PAPER

A Study of Control Plane Stability with Retry Traffic: Comparison of Hard- and Soft-State Protocols

Masaki AIDA[†a)], **Chisa TAKANO**[†,††], ***Members***, **Masayuki MURATA**[†††], ***Fellow***, ***and*** **Makoto IMASE**[†††], ***Member***

SUMMARY Recently problems with commercial IP telephony systems have been reported one after another, in Japan. One of the important causes is congestion in the control plane. It has been recognized that with the current Internet it is important to control not only congestion caused by overload of the data plane but also congestion caused by overload of the control plane. In particular, "retry traffic," such as repeated attempts to set up a connection, tends to cause congestion. In general, users make repeated attempt to set up connections not only when the data plane is congested but also when the control plane in the network is overloaded. The latter is caused by user behavior: an increase in the waiting time for the processing of connection establishment to be completed tends to increase his or her initiation of reattempts. Thus, it is important to manage both data plane and control-plane resources effectively. In this paper, we focus on RSVP-based communication services including IP telephony, and introduce a model that takes account of both data-plane and control-plane systems, and we examine the behavior of retry traffic. In addition, we compare the system stability achieved by two different resource management methods, the hard-state method and the soft-state method.

key words: *IP telephony, VoIP, retry traffic, control plane, soft-state, stability*

1. Introduction

In the Internet, congestion due to overload of the control plane has become an important issue in addition to congestion due to overload of the data plane. In particular, "retry traffic," such as repeated attempts to set up a connection or reload a file, tends to cause congestion. Therefore, it is important to develop a mechanism to avoid congestion caused by retry traffic.

In general, retry traffic is generated by factors in both the data plane and the control plane as described below.

- Retry traffic generated due to a shortage of data-plane resources. A shortage of resources in the data plane causes requests to establish a connection or secure bandwidth to be discarded. This will induce retries.
- Retry traffic generated due to a shortage of control-plane resources. A shortage of processing resources causes the response time of the processing system to increase. This will induce impatient users to repeat their requests.

Since retry traffic itself consumes network resources, it is not appropriate to handle congestion on the data plane and that on the control plane separately. It is also important to manage network resources appropriately in order to prevent retry traffic from increasing.

Network resource management methods can be broadly classified into two categories:

- **Hard-state methods:** A network resource is secured or released at a user's explicit request.
- **Soft-state methods:** A network resource is secured at a user's explicit request but is released automatically after the expiry of a timer rather than at the user's explicit request. If the user wishes to retain a resource for a prolonged period, he or she needs to send a refresh message periodically to initialize the timer, thereby preventing the resource from being released.

In general, if requests from users can be relied on to arrive without fail, the hard-state method is more scalable than the soft-state method when faced with an increase in the number of user connections to be managed. This is because the hard-state method does not require refresh-messages to be sent. However, if requests from users do not necessarily arrive without fail, and thus the probability of failing to release a resource is not negligible, the balance between the overhead of sending refresh messages and the failure to manage resources properly determine which method is more advantageous, the hard-state method or the soft-state method. The soft-state method is widely used in the current Internet design. The advantages and disadvantages of the two methods have been evaluated from a variety of perspectives [1]–[3]. However, it is not a simple task to determine which is better because the outcome is influenced by the type of traffic and other conditions used in any comparison.

This paper investigates the stability issues in RSVP-based communication services including IP telephony, and discusses the properties of retry traffic caused by resource shortage in the data plane or the control plane. In order to take retry traffic generation into consideration, it is necessary to consider the interaction between users and the system. First, we introduce a quasi-static retry traffic model that describes the behavior of input traffic including retry traffic. Next, we demonstrate that under certain conditions a system

Manuscript received March 22, 2007.
Manuscript revised July 17, 2007.

[†]The authors are with the Graduate School of System Design, Tokyo Metropolitan University, Hino-shi, 191-0065 Japan.
[††]The author is with the Traffic Engineering Division, NTT Advanced Technology Corporation (NTT-AT), Musashino-shi, 180-0006 Japan.
[†††]The authors are with the Graduate School of Information Science and Technology, Osaka University, Suita-shi, 565-0871 Japan.
a) E-mail: maida@sd.tmu.ac.jp

DOI: 10.1093/ietcom/e91–b.2.437

can become unstable abruptly. We also indicate that network resource management is important in protecting the control plane from the proliferation of retry traffic, and that resource management based on the soft-state method offers effective and robust protection against changes in the network state.

This paper is organized as follows. In Sect. 2, we briefly summarize the related studies concerning retry traffic and control-plane models. Section 3 introduces a system model that takes account of retry traffic in both control and data planes. In addition, we propose a quasi-static retry traffic model for analyzing the stability of the model. Using the quasi-static retry traffic model, Sect. 4 investigates the stability of the system. Section 5 shows approximations of parameter values that concern the system stability. In Sect. 6, we compare both hard- and soft-state based resource management protocols, with respect to the stability of the system. Our conclusions are summarized in Sect. 7.

2. Related Studies

Studies dealing with retry traffic on an M/G/s/s retrial queue model are well known [4]–[6]. The expression, M/G/s/s, represents a model in which service requests arise in a Poisson manner, enter the system, receive service from one of s servers, and then leave the system, and in which service requests are discarded if all s servers are busy. An M/G/s/s retrial queue model represents an M/G/s/s model in which discarded service requests are stored in a retrial queue and re-enter the system after a certain elapsed time determined by an exponential distribution (Fig. 1). It is known that the stability of the M/G/s/s retrial queue model can be derived if the length of the retrial queue does not diverge [4], and

$$\frac{\lambda_0}{\mu} < s, \tag{1}$$

where λ_0 is the arrival rate of service requests, excluding retry requests, per unit time, and $1/\mu$ is the average service time.

