ISSN:2231-0711
www.ijcset.net

Multicasting through Hop-by-Hop Routing Protocol Using Modified Recursive Unicast

U.Raghunath Reddy, K.Sekhar, P.Prabhavathi

Dept of Information Technology, G.Pullaiah College of Engineering & Technology
Kurnool, Andhra Pradesh, India
rag.knl@gmail.com
kraja.sekhar99@gmail.com
prabha1206@gmail.com

Abstract-IP Multicast is a hotly debated topic since many years. For this many reasons are responsible. Hence, the Internet is likely to be organized with both unicast and multicast enabled networks. Thus, it is of utmost importance to design protocols that allow the progressive deployment of the multicast service by supporting unicast clouds. This paper proposes HBH (Hop-By-Hop multicast routing protocol) which uses modified recursive unicast approach for Multicasting. HBH adopts the channel abstraction to simplify address allocation and implements data distribution using recursive unicast trees, which allow the transparent support of unicast only routers. Additionally, HBH is original because its tree construction algorithm takes into account the unicast routing asymmetries. As most multicast routing protocols rely on the unicast infrastructure, these asymmetries impact the structure of the multicast trees. HBH outperforms other multicast routing protocols in terms of the delay experienced by the receivers and the bandwidth consumption of the multicast trees.

***Keywords*: Multicast, Unicast , Incremental Service Deployment , Routing .**

1. INTRODUCTION

IP multicast is a technique for one-to-many communication over an IP infrastructure in a network. It scales to a larger receiver population by not requiring prior knowledge of who or how many receivers there are from M sources to N receivers. Multicast uses an addressing scheme based on IP class-D addresses, and routing protocols. In IP Multicast any host can send to a multicast group and any host can join it and receive data. The IP Unicast service model is also completely open, but the potential burden caused by unauthorized senders is amplified by the group size in multicast.The IP Multicast architecture is completed by group ad- dressing and routing protocols. A multicast group is identified by a class-D IP address which is not related to any topological information, as opposed to the hierarchical unicast addressing model. Therefore, multicast address allocation is complicated and multicast forwarding state is difficult to aggregate. Currently, there is no scalable solution to inter-domain multicast routing. Nevertheless, ISPs (Internet Service Providers) could be interested in multicast to face the increasing demand for network resources and content distribution. As a consequence, the Internet is likely to be organized with both unicast and multicast enabled networks. Therefore, it is of utmost importance to design protocols that allow the progressive deployment of the multicast service by supporting unicast clouds.

Multicast Groups: There are three types of IPv4 and IPv6 addresses: unicast, broadcast, and multicast. Unicast addresses are used for transmitting a message to a single destination node. Broadcast addresses are used when a message is supposed to be transmitted to all nodes in a subnetwork. For delivering a message to a group of destination nodes which are not necessarily in the same subnetwork, multicast addresses are used. While Class A, B, and C IP addresses are used for unicast messages, Class D addresses (224.0.0.0 - 239:255.255.255) are employed by multicast messages.

Multicast Addressing: A Class D IP address Figure1 is assigned to a group of nodes defining a multicast group. The most significant four bits of Class D addresses are set to "1110". The 28-bit number following these four bits is called "multicast group ID". Some of the Class D addresses are registered with the Internet Assigned Numbers Authority (IANA) for special purposes. The block of multicast addresses ranging from 224.0.0.1 to 224.0.0.255 is reserved for the use of routing protocols and some other low-level topology discovery or maintenance protocols. The block of multicast addresses ranging from 224.0.0.1 to 224.0.0.255 is reserved for the use Addresses ranging from 239.0.0.0 to 239.255.255.255 are reserved to be used for site-local "administratively scoped" applications, and not Internet-wide applications.

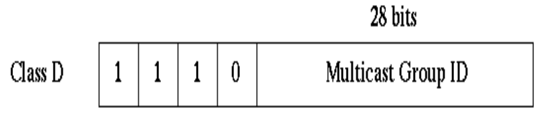


Figure1: Format of a class D IP address

2. BACKGROUND

2.1 Multicast distribution through recursive unicast

The basic idea of the (REUNITE) recursive unicast approach is that packets have *unicast destination* addresses. The routers that act as branching nodes for a specific multicast group are re- sponsible of creating packet copies with *modified* destination address in such a way that all group members receive the information. Figure 2 gives an example of the recursive unicast data distribution in REUNITE. The source sends data in unicast to the first receiver that joined the group. At a branching node, RB, incoming packets are addressed to the first receiver, ri, that joined the group in the sub-tree below RB. ri is stored in a special MFT entry, MFT<*S*>.*dst*. RB creates one packet copy for each receiver in its MFT (the destination address of each packet copy is set to the receiver's unicast address). The original packet is also forwarded to ri. In the example, *S* produces data packets addressed to $r1$ (these packets reach $r1$ unchanged). $R1$ creates one packet copy and sends it to $r4$. Since $R3$ is a non-branching node, it simply forwards the packets without consulting its MFT. $R5$ creates one packet copy to $r8$ and finally $R7$ creates copies to $r5$ and $r6$.The recursive unicast technique allows the progressive deployment of the multicast service because data forwarding is based on unicast addresses. Unicast only routers in the distribution tree are transparently supported. These routers are unable to be branching nodes of the tree but can forward data since unicast destination addresses are used.

