Rate-Based Flow Control Model in Group Communication

Iguchi Akihito, Takayuki Tachikawa, and Makoto Takizawa
Tokyo Denki University
E-mail {igu,tachi,taki}@takilab.k.dendai.ac.jp

The transmission speed of the high-speed network is faster than the processing speed of the process. The process loses messages if the messages arrive at the process faster than the processing speed. In addition, a group of the processes have to be cooperated by exchanging multimedia messages in the high-speed network. Here, each process not only sends messages to multiple processes but also receives messages from multiple processes in the group. In the group communication, every process p_i is required to receive all the messages in some order like the causal one. In this paper, we discuss how to allocate the transmission speed to the processes in the group so as to satisfy the quality of service (QoS) required.

グループ通信における送信率割り当て方式

井口 昭人　立川 敬行　滝沢 誠
東京電機大学理工学部経営工学科
E-mail {igu,tachi,taki}@takilab.k.dendai.ac.jp

Gbps の伝送速度を持つ高速通信網では、メッセージの伝送時間は伝搬遅延時間よりも小さく、通信網の伝送速度は送信プロセスの処理速度よりも大きい。このため、メッセージの送信間隔がある一定時間以下であると、バッファオーバーランが起こり、スループットが低下する。バッファオーバーランを防ぐために、プロセス間でのメッセージの送信率を制御する必要がある。また、マルチメディアデータの転送では、各プロセスが各メディア毎に割り当てたい送信率が異り、このための制御が複雑なものとなる。テレビ会議等の応用では、複数のプロセスがグループを作り、グループ内のプロセスが互いにメッセージ通信を行なっている。これまでに、1対1または同報型の送信率制御方式について論じられてきたが、グループでの送信率制御は論じられていない。本論文では、グループ内の各プロセスが要求する送信率に対して、グループとして最適な送信率を割り当てるための手法を提案する。

1 Introduction

High-speed communication networks like ATM networks [5] and gigabit local area networks [7] have been available now. Distributed applications are realized by the cooperation of multiple processes $p_1, \ldots, p_n$ interconnected by the high-speed communication networks. The transmission speed of the network is faster than the processing speed of the process p_i. p_i loses messages if the messages arrive at p_i faster than the processing speed of p_i. Kinds of high-speed communication protocols like XTP [3], VMTP [2], NETBLT [4], RTP [13], RTCP [13], and RSVP [18] are discussed so far. In these protocols, each process is assumed to send messages to one process or a group of multiple processes. In the one-to-one communication, the papers [2–4] discuss how to adjust the transmission rate of the sender so that the receivers could receive all the messages sent by the sender. In the multicast communication [2], only one sender sends messages to a group of multiple processes like a collection of replicated servers.

In distributed applications like groupware [6], a group G of multiple processes $p_1, \ldots, p_n$ are cooperated to achieve some objective by exchanging messages. Here, each process p_i not only sends messages to multiple processes but also receives messages from multiple processes in G while one sender sends messages to one or more than one process in the one-to-one and multicast communication. In the group communication, every process p_i is required to receive all the messages sent to p_i in G in some receipt order. Many papers [1, 9, 12, 15, 17] discuss kinds of group communication protocols which support a collection of multiple processes with the causally [1,8] or totally ordered delivery of messages. Nakamura and Takizawa [11] discuss how to decide the transmission rates of the processes interconnected by a high-speed broadcast channel in the group so that the total transmission rate of the channel can be fairly shared by every process in G. However, only a single stream is transmitted by each process in the group and there is no relation among the streams.

In the multimedia applications, the processes exchange multiple multimedia streams like voice and image streams in G. The streams are required to be delivered to the destinations with some *quality of service* (QoS). For example, a voice stream has to be delivered in some time constraint even if the some data, i.e. frames are lost. In addition to supporting each stream with some QoS, there is some constraint on the QoS among the streams. For example, the transmissions of like voice and image streams have to be synchronized, e.g. two streams are transmitted and received at the same rate. It is critical for every process to be able to receive multiple streams of messages sent by multiple processes so as to satisfy the QoS constraints on the streams, e.g. transmissions of multiple streams are synchronized, and to increase the throughput of the system. In this paper, we present a general model where each process sends and receives multiple streams to and from multiple processes, which are interrelated in G. We discuss how to allocate the transmission rates to the processes in G so that the constraints and requirements on the streams are satisfied.

