Pantheon: the training ground for Internet congestion-control research

https://pantheon.stanford.edu

Francis Y. Yan[†], Jestin Ma[†], Greg D. Hill[†], Deepti Raghavan[¶], Riad S. Wahby[†], Philip Levis[†], Keith Winstein[†]

[†]Stanford University, [¶]Massachusetts Institute of Technology

July 13, 2018

Congestion control

Cornerstone problem in computer networking

- Avoids congestion collapse
- Allocates resources among users
- Affects every application using TCP socket

Status quo of congestion control research

MEASURING BOTTLENECK BANDWIDTH AND ROUND-TRIP PROPAGATION TIME

NEAL CARDWELL
YUCHUNG CHENG
C. STEPHEN GUNN
SOHEIL HASSAS YEGANEH
VAN JACOBSON

By all accounts, today's Internet is not moving data as well as it should. Most of the world's cellular users experience delays of seconds to minutes; public Wi-Fi in airports and conference venues is often worse. Physics and climate researchers need to exchange petabytes of data with global collaborators but find their carefully engineered multi-Gbps infrastructure often delivers at only a few Mbps over intercontinental distances.[6]

These problems result from a design choice made when TCP congestion control was created in the 1980s—interpreting packet loss as "congestion."[1,3] This equivalence was true at the time but was because of technology limitations, not first principles. As NICs (network interface controllers) evolved from Mbps to Gbps and memory chips from KB to GB, the relationship between packet loss and congestion became more tenuous.

Today TCP's loss-based congestion control—even with the current best of breed, CUBIC[11]—is the primary cause of these problems. When bottleneck buffers are large,

Status quo of congestion control research

Stochastic Forecasts Achieve High Throughput and Low Delay over Cellular Networks

Keith Winstein, Anirudh Sivaraman, and Hari Balakrishnan
MIT Computer Science and Artificial Intelligence Laboratory, Cambridge, Mass.
{keithw,anirudh,hari}@mit.edu

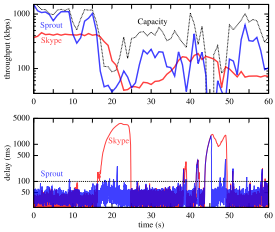
Abstract

Sprout is an end-to-end transport protocol for interactive applications that desire high throughput and low delay. Sprout works well over cellular wireless networks, where link speeds change dramatically with time, and current protocols build up multi-second queues in network gateways. Sprout does not use TCP-style reactive congestion control; instead the receiver observes the packet arrival times to infer the uncertain dynamics of the network path. This inference is used to forecast how many bytes may be sent by the sender, while bounding the risk that packets will be delayed inside the network for too long.

In evaluations on traces from four commercial LTE and 3G networks, Sprout, compared with Skype, reduced self-inflicted end-to-end delay by a factor of 7.9 and achieved 2.2× the transmitted bit rate on average. Compared with Google's Hangout, Sprout reduced delay by a factor of 7.2 while achieving 4.4× the bit rate, and compared with Apple's Facetime, Sprout reduced delay by a factor of 8.7 with 1.9× the bit rate.

Although it is end-to-end, Sprout matched or outperformed TCP Cubic running over the CoDel active queue management algorithm, which requires changes to cellular carrier equipment to deploy. We also tested Sprout as a tunnel to carry competing interactive and bulk traffic (Skype and TCP Cubic), and found that Sprout was able to isolate client application flows from one another.

Figure 1: Skype and Sprout on the Verizon LTE downlink trace. For Skype, overshoots in throughput lead to large standing queues. Sprout tries to keep each packet's delay less than 100 ms with 95% probability.



For an interactive application such as a videoconferencing program that requires both high throughput and low delay, these conditions are challenging. If the application sends at too low a rate, it will waste the oppor-

Status quo of congestion control research

PCC: Re-architecting Congestion Control for Consistent High Performance

Mo Dong[*], Qingxi Li[*], Doron Zarchy[**], P. Brighten Godfrey[*], and Michael Schapira[**]

[*]University of Illinois at Urbana-Champaign
[**]Hebrew University of Jerusalem

Abstract

TCP and its variants have suffered from surprisingly poor performance for decades. We argue the TCP family has little hope of achieving consistent high performance due to a fundamental architectural deficiency: hardwiring packet-level events to control responses. We propose Performance-oriented Congestion Control (PCC), a new congestion control architecture in which each sender continuously observes the connection between its *actions* and *empirically experienced performance*, enabling it to consistently adopt actions that result in high performance. We prove that PCC converges to a stable and fair equilibrium. Across many real-world and challenging environments, PCC shows consistent and often 10× performance improvement, with better fairness and stability than TCP. PCC requires no router hardware support or new packet format.

a very difficult task within TCP's rate control architecture, which we refer to as **hardwired mapping**: certain predefined packet-level events are hardwired to certain predefined control responses. TCP reacts to events that can be as simple as "one packet loss" (TCP New Reno) or can involve multiple signals like "one packet loss and RTT increased by $x\%$" (TCP Illinois). Similarly, the control response might be "halve the rate" (New Reno) or a more complex action like "reduce the window size w to $f(\Delta RTT)w$" (Illinois). The defining feature is that the control action is a direct function of packet-level events.

