A Reliable And Scalable Multicast Model (RSM2)

Ruchi Gupta, Pramod Kumar Sethi

Abstract: Multicasting is the ability of a communication network to accept a single message from an application and to deliver copies of the message to multiple recipients at different location[1]. With the emergence of mobile users, many existing Internet -protocols , including those with multicast support, need to be adapted in order to offer support to this increasingly growing class of users. Our research in multicasting, as to design a Multicast Model , which provides reliability & scalability with best path for data delivery . Reliability means guaranteed Delivery of packets. Scalability means capability to serve growing needs .In this context , A few concepts of Proactive routing technique are used to make available this model in Infrastructured wireless also. Minimum Spanning path is used to deliver the packets ,to reduce the cost & delay.

Index Terms: Combo-Casting, Minimum Spanning Path, Multicasting, Reliable , Scalable.

I. INTRODUCTION

In the last few years, the Internet has changed from a pure scientific network to the basis of data communication in every-day life. The number of users grows still exponentially and has already reached the order of magnitude of tens of millions. The added spectrum and number of users introduce also new forms of communication into the Internet: Communication not just between two peers,but true group communication. The foundation for the group communication in the Internet is the IP-multicast service.
Althogh most of the network connections that were used before , multicasting have been known ,were Unicast (one-to-one) connections. They were basic and can be used for reliable data transmissions back and forth between the two connected nodes. However, these connections are not appropriate for communications from one sender to multiple receivers (one-to-many) or for many senders to many receivers (many-to-many). Multicast connections, particularly IP Multicast connections, may be more appropriate. IP Multicast connections may be particularly useful for one-to-many communications, as the sender need only transmit data packets once, and multicast enabled routers will copy the packets and send them to joined receivers. As the standard is implemented, there is no need for a sender to generate as many copies of the packets ,as the number of receivers. The need only ,is to , know the multicast IP address (from the D block of IP addresses).
Multicasting represents an efficient mechanism that implements point-to-multipoint communications[2].
Applications that utilize it falls into two classes:

- Soft real-time multicast applications : These have the capability to handle delay, for e.g. video conferencing ,service discovery and distance learning.
- Fully reliable multicast applications : These expect reliable data transmission ,for e.g. software distribution and access to distributed databases.

Reliable multicast transport is considerably more complex than reliable unicast. It is generally difficult to build a generic reliable transport protocol for multicast , such as TCP which is a generic transport protocol for unicast . It is a good idea to implement different kinds of protocols to serve different kinds of applications instead of having one protocol to serve all kinds of applications. Doing so will make the protocols simpler to implement and more efficient.

Since the early 1980’s, many protocols have been introduced for reliable multicast communications. There are many ways to implement a reliable multicast protocol . Protocols are referred to as[3]:

- Sender-Oriented : if the sender of a packet is responsible for ensuring that all receivers have obtained a copy of the packet.
- Receiver-Oriented : if the responsibilities for detecting missing packets lie with receivers. A protocol with unordered delivery does not make guarantee on the order in which packets are delivered to the receiver.
- Source-Ordered : this protocol maintains the order of transmission for each sender ,i.e., a receiver obtains the packets from a specific sender in the same order in which packets were transmitted by that sender.
- Totally-Ordered protocol : it ensures that all packets are received by all receivers in the same order.

Typically, with a sender-initiated approach, the receivers acknowledge (ACK) each packet they receive. It is upto the sender to keep track of which packets have been acknowledged, and to resend those which have not been acknowledged within a certain time interval.

Revised Manuscript Received on 30 June 2012.
* Correspondence Author
Ruchi Gupta*, Department of Computer Science, Mahamaya Technical University, Krishna Engineering College , Mohan Nagar, Ghaziabad , India .
Pramod Kumar Sethi, Department of Computer Science, Mahamaya Technical University, Krishna Engineering College, Mohan Nagar, Ghaziabad, India .

