Throughput Stability of Reliable Multicast Protocols *

Öznur Özkasap[1] Kenneth P. Birman[2]

[1] Ege University, Department of Computer Engineering, 35100 Bornova, Izmir, Turkey
ozkasap@bornova.ege.edu.tr
[2] Cornell University, Department of Computer Science, 14853 Ithaca, NY, USA
ken@cs.cornell.edu

Abstract. Traditional reliable multicast protocols depend on assumptions about flow control and reliability mechanisms, and they suffer from a kind of interference between these mechanisms. This in turn affects the overall performance, throughput and scalability of group applications utilizing these protocols. However, there exists a substantial class of distributed applications for which the throughput stability guarantee is indispensable. Pbcast protocol is a new option in scalable reliable multicast protocols that offers throughput stability, scalability and a bimodal delivery guarantee as the key features. In this paper, we focus on the throughput stability of reliable multicast protocols. We describe an experimental model developed for Pbcast and virtually synchronous protocols on a real system. We then give the analysis results of our study.

1 Introduction

Several distributed applications require reliable delivery of messages or data to all participants. Example applications include electronic stock exchanges, air traffic control systems, health care systems, and factory automation systems. Multicast is an efficient communication paradigm and a reliable multicast protocol is the basic building block of such applications. Communication properties and the degree of reliability guarantees required by such applications differ from one setting to another. Thus, reliable multicast protocols can be broadly divided into two classes, based on the reliability guarantees they provide. One class of protocols offers strong reliability guarantees such as atomicity, delivery ordering, virtual synchrony, real-time support, security properties and network-partitioning support. The other class offers support for best-effort reliability in large-scale settings.

For large-scale applications such as Internet media distribution, electronic stock exchange and distribution of radar and flight track data in air traffic control systems, the throughput stability guarantee is extremely important. This property entails the steady delivery of multicast data stream to correct destinations. For instance, Internet media distribution applications, that transmit media such as TV and radio, or teleconferencing data over the Internet, disseminate media with a steady rate. An important requirement is the steady delivery of media to all correct participants in

* This work was partially supported by a TÜBITAK-NATO A2 grant.

spite of possible failures in the system. Another application group is electronic stock exchange and trading environments like the Swiss Exchange Trading System (SWX) [1]. Similarly, such applications use multicast communication protocols to disseminate trading information to all participants at the same time and with minimal delay. Throughput instability problem applies to both classes of reliable multicast protocols that we mentioned.

In this study, we focus on a new option in reliable multicast protocols. We call this protocol Bimodal Multicast, or Pbcast (probabilistic multicast) for short [2]. Pbcast offers throughput stability, scalability and a bimodal delivery guarantee. The protocol is based on an epidemic loss recovery mechanism. It exhibits stable throughput under failure scenarios that are common on real large-scale networks. In contrast, this kind of behavior can cause other reliable multicast protocols to exhibit unstable throughput.

In this study, we develop an experimental model for Pbcast protocol and virtually synchronous reliable multicast protocols offering strong reliability guarantees. We construct several group communication applications using these protocols on a real system. The aim is to investigate protocol properties, especially the throughput stability and scalability guarantees, in practice. The work has been performed on the IBM SP2 Supercomputer of Cornell Theory Center that offers an isolated network behavior. We use emulation methods to model process and link failures. Ensemble system has been ported on SP2, and a detailed analysis study of Pbcast protocol and its comparison with Ensemble's virtual synchrony protocols has been accomplished.

The paper is organized as follows: Section 2 describes reliability properties offered by two broad classes of multicast protocols, and Pbcast protocol. In section 3, we describe the throughput stability requirement and causes of the instability problem. Section 4 presents our experimental model and settings. Section 5 includes analysis and results of the study. Section 6 concludes the paper.

2 Reliability Properties of Multicast Protocols

Reliability guarantees provided by multicast protocols split them into two broad classes: *Strong reliability* and *best-effort reliability*. There is a great deal of work on communication tools offering protocols with strong reliability guarantees. Example systems include Isis [3,4], Horus [5,6], Totem [7], Transis [8] and Ensemble [9]. The other class of protocols offers support for best-effort reliability in large-scale settings. Example systems are Internet Muse protocol for network news distribution [10], the Scalable Reliable Multicast (SRM) protocol [11], the Pragmatic General Multicast (PGM) protocol [12], and the Reliable Message Transfer Protocol (RMTP) [13,14]. A new option in the spectrum of reliable multicast protocols is the Pbcast protocol [2]. We now describe basic properties offered by these multicast protocols.