Although the M/G/s/s retrial queue model incorporates retry traffic that arises due to a resource shortage on the data plane, it does not incorporate retry traffic that arises due to a resource shortage on the control plane.

Reference [7] points out that an increase in the complexity of call control processing due to the addition of new processing requirements, such as the determination of QoS, causes an increase in the load on the control plane, and discusses the properties of call processing delay. Although [7] addresses both the data and control planes, it does not take retry traffic into consideration.

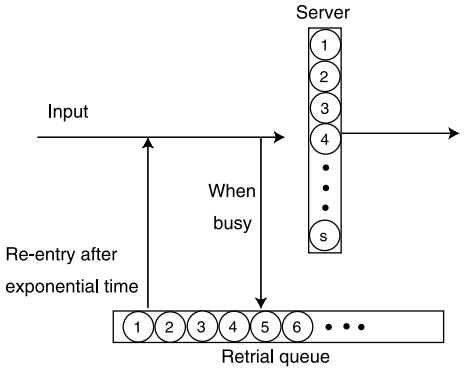

Fig. 1 M/G/s/s retrial queue model.

3. Model that Takes Account of Retry Traffic in Both the Control Plane and the Data Plane

This section describes a model for examining the properties of retry traffic that arises due to resource shortages in the data and control planes, and the method used to analyze these properties.

3.1 Basic Model for RSVP-Based Communications

In order to describe retry traffic that arises due to resource shortages in the data and control planes, we have developed a model that incorporates a serial combination of a data plane model and a control plane model (Fig. 2).

The data plane model describes the behavior of data transmission processing, such as securing link bandwidth, and so we use an M/G/s/s retrial queue model for it. If the system fails to secure bandwidth due to a resource shortage

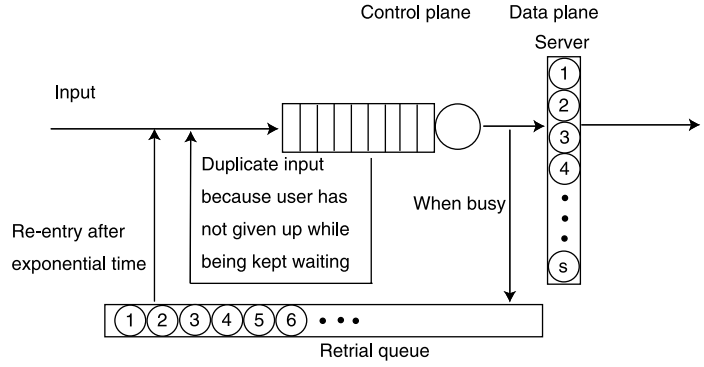

Fig. 2 Model incorporating retry traffic in both the control plane model (M/M/1) and the data plane model (M/G/s/s).

in the data plane, it means that all s servers are busy. In this case a service request is kept waiting in the retrial queue for a period of time (exponential), and then is re-entered into the system as a new request.

The control plane model describes the system behavior related to routing and processing of protocols, and so M/M/1 has been adopted for it. The expression, M/M/1, represents a model in which service requests arise in a Poisson manner, enter the system, receive service from one server with exponential service time, and then leave the system, and in which service requests are queued if the server is busy. The reasons why we apply M/M/1 to the control-plane model are that it allows simple and tractable analysis of the stability of the retry system. However, the validity of the M/M/1-based model needs justification. If the control-plane model describes session initiation processes of IP telephony, features of the input process and the use of a single server are acceptable assumptions. Although the accuracy of the M/M/1-based model with respect to the service time of the control plane model is unknown, the results obtained from this model will give a foundation for analyzing more complex models in future studies.

In addition to the ordinary M/M/1 model, some of the users who have been kept waiting due to increased processing time may attempt to initiate a new service request. Therefore, we have modified this model slightly to take account of retry traffic that will arise depending on the length of the queue in the M/M/1 model. Since such retry traffic will be generated without the original service requests being cancelled, our model assumes that new service requests arise without any of the service requests waiting in the queue in the M/M/1 model being withdrawn.

Note that we assume the input to the data plane follows a Poisson process. If there is no retry traffic, this assumption is true because M/M/1 is chosen for the control-plane model. This is because service requests that leaves the control plane and enter in the data plane follow a Poisson process. As shown in the subsequent sections, we also model the input of retry traffic as a Poisson process. Thus, the input to the data plane follows a Poisson process even when there is retry traffic.

3.2 Quasi-Static Retry Traffic Model

This subsection considers how retry traffic arises from the control-plane system shown in Fig. 2. If retry traffic that is dependent on the length of the queue in the control plane arises, one possible case is that retry traffic volume is proportional to the length of the queue in the control plane. If we assume that service requests waiting in the queue generate retry traffic at a certain rate ϵ, a diagram depicting the state transition rate with respect to the queue length of the control-plane system is as shown in Fig. 3. In this figure, λ_0 is the arrival rate of service requests, excluding retry traffic, per unit time, while $1/\eta$ is the average service time of the control plane system. For this system to have a steady-state probability, the infinite sum on the right-hand side of

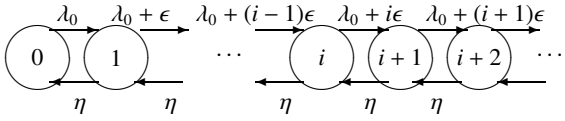

Fig. 3 State transition diagram for the case where retry traffic is generated in proportion to the queue length of the control-plane system.

the following equation must be finite.

$$p_0 = \left[1 + \sum_{i=1}^{\infty} \prod_{j=0}^{i} \left(\frac{\lambda_0 + j\epsilon}{\eta}\right)\right]^{-1}.$$

Therefore, if $\epsilon > 0$, the system is unstable. Since an increase in retry traffic does not result in the divergence of the waiting time of an actual control plane under normal operating conditions, we can conclude that a model in which retry traffic is generated in proportion to the queue length of the control plane is not realistic.

