In IEEE International Conference on Telecommunications 2001

A Framework for Practical Performance Evaluation and Traffic Engineering in IP Networks

Victor Firoiu
Advanced Technology
Nortel Networks
600 Tech Park
Billerica, MA 01821 USA
vfiroiu@nortelnetworks.com

Ikjun Yeom *
Center for Next Generation
Information Technology
Kyungpook National University
1370, Sankyukdong, Buk-ku,
Taegu, 702-701, South Korea
ikjun@ee.knu.ac.kr

Xiaohui Zhang
Advanced Technology
Nortel Networks
600 Tech Park
Billerica, MA 01821 USA
xiaohui@nortelnetworks.com

Abstract

We propose a model for performance evaluation (average throughput, packet delay and loss probability) of IP networks that are dominated by congestion-controlled (i.e., TCP-like) traffic. Our model includes heterogeneous TCP flows, UDP flows, short-lived TCP flows, and TCP flows in a differentiated services network. We also introduce a practical approach to estimating the performance of large scale networks where flows may encounter multiple congested links. Through extensive simulations, we find our models to be accurate and efficient to compute.

The model presented here is one of the first attempts[1] to providing a network calculus for congestion-controlled traffic. This quantitative network analysis is an essential part of traffic engineering and network provisioning for IP networks with Best Effort and/or Differentiated Services traffic.

I Introduction

Current Internet suffers from congestion due to increasing traffic. To solve the congestion, more resources are being deployed. *Traffic engineering* aims at solving the congestion through efficient traffic distribution [2]. Traffic engineering is a process to avoid uneven traffic distribution and improve overall network performance. In traffic engineering, an analytic model for network behaviors is essential for network dimensioning, network provisioning, and admission control. In this paper, we provide a framework for computing the average delay and loss rate per end-to-end TCP and UDP flow and at each network element, given a characterization of active flows, routing, and link attributes.

Congestion control becomes more important as the Internet grows. We can divide congestion control into two parts: (1) Congestion control at sources of traffic, and (2) Queue management in routers.

Congestion control at sources is usually performed by adjusting the sending rate according to the current available bandwidth. We say that a flow is *responsive* when the flow adjusts its own sending rate reacting to packet losses. The most widely used type of responsive flow is the TCP flow. In fact, TCP has sophisticated mechanisms for congestion control, such as slow start, additive increase multiplicative decrease, timeouts, duplicated ACKs [20],[21] and many other techniques are still being developed.

Queue management associated with drop policy is another part of congestion control. When queue build-up at a router exceeds a certain threshold, the router detects it as an indication of congestion and takes an action. The action is typically to drop packets. There are a lot of issues related to drop policy: congestion control, fairness among flows, and queuing delay. Many approaches to these issues have been proposed, including tail drop, random drop, early drop, random early detection (RED) [13], and recently RIO (RED In/Out) [8].

Congestion controls at sources and routers are highly correlated to one another: The drop policy at routers impacts the sending rate at sources, and sources' reaction to packet losses impacts queue build-up at routers. Changes to queue lengths result in changes to drop rates. These procedures are repeated until a system (network) reaches a steady state (if a steady state exists). What the steady state is and how fast the system reaches the steady state are determined by both (1) and (2). Congestion control, therefore, should be examined and evaluated as an interaction between (1) and (2). However, complicated mechanisms in (1) and (2) make it difficult to evaluate this interaction.

In this paper, we look at this interaction focusing mainly on TCP flows and RED routers given that TCP flows comprise about than 90% of recently measured Internet traffic [15], and that RED is increasingly used in the current Internet. Our approach is to start from the performance model of TCP and

*This work was done when Ikjun Yeom was working as a summer intern at Nortel Networks.

[1] We have recently learned about an independent and simultaneous work [5]

RED presented in [12]. We first extend it to heterogeneous traffic including TCP flows with different RTTs, maximum window sizes, different packet sizes, heterogeneity of flows (TCP and UDP), and short lived flows. We then extend it to the differentiated services Assured Forwarding Per-Hop Behavior [14]. Next, we further extend this model to large-scale networks where flows may encounter multiple points of congestion. Our resulting models are supported by analytical expressions; we provide effective methods for solving them numerically.

The main contributions of this paper include: (1) a generalized model for performance evaluation in networks with single point of congestion; (2) a number of simulations in heterogeneous networks to validate the model; (3) a general model for average throughput, delay and loss for networks with multiple points of congestion; (4) an efficient method for solving the network model numerically; and (5) a set of simulation experiments validating the network model and its numerical solution.

We believe these contributions play an important role in traffic engineering, especially in the following areas: (1) admission control in DiffServ, (2) traffic engineering of DS and/or Best Effort by computing what-if scenarios when establishing/changing routes for flows, (3) network dimensioning, provisioning; compute impact on delay, loss when increase link capacity, and (4) choosing RED parameters given all network conditions.

The rest of the paper is organized as follows: In Section II we propose a model for performance evaluation in single bottleneck networks and validate our model through a number of simulations. In Section III we propose a numerical approach for network-wide performance evaluation using our model. We apply the approach to large scale networks and present the results. In Section IV, we introduce some useful applications using the work here and discuss limitations. In Section V, we present related work. We conclude the paper in Section VI.

