

Tree-Based Reliable Multicast in Combined Fixed/Mobile IP Networks

Wonyong Yoon Dongman Lee Chansu Yu Myungchul Kim
School of Engineering, Information and Communications University
58-4 Hwaam-Dong, Yuseong-Ku, Taejon 305-732, Korea
{ wyyoon, dlee, cyu, mckim }@icu.ac.kr

Abstract

This paper proposes a solution to delivering multicast data reliably not only to/from fixed hosts but also to/from mobile hosts in combined fixed/mobile IP networks, particularly with regard to the per-source shortest path multicast routing protocol and remote subscription option in IETF mobile IP multicast standard. We propose exploiting tree-based error recovery, well-established concept for reliable multicast in fixed networks, where a hierarchical logical tree of a source and receivers is constructed over which the responsibility of error recovery is distributed. We introduce a new entity named ARMM (Agent for Reliable Mobile Multicast) located at each cell of wireless/mobile networks. Each ARMM participates in a logical tree as a parent of mobile hosts in the corresponding cell. It is responsible for retransmission and requesting for retransmission on behalf of the mobile hosts. With aid of ARMM, the proposed scheme features (1) adaptive reconstruction of logical trees due to host mobility, (2) retransmission to mobile hosts via optimal routes, (3) abstracting a group of mobile hosts into a single ARMM (and thus reducing logical tree maintenance overhead), (4) efficient use of wireless bandwidth for retransmission, and (5) low processing burden on mobile hosts themselves. The experimental results show that the proposed scheme provides higher scalability than a scheme applying a generalized tree-based reliable multicast in mobile environments.

1. Introduction

Reliable multicasting guarantees that every packet from a source is transmitted to the receivers of a multicast group without any error and also in order. It is a valuable transport layer service to applications such as software distribution or multimedia conferencing. There have been many research works on reliable multicast services [1][2][3][4][5][6][7]. They attempt to provide scalable and efficient solutions to acknowledge implosion and retransmission exposure.

These issues get more complicated as receivers dynamically join and leave a multicast group.

As mobile networks become common, it is well anticipated that the members of a multicast group are a combination of fixed and mobile hosts. Although a number of various approaches for reliable multicast have been recently proposed [7], most of them do not consider mobile members in a session and thus there is no guaranteeing that they work that much well also in the presence of mobile characteristics such as handoff of mobile hosts or heterogeneity of hosts and networks. These characteristics may give rise to some new important issues [8] that can be appropriately dealt with only by an extension or modification of the existing approaches. A few mobile reliable multicast protocols [9][10][11] do consider a combined fixed/mobile environment. Assuming the existence of a reliable multicast protocol for fixed hosts, they handle only the reliable delivery of multicast packets to mobile hosts from agents that relay the packets from the fixed hosts on behalf of the mobile hosts. As a mobile host likely roams from one cell to another, the agents cooperate with one another for the appropriate buffer management so that the mobile host may not lose the chance of receiving data packets due to handoff. In addition, it is critical for overall performance of reliable multicast in a combined host environment to use a reliable multicast protocol that can adapt to host mobility, i.e., dynamic membership change of the agents, as well in a scalable manner. However, the current mobile reliable multicasts do not allow the fixed part of sessions to adapt to dynamics of the mobile counterpart.

In this paper, we propose a scalable tree-based reliable multicast protocol that transparently handles membership dynamics due to host mobility as well as that of fixed hosts, and supports reliable delivery to/from mobile hosts. We introduce a new entity named ARMM (Agent for Reliable Mobile Multicast) in each cell. Each ARMM, participating in a logical tree, is responsible for retransmission and requesting for retransmission on behalf of the mobile hosts. It abstracts a group of mobile hosts into a single ARMM and thus reduces logical tree maintenance overhead. With

ARMM, the proposed scheme enables retransmission to mobile hosts via optimal routes. This leads to efficient use of wireless bandwidth for retransmission and low processing burden on mobile hosts. The proposed scheme also deals with the reliable delivery of multicast data from mobile hosts while most of previous works consider only reliable multicast delivery to mobile hosts. The experimental results show that the proposed scheme reduces acknowledgment implosion from 1.2 to 5 times compared with a generic tree-based scheme, depending on the session size and the ratio of mobiles. We assume that IETF remote subscription is used as mobile IP multicast [12].

The rest of the paper is organized as follows. Section 2 gives an overview of existing approaches. Section 3 identifies the issues that host mobility may cause when a tree-based error recovery is applied. Section 4 describes in detail the proposed solution. Section 5 presents and analyzes the results of simulation experiments on the proposed scheme. Conclusion is drawn in Section 6.

2. Related works

There have been various approaches for providing a reliable multicast service in fixed IP networks [7]. Tree-based reliable multicast protocols such as RMTP (Reliable Multicast Transport Protocol) [1] are known to be most scalable in terms of throughput. They construct a hierarchical logical tree in which a source and receivers are configured as a root and intermediate nodes or leaves respectively. In the logical tree, a parent node is responsible for reliability of its children nodes. The parent node receives acknowledgments (positive and/or negative) from its children and retransmits packets missed by them. By capitalizing the inherent structure of trees, tree-based protocols efficiently distribute the responsibility of reliable multicast to the source and the receivers of multicast, and also localize retransmission of lost packets to retransmitter's descendants. This combined distributed/localized error recovery contributes largely to alleviate implosion of acknowledgments and exposure to retransmissions and thus to enhance the protocol scalability. Our previous experience shows that the approximation of a logical tree to a multicast routing tree improves the scalability further [3].

