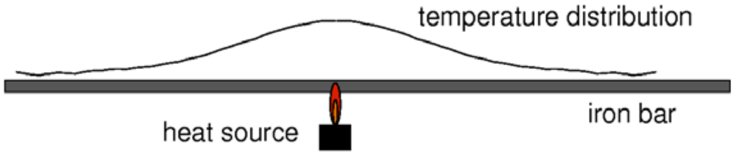

Figure 1: Example of thermal diffusion phenomena.

information about the network, but it is impossible to achieve such a centralized control mechanism in high-speed networks. In addition, solving these optimization problems requires enough time to be available for calculation, so it is difficult to apply these methods to decision-making on a very short time-scale. Therefore, in a high-speed network, the principles adopted for time-sensitive control are inevitably those of autonomous decentralized systems.

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 [3,4]. 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 inadequate for application to support decision-making on a very short time-scale. In low-speed networks, a control delay on the order of the round-trip time (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 the RTT becomes large relative to the unit of time determined by node's processing speed, although the RTT is itself unchanged. This means that nodes in high-speed networks experience a larger RTT, and this causes an increase in the sensitivity to control delay. To achieve rapid control on a shorter time scale than the RTT, it is preferable to apply control by the nodes rather than by the end hosts.

We therefore considered a control mechanism in which the nodes in a network handle their local traffic flows themselves, based only on the local information directly available to them. This mechanism can immediately detect a change in the network state around the node and apply quick decision-making. Although decision-making at a local node should lead to action suitable for the local performance of the networks, the action is not guaranteed to be appropriate for the overall network-wide performance. Therefore, the implementation of decision-making at each node cannot lead to optimum performance for the whole network.

In our previous studies, we proposed a diffusion-type flow control (DFC) [5,6]. 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 [7].

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. 1). In this process, the action in a minute segment of the iron bar is very simple: heat flows from the hotter side towards cooler side. The rate of heat flow is proportional to the temperature gradient. There is no 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, which is proportional to the difference between the number of packets in the

information, and it adjusts its transmission rate towards the downstream node $i+1$. The framework for node behavior and flow control is summarized as follows:

- Each node i autonomously determines the transmission rate J_i based only on information directly available to it, that is, the feedback information obtained from the downstream node $i+1$ and its own feedback information.
- 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.
- The above rules are 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 $J_i(\alpha, t) = \max(0, \min(L_i(t), \tilde{J}_i(\alpha, t)))$ and

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

where L_i denotes the value of the link capacity from node i to node $i+1$, $n_i(t)$ 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 $i+1$.

In addition, $r_i(t-d_i)$ and $n_{i+1}(t-d_i)$ are reported every fixed period τ_{i+1} from the downstream node $i+1$ with propagation delay d_i. Parameter α (≥ 1), which is a constant, is the flow intensity multiplier. Parameter D_i is chosen to be inversely proportional to the propagation delay [6] as $D_i = D/d_i$, where D (> 0), which is a positive constant, is the diffusion coefficient.

The feedback information $\mathbf{F}_i(t)$ created every fixed period τ_i by node i consists of the following two quantities: $\mathbf{F}_i(t) = (r_{i-1}(t), n_i(t))$. 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 $r_{i-1}(t) = J_i(1, t)$. Moreover, the packet flow $J_i(t)$ in node i is renewed whenever feedback information arrives from the downstream node $i+1$ (with a period of $\tau_{i+1} = d_i$).

To enable an intuitive understanding, we briefly explain the physical meaning of DFC. We 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 (1) is expressed as

$$\tilde{J}(\alpha, x, t) = \alpha\, r(x, t) - D\, \frac{\partial n(x, t)}{\partial x}, \tag{2}$$

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

$$\frac{\partial n(x, t)}{\partial t} = -\alpha\, \frac{\partial r(x, t)}{\partial x} + D\, \frac{\partial^2 n(x, t)}{\partial x^2}, \tag{3}$$

to the node outside of the network. Then, at least, the node outside of the network can regulate the input traffic according to $\min(\ell_1(t), r_0(t))$ reported by node 1.

3. CLASSIFICATION OF PACKET DENSITY UNIFORMIZATION

In this section, we reconsider the uniformization of packet density and classify its methods.

