Mobile Networks and Applications 6, 361–376, 2001

AMTree: An Active Approach to Multicasting in Mobile Networks

KWAN-WU CHIN
Motorola Australia Research Center, Locked Bag 5028, Botany NSW 1455, Australia

MOHAN KUMAR
Department of Computer Science and Engineering, University of Texas at Arlington, Box 19015, Arlington, TX 76019-0015, USA

Abstract. Active networks (ANs) are a new paradigm in computer networking. In ANs, programs can be injected into routers and switches to extend the functionalities of the network. This allows programmers to enhance existing protocols and enables the rapid deployment of new protocols. Little work has been done in the area of multicast routing in heterogeneous environments. In this paper, we propose AMTree, an AN-based multicast tree that is bidirectional, optimizable on demand and adaptive to source migration. We show how ANs can be exploited to enable multicast tree to be modified and optimized efficiently. By filtering unnecessary signaling messages, maintaining minimal storage at routers and incorporating features from shared-tree methods we are able to achieve a scalable solution. Furthermore, we introduce an AN-based optimisation algorithm that is executed on demand by receivers. Besides that we introduce a fast rejoin protocol for receiver migration that makes no assumptions about the existence of multicast services in foreign networks. The performance of AMTree is compared to those of the bidirectional home agent (HA) method and the remote subscription method. We found that compared to the bidirectional HA method AMTree has a much lower handoff and end-to-end latency. Unlike the bidirectional HA method where end-to-end latency increases as the mobile host (MH) migrates further away from its HA, AMTree's latency remains fairly constant. We found that after optimization, the resulting tree's end-to-end latency to be comparable to the remote subscription method but without the need for building a new multicast tree after each handoff.

Keywords: multicast, mobile/wireless networks, active networks

1. Introduction

Multicasting is increasingly used by applications to dissiminate data across the network. Applications range from video conferencing to resource discovery. In the Internet, IP multicast [1] is used and the most popular implementation of IP multicast is based on the distance vector multicast routing protocol (DVMRP) [2]. Each group is identified by a group address [3] and members join and leave this group as they wish. Multicasting in heterogeneous environments is an area under investigation. Traditional multicast protocols [2,4–6] did not consider sources and receivers to be mobile. There has been little work [7–11] done in the area of multicasting specifically routing to MHs and current solutions suffer from limitations such as sub-optimal path and scalability issues.

ANs [12] are a new paradigm for solving network problems. This paradigm uses the computational power at intermediate network nodes to facilitate processing of traffic passing through. This work describes an AN-based approach called AMTree. An AMTree takes advantage of the processing capabilities at routers which enable MHs to continue sending packets to receivers after migration. Hence, the multicast tree can be maintained without much modifications and incurs minimal packet latency. This means low handoff latencies and minimal disruptions to packet flow. Furthermore due to intra-network processing, signaling overheads are low and tree is updated dynamically in an efficient way. AMTree uses the source-rooted tree (SRT) approach. The multicast tree incorporates the bidirectional state of core-based tree (CBT) [5] and is not dependent on the state maintained by the underlying routing protocol. Although we incorporate some features of CBT [5], no selection and management of cores are required. In AMTree, the term core refers to an active router (AR) that has a degree greater than one and its main functionality is to provide abstraction during handoff. In AMTree the designated cores to send packets to are dynamic and distributed. Hence, no traffic concentration is experienced. A core can be easily programmed with monitoring and traffic management protocols applied only to parts of the tree. Unlike DVMRP [2] where one separate tree is constructed for each source, in AMTree only one multicast tree is required for a given session. Furthermore, periodic messages are not required to acquire topology changes and routers do not need to maintain prune information. Senders that are not members of the multicast session can send packets to the tree although in our work we assume a primary source. AMTree encompasses the advantages of both source-rooted and shared tree with adaptation to host migration. Our work herein also can be applied to other work that uses the shortest-path tree[1] such as protocol independent multicast (PIM) (sparse-mode).[2] In PIM (sparse-mode) [4], receivers have the option to switch from an existing shared-tree to the shortest-path tree [4]. Therefore, if the

[1] Shortest-path tree and source rooted tree are used interchangeably.
[2] PIM is separated into dense and sparse mode.

Table 1
A brief comparison of AMTree with other multicast protocols.

Category	AMTree	DVMRP	MOSPF	CBT	PIM-dense	PIM-sparse
Tree type	SRT	SRT	SRT	Shared	Shared	SRT
Multiple sources	Yes	No	No	Yes	Yes	No
Number of trees in multiple sources	1	S	S	1	1	S
Router's state	$O(R)$	$O(S \times R)$	$O(S \times R)$	$O(R)$	$O(R)$	$O(S \times R)$
Use of core	Yes	No	No	Yes	Yes	No
Mobility support	Yes	No	No	Indirectly	Indirectly	No
Mobility effect to tree	Adapts	Invalidated	Invalidated	New connection	New connection	Invalidated
Core's adaptability to mobility	Yes	DNA	DNA	No	No	DNA
Core's position	Dynamic	DNA	DNA	Static	Static	DNA

source is mobile the problems discussed in the next section will need to be addressed. Apart from source migration, we also present a viable and simple receiver migration scheme. The migration scheme proposed works in foreign networks where no multicast service is present and does not suffer due to rejoining delay. Furthermore no packet loss and/or packet duplication are incurred.

A brief comparison of AMTree and existing multicast protocols is shown in table 1. In table 1, S and R refer to the number of sources and receivers, respectively, and DNA stands for does not apply. In table 1, we see that under the category *mobility support*, shared tree approaches support group members' mobility indirectly. When a source is mobile the source only has to form a new connection to the core. As far as the core is concerned the new connection formed is from a new source given that the mobile source has obtained a new care-of address. At the receiving end, receivers may experience delays due to the reconnection but the multicast tree formed is still valid. Hence, traffic can still flow to receivers whereas in approaches based on SRT traffic flow ceases.

1.1. The problems

Current multicast protocols such as DVMRP [2], MOSPF [6], CBT[5] and PIM [4] are designed with static host in mind and hence are prone to problems in mobile networks. Typical problems with multicasting in mobile networks are:

1. After migration, multicast protocols that are based on shortest path tree such as DVMRP may route packets incorrectly or drop packets due to reverse path forwarding.

2. In shared-tree approach such as PIM [4] or CBT [5], an algorithm is needed to determine core(s) or rendezvous point (RP) strategic location(s) in the network. This ensures that receivers obtain the least possible delay. The core's location is usually selected at the start of the multicast session. In mobile networks if sources/receivers are mobile then a core's or RP's position will be sub-optimal after each receiver/source migration. What is needed is an algorithm to relocate the core (dynamic core), but we conjecture that such algorithm(s) may prove to be infeasible due to high signaling messages involved and multiple constraints that have to be satisfied due to host movement.

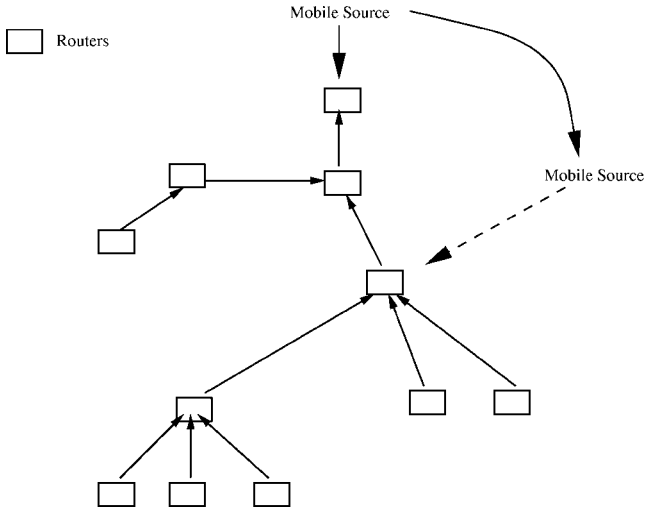

Figure 1. Handoff latency for varying receiver size.

3. At the receiving end, when a MH migrates to a cell with no other group members, it will experience delay. This is mainly caused by subscription delay, tree rebuild or due to non-existent multicast routers in the region.

4. The time to live (TTL) specified may be inappropriate. For example, a TTL set for one region maybe inappropriate for another. Once the MH migrates out of a region, the specified TTL value earlier maybe too small.

In this paper, AMTree tackles the first and third problems. Indirectly we show how the second problem can be avoided. As mentioned earlier existing multicast routing protocols assume a static source. In figure 1, we see a source-rooted tree constructed using reverse path forwarding. The root of the tree is a MH. The arrows show the direction in which the source's packet should arrive. When the MH migrates, any packets sent from the new care-of address will be discarded due to reverse path forwarding. This is because in reverse path forwarding the multicast router expects packets to come from an upstream interface. Since routers expect packets to arrive on the shortest path to the source, packets originating from the new location are dropped.

When a receiver migrates, three issues need to be considered. Firstly, the foreign network may not support multicast service. Therefore, the receiver is unable to rejoin the multicast session until it migrates to a network which supports

multicast. Secondly, the foreign network may support multicast service but does not join the multicast group(s) in which the visiting MH is subscribed to. Hence, the MH has to wait for Internet Group Management Protocol (IGMP) [3] next membership query cycle (at most one request per minute) and endure rejoining delay. Thirdly, in case where the foreign network has joined the multicast group, the receiver may receive duplicate packets or subsequent packets. Therefore, the receiver may experience duplicate packets or packet loss.

