PAPER

Mutual Complementarity between Diffusion-Type Flow Control and TCP

Chisa TAKANO[†,††a)], Kaori MURANAKA[†], *Members*, Keita SUGIYAMA[††], *Student Member*, *and* Masaki AIDA[††], *Member*

SUMMARY In current IP-based networks, the application of window-based end-to-end flow control, including TCP, to ensure reliable flows is an essential factor. However, since such a flow control is provided by the end hosts, end-to-end control cannot be applied to decision-making in a time-scale shorter than the round-trip delay. We have previously proposed a diffusion-type flow control mechanism to realize the extremely time sensitive flow control that is required for high-speed networks. In this mechanism, each network node manages its own traffic only on the basis of the local information directly available to it, by using predetermined rules. The implementation of decision-making at each node can lead to optimal performance for the whole network. Our previous studies showed that the mechanism works well, by itself, in high-speed networks. However, to apply this mechanism to actual networks, it needs to be able to coexist with other existing protocols. In this paper, we investigate the performance of diffusion-type flow control coexisting with TCP. We show that diffusion-type flow control can coexist with TCP and the two can be complementary. Then, we show that a combination of both controls achieves higher network performance than TCP alone in high-speed networks.

key words: *flow control, autonomous decentralized control, diffusion equation, high-speed networks*

1. Introduction

The rapid spread of the Internet will necessitate the construction of higher-speed backbone networks in the near future. In a high-speed network, it is impossible to implement time-sensitive control based on collecting global information about the whole network because the state of a node varies rapidly with time, and is dependent on its processing speed, even if the propagation delay is identical with that in low-speed networks. We have previously proposed a diffusion-type flow control (DFC) mechanism as a solution for the severely time-sensitive flow control that is required for high-speed networks [1]–[3].

The different control mechanisms in networks can be classified from the point of view of their particular time-scale of control operations. Figure 1 shows the mutual relationship of different types of control according to such a classification. They form a layered structure with respect to time-scale. For example, routing and call admission control fall into the long and medium time-scales, respectively.

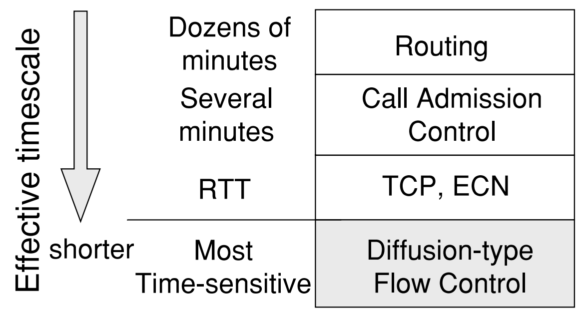


Fig. 1 Classification of various control mechanisms with respect to their effective time-scales.

Individual control mechanisms work well for their appropriate time-scales and they cooperate with each other. An end-to-end control such as TCP acts on the time-scale of the round-trip delay time (RTT). In high-speed networks, since a lot of packets are in transit on links, the delay in applying control greatly affects the network performance. However, since end hosts provide the flow control, TCP cannot be applied to decision-making in a time-scale shorter than the RTT. The target of DFC is a time-scale shorter than the RTT.

To overcome the inefficiency of TCP in high-speed networks, eXplicit Control Protocol (XCP) has been proposed [4]. This protocol is a generalization of Explicit Congestion Notification (ECN). By notifying information about a router to an end host, XCP can stabilize TCP performance. However, XCP is an end-to-node control and the notification of router information to an end host experiences delay proportional to the RTT. Therefore, TCP assisted by XCP, cannot be applied to decision-making in a time-scale very much shorter than the RTT.

Most research into flow control mechanisms mainly focuses on optimization problems [5]–[9]. These studies do not address the requirement of time-sensitivity in network control. The principles adopted for time-sensitive control are inevitably those of autonomous decentralized systems [10]. DFC is a solution for the extremely time-sensitive flow control required for high-speed networks, and is designed so that can satisfy the following requirements:

- It must be possible to collect the information required for the control method.
- The control should take effect immediately.

In DFC, by using predetermined rules, each node in a net-

Manuscript received January 23, 2006.
Manuscript revised April 12, 2006.

[†]The authors are with Traffic Engineering Division, NTT Advanced Technology Corporation (NTT-AT), Musashino-shi, 180-0006 Japan.
[††]The authors are with the Graduate School of System Design, Tokyo Metropolitan University, Hino-shi, 191-0065 Japan.
a) E-mail: chisa.takano@ntt-at.co.jp

DOI: 10.1093/ietcom/e89–b.10.2850

work manages its local traffic flow on the basis of only the local information directly available to it. In addition, the implementation of decision-making at each node can lead to optimal performance for the whole network. That is, the state of the whole network is controlled indirectly through the autonomous action of each node.

Decentralized flow control by end hosts, including TCP, is widely used in current networks, and there has been a lot of research in this area [8], [9], [11]. However, since end-to-end or end-to-node control cannot be applied to decision-making on a time-scale shorter than the round-trip delay, it is not capable of supporting decision-making on a very short time-scale. In low-speed networks, a control delay of the order of the RTT has a negligible effect on the network performance. However, in high-speed networks, the control delay greatly affects the network performance. This is because, although the RTT is itself unchanged, it becomes larger relative to the unit of time determined by the node's processing speed. This means that nodes in high-speed networks experience a larger RTT relative to the processing speed, and this causes an increase in the sensitivity to control delay. To achieve rapid control in a time-scale shorter than the RTT, it is preferable to apply control by the nodes rather than by the end hosts (see Fig. 2).

Let us consider the situation where the RTT is 100 ms when a network is congested. The upper graph in Fig. 2 shows the relationship between the speed of the network and the number of packets influenced by the control delay, when flow control by end hosts is applied. If the network speed is 10 Mbps, the number of packets influenced by control delay from an end host is only a few hundred. However, if the network speed is 100 Gbps, the number of packets is several million. Even though the RTT is unchanged, the increase in the network speed has a severe influence on network performance. If we apply node-by-node control (the lower graph in Fig. 2), the control delay is reduced typically by a factor of 2000 compared to end host control.