A hardwired mapping has to make *assumptions* about the network. Take a textbook event-control pair: a packet loss halves the congestion window. TCP *assumes* that the loss indicates congestion in the network. When the assumption is violated, halving the window size can severely degrade performance (e.g. if loss is random, rate

Inconsistent behaviors

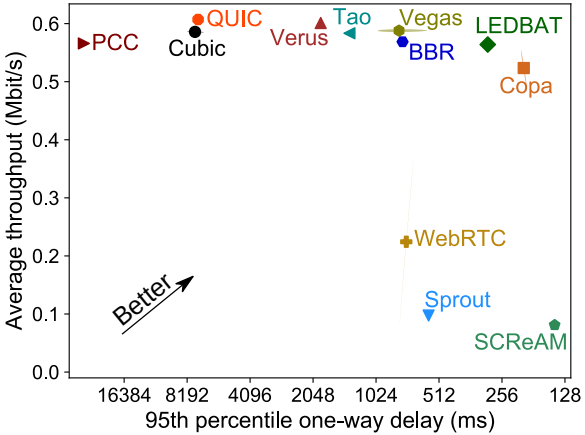


Figure: Colombia to AWS Brazil (cellular, 1 flow, 3 trials, P1391)

Inconsistent behaviors

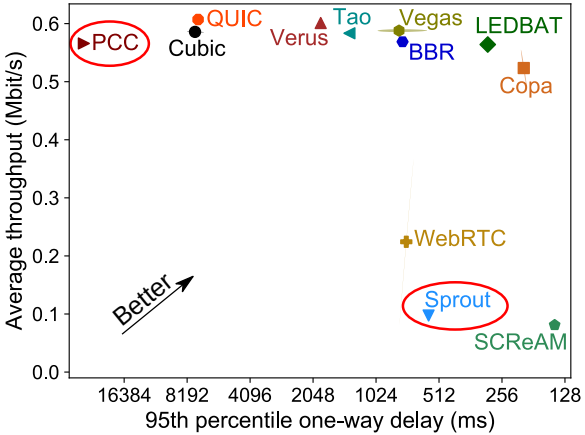


Figure: Colombia to AWS Brazil (cellular, 1 flow, 3 trials, P1391)

Challenges and problems

Every emerging algorithm claims to be the "state-of-the-art"

- ... compared with other algorithms that they picked
- ... evaluated on their own testbeds in real world
- ... and/or on simulators/emulators with their settings
- ... based on the specific results that they collected

Challenges and problems

... compared with other algorithms that they picked

... evaluated on their own testbeds in real world

... and/or on simulators/emulators with their settings

... based on the specific results that they collected

Challenges and problems

... compared with other algorithms that they picked

$\Longrightarrow$ must acquire, compile, and execute prior algorithms

... evaluated on their own testbeds in real world

... and/or on simulators/emulators with their settings

... based on the specific results that they collected

Challenges and problems

... compared with other algorithms that they picked

$\Longrightarrow$ must acquire, compile, and execute prior algorithms

... evaluated on their own testbeds in real world

$\Longrightarrow$ large service operators: risky to deploy, long turnaround time

... and/or on simulators/emulators with their settings

... based on the specific results that they collected

Challenges and problems

... compared with other algorithms that they picked

$\Longrightarrow$ must acquire, compile, and execute prior algorithms

... evaluated on their own testbeds in real world

$\Longrightarrow$ large service operators: risky to deploy, long turnaround time

$\Longrightarrow$ researchers: on a much smaller scale, results may not generalize

... and/or on simulators/emulators with their settings

... based on the specific results that they collected

Challenges and problems

... compared with other algorithms that they picked

$\Longrightarrow$ must acquire, compile, and execute prior algorithms

... evaluated on their own testbeds in real world

$\Longrightarrow$ large service operators: risky to deploy, long turnaround time

$\Longrightarrow$ researchers: on a much smaller scale, results may not generalize

... and/or on simulators/emulators with their settings

$\Longrightarrow$ how to configure the settings?

... based on the specific results that they collected

Challenges and problems

... compared with other algorithms that they picked

$\Longrightarrow$ must acquire, compile, and execute prior algorithms

... evaluated on their own testbeds in real world

$\Longrightarrow$ large service operators: risky to deploy, long turnaround time

$\Longrightarrow$ researchers: on a much smaller scale, results may not generalize

... and/or on simulators/emulators with their settings

$\Longrightarrow$ how to configure the settings?

