Evaluation of Gossip to Build Scalable and Reliable Multicast Protocols

Soontaree Tanaraksiritavorn and Shivakant Mishra
Department of Computer Science
University of Colorado, Campus Box 0430
Boulder, CO 80309-0430.
Email: {tanaraks|mishras}@cs.colorado.edu

Abstract

This paper provides an extensive experimental evaluation of the gossip technique that has been proposed to construct scalable and reliable multicast protocols. This evaluation has been done by simulating several gossip and non gossip-based multicast protocols in a network simulator NS2. The evaluation comprises of three parts: (1) a comparison of the gossip technique with other traditional message loss detection and recovery techniques, (2) an extensive performance evaluation of the gossip technique under several different computing environments, and (3) an evaluation of how the performance of a gossip-based multicast protocol is affected by varying various gossip parameters. The main conclusion of the paper is that the gossip technique is certainly a useful technique for constructing scalable and reliable multicast protocols. However, it performs poorly under some specific operating conditions, and its performance is affected by the selection criteria used to select a gossip subgroup and gossip period.

1. Introduction

Reliable multicast protocols have been used to construct critical as well as non-critical applications. These protocols ensure that all group members receive a multicast message despite communication, processor, or other component failures. A fundamental research issue in designing these protocols at present is the scalability of these protocols in terms of group size. Scalability in terms of group size requires that these protocols continue to work correctly and provide a good and stable performance as the group size increases.

Gossip is a technique that has been proposed to construct scalable and reliable multicast protocols [3, 1, 4]. Informally, this technique comprises of group members exchanging messages with one another at random to detect and recover lost multicast messages. There are three parameters that can affect the performance of a gossip-based multicast protocol: (1) selection of an appropriate gossip subgroup with which a group member exchanges multicast state information, (2) adopting a suitable retransmission strategy: gossip-push vs gossip-pull, and (3) fixing an optimal gossip period: the time interval in which a group member gossips with the members of its gossip subgroup.

The main goal of this paper is to evaluate the effectiveness of gossip technique in building scalable and reliable multicast protocols. We have implemented two reliable multicast protocols: one gossip-based (Gossip) and one negative acknowledgement-based (NAK). We have also used the Scalable Reliable Multicast (SRM) protocol developed at UC-Berkeley/LBNL in our evaluation. We measured the performance of these three protocols under several different operating conditions by simulating them in a network simulator, NS2. The operating parameters that we varied in our experiments are (1) group size, (2) message arrival rate, (3) message arrival patterns, (4) communication failure probability, (5) gossip subgroup, and (6) gossip period. The performance indices we measured are protocol throughput, protocol reliability, and protocol message overhead. In this paper, we present the performance measured and our evaluation based on this performance.

2. Protocol Description

2.1. NAK

In this protocol, messages are disseminated using IP multicast. A receiver checks for message loss as soon a new message arrives by comparing the sequence number of the new message with the sequence number of the last message received. If a gap in sequence numbers is detected, the receiver sends a *retransmission request* for each missing message to the sender of the message. Retransmission request and *repair packets* are sent using unicast. In addition, a member sends a *session message* containing sender id and the sequence number of the last message received during long idle periods to detect messages losses.

2.2. Gossip

In this protocol, a group member multicasts a message by disseminating it using the IP multicast. Gossip differs from NAK in the mechanism used for recovering from the loss of messages. Any member in the Gossip protocol can act as a source of message loss detection and recovery mechanism. Unlike NAK, group members do not attempt to detect message losses on receiving a new message. Instead, each group member has a gossip subgroup, which is a (small) subgroup of the multicast group. Group members periodically send gossip messages to all members with in their gossip subgroup. A gossip message sent by a group member contains the current state of that member. Using this message, a group member informs the members of its gossip subgroup about all messages it has received. When a gossip message is received by a group member, the group member checks information in the message to detect if it is missing some messages. A retransmission request for missing messages is sent directly to the gossip message sender. This technique is also called gossip-push method. An alternate technique is gossip-pull method, wherein a gossip sender informs all members of its gossip subgroup about all messages it is missing. Members of a gossip subgroup of a group member can be chosen at random, or from the neighborhood of the group member. We have experimented with both of these methods of selecting a gossip subgroup.

2.3. SRM

SRM [4] provides many-to-many multicast delivery. Like NAK and Gossip, messages are multicast using IP multicast. Message loss is detected when a gap in sequence numbers between the recently received message and the last received message is detected. Like NAK, once a receiver detects a message loss, it multicasts a retransmission request. Any member that receives the retransmission request can act as a source for recovering lost messages by multicasting the requested message. To reduce multicast traffic load and redundant retransmissions, each member waits for a random period of time before responding to a retransmission request. A group member does not respond to a retransmission request, if it has already seen a response sent by some other member. Like NAK, group members also send session messages periodically to detect message losses that may occur just before a long idle period.