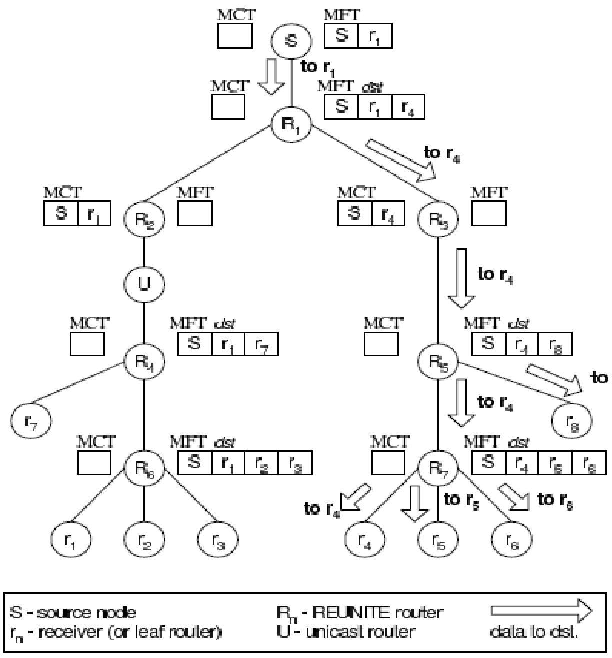


Figure 2: Data distribution in the recursive unicast approach.

2.2 The risks of asymmetric routing

Asymmetric routing means that the unicast path from *A* to *B* may differ from the path from *B* to *A*. Asymmetric unicast routing affects multicast routing since the majority of multicast routing protocols construct *Reverse* Shortest-Path Trees. Data packets from the source to a receiver follow the unicast route used to go from the receiver to the source. If these paths have different characteristics, e.g. different delays, the use of the reverse SPT may be problematic to Quos deployment. The ability to construct Shortest-Path Trees is therefore advantageous for a multicast routing protocol.

3. HOP-by-HOP MULTICAST PROTOCOL

HBH has a tree construction algorithm that is able to better treat the pathological cases due to asymmetric unicast routes. HBH uses two tables, one MCT and one MFT that have nearly the same function as in REUNITE. The difference is that one entry table in HBH stores the address of a *next branching node* instead of the address of a *receiver (*excepted the branching router nearest the receiver). The MFT has no *dst* entry. Data received by a branching router, HB, has unicast destination address set to HB (in REUNITE data is addressed to MFT<*S*>.*dst*).This choice turns the tree structure more stable than in REUNITE. A multicast channel in HBH is identified by < *S, G* >, where *S is* the unicast address of the source and *G* is a class-D IP address allocated by the source. This definition solves the address allocation problem while being compatible with IP Multi- cast. Therefore HBH can support IP Multicast clouds as leaves of the distribution tree.

HBH's tree structure has the advantage of an enhanced stability of the table entries when compared to REUNITE. The tradeoff is that in HBH each data packet received by a branching node produces $n + 1$ modified packet copies while in REUNITE it produces n modified packets. The tree management scheme of HBH minimizes the impact of member departures in the tree structure. This is possible because the MFT receiver entry is located at the branching node nearest. the receiver. For example, the departure of $r1$ in Figure 4 has a greater impact in the tree structure of REUNITE than in that of HBH. In the worst case, HBH may need one more change than REUNITE (it happens when a branching node becomes a non-branching one, e.g. after the departure of $r8$). In this example routes are symmetric so there are no route changes for other members when a member leaves the group. Tree reconfiguration in REUNITE may cause route changes to the remaining receivers. This is avoided in HBH.

3.1 Tree management in HBH

HBH has three message types: *join*, *tree*, and *fusion*. *Join* messages are periodically unicast by the receivers in the direction of the source and refresh the forwarding state (MFT entry) at the router where the receiver joined. A branching router "joins" the group itself at the next upstream branching router. Thus the *join* messages may be intercepted by the branching nodes which sign themselves *join* messages. The source periodically multicasts a *tree* message that refreshes the rest of the tree structure. *Fusion* messages are sent by potential branching routers and construct the distribution tree together with the *tree messages*. Each HBH router in *S*'s distribution tree has either a MCT<*S* > *or* a MFT<*S*>. A non-branching node in *S*'s distribution tree has a MCT<*S*>. MCT<*S*> has one Single entry to which two timers are associated, *t1* and *t2*. At the expiration of *t1* the MCT becomes stale and at the expiration of *t2* the MCT is destroyed.