In section 2, we present the system model. In section 3, we discuss how to control the transmission rates of the processes in the group. In section 4, we present the implementation.

2 System Model

A distributed application is realized by the cooperation of multiple processes $p_1, ..., p_n$. A *group* G is a collection of p_1, ..., p_n, i.e. $G = \{p_1, ..., p_n\}$. First, G is established among $p_1, \dots, p_n$. Then, the processes exchange messages with only the processes in G, respectively. Messages can be delivered in the same causal order as received in G [1, 9, 12, 15, 17]. In this paper, we assume that the processes are interconnected by the high-speed point-to-point channels. G is logically considered to support every pair of processes p_i and p_j with a reliable, bidirectional *logical* communication channel $\langle p_i, p_j \rangle$ [Figure 1]. p_i and p_j can exchange messages through the channel $\langle p_i, p_j \rangle$. p_i and p_j can reliably deliver messages to one another in the sending order, i.e. FIFO order without any message loss by using the channel $\langle p_i, p_j \rangle$. We first assume that every pair of channels $\langle p_i, p_j \rangle$ and $\langle p_h, p_l \rangle$ are independent. That is, the transmission rate of $\langle p_i, p_j \rangle$ is independent of how congested $\langle p_h, p_l \rangle$ is. If multiple processes share one physical channel like the Ethernet, multiple logical channels are not independent. In order to make the discussion simple, we first make the assumption. Then, we extend the discussion to more general cases where the channels are dependent on others.

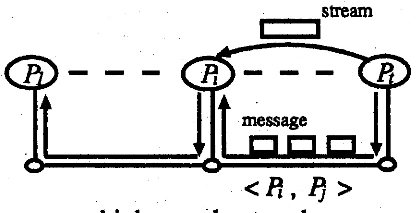

Figure 1: System configuration.

Each channel $\langle p_i, p_j \rangle$ in addition supports p_i and p_j with the high-speed data transmission of messages. The transmission speed of the channel $\langle p_i, p_j \rangle$ is faster than the processing speed of p_i or p_j. p_j loses messages due to the buffer overrun if the messages arrive at p_j from p_i faster than p_j could receive even if the channel $\langle p_i, p_j \rangle$ itself is reliable. Hence, p_i has to transmit messages to p_j at such a fast rate that p_j can receive all the messages without the overrun. This is the rate-based flow control [3, 2, 4].

Each process p_i sends data to other processes in the group G. A unit of data exchanged between the processes is referred to as a *stream*. For example, suppose that four processes p_i, p_j, p_k, and p_h communicate with each other. Here, suppose that p_i sends two streams, i.e. video stream V_i and image stream I_i to p_j and p_k in G. p_j sends a voice stream V_j to p_i and p_h. p_k sends a voice stream V_k to p_j. Here, p_j receives V_i and I_i from p_i and V_k from p_k while sending V_j to p_i and p_h. Thus, each process p_i not only sends but also receives one or more than one stream to and from other processes in G. In addition, some streams may be related in applications. For example, p_j is required to receive I_i from p_i and V_k from p_k at the same rate. That is, the transmissions of I_i and V_k are required to be synchronized.

A *message* is a unit of data transmitted in the network. p_i decomposes a stream issued by the application process to a sequence of smaller messages [Figure 1]. p_i sends messages decomposed from the streams to the network. p_i constructs the streams from the messages received in the network and then delivers the streams to the applications.

3 Transmission and Receipt Rates

3.1 Massage rates

Each logical channel $\langle p_i, p_j \rangle$ in the network supports p_i and p_j with the high-speed transmission of messages. Here, the transmission speed of the channel is faster than the receipt speed of the process. Each process p_i in the group G receives messages if the messages arrive at p_i at a slower rate than p_i could receive. Here, let $maxRR_i(t)$ be a maximum receipt rate of p_i at time t. We assume that $maxRR_i(t)$ is an invariant constant $maxRR_i$. p_i can receive at most $maxRR_i$ messages per a time unit. Let $RR_i(t)$ be a receipt rate of p_i at time t, i.e. how many messages p_i receives per a time unit at t. That is, p_i receives $RR_i(t)$ messages per a time unit at t. Here, $RR_i(t) \leq maxRR_i$.

p_i sends messages to the processes $p_1, \dots, p_n$ in G. $TR_i(t)$ denotes a transmission rate of p_i at time t. That is, p_i sends $TR_i(t)$ messages per a time unit at t. Let $maxTR_i$ be a maximum transmission rate of p_i which is time-invariant. Each message is destined to the destination processes, not necessarly all the processes in G. p_i sends $TR_{ij}(t)$ messages among $TR_i(t)$ messages to each p_j at t ($j = 1, \dots, n$). If p_i sends no message to p_j, $TR_{ij}(t)$ is 0. Thus, $TR_i(t)$ is given as follows:

- $TR_i(t) = TR_{i1}(t) + ... + TR_{in}(t)$.