Among the key properties provided as strong reliability guarantees are atomicity, ordered message delivery, real-time support and virtual synchrony. *Atomicity* means that a multicast message is either received by all destinations that do not fail or by none of them. Atomicity, which is also called all-or-nothing delivery, is a useful property, because a process that delivers an atomic multicast knows that all the

operational destinations will also deliver the same message. This guarantees consistency with the actions taken by group members [15]. Some applications also require o*rdered message delivery.* Ordered multicast protocols ensure that the order of messages delivered is the same on each operational destination. Different forms of ordering are possible such as FIFO, causal and total ordering. The strongest form among these is the total order guarantee that ensures that multicast messages reach all of the members in the same order [16]. Distributed real-time and control applications need *real-time support* in reliable multicast protocols. In these systems, multicast messages must be delivered at each destination by their deadlines. The *virtual synchrony* model [17] was introduced in the Isis system. In addition to message ordering, this model guarantees that membership changes are observed in the same order by all the members of a group. In addition, membership changes are totally ordered with respect to all regular messages. The model ensures that failures do not cause incomplete delivery of multicast messages. If two group members proceed from one view of membership to the next, they deliver the same set of messages in the first view. The virtual synchrony model has been adopted by various group communication systems. Examples include Transis [8], and Totem [7].

The other category includes scalable reliable multicast protocols that focus on best-effort reliability in large-scale systems. Basic properties offered are: *best-effort delivery*, *scalability* as the number of participants increases, *minimal delivery latency* of multicast messages. This class of protocols overcomes message loss and failures, but they do not guarantee end-to-end reliability. For instance, group members may not have a consistent knowledge of group membership, or a member may leave the group without informing the others.

Pbcast, which is a new option in reliable multicast protocols, is constructed using a novel gossip based transport layer. The transport layer employs random behavior to overcome scalability and stability problems. Higher level mechanisms implementing stronger protocol properties such as message ordering and security can be layered over the gossip mechanisms. In this paper, we do not go into details of the protocol. Detailed information on Pbcast is given in [2]. The protocol has the following properties:

Bimodal delivery: The atomicity property of Pbcast has a slightly different meaning than the traditional 'all-or-nothing' guarantee offered by reliable multicast protocols. Atomicity is in the form of 'almost all or almost none', which is called bimodal delivery guarantee.

Message ordering: Each participant in the group delivers Pbcast messages in FIFO order. In other words, multicasts originated from a sender are delivered by each member in the order of generation. As mentioned in [18], stronger forms of ordering like total order can be provided by the protocol. [19] includes a similar protocol providing total ordering.

Scalability: As the network and group size increase, overheads of the protocol remain almost constant or grow slowly compared to other reliable multicast protocols. In addition, throughput variation grows slowly with the log of the group size.

Throughput stability: Throughput variation observed at the participants of a group is low when compared to multicast rates. This leads to steady delivery of multicast messages at the correct processes.

Multicast stability detection: Pbcast protocol detects the stability of multicast messages. This means that the bimodal delivery guarantee has been achieved. If a message is detected as stable, it can be safely garbage collected. If needed, the application can be informed as well. Although some reliable multicast protocols like SRM do not provide stability detection, virtual synchrony protocols like the ones offered in Ensemble communication toolkit include stability detection mechanisms.

Loss detection: Because of process and link failures, there is a small probability that some multicast messages will not be delivered by some processes. The message loss is common at faulty processes. If such an event occurs, processes that do not receive a message are informed via an up-call.

3 Throughput Stability

For large-scale distributed applications that motivate our work, the throughput stability guarantee is extremely important. This property entails the steady delivery of multicast data stream to correct destinations.

