NORTHWESTERN
UNIVERSITY

Electrical Engineering and Computer Science Department

Technical Report
NWU-EECS-07-12
October 2, 2007

What Lies Beneath: Understanding Internet Congestion

Leiwen Deng, Aleksandar Kuzmanovic, and Bruce Davie

Abstract

Developing measurement tools that can concurrently monitor congested Internet links at a large scale would significantly help us understand how the Internet operates. While congestion at the Internet edge typically arises due to bottlenecks existent at a connection's last mile, congestion in the core could be more complex. This is because it may depend upon internal network policies and hence can reveal systematic problems such as routing pathologies, poorly-engineered network policies, or non-cooperative inter-AS relationships. Therefore, enabling the tools to provide deeper insights about congestion in the core is certainly beneficial.

In this paper, we present the design and implementation of a large-scale triggered monitoring system that focuses on monitoring a subset of Internet core links that exhibit relatively strong and persistent congestion, *i.e.*, hot spots. The system exploits triggered mechanisms to address its scalability; moreover, it automates selection of good vantage points to handle the common measurement experience that the much more congested Internet edges could often overshadow the observation for congestion in the core. Using the system, we characterize the properties of concurrently monitored hot spots. Contrary to common belief, we find that strong time-invariant hot spots reside in the Internet core — both within and between large backbone networks. Moreover, we find that congestion events at these hot spots can be highly correlated and such correlated congestion events can span across up to three neighboring ASes. We provide a root-cause analysis to explain this phenomenon and discuss implications of our findings.

This work is supported by a Cisco Collaborative Research grant.

Keywords: network congestion, Internet measurement, triggered monitoring, congestion correlation

What Lies Beneath: Understanding Internet Congestion

Leiwen Deng, Aleksandar Kuzmanovic
Northwestern University
{karldeng,akuzma}@cs.northwestern.edu

Bruce Davie
Cisco Systems
bdavie@cisco.com

ABSTRACT

Developing measurement tools that can concurrently monitor congested Internet links at a large scale would significantly help us understand how the Internet operates. While congestion at the Internet edge typically arises due to bottlenecks existent at a connection's last mile, congestion in the core could be more complex. This is because it may depend upon internal network policies and hence can reveal systematic problems such as routing pathologies, poorly-engineered network policies, or non-cooperative inter-AS relationships. Therefore, enabling the tools to provide deeper insights about congestion in the core is certainly beneficial.

In this paper, we present the design and implementation of a large-scale triggered monitoring system that focuses on monitoring a subset of Internet core links that exhibit relatively strong and persistent congestion, *i.e.,* hot spots. The system exploits triggered mechanisms to address its scalability; moreover, it automates selection of good vantage points to handle the common measurement experience that the much more congested Internet edges could often overshadow the observation for congestion in the core. Using the system, we characterize the properties of concurrently monitored hot spots. Contrary to common belief, we find that strong time-invariant hot spots reside in the Internet core — both within and between large backbone networks. Moreover, we find that congestion events at these hot spots can be highly correlated and such correlated congestion events can span across up to three neighboring ASes. We provide a root-cause analysis to explain this phenomenon and discuss implications of our findings.

1. INTRODUCTION

The common wisdom is that very little to no congestion occurs in the Internet core (*e.g.*, Tier-1 or -2 providers). Given that ISPs are aggressively overprovisioning the capacity of their pipes, and that end-to-end data transfers are typically constrained at Internet edges, the "lightly-congested core" hypothesis makes a lot of sense. Indeed, measurements conducted within Tier-1 providers such as *Sprint* report almost no packet losses [26]; likewise, ISPs like *Verio* advertise SLAs that guarantee negligible packet-loss rates [8]. As a result, researchers are focusing on characterizing congestion at access networks such as DSL and cable [23].

Congestion in the Internet core — why do we care? While numerous other measurement studies do confirm that the network edge is more congested than the core [10], understanding the properties of congestion events residing in the Internet core is meaningful for (at least) the following two reasons.

First, despite negligible packet losses in the core, *queuing delay*, which appears whenever the arrival rate at a router is larger than its service rate, may be non-negligible. Variable queuing delay leads to jitter, which can hurt the performance of delay-based congestion control algorithms (*e.g.*, [14, 17]), or real-time applications such as VoIP. And whereas it is difficult to route around congestion in access-network congestion scenarios [23] unless a client is multihomed [12], congested links in the core could be effectively avoided in many cases [11, 13]. Hence, identifying such congested locations, and characterizing their properties in space (how "big" they are) and time (the time-scales at which they occur and repeat) is valuable for latency-sensitive applications such as VoIP [7].

Second, the Internet is a complex system composed of thousands of independently administered ASes. Events happening at one point in the network can propagate over inter-AS borders and have repercussions on other parts of the network (*e.g.*, [35]). Consequently, measurements from each independent AS (*e.g.*, [26]) are inherently limited: even when a congestion location is identified, establishing potential dependencies among events happening at different ASes, or revealing underlying mechanisms responsible for propagating such events, is simply infeasible. To achieve such goals, it is essential to have global views, *i.e.*, to *concurrently* monitor the Internet congestion locations at a *large scale* across the Internet.

Contributions. This paper makes two primary contributions. First, we present the design of a large-scale *triggered* monitoring system, the goal of which is to quantify, locate, track, correlate, and analyze congestion events happening in the Internet core. The system

focuses on a subset of core links that exhibit relatively strong and persistent congestion, *i.e.*, *hot spots*. Second, we use a PlanetLab [5] implementation of our system to provide, to the best of our knowledge, the first-of-its-kind measurement study that characterizes the properties of concurrently monitored hot spots across the world.

Methodology. At a high level, our methodology is as follows. Using about 200 PlanetLab vantage points, we initially "jumpstart" measurements by collecting underlying IP-level topology. Once sufficient topology information is revealed, we start light end-to-end probing from vantage points. Whenever a path experiences excessive *queuing delay* over longer time scales, we accurately locate a congested link and designate it a hot spot. In addition, we trigger large-scale *coordinated measurements* to explore the entire area around that hot spot, both topologically close to and distant from it. Our system covers close to 30,000 Internet core links using over 35,000 distinct end-to-end paths. It is capable of monitoring up to 8,000 links *concurrently*, 350 of which could be inspected in depth using a tool recently proposed in [22].

Designing a system capable of performing coordinated measurements at such a large scale is itself a challenging systems engineering problem. Some of the questions we must address are as follows. How to effectively collect the underlying topology and track routing changes? How to handle clock skew, routing alterations, and other anomalies? How to effectively balance the measurement load across vantage points? How to minimize the amount of measurement resources while still covering a large area? How to select vantage points that can achieve high *measurement quality* with high probability? In this paper, we provide a "from scratch" system design and answer all these questions.

Applications. Our system has important implications for emerging protocols and systems, and they open avenues for future research. For instance, given the very light overhead on each node in our system (3.9 KBps on average), it could be easily ported into emerging VoIP systems to help avoid hot spots in the network. Further, our monitoring system should be viewed as a rudiment of a global Internet "debugging" system that we plan to develop. Such a system would not only track "anomalies" (*i.e.*, hot spots or outages), but would also automate a root-cause analysis (*e.g.*, automatically detect if a new policy implemented by an AS causes "trouble" to its neighbors). Finally, others will be able not just to leverage our data sets, but to actively use our system to gather their own data, performing independent analysis.

Findings. Our key findings are as follows. (*i*) Congestion events on some core links can be highly correlated. Such correlation can span across up to three neighboring ASes. (*ii*) There are a small number of hot spots between or within large backbone networks exhibiting highly intensive time-independent congestion. (*iii*) Both phenomena bear close relationships with the AS-level traffic aggregation effect, *i.e.*, the effect when upstream traffic converges to a *thin* aggregation point relative to its upstream traffic volume.

