclusion that there is a significant amount of points in end-to-end delay measurements, which cannot be explained by these artifacts even if we do our best for doing so. The only plausible explanation for these points is the presence of significant queuing delay comparable with the delay caused by all the other factors (propagation, transmission and processing).**

I. Introduction

With the proliferation of massive media consumption through the Internet two opposing phenomena are seen: the ever-increasing load on the networks leads to congestion at some bottlenecks during peak hours while the requirement of instant access to content has become standard. The queuing delay at network elements is at the same time the result of the former and the obstacle to the latter. The motivation of our work stems from the fact that although the academic literature tackling the performance of the Internet is vast, one cannot find any results that would yield a rough estimate of expected end-to-end delays in commercial networks. We argue that this is the consequence of the known issues of measuring delays in the context of the Internet.

First of all, the available **measurement tools** are imperfect. A thorough survey in [1] presents all the improvements and add-ons that had been done to standard measurement tools, such as *ping* and *traceroute*, before 2007. The survey summarizes all the problems that might hinder correct path detection in the Internet on every level: missing nodes and links [2] [3], false links due to load balancing [4] [5] [6], IP aliasing [7] [8] [9], IP dynamism [10], multi-homing [11], MPLS (Multiprotocol Label Switching) tunneling [12] just to name a few. In order to overcome these issues, numerous measurement tools have been proposed to improve and/or to complement existing methods.

L. Toka is with the MTA-BME Information Systems Research Group, and this paper was supported by the Janos Bolyai Research Scholarship of the Hungarian Academy of Sciences.

L. Toka and A. Gulyás are with the MTA-BME Future Internet Research Group

All authors are with the HSNLab, Dept. of Telecommunications and Media Informatics Budapest University of Technology and Economics

Secondly, if the measurement tools experience different chains of links when traversing the Internet between the same source-destination pairs multiple times, the topology results can be confusing and the measured delays might vary. In order to grasp the high level dynamism of the network, and of course to unfold the non-trivial results of the measurements performed for longer periods, the research community has also focused on the features of **path variation**. The authors of [13] state that Internet paths are heavily dominated by a single prevalent route, but the time periods over which routes persist show wide variation, ranging from seconds up to days. Slightly easing the complexity, in [14] the authors argue that delay variations are mostly observed within the routes, and only 15-20% of the source-destination pairs show a significant difference between the delays of different routes. In [15] the authors map delay figures to specific segments of both the forward and reverse paths which in turn can be asymmetric.

Due to these issues, our understanding of queuing delay in general, based on the art-of-the-state, is that measuring it and then building a model of it is difficult, if not impossible. Most works try to neglect it, since for their goals it is just an unnecessary noise, other works try to grasp the nature of it but only in a well-confined, measurable setting. Due to the fact that most of these works base their conclusions on measurements carried out either fully in an academic network, e.g., among vantage points placed in PlanetLab, or on single links, we argue that the effects of queuing is much higher than what it seems in those results.

In this paper we show that the queuing delay in today's networks is unfortunately not negligible and we present a novel approach that provides a qualitative estimate of it. Our method first assumes no queuing delay and tries to attribute the observed delay variations to other causes, but in the end fails to do so, leading to confute the initial hypothesis. To achieve this goal, we take standard measurement tools, a bulk of network measurement results and an adversary analytic approach. Our contribution can be summarized in the next points:

- collecting dense periodic *ping* and *traceroute* measurements to a set of hosts worldwide over weeks;
- geolocalizing the hosts recorded in the *traceroute* measurements with Spotter [16];
- merging paths that are equivalent in terms of propagation delay;
- assigning most of the recorded *ping* RTT to one of the merged paths in an adversary manner;
- distilling the fraction of *ping* measurements that cannot be reasoned by path variation.

The paper is constructed as follows: in Sec. II we give an outlook on the related work considering queuing delay; in Sec. III we describe our simplistic model and the methods we apply; we lay out our results in Sec. IV; and finally we discuss possible concerns about our conservative estimation method and sketch the direction of our future work in Sec. V.