Traditional reliable multicast protocols depend on assumptions about response delay, failure detection and flow control mechanisms. Low-probability events caused by these mechanisms, such as random delay fluctuations in the form of scheduling or paging delays, emerge as an obstacle to scalability in reliable multicast protocols. For example, in a virtual synchrony reliability model, a less responsive member exposing such events can impact the throughput of the other healthy members in the group. The reason is as follows. For the reliability purposes, such a protocol requires the sender to buffer messages until all members acknowledge receipt. Since the perturbed member is less responsive, the flow control mechanism begins to limit the transmission bandwidth of the sender. This in turn affects the overall performance and throughput of the multicast group. In effect, these protocols suffer from a kind of interference between reliability and flow control mechanisms. Moreover, as the system size is scaled up, the frequency of these events rises, and this situation can cause unstable throughput.

An observation on the throughput instability problem of reliable multicast protocols offering strong reliability is mentioned in the Swiss Exchange Trading System (SWX) [1]. SWX developers observed some shortcomings that they attribute to the multicast protocols (and strong reliability guarantees) provided by Isis. For instance, one slow client could affect the entire system, especially under peak load. Also, multicast throughput was found to degrade linearly as the number of clients increased.

Throughput instability problem does not only apply to the traditional protocols using virtually synchronous reliability model. Scalable protocols based on best-effort reliability exhibit the same problem. As an example, recent studies [20,21] have shown that, for the SRM protocol, random packet loss can trigger high rates of request and retransmission messages. In addition, this overhead grows with the size of the system.

4 Experimental Model

A theoretical analysis of Pbcast is given in [2]. A simulation model and analysis of the protocol, and also its comparison with a best-effort protocol are discussed in [22]. In this paper, we describe our experimental study for the protocol and Ensemble's virtually synchronous protocols. The main focus is to investigate and analyze protocol properties, giving attention to stability and scalability, in practice.

The experimental work has been performed on the SP2 system of Cornell Theory Center that offers an isolated network behavior. SP2 consists of nodes connected by an ethernet and a switch. A node is a processor with associated memory and disk. Cornell Theory Center's SP2 system has total 160 nodes that share data via message passing over a high performance two-level cross bar switch.

In this work, as shown in fig. 1, Ensemble group communication system has been ported on SP2, and many process group applications utilizing Pbcast and Ensemble's traditional reliable multicast protocols have been designed. Emulation methods have been used to model process and link failures. A detailed experimental study and analysis of Pbcast, and also its comparison with Ensemble's reliable multicast protocols has been accomplished.

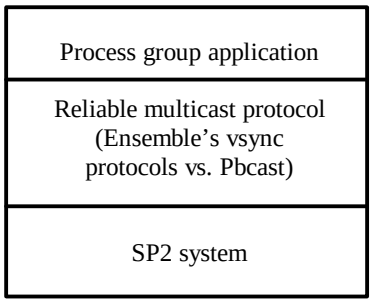

Fig. 1. Block diagram of the model

Our interest is in the performance of the protocols in the case of soft process failures. We emulate a process failure, such as a slow or overloaded member, by forcing the process to sleep with varied probabilities. We call a group member subject to such a failure as '*perturbed*', and the probability of failure that impacts the process as '*perturb rate*'. We have constructed process group applications on Ensemble toolkit for various group sizes starting from 8-member case up to 128-member process groups. There exists one sender process that disseminates 200 multicast messages per second to the group participants. During the execution of group application, some members were perturbed, that is forced to sleep during 100 millisecond intervals with varied perturb rates. First, we designed experiments so that one member is perturbed for various group sizes. Then, we increased the percentage of perturbed members up to 25% of the group size. In other words, we arranged the application so that, one or more group members would occasionally pause, allowing incoming buffers to fill and eventually overflow, but then resume computing and communication before the background failure detection used by the system have detected the situation. This behavior is common in the real world, where multicast applications often share platforms with other applications.

5 Analysis and Results

Based on the results of process group executions described above, we first examine the throughput behavior of Ensemble's virtually synchronous protocols. We then focus on the throughput behavior of Pbcast and also protocol overhead associated with soft failure recovery. During our experiments, we varied a number of operating parameters. These are; n: size of process group (8 to 128), f: number of perturbed processes (1 to n/4), p: degree of perturbation (0.1 to 0.9).