We have evaluated the performance of DFC in a high-speed network environment [1]–[3]. In particular, DFC for an end-to-end flow has been evaluated in [3]. Since the previous studies have not introduced window flow control such as TCP as a means of end-to-end flow control, we should verify the compatibility and complementarity of DFC with TCP.

In this paper, we investigate the performance of DFC coexisting with TCP. We show that DFC can coexist with TCP and the two can complement each other. In addition, we show that the coexistence of both controls achieves higher network performance than TCP alone in high-speed networks. The remainder of this paper is organized as follows. In Sect. 2, we describe the framework of DFC. In Sect. 3, we describe some requirements of DFC which are necessary to ensure that DFC works appropriately in actual networks. In Secs. 4–6, we evaluate the performance of TCP with DFC in various situations: network speed, network size, and different TCP implementations. The comparisons were made using the ns2 simulator. Finally, Sect. 7

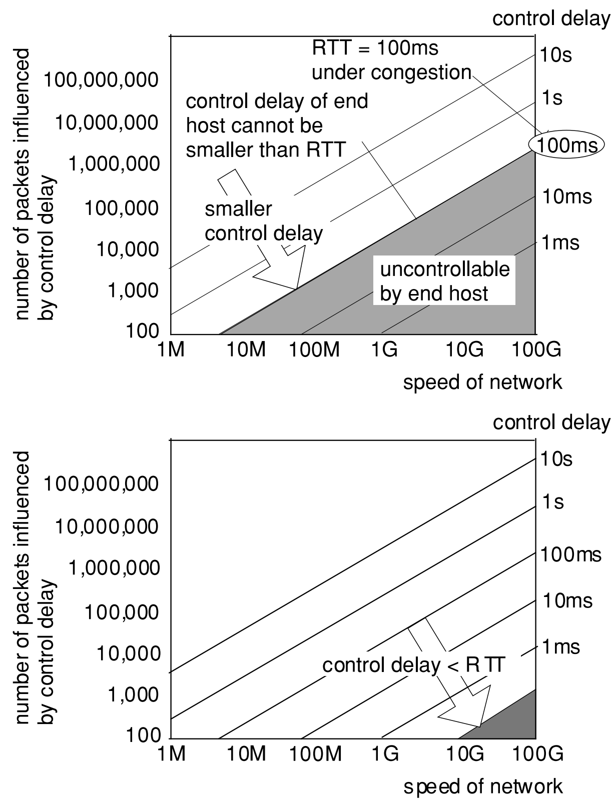


Fig. 2 Relationship between the number of packets influenced by control delay and the speed of the network. The upper graph illustrates control by end hosts and the lower one illustrates node-by-node control. (A trial calculation was made with an average distance between nodes of 10 km, average number of hops = 5, and a link utilization of 0.5.)

provides the conclusions to this paper.

2. Diffusion-Type Flow Control

2.1 Concept

In the case of Internet-based networks, to guarantee the end-to-end quality of service (QoS) of a flow, a QoS-sensitive flow uses a static route (*e.g.*, RSVP). Thus, we assume that a target flow has a static route. In addition, we assume all routers in the network can employ per-flow queuing for all the target flows†. DFC provides a framework in which the implementation of the decision-making of each node leads to high performance for the whole network. The principle of our flow control model can be explained through the following analogy [3].

When we heat a point on a cold iron bar, the temperature distribution follows a normal distribution and heat spreads through the whole bar by diffusion (Fig. 3). In this process, the action in a minute segment of the iron bar is

†The assumption of per-flow queuing is not mandatory in the framework of DFC, but it is convenient to use it to simplify the explanation of the framework. In actual fact, it is hard to implement per-flow queuing in high-speed networks Fundamentally, DFC only requires "per-input port" queueing.

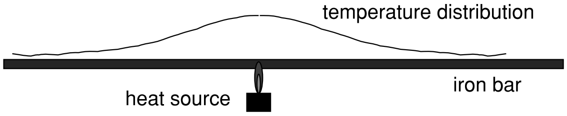


Fig. 3 Example of thermal diffusion phenomenon.

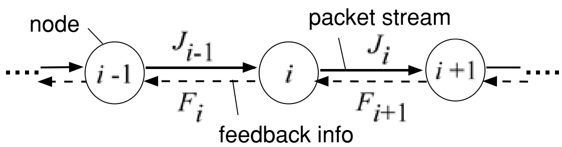


Fig. 4 Node interactions in our flow control model.

very simple: heat flows from the hotter side towards the cooler side. The rate of heat flow is proportional to the temperature gradient. There is no direct communication between two distant segments of the iron bar. Although each segment acts autonomously, based on its local information, the temperature distribution of the whole iron bar exhibits orderly behavior. In DFC, each node controls its local packet flow. The packet rate includes the quantity that is proportional to the difference between the number of packets in the node and that in an adjacent node. Thus, the distribution of the total number of packets in a node in the network becomes uniform over time. In this control mechanism, the state of the whole network is controlled indirectly through the autonomous action of each node.

2.2 Main Principle of Diffusion-Type Flow Control Mechanism

In DFC, each node controls its local packet flow autonomously. Figure 4 shows the interactions between nodes (routers) in our flow control method, using a network model with a simple 1-dimensional configuration. All nodes have two incoming and two outgoing links, for a one-way packet stream and for feedback information, that is, node i (i = 1, 2, ...) transfers packets to node $i+1$, and node $i+1$ sends feedback information to node i. For simplicity, we assume that packets have a fixed length in bits.

All nodes are capable of receiving feedback information from adjacent downstream nodes, and sending it to adjacent upstream nodes. Each node i can receive feedback information sent from the downstream node $i + 1$ and can send feedback information about itself to the upstream node $i - 1$.