A branching node in *S*'s distribution tree has a MFT<*S*>.Two timers, *t1* and *t2*, are associated to each entry in MFT<*S*>. When *t1* times out the MFT entry becomes stale and it is destroyed when *t2* expires. In HBH, a *stale* entry is used for data forwarding but produces no downstream *tree message*. A MFT entry in HBH can also be *marked*. A *marked* entry is used to forward *tree messages* but not for data forwarding. The basic ideas are: the first join *issued* by a receiver is never intercepted, reaching the source; the *tree messages* are periodically multicast by the source; these are combined with *fusion messages* sent by Potential branching nodes to construct and refine the tree structure.*r*1 joins the multicast channel at *S* which starts sending *tree*(*S, r*1) messages. These messages create a MCT<*S*> containing *r*1 at *H*1 and *H*3 (Figure 3(a)). When *r*2 joins the group by sending the first join(*S, r*2), this message is not intercepted and reaches *S* (the first *join* message is never intercepted). The *tree*(*S, r*2) produced by the source create MCT<*S*> state at *H*4 (Figure 3(b)).Both receivers are connected to the source through the shortest-path. Suppose now that *r*3 (unicast routes: *S* → *H*1→ *H*3 → *r*3 and *r*3 → *H3* → *H1* → *S*) joins the channel. It sends a join(*S, r*3) to S, which starts sending *tree*(*S, r*3) messages. As *H*1 receives two different *tree messages*, it sends a *fusion*(*S, r*1, *r*3) to the source. The reception of the *fusion* causes *S to* mark the *r*1 and *r*3 entries in its MFT and to add *H*1 to it. In the same way as *H*1, *H*3 receives *tree*(*S, r*1) and *tree*(*S, r*3) messages and thus send a *fusion*(*S, r*1, *r*3) to the source (Figure 3(c)).*H*3's MFT now contains *r*1 and *r*3. Subsequent *join*(*S, r*1) messages are intercepted by *H*1 and refresh the *r*1 marked entry in *H*1's MFT. The *join*(*S, r*3) messages refresh the *r*3 MFT entry at H3.*S sends* data addressed to *H*1, that sends it addressed to *H*3. *H*3 sends copies to *r*1 and *r*3. Subsequently, as *S receives* no more *join*(*S, r*1) neither *join*(*S, r*3) messages, its corresponding MFT entries are destroyed. The final structure is shown in Figure 3(d).In this way, HBH is able to use the good branching point to the distribution tree.