5.1 Throughput Behavior of Virtually Synchronous Protocols

In this part, process group applications utilize Ensemble's traditional and scalable multicast protocols based on virtual synchrony reliability model. We investigate and analyze the throughput behavior of these protocols. We varied operating parameters n, f and p. We measure throughput at the unperturbed or correct group members. The data points in the analysis correspond to values measured during 500 millisecond intervals. Fig. 2 shows some analysis results for 32 and 96-member process groups. Graphs show the superimposed data for cases f=1 and f=n/4. We see that even a single perturbed group member impacts the throughput of unperturbed members negatively. The problem becomes worse as the group size (n), percentage of perturbed members (f), and perturb rate (p) grow. If we focus on the data points for a single perturb rate, we see that the number of perturbed members affects the throughput degradation. For instance, in fig. 2, for a 96-member group when the perturb rate is 0.1, the throughput on non-perturbed members for the scalable Ensemble multicast protocol is about 90 messages/second when there is one perturbed member in the group. The throughput for the same protocol decreases to about 50 messages/second when the number of perturbed members is increased to 24. The same observation is valid for the traditional Ensemble multicast protocol. Among the two protocols, the traditional Ensemble multicast protocol shows the worst throughput behavior.

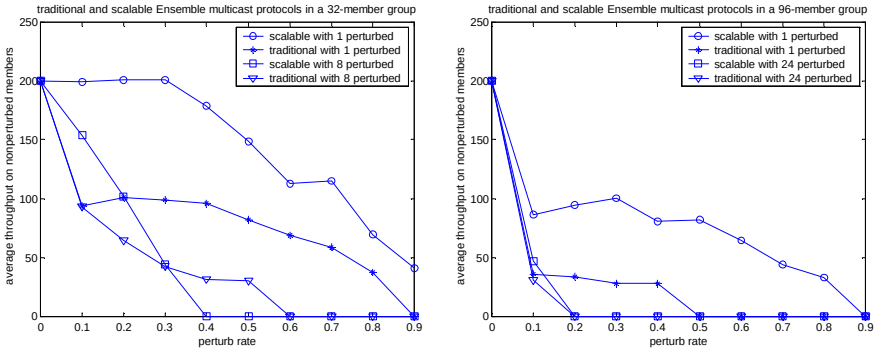

Fig. 2. Throughput performance of Ensemble's reliable multicast protocols

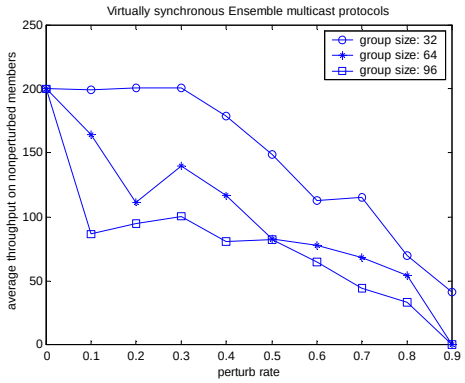

Fig. 3 shows the impact of an increase of group size on the throughput behavior clearly, when f=1. In the next section, we show that, under the same conditions, Pbcast achieves the ideal output rate even with high percentage of perturbed members.

Fig. 3. Throughput behavior as a function of group size

5.2 Throughput Behavior of Pbcast Protocol

In this part, process group applications utilize Pbcast protocol. We investigate the scalability and stability properties of Pbcast. We mainly focus on the following analysis cases:

a) Throughput as a function of perturb rate for various group sizes

b) Throughput as a function of proportion of perturbed members

c) Protocol overhead associated with soft failure recovery as a function of group size.

We varied operating parameters n, f and p. We measure throughput at the unperturbed or correct group members. The data points in the analysis correspond to values measured during 500 millisecond intervals. Since the throughput was steady, we also computed the variance of these data points. Fig. 4 shows variation of throughput measured at a typical receiver as the perturb rate and group size increase. The group size is 8 and 128 respectively. These sample results are for the experiments where f=n/4. We can conclude that as we scale a process group, throughput can be maintained even if we perturb some group members. The throughput behavior remains stable as we scale the process group size even with high rates of failures. During these runs no message loss at all was observed at unperturbed members. On the other hand, the variance does grow as a function of group size. Fig. 5 shows throughput variance as group size increases. Although the scale of our experiments was insufficient to test the log-growth predictions of computational results for Pbcast [2], the data is consistent with those predictions. As we saw in the previous section, the same conditions provoke degraded throughput for traditional virtually synchronous protocols.