Multicast support for mobile hosts has been studied largely based on IETF mobile IP standard. IETF mobile IP multicast specification [12] provides two options for receiving multicast datagrams. In remote subscription, a mobile host with a co-located care-of address receives multicast packets directly by link-level multicast in the foreign network. In bi-directional tunneling, a mobile host receives multicast packets via a tunnel from its home agent to the current foreign agent. Remote subscription provides optimal routing at the cost of exchange of multicast routing

messages due to join/leave of mobile hosts. Meanwhile, bi-directional tunneling tolerates route non-optimality to eliminate cost involved with join/leave of mobile hosts. MoM protocol [13] further enhances bi-directional tunneling by addressing the tunnel convergence problem. It ensures that only one home agent tunnels to a foreign agent even when more than one mobile hosts from different home cells reside in the foreign cell. Xylomenos and Plozoz point out that remote subscription will be eventually dominant due to the strength of optimal routing [14].

There have been a few research works that focus on the reliable delivery from mobile agents to mobile hosts. HVMP (Host View Membership Protocol) provides exactly-once semantics for multicast [9][8]. For this, each Mobile Support Station does not discard packets from its memory so that newly incoming mobile hosts can receive the packets immediately from it (the new MSS), even though all the mobile hosts local to the MSS have received the packets successfully. RelM also guarantees exactly-once delivery of multicast messages to mobile hosts in a session [10]. In its new three-level hierarchy, Supervisor Hosts on top of multiple Mobile Support Stations collect acknowledgments and forward them to the source of the multicast. Unlikely to HVMP, Supervisor Hosts with no session members need neither receive multicast packets nor be involved with procedure of reliability enforcement. A logical ring reliable multicast protocol for mobile nodes is most recently proposed [11]. The motivation is the same as HVMP. Using a token circulated along a logical ring of base stations, each base station agrees on the maximum sequence number below which all packets can be considered as delivered to all the mobiles. Thus packets can be safely discarded from memory without newly incoming mobile hosts losing the chance of receiving these packets.

3. Design considerations

In Figure 1, rectangles and circles represent multicast routers and end hosts respectively. Bold rectangles - R4, R5, and R6 - are home agents and/or foreign agents and bold circles D and E are mobile hosts. We assume that R4 is a home agent for a mobile host E and R5 is that for D.

We consider the case when a fixed host S is a multicast source. Based on tree-based error recovery, we construct the corresponding logical tree (detailed in the next sections). In the tree, a pair of nodes has parent-child relationship with each other and each parent is responsible for the reliability of its own children.

Suppose that the corresponding logical tree remains unchanged when a mobile host D moves into a foreign agent R4's network. Based on our assumption that remote subscription is used for mobile IP multicast, D even then continues receiving multicast packets from the source via the

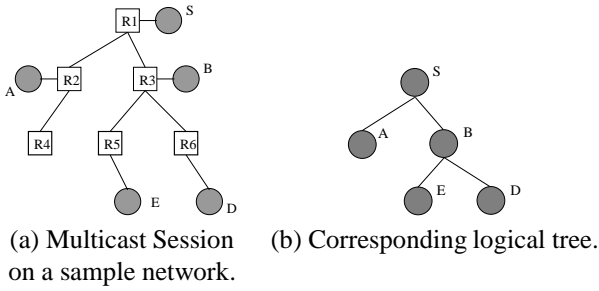


Figure 1. Multicast on a combined fixed/mobile network.

link between R1 and R2, the link between R2 and R4, and the wireless link from R4 to E. However, since D's logical parent is still B, if D experiences a loss, it requests B to retransmit the lost packet. Now that the transmission path from S to B and that from S to D are totally independent (both paths do not share any link), losses at the mobile host D does not have any correlation with those at B. Considering packet loss correlation, A is more eligible for a logical parent of D than B. Therefore the logical tree gets changed as well so that more eligible parents can be liable for the reliability of children. However, with mobile IP standard and current reliable multicast solutions, the need to modify the logical tree cannot be even recognized. We refer to this problem as the *irresponsive logical tree* problem.

Suppose that some packets from the source S to a mobile host E are lost in transit either on a link between R3 and R5 or on a wireless link between R5 and E. The tree-based error recovery mandates that the logical parent B retransmit the lost packets to the logical child E. Based on the IETF mobile IP standard [12], packets which are unicast from B to E should be transmitted first to E's home agent R4. The packets reach the destination node E via a tunnel from R4 to the foreign agent R5. This is clearly inefficient in terms of bandwidth consumption and routing delay. We refer to this as the *retransmission route non-optimality* problem.

When more than one mobile hosts in the same cell experience the loss of the same packet, multiple unicast retransmissions to each mobile hosts may converge on a single wireless link from a mobile (foreign/home) agent to the mobile hosts thus resulting in duplicate retransmissions. This situation is referred to as the *retransmission duplication* problem.

The tree-based error recovery requires that each host in a given reliable multicast session should maintain a regional part of a logical tree, that is, its (logical) parent and children. As the number of active sources in the multicast session increases, the processing overhead imposed to maintain the logical tree may not be acceptable for mobile hosts which usually have rather low computing power. We refer to this

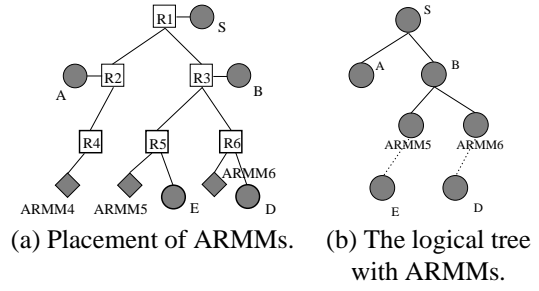


Figure 2. Introduction of ARMM (Agent for Reliable Mobile Multicast).

as the *high processing burden at mobile hosts* problem.