This means that the sender must keep state information and timers for each receiver. Keeping state for each receivers becomes infeasible as, the number of receivers stretches up into the millions. Furthermore , the volume of acknowledgment-messages may overwhelm the sender , resulting in what is referred to as ACK- IMPLOSION problem(shown below in figure).Hence , this scheme is not scalable.To remove this Ack-implosion problems, various protocols have been proposed like- SRM (Scalable Reliable multicast),RMTP (Reliable Multicast Transport Protocol) .
We will work on To formulate a model(RSM2) which provides scalable & reliable multicast ,where each source ,who wants to send the message ,will use best-path , that is obtained via “Kruskal’s Minimum Spanning Path Algorithm”.

II. PROBLEM STATEMENT

There have been many models and protocols (SRM , RMTP..) have been introduced , to provide reliability and scalability in multicasting communications . The goal of the research is **to design an algorithm, describing Multicast Model (RSM2) ,which achieves scalability and reliability**. Along with it, our model opt flat approach . The problem with hierarchical approach , is that , everytime when a receiver becomes the sender , entire hierarchy get modified .The hierarchical Model , RMTP, does not fit in a situation of, where, many nodes can send data simultaneously at the same time . Another problem was to choose an effective Designated Receiver , to deliver the packets to all the nodes , under that DR. Our Multicast Model removes all the above stated problems and provides a cost-effective , delay-effective path to deliver the packets .

III. DESIGN OBJECTIVES

The design of RSM2 is motivated by the local recovery scheme of RMTP and other existing reliable multicast protocols. RSM2 is intended to take advantages from the existing protocols and avoid their drawbacks. The model has been designed inorder to meet meet the top level goals which must be met at very first stage. Since , many of reliable multicast issues are still under research , some second level goals are established and are expected to be fully met at the second stage.

In the Internet environment , hosts are generally interconnected via heterogeneous links and dispersed world-wide. The quality of service of their network links such as the bandwidth and packet propagation time varies from one to another. The primary goal is thus to provide a reliable data delivery service over such a heterogeneous environment[4].

The goals of RSM2 are summarized as follows :

- It must ensure reliable data delivery.
- It must scale well to large group of users.
- It must continue to work for majority of receivers even in case where some receivers may leave their group , or some new receivers join the group , during the transmission.
- It must provide reasonable performance .
- It must work in Infrastured wireless networks.

IV. RSM2 ARCHITECTURE AND ASSUMPTIONS

RSM2 is based on flat architecture . Let all the nodes in the network are connected with each other ,through local area switches or routers . These routers are colocated with Dynamics Manager(DM).
The assumptions made in the design of model are as follows :

1. Active Server Based Local Recovery : It makes use of Specially designated hosts ,that have all the network ‘s computational ability, as Known as Dynamics Manager(DM).
2. Dynamics Manager : DMs are colocated with each router of the network. They have all the essential network computational ability like – to maintain a proper data of the nodes that are linked with it , to compute a Partial_cost_ matrix , and to assign a priority .
3. Cost Matrix : In an Heterogeneous environment , it is not possible that all links are alike. Hence , on the basis of their property we assign a cost to each link . Cost matrix shows the cost associated with each link.
   If there is no link between any two node , in that case , matrix assigns the cost as an infinity.
4. Priority Matrix : In the model , priority matrix is designed from the cost matrix . To send the packets , the path is decided on the basis of priority matrix.
5. Echo packet : Whenever a node wants to send the data to others , then sender first sends an echo packet . In that packet , there are two fields :

- Group-id : It indicates the group to which sender wants to communicate.
- Sender –id : It defines the address of the sender .

Echo packet as moves through the network , it stores the information about path and cost .

6. Response Packet : This packet is sent by DM ,to the sender , in response to Echo Packet.
7. IGMP drive : As the Echo packet received by the DMs , they run IGMP protocol , to know the group status of the nodes under them.This report is forwarded to their neighbouring DMs . These reports make the DMs up-to-date always. Also , DMs periodically .
8. NACK –Based Combo-Casting Approach : Since , each data packet has a unique sequence no. .Hence , if a packet is missed by a node , it sends a NACK to the DM . DM do not retransmits that packet immediately . As the DM gets an ACK / NACK for the last packet , DM analyses NACKs. On the basis of no. of NACKs, DM decides to retransmit the packet by unicasting or multicasting.