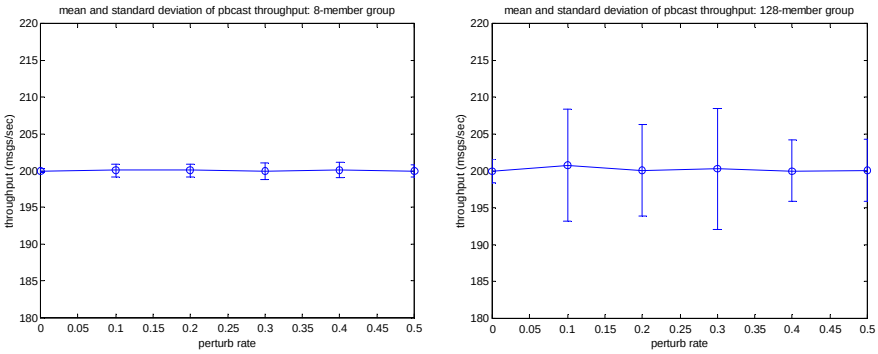

Fig. 4. Variation of pbcast throughput

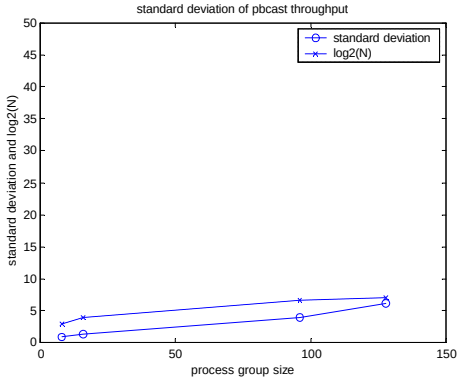

Fig. 5. Throughput variance of Pbcast as a function of group size

We can conclude that Pbcast is more stable and scalable compared to the traditional multicast protocols. As the perturbed process begins to sleep for long enough to significantly impact Ensemble's flow control and windowed acknowledgement, the fragility of the traditional multicast protocols becomes evident very quickly,. Furthermore, in such a condition, high data dissemination rates can quickly fill up message buffers of receivers, and hence can cause message losses due to buffer overflows.

In the case of virtually synchronous protocols, a perturbed process is particularly difficult to manage. Since the process is sending and receiving messages, it is not considered to have failed. But, it is slow and may experience high message loss rates, especially in the case of buffer overflows. The sender and correct receivers keep copies of unacknowledged messages until all members deliver them. It causes available buffer spaces to fill up quickly, and activates background flow control mechanisms. Setting failure detection parameters more aggressively has been proposed as a solution [1]. But, doing so increases the risk of erroneous failure detection approximately as the square of the group size in the worst-case. Because, all group members monitor one another and every member can mistakenly classify all the other (n-1) members as faulty where n is the group size. Then, the whole group has n*(n-1) chances to make a mistake during failure detection. Since the failure detection parameters are set aggressively in such an approach, it is more likely that randomized events such as paging and scheduling delays will be interpreted as a member's crash. As group size increases, failure

detection accuracy becomes a significant problem. Most success scenarios with virtual synchrony use fairly small groups, sometimes structured hierarchically. In addition, the largest systems have performance demands that are typically limited to short bursts of multicast.

In this study, we analyzed protocol overhead associated with soft failure recovery, as well. For this purpose, retransmission behavior at a correct member was investigated. Fig. 6 shows overhead as perturb rate increases, for 8 and 128-member groups, respectively. For these graphs f=n/4, and each region in the graphs illustrates data points measured during 500 msec intervals for a certain perturb rate. For instance, the first region contains data points for p=0.1, second one is for p=0.2, and so on. Fig. 7.a superimposes the data for n= 8, 16, 64 and 128, and shows the percentage of messages retransmitted as p increases for various n.

