

# Near Receiver Branching Point Multicast Protocol

<sup>†</sup>Mozafar Bag-Mohammadi, <sup>†</sup>Siavash Samadian-Barzoki and <sup>‡</sup>Nasser Yazdani

Router Lab.,  
University of Tehran,  
Tehran, Iran

<sup>†</sup>{mozafarb, s.samadian}@ece.ut.ac.ir, <sup>‡</sup>yazdani@ut.ac.ir

**Abstract**— Many multicast routing protocols construct SPTs (Shortest Path Trees) to deliver multicast data packets to the receivers with the minimum delay. Some protocols, on the other hand, try to minimize the bandwidth consumption in the multicast distribution tree using Steiner trees. In this paper, we propose a new multicast routing protocol called NRBP (Near Receiver Branching Point) which can construct a wide range of multicast distribution trees between these two extremes. Changing the protocol parameters controls the tree characteristics. Our protocol is not only able to build SPT with minimum total tree cost, but also it can create semi-Steiner tree with less data delivery delay. Benefits of the proposed protocol include some sort of congestion avoidance, controlling the MFT (Multicast Forwarding Table) memory requirements and QoS provisioning.

**Keywords**—multicast; multicast tree; SPT; Steiner tree; Multicast routing protocol; MPR

## I. INTRODUCTION

Many applications like video conferencing use multicast [10][13], which reduces the network load and data distribution delay dramatically compared to the multiple unicast data delivery. Multicast routing protocols construct multicast distribution trees in different ways aiming to achieve diverse goals. Minimizing data distribution delay and bandwidth consumption in the whole network are the two major criteria in the tree construction. The SPT (Shortest Path Tree) considers the first goal and the Steiner tree the second one.

Many protocols like PIM-SM (Protocol Independent Multicast – Sparse Mode) [5][8] use the SPT as the target. Some of them build an approximation of the tree using the shortest reverse path selection at each new receiver join. This results in a poor approximation in asymmetric networks which is the usual case [11]. Others build the tree in the forward direction trading off more join latency [9].

Steiner tree construction is an NP-complete problem [4]. Therefore, multicast routing protocols use heuristics to approximate it. Some methods try to achieve this goal in a static environment using a central node [4]. This node should be aware of network topology and membership status of each node. Thus, these approaches are not suitable for dynamic environments like Internet. Other protocols attempt to achieve the same objective in a dynamic environment [1][2].

Generally speaking, the SPT is not optimal in terms of bandwidth consumption. Meanwhile, Steiner trees may introduce extra delay in the data distribution. Hence, it is

attractive having a protocol that takes into account both criteria simultaneously. There are different solutions to this problem, which can be categorized as MPR (Multi-Path Routing) protocols [14]. Based on how the new member is connected to the tree, multicast routing protocols can be classified into two broad categories: SPR (Single-Path Routing) and MPR. SPR provides a single path to connect a new member to the tree whereas MPR provides multiple candidate paths [14]. An MPR protocol selects a path among the candidates using some special criteria.

In this paper, we propose NRBP protocol which can act as an SPR or MPR protocol. When the join latency is more important, the protocol can build the same approximate SPT selecting the shortest reverse path to the existing tree. Otherwise, the protocol acts as an MPR protocol. In this case, one can build a range of different possible trees between SPTs in forward direction and Steiner trees by using two control parameters namely  $C_{Max}$  and  $K$ . The value of zero for  $K$  results in an SPT with the minimum possible bandwidth consumption while value of infinity ( $K_{Max}$ ) makes NRBP to construct a Steiner tree with the minimum possible distribution delay. Values between these two extremes generate the trees that lie between the SPT and Steiner tree by placing more importance on one of the two main characteristics.  $C_{Max}$  controls the quality of the resulted tree by changing the number of candidate paths for new joins. The greater the value of  $C_{Max}$ , the higher the quality of the resulted tree. For example, one can achieve a more optimum SPT with a higher value of  $C_{Max}$  when the reduction in the bandwidth consumption is desired.