We have considered the case where the source of multicast is a fixed host. Even when a mobile host, say, D in Figure 1, is the source of multicast, similar problems also can occur. If the mobile host D moves into other networks away from the current R6's network, then the multicast routing path for packets emitted by D should be recalculated by the underlying multicast routing protocol (in this paper we assume the protocol to be per-source shortest-path multicast routing). Since the change of multicast routing path may result in inefficiency of a current logical tree rooted at D, the logical tree needs to be modified reflecting the change. Here the same problem arises - the *irresponsive logical tree* problem. Furthermore a mobile host as a source of multicast still should maintain information on its own children in a logical tree. Retransmission duplication problem and retransmission route non-optimality problem also may arise in a case that more than one mobile host beside the multicast source mobile host is in the same cell. If the source mobile host retransmits packets to other receiving mobile hosts via unicast tunnels, the retransmitted packets will follow routes far from optimal and wireless bandwidth within the same cell will be redundantly used for the retransmission to each receiver.

4. The proposed scheme

In order to address the aforementioned issues, we propose a scheme based on ARMM (Agent for Reliable Mobile Multicast). Placed in each wireless cell, each ARMM acts as a proxy for mobile hosts in the cell. On behalf of the mobile hosts, ARMM participates in a logical tree and takes the responsibility of error recovery - retransmission and request for retransmission. In Figure 2 (a), we place three ARMMs and the resulting logical tree is shown in Figure 2 (b).

4.1. Construction/reconstruction of logical trees

The way we construct logical trees is a key point to dealing appropriately with the problems described in Section 3. For efficient error recovery, it is important to ensure that a logical tree be organized as close to the underlying multicast routing tree as possible [3].

4.1.1. Error bitmap. In order to construct a logical tree as close to a multicast routing tree as possible, we exploit the fate sharing property in IP multicasting [15]. Each receiver in a multicast session maintains a packet delivery status called *error bitmap* similar to the status message of RMTP [1]. An error bitmap consists of two parts: sequence number (N_s) and actual bitmap (B). N_s is the sequence number of the first packet in a sequence of packets. One bit of B indicates the status of transmission of the corresponding packet; 1 means a successful packet delivery to the receiver and 0 indicates a packet loss. For example, $N_s = 5$ and $B = 11010$ means that the receiver has successfully received packets with sequence numbers 5, 6, and 8.

Each receiver feeds the error bitmap information back to the source or its parent in the logical tree. If a bit in the error bitmap at a node is set to 1, the corresponding bit in the error bitmap of its parent is much likely to be set to 1. Note that the error bitmap of a source always consists of all 1's.

4.1.2. Construction of logical trees. We first define two relational operators and an operation between a pair of nodes in a logical tree as follows.

Definition 1 (Child) We say that two nodes N_i and N_j have the relationship $N_i \subset N_j$ (which reads N_i is a child of N_j) if and only if $B_k(N_i) \leq B_k(N_j)$ for all $k = 1, 2, \dots, l$. Here $B_k(N_i)$ denotes the k -th bit (from the left) of the bitmap of node N_i , while the bitmap length is assumed to be l .

Definition 2 (Parent) Similarly, $N_i \supset N_j$ (which reads N_i is a parent of N_j) if and only if $B_k(N_i) \geq B_k(N_j)$ for all $k = 1, 2, \dots, l$. The relationship, \supset , is thus a complementary relationship to \subset .

Definition 3 (Adoption) $N_i \Leftarrow N_j$ (which reads N_i adopts N_j) when N_j is attached to N_i as a child in the logical tree.

Based on these definitions, a logical tree is being formed as follows. A newcomer, N , in a session is first attached to the tree as a *tentative* child of the source, S , by informing the source of its IP address. We assume that the newcomers are aware of the multicast group address and the source address. The newcomer N feeds its error bitmap back to the

source S when 32 bits get filled up. For the discussion of bitmap size refer to our previous work [3]. Receiving the error bitmap information from the newcomer, the source S compares it with those of its children to check if the newcomer has child or parent relationship with any child C . Join algorithm can be divided into the following four cases.

Case i) N has no relationship with any child of S . Then $S \Leftarrow N$ as a *regular* child.

Case ii) $N \supset C$ but not $N \subset C$. Then $S \Leftarrow N$, and $N \Leftarrow C$.

Case iii) $N \subset C$ but not $N \supset C$. Then S sends N 's error bitmap information and IP address to C . The procedure is repeated downward the tree until N 's proper parent is found.

Case iv) $N \supset C$ and $N \subset C$. Then $C \Leftarrow N$.

Note that the logical tree in Figure 1 (b) is constructed using this Join algorithm.

Leave algorithm works as follows. The logical tree is also reconfigured when a receiver leaves a session. A receiver wishing to leave a session notifies its parent. The parent, P , adopts the children of the leaving receiver as immediate children, and orders them to change their parent to P . Due to failures, a receiver may leave the tree without notifying its parent. This results in its children detached from the tree. When the children become to find out that their parent does not exist, they join the group as tentative children of the source. Afterward, Join algorithm is processed again until the children find a right parent.

When a mobile host joins as a receiver and it is the first one in the cell, it requests its corresponding ARMM to perform Join algorithm and thus to join the logical tree on behalf of the mobile host.

The proposed scheme also allows a mobile host to be a sender. Since a mobile host itself does not participate in a logical tree, fixed hosts and other ARMMs should be notified that the mobile host is represented by its corresponding ARMM as they join the session. We refer to this notification as *binding*. This makes it possible that the mobile host transparently behaves as a sender while its ARMM plays a root of a logical tree on behalf of it for error recovery. The binding is also used when a mobile host roams from one cell to another. In Section 4.5, the binding procedure will be described in detail.

4.1.3. Reconstruction of logical trees adaptive to host mobility. As a mobile host as a sender or a receiver of a multicast group moves to a different cell in a mobile network, its corresponding ARMMs may need to join or leave the session. When a mobile host MH leaves the current cell, the corresponding ARMM, A1, notifies the session of this

event. A1 also checks if there remains any other mobile host in the cell belonging to the same session. If so, A1 continues to stay in the session and the logical tree. Otherwise, it performs Leave algorithm thus leaves both the session and the logical tree.