3. Implementation

We implemented Gossip and NAK in C++ in NS2 that was developed at UC Berkeley. We already had an implementation of SRM in NS-2 available. The network topology was generated by Georgia Tech's Internetwork Topology Models (GT-ITM). Each link in the network is a 10Mb/sec with varied communication delay, automatically generated by GT-ITM. We set the router queue limit to 50 packets. The network topology constructed by GT-ITM is called the transit-stub model [2]. The graph generated by this model reflects the hierarchical structure and locality present in the Internet. Each routing domain in the internetwork can be classified as either a stub or a transit domain. Stub domains correspond to interconnected LANs while transit domains represent WANs or MANs. All topologies generated in our experiments had one transit domain with varied sizes of interconnected LANs. A transit domain consists of N_t backbone nodes on an average. Each of these backbone nodes consist of S stub domains on an average attached to it, with approximately N_s nodes per stub domain. A node in stub domains represents a router or switch. The total number of nodes is $N_t * (1 + S * N_s)$.

3.1. Operating Parameters

We have varied six operating parameters to create different operating conditions. These six parameters are *group size*, *message arrival pattern*, *message arrival rate*, *failure probability*, and two gossip parameters, *gossip subgroup* and *gossip period*. Group size was varied from 20 to 124. Message arrival pattern indicates the pattern in which one or more group members multicast messages over a relatively long period of time. We have considered three types of message arrival patterns: *uniform*, *one active*, and *bursty*. In the uniform message arrival pattern, a large percentage of group members multicast messages at a constant rate. In our experiments, 50% of the group members multicast messages under uniform message arrival pattern. In the one-active message arrival pattern, one group member multicasts all messages. Finally, in the bursty message arrival pattern, group members multicast messages alternating between short periods (bursty periods) at a very high rate, and relatively long periods (idle periods) at very low rate.

Message arrival rate is the rate at which messages are multicast in the group. We varied the message arrival rate from 3000 to 10,000 messages per second. Failure probability is the probability of message losses in the communication network. We measured performance under 0.1% and 1% failure probabilities. Finally, we varied two gossip parameters: *gossip subgroup size* and *gossip period*. We experimented with gossip subgroup sizes 2, 10% of the total group size, and 50% of the total group size, and gossip periods of 0.25 seconds and 2 seconds.

3.2. Performance Indices

We consider the following three important performance indices: *throughput*, *reliability*, and *message overhead*.

Throughput is the number of messages received by a group member per second for a given arrival rate and arrival pattern. Reliability is the percentage of multicast messages received by a group member for a given arrival rate and arrival pattern. Finally, message overhead is the percentage of overhead messages (session, request, and repair messages) exchanged in the multicast.

4. Simulation Results

Each simulation experiment was performed over a 14.0 second period. Nodes joined the group 0.5 seconds after the simulation run was started, and the membership remained constant. Members started multicasting messages 0.5 seconds after joining the group for a period of 10 seconds.

We started by measuring the throughput and reliability of the three protocols under different operating conditions. However, soon it became evident that SRM provides very poor performance for even moderate group sizes (around 50), or moderate arrival rates (around 2000 messages per second). This was the case for all three message arrival patterns. For example, Figures 1 and 2 show the throughput of SRM and Gossip for uniform and bursty arrival patterns respectively. As we can see, the throughput of SRM decreases significantly faster with increase in group size. We note that several earlier works have also evaluated SRM and arrived at the same conclusion (e.g see [1]). These earlier evaluations of SRM were done for only one-active arrival pattern. Our experiments have confirmed that SRM is not scalable at even moderate group sizes and arrival rates under uniform or bursty arrival patterns as well.

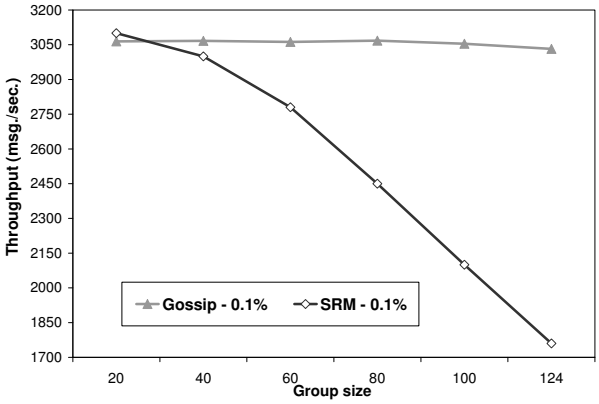

Figure 1. Throughput (uniform arrival pattern).

The main reason for poor performance of SRM at larger group sizes or higher arrival rates is that it consumes significantly large bandwidth. As the group size scales up, or the arrival rates become larger, the number of overhead messages increases significantly. Because of the significantly poor performance of SRM, we will not included its performance in our performance comparison henceforth. A detailed performance results of SRM from our experiments can be found in [5].