II A model for single bottleneck networks

In [12], we have proposed a model for TCP traffic performance over a network with a single point of congestion and with queue management implementing the RED algorithm [13]. In that model, the average queue length and link utilization are predicted as a function of drop probability over a link wherein all TCP flows have the same characteristics (e.g. packet size, round trip time (RTT), and upper limit of advertised window size). In this section, we extend the model to arbitrary heterogeneous traffic. We assume that the reader is familiar with the TCP Reno congestion control mechanism as presented in [20].

A *A model for heterogeneous traffic*

We consider that n_t TCP flows (f_i, $1 \le i \le n_t$) and n_u UDP CBR (Constant Bit Rate) flows (u_j, $1 \le j \le n_u$) share a single bottleneck link with capacity c. We start from the TCP throughput model proposed in [19] and summarized below in eq. (1) to (4). In steady state, throughput of each i^{th} individual TCP flow is approximated by $t_i(p, R_i) =$

$$\begin{cases} \dfrac{M_i(\frac{1-p}{p}+\frac{W(p)}{2}+Q(p,W(p))}{R_i(\frac{b}{2}W(p)+1)+\frac{Q(p,W(p))F(p)T_0}{1-p}} & \text{if } W(p) < W_i \\ \dfrac{M_i(\frac{1-p}{p}+\frac{W_i}{2}+Q(p,W_i)}{R_i(\frac{b}{8}W_i+\frac{1-p}{pW_i}+2)+\frac{Q(p,W_i)F(p)T_0}{1-p}} & \text{otherwise} \end{cases} \tag{1}$$

where p is the drop probability of the link, R_i and M_i are the RTT and the packet size of i^{th} TCP flow, W_i is the maximum window size limited by sender's or receiver's buffer size, and T_0 is the latency of the first timeout without backoff (typical value $5 * R_i$). $W(p)$ is the window size at the time of loss events and given by

$$W(p) = \frac{2+b}{3b} + \sqrt{\frac{8(1-p)}{3bp} + (\frac{2+b}{3b})^2} \tag{2}$$

where b is the number of packets acknowledged by one ACK, usually equal to 2. $Q(p,w)$ is the probability that a sender detects packet loss by timeout and expressed by

$$Q(p,w) = \min\{1, \frac{(1+(1-p)^3(1-(1-p)^{w-3}))}{(1-(1-p)^w)/(1-(1-p)^3)}\} \tag{3}$$

$F(p)$ is the expected number of doubling of T_0 by backoff and given by

$$F(p) = 1 + p + 2p^2 + 4p^3 + 8p^4 + 16p^5 + 32p^6 \tag{4}$$

The throughput of a CBR flow, r_j, is simply given by

$$r_j(p) = (1-p)\lambda_j \tag{5}$$

where λ_j is the sending rate of the j^{th} CBR flow. Then, the total throughput T of the link is

$$T(p, \{R_i\}) = \sum_{i=1}^{n_t} t_i(p, R_i) + \sum_{j=1}^{n_u} r_j(p) \tag{6}$$

Assuming that p is independent from the queue length[2], R_i is given by

$$R_i = R_{0,i} + \overline{q}/c \tag{7}$$

where $R_{0,i}$ is the sum of propagation delay experienced by the i^{th} TCP flow, and $\overline{q}$ is the average queue length of the link. From the assumption that all the flows pass through one bottleneck link, $\overline{q}$ is the same for all the flows, and we can reduce $T(p, \{R_i\})$ to $T(p, \overline{q})$ at given $\{R_{0,i}\}$ and c without loss of generality.

[2] In most drop policies, p is the function of queue length. This assumption is to examine the relation between queue length and p. In Section III, we present a general form.

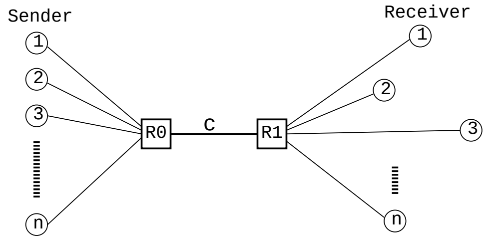


Figure 1: Simulation topology

Similar to [12], we can determine p_0 at which

$$T(p_0, \overline{q} = 0) = c \tag{8}$$

We can see from (1), (5) and (6) that $T(p, \overline{q})$ is monotonic in p increases for $0 < p \leq 1$, and thus there exists a unique p_0 solution for (8). We denote $p_0 = T_p^{-1}(c, 0)$ where $T_p^{-1}(c, \overline{q})$ is the inverse of $T(p, \overline{q})$ in p. $T_p^{-1}(c, \overline{q})$ does not have a closed form, but can be calculated numerically from (1)-(7) (e.g., using Newton-Raphson's method).

