PAPER

Capacity Dimensioning of VPN Access Links for Elastic Traffic in the Hose Model

Keisuke ISHIBASHI[†a)], **Mika ISHIZUKA**[†], **Masaki AIDA**[†], *and* **Shin-ichi KURIBAYASHI**[†], *Members*

SUMMARY This paper discusses research into the capacity dimensioning of Virtual Private Network (VPN) access links for elastic traffic, such as the Web or ftp. Assuming that the core-VPN network is provisioned with a sufficiently large capacity, managing the capacity of the VPN access link comes to sharing the bandwidth for the elastic traffic of the two bottlenecks, the ingress and egress access links. In the case of a single bottleneck with a limited capacity for access links, the processor-sharing model provides a simple formula for mean transfer time, but here, the value may be less than the actual transfer time because multiple flow may compete the bandwidth of both ingress and egress links. In contrast, max-min fair sharing provides an accurate sharing model which is similar to the TCP, but it is difficult to obtain a closed form for performance statistics. We propose a closed form approximation for a max-min fair sharing model, within a specific but realistic topology, through an investigation into the difference between the max-min and the processor sharing model. Using approximation, we calculate the capacity dimensioning of VPN access links.

key words: *VPN, hose model, max-min fair sharing*

1. Introduction

Virtual Private Networks (VPNs) are widely used because they provide secure connections between customer-LANs at reasonable cost. The VPN service model is divided into two categories: the customer pipe model and the hose model [1].

The customer pipe model is an emulation of a leased lines. Using this model, the customer needs to prepare individual customer pipes for every pair of customer LANs. Traffic from one LAN to other LANs is separated among corresponding pipes. Traffic engineering for this model is essentially the same as that for leased lines. On the other hand, in the hose model, the customer only needs to prepare an access link to the VPN core network from each customer LAN, not each pair of LANs*. Traffic from the LAN to all other LANs is aggregated into the link. From the customer's point of view, one of the advantages of the hose model is its statistical multiplexing gain at the access link.

Duffield et al. evaluated the gain using the actual traffic data of corporate network [1]. They allocated the capacity based on the peak rate, and stated that the capacity could be reduced by 6.5% up to 72%. Lee et al. also studied the gain by calculating the capacity to meet given packet loss probability [4]. They used a sample traffic matrix and showed the gain was from 16% to 42%. In our paper, we focus on flow level performance, such as the mean transfer time for TCP traffic in dimensioning access-link capacity. Because a large portion of current traffic is TCP [3], it is adequate to consider performance at the TCP flow level for capacity dimensioning.

To evaluate the performance of TCP flow, we must consider both the core network, provisioned by the provider and the access network provisioned by the customer. However, providers normally provision a core network with sufficient capacity so that the network cannot or rarely becomes a bottleneck [1], [5]. Therefore, we can assume that the user can fully utilize the capacity of their access links. Based on this assumption, traffic engineering for the VPN access link becomes a bandwidth-sharing problem for elastic traffic that goes through two bottlenecks: the ingress and egress access links.

In this paper, we consider the case of a network that consists of one central LAN and multiple branch LANs ("star-topology"), which is the most typical network topology for VPN of enterprises which consists one central office and several branch offices, and where elastic traffic is transferred between the central LAN and branch LANs (Fig. 1). We can expect to obtain a statistical multiplexing gain for the access link of the central LAN, where many flows from/to other branch LANs share the bandwidth.

Many papers have addressed TCP bandwidth sharing issues using the processor sharing model [6], [9]–[12]. In this model, flows share the bandwidth equally and the throughput is equal to the capacity divided by the number of competing flows. When a flow cannnot utilize the capacity itself due to the rate-limited access link, the throughput of a flow is determined by the min-

Manuscript received April 2, 2003.
Manuscript revised August 13, 2003.

[†]The authors are with the NTT Information Sharing Platform Laboratories, NTT Corporation, Musashino-shi, 180-8585 Japan.
a) E-mail: ishibashi.keisuke@lab.ntt.co.jp

*More precisely pipe model specifies the traffic/QoS requirements for each pair of LANs and the hose model specifies the requirements for each LAN and core network. But, in this paper, we have considered the minimum bandwidth guarantee for QoS and identify the guranteed bandwidth as the physical bandwidth for access links.

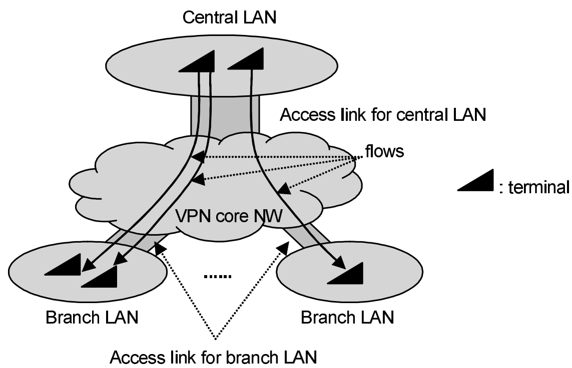

Fig. 1 Hose model VPN.

imum of the equally-shared bandwidth of a bottleneck link and the bandwidth of an access link. In this model, the performance index such as mean transfer time can be calculated using the capacity and utilization ratio for a Poisson arrival case. It should be mentioned that the indexes are independent of the file size distribution.

It seems that this model can be applied to our case, because access links with limited capacity in the model correspond to branch-LAN access links, and the bottleneck link where many flows compete for its bandwidth corresponds to the central-LAN access link in our case. However, because this model does not consider when multiple flows compete with the access link of a branch LAN, the mean transfer time for VPN access may be underestimated when applied to our scenario.

In terms of multiple bottleneck links, there are many papers that deal with TCP flows that share the bandwidth of links with max-min fair sharing or proportional sharing for general topology [15]–[18].

Most of these consider persistent flow i.e., the number of flows remain fixed, but some deal with a variable number of flows expressed as a random variable [18], [19], where max-min sharing is presumed. However, because the exact mean transfer time for max-min sharing is difficult to derive, they propose approximations of sharing. While these approximations can be applied to general topology, both Chanda and Fayolle et al.' results do not have sufficient accuracy for the capacity dimensioning.