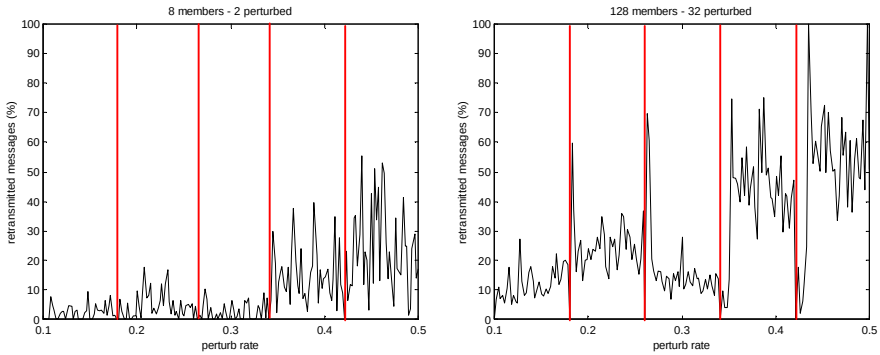

Fig. 6. Pbcast overhead associated with soft failure recovery

For these experiments, we also compute the theoretical worst-case bounds for retransmission behavior at a correct member (fig. 7.b). Assume r is the number of multicast data messages per second disseminated to the group by the sender, and p is the perturb rate. In every 100 msec (which is the duration of a gossip round in the experiments), at most ((r/10)*p) messages are missed by a faulty member, and a correct member gossips to two randomly selected group members. In the worst-case, if these two members are faulty and they lack all ((r/10)*p) data messages, they request retransmissions of these messages from the correct member. Then, the correct member retransmits at most 2*((r/10)*p) = (r*p)/5 messages in every 100 msec. In our experiments, we measured data points during 500 msec intervals, and computed the percentage of retransmitted messages to the multicast data messages disseminated by the sender during each interval. If we compute theoretical values for 500 msec intervals, the correct member retransmits at most 5*(r*p)/5 = r*p messages, and the sender disseminates r/2 messages during every 500 msec interval. Then, the bound for the percentage of retransmitted messages would be (r*p)/(r/2) = 2*p in the worst-case. Fig. 7.b shows the computed theoretical worst-case bounds. Note that, our experimental results are below the theoretical bound, and the results confirm that overhead on the correct processes is bounded as the size of process group increases.

As the group size increases, we observed an increase in the percentage of retransmitted messages. We believe, this is mainly due to the increase in the number of perturbed members with the group size. Because, in these experiments, number of perturbed members equals 25% of the group size (f = n/4).

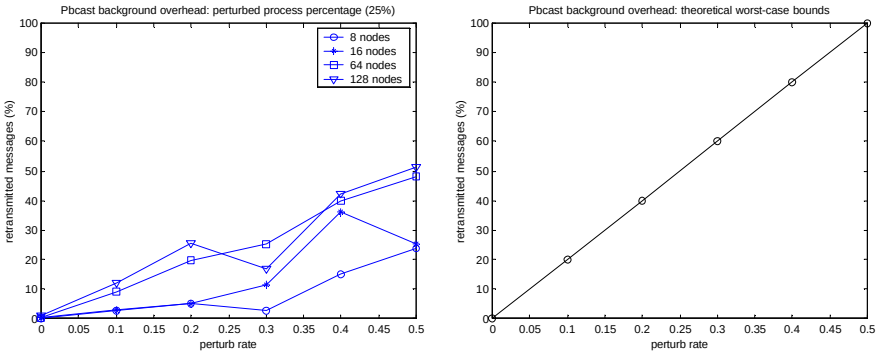

Fig. 7. Percentage of message retransmissions as a function of p. a) Experimental results, b) Theoretical worst-case bounds

6 Conclusion

Our study yields some general conclusion about the behavior of basic Pbcast and virtually synchronous multicast protocols. In the first part of the study, we focused on the virtually synchronous Ensemble multicast protocols in the case of soft process failures. We showed that even a single perturbed group member impacts the throughput of unperturbed members negatively. On the other hand, Pbcast achieves the ideal throughput rate even with high percentage of perturbed members. In the second part of the study, we focused on the performance of Pbcast in the case of soft process failures. We showed that the throughput behavior of Pbcast remains stable as we scale the process group size even with high rates of failures. Furthermore, our results confirm that overhead on the correct processes is bounded as the size of process group increases.