When $p > p_0$, the link is underutilized, and $\overline{q} = 0$. Then the utilization $u(p)$ is given by

$$u(p) = \frac{T(p, 0)}{c} \tag{9}$$

When $p \leq p_0$, the link is fully utilized ($u(p) = 1$), and $\overline{q}$ is given by

$$\overline{q} = T_{\overline{q}}^{-1}(p, c) \tag{10}$$

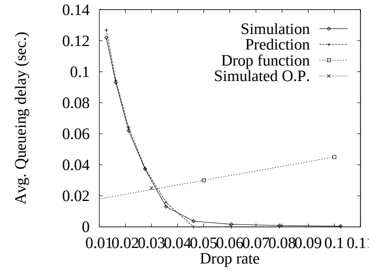
where $T_{\overline{q}}^{-1}(p, c)$ is the inverse of $T(p, \overline{q})$ in $\overline{q}$. $T_{\overline{q}}^{-1}(p, c)$ does not have a closed form, but can be computed numerically from (1)-(7).

B *Validation through simulation*

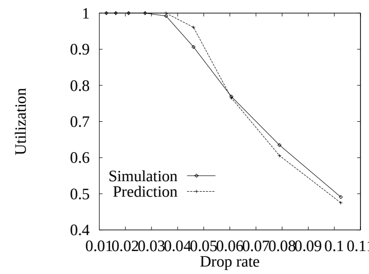
In this section, we present a number of simulation results that verify our models. Fig. 1 shows the network topology. The capacity c of the bottleneck link ($R0 - R1$) is 9 Mbps, while all the other links have larger bandwidth. We use ns-2 [18] with a fixed drop rate module. For each simulation setup, we conduct several experiments with different drop probabilities. In each experiment, we measure average queueing delay ($\overline{q}/c$) and utilization at the bottleneck link. We also conduct an experiment using RED drop function with parameters {`50/150/0.1`}[3] and present the average operating point.

B.1 Heterogeneous TCP flows

In this simulation, we use 120 heterogeneous TCP flows. Each flow is randomized with respect to the following parameters: (1) $R_{0,i}$, RTT except queueing delay, is randomly selected from 0.06 to 0.14 seconds; (2) the packet size for each flow is randomly selected from the set $\{2^5, 2^6, 2^7, 2^8, 2^9, 2^{10}\}$ bytes; (3) W_i, the maximum TCP window size, is selected from 6 to 50 packets.



(a) Avg. queueing delay



(b) Avg. link utilization

Figure 2: Heterogeneous TCP flows

Fig. 2 shows the results. It is observed that our model is quite accurate in both queueing delay and utilization, and this confirms the results presented in [12], this time with heterogeneous TCP flows.

B.2 TCP and UDP flows

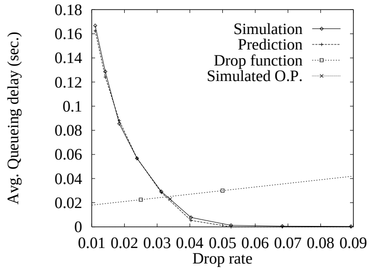
In this simulation, we use both TCP and UDP flows. TCP flows are the same as in the previous section. UDP traffic is constituted by a number of constant bit rate (CBR) flows with 0.1 Mbps sending rate, for a total of 30of each UDP flow is also randomly selected from the set $\{2^5, 2^6, 2^7, 2^8, 2^9, 2^{10}\}$ byte. Fig. 3 presents simulation results showing that our model has the same good level of accuracy as the TCP-only model.

B.3 Short-lived TCP flows

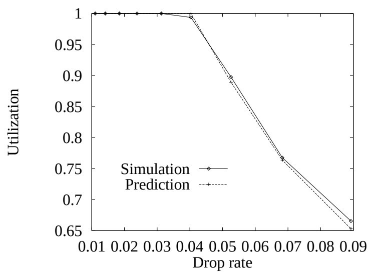
So far we assumed that each flow has infinite data to be transferred. However, as WWW with HTTP are extremely popular, the TCP flows become short. Recent studies have shown that mean size of transferred data by each TCP flow is around 10KB [9],[7]. To validate our model with more realistic traffic, we consider short-lived TCP flows as well.

We apply the throughput model for a short-lived TCP flow proposed in [6] to $t_i(p, R_i)$ instead of (1). In [6], TCP throughput is a function of D, the amount of data to be transferred, in addition to the

[3] In this paper, we use the notation {`50/150/0.1`} as short for $q_{min} = 50, q_{max} = 150, p_{max} = 0.1$.

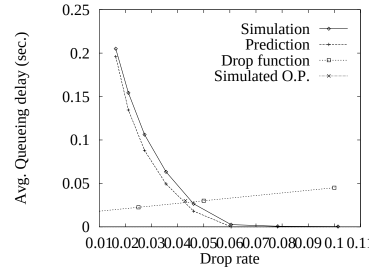


(a) Avg. queueing delay with 30% CBR traffic

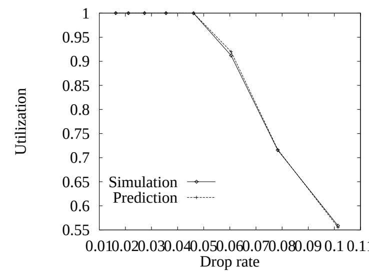


(b) Avg. link utilization with 30% CBR traffic

Figure 3: Traffic Heterogeneity



(a) Avg. queueing delay when considering short-lived flows



(b) Utilization when considering short-lived flows

Figure 4: Heterogeneity of flows' file size

drop probability and RTT.

$$t_i(D_i, p, R_i) = \frac{D_i}{T_{ss} + T_{loss} + T_{ca} + T_{delack}} \quad (11)$$

where T_{ss} is the time spent in initial slow start, T_{loss} is the time caused by the first packet loss followed by the initial slow start, T_{ca} is the time to send the remaining data, and T_{delack} is the TCP latency caused by delayed ACKs[4].