In NRBP, each new member trying to join to the group sends a *Join-Req* message toward the root of the existing multicast distribution tree. The root is either the sender in source specific trees or the RP (Rendezvous Point) in shared trees [5][8][6][7]. In the MPR mode, some tree nodes answer with *Join-Bid* messages and the new receiver can choose between them considering various criteria. The *Join-Bid* message can contain the following information:

- Path parameters: the amount of delay and bandwidth consumption resulting from the new path.
- Routers status in the new path: limitations in the processing power and MFT memory requirements and ability to tolerate the new traffic.

Therefore, the receiver can take into account the QoS constraints or avoid the congested path to select the desired path. On-tree nodes can manage their MFT memory requirements. They simply avoid sending *Join-Bid* messages

when they are in the shortage of memory or prefer to save their memory for more critical groups. NRBP outperforms the existing solutions in a sense that it can construct a bigger range of distribution trees with less control messages.

In section 2, we discuss related work and compare NRBP with previous methods. Section 3 presents the motivation and basic idea behind NRBP. We explain the proposed protocol in more detail in section 4. Finally, the paper is concluded in section 5 while showing the roadmap to future work.

## II. RELATED WORK

SBPT (Shortest Best Path Tree) [1] is the first algorithm that aims to build an SPT with the minimum bandwidth consumption. In this algorithm, a new member finds all shortest paths in reverse direction to the source and selects the one which adds minimum bandwidth to the tree. They consider hop count as a measure for the bandwidth consumption. The algorithm does not consider the join latency and checks all of the on-tree nodes for each new join. In addition, each node needs to know the network topology and the membership status of the other nodes.

Reference [2] introduces an MPR protocol that establishes and maintains multicast trees maximizing the bandwidth to be shared by multiple receivers and satisfying the maximum path length bounds for each receiver. For a delay sensitive multicast application, PLC-MS (Path Length Control – Most Strict) can save bandwidth which would be otherwise wasted by the SPT algorithm, while achieving the same shortest reverse path length. Furthermore, for multicast applications in which delay is not the highest priority, the PLC-S (Path Length Control – Strict) algorithm can achieve the lowest bandwidth consumption with only a moderate increase in the path length. To find the potential neighbor nodes for a connection, the joining node broadcasts a polling message. Neighbor nodes reply to let their existence and distances from the sender to be known to the joining node. This broadcast process generates a high control traffic, which can be controlled by placing a small value in the TTL (Time-To-Live) field. The value in the TTL field limits the searching area and, hence, setting a small value to this field deteriorates the performance of the protocol.

The first MPR multicast routing protocol was proposed in [3]. The authors suggest a mechanism called DSJ (Directed Spanning Join) to reduce the aforementioned traffic load in finding on-tree nodes. While it is still a directed broadcast scheme, as the directed spanning joins get closer to the root of the tree, the edges become pruned at a greater rate. A new MPR routing protocol is proposed in [12] which generates significantly less messaging overhead than flooding with TTL and DSJ protocols. Messaging overhead is reduced by allowing each receiver to specify how broad the area of multiple-path search should be, depending on its end-to-end delay requirements.

## III. PROBLEM DEFINITION

Fig. 1 illustrates a sender  $S$  sending data packets with size  $B$  to receivers  $r1$  and  $r2$  using unicast data delivery. We define  $B_{Total}$  to be the total amount of the bandwidth consumed in a data delivery from a sender to the receivers at a given time.

This value can be computed as a multiplication of the total traversed links and the packet size. In addition, we define  $P_{Total}$  to be the total path length which is the sum of the path links from all of the multicast receivers to the sender. For example, in Fig. 1,  $B_{Total}$  is equal to  $10B$  since 10 instances of the data packet passes the links and  $P_{Total}=5+5=10$  since the path length from the sender to each receiver is 5.