II. Related work

One can find a myriad of scientific papers tackling network **delay modeling** in more details. At the lowest level, the measuring tools themselves incur some delay on their path [17]. Going forward, authors of [18] say that the network processing delay can reach the magnitude of long-distance propagation delay and thus becomes a significant contributor to the overall packet delay. In [19] the authors found that end-to-end delays on a given path mostly follow Gamma-like shape with heavy-tail. Others showed that end-to-end network delay measurements can be modeled using a finite combination of Weibull distributions [20]. In order to estimate geographic distances between network routers and measurement nodes more precisely, in [21] a detailed path-latency model is proposed. Building on it, the same authors implemented an IP **geolocalization** service estimating the position of Internet devices with remarkable precision [16] [22]. In [23] the authors propose not less than **diagnosing network component failures** based on the difference of the measured and the expected delay.

While in the aforementioned geolocalization models the queuing delays are deliberately ruled out, some works are dedicated to model delays specifically due to high network load, called as **queuing delay**. By sending UDP probe packets at regular time intervals, and changing the regularity to different time scales, the author of [24] showed rapid fluctuations of queuing delays. In [25], the authors present an analytical approach for estimating the queuing delay distribution on an Internet link carrying realistic TCP traffic. In [26] such findings are presented that low rank tiers have undersized links, hence queuing, although altogether is not typical to inter-AS links, can occur there. [27] reveals that the average queuing delay on different network segments spans more than two orders of magnitude and follows closely a log-normal distribution. They found that the average queuing delay is roughly 1 ms. The authors of [28] show that the 99th percentile variable delay remains under 1 ms over several hops and under link utilization of below 90% on a bottleneck.

Bufferbloat is a recently introduced phenomenon that causes queueing delay. It occurs when packets suffer unnecessary delays because of unmanaged and oversized buffers which hinder TCP from functioning properly. Authors of [29] revived the term bufferbloat and described the phenomenon and its causes in details. Based on their work, [30] tried to capture this delay type and measure its significance by passively monitoring BitTorrent traffic. [31] analyzed bufferbloat in a testbed with typical network device buffer architecture. Although research results typically suggest that the extent of bufferbloat is not so significant, the question is still open and it remains an actively researched area (e.g. in [32]). Note that bufferbloat is only one type of queueing delay, but in this paper we refer to all types of delays that add up to the time minimally required to traverse through a given route as queueing delay.

III. Internet queuing delay analysis

Our qualitative queuing delay analysis will take the following steps: (i) we define a delay model and clarify what we mean by queuing delay, (ii) as opposed to standard techniques of eliminating measurement artifacts, we take an adversary approach and try to attribute the measurement results that cannot be explained solely with propagation, transmission and processing delay *completely* to measurement artifacts and (iii) show with extensive measurements that this is impossible. This means that a significant portion of our end-to-end delay measurements can be explained *only* by assuming serious queuing delays on the Internet paths, at least comparable with the end-to-end delay.

A. Delay model

Based on previous works (e.g [21]) delay suffered by a packet traversing to a destination can be described as:

$$D = D_{tran} + D_{prop} + D_{proc} + D_{que} + D_{icmp} \qquad (1)$$

where D is the time required for a packet to reach its destination, i.e., one-way delay. Transmission delay D_{tran} represents the delay caused by pushing all bits of a given packet into the transport medium with given capacity. Assuming a link with at least 1 Gbps data-rate (R) and overestimating ping message's size with 100 byte (S) the calculated D_{tran} would be $S/R = 10^{-6}$ s. The propagation time, i.e., the time spent to send the signal representing the packet through the used transport medium, is given in D_{prop}. The value of this component depends from the distance traversed by the signal and at longer links it can be significant, i.e., hundreds of ms. Because IP routing require some search in routing tables and packet modifications, there is a so-called processing delay denoted by D_{proc}. Relying on [33] ICMP packet processing time is in the order of μs and the difference between best and worst cases is only 5 μs. D_{que} stands for the queuing delay. To handle delay caused by the destination node by processing and generating answer for an ICMP request, we also have a D_{icmp} member. Based on [21] the value of D_{icmp} falls typically between $300 - 600$ μs.

Many papers in the literature assumes $D_{que} = 0$ for simplicity. From now on we follow the spirit of these papers and try to attribute the variance in our latency measurements to the variance in the other factors (D_{tran},D_{prop}, D_{proc} and D_{icmp}) plus measurement artifacts, e.g., multiple fake routes recovered by `traceroute`.