To generate short-lived TCP flows, each flow transmits a random number of packets between 50 to 200 packets. When a flow completes sending the given number of packets, it closes its connection and opens a new connection immediately so that the total number of running flows is maintained. In this simulation the number of concurrent TCP flows is 120.

Fig. 4 shows the result. It is observed that queueing delay in Fig. 4(a) is relatively high compared to Fig. 2(a) at the same drop rate. It is because a TCP sender increases its window size exponentially[5] in slow start period until the first packet loss, and this behavior makes traffic burstier. We also observe that our prediction is quite close to the simulation results.

[4]We do not present the detail model in this paper. Please refer to the original paper [6].

[5]In slow start period, the window size increases by one packet at every ACK arrival. Thus, the window size is doubled at every RTT.

B.4 Differentiated services

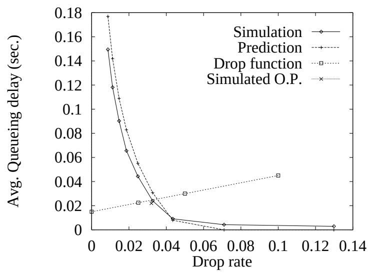
Differentiated services (diff-serv) have been introduced to provide different level of services to different users [17],[3]. The Assured Forwarding Per-Hop Behavior (AF PHB) provides different drop rates to packets according to their subscribed level of service [14].

In this section, we apply our model to AF PHB with two drop precedences. When the packet of a flow enters a diff-serv domain, an edge device marks the packet IN if the flow's current sending rate is within its contract rate. Otherwise, the packet is marked as OUT. In the domain, an OUT packet is discarded first when congestion occurs. The forwarding of IN packet is meant to be protected until every OUT packet is dropped.

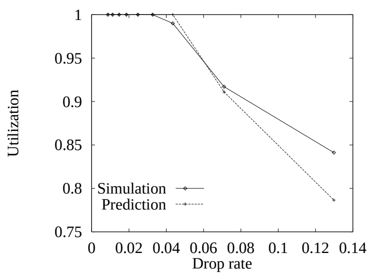
A TCP throughput model in a diff-serv network has been proposed in [22]. In this model, TCP throughput $t_i(C_i, p_{out}, p_{in}, R_i)$ is defined as a function of its contract rate (C_i), drop rates of IN and OUT packets (p_{out} and p_{in}), and RTT (R_i). We assume that $p_{in} = 0$ for simplicity[6] and thus the model is reduced to $t_i(C_i, p_{out}, R_i)$. We combine this model for TCP throughput with our model framework (6),(7),(9),(10) and obtain a model of TCP in a diff-serv network.

In our simulation, there are 30 TCP flows with randomly selected contract rates (C_i). The range of $\{C_i\}_i$ in the range 0.01 to 0.5 Mbps, and $\sum_{i=1}^{30} C_i =$

[6]In a well configured diff-serv domain, IN packets are expected not to be dropped.



(a) Avg. queueing delay



(b) Utilization

Figure 5: TCP flows in AF PHB

$7.5Mbps$, which is less than c (9 Mbps), so that $t_i > C_i$ and some packets will be marked as OUT. Fig. 5 shows the simulation results that confirm the accuracy of our model for TCP in an AF PHB network.

III An approach to the performance evaluation of large scale networks

In the previous section, we presented a model for a network with heterogeneous traffic and a single congested link. In this section, we extend this model to large scale networks with possible multiple congested links.

A The model

We consider the following network calculus problem.
Given

- A network with a set of links L, its topology and routing;
- For each link $k \in L$, its capacity c_k, its propagation delay R_0^k and its congestion control function

$$p_k = H_k(q_k) \tag{12}$$

where p_k is the drop probability and q_k is the average queue size at link k. For example, if RED control is used [13], $H_k(q_k)$ has the form

$$\begin{cases} 0 & q_k < q_{min}^k \\ \frac{p_{max}^k}{q_{max}^k - q_{min}^k}(q_k - q_{min}^k) & q_{min}^k \le q_k \le q_{max}^k \\ 1 & q_{max}^k < q_k \end{cases} \tag{13}$$

where q_{min}^k, q_{max}^k and p_{max}^k are configurable RED parameters;

- A set of TCP flows $f \in F$ with their characteristics: packet size M_f and maximum window size W_f;
- A set of UDP flows $f \in U$ and their sending rate λ_f.

Find

- The steady-state average queue size $\overline{q}_k$ and drop probability p_k for each link k;
- The average RTT R_f, end-to-end drop probability p_f and sending rate t_f for every TCP flow f.

While

- maximizing the utilization of all links.