Implications. The implications of our findings are the following. First, the common wisdom is that the diversity of traffic and links makes large and long-lasting spatial link congestion dependence unlikely in real networks such as the Internet [19]. In this paper, we show that this is not the case: not only that correlation between congestion events could be excessive, but it can cover a large area. Still, most (if not all) network tomography models assume link congestion independence, *e.g.*, [18,19,24,30,34,40]. This is because deriving these models without such an assumption is hard or often infeasible. Our research suggests that each such model should be carefully evaluated to understand how link-level correlations affect its accuracy. Finally, our results about the typical correlation "spreading" provide guidelines for overlay re-routing systems. To avoid an entire hot spot (not just a part of it), it makes sense to choose overlay paths that are at least 3 ASes disjoint from the one experiencing congestion, if such paths are available.

2. BACKGROUND

In the recent work, the authors of [22] proposed a lightweight measurement tool, *Pong*, capable of accurately locating and monitoring Internet links that exhibit persistent congestion, *i.e.*, excessive queuing delays, over longer time scales. In this paper, we use Pong as the main building block of the large-scale congestion monitoring system. Below, we summarize the main features of Pong.

Pong exploits a novel variant of network tomography approach [18] to achieve: (*i*) low measurement overhead (per-path overhead is only 4.4 KBps), (*ii*) significantly improved resolution for congestion on *non-access links* relative to other delay-based active measurement tools, and (*iii*) the capability to quantify its own measurement quality, which serves as the basis for us to optimize vantage point selection.

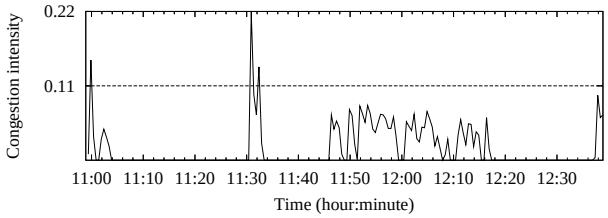
High link congestion-measurement resolution. Pong deploys its measurement on the two endpoints of a path. By coordinating active probes (including end-to-end probes and probes to intermediate routers) sent from *both endpoints*, Pong significantly improves the ability to *detect and locate* congestion on non-access links of the measuring path relative to the state of the art delay-based congestion measurement approaches [22].

Link measurability score. Pong makes a best effort to minimize the effect of measurement errors caused by anomalies such as clock skew at end hosts, router

alteration of the path, ICMP queuing at routers, *etc.* In addition, its unique network tomography approach allows it to be able to quantify such measurement errors rather than just detect them. Pong quantifies its measurement quality using the *link measurability score* (LMS), a measure associated with each of the links on an end-to-end path. It defines how well we can measure a specific link from a specific measuring path. The LMS is the key for us to optimize the measuring path selection in our large-scale congestion monitoring system, as we explain later in Section 4.

Congestion event and congestion intensity. Congestion has several manifestations. The most severe one induces packet losses. By contrast, Pong detects congestion in its early stage, as indicated by increased queuing delays at links. From a microscopic view, a continuous queue building-up and draining period typically lasts from *several ms to hundreds of ms* [39]. On time scales of *seconds to minutes*, queue building-up can repeatedly happen, as shown in Figure 1.

Pong can report congestion online on-demand at 30-second intervals when it detects congestion. Each report is associated with a link for a 30-second-long epoch and we call each report a *congestion event*. Pong annotates each congestion event with two measures: *congestion intensity* and the corresponding LMS. The congestion intensity quantifies how frequently the link is congested during the 30-second-long time period. It is an important measure that we exploit in our system.



The Y axis shows the congestion intensity. It is computed every 30 seconds.

Figure 1: Congestion events on a link monitored by Pong

3. A TRIGGERED MONITORING SYSTEM

In this section, we present T-Pong, a large-scale congestion monitoring system. To accurately measure points of congestion in the Internet core and to effectively and scalably allocate measurement resources, T-Pong deploys an on-demand triggered monitoring approach.

3.1 Design Goals

We first summarize our design goals:

Good coverage. The system should have a good coverage of links. By "good" we mean: (*i*) High coverage, low overhead. The system should be able to monitor a large percent of network links concurrently while keeping its traffic overhead low. (*ii*) Balanced coverage. Measuring paths should cover different congested links in a balanced way. The system should reduce the chance that the number of measuring paths that cover two different links differ significantly; otherwise, it could result in a non-negligible measurement bias.

High measurement quality. The system should allocate paths that provide the best measurement qualities when measuring congested links.

Online processing. To support triggering mechanisms, the system should be capable of processing raw data online. It should provide: (*i*) Fast algorithms. To operate in real time, algorithms must be fast enough to keep up with the data input rate. (*ii*) Extensible triggering logic. Triggering logic may be frequently adjusted for research purposes, the system therefore should provide a convenient interface to update triggering logic.

Integrity and robustness. To be capable of running long-term monitoring tasks, the system must guarantee the integrity of critical data and should be able to quickly recover from errors.

3.2 System Structure

Figure 2 illustrates the structure of T-Pong. It consists of one *control node* and hundreds of *measuring nodes*. The control node performs centralized control of the entire system. It collects measurement reports from measuring nodes and adjusts monitoring deployment based on a set of vantage points selection algorithms. The measuring nodes monitor network congestion from vantage points scattered all around the world. Each of them runs two measurement processes: TMon and Pong. TMon performs end-to-end congestion monitoring while Pong performs fine-grained link congestion measurement upon triggering.

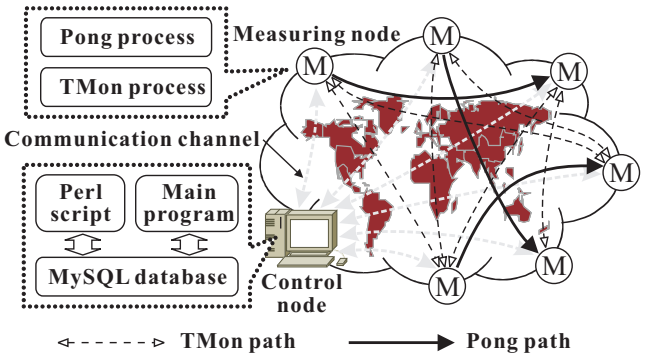


Figure 2: T-Pong system structure

Control Node. In our implementation, the control node is Pentium 4 PC that runs Fedora core 5 Linux. Its application software includes a main program written in C++, a MySQL database that stores measurement

Paths used	Path selection algorithm	Probing method	Probing rate	Objective
All paths	No selection, full mesh	Low-rate probing (Section 3.6.2)	Once every 5 minutes	Track topology and path reachability
TMon paths – a subset of all paths	Greedy TMon path selection (Section 3.5.1)	Fast-rate probing (Section 3.6.3)	5 probes/sec	Monitor end-to-end congestion
Pong paths – a subset of TMon paths upon triggering	Priority-based Pong path allocation (Section 3.5.2)	Pong's coordinated probing (Section 2)	10 probes/sec for e2e probing, 2 probes/sec for router-targeted probing	Locate and monitor link-level congestion

Table 1: A summary of T-Pong's major measurement techniques

data, and an optional Perl script that allows customized triggering logic. Fault tolerance of the control node is a concern for the implementation, and we plan to address it in the future using a failover mechanism to a standby control node.

Measuring Nodes. The measuring nodes are PlanetLab hosts running the TMon and Pong programs. Both programs are written in C++. TMon keeps track of path routes and monitors end-to-end congestion events. Pong measures link-level congestion of hot spots upon triggering.

3.3 Measurement Procedure Overview

In this section, we introduce the major steps of T-Pong's measurement procedure. We summarize techniques associated with these steps in Table 1. (*i*) During the system bootstrap phase, the control node collects route information from all measuring nodes and updates its topology database accordingly (Section 3.4). The topology database stores the whole set of routes. It does not need to resolve extensive topology information such as unique nodes and unique links via IP alias approaches. In our measurement and analysis in Section 4, we do not require such detailed topology estimates.