2. Background

Archarya et al. [7] extended Columbia's version of Mobile IP [13] where unicast tunnels are setup between mobile support routers. In other words, a virtual link layer connectiviity is established that enables DVMRP to work properly. This work was then extended in [8] where mechanisms for resolving duplicate packets and delivery of in-sequence packets were proposed.

The methods proposed in the IETF Mobile IP [14] specification are as follows:

- *Bidirectional HA tunneling*. The idea for a bidirectional tunnel between the MH and its HA was presented initially in [11]. The HA becomes the source/receiver of multicast traffic. Whenever the MH migrates to a new subnet a bidirectional tunnel is created from the MH's care-of address to the HA. As a result, any traffic generated by the MH or directed towards the MH has to traverse through the HA. The tunnel is then used to send packets going to/from the MH. From the receivers point of view the source never left its subnet at all. Obviously, this incurs a high handoff latency as the MH moves further away from the HA. Apart from that, this method introduces a problem called the tunnel convergence problem. This is where multiple MHs are serviced by one foreign agent (FA) and some/all MHs have different HAs. As a result, each HA will have a tunnel to the FA. To solve the convergence problem, Wang et al. [9] and Harrison et al. [10] have looked at ways of reducing tunneling cost. Harrison et al. [10] investigated different policies for choosing designated HA for multicast transmissions. Therefore, the convergence problem is solved by having only one bidirectional tunnel to the designated HA. Wang et al. [9] then extended the work in [10] where they used an architecture consisting of multicast agents (MAs). Basically their idea is to have MAs serving a unique set of FAs, therefore the HA was required to tunnel only to these MAs in the network. The difference between the work in [9] and in [10] is that, in [9] a tunnel is setup directly to the MA (thus reducing the length of the tunnel) and no algorithms are employed for choosing a designated HA.
- *Remote subscription*. A care-of address is allocated when the care-o30-331.2(addr7365.1gn)8it

2. *Capsules approach*. In this approach each packet carries a set of instructions which are executed at network elements. A comprehensive capsule based architecture was later presented by Wetherall et al. [22].

In [23], ARs are used to suppress duplicate NACKs and limit the delivery of repair packets to receivers experiencing loss. Furthermore, in this work multicast packets are cached on a best-effort basis for local retransmission. A similar approach was also taken in [24] where quality of service (QoS) filters and error control mechanisms are embedeed within active nodes. The active nodes in the architecture of [24] comprise different service modules (MPEG-1 filters) that deliver varying QoS streams depending on available bandwidth. In other words different branches of the multicast tree have different qualities of service. Lau [25] worked on an active receiver driven layered multicast protocol. The protocol does network filtering and achieves bandwidth convergence via the use of active routers. In the area of video delivery, Baldi et al. [26] designed a video conference system based on ANs. Their systems allow the cloning of video servers and customisation of packet delivery.

AMTree closely resembles the active multicast protocol proposed by Wetherall et al. in [22] where two capsules are defined: *subscribe* and *multicast data*. Receivers joining the multicast tree send a *subscribe* capsule to the sender. The program in the *subscribe* capsule installs (or refreshes) forwarding pointers in the cache. The *multicast data* capsule is used to carry data and a program for routing the carried data. At each router, data is routed dynamically based on the state information and program in the capsule. AMTree is similar in that we use the *subscribe* capsule to install state at routers but our data are routed by the routers themselves. Furthermore, our multicast tree is tailor-made for MHs.

To the best of our knowledge, no work has considered the use of ANs in adapting the multicast tree in the presence of mobility nor has there been an AN based protocol developed to address the problem of multicast routing. We intend to exploit the computing capabilities and state of routers to develop a solution that is adaptive to MH's characteristics. The information maintained and used will be presented in section 3.1.1.

The service provided by AMTree is unreliable, best-effort and connectionless delivery of packets. In other words, packets may be lost, duplicated, delayed or delivered out of sequence. This comforms to the IP service model where higher layer applications take responsibility of reliability.

In [7,15], it was mentioned that the update of multicast tree, specifically SRT, is computational expensive. We show that that a SRT can be updated efficiently. Further, previous work [10,15] suffers from tunnel convergence and algorithms are needed to overcome the problem. AMTree avoids the tunnel convergence problem because at any given time only one branch leads to a LAN with multiple MHs.

AMTree has a low handoff latency and disruptions to packet flow are minimal. This is made possible by using states and computation in the network which allows a MH to find a shortest path to the tree once it has migrated. As the tree undergoes minimal modifications the MH can continue to multicast after migration. Therefore, the update resulting from migration is kept low whereas in schemes such as *remote subscription* [14], the entire tree needs to be rebuilt to accomodate the MH's new location. In AMTree, the multicast tree remains intact and it is optimised dynamically. Furthermore, ARs filter unnecessary control messages and additional customisation of traffic can be added easily. AMTree incorporates the advantages of both source-rooted and shared tree. The end-to-end latency is kept low due to the tree being constructed as a source-rooted tree. It has the advantages of CBT such as bidirectional link and scalability. A detailed comparsion of AMTree, source rooted tree and shared tree will be presented in section 5.5. Besides that we show how receiver migration is achieved. The receiver migration model makes no assumptions about the existence of multicast service at the foreign network nor any group members already subscribed. Therefore, receivers are able to experience seamless handoff and do not suffer due to rejoin delay.

3. AMTree

The AMTree protocol allows for an efficient method for updating the multicast tree during migration by exploiting the characteristics of ANs. This enables senders to continue multicasting to a given session after handoff and does not incur any drastic changes in end-to-end latency.

The rationale of AMTree is to enable the adaptation of multicast tree during handoff. Current methods [7–11] are ineffecient, require significant computational overheads, or do not consider the source to be mobile. To update the tree from the edge of the network, the signaling overheads and the number of updates would prove to be inefficient. In AMTree, we provide an efficient solution that requires no modifications to the distribution tree after handoff. This enables a multicast source to be mobile while still being able to send data down the tree. Receivers have the option of optimising the tree where an alternative path to an optimal portion of the tree is found. Furthermore, no periodic control messages are used to catch topology changes. This is because receivers are required to join the tree explicitly, thus data only flows over links that lead from the source to receivers. In the case where the link to the parent AR is down, an optimization process is executed to graft to another portion of the tree. AMTree does not suffer from traffic bottlenecks encountered in PIM [4] and CBT [5] since the tree is basically a source rooted tree. ARs dynamically filter out unnecessary control messages. For example, join/optimization/NACK/ACK messages are filtered out by ARs nearest to the subscribing receiver.

The AMTree protocol is separated into three phases: (i) construction of active multicast tree, (ii) update process during migration and (iii) tree optimization by active nodes.

3.1. Building the multicast tree

We assume that a distributed location directory (LD) service exists in the AN which maintains the contact point of a given group, in this case the source. The distribution and access to the LD can be implemented similar to the domain name service (DNS). The construction of the multicast tree comprises three processes: join, leave and send.

3.1.1. Join process

As in the traditional IP multicast protocol a receiver indicates its interest in a multicast session to its local router.[4] If an active session already exists then no request is made to the LD. Otherwise the local router queries the LD for the contact point. The contact point in this case is the source address or the router local to the MH and it is updated whenever the source migrates. A JOIN_REQUEST message is then sent hop-by-hop towards the contact point. Once the JOIN_REQUEST message is intercepted by an AR that is subscribed to the session, a JOIN_ACK is sent back to the initiating router. This completes the join process. If a receiver is the first to be subscribed then the processing AR loads the AMTree program and creates a state pertaining to the session. The router then is subscribed to the session. Each router does not know how many receivers are subscribed to its child nodes. An AR only knows how many subscribers (end-hosts or ARs) are directly subscribed to it. Each router maintain a link to an upstream AR and a list of subscribers (which can be a neigbouring AR or end-hosts). The states created at each AR for each session are shown in table 2. The corresponding number of bytes for each data stored are shown in table 2. As we can see from table 2, we can easily associate properties to each subscriber which enable employment of service differentiation such as filtering and scoping to parts of the tree. Besides that each state (parent and subscribers) is allocated a TTL value. The TTL value is refreshed after data are forwarded or received. When the TTL expires, for example, when the link leading to a downstream AR is severed, the corresponding states are removed. Interested readers are referred to [27] for an analysis and design of active multicast services. Of importance are the states required to store receivers. For a given multicast session, the number of receiver states maintained at each AR is less than or equal to its degree (i.e. the number of neighbors an AR has).

In the multicast tree, ARs with at least one receiver are termed *core* routers. An example of a multicast tree with cores is shown in figure 2. The main function of these routers is to provide an abstraction during handoff. In other words, no modifications are required to routers that are downstream from the core router during handoff. Furthermore, the shortest path from each connected receiver to a core is maintained given that migration happens higher up in the tree. In addition, control programs to monitor and customise traffic at a given subtree can be loaded at the core.

[4] Assuming that this router supports the multicast protocol.

Table 2
Data structure maintained by each AR.