When node i receives feedback information from downstream node $i + 1$, it determines the transmission rate for packets to the downstream node $i + 1$ using the received feedback information, and it adjusts its transmission rate towards the downstream node $i + 1$ accordingly. The framework for node behavior and flow control may be summarized as follows:

- Each node i autonomously determines the transmission rate J_i on the basis of only the local information directly available to it, that is, the feedback information obtained from the downstream node $i + 1$ and its own feedback information.
- The rule for determining the transmission rate is the same for all nodes.
- Each node i adjusts its transmission rate towards the downstream node $i + 1$ to J_i.
  (If there are no packets in node i, the packet transmission rate is 0.)
- Each node i autonomously creates feedback information according to a predefined rule and sends it to the upstream node $i - 1$. Feedback information is created periodically with a fixed interval τ_i for each link.
- The rule for creating the feedback information is the same for all nodes.
- Packets and feedback information both experience the same propagation delay.

As mentioned above, the framework of our flow control model involves both autonomous decision-making by each node and interaction between adjacent nodes. There is no centralized control mechanism in the network.

Next, we explain the details of DFC. The transmission rate $J_i(\alpha, t)$ of node i at time t is determined by

$$J_i(\alpha, t) = \max(0, \min(L_i(t), \tilde{J}_i(\alpha, t))), \quad \text{and} \tag{1}$$

$$\tilde{J}_i(\alpha, t) = \alpha\, r_i(t - d_i) - D_i\, (n_{i+1}(t - d_i) - n_i(t)), \tag{2}$$

where $L_i(t)$ denotes the value of the available bandwidth of the link from node i to node $i+1$ for target flow at time t, $n_i(t)$ denotes the number of packets in node i at time t, $r_i(t - d_i)$ is the target transmission rate specified by the downstream node $i + 1$ as feedback information, and d_i denotes the propagation delay between nodes i and $i + 1$. The determination of $L_i(t)$ is explained in the next subsection.

In addition, $r_i(t - d_i)$ and $n_{i+1}(t - d_i)$ are reported from the downstream node $i + 1$ as feedback information with propagation delay d_i. Parameter α (≥ 1), which is a constant, is the flow intensity multiplier. When $\alpha = 1$, the first term on the right-hand side of (2) becomes $r_i(t - d_i)$. This parameter setting influences the balance between input and output traffic at a node. Our previous study [3] showed that $\alpha = 1$ exhibits appropriate performance. Parameter D_i is chosen to be inversely proportional to the propagation delay [2] as follows:

$$D_i = D\,\frac{1}{d_i} \propto (d_i)^{-1}, \tag{3}$$

where D (> 0), which is a positive constant, is the diffusion coefficient.

The feedback information $\mathbf{F}_i(t)$ created at regular fixed intervals of periodicity τ_i by node i consists of the two quantities shown in (4):

$$\mathbf{F}_i(t) = (r_{i-1}(t), n_i(t)). \tag{4}$$

Node i reports this to the upstream node $i - 1$ with a period of $\tau_i = d_{i-1}$. Here, the target transmission rate is determined as

$$r_{i-1}(t) = J_i(1, t). \tag{5}$$

Moreover, the packet flow $J_i(t)$ in node i is adjusted whenever feedback information arrives from the downstream node $i+1$ (with a periodicity of $\tau_{i+1} = d_i$).

To allow an intuitive understanding, we will briefly explain the physical meaning of DFC. Let us replace i with x and apply a continuous approximation. Then the propagation delay becomes $d_i \to 0$ for all i and the packet flow (2) may be expressed as

$$\tilde{J}(\alpha, x, t) = \alpha\, r(x, t) - D\, \frac{\partial n(x, t)}{\partial x}, \tag{6}$$

and the temporal evolution of the packet density $n(x, t)$ may be represented by a diffusion-type equation,

$$\frac{\partial n(x, t)}{\partial t} = -\alpha\, \frac{\partial r(x, t)}{\partial x} + D\, \frac{\partial^2 n(x, t)}{\partial x^2}, \tag{7}$$

using the continuous equation

$$\frac{\partial n(x, t)}{\partial t} = -\frac{\partial \tilde{J}(\alpha, x, t)}{\partial x}. \tag{8}$$

As explained in Sect. 2.1, our method aims to perform flow control using the analogy of diffusion. We can therefore expect excess packets in a congested node to become distributed over the whole network and normal network conditions to be restored after some time.

In addition to the above framework, we consider the boundary condition of the rule for determining the transmission rate in the DFC.

Here we consider the situation where nodes and/or end hosts in other networks do not support the DFC mechanism. We call the nodes and/or end hosts that are connected directly to the ingress node in our network external nodes. We assume that the external nodes only have a traffic shaping function, which can adjust the transmission rate to the requested rate reported by the downstream node.

That is, an external node 0 cannot calculate the transmission rate $J_0(\alpha, t)$ using (2), but can adjust its transmission rate to $r_0(t - d_0)$, which was reported by node 1.

We consider a rule for determining $r_0(t)$ as a boundary condition. Node 1 can calculate $J_0(\alpha, t)$ if we assume that the number of packets stored in the other networks' node is $i = 0$. The target rate $r_0(t)$, reported by node 1, is created as $\tilde{J}_0(\alpha, t)$ with the above assumption. That is,

$$r_0(t) := \tilde{J}_0(\alpha, t + d_0) = \alpha\, J_1(1, t) - D_0\, n_1(t). \tag{9}$$

This quantity can be calculated just from information known to node 1.