After that the system keeps track of route changes and path reachability, and incrementally updates the topology database. (*ii*) Once the topology database is available, the control node starts a greedy algorithm (Section 3.5.1) to select a subset of paths (called *TMon paths*) that can cover a relatively large percent of links. The TMon processes on measuring nodes associated with these TMon paths then start *fast-rate probing*. The fast-rate probing monitors end-to-end congestion for each TMon path. (*iii*) Once end-to-end congestion on a TMon path is detected, the system uses its default triggering logic (a priority-based algorithm, Section 3.5.2) to decide whether it should start Pong on that path. It might also decide to start Pong on other paths based on customized triggering logic (Section 3.5.3). Measuring nodes associated with these paths then use Pong (the paths are therefore called *Pong paths*) to locate and monitor congestion on these paths at the granularity of a single link.

The system keeps track of congestion events as well as anomalies such as route changes, clock skews and path failures. It responds to such events by (*i*) processing congestion events and analyzing measurement accuracy online, (*ii*) updating the topology database, (*iii*) suppressing measurement on paths affected by anomalies, and (*iv*) rearranging the deployment of TMon and Pong paths. In this way, the system gradually transits into a stable state, in which it collects measurement reports, processes them, applies triggering logic, and adjusts measurement deployment.

3.4 Initializing Topology Database

When the system initially starts up, all measuring nodes are sending topology messages to the control node (after which, they send incremental reports upon route changes). The control node therefore experiences a burst of topology messages during bootstrap.

As we will discuss in Section 3.5.1, updating topology information to a database is not a cheap operation, since we have to update additional parameters used for selecting TMon paths, which incurs costly database queries. Database updates therefore would be unable to keep up with arrival speed of topology messages. To solve this, we only perform simple pre-processing operations for each topology message in real time, while buffering all database updates. In addition, we attach a *version* number and an 8-byte *path digest* value (a fingerprint of the route) in each topology message. This helps the control node to easily filter outdated or duplicate updates thereby avoiding unnecessary database updates.

3.5 Path Selection Algorithms

3.5.1 Greedy TMon Path Selection

The control node uses a greedy algorithm to select TMon paths. The goal of this algorithm is to select only a small fraction of paths while covering a relatively large percentage of links[1].

Algorithm. This selection algorithm runs online in an iterative way. During each iteration, it selects a new TMon path that covers the most *remaining* links. A remaining link is a link that has currently been covered

[1]In [15], the authors show that we can see a large fraction of the Internet (in particular, the "switching core") from its edges when properly deploying a relatively small scale end-to-end measurement infrastructure.

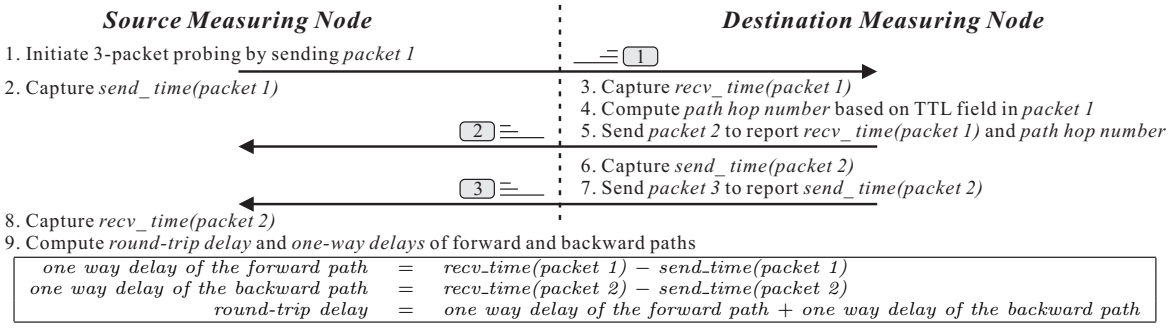


Figure 3: Three-packet probing procedure

by *less than two* TMon paths.

In addition, the new TMon path must satisfy the following conditions: (*i*) Each of the two endpoints of this path must not participate in more than 20 TMon paths. Restricting to 20 TMon paths is a good tradeoff between the link coverage and the measurement overhead as we show in Section 4.2.1. (*ii*) The path must be *active*. A path becomes inactive when it is unreachable or experiences considerable route alterations. (*iii*) The path must not be *suppressed*. A path is suppressed from being a TMon path for a short period of time (15 minutes) when anomalies (*e.g.*, clock skews) are detected. A path is suppressed for a long period of time (6 hours) when it shows poor measurement quality while running Pong.

The iterative selection stops when there are no links currently satisfying the above conditions. Then the algorithm transits into a sleeping state for 5 minutes, after which it tries the iterative selection again.

Making it online-capable. Enabling this algorithm to run online is a non-trivial task since each iteration involves complex logic. If implemented improperly, the algorithm could incur formidable database queries. To address this, we pre-compute per-link, per-path, and per-node parameters used in this algorithm during other operations. For example, when we update topology information of a path, we additionally compute the number of paths and the number of TMon paths that cover each link of the path. Although pre-computing adds some complexity to other operations, our implementation shows it to be a good tradeoff.

3.5.2 Priority-based Pong Path Allocation

The control node allocates Pong paths using a priority-based algorithm. In order to minimize the interference among Pong paths, we restrict each measuring node to participating in at most one Pong path. Given this constraint, the priority-based algorithm selects paths that provide the best measurement quality and link coverage when measuring points of congestion. In addition, it effectively reduces the chance that multiple Pong paths measure the same congested link, thereby achieving a balanced measurement coverage.

To achieve these goals, the systems incorporates the maximum *link measurability score* (*LMS*, Section 2) over all non-edge links on a path, a measure that indicates the best measurement quality a path can achieve. In addition, because multiple paths can measure the same core links concurrently, the system excludes LMSes of those links currently covered by other Pong paths with a higher LMS, as we explain below.

Algorithm. To allocate Pong paths, we follow these rules in sequence: (*i*) Give priority to a path that shows higher maximum LMS over non-edge links on a path, excluding the LMSes of those links that are currently covered by other Pong paths with a higher LMS. We use the LMS history of the recent 24 hours for a candidate path on which we try to start Pong, and we use the LMS history of the recent 1 hour for a currently running Pong path that conflicts with the candidate path. (*ii*) Give priority to a path that shows higher *congestion intensity* (Section 2) unless it is filtered by the triggering logic. (*iii*) Do not override a path that has just started Pong measurement for a short period of time ($<$ 30 minutes). (*iv*) Suppress paths showing small maximum LMS over non-edge links of a path (using LMS history of the recent 24 hours). This priority-based algorithm allows measurement resources to be effectively allocated to track hot spots.

3.5.3 Programmable Triggering Logic

The system allows us to customize our triggering strategies on the fly through an add-on Perl script. The interface between the main program and the Perl script is the database. The main program records each congestion event to the database. The Perl script, which runs in a loop, periodically checks new events, selects concerned ones, and performs its triggering logic. Based on that, it issues measurement commands and enqueues them to the database. The main program checks the database for enqueued commands every minute, and adjusts measurement deployment accordingly.

This feature provides great flexibility for our conges-

tion measurement experiments with the T-Pong system. We can easily change our focused network area and concerned network-wide congestion pattern, and easily test and verify our research hypotheses.

3.6 TMon's Active Probing

3.6.1 3-packet Probing

TMon keeps track of path routes using low-rate probing and measures end-to-end congestion on TMon paths via fast-rate probing. Both low-rate and fast-rate probings exploit *3-packet probing*, which includes three probing packets. The 3-packet probing measures: (*i*) path reachability, (*ii*) round-trip delay, (*iii*) one-way delays of both forward and backward paths, and (*iv*) the number of hops on the forward path. Figure 3 illustrates its procedure. Note that the measured one-way delays include the clock offset between the two measuring nodes, but it is canceled when we compute round-trip delay or queuing delays.

3.6.2 Low-rate Probing

A measuring node sends low-rate probes to all other measuring nodes every five minutes. Each low-rate probe consists of a *3-packet probing* and an optional traceroute. The 3-packet probing explores path reachability and the total hop number of a path. For each reachable path, the traceroute is performed. Since we know the total path hop number in advance, we develop an efficient version of traceroute, which sends TTL limited probes to all hops along the path concurrently.