We also make the following notation: P_f and RP_f are the set of links traversed by flow f on its forward path and round trip respectively, and $P_f^k \subset P_f$ is the set of links on f's forward path up to and including link k. F_k is the set of TCP flows traversing link k in their forward direction (i.e., transiting TCP data packets and not ACK packets). U_k is the set of UDP flows traversing link k. P_f, RP_f, P_f^k, F_k and U_k can be derived from the given network topology and routing.

To solve this problem, we have that, for a network in steady-state, there is an equilibrium between the TCP congestion control algorithm expressed in eq (1)-(4), the UDP throughput reduced by packet drops expressed by (5), and the congestion control algorithm at each link H_k such as the one in (13), i.e., eq (1)-(4), (5) and (12) should simultaneously hold throughput the network.

More precisely, we have the following. The average round trip time for TCP flow f is

$$R_f = \sum_{k \in RP_f} (R_0^k + \overline{q_k}) \tag{14}$$

The average drop probability experienced by TCP flow f on its forward path is

$$p_f = 1 - \prod_{k \in P_f} (1 - H_k(\overline{q}_k)) \tag{15}$$

The average drop probability experienced by flow f up to link k is

$$p_f^k = 1 - \prod_{k \in P_f^k} (1 - H_k(\overline{q}_k)) \tag{16}$$

The total throughput at link k is

$$T_k = \sum_{f\in F_k} t_f(p_f, R_f)(1-p_f^k) + \sum_{f\in U_k} \lambda_f(1-p_f^k) \quad (17)$$

where t_f is given in eq (1)-(4). Observe that T_k depends on $\{\overline{q}_j\}_{j\in L}$ through p_f, p_f^k and R_f. Moreover, it is easy to see that T_k is monotonically decreasing in $\{\overline{q}_j\}_{j\in L}$. Here we have ignored the contribution of ACK packets to the total link throughput since their size is relatively small compared to data packets. We have also ignored the drop rate of ACK packets (drop rate on TCP's return path), since acknowledgements are cummulative (confirm reception of all previous packets) and thus the loss of an ACK has only a small effect on TCP throughput. Finally, observe that in (17), both TCP and UDP sending rates t_f and λ_f are reduced at link k by a cummulative drop probability p_f^k.

Our initial problem is now reduced to

> ***Problem 1.*** *Find $\overline{q}_k$ for all links $k \in L$ such that*
>
> $$T_k \le c_k \quad \forall k \in L \quad (18)$$
>
> *and t_f are maximized for all $f \in F$.*

The first conditions says that traffic cannot surpass link's capacity, while the maximization of all TCP sending rates follows from TCP's congestion control algorithm that increases the sending rate until a drop occurs.

To solve this problem, we propose the following numerical, iterative method.

1. **Initial step.** For each link k, find an initial approximation of average queue size $\{\overline{q}_{k,0}\}$ by assuming that link k is the only congestion point for all the flows traversing it:

$$T_{k,0} = \sum_{f\in F_k} t_f(p_{f,0}, R_{f,0})(1-p_{f,0}^k) + \sum_{f\in U_k} \lambda_f(1-p_{f,0}^k) = c_k \quad (19)$$

where

$$p_{f,0} = p_{f,0}^k = H_k(\overline{q}_{k,0}) \quad (20)$$

$$R_{f,0} = \overline{q}_{k,0} + \sum_{k\in RP_f} R_0^k \quad (21)$$

This problem can be solved as presented in Section II. Observe that the result of this initial step provides a solution satisfying condition (18), since the existence of multiple congested links on a flow's path can only reduce its throughput compared to the throughput derived from the one-congested link condition in (19). But this result does not necessarily give the maximum TCP flow throughput as required by our problem statement above.

2. **Iteration step n+1.** For each link k, adjust the queue size

$$\overline{q}_{k,n+1} = \frac{T_{k,n}}{c_k}\overline{q}_{k,n} \quad (22)$$

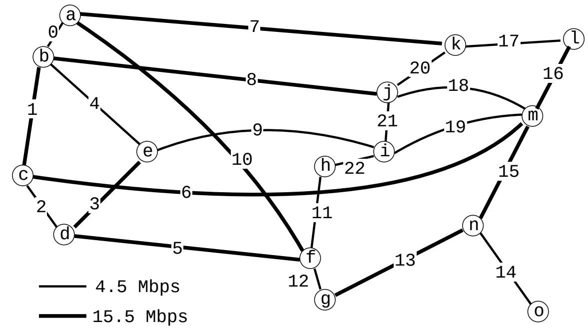


Figure 6: A large scale network topology

Compute new link probability values

$$p_{k,n+1} = H_k(\overline{q}_{k,n+1}) \quad (23)$$

For all flows f, compute $R_{f,n+1}$, $p_{f,n+1}$ and $p_{f,n+1}^k$ from eq (14), (15) and (16). Using these values, compute new values for $T_{k,n+1}$ from (17).

Observe that if $T_{k,n} < c_k$, then $\overline{q}_{k,n+1}$ is decreased below $\overline{q}_{k,n}$, $p_{f,n+1}$ are decreased, $t_{f,n+1}$ are increased and thus $T_{k,n+1}$ is increased. The reciprocal is also true, and thus a repeated application of eq (22) has the effect of bringing $T_{k,i}$ as close as possible to c_k, which is the goal of Problem 1.