In this paper, we assume that TCP shares bandwidth according to max-min fair sharing and we propose an accurate approximation of performance for star topology by focusing on the difference between max-min sharing and processor sharing with a narrow access link. Using this approximation, we provide some examples of methods for the capacity dimensioning of VPN access links to achieve various performance objectives.

The rest of the paper is organized as follows. We begin by describing conditions this paper concerned and observing some bandwidth sharing models. We also provide an approximation in Sect. 2. Section 3 evaluates the approximation in various network scenarios. Finally, we provide methods for capacity dimensioning of access links, using the approximation in Sect. 4.

2. Bandwidth Sharing for VPN Access Links

2.1 Bandwidth Sharing—Processor Sharing and Max-Min Sharing

In this subsection, we compare two bandwidth sharing policies: processor sharing with an access link of limited capacity and max-min sharing.

Let us describe the conditions we have assumed in this paper.

- A customer VPN consists of one central LAN, whose access-link capacity is C [bps] and k branch LANs, whose access-link capacity is r [bps]. (There is a homogeneous environment among branch LANs.) The capacity of LANs and the core network are sufficiently large so that these networks cannot become bottlenecked.
- Traffic type is TCP file transfer between the central LAN and k branch LANs.
- Flow-arrival process is Poisson [22] with arrival rate λ [1/s] for each branch-LAN (for central-LAN, arrival rate is $k\lambda$).
- File size is exponentially distributed and its mean is s [bits].
- Mean file size is sufficiently large so that we can ignore slow-start effect, and flow achieves fair sharing instantaneously.
- Window size is sufficiently large so that a flow can utilize access-link capacity when there are no other flows within the link.
- The performance criterion is the mean transfer time.

We also prepared notations of utilization for the central-LAN access link that are $\rho_C := k\lambda s/C$ and that for branch-LANs as $\rho_r := \lambda s/r$. Using these notations, multiplex gain for central-LAN access links can be evaluated with $(kr - C)/C$.

According to the processor sharing model with limited capacity, r [bps], available bandwidth for a flow is determined depending on the number of competing flows in a central-LAN access link, and equals $\min(C/n, r)$, where n is the number of flows in the central-LAN access link at a point of time. Then, the mean transfer time under the processor sharing policy, T_{PS}, is calculated for general distributed file size [12] as

$$T_{\mathrm{PS}}(C, \rho_r) = s\left[\frac{1}{r} + \frac{\mathrm{E}_{2,R}(k\rho_r)}{(1-\rho_C)C}\left\{1 - (1-\rho_C)\left(\frac{C}{r} - R\right)\right\}\right], \quad (1)$$

where $R = \lfloor \frac{C}{r} \rfloor$ and $\mathrm{E}_{2,R}(k\rho_r)$ is Erlang's C formula [20], which can be written as:

$$\mathrm{E}_{2,R}(k\rho_r) := \frac{\left(\frac{(k\rho_r)^R}{R!}\right)\left(\frac{1}{1-\rho_r}\right)}{\sum_{k=0}^{R-1}\frac{(k\rho_r)^k}{k!} + \left(\frac{(k\rho_r)^R}{R!}\right)\left(\frac{1}{1-\rho_r}\right)}. \tag{2}$$

However, this result cannot be directly applied to our case. This is because when multiple flows share an access link of a branch-LAN, a flow cannot utilize the full bandwidth of the access-link as flows compete for the bandwidth, and the available bandwidth is smaller than $\min(C/n, r)$.

On the other hand, according to the max-min model, multiple flows in a branch-LAN access link are considered. In the model, the available bandwidth for a flow in i-th branch-LAN $b_i(N)$ is determined not only by n, but also by state vector $N = \{n_i\}_{i=1}^k$ whose element n_i is the number of flows for the i-th branch-LAN access link as follows [21]. First, without loss of generality, suppose that the index is descendingly ordered by n_i, i.e., $n_i \geq n_{i+1}$. Let $g = \sum_{i=1}^k \mathbf{1}_{\{n_i>0\}}$ (number of branch-LANs that has at least one flow to central-LAN) and $n_0 = 0$.

Step 1: Let $i = 1$.
Step 2: If

$$\frac{r}{n_i} \leq \frac{C-(i-1)r}{n-\sum_{j=0}^{i-1} n_j}. \tag{3}$$

then go to **Step 3**. Otherwise for flows in branch LANs $i, i+1, \ldots g$, the available bandwidth is limited by the capacity of the central-LAN access-link, and

$$b_l(N) = \frac{C-(i-1)r}{n-\sum_{j=0}^{i-1} n_j}, \quad l = i, \ldots, g. \tag{4}$$

Stop the procedure.

Step 3: Access link for branch i is bottlenecked and $b_i(N) = r/n_i$. Set $i \leftarrow i+1$ and return to **Step 2**.

To see which sharing policy represents the TCP bandwidth sharing mechanism, we compared the performance obtained by the processor sharing model using (1), the max-min model and *ns* simulation [23] as a reference for TCP performance. Because it is difficult to obtain a closed form for performance statistics for the max-min model, we ran simulations of the Markov process for N including 100,000 transissions where the departure rate-vector is given as $\{n_i b_i(N)/s\}_{i=1}^k$. Through the simulations, we obtained a mean of n from which we calculated the mean transfer time using Little's Law [20]. The parameter set used here is listed in Table 1.

For *ns* simulations, we set the propagation delay for an access link to 10 ms, with a maximum window size of 20 kbytes, and the number of terminals in a branch LAN to 100. The core network was represented by a single node. We set the buffer size of every router to 10,000 packets so that we could consider that routers had infinitely large buffers.