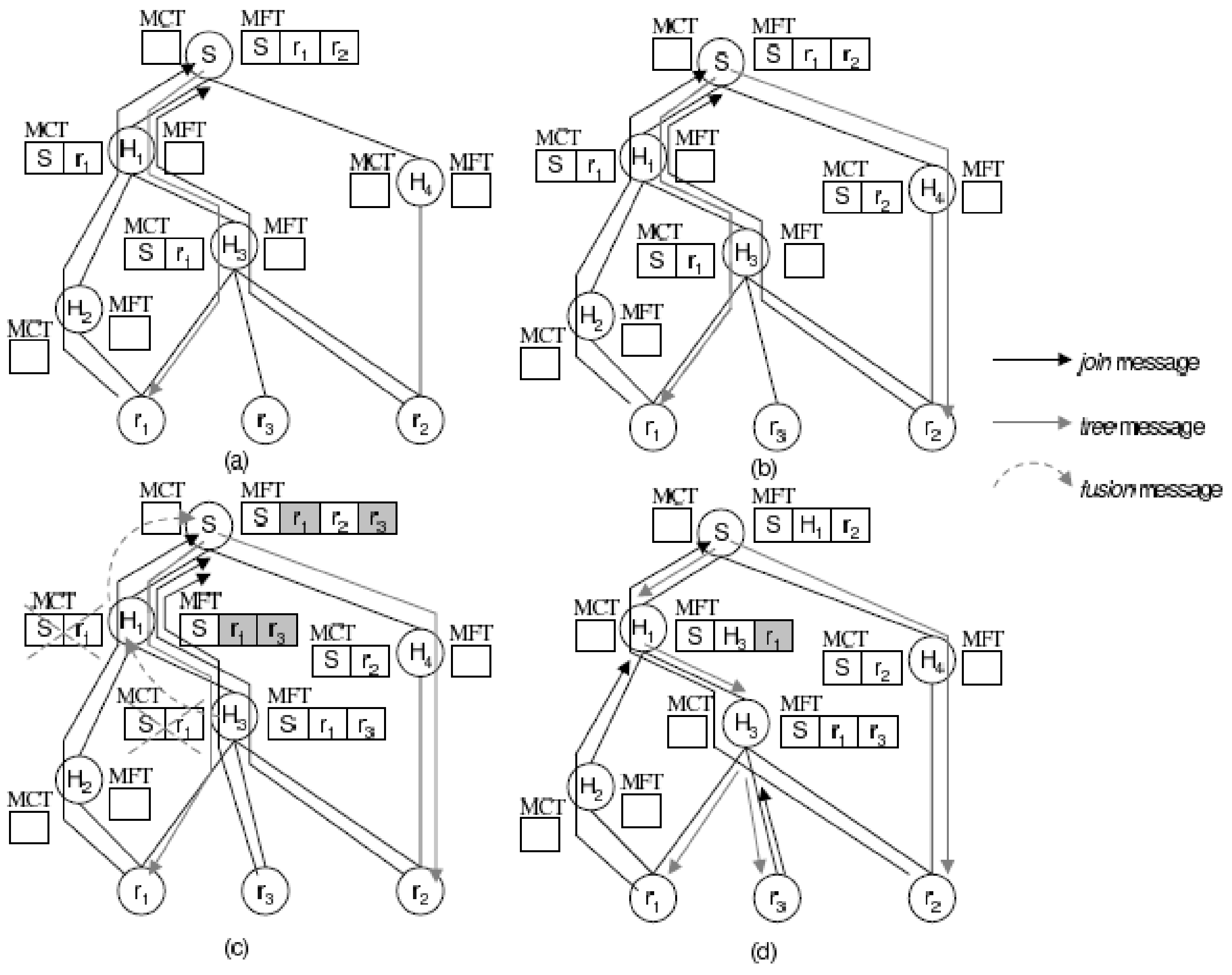


Figure 3: HBH tree construction mechanism.

3.2MESSAGE PROCESSING IN HBH

Figure 4 presents the processing rules of the three message types used by HBH to construct the distribution tree. Each receiver *r periodically* sends a *join*(*S, r*) in unicast to the source. The source periodically multicasts a *tree message* for each <*S, G*> *channels*.

Join message (Figure 4(a)) - When router *B* receives a *join*(*S, R*), it should forward this message unchanged if *B* has no MFT (1) or if *R* is not in *B*'s MFT (2). Only if *B*'s MFT has a *R* entry, *B* intercepts the *join*(*S, R*) and sends a *join*(*S, B*) afterwards. It means that *B* is a branching node of the channel <*S, G*>.*Tree message* (Figure 4(c)) - A *tree*(*S, R*) message received by router *B* is treated and forwarded in multicast. If B is a branching node, it may receive a tree *message* addressed to *B*. In this case *B* discards the message and sends a *tree message* to each node present in its MFT (1). If *B* is a branching node and *tree*(*S, R*) is not addressed to *B*, there is two possibilities: *R* is a new receiver (in which case *B* inserts *R* in its MFT and sends a *fusion* message upstream) (2) or *R* is present in *B*'s MFT which means that *B* does not receive *join*(*S, R*) messages from *R* - and in this case *B* simply has to refresh the *R* entry in its MFT and to send a *fusion* message upstream (3). If *B* is not a branching node, there is two possible cases: *B* was not in *S*'s distribution tree in which case *B* creates a MCT containing *R* (4), or *B* was already in the distribution tree but was not a branching node (in which case *B* has a MCT<*S* >) (5). If *R* was already present in *B*'s MCT, nothing has changed and *B* simply refreshes its MCT (6). If *R* is not present in the MCT, then maybe the MCT is stale in which case *R* replaces the previous MCT entry, or the MCT is up to date which means that there is a second receiver that will receive data through *B*, so *B* becomes a branching node. This implies the creation of a MFT, the destruction of the MCT, and a *fusion* message to be sent upstream (8). The *fusion* messages produced by *B* contain all the nodes that *B* maintains in its MFT - the nodes for which *B* is branching node.

Fusion message (Figure 4(b)) - Suppose router *B* receives a *fusion*(*S, R ...Rn)* from node *Bp.* If the message is not addressed to *B*, then *B* simply forwards it upstream

(1). If the message is addressed to *B*, then *B* marks the receiver entries in its MFT that are listed in the fusion *message*

(2). A marked entry in the MFT is used to *tree message* forwarding but not for data distribution. *Bp* is added to *B*'s MFT if it was not previously present. In addition, *Bp's t1* timer is expired - *Bp becomes stale.*

(3). Consequently, *Bp* will be used for data forwarding, but not for *tree message* forwarding. If *Bp was* already present in *B*'s MFT, then *Bp's t2* timer is refreshed (it avoids the destruction of the *Bp* entry), but its *t1* timer is kept expired

(4). If afterwards *Bp* (the node that produced the *fusion* for *R ...Rn)* receives the join messages produced by any of *R ...Rn,* it intercepts them and send a *join*(*S, Bp)* upstream. In this case the *R ...Rn entries* in *B*'s MFT will eventually timeout and be destroyed, while the *Bp entry* in *B*'s MFT is refreshed by the *join*(*S, Bp).* If *Bp* does not receive *join* messages from *R, ...Rn* , the emission of *fusion* messages continues since there is another node upper in the tree that receives the *join*(*S, R, ...Rn*) and periodically produce the *tree*(*S, R, ...Rn*) messages to these receivers. Nevertheless, this node will not forward data to these receivers, but to *Bp instead* since the receivers' entries are marked. *Bp is* responsible for data duplication.