2.3 Determination of Available Bandwidth

Since the link bandwidth is shared by multiple flows, we need to adjust the available bandwidth, $L_i(t)$, appropriately among the flows. In this subsection, we take multiple flows into account. So, we have generalized some quantities as follows: $L_i(t) \to L_i^j(t)$, $\tilde{J}_i(\alpha, t) \to \tilde{J}_i^j(\alpha, t)$, etc., where these mean the quantities for flow j. Let the bandwidth of the link from node i to $i+1$ be B_i. Then, $L_i^j(t)$ ($j = 1, 2, \ldots, M_i$) must satisfy

$$\sum_{j=1}^{M_i} L_i^j(t) \leq B_i, \tag{10}$$

where M_i denotes the number of flows at node i.

In DFC, the ideal transmission rate is $\tilde{J}_i^j(\alpha, t)$, and $J_i^j(\alpha, t)$ is restricted by its available bandwidth. In particular, if there are a lot of flows, the ideal transmission rate $\tilde{J}_i^j(\alpha, t)$ is frequently

$$\sum_{j=1}^{M_i} \tilde{J}_i^j(\alpha, t) > B_i, \tag{11}$$

and the probability of each flow getting its ideal transmission rate $\tilde{J}_i^j(\alpha, t)$ is low. This prevents the smooth equalization of the packet density. So, we take into consideration two conditions: the relative value of the ideal transmission rate of each flow, and the following equation.

$$\sum_{j=1}^{M_i} J_i^j(\alpha, t) \leq B_i. \tag{12}$$

Then, the simplest way to determine $L_i^i(t)$ is to assume that the bandwidth B_i is shared between flows according to a flow weight $\tilde{J}_i^j(\alpha, t)$ [12], that is,

$$L_i^j(t) = B_i\, \frac{\tilde{J}_i^j(\alpha, t)}{\sum_{j=1}^{M_i} \tilde{J}_i^j(\alpha, t)}. \tag{13}$$

This rule means that a flow with larger $\tilde{J}_i^j(\alpha, t)$ can have a larger transmission rate and can transmit a larger volume of traffic to the downstream node. Thus, the transmission rates of other flows are regulated to be smaller.

Hereafter, we focus on models having two flows, a target flow and a background flow, and investigate the extent to which DFC and TCP complement each other in operation. The performance of DFC for multiple flows was described in our previous study [12].

3. Requirements of Characteristics of TCP Coexistence with DFC

If DFC is to be installed in actual networks, it must work well with UDP and TCP. Our previous studies showed that DFC itself exhibits improved performance. These results imply that DFC should works well in combination with some kinds of protocols, including UDP, that have no flow control. Since TCP includes its own flow control, we need to verify the compatibility and complementarity of DFC with respect to TCP. We extend the simulation tool ns2 [13] capability with the function of DFC to investigate the performance of DFC when coexisting with TCP.

In general, it is difficult to for two different flow control mechanisms to coexist in the same network, especially if they act in the same time-scale. For example, both TCP and ATM ABR include end-to-end flow control mechanisms, which act in the time-scale of the RTT. Since they make decisions in the same time-scale, but not in a manner which is cooperative, it is hard to obtain synergy between them.

One of important features of DFC, which can help overcome this problem and to achieve complementarity between DFC and TCP, is that they act in different time-scales. DFC is a node-by-node control and acts in a time-scale shorter than the RTT. The difference of the time-scales is larger for high-speed and large-scale networks. Even though it is more difficult to control higher-speed and larger-scale networks using TCP, we expect a coexistence of DFC with TCP to give dependable performance in this situation.

To verify the performance of the coexistence of DFC and TCP, we evaluate their complementarity from the following three points of view:

- Performance characteristics in high-speed networks.
- Performance characteristics in large-scale networks.
- Performance characteristics with different TCP implementations.

4. Complementarity of DFC and TCP with Respect to Network Speed

4.1 Simulation Model

Figure 5 shows our network model with 30 nodes, which is used in the simulations. Although this 1-dimensional model looks simple, it represents a part of a network and describes a path of the target end-to-end flow extracted from the whole network. The propagation delay of each link between nodes is 0.1 ms, and the buffer capacity at each node is 1800 packets. The reason why the capacity of buffer is fixed in our model is as follows. In actual network design, it is natural to choose a larger capacity of buffer for a larger link bandwidth. However, the appropriate capacity with respect to packet loss and delay depends on the RTT of each flow. Since we cannot predict the RTT of incoming flows to a router in advance at the router design stage, the appropriate capacity cannot be designed in advance. So, it is preferable that the appropriate performance should be obtained by using as small a capacity of buffer as possible. One aim of our simulation models is to show that DFC exhibits appropriate performance even when the buffer capacity is small.

The target flow is between node 1 and node 30, while the background traffic flows between node 15 and node 30. The target flow and the background flow start at simulation time $t = 0$ s and $t = 0.1$ s, respectively.

We investigated the network performance with respect to network speed in the cases of TCP flow control without DFC and TCP coexisting with DFC by the use of six simulation scenarios. The background flow was controlled by the same method as the target flow, that is, TCP without DFC or TCP with DFC. Unlike TCP with DFC, TCP without DFC does not use per-flow queuing but uses FIFO. The network parameters and flow control parameters for each simulation scenario are summarized in Table 1. The maximum TCP window size was chosen to be sufficiently larger than the bandwidth-delay product. The source traffic model is greedy, that is, input traffic to the network is the same as the quantity allowed by the flow control. For simplicity, the packet size was a fixed length of 1500 bytes. In addition, we chose $\alpha = 1.0$ in DFC. The implementation of TCP used in the results reported in this section was TCP Tahoe. The buffer capacity of the nodes and the distances between the nodes were the same in all scenarios. The link bandwidth was altered, becoming progressively greater in the different scenarios. The window size was also increased to cater for this larger bandwidth. That is, scenario 1 represented a network with the smallest link bandwidth, while scenario 6

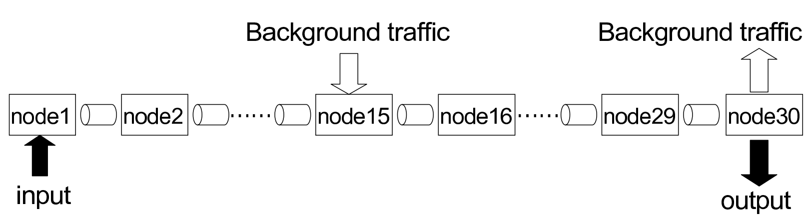
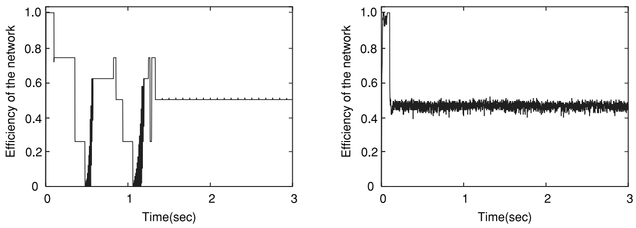


Fig. 5 Simulation model.