Table 1 Evaluation conditions.

k (number of branch-LANs)	10
r (bandwidth branch-LAN link)	1.5 Mbps
s (mean file size)	1 Mbyte
ρ_r (utlization ratio for branch-LAN)	0.2, 0.4, 0.6, and 0.8

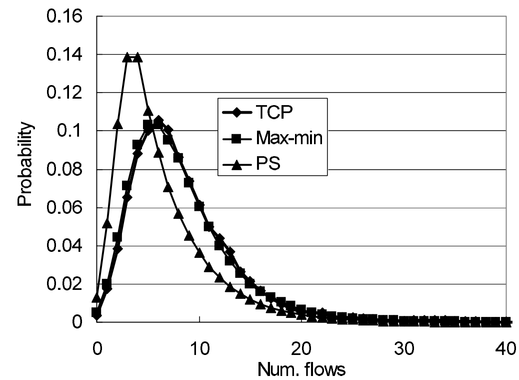

Fig. 2 Distribution of number of flows for $\rho_r = 0.4$ and $C = 7.5$ [Mbps].

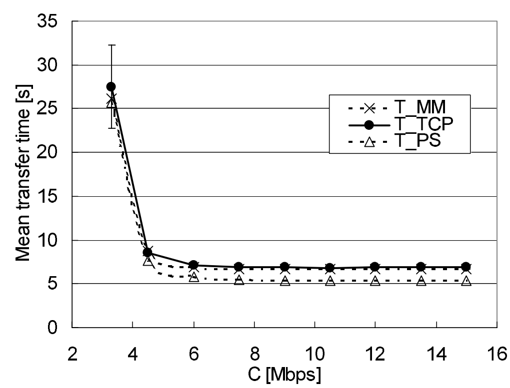

Fig. 3 Mean transfer time for $\rho_r = 0.2$.

We used TCP Tahoe for the simulation†. We ran five simulations each lasting 36,000-s and derived average and 95% confidence intervals.

Figure 2 shows the distribution of n when the utilization ratio ρ_r is 0.4 and $C = 7.5$ Mbps. We found that while the max-min sharing model and TCP agree in the distribution, the processor-sharing model has a mass probability with lower n than TCP.

†The mean transfer time with TCP Tahoe may be larger than those with TCP Reno or TCP SACK particularly when packet loss occurs [24], [25]. And the max-min model gives the minimum mean file transfer time because it assumes that a flow can fully utilize the available bandwidth. In our case, as will be seen later (espesially in Sect. 3.4 where packet loss occurs), the mean file transfer time with TCP Tahoe is roughly as same as those with the max-min model. Thus the mean file transfer time with TCP Reno or SACK is also expected to be approximately same as those with the max-min model.

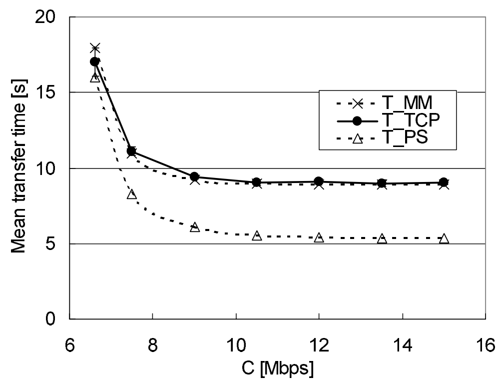

Fig. 4 Mean transfer time for $\rho_r = 0.4$.

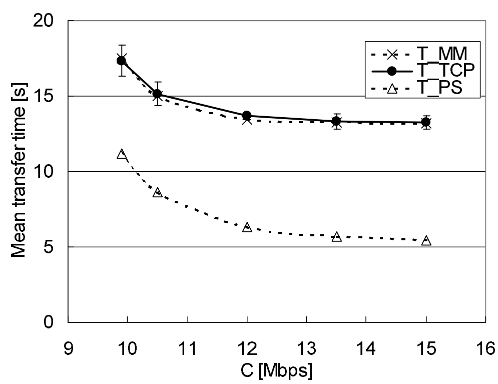

Fig. 5 Mean transfer time for $\rho_r = 0.6$.

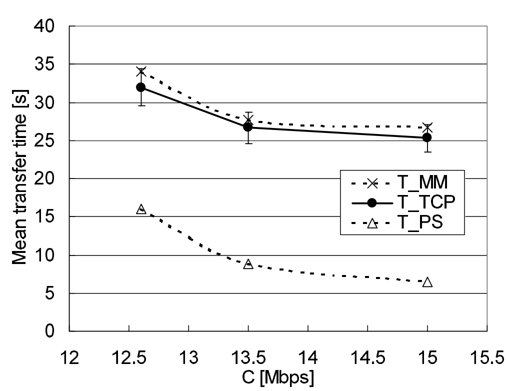

Fig. 6 Mean transfer time for $\rho_r = 0.8$.

Figures 3–6 have the mean transfer time for TCP ($T_{\rm TCP}(C, \rho_r)$), the processor-sharing model ($T_{\rm PS}(C, \rho_r)$), and the max-min model ($T_{\rm MM}(C, \rho_r)$) with respect to the capacity of central-LAN access links. From these figures, we can see that when ρ_r is high and C is large, there is considerable discrepancy between $T_{\rm PS}$ and $T_{\rm TCP}$.

In contrast, with the max-min sharing model, both the distribution of the number of flows and the mean transfer time are very similar to TCP results.

Thus, we can consider the performance obtained through the max-min model as the actual performance in our scenario where a TCP flow fairly shares the bandwidth among other competing flows. This is the condition that RTT is small and mean file size are sufficiently large.

In general, however, the max-min sharing model is not tractable to obtain a closed form of the stationary distribution for the number of flows or mean transfer time [7], [18], [19]. In the next subsection, we propose an approximation for the mean transfer time using $T_{\rm PS}$.

2.2 Approximation of Mean Transfer Time

To derive an approximation of $T_{\rm MM}$ using $T_{\rm PS}$, let us first describe the difference between $T_{\rm MM}$ and $T_{\rm PS}$ for several special cases.

