FAST Kernel:

Background Theory and Experimental Results *

(Extended Abstract)

C. Jin, D. Wei, S. H. Low
G. Buhrmaster, J. Bunn, D. H. Choe, R. L. A. Cottrell
J. C. Doyle, H. Newman, F. Paganini, S. Ravot, S. Singh †

http://netlab.caltech.edu/FAST/

January 12, 2003

A breakthrough that has allowed the Internet to expand by five orders of magnitude in size and in backbone speed in the last 15 years was the invention in 1988 by Jacobson of an end-to-end congestion control algorithm in TCP (Transmission Control Protocol) [1]. The algorithm is a distributed and asynchronous method to share network resources among competing users. TCP has been carrying more than 90% of the Internet traffic and Jacobson's algorithm is instrumental in preventing the Internet from congestion collapse while the Web exploded in the 1990s.

This algorithm, designed when most parts of the Internet could barely carry the traffic of a single uncompressed voice call, however, cannot scale to the future ultrascale networks that must be able to carry the traffic of 1.5 million concurrent voice calls. This is due to serious equilibrium and stability problems in high capacity long distance networks, and has led to doubts on whether the current TCP paradigm of end-to-end control coupled with packet switching is suitable for future networks.

In this talk, we describe a new TCP congestion control method that can stably achieve high utilization and throughput at multi-Gbps over long distance.

1 Background theory

There is a vast literature on experimental and analytical work on TCP congestion control algorithms, and on understanding the equilibrium and stability properties of large scale networks under end-to-end control. Here, we only provide references that are *directly* related to the FAST kernel.[1]

A congestion control algorithm consists of two components, a source algorithm, implemented in TCP, that adapts sending rate (or window) to congestion information in its path, and a link algorithm, implemented in routers, that updates and feeds back a measure of congestion to sources that traverse the link. Typically, the link algorithm is implicit and the measure of congestion is either packet loss probability or queueing delay. For example, the current protocol TCP Reno and its variants use loss probability as a congestion measure, and TCP Vegas [2] uses queueing delay as a congestion measure

*To be presented at the First International Workshop on Protocols for Fast Long-Distance Networks, February 3-4, 2003, CERN, Geneva, Switzerland.

†G. Bugrmaster and L. Cottrell are with SLAC (Stanford Linear Accelerator Center), Stanford, CA. F. Paganini is with EE Department, UCLA. S. Ravot is with both Caltech, Pasadena, CA and CERN, Geneva, Switzerland. All other authors are with Caltech, Pasadena, CA.

[1]We emphasize that we have *not* provided full reference to related work in this extended abstract, and will provide it in the full paper.

[3, 4]. Both are implicitly updated by the queueing process and implicitly fed back to sources via end-to-end loss and delay, respectively.

The source-link algorithm pair, referred to here as TCP/AQM (active queue management) algorithms[2], forms a distributed feedback system. The equilibrium and dynamic properties of this system determine the network performance, such as throughput, utilization, delay, loss, fairness, response to congestion, and robustness to uncertainties.

The equilibrium properties of the network can be readily understood by interpreting TCP/AQM as a distributed algorithm over the Internet to solve a global optimization problem [5, 6, 4]. Different TCP and AQM protocols all solve the same prototypical problem, but with different objective (utility) functions and using different iterative procedures. Indeed, we can regard each source as having a utility function, as a function of its rate. The goal of TCP/AQM is to maximize the aggregate utility subject to link capacity constraints. TCP iterates on the source rates and AQM iterates on the congestion measures. The throughput, utilization, loss, delay and fairness of the network are determined by the equilibrium values of these variables, and therefore can be understood by studying the underlying utility maximization problem.[3]

This is the case if the TCP/AQM algorithms are stable. It has been shown, however, that the current algorithms can become unstable as delay increases, or more strikingly, as network capacity increases [7, 8]! This is one of the main difficulties in operating in fast long-distance networks.