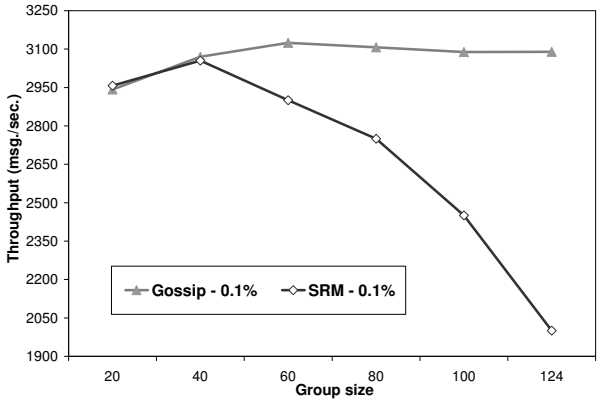

Figure 2. Throughput (bursty arrival pattern).

4.1. Throughput

Figure 3 shows the throughput for Gossip and NAK under three arrival patterns at a high arrival rate (7500 messages/second), and low failure probability (0.1%). There are three observations we make from this figure. First, for both Gossip and NAK, best throughput occurs under uniform arrival pattern, and worst throughput occurs under one-active arrival pattern. This shows that both protocols are sensitive to how fast a single group member is multicasting data. In case of one-active arrival pattern, one group member multicasts at a very high rate, while in case of uniform arrival pattern, rate at which a single group member multicasts is slowest. The main reason for this sensitivity to the multicast rate of a single group member is the increased congestion in the network near that member. The increased congestion results in delayed message delivery, message losses, and subsequent transmission of control messages.

The second observation is that the throughput of Gossip is significantly better than that of NAK under all three arrival patterns. This shows that under high arrival rate, gossip is a useful technique even for relatively smaller group sizes. Finally, as group size increases, the throughput of both Gossip and NAK decreases. However, the throughput of NAK decreases much faster than that of Gossip. This shows that Gossip scales much better with group size than NAK under high arrival rate and low failure probability.

Figures 4, 5, and 6 show the effect of failure rate on the throughput of Gossip and NAK at low arrival rate. We make several observations from these measurements. Gossip's throughput is higher than NAK's in one active and bursty arrival patterns. It is significantly higher at high failure rate. However, Gossip's throughput is lower than NAK's in uniform arrival pattern. The reason for this observation is once

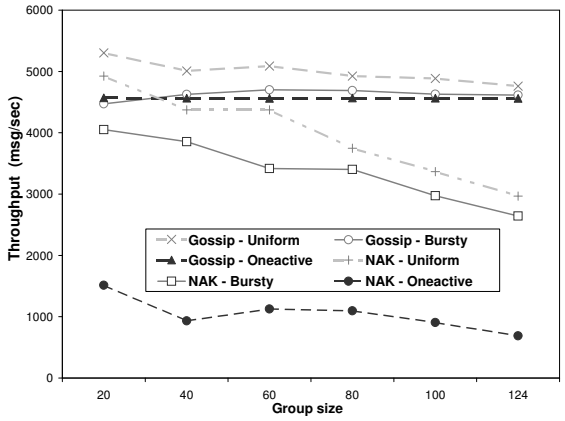

Figure 3. Throughput as a function of group size.

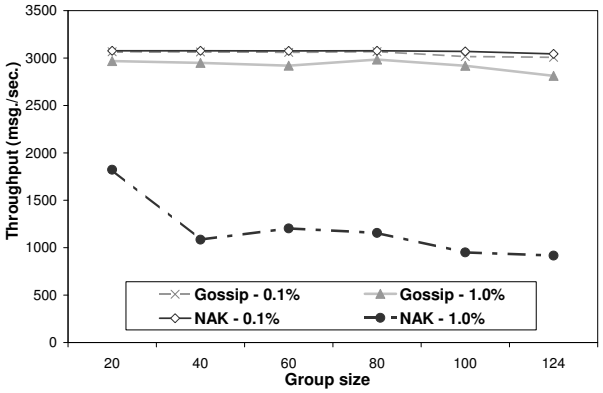

Figure 4. Throughput (one-active arrival pattern).

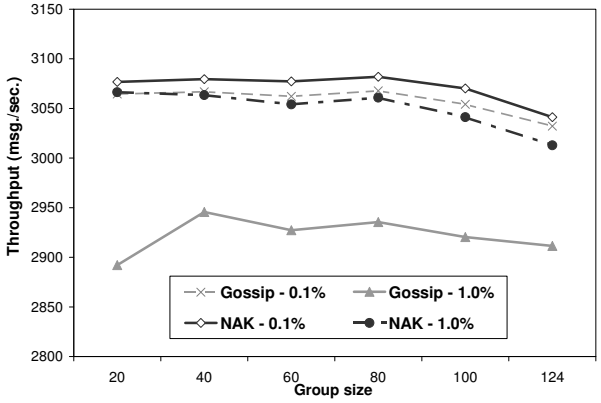

Figure 5. Throughput (uniform arrival pattern).