(A) In general, when $C = r$, the access links of branch LANs cannot be bottlenecked, and the access link of the central LAN is the only single bottleneck for the network. Thus, the performance of the max-min model is the same as that of the processor-sharing model because it assumes equal bandwidth sharing among flows for a single-bottleneck case. When $\rho_C \simeq 1$ ($C \simeq kr\rho_r$) and $\rho_r < 1$, both performance outcomes also agree because

$$b_i(N) \to C/n, \ \text{ as } \ \rho_C \to 1. \tag{5}$$

(B) When $C = kr$, then in the max-min model, the central-LAN access link is no longer bottlenecked, and $T_{\rm MM}(kr, \rho_r) = T_{\rm MM}(\infty, \rho_r)$.

(C) When $C = \infty$, then available bandwidth for the processor-sharing model $\min(C/n, r) = r$, thus the model derives s/r as a mean transfer time. On the other hand, the available bandwidth through the max-min sharing model is r/n_i. By applying processor sharing for each access link for branch LAN independently, we derive the mean transfer time as $\frac{s}{r(1-\rho_r)}$ [7]. Thus, $T_{\rm MM}(\infty, \rho_r)$ is $1/(1-\rho_r)$ times larger than $T_{\rm PS}(\infty, \rho_r)$.

To explore $T_{\rm MM}$ in general cases, we defined the compensation formula $F(C, \rho_r) := \frac{T_{\rm MM}(C,\rho_r)}{T_{\rm PS}(C,\rho_r)}$ and plot

$$\frac{1}{1-\rho_r} - F(C, \rho_r) \tag{6}$$

for several ρ_r against C semi-logarithmically (Fig. 7).

From the above condition (C), we expected that (6) would approach to 0 as C increases. In addition, from Fig. 7, we observed that (6) decayed exponentially as C increased for every ρ_r. Thus, we expected that $F(C, \rho_r)$ can be written as

$$F(C, \rho_r) \approx \frac{1}{1-\rho_r} - \alpha \exp(-\beta C). \tag{7}$$

Then parameters α and β can be calculated using conditions (A) and (B). Here, we calculated α and β when

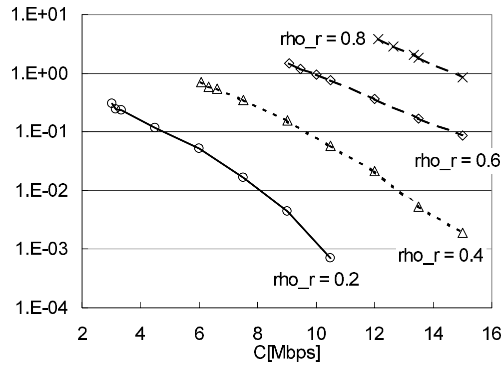

Fig. 7 Plot of $\frac{1}{1-\rho} - F(C, \rho_r)$.

$k\rho_r > 1$ ($C = r$ means $\rho_C > 1$). Then from condition (A), we derive

$$\alpha \exp(-\beta k \rho_r r) = 1. \tag{8}$$

From condition (B), we derive

$$\alpha \exp(-\beta k r) = \frac{s}{r} \frac{1-\rho_r}{T_{\mathrm{PS}}(kr, \rho_r)}. \tag{9}$$

Therefore, we have an approximation $F'(C, \rho_r)$ of $F(C, \rho_r)$ as follows:

$$F'(C, \rho_r) := \frac{1}{1-\rho_r} - \frac{\rho_r}{1-\rho_r} \left(\frac{1 - \frac{s/r}{T_{\mathrm{PS}}(kr, \rho_r)}}{\rho_r} \right)^{\frac{C/kr - \rho_r}{1-\rho_r}}. \tag{10}$$

Thus, the approximated mean transfer time T_{APX} is given using T_{PS}, which can be given in closed form, as

$$T_{\mathrm{APX}}(C, \rho_r) := T_{\mathrm{PS}}(C, \rho_r) F'(C, \rho_r). \tag{11}$$

3. Simulation Results

In this section, we evaluated the approximation (11) for various environments. The reference transfer time for TCP was obtained through *ns*-simulations as well.

3.1 Conditions in Sect. 2

We first evaluated the conditions used in Sect. 2. We plotted the mean transfer time obtained with (11) for the situations used in Sect. 2 (Fig. 8). As a natural expectation from the results of Fig. 7, it can be found that the (11) approximates TCP performance well, even when utilization is high.

3.2 Pareto Distributed File Size

We also evaluated our approximation with a Pareto distributed file size. Because the processor-sharing model, which our approximation agrees with in extreme cases,

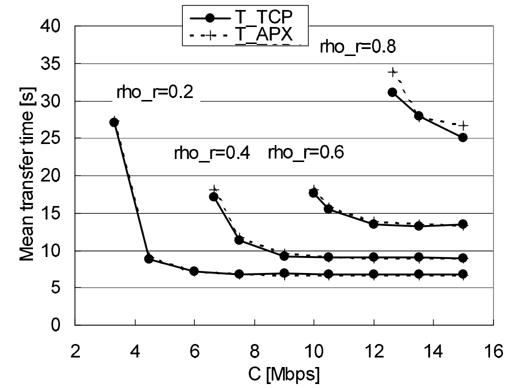

Fig. 8 Approximated mean transfer time.

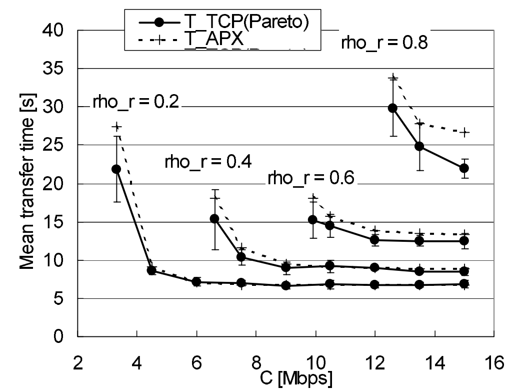

Fig. 9 Mean transfer time for Pareto distributed file size.

proved to be insensitive to the distribution of file size [26], we expected that our approximation would also be insensitive. Figure 9 shows the mean transfer time for Pareto distribution and our approximation for the shape parameter 1.5. Though the fit is not as good as those for the exponential case, it can be said that (11) can also approximate performance for Pareto distribution. The deviation of T_{TCP} from T_{APX} was considered to mainly derive from the large variation in file transfer time due to heavy tail distribution of file size.