(a) Serial Diffusion: To avoid packet losses, the number of packets of the target flow in routers along the flow's path is uniformized. This type is further divided into two types; (a-1) Backward Serial Diffusion: with respect to a bottleneck link, the number of packets is uniformized toward the upstream direction. (a-2) Forward Serial Diffusion: with respect to a bottleneck link, the number of packets is uniformized toward the downstream direction.

(b) Parallel Diffusion: For multiple flows, which share all or a part of the path, the number of packets at a node on the common path is uniformized among all the flows.

In our previous work, our target was serial diffusion, in particular, backward serial diffusion, because, for a single flow environment, the bottleneck link with bandwidth L_i prevents the packet flow at node i being greater than L_i (see Sec. 5.2). However, in a multiple flow environment, the bandwidth of the bottleneck link is shared by multiple flows, so some flows may have larger rates than others. For fast congestion recovery, both backward and forward serial diffusion are important. We can expect both to be achieved by appropriately setting the available bandwidth of each flow.

We hope that the effect of packet uniformization will spread not only along the path of the target flows but also to the whole nelwork, so packet uniformization among flows by parallel diffusion is important. To achieve parallel diffusion, we asign an appropriate available bandwidth L_i for each flow passing through node i.

From the above considerations, appropriate setting of L_i is the key issue to achieve DFC with backward and forward diffusion and parallel difuusion.

4. DIFFUSION-TYPE FLOW CONTROL FOR MULTIPLE FLOWS

4.1 FRAMEWORK

In this paper, all flows are in the same priority class and it is desirable that all active flows share link bandwidth fairly. Extension to the case where flows require different bandwidths is easy.

There are M_i flows sharing the link between nodes i and $i+1$, and they are identified by j ($j = 1, 2, \ldots, M_i$). Some quantities are redefined for each flow j as follows: $n_i^j(t)$: the number of packets of flow j in node i at time t, $r_i^j(t-d_i)$: the rate for flow j reported by feedback information from the downstream node $i+1$, $n_{i+1}^j(t-d_i)$: the number of packets of flow j in node $i+1$ reported by feedback information from the downstream node $i+1$, and $L_i^j(t)$: the available bandwidth for flow j of a link from node i to $i+1$.

Using these quantities, we consider the framework of DFC for multiple flows. When the feedback information from downstream node $i+1$ is received, each node i autonomously

5. SIMULATION RESULTS
5.1 EVALUATION MODEL

To demonstrate the characteristics of DFC, this section compares simulation results of DFCs for multiple flows and for a single flow. In our evaluation, we used packet shaping with $\ell_i(t)$, and the DFC parameters were set as $D = 0.1$ and $\alpha = 1$, and the interval of the feedback information $\mathbf{F}_i^j(t)$ was set to d_{i-1}.

Figure 3 shows our network model with 60 nodes, which was used in the simulations. Although each 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. We represent the lengths of links by their delays, and the length of links is 1.0 [unit time]. A packet has a fixed length and the link bandwidth is 100 packets per unit time.

The simulation scenario was as follows. There were two flows. Flow 1 began at time $t = 1000$ and background flow 2 began at time $t = 0$. The path of flow 1 was from node 0 to 60 and that of background flow 2 was from node 30 to 60. The maximum rate (without traffic regulation) of both flows was 100 packets per unit time (same as link bandwidth). After the flow 1 traffic entered the network, the link from node 30 to 31 became a bottleneck, and traffic of both flows was regulated by predefined rules for DFC and packet shaping. After congestion occurred,we investigated the temporal evolution of the network state, for DFC for both multiple flows and a single flow.

5.2 SIMULATION RESULTS FOR SERIAL DIFFUSION

The upper panel of Fig. 4 shows the temporal evolution of the total number of packets stored in each node, for flow 1, when DFC for a single flow was applied. We chose the available bandwidth at the bottleneck link as $L_{30}^1 = L_{30}^2 = B_{30}/2$. Because the transmission rate $J_{30}^1(1,t)$ was regulated to less than or equal to $B_{30}/2$ at node 30, the nodes downstream from node 30 were not congested. The effect of backward serial diffusion caused congestion at node 30 to spread to the upstream nodes and prevented packet losses at node 30. If we had not applied DFC, all stored packets would have concentrated at node 30, which might have caused packet losses.