In general, state transitions of the control-plane system occur on a timescale much shorter than humans can perceive. It is natural to assume that the reason that users grow impatient and initiate repeat reattempts to send service requests is not in response to the queue length at the present time but in response to the average waiting time that occurs on a longer timescale, a timescale long enough for humans to perceive. Since the rate of input of retry traffic by users changes very slowly on such a longer timescale, the change of the mean of system behavior occurs slowly on a timescale long enough for humans to perceive. Let T be the minimum timescale perceptible to humans. We assume that the control plane system is in a steady state on a timescale smaller than T, and that retry traffic affects the system on a timescale larger than T. In the following, we assume a quasi-static steady state for the generation of retry traffic from the control plane. The key assumptions are as follows:

- The system can be assumed to be in a steady state on a timescale shorter than T.
- Changes in the system are observed at discrete times occurring at an interval of T.
- Retry traffic from the system at a certain time, $t = k$, is determined by the steady-state probability of the system at $t = k - 1$. (More specifically, retry traffic from the control-plane system is proportional to the average queue length at $t = k - 1$, and that from the data-plane system is proportional to the loss ratio at $t = k - 1$.)

3.3 Value of Timescale T

Since this paper deals with retry traffic generated as users respond to an increase in waiting time, we treat the behavior of the control-plane system occurring on a timescale shorter than T separately, and consider the system as being in a quasi-static steady state on such a timescale. To do so, it is necessary to determine the appropriate value of T. The value of T must satisfy the following requirements:

- T must be within a timescale in which humans can actually perceive an increase in the waiting time for their requests to be processed.
- T must be sufficiently longer than the timescale in which state transitions occur in the system so that it is possible to assume that the system is in a steady state in a timescale shorter than T.

It is well known that the tolerable waiting time (i.e., the maximum length of time for which a user's attention can be retained) for a webpage to be displayed completely is 8 seconds. To satisfy this so-called 8-second rule, many webpages are so designed that they can be displayed completely within 8 seconds. Measurement of tolerable waiting time in actual experiments have verified that a user's interest drifts away after 10 seconds or so [8]. It is possible that the maximum tolerable waiting time for a user in our case related to retry traffic is shorter than in the case of webpage browsing because our case may involve urgent service requests, such as attempts to make reservations for popular services. Reference [9] classifies the timescales according to how humans perceive them as follows:

1. A delay of 0.1 second is perceived as instantaneous access.
2. A delay of 1 second is the limit for the users' flow of thought to stay uninterrupted.
3. A delay of 10 seconds is the limit for retaining the users' attention/focus on the dialogue.

The last category coincides fairly well with the 8-second rule for websites in terms of both the timescale and the reason. Since the generation of retry traffic seems to correspond to the second category, one second should be a reasonable value for T.

4. Input of Traffic (Including Retry Traffic) and Stability

The properties of input traffic and the system's stability are examined here using a quasi-static model of the system, a model that takes account of retry traffic caused by resource shortages in both the control plane and the data plane in combination.

Details of the model are as follows. Let $\lambda_0 > 0$ be the traffic input rate without retry traffic. We assume that $\lambda_0 > 0$. Changes in the state of the control-plane system are examined at discrete intervals T. We assume that the retry traffic at time $t = k$ is determined by the steady-state probability of the system at time $t = k - 1$. We assume that users whose service requests have not been dealt with by any server of the data-plane system keep sending service requests until they are finally processed, and that users kept waiting for their requests to be processed by the control-plane system send service requests at a rate proportional to the average queue length.

Let λ_k be the input load, including retry traffic, at time k. We assume that the input load at time $k + 1$ is

$$\lambda_{k+1} = \lambda_0 + \lambda_k B(\rho_k, s) + \epsilon \frac{\rho_k/a}{1 - \rho_k/a}, \tag{2}$$

where $\rho_k = \lambda_k/\mu$, $1/\mu$ is the average service time of the data-plane system, a is the ratio of $1/\mu$ to the processing time of the control-plane server, i.e., the ratio between the relative processing powers of the two servers ($a = \eta/\mu$ where $1/\eta$ is the average service time of the control-plane system), and ϵ is a positive constant indicating the intensity of the retry traffic generated due to a control-plane resource shortage. $B(x, s)$ is an Erlang B formula, i.e.,

$$B(x, s) = \frac{\dfrac{x^s}{s!}}{1 + x + \dfrac{x^2}{2!} + \cdots + \dfrac{x^s}{s!}}, \tag{3}$$

for $x \geq 0$. The structure of Eq. (2) is as follows. The first term on the right-hand side is the input rate of original traffic (excluding retry traffic). The second term corresponds to the rate of retries from the data-plane system, and denotes the number of losses occurring at M/G/s/s per unit time at time k. The last term corresponds to the rate of retries from the control-plane system. The quantity $(\rho_k/a)/(1 - (\rho_k/a))$ denotes the average number of service requests in the M/M/1 system.