$maxTR_{ij}(t)$ is a maximum transmission rate of p_i to p_j at t. Here, $TR_i(t) \leq maxTR_i(t)$ and $TR_{ij}(t) \leq maxTR_{ij}(t) \leq maxTR_i(t)$.

p_i receives messages from multiple processes while sending messages to multiple processes in G. Here, let δ_{ij} denote the propagation delay from a process p_i to p_j. Assume that $\delta_{ij} = \delta_{ji}$ for every pair of p_i and p_j in G and δ_{ij} is time-invariant. p_i receives messages at time t which p_j has sent to p_i at $t - \delta_{ij}$. Hence, totally $TR_{i1}(t - \delta_{i1}) + ... + TR_{in}(t - \delta_{in})$ messages arrive at p_i at time t [Figure 2]. $AR_{ij}(t)$ is an arrival rate showing how many messages arrive at p_i from p_j per a time unit at t. Let $AR_i(t)$ be an *arrival* rate of p_i at t which is given as follows:

- $AR_i(t) = AR_{i1}(t) + \ldots + AR_{in}(t)$.
- $AR_{ij}(t) = TR_{ji}(t - \delta_{ij})$.

If $AR_i(t) \leq maxRR_i$, p_i receives all the messages sent to p_i from the processes in G. That is, $RR_i(t) = AR_i(t)$. Otherwise, p_i loses some of the messages sent to p_i due to the buffer overrun, i.e. $RR_i(t) < AR_i(t)$. $maxRR_{ij}(t)$ is the maximum receipt rate of p_i from p_j, i.e. $RR_{ij}(t) \leq maxRR_{ij}(t)$. $maxRR_{i1}(t) + \cdots + maxRR_{in}(t) \leq maxRR_i$. If $AR_{ij}(t) > maxRR_{ij}(t)$, $RR_{ij}(t) = maxRR_{ij}(t)$. Here, p_i loses $AR_{ij}(t) - RR_{ij}(t)$ messages at t. In this paper, we make a following assumption on how many messages are lost due to the overrun.

[Assumption] If $AR_{ij}(t) > maxRR_{ij}(t)$, p_i loses $AR_{ij}(t) - maxRR_{ij}$ messages sent by p_j per a time unit at time t. □

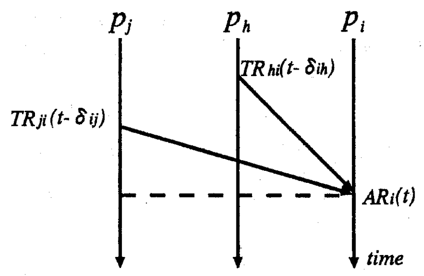

Figure 2: Arrival rate.

This assumption means that each p_i can receive $maxRR_{ij}(t)$ messages from p_j at every time t. That is, $RR_{ij}(t) = maxRR_{ij}(t)$ if $AR_{ij}(t) \geq maxRR_{ij}(t)$. $L_{ij}(t)$ is $AR_{ij}(t) - maxRR_{ij}(t)$ if $AR_{ij}(t) > maxRR_{ij}(t)$. A *loss* rate $L_i(t)$ shows how many messages per a time unit are lost by p_i. Even if $AR_i(t) < maxRR_i$, $L_{ij}(t)$ may not be zero. For example, p_i decides the maximum receipt rate $maxRR_{ij}(t)$ of messages from p_j. Here, $L_{ij}(t) > 0$ if $AR_{ij}(t) > RR_{ij}(t)$. $L_i(t)$ is defined as follows:

- $L_i(t) = L_{i1}(t) + \cdots + L_{in}(t)$.

Even if $RR_i(t) \leq maxRR_i$, $L_i(t)$ may not be zero since $L_{ij}(t) > 0$ for some j. Each p_i receives messages from $p_1, \ldots, p_n$ at the receipt rate $RR_i(t)$ at time t as presented here. $RR_{ij}(t)$ is a receipt rate of p_i from p_j. That is, p_i receives $RR_{ij}(t)$ messages from p_j per a time unit at t.