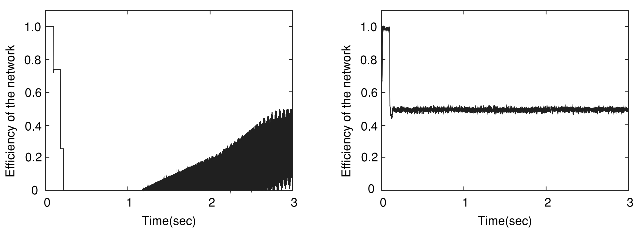
Table 1 Simulation conditions.

	network parameters			flow control parameters	
	link bandwidth (Mbps)	link delay (msec)	buffer capacity (packets)	diffusion coefficient	max. window size
scenario 1	370	0.1	1800	0.1	5,000
scenario 2	1,230	0.1	1800	0.1	5,000
scenario 3	3,700	0.1	1800	0.1	5,000
scenario 4	12,300	0.1	1800	0.1	10,000
scenario 5	37,000	0.1	1800	0.1	20,000
scenario 6	123,000	0.1	1800	0.1	100,000



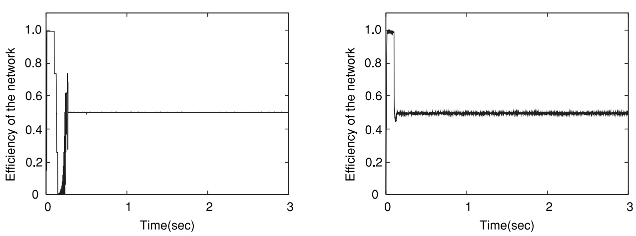
TCP without DFC TCP with DFC

Fig. 6 Simulation scenario 1.



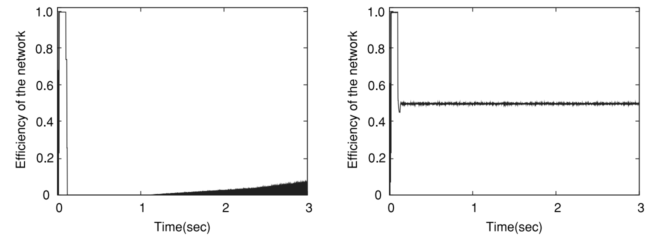
TCP without DFC TCP with DFC

Fig. 7 Simulation scenario 2.



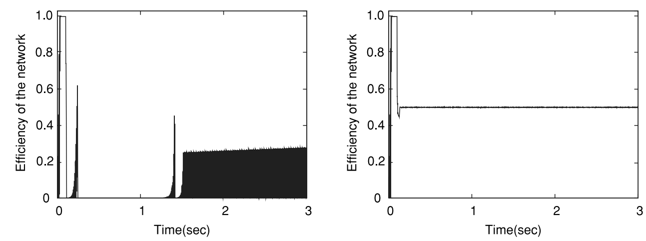
TCP without DFC TCP with DFC

Fig. 8 Simulation scenario 3.



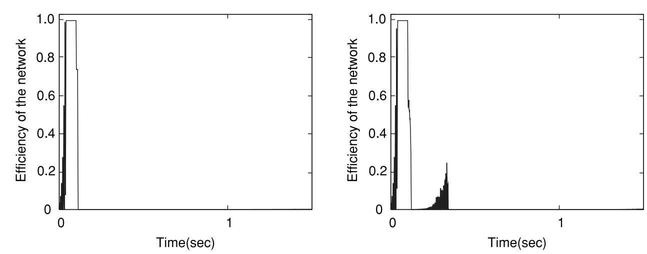
TCP without DFC TCP with DFC

Fig. 9 Simulation scenario 4.



TCP without DFC TCP with DFC

Fig. 10 Simulation scenario 5.



TCP without DFC TCP with DFC

Fig. 11 Simulation scenario 6.

represented a network with the largest link bandwidth.

4.2 Simulation Results

Figures 6–11 show the simulation results obtained from scenarios 1–6, respectively. The horizontal axes denote the simulation time and the vertical axes denote the efficiency of the networks. Here, the efficiency of the network represents the normalized value of the total number of packets that are in transit on links, that is, a ratio of the total number of packets that are in transit on links to the maximum number of packets that can be in transit on links. The total number of packets means the number of packets being transported by the network at a particular instant. The left-hand graph of each figure shows the result obtained using TCP without DFC, and the right-hand graph shows the result obtained using TCP with DFC. The simulation time was 3 s for scenarios 1–5, and 1.5 s for scenario 6. After the time when the background traffic started (after 0.1 s), the available bandwidth for the target flow was reduced to a half. If the efficiency of networks reaches 0.5 after 0.1 s, the target flow is sharing the link bandwidth fairly, with high utilization.

