Modeling TCP-Vegas under On/Off Traffic *

Adam Wierman
Computer Science Department
Carnegie Mellon University
acw@cs.cmu.edu

Takayuki Osogami
Computer Science Department
Carnegie Mellon University
osogami@cs.cmu.edu

Jörgen Olsén
Department of Mathematics
Uppsala University, Sweden
jorgen@math.uu.se

1. INTRODUCTION

There has been a significant amount of research toward modeling variants of the Transmission Control Protocol (TCP) in order to understand the impact of this protocol on file transmission times and network utilization. Analytical models have emerged as a way to reduce the time required for evaluation when compared with more traditional evaluations performed using event driven simulators such as *ns*. In addition, when designed carefully, analytical models help researchers make design decisions about novel TCP mechanisms.

A wide variety of techniques have been applied to the problem of TCP modeling with a fair amount of success (see [16] for a detailed discussion). These techniques range over renewal theory [10, 17], Markov chains [11], and fluid models [1]. However, until the recent seminal work of papers using fixed-point methods [3, 4, 5, 6, 7, 8, 13, 14, 18], no modeling technique was able to mimic both the structure of a TCP source and the interaction a source has with the network. The fixed-point methods take the novel approach of separating the modeling of network behavior from the modeling of the behavior within a TCP source, and then allowing the two to tune each other via feedback.

The fixed-point framework allows general network topologies to be analyzed, and issues such as the interpretation of end-to-end loss rates in multiple bottleneck networks are addressed in [3, 7, 8, 14, 18].

In this paper, we generalize the framework based on a fixed-point method introduced by Casetti and Meo in [4, 5] in order to allow us to model TCP-Vegas connections. The framework of Casetti and Meo, which uses a Markov chain to model the TCP source, has a few advantages over other fixed-point methods: (1) it can model explicit details of TCP, making it possible to distinguish different flavors of TCP; (2) it allows modeling on-off traffic sources; (3) it gives the fraction of time that TCP spends in each state, from which we can evaluate the effectiveness of each mechanism of the protocol.

A majority of existing analytical models focus on TCP-Reno, the most widely deployed variant of TCP, and there has been little research on analytical models of TCP-Vegas, a more recently proposed variant. Analytical models of TCP-Vegas have been difficult to develop because TCP-Vegas uses observed delay to detect an incipient stage of congestion and try to adjust the sending rate before packets are lost. Thus, unlike TCP-Reno, TCP-Vegas attempts to determine the correct sending rate without relying on packet losses.

Prior studies on measurement and simulation of the performance of TCP-Vegas suggest that in many situations it is able to provide users higher throughput and lower loss rates than TCP-Reno. Hence, it is an important task to model the performance of TCP-Vegas in order to understand how this protocol performs in a network shared with other variants of TCP.

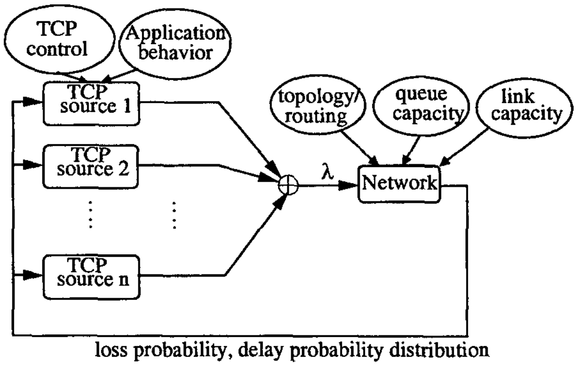


Figure 1: TCP analysis methodology

Because TCP-Vegas uses observed delay, as well as loss events, to adjust the congestion window size, while TCP-Reno uses only loss events, the extension made in this work to the framework of Casetti and Meo for modeling TCP-Reno is nontrivial. We need to make two major changes. First, analyzing the network model to determine the loss rate and the mean queueing delay is no longer enough; the full distribution of queueing delay must be obtained. Second, the Markov chain used to model a TCP-Reno source must be extended to use information from the delay distribution.