When MH moves into a new cell, a new ARMM, A2, will join the session and be attached to the logical tree if no mobile host in the cell has joined the session before. At first, A2 will be adopted as a child of the parent of A1.¹ The parent then checks the error bitmap of A2 to further find a proper position of A2. If there exists a child of the parent such that the child \supset A2, a similar procedure to Case iii) in Join algorithm will be performed. If the parent finds that \supset relationship with A2 is no longer valid, it requests its logical parent to locate a new position of A2. This will repeat upward the logical tree until a node having \supset relationship with A2 is found. Then, Join algorithm will be performed again at the node downward the tree. A2 also sends a binding message telling that it will act on behalf of MH1. In case that the mobile host is a source of multicast, A1-rooted logical tree should be changed into A2-rooted logical tree. For smooth change, the children of A1 are first attached to the child nodes of A2 and other descendants of A1 participate in the logical tree with the same parent-child relationship as in the previous A1-rooted logical tree. Later by exchanging error bitmap information, the nodes may reconstitute more optimal logical tree. If one or more mobile hosts have been already staying in the session at the time of handoff, no work is needed except the transmission of binding messages regardless of whether the mobile host is a source or a receiver.

The main advantage of binding multiple mobile hosts to a single ARMM is that only the ARMM is involved with the corresponding logical tree. Another advantage is that the handoff of mobile hosts does not always entail join/leave of ARMM and thus reduces messaging overhead - such as exchange of prune/graft messages - in an underlying multicast routing protocol due to this join/leave. A careful reader will notice that the enforcement of Join/Leave algorithm both in initial construction of logical trees and in adaptive reconstruction of those trees depending on host mobility clearly addresses the irresponsive logical tree problem.

4.2. Retransmission via ARMM

The proposed scheme, as a generic tree-based reliable multicast protocol, requires each node in a logical tree to be responsible for its children and to depend on its parent for reliable packet delivery. Each node feeds two kinds of control information back to its parent, i.e., positive acknowledgment and negative acknowledgment. Each node checks

¹This is different from the case of a fixed host other than ARMM, which is attached to the tree as a child of the source.

if all of its children have received the packet sent to them as it receives positive acknowledgments from its children. Then it releases the buffer space assigned for the packet. Note that a positive acknowledgment need not be sent per packet since it mainly concerns buffer management. In fact, in our scheme positive acknowledgments can be fed back along with transmission of error bitmap information that is sent periodically for the construction and maintenance of logical trees.

Negative acknowledgments ensure timely recovery of lost packets. Each node sends a negative acknowledgment to its parent immediately detecting a gap in the received packets, and sets a timer for the packet. When the parent receives a negative acknowledgment, it unicasts the lost packet to the requester if it has the packet. This process repeats if the timer expires without receiving the repair packet. The timeout implies that either a negative acknowledgment or retransmission is lost.

The proposed scheme allows only for unicast retransmission from a parent to a child in a logical tree. Although unicast retransmission has a weakness in that one retransmission can repair only one requester, it helps greatly reduce overall bandwidth consumption for error recovery since the retransmitted packets follow near-optimal recovery paths in the corresponding logical tree. Recall that the near-optimal logical tree is ascribed to Join/Leave algorithm.

The acknowledgment and unicast retransmission described above is applicable only to fixed hosts. Should fixed hosts retransmit lost packets to mobile hosts in the same manner, that is, via unicast, they would suffer from what we call the retransmission route non-optimality problem. The unicast packet should get first to the home agent of a mobile host and then be passed to the foreign agent of the current foreign network and lastly to the mobile host. It is due to triangular routing in mobile IP [12]. Consequently we need to put in force a different method of retransmission for mobile hosts.

Mobile hosts themselves in fact do not join the logical tree for error recovery. Detecting a loss of a packet (identified by gaps in the sequence numbers of packets received), the mobile hosts send their own negative acknowledgments to the corresponding ARMM. Receiving negative acknowledgments, the ARMM requests for the packet if it has not detected the loss yet. If it has already received the packet and keeps the packet still in its buffer, it is ready to retransmit the packet.² It multicasts the packet using link-level multicast so that multiple mobile hosts who experience the same loss can be repaired by the single multicast packet. The repair packet is emitted immediately at the receipt of

²It is possible in a worst case that the ARMM has already received the packet but it has released the packet from the buffer since mobile hosts having already joined the cell have also received the packet successfully before new one joins the cell. This case will be deferred for later discussion.

the negative acknowledgment.

In case that the source of a multicast group is a mobile host, unicast retransmission from the mobile host to its corresponding ARMM will be done if the ARMM experiences losses. The rest of receivers never contact directly the mobile host to request for retransmission. They contact the ARMM instead. The mobile receivers who are in the same cell as the source get repair services from the ARMM via multicast retransmission.

To summarize, retransmission route optimality in the proposed scheme comes out twofold. One is optimization of a recovery path between fixed nodes in a logical tree by approximating the tree as close to the corresponding underlying routing tree as possible. The other is elimination of triangular routing in retransmission to mobile hosts by dividing a single retransmission into two steps: unicast retransmission from a fixed host to an ARMM and multicast retransmission from the ARMM to mobile hosts.

4.3. Tree maintenance by ARMM

ARMM should maintain the information on its mobile hosts and update the information if necessary. The information is fed by the binding protocol (explained in Section 4.5). ARMM also feeds back periodically its own error bitmap to logical parents and processes error bitmaps from its logical children to cope with the changes in the fixed part of the combined networks. The changes include the join/leave of fixed hosts or other ARMMs and the change in the underlying multicast routing tree. As mobile hosts join and leave a cell, the corresponding ARMM is involved with the reconstruction of a logical tree by performing Join/Leave algorithm.