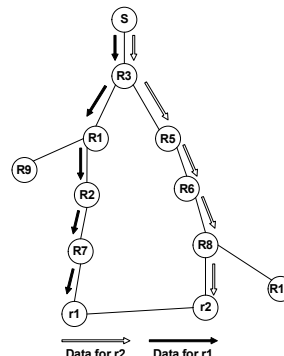


Figure 1. Data distribution using separate unicast delivery

As depicted in Fig.2b, if we use the SPT multicast data delivery, then, we will achieve  $B_{Total}=9B$  and  $P_{Total}=10$ . This results in a small bandwidth saving compared to the unicast data delivery while the same shortest paths are used. The reason for this little benefit is that the branching point of the multicast distribution tree is far from the receiver and near to the sender. Thus, to achieve more bandwidth saving with the same shortest paths, the protocol should try to keep the branching points as much as possible near the receivers.

To construct a multicast tree with the minimum possible bandwidth consumption, one can approximate a Steiner tree using a greedy algorithm [4]. This algorithm develops a multicast tree by connecting each new receiver node to its nearest multicast receiver that has been already connected to the sender of the target multicast stream. An example of this tree is shown in Fig.2a which results in  $B_{Total}=6B$  and  $P_{Total}=5+6=11$ . As mentioned previously, this kind of tree has lower bandwidth consumption than the SPT and more distribution delay.

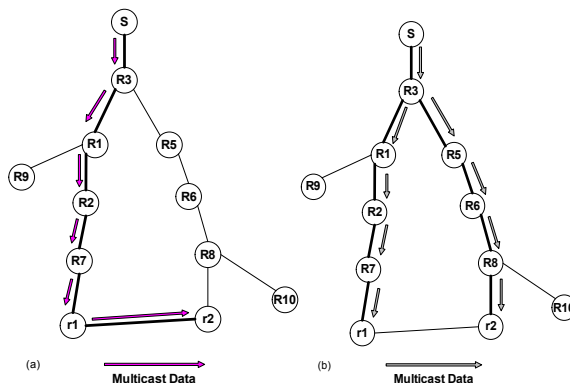


Figure 2. Multicast data distribution using a) Greedy algorithm to construct Steiner tree b) SPT

For all 5 receivers to receive the multicast data packet from node  $S$  in Fig.3, exactly 4 copies of the original packet

must be created somewhere in the distribution tree. This can be generalized for  $n$  receivers where the number of copies will be  $n-1$ . Therefore, the problem is how to make these copies in the most appropriate nodes to reduce the bandwidth consumption and the data distribution delay. The most appropriate node is one of the on-tree nodes that can satisfy those criteria.

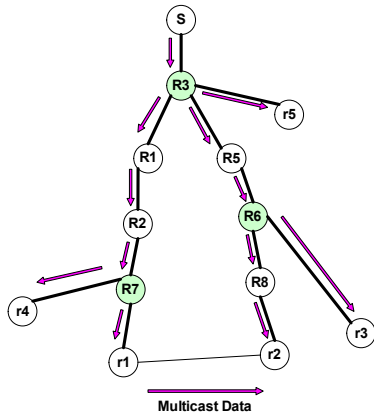


Figure 3. Multicast data distribution through SPT

In constructing the SPT, the main idea in NRBP, in order to reduce the bandwidth consumption is to join the tree as close to the joining node as possible. Fig.4 illustrates an example where  $r_{new}$  wants to join the SPT identified by thick lines. The four possible paths to join the tree are shown with numbered arrow lines. SPT selects path number 1 while different Steiner tree construction approaches may select paths number 2 through 4. For example, the greedy algorithm chooses path number 4. Path number 2 is the path with the same delay characteristics as path number 1 and with lower extra bandwidth consumption. Neither SPT nor the greedy algorithm guarantees to select this path.