The model that results from these extensions represents a leap in our ability to model TCP-Vegas sources. In particular two major limitations of other TCP models are overcome in our model:

1. Our model is the first model of TCP-Vegas that accurately predicts the operating point of the network; i.e., it predicts both the throughput and the loss rate of TCP sources. In particular, all but one previous work assumes loss-free operation [1, 2, 9, 12, 15]. Samios and Vernon [19] proposed the first analytical model to incorporate loss rate. They give a closed-form formula for the throughput achieved by a single TCP-Vegas sender as a function of the loss rate and the round trip time (RTT) of the network. However, they are not able to predict the full operating point of the network.

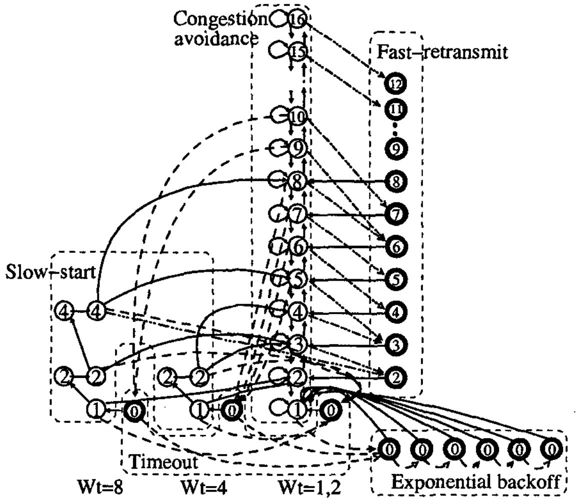


Figure 2: **A continuous time Markov chain of TCP-Vegas for a maximum window size of 16, where states represents the window size and the phase of transmission: slow-start, congestion avoidance, fast-transmit, timeout, and exponential backoff. W_t denotes the slow-start threshold value.**

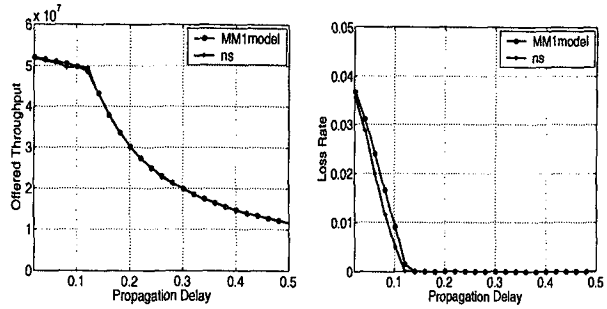
2. Our model is the first model of TCP-Vegas for arbitrary on-off traffic; all previous work on modeling TCP-Vegas is based on the analysis of TCP-Vegas for bulk transfers [1, 2, 9, 12, 15, 19]. Bulk transfer models are based on the assumption that a large amount of data is sent along a single connection. On-off traffic provides a more general model, including bulk transfer as a special case (when the on periods are long). On-off traffic applies both to FTP connections, the case of bulk transfer, and HTTP traffic, which inherently is made up of many short connections.

We validate our model under a single bottleneck network topology against *ns* simulations. It is also possible to extend the network modeling to more general network topologies. We find that our model is accurate in a wide range of situations [20].

2. MODEL OVERVIEW

Our framework consists of two components: the TCP sources and the network (Figure 1). The throughput of each source is independently analyzed via a Markov chain, while the loss rate and the probability distribution of the delay of the network is analyzed separately using queueing models. For the Tahoe and Reno models in [4, 5], only the loss rate is analyzed, but we require the delay distribution for modeling TCP-Vegas. Aspects such as network topology, queueing capacity, link capacity, packet arrival pattern, queue management scheme (DropTail, RED, class-based RED, etc.), and the network environment (fixed links or Wireless transmission) are taken into account within the queuing model.

Although the two components are analyzed separately, the parameters in the source models and the network model are tuned by iterative analysis with feedback from each other. That is, the source models are analyzed to obtain an aggregate traffic, which is used as an arrival process of the network model. The network model, which for this work is an $M/M/1/B$ queue, is analyzed to obtain the loss rate and delay probability distribution, which are used as the parameters in source models. Notice that a TCP source adjusts its sending rate (its congestion window size) based on the observation of events such as packet loss and delay of acknowledgments. The source models are analyzed again, and the procedure is continued until a stable solution is obtained. Given accurate source and network models, the fixed point of this iteration matches the operating point of the true network. We assume that the sources are independent. Therefore, when the sources are homogeneous, it suffices to analyze only one Markov chain; when the sources are heterogeneous, the number of Markov chains to be analyzed is the number of different types of sources.