9. Buffer Management : Each DM has two buffers , one buffer for data packets , and another buffer for NACKs .Initially , the buffer capacity is assumed to be unlimited. But , the capacity is confined , as the first packet is received .

V. DESCRIPTION OF RSM2 MODEL

A. *Optimized Flooding Algorithm (OFA)*

In this model , we use flooding technique , but it has been optimized to overcome the drawbacks of Flooding . Each packet has a sequence no. When the DM gets the packet ,starts T_{stores} timer for t_{store} time . As the t_{store} time gets out , DM drops the packet , but save it's seq. No for ($2 * t_{store}$) some time. It prevents looping. As DM receives the same packet or packet with same sequence no. , then DM will discard it and not floods to the network. As, the DM gets updated information regarding packet or topology's change , then it immediately floods to the network.

```
Procedure OFA()
{  //  $t_{store} \leftarrow$ time to store the packet.
Step 1:  DM $\leftarrow$ getpacket();
Step 2: pck $\leftarrow$ no. of packets to be send in one session.
Step 3: For (i=1;i<=pck;i++)
{  $T_{store} = 20ns$.  // assume each incoming packet will be
stored in buffer for 20ns, first . While ($T_{store} > 0$)
{ Stores the packet;   If ($Packet_{in} == Packet_{store}$)
{ Discard the packet   Stop flooding ; }
        Else
 { store the packet  forward to connected node except that
from which it come }
{ Step 4: Drop the packet;
Step 5: Save the sequence no;
Step 6: If ($seq\_noPacket_{in} == seq\_noPacket_{empty}$)
          Discard the packet ;
Step 7: Else store the packet  and Go to Step-3.
```

B. NACK-Based Packet Recovery

RSM2 model opts NACK-Based packet recovery. The recipient will not acknowledge the DM(s) or sender for each received packet . It prevents the ACK-implosion problem. If sender misses a packet , then immediately send NACK for it. As DM gets NACK , will not serve immediately , wait for a random amount of time , and after that serves the request using Combo-casting. There are three possibilities ,as follows

- Possible Case- I :

As , the receiver missed the packet , it sends the NACK immediately. So, if NACK for Packet-5 comes to DM then it acknowledges the receipt of all the packets , sent before packet-5 .

- Possible Case-II :

DMs receives a request for that packet , which has been emptied out the buffer . In that case , DMs will wait for a random amount of time , and then sends this request to the recivers under them . If receiver (under DM) get that packet, will serve the request .Otherwise DMs will flood this request to other DMs , using optimized flooding . Incase , no one can serve this request , only then , the request will go to the sender.

- Possible Case-III :

As , the receiver gets the last packet of the sequence, it sends an Over-packet to DM.This packet acknowledges that , all the packets have been received successfully.

C. *NEED TO MODIFY PACKET STRUCTURE AND BUFFERING*

Dynamics Manager is the main focus of this model. Dynamics Manager's functionality makes available this model to work in both wired and wireless networks.
As DMs use two kinds of Buffers.

1. Data_buffer : The buffer stores all the data packet to send ,and do not get empty immediately.The buffer is emptied , after $T_{prio.}$ Stops.
2. Nack_buffer : This buffer stores the NACKs , received from receiver nodes. Since , NACKs are not served immediately , hence , we need this buffer

Since , after the priority path formation , some information is added to the header of the packet , discussed below . This information is used further in path maintenance and buffer maintenance :

1. Dseq_No. : it is the sequence no. , which is added to the packet , to distinguish it from others , and to assemble in proper order.
2. Path : this field contains the path , through which packet goes to the recipients. DMs will forward to next node , using this path .
3. More(m) /End(e) : In each packet , the last field either sets to 'm' or to 'e'.

- If the last field sets to 'm' : This field indicates how many packets will be sent after this packet .
- If the last field sets to 'e' :It indicates , it is the last packet of this session.

4. Priority : if the value of Priority is low , means lesser cost , hence for each incoming packet , DM will buffers the packet and starts . As T_{prio} gets out , packet will be dropped and buffer get emptied.