Data	Details
Upstream AR's IP	A link to the upstream AR is required to return any responses from receivers
Subsribers' IP	The downstream ARs/end-hosts to multicast packets to
Multicast address	The multicast address associated with the current session
Time to live	The maximum duration in which the state is maintained
Contact point	The root node of the multicast tree and is obtained from the LD or HOP_DISCOVERY message
Forwarding IP	This is used to tunnel packets to a recently migrated receiver

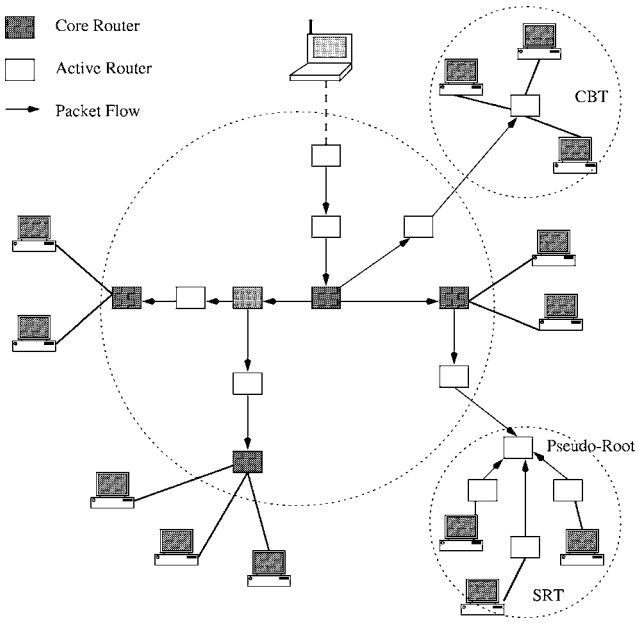

Figure 2. Example of a multicast tree building using the AMTree protocol.

Although the cores in AMTree are formed dynamically they can be used to interconnect different networks using different multicast protocols. For example, if a network uses CBT then the $core_{\mathrm{CBT}}$ (that belongs to CBT) becomes one of the cores in AMTree. An illustration of AMTree working with different multicast routing protocols is shown in figure 2. Note that for a network using SRT, the source in the network is a pseudo-root. As far as the receivers are concerned the source of the multicast is rooted at the pseudo-root. The only requirement is that $core_{\mathrm{CBT}}$ (pseudo-root) understands the control messages sent by the root. Also $core_{\mathrm{CBT}}$ is fixed unlike cores formed naturally in AMTree. Here we only point out the possibilities of AMTree interoperating with other multicast protocols. Since we are concerned with AMTree in this paper, we will present the interoperation of AMTree with other multicast protocols in our future work.

MH migrations are generally local [28]. This means MHs more likely to stay within a domain. which translates to movement at the upper part of the multicast tree. Given this

Table 3
Number of bytes maintained at each AR.

Data	Bytes
Upstream AR's IP	4
Subsribers' IP	Degree × 4
Multicast address	4
Time to live	(1 × Number of subscribers) + 1
Contact point	4
Forwarding IP	4

observation, modifications to the multicast tree happen more often at the upper part (i.e. near the source's subnetwork) than at the leaf nodes. For example, if we have a multicast tree spanning several clusters of networks, modifications to the tree are confined to the cluster in which the MH is located. Furthermore, these core routers are responsible for filtering out optimization messages if an optimization has been completed or the core router is currently undergoing optimization. Since the tree is a source-rooted tree, no traffic concentration occurs and cores in AMTree are distributed. This improves the scalability of AMTree.

As can be seen from figure 2, the cores within the multicast tree form a structure that is quite similar to a CBT. In contrast, the *cores* in AMTree are dynamically assigned and receivers do not join using cores as rendezvous points. The tree is built with each receiver having a shortest path to the source and subsequent optimizations are updated on a demand basis. After handoff, some receivers might experience lower latency while others might experience increased latency. For example, when a MH migrates to a domain containing leaf nodes, receivers in that cluster will experience lower delay while others may observe higher delays. Depending on the observed latency, a receiver can request for an optimization if the latency exceeds a threshold.

3.1.2. Leave process

Leave operation in AMTree is explicit. Each subscriber wishing to stop receiving packets[5] sends a LEAVE_MESG to its corresponding AR. As a result the subscriber will be pruned from that AR. Apart from that each subscriber maintained at each AR then checks whether it has other subscribers. If no subscribers are found, then a prune message is sent upstream. The upstream AR then removes the downstream AR from its list of subscribers. A check is then performed to determine whether there are any subscriber(s) left, if none then a prune message is generated. AR has an associated TTL value. The TTL value is refreshed after packets pertaining to the session are processed. When a TTL value for a given subscriber has expired, its entry is removed.

3.1.3. Send process

A source interested in sending to the multicast tree indicates its interest in creating/transmitting to a multicast tree through its local router. The local router than registers with the LD stating the current contact point for the session (i.e. specifying the root of the tree). The AR does not send any packets unless there is at least one receiver subscribed to the session. Once the AR has a subscribed member (either AR(s) or receiver(s)), it proceeds to multicast any packets sent by the source. When a packet arrives at an AR, the session details are accessed and a list of subscribers to the router is returned. The packet is then duplicated and forwarded to each subscriber.

In our discussion, we assumed one source per multicast tree although multiple sources are possible. This simplifies the optimization algorithm which will be presented in section 3.3. Due to the bidirectional nature of the state maintained at each AR, any non-member source that wishes to send packets to the session can do so. The storage space required at each AR is O(N), where N is the number of receivers subscribed to the session. This is similar to CBT [5] and PIM [4].

3.2. Handoff

The migration of the source entails two main processes: registration of the source's care-of address and connection to the nearest core. We assume the radio establishment between the base station (BS) and the MH is handled by the link- layer at the BS and MH. Also a mechanism such as dynamic host configuration protocol (DHCP) [29] is used to allocate a new care-of address to the MH. Once the connection to the BS is established, the handoff protocol executes the following:

1. A REG_COF is sent to the LD. This updates the source entry at the LD with the MH's current care-of address. If an AR that has subscribed to the session is encountered by the REG_COF message the intercepting AR sends a CORE-CONNECT to the MH's current location. The CORE-CONNECT message is used by the AMTree core discovery protocol to determine the nearest node/core to connect to after handoff. The details of the protocol will be explained in the next section.
2. A HO_UPDATE message containing the MH's care-of address is then sent to the previous contact point. A HOP_DISCOVERY with the MH's current care-of address is then generated and multicast along the tree (rooted at the old source). Cores on the tree receiving this message will then generate a CORE-CONNECT to the address specified in the HOP_DISCOVERY message. The HOP_DISCOVERY message continues to be forwarded along the tree until it reaches a leaf node where it is discarded.
3. A HO_COMPLETE message is returned to the MH once the local AR has processed the CORE-CONNECT message.

3.2.1. Core discovery

Core discovery is performed after handoff and is inititated by the core routers within the multicast tree. The discovery protocol uses the computation at each AR to determine

[5] Applies to receiver migration as well.

which core/router is nearest to the MH. The designated core is then used by the MH to unicast packets, the core in turn will multicast packets along the tree.

The CORE-CONNECT message contains four pieces of information: hop count, designated core, multicast address and MH's care-of address. The message (destined for the MH) is then processed, by each AR between the core and the MH. At each AR the hop count is compared with the observed hop-count (given that it has processed another CORE-CONNECT message) and incremented if required before being forwarded. If the message is the first to be processed, a new state is created and information in the HOP_DISCOVERY message is copied. Furthermore, if the AR is not part of the multicast tree, data shown in table 2 are created. Basically, the AR is now subscribed to the session. If an AR that has already subscribed to the tree is encountered and it is not a core router then the hop count is set to zero and the designated core field is set to the current AR's address. A release mesage is then sent back to the old designated core. Unlike shared-tree methods such as CBT [5] the MH does not need to connect to any of the predefined cores.

In figure 3, AR_1–AR_4 are core routers and A, B, C and D are routers. We can see that for core routers AR_1 and AR_2 hop counts to router A are different. Consider A and that a soft-state for AR_1 has already been created with a hop count of two. When the CORE-CONNECT message from AR_2 arrives at A, router A compares the hop count within the message with the hop count stored in its soft-state (i.e. the hop count for AR_1). Since the stored hop count for AR_1 is smaller, the CORE-CONNECT message from AR_2 is discarded. Then a RELEASE_MESG is sent to AR_2 and all soft-states allocated between routers A and AR_2 are freed. Note that the release message only affects state created by the CORE-CONNECT message. At D the CORE-CONNECT message from AR_1 has incurred a hop count of five. On the other hand, the message from AR_3 has only incurred a hop count of four. Therefore, D updates its state with AR_3 and sends a RELEASE_MESG to AR_1 which unsubscribes all routers along the path to AR_1. As a result, multicast packets are then sent down the path D–C–B to AR_3. Note that the multicast tree is augmented with routers D, C and B. Apart from that if a CORE-CONNECT message passes through a core router the message is dropped. Thus, unnecessary signaling messages are prevented from flooding the network during migration. For example, if C is a core router, clearly the shortest core to connect to is C, therefore, any subsequent CORE-CONNECT messages received by C are discarded and release messages sent. The pseudocode in algorithm 1 shows the core discovery process at the AR.

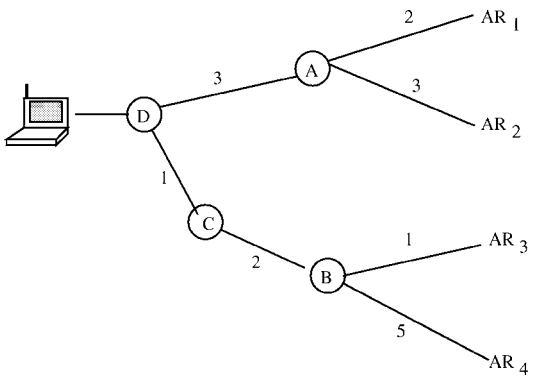

Figure 3. A network example showing hop count.