Low-rate probes to different nodes are paced evenly during each 5-minute period to smooth the traffic and to avoid interference between traceroutes. For paths which are temporarily unreachable, we back off probing rate up to 1/16 of the normal rate (*i.e.*, one probe every 80 minutes) to reduce unnecessary overhead.

3.6.3 Fast-rate Probing

A node sends fast-rate probes (5 times per second) on each TMon path sourced from it. Each fast-rate probing is a 3-packet probing. To measure congestion, the source node keeps track of round-trip and one-way delays. Based on that, it computes the corresponding queuing delays and infers congestion. Fast-rate probing backs off up to 1/64 of its normal rate when a path becomes unreachable.

4. EVALUATION AND MEASUREMENTS

4.1 Experimental Setup

We conducted six PlanetLab based experiments using our congestion monitoring system in a two-month period. Each experiment lasts for $4 \sim 7$ days, and we therefore collected a set of one month-long measurement data altogether. We intentionally collected data in different time intervals in order to verify that our results are time-invariant. Indeed, *all our findings hold for all the distinct traces we took.* We use a total of 191 PlanetLab nodes in our experiments. Table 2 shows a summary of these nodes classified by continents.

Continent	Number of nodes
North America, South America	110
Europe	63
Asia, Australia	18

Table 2: PlanetLab nodes used in our experiments

4.2 Evaluation

4.2.1 Coverage and Overhead

Here, we provide statistics about the network coverage, *i.e.*, how many end-to-end paths and internal links our system has covered. Also, we quantify the measurement overhead imposed by our system.

Total paths and links. We observe about 36,000 paths (N^2, where N=191 PlanetLab nodes), which expose about 12,100 distinct links at a time. In addition, due to routing changes, we are able to observe about 29,000 distinct links totally.

Coverage and overhead of TMon paths. Our measurement log shows that there are $1{,}500 \sim 2{,}000$ paths running TMon concurrently. These paths cover about $7{,}600 \sim 8{,}000$ distinct links concurrently. Comparing with the number of total paths and links, it shows that we manage to use 4.9% of total paths to cover 65% of total links. The discrepancy between the percent of paths and links is not a surprise, since this effect has been observed previously (*e.g.*, [15, 20]).

Each PlanetLab node that we use participates in 9.2 TMon paths on average, which corresponds to an average per-node traffic overhead of 3.9 KBps, while the peak per-node traffic overhead is 8.4 KBps. This is because we restrict each node to participate in at most 20 TMon paths (Section 3.5.1). Our experiments show that we can still improve the coverage of links considerably if we relax this restriction to some extent, *e.g.*, to allow each node to participate in up to 40 TMon paths. However, this would come at the cost of a higher overhead. The overhead of TMon paths constitutes the major component of the total overhead. By contrast, the overhead of Pong paths and of the communication channels between the control node and all measuring nodes only accounts for a very small fraction. Therefore, we can roughly treat the overhead of TMon paths as the total overhead of the system.

Coverage of Pong paths. Our measurement log shows that there are $25 \sim 30$ paths running Pong concurrently. This means that there are $50 \sim 60$ PlanetLab nodes run-

ning Pong at a moment. These paths cover 250 $\sim$ 350 distinct links concurrently.

4.2.2 Measurement Quality

One important principle of our system design is to select measuring paths (or vantage points) that provide the best measurement quality on link congestion. Here we evaluate the measurement quality that we achieve in our experiments. We use the link measurability score (LMS, Section 2) annotated with each link congestion event as the evaluation measure. According to [22], the value of LMS is a float number between 0 and 6. And it has the following major levels:

"LMS=0" means undesirable path conditions for the congestion measurement using Pong. We have observed end-to-end congestion on the path at a moment, but Pong is unable to accurately locate this congestion. The conclusion that the reported link is congested may not be reliable. "LMS=1" implies that a desirable path condition for Pong has been satisfied. This is the lowest level that a link congestion event report is considered acceptable. "LMS=2" implies that a desirable path condition has been satisfied and indicates moderate measurement quality. "LMS=3" implies desirable path conditions for Pong and indicates good measurement quality. "LMS$\geq$4" indicates very good measurement quality.

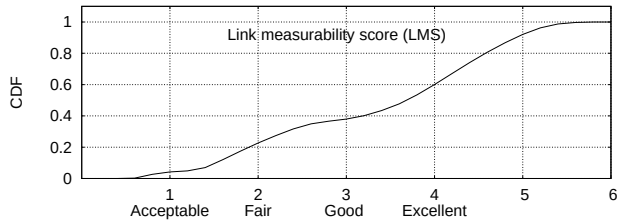


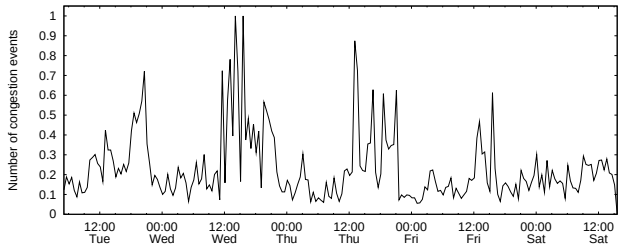
Figure 4: Link measurability score distribution

We use measurement data of a one week long experiment to exemplify our result. During this week, we collected a total of 54,991 link congestion events. 35,728 (65%) of them have a non-zero LMS. Figure 4 shows the cumulative distribution function (CDF) of LMS for these 35,728 events. As we can see, (*i*) 96% of these congestion events have an acceptable measurement quality (LMS$\geq$1), (*ii*) 77% of them have a better-than-fair measurement quality (LMS$\geq$2), (*iii*) 63% of them have a good measurement quality (LMS$\geq$3), and (*iv*) 40% of them have an excellent measurement quality (LMS$\geq$4).

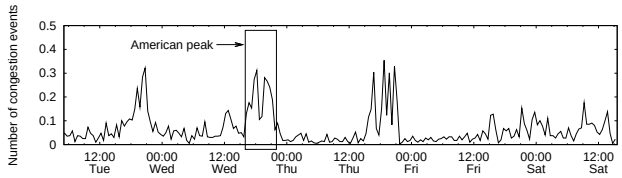
4.3 Link-level vs. End-to-end Congestion

In this section, we study the properties of end-to-end congestion events (observed by our TMon system), and link-level congestion events (observed by the triggered Pong-based measurements) and emphasize fundamental differences between the two. Throughout the section, we leverage the fact that traffic loads exhibit a well-known time-of-day pattern. As a result, network congestion also exhibits a similar pattern, which our system successfully captures.

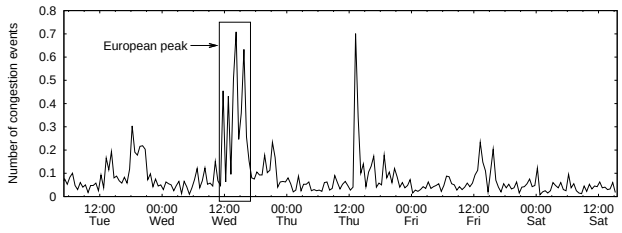
The Y axis in each of the following sub-figures is normalized using the peak value of the curve in sub-figure (a).



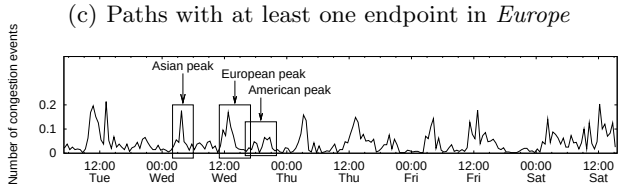
(a) All paths



(b) Paths with both endpoints in *South, North America*



(c) Paths with at least one endpoint in *Europe*



(d) Paths with at least one endpoint in *Asia*

Figure 5: Time of day effect of end-to-end congestion observation

4.3.1 End-to-end Congestion Properties

Here, we evaluate properties of the end-to-end congestion captured by TMon paths. These results provide a basis for understanding properties of the link-level congestion that we describe in the next section. We use a 4.5-day-long measurement data set to exemplify our result of end-to-end congestion events on all paths. To quantify the congestion scale, in Figure 5(a) we show the (normalized) number of congestion events computed at 30-minute-long time intervals.