If we can rebuild both TCP (source) algorithm and AQM (link) algorithm from scratch, then we now know [9] how to design TCP/AQM algorithm pairs, that are as simple and decentralized as the current protocol, but that maintain linear stability in networks of *arbitrary* capacity, size, delay and load. The main insight from this work is that, to maintain stability in high capacity large distance networks, sources should scale down their responses by their individual round trip delays and links should scale down their responses by their individual capacity. This insight combined with that from [4] leads to a TCP algorithm that can maintain linear stability without having to change the current link algorithm [10, 11]. Moreover, it suggests an incremental deployment strategy where performance steadily improves as ECN (Explicit Congestion Notification) deployment proliferates.

This suggests that by modifying just the TCP kernel at the *sending hosts*, we can stabilize the Internet with the current routers. It motivates the implementation of the FAST kernel.

The implementation of the FAST kernel involves a number of innovations that are crucial to achieve scalability. As the Internet scales up in speed and size, its stability and performance becomes harder to control. The emerging theory allows us to understand the equilibrium and stability properties of large networks under end-to-end control and is indispensable to the design and optimization of the Internet. It is the foundation of the FAST kernel and plays an important role in its implementation, providing a framework to understand issues, clarify ideas and suggest directions, leading to a more robust and better performing implementation.

2 Experimental results

The Caltech FAST kernel was demonstrated publicly for the first time in a series of experiments conducted during the SuperComputing Conference (SC2002) in Baltimore, MD, in November 16–22 2002 by a Caltech-SLAC research team working in partnership with CERN, DataTAG, StarLight, Cisco, and Level(3).

The demonstrations used a 10 Gbps link donated by Level(3) between Starlight (Chicago) and Sunnyvale, as well as the DataTAG 2.5 Gbps link between Starlight and CERN (Geneva), and the

[2]We will henceforth refer it as a "TCP algorithm" even though we really mean the congestion control algorithm in TCP.

[3]The source rates and congestion measures correspond to primal and dual variables, respectively, in the underlying constrained optimization problem. Their properties are well studied in optimization theory and have direct implications on the equilibrium properties of the network.

Abilene backbone of Internet2. The network routers and switches at Starlight and CERN were used together with a GSR 12406 router loaned by Cisco at Sunnyvale, additional Cisco modules loaned at Starlight, and sets of dual Pentium 4 servers each with dual Gigabit Ethernet connections at Starlight, Sunnyvale, CERN and the SC2002 show floor provided by Caltech, SLAC and CERN. The network setup is shown in Figure 1.