In Figs. 6–10 the efficiency of TCP without DFC is low and unstable. This is because the background traffic entering at node 15 causes packet losses and with the result that TCP reduces its window size. On the other hand, the efficiency of TCP with DFC remains high and stable. This is because DFC acts to prevent packet loss. Through the diffusion effect of DFC, the number of packets stored at nodes becomes uniform over time. These results show that when TCP is used in conjunction with DFC it is possible to maintain high and stable network efficiency, even if background traffic changes. In Fig. 11, both TCP without and TCP with DFC exhibit low and unstable efficiency. At this link speed, the capacity of the buffers is too small to prevent packet loss even when DFC is applied. Thus, packet losses can occur not only in the case of TCP without DFC but also when TCP is used with DFC, if the buffer size is not adequate for the link speed.

Next, to investigate the difference between TCP without DFC and TCP with DFC, we show the temporal evolution of the number of packets stored in each node. Figures 12 and 13 show the results obtained from TCP wthout and with DFC (both are scenario 4). The horizontal axes denote node ID (1–29) and the vertical axes denote the number of packet stored at the node. The five different graphs for each case represent different instants during the simulation,

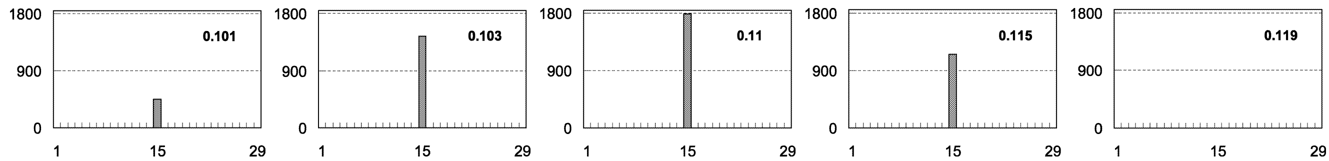


Fig. 12 Temporal evolution of distribution of packets stored in each node (TCP without DFC).

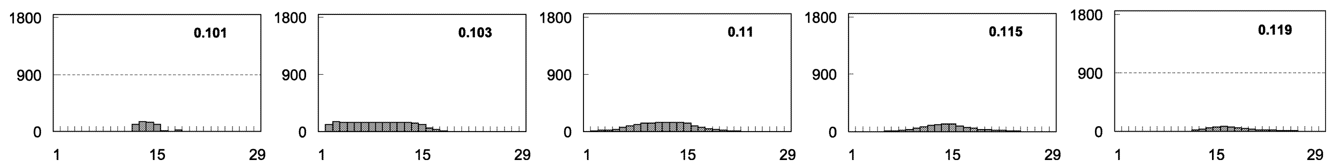


Fig. 13 Temporal evolution of distribution of packets stored in each node (TCP with DFC).

the time being shown on each graph.

In the case of Fig. 12, after the time when the background traffic started (after 0.1 s), all the stored packets were at node 15, resulting in packet loss. The number of stored packet at node 15 reached the buffer size of 1800 and packet loss occured. After that, the TCP window size was reduced and the number of stored packet decreased. In the case of Fig. 13, TCP with DFC prevents the stored packets from building up at a particular node. This effect is due to the operation of DFC. Since packet loss was avoided, the TCP window size did not reduce and high network efficiency was achieved, as shown as Figs. 6–10. Through the introduction of DFC, each node acts cooperatively to avoid packet loss even though the decision-making of each node is based only on the local information.

5. Complementarity of DFC and TCP with Respect to the Number of Hops

5.1 Simulation Model

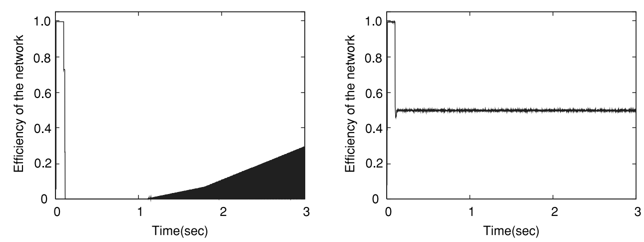
In this section, we investigate the performance characteristics of TCP with DFC with respect to the network size. We take scenario 4 in Sect. 4 as a reference model, and add new models which have 16 nodes and 60 nodes. We call them scenarios 4S and 4L, respectively. The background traffic is entered at node 8 in scenario 4S and node 30 in scenario 4L. In scenario 4L, the maximum TCP window size was changed from scenario 4 because it should be chosen to be sufficiently larger than the bandwidth-delay product. Other simulation conditions were the same as for scenario 4 (Table 2).

5.2 Simulation Results

Figures 14 and 15 show the results obtained from scenarios 4S and 4L, respectively. The horizontal axes denote the simulation time and the vertical axes denote the efficiency of the network. The left-hand graph of each figure shows the result obtained using TCP without DFC, and the right-hand graph shows the result obtained using TCP with DFC. By comparing these figures with Fig. 9, the performance of TCP without DFC can be seen to be degraded as the network size increases. In other words, a larger RTT degrades the performance of TCP. On the other hand, TCP with DFC exhibits high and stable network efficiency for all cases. These results show that DFC can be effective irrespective of network size.

Table 2 Simulation conditions 2.

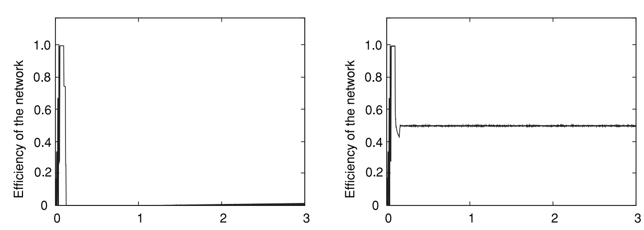
	number of nodes	window capacity	other parameters
scenario 4S	16	10,000	same as scenario 4
scenario 4L	60	20,000	same as scenario 4



TCP without DFC TCP with DFC

TCP without DFC TCP with DFC

Fig. 14 Simulation scenario 4S.



TCP without DFC TCP with DFC

Fig. 15 Simulation scenario 4L.