Figure 5(a) shows a typical time-of-day effect in end-

to-end congestion. The measures show consistent time-of-day pattern with a daily peak at 12:00∼21:00 GMT. However, this is a mixed time-of-day pattern for traffic generated by users all over the world. To get a clearer picture, we decompose it into different user time zones and compare each component with its local time.

To this end, we classify paths according to the continent that their endpoints reside in. Since each path has two endpoints that can reside at different continents, we make an approximation by dividing paths into the following three groups: (*i*) paths with both endpoints in North or South America, (*ii*) paths with at least one endpoint in Europe, and (*iii*) paths with at least one endpoint in Asia. Figures 5(b)-(d) plot congestion events captured by the three path groups respectively.

From the three figures we can resolve three different daily peaks which can be well mapped to local *diurnal* time of the three continents. We denote them by "American peak", "European peak", and "Asian peak" respectively as summarized in the following table.

Peak	GMT time	Local time
American peak	16:00–22:00	10:00–16:00 (GMT-6)
European peak	11:00–17:00	12:00–18:00 (GMT+1)
Asian peak	02:00–06:00	10:00–14:00 (GMT+8)

Now we explain the above observations. First, it is well-known that links at network edges are more congested than that in the core [10]. As a result, the main component of end-to-end congestion is the congestion at edges, *i.e.*, congestion close to measuring endpoints. In Figure 5(b), since both endpoints of a measuring path locate in the American continent, the figure shows a clear "American peak" in the sense that it matches the local diurnal time.

Figures 5(c) and 5(d) expose the same causality, which captures the "European peak" and "Asian peak," respectively. However, since we only apply the geographic constraint for one endpoint, these two figures capture more than one daily peak. This is most apparent in figure 5(d). In addition to the "Asian peak," this figure captures the "European peak" and the "American peak" as well. This is because only one endpoint of each path is required to be in Asia. The other endpoint could reside in America or Europe.

4.3.2 Link-level Monitoring

Here, we evaluate the properties of link-level congestion monitored via Pong by comparing them with properties of the end-to-end congestion monitored via TMon. We emphasize the following two fundamental differences between the two approaches and reveal advantages of the link-level congestion monitoring.

First, link-level congestion monitoring allows us to focus on congestion events at a specific location in the Internet *core*, which could show different properties than the congestion at edges. By contrast, the end-to-end congestion monitoring inevitably multiplexes congestion from all locations on a path and such congestion is usually dominated by the congestion at edges.

Second, when monitoring a wide network area, our link-level congestion monitoring approach makes it possible to cover the area in a balanced way (Section 3.5.2). By contrast, the location-unaware end-to-end monitoring would inevitably lead to big discrepancy on the numbers of measuring paths that cover different links. Such unbalanced coverage will lead to biased measurement results because congestion on some links could be repeatedly counted for much more times than congestion on other links.

Figure 6(a) shows results of the link-level congestion monitored via Pong. Since we know the congestion location, we can filter out congestion events at edge links. As a result, the figure actually shows aggregate link-level congestion on core links. Similarly to the way we analyze end-to-end congestion, we use the normalized number of congestion events as the representative measure and the measure is computed at 30-minute-long intervals.

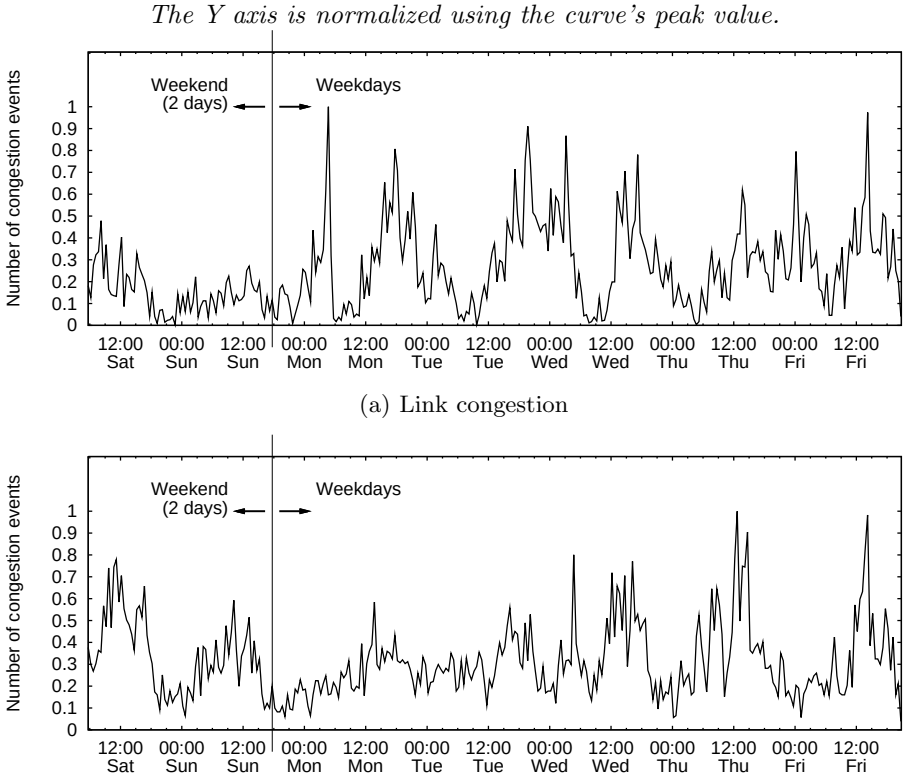
Figure 6(b) shows results of the end-to-end congestion monitored via TMon during the same week. The two figures reveal important differences in link-level and end-to-end congestion dynamics. In particular, Figure 6(a) shows a clearer time-of-day pattern — peaks are higher and valleys are deeper. Daily peaks in Figure 6(a) do not necessarily correspond to daily peaks in Figure 6(b), but it still corresponds to some of its minor peaks. This is because congestion on core links will definitely lead to end-to-end congestion, but it does not dominantly affect the end-to-end congestion. For the valleys, the two figures show an even larger mismatch. For example, the valleys in Figure 6(a) could correspond to epochs in Figure 6(b) at which there is high congestion. This happens because the end-to-end congestion is dominated by congestion at edges.

In addition to the daily differences, the two figures also show a significant weekly difference. As summarized in Table 3, Figure 6(a) shows much fewer congestion events per hour during the weekend than the weekdays. On the contrary, Figure 6(b) shows slightly more congestion events per hour during the weekend than the weekdays.

	Figure 6(a)	Figure 6(b)
Weekend	10,060 (279/hour)	5,684 (158/hour)
Weekdays	44,931 (365/hour)	17,552 (143/hour)
Total	54,991 (346/hour)	23,236 (146/hour)

Table 3: Number of congestion events corresponding to data in Figure 6

Overall, we find that congestion in the core tends to behave very differently from end-to-end congestion; our



(a) Link congestion

(b) End-to-end congestion

Figure 6: Comparison between link congestion and end-to-end congestion observations

link-level monitoring approach therefore makes possible a much more accurate insight about the congestion in the core. In addition, the clearer time-of-day effect observed in Figure 6(a) is also result from the unbiased coverage achieved by our algorithm. In particular, the link coverage of Pong paths is optimized to be *uniform* because our triggering logic effectively reduces the chance that multiple Pong paths repeatedly measure the same congested link.

4.4 Link-level Congestion Correlation

In this section, we analyze the congestion correlation across core links and describe our basic findings. We present an in-depth analysis about underlying correlation causes in the next section.