Next, we will consider the stability of the system. We assume that the requirement for the system to be stable is that the volume of traffic, including retry traffic, does not diverge after a sufficient period of elapsed time. Namely,

$$\lim_{k\to\infty} \lambda_k < \infty. \tag{4}$$

This can be verified by defining two functions of λ, $f(\lambda)$ and $g(\lambda)$ as follows, and examining whether they intersect.

$$f(\lambda) = \lambda_0 + \lambda B(\rho, s) + \epsilon \frac{\rho/a}{1 - \rho/a}, \tag{5}$$

$$g(\lambda) = \lambda, \tag{6}$$

where $\rho = \lambda/\mu$. Since $\lambda_0 > 0$, $f(0) > g(0)$. The relationship between $f(\lambda)$ and $g(\lambda)$ in some typical cases is as shown in Fig. 4. If the input traffic at a certain time is λ, the input traffic after one unit time becomes $\{g^{-1} \circ f\}(\lambda)$, followed by $\{g^{-1} \circ f\}^2(\lambda)$, etc.. In general, the input traffic after an elapse of n units of times becomes $\{g^{-1} \circ f\}^n(\lambda)$. As shown in the left-hand chart in Fig. 4, if $f(\lambda)$ and $g(\lambda)$ do not intersect, $\lim_{n\to\infty}\{g^{-1} \circ f\}^n(\lambda) = \infty$. If, on the other hand, the two functions intersect as shown in the middle chart, and if $\lim_{\lambda\to\infty} f(\lambda)/g(\lambda) < 1$, then the system is stable. If the two functions intersect in the manner shown in the right-hand chart of Fig. 4, and if $\lim_{\lambda\to\infty} f(\lambda)/g(\lambda) > 1$, then the system is stable for all λ to the left of the right-hand intersection.

If the retry traffic from the control-plane system is negligible ($\epsilon = 0$ or $a = \infty$), $f(\lambda) = \lambda_0 + \lambda B(\rho, s)$. Considering that $\lambda_0 > 0$ and $B(\rho, s) < 1$, $f(\lambda)$ and $g(\lambda)$ intersect at one point if Eq. (1) is satisfied, and do not intersect at all if Eq. (1) is not satisfied. If, on the other hand, the retry traffic from the control-plane system is not negligible ($\epsilon > 0$

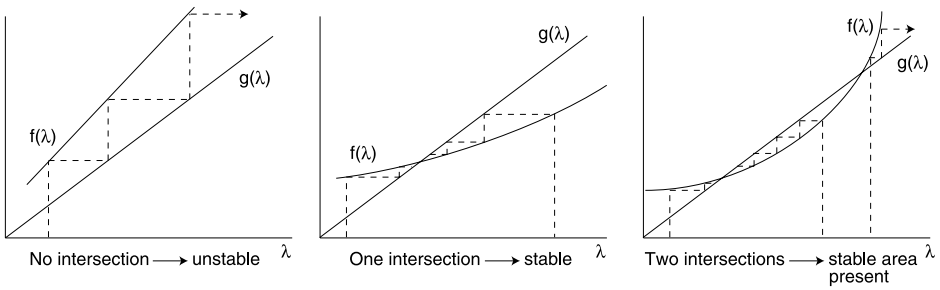

Fig. 4 System stability depends on the presence of intersections between $f(\lambda)$ and $g(\lambda)$.

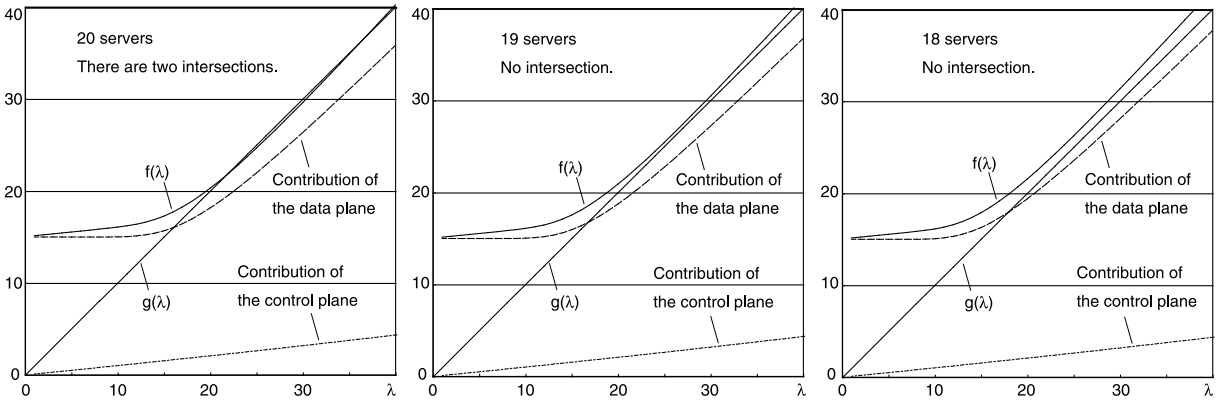

Fig. 5 Relationship between $f(\lambda)$ and $g(\lambda)$ for different number of servers.

and $a < \infty$), the third term on the right-hand side of Eq. (5) always yields $\lim_{\lambda\to\infty} f(\lambda)/g(\lambda) > 1$. Therefore, $f(\lambda)$ and $g(\lambda)$ may intersect at two points or at one point (a point of contact), or may not intersect at all.

Next, let us look at system stability using examples with specific values. We will consider the case where $\lambda_0 = 15$, $\mu = 1$, $\epsilon = 50$, and $a = 500$. The behavior of $f(\lambda)$ and $g(\lambda)$ for $s = 20$, 19, and 18 servers is shown in Fig. 5. In order to consider the influence of retry traffic on the control plane and the data plane separately, Fig. 5 shows separately the first and the second terms (original traffic and contribution of the data plane), and the third term (contribution of the control plane) on the right-hand side of Eq. (5), in addition to the value of $f(\lambda)$. In these evaluation conditions, $\lambda_0/\mu < s$ always holds, satisfying Eq. (1). Consequently, the data-plane system is stable against the presence of retry traffic, i.e. the curve representing the contribution of the data plane and $g(\lambda)$ intersect at one point. Therefore, as long as the influence of the control-plane system is negligible, the system is stable even against the influence of retry traffic provided that a sufficient number of servers is provided to handle the demand, λ_0, of the input traffic. If, on the other hand, the contribution of the control plane is not negligible, the system is stable if the number of servers is 20 but not stable if it is 19 or 18, as can be determined from the presence or absence of intersections in Fig. 5. Thus, we can conclude that if the retry traffic from the control plane is not negligible, the system may not be stable against retry traffic even when the number of servers provided is sufficient to satisfy

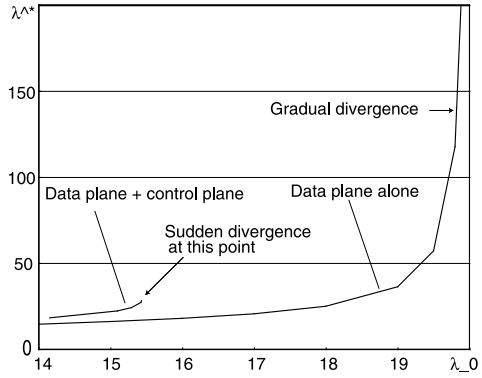

Fig. 6 Change in λ^* as λ_0 is increased.

