Identifying Bottleneck Links Using Distributed End-to-end Available Bandwidth Measurements

Alok Shriram and **Jasleen Kaur**
Department of Computer Science
University of North Carolina at Chapel Hill
{alok, jasleen}@cs.unc.edu

1. Problem Statement

Why discover bottleneck links? The ability to identify overloaded network links has utility for both network operators as well as end-users. For network operators, access to an up-to-date snapshot of overloaded links can help quickly detect outages. Continual monitoring of overloaded links can also help verify the efficacy of load-balancing mechanisms used at the edges of a provider network. For end-users that rely on overlay networks for improving end-to-end transport performance, information about the location of bottleneck links can help select good alternate paths. Furthermore, a collection of bottleneck link snapshots with time can be used to answer fundamental questions such as: *Does congestion occur frequently? If so, when and on what kind of links does it occur?* In this work, we design and evaluate a tool for estimating bottleneck links—links with the least amount of available bandwidth (AB) [Jain02]—using distributed end-to-end AB measurements.

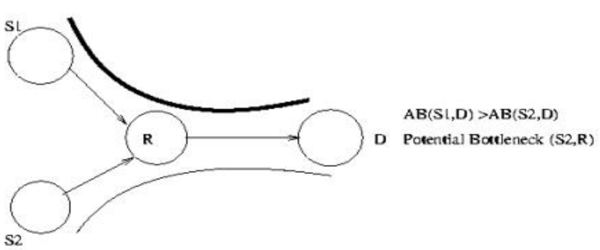
Why end-to-end measurements? Two factors prevent us from monitoring packet transmission at *individual* links in order to identify bottleneck links. First, access to routers and links is available only to network operators. Furthermore, end-users may not be allowed access to even any collected data due to protective business policies and privacy issues. Second, deploying a monitoring infrastructure—that can keep up with heavy traffic on high-speed links—at all links in a network is an expensive undertaking. Thus, even network operators may prefer to rely on a tool that uses edge-to-edge measurements to identify heavily-loaded links [Avaya].

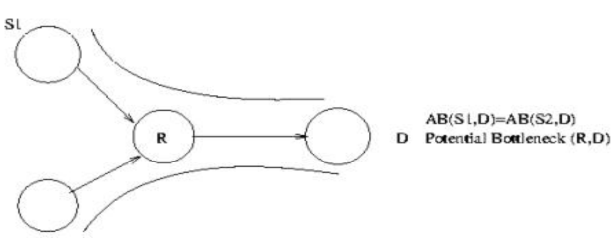
The idea of using end-to-end measurements to discover bottleneck links has also been explored in the design of Pipechar [Jin01], a tool that uses back-to-back packets to elicit ICMP responses from a given router. The AB of the corresponding link is then estimated using the dispersion in the response stream. Unfortunately, this technique suffers from two limitations that limit its ability to measure AB accurately. First, it has been shown in [Dovrolis01] that packet dispersion techniques do not measure AB, but a different throughput metric. Second, routers may discriminate between the handling of ICMP messages and ongoing data traffic; dispersion in the ICMP stream, therefore, may not be correlated to traffic load on a router. In fact, in the current Internet, this second limitation is likely to plague any effort that relies on end-to-end measurements on a *single* path to estimate bottleneck links.

2. Key Idea

In this work, we take the approach of using distributed measurements of end-to-end AB on *multiple* paths to identify the bottleneck links on each. Specifically, we envision that entities with access to multiple end-points will cooperate to exploit a key observation:

> *Simultaneous measurement of end-to-end AB on* ***multiple*** *paths, such that they share only a* ***subset*** *of their links with each other, increases the likelihood of identifying bottleneck links on each.*





The figure to the left illustrates two specific inference rules that exploit this observation.

- ***Rule 1:*** If AB on the end-to-end path between S1-D is greater than AB on path S2-D, the bottleneck links of S2-D must lie on the non-shared portion of its path.

- ***Rule 2:*** If the end-to-end AB on both paths is equal, it is highly likely that the bottleneck links of each path lie on the shared portion of their paths. Observe that this inference rule may lead to erroneous conclusions in case two or more links on the non-shared portions of each path have exactly the same AB. To assess the impact of this rule, we compare the impact of using just Rule 1, and using both Rules 1 and 2, on the estimation of bottleneck links.

3. Challenges

In practice, several challenges make it difficult to use the above ideas to identify bottleneck links:

Scheduling multiple end-to-end measurements: The schedule for measuring AB on multiple paths needs to satisfy two conflicting requirements: **(i)** probes should be run simultaneously to capture the same network conditions, and **(ii)** probes should be run separately on paths that share links in order to avoid interference. We show that achieving the optimal balance in this tradeoff is NP-hard and use heuristic algorithms to run probes on multiple paths close in time.