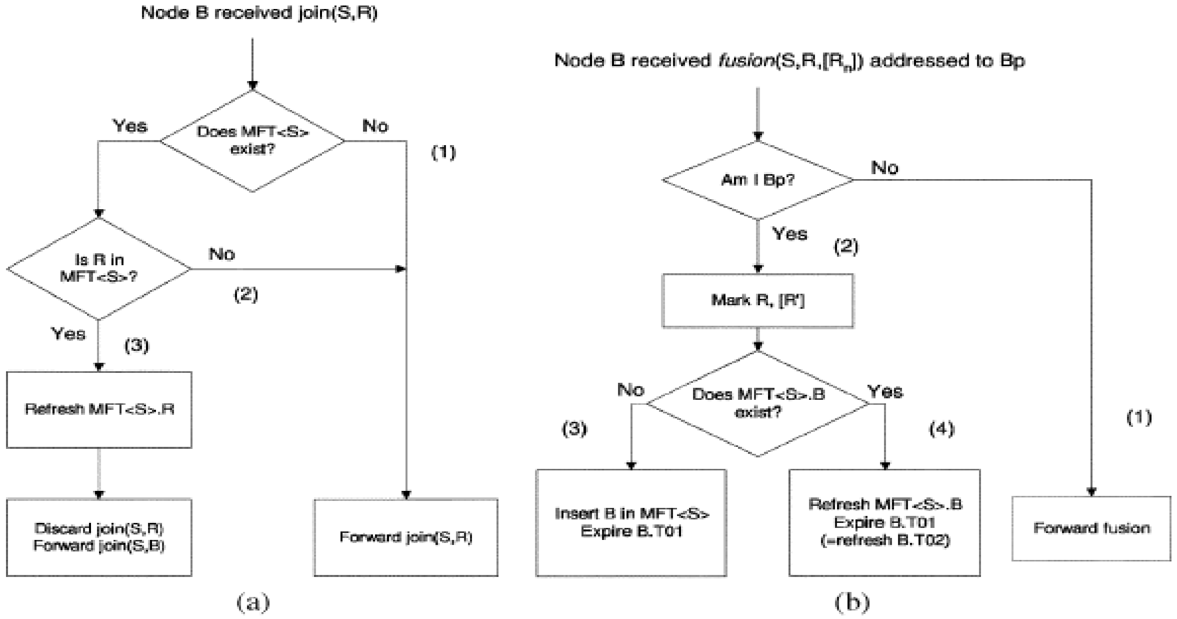
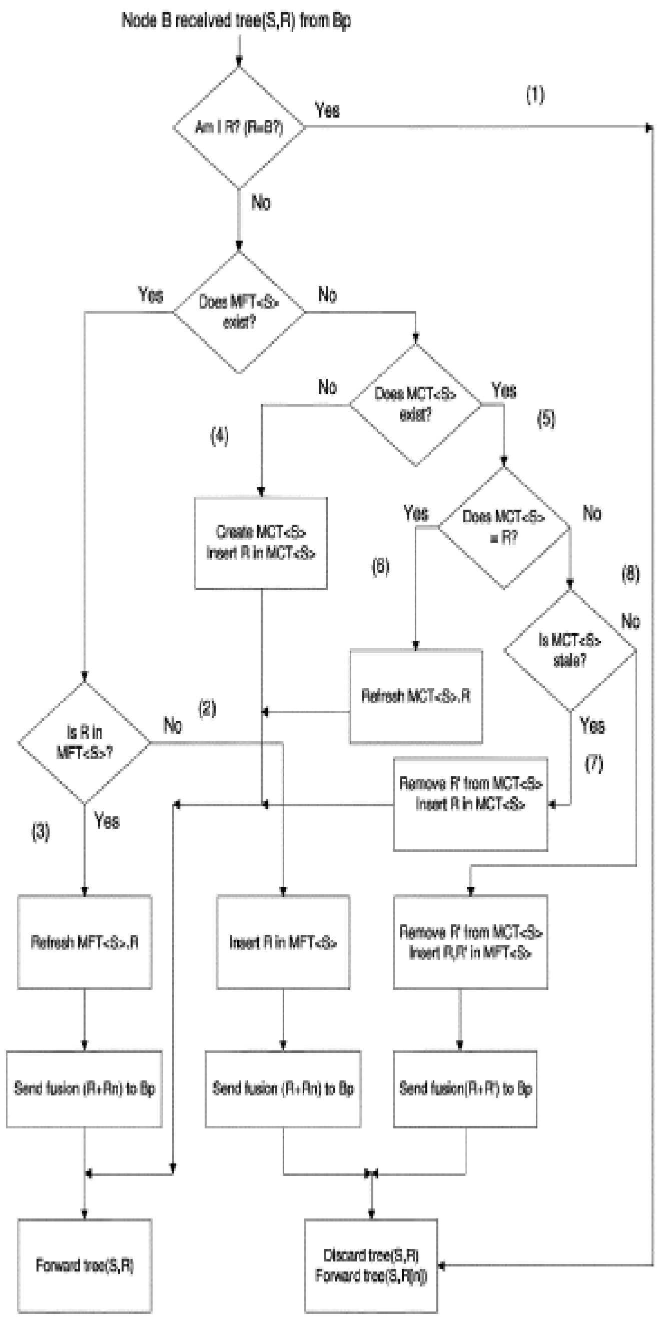