To quantify the congestion correlation, we define *observed correlation*. It approximates the pairwise congestion correlation between two links. To compute it, we first search concurrent link congestion events during each 30-second time period and update a matrix that records the times each link pair being concurrently congested (we call this measure *overlap count*). Then, we divide the overlap count by the smaller total congestion event number of the two links in the pair. The quotient is the observed correlation. We do not use the classical statistical measure of correlation due to the difficulty to acquire the overlapped measurement period of two links in our trigger-based measurement context. This is because we only know the overlapped measurement period of paths and it is hard to resolve whether a specific path measurement period is ascribed to a specific link or not.

In addition, to prevent scenarios in which the smaller total congestion event number is too small, such that it might lead to an unreasonably high correlation, we require the overlap count to be at least 10; otherwise, we filter this link pair. We put a requirement on the overlap count instead of the smaller total congestion event number because we also want to filter the cases when two links do not share many concurrent measurement periods, hence an unreasonably low correlation. Indeed, because concurrent measurements might not always be available, the observed correlation may slightly *underestimate* congestion correlation between links.

To present the results, we use the data corresponding to Figure 6(a), which are data of link-level congestion on the core links in a one-week-long experiment. We plot the CDF of observed correlations for the weekend and weekdays separately as shown in Figure 7. The figure shows that (i) congestion correlation across core links

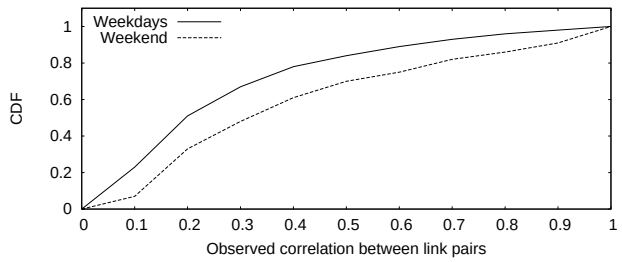


Figure 7: Distribution of observed correlation

is *not rare* and there are quite a number of repetitively congested links that bear relatively high correlations; (*ii*) the observed correlations are much higher during the weekend than during the weekdays.

The second phenomenon implies a relationship between the observed correlations and overall congestion scale on the core links. Recall that in Figure 6(a) we observe a much lower overall congestion scale on the core links during the weekend than during the weekdays. We infer that the reason we observe higher correlations during the weekend is that a pairwise correlation is much easier to observe when the overall congestion scale is low. We hypothesize that we observe a lower correlation during weekdays not because the same pairwise correlation does not exist, but because it is blurred as a result of the interference among multiple pairwise correlations when the overall congestion scale is high.

To understand how far a pair of correlated links can separate from each other, we select the top 25% (in terms of observed correlation) link pairs in the weekdays and weekend respectively. We approximately estimate the shortest distance between the two links in each pair based on our topology database. We represent the distance by two measures: *AS distance* and *hop distance*. The former quantifies the AS level distance, the latter quantifies the router level distance. We summarize the average and standard deviation of estimated distances in Table 4. In addition, our result shows that the correlated links can span across up to three neighboring ASes (*i.e.*, AS distance ≤ 3).

AS Distance				Hop Distance			
Weekdays		Weekend		Weekdays		Weekend	
Avg	Dev	Avg	Dev	Avg	Dev	Avg	Dev
0.82	0.76	0.85	0.77	3.1	1.9	3.7	1.9

Avg: Average distance; Dev: Standard deviation of distance.

Table 4: Distance between correlated links

4.5 Aggregation Effect Hypothesis

One important finding from our experiments is that the *traffic aggregation effect* tends to play an important role on congestion behavior at core links. Here, traffic aggregation means the situation that traffic from a number of upstream links converges at a downstream AS-level aggregation link. We make this hypothesis based on two phenomena that we observed: (*i*) The spatial distribution of correlated links tends to result from the aggregation effect (Section 4.5.1). (*ii*) There are some hot spots in the core exhibiting time-independent high congestion, which also tend to result from the aggregation effect (Section 4.5.2). A comparison between congestion locations related to these two phenomena and locations that the aggregation effect is most likely to happen fortifies our hypothesis (Section 4.5.3).

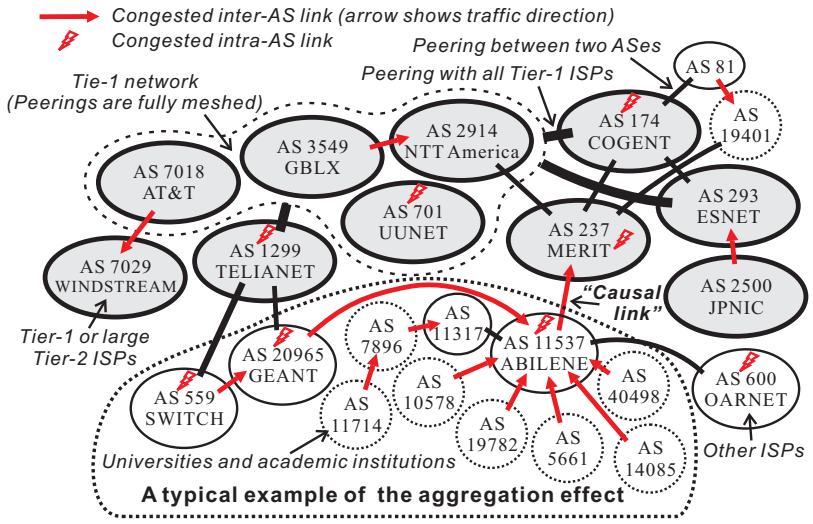
4.5.1 Spatial Distribution of Correlated Links

To perform an in-depth analysis about congestion correlation, we select top 20 links in terms that they are commonly correlated with the largest number of other links. We call them the "causal links." Note that these links are not necessarily the real cause, but they tend to be the most sensitive indicators for underlying causes; hence, we can *roughly treat them as the cause*.

We analyze the locations of each "causal link" and of the links highly correlated with it. To represent the location, we classify links into intra-AS links and inter-AS links. Classifying intra- and inter-AS links is a non-trivial task. To do this, we use a best-effort algorithm that is based on IP ownerships of the two nodes of each link. We find that the above links can reside at diversified locations, including (*i*) intra-AS links within large ISPs (including Tier-1, Tier-2 and other backbone networks); (*ii*) inter-AS links between large ISPs; and (*iii*) inter-AS links from a lower tier to a higher tier.

To describe our in-depth analysis results about correlated links, we use the top "causal link" as the example. It is an inter-AS link between AS 11537 (Abilene) and AS 237 (Merit). This link shows high correlation with 25 other links. Figure 8 illustrates AS-level locations for 23 of these links that we can resolve. The 23 links cover an area that consists of as much as 24 distinct ASes. All of them are no more than 3 ASes away from the "causal link". As we can see from Figure 8, most links reside *upstream* from the "causal link". Examples are: (*i*) an intra-AS link within AS 11537. (*ii*) Inter-AS links from seven universities that directly or indirectly access AS 11537. (*iii*) Intra-AS links within Tier-1, Tier-2 ISPs, and backbone networks that are close to AS 11537.

Analysis on other "causal links" shows a similar congestion spatial distribution pattern. Such distribution pattern fortifies our hypothesis that the congestion correlation could result from the aggregation effect: When upstream traffic converges to a relatively *thin* aggregation point, upstream traffic surges can cause congestion at the aggregation point, hence a high probability that congestion at the two places (the upstream network and



This figure shows locations of congested links correlated with an inter-AS link between AS 11537 and AS 237. The lower part shows a typical example of the aggregation effect.

Figure 8: In-depth analysis on spatial distribution of correlated links

the aggregation point) is correlated. Take Abilene for example. Its cross-country backbone is 10 Gbps, while the two aggregation points (one in Chicago, the other in Michigan) from Abilene to Merit tend to have less provisioned bandwidths, *e.g.*, the one in Chicago still uses the OC-12 (622 Mbps) connection [4]. In addition, Abilene aggregate network statistics in 2007 [1] shows that aggregate traffic from Abilene to Merit is usually about twice as much as that in the reverse direction. This matches the congestion direction that we observe in Figure 8.