D. PRIORITY TIMER

The designing of Priority timer is based on Priority matrix. If the path has low cost , then it also has low priority . Then , for the low-cost path , the T_{prio} will be set to small time units .As , the T_{prio} gets out , the buffer will be emptied.
For the high-cost path , the T_{prio} will be set to large time units . As , the T_{prio} gets stop, the buffer will be emptied. Incase , the Tprio gets out and then , NACK comes for the Packet , then it will be served by its predecessor node or successor node . It might be the possibility , if no one can serve that request , then that request is served by Sender directly.

```
Priority Timer()
{  //Assume that if a link (connects two nodes-A and B) has
least cost ,  then it takes 10 ns from A to B or vice-versa.
          $T_{init} = 10ns$;
```

```
  p = priority assigned to the edge
 for(i=1; i<p; i++)
{ $T_{prio.} = i * T_{init}$ ; }
```

E. MINIMIZING CONGESTION DUE TO TRANSMISSION-ERROR

RSM2 model reduces the congestion due to packet retransmission .Since , we have introduced a new approach of NACK-buffer to fulfil NACK-Based recovery. Dynamic Manager first stores all the NACKS , and wait for a random amount of time, T_{NACK} .As , the time gets out , DMs retransmit the data via combo-casting.

F. COMBO-CASTING ALGORITHM

Dynamic Managers first stores all the NACKS , and wait for a random amount of time , T_{NACK}. As , the time gets out DMs decide either to retransmit the data through multicasting or via unicasting. If more NACKs are received for the same packet ,then DM subgroups them into a new group , and multicast the missed data-packet . In case , there is one or two nodes send NACK for the same packet , then , DMs unicast the missed data packet .

```
Procedure Combo-casting( )
{
   $T_{NACK} \longleftarrow$ Storage time for NACKs.
 while ($T_{NACK}$ >0)
  {
   do nothing , only stores the packet ;
  }
     if (i<2)
      {
           Unicast()
           {
            Transmit the data via unicasting ;
            }
      Else
       {
          Multicasting()
               {
            Transmit the data via multicasting;
               }
       }
```

G. DYNAMICS MANAGER

In RSM2 model , Dynamics Manager plays an important role. Dynamics Manager are the specialised Machines , with network computational capabilities . Dynamics Manager is the main focus of this model. Dynamics Manager's functionality makes available this model to work in both wired and wireless networks. Dynamics Managers act as listeners and calculater to perform network computations.
The importance of Dynamics Manager is described as follows:

- Sends Status report Periodically or at topology's update :
- Create Partial Matrix : Dynamics Manager on receiving the echo packet or getting the topology-updates , create the matrix .This matrix has the information of those nodes that are directly connected with this DM. Hence, this matrix is called Partial-Matrix.

$$P[i][j] = \{ \textit{Exclude Edge}(i,j), \textit{if } \forall (i,j) \in E \textit{ forms loop} \\ \textit{otherwise}, \textit{Assign Priority} \forall (i,j) \}$$

- Calculate Priority of the link : Dynamics Manager calculates the priority matrix from the work matrix using the Priority Constraint , as follows :

```
  Priority Matrix()
   {
 for (i=1; i<=n; i++)
    {
   for (j=1 ; j<=2 ;j++)
      {
If  (W[i][j] == Least_cost)
    P[i][j]  $\longleftarrow$  Min_Prio;
    Else (C[i][j] == High_cost)
    P[i][j]  $\longleftarrow$  Min_prio;
         }
```

- The Priority matrix is used to set the priority timer . Since the packet already contains the path and priority of each path. DMs gets the priority for the path , through which packet will reach to destination via DM(s). DM(s) .If DM finds the link has lower priority-value , then Tprio. will be in function for small unit of time. Since , the link has low cost and having less chances to miss the data , if passes through this link. So, as the Tprio gets out . data will be emptied out of the buffer.