... based on the specific results that they collected

$\Longrightarrow$ but the Internet is diverse and evolving

TPC™

performance data to the industry... The TPC is a non-profit corporation focused on developing

- Home
- About the TPC
- Benchmarks
  - Enterprise BMs
    - TPC-C
    - TPC-DI
    - TPC-DS
    - TPC-E
    - TPC-H
    - TPC-VMS
  - Express BMs
    - TPCx-BB
    - TPCx-HCI
    - TPCx-HS
    - TPCx-IoT
    - TPCx-V
  - Common Specifications
    - TPC-Pricing
    - TPC-Energy
  - Obsolete BMs
    - TPC-A
    - TPC-App
    - TPC-B
    - TPC-D
    - TPC-R
    - TPC-W
  - Submission Checklist

TPC-C

TPC-C is an On-Line Transaction Processing Benchmark

Approved in July of 1992, TPC Benchmark C is an on-line transaction processing (OLTP) benchmark. types, more complex database and overall execution structure. TPC-C involves a mix of five concurrer database is comprised of nine types of tables with a wide range of record and population sizes. TPC-C supplier, TPC-C is not limited to the activity of any particular business segment, but, rather represents

Specification

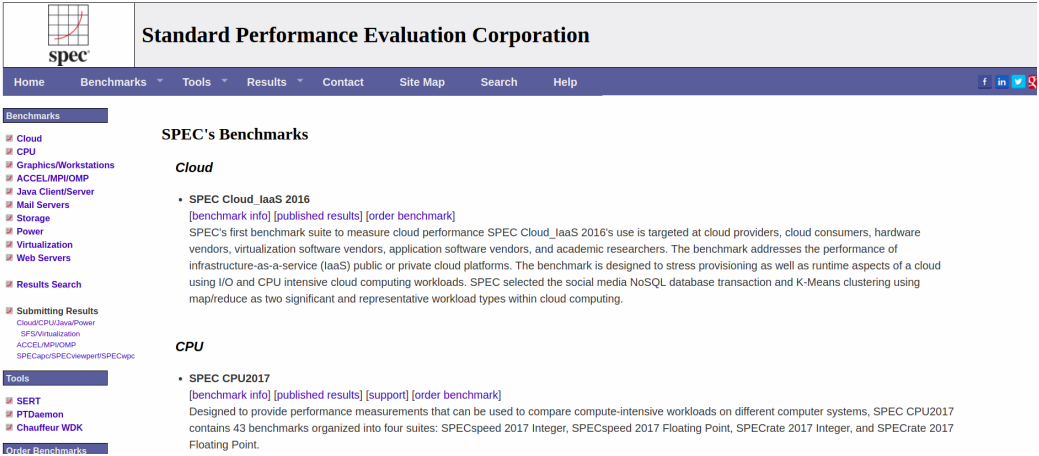
- The current TPC-C specification can be found on the TPC Documentation Webpage.

More Information

- Detailed TPC-C Description
- Frequently Asked TPC-C Questions
- Sigmod HTML Presentation
- Order-Of-Magnitude Advantage on TPC-C Through Massive Parallelism
- Thousands of DebitCredit Transactions-Per-Second: Easy and Inexpensive
- Transaction Performance vs. Moore's Law: A Trend Analysis

Results

- Advanced TPC-C Sorting and Filtering Options
- Results Spreadsheet (downloadable file)



Standard Performance Evaluation Corporation

Home | Benchmarks | Tools | Results | Contact | Site Map | Search | Help

Benchmarks

- Cloud
- CPU
- Graphics/Workstations
- ACCEL/MPI/OMP
- Java Client/Server
- Mail Servers
- Storage
- Power
- Virtualization
- Web Servers

Results Search

Submitting Results
Cloud/CPU/Java/Power
SFS/Virtualization
ACCEL/MPI/OMP
SPECapc/SPECviewperf/SPECwpc

Tools

- SERT
- PTDaemon
- Chauffeur WDK

Order Benchmarks

SPEC's Benchmarks

Cloud

- SPEC Cloud_IaaS 2016
  [benchmark info] [published results] [order benchmark]
  SPEC's first benchmark suite to measure cloud performance SPEC Cloud_IaaS 2016's use is targeted at cloud providers, cloud consumers, hardware vendors, virtualization software vendors, application software vendors, and academic researchers. The benchmark addresses the performance of infrastructure-as-a-service (IaaS) public or private cloud platforms. The benchmark is designed to stress provisioning as well as runtime aspects of a cloud using I/O and CPU intensive cloud computing workloads. SPEC selected the social media NoSQL database transaction and K-Means clustering using map/reduce as two significant and representative workload types within cloud computing.

CPU

- SPEC CPU2017
  [benchmark info] [published results] [support] [order benchmark]
  Designed to provide performance measurements that can be used to compare compute-intensive workloads on different computer systems, SPEC CPU2017 contains 43 benchmarks organized into four suites: SPECspeed 2017 Integer, SPECspeed 2017 Floating Point, SPECrate 2017 Integer, and SPECrate 2017 Floating Point.