Eq. (1) for input traffic demand λ_0.

Let λ^* be the input traffic after a sufficient time has elapsed as follows:

$$\lambda^* := \lim_{k\to\infty} \lambda_k. \tag{7}$$

Let us consider that Fig. 5 shows a case where the number of servers is reduced from 20 to 19. When $s = 20$, the system is stable at $\lambda^* \simeq 22$. However, when $s = 19$, the intersection disappears to make $\lambda^* = \infty$. Thus, the system suddenly becomes unstable. Such a change in stability also occurs when other parameters, such as λ_0, are gradually increased. Figure 6 shows how the value of λ^* changes as λ_0 is increased, for the case where $s = 20$. For comparison, the value of λ^* for the data plane alone is also shown (i.e., the processing

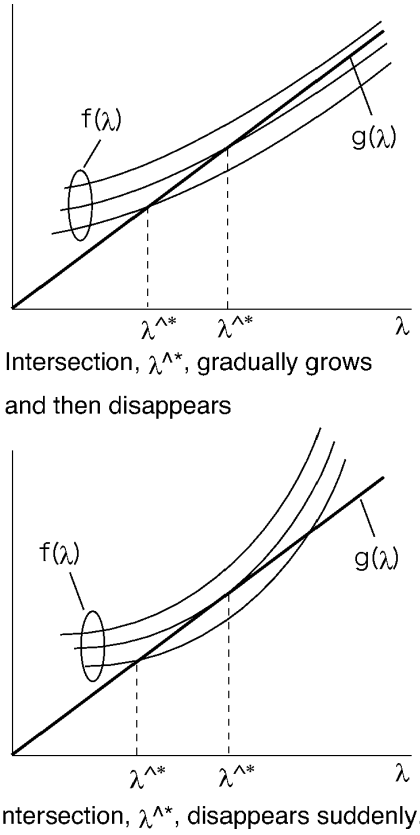

Fig. 7 An intersection is present when retry traffic from the control plane is absent (upper figure), but disappears when it is present (lower figure).

power of the control-plane system is made such that $\eta = \infty$). If we consider the data plane alone, λ^* increases as λ_0 approaches $s = 20$, and the system is stable within the range where Eq. (1) is satisfied. On the other hand, if we take the contribution of the control plane into consideration, the early warning sign as λ^* gradually grows and diverges does not occur, and when the intersection disappears, the system suddenly becomes unstable with $\lambda^* = \infty$.

Figure 7 explains why this sudden change occurs. It shows that in the absence of retry traffic from the control plane, $f(\lambda)$ and $g(\lambda)$ intersect at one point at most, and that when the intersection disappears, $\lambda^* \to \infty$. In the presence of retry traffic from the control plane, as λ_0 is altered from low to high, $f(\lambda)$ and $g(\lambda)$ intersect at two points, and then at one point, and finally do not intersect at all. When the intersection disappears, any prior signs of the phenomenon of $\lambda^* \to \infty$ is not observed.

Conversely, let us consider the case where an intersection between $f(\lambda)$ and $g(\lambda)$ reappears after once it disappears. In the absence of retry traffic from the control plane, $f(\lambda)$ and $g(\lambda)$ intersect at one point. Therefore, even if λ increases while there is no intersection, the system regains its stability when an intersection reappears. On the other hand, in the presence of retry traffic from the control plane, the behavior of the system is different from the former case. If λ increases, while there is no intersection, and exceeds the value of the right-hand intersection, the system does not regain its stability even when an intersection reappears. Therefore, if the system is to be kept stable in the presence of retry traffic from the control plane, it is important to ensure that the intersection between $f(\lambda)$ and $g(\lambda)$ does not disappear.

From the above discussion we can conclude that if control-plane resources are not fully usable for one reason or another, so reducing their availability, the system may suddenly become unstable, and that once this instability occurs, the system may not regain its stability even after control-plane resources have been fully restored. Therefore, to ensure the stability of the system, it is extremely important to manage control-plane resources properly.

5. Design of System Stability

As shown in the previous section, the value of the right-hand intersection of the curves $f(\lambda)$ and $g(\lambda)$ is important in ensuring the stability of the system. First, we give an approximation of the value of the right-hand intersection. If there are two intersections of $f(\lambda)$ and $g(\lambda)$, then

$$f(\lambda) = \lambda_0 + \lambda\, B(\rho,\, s) + \epsilon\, \frac{\rho/a}{1-\rho/a} = \lambda \tag{8}$$

has two solutions. One is λ^* and let the other be $\tilde{\lambda}$, which is the value of the right-hand intersection and $\lambda^* < \tilde{\lambda}$.