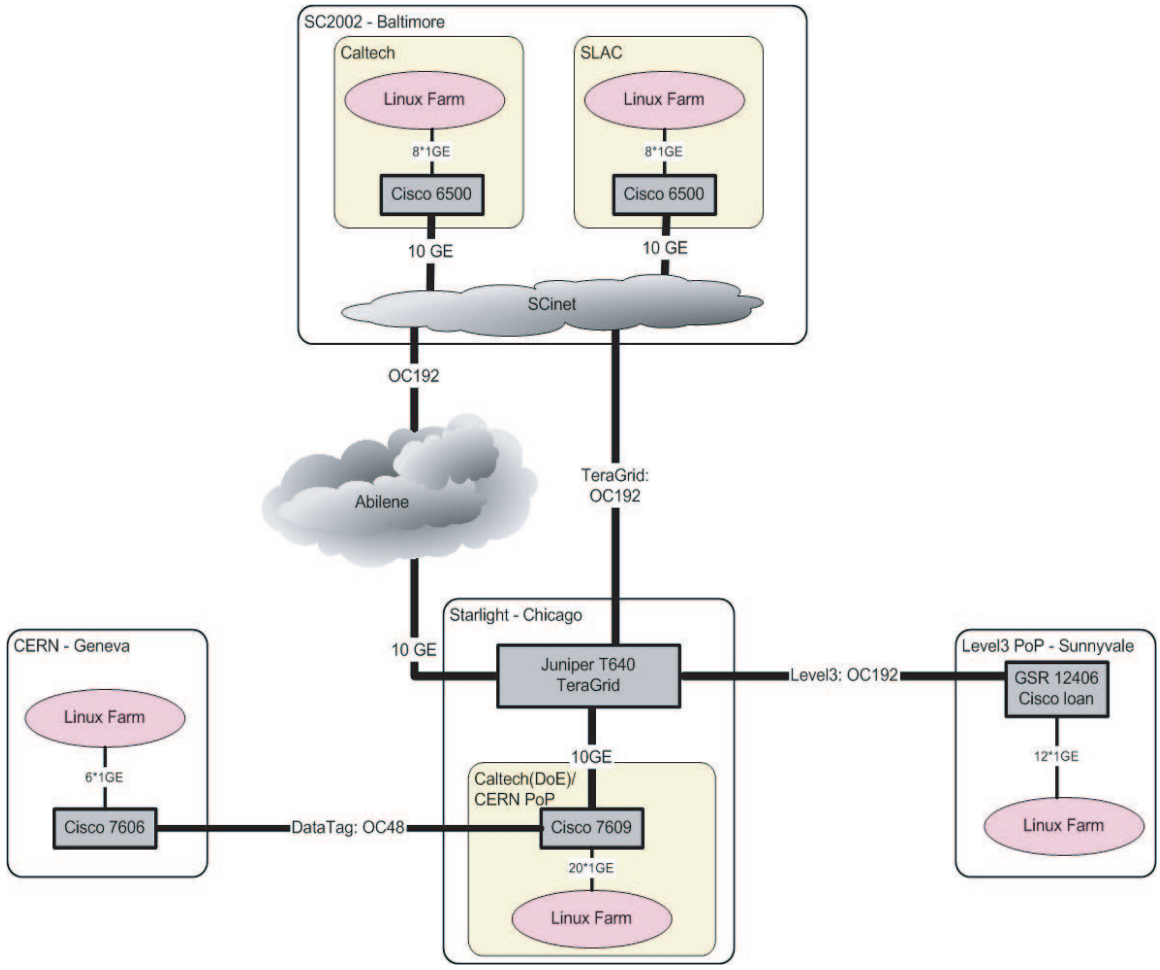

Figure 1: Network setup in SC2002

We have conducted a number of experiments, all using standard MTU (Maximum Transmission Unit), 1500 bytes including TCP and IP headers. In particular, we have demonstrated more than 950 Mbps stably with a single TCP flow between CERN in Geneva and Level(3)'s PoP (point of presence) in Sunnyvale, over a distance of over 6,000 miles on a single Gigabit Ethernet port at each end of the path. The details of five of these experiments are shown in Table 1.

These five experiments are chosen because each of them has run for more than an hour and we have their complete throughput traces. Since the SCinet at SC2002 was shared by all conference participants, it was important to run the experiments long enough to observe how FAST kernel reacts to congestion and packet losses that were inevitable over such a long period. All statistics below are *averages* over the duration of the experiments.

The throughput in each experiment is the ratio of total amount of data transferred and the duration of the transfer. Utilization is the ratio of throughput and bottleneck capacity (Gigabit Ethernet card), excluding the (40-byte) overhead of TCP/IP headers. The "bmps" column is the product of throughput and distance of transfer, measured in bit-meter-per-second. Delay is the minimum round trip time.

The throughput traces of these experiments are shown in Figure 2. These traces, especially those

#flow	bmps 10^{15}	throughput Mbps	utilization	distance km	delay ms	duration s	transfer GB	MTU B
1	9.28	925	95%	10,037	180	3,600	387	1,500
2	18.03	1,797	92%	10,037	180	3,600	753	1,500
7	24.17	6,123	90%	3,948	85	21,600	15,396	1,500
9	31.35	7,940	90%	3,948	85	4,030	3,725	1,500
10	33.99	8,609	88%	3,948	85	21,600	21,647	1,500