IMAGENET

14,197,122 images, 21841 synsets indexed

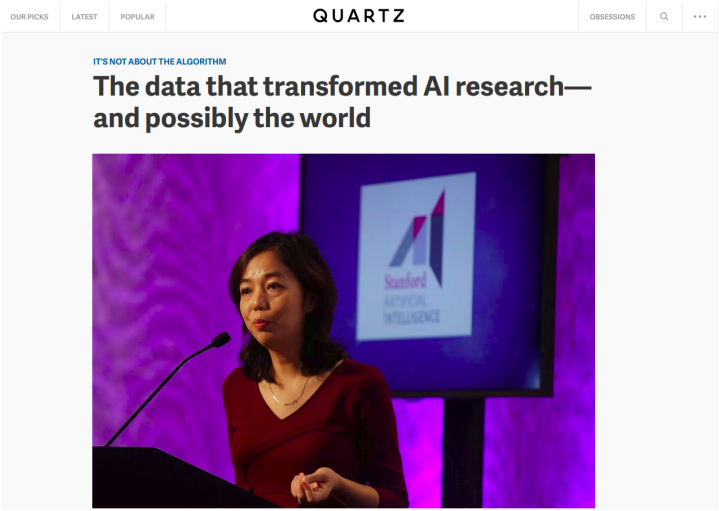
Explore Download Challenges Publications CoolStuff About

Not logged in. Login | Signup

ImageNet is an image database organized according to the WordNet hierarchy (currently only the nouns), in which each node of the hierarchy is depicted by hundreds and thousands of images. Currently we have an average of over five hundred images per node. We hope ImageNet will become a useful resource for researchers, educators, students and all of you who share our passion for pictures.
Click here to learn more about ImageNet, Click here to join the ImageNet mailing list.

What do these images have in common? *Find out!*

Check out the ImageNet Challenge on Kaggle!


OUR PICKS
LATEST
POPULAR
QUARTZ
OBSESSIONS
IT'S NOT ABOUT THE ALGORITHM
The data that transformed AI research—and possibly the world
Stanford
ARTIFICIAL
INTELLIGENCE

Shared, reproducible benchmarks can lead to huge leaps performance and transform technologies by making them scientific.

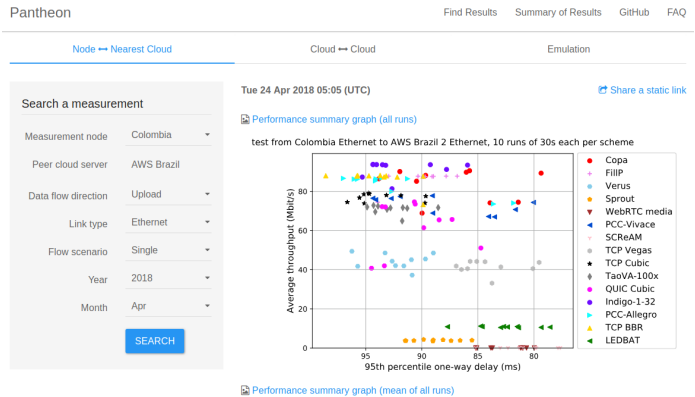
Pantheon: a community evaluation platform for congestion control

- a common reference set of 15+ benchmark algorithms
- a diverse testbed of network nodes in 10+ countries
  - Cellular and wired: U.S., Mexico, Brazil, Colombia, India, China
  - Wired networks only: U.K., Australia, Japan, Korea, Saudi Arabia
- a collection of *calibrated emulators* and pathological emulators
- a continous-testing system and a public archive of searchable results at `https://pantheon.stanford.edu`

This is a reproducible talk!

e.g., P123: https://pantheon.stanford.edu/result/123/

pantheon.stanford.edu



pantheon.stanford.edu

Summary of Results

We summarize the performance of each scheme using the metric **log(mean throughput / mean 95th percentile delay)** (a version of Kleinrock's power metric).

For each test, the highest-power scheme is shown in light green, and the lowest-power scheme is in black.

Best — Worst

☐ *represents no data available, either if the scheme did not run, or failed during tests. This summary includes only single-flow experiments; for individual results (including multi-flow experiments) please see the Find Results tab.*

Date (UTC)	Description	TCP BBR	Copa	TCP Cubic	Fillp	Indigo	LEDBAT	PCC-Allegro	PCC-Expr	QUIC Cubic	SCReAM	Sprout	TaoVA-100x	TCP Vegas	Verus	PCC-Vivace	WebRTC media
06/06/2018	GCE Tokyo to GCE Iowa, Ethernet																
06/06/2018	GCE London to GCE Sydney, Ethernet																
06/06/2018	GCE Iowa to GCE Tokyo, Ethernet																
06/06/2018	GCE Sydney to GCE London, Ethernet																
06/06/2018	Brazil to AWS Brazil 1, Ethernet																
06/06/2018	India to AWS India 1, Ethernet																
06/06/2018	Colombia to AWS Brazil 2, Ethernet																
06/06/2018	Mexico to AWS California 2, Ethernet																