- $RR_i(t) = RR_{i1}(t) + \ldots + RR_{in}(t)$.

p_i control the receipt rate $RR_{ij}(t)$ for p_j by giving some part of $maxRR_i$ to $maxRR_{ij}(t)$. Let $L_{ij}(t)$ denote a loss rate of messages which p_j loses from p_i at t.

$$L_{ij}(t) = \begin{cases} AR_{ij}(t) - maxRR_{ij}(t) \\ \quad\quad if AR_{ij}(t) \geq maxRR_{ij}(t) \\ 0 \quad otherwise. \end{cases}$$

Let $maxRR$ be $maxRR_1$ +...+ $maxRR_n$. $maxRR$ is named a *total capacity* of the system. At most $maxRR$ messages can be transmitted at the same time in the network. Let $totalTR(t)$ be a *total transmission* rate of the system at time t. In the system, $totalTR(t)$ shows the amount of messages which all the processes are transmitting at time t. Let $totalAR(t)$ be a *total arrival* rate of the system at t. $totalAR(t)$ shows the number of messages which are arriving at the processes in G at t. Let $totalRR(t)$ be a *total receipt* rate of the system at t. Totally $totalRR(t)$ messages are received by the processes in the system at t.

- $totalTR(t) = TR_1(t) + \cdots + TR_n(t)$.
- $totalAR(t) = AR_1(t) + \cdots + AR_n(t)$.
- $totalRR(t) = RR_1(t) + \cdots + RR_n(t)$.

$totalRR(t)$ shows the *throughput* of the system at t. Here, $totalRR(t) \leq maxRR$ and $totalRR(t) \leq totalAR(t)$. Let $totalL(t)$ be a *total loss* rate of the system at t.

- $totalL(t) = L_1(t) + \cdots + L_n(t)$.

The *efficiency* $E(t)$ of the system at t is defined to be $totalRR(t)$ / $totalL(t)$. The larger $E(t)$ is, the more efficient the system is.

3.2 Stream rates

Each process p_i sends k_i (≥ 0) streams S_i^1, ..., $S_i^{k_i}$ in the group G at the same time. Messages decomposed from each stream S_i^h are transmitted to the destinations at some transmission rate $TR_i^h(t)$ ($h = 1,\ldots,k_i$).

- $TR_{ij}(t) = TR_{ij}^1(t) + \cdots + TR_{ij}^{k_j}(t)$.

$AR_{ji}^h(t) = TR_{ij}^h(t - \delta_{ij})$ which is the arrival rate of S_i^k from p_i to p_j. Let $RR_{ji}^h(t)$ denote a receipt rate of messages of S_i^h which p_i receives from p_j at t. $maxRR_{ji}^h(t)$ is the maximum receipt rate of $RR_{ji}^h(t)$, i.e. $RR_{ji}^h(t) \leq maxRR_{ji}^h(t)$. Let $L_{ji}^h(t)$ be a loss rate of messages in S_j^h which p_i loses from p_j at t.

- $AR_{ji}(t) = AR_{ji}^1(t) + \cdots + AR_{ji}^{k_j}(t)$.
- $RR_{ji}(t) = RR_{ji}^1(t) + \cdots + RR_{ji}^{k_j}(t)$.
- $L_{ji}(t) = L_{ji}^1(t) + \cdots + L_{ji}^{k_j}(t)$.
- $L_{ji}^h(t) = AR_{ji}^h(t) - RR_{ji}^h(t) \quad (h = 1, \cdots, k_j)$.

Here, even if $RR_{ji}(t) > AR_{ji}(t)$, $L_{ji}(t) > 0$ if p_i loses messages of some S_i^h, i.e. $L_{ji}^h(t) = AR_{ji}^h(t)$ -

$maxRR_j^h(t) > 0$. Even if $RR_{ji}(t) \leq maxRR_{ji}(t)$, $L_{ji}(t)$ may not be zero.