Table 1: Experimental results: average statistics

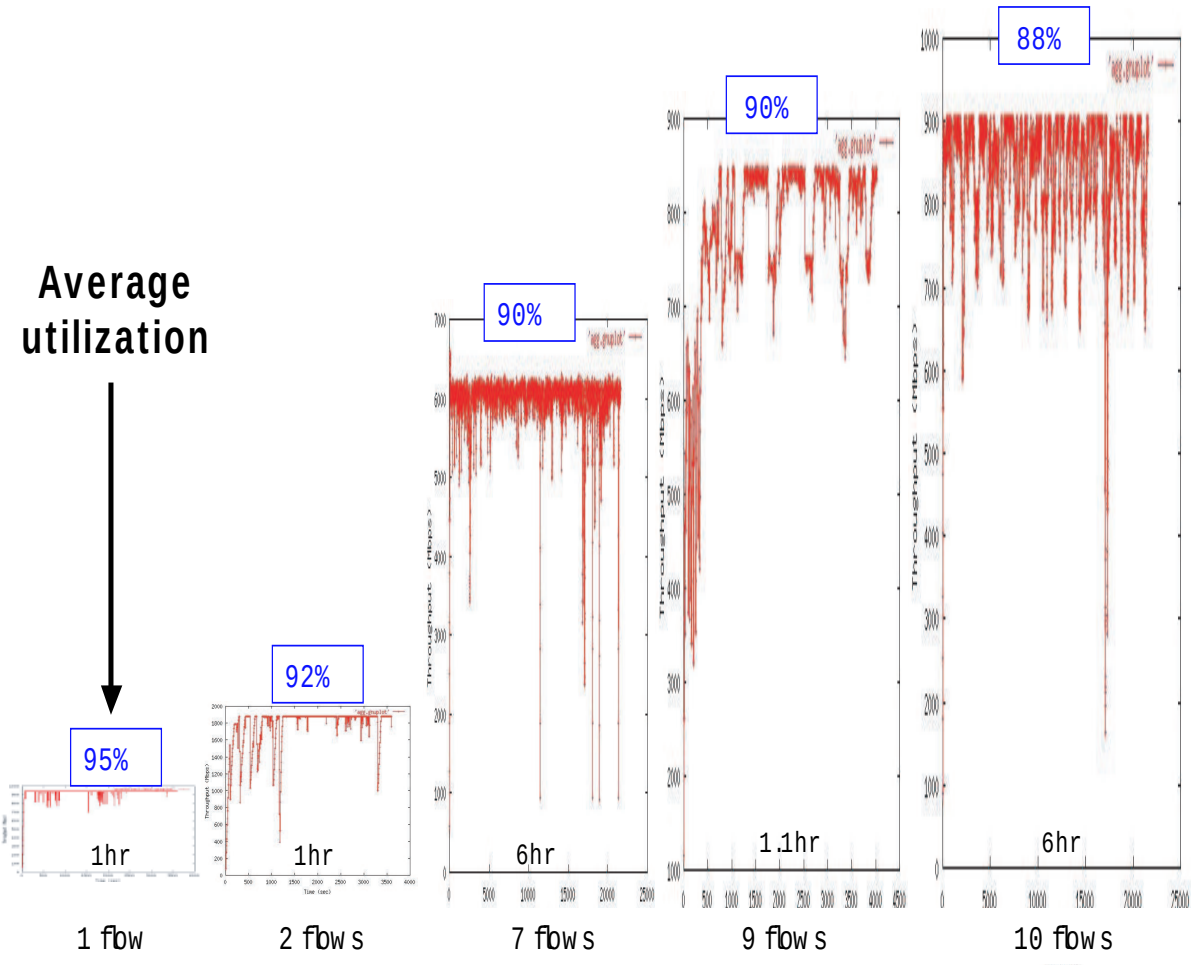

Figure 2: Throughput traces for experiments in Table 1. From left: 1 flow, 2 flows, 7 flows, 9 flows, 10 flows; x-axis is time, y-axis is aggregate throughput, and percentage is utilization.

for 9 and 10 flows, display stable reduction in throughput over several intervals of several minutes each, suggesting significant sharing with other conference participants of network bandwidth.

3 Conclusion

One of the drivers of these developments has been the High Energy and Nuclear Physics (HENP) community, whose explorations at the high energy frontier are breaking new ground in our understanding of the fundamental interactions, structures and symmetries that govern the nature of matter and space-time in our universe. The CMS (Compact Muon Solenoid) Collaboration, now building next-generation experiments scheduled to begin operation at CERN's Large Hadron Collider (LHC) in 2007, along with the other LHC Collaborations, is facing unprecedented challenges in managing, processing and analyzing massive data volumes, rising from the Petabyte (10^{15} Bytes) to the Exabyte (10^{18} Bytes) scale over the coming decade. The current generation of experiments now in operation and taking data at SLAC (BaBar) and Fermilab (D0 and CDF) face similar challenges. BaBar has already accumulated nearly a Petabyte of stored data. Effective data sharing will require 10 Gbps of sustained throughput on the major HENP network links within the next 2 to 3 years, rising to the Terabit/sec range within the coming decade. In this talk, we have described a new TCP kernel that can make efficient use of the raw capacities in the future ultrascale network, enabled by the rapid advances in computing, storage and communication technologies.