The contribution of the data-plane retries in Eq. (5), the second term on the right-hand side, increases linearly with λ for $\lambda \gg 1$. Using this property, we have

$$\lambda\, B(\rho,\, s) \simeq \lambda - s\mu, \quad (\lambda \gg 1). \tag{9}$$

This approximation means the proportion of input traffic which exceeds the server capacity, given by $(\lambda - s\mu)$, is all lost. Then, $\tilde{\lambda}$ should satisfy

$$\lambda_0 + \tilde{\lambda} - s\mu + \epsilon\, \frac{\tilde{\lambda}/(a\mu)}{1-\tilde{\lambda}/(a\mu)} \simeq \tilde{\lambda}, \tag{10}$$

and we have

$$\tilde{\lambda} \simeq a\,\mu\, \frac{s\mu - \lambda_0}{s\mu - \lambda_0 + \epsilon}, \quad (s\mu > \lambda_0). \tag{11}$$

To demonstrate the accuracy of the approximation, we show numerical examples. Comparisons between the exact value and the approximation of $\tilde{\lambda}$ are shown in Figs. 8 and 9, and Tables 1 and 2. Figure 8 and Table 1 show an example, comparing the exact and approximate values of $\tilde{\lambda}$ with respect to λ_0, and Fig. 9 and Table 2 show a similar comparison for $\tilde{\lambda}$ with respect to a. These examples illustrate that Eq. (11) gives a good approximation.

Next, we consider a design issue with the condition that $\tilde{\lambda} \geq c\,\lambda_0$, that is,

$$\tilde{\lambda} = a\,\mu\, \frac{s\mu - \lambda_0}{s\mu - \lambda_0 + \epsilon} \geq c\,\lambda_0, \tag{12}$$

where c is a positive constant. From this inequality, we have

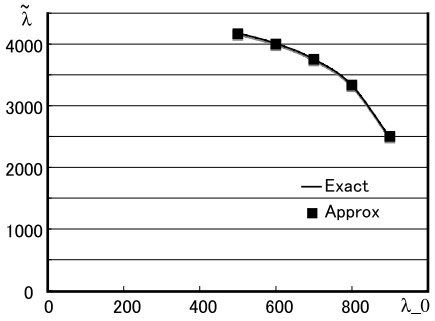

Fig. 8 Approximated $\tilde{\lambda}$ with respect to λ_0.

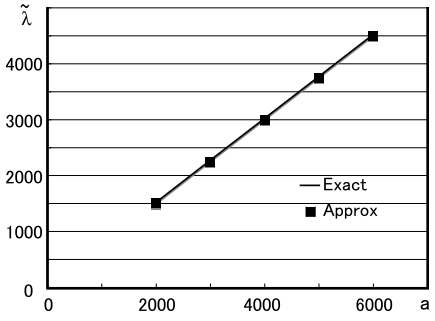

Fig. 9 Approximated $\tilde{\lambda}$ with respect to a.

Table 1 Approximated $\tilde{\lambda}$ with respect to λ_0.

a	s	μ	λ_0	ϵ	Exact $\tilde{\lambda}$	Approx. $\tilde{\lambda}$
5000	1000	1	900	100	2491	2500
5000	1000	1	800	100	3331	3333
5000	1000	1	700	100	3749	3750
5000	1000	1	600	100	3999	4000
5000	1000	1	500	100	4165	4166

Table 2 Approximated $\tilde{\lambda}$ with respect to a.

a	s	μ	λ_0	ϵ	Exact $\tilde{\lambda}$	Approx. $\tilde{\lambda}$
2000	1000	1	700	100	1497	1500
3000	1000	1	700	100	2248	2250
4000	1000	1	700	100	2999	3000
5000	1000	1	700	100	3749	3750
6000	1000	1	700	100	4499	4500

two conditions

$$a \geq \frac{c\lambda_0}{\mu} \cdot \frac{s\mu - \lambda_0 + \epsilon}{s\mu - \lambda_0}, \quad (\text{if } s\mu > \lambda_0), \tag{13}$$

$$s \geq \frac{\lambda_0}{\mu} \cdot \frac{a\mu - c\lambda_0 + c\epsilon}{a\mu - c\lambda_0}, \quad (\text{if } a\mu > c\lambda_0). \tag{14}$$

Inequality (13) is a condition for the design of the control-plane system and $s\mu > \lambda_0$ is the stability condition for the data-plane system. Similarly, inequality (14) is a condition for the design of the data-plane system and $a\mu > c\lambda_0$ is the stability condition for the control-plane system.

6. Comparison of Soft-State and Hard-State Methods for Managing Control-Plane Resources

As we have confirmed in the previous sections, in a model that takes the retry traffic from both data and control planes into consideration, the system can suddenly become unstable as a result of changes in traffic or network resources. A safeguard that can be taken on the network side to avoid, as far as possible, such a sudden degeneration into instability as far as possible is to manage the control-plane resources properly. This section compares two different methods of managing control-plane resources, namely the hard-state and the soft-state methods. It then demonstrates that the soft-state method is more robust in withstanding changes in the network state.

The two methods are described below.

- **Hard-state method**: In this method, data-plane resources are released at the request of a user. If a resource release request from a user does not arrive due to a problem in the network (e.g., loss of the resource release request packet), the system administrator releases the resource after a specified time (human recovery). In our evaluation, we assume that the system administrator releases the resource every 7,200 seconds (2 hours).
- **Soft-state method**: In this method, a data-plane resource is released automatically after the expiry of a timer. If the user wants to retain the data-plane resource for a period longer than the timeout period, he or she needs to send a refresh message to reset the timer. In our evaluation, the data-plane resource is automatically released two seconds after it was seized or after the arrival of the last refresh message (automatic recovery).

In addition to the above, we will also evaluate the following methods:

- **Ideal type**: The operation of this method is the same as that of the hard-state method except that it is assumed that release requests from users will always arrive without any problem — an ideal situation, as the name of the method implies.
- **Compound type**: The operation of this method is the same as that of the soft-state method except that a resource can be released by explicit resource release requests from users in addition to timeouts.

