マルチメディア通信と分散処理 85－2
（1997. 11. 6）

ARQ Protocols for Bi-directional Data Transmission in Group Communications

Takayuki Tachikawa and Makoto Takizawa
{tachi, taki}@takilab.k.dendai.ac.jp
Dept. of Computers and Systems Engineering
Tokyo Denki University

Automatic repeat request (ARQ) protocols are discussed so far to analyze the buffer size and the delay required for two peer processes. However, ARQ protocols in group communications are not discussed, where multiple senders send packets to multiple receivers. In the group communication protocols, acknowledgments of packets received are piggy-backed by packets. Hence, the forward and feedback channels are considered to be combined to one bi-directional channel. In addition, since each process has the buffer both for retransmitting and for resequencing packets sent and received, efficient buffer allocation schemes have to be adopted. In this paper, we analyze the throughput of the ARQ protocol for the group communications.

グループ通信における双方向データ転送のためのARQプロトコル

立川 敬行　滝沢 誠
東京電機大学理工学部経営工学科

従来の多くの ARQ プロトコルは、複数の送受信が存在するグループ通信に対してあまり議論されてこなかった。グループ通信では、確認メッセージがデータ・メッセージにピギーバックされるため、フォワード・チャネルとフィードバック・チャネルを同種のチャネルとみなすことができる。さらに、各プロセスは送信と受信用の両方のバッファを持つため、これらのバッファの効果的な配分を考える必要がある。本論文では、このようなグループ通信における ARQ プロトコルの解析を行う。

1 Introduction

Automatic repeat request (ARQ) protocols [1–5, 9, 10] are used to reliably deliver packets to the destination process in the correct sequence. There are following three basic types of ARQ protocols:

- Stop and Wait (SW).
- Go Back N (GBN) [9].
- Selective Retransmission (SR) [2].

These basic protocols are analyzed and expanded so far, e.g. multiple copies scheme [4] and postponed retransmission scheme [1]. Most of them are discussed to realize reliable delivery of packets from one transmitter to one receiver by using a uni-directional data channel. There are two kinds of channels, i.e. *forward* and *feedback* ones between the transmitter and the receiver. The data is transmitted only from the transmitter to the receiver through the forward channel but the receiver does not transmit data. The receiver sends ACK and NACK to the transmitter through the feedback channel. The transmitter stores packets which the transmitter sends in the retransmission buffer. If the receiver fails to receive packets, the receiver sends NACK to the transmitter and then the transmitter retransmits the packets stored in the retransmission buffer. Since the packets arrive at the receiver out of the transmission order, the receiver has a buffer for resequencing the packets received. In order to exchange packets between two processes, i.e. both processes transmit data packets, two independent uni-directional channels are needed. Therefore, we discuss the bi-directional channel which can be used as both the forward and feedback channels. If two processes use the bi-directional channel, each process has both the retransmission and resequencing buffers to send and receive packets. In addition, the ACK or NACK information of a data packet m is piggy-backed by data packets which a process sends after receiving m. While most of previous approaches [1, 5] assume that the transmitter always has new packets to transmit and transmits data packets at the constant rate, we assume that the transmission rates required by two processes are not same and they are furthermore dynamically changing. We try to allocate retransmission and resequencing buffers dynamically so as to meet their desired transmission rates.

In group communications [6], a group is composed of multiple processes, where multiple processes transmit data packets to multiple receivers. If a process P_i allocates all the available buffers to receive packets from a process P_j, P_i cannot allocate any buffer to receive from other process. Hence, we can not consider that the group communication protocol is a simple combination of multicast protocols [7, 11]. Each process has both the retransmission and resequencing buffers for each process in the group. We consider how each process allocates the available buffers for every other process in the group.