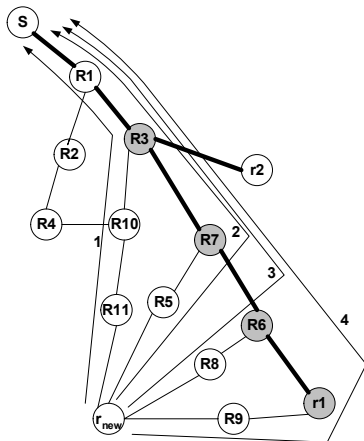


Figure 4. Possible path for addition of a new receiver

#### IV. NRBP PROTOCOL

In NRBP, there are 3 different control messages, namely *Join-Req*, *Join-Bid* and *Join*. The protocol can operate either in the SPR or MPR mode. When the join latency is more important, the administrator sets a special field named  $J$  to

enforce the SPR operation which results in a shortest reverse path multicast tree. Otherwise, the protocol works in the MPR mode. When a new member wants to join the tree, it sends a *Join-Req* message toward the root of the tree which is either the sender in source specific trees or the RP in shared trees. The message also includes the group address. In the SPR mode, this message installs the appropriate states in the path until it reaches an on-tree node and consequently the tree is constructed in the shortest reverse path. On the other hand, in the MPR mode, the *Join-Req* message reaches an on-tree node with no intermediate operation and triggers some *Join-Bid* messages in candidate nodes in response. These nodes are the result of a local search process around the on-tree node that received the *Join-Req* message. The administrator can control the depth of this local search in the tree using a special parameter named  $C_{Max}$ .

Pseudo-code 1 shows the operation in node  $A$  when it receives a *Join-Req* message:

```

if ( $J=1$ ) /* SPR mode */
{
  Insert  $iif_{Join-Req}$  in MFT;
  if ( $A \notin tree$ )
    Forward the Join-Req message
    toward the root of the tree;
}
else /* MPR mode */
{
  if ( $A \in tree$ )
  {
    if (resource exists)
      Send a Join-Bid message to
      the sender of Join-Req;
    if ( $C < C_{Max}$ )
    {
      Increase  $C$ ;
      Send Join-Req to
       $AllTreeLinks - iif_{Join-Req}$  ;
    }
    else
      Discard the Join-Req
      message;
  }
  else
    Forward the Join-Req message
    to the next-hop toward the root
    of the tree;
}

```

#### Pseudo-code 1: Node $A$ received a *Join-Req* message

In this pseudo-code,  $iif_{Join-Req}$  is the interface on which node  $A$  receives the *Join-Req* message. MFT is the Multicast Forwarding Table which includes the set of all tree branches for each multicast tree ( $AllTreeLinks$ ).  $C$  is a special field in *Join-Req* messages that is set to zero by the new member. It is used as a counter which indicates the current search depth in the tree.

It is possible that a new member receives no *Join-Bid* message in response of its original *Join-Req*. This can happen due to the shortage of resources in on-tree nodes. To consider this situation, the new member sets a timer called  $T_{Join-Req}$  upon

sending the *Join-Req* message. This timer is computed as  $T_{Join-Req} = T_{Root-RTT} + T_p * C_{Max}$ .  $T_{Root-RTT}$  is the RTT (Round Trip Time) estimation of the new member to the root of the tree and  $T_p$  is the process time of a *Join-Req* in an on-tree node. When  $T_{Join-Req}$  expires, the new member can use the received *Join-Bid* messages to choose the best candidate on-tree node for its *Join* message. If there were no *Join-Bid* message received, the new member sets  $T_{Join-Req}$  to the new value in an exponential back-off process.

A *Join-Bid* message includes a copy of the  $C$  field in *Join-Req*,  $D_{br}$  (the distance from the bidder node to the new member) and  $D_{sb}$  (the distance from the root of the tree to the bidder).  $D_{sb}$  is computed and stored previously using the *Join* messages of other group members during the tree construction process. In addition, *Join-Bid* contains a variable length field  $P^*$  which collects a path vector of the intermediate nodes from the bidder to the new member. Upon reception of a *Join-Bid* message, each node on the path, except the new member, adds its IP address to the  $P^*$  field. This path vector would be used by *Join* messages to construct the same path if the bidder were selected as the best candidate on-tree node based on the protocol parameters.