Algorithm 1 (Discovery process at AR).

```
IF we are subscribed but not a core THEN {
 Generate a new CORE-CONNECT message.
 Set hop to zero
 Set designated core to our address
 Discard old CORE-CONNECT message.
} ELSE IF no soft-state exists THEN {
   Create_State(hop, mcast addr, core addr)
   hop ++
   Forward CORE-CONNECT to MH's care-of-addr
  }
ELSE
{
 Get hop from CORE-CONNECT message
 IF hop < hop in soft-state THEN {
 Update_State(data from current message)
 hop ++
 Forward CORE-CONNECT to MH's care-of-addr
} ELSE {
   Discard message
   Send RELEASE_MESG to originating core
  }
 }
}
```

3.2.2. Packet delivery after handoff

Once the tree is augmented with the MH's new location, the designated core router has to multicast packets up the tree as well as down. If a non-member source sends packets to the session the algorithm presented in this section applies. The change in direction of packets after handoff is shown in figure 4. In figure 4, the routers leading up to the old contact point do not have any receivers subscribed, therefore, they are pruned from the tree. The algorithm to forward packets is as follows:

- Make a duplicate of the message and forward each message to downstream subscriber(s).
- If the source message did not come from an upstream AR then forward the message upstream given that the upstream AR did not prune itself from the session.
- The message is terminated once it reaches the old contact point.

3.3. Optimization

Two questions arise with respect to optimisation: *when should optimization be invoked*? and *how is optimization performed*? We will first address the first question. Basically, the optimization can be invoked when the receiver notices a significant increase in end-to-end latency.

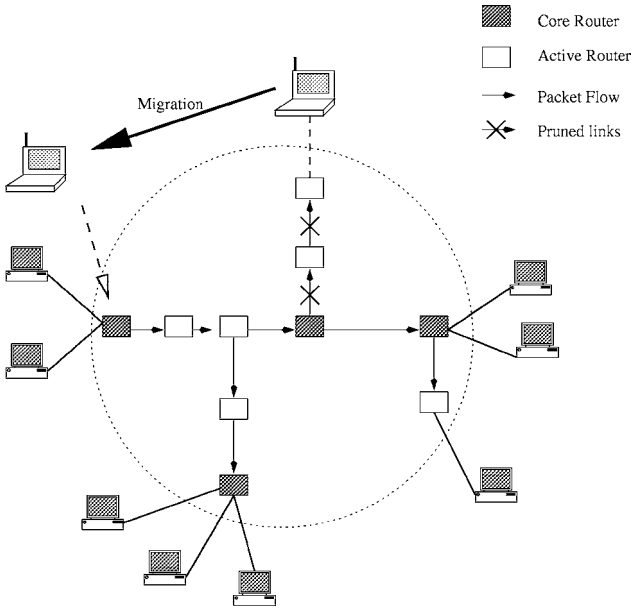

Figure 4. Example of packet flow after migration.

In the receiver's case, the optimization process is performed on a demand basis. This is due to some receivers having lower latencies after handoff or vice versa. Therefore, based on a given receiver's tolerance to delay, optimization might only happen periodically or randomly. Hence, the optimization process is performed when quality of service constraints are violated. Note that receivers are neither aware of source migration nor shorter paths. Without any probing of the network, the only indication (albeit not reliable) that a shorter path exists is when a significant increase in end-to-end delay is observed. Second, in the periodic invocation case, core ARs invoke the optimization process periodically to determine whether an alternative path exists. The downside is that optimization may be performed when no source migration has taken place, thus causing unnecessary signaling messages. Finally, the optimization process can be invoked explicitly. After source migration, a message is multicast to all cores to invoke optimization. Each core determines whether the parent node is valid, given the source's new location. If not then a new parent is found. The approach taken by us consists of on demand routing and explicit notification. This means that receivers invoke the optimization when quality of service constraints are violated. For example, when RSVP [30] is used, migration of source may cause the agreed upon contract to be violated. As mentioned, we have also adopted the use of explicit notification. For simplicity, the source invokes the optimization process explicitly when it migrates across LANs. This is because migration within a LAN will not cause significant increase in delay. Moreover, updating is only required at the local core, thus, subscribers downstream are not affected. Besides the above, the optimization message is also used when topology changes and the link to the parent AR is severed. Note that ARs themselves can choose to invoke the optimization when necessary. At the AR, when the TTL of the parent node expires and the multicast session has not ended yet,[6] the core AR then invokes the optimization process. As a result, given an alternative path, a new parent link is formed which links the core AR to a different part of the multicast tree. In the following paragraphs we describe the optimization process from a receiver's point of view. Note that similar processes can be invoked from a core AR as well.

The next question that needs to be addressed is: *how is optimization performed*? The optimization process entails sending an OP_DISCOVER message to the MH's care-of address. If a shorter path exists then optimization is possible. The multicast tree is then augmented and the original link is pruned. In our optimization algorithm, we assume there is only one primary data source. Since our multicast tree is bidirectional (i.e. the receivers are able to send feedbacks to the source) we do need to build a separate tree for each source. Hence, our tree has the benefits of CBT but no discovery is required to find the optimal core(s). Note that the OP_DISCOVER message is routed based on the recorded MH's care-of address at any given time. Therefore, if the source migrates during the optimization process, and the OP_DISCOVER message and ARs along the path do not have the updated MH's care-of address, then the message will be routed to the recorded care-of address. Therefore, the optimization process needs to be repeated for the given receiver(s). To prevent the network from being flooded with OP_DISCOVER messages, only one OP_DISCOVER message is sent for a given core. This is sufficient to ensure an optimal path for other receivers that directly subscribe to the core. For example, if an AR receives an optimization request from its downstream subscriber and an optimization has already been performed, the request is discarded. The pseudocode for the optimization process at a given router is shown in the pseudocode of algorithm 2.

Algorithm 2 (Optimization process at AR).

```
// Neighboring AR's address
Get AR_ADDRESS from OP_DISCOVER
IF NOT SUBSCRIBED TO SESSION THEN {
  Create new multicast session state
  Add AR_ADDRESS to subscriber list
  Set AR_ADDRESS to our IP
  Forward OP_DISCOVER to next hop
}
ELSE {
 IF NOT AR_ADDRESS in soft-state {
  Add AR_ADDRESS to subscriber list
  Send GRAFT_MESG to AR_ADDRESS
 }
 IF we have not optimized before THEN {
  Set AR_ADDRESS to our IP
  Forward OP_DISCOVER to next hop
 }
 ELSE discard message
}
```

[6] Given that the session does not end when link failure occurs.

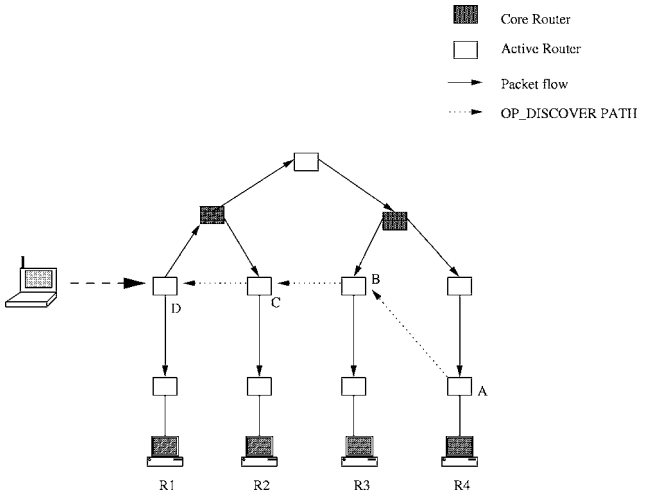

Figure 5. Example of an optimization being performed.

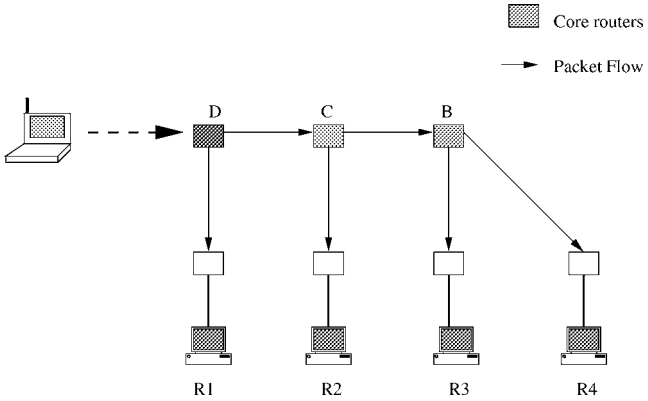

Figure 6. Resulting tree after optimization from R4.