B. Measurement data

`Ping` and `traceroute` are well-known tools designed for measuring delays and discovering topology on the Internet. While they are widely used in scientific measurements, they have their shortcomings.

`Ping` measures round-trip time between two nodes over the Internet, but the forward and reverse routes between the two nodes could significantly differ from each other, e.g., due to the hot potato effect [26]. Moreover, there is an additional delay at the destination side: the time required to process the ping request and generate a response for it. Although there are tools like OWAMP [34] (One-Way Active Measurement Protocol) to measure one-way delay and by-pass the inaccuracies caused by the return path, these require time synchronization between endpoints, which assumes access to both of them. However, when measuring the packet delay to a random node over the Internet, access to that node is typically not be possible.

`Traceroute` is the trivial tool for discovering IP level routes between endpoints. Although `ping` has an option to record the return path of packets using its "record route" option, it is barely supported and its output usually differs from that of `traceroute`. In addition, it can not store more then nine IP addresses. `traceroute` was designed to record IP hops on a forward path to a destination node. Nevertheless, its weakness against load balancers ([35]) makes its results unreliable. To avoid `traceroute`'s anomalies [4] developed `paris-traceroute` with the capability of mitigating the effects of load balancing nodes on route discovery. As described in [35], `paris-traceroute` also has its own limitations, e.g., identifying a simple router as a load balancer and overestimate the number of load balanced paths.

From our host deployed at the campus of BME (Budapest University of Technology and Economics) we schedule `ping` measurements towards various destinations at every 8 seconds. The RTT results of these measurements give us the value of D for a given packet. In parallel we continuously monitor all the possible paths our `ping` packets can take by running `paris-traceroute` to reach state-of-the-art accuracy. In addition we geotag the hops in these routes with the most accurate latitude and longitude values we can get from the Spotter IP geolocation tool [16]. At the end in one measurement we get the values of D for all packets and the corresponding possible paths with geographical tags the packets could have taken.

C. *Inference method*

Extracting queuing delay for a given packet directly from `ping` and `paris-traceroute` measurements is impossible as they cannot tell which packet traversed which path. Our idea is to simply by-pass this problem and work out an inference method (see Fig. 1) which can provide a conservative lower bound on the queuing delay the packets suffer without knowing over which path they are actually transmitted. We do the following: (i) we take the paths returned by `paris-traceroute` and group them into geographically equivalent classes with the geotags provided by Spotter, direclty defining D_{prop}, (ii) define a delay interval for these path classes representing the variation of RTT on these paths according to $D_{tran}+D_{proc}+D_{icmp}$ and (iii) try to maximally cover the results of the RTT values obtained from our `ping` measurements with these intervals. The uncovered measurement points indicate RTT values which cannot be explained by assuming $D_{que}=0$. The analysis of these uncovered points then give us conservative lower bounds on the end-to-end queuing delay. We describe these steps in details:

Geographically equivalent classes – Since in our delay model of Sec. III-A D_{tran} and D_{icmp} are quasi-constant, D_{proc} depends only on the number of hops and $D_{que}=0$,

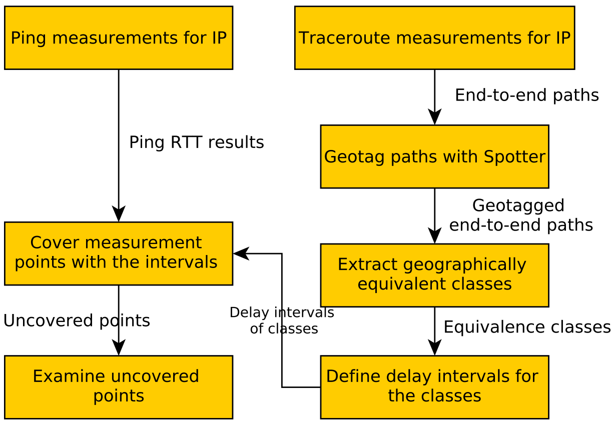
Ping measurements for IP
Traceroute measurements for IP
End-to-end paths
Ping RTT results
Geotag paths with Spotter
Geotagged end-to-end paths
Cover measurement points with the intervals
Extract geographically equivalent classes
Uncovered points
Delay intervals of classes
Equivalence classes
Examine uncovered points
Define delay intervals for the classes

Fig. 1. The measurement process used get a conservative estimation on the queuing delay.

the only highly variable parameter is D_{prop}, which in turn depends solely on the geographical path the packet takes. In order to model the measurement set in a tractable manner, we cluster the paths recovered by `paris-traceroute` into geographically equivalent classes: we group measured paths of which the geographical footprint can be considered the same.