The tree maintenance by ARMM on behalf of its corresponding mobile hosts has two distinguished advantages. One is that by abstracting a group of mobile hosts into a single ARMM, we can reduce logical tree maintenance overhead as much as the number of such hosts. The other is that mobile hosts with low computing power are not directly involved with logical trees and less processing burden is imposed on them.

4.4. Message stability

A multicast packet is called stable if it is received by all of the receivers of multicast. If an unstable packet is discarded from the buffers of all other members in the session including the source of multicast, the receivers that has not yet received the packet would be in deadlock. In other words, they would request continuously for the packet that does not exist any longer.

Using positive acknowledgments for buffer management is sufficient for tree-based reliable multicast in fixed net-

works if we assume failure-free environments. A node can safely discard a packet if it receives positive acknowledgments from all of its children. However, with such a buffer management mechanism alone, we cannot provide mobile hosts with message stability.

4.4.1. Receiver message stability. We do not allow ARMM to discard packets even though it receives positive acknowledgments from all the corresponding member mobile hosts in the cell. It should keep the packets until it is confirmed that the mobile hosts will have eventually received the packets. Suppose that a source of a multicast group sends packets with sequence numbers from 1 and a mobile host MH hands off from the cell C1 to the cell C2. Suppose that A1, C1's ARMM, has received packets up to l but MH receives packets from 1 to k in C1. Suppose also that A1 gets the acknowledgement of the packets up to m from all of its mobile hosts including MH ($m < k < l$). Thus A1 retains the buffer space for packets with sequence numbers from $m + 1$ to l . Then there exist two cases to consider due to handoff.

Case i) ARMM in the new cell has not yet joined the session.

A2, C2's ARMM, joins the session immediately when it finds that the new mobile host MH has moved into its cell. At that time MH has already joined the session (remote subscription). Suppose that MH receives the first packet with sequence number p and A2 receives the first packet with sequence number q where $p \leq q$. MH requests to A2 for packets from $k + 1$ to $p - 1$. In the request, MH also loads information that its previous ARMM was A1. Receiving the request, A2 will resort to A1 since it does not have those packets. A2 sends to A1 a request packet containing k and q . A1 then hands over the packets from $k + 1$ to $q - 1$ to A2 and discards the packets from memory. Until the completion of hand-over of those packets, A1 is not allowed to leave the session even if no mobile host resides at C1 any longer. A2 relays those packets to MH via unicast. If more than one mobile host has roamed to C2, multicast relay of missing packets would be more desirable since multiple mobile hosts can receive the packets at one time.³

Case ii) ARMM in the new cell is already a member of the session.

Receiving the first packet with sequence number p in C2 and detecting the gaps, MH will request to A2 for the pack-

³In general, which of unicast or multicast relay we should choose may depend on which of processing power of mobile host or wireless bandwidth is more important. Multicast relay consumes sparingly rather scarce wireless bandwidth with other hosts being redundantly exposed to the packets while unicast relay avoids host exposure at the redundant use of wireless bandwidth for the same packet.

ets from $k + 1$ to $p - 1$. In the request, MH1 loads information that its previous ARMM was A1. At that time, A2 has already received packets before p . Suppose that A2 retains packets from $q + 1$ to r in memory. If r is less than or equal to k , then A2 tells A1 that it can release memory space for the packets from $m + 1$ to l since A2 will eventually receive those packets later and thus send them to MH. If k is less than r and q is less than or equal to k , it again informs A1 that it can safely discard the packets since A2 has already received the packets and still keep them in buffer. Finally if k is less than r and k is less than q , it has no choice but to resort to A1. This happens when A2 has already received the packets but has just deleted them because all the previous mobile hosts had successfully received those packets since A2 retains packets with the sequence number starting from $q + 1$. A1 should hand over the packets from $k + 1$ to q to A2. Again, A1 is not allowed to leave the session until the completion of hand-over. Then A2 sends those packets to MH via unicast or link-level multicast according to a chosen policy.

In both cases, a key point is how we can make A1 keep the missing packets. We must not let A1 leave the logical tree and delete all the information about the session or all the packets that are received by A1 but not acknowledged (the packets from sequence number $m + 1$) even though MH was the last one that stayed in C1. We do not allow the leave of A1 until A1 receives either an explicit *LEAVE_SESSION* message from MH or a message from A2 requesting for hand-over of missing packets (this is *BIND_QUERY* message that will be described in Section 4.5).

4.4.2. Sender message stability. When a source of multicast is a mobile host, the provision of stable delivery of packets from the source to fixed or mobile hosts is much easier. Let the source mobile host MH send packets with sequence numbers from 1 to n to a session while it stays in C1 and then move from C1 to C2. Suppose that at that time A1 has received from MH the packets from 1 to m and has sent acknowledgments for the packets from l to l to MH ($l \leq m \leq n$). Suppose also that A1 has received acknowledgments from all its logical children for the packets from 1 to k ($k \leq l$). When MH moves into C2, it resumes emitting packets from sequence number $n + 1$. A2, knowing that the previous ARMM of MH was A1 and MH has received acknowledgments from A1 up to l by the binding procedure, sends a message containing that information to A1. Then A1 informs the session that all the acknowledgments for packets up to l should be fed back using the current logical tree and a new logical tree rooted at A2 should be constructed for the packets from $l + 1$. In the new logical tree, only the root is changed from A1 to A2 as described in Section 4.1.3. Since MH resumes transmitting packets from $n + 1$ in C2, A2 may not have the packets from $l + 1$

to n . Thus A1 should hand over the packets it has received but has not yet acknowledged (from $l + 1$ to m) to A2 via a binding procedure. A2 will receive the packets from $m + 1$ to n directly from MH. A1 should act as the root of the previous logical tree up to sequence number l and therefore is not allowed to leave the tree until the packets up to sequence number l are delivered stably even though no mobile host is attached to itself.