Design of a probing tool: The probing tool used to measure end-to-end AB on a single path needs to do so *accurately*, *quickly*, and in a manner *non-intrusive* to ongoing traffic. Our preliminary evaluation suggests that Pathload is currently the most accurate tool [Jain02]. Unfortunately, Pathload may take several tens of seconds to measure AB on a single path, which is inadequate for the purpose of running probes on multiple paths "close in time". We are currently working on improving the run-time of Pathload. It is important to note that our ideas can be used in conjunction with any probing tool.

Accurately estimating bottlenecks: Two factors can adversely affect the accuracy of estimating bottlenecks. First, if ε is the maximum inconsistency associated with the probing tool, then two paths with equal AB may yield readings that differ by up to ε. Second, due to a time-lag between measurements on different paths, two paths may yield different end-to-end AB, even if they have common bottleneck links. Bottleneck estimation must, therefore, account for these errors. We estimate reasonable bounds on these errors by conducting extensive evaluation of the probing tool and by measuring the amount by which AB changes with time on real paths.

4. PlanetLab Experiments & Results

We have instantiated our ideas in a distributed measurement and analysis tool (we currently use Pathload as the probing component). We present results from a deployment of the tool on 4 different topologies, each with 4 participating PlanetLab end-hosts (total of 48 paths studied). We run our tool 250 times over the duration of several hours. We label a link as a *potential bottleneck link* if comparison of all pairs of end-to-end paths to which it belongs indicates it to be a candidate bottleneck link. The following figures summarize our main findings.

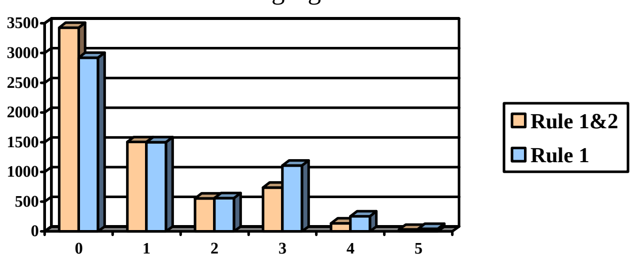


Fig 1: Number of potential bottleneck links per path

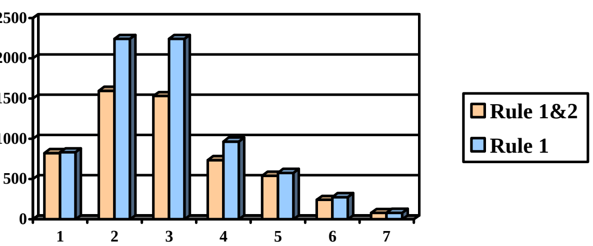


Fig 2: Distance of potential bottleneck links from source

Observe that there are several factors—including more unknown (per-link AB) than known quantities (end-to-end AB), measurement inaccuracy and long run-time of Pathload, etc—that could limit the ability of our tool to identify bottleneck links on all paths studied. Fig 1 plots the number of potential bottleneck links discovered for each path. We find that our current setup: **(i)** detects at least one bottleneck link for around 47% of the 12,000 path snapshots we study, and **(ii)** estimates three or less bottlenecks for 97% of the paths. We believe that a faster and more accurate probing tool will significantly improve these results. Use of the inference Rule 1 in isolation helps increase the number of paths for which we can estimate at least one bottleneck link, but it also limits the ability to narrow down such links to 1-2 candidates per path.

Fig 2 plots the distance, in hop-count, of all potential bottleneck links from the source of the corresponding paths. We find:

1. Around 71% (73% with just Rule 1) of potential bottleneck links occur within 3 hops from the source. This observation seems to support a popular opinion that bottlenecks occur near the edges of a path.
2. Only 14% (11% with just Rule 1) of potential bottleneck links occur within a single hop from the source, whereas more than 57% occur at 2-3 hops from the source. This suggests that links belonging to regional providers are more likely to be bottlenecks on a path than the last-mile LAN technology.

References

[Avaya] Avaya Networks, http://www.research.avayalabs.com/.

[Dovrolis01] C. Dovrolis and P. Ramanathan and D. Moore. What Do Packet Dispersion Techniques Measure?, Proceedings of IEEE INFOCOM, April 2001.

[Jain02] M. Jain and C. Dovrolis. End-to-end Available Bandwidth: Measurement Methodology, Dynamics, and Relation to TCP Throughput, Proceedings of ACM SIGCOMM, Pittsburgh, PA, August 2002.

[Jin01] G. Jin and G. Yang and B.R. Crowley and D.A. Agarwal. Network Characterization Service (NCS), Proceedings of the 10th IEEE Symposium on High Performance Distributed Computing, August 2001.