Fig 4 - Message processing in HBH. (a) Join message. (b) Fusion message.



(c)

Figure 4: Message processing in HBH. (c) Tree message

4. PROPOSED MODIFIED RECURSIVE UNICAST APPROACH (REUNITE II):

In the control protocol as in REUNITE and HBH, each router needs to maintain a MCT on the control Plane. The purpose of having MCTs is to mark routers that are not branching points as a part of the multicast tree. With this information in the control path, new branching points can be easily created using JOIN messages.

In this section, we describe a modified version of the control protocol; called REUNITE II.This eliminates the need for maintaining MCTs at routers. In addition, it also eliminates the race condition of joins experienced by REUNITE. As will be discussed later, these benefits are achieved at the expense of having more protocol message types and slightly more complex protocol state machines than the original control protocol. In the following, we will describe REUNITE II, and discuss the tradeoffs between REUNITE and REUNITE II. In order to remove MCTs, we need to introduce a new mechanism to create branching points for new receivers. The key idea is to rely on the forwarding path to discover where a new receiver can join the tree. The outline of REUNITE II is as follows. As in REUNITE, each receiver sends Periodic JOIN messages towards the root node. These messages are intercepted by the first node that maintains group state on the messages' paths. Note that unlike REUNITE, in REUNITE II the JOIN message can be intercepted only by nodes that are already branching points in the Multicast tree, as these are the *only* nodes that maintain group state in their MFTs. Once a node Intercepts a JOIN message, it either inserts the new receiver in its MFT (if the node is a "suitable" Branching point for the receiver), or generates a new message, called BRANCH, and forwards it down the tree. The purpose of the BRANCH message is to find a branching point for the new receiver. Ideally, a branching point is created at the first node at which the path towards the new receiver diverges from the path followed by the BRANCH message..The only difference is that, in order to better illustrate the behavior of the protocol, the path from S to R2 is changed to S → N1→ N3→ R2 (If we maintain the same route, i.e., S→ N4→ R2, there would be no interaction between R1 and R2 in REUNITE II, as both of them would join at S. The operations of REUNITE II are shown in Figures 5 and 6. For simplicity, TREE messages are not shown. We consider two cases: (a) all nodes implement REUNITE II, and (b) only a subset of nodes implements the Protocol.

4.1 Join Operation When All Nodes Implement REUNITE II

Figure 5 shows the main messages exchanged as a result of R1 and R2 joining the group. R1 joins first by sending a JOIN message towards the root (Figure 5(a)). Since no node maintains multicast state, the message is delivered to the root S. Upon receiving the message, node S creates an entry for the new receiver, as this is the first receiver to join the group. Next, R2 joins by sending a JOIN message towards S (Figure 6(b)). When S intercepts this message it first checks whether there is any receiver in the MFT of S that uses the same output interface as R2. In this example, such a receiver, R1, exists, as both paths
S → R1 and S → R2 share the link S: N1. As a result, node S generates a message, called BRANCH, and sends it towards R1. The message contains a field that specifies the group S, and a field that specifies the receiver that wants to join, i.e., R2. When the BRANCH message arrives at N1, N1 checks whether the traffic towards R1 and R2 uses the same output

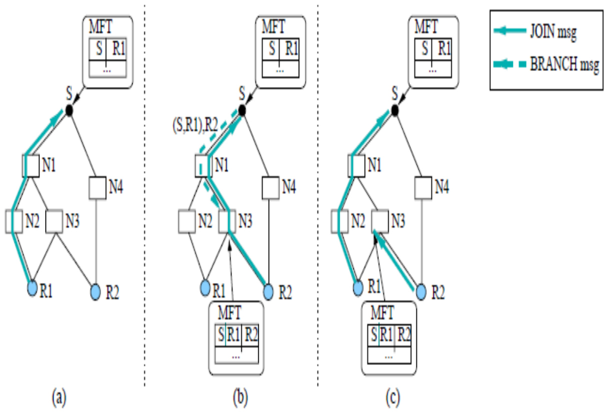


Figure 5: Example illustrating the join operation when all nodes are REUNITE II aware.