In this paper, we discuss two points of the ARQ protocol. First, we consider how to allocate a pool of available buffers to the retransmission and the resequencing buffers in the one-to-one communication. Second, we consider the buffer management with the group communication. Here, we discuss how to allocate the available buffers to the retransmission and resequencing buffers in the change of the transmission rates of the processes in the group.

In section 2, we present the system model. In section 3, we discuss the buffer allocation in the one-to-one communication. In section 4, we discuss the ARQ protocol for the group communication.

2 System Model

A communication system is composed of processes interconnected by communication channels. Each process can send not only fixed length data packets with acknowledgment (ACK) information, i.e. piggy back, but also acknowledgment packets which do not include data. Each channel is not reliable. Each packet contains error detection and correction bits like CRC. Hence, we assume that only packet loss occurs in the network due to congestions and buffer overruns.

There are four types of communication among processes in the system as shown in Figure 1. The first type is one-to-one communication which is discussed in the ARQ protocols [1–5, 7, 9–11]. Here, the channel is one way. That is, one transmitter sends data packets to another receiver. The third type is one-to-many communication. It is also named a *multicast* (multi-receivers) communication. There are one transmitter and multiple receivers. The transmitter sends data packets to the receivers. This type can be considered to be a combination of the one-to-one communication from the transmitter to each receiver. In the bi-directional one-to-one communication type, two processes exchange data packets. Each process plays two roles, i.e. transmitter and receiver. There is a bi-directional channel between the processes. In the one-to-one and one-to-many communications, the receiver has to receive all the data packets sent by transmitter in the sending, i.e. FIFO order in the presence of the packet loss. In the fourth *group* communication, each process sends data packets to multiple processes while each process receives from multiple processes. There are logically a bi-directional channel between every pair of processes. In addition to the FIFO order, the processes have to receive the data packets in the *causal* order [8] and total order [6]. In this paper, we consider ARQ protocols in (2) one-to-one (bi-directional) and (4) group communications.

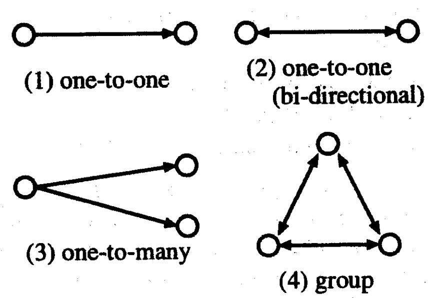

Figure 1: Types of communications.

3 One-to-One Communication

First, let us consider that two processes P_1 and P_2 communicate in the one-to-one channel. One process P_i sends packets to P_j. Since the channel is not reliable, P_j may fail to receive the packets sent by P_i. Here, P_i retransmits the packets to P_j. In order to retransmit the packets, P_i has to have to buffer the packets which P_i has sent but whose ACK P_i has not received. The buffer is named a retransmission buffer TB_i. On the other hand, P_j has to receive the packets sent by P_i in the sending order. Due to the loss and retransmission of the packets, P_j may receive the packets out of the sending order. P_j has to have another buffer to store the packets received in order to resequence the packets. This buffer is named a resequencing buffer SB_j. Thus, each process has two types of buffers; retransmission buffer TB_i and resequencing buffer SB_i for the one-to-one channel. Figure 2(1) and (2) show the one-to-one communication between P_1 and P_2 using a uni-directional channel and bi-directional channel, respectively. Here, let RTB_i and RSB_i be the sizes of the buffers TB_i and SB_i in P_i, respectively. Let us consider two processes P_1 and P_2 communicate in the one-to-one channel as shown in Figure 2. In the uni-directional channel, only the transmitter P_1 sends data packets to the receiver P_2. Hence, the size RTB_1 of the retransmission buffer TB_1 of the transmitter P_1 and the size RSB_2 of the resequencing buffer SB_2 of the receiver P_2 are same, i.e. $RTB_1 = RSB_2$.