Let $|S_i^h|$ be a volume of S_i^h. Suppose that p_i starts to transmit messages of S_i^h to p_j at time t_1 and ends up at t_2, i.e. $\int_{t_1}^{t_2} RR_{ij}^h(t+\delta_{ij})dt = |S_i^h|$. If p_j loses some messages of S_i^h sent by p_i, p_i retransmits p_j the messages lost by p_j. Here, $\int_{t_1}^{t_2} TR_{ij}^h(t)dt \geq |S_i^h|$. Let ATR_{ij}^h be an average transmission rate of S_j^h, i.e. $ATR_{ij}^h = \int_{t_1}^{t_2} TR_{ij}^h(t)dt/(t_2 - t_1)$. Let ARR_{ij}^h be an average receipt rate of S_i^h, i.e. $ARR_{ij}^h = |S_i^h|/(t_2 - t_1)$.

Each process p_i can control the transmission of S_i^h by giving $TR_i^h(t)$. p_j controls the receipt of S_i^h by giving $maxRR_{ji}^h(t)$.

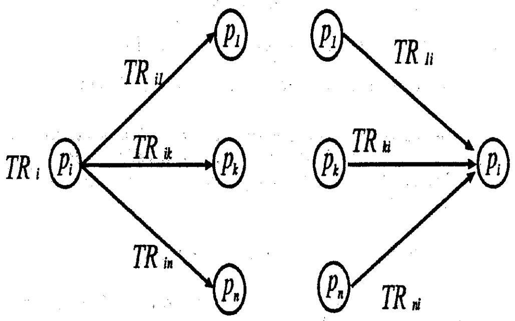

Figure 3: Transmission and receipt rates.

3.3 Fairness

The system has to *fairly* support the processes in the group $G = \{p_1, \cdots, p_n\}$ with the data transmission. First, let us consider the transmission rates of the processes. Suppose that p_i and p_j would send messages. Suppose that there are two systems T_1 and T_2 which support G with different rate-based flow control strategies. Let $TR_i^{(k)}(t)$ be a transmission rate of p_i by the system T_k at time t. T_1 is defined to be more *fair* than T_2 on the transmission rates of p_i and p_j at t if $|TR_i^{(1)}(t) - TR_j^{(1)}(t)| \leq |TR_i^{(2)}(t) - TR_j^{(2)}(t)|$. Here, let $VTR^{(k)}(t)$ be the variance of $TR_1^{(k)}(t)$, ..., $TR_n^{(k)}(t)$ in the system T_k.

[Definition] A system T_1 is more *fair* than T_2 on the transmission rate at time t iff $VTR^{(1)}(t) \leq VTR^{(2)}(t)$. □

In the most fair system, every process p_i transmits messages at the same rate, i.e. $TR_i(t) = TR_j(t)$ for every pair of p_i and p_j.

Here, let $VRR^{(k)}(t)$ and $VL^{(k)}(t)$ be the variances of the receipt rates and the loss rates of the processes in the system T_k at time t.

[Definition] T_1 is more fair than T_2 on the receipt rate and loss rate iff $VRR^{(1)}(t) \leq VRR^{(2)}(t)$ and $VL^{(1)}(t) \leq VL^{(2)}(t)$, respectively. □

3.4 Satisfiability

Distributed applications specify requirements on the transmission rates of messages in the group G. Let QR_{ij}^h be a transmission requirement for a stream S_i^h sent by p_i to p_j which shows how many messages of S_i^h per a time unit to be transmitted from p_i to p_j. QR_{ij} is a transmission requirement of messages sent by p_i to p_j.

- $QR_{ij} = QR_{ij}^1 + \cdots + QR_{ij}^{k_i}$.

In addition, there is a transmission requirement among the streams S_i^h and S_k^l. Suppose that S_i^h and S_k^l are sent to p_j. For example, applications require that p_j receive the messages of S_i^h and S_k^l at the same rate, i.e. $QR_{ij}^h = QR_{kj}^l$. p_j may be required to receive S_i^h three times faster than S_i^l, i.e. $Q_{ij}^h = 3Q_{ij}^l$. Thus, the relation among the streams $S_i^h, \ldots, S_j^l$ is represented by the relation among the receipt rates $Q_{ih}^h, \ldots, Q_{ij}^l$. The satisfiability $SQ_{ij}^h(t)$ of the system is defined to be $TR_{ij}^h(t)/QR_{ij}^h(t)$. The bigger $SQ_{ij}^h(t)$ is, the more satisfied p_i is for S_i^h. $SQ_{ij}(t)$ and $SQ_i(t)$ are defined as follows:

- $SQ_{ij}(t) = (SQ_{ij}^1(t) + \cdots + SQ_{ij}^{k_i}(t))/k_i$.
- $SQ_i(t) = (SQ_{i1}(t) + \cdots + SQ_{in}(t))/n$.