The following pseudo-code represents the selection algorithm in the new group member.

```

Let  $D_1 \leftarrow \{\}, D_{min} \leftarrow +\infty;$ 
For ( $\forall$  Join-Bid received before
       $T_{Join-Req}$  expires)
{
  Let  $D_{sr} \leftarrow D_{sb} + D_{br};$ 
       $D_{min} \leftarrow \min(D_{min}, D_{sr});$ 
       $D_1 \leftarrow D_1 \cup \{(D_{sr}, D_{sb}, D_{br}, P^*)\};$ 
}
Let  $D_2 \leftarrow \{\}, D_{opt} \leftarrow +\infty;$ 
For ( $\forall$   $(D_x, D_y, D_z, P) \in D_1$ )
{
  if ( $D_x \leq D_{min} + K$ )
    if ( $D_z < D_{opt}$ )
    {
       $D_2 \leftarrow \{(D_x, D_y, D_z, P)\};$ 
       $D_{opt} \leftarrow D_z;$ 
    }
    else if ( $D_z = D_{opt}$ )
       $D_2 \leftarrow D_2 \cup \{(D_x, D_y, D_z, P)\};$ 
}
Let  $D_{last} \leftarrow +\infty;$ 
For ( $\forall$   $(D_x, D_y, D_z, P) \in D_2$ )
{
  if ( $D_y < D_{last}$ )
  {
     $D_{last} \leftarrow D_y;$ 
     $(D_{select}, P_{select}) \leftarrow (D_x, P);$ 
  }
}
Send Join message through  $P_{select}$  including
 $D_{select}$ ;

```

**Pseudo Code 2: Node A is joining group G with sender S**

In this algorithm,  $K$  is a parameter that controls the type of tree to be constructed. The administrator sets this value to zero to achieve the SPT with the minimum total tree cost. If the parameter is set to infinity, the protocol constructs a Steiner tree with the minimum delay.

First, the algorithm finds the *Join-Bid* with the minimum cost from the root to the new member ( $D_{min}$ ) among the set of all received *Join-Bid* messages ( $D_1$ ). Then, it extracts the set of *Join-Bid* messages ( $D_2$ ) from  $D_1$  which have the minimum cost from the bidder node to the new member and its cost is at most  $D_{min} + K$ . The candidate node which has the minimum distance from the root is selected among  $D_2$  as the best candidate. The new member then sends a *Join* message to the selected node through the  $P_{select}$  path vector. The *Join* message contains the distance through the selected node between the root of the tree and the new member ( $D_{select}$ ) as well. Each node  $A$  on the path vector calculates  $D_{sa} = D_{select} - D_{ar}$  to be used for future *Join-Bid* messages where  $D_{sa}$  is the distance from the root to node  $A$  and  $D_{ar}$  is the distance from this node to the new member. The *Join* message also installs appropriate states in the path to construct the new tree branch. When the *Join* message reaches the selected candidate node, the tree construction is completed.

#### A. Example

Fig.5 shows an example of the NRBP protocol operations with  $J=0$ ,  $C_{Max}=2$  and  $K=0$ . In this figure, the network is symmetric i.e. each path has equal cost in both directions. Furthermore, hop count is considered as the cost in the network. In Fig.5a., the new member  $r_{new}$  sends a *Join-Req* message with  $C=0$  toward the root of the tree (node  $S$ ) and sets  $T_{Join-Req}$  to wait for candidate nodes' responses. The message meets the tree (indicated with thick lines) in  $R3$  which sends the same message to  $R1$  and  $R7$  on the tree increasing the value of  $C$  to 1. Since the value of  $C$  in *Join-Req* message is less than the control parameter  $C_{Max}$ , the on-tree nodes  $R1$  and  $R7$  send this message with  $C=2$  to  $S$ ,  $r2$  and  $R6$  respectively. This message goes no further in the tree (Fig.5b) since the  $C$  field is no longer less than  $C_{Max}$ . Upon receiving the *Join-Req* message, each on-tree node sends a *Join-Bid* message in response toward  $r_{new}$ . As can be seen in Fig.5b,  $R1$  and  $S$  do not send *Join-Bid* message even though they have received the *Join-Req* message. The reason is that an on-tree node should not send *Join-Bid* message on the same interface it has received or sent a *Join-Req* message. This is omitted from Pseudo-code 1 for the sake of simplicity but should be considered to eliminate unnecessary *Join-Bid* messages.