In the bi-directional channel, P_1 and P_2 send and receive data packets. Hence, $RTB_1 = RSB_2$ and $RTB_2 = RSB_1$. The total size of buffers in the process is fixed and equal to N, i.e. $RTB_1 + RSB_1 = RTB_2 + RSB_2 = N$. Each process can have one pool AVB of available buffers for retransmitting and resequencing data packets sent and received. At the beginning of transmission, half of AVB is initially allocated to RTB_i and the another half to RSB_i in each process P_i, i.e. $RTB_1 = RSB_1 = RTB_2 = RSB_2 = N/2$. If the desired transmission rate of the process P_1 is decreased, P_1 decreases the retransmission buffer size RTB_1 and increases the resequencing buffer size RSB_1. P_1 notifies another process P_2 of the change of buffer size. Then, P_2 decreases RSB_2 and increases RTB_2. Conversely, if the desired transmission rate of P_1 is increased, P_1

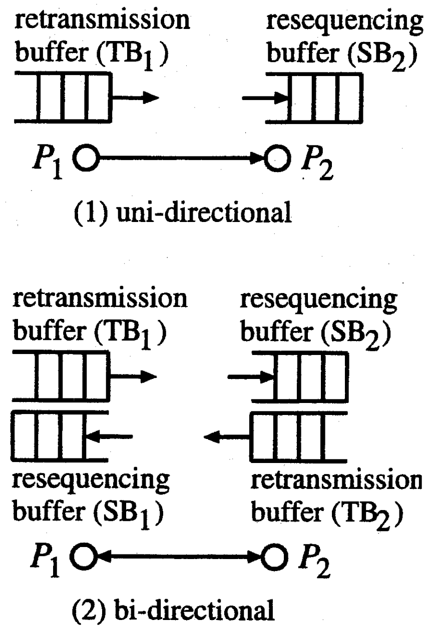

Figure 2: Buffering capacities.

increases RTB_1 and decreases RSB_1, and then notifies P_2 of it. If P_2 receives the notification from P_1, P_2 immediately increases RSB_2 and decreases RTB_2.

Suppose that two independent uni-directional channels are used for P_1 and P_2 to communicate with one another in stead of one bi-directional channel. Each process has to manage the buffers of one channel independently of the other channel. That is, the process P_i cannot share the available buffer pool AVB to buffer packets sent and received. On the other hand, if the bi-directional channel is used, each process P_i has one pool of available buffers which are used to buffer packets sent and received. Here, the sizes of TB_i and SB_i can be dynamically changed in order to increase the throughput. Hence, RTB_i and RSB_i are maintained to be proportional to the current desired transmission rates of P_i and P_j, respectively.

Let us consider a case that $N = 6$ as shown in Figure 3. First, $RTB_A = RSB_A = RTB_B = RSB_B = 3$. Both sequence number (SEQ) and acknowledgment number (ACK) are piggy-backed by each data packet. A sends packets whose sequence nubers are 1, 2, and 3. The data packets are stored in TB_A. Then, A stops sending data packets and waits for a packet whose acknowledgment number (ACK) is larger than 1. On the other hand, B sends packets to A. B sends a packet m with SEQ = 2 after receiving the packet with SEQ = 1 from A. Here, the packet m carries ACK = 1 to A. On receipt of m, A knows that B receives the packets where SEQ $\leq$ 1. Here, A receives the first packet from TB_A and TB_A can include one more packet. A starts to send data packet to B.

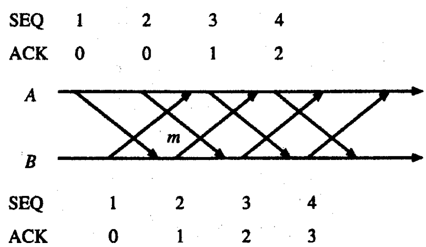

Figure 3: Basic data transmission.