4.5.2 Locations of Time-Independent Hot Spots

In addition to the spatial distribution of correlated links, we observe another phenomenon that fortifies our aggregation hypothesis. This phenomenon is that there are a number of hot spots exhibiting time-independent high congestion. By analyzing locations of such hot spots, we find a high likelihood that such hot spots could result from the aggregation effect.

To understand properties of such time-independent hot spots, we illustrate their effect on end-to-end congestion observation as shown in Figure 9. The figure plots dynamics of two measures: *number of congestion events* and *average per-event congestion intensity*. Both measures are computed at 30-minute-long time intervals and are normalized by their peak values. The figure shows that the number of congestion events exhibits a clear time-of-day pattern. On the contrary, the average congestion intensity exhibits a very different dynamics. It is largely independent on the time-of-day effect, and even shows its peaks at the valleys of the number of congested events curve, and vice versa.

Through investigation, we find the above phenomenon results from the following properties of time-independent hot spots: Although such hot spots exhibit the time-off-day effect in terms of the number of congestion events (fewer events when overall network congestion is low), their *per-event* congestion intensities always remain high. This is why we can often observe much higher average congestion intensity when the total number of congestion events is small.

We find that such time-independent hot spots are inter-AS links between large backbone networks all over the world *as well as intra-AS links* within these networks. The former ones are about 1.5 times as many as the latter ones. Most of these links are not intercontinental links as we initially hypothesized. Table 5 shows the top ten ASes that host the most of such time-independent hot spots. We list them in descending order of the maximum congestion intensity measured on the hot spots they host, *i.e.*, AS 174 shows the largest maximum congestion intensity. In the next section, we show that these locations are the places where the aggregation effect is most likely to happen.

4.5.3 AS-level Traffic Aggregation Effect

To analyze the aggregate effect hypothesis, we compare the top 20 correlated links (that are commonly correlated with the largest number of links) mentioned in Section 4.5.1 and locations of the time-independent hot spots mentioned in the previous section with locations where the aggregation effect is most likely to take place.

The aggregation effect could happen most probably at (i) networks that have the largest number of peers,

The Y axis is normalized using each curve's peak value.

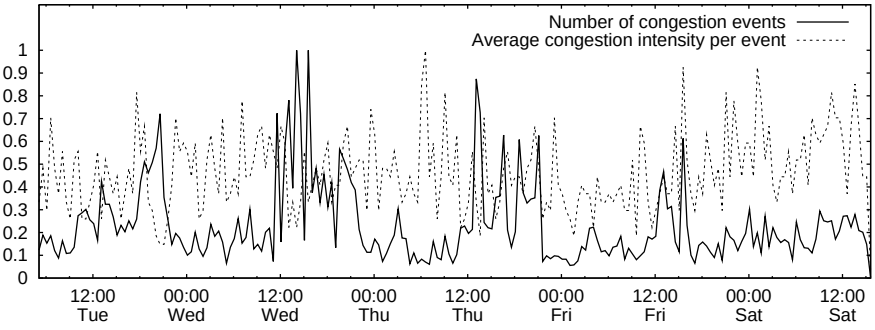


Figure 9: Effect of time-independent hot spots

AS#	Description
174	Cogent Communications, a large Tier-2 ISP.
1299	TeliaNet Global Network, a large Tier-2 ISP.
20965	GEANT, a main European multi-gigabit computer network for research and education purposes, Tier-2.
4323	Time Warner Telecom, a Tier-2 ISP in US.
3356	Level 3 Communications, a Tier-1 ISPs.
237	Merit, a Tier-2 network in US.
6461	Abovenet Communications, a large Tier-2 ISP.
27750	RedCLARA, a backbone connects the Latin-American National Research and Education Networks to Europe.
6453	Teleglobe, a Tier-2 ISP.
2914	NTT America, a Tier-1 ISPs.
3549	Global Crossing, a Tier-1 ISPs.
11537	Abilene, an Internet2 backbone network in US.
4538	China Education and Research Network.

Table 5: Backbone networks with strong hot spots

and (*ii*) ISPs that are most aggressively promoting customer access. Table 6 shows the networks within the top ten of both categories that match locations of the top 20 correlated links and the time-independent hot spots that we measured. For the top ten networks with the largest number of peers, we use ranks provided by *FixedOrbit* [3]. For the top ten networks most aggressively promoting customer access, we use the ranks in terms of the *Renesys customer base index* [6] of three continents. The definition of the Renesys customer base index is highlighted in the table.

Table 6 shows a remarkable location match among our results and statistics collected by others. Indeed, Table 6 reveals that almost all the hot spots shown in Table 5 and Figure 8 locate at (*i*) networks that have the largest number of peers [3], or (*ii*) ISPs that are aggressively promoting customer access. Therefore, we infer that the phenomena of both the congestion correlation and time-independent hot spots could be closely related to the *AS-level traffic aggregation effect*. First, an upstream link could bear congestion correlation with a downstream AS-level aggregation link. Second, time-independent hot spots usually take place at AS-level aggregation points.

5. RELATED WORK

Our system and findings relate to other network monitoring systems, Internet tomography, root-cause- and performance-modeling analysis that we outline below.

Triggered-based Monitoring. One of the key features of our monitoring system is its triggered measurement nature. In this context, Zhang *et al.*'s PlanetSeer [38] monitoring system is closest to ours, both in spirit and design. PlanetSeer detects network path anomalies such as outages or loops using passive observations of the CoDeeN CDN [2], and then initiates active probing. Contrary to PlanetSeer, in absence of passive traffic monitoring, we use a mesh-like light active monitoring subsystem TMon. Also, instead of monitoring loops and outages, we focus on congestion hotspots. Understanding if and how loops or outages affect congestion is a part of our intended future work.

Other systems that apply a triggered-measurement approach include Boschi *et al.*'s SLA-validation system [16] and Wen *et al.*'s on-demand monitoring system [36]. In addition to the fact that our system's goals are fundamentally different from all the above, it also requires measuring entire network *areas*, and hence triggering a large number of vantage points, *concurrently*.

Information Plane. Our congestion monitoring system relates to information dissemination systems that monitor and propagate important network performance inferences to end-points, (*e.g.*, [13, 21, 28, 37]). In general, such systems could be divided into two types. The

Rank	Network	Peers
1	UUNET	2,346
2	AT&T WorldNet	2,092
3	Level 3 Comm.	1,742
5	Cogent Comm.	1,642
7	Global Crossing	1,041
8	Time Warner	918
9	Abovenet	798

(a) Matched locations in the top ten networks defined by the number of peers

North America

Rank	ISP
1	Level 3 Comm.
2	UUNET
3	AT&T WorldNet
6	Cogent Comm.
9	Global Crossing

Europe

Rank	ISP
1	Level 3 Comm.
2	TeliaNet Global Network
4	Global Crossing
8	Teleglobe

Asia

Rank	ISP
2	NTT America
6	UUNET
8	AT&T WorldNet
9	Level 3 Comm.
10	Teleglobe

(b) Matched locations in the top ten ISPs defined by the Renesys customer base index across three continents 02/2006

The Renesys Customer Base Index [6] is defined based on the following five criteria: (1) Which service providers have the most customer networks? (2) Which service providers are acquiring customer networks at the fastest rate? (3) Which service providers are experiencing the least customer churn? (4) Which service providers have the most customers with only one link to the Internet? (5) Which service providers connect to the most customer networks, both directly and through peering relationships?

Table 6: Matched locations in top ten networks/ISPs defined by the number of peers and the Renesys Customer Base Index

first manages information about network or nodes *under control* of the information plane, (*e.g.* RON [13]), while the second type predicts path performance at *Internet-scale*, (*e.g.*, iPlane [28]). As a result, the first type provides information over short time-scales, while the second one does it over much longer time scales, *e.g.*, 6 hours [28].