We are working to further these preliminary results, to improve and evaluate the stability, responsiveness, fairness of the FAST kernel, and its interaction with the current protocols. We look forward to testing FAST in other high speed networks and grid facilities in the future, including Abilene, TeraGrid links and the National Light Rail network.

Acknowledgments: We gratefully acknowledge the support of the

- Cisco team, in particular, B. Aiken, V. Doraiswami, M. Potter, R. Sepulveda, M. Turzanski, D. Walsten and S. Yip
- Level(3) team, in particular, P. Fernes and R. Struble
- StarLight team, in particular, T. deFanti and L. Winkler
- CERN team, in particular, O. Martin and P. Moroni
- DataTAG team, in particular, E. Martelli and J. P. Martin-Flatin
- SCinet team, in particular, G. Goddard and J. Patton
- TeraGrid team, in particular, L. Winkler
- SLAC team, in particular, C. Granieri C. Logg, I. Mei, W. Matthews, R. Mount, J. Navratil and J. Williams
- Caltech-UCLA team, in particular, C. Chapman, C. Hu (Williams/Caltech), J. Pool, J. Wang and Z. Wang (UCLA)

and the funding support of NSF, DoE, ARO, and the Caltech Lee Center for Advanced Networking.

References

[1] V. Jacobson. Congestion avoidance and control. *Proceedings of SIGCOMM'88, ACM*, August 1988. An updated version is available via `ftp://ftp.ee.lbl.gov/papers/congavoid.ps.Z`.

[2] Lawrence S. Brakmo and Larry L. Peterson. TCP Vegas: end-to-end congestion avoidance on a global Internet. *IEEE Journal on Selected Areas in Communications*, 13(8):1465–80, October 1995. `http://cs.princeton.edu/nsg/papers/jsac-vegas.ps`.

[3] Jeonghoon Mo and Jean Walrand. Fair end-to-end window-based congestion control. *IEEE/ACM Transactions on Networking*, 8(5):556–567, October 2000.

[4] Steven H. Low, Larry Peterson, and Limin Wang. Understanding Vegas: a duality model. *J. of ACM*, 49(2):207–235, March 2002. `http://netlab.caltech.edu`.

[5] Steven H. Low and David E. Lapsley. Optimization flow control, I: basic algorithm and convergence. *IEEE/ACM Transactions on Networking*, 7(6):861–874, December 1999. `http://netlab.caltech.edu`.

[6] Steven H. Low. A duality model of TCP and queue management algorithms. *IEEE/ACM Trans. on Networking, to appear*, October 2003. `http://netlab.caltech.edu`.

[7] Chris Hollot, Vishal Misra, Don Towsley, and Wei-Bo Gong. A control theoretic analysis of RED. In *Proceedings of IEEE Infocom*, April 2001. `http://www-net.cs.umass.edu/papers/papers.html`.

[8] S. H. Low, F. Paganini, J. Wang, S. A. Adlakha, and J. C. Doyle. Dynamics of TCP/RED and a scalable control. In *Proc. of IEEE Infocom*, June 2002. `http://netlab.caltech.edu`.

[9] Fernando Paganini, John C. Doyle, and Steven H. Low. Scalable laws for stable network congestion control. In *Proceedings of Conference on Decision and Control*, December 2001. `http://www.ee.ucla.edu/~paganini`.

[10] Hyojeong Choe and Steven H. Low. Stabilized Vegas. In *Proc. of IEEE Infocom*, April 2003. `http://netlab.caltech.edu`.

[11] Fernando Paganini, Zhikui Wang, Steven H. Low, and John C. Doyle. A new TCP/AQM for stability and performance in fast networks. In *Proc. of IEEE Infocom*, April 2003. `http://netlab.caltech.edu`.