Since the channels are less reliable, some packets are lost in the channel. If a packet is lost in bi-directional channel, both the data and the acknowledgement information carried by the packet are lost. In Figure 4, the second packet sent by B is lost. The packet carries data with the receipt acknowledgment of the first packet sent by A, i.e. ACK = 1. Each process has to check a sequence number of packets to detect the packet loss. After A sends a packet whose sequence number (SEQ) is 3, A waits to receive a packet whose acknowledge number (ACK) is larger than 1. When A receives a packet whose SEQ is 3 and ACK is 2, A detects a loss of a packet whose SEQ is 2. Since A knows that B receives two packets whose SEQs are 1 and 2, A removes the packets from TB_A. Here, A can send two more packets. A sends a packet whose SEQ is 4 with ACK = 1. When B receives the packet, B retransmits a packet whose SEQ is 1. The way to retransmit lost packets depends on the type of protocols, i.e. go-back-n [9]or selective retransmission [2]. In the go-back-n method, B retransmits these packets where SEQ = 2 and 3. In the selective retransmission, only second packet whose SEQ = 2 is retransmitted.

Next, let us consider a case that the desired buffer size is dynamically changing. Suppose A decreases RTB_A to 2 and increases RSB_A to 4 in Figure 5. Then, A notifies B of it by a packet

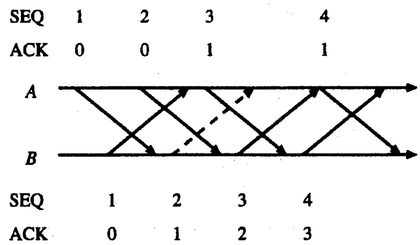

Figure 4: Packet loss.

whose SEQ is 10. When B receives the packet, B changes RTB_B to 4 and RSB to 2 if there are less than two packets stored in SB_B. After changing RSB_B, B can send four packets without A's acknowledgment information. On the other hand, A can send only two packets without B's acknowledgment information. The third packet sent by A is an ACK packet, i.e. no-data is included.

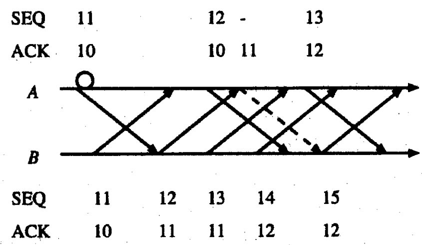

Figure 5: Buffer size changing.

4 Group Communication

In the group communication, the group is composed of n (≥ 2) processes P_1, ..., P_n where every pair of the processes are interconnected by a bidirectional channel. Each process P_i can transmit data packets to $n - 1$ processes P_1, ..., P_{i-1}, P_{i+1}, ..., P_n in the group. Hence, each process receives packets from $n - 1$ processes as shown in Figure 6. As presented in the one-to-one communication, packets may be lost in the channel. If a packet m is lost, m is retransmitted by the transmitter P_i of m.

On sending m, P_i stores m in the retransmission buffer TB_i until P_i receives the acknowledgment of m from all the other processes. Each process has to buffer packets received in order to sequence the packets in the sending order. In addition, the packets have to be sequenced in the *causal order* [6, 8] in the group communication. In this paper, we discuss a way that every process receives all the packets sent by each process in the sending order. Thus, the packets received by P_i are buffered in the resequencing buffer SB_i of P_i.

P_i has a pool of available buffers AVB. On receipt of a data packet m from P_j, P_i obtains one available buffer from AVB, stores m in the buffer, and puts it in the resequencing buffer SB_i. P_i obtains the available buffer from AVB to buffer packets received by every other process in the group.

The desired transmission rate of each process is fratuated. That is, P_i may send more data packets at some time but may not send at another time. P_i allocates the buffer size to TB_i and SB_i in proportion to the transmission rates of P_i and the other processes, respectively.