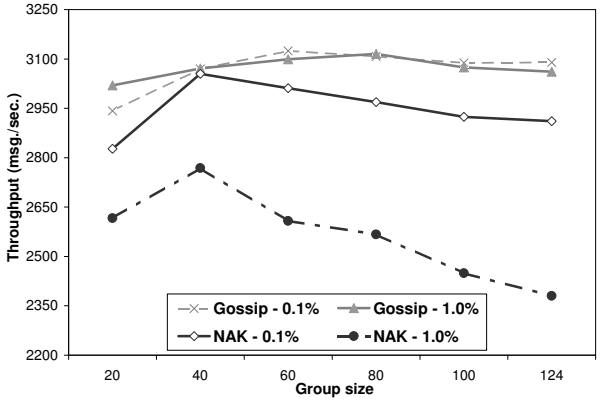

Figure 6. Throughput (bursty arrival pattern).

again attributed to the relative sensitivity of these protocols to rate of multicast of a single member. Gossip performs better than NAK when the rate at which a single member is multicasting data is high (i.e. one active and bursty arrival patterns). However, NAK performs better when this rate is low (i.e. uniform arrival pattern). The second observation is that this sensitivity is also related to failure rate. The difference in throughput between NAK and Gossip is higher under high failure rate than under low failure rate. In fact, the throughput generally drops more significantly in case of NAK than in case Gossip as failure probability increases (under one active and bursty arrival patterns). This shows that Gossip provides better scalability with increase in failure probability. In other words, Gossip handles message losses more efficiently than NAK.

Figures 7, 8, and 9 show the throughput of NAK and Gossip as a function of message arrival rate for one-active, uniform, and bursty arrival patterns respectively. There are three observations we make from these figures. First, the throughput of both the protocols increases with increase in arrival rate until a peak is reached. After this peak, the throughput drops significantly with increase in arrival rate. The reason for this behavior is that initially at low arrival rates, the additional messages multicast due to increase in arrival rates do not congest the network. As a result, as the arrival rate increases, so does the throughput. However, after a particular point, any additional messages due to increasing arrival rate start congesting the network. This results in increased message losses and lower throughput.

The second observation we make is that the peak after which the throughput starts decreasing with increase in arrival rate occurs at a much higher arrival rate in Gossip than in NAK. This shows that the network gets congested in NAK at much lower arrival rate than in Gossip. The main reason for this is that the number of control messages Gossip uses to implement reliability gets significantly lower than the number of control messages NAK uses as the arrival rate increases. This will become evident from the message overhead measurements described later in this section. Finally, as the arrival rate increases, the difference in throughput between Gossip and NAK also increases. In other words, throughput of NAK gets much worse, compared to the throughput of Gossip, as the arrival rate increases. This shows that Gossip copes much better with higher arrival rates than NAK. All these observations show that Gossip can maintain a better, stable throughput for much higher arrival rates than NAK. Hence it scales much

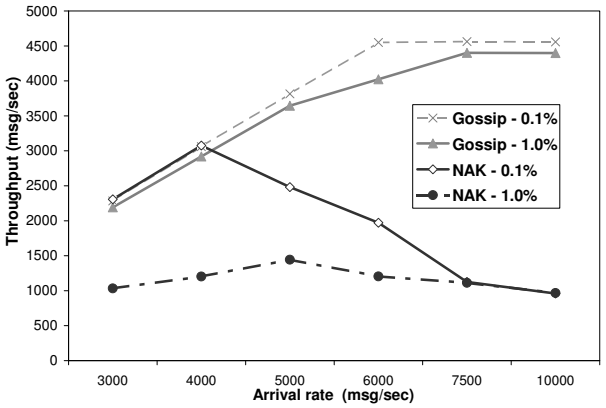

Figure 7. Throughput (one-active arrival pattern).

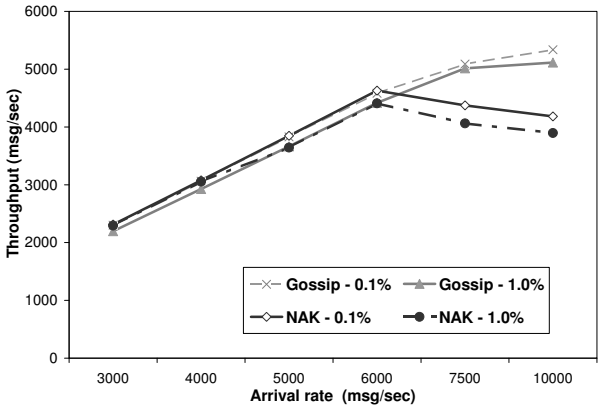

Figure 8. Throughput (uniform arrival pattern).