3.3 Small File Size

We changed the mean file size from 1,000 KB to 50 KB to access the effect of TCP's initial slow start on mean transfer time Figures 10–13 have the mean transfer time for the hose model, the approximation of time for the hose model, and the time for the pipe model. We can see that there are non-negligible differences between actual transfer time and approximated transfer time. Several papers [13], [14], [27] pointed out this difference for the processor sharing model and proposed ways to compensate for this difference. For example, Ozawa proposed adding delay for slow-start phase to the transfer time calculated with PS model in [13].

Thus, if we use the approximation formula (11) to calculate the exact mean transfer time for small files, we

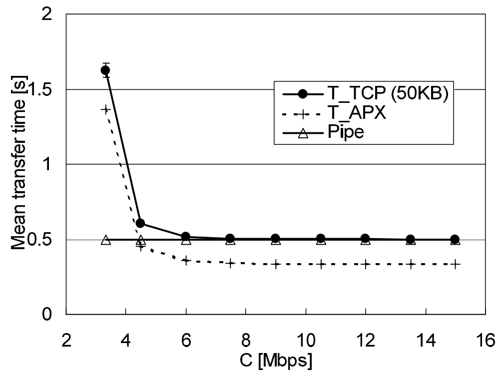

Fig. 10 Mean transfer time for 50 KB average file size (ρ_r = 0.2).

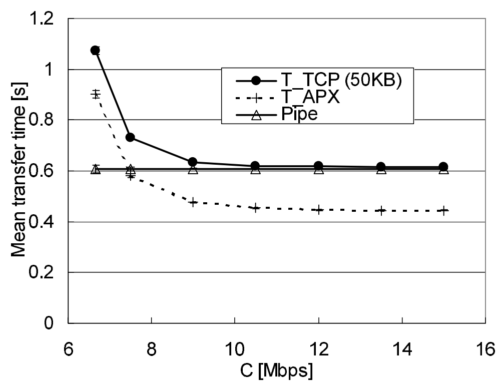

Fig. 11 Mean transfer time for 50 KB average file size (ρ_r = 0.4).

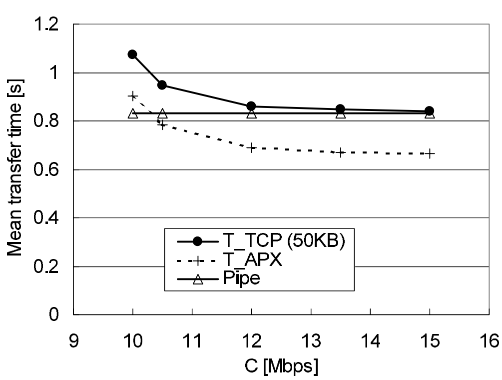

Fig. 12 Mean transfer time for 50 KB average file size (ρ_r = 0.6).

should introduce this compensation into our formula.

Note that the mean transfer time for the pipe model also increased due to the slow start effect (without slow start, the transfer time should equal $T_{\mathrm{APX}}(15, \rho_r)$). Thus, even for small files, the approximation formula (11) may suffice for the capacity dimensioning method where a capacity for the hose model is calculated to achieve the same mean transfer time as that for the pipe model. Section 4 presents the results obtained from simulating capacity dimensioning

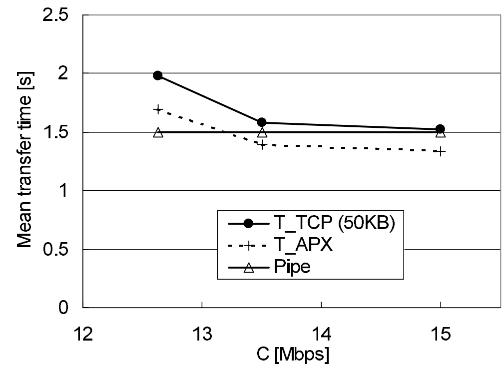

Fig. 13 Mean transfer time for 50 KB average file size (ρ_r = 0.8).

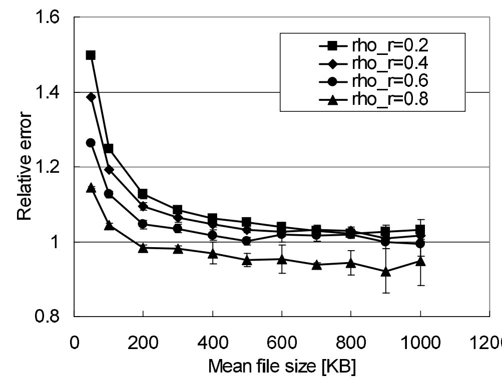

Fig. 14 Relative error of the approximation for various mean file size when C is fixed to 15 [Mbps].

for small files.

We also investigate the relative error of our approximation for various mean flow sizes (Fig. 14). We set capacity of central LAN access link (C) as 15 Mbps, where the slow start effects are largest as can be seen from Figs. 10–13 (For C larger than 15 Mbps, the mean transfer time will be the same as 15 Mbps case). From the figure, we can see that slow-start effect is negligble when mean file size is larger than 500 KB.

3.4 Small Buffer Size

So far, we have evaluated for the buffer size of 10,000 packet case, where no packet loss occured. Here, we changed the buffer size to 50 packet and ran simulations to see the effect of packet losses (Fig. 15). We can see that in the case for narrow capacity of central-LAN access links, actual TCP transfer time was longer than the approximated transfer time (T_{APX}), which was calculated using nominal utilization ratio (utilization ratio calculated with the input parameter of the simulations). Followings can be considered as the reasons of this increase:

- due to the slow start after packet losses, the bandwidth may not fully utilized

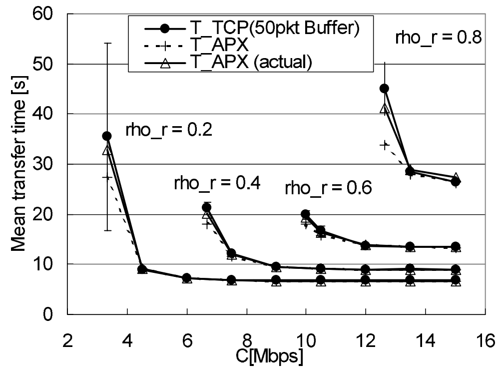

Fig. 15 Mean transfer time for 50 packet buffer.