(a) throughput

(b) loss rate

Figure 3: Comparison of *ns* and Markov model for TCP-Vegas with respect to (a) offered traffic and (b) loss rate. The network is set up with 100 TCP sources each with on-off traffic (On/Off = 50/5) sharing a 50 Mbps core network (the maximum window size $W = 16$ and the buffer size $B = 100$).

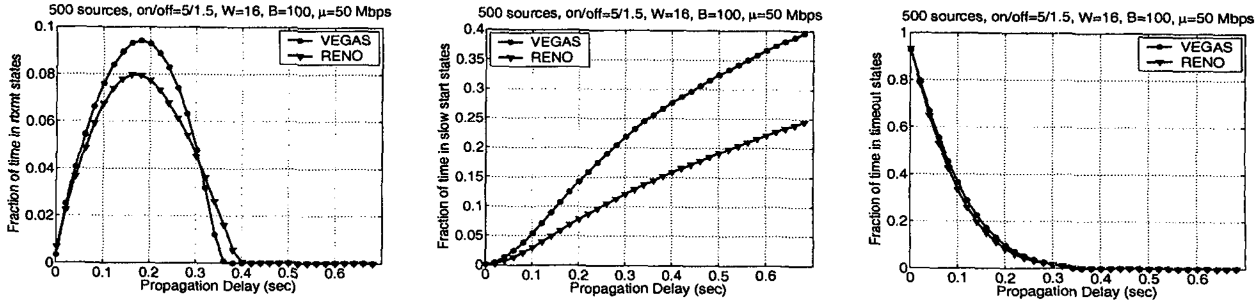
Mimicking the structure of real-world network hosts, our model of an individual source is composed of two levels: an application level and a transport level. The application level alternates between a busy state and an idle state. While the application level is in a busy state, the transport level moves among TCP states, changing the window size and the phase of transmission (such as slow-start, congestion avoidance, fast-retransmit, and timeout).

A continuous time Markov chain for TCP-Vegas is shown in Figure 2. The states keep track of the congestion window size, as well as other information such as the slow-start threshold and whether we need to recover from loss or not. States with thin borders correspond to states where no loss event has occurred: slow-start states and congestion avoidance states. States with thick borders correspond to states where loss has occurred but has not yet been recovered from: fast retransmit states, timeout states, and exponential backoff states. The transition rate is determined by the loss rate and the distribution of the RTT, which are provided by the network model. By solving this Markov chain, we can determine the limiting probabilities of the congestion window size, which determines the throughput of the sender. See [20] for more details.

3. MODEL ANALYSIS

Using the model described in the previous section, it is possible to predict two types of information about how TCP flavors perform. First, the model predicts external performance metrics such as mean throughput and mean loss rate. Further, the model predicts internal performance metrics such as how much time is spent performing different TCP mechanisms.

To validate the accuracy of the model predictions about external metrics we include plots for a small set of validations in Figure 3. We validate our model using a network with a bottleneck topology, where the bottleneck link is fed by N independent on-off sources.



(a) fast retransmit (b) slow-start (c) timeout

Figure 4: Comparison of time spent TCP states under TCP-Reno and TCP-Vegas.

The sources are connected to the bottleneck link via separate access links. A range of propagation delays, capacities, and buffer sizes are tested for the bottleneck link. The plots in Figure 3 represent a small subset of the network configurations the model has been tested on. For a more detailed validation of the model the interested reader should refer to [20].

Apart from accurate prediction of the performance of various TCP flavors, an advantage of our framework is that it allows us to investigate such information as the fraction of time TCP spends in each state: slow-start, congestion avoidance, backoff, fast retransmit, and timeout. Since the states in our model correspond exactly to the states of TCP, we can draw conclusions about the effectiveness of the new mechanisms introduced in TCP-Vegas, using the fraction of time spent in each state.