When a receiver decides to perform an optimization, a request is made to the local AR. The local AR then transmits an OP_DISCOVER message with the MH's care-of address as its destination. In figure 5, we see that R4 has initiated an optimization. The OP_DISCOVER message is relayed through other parts of the tree.[7] At each AR, a check is performed to determine whether it is subscribed to the session and if so whether its downstream link is where the OP_DISCOVER message originates from. If the AR is subscribed to the session (e.g., the parent of R4 in figure 5) no computation is performed. Otherwise the session state in table 2 is created. The AR now is subscribed to the session. In figure 5, router *B* is subscribed to the session but it does not have router *A* as one of its downstream links. Clearly, an alternative path exists. As a result, router *B* creates a new state for router *A*. A GRAFT_ACK is then sent back to *A* indicating a successful graft and the old link can be pruned. When the OP_DISCOVER message reaches *C* where no entry exists for router *B* as before a new state is created and a GRAFT_MESG sent. In this case, the GRAFT_MESG message is sent to router *B*. This process is also repeated at router *D* and resulting tree obtained is shown in figure 6.

If a given router has already processed the OP_DISCOVER message, it assumes the route to the MH to be optimal. Therefore, any subsequent optimization requests by subscribers are dropped. If the subscriber requesting the optimization does not have an entry then an entry is created for the subscriber and GRAFT_ACK is sent. If we look at figure 5, we see that receivers R1–R3 do not require any optimization once the optimization from R4 has been performed. If an alternative path exists for R3 then the optimization process is carried out.

3.3.1. Pruning

Upon receipt of a GRAFT_ACK message, an AR transmits a PRUNE_MESG on the up link to its parent AR. At the parent AR the corresponding subscriber specified in the PRUNE_MESG is unsubscribed. If no subscriber(s) is left then the AR prunes itself from its parent AR. In the case where the prune message reaches an AR that has already pruned itself the message is discarded.

The PRUNE_MESG maybe lost because the service provided by the IP are unreliable. Therefore, upstream ARs continue to multicast packets even though a prune message has been sent. As a result, downstream ARs receive duplicate packets. Moreover, loops may form. To overcome this problem, when a packet arrives from the parent node in which a PRUNE_MESG has been sent, the packet is discarded and a PRUNE_MESG retransmitted. If the packet contains sequence number, then it is also used. For example, when an AR receives packets with similar sequence number from two or more neigbors. A copy of the packet is then multicast to all subscribers and a PRUNE_MESG sent out to all neighbors (except the parent AR) responsible for the duplicate packet.

3.4. Receiver migration

So far we discussed the source migration process. Now we will present a simple scheme for receiver migration that does not assume the existence of multicast service at the foreign network. The AMTree scheme minimizes rejoining delay, packet duplication and packet loss for receiver migrating as well. Note that the scheme proposed here is different to those of reliable multicast where the aim is to ensure that a given packet is received by all group members. Here the aim is to enable seamless handoff. From the receiver's view, during handoff it remains subscribed and causes little disruptions to traffic flow.

To ensure that the receiver continously receives multicast data, a forwarding pointer is used. This means that after handoff, a receiver sends a notification to the previous BS while waiting for a rejoined notification at the current network. Once the receiver rejoined has successfully rejoin the multicast session the forwarding pointer is terminated. The advantages include:

[7] Here we assumed the underlying routing protocol will route the message through the shortest path. If an alternative path exists we assume it is optimal.

- Receivers are able to receive multicast traffic in networks that do not support multicast service.
- A fast rejoin is possible since there is no need to wait for IGMP query request.
- Receivers are not susceptible to lengthy rejoin process in networks where no group membership exists.

3.4.1. Handoff

Figure 7 gives an illustration of the processes involved during receiver migration. When MH_1 migrates to FA_2 a check is performed to determine whether the current network has any members subscribed to the multicast session. If so, the receiver notifies the local multicast agent (FA or a multicast router) to update its multicast table. Otherwise, a binding update is sent to FA_1. The receiver also makes a request to rejoin the multicast session. Once FA_2 has intercepted the binding update message, it is considered subscribed to the session, therefore any subsequent receivers of the same group migrating into the current network are not required to send a binding update. As a result, at any given time only one forwarding path exists, thereby avoiding the tunnel convergence problem. After processing the binding update message FA_2 forwards the message to FA_1.

The binding update contains the receiver's new care-of address. If an AR that is subscribed to the multicast session intercepts the binding update message (as the case in figure 7), the AR (C_1) updates its *Forwarding IP* state to the care-of address stated in the binding update message. In this case, the binding update is not forwarded to the FA_1 but any multicast packets received by the AR will be tunelled to the MH until a STOP_FORWARD message is received. The STOP_FORWARD message is sent by FA_2 once the rejoin process is completed.

The receiver is not considered to be part of the multicast session until its rejoin request has been acknowledged. Since the destination address of multicast packets contain class D addresses, the packet has to be tunelled to the receiver. Therefore, the receiver needs to perform a check (*Forwarding IP* field) whether forwarding is required, if so a copy of the multicast packet is encapsulated and forwarded

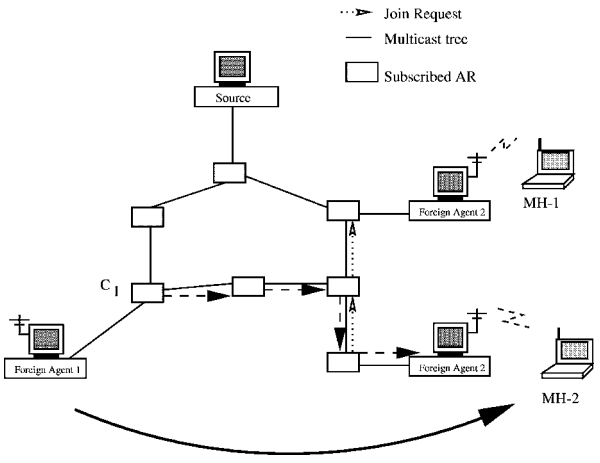

Figure 7. Illustration of receiver migration and rejoining multicast session.

to the care-of address stored. This means if the *Forwarding IP* field is set, then forwarding is required.

From figure 7, we can also see a join message heading up to the source. The join process is similar to that described in section 3.1.1. Once the receiver has rejoined the multicast session, a STOP_FORWARD message is sent to FA_1. Upon receiving the STOP_FORWARD message, FA_1 or an AR removes the receiver from its *Forwarding IP* list.

Since the receiver does not need to wait for IGMP query or delay incurred by the rejoining process, a fast handoff is achieved. In networks, where no multicast service is available, the proposed handoff method is not affected since it is not a requirement that multicast service exists. The main advantage offered by AMTree is that while receiving packets from the previous BS, multicast service can be installed. Recall that we are able to augment services provided by ARs and since when a join is performed AMTree is loaded into AR. As a result, subsequent receivers migrating into the network can utilise multicast service. Unlike approaches such as [10] where a HA handoff is required, in AMTree the branch created by the first receiver persists until all group members have left the network. Since at any given time only one branch leads to a given network and the branch persists until all receivers leave, the problems associated with tunnel convergence are avoided.

4. Simulation model

Our simulation studies were done using Opnet [31] and the following investigations are carried out:

- A comparison between AMTree and bidirectional HA method. We looked at the resulting end-to-end latency incurred before/after handoff and after optimisation. Furthermore, the handoff latency incurred for both schemes are investigated. In AMTree the handoff latency is calculated from the time the MH establishes a link with the BS to receiving a HO_COMPLETE message from the local AR. Here we looked at how the latency varies with different receiver sizes.
- We determine the increase in end-to-end latency after handoff due to the resulting non-optimal tree. This is to show that while no immediate modifications are done to the tree the end-to-end latency is tolerable.
- We looked at the effects of receiver sizes on the number of cores formed within a multicast tree. This is significant in that the number of cores represent the varying degree of abstraction achieved.

Our network is modeled after traditional networks with several enhancements. These enhancements allow customised programs to be injected into routers. Here we take a programmable approach where a multicast join message loads the AMTree algorithm into routers. The join message issued by receivers are active packets in that they hold a flag which loads the appropriate active program in each

router it traverses through. Once a program is loaded we assume the router will pass all packets related to the multicast session to the AMTree program accordingly. Furthermore, the router provides predefined interfaces for accessing routing information, creating soft-state and forwarding of capsule/packets. As in traditional network models our network provides "best-effort" packet delivery in that the underlying network is unreliable where packets can be lost, duplicated or delayed. Hence, multicast applications are needed to provide reliable transport of data packets. We assume that for each subnet connected to the network there exists a base station (BS) which enables MHs to connect to the network. MH migrations in our model are random in that they can migrate locally or globally.

The receivers are considered static in our model and each receiver that is interested in joining a multicast session contacts its local router. IGMP [3] is used to relay membership information. RIP [32] is used to route control messages such as join/leave and we assume that given a destination address will route packets on the shortest path. The multicast tree built is symmetric in that each router only knows who its upstream and downstream routers are. Therefore, upstream and downstream packets will follow the same route.

The simulation was done on a 50 node mesh topology with degree ranging from three to six. The data rate of interconnecting links between ARs is set to 10 Mb/s. The wireless links have a data rate of 2 Mb/s. Bit errors at the wireless link are assumed to be handled by a data link layer protocol. Each FA (or BS) is assumed to manage a cell and overlapping of cells means there are no *silent* areas. The MH controls the time of handoff, meaning the time of handoff and the duration of the handoff procedure can be measured. The MH migrates randomly to any of the receiver's subnet. Hence, the migrations are not necessarily local. The BSs broadcast advertisement messages at intervals of one second. The advertisements are needed to enable MHs to detect that they are in the cell area of a given BS.