Our system takes the best of the two worlds: its light mesh-like monitoring approach enables covering Internet-wide areas, yet its triggered-based approach helps effectively focus on a smaller number of *important events* [32], and consequently disseminate the information about the same over shorter time scales.

Locating Internet Bottlenecks. A number of tools have been designed to detect and locate Internet bottlenecks and their properties, *e.g.*, [10,27,29,31]. Common to most of these tools is that they are designed for monitoring a *single* end-to-end path, and hence may not be suitable for large-scale Internet-scale monitoring due to large measurement overhead. Because we depend upon a lightweight measurement tool [22], we are capable of generating concurrent measurements over a larger Internet area and reveal correlation among concurrently congested hot spots, yet without overloading either the network or the monitoring vantage points.

Internet Correlations. The Internet is a complex system composed of thousands of ASes. Necessarily, events happening in one part of the network can have repercussions on other network parts. For example, Sridharan *et al.* [33] find correlation between route dynamics and routing loops; Feamster *et al.* [25] show that there exists correlation between link failures and BGP routing messages; Teixeira *et al.* [35] find that hot-potato routing can trigger BGP events; on the other side, Agarwal *et al.* [9] show that BGP routing changes can cause traffic shifts in a single backbone network.

6. CONCLUSIONS

In this paper, we performed an in-depth study on Internet congestion behavior with a focus on the Internet core. We developed a large-scale triggered monitoring system which integrates the following unique properties: (*i*) it is able to locate and track congestion events *concurrently* at a large fraction of Internet links; (*ii*) it exploits triggering mechanisms to effectively allocate measurement resources to hot spots; (*iii*) it deploys a set of novel online path-selection algorithms that deliver highly-accurate observations about the underlying con-

gestion. Our system's ability to directly observe distinct link-level congestion events concurrently allows a much deeper understanding of Internet-wide congestion.

Our major findings from experiments using this system are as follows: (*i*) Congestion events in the core can be highly correlated. Such correlation can span across up to three neighboring ASes. (*ii*) There are a small number of hot spots between or within large backbone networks exhibiting highly intensive time-independent congestion. (*iii*) The phenomena of both the congestion correlation and the time-independent hot spots could be closely related to the AS-level traffic aggregation effect. (*iv*) Congestion dynamics in the core is quite different from that at edges.

7. REFERENCES

[1] Abilene aggregate network statistics. http://stryper.grnoc.iu.edu/abilene/aggregate/html/.
[2] CoDeeN. http://codeen.cs.princeton.edu/.
[3] FixedOrbit. http://fixedorbit.com/.
[4] Merit Network. http://www.merit.edu/.
[5] PlanetLab. http://www.planet-lab.org/.
[6] Renesys Corporation. http://www.renesys.com/.
[7] Skype. http://www.skype.com/.
[8] Verio backbone SLA terms and conditions. http://www.verio.com/global-ip-guarantee/.
[9] S. Agarwal, C. Chuah, S. Bhattacharyya, and C. Diot. The impact of BGP dynamics on intra-domain traffic. In *SIGMETRICS'04*.
[10] A. Akella, S. Seshan, and A. Shaikh. An empirical evaluation of wide-area Internet bottlenecks. In *IMC'03*.
[11] A. Akella, A. Shaikh, and S. Seshan. A comparison of overlay routing and multihoming route control. In *SIGCOMM'04*.
[12] A. Akella, A. Shaikh, and S. Seshan. Multihoming performance benefits: An experimental evaluation of practical enterprise strategies. In *USENIX Technical Conference*, 2004.
[13] D. Anderson, H. Balakrishnan, M. Kaashoek, and R. Morris. Resilient overlay networks. In *SOSP'01*.
[14] T. Anderson, A. Collins, A. Krishnamurthy, and J. Zahorjan. PCP: Efficient endpoint congestoin control. In *NSDI'06*.
[15] P. Barford, A. Bestavros, J. Byers, and M. Crovella. On the marginal utility of network topology measurements. In *SIGCOMM IMW'01*.
[16] E. Boschi, M. Bossardt, and T. Dubendorfer. Validating inter-domain SLAs with a programmable traffic control system. In *IWAN'05*.
[17] L. Brakmo and L. Peterson. TCP Vegas: End to end congestion avoidance on a global Internet. *IEEE JSAC*, 1995.
[18] T. Bu, N. Duffield, F. Presti, and D. Towsley. Network tomography on general topologies. In *SIGMETRICS'02*.
[19] C. Caceres, N. Duffield, J. Horowitz, and D. Towsley. Multicast-based inference of network-internal loss characteristics. *IEEE Transactions on Information Theory*, 1999.
[20] Y. Chen, D. Bindel, H. Song, and R. Katz. An algebraic approach to practical and scalable overlay network monitoring. In *SIGCOMM'04*.
[21] D. Clark, C. Partridge, J. Ramming, and J. Wroclawski. A knowledge plain for the Internet. In *SIGCOMM'03*.
[22] L. Deng and A. Kuzmanovic. Monitoring persistently congested Internet links. In *ICNP'08*.
[23] M. Dischinger, A. Haeberlen, K. Gummadi, and S. Saroiu. Characterizing residential broadband networks. In *IMC'07*.
[24] N. Duffield, J. Horowits, D. Towsley, W. Wei, and T. Friedman. Multicast-based loss inference with missing data. *IEEE JSAC*, 2002.
[25] N. Feamster, D. Anderson, H. Balakrishnan, and F. Kaashoek. Measuring the effects of Internet path failures on reactive routing. In *SIGMETRICS'03*.
[26] C. Fraleigh, S. Moon, B. Lyles, C. Cotton, M. Khan, D. Moll, R. Rockwell, T. Seely, and C. Diot. Packet-level traffic measurements from the Sprint IP backbone. *IEEE Network*, 2003.
[27] N. Hu, L. Li, Z. Mao, P. Steenkiste, and J. Wang. Locating Internet bottlenecks: Algorithms, measurements, and implications. In *SIGCOMM'04*.
[28] H. Madhyastha, T. Isdal, M. Piatek, C. Dixon, T. Anderson, A. Krishnamurthy, and A. Venkataramani. iPlane: An information plane for distributed services. In *OSDI'06*.
[29] R. Mahajan, N. Spring, D. Wetherall, and T. Anderson. User-level Internet path diagnostics. In *SOSP'03*.
[30] V. Padmanabhan, L. Qiu, and H. Wang. Server-based inference of Internet link lossiness. In *INFOCOM'03*.
[31] V. Ribeiro, R. Riedi, and R. Baraniuk. Locating available bandwidth bottlenecks. *IEEE Internet Computing*, 2004.
[32] H. Song, L. Qiu, and Y. Zhang. NetQuest: A flexible framework for large-scale network measurement. In *SIGMETRICS'06*.
[33] A. Sridharan, S. Moon, and C. Diot. On the correlation between route dynamics and routing loops. In *IMC'03*.
[34] C. Tang and P. McKinley. On the cost-quality tradeoff in topology-aware overlay path probing. In *ICNP'03*.
[35] R. Teixeira, A. Shaikh, T. Griffin, and J. Rexford. Dynamics of hot-potato routing in IP networks. In *SIGMETRICS'04*.
[36] Z. Wen, S. Triukose, and M. Rabinovich. Facilitating focused Internet measurements. In *SIGMETRICS'07*.
[37] P. Yalagandula, P. Sharma, S. Banarjee, S. Basu, and S. Lee. S3: A scalable sensing service for monitoring large networked systems. In *SIGCOMM INM'06*.
[38] M. Zhang, C. Zhang, V. Pai, L. Peterson, and R. Wang. PlanetSeer: Internet path failure monitoring and characterization in wide-area services. In *OSDI'04*.
[39] Y. Zhang, N. Duffield, V. Paxson, and S. Shenker. On the consistency of Internet path properties. In *SIGCOMM IMW'01*.
[40] Y. Zhao, Y. Chen, and D. Bindel. Towards unbiased end-to-end network diagnosis. In *SIGCOMM'06*.