VI. BASIS FOR PROPOSED ALGORITHM

- DESIGN OF COST MATRIX :

The cost matrix is designed to define the cost between each pair of nodes , with the following cost constraint :

C[i][j] = { INFINITY, iff (i , j) do not belongs to
set , E V ($\forall$ (i , j),i = j.
Otherwise , Assign Cost }

- DESIGN OF WORK MATRIX :

The work matrix is designed to define the priority to the edges , with the following work constraint :
W[i][j] = { Leave Edge (i´, j´), if (i= j´) Λ (j=i´),
$\forall$ (i , j) Λ (i´, j´) є E
Otherwise, Include Edge (i´,j´)}

- **DESIGN OF PRIORITY MATRIX :**

The priority matrix is designed to define the priority to the edges , with the following priority constraint :

P[i][j] = { Exclude Edge (i , j) , if
$\forall$ (i , j) є E forms Loop
Otherwise , Assign Priority $\forall$ (i , j)}

DESIGN OF DATA-PATH :

The data delivery path is designed to deliver the data to the desirable recipients .Desirable recipients belongs to the specific group(s) , to whom Sender wants to communicate .

Data path $\leftarrow$ Choose (u,v) є {specific group(s)}.(u,v) є E.

- COMBO-CASTING :

Dynamic Managers first stores all the NACKs , and wait for a random amount of time , T_{NACK} . As , the time gets out DMs decide either to retransmit the data through multicasting or via unicasting. If more NACKs are received for the same packet ,then DM subgroups them into a new group , and multicast the missed data-packet . In case , there is one or two nodes send NACK for the same packet , then , DMs unicast the missed data packet .

- PRIORITY TIMER MANAGEMENT :

The Priority matrix is used to set the priority timer . Since the packet already contains the path and priority of each path. DMs gets the priority for the path , through which packet will reach to destination via DM(s). DM(s) .If DM finds the link has lower priority-value , then $T_{prio.}$ will be in function for small unit of time. Since , the link has low cost and having less chances to miss the data , if passes through this link. So, as the T_{prio} gets out . data will be emptied out of the buffer.

- **OPTIMIZED FLOODING ALGORITHM (OFA)**

When the DM gets the packet ,starts T_{stores} timer for t_{store} time . As the t_{store} time gets out , DM drops the packet , but save it's seq. No for (2 t_{store}) some time. It prevents looping. As DM receives the same packet or packet with same sequence no. , then DM will discard it and not floods to the network.
As , the DM gets updated information regarding packet or topology's change , then it immediately floods to the network.

VII. PROPOSED ALGORITHM (MCPA)

Any reliable multicast protocol requires some recovery mechanism. A generic description of a recovery mechanism consists of a prioritized list of recovery servers/receivers (clients), hierarchically and/or geographically and/or randomly organized. Recovery requests are sent to the recovery clients on the list one-by-one until the recovery effort is successful. There are many recovery strategies available in literature fitting the generic description [5]. MCPA (Minimum – Cost Path Algorithm) is introduced as a new step in this field.

```
Procedure MCPA()
  {
// d  ← no. of Dynamics Managers.
// E  ← set of edges.
// S  ← set of senders.
// n  ← No. of nodes.

Step1: Sender sends ECHO-packet.

Step 2: DMs forms partial-cost matrix.

    Create_partial_matrix()
    {
    For ( i=1 ; i<=n; i++)
      {
      For (j=1; j<=n ;j++)
         {
           If (i==j OR No edge between i and j)
          C[i][j] ← INF ;
         else
          C[i][j] ← Cost of the link ;
         }
       }
     }
    Step 3: flood_partial_ matrix()

         {
  for( i=1;i<=d; i++)
           {

            While (next_node is not sender)
                 {
          Next_node ← Partial-Cost Matrix;
                 }
               }
             }
      Step 3.1 If topology ← updates
                  Goto step 2.

      Step 3.2 Else
                  Go to step 4.
```

Step 4: Sender constructs the final cost matrix using , the Partial-Cost matrix.

Step 5 : Final cost matrix is given to a chosen DM. Choice depends on what criteria (cost , distance .....etc.) is selected by the sender.

Step 6: At Dynamics Manager's site :

Step 6.1 : Work-Matrix is created using Cost-Matrix , with following work-constraint.