P_i stores the packets received by the other processes into one resequencing buffer SB_i. Here, let RSB_{ij} show the size of SB_i which is used to store the packets from P_j. $RSB_i = RSB_{i1} + \cdots + RSB_{in}$. At the beginning of transmission, each RSB_{ij} is equal for each process P_j, i.e. N/n. $RTB_i = N/n$. During the transmission, each process P_i changes the transmission rates of the packets to P_j if the amount of data which P_i sends to P_j is changed. P_i notifies the other processes of the change of the transmission rate. On receipt of the notification, P_j changes RSB_{ji} so as to meet the transmission rate required by P_i.

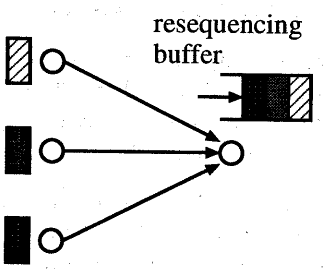

Figure 6: Packets buffering from multiple sender.

Let us consider a group comosed of three processes A, B, and C. Suppose that the available buffer size for each process is 6, i.e. $N = 6$. At the begining of transmission, the resequencing buffer is equally allocated to each process in the group, i.e. $RSB_{AB} = RSB_{AC} = RSB_{BA} = RSB_{BC} = RSB_{CA} = RSB_{CB} = N/3 = 2$. The size of retransmission buffer is the same as the size of the resequencing buffer for each process, i.e. $RTB_A = RSB_{BA} = RSB_{CA} = N/3 = 2$. After that, suppose that A decreases RTB_A from 2 to 1. Since $RTB_A = RSB_{BA} = RSB_{CA}$, B and C can decrease RSB_{BA} and RSB_{CA} from 2 to 1, respectively if the number of packets stored in the resequencing buffer is smaller than 2. Now, B and C have one available buffer to allocate. Here, suppose that B needs more transmission rate than C. C increases RTB_C to 3. RSB_{AB} and RSB_{CB} are also increased to 3 by A and C, respectively. Next, suppose that A has no packet to send at some time. A decreases RTB_A to 0, and notifies it to B and C. C would like to send packets at more transmission rate, and C notifies B. If B accepts it, C increases RTB_C to 3, and other buffer sizes are chaned as shown in Figure 7.

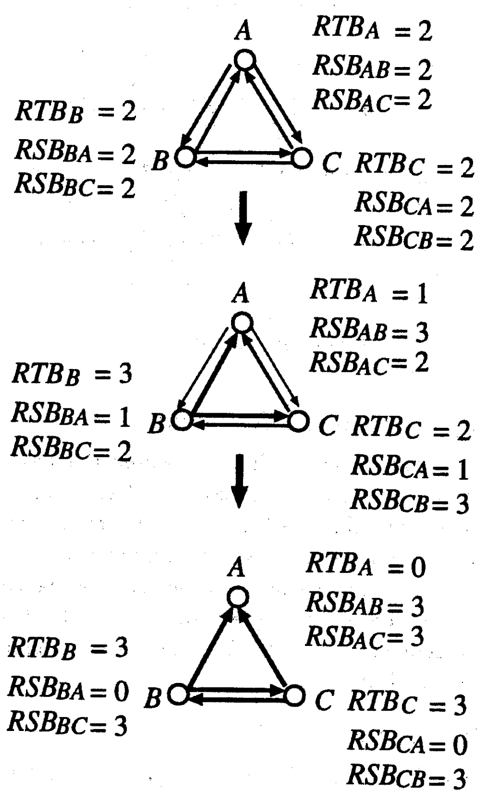

Figure 7: Dynamic buffer allocation.

On the other hand, from the transmitter point of view, a packet m multicasted by a transmitter P_i has to be stored in the retransmission buffer until P_i receives the acknowledgment of m meaning that every process in the group receives m.