Let SQ_{ji} and SQ_i be averages of $SQ_{ij}(t)$ and $SQ_i(t)$, respectively.

4 Rate Control

We discuss how to allocate the transmission rates to the processes p_1, ..., p_n in the group G. Each p_i sends multiple streams $S_i^1, \ldots, S_i^{k_i}$ to multiple destination processes while p_i receives multiple streams from multiple processes in G. In this paper, we consider the following requirements:

(1) Transmission rate.
(2) Synchronization among streams.

4.1 Transmission rate

Each process p_i sends a stream S_i^h to the destination processes in G. For each stream S_i^h and each destination process p_j of S_i^h, a transmission rate requirement QR_{ij}^h is specified. p_i is required to send S_i^h to p_j at a transmission rate $TR_i^h(t) \geq QR_{ij}^h$. p_i has to decide $TR_i^h(t) = TR_{i1}^h(t) + \cdots + TR_{in}^h(t)$ so as to satisfy the following constraints:

- $\sum_{h=1}^{k_i} TR_i^h(t) \leq maxTR_i$.
- $TR_{ij}(t) \leq maxRR_{ji}(t)$ for every destination process p_j.

Before sending streams $S_i^1, \ldots, S_i^h$, each p_i sends the requirements $QR_{i1}^h, \ldots, QR_{in}^h$ for every S_i^h to $p_1, \ldots, p_n$ in G. p_i receives the transmission requirements from all the processes, i.e. $\{QR_{jh}^l | j, h = 1, \ldots, n,\ l = 1, \ldots, k_j\}$. Based on the rate requirements obtained, p_i decides $TR_{ij}(t)$ for each p_j and S_i^h.

(1) If $\sum_{l=1}^{n} QR_{lj} \leq maxRR_j$ for p_j, p_i sends messages of the stream S_i^h to p_j at a transmission rate $TR_{ij}^h(t) = QR_{ij}^h$. p_j receives the messages of S_i^h from p_i at a receipt rate $RR_{ji}^h(t) = QR_{ij}^h$.

(2) Otherwise, the transmission rate of p_i has to be changed as follows:
If $\sum_{l=1}^{n} QR_{lj} > maxRR_j$, $TR_{ij}(t) = QR_{ij}^h \times (maxRR_j / \sum_{l=1}^{n} QR_{lj})$.

In (2), some messages of some stream are lost by p_j due to the overrun. Each stream S_i^h has a priority $\pi(S_i^h)$ given by p_i. If p_j losses messages of S_i^h, p_j reduces the receipt rate of the lower-priority stream S_i^l to give more rates to $RR_{ji}^h(t)$. Here, suppose that messages are lost by p_i, $L_{ji}^h > 0$.

(1) If $RR_{ji}(t) \geq AR_{ji}(t)$, some parts of $RR_{ji}^{l_1}(t), \ldots, RR_{ji}^{l_k}(t)$ are moved to $RR_{ji}^h(t)$ if $\pi(S_i^{l_k}) < \pi(S_i^h)$. That is, $RR_{ji}^h(t) = RR_{ji}^h(t+\delta) + \alpha$ and $RR_{ij}^{l_k}(t+\delta) = RR_{ji}^{l_k}(t) - \alpha_k$ so that $RR_{ji}^h(t) \geq AR_{ji}^h(t)$. Here, $\alpha = \alpha_1 + \cdots + \alpha_n$ and $\alpha_k > \alpha_{k'}$ if $\pi(S_i^{l_k}) < \pi(S_i^{l_{k'}})$.

(2) If $RR_j(t) \geq AR_j(t)$, some parts of $RR_{ji}^{l_1}(t), \ldots, RR_{ji}^{l_k}(t)$ are moved to $RR_{ji}^h(t)$ if $\pi(S_j^{l_k}) < \pi(S_i^h)$, i.e. $RR_{ji}^h(t+\delta) = RR_{ji}^h(t) + \alpha$ and $RR_{ji}^{l_k}(t+\delta) = RR_{ji}^{l_k}(t) - \alpha_k$. Here, $\alpha = \alpha_1 + \cdots + \alpha_n$ and $\alpha_k > \alpha_{k'}$ if $\pi(S_i^{l_k}) < \pi(S_i^{l_{k'}})$.