Building on the nodes found by `paris-traceroute` and positioned by Spotter, we must take the precision of Spotter into account. This is said to be 50 km, so we group paths that differ less than 50 km in their node positions: we consider two measured paths as geographically equivalent, hence belonging to the same class, if their *city-wise* geographical footprints are identical. Within an equivalence class the variation of D_{prop} among the paths grouped together is less than 2 ms for the following reasons. Light travels 50 km in fiber within 0.3-0.4 ms and the hop count difference of the paths in the same class is usually below 4-6 hops (Table I supports this argument by showing that only an extremely small ($<0.5\%$) fraction of our classes contain paths with hopcount difference larger than 4 hops).

TABLE I
HOP COUNT DIFFERENCE DISTRIBUTION

max hop diff.	0	1	2	3	4	5	6	7	8	9
no. of samples	1374	78	29	6	0	4	1	2	0	1

The classes are built from `paris-traceroute` results by first merging nodes using the 50 km threshold, i.e., nodes closer than 50 km to their previous neighbor are removed from comparisons. Next, a route is merged into a class in which all routes travel through exactly the same city-level areas; if there is no such class a new one is created. In the end D_{prop} for paths within a class is nearly the same.

Delay intervals for geographically equivalent classes – Here we give an estimate on the range of RTTs the paths belonging to an equivalence class might carry. In other words, we provide the confidence interval within which the delay measurements of different paths of the same class must fall due to their similarity in length. First we indentify the delay components which can vary, assuming 0 queuing delay all along. D_{icmp} only occurs at the destination and as shown in Sec. III-A its fluctuation is negligible: $\bar{D}_{icmp}=0$ (we denote the variance of a delay component by $\bar{D}$). Although D_{proc} is only in the μs order of magnitude, it occurs at every

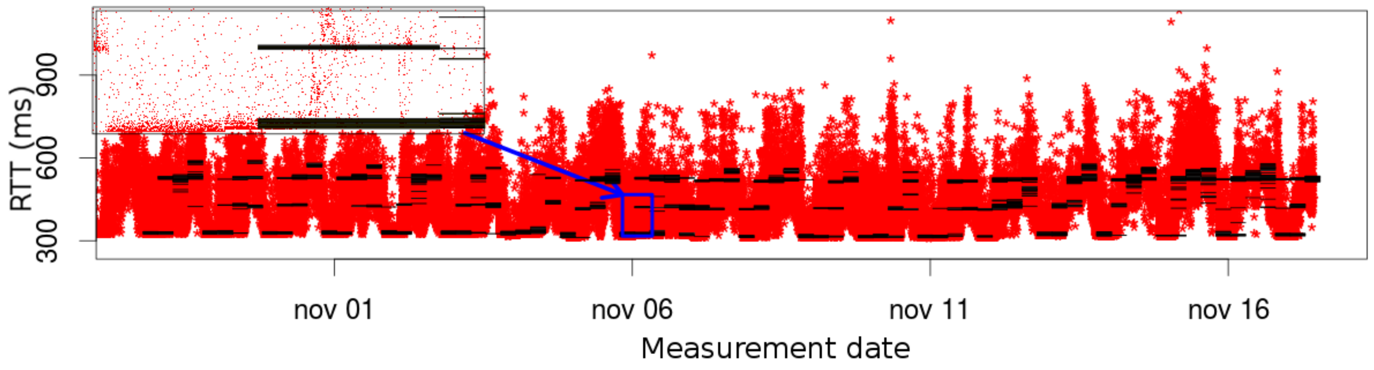

Fig. 2. RTT measurements covered with delay intervals associated to routes. x-axis represents measurement time, y-axis RTT values in ms

node and easily could scale up on a longer path. However, assuming a route with 60 hops (highest possible length in our measurements) and previously mentioned 5 μs worst case variance for it then $\bar{D}_{proc} = 300\ \mu s$ still negligible. As shown previously for a slower 1 Gbps link D_{tran} is 0.001 ms. Assuming two routes in the same class: one with 60 hops (highest possible length) and with 1 Gbps links, the other with supposedly 0 transmission time on all its links, the difference in D_{tran} is $60 * 0.001 = 0.06$ ms, hence still negligible ($\bar{D}_{tran} = 0$).