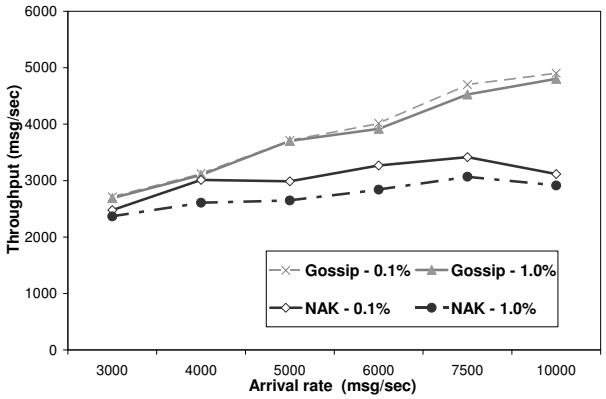

Figure 9. Throughput (bursty arrival pattern).

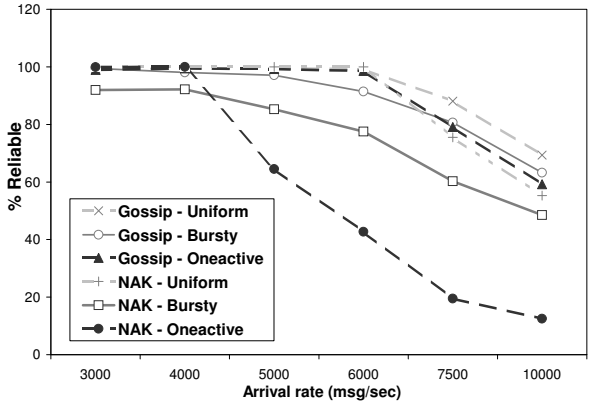

Figure 10. Reliability as a function of arrival rate.

better than NAK with increase in arrival rate.

4.2. Reliability

Figure 10 shows the reliability of Gossip and NAK as a function of message arrival rate. The failure probability is low (0.1%), and the group size is 60. There are three observations we make from this figure. First, both protocols exhibit high reliability (close to 100%) at low arrival rates. This indicates that both protocols are able to recover from all message losses when the arrival rate is low. In general, NAK provides a (very) slightly higher (except in bursty arrival pattern) reliability than Gossip under low arrival rate. Hence, as far as reliability is concerned, there is little to choose between the two protocols.

The second observation is that the reliability starts decreasing significantly with increase in arrival rate after a certain value of arrival rate in both protocols. The reason for this is increase in network congestion. For lower arrival rate, both protocols are able to recover from almost all message losses. However, as network starts getting congested with increase in arrival rate, both protocols start losing more messages. An important observation here is that the rate of decrease in reliability is significantly higher in NAK than in Gossip. The main reason for this is the technique used to recover from message losses. A message loss results in more control messages being transmitted per unit time than Gossip. As a result, a message loss further congests the network in NAK, and hence, worsens the protocol reliability.

Finally, at higher arrival rates, reliability exhibited by gossip is significantly higher than that of NAK. This shows that Gossip is increasingly able to provide a higher reliability with increase in arrival rate that NAK. All these observations show that with respect to reliability, Gossip scales much better than NAK with increase in arrival rate.

Figures 11, 12, and 13 show the reliability of NAK and Gossip as a function of message arrival rate for one-active, uniform, and bursty arrival patterns respectively at both low and high failure rates. There are three important observations we can make from these figures. First, as with low failure probability, the reliability of both the protocols remains constant and high at low arrival rates for high failure probability as well. Also, again as with ow failure probability, after a certain value of arrival rate, the reliability

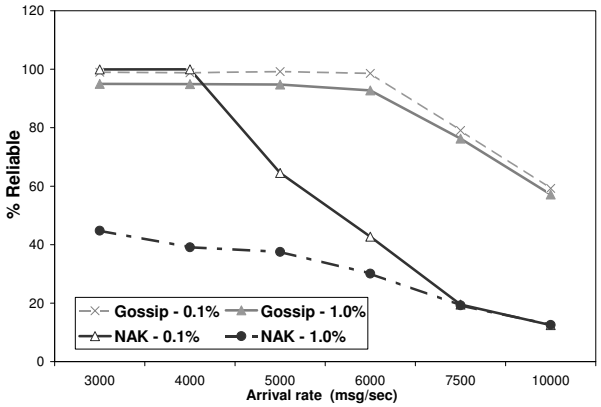

Figure 11. Reliability (one-active arrival pattern).