Pantheon: a community resource

- A common language in congestion control
  - benchmark algorithms
  - shared testbeds
  - public data

Pantheon: a community resource

- A common language in congestion control
  - benchmark algorithms
  - shared testbeds
  - public data
- A training ground for congestion control
  - enables faster innovation and more reproducible research
  - e.g., Vivace (NSDI '18), Copa (NSDI '18), Indigo: a machine-learned congestion control

Outline

A software library of congestion-control algorithms

15+ algorithms

- TCP Cubic, TCP Vegas, TCP BBR, QUIC Cubic, LEDBAT, WebRTC (media), Sprout, Remy, Verus, PCC, SCReAM, FillP, Vivace, Copa, Indigo, ...
- Add your own transport protocol (instructions at pantheon.stanford.edu)

Common testing interface

- A full-throttle flow that runs until killed

Measure performance faithfully without modifications

Key findings

- Measurement study from more than a year of data
- Performance of congestion-control algorithms varies across the type of network path, path direction, and time

Key finding 1: scheme performance varies by path

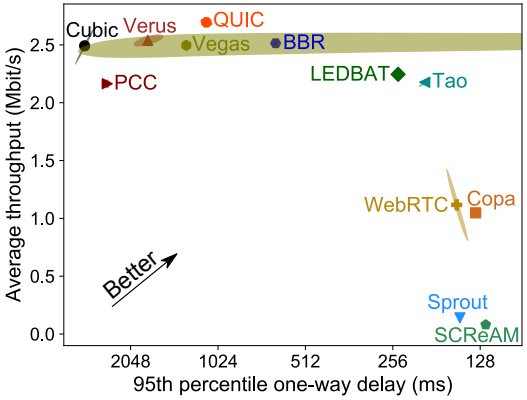


Figure: AWS Brazil to Colombia (cellular, 1 flow, 3 trials, P1392)

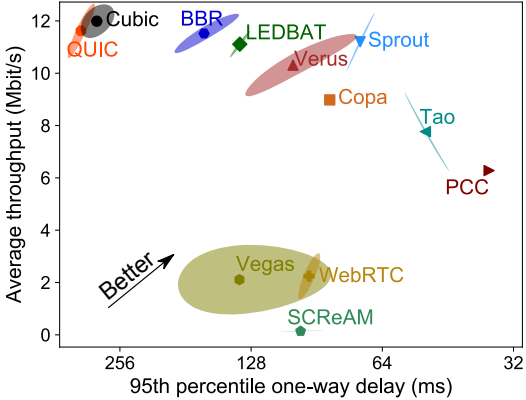


Figure: Stanford to AWS California (cellular, 1 flow, 3 trials, P950)

Key finding 1: scheme performance varies by path

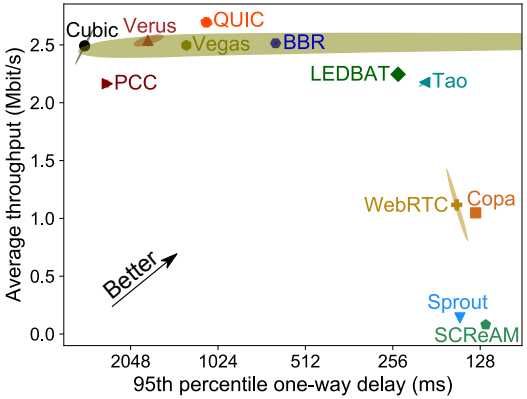


Figure: AWS Brazil to Colombia (cellular, 1 flow, 3 trials, P1392)

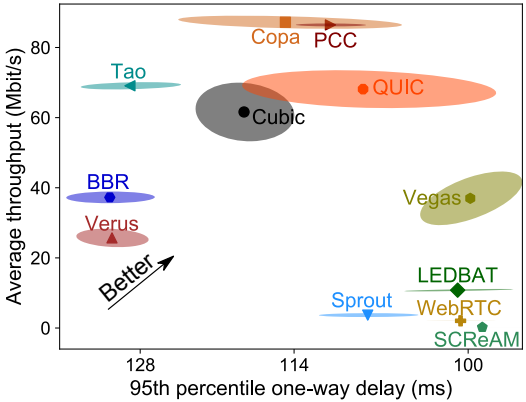


Figure: AWS Brazil to Colombia (wired, 1 flow, 10 trials, P1271)

Key finding 2: scheme performance varies by path direction

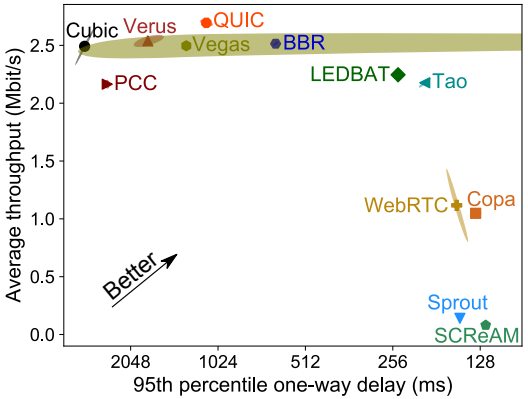


Figure: AWS Brazil to Colombia (cellular, 1 flow, 3 trials, P1392)