We derive the concrete expression of $f(\lambda)$ for each method. Let p be the probability of loss of resource release request packets. $f_x(\lambda)$ represents $f(\lambda)$ for each method x, that is, $f_{\mathrm{h}}(\lambda)$, $f_{\mathrm{s}}(\lambda)$, $f_{\mathrm{i}}(\lambda)$, and $f_{\mathrm{c}}(\lambda)$ represent $f(\lambda)$ for the hard-state, soft-state, ideal, and compound methods, respectively. Since all the methods are ways of managing data-plane resources, the differences between the various functions $f_x(\lambda)$ with respect to method x, relate only to the holding time of the data-plane resources, and we have

$$f_x(\lambda) = \lambda_0 + \lambda\, B(\rho_x,\, s) + \epsilon\, \frac{\rho/a}{1 - \rho/a}, \tag{15}$$

where ρ_x denotes ρ for a method x ($x \in \{\mathrm{h},\, \mathrm{s},\, \mathrm{i},\, \mathrm{c}\}$).

For the hard-state method, if a resource release request is lost, the system administrator releases the resource at the

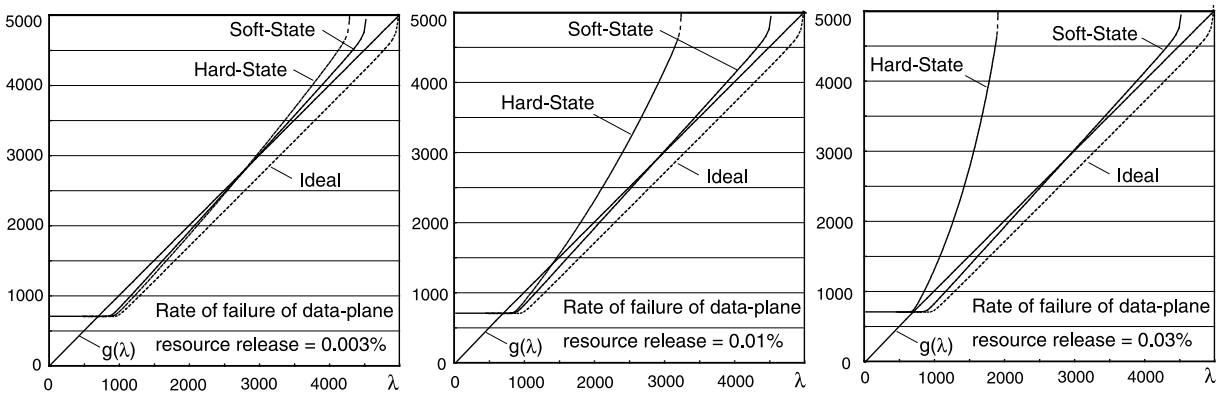

Fig. 10 Comparison of system stability for different data-plane resource management methods (soft-state method, hard-state method and ideal/compound methods).

next instance of the periodic manual (human) recovery operation. If the interval of the manual recovery operations is d, the average waiting time until the next manual recovery is $d/2$. So, the average holding time $(1/\mu_{\rm h})$ of data-plane resources increases by $d/2$ if a resource release request is lost. That is,

$$\frac{1}{\mu_{\rm h}} = (1-p)\frac{1}{\mu} + p\left(\frac{1}{\mu} + \frac{d}{2}\right),$$

and we obtain $f_{\rm h}(\lambda)$ by using $\rho_{\rm h} = \lambda/\mu_{\rm h}$ and Eq. (15).

For the soft-state method, data-plane resources are always released by timeout. If the interval of automatic recovery operations by timeout is τ, the average waiting time until the next automatic recovery is $\tau/2$. So, the average holding time $(1/\mu_{\rm s})$ of data-plane resources is always increased by $\tau/2$. That is,

$$\frac{1}{\mu_{\rm s}} = \frac{1}{\mu} + \frac{\tau}{2},$$

and we obtain $f_{\rm s}(\lambda)$ by using $\rho_{\rm s} = \lambda/\mu_{\rm s}$ and Eq. (15).

For the ideal type, since the probability of loss of resource release request is $p = 0$, we have

$$\frac{1}{\mu_{\rm i}} = \frac{1}{\mu},$$

and we obtain $f_{\rm i}(\lambda) = f(\lambda)$ and $\rho_{\rm i} = \rho$. Similarly, for the compound type, we have

$$\frac{1}{\mu_{\rm c}} = (1-p)\frac{1}{\mu} + p\left(\frac{1}{\mu} + \frac{\tau}{2}\right),$$

and we obtain $f_{\rm c}(\lambda)$ by using $\rho_{\rm c} = \lambda/\mu_{\rm c}$.

Figure 10 shows the calculated results for $f(\lambda)$ and $g(\lambda)$ for cases where the number of servers is 1000 (i.e., $s = 1000$). We assumed that $\lambda_0 = 700$, $\mu = 1$, $\epsilon = 1$, and $a = 5000$. In this specific example, the average connection holding time was 10 s, and the average time it took to set up a connection was 2 ms. The three charts in Fig. 10 are for three values (0.003%, 0.01%, and 0.03%) of the probability of failing to release the data-plane resource as a result of resource release requests not being delivered correctly. Since the results for the ideal type and the compound type coincide almost completely, the figure shows only the result for the ideal type.

The hard-state method keeps the system stable when the probability of failing to release the data-plane resource is low, but leaves the system less stable when the probability is higher, so the system stability is sensitive to changes in this probability. On the other hand, with the soft-state method and the ideal type, the system is stable and not sensitive to changes in the release failure probability. Therefore, we can conclude that it is advisable for resources to be managed using the soft-state method in order to avoid the occurrence of undesirable situations where the system becomes unstable due to a shortage in effective data-plane resources resulting from failure to release resources.