The last delay component is D_{prop}, and we introduced the geographically equivalent classes exactly for modeling the variance of this term. Since we merge nodes on the granularity of cities (with 50 km threshold on the distance), we possibly bring in a 0.3 ms large variance to D_{prop} with every merge as discussed above. Thus the variance of path delays in a given class is:

$$h\bar{D}_{tran} + h\bar{D}_{proc} + \bar{D}_{icmp} + m\bar{D}_{prop} \approx m\bar{D}_{prop}, \tag{2}$$

where h denotes the hop count of the longest path in the class and m denotes the maximal merging steps performed to merge geographically equivalent nodes in a measured path, finally $\bar{D}_{prop} = 0.3$ ms is assumed. E.g. if an equivalence class contains a path $A \rightarrow B \rightarrow C \rightarrow E$ where B is in the same city as C, then $h = 4$, $m = 1$.

Interval covering of the `ping` measurement data – Now we can fit these confidence intervals to the ping measurement data to cover the maximal number of points, thus explaining most results we can with variation in $D_{tran} + D_{proc} + D_{icmp} + D_{prop}$ plus measurement artifacts.

It is known from [36] that optimal covering of a dataset with intervals is NP-hard. To solve this problem on our ping dataset we use greedy heuristics. We sort the interval widths in a decreasing order. We take the first interval, and go through our dataset with the interval as a sliding window and take the maximum number of points the interval can cover. We remove the covered points from the dataset and continue with the next interval. As an example consider a vector of RTT values $x = \{80, 50, 65, 52, 120, 66, 44, 85\}$ and that we have two classes, hence two intervals, of length 10 and 20 ms. Starting with the 20 ms interval we find that it can cover at most 4 points (50,65,52,66). Deleting these from x and proceeding with the 10ms interval, we can cover at most 2 points (80,85). At the end we have two uncovered points (44,120).

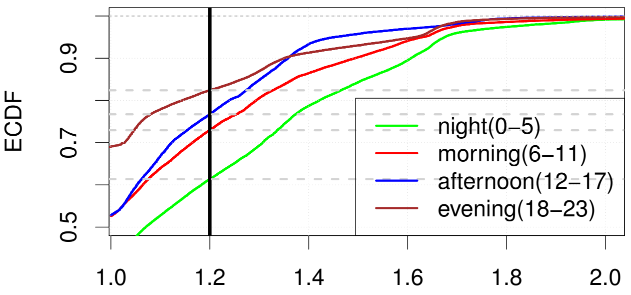

Fig. 3. ECDF of the ratio of uncovered RTT values to its parent interval.

IV. Results

In order to perform our adversary approach described in Sec. III we created a testbed and carried out `ping`, `paris-traceroute` and Spotter measurements to random IP addresses. On the measurement data we used the presented node merging, path grouping and RTT delay interval covering algorithms. We observed a large number of measurements where the variation of RTT values was so significant that it cannot be explain with measurement artifacts, as the full covering failed. We demonstrate our method in an example.

The result of our covering algorithm on measurement data to a specific destination is shown in Fig. 2: points represent RTT values from `ping` measurements, their variance is very large spanning from 350 ms to over 1 s. We performed 37 RTT measurements, equally distributed in time, between two `paris-traceroute` measurements. We assume that route change between `paris-traceroute` measurements is rare.

Our merging and path grouping algorithm found that most of the RTT measurements are concentrated near three "trails". Three different routes and an intermediate load balancer which periodically changes route priorities may cause this. The figure also shows that the whole span of data points cannot be explained by a high number of different routes and the rapid switching between them. Since determining which `ping` result belongs to which route is beyond our capability, we associate every uncovered RTT value to the closest, not larger covering interval, called as a parent interval. With this approach our aim is to minimize the demonstrable queuing delay. Note that the ratio of uncovered RTT values to all RTT measurements is 51%.