4.5. Binding Protocol

A binding between an ARMM and a mobile host means that the ARMM acts as a proxy in logical trees for the mobile host. A binding can be thought of as a function from the set of mobile hosts to the set of ARMMs. As it will become clear, our binding protocol is closely related to sender/receiver message stability.

When MH hands off from C1 to C2, it may know the fact in two ways. First, each ARMM periodically sends an advertise message telling its existence so that a mobile host may detect handoff upon receiving the message. Second, mobile IP software at a mobile host, knowing the hand-off via a well-defined method [12], signals the fact to reliable multicast software - typically implemented at transport or application layer - at the mobile host. When MH knows hand-off in either way and receives a multicast packet in the new cell for the first time, it sends A2 a *BIND_REQUEST* message containing information on its previous ARMM and some sequence numbers. The sequence numbers are ss , $rs(j)$, and $ns(j)$. ss is the maximum sequence number until which MH has sent successfully to A1 - that is, has received acknowledgments. $rs(j)$ is the maximum sequence number until which MH received successfully for each multicast source j . $ns(j)$ is the sequence number of the packet that MH firstly receives in C2. For the simplicity of description we consider MH as a sender and a receiver only for the source s in the following discussion. A2 sends A1 *BIND_QUERY* message containing $ns(s)$, $rs(s)$, ss and some more sequence numbers necessary for message stability as explained in the previous section. Receiving *BIND_QUERY*, A1 knows that it should behave as a source of a logical tree up to the sequence number ss . It also sends A2 *BIND_OK* message containing all the received packets emitted by MH from the sequence number $ss + 1$ to the maximum of retained packets at the time of hand-off and all the received packets emitted by the source s but missed by MH due to handoff. Receiving *BIND_OK*, A2 multicasts to the whole session *BIND* message telling that MH is engaged with itself from the sequence number $ss + 1$.

Upon receiving *BIND* messages, the members in the session behave as follows. If A2 joins a tree as a root for the first time, the members which were the children of A1 in the previous logical tree for the source s are attached to a new

logical tree as tentative children of A2. Other members create an entry for the new root A2 by copying the information on regional part of the previous tree. If A2 already participates in a logical tree because other source mobile hosts have previously existed in the cell, other members need only to add information that A2 will manage MH too. For both cases, all the member nodes in the new logical tree should send acknowledgments for packets from $ss + 1$ to their own parents again even though they have already sent the same acknowledgments along the old logical tree. This ensures sender message stability.

After sending *BIND* messages, A2 as a receiver either participates in a new logical tree as a child of the parent of A1 if A2 joins the tree for the first time, or adds MH to a list of the mobile hosts which it should manage in the existing logical tree if A2 has already participated in the logical tree. Since A1 hands over the packets that it has received but has not yet received acknowledgments for from MH, A2 can take over the management of MH without hurting receiver message stability.

Recall that even if A1 has no mobile hosts to manage after MH left the cell, A1 should not leave the session immediately. It should check if it has received the acknowledgments for the packets up to sequence number ss for sender message stability. Similarly for receiver message stability, it also should make sure that it has handed over all the packets that A2 misses. After the procedures for messages stability complete, A1 sends *UNBIND(A1, MH)* message to the session, leaving the session and the logical tree.

5. Performance evaluation

In this section we evaluate the performance of the proposed scheme by a comprehensive simulation and compare it with a static tree-based reliable multicast scheme (similar to RMTP [1]).

5.1. Simulation model

The simulation model consists of network model, host mobility model and multicast group model. Based on the models described in the following, multicast sessions are simulated using our event-driven simulation program [16].

Network model. For each simulation run, a sample network topology is generated using GT-ITM [17][18]. The network size grows up to L nodes (L is set to 100). Each node is a multicast router. Members in a multicast session attach themselves at each node (in real world member hosts reside in the subnet of the multicast router). More than one member, either fixed or mobile, can be placed. Mobile agents are assumed to be co-located statically at multicast routers.

We assume that the fixed part of networks do not change in routing. The link latency is assumed to be constant, whose value is obtained randomly from a uniform distribution $U[0.02, 0.03]$. Packet loss probabilities at each wired link are identical, and the value is randomly chosen from a uniform distribution $U[0.01, 0.1]$. If the packet loss probability at the link is p , each packet is lost at the link with the same probability p , without being affected by factors such as bursty loss. Given p , packet loss probability at each wireless link is set to wp (w is simply set to 2). The timer value for resending NACK is set close to the round-trip time between a requester and a retransmitter.

Multicast group model. For a given session, one node is selected as a source and programmed to multicast packets at a constant rate X (set to 10 packets per one time unit) during the lifetime of the session. Total N members are selected and placed randomly at nodes of the sample network. Among N members, K members are mobile ($0 \leq K \leq N$). Two kinds of multicast sessions are simulated at each run: one with a fixed source and the other with a mobile source. Mobile hosts, either a source or receivers, follow a host mobility model described below. We assume static memberships. Although our scheme allows for graceful join/leave procedure, we do not evaluate membership dynamics in this paper.

Host mobility model. A mobile host begins simulation in a home cell of the sample network and moves at random around to/from its adjacent cells. When the mobile host hands off, it is assumed to move to each adjacent cell equiprobably. Given c adjacent cells, p_d (probability of handoff to a adjacent cell) is $1/c$. Handoff time denoted by T_h is delay from the start of handoff of a mobile host to the end (T_h is set to 0.5). After handoff, the mobile host resides in a new cell during inter-handoff time T_{ih} . This is time taken for a mobile host to remain in its current cell until moving into another cell. It is exponentially distributed with the mean $1/\lambda_{ih}$. Therefore, how long the mobile host has resided in a cell without handoff in past has no influence on how soon it will leave the current cell (memory-less property).