- actual utilization ratio may increase due to the retransmisions after packet losses.

To eliminate the latter effect, we also calculate approximated mean transfer time using acutal utilizatoin ratio (T_{APX}(actual)). We can see that TCP transfer time can fairly be approximated by using the actual utilization ratio, which can be obtaind through easy traffic measurement.

3.5 Heterogenous Access Links

Until now, we have only evaluated our approximation formula (11) in a homogenous access environment. In this subsection, we test this formula in a heterogenous access environment. We derive the approximation formula on the basis of two processor sharing models for extreme cases, the M/G/1/PS for $C = kr$ case and the M/G/R/PS for $C \simeq kr\rho_r$ case. The mean transfer time for the M/G/1/PS model is calculated independently for each access link, thus heterogenity does not matter. However, the M/G/R/PS model, where multiple flows share the bandwidth of central-LAN access links, has not yet been completely extended to heterogenous access links, to the best of our knowledge. Lindberger suggested that by using the average capacity of access links weighted by its utilization ratio in place of r in (1), the mean transfer time could be roughly approximated [28], [29]. However, the number of different access links is limited to two types in deriviating the approximation. Thus, our formula cannot be directly extended to a heterogenous environment due to the lack of a basic model. Let us now discuss how the heterogenity affects mean transfer time.

Figure 16 has the mean transfer time for multiple access capacity, one 3 Mbps, two 2.25 Mbps, three 1.5 Mbps and four 0.75 Mbps links (1.5 Mbps on average) and its approximation assuming the bandwidth of all access links is 1.5 Mbps (mean transfer times for the simulation results and approximation will be the same for $C = kr$). As expected, there is a difference between T_{APX} and T_{TCP} for $C \simeq kr\rho_r$. However, except for this extreme case, our approximation, assuming homogenous access links, can predict the actual mean transfer time fairly well and can be applied to this network environment.

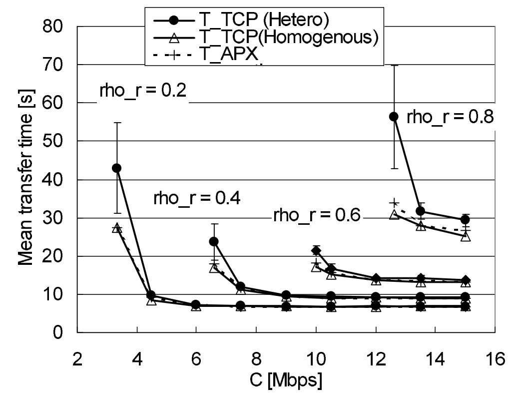

Fig. 16 Mean transfer time for multiple access capacity.

4. Capacity Dimensioning

This section presents two example methods for capacity dimensioning of VPN access links using the approximation given in Sect. 2.

We first fix the capacity for a branch LAN, and calculate the required capacity of the access to the central LAN to achieve the target mean transfer time. The target was set as γ (> 1) times the quantity of the mean transfer time obtained from the pipe model where the capacity of the pipe was configured the same as for the branch LAN access link. Note that the mean transfer time for $\gamma = 1$ cannot be achieved if the capacity of the branch LAN is the same as the pipe model and $C < kr$. This capacity is given using (11) as follows:

$$C_r := \min\left\{ C \,\middle|\, T_{\mathrm{APX}}(C, \rho_r) \leq \frac{\gamma s}{r(1-\rho_r)} \right\} \qquad (12)$$

We calculated C_r for $\gamma = 1.1$ and $\rho_r = 0.2$, 0.4, 0.6, and 0.8 for the same environment in Sect. 2. Figure 17 shows the results. We also calculate the required capacity using T_{TCP}. We found that using our approximation, the required capacity could be calculated accurately and multiplexing gain kr/C_r was large when the utilization of branch LAN, ρ_r, was low.

As another example, we set $\gamma = 1$ and increased the capacity for branch LANs, r. Then, given the capacity for branch LANs, we determined the required capacity for the central LAN, C_r.

In this example, as r increases, the C_r to achieve the same mean transfer time for the pipe model decreases, and the degree-of-freedom increases.

Thus, to determine capacities, we intoduce a cost function $P(\cdot)$ of capacity, which is, for example, monthly fee for the access link determined by its capacity.

Then, we seek the capacity of the branch LAN that

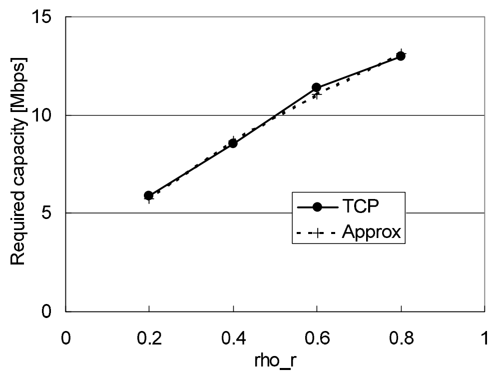

Fig. 17 Required capacity to achieve the objective performance.

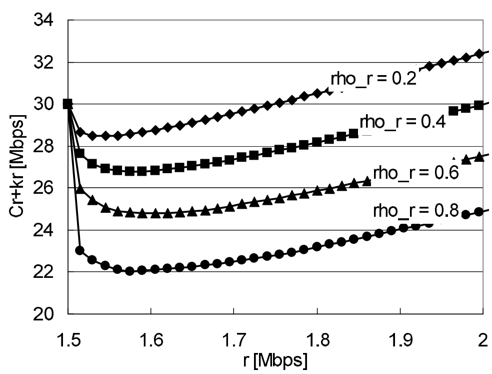

Fig. 18 Total required capacity ($kr + C_r$) for hose model.

would minimize the total cost as

$$r^* = \operatorname*{argmin}_r kP(r) + P(C_r). \tag{13}$$

When the cost function P is linear and non-decreasing [30], then (13) is equivalent to

$$r^* = \operatorname*{argmin}_r kr + C_r \tag{14}$$

Figure 18 show the total required capacity ($= kr + C_r$) changing the capacity of branch-LAN access links. We see that the total required capacity take its minimum for r at about 1.6 Mbps.