We also find that, by adding an explicit resource-releasing process to the mechanism of the soft-state method, it is possible to achieve a high stability, comparable to that of the ideal type irrespective of the network state.

7. Conclusions

This paper has considered congestion caused by overloads not only in the data plane but also in the control plane. In particular, we have evaluated methods of managing data-plane resources in order to avoid congestion caused by retry traffic resulting from a shortage of both data-plane and control-plane resources.

In order to describe retry traffic resulting from a shortage of both data-plane and control-plane resources, we have developed a model that combines the M/M/1 and M/G/s/s models. We have examined the stability of the system by checking whether retry traffic diverges. In order to describe users' tendency to generate retry traffic when their waiting time increases due to control-plane system issues, we have adopted a quasi-static traffic model in which the system state is considered to be steady below a certain appropriate timescale, and begins to change gradually on a timescale perceptible to humans.

Examination using the above approach has revealed that when retry traffic resulting from a shortage of control-

plane resources is considered, the system state can change more abruptly than when only retry traffic resulting from a shortage in data-plane resources is considered. In fact, the system can suddenly become unstable if the system conditions change only slightly, without any prior signs of warning, such as an increase in retry traffic.

To avoid having the system become unstable in this manner, it is important to manage data-plane resources appropriately. We have examined how two different methods of data-plane resource management, the hard-state method and the soft-state method, affect system stability. It has been shown that use of the soft-state method makes the system stable and robust against changes in the network state.

Since our primary goal so far has been to examine the basic characteristics of retry traffic, we have evaluated models using a relatively small number of servers. We plan to continue the evaluation with a more realistic number of servers and with the detailed specification of parameters and other conditions. We will also study system design methods that will strike a good balance between the data-plane and the control-plane systems.

Acknowledgements

A part of this research was made possible by the Grant-in-Aid for Scientific Research (A) No. 16200003 (2004–2007) from the Japan Society for the Promotion of Science.

References

[1] P. Ji, Z. Ge, J. Kurose, and D. Towsley, "A comparison of hard-state and soft-state signaling protocols," IEEE/ACM Trans. Netw., vol.15, pp.281–294, April 2007.
[2] J.C.S. Lui, V. Misra, and D. Rubenstein, "On the robustness of soft state protocols," ICNP 2004, pp.50–60, 2004.
[3] T. Yamada, T. Tada, and M. Imase, "Evaluation of the robustness of soft-state for data management in a distributed environment," IEICE Technical Report, IN2005-111, pp.7–12, 2005.
[4] G.I. Falin and J.G.C. Templeton, Retrial Queues, Chapman & Hall, London, 1997.
[5] H. Alshaer and E. Horlait, "The joint distribution of server state and queue length of M/M/1/1 retrial queue with abandonment and feedback," 8th International Symposium on DSP and Communication System, Australia, Oct. 2005.
[6] W. Shang, L. Liu, and Q.-L. Li, "Tail asymptotics for the queue length in an M/G/1 retrial queue," Queueing Systems, vol.52, pp.193–198, 2006.
[7] R.-H. Hwang, C.-Yi, J.F. Kurose, and D. Towsley, "On-call processing delay in high speed networks," IEEE/ACM Trans. Netw., vol.3, no.6, pp.628–639, Dec. 1995.
[8] F.F.-H. Nah, "A study on tolerable waiting time: How long are Web users willing to wait?," Behaviour & Information Technology, vol.23, no.3, pp.153–163, May 2004.
[9] J. Nielsen, "Response times: The three important limits," Excerpt from Chapter 5 of Usability Engineering by J. Nielsen, Academic Press, 1993. Available at http://www.useit.com/papers/responsetime.html

Masaki Aida received his B.S. and M.S. in Theoretical Physics from St. Paul's University, Tokyo, Japan, in 1987 and 1989, respectively, and received the Ph.D. in Telecommunications Engineering from the University of Tokyo, Japan, in 1999. In April 1989, he joined NTT Laboratories. From April 2005 to March 2007, he was an Associate Professor at the Faculty of System Design, Tokyo Metropolitan University. He has been a Professor of the Graduate School of System Design, Tokyo Metropolitan University since April 2007. His current interests include traffic issues in computer communication networks. He received the Young Researchers' Award of the IEICE in 1996. He is a member of the IEEE and the Operations Research Society of Japan.

Chisa Takano received the B.E. degree in Telecommunication Engineering from Osaka University, Japan, in 2000. In 2000 she joined the Traffic Research Center, NTT Advanced Technology Corporation (NTT-AT). She has been engaged in research and development of computer networks. She received the IEICE's Young Researchers' Award in 2003.

Masayuki Murata received the M.E. and D.E. degrees in Information and Computer Sciences from Osaka University, Japan, in 1984 and 1988, respectively. In April 1984, he joined IBM Japan's Tokyo Research Laboratory, as a Researcher. From September 1987 to January 1989, he was an Assistant Professor with the Computation Center, Osaka University. In February 1989, he moved to the Department of Information and Computer Sciences, Faculty of Engineering Science, Osaka University. In April 1999, he became a Professor of Osaka University, and since April 2004, he has been with the Graduate School of Information Science and Technology, Osaka University. He has contributed more than four hundred and fifty papers to international and domestic journals and conferences. His research interests include computer communication networks and performance modeling and evaluation. He is a member of the IEEE, ACM, the Internet Society, and IPSJ.

Makoto Imase received his B.E. and M.E. degrees in Information Engineering from Osaka University in 1975 and 1977, respectively. He received his D.E. degree from Osaka University in 1986. From 1977 to 2001, he worked for Nippon Telegraph and Telephone Corporation (NTT). Since 2002 he has been a Professor of the Graduate School of Information Science and Technology at Osaka University. His research interests are in the area of information networks, distributed systems and graph theory. He is a member of IPSJ and JSIAM.