6. Complementarity of DFC and TCP with Respect to Differences in TCPs Implementation

6.1 Simulation Model

In previous sections, our evaluation results were obtained using TCP Tahoe. Other popular implementations of TCP are Reno, NewReno, and Vegas. The main difference of Reno and NewReno with respect to Tahoe is in the behavior of TCP window size after the detection of packet loss.

In the evaluation of TCP with DFC it is possible to avoid packet losses if each node has appropriate buffer size (but the size is still smaller than that required for TCP without DFC). Thus, we may expect that replacing Tahoe with Reno or NewReno in TCP with DFC will cause no essential change in the results described in the previous sections. However, Vegas is rather different from other implementations. In this section, we evaluate the performance characteristics of TCP with DFC with respect to differences in TCP implementation.

As in Sect. 5, we take scenario 4 in Sect. 4 as a reference model, and add new models using TCP Reno, NewReno, and Vegas. We call them scenarios 4R, 4N, and 4 V, respectively. Other simulation conditions are same as in scenario 4 (see Table 3). TCP window control in Vegas is triggered by the detection of a variation in RTT, variations while in the other implementations (Reno and NewReno) window control is triggered by detection of packet losses. Thus, Vegas can act in a smaller time-scale than the others. That is, Vegas has a smaller control delay than the other implementations. To further investigate the behavior of TCP Vegas without and with DFC, we introduce a new model in addition to the above models. This model has a smaller buffer size (400 packets), but the other conditions are the same as for scenario 4 V. We call it scenario 4 V' (Table 4).

6.2 Simulation Results

Figures 16 and 17 show the results obtained from scenarios 4R and 4N, respectively. The horizontal axes denote simulation time and the vertical axes denote the efficiency of the network. The left-hand graph in each figure shows the result obtained from TCP without DFC, and the right-hand graph shows the result obtained from TCP with DFC. By comparing these figure with Fig. 9, we can see that the performance of TCP without DFC for Reno or NewReno is improved compared with than that for Tahoe. However, the improved performance is still inferior to TCP with DFC. TCP with DFC exhibits high and stable efficiency of the network for all three of the TCP implementations compared here, that is Tahoe, Reno, and NewReno.

Figure 18 shows the results obtained from scenario 4 V. This result is completely different from the previous results. Strong oscillations appears in the behavior of the efficiency of the network. Since high frequency oscillations in the behavior of the efficiency of the network do not contribute to throughput (the efficiency of the network averaged over a longer time-scale), the throughput tends to decrease and to converge slowly. The results for TCP without and TCP with DFC are almost the same, and this similarity means the origin of the behavior is a feature of TCP Vegas. The addition of DFC does not influence the behavior of the efficiency of t he network in this case.

In the evaluation of scenario 4 V, there was no packet loss. This is because Vegas has a smaller control delay than

Table 3 Simulation conditions 3.

	implementation of TCP	buffer capacity	other parameters
scenario 4R	Reno	same as scenario 4	same as scenario 4
scenario 4N	Newreno	same as scenario 4	same as scenario 4
scenario 4 V	Vegas	same as scenario 4	same as scenario 4

Table 4 Simulation conditions 4.

	implementation of TCP	buffer capacity	other parameters
scenario 4 V	Vegas	same as scenario 4	same as scenario 4
scenario 4 V'	Vegas	400	same as scenario 4

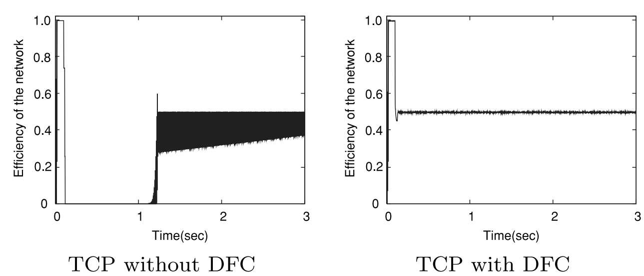


Fig. 16 Simulation scenario 4R.

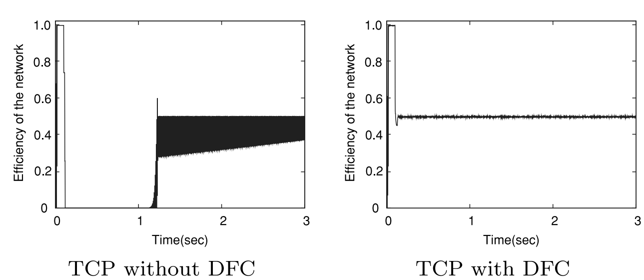


Fig. 17 Simulation scenario 4N.

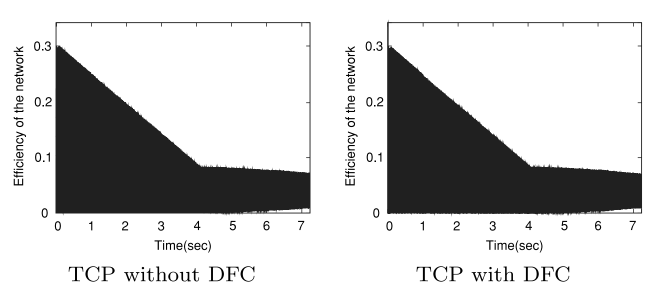
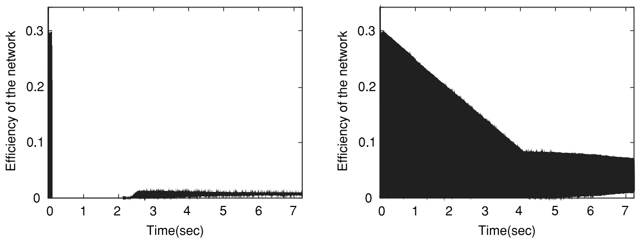


Fig. 18 Simulation scenario 4 V.



TCP without DFC TCP with DFC

Fig. 19 Simulation scenario 4 V'.

other implementations. To further clarify the difference between TCP without and TCP with DFC, we evaluate scenario 4 V', in which a smaller buffer size is used. Figure 19 shows the results obtained from scenario 4 V'. In the case of TCP without DFC, there is packet loss and the efficiency of the network is degraded remarkably. In the case of TCP with DFC, the efficiency obtained is same as in scenario 4 V.