Thus, by increasing the bandwidth of branch-LAN access links by about 5%, we can decrease the total required bandwidth, for example, by about 26% when $\rho_r = 0.2$. We observed that the total required capacity was small for a low utilization ratio, and increased as the utilization ratio increased, the same as for Fig. 17. Of course these values depend on the condition such as the number of branch LAN. However, we can numerically evaluate the relationship between the increase of branch-LAN access links and the decrease of the total required bandwidth using our method.

From Table 2, we can also see that r^*, optimal capacity for branch LAN access links, take its minimum at $\rho_r = 0.4$. However, because we have not yet derived a formulae that derives r^* as a closed form, we cannnot tell directly how $r*$ changes as a function of ρ_r.

Table 2 Simulation results for r^* and C_{r^*}.

ρ_r	0.2	0.4	0.6	0.8
r^* (Mbps)	1.58	1.67	1.58	1.55
C_r^* (Mbps)	6.27	8.75	11.00	13.01
$kr^* + C_{r^*}$ (Mbps)	22.0	25.4	26.8	28.5
r^* (with PS) (Mbps)	1.50	1.50	1.50	1.50
C_r^* (with PS) (Mbps)	5.04	7.34	9.79	12.3
$kr + C_r^*$ (with PS) (Mbps)	20.0	22.3	24.8	27.3
T_{Pipe} (50 KB) (s)	0.495	0.608	0.833	1.50
T_{Min} (50 KB) (s)	0.496	0.601	0.826	1.54
T_{Min} (50 KB) (with PS) (s)	0.55	0.76	1.20	3.28
T_{APX} (50 KB) (s)	0.333	0.444	0.667	1.33
T_{Pipe} (1,000 KB) (s)	6.61	8.96	12.9	26.5
T_{Min} (1,000 KB) (s)	6.60	8.64	12.9	25.5
T_{Min} (1,000 KB) (with PS) (s)	7.79	11.6	18.8	40.4
T_{APX} (1,000 KB) (s)	6.67	8.89	13.3	26.7

To prove the effectiveness of the dimensioning method, we ran simulations for the hose model where the bandwidth of branch-LAN and central LAN access links were r^* and C_{r^*}, and for the pipe model where pipe bandwidth is 1.5 Mbps. Two mean file sizes, 1,000 KB and 50 KB, were evaluated. The other simulation conditions were the same as in Sect. 2. We compared the mean transfer time obtaind through the hose with r^* and C_{r^*} simulation results (which are denoted as T_{Min}), pipe simulation results (which are denoted as T_{Pipe}), and those calculated the approximation formulae for $C = kr$ case (T_{APX}). Note that T_{APX} should be equal to those for the pipe model if TCP strictly achives max-min sharing (without slowstart). Table 2 shows the results. We can see that the mean transfer time for the dimensioning method was almost the same as that for the pipe model. It should be mentioned that while T_{Min} was larger than T_{APX} for a 50-KB file size, the above equation still held, as indicated in Sect. 3.

We also calculate r^* and C^* using PS model and evaluate the mean transfer time through simulations using these bandwidth. The results are shown in Table 2 with the notation "(with PS)." As is expected from Figs. 3–6, calculated bandwidthes with PS model are smaller than those with the proposed approximation formulae. As a result, the mean transfer time increases, especially when high utilization ratio.

5. Conclusion

We considered capacity dimensioning for VPN access links for elastic traffic with star topology. We confirmed that the simple processor sharing model could not be applied to our case, and the max-min sharing model was able to predict TCP performance. Because the exact calculation of performance is difficult, we proposed a closed form approximation of max-min fair sharing. The approximation was derived by considering the difference between processor sharing and max-min sharing models. We discussed two methods of dimensioning using the approximation.

Our work currently focus on only TCP traffic. Current traffic includes both TCP and UDP traffic. In that case, our method cannot be applied diretly to capacity dimensioning. And modification of our method to those case remains for further study. However, a rough estimate of required capacity can be calculated by adding capacity for UDP traffic to that calculated by our method, because UDP does not have any congestion control mechanism and its traffic tends to utilize bandwidth as possible.

Acknowledgements

We would like to thank Dr. Ryoichi Kawahara for his valuable comments and Ms. Kyoko Ashitagawa for her help with the simulations.