4.2 Synchronization

Some streams transmitted are interrelated. First, p_i has to send messages of each stream S_i^h to each p_j so that p_j could receive all the messages of S_i^h as presented in the preceding subsection. In addition, we have to consider the relations among the streams. For example, suppose that p_i sends two streams, i.e. voice stream S_i^1 and image one S_i^2 to p_j. p_j is required to receive the messages of S_i^1 and S_i^2 in a synchronous mode. That is, it is required that $RR_i^1(t)$ / $RR_i^2(t)$ be invariant for every time t. In another example, suppose that p_j and p_k send streams S_j and S_k to p_i, respectively. p_i has to send S_j and S_k in a synchronous mode if S_j and S_k are related. Thus, each p_i has to send messages to the destination processes so as to satisfy the requirements. There are the following cases [Figure4]:

(1) p_i sends a steam S_i^h to $p_1, \ldots, p_n$ in G [Figure 4(1)]. For every pair of destinations p_j and p_k of S_i^h, $TR_{ij}^h(t) = c_{jk} \times TR_{ik}^h(t)$ where c_{jk}^h is a constant.

(2) p_i receives streams from $p_1, \ldots, p_n$ in G [Figure 4(2)]. Suppose that p_i receives a stream S_j^k from p_j and S_k^l from p_k, $RR_{ji}^h(t) = b_{jk} \times RR_{ki}^l(t)$ where b_{jk} is a constant.

In (1), if $c_{jk} = 1$, p_i sends the messages of S_i^h to every destination p_j at the same transmission rate. If there is no relation between p_j and p_k, p_i

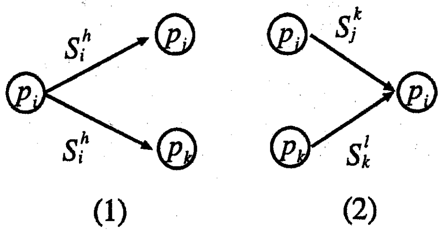

Figure 4: Related streams.

can transmit messages of S_i^h to p_j and p_k at different transmission rates $TR_{ij}^h(t)$ and $TR_{ih}^h(t)$ so that $TR_{ij}^h(t) \leq maxRR_j$ and $TR_{ij}^h(t) \leq maxRR_k$. In (2), if $b_{jk} = 1$, p_i has to receive the messages of S_j^h and S_k^l at the same receipt rate, i.e. $TR_{ij}^h(t) = TR_{ik}^l(t)$. Here, p_i receives the messages in a synchronous mode. If the overrun occurs in the receiver process, the transmission rate is decreased so as to satisfy the constraints (1) and (2).

Every process p_i negotiates with the other processes before transmitting the messages of the stream. Each process p_i first sends the transmission requirement $QR_i^h = \langle QR_{i1}^h, \cdots, QR_{in}^h \rangle$ to all the processes $p_1, \ldots, p_n$ in G before sending the messages of a stream S_i^h. The requirement QR_i includes QR_i^h for each destination p_j. QR_i^h also includes the inter-stream requirement in from $\langle QR_{ij}^h, QR_{ik}^h, c_{jk}^h \rangle$. On receipt of QR_{ij}^h from p_j, p_j decides the maximum receipt rate $maxRR_{ji}^h$ for S_i^h. If there is no conflict among the stream requirements in p_j, $maxRR_{ji}^h = QR_{ij}^h$. Otherwise, p_j has to do the negotiation with p_i and the processes with which p_j is communicating.

5 Implementation

The algorithms for allocating the transmission rates to the processes $p_1, \ldots, p_n$ in the group G are implemented in Sparc workstations intercorrected by the fast Etherenet 100baseTX [Figure 5], where each p_i is in one station. Each process p_i has four modules A_i, V_i, WB_i, and GR_i. The modules A_i, V_i, and WB_i are application interfaces for audio, visual, and whiteboard media, respectively, of p_i. A_i, V_i, and WB_i send and receive audio, visual, and whiteboard streams, respectively to the other processes in G. GR_i is a module of the group communication protocol [15–17]. GR_i is composed of two modules, a group agent AG_i and a data transmission module TR_i. AG_i does the negotiation with the agents in the other processes in G to allocate the transmission rates to the processes. TR_i sends audio, video, and whiteboard streams to each destination at the rates obtained by the negotiation of AR_i module. TR_i also receives the streams from the other processes at the rates negotiated. TR_i sends and receives the messages at the rates allocated by AG, and delivers the messages in the causal order.

