# A DELAY – CONSTRAINED IN MULTICAST ROUTING USING JIA ALGORITHM

Akhilesh Kumar, Deepa Verma, Rakesh Kumar, and Ashish Kumar Srivastava

Rajkiya Engineering College, Ambedkar Nagar, U.P

## *ABSTRACT*

*The Distributed multicast routing protocol under delay constraints, is one of the software, which requires simultaneous transmission of message from a source to a group of destinations within specified time delay. For example. Video Conferencing system. Multicast routing is to find a routing tree which is routed from the source and contains all the destinations. The principle goal of multicast routing is to minimize the network cost. A tree with minimal overall cost is called a Steiner tree. Finding such tree is the principle of the NP complete.*

*Many inexpensive heuristic algorithms have been proposed for the Steiner tree problem. However, most of the proposed algorithms are centralized in nature. Centralized algorithm requires a central node to be responsible for computing the tree and this central node must have full knowledge about the global network. But, this is not practical in large networks. Therefore, existing algorithms suffer from the drawback such as heavy communication cost, long connection setup time and poor quality of produced routing trees. So far, a little work has been done on finding delay bounded Steiner tree in a distributed manner. An effort is made in this paper to this effect. The Study reveals that the drawbacks mentioned above has been sufficiently reduced. This paper gives complete guidelines for authors submitting papers for the AIRCC Journals.*

## 1. INTRODUCTION

Multicast is a kind of group communication which requires simultaneous transmission of message from source to group of destination. Real time multicast refers to a multicast in which a message should receive by all destinations within specified time delay. Here are many applications relying on time multicast services. For example, in a video conferencing system, each participant interacts with the other by sending and receiving message. Each message, originating from a participant, must be delivered to all the other in a real time manner.

In real time communication, a connection (logical connection) from the source to the destination(s) has to be established before any data transmission occurs. During the connection setup, sufficient network resources (i.e. network bandwidth, buffer etc.) are received at each network node on the connections, so user QOS (Quality of Service) can be generated at run time.
Routing is an important step of connection setup. It concerns selecting a route from the source to destination(s) on which the connection will be established. MULTICAST routing is to find a route tree which is routed from the source and contains all the destinations. Because most networks nowadays have the feature of bandwidth sharing when transmitting same message to multiple destinations along common segment of path only one copy of message is needed to flow through the common part of the path to multiple destinations. Therefore, choosing a proper routing tree, which maximizes the path sharing, can significantly reduce the bandwidth consumption of the multicasting. We define the network cost of routing tree as the distance of all the links in the tree, which is total distance a multicast message should travel to reach all the

destinations. One principle goal of multicast routing is to minimize the cost. A tree with minimal overall cost is called a Steiner tree. Finding such tree is problem of NP complete.

Many real applications also have a time constraint, which requires communication to be done within pre specified delay bound. Therefore multicast routing for the real time application should generate a routing tree in which delay from the source (the root) to any destination does not exceed the bound. Finding such a tree which has minimal network cost under a delay constraint is called delay bound Steiner's tree problem.

A lot of research has been done on the Steiner tree problem many inexpensive heuristic algorithms have been proposed. However, most of the proposed algorithms are centralized in nature. Centralized algorithms require a central node to be responsible for computing the tree and this central node must have full knowledge about the global network. This is not practical in large networks. However, existing distributed algorithms suffer from the drawback such as heavy communication cost, long connection setup time and poor quality of the produced routing trees. So far, little work has been done on finding delay bounded Steiner tree in a distributed manner.
In the project the work on a distributed protocol of multicast routing for real time applications has been developed. The protocol has the following advantages:

1. Fully Distributed: - Each node operates based on its routing information and coordination with other nodes is done via network message passing.

2. Near optimal routing trees under delay bound: - The network cost of the tree is close to optimal under the condition that the delay from a source to any destination along the routing tree does not exceed a pre-specified bound.

3. Low communication cost and flexible in dynamic membership changes. It takes a small number of network messages for routing and the multicast membership can be dynamically be changed without effecting the existing traffic on connection.