The only traffic in the network is that solely of the multicast session and the AMTree protocol. At any moment in time there is only one ongoing session. This is in accordance with our design where a specific tree is built for a unique session. The packets generated by the source are of size 1024 bytes at a rate of one packet per second.

At the start of a simulation each receiver is randomly connected to the topology. We studied the performance of AMTree on varying number of receivers, ranging from 10 to 50. This serves to simulate low to high density multicast tree. The AMTree algorithm is loaded into ARs at the start of simulation. Ten simulation runs were performed for each case and the results averaged.

5. Results

5.1. Handoff latency

The handoff latencies for different number of receivers are shown in figure 8. The latencies range from 4.4 to 4.8 ms.

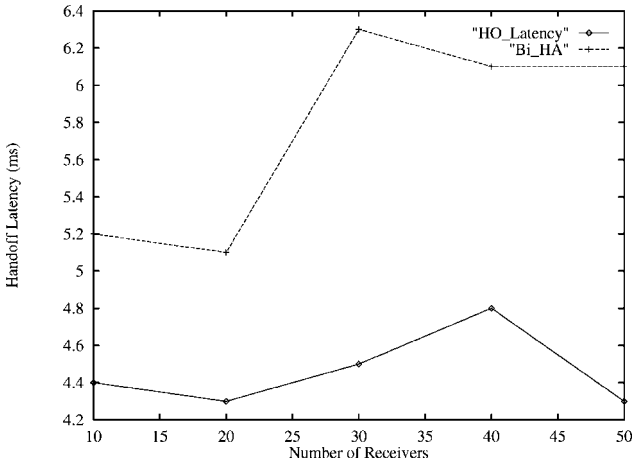

Figure 8. Handoff latency for varying receiver size.

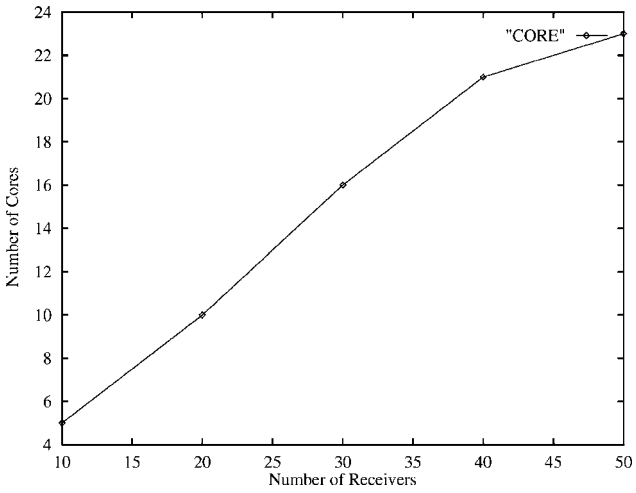

Figure 9. Number of cores versus number of receivers.

As can be seen the handoff latency achieved by AMTree is much lower than that of bidirectional HA method [14]. In the bidirectional HA method the increase in latency when the number of receivers reaches 30 is mainly due to the increase in traffic. The latencies measured in AMtree are for cases where the full core discovery protocol is employed rather than of connection to a core encountered during registration of care-of address (i.e. core encountered from REG_COF message). A significantly lower handoff latency will be observed in highly dense groups due to the higher probability of encountering a core during registration. A significant feature of our work is that the discovery protocol does not stop in finding a core. The discovery protocol is executed further to determine the shortest path possible to a core.

5.2. Number of cores

The number of cores discovered for different number of receiver sizes are shown in figure 9. The higher the number of cores the higher the degree of abstraction. This means that less modifications are required, which leaves a large proportion of the tree unaffected after handoff. Although from or

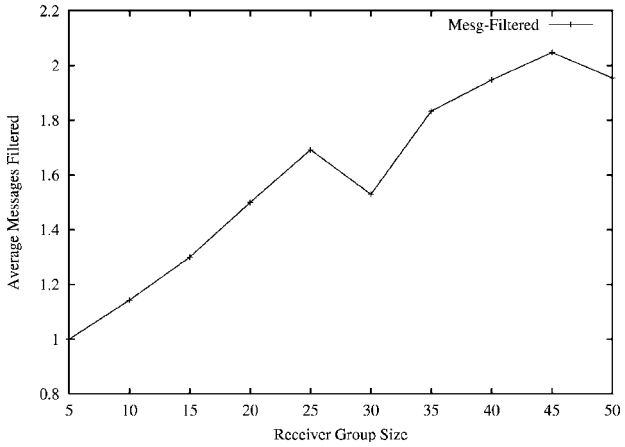

Figure 10. The corresponding number of messages filtered.

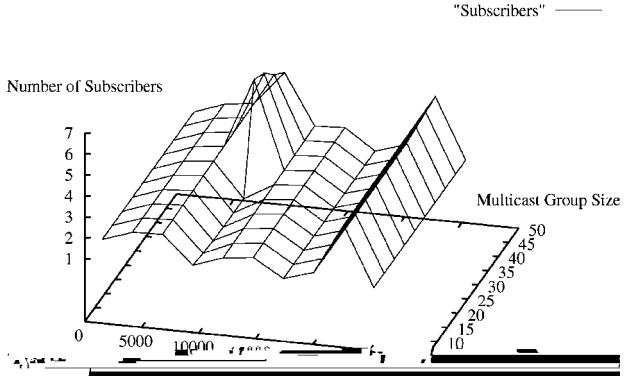

Figure 11. Number of subscribers maintain by each core with respect to multicast group size.

figure 9 we see that the number of cores is propotional to the number of receivers, the increase in cores does not translate to a higher consumption of bandwidth due to core discovery and/or optimization. On the contrary, it increases the number of messages filtered out thereby conserving the overall bandwidth. We can see from figure 10 that as the group size increases, the number of messages filtered increases. By observing figures 9 and 10, we see that when the multicast group size is at 45, there are 22 cores and on average two messages being filtered. That translates to 44 messages filtered in the entire network.

In figure 11 the number of subscribers (which include routers and end-hosts) maintained by each core router is shown. The storage requirements for a given session is distributed across 20 cores. Note that ARs are also classified as subscribers. From our observation of the state maintained at ARs, we find that as the multicast group size increases, ARs at the edge of the network maintain a higher number of subscribers. This fact is only true in dense mode situation where most ARs will have at least one subscriber downstream and subscribers will join the multicast session at the AR that the *join message* first encountered.

5.3. End-to-end latency

We compared AMTree's end-to-end latency to the bidirectional HA method specified in the Mobile IP's specification [14]. Figure 12 shows the latency incurred by each receiver (40 receivers scenario) in the simulation. The receivers identification indicates the latency observed for receiver n.

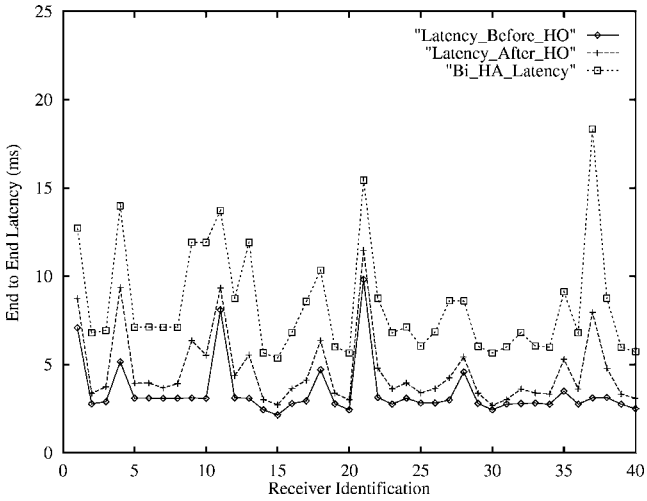

Figure 12. Comparison between AMTree and bidirectional HA method. The number of receivers in this 50 node topology is 40.

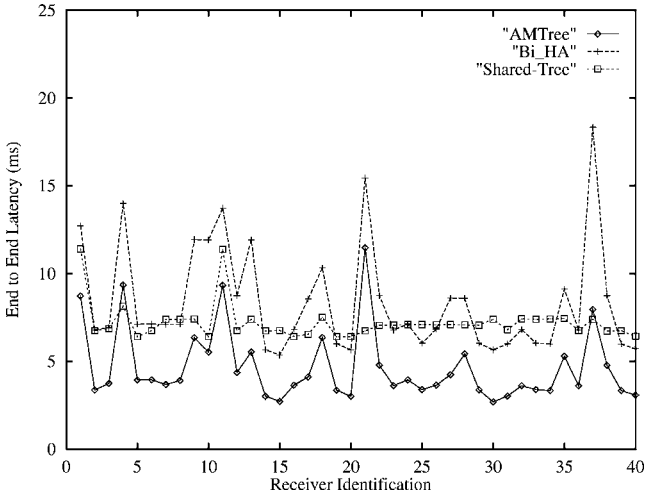

Figure 13. This graph compares the end-to-end latency of AMTree (no optimization) to shared tree approach. The number of receivers is 40 in a network topology with 50 nodes.

In figure 12, we see that the bidirectional method incurs a high latency due to triangle routing. On the other hand, with AMTree the latency after handoff registered a slight increase. This increase is partly due to an increase in hop count to the migrated subnet. For example, when the MH is at subnet 1, receiver i might have a latency of 4 ms. After the MH migrates to subnet 2, the best possible latency obtainable is 4.5 ms. Therefore, a slight increase in end-to-end latency is observed. This slight increase is unavoidable and video applications need to adapt to this slight increase in delay.