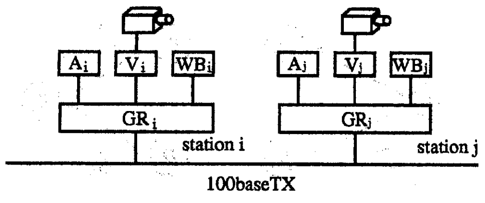

Figure 5: System model

6 Concluding Remarks

The traditional rate-based flow control schemes discuss how one process sends messages to one destination process or multicasts them to a group of multiple processes. In the group communication, each process not only sends messages to multiple processes but also receives messages sent by multiple processes in the group. In this paper, we have presented the general models of the group communication where the processes transmit multiple streams of multimedia data which are interrelated. In addition, we have discussed how to allocate the transmission rates to the processes in the group.

References

[1] Birman, K., Schiper, A., and Stephenson, P., "Lightweight Causal and Atomic Group Multicast," *ACM Trans. on Computer Systems*, Vol.9, No.3, 1991, pp.272–314.

[2] Cheriton, D. R., "VMTP: A Transport Protocol for the Next Generation of Communication Systems," *Proc. of the ACM SIGCOMM'86*, 1986, pp.406–415.

[3] Chesson, G., "XTP/PE Overview," *Proc. of the 13th IEEE Conf. on Local Computer Networks*, 1988, pp.292–296.

[4] Clark, D. D., Lambert, M. L., and Zhang, L., "NETBLT: A High Throughput Transport Protocol," *Proc. of the ACM SIGCOMM'87*, 1987, pp.353–359.

[5] Doeringer, W. A., Dykeman, D., Kaiserswerth, M., Meister, B. W., Rudin, H., and Williamson, R., "A Survey of Light-Weight Transport Protocols for High-Speed Networks," *IEEE Trans. on Communications*, Vol.38, No.11, 1990, pp.2025–2039.

[6] Ellis, C. A., Gibbs, S. J., and Rein, G. L., "Groupware," *Comm. ACM*, Vol.34, No.1, 1991, pp.38-58.

[7] IEEE802.3z Recording Secretary, "Overview & Guide TO IEEE802/LMSC," 1996, pp.802–96/02D.

[8] Lamport, L., "Time, Clocks, and the Ordering of Events in a Distributed System," *Comm. ACM*, Vol.21, No.7, 1978, pp.558–565.

[9] Melliar-Smith, P. M., Moser, L. E., and Agrawala, V., "Broadcast Protocols for Distributed Systems," *IEEE Trans. on Parallel and Distributed Systems*, Vol.1, No.1, 1990, pp.17-25.

[10] Nakamura, A. and Takizawa, M., "Starvation Prevented Priority-Based Total Ordering (PriTO) Protocol on High-Speed One-Channel Network," *Proc. of the 2nd IEEE Int'l Symp. on High-Performance Distributed Computing (HPDC-2)*, 1993, pp.281–288.

[11] Nakamura, A. and Takizawa, M., "Priority-Based Total and Semi-Total Ordering Broadcast Protocols," *Proc. of ICPADS-92*, 1992, pp.178–185.

[12] Nakamura, A. and Takizawa, M., "Causally Ordering Broadcast Protocol," *Proc. of the IEEE ICDCS-14*, 1994, pp.48–55.

[13] Schulzrinne, H., Casner, S., Frederick, R., and Jacobson, V., "RTP: A Transport Protocol for Real-Time Applications," *RFC-1889*, 1996.

[14] Strayer, W. T., Dempsey, B. J., and Weaver, A. C., "XTP: The Xpress Transfer Protocol (XTP)," *Addison-Wesley*, 1992.

[15] Tachikawa, T. and Takizawa, M., "Selective Total Ordering Broadcast Protocol," *Proc. of IEEE ICNP-94*, 1994, pp.212–219.

[16] Tachikawa, T. and Takizawa, M., "Multimedia Intra-Group Communication Protocol," *Proc. of IEEE HPDC-4*, 1995, pp.180–187.

[17] Tachikawa, T. and Takizawa, M., "Communication Protocol for Wide-area Group," *Proc. of the 11th Int'l Conf. on Information Networks ICOIN-11*, 1997, pp.3D-5.1–3D-5.9.

[18] Zhang, L., S.E. Deering, D.Estrin, S.Shenker, and D. Zappala., "RSVP: A New Resource ReSerVation Protocol," *IEEE Network Magazine*, Vol.9, No.5, 1993.