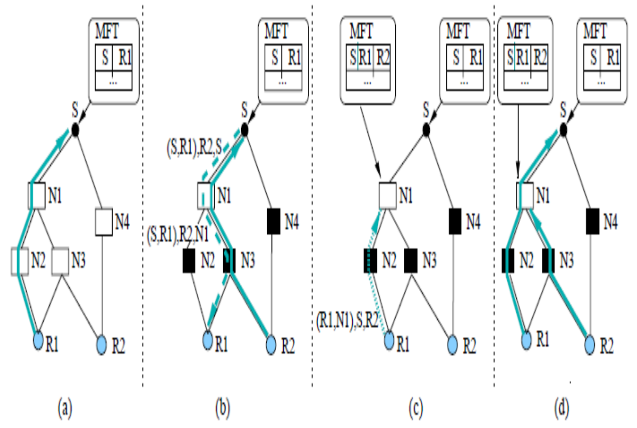


Figure 6: Example illustrating the join operation for the case when only node N1 implements REUNITE II

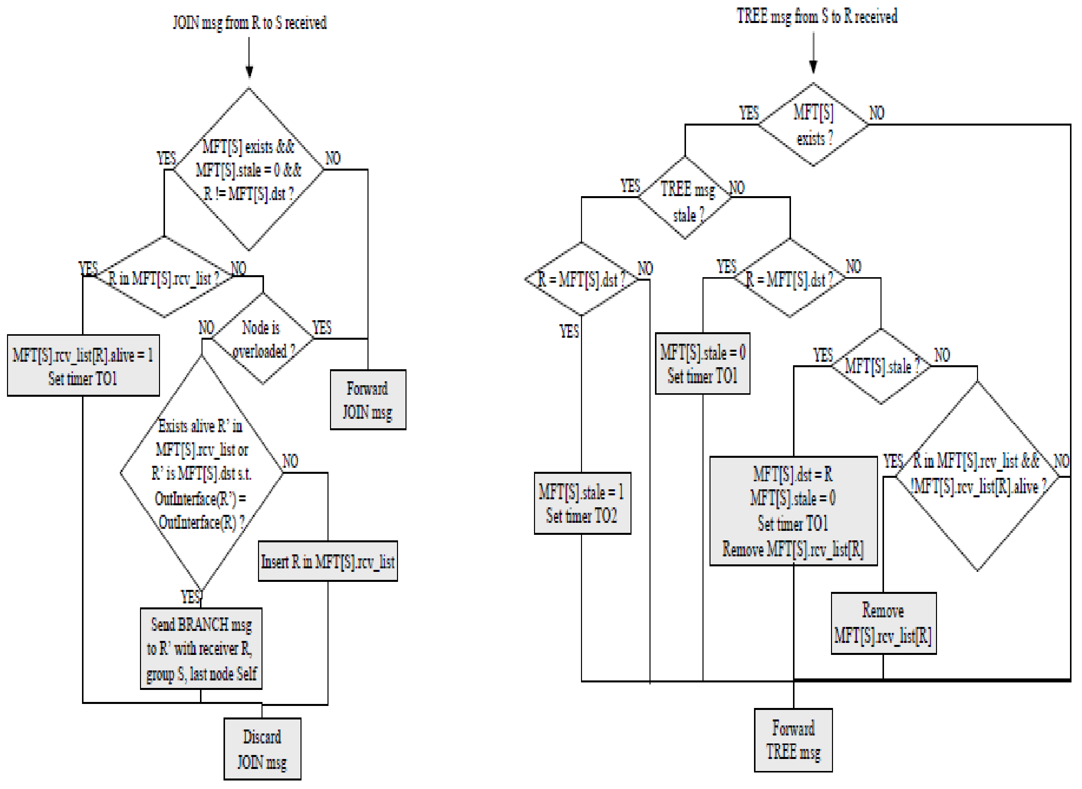

JOIN msg from R to S received
MFT[S] exists && MFT[S].stale = 0 && R != MFT[S].dst ?
YES
NO
R in MFT[S].rcv_list ?
Node is overloaded ?
MFT[S].rcv_list[R].alive = 1 Set timer TO1
Forward JOIN msg
Exists alive R' in MFT[S].rcv_list or R' is MFT[S].dst s.t. OutInterface(R') = OutInterface(R) ?
Insert R in MFT[S].rcv_list
Send BRANCH msg to R' with receiver R, group S, last node Self
Discard JOIN msg
TREE msg from S to R received
MFT[S] exists ?
TREE msg stale ?
R = MFT[S].dst ?
MFT[S].stale = 0 Set timer TO1
MFT[S].stale = 1 Set timer TO2
MFT[S].stale ?
MFT[S].dst = R MFT[S].stale = 0 Set timer TO1 Remove MFT[S].rcv_list[R]
R in MFT[S].rcv_list && !MFT[S].rcv_list[R].alive ?
Remove MFT[S].rcv_list[R]
Forward TREE msg