Figure 4 shows the fraction of time spent in fast retransmit, timeout, and slow-start states under each flavor of TCP with a bottleneck link having a 50 Mbps capacity and a buffer size of 100 packets shared by 500 sources. The important point illustrated by these plots is that the modified slow-start mechanism in TCP-Vegas does not help avoid future losses or timeouts, but does result in spending more time in the slow-start phase. Comparing the performance of TCP-Vegas and TCP-Reno, we can observe that the time TCP-Vegas spends in slow-start states is much greater than the time TCP-Reno spends in these states (Figure 4 (b)), and that the time TCP-Vegas spends in timeout states is almost the same as the time TCP-Reno spends in these states (Figure 4 (c)).

4. REFERENCES

[1] T. Bonald. Comparison of TCP Reno and TCP Vegas via fluid approximation. Technical Report RR-3563, INRIA, 1998.

[2] C. Boutremans and J. L. Boudec. A note on the fairness of TCP Vegas. In *Proc. of* International Zurich Seminar on Broadband Communications, pages 163–170, Feb. 2000.

[3] T. Bu and D. Towsley. Fixed point approximations for TCP behavior in an AQM network. In *Proc. of* ACM Sigmetrics, pages 216–225, Jun. 2001.

[4] C. Casetti and M. Meo. A new approach to model the stationary behavior of TCP connections. In *Proc. of* IEEE INFOCOM, pages 367–375, March 2000.

[5] C. Casetti and M. Meo. An analytical framework for the performance evaluation of TCP Reno connections. *Computer Networks*, 37:669–682, 2001.

[6] V. Firoiu and M. Borden. A study of active queue management for congestion control. In *Proc. of* IEEE INFOCOM, pages 1435–1444, Mar. 2000.

[7] V. Firoiu, I. Yeom, and X. Zhang. A framework for practical performance evaluation and traffic engineering in IP networks. In *Proc. of* IEEE ICT, Jun. 2001.

[8] R. Gibbens, S. Sargood, C. V. Eijl, F. Kelly, H. Azmoodeh, R. Macfadyen, and N. Macfadyen. Fixed-point models for the end-to-end performance analysis of IP networks. In *13th ITC Special Seminar*, Sep. 2000.

[9] G. Hasegawa, M. Murata, and H. Miyahara. Fairness and stability of congestion control mechanisms of TCP. In *Proc. of* IEEE INFOCOM, pages 1329–1336, Mar. 1999.

[10] I. Kaj and J. Olsén. Stochastic equilibrium modeling of the TCP dynamics in various AQM environments. In *Proc. of* SPECTS, pages 481–493, Jul. 2002.

[11] A. Kumar. Comparative performance analysis of versions of TCP in a local network with a lossy link. *IEEE/ACM Trans./ Networking*, 6:485–498, 1998.

[12] S. Low, L. Peterson, and L. Wang. Understanding TCP Vegas: A duality model. In *Proc. of* ACM Sigmetrics, pages 226–235, Jun. 2001.

[13] A. Misra, T. Ott, and J. Baras. The window distribution of multiple TCPs with random queues. In *Proc. of* IEEE GLOBECOM, pages 1714–1726, Dec. 1999.

[14] V. Misra, W.-B. Gong, and D. F. Towsley. Fluid-based analysis of a network of AQM routers supporting TCP flows with an application to RED. *ACM SIGCOMM Computer Communication Review*, 30:151–160, 2000.

[15] J. Mo, R. La, V. Anantharam, and J. Walrand. Analysis and comparison of TCP Reno and Vegas. In *Proc. of* IEEE INFOCOM, pages 1556–1563, Mar 1999.

[16] J. Olsén. *Stochastic Modeling and Simulation of the TCP Protocol.* PhD thesis, Department of Mathematics, Uppsala University, Sweden, Oct. 2003.

[17] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose. Modeling TCP throughput: a simple model and its empirical validation. *ACM Computer Communication Review*, 28:303–314, 1998.

[18] M. Roughan, A. Erramilli, and D. Veitch. Network performance for TCP networks part I: Persistent sources. In *Proc. of* the International Teletraffic Congress-ITC-17, pages 24–28, Sep. 2001.

[19] C. Samios and M. Vernon. Modeling the throughput of TCP Vegas. In *Proc. of* ACM Sigmetrics, pages 71–81, Jun. 2003.

[20] A. Wierman, T. Osogami, and J. Olsén. A unified framework for modeling TCP-Vegas, TCP-SACK, and TCP-Reno. In *Proc. of International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems*, Oct. 2003.