W[i][j] = { Leave Edge (i′, j′), if (i= j′) Λ (j=i′),
$\forall$ (i , j) Λ(i′, j′)є E
Otherwise, Include Edge (i′,j′)}

```
        for (i=1; i<=n; i++)
          {
        for (j=1 ; j<=2 ;j++)
        {
       i ← No. of edges;
       j1 ← Connected Edge;
       j2 ← Cost to edge;
        }
     }
```

Step 6.2 : Heap Sort the WorkMatrix on the basis of cost.

Step 6.3: Create the priority matrix , with following Priority constraint :

```
 P[i][j] = { Exclude Edge (i , j) , if
                    ∀ (i , j) ϵ E forms Loop
       Otherwise , Assign Priority ∀ (i , j)}
```

```
for (i=1; i<=n; i++)
          {
         for (j=1 ; j<=2 ;j++)
            {
            Create the priority matrix
               If (W[i][j] ==Least_cost)
                    P[i][j] ← Min_Prio;
             Else (C[i][j]==High_cost)
              P[i][j] ← Min_prio;
              }
          }
```

Step 6.4 : the priority matrix is sent to Sender .

Step7 : At Sender :Create Data Delivery path using Priority Matrix.

Data path ← Choose (u,v) ϵ {specific group(s)}.(u,v) ϵ E.

Step 8 : Packet is created and forwarded according to the path mentioned in header of the packet

Step8.1 : DMs run Priority Timer() algorithm for Data-buffer maintenance on the basis of priority.

```
  Priority Timer()
         { //Assume that if a link (connects two nodes-A and
         B) has least cost , then it takes 10 ns from A to B or
         vice-versa.
          $T_{init}$ = 10ns;
           p = priority assigned to the edge
          for(i=1; i<p; i++)
         {
         $T_{prio.}$ = i* $T_{init}$ ;
          }
          }
```

Step 8.2 : DMs run Combo-casting algo() for NACK-serving.

```
     If( NACK received before the buffer empties the
missed packet)
      {
     Procedure Combo-casting( )
         {
            $t_{NACK}$ ← Storage time for NACKs.
            while ( $t_{NACK}$ >0)
              {
              do nothing , only stores the NACK ;
                }
             if (i<2)
                 Unicast()
             Else
                 Multicasting()
          Else
           {
          DMs will flood that request to all DMs , and set its
timer $t_{NACK}$.
            If time gets out and no packet receives , then DM will
send NACK to Original Sender directly .
  Else go to step- 8.1.
        }
       }     } //END
```

VIII. COMPARISON OF RSM2 WITH RMTP

There is a wealth of literature on reliable multicasting. Several new papers have also appeared in the recent literature that focus on Wide Area networks. [1] describes the design of Reliable Multicast Transport Protocol , that use efficient local-recovery technique for serving the missing packets . Our work is closely related to RMTP , with significant differences .Let us study in details the comparative study of RSM2 over RMTP.

1. ***Design*** : RSM2 opts flat design approach while RMTP works on hierarchical design . Flat design facilitates RSM2 to work in all the kinds of situation i.e.
   - One-to-Many : RSM2 fits in the situation , where one sender communicates with many receivers on the network. RMTP is also good in this scenario .
   - Many-to-Many : RSM2 fits in this situation , where many senders want to starts communication at the same time .RMTP does not fit in this scenario , it opts hierarchical approach , and in hierarchical , at one time only one node be the root of that hierarchy , i.e. the original sender.
   - All-to-All : RSM2 fits in this situation , where all the nodes at the network wants to go in communication , with each other .Flat design approach facilitates to work in all the three scenarios. In flat design , there is no root and no leaf .RMTP again does not fit in this scenario because of it's hierarchical design .since As, the no. of senders will start the communication , simultaneously, then a dilemma will occur about , who will be the root of that hierarchy .
2. ***Best Path***: RSM2 provides best path for the packet delivery . RSM2 use Kruskal's Algorithm for minimum spanning path , to reach to the all presentable nodes on the network. This path is further reduced if it includes some undesirable nodes. So, the data-packet is transmitted over short path , with minimum cost .But , in RMTP these concepts are not introduced. In RMTP , data is just multicasted without concerning cost or delay of the links , and the responsibility of sender is transferred to DR , i.e. Designated Receiver .