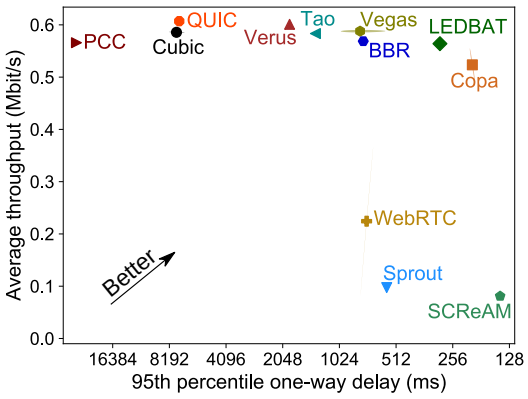


Figure: Colombia to AWS Brazil (cellular, 1 flow, 3 trials, P1391)

Key finding 3: scheme performance varies in time

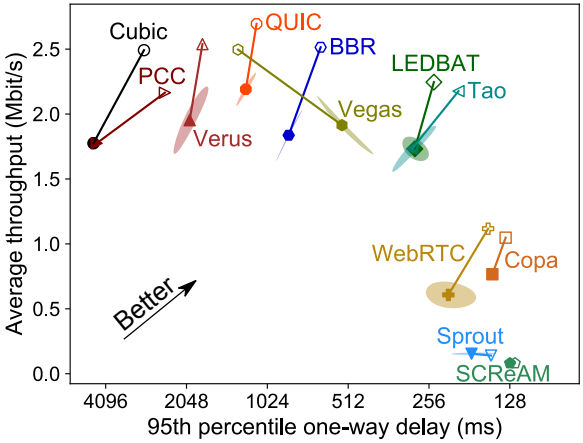


Figure: AWS Brazil to Colombia (cellular, 1 flow, 3 trials, filled dots show performance after 2 days)

Limitations

- Only tests schemes at full throttle
- Nodes are not necessarily representative
- Does not measure interactions between different schemes (ongoing collaboration with CMU)

Outline

Motivations

Simulation/emulation: reproducible and allows rapid experimentation

- ns-2/ns-3, Mininet, Mahimahi, etc.
- fine-grained and detailed, providing a number of parameters

Motivations

Simulation/emulation: reproducible and allows rapid experimentation

- ns-2/ns-3, Mininet, Mahimahi, etc.
- fine-grained and detailed, providing a number of parameters

Open problem

What is the choice of parameter values to faithfully emulate a particular target network?

New figure of merit for network emulators

Replication error

Average difference of the performance of a set of transport algorithms run over the emulator compared with over the target real network path.

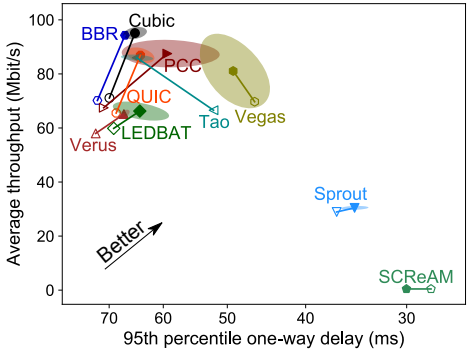


Figure: Filled dots: real results over a network path; open dots: results over an emulator.

Emulator characteristics

Five parameters

- a bottleneck link rate
- a constant propagation delay
- a DropTail threshold for the sender's queue
- a loss rate (per-packet, i.i.d)
- a bit that selects constant rate or Poisson-governed rate

Automatically calibrate emulators to match a network path

Collect a set of results over a particular network path on Pantheon

- average throughput and 95th percentile delay of a dozen algorithms

Automatically calibrate emulators to match a network path

Collect a set of results over a particular network path on Pantheon

- average throughput and 95th percentile delay of a dozen algorithms

Run Bayesian optimization

- Input x: <rate, propagation delay, queue size, loss rate>
- Run twice: constant rate and Poisson-governed rate
- Objective function $f(x)$: mean replication error
- Prior: Gaussian process
- Acquisition function: expected improvement

Results of calibrated emulators

- Trained emulators calibrated to 6 of Pantheon's paths
  - Nepal (Wi-Fi), Colombia (cellular), Mexico (cellular), China (wired), India (wired), and Mexico (wired)
  - including single flow and three flows for Mexico (wired)
- Each for about 2 hours on 30 machines with 4 cores each

Results of calibrated emulators

- Trained emulators calibrated to 6 of Pantheon's paths
  - Nepal (Wi-Fi), Colombia (cellular), Mexico (cellular), China (wired), India (wired), and Mexico (wired)
  - including single flow and three flows for Mexico (wired)
- Each for about 2 hours on 30 machines with 4 cores each
- Replication error is within **17%** on average