5.2. Simulation results

The numbers in the legends in the figures below represent the scenario: 1 for one with a fixed source and 2 for one with a mobile source, respectively. M in the legends means that the value is for mobile nodes only. For description, we denote our scheme as ARMM and the static tree approach as NAIVE.

Interesting performance measures in our experiments are average implosion, average exposure, average delay, and average stability time. Average implosion counts how many times on average control packets for a data packet pass over

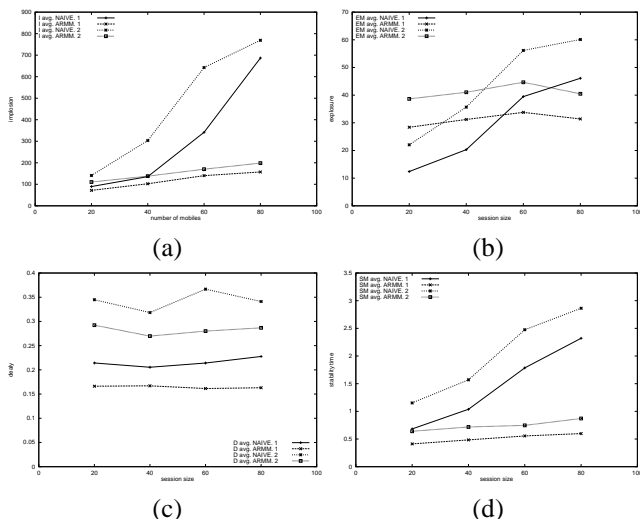


Figure 3. Scaling characteristics with session size: $p = 0.02$, $X=10$, $1/\lambda=50$, $K=N/2$.

the network links. Average exposure represents how many times on average data packets, both transmission and retransmission, pass over the network links. Average delay is the average of time taken for hosts to receive a packet. Average stability time is the average of time taken until all the members in the session successfully receive a packet.

Figure 3 illustrates how well ARMM and NAIVE scale with the number of session members. We increase the session size N up to 80, with the ratio of the number of mobile members to the session size fixed to $1/2$. Figure 3 (a) shows that average implosion increases linearly with session size in both schemes. However, the rate is substantially higher in NAIVE since much more unicast acknowledgments between fixed nodes and mobile nodes or among mobile nodes suffer from triangular routing. Figure 3 (b) illustrates the average exposure on wireless links. Exposure of ARMM remains only a little changed as the session size increases while that of NAIVE shows significant increase. This is because in ARMM, retransmission is not duplicated in a cell when more than two mobiles reside in the same cell. Note that in NAIVE, retransmission on wireless links is duplicated as much as the number of mobiles in the cell. Figure 3 (c) shows that the average of message delivery time has little correlation with the session size while Figure 3 (d) does that the average of stability time has linear relation with the session size. But the increase rate is quite different.

Figure 4 compares the scalability of ARMM and NAIVE with the number of mobile members. Figure 4 (a) shows that average implosion of NAIVE is much severer than that of ARMM in both scenarios. This is because as the number of mobiles increases, so does the number of acknowl-

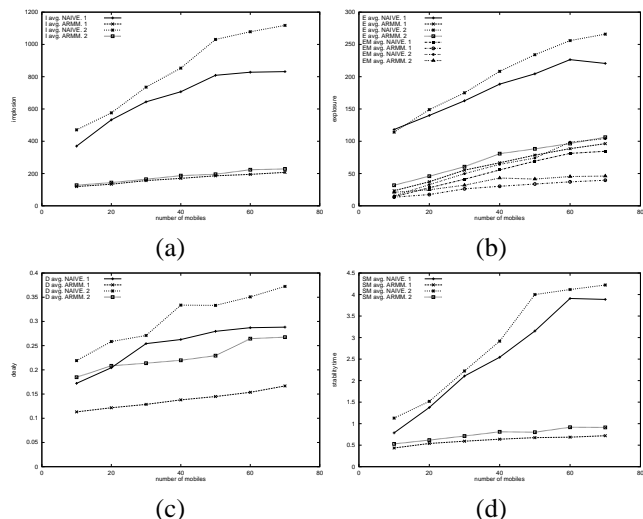


Figure 4. Scaling characteristics with number of mobiles: $p = 0.02$, $X=10$, $1/\lambda=50$, $N=80$.

gments to those mobiles which are sent over triangular routes. Average exposure in Figure 4 (b) can be explained in a similar manner. In Figure 4 (c), NAIVE has higher average delay (of both fixed and mobile hosts) again for the same reason. Average delay for mobiles remain almost unchanged as the number of mobile nodes increases. Figure 4 (d) shows average stability time for mobiles, which linearly increases in both cases.

Figure 5 shows the influence of X/λ_{ih} , the ratio of transmission rate to handoff arrival rate. The experiments were conducted by varying $1/\lambda_{ih}$. The higher value of X/λ_{ih} means that handoff occurs less frequently. The influence is noticeable only in the ranges of small mean inter-handoff time. This implies that inter-handoff time larger compared to transmission rate has little impact on protocol performance. Figure 6 shows the results in networks with different link loss probability. For loss-prone networks, average delay and average stability time are higher in both schemes and also in both scenarios.

6. Concluding remarks

Considering growing interests in mobile computing, multicast sessions involving not just fixed hosts but mobile hosts are anticipated to be prevalent in near future. Most mobile reliable multicast protocols deal with the reliable delivery of multicast packets to mobile hosts from agents while an existing reliable multicast protocol is used separately for fixed hosts. Due to host mobility, there may happen frequent joins and leaves of the agents. Overall performance of reliable multicast in a combined host environment

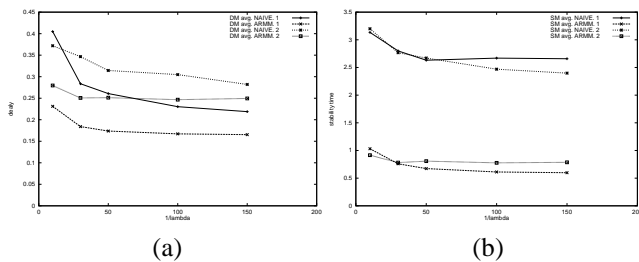


Figure 5. Impact of inter-handoff time: $p = 0.02$, $X=10$, $N = 80$, $K=40$.