References

[1] N.G. Duffield, P. Goyal, and A. Greenberg, "A flexible model for resource management in virtual private networks," Proc. ACM SIGCOMM'99, 1999.
[2] B. Davie and Y. Rekhter, MPLS: Technology and applications, Morgan Kaufmann, San Francisco, 2000.
[3] K. Thompson, G.J. Miller, and R. Wilder, "Wide-area Internet traffic patterns and characteristics," IEEE Netw., Nov./Dec. 1997.
[4] H. Lee, Y. Uh, M.-T. Hwang, J.-H. Eom, Y.-G. Lee, and Y. Nemoto, "Capacity design of guaranteed-QoS VPN," IEICE Trans. Commun., vol.E85-B, no.5, pp.1042–1045, May 2002.
[5] A. Kumar, R. Rastogi, A. Silberschatz, and B. Yener, "Algorithms for provisioning virtual private networks in the hose model," Proc. ACM SIGCOMM'01, 2001.
[6] M. Nabe, M. Murata, and H. Miyahara, "Analysis and modeling of World Wide Web traffic for capacity dimensioning of Internet access lines," Performance Evaluation, vol.34, no.4, 1998.
[7] J.W. Roberts and L. Massoule, "Bandwidth sharing and admission control for elastic traffic," Proc. ITC Specialist Seminar 1998.
[8] A.W. Berger and Y. Kogan, "Dimensioning bandwidth for elastic traffic in high-speed data networks," IEEE/ACM Trans. Netw., vol.8, no.5, pp.643–654, 2000.
[9] D.P. Heyman, T.V. Lakshman, and A.L. Neidhardt, "A new method for analysing feedback-based protocols with applications to engineering web traffic over the Internet," Proc. ACM SIGMETRICS'97, 1997.
[10] A. Riedl, T. Bauschert, M. Perske, and A. Probst, "Investigation of the M/G/R processor sharing model for dimensioning of IP access networks with elastic traffic," Proc. First Polish-German Teletraffic Symposium, 2000.
[11] J.V.L. Beckers, I. Hendrawan, R.E. Kooij, and R.D. van der Mei, "Generalized processor sharing performance models for Internet access lines," Proc. 9th IFIP Conf. on Performance Modelling and Evaluation of ATM & IP Networks, 2001.
[12] S.B. Fredj, T. Bonald, A. Proutiere, G. Regnie, and J.W. Roberts, "Statistical bandwidth sharing: A study of congestion at flow level," Proc. ACM SIGCOMM'01, 2001.
[13] T. Ozawa, "Analysis of the mean response time using a processor sharing model," IEICE Technical Report, NS2002-12, 2002.
[14] M. Fukushima, H. Nakamura, and S. Nomoto, "A study on a model of bandwidth sharing among multiple TCP connections with slow start," Proc. IEICE-CS Conf. IEICE2002, SB-13-3, 2002.
[15] L. Massoulie and J. Roberts, "Bandwidth sharing: Objectives and algorithms," IEEE/ACM Trans. Netw., vol.10, no.3, pp.320–328, 2002.
[16] T. Bonald and L. Massoulie, "Impact of fairness on Internet performance," Proc. ACM SIGMETRICS'01, 2001.
[17] J. Mo and J. Walrand, "Fair end-to-end window-based congestion control," IEEE/ACM Trans. Netw., vol.8, no.5, pp.556–567, 2000.
[18] P.S. Chanda, A. Kumar, and A.A. Kherani, "An approximate calculation of max-min fair throughputs for non-persistent elastic flows," Proc. IEEE GLOBECOM'01, 2001.
[19] G. Fayolle, A. Fortelle, J.-M. Lasgouttes, L. Massoulie, and J. Roberts, "Best-effort networks: Modeling and performance analysis via large networks asymptotics," Proc. IEEE INFOCOM'01, 2001.
[20] L. Kleinrock, Queuing Systems, vol.1, John Wiley & Sons, New York, 1976.
[21] D. Bertsekas and R. Gallager, Data Networks, 2nd ed., Prentice-Hall, Upper Saddle River, NJ, 1992.
[22] V. Paxon and S. Floyd, "Wide-area traffic: The failure of Poisson modeling," IEEE/ACM Trans. Netw., vol.3, no.3, pp.226–244, June 1995.
[23] UCB/LBNL/VINT Network Simulator—ns (version 2). http://www.isi.edu/nsnam/ns.
[24] K. Fall and S. Floyd, "Simulation-based comparisons of Tahoe, Reno and SACK TCP," ACM SIGCOMM Computer Communication Review, vol.26, no.3, July 1996.
[25] G. Hasegawa, M. Murata, and H. Miyahara, "Fairness and stability of congestion control mechanisms of TCP," Telecommunication Systems Journal, pp.167–184, Nov. 2000.
[26] F.P. Kelly, Reversibility and Stochastic Networks, John Wiley & Sons, New York, 1979.
[27] A.A. Kherani and A. Kumar, "Stochastic models for throughput analysis of randomly arriving elastic flows in the Internet," Proc. IEEE INFOCOM'02, 2002.
[28] K. Lindberger, "Delays in ABR-like traffic and relations to processor sharing," COST 257 TD(98) 10, 1998.
[29] K. Lindberger, "Balancing quality of service, pricing and utilization in multiservice networks with stream and elastic traffic," Proc. ITC 16, 1999.
[30] http://www.ntt-east.co.jp/mdn/ryokin.html (in Japanese).

Keisuke Ishibashi received his B.S. and M.S. in Mathematics from Tohoku University, Miyagi, Japan in 1993 and 1995, respectively. Since joining NTT in 1995, he has been engaged in research on traffic issues in computer communication networks. He received IEICE's Young Investigators Award and Information Network research award in 2002. Mr. Ishibashi is a member of the IEEE and the Operations Research Society of Japan.

Mika Ishizuka graduated from Keio University, Tokyo, with B.E. and M.E. degrees in Control Engineering in 1992 and 1994, respectively. In 1994, she joined NTT and has been engaged in research on traffic control in computer networks. She received the IEICE Information Networks Research Award in 2003. Ms. Ishizuka is a member of the Operations Research Society of Japan.

Masaki Aida received his B.Sc. and M.Sc. in Theoretical Physics from St. Paul's University, Tokyo, in 1987 and 1989, and received his Ph.D. in Telecommunications Engineering from the University of Tokyo, in 1999. Since joining NTT Laboratories in 1989, he has been mainly engaged in research on traffic issues in ATM networks. From March 1998 to March 2001, he was a manager at the Traffic Research Center of NTT Advanced Technology Corporation (NTT-AT). He is currently a Senior Research Engineer at NTT Information Sharing Platform Laboratories. His current interests include traffic issues in communications systems. He received IEICE's Young Investigators Award in 1996. Dr. Aida is a member of the IEEE and the Operations Research Society of Japan.

Shin-ichi Kuribayashi received his B.S., M.S., and Ph.D. in Engineering from Tohoku University, Sendai, in 1978, 1980, and 1988 respectively. He joined NTT Electrical Communications Labs in 1980. He has been engaged in the design and development of DDX and ISDN packet switching, ATM, PHS, and IMT2000 and IP-VPN systems. He researched distributed communication systems at Stanford University from December 1988 through December 1989. He participated in international standardization on ATM signaling and IMT2000 signaling protocols at the ITU-T SG11 from 1990 through 2000. He is now responsible for next-generation IP systems work at NTT Information Sharing Platform Labs.