3. ***Dynamics Cooperativity*** : RSM2 model is designed in such a way , to work in dynamics also. RSM2 works well in Wired Networks and in Infrastured Wireless Networks also. Working of RSM2 in Infrasture-less Wireless Networks is the part of future scope. Hence , RSM2 is capable to cooperate dynamics well. RMTP works good in static environment only and do not cooperate well with dynamics of nodes . Since , in RMTP designated receivers are chosen statically , based on approximate location of receivers . So, it halts in wireless networks . Hence , RMTP does not provide Dynamics Cooperativity.
4. ***Combo-Casting***: RSM2 use Combo-Casting to serve NACKs hence reduces duplication of packets effectively. Dynamic Managers first stores all the NACKs , and wait for a random amount of time , TNACK . As , the time gets out DMs decide either to retransmit the data through multicasting or via unicasting. If more NACKs are received for the same packet , then DM subgroups them into a new group , and multicast the missed data-packet . In case , there is one or two nodes send NACK for the same packet , then , DMs unicast the missed data packet .In RMTP , Designated Receiver buffers the packet and if NACK comes for a missing packet , then it is first served by the DR , who is looking after that area .If this DR , becomes unable to serve that request , then request will go to it's Parent DR and so on. Finally reach to the sender , if no DR serves that request .

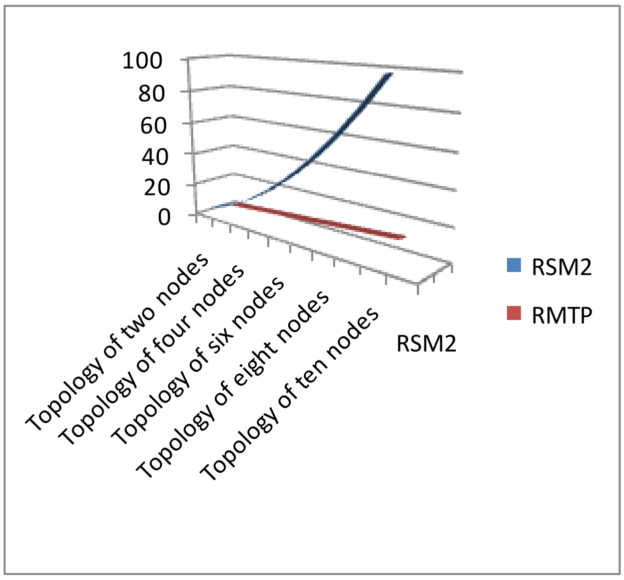

Fig : Complexity of RSM2 Vs. RMTP over the no. of nodes in network topology.

REFERENCES

1. Sanjoy Paul, Member, IEEE, Krishan K. Sabnani, Fellow, IEEE, John C.-H. Lin, and Supratik Bhattacharyya " Reliable Multicast Transport Protocol (RMTP) ".IEEE journal on Selected Areas in Communications , Vol. 15, No. 3, April 1997.
2. Ali Alsaih and Tariq Alahdal. ,"Non-Real Time Reliable Multicast Protocol Using Sub-Sub Casting ," The International Arab Journal of Information Technology , Vol. 4 , No. 1 , January 2007.
3. Jim Gemmell, Jorg Liebeherr, Dave Bassett ,"An API for Scalable Reliable Multicast".
4. Tie Liao , "Light-weight Reliable Multicast Protocol" ,INRIA , Rocquencourt , BP 105 ,78153 Le Chesnay Cedex, France .
5. Danyang Zhang , Sibabrata Ray , Rajgopal Kannan , S. Sitharama Iyengar "A Recovery Algorithm for Reliable Multicasting in reliable networks." Proceedings of the 2003 International Conference on Parallel Processing (ICPP'03).

AUTHOR PROFILE

Ruchi Gupta , M-Tech Scholar , Computer Sc. Department , Krishna Engineering College , MohanNagar, .GZB . India. Currently working as an Assistant Professor ,C.S Deptt.,in Mewar University , Rajasthan. India.

Pramod Kumar Sethi , Assistant Professor in Computer Science Department , Krishna Engineering College , MohanNagar , Ghaziabad , India.