When  $T_{Join-Req}$  expires,  $r_{new}$  uses the received *Join-Bid* messages from  $r2$ ,  $R3$ ,  $R7$  and  $R6$  to select the best on-tree node candidate. After processing these messages,  $r_{new}$  selects  $R7$  and sends a *Join* message towards it (Fig.5c). The *Join-Bid* message from  $R7$  included  $R5$  and  $R7$  in its path vector  $P^*$ . This path vector is thus used to construct the tree using the *Join* message back to  $R7$ . This technique enables NRBP to construct the tree in a shortest path in the forward direction from the root to the receivers when the network is asymmetric. Besides,  $R5$  computes the distance from node  $S$  to itself in the forward direction using  $D_{sr}$  (the distance from node  $S$  to  $r_{new}$  in forward direction). Consequently,  $D_{sr5} = 5-1=4$  where  $D_{sr5}$  is the distance from node  $S$  to  $R5$ ,  $D_{sr} = 5$  and  $D_{r5r} = 1$  (the

distance from node  $R5$  to  $r_{new}$ ). Fig.5d shows the resulting SPT with the minimum possible extra cost.

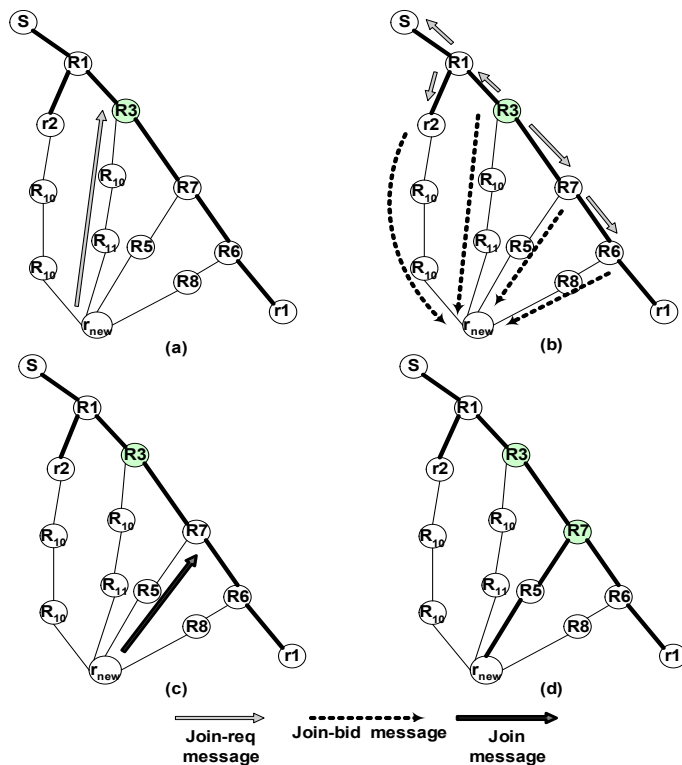


Figure 5. NRBP operation Example

### B. Leave messages

When a group member wants to leave the group, it sends a *Leave* message using the same path vector  $P$  that it used for the *Join* message. Therefore, a new member must save the associated fields that correspond to the *Join* message. Upon receiving of *Leave* message, each node on the path deletes the appropriate state from its MFT to remove the tree branch leading to the leaving member.