Apart from that we have also compared AMTree to shared-tree approach. Figure 13 shows a comparison between AMTree, Bi-HA and shared-tree. We can see that shared-tree generally has a stable end-to-end latency given

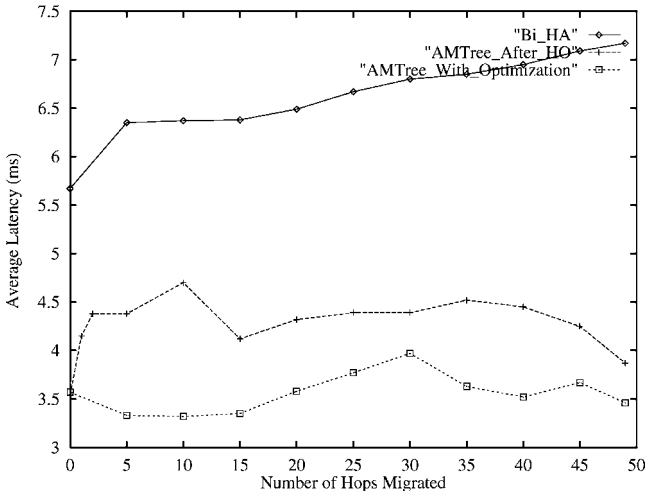

Figure 14. This graph shows the latency incurred when the MH migrates increasingly further away from its previous position.

that the RP is positioned in a strategic location. AMTree still outperforms shared-tree on most occasions in terms of end-to-end latency. Also for shared-tree we see that the end-to-end latency observed does not vary as much as Bi-HA or AMTree. This is due to the position of the core/RP. Since we manually positioned the core/RP in the centre of the topology, all BSs have approximately equal distance to the core. Hence, as the source migrates to each BS, the distance to the core will be approximately equal to that of its previous BS. Therefore, the delay experienced by the shared-tree approach in our experiment represents a "perfect" situation where end-to-end delay is not affected considerably by source migration. When receivers/sources migrate further away from the core, the performance of the shared-tree is similar to that of the Bi-HA method.

5.3.1. Hop count

We also analysed the performance of AMTree in cases where the MH migrates with an increasing number of hops. From figure 14 we see that the delay after optimization remains fairly constant and is significantly lower than the bidirectional HA method and also AMTree with no optimization. As the MH moves further away from its HA we see that for the bidirectional HA scheme, the end-to-end latency increases rapidly due to triangle routing. On the other hand, in our AMTree protocol the increase in latency is restricted becase the probability of encountering ARs is high with increase in the number of receivers. This is because we take advantage of the geographical placement of ARs in the tree and connect to the closest one. Therefore, as the number of receivers increases the probability of encountering an AR is high.

5.3.2. Optimization

The optimization metrics we measured are the end-to-end latency and hop count. We show results for a network with 40 receivers[8] shown in figure 15. As can be see from figures 15(a) and (b), a significant reduction in number of hops is observed. Hence, a lower end-to-end latency is observed. Note that in figure 15 some receivers do not gain from the optimization process and may suffer from a slight increase in latency. Certain cores sustain an increased number of receivers thereby increasing the processing time after optimization. Note that in our study the receivers join the multicast tree at the start and they are static throughout the simulations. If receivers are dynamic and some receivers join after source migration the path from those receivers to the source would be optimal.

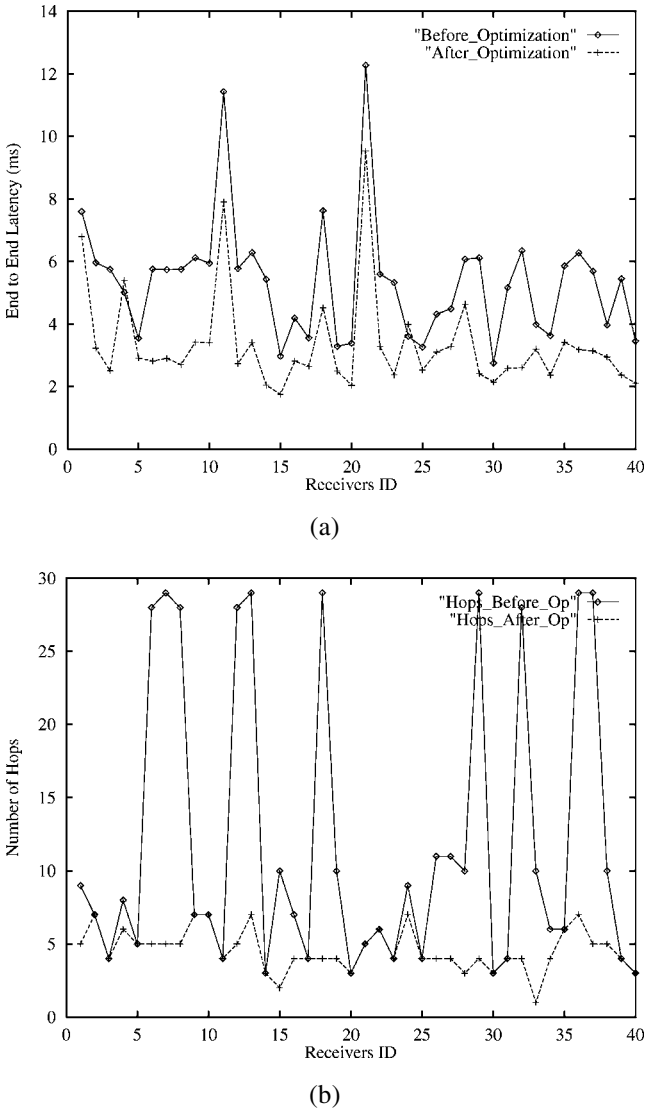

Figure 15. Hops and end-to-end latency after optimization on a 50 node mesh topology with 40 receivers.

We also investigated our optimization algorithm in the event of remote subscription when the entire multicast tree is rebuilt to accomodate the MH's new location. Hence, the latencies incurred should be the best achievable for the given MH's location. The latency incurred for both algorithms is shown in figure 16. In figure 16, we see that some receivers achieve better latency compared to remote subscription. Therefore, at best the AMTree's end-to-end latency is

[8] We have done tests for receivers ranging from 10 to 50.

lower than remote subscription.

5.4. Tree efficiency

Table 4 shows the tree efficiency achieved by AMTree. Essentially the results in table 4 justify our arguments that most of the tree can remain unmodified. Due to the fact that ARs only maintain the parent node, any changes due to source migration will only update the given state leading to the parent node. Downstream receivers are unmodified. As can be seen in table 4, the group density increases the efficiency. This means that for a receiver size of 40 there is only 4.4% difference between our multicast tree and Total-Rebuild. Note that this high efficiency is due to increasing number of cores which modify the tree structure without requiring changes to downtream subscribers. Also note that due to the increased number of receivers, most subscribers will be concentrated at edge routers, therefore, for most cases these receivers are unaffected.

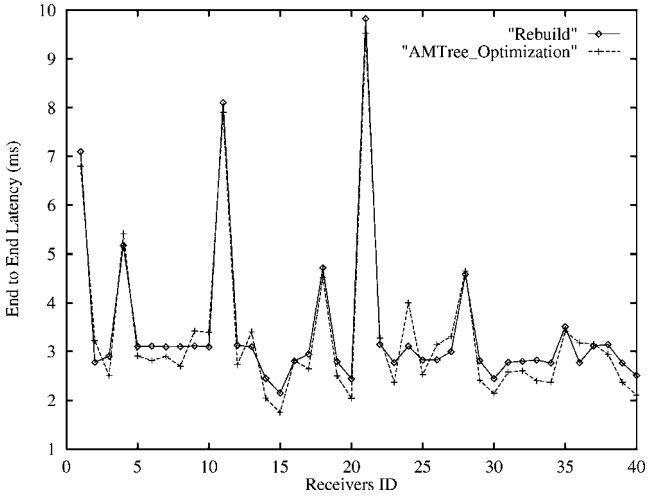

Figure 16. This graph compares the latency of AMTree's optimization to rebuilding a new multicast tree.

Table 4
Efficiency of the AMTree.

Number of receivers	Efficiency
10	73.5
20	68.75
30	76.4
40	94.6
50	97.3

5.5. Qualitative comparison with other approaches

In table 5, we compare the features of AMTree with other mobile multicast approaches. The category *Join and graft delays* refers to the time it takes for receivers to rejoin the multicast session. For Bi-HA and MoM the join delay is dependent on the receiver's distance from its HA. As for RBMoM the join delay is range dependent. As mentioned in [15], the worst case and ideal performance of their scheme is that of Bi-HA and Rebuild, respectively. In AMTree, the join latency is dependent on the distance to the previous BS. Since migration is generally local, the rejoin process incurs minimal delay. Note that in AMTree the second rejoin process was not taken into account because it does not affect the disruptions observed by the receiver. In table 5, we also looked at transparency. Transparency here means whether subscribers experience any disruptions due to mobility (source or receiver). Only Rebuild and Shared-Tree do not provide transparency. We have included Shared-Tree in table 5 because it supports mobility indirectly and no modifications to the current protocols are required. The problems that arise with CBT and PIM are: (i) receivers suffer from high disruptions delay, (ii) high packet loss during handoff, and (iii) possibly increased end-to-end delay. Further, packet duplication may occur when receivers rejoin the multicast session after migration. Despite these problems, packets are able to reach multicast group members after handoff unlike SRT approaches.