3. **Stop condition.** We end the iteration when

$$\sqrt{\sum_{k\in L}(\overline{q}_{k,n+1} - \overline{q}_{k,n})^2} < \delta\sqrt{\sum_{k\in L}\overline{q}_{k,n}^2} \quad (24)$$

where δ is a small constant that determines the accuracy and the running time of this numerical method. This condition says that the iteration can stop when changes to $\overline{q}_k$ values are below a small threshold, and thus the solution has converged to a set of steady-state values.

B *Simulation study*

To study the effectiveness of our model, we have conducted extensive simulation experiments. In the following, we present a set of simulations with the network shown in Fig. 6. There are 15 nodes and 23 links. Each link capacity is given by 15.5 or 4.5 Mbps as shown in the figure. RED parameter is {0/300/0.1} for 15.5 Mbps link. For 4.5 Mbps link, we set {0/500/0.1} so that 4.5 Mbps link can resolve the burst traffic from 15.5 Mbps without forcely drop[7]. Propagation delay for each link is set to 20 msec. for relative long links (e.g., link 6, 7, 8...) and 10 msec. for short links (e.g., link 0, 1, 2, 3...). In this topology, each node establishes a set of connections to all the other nodes. Each set consists of 9 TCP flows for a total of 1890 (= $15 \times 14 \times 9$) flows in the network.

[7] In this paper we use "forcely drop" to mean packet drop with $p = 1$ when $\overline{q} > q_{max}$.

In the simulation, W_i is randomly selected from 6 to 50 packets, and packet is 1 KB. We measure the throughput of each flows, $\overline{q}$ and p at both directions of each link.

We also compute the average queue size, drop probability and flow throughput predicted by our model. We present in Fig. 7(a) the simulated and predicted throughput of each flow set, which is the sum of 9 flows with the same source and destination. We see that our prediction of flows with relatively low throughput is quite accurate. For flows with high throughput, we see that prediction is higher than simulation but still fairly accurate. Fig. 7(b) shows the histogram of relative errors[8]. We see that most errors are less than 10%.

We also present drop rate and queueing delay in Fig. 7(c) and 7(d). In each figure, the x-axis shows simulated values, and the y-axis shows predicted values. Thus each point represents simulated and predicted values of a direction of each link. Total 26 points are presented in each figure. We do not specify the link corresponding to each point because it does not give any better understanding of the results. $x = y$ is the ideal case. It is shown that predictions of low drop rate (less than 0.015) in Fig. 7(c) and small delay (less than 0.1 sec.) are very close to simulated results. For higher drop rate and larger queueing delay, it is observed that prediction is slightly less than simulation.

IV Applications and limitations

So far, we have presented an approach for performance evaluation in large networks and verified through simulations. In this section, we present some examples of useful applications using this approach and discuss limitations of our approach.

A *Identification of congested links*

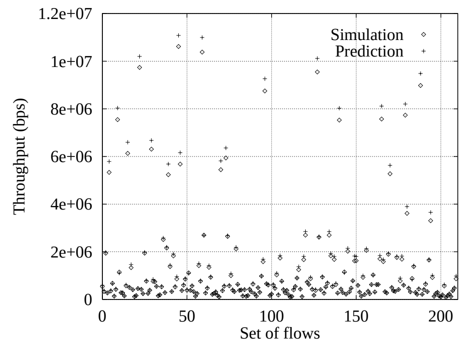
Usually we consider that a link is bottleneck when arrival rate of incoming packets is higher than service rate (departure rate), and the packets are enqueued. A TCP flow's throughput is typically limited by the out-most bottleneck link, or the most congested link on its path. Therefore, identifying bottleneck links in a given network is an important part of traffic engineering.

Average queueing delay and drop rate produced by a link is a good indication of level of congestion and can be used to identify if the link is a bottleneck in the network. Our work gives an accurate estimate of each link at a given network. Fig. 8 shows an example. In the figure, each bar represents predicted queueing

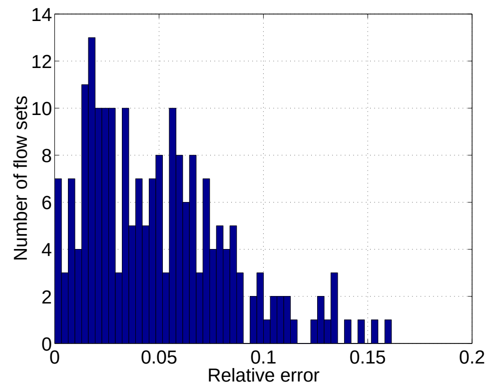
[8]In this paper, we define

$$\text{Relative error} = \frac{|\text{Simulation} - \text{Prediction}|}{\text{Simulation}}$$

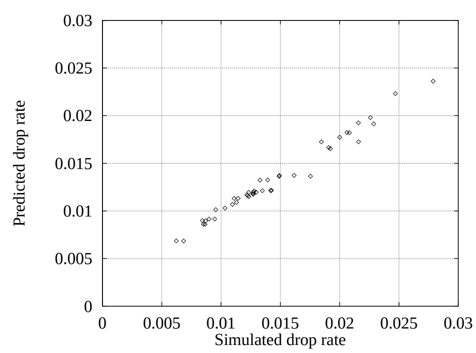
.