The multicast state corresponding to each tree link is also destroyed periodically after its creation unless it receives the same *Join* message through that link. This implies that the states are soft and each member should repeat its join process periodically as in PIM-SM protocol.

## V. CONCLUSION

NRBP protocol can operate in the SPR or MPR modes. When the join latency of the new group members is more important, the protocol builds the shortest reverse path tree. In the MPR mode, the protocol builds a range of different possible trees between an SPT in the forward direction and Steiner trees using  $K$  and  $C_{Max}$  as control parameters. The value of zero for  $K$  results in an SPT with the minimum possible total tree cost while value of infinity constructs a Steiner tree with the minimum distribution delay. Values between these two extremes results in a wide range of trees which balance the total tree cost and end-to-end distribution delay.  $C_{Max}$  controls the quality of the resulted tree by changing the number of candidate paths for new joins. The

protocol includes many attractive features such as QoS provisioning and memory management of on-tree nodes.

The performance analysis of the protocol and its behavior in different dynamic networks is on the way. We consider fault tolerance study of the protocol as a future work.

## REFERENCES

- [1] H. Fujinoki and K. Christensen, "The New Shortest Best Path Tree (SBPT) Algorithm for Dynamic Multicast Trees," Proceedings of the IEEE 24th Conference on Local Computer Networks, pp. 204-211, October 1999, Boston, Massachusetts
- [2] H. Fujinoki and K. Christensen, "A Routing Algorithm for Dynamic Multicast Trees with End-to-End Path Length Control," Computer Communications, Vol. 23, No. 2, pp. 101-114, January 2000.
- [3] K. Carlberg, J. Crowcroft, "Building shared trees using a one-to-many joining mechanism", ACM SIGCOMM Computer Communication Review, v.27 n.1, p.5-11, Jan. 1997
- [4] P. Winter, "Steiner problem in networks: a survey", Networks, v.17 n.2, p.129-167, summer 1987
- [5] S.Deering, D.Estrin, D.Faranacci, V.Jacobson, C.G.Liu, L.Wei, "The PIM Architecture for Wide-Area Multicast Routing", IEEE/ACM Transactions on Networking, Vol.4, No.2, April 1996
- [6] T.Ballardie, P.Francis, J.Crowcroft, "Core Based Trees (CBT): An Architecture for Scalable Inter-Domain Multicast Routing", ACM SIGCOMM'93
- [7] K.C.Almeroth, "The Evolution of Multicast: From the MBone to Inter-Domain Multicast to Internet2 Deployment", IEEE Network, Jan./Feb. 2000
- [8] S.Deering, D.Estrin, D.Faranacci, V.Jacobson, C.G.Liu, L.Wei, "An Architecture for Wide-Area Multicast Routing", Proc. SIGCOMM'94, Computer Communication Review, Vol.24, No.4, Oct.1994
- [9] Luis Henrique M.K. Costa, Serge Fdida, Otto Carlos M.B. Duarte, "Hop-by-hop Multicast Routing Protocol", ACM SIGCOMM'2001
- [10] S.Casner, S.Deering, "First IETF Internet Audiocast", ACM SIGCOMM, Computer Communications Review, Vol. 22, No. 3, July 1992
- [11] Paxson, "End-to-End Routing Behavior in the Internet", In Proceedings of SIGCOMM '96 (Stanford, CA, August 1996).
- [12] H. Fujinoki and K. Christensen, "The Directed Reverse Path Join (DRPJ) Protocol: An Efficient Multicast Routing Protocol," Computer Communications, Vol. 24, No. 12, pp. 1121-1133, 2001.
- [13] M.R. Macedonia, D.P. Brutzman, "MBone Provides Audio and Video Across the Internet", IEEE Computer Magazine, Vol. 27, No. 4, pp. 30-36, April 1994
- [14] Striegel, G. Manimaran, "A Survey of QoS Multicast Routing Issues," IEEE Communications, June 2002.