With regard to tunneling, note that unlike Bi-HA, AMTree requires minimal tunneling because tunneling is only used during receiver migration. Since tunneling is only used when there are no members at a foreign network and cease to exist after the receiver has rejoined the multicast session, the tunnel convergence problem is avoided.

As for seamless receiver migration, current multicast protocols do not ensure smooth handoff and rejoin. Bi-HA and MoM suffer from rejoin latency which is dependent on the HA's distance from the receiver's location. In Rebuild and Shared-Tree, the existence of multicast services is assumed and considerable delay is incurred if the foreign network does not have group members. RBMoM's rejoin latency is range dependent. Depending on the range, at worst its performance may be similar to Bi-HA and at best similar to Rebuild.

Table 5
A comparison between AMTree and other mobile multicast approaches.

Category	Rebuild	Bi-HA	MoM	RBMoM	Shared-Tree	AMTree
Optimal routing	Yes	No	No	Range dependent	No	Yes
Tunnell convergence	No	Yes	Yes	Yes	Yes	No
Transparency	No	Yes	Yes	Yes	No	Yes
Tuneling	No	Yes	Yes	Yes	No	Minimal
Join & graft delays	Minimal	High	High	Range dependent	Moderate	Minimal
Seamless receiver migration	No	No	No	Range dependent	No	Yes
Agents involved in routing	FA	HA, FA	HA, FA	HA, FA, MHA	FA	FA

6. Conclusion and future work

We have presented an active multicasting protocol that enables the adaption of the multicast distribution tree during handoff. AMTree enables the multicast tree to remain intact after handoff. The end-to-end latency incurred after migration is shown to be minimal and significanly lower than that of bidirectional HA and comparable to that of remote subscription after the AMTree optimization. The handoff latency is shown to scale well as the number of receivers increase. The handoff latency will remain fairly constant as the number of receivers increases due to higher probability of finding an AR that is subscribed to the session. In the bidirectional HA method the handoff latency is proportional to the round trip time from the MH's current location to its HA. We have also shown that AMTree can be easily optimised and end-to-end latency achieved is comparable to rebuilding the tree from the new location. Unlike remote subscription no new tree is constructed at the MH's new location. Apart from that we present a scalable solution in that the storage requirement is kept at a minimum and ARs take an active role in reducing unnecesary control packets. We have shown the viability of ANs in solving the problem of multicasting to MHs. Furthermore, we demonstrate how the advantages of source-rooted tree and shared-tree can be incorporated to handle MHs. The functionalities embedded in ARs can be extended easily to include bandwidth adaptation programs if any of its subscribers are mobile. We have also shown how receiver migration is performed. The algorithm is simple, fast and does not assume the existence of multicast service at foreign networks.

An intermediate follow up of AMTree is to investigate the incorporation of multiple sources. The multicast aspects of having multiple sources are trivial but the challenge is in the optimization process. Also we will be looking into the application of ANs in alleviating the problem of scoping, quality of service and security. Besides that we are looking at the possibilities of applying features of AMTree in ad-hoc networks.

References

[1] S.E. Deering and D.R. Cheriton, Multicast routing in datagram internetworks and extended lans, ACM Transactions on Computer Systems 7 (May 1990) 85–110.

[2] D. Waitzman, C. Patridge and S. Deering, Distance vector multicast routing. RFC1075 (November 1988).

[3] S. Deering, Host extension for IP multicasting, RFC 1112 (August 1989).

[4] S. Deering, D. Estrin, D. Farinacci, V. Jacobson, C.-G. Liu and L. Wei, An architecture for wide-area multicast routing, in: *SIGCOMM'94* (August 1994) pp. 126–135.

[5] A. Ballardie, P. Francis and J. Crowcroft, Core Based Trees (CBT): An architecture for scalable inter-domain multicast routing, in: *SIGCOMM'93*, San Francisco, CA (1993) pp. 85–95.

[6] J. Moy, Extension to OSPF, Internet draft 1584 (1994).

[7] A. Acharya, A. Bakre and B.R. Badrinath, IP multicast extensions for mobile internetworking, in: *INFOCOM'96*, San Francisco, CA (1996).

[8] A. Acharya and B.R. Badrinath, A framework for delivering multicast messages in networks with mobile hosts, Wireless Networks (1997) (to appear).

[9] Y. Wang and W. Chen, Supporting IP multicast for mobile hosts, Mobile Networks and Applications 6 (2001) 57–66.

[10] T.G. Harrison, C.L. Williamson, W.L. Mackrell and R.B. Bunt, Mobile multicast (mom) protocol: Multicast support for mobile hosts, in: *ACM International Conference on Mobile Computing and Networking (Mobicom)* (September 1997).

[11] V. Chikarmane, R. Bunt and C. Williamson, Mobile IP-based multicast as a service for mobile hosts, in: *Proceedings of the 2nd IEEE International Conference on Services in Distributed and Networks Environments* (1995).

[12] D.L. Tennenhouse and D.J. Wetherall, Towards an active network architecture, Computer Communication Review 26 (April 1996) 5–18.

[13] J. Ioannidis, D. Duchamp and G.Q.M. Jr, IP-Based protocols for mobile internetworking, in: *ACM SIGCOMM'91* (September 1991) pp. 235–245.

[14] C. Perkins, IP mobility support, RFC2002 (October 1996).

[15] C.R. Lin and K.-M. Wang, Mobile multicast support in IP networks, in: *INFOCOM'2000* (2000).

[16] T.-E. Kim and V. Bharghavan, A multicast routing algorithm for mobile computing environments, in: *Proceedings of IEEE Wireless Communications and Networking Conference* (September 1999).

[17] D.L. Tennenhouse, J.M. Smith, W.D. Sincoskie, D.J. Wetherall and G.J. Minden, A survey of active network research, IEEE Communications Magazine 35 (January 1997) pp. 80–86.

[18] D.L. Tennenhouse, S.J. Garland, L. Shrira and M. Kaashoek, From Internet to activenet (1996) `http://www.tns.lcs.mit.edu/publications/rfc96/rfc96.html`.

[19] J.M. Smith, D.J. Farber, C.A. Gunter and S.M. Nettles, Switchware: Accelerating network evolution, Technical report, MS-CIS-96-38, University of Pennsylvania (1996).

[20] D.S. Alexander, M. Shaw, S.M. Nettles and J.M. Smith, Active bridging, in: *Proceedings of SIGCOMM '97* (1997).

[21] S. Bhattacharjee, K. L. Calvert and E. Zegura, An architecture for active networking, in: *High Performance Networking (HPN'97)* (April 1997).

[22] D.J. Wetherall, J. Guttag and D.L. Tennenhouse, ANTS: A toolkit for building and dynamically deploying network protocols, in: *IEEE OPENARCH'98*, San Francisco, CA (April 1998).

[23] L.-W. Lehman, S.J. Garland and D.L. Tennenhouse, Active reliable multicast, in: *IEEE INFOCOM'98*, San Francisco, CA (1998).

[24] R. Wittmann and M. Zitterbart, Amnet: Active multicasting network, in: *Proceedings of International Conference on Communications (ICC'98)* (1998).

[25] W. Lau, On active networks and the receiver heterogeneity problem in multicast session, Honours Thesis, Curtin University of Technology, Western Australia (1998).

[26] M. Baldi, G.P. Picco and F. Risso, Designing a video conference system for active networks, in: *Proceedings of 2nd International Workshop on Mobile Agents*, Stuggart, Germany (1998).

[27] M. Calderon, M. Sedano, A. Azcorra and C. Alonso, Active network support for multicast applications, IEEE Network 12 (May 1998) 46–53.

[28] R. Caceres and V.N. Padmanabhan, Fast and scalable handoffs for wireless internetworks, in: *ACM/IEEE MOBICOM'96*, Rye, New York, USA (November 1996).

[29] R. Droms, Dynamic host configuration protocol, RFC1542 (October 1993).

[30] L. Zhang, S. Deering, D. Estrin, S. Shenker and D. Zappala, RSVP: A new resource ReSerVation protocol, IEEE Network (September 1993).

[31] MIL3 Inc., Opnet Modeler 3.5.A. Network Simulation Software (1997).

[32] C.L. Hedrick, Routing information protocol, RFC1058 (June 1988).

Kwan-Wu Chin is a PhD candidate at the School of Computer Science, Curtin University of Technology, Western Australia. He is currently investigating the application of active networking technologies to unicast and multicast routing in mobile networks. His research interests include mobile networks, active networks, location management, multicasting, and performance analysis of transport protocols.
E-mail: chinkw@cs.curtin.edu.au

Mohan Kumar is a Senior Lecturer at the School of Computing Science, Curtin University of Technology. Prior to joining Curtin University in 1992, he worked as a Scientific Officer at the Indian Institute of Science for six years. He is on the Edititorial Board of The Computer Journal and recently guest edited a special issue of The Computer Journal on Mobile Computing. Kumar has published over 60 articles in refereed journals and conference proceedings in the areas of parallel and distributed computing and mobile computing. He was awarded the Alumni Medal of the Indian Institute of Science 1993, elected senior member of the IEEE in 1995, and received the Excellence award at Curtin University in 1999. His areas of research interest include mobile computing and wireless networks, parallel and distributed computing, and computer architecture.
E-mail: kumar@cs.curtin.edu.au