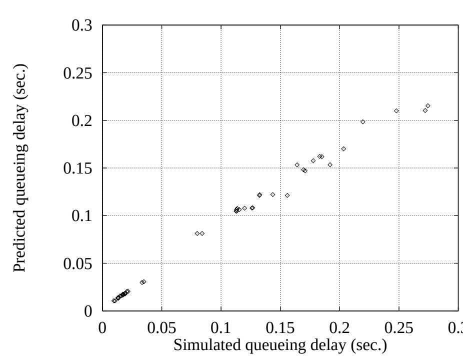
(a) Throughput



(b) Histogram of relative error



(c) Drop rate



(d) Queueing delay

Figure 7: Simulation and prediction with TCP traffic

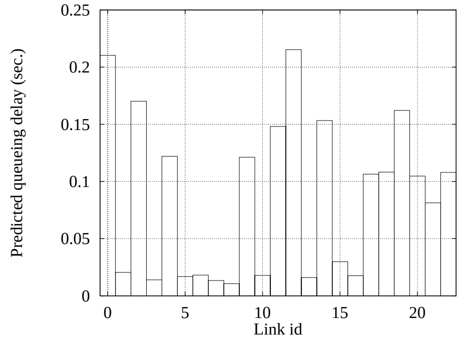


Figure 8: Queueing delay in each link

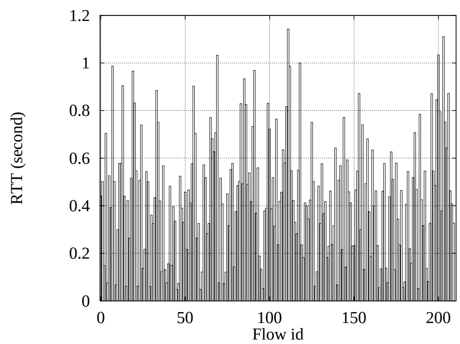


Figure 9: Predicted RTTs of individual flows

delay of each link in Fig. 6 with traffic described in Section III-B. We can easily identify which links are the most congested in the network.

B *Admission control*

As multimedia/real-time applications become popular, end-to-end packet delay becomes important. It is known that bounded delay service is difficult to be provided without admission control and explicit resource allocation. Our framework can be effectively used for admission control. For example, consider again the network in B. With the given offered traffic, average RTT of each flow is estimated in Fig. 9.

In the figure, the maximum average RTT is 1.14 sec. for the 111^{th} flow, which sends packets from node h to node o through {11,12,13,14} links. Now we perform admission control at this given network and

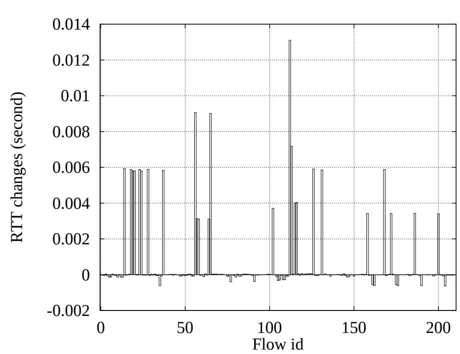


Figure 10: Predicted RTT changes after establishing the requested connections through {0,4,9} links

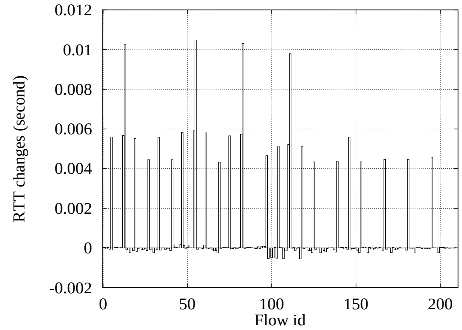


Figure 11: Predicted RTT changes after establishing the requested connections through {5,12,13,14} links

require that the average RTTs of every flows be less than or equal to 1.14 sec. Suppose that the following connection requests show up:

- 5 TCP flows from node a to node i through {0,4,9} links.
  Fig. 10 shows the predicted RTT changes from as in Fig. 9 caused by the requested connections. As a result, the maximum average RTT is still 1.14 sec., and the request is accepted.

- 5 TCP flows from node d to node o through {5,12,13,14} links.
  Fig. 11 shows the predicted RTT changes upon this request, and it is observed that the RTT of the 111^{th} flow is predicted to be 1.15 msec. This breaks the condition, and thus this request is declined.

Similarly, we can perform admission control on average drop rate of each flow, average throughput of each flow, and average drop rate and/or queue length at each link.

C *Limitations*

While developing our model and approach, we have found that there are two potential limitations:

1. Our model is based on the flow's throughput model (in this paper we use TCP's throughput model [19]). The accuracy of our model is thus highly dependent on the accuracy of the throughput model. The results presented in Fig. 7(a) show that the throughput of the individual flows is slightly overestimated, and this overestimation in throughput results in the underestimation of queueing delay and drop rate shown in Fig. 7.