The lower panel of Fig. 4 shows the result in the case where DFC for multiple flow was applied. The available bandwidth at the bottleneck link was determined by (6), and its value dynamically changed depending on the situation. This result reveals both backward and forward serial diffusion. We can also see that the time taken to recover

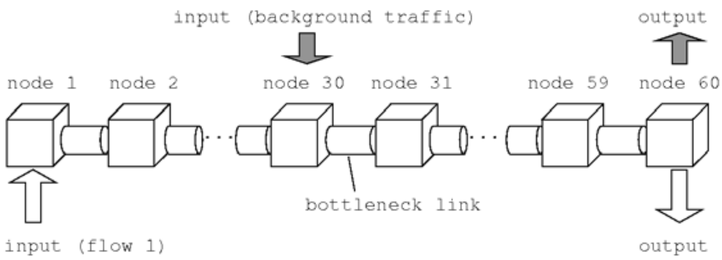

Figure 3: Network model.

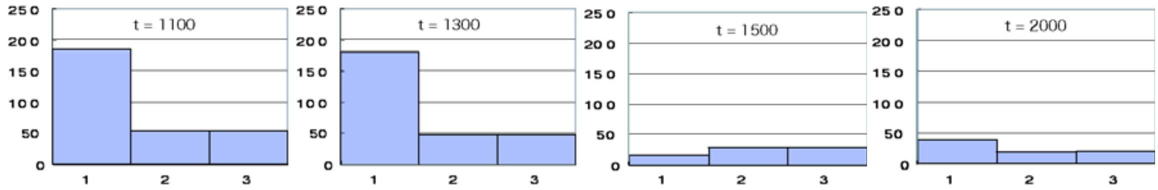

Figure 6: Comparison of the number of each flow's packets stored in node 30.

at time $t = 0$ and background flows 2 and 3 began at time $t = 1000$. The path of flow 1 was from node 1 to 60, and the paths of background flows 2 and 3 were from node 30 to 60. The maximum rate (without traffic regulation) of flow 1 was 100 packets per unit time (same as link bandwidth), and each maximum rate of background flow 2 or 3 was 20 packets per unit time. After the traffic of background flows 2 and 3 entered the network, the link from node 30 to 31 became a bottleneck. Then the traffic was regulated by the predefined rules of DFC and packet shaping.

Figure 6 shows the temporal evolution of the number of each flow's packets stored in the congested node 30. We can see that the number of flow 1's packets is large. This is because the distance between the ingress node of flow 1 and node 30 was longer than that of other flows. After that, the number of packets for flow 1 decreased and was uniformized among other flows. If we had used DFC for a single flow with $L^1_{30} = L^2_{30} = L^3_{30} = B_{30}/3$, the numbers of stored packets for background flows 2 and 3 would have been 0. Figure 6 implies that DFC for multiple flows achieves fast recovery from congestion by influencing the background flows 2 and 3. However, since the numbers of their flow's packets stored in node 30 were sufficiently small, the influence on the background flows was small.

6. CONCLUSIONS

In this paper, we extended DFC to be applicable to multiple flows. DFC aims to prevent packet concentration at the congested node and to avoid packet losses. Uniformization of the number of packets is classified into two types: serial diffusion along the path of the flow and parallel diffusion among different flows. Our previous work on DFC treated only a single flow and it achieved only partial serial diffusion (backward serial diffusion). By choosing the available bandwidth for each flow appropriately, DFC for multiple flows achieves not only backward serial diffusion, but also forward serial diffusion and parallel diffusion. In addition, the proposed flow control can shorten the time taken to recover from congestion.

REFERENCES

1. Y. Bartal, *et al.*, 38th Ann. IEEE Symp. on Foundations of Computer Science (1997).
2. S. H. Low and D. E. Lapsley, IEEE/ACM Trans. Netw., Vol. 7, No. 6 (1999) 861.
3. J. Mo and J. Walrand, IEEE/ACM Trans. Netw., Vol. 8, No. 5 (1999) 556.
4. R. Johari and D. Tan, IEEE/ACM Trans. Netw., Vol. 9, No. 6 (2001) 818.
5. C. Takano and M. Aida, IEICE Trans. Commun., Vol. E86-B, No. 10 (2003) 2882.
6. C. Takano, *et al.*, IEICE Trans. Commun., Vol. E87-B, No. 6 (2004) 1551.
7. C. Takano and M. Aida, IEICE Trans. Commun., Vol. E88-B, No. 4 (2005) 1559.