References

1. Piantoni, R. and Stancescu, C., 1997, Implementing the Swiss Exchange Trading System, FTCS 27, Seattle, WA, 309-313p.
2. Birman, K.P., Hayden, M., Ozkasap, O., Xiao, Z., Budiu, M. and Minsky, Y., 1999, Bimodal Multicast, ACM Transactions on Computer Systems, 17(2), 41-88p.

3. Birman, K.P. and van Renesse, R., 1994, Reliable Distributed Computing with the Isis Toolkit, New York: IEEE Computer Society Press.
4. Birman, K.P., 1993, The Process Group Approach to Reliable Distributed Computing, Communications of the ACM, 36(12), 37-53p.
5. Van Renesse, R. and Birman, K.P., 1995, Protocol Composition in Horus, Technical Report, TR95-1505, Department of Computer Science, Cornell University.
6. Van Renesse, R., Birman, K.P. and Maffeis, S., 1996, Horus: A Flexible Group Communication System, Communications of the ACM, 39(4), 76-83p.
7. Moser, L.E., Melliar-Smith, P.M., Agarwal, D.A., Budhia, R.K., et.al, 1996, Totem: A Fault-tolerant Multicast Group Communication System, Communications of the ACM, 39(4), 54-63p.
8. Dolev, D. and Malki, D., 1996, The Transis Approach to High Availability Cluster Communication, Communications of the ACM, 39(4), 64-70p.
9. Hayden, M., 1998, The Ensemble System, Ph.D. dissertation, Cornell University Dept. of Computer Science.
10. Lidl, K., Osborne, J. and Malcome, J., 1994, Drinking from the Firehose: Multicast USENET News, USENIX Winter 1994, 33-45p.
11. Floyd, S., Jacobson, V., Liu, C., McCanne, S. and Zhang, L., 1997, A Reliable Multicast Framework for Light-weight Sessions and Application Level Framing, IEEE/ACM Transactions on Networking, 5(6), 784-803p. http://www-nrg.ee.lbl.gov/floyd/srm.html
12. Speakman, T., Farinacci, D., Lin, S. and Tweedly, A., 1998, PGM Reliable Transport Protocol, Internet-Draft.
13. Paul, S., Sabnani, K., Lin, J. C. and Bhattacharyya, S., 1997, Reliable Multicast Transport Protocol (RMTP), IEEE Journal on Selected Areas in Communications, special issue on Network Support for Multipoint Communication, 15(3), http://www.bell-labs.com/user/sanjoy/rmtp2.ps
14. Lin, J.C. and Paul, S., 1996, A Reliable Multicast Transport Protocol, Proceedings of IEEE INFOCOM '96, 1414-1424p. http://www.bell-labs.com/user/sanjoy/rmtp.ps
15. Cristian, F., Aghili, H., Strong, R. and Dolev, D., 1985, Atomic Broadcast: From Simple Message Diffusion to Byzantine Agreement. Proc. 15th International FTCS, 200-206p.
16. Lamport, L., 1978, The Implementation of Reliable Distributed Multiprocess Systems, Computer Networks, 2, 95-114p.
17. Birman, K.P. and Joseph, T.A., 1987, Exploiting Virtual Synchrony in Distributed Systems, Proceedings of the 11th Symposium on Operating System Principles, New York: ACM Press, 123-128p.
18. Birman, K.P., 1997, Building Secure and Reliable Network Applications, Manning Publishing Company and Prentice Hall, Greenwich, CT. http://www.browsebooks.com/Birman/index.html
19. Hayden, M. and Birman, K.P., 1996, Probabilistic Broadcast, Technical Report, TR96-1606, Department of Computer Science, Cornell University.
20. Liu, C., 1997, Error Recovery in Scalable Reliable Multicast, Ph.D. dissertation, University of Southern California.
21. Lucas, M., 1998, Efficient Data Distribution in Large-Scale Multicast Networks, Ph.D. dissertation, Dept. of Computer Science, University of Virginia.
22. Ozkasap, O., Xiao, Z. and Birman, K.P., 1999.a, Scalability of Two Reliable Multicast Protocols, Technical Report, TR99-1748, Department of Computer Science, Cornell University.