Consequently, we may conclude that introducing DFC makes it possible to stabilize TCP performance for all TCP implementations: Tahoe, Reno, NewReno, and Vegas.

7. Conclusions

To overcome the difficulty in controlling high-speed networks, we have proposed DFC, where each node manages its local traffic flow on the basis of only the local information directly available to it, by using predetermined rules.

One of the important issues in the application of DFC is the verification of the applicability of DFC to conventional networks. DFC needs to work well in conventional networks. This paper investigates the issue of the complementarity of DFC and TCP. This is the central issue for DFC applicability. We extended the capabilities of the simulation tool ns2, by adding the function of DFC, and investigated the performance of TCP without DFC and TCP with DFC. We found that the efficiency of the network is low and unstable in the case of TCP without DFC, while in case of TCP with DFC the efficiency of the network becomes stable and high, even for high speed networks.

In addition, we compared the behavior of the distribution of packets among the individual nodes for TCP without DFC and for TCP with DFC, in order to demonstrate the performance of DFC. The simulation results showed that packet loss occurs at congestion nodes for TCP without DFC, while TCP with DFC avoids packet losses.

Moreover, we investigated the performances of TCP with DFC with respect to different network sizes and different TCP implementations. These results tell us that introducing DFC enables to stabilize TCP performance in various situations.

These results indicate that the desirable characteristics of DFC, including the cooperative behavior of nodes to avoid packet loss and the ability to recover rapidly from congestion, are also effective for a TCP controlled flow.

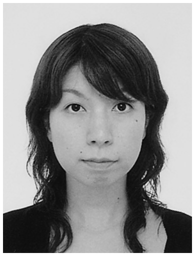
Acknowledgments

This research was partially supported by the Grant-in-Aid for Scientific Research (S) No. 18100001 (2006–2010) from the Japan Society for the Promotion of Science.

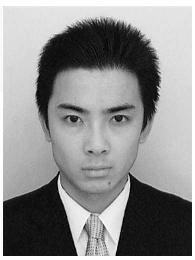
References

[1] C. Takano and M. Aida, "Stability and adaptability of autonomous decentralized flow control in high-speed networks," IEICE Trans. Commun., vol.E86-B, no.10, pp.2882–2890, Oct. 2003.
[2] C. Takano, M. Aida, and S. Kuribayashi, "Autonomous decentralized flow control in high-speed networks with inhomogeneous configurations," IEICE Trans. Commun., vol.E87-B, no.6, pp.1551–1560, June 2004.
[3] C. Takano and M. Aida, "Diffusion-type autonomous decentralized flow control for end-to-end flow in high-speed networks," IEICE Trans. Commun., vol.E88-B, no.4, pp.1559–1567, April 2005.
[4] D. Katabi, M. Handley, and C. Rohrs, "Congestion control for high bandwidth-delay product networks," Proc. ACM SIGCOMM 2002, pp.89–102, Aug. 2002.
[5] Y. Bartal, J. Byers, and D. Raz, "Global optimization using local information with applications to flow control," Proc. 38th Ann. IEEE Symp. on Foundations of Computer Science, pp.303–312, Oct. 1997.
[6] S.H. Low and D.E. Lapsley, "Optimization flow control-I: Basic algorithm and convergence," IEEE/ACM Trans. Netw., vol.7, no.6, pp.861–874, 1999.
[7] K. Kar, S. Sarkar, and L. Tassiulas, "A simple rate control algorithm for maximizing total user utility," Proc. IEEE INFOCOM 2001, pp.133–141, 2001.
[8] J. Mo and J. Walrand, "Fair end-to-end window based congestion control," IEEE/ACM Trans. Netw., vol.8, no.5, pp.556–567, Oct. 1999.
[9] S. Kunniyur and R. Srikant, "A decentralized adaptive ECN marking algorithm," Proc. IEEE GLOBECOM'00, pp.1719–1723, 2000.
[10] M. Aida and K. Horikawa, "Stability analysis for global performance of flow control in high-speed networks based on statistical physics," IEICE Trans. Commun., vol.E82-B, no.12, pp.2095–2106, Dec. 1999.
[11] R. Johari and D. Tan, "End-to-end congestion control for the Internet: Delays and stability," IEEE/ACM Trans. Netw., vol.9, no.6, pp.818–832, Dec. 2001.
[12] M. Aida, C. Takano, and A. Miura, "Diffusion-type flow control scheme for multiple flows," 19th International Teletraffic Congress (ITC19), pp.133–142, 2005.
[13] The Network Simulator — ns-2. http://www.isi.edu/nsnam/ns/

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



Kaori Muranaka received the B.S. in Mathematics from Tsuda College, Tokyo, Japan, in 1999. In 1999 she joined Traffic Research Center, NTT Advanced Technology Corporation (NTT-AT). She has been engaged in research on traffic issues in computer communication networks.



Keita Sugiyama received his B.E. in Information Systems Engineering from Tokyo Institute of Technology, Japan, in 2006. Now, he is a graduate student of the Graduate School of System Design, Tokyo Metropolitan University.

Masaki Aida received his B.Sc. and M.Sc. in Theoretical Physics from St. Paul's University, Tokyo, Japan, in 1987 and 1989, and received the Ph.D. in Telecommunications Engineering from the University of Tokyo, Japan, in 1999. He joined NTT Laboratories in 1989. He was also a manager at Traffic Research Center, NTT Advanced Technology Corporation (NTT-AT) from 1998 to 2001. His current interests include traffic issues in communication systems. He is currently an Associate Professor at the Graduate School of System Design, Tokyo Metropolitan University. Dr. Aida received the Young Researchers' Award of IEICE in 1996. He is a member of the IEEE and the Operations Research Society of Japan.