2. The existence and uniqueness of a solution for Problem 1 is an open question. From our simulation experiments, we observe that ur numerical solution seems to be quite robust, always converging in a small number of steps and producing results close to those obtained from simulation.

V Related work

The problem of RED configuration under different traffic patterns has been considered in [11], where a self-tuning mechanism has been. A simple analytic model for RED was developed in [4]. Based on the model, several problems with RED have been addressed such as increasing loss rate of non-bursty traffic, higher number of consecutive packet drops, and increasing delay jitter. A model for transient analysis of RED routers has been proposed in [16]. An independent and simultaneous work [5] proposes a network model similar to ours, but does not include a model for differentiated services and uses a different numerical method based on fixed point.

Traffic engineering is focusing on optimizing overall network performance via the measurement, modeling, and control of traffic. In [1], some applications of MPLS (Multiprotocol Label Switching) to traffic engineering in IP networks have been proposed. In [10], a tool, called *NetScope*, for traffic engineering associated to the configuration of intra-domain routing has been proposed. This tool is useful but different from our work in that NetScope evaluates network performance depending on routing protocol while our work focuses on congestion control.

VI Conclusion

In this paper, we have proposed a framework for performance evaluation in IP networks. We have first proposed an analytic model for networks with single point of congestion with heterogenous TCP and UDP flows, short-lived TCP flows and with Assured Forwarding Differentiated Services network and presneted simulation results that confirm our model.

Next, we presented a practical and numerical approach for performance evaluation of large scale networks with multiple congested links and validated our model through simulation.

For future work, we plan to study the conditions for existence and uniqueness of solutions to Problem 1 and develop models for various drop functions and include them in our model.

Acknowledgements

We would like to thank Don Towsley for many useful discussions.

References

[1] D. Awduche. MPLS and Traffic Engineering in IP Networks. *IEEE Communications Magazine*, Dec 1999.

[2] D. Awduche, A. Chiu, A. Elwalid, I. Widjaja, and X. Xiao. A Framework for Internet Traffic Engineering. IETF Draft, March 2001.

[3] S. Blake, D. Black, M. Carlson, E. Davies, Z. Whang, and W. Weiss. An Architecture for Differentiated Services. RFC 2475, December 1998.

[4] T. Bonald, M. May, and J.-C. Bolot. Analytic Evaluation of RED Performance. In *IEEE Infocom*, 2000.

[5] T. Bu and D. Towsley. Fixed Point Approximations for TCP Behavior in an AQM Network. In *ACM Sigmetrics*, June 2001.

[6] N. Cardwell, S. Savage, and T. Anderson. Modeling TCP latency. In *IEEE Infocom*, 2000.

[7] K. Claffy, G. Miller, and K. Thompson. The nature of the beast: recent traffic measurements from an Internet backbone. In *Inet98*, 1998.

[8] D. Clark and W. Fang. "Explicit Allocation of Best Effort Packet Delivery Service". *IEEE/ACM Transactions on Networking*, 6, August 1998.

[9] C. R. Cunha, A. Bestavros, and M. E. Crovella. Characteristics of WWW client-based traces. Technical Report BU-CS-95-010, Boston University, July 1995.

[10] A. Feldmann, A. Greenberg, C. Lund, N. Reingold, and J. Rexford. NetScope: Traffic Enginnering for IP Networks. *IEEE Network Magazine*, 2000. Special issue on Internet Traffic Engineering.

[11] W. C. Feng, D. Kandlur, D. Saha, and K. Shin. A Self-configuring RED Gateway. In *IEEE Infocom*, 1999.

[12] V. Firoiu and M. Borden. A Study of Active Queue Management for Congestion Control. In *IEEE Infocom*, 2000.

[13] S. Floyd and V. Jacobson. Random Early Detection gateways for Congestion Avoidance. *IEEE/ACM Transactions on Networking*, 1(4), August 1997.

[14] J. Heinanen, F. Baker, W. Weiss, and J. Wroclawski. Assured Forwarding PHB Group. Internet RFC2597, June 1999.

[15] S. McCreary and K. Claffy. Trends in Wide Area IP Traffic Patterns. CAIDA, May 2000.

[16] V. Misra, W. Gong, and D. Towsley. A Fluid-based Analysis of a Network of AQM Routers Supporting TCP Flows with an Application to RED. In *Proceedings of SIGCOMM*, 2000.

[17] K. Nichols, S. Blake, F. Baker, and D. L. Black. Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers. IETF RFC 2474, Dec 1998.

[18] ns2. Network Simulator V.2. http://www.isi.edu/nsnam/ns/.

[19] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose. Modeling TCP Reno Performance: A Simple Model and Its Empirical Validation. *IEEE/ACM Transactions on Networking*, 8(2), April 2000.

[20] W. Stevens. *TCP/IP Illustrated, Vol.1 The Protocols*. Addison-Wesley, 1994.

[21] W. Stevens. TCP Slow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithms. RFC2001, Jan 1997.

[22] I. Yeom and A. L. N. Reddy. Modeling TCP Behavior in a Differentiated Services Network. Technical Report TR, Texas A&M University, 1998.