## 2. SYSTEM MODEL AND PROBLEM SPECIFICATION

### Real-time Optimal Multicast Routing

The network is modeled by a connected graph *G (V, E),* where the nodes in the graph represent communication endpoints and the edges represent links. For each link *e € E, d (e)* is the distance of the link. It can be the number of hops in wide area networks. the network delay of a link is proportional to the distance of the link. Thus *d (e)* is also used to indicate the delay (the unit of time required for traveling distance *d (e))* of *e.*

Given a source node *s € V,* a set of destination nodes D c *V,* with $s \notin$ D, a routing tree for multicast connections is a sub tree of the graph *G (V, E)* rooted from s, that contains all of the nodes of *D* and an arbitrary subset of *(V -D),* whose leaf set consists only of a subset of nodes of *D.* When multicasting a message to D, node s sends a copy of the message to each child of s in the tree. These children in turn transmit the message to their children until all nodes in the tree (thus, all nodes in *D)* have received the message.

According to the nature of a tree, a multicast message will flow through each branch of the routing tree once and only once to reach all the destinations. Therefore the bandwidth consumption of a multicasting is in proportion to the sum of the distance of all links in the tree. The network cost of a routing tree T is defined as:

$$\text{Network Cost (T)} = \sum_{e \in T} d(e) \quad \text{.........(1)}$$

One objective of multicast routing is to optimize this network cost, so that the bandwidth consumption for each multicast will be minimal. -This is particularly important for multimedia applications where messages containing audio or video files are usually very large in size. The bandwidth saving by using a good routing tree can be very significant in these cases. In addition to the requirement of network cost, many interactive multimedia applications have a stringent delay constraint. It is often required that the delay from the source s to any destination should be within a time bound A. Suppose *P(s,u), ED,* is the path from s to U along the routing tree *T',* then the bounded delay requirement can be expressed as:

$$\square u \in D: \sum_{e \in P(s,u)} d(e) < \Delta \quad \text{.........(2)}$$

Related Work

The network cost and delay requirements of multicast routing tree often conflict with each other. A Steiner tree with optimal network cost may have a long delay to the farthest destination. Shortest path tree (SPT), in which each path from the source to a destination is a shortest path (in terms of network delay), has the shortest delay, but it may incur a high network cost. A tradeoff algorithm between optimal network cost and minimum average delay was proposed by Kumar *et a1* in [3]. It generates two routing trees, a shortest path tree T and a Steiner tree T. It performs *k* (a positive integer specified by the user) degree tradeoff by first identifying out *k* destinations to whom the difference between the delay in T and the delay in *7"* is the largest. Then it replaces the paths to the *k* destinations in *T* by their shortest paths in T. Therefore it obtains a less optimal routing T', but with a shorter average delay. This idea can be easily extended to delay bounded routing by simply replacing the paths to the nodes whose delay exceeds the time bound by their shortest paths.

In real-time applications, there is usually no need to minimize the average delay to all destinations. It is often required that the delay to any destinations should be within a bound. The delay bounded Steiner trees were investigated by Kompella *et a1* in [14]. It proposed an algorithm based on MST (minimum spanning tree) heuristic, generating a routing tree starting from the source s. Each time of expanding the tree, it proposed two heuristics to select the next non-tree node *v* to be included into the tree. One is called the *cost delay heuristic*, which uses a cost and delay of link into weight. The weight function is

$$w[u, v]=c[u, v]/(\Delta-(D[u]+t[u, v]))$$
$$\text{If } (D[u] + t[u, v]) < \Delta$$
$$w[u, v]=\infty$$

Otherwise

Where C and T is the cost matrix and delay matrix respectively, D[u] is the delay from s to u along the tree. Then simply adds node v that has the minimal weight to a tree node. This heuristic, although producing a delay bound tree, has a tendency to optimize on delay. It may find paths with delays far lower than, at the expense of adding cost to the tree.

The other heuristic, called cost heuristic is to use MST algorithm directly. Each time it simply adds node v whose cost to the tree is minimal and condition D[v] <     , holds into the tree. This heuristic is too greedy initially when D[v] is small. It may fail to find a delay bounded routing tree, though it does exists, due to bad tree structure constructed at the beginning.

As we mentioned before, distributed routing algorithms are much more practical in large networks. SPT can be easily implemented by Bellman-Ford's routing algorithm [2, 9] in a distributed manner. Some other wide accepted distributed solutions to Steiner tree problem are based on MST heuristics. A MST heuristic is to generate a MST of the network graph G (E, V), spanning all nodes in V. A Steiner tree (approximate) is then obtained by removing, from MST, sub trees containing no nodes in {s} U D. Some distributed MST algorithms are introduced to generate Steiner tree in a distributed way. However, in distributed MST heuristics, all the nodes in the network should participate in the execution of algorithms, which is costly in large networks.

## 3. JIA ALGORITHMS FOR MULTICAST TREE CONSTRUCTION

### 3.1 Assumption and Basic Idea

The idea of