Figure 7: Detailed message processing algorithms for the REUNITE II protocol.

interface. Since this is the case, the BRANCH message is just forwarded to the next node N3. Upon receiving the BRANCH message, N3 checks similarly whether the next node along the paths towardsR1 and R2 is the same. Since this is not the case, N3 concludes that it is a branching point for R2 and as a result it installs the corresponding state in its MFT. Subsequent JOIN messages sent by R2 will be intercepted directly by N3, and issued to refresh R2's entry in the MFT.

5. CONCLUSIONS

We presented HBH; a multicast routing protocol that implements multicast distribution through recursive unicast trees, idea originally proposed in REUNITE.HBH allows the incremental deployment of the multicast service as unicast routers inside the network are transparently supported. The observation of the strengths and weaknesses of REUNITE directed the design of HBH.

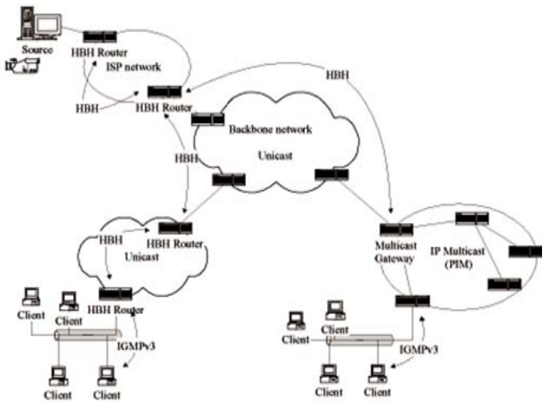


Figure 8: HBH Multicast Service Deployment

We also compared REUNITE and REUNITE II. The obvious advantage of REUNITE II over REUNITE is that it eliminates the need for control path state. REUNITE II has another advantage. Simultaneous joins can lead to a race condition such that a sub-optimal multicast tree is created for short transient periods. The cause is that control path state is not instantaneously created in REUNITE when a new receiver joins. In contrast, REUNITE II eliminates this problem. In REUNITE II, simultaneous joins can independently discover the optimal branching points without relying on any control path state, thus eliminating the race condition. Thus HBH performs better using REUNITE II .The objectives of HBH are:

- To go through unicast clouds**.**
- Member departure should have minimum impact on the tree structure.
- To provide lower cost trees.
- To guarantee that members receive data through the shortest path from the source.

REFERENCES:

[1] Costa, L.H.M.K.; Fdida, S.; Duarte, O.C.M.B.; Univ. Fed. do Rio de Janeiro "Incremental service deployment using the hop-by-hop multicast routing protocol",(IEEE/ACM Transactions on Networking, JUNE 2006).

[2] S. Bhattacharyya, C. Diot, L. Giuliano, R. Rockell, J. Meylor, D. Meyer,G. Shepherd, and B. Haberman, an Overview of Source Specific Multicast(SSM), July 2005.

[3] P.Radoslavov,C.Papadopoulos,R.Govindan, and D. Estrin, "A comparison of application-level and router-assisted hierarchical schemes for reliable multicast," *IEEE/ACM Trans. Netw.*, vol.12, no.3, Jun. 2004.

. [4] R. Vida and L. Costa, Multicast Listener Discovery Version 2 (MLDv2) for IPv6, RFC 3810, June 2004.

[5] C. Diot, B. N. Levine, B. Liles, H. Kassem, and D. Balensiefen,"Deployment issues for the IP multicast service and architecture," IEEE Network, pp. 78–88, Jan. 2000.

[6] http://www.cc.gatech.edu/fac/Mostafa.Ammar/MCAST.html

ABOUT AUTHORS:

U.Raghunath is currently pursuing B.tech Degree in Information Technology at G.Pullaiah College of Engineering and Technology, JNTU Anantapur, India. He is a Member of IEI. He did R&D in DeskStream (Desktop Virtualization) at GPCET. His research interests are in the area of routing, group communication, quality of service, multicast.

K.Sekhar is Currently pursuing B.tech Degree in Information Technology at G.Pullaiah College of Engineering & Technology, JNTU Anantapur, India. His research areas include routing protocols, wireless QoS, Wireless LAN protocols.

P.Prabhavathi is currently pursuing B.tech Degree in Information Technology at G.Pullaiah College of Engineering & Technology, JNTU Anantapur, India. Her research areas include multicast, security and mobile communications