Before summarizing the results, we split days into 6-hour long periods in order to mitigate the effects of daily traffic patterns, and approach stationarity. To determine the extent of queuing delay we calculated the ratio for all uncovered RTT values to the smallest RTT value in their parent intervals. This ratio, denoted by R, is always larger than 1 due to our parent interval association method. If an RTT data point has an $R = 1.4$ then there is a supposed queuing delay of 40% of the minimal RTT on a given path, and this is after trying to assign all RTT points to a path in an adversary, against queuing delay, manner. In Fig. 3 we show the empirical distribution of R in 6-hour time windows separately. We marked $R = 1.2$ with a vertical line.

Against our expectations, the most uncovered RTT values (40%) with $R \geq 1.2$ found during the night period (GMT timezone). Nevertheless, uncovered RTT points in other time windows also suffer from high queuing delay: at least 20% of the packets have $R > 1.2$.

V. Conclusion

In this paper we showed qualitatively that queuing delay over the Internet can be indeed significant. As opposed to previous works in delay analysis our methodology was sort of indirect: we tried our best to explain the observed delay measurement values by assuming zero queuing delay and failed doing so.

The proposed method gives a conservative qualitative estimate on queuing delay experienced on random paths over the Internet. One might argue that the validity of our results is somehow limited due to the fact that we used only one vantage point. Indeed, our future work aims at leveraging the vast amount of measurement data of CAIDA (Center for Applied Internet Data Analysis). Another weakness of our work is the simplistic methods that we applied when merging IP nodes geographically close to each other, when grouping paths based on the similarity of their (merged) nodes, finally when greedily covering the RTT measurement points with our adversary goal in mind against queuing delay. Also as a future track, we will develop more sophisticated methods to clean the gathered measurement data and model the propagation delay in a more appropriate way.

References

[1] Benoit Donnet and Timur Friedman. Internet topology discovery: a survey. *Communications Surveys & Tutorials, IEEE*, 9(4): pp. 56–69, 2007.

[2] M Tozal and Kamil Sarac. Tracenet: an internet topology data collector. In *IMC*. ACM, 2010.

[3] Yihua He, Georgos Siganos, Michalis Faloutsos, and Srikanth Krishnamurthy. Lord of the links: A framework for discovering missing links in the internet topology. *IEEE/ACM ToN*, 17(2): pp. 391–404, 2009.

[4] Brice Augustin, Timur Friedman, and Renata Teixeira. Multipath tracing with paris traceroute. In *E2EMON'07*. IEEE, 2007.

[5] Brice Augustin, Timur Friedman, and Renata Teixeira. Measuring load-balanced paths in the internet. In *IMC*. ACM, 2007.

[6] Brice Augustin, Xavier Cuvellier, Benjamin Orgogozo, Fabien Viger, Timur Friedman, Matthieu Latapy, Clémence Magnien, and Renata Teixeira. Avoiding traceroute anomalies with paris traceroute. In *IMC*. ACM, 2006.

[7] Santiago Garcia-Jimenez, Eduardo Magana, Daniel Morató, and Mikel Izal. Pamplona-traceroute: topology discovery and alias resolution to build router level internet maps. In *GIIS*. IEEE, 2013.

[8] Yu Zhang, Ricardo Oliveira, Yangyang Wang, Shen Su, Baobao Zhang, Jun Bi, Hongli Zhang, and Lixia Zhang. A framework to quantify the pitfalls of using traceroute in AS-level topology measurement. *IEEE JSAC*, 29(9): pp. 1822–1836, 2011.

[9] Mehmet H Gunes and Kamil Sarac. Resolving IP aliases in building traceroute-based internet maps. *IEEE/ACM ToN*, 17(6): pp. 1738–1751, 2009.

[10] Thomas Bourgeau. Monitoring network topology dynamism of large-scale traceroute-based measurements. In *CNSM, 2011*. IEEE, 2011.

[11] Zhuoqing Morley Mao, Jennifer Rexford, Jia Wang, and Randy H Katz. Towards an accurate AS-level traceroute tool. In *SIGCOMM*. ACM, 2003.

[12] Benoit Donnet, Matthew Luckie, Pascal Mérindol, and Jean-Jacques Pansiot. Revealing MPLS tunnels obscured from traceroute. *ACM SIGCOMM*, 42(2), 2012.