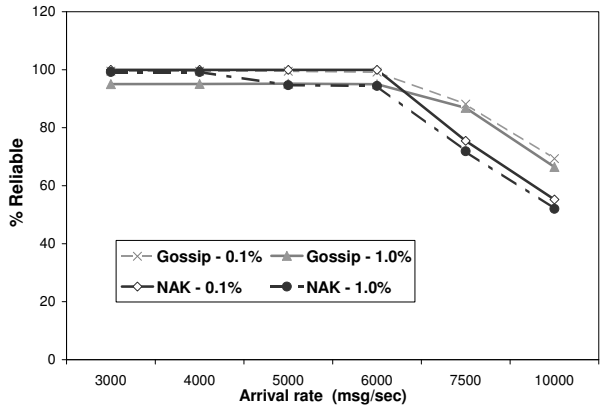

Figure 12. Reliability (uniform arrival pattern).

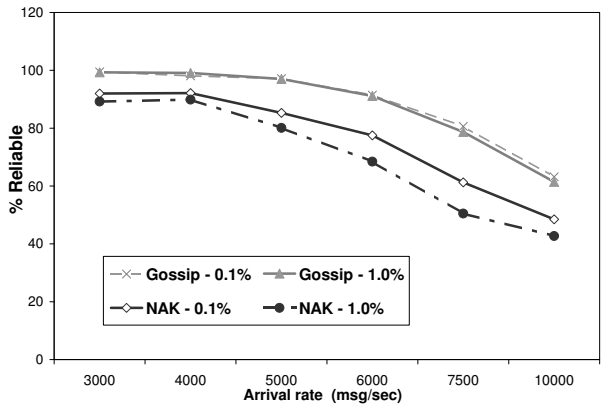

Figure 13. Reliability (bursty arrival pattern).

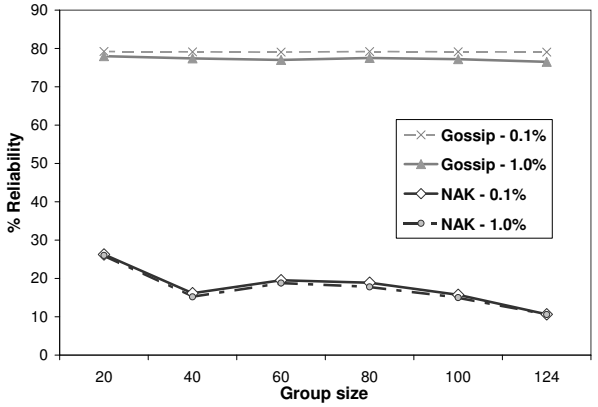

Figure 14. Reliability (one-active arrival pattern).

drops significantly with increase in arrival rate for high failure probability as well. The reason for this is same as the one mentioned above for low failure probability.

The second observation is that the reliability of NAK drops significantly with increase in failure probability. On the other hand, reliability of Gossip is not affected significantly by increase in failure probability. This shows that the technique used to recover from message losses in NAK is not well-suited for high failure computing environments. The main reason for this is that the number of control messages sent as result of a message loss in NAK is much higher than in Gossip. This results in increased congestion and in turn increased message losses in NAK.

Finally, the rate at which reliability starts decreasing after a certain arrival rate is much higher in NAK than in Gossip, both at low and high failure probability. All these observations show that Gossip can maintain a better, stable reliability for much higher arrival rates than NAK.

Figures 14, 15, and 16 show the reliability of NAK and Gossip as a function of group size for one-active, uniform, and bursty arrival patterns respectively for a high arrival rate. There are three observations we make from these figures. First, reliability of Gossip is better than that of NAK under all three arrival patterns. Even at low group size of 20, Gossip provides better reliability than NAK. Both NAK and Gossip provide best reliability under uniform arrival pattern, and worst under one-active arrival pattern. Once again this is attributed to these protocols' sensitivity to the rate at which a single group member is multicasting data. The second observation is that the reliability of NAK varies significantly with change in arrival patterns. It is very low in one-active arrival pattern, and only moderate in uniform and bursty arrival patterns. This shows that NAK is more sensitive to the to the rate at which a single group member is multicasting data than Gossip. Finally, the reliability decreases with increase in group size in both protocols. However the rate of this decrease is much higher in NAK than in Gossip. This shows that Gossip can maintain a better, stable reliability with increase in group size.

4.3. Message Overhead

Figure 17 shows the message overhead in Gossip and NAK as a function of group size. The arrival pattern is uni-

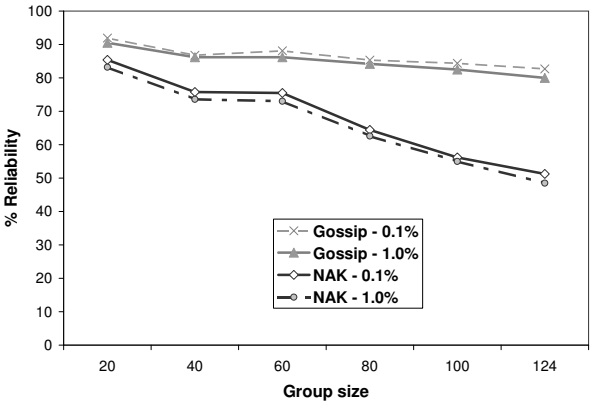

Figure 15. Reliability (uniform arrival pattern).

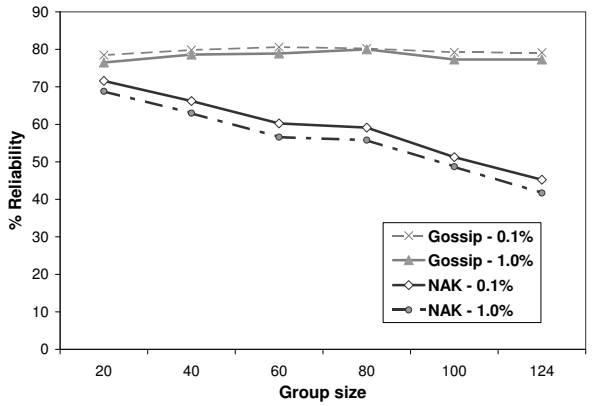

Figure 16. Reliability (bursty arrival pattern).

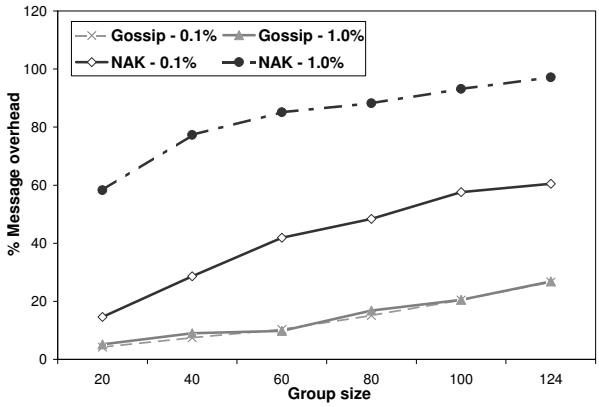

Figure 17. Message overhead as a function of group size.

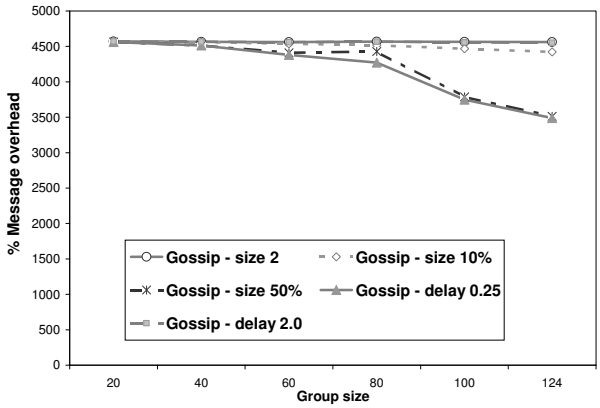

Figure 18. Throughput for one-active arrival pattern.

form, and the arrival rate is 4000 messages/second. There are three observations we make from this figure. First, the message overhead is much higher in NAK than in Gossip. Even at smaller group sizes (e.g. 20), message overhead in NAK is more than twice the message overhead in Gossip. The main reason for this is that Gossip avoids sending control messages when multicast data arrives out of order. On the other hand, a member in NAK sends out a retransmission request as soon as it sees a gap in the sequence numbers. If messages arrive out of order (more probable in large arrival rates and group sizes), these retransmission requests and the corresponding repair packets are unnecessary.

The second observation is that the message overhead increases significantly in NAK with increase in failure probability. The message overhead in Gossip is not affected in any significant manner with increase in failure probability. The main reason for this is that a group member in NAK sends retransmission requests and corresponding repair packets every time it detects a gap in the received message sequence numbers. Gossip on the other hand, sends control messages only periodically. In particular, multiple message losses may result in the transmission of several control messages in NAK, and only a single control message in Gossip. The final observation is that the message overhead increases with increase in group size in both the protocols. However, the rate of increase is much smaller in Gossip than in NAK. All these three observations show that Gossip maintains a relatively slow increase in the percentage of overhead message sent with increase in group size or failure probability. On the other hand, the percentage of overhead messages sent in NAK increases significantly faster with increase in group size or failure probability. Message overhead measured for other arrival patterns and arrival rates showed similar characteristics.

4.4. Gossip Parameters

Figures 18, 19, and 20 show the effect of variation in gossip subgroup size and gossip period on the throughput in Gossip for one-active uniform, and bursty arrival patterns respectively. The arrival rate is 7500 messages/second and failure probability is 0.1%. With increase in group size, the throughput decreases significantly when the gossip sub-

group size is 50%. On the other hand, throughput almost remains the same when gossip subgroup sizes are 2 or 10%. This shows that the throughput of Gossip is affected by the gossip subgroup size only when the subgroup size is very large. Variation in the throughput with gossip period is more pronounced though. Under both one-active and uniform arrival patterns, the throughput decreases significantly by changing the gossip period from 2.0 ms to 0.25 ms.

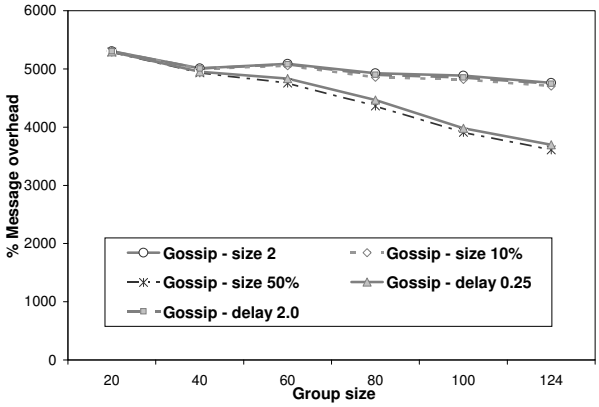

Figure 19. Throughput for uniform arrival pattern.

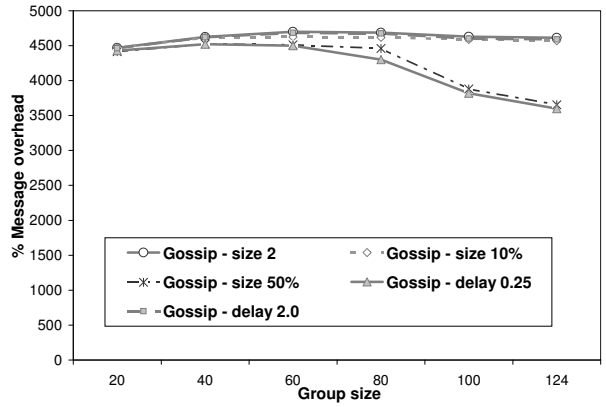

Figure 20. Throughput for bursty arrival pattern.

5. Discussion

Our first conclusion is that the gossip technique is a very useful technique for constructing scalable and reliable multicast protocols. It can sustain good, stable throughput and reliability for much larger values of group size or message arrival rates than NAK or SRM. NAK outperformed Gossip for some operating scenarios that involved smaller group sizes, or lower message arrival rates. However, as the group size or update arrival rate increases, the performance of NAK drops very significantly, while Gossip continues to maintain its good performance. Another important observation is that even when failure probability increases, Gossip is able to maintain a relatively stable throughput and reliability. Performance of NAK on the other hand is significantly reduced by an increase in failure probability.

The main reason for the better performance of Gossip for larger values of group size or update arrival rate is evident when we look at the number of overhead messages. In comparison with Gossip, NAK injects a significantly large number of overhead messages. The rate of increase in the number of overhead messages with group size or message arrival rate is significantly higher in NAK than in Gossip. These overhead messages congest the network as group size or message arrival rate increases. This results in additional message losses, and hence poor performance.

The second conclusion is that the effectiveness of Gossip is significantly dependent on the operating conditions. Performance of Gossip is best under uniform message arrival pattern, and worst under one-active message arrival pattern. Similar behavior was observed for NAK as well. This shows that both of these protocols are sensitive to the rate at which a single member in the group is multicasting. Again, the reason for this becomes evident from looking at the message overhead of the two protocols. We observed that the nodes closer to the active sender in one-active arrival pattern tend to overload very fast, while in uniform arrival pattern load was shared equally over the network. This results in the overloaded nodes near a fast-sending sender to become performance bottleneck.

The final conclusion is that the performance of gossip technique is affected to some extent by choosing appropriate gossip subgroup size and gossip period. The simulation results show that we should avoid configuring Gossip with a very large gossip subgroup size or a low gossip period. The main reason is that under a large gossip subgroup size or a low gossip period, Gossip injects too many overhead messages, and that results in poor performance.

References

[1] K. Birman, M. Hayden, O. Ozkasao, Z. Xial, M. Budiu, and Y. Minsky. Bimodal multicast. *ACM Transactions on Computer Systems*, 17(2):41–88, May 1999.

[2] K. Calvert, M. Doar, and E. Zegura. Modeling internet topology. *IEEE Communications Magazine*, June 1997.

[3] A. Demers, D. Greene, C. Hauser, W. Irish, and J. Larson. Epidemic algorithms for replicated database maintenance. In *Proceedings of the 6th ACM Symposium on Principles of Distributed Computing*, 1987.

[4] S. Floyd, V. Jacobson, C.-G. Liu, S. McCanne, and L. Zhang. A reliable multicast framework for light-weight sessions and application level framing. *IEEE/ACM Transcations on Networking*, 5(6), December 1997.

[5] S. Tanaraksiritavorn. Evaluation of gossip technique to build scalable, reliable multicast protocol. Master's thesis, University of Colorado, Department of Computer Science, Boulder, CO, 2001.