Representative calibration result

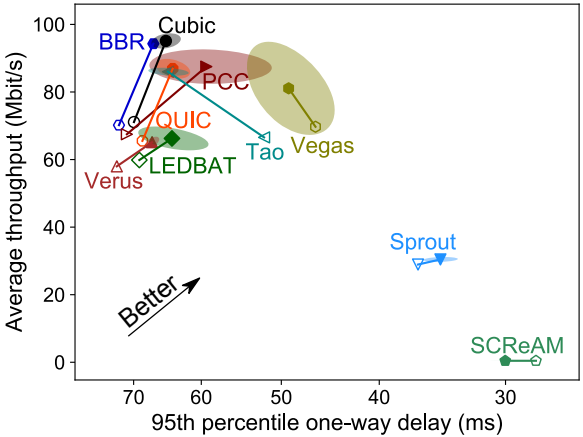


Figure: AWS California to Mexico (wired, 3 flows, 10 trials, P1237). Mean replication error: 14.4%.

Pathological emulators

Suggested by BBR team at Google

- Very small buffer sizes
- Severe ACK aggregation
- Token-bucket policers

Outline

Pantheon use cases

- Vivace (NSDI '18): validating a new scheme in the real world
- Copa (NSDI '18): iterative design with measurements
- Other ongoing projects:
  - Mixed-scheme multi-flow measurements (CMU)
  - FillP (Huawei)

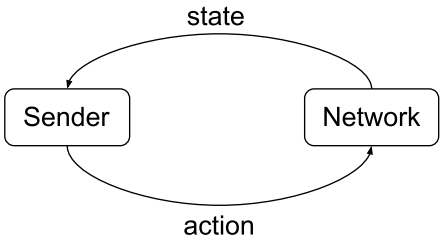
Indigo: a machine learning design enabled by Pantheon

At step t:

- state_t: congestion signals
- action_t: congestion window adjustment

Indigo's goal

Learning to map state_t to action_t using a model



Design

step

- 10 ms

state

- EWMA of the queuing delay
- EWMA of the sending rate
- EWMA of the receiving rate
- Current congestion window size
- Previous action taken

action

- $\div 2, -10$ (packets), $+0, +10$ (packets), $\times 2$

model

- Input: a state
- $\rightarrow$ 1-layer LSTM network
- $\rightarrow$ Softmax classifier
- $\rightarrow$ Output: an action

Imitation learning: expert policy

Congestion-control oracle

- $\text{state}_t \rightarrow \text{action}_t^*$
- Outputs an action that brings congestion window closest to the ideal size

Imitation learning: expert policy

Congestion-control oracle

- $\text{state}_t \rightarrow \text{action}_t^*$
- Outputs an action that brings congestion window closest to the ideal size

Ideal size

- Only exists in emulators
- BDP: simple emulated links with a fixed bandwidth and min RTT
- Search around BDP otherwise

Method

- Imitation learning with DAgger (dataset aggregation)
- Trained on 24 synthetic emulators
  - all combinations of (5, 10, 20, 50, 100, 200 Mbps) link rate and (10, 20, 40, 80 ms) min one-way delay
  - infinite queues and no loss
- and 6 calibrated emulators of Pantheon
  - help mitigate the "distribution mismatch"

Real-world results

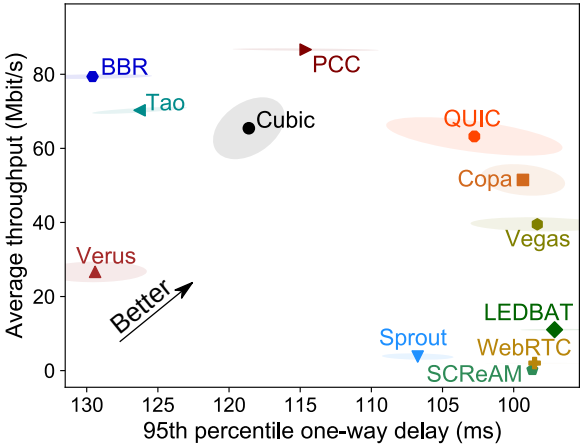


Figure: AWS Brazil to Colombia (wired, 1 flow, 10 trials, P1439)

Real-world results

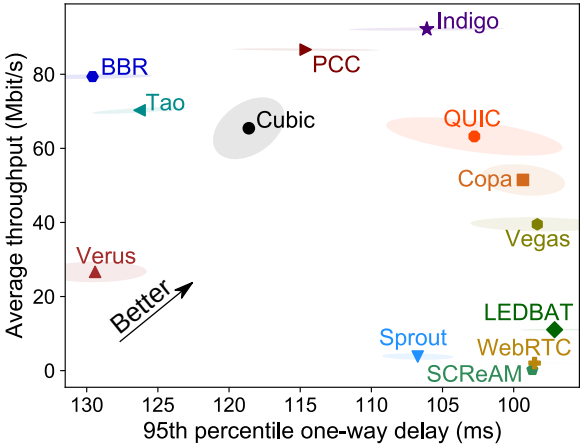


Figure: AWS Brazil to Colombia (wired, 1 flow, 10 trials, P1439)