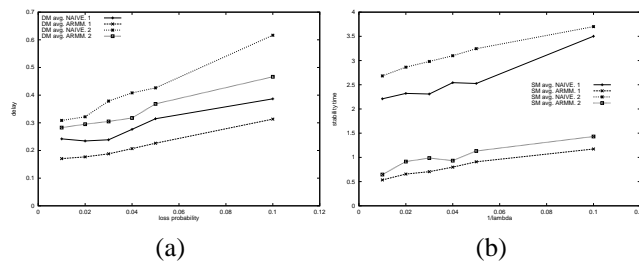


Figure 6. Impact of link loss probability: $X=10$, $1/\lambda=50$, $N = 80$, $K=40$.

depends on a reliable multicast protocol that can adapt to host mobility (i.e. dynamic membership change) as well in a scalable manner. We propose a scalable reliable multicast protocol that supports reliable multicast delivery in a combination of fixed and mobile hosts. The proposed scheme is based on our previous work that was designed for fixed hosts [3].

We demonstrate the usefulness of Join/Leave algorithm in that it can be commonly applied to host mobility and dynamic member join/leave. Similarity between them is fully exploited since host mobility could be mapped to join and/or leave of the corresponding ARMM in a logical tree. ARMM is a newly introduced entity to provide mobility transparency to fixed member hosts. It can be implemented as a server process running either at a mobile agent as in other contemporary approaches or at a dedicated server.

The simulation results verify that the proposed scheme is a viable solution. By constructing and reconstructing logical trees adaptive to host movements and also by eliminating triangular routing via ARMM, the proposed scheme achieves optimal routes for error recovery - optimal in terms of reduction of acknowledge implosion. Significantly low implosion, exposure and delay in the experiments demonstrate this. Abstraction of a group of mobile hosts into a single ARMM leads to efficient use of wireless bandwidth for retransmission and low processing burden on mobile hosts themselves. Future work includes the evaluation of efficiency in the buffer management for handoff.

References

- [1] J. C. Lin and S. Paul, "RMTP: A Reliable Multicast Transport Protocol," *IEEE INFOCOM '96*, pp.1414-1424, March 1996
- [2] R. Yavatkar, J. Griffioen, and M Sudan, "A Reliable Dissemination Protocol for Interactive Collaborative Applications," *ACM Multimedia '95*, pp. 333-344, November 1995
- [3] W. Yoon and D. Lee, "Adaptive Tree-based Recovery for Scalable Reliable Multicast," *IEEE ICCCN '99*, pp.126-131, October 1999
- [4] B. N. Levine, D. B. Lavo and J. Garcia-Luna-Aceves, "The Case for Reliable Concurrent Multicasting Using Shared Ack Trees," *ACM Multimedia '96*, pp. 365-376, November 1996
- [5] C. Papadopoulos, G. Parulkar, and G. Varghese, "An Error Control Scheme for Large-Scale Multicast Applications," *IEEE INFOCOM '98*, pp. 1188-1196, March 1998
- [6] S. Floyd, V. Jacobson, C. Liu, S. McCanne, and L. Zhang, "A Reliable Multicast Framework for Light-weight Sessions and Application Level Framing," *ACM SIGCOMM '95*, pp. 342-356, August 1995
- [7] K. Obraczka, "Multicast Transport Protocols: A Survey and Taxonomy," *IEEE Communications Magazine*, pp. 94-102, January 1998
- [8] A. Acharya and B. R. Badrinath, "Delivering Multicast Messages in Network with Mobile Hosts," *IEEE ICDCS '93*, pp. 292-299, 1993
- [9] A. Acharya and B. R. Badrinath, "A Framework for Delivering Multicast Messages in Networks with Mobile Hosts," *ACM/Baltzer Mobile Networks and Applications, 1*, pp. 199-219, 1996
- [10] K. Brown and S. Singh, "RelM: Reliable Multicast for Mobile Networks," *Journal of Computer Communications, 21(16)*, pp. 1379-1400, 1998
- [11] I. Nikolaidis and J. J. Harms, "A Logical Ring Reliable Multicast Protocol for Mobile Nodes," *IEEE ICNP '99*, pp. 106-113, 1999
- [12] C. Perkins, et. al., "IP Mobility Support," RFC2002, October 1996
- [13] V. Chikarmane, C. Williamson, R. Bunt and W. Markrell, "Multicast Support for Mobile Hosts Using Mobile IP: Design issues and Proposed Architecture," *ACM/Baltzer Mobile Networks and Applications, 3*, pp. 365-379, 1998
- [14] G. Xylomenos and G. Polzos, "IP Multicast for Mobile Hosts," *IEEE Communications, 35(1)*, pp. 54-58, 1997
- [15] S. Deering, "Multicast Routing in Datagram Internetworks and Extended LANs," *ACM Transactions on Computer Systems, 8(2)*, pp. 85-111, May 1990
- [16] W. Yoon, Multicast Simulator, <http://vega.icu.ac.kr/~wyyoon/reliable/sim/>
- [17] K. Calvert and E. Zegura, GT-ITM: Georgia Tech Internetwork Topology Models, <http://www.cc.gatech.edu/fac/Ellen.Zegura/graphs.html>
- [18] E. W. Zegura, K. Calvert and S. Bhattacharjee, "How to Model an Internetwork," *IEEE INFOCOM '96*, 1996