That is, packets multicasted in the group have to be *atomically* received [6] by every process. In Figure 8, A sends a packet m to B and C and stores m in RTB_A. On receipt of m sent by A, B, and C send back the packets with the acknowledgment information of m. A removes m from TB_A only if A receives the acknowledgmwnts of m from B and C. In Figure 8, A knows that both B and C receive the first packet sent by A at the plus (+) mark where A receives the packets from B and C whose $ACK_A = 1$ and $ACK_A = 2$, respectively. A removes the first packet from TB_A. Then, at the star (*) mark, A receives a packet from B whose $ACK_A = 3$. Here, A knows that the second packet with SEQ = 2 sent by A is received by B and C. Then second packet is removed from TB_A. However, the third packet with SEQ = 3 is still stored in TB_A.

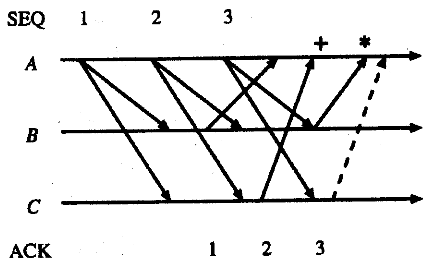

Figure 8: Atomic receipt.

5 Concluding Remarks

In this paper, we have presented the way to share both the retransmission and resequencing buffers by using the bi-directional channel. This method is suitable for two-way communications in the presence of desired transmission rate changes. In addition, we have presented the way to share the resequencing buffer for each process in group communications.

References

[1] Cam, R. and Leung, C., "Throughput Analysis of Some ARQ Protocols in the Presence of Feedback Errors," *IEEE Trans. on Communication*, Vol. 45, No. 1, 1997, pp. 35–44.

[2] Chang, J. and Yang, T., "End-to-End Delay of an Adaptive Selective Repeat ARQ Protocol," *IEEE Trans. on Communication*, Vol. 42, No. 11, 1994, pp. 2926–2933.

[3] Djuknic, G. and Schilling, D., "Performance Analysis of an ARQ Transmission Scheme for Meteor Burst Communications," *IEEE Trans. on Communication*, Vol. 42, No. 2/3/4, 1994, pp. 268–271.

[4] Kallel, S. and Leung, C., "Efficient ARQ Schemes with Multiple Copy Decoding," *IEEE Trans. on Communication*, Vol. 40, No. 3, 1992, pp. 642–650.

[5] Kallel, S., "Efficient Hybrid ARQ Protocols with Adaptive Forward Error Correction," *IEEE Trans. on Communication*, Vol. 42, No. 2/3/4, 1994, pp. 281–289.

[6] Nakamura, A., Tachikawa, T., and Takizawa, M., "Group Communication Protocols: Properties and Evaluation," *Proc. of IEEE Workshop on Services in Distributed and Networked Environments (SDNE)*, 1994, pp.139–146.

[7] Sakakibara, K. and Kasahara, M., "Multicast Hybrid ARQ Scheme Using MDS Codes and GMD Decoding," *IEEE Trans. on Communication*, Vol. 43, No. 12, 1995, pp. 2933–2940.

[8] Tachikawa, T. and Takizawa, M., "Distributed Protocol for Selective Intra-group Communication," *Proc. of the 3rd IEEE International Conf. on Network Protocols (ICNP-95)*, 1995, pp.234–231.

[9] Yao, Y., "An Effective Go-Back-N ARQ Scheme for Variable-Error-Rate Channel," *IEEE Trans. on Communication*, Vol. 43, No. 1, 1995, pp. 20–23.

[10] Yoshimoto, M., Takine, T., Takahashi, Y., and Hasegawa, T. "Waiting Time and Queue Length Distributions for Go-Back-N and Selective-Repeat ARQ Protocols," *IEEE Trans. on Communication*, Vol. 41, No. 11, 1993, pp. 1687–1693.

[11] Wang, J. and Silvester, J., "Optimal Adaptive Multireceiver ARQ Protocol," *IEEE Trans. on Communication*, Vol. 41, No. 12, 1993, pp. 1816–1829.