Real-world results

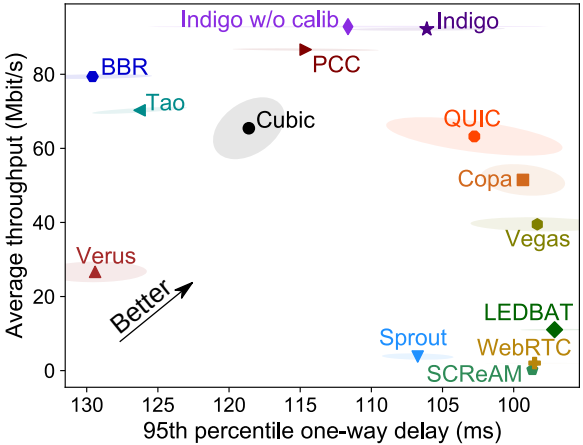


Figure: AWS Brazil to Colombia (wired, 1 flow, 10 trials, P1439)

Real-world results

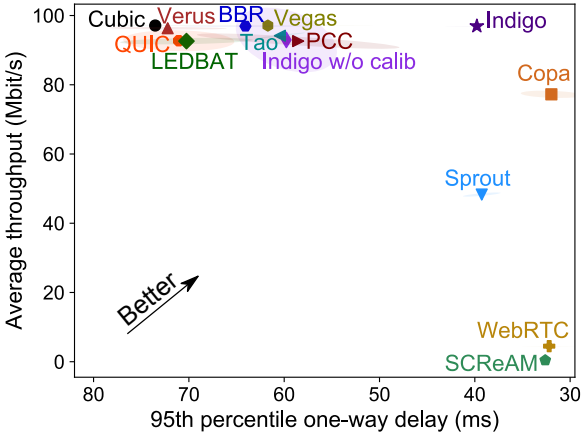


Figure: India to AWS India (wired, 3 flows, 10 trials, P1476)

Outline

Conclusion

Pantheon

- A community evaluation platform for congestion control
  - benchmark algorithms, shared testbeds, and public data
- A training ground for congestion control
  - enables faster innovation and more reproducible research
  - e.g., Vivace (NSDI '18), Copa (NSDI '18), Indigo: a machine-learned congestion control

Conclusion

Pantheon

- A community evaluation platform for congestion control
  - benchmark algorithms, shared testbeds, and public data
- A training ground for congestion control
  - enables faster innovation and more reproducible research
  - e.g., Vivace (NSDI '18), Copa (NSDI '18), Indigo: a machine-learned congestion control

Calibrated emulators and pathological emulators

- Replication error — new figure of merit for network emulators
- Automatically calibrate an emulator to accurately model real networks

Q&A

Visit https://pantheon.stanford.edu for more results and the paper!

Pantheon-tunnel

- Virtual private network (VPN)
- | IP | UDP | UID (unique identifier) | original IP datagram |
- Unambiguously logs every packet
- Tracks the size, time sent and time received of each IP datagram
- Either endpoint can be the sender or receiver even if behind a NAT

Evaluation of Pantheon-tunnel

- Causes no significant change in throughput or delay ($p < 0.2$)

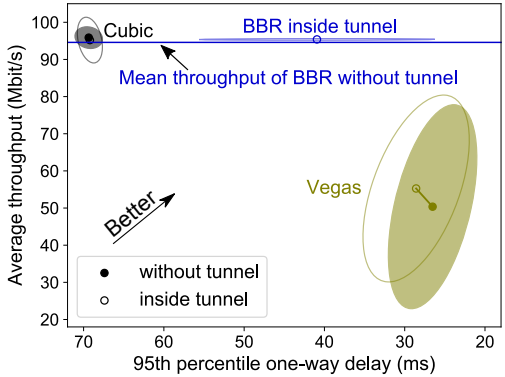


Figure: India to AWS India (wired, 1 flow, 50 trials)

Indigo: fairness evaluation

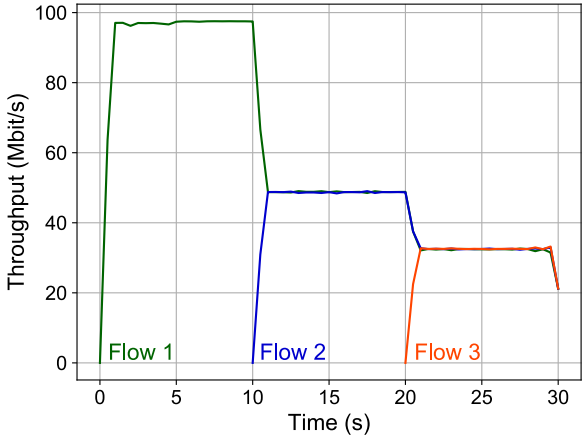


Figure: Time-domain three-flow test (one trial in P1476)