[13] Vern Paxson. End-to-end routing behavior in the internet. In *SIGCOMM*. ACM, 1996.

[14] Yaron Schwartz, Yuval Shavitt, and Udi Weinsberg. A measurement study of the origins of end-to-end delay variations. In *PAM*, 2010.

[15] Pietro Marchetta, Alessio Botta, Ethan Katz-Bassett, and Antonio Pescapé. Dissecting round trip time on the slow path with a single packet. In *PAM*, 2014.

[16] S. Laki et al. Spotter: A model based active geolocation service. In *INFOCOM*. IEEE, 2011.

[17] Klaus Mochalski, Jörg Micheel, and Stephen Donnelly. Packet delay and loss at the auckland internet access path. In *PAM*, 2002.

[18] Ramaswamy Ramaswamy, Ning Weng, and Tilman Wolf. Characterizing network processing delay. In *GLOBECOM*. IEEE, 2004.

[19] CJ Bovy, HT Mertodimedjo, G Hooghiemstra, H Uijterwaal, and Piet Van Mieghem. Analysis of end-to-end delay measurements in internet. *PAM*, 2002.

[20] José-Alberto Hernández and Iain W Phillips. Weibull mixture model to characterise end-to-end internet delay at coarse time-scales. *IEE Proceedings Communications*, 153(2): pp. 295–304, 2006.

[21] Sándor Laki, Péter Mátray, Péter Hága, István Csabai, and Gábor Vattay. A detailed path-latency model for router geolocation. In *TridentCom*. IEEE, 2009.

[22] Péter Mátray, Péter Hága, Sándor Laki, Gábor Vattay, and István Csabai. On the spatial properties of internet routes. *Computer Networks*, 56(9): pp. 2237–2248, 2012.

[23] Roni Stern and Meir Kalech. Model-based diagnosis techniques for internet delay diagnosis with dynamic routing. *Applied Intelligence*, 41(1): pp. 167–183, 2014.

[24] Jean-Chrysotome Bolot. End-to-end packet delay and loss behavior in the internet. *ACM SIGCOMM*, 23(4), 1993.

[25] Michele Garetto and Don Towsley. Modeling, simulation and measurements of queuing delay under long-tail internet traffic. *ACM SIGMETRICS*, 31(1), 2003.

[26] Amgad Zeitoun, Chen-Nee Chuah, Supratik Bhattacharyya, and Christophe Diot. An as-level study of internet path delay characteristics. In *GLOBECOM*. IEEE, 2004.

[27] István Csabai, Péter Hága, Péter Mátray, Gábor Simon, József Stéger, and Gábor Vattay. Results of large-scale queueing delay tomography performed in the ETOMIC infrastructure. In *INFOCOM*, 2006.

[28] Baek-Young Choi, Sue Moon, Zhi-Li Zhang, Konstantina Papagiannaki, and Christophe Diot. Analysis of point-to-point packet delay in an operational network. *Computer Networks*, 51(13): pp. 3812–3827, 2007.

[29] J. Gettys et al. Bufferbloat: Dark buffers in the internet. *Queue*, 9(11): pp. 40, 2011.

[30] Chiara Chirichella and Dario Rossi. To the moon and back: are internet bufferbloat delays really that large? In *INFOCOM WKSHPS*. IEEE.

[31] T.B. Cardozo et al. Bufferbloat systematic analysis. In *ITS*. IEEE, 2014.

[32] P. Casoria et al. Distributed active measurement of internet queuing delays. In *PAM*, 2014.

[33] Modeling of one-way transit time in IP routers. Modeling of one-way transit time in IP routers. In *AICT-ICIW'06*. IEEE, 2006.

[34] Matthew J Zekauskas, Anatoly Karp, Benjamin Teitelbaum, Stanislav Shalunov, and Jeff W Boote. A one-way active measurement protocol (OWAMP). RFC 4656, September 2006.

[35] Pietro Marchetta, Valerio Persico, Antonio Pescapé, and Ethan Katz-Bassett. Don't trust traceroute (completely). pages 5–8, 2013.

[36] Chandra Chekuri and Sariel Har-Peled. Covering by translated intervals is NP-Hard. 2006.