Int. J. Communications, Network and System Sciences, 2010, 3, 72-78
doi:10.4236/ijcns.2010.310010 Published Online January 2010 (http://www.SciRP.org/journal/ijcns/).

Cooperative Loss Recovery for Reliable Multicast in Ad Hoc Networks

Mengjie HUANG, Gang FENG, Yide ZHANG
National Key Laboratory of Science and Technology on Communications
University of Electronic Science and Technology of China, Chengdu, China
E-mail: *fenggang@uestc.edu.cn*

Received September 22, 2009; *revised October* 24, 2009; *accepted December* 2, 2009

Abstract

Providing reliable multicast service is very challenging in Ad Hoc networks. In this paper, we propose an efficient loss recovery scheme for reliable multicast (CoreRM). Our basic idea is to apply the notion of cooperative communications to support local loss recovery in multicast. A receiver node experiencing a packet loss tries to recover the lost packet through progressively cooperating with neighboring nodes, upstream nodes or even source node. In order to reduce recovery latency and retransmission overhead, CoreRM caches not only data packets but also the path which could be used for future possible use to expedite the loss recovery process. Both analytical and simulation results reveal that CoreRM significantly improves the reliable multicast performance in terms of delivery ratio, throughput and recovery latency compared with UDP and PGM.

Keywords: Ad Hoc Networks, Reliable Multicast, Cooperative Communication, Loss Recovery

1. Introduction

Multicast is an efficient approach to deliver copies of data to multiple specific receivers, or the group members in Ad Hoc networks. Although some multicast applications, such as audio/video conferencing, can tolerate certain degree of packet errors or losses, other applications, such as one-to- many file transfer and military applications do not. Hence, reliable multicast is an inevitable service to support those reliability-dependent applications in Ad Hoc networks. Reliable multicast in Ad Hoc networks faces various technical challenges, e.g. high error rate, low bandwidth, and highly dynamic and unpredictable topology changes. Receivers in a multicast group may experience wide different packet loss rates depending on their locations on the multicast tree. Having the sender or a separate node retransmit to the entire group when only a small subset of the receivers experience losses wastes network resource. Another main challenge in reliable multicast is the frequent group membership changes due to node mobility. Such changes make it difficult to designate several nodes as repair nodes in retransmitting lost packets. In addition, mobile nodes in Ad Hoc networks usually have limited capacity for separately responding to reports of data loss. Therefore, developing reliable multicast service calls for a robust and efficient loss recovery scheme to cope with dynamic group membership changes and scale loss recovery to large multicast groups.

In recent year, a number of reliable multicast protocols [1,2] have been proposed. The Pragmatic General Multicast (PGM) [1] is a known reliable multicast protocol for wired network. PGM depends on source node to recover the lost. Each receiver maintains the receiving a record which caches the maximum sequence number of received packet, and request repairs via a NAK when error occurs. In [2] we develop efficient reliable multicast protocols by jointly considering loss recovery and congestion control. However, providing reliable multicast service is more challenging in Ad Hoc networks. Recently, a number of reliable multicast protocols [3–6] for Ad Hoc networks have be proposed. These protocols use different approaches to improve the reliability of the multicast. According to the position of repair node, these protocols can be classified into three categories: 1) source node dependent, 2) receiver node dependent; and 3) nearby node dependent. The Reliable Multicast Algorithm (RMA) [3] depends on source node for recovering loss packets. In RMA all the receivers must send ACKs to the sender for received data packets. The Reliable, Adaptive, Congestion-Controlled Ad Hoc Multicast Transport Protocol (ReAct) [6] recovers loss packets with upstream-node and source node. In [5], The Anonymous Gossip (AG) randomly selects one of nearby members as repair node. However, AG cannot guarantee to recover all of missing packets for the random choose. However, depending upon the source or several designated nodes to recover the loss will make these nodes become the bottleneck of transmission as they need to deal with abundant retransmission traffic. It is a good point to divide receivers into group for controlling NAK implosion and avoiding bottle-neck in the network. In [7], Alex Fung *et al.* suggest a reliable multicast scheme using grouping. But it is a pity that the protocol consumes substantial network resources to maintain the group relationship due to mobility in Ad Hoc networks.

In this paper, we address some key issues in designing efficient reliable multicast protocol in Ad Hoc networks: how to control NAK implosion, to reduce the recovery latency, to distribute the load for retransmission in Ad Hoc networks. We propose an efficient approach which applies the idea of cooperative communications to loss recovery for reliable multicast in Ad Hoc networks. We call this approach as Cooperative loss Recovery for Reliable Multicast (CoreRM) to address the challenges. In CoreRM, a receiving node experiencing a packet loss tries to recover the lost packet through progressively cooperating with neighboring nodes, upstream nodes or even source node. CoreRM aims at recovering the data loss within minimal number of hops and latency. In addition, CoreRM controls NAK implosion by not sending NAK immediately to avoid channel from competing with other nodes. CoreRM is robust to group topology changes, as nodes periodically exchange messages to obtain the information of upstream node by sending Source Path Messages (SPMs). Another feature of loss recovery scheme of CoreRM is the caching strategy. Nodes in the multicast tree cache not only data but also the recovery path which is carried by negatively acknowledge (NAK) or NAK confirmation (NCF). When the node has the recovery path of the loss packet, it can directly send NAK to the address recorded in the recovery path to request the retransmission of the lost packets. This strategy aims at maximizing reliability with minimal recovery latency through dispersing the recover traffic around the whole network.

The rest of the paper is organized as follows. Section 2 presents the proposed cooperative loss recovery scheme for reliable multicast CoreRM. Section 3 gives performance evaluation of CoreRM. Section 4 presents simulations experiments and numerical results. Finally, Section 5 concludes the paper and suggests future work.

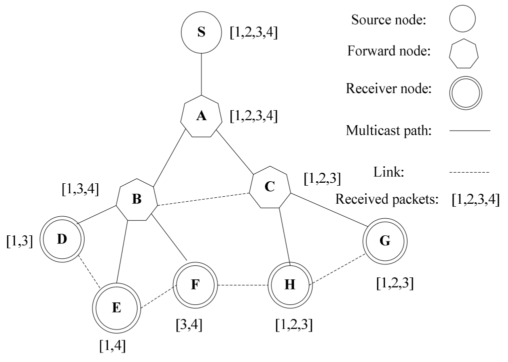

Figure 1. A simple example.

2. Cooperative Loss Recovery for Reliable Multicast

In this section, we first give sort of a general introduction concerning CoreRM, then present the details of Loss Recovery Scheme, Caching Scheme, NAK Controlling Scheme and Source Path Message (SPM) which are the primary parts of CoreRM.

2.1. Overview of CoreRM

CoreRM is a receiver-initialed, NAK-based scheme in which receivers are responsible for detecting and requesting the lost packets. CoreRM employs tree-based routing protocol, such as MAODV [8].With the cooperation among nodes in the network, CoreRM can distribute the burden of loss recovery into the entire network. Furthermore, every node in CoreRM maintains a Loss Recovery Path Vector (LRPV) routing table to register lost data packets' recovery path. Unlike data packets, LRPV table entries are compact and consume small amount of space. Additionally, LRPV does not use proactive or explicit messages thus avoiding extra communication overheads.

As a key component of CoreRM, our loss recovery scheme has four stages:

1) LRPV recovery. If a node experiencing the loss find that its LRPV has the recovery path of the lost packet, it sends NAK to the node indicated in the LRPV entry according to the recovery path. The details of LRPV and recovery path will be elaborated at Subsection 2.2.

2) Local recovery. If LRPV recovery fails, local recovery is executed. In this stages, receivers try to recover the missing packets from nodes of cooperation zone (collection any one-hop neighboring node). At beginning, node sends a NAK with broadcast network address, and any one-hop neighbor sends back an NCF. If the neighbor has cached some of the lost packets, it will retransmit the packet to the node which sends the NAK. At the same time, the routing table of LRPV is updated upon receiving NAK or NCF.

3) Global recovery. For the remaining missing packets, the receivers unicast a NAK to their individual upstream nodes along the reverse path of the tree. If those nodes cache the lost packets, they will retransmit them to the requester.

4) Source recovery. If all of above stages fail, source node will eventually receive the NAK. Then the source broadcasts the packet.

We give a simple example with a single multicast group in Figure 1 to illustrate the operation of loss recovery in CoreRM. Let S denote the source node, {A, B, C} denote forwarding nodes, and {D, E, F, G, H} denote receivers of the multicast group. Two nodes within each other's transmission range are connected by a solid link if the link is in the underlying multicast tree or by a dotted link otherwise. The source S sends packets with sequence numbers from 1 to 4. The underlying multicast protocol delivers them to multicast group members (receivers), and some packets are lost due to the lossy link. The bracket beside a forwarder or a member represents the set of cached packets, for an instance E has received packet 1 and packet 4, but missed packet 2 and packet 3. We now briefly describe CoreRM recovery process on members E and H.

1) E detects that packets 2 and 3 are lost. As there is no information about LRPV of other nodes exchanged with E at the beginning of loss recovery process, E's LRPV returns no result for helping loss recovery. 2) E initiates local recovery process by broadcasting NAK to its one-hop neighbors. The recovery destination address in LRPV piggybacked in the NAK sent by E is E's address for received packets 1 and 4 and null for the lost packets 2 and 3. All local nodes update their routing table of LRPV for packets 1 to 4 on hearing the NAK sent by E. They know that E has cached the packets 1 and 4, and lost packets 2 and 3. If they lose packets 1 or 4, they will add the address of E to the recovery path of packet 4. At the same time, they check their cache whether they have cached packets 2 or 3. 3) Neighbor nodes send NCF after receiving NAK, which also piggybacks the information of their LRPV. At the same time E updates its routing table of LRPV according to NCF and notes that packet 3 is one hop away in the direction of B, D and F. After receiving the retransmission of packet 3, E still misses packet 2. 4) E enters the global recovery stage. A NAK is sent to E's upstream node B. B cannot recover packet 2, and then sends a NAK to B's upstream node A. A finds that there is packet 2 in its buffer, and then broadcasts packet 2 to help all of down-stream nodes recover the loss. As the result, E recovers packet 2 and its CoreRM recovery process ends. 5) Now H starts loss recovery. With the recovery path of its LRPV, H knows that packet 4 is one hop away from F. H sends NAK to F for requesting retransmission of packet 4. After receiving packet 4, H ends recovery process. In addition, all nodes hearing H's NAK will update their routing table of LRPV.

2.2. Caching Scheme

In traditional recovery schemes, only source node or several receivers are responsible for the loss recovery [3–5]. When packet losses frequently occur in Ad Hoc networks, the repair nodes will become the bottleneck of transmission for their limited resources. As the forwarder may be closer to the sender than the receiver, the data caching at forwarders is useful for recovering the receivers' loss. CoreRM caches multicast packets at all the members of multicast tree, including forwarders and receivers for possible retransmission. Therefore, in CoreRM any forwarder or receiver which receives NAK can retransmit the requested packet if it has cached. Since end-to-end loss recovery latency over a large size network may be long, retransmissions from an intermediate router or short-range neighbor can significantly reduce recovery latency. Additionally, by distributing the burden of retransmission to nodes along the multicast tree and one-hop neighboring nodes, CoreRM tries to minimize the bottleneck nodes overwhelmed by retransmission requests and repair traffic.

Each packet with a unique ID (sequence, source address) is sent by multicast sender in serial. Receivers detect losses by sequence gaps in the data packets or by arriving no data within a certain interval. Receivers update the maximum sequence recorded in the buffer when new data packets arrive. If the new arriving data's sequence number is two or more greater than the maximum sequence number recorded, it means that the data does not arrive in order and some data packets are lost. The arrival packet is duplicated, if the arriving packet's sequence number is smaller than the maximum. Otherwise, the maximum sequence number of receiver's buffer is changed according to the arriving packet. When receivers detect a data missing, a NAK packet is generated to request the retransmission according to the loss packet, and the sequence number of NAK equals to that of the missing packet. Matching the sequence number of NAK and lost data packet is good for loss recovery without any confusion.

Each node maintains a LRPV routing table for a multicast group. It records the messages of missing packets' recovery path. Each LRPV routing table has the format as follows:

Table 1. LRPV routing table format.

PacketID=(S, seqNo)		Dst1	hops
32-bit	32-bit	32-bit	32-bit

where the packet ID is the key column identifying each lost packet. S is the address of sender and seqNo is the sequence number of the data packet. Dst1 is the address to reach the destination which caches the data packet identified by ID. Like other distance vector schemes, in the LRPV table each node only keeps track of the best target that has the minimal hops to a node experiencing packet loss. If the packet has been received, then the Dst1 of this packet is the address of receiver and the hops is 0.

The routing table of LRPV is exchanged via piggybacking on NAK or NCF packets. The packet format of a NAK/NCF packet is:

Table 2. NAK/NCF packet format.

Type	G	S	Rcv	SeqNo	[DV]
4-bit	32-bit	32-bit	32-bit	32-bit	N-unit

where Type is the packet type, NAK or NCF; G is the address of multicast group; S is the source address of the

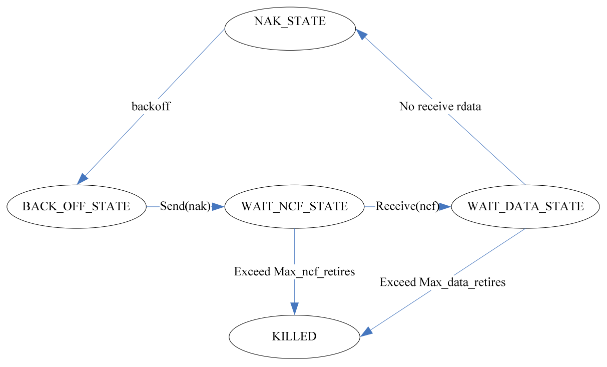

Figure 2. State transition diagram of CoreRM.

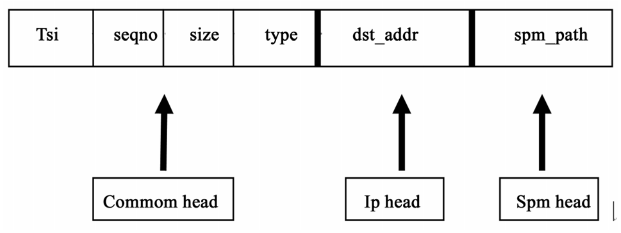

Figure 3. SPM packet format.

application session of multicast; Rcv is the address of receiver, e.g., a broadcast address in a NAK-broadcast or a unicast address of up-stream node in NAK-unicast; SeqNo is the data sequence number of the lost packet; and [DV] is a fixed field for piggybacking LRPV. In the [DV] field, the i-th unit [DV (i)] is the routing table about packet ID <S, seqNo + i>.

In details, a node updates its LRPV table upon receiving or overhearing a DATA, a NAK, an NCF, or the retransmitted packet (RDATA) as follows: 1) Upon forwarding or receiving a data packet, the node A caches the packet, then creates or updates the routing table of LRPV. <(S, seqNo), A, 0, 0>. 2) Upon overhearing a NAK or NCF, A updates its LRPV table according to the [DV] list. Note that [DV (i)] effects A's LRPV table entry identified by <S, seqNo + i> in the following two cases: 1) If the sequence number in [DV] list is equal to that in the corresponding LRPV, and the number of hops in D is lesser, A updates the entry with address in the LRPV. 2) If the data packet is not cached in local buffer, A creates the LRPV routing table according to the [DV] list.

2.3. NAK Fusion/Suppression

NAK implosion and explosion may take place when different nodes send NAK in the same time, different nodes send the NAK with the same sequence number, or a node incessantly sends NAK during the time of waiting retransmission. Therefore, we design NAK fusion/suppression mechanism in CoreRM to tackle the problem of NAK implosion and explosion.

Figure 2 presents the state transition of CoreRM. If the node detects a packet loss, it enters NAK_STATE and sets a random timer. Only after the timer expires, the node sends a NAK. Then the node's state enters WAIT_NCF_STATE and waits for receiving responding NCF. Node's state will change into WAIT_DATA_STATE on receiving NCF. The KILLED state can be entered in one of two ways. First, the counter of the node has exceeded the maximum number of MAX_NCF_RETIRES on waiting for NCF. Second, the counter of the node has exceeded the maximum number of MAX_DATA_RETIRES. CoreRM designs KILLED state to cope with the transmission link broken, and retransmission stops after trying maximum number of times to wait for RDATA or NCF. During the WAIT_NCF_STATE, nodes stop sending duplicate NAK to request the same lost packets. Only when nodes enter BACK_OFF_STATE, it sends NAK again.

2.4. Source Path Message (SPM)

CoreRM designs Source Path Message (SPM) packet to address the challenge of mobility in Ad Hoc networks. Nodes' moving will cause link frequently breaking and restructure of multicast topology. If the node can be aware of the link breaking immediately, it will stop sending packets through the broken link and promptly choose another path to transmit packets. Therefore, CoreRM periodically transmits session messages called Source Path Message (SPM) to tackle the mobility. SPM enables the network nodes to obtain the information of upstream node.

Figure 3 shows the SPM packet format. Tsi is the packet ID of SPM, which can be used to identify different data flows; Type is packet type, such as SPM; dst_addr is the address of destination; spm_path is the address of up-stream node. SPM sent by the source node sets the spm_path to the address of source. While a node receives the SPM packet, it has the information about up-stream node from SPM. Before forwarding SPM, node changes the address of spm_path into its own address. In this way, down-stream nodes can know the up-stream node's changing immediately.

3. Performance Evaluation

In this section we develop an analytical model to analyze the performance of the proposed CoreRM and compare it with that of PGM. We consider a scenario of multicast with one sender and multiple receivers. S is source node, and MHi is host i in the multicast group. Number of hops between S and MHi is Hi. The probability of data d_j being cached in MHi is p_{ij}. Transmission range of a host in the multicast is r (meters), and density of the nodes in the network is ρ. L_j is the number of hops to retrieve a data item, and T_j is the recovery latency to retrieve data item d_j. L_{avg} is the average number of hops which is the hops to recovery data. Reducing the hop count can reduce the query latency, bandwidth and the power consumption since few nodes are involved in the recovery process. Reduction in the hop count can also alleviate the load at data source since some of the requests are satisfied by local cache. T_{avg} is the average query latency which is the time elapsed between the retransmission request is sent and the data is transmitted back to the requester averaged upon all the successful recoveries. The latency of different loss recovery stages is different. T_Z is the average time to retrieve a data item from the local cache, T_g is average time to retrieve a data item from the global loss recovery, T_s is the average time to retrieve a data from S, and T_r is the maximum time to wait for a retransmission.

If P is the probability of cache hit at a node, then the probability of cache hit in a cooperation zone is $1-(1-P)^{\rho\pi r^2-1}$, as the area of a cooperation zone is πr^2. Considering the density of nodes in the network, the number of nodes in a recovery zone is $N_Z=\rho\pi r^2$. Because the requester itself cannot provide loss recovery, the number of nodes contributing for the cooperative zone hit equal $\rho\pi r^2-1$, and the probability of cache miss in a cooperative zone $=(1-P)^{\rho\pi r^2-1}$. Thus the probability of cache hit in a cooperative zone is $1-(1-P)^{\rho\pi r^2-1}$

So, L_j and T_j are given by

$$L_j=\left[1-\left(1-P_{ij}\right)^{\rho\pi r^2-1}\right]\cdot 1+\sum_{k=1}^{H_i-1}\left(1-P_{ij}\right)^{\rho\pi r^2}\cdot\left(1-P_{ij}\right)^{k-1}\cdot P_{ij}\cdot k$$
$$+\left(1-P_{ij}\right)^{\rho\pi r^2}\cdot\left(1-P_{ij}\right)^{H_i-1}\cdot P_{ij}\cdot H_i \quad (1)$$

$$T_j=\left[1-\left(1-P_{ij}\right)^{\rho\pi r^2-1}\right]\cdot T_z+\sum_{k=1}^{H_i-1}\left(1-P_{ij}\right)^{\rho\pi r^2}\cdot\left(1-P_{ij}\right)^{k-1}\cdot P_{ij}\cdot T_g$$
$$+\left(1-P_{ij}\right)^{\rho\pi r^2}\cdot\left(1-P_{ij}\right)^{H_i-1}\cdot P_{ij}\cdot T_S \quad (2)$$

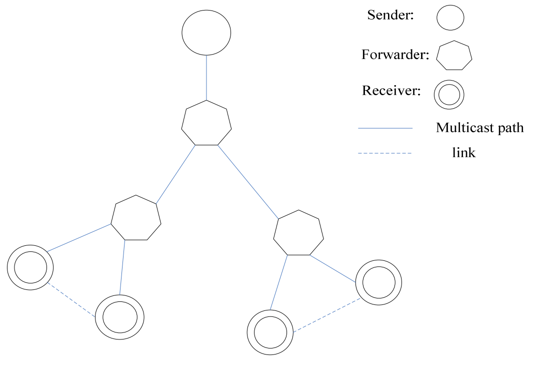

Figure 4. A simple multicast tree.

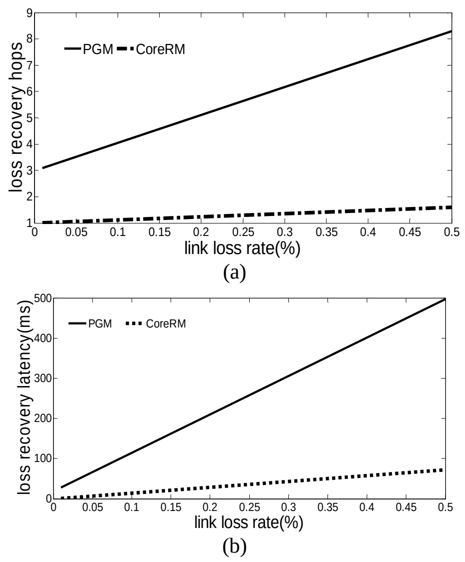

Figure 5. Effect of link loss rate. (a) Average number of hops; (b) Average recovery latency.

To simplify the model, some assumptions have been made as follows: 1) Every link has the same data loss probability p, so p_{ij} of each node can be compute by p. For instance, D caches data item d_j with the probability $(1-p)^3$. 2) Every node on the multicast tree will cache the transmitting data packet. The space of node's buffer is infinite. 3) Source node can complete all the retransmission. 4) Only data packet and RDATA packet will experience loss. Therefore, it maybe need same times retransmission to recovery loss.

For PGM with the multicast topology in Figure 4, we have

$$L_{avg}=\sum_{n=1}^{\infty}3\cdot(1-p)^3\cdot\left[p\cdot 1+(1-p)\cdot p\cdot 2+(1-p)^2\cdot p\cdot 3\right]^{n-1} \quad (3)$$

$$T_{avg}=\sum_{n=1}^{\infty}T_s\cdot(1-p)^3\cdot(n-1)\cdot T_r\cdot\left[p+(1-p)\cdot p+(1-p)^2\cdot p\right]^{n-1} \quad (4)$$

For our CoreRM with the multicast topology in Figure 4, we have

$$L_{avg}=\sum_{n=1}^{\infty}\left[\left(1-p^5\right)\cdot 1+(1-p)\cdot p^5\cdot 2+3\cdot p^6\right]$$
$$\cdot\left\{p^2\cdot\left[(1-p)\cdot p\cdot 2+p\cdot 1\right]\cdot\left[p\cdot 1+(1-p)\cdot p\cdot 2+(1-p)^2\cdot p\cdot 3\right]\right\}^{n-1} \quad (5)$$

$$T_{avg}=\sum_{n=1}^{\infty}\left[\left(1-p^5\right)\cdot T_z+(1-p)\cdot p^5\cdot T_g+p^6\cdot T_s\right]$$
$$\cdot(n-1)\cdot T_r\left\{p^2\left[(1-p)\cdot p+p\right]\cdot\left[p+(1-p)\cdot p+(1-p)^2\cdot p\right]\right\} \quad (6)$$

Figure 5 illustrates the effect of transmission loss on average number of recovery hops and average recovery

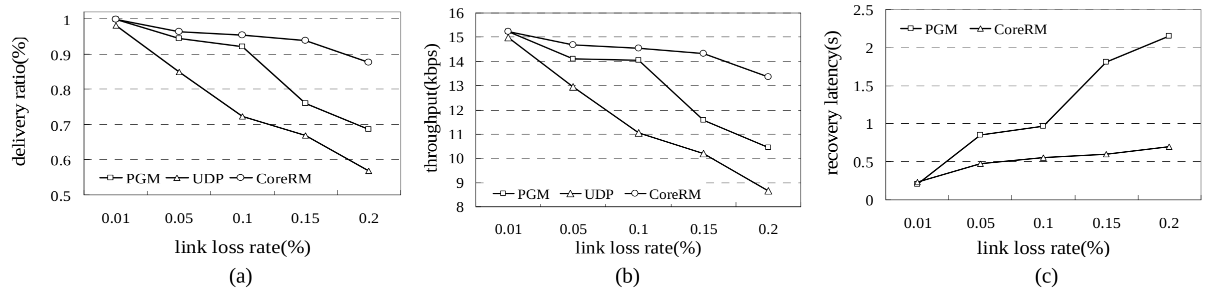

Figure 6. (a) Multicast packet delivery ratio; (b) Multicast throughput; (c) Recovery latency.

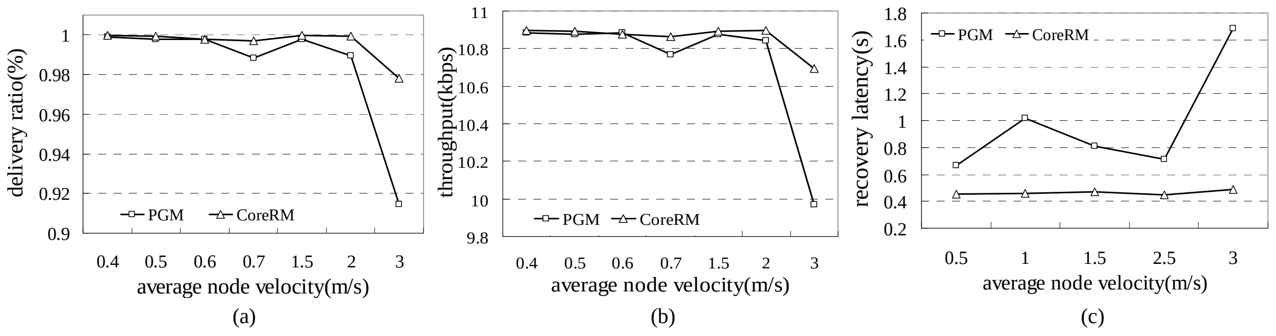

Figure 7. (a) Multicast packet delivery ratio; (b) Multicast throughput; (c) Recovery latency.

latency for CoreRM and PGM respectively. As expected, CoreRM's exhibits significantly lower recovery hops and latency than PGM as the latter uses source node to recover the lost. As the error rate of the link increases, the number of lost packets increases. In PGM, as the source node is in charge of recovering the packet loss, every request is sent to the sender and the source node broadcasts the retransmitted packets through the whole tree. So PGM perform poorly in Ad Hoc networks due to superfluous overhead. Figure 5(a) also shows that the number of recovery hops with CoreRM is less than 2, while it is more than 3 with PGM. Figure 5(b) shows that PGM's recovery latency increases much more rapidly than CoreRM's.

4. Simulation and Numerical Results

In this section, we use the ns-2 simulation experiments to evaluate the performance of CoreRM and compare it with PGM, UDP. In our simulations, MAC/802.11 based Ad Hoc network is assumed. MAODV [7] is used as multicast routing protocol. As PGM was designed for wired network environment, we make some modifications (such as interfaces) to make it working in Ad Hoc networks. For all the experiments, 16 nodes are randomly placed in a 670m × 670m area. The transmission range of every node is 250m. Network bandwidth is 2Mbit/sec. The source node continuously sends CBR traffic with data payload of 210 byts to the multicast group.

We examine three protocols' behaviors under the following two setting: 1) In static network environment where we assume the nodes do not move, we focus on the performance of three protocols with different link packet loss rate. 2) In mobile network environment where the nodes move according to Random Way Point [9] pattern, where we focus on the performance as a function of node moving velocity, and we explore the effect of node mobility. In our study, we use the following performance metrics: 1) Multicast Packet Delivery Ratio: is the ratio of packets successfully received by all multicast receivers to the total number of data packets sent by a sender. 2) Multicast Throughput: measures throughput of packets reliably delivered, i.e., packets that are received by all members. 3) Recovery Latency: is the time elapsed between the request is sent and the data is transmitted back to the requester averaged upon all the successful queries.

Figure 6 shows the performance of CoreRM in terms of multicast packet delivery ratio, multicast throughput and recovery latency in the static network. For comparison purpose, we also show the performance of UDP and PGM. From Figure 6(a) and Figure 6(b), we can observe that CoreRM always has the highest data delivery ratio and multicast throughput. When the link packet loss rate increases from 0.01 to 0.2, the difference between the performance of CoreRM and PGM and UDP becomes more significant. UDP has the worst delivery ratio and throughput performance as UDP is a connectionless protocol which takes no measure to recover the lost packets. PGM is a reliable multicast protocol using source node to recover the lost packet. However, source-based loss recovery will increase the overhead of the whole network.

Thus PGM's performance is better than UDP, but worse than CoreRM. CoreRM chooses neighboring nodes, up-stream nodes and even source node to distribute the overhead of loss recovery. As a result, the recovery latency of CoreRM is shorter than that of PGM, as shown in Figure 6 (c). When the link packet loss probability is large, the recovery latency of PGM is almost three times of that of CoreRM. Figure 7 shows the performance of PGM and CoreRM in terms of delivery ratio, throughput and recovery latency in mobile Ad Hoc networks. Frequent link breaking and reconstruction of multicast topology cause lots of retransmission overhead and consume resources, thus the mobility effect the performance of PGM and CoreRM substantially in mobile Ad Hoc networks. In Figures 7(a) and (b), CoreRM always outperforms PGM in terms of both delivery ratio and throughput, especially when node moving velocity is greater than 2.5m/s. The packet delivery ratio is close to 100% when the node moving velocity is small for CoreRM. From Figure 7(c), we can see that the recovery latency of PGM is significantly greater than that of CoreRM, especially when the node moving velocity becomes large. This is mainly due to the long distance transmission of NAK and recovery packets in PGM. Instead, our CoreRM is able to successfully recover the packets with mobility and guarantees perfect reliability.

5. Conclusions

In this paper we have developed a cooperative loss recovery scheme for reliable multicast in Ad Hoc networks called CoreRM. CoreRM's loss recovery process combines the nodes located inside cooperation zone, multicast path and even source to expedite the loss recovery. The aim of CoreRM design is to balance the overhead incurred by the loss recovery to the whole network, and try to minimize the recovery overhead and latency through short-range nodes' caching. Another distinguished feature of CoreRM is that every node caches data packet and routing table to register lost data packets' recovery path. In addition, CoreRM periodically transmits SPM to enable the network nodes to obtain the information of network topology changing in time. In addition, CoreRM takes some measure to tackle the NAK implosion and explosion. Through simulation experiments, we evaluate CoreRM's performance in static Ad Hoc networks and mobile Ad Hoc networks. Our numerical results show that CoreRM outperforms PGM and UDP in terms of multicast packet delivery ratio, throughput and recovery latency.

In this paper, we have not addressed the issue of congestion control in CoreRM. Joint congestion control and cooperative loss recovery may bring in significant performance improvement, which is one of our future research focuses. Another interesting idea is to exploit network coding for local loss recovery to further improve the performance for reliable multicast in Ad Hoc networks.

6. Acknowledgement

This work was supported by NSFC (National Science Foundation of China) grant No. 60833002 and grant No. 60772142.

7. References

[1] J. Gemmell, T. Montgomery, T. Speakman, N. Bhaskar, and J. Crowcroft, "The PGM reliable multicast protocol," Network, IEEE, Vol. 17, No. 1, January–February, 2003.

[2] F. Xie and G. Feng, "The impact of loss recovery on congestion control for reliable multicast [J]," IEEE/ACM Transaction on Networking, Vol. 14, No. 6, pp. 1323–1335, 2006.

[3] T. G. Mukesh, S. D. Panda, and P. Sadayappan, "A reliable multicast algorithm for mobile ad hoc networks," Distributed Computing Systems, Proceedings of the 22nd International Conference, pp. 563–570, July 2–5, 2002.

[4] B. Ouyang, X. Y. Hong, and Y. J. Yi. "A comparison of reliable multicast protocols for mobile ad hoc networks," SoutheastCon, Proceeding of IEEE, pp. 339–344, April 8–10, 2005.

[5] R. Chandra, V. Ramasubramanian, and K. P. Birman. "Anonymous gossip: Improving multicast reliability in mobile ad hoc networks, distributed computing systems," In IEEE ICDCS, pp. 275–283, 2001.

[6] V. Rajendran, Y. Yi, K. Obraczka, S. J. Lee, K. Tang, and M. Gerla, "Reliable, adaptive, congestion-controlled ad hoc multicast transport protocol: Combining source-based andlocal recovery," UCSC Technical Report, 2003.

[7] A. Fung and I. Sasase, "Hybrid local recovery scheme for reliable multicast using group-aided multicast scheme," pp. 3478–3482, March 11–15, WCNC, 2007.

[8] Y. F. Zhu and T. Kunz, "MAODV implementation for NS-2.26," Systems and Computing Engineering, Carleton University, Technical Report SCE-04-01, January 2004.

[9] B. Johnson and D. A. Maltz, "Dynamic source routing in ad hoc wireless networks," Mobile Computing, T. Imielinski and H. Korth, eds., Chapter 5, Kluwer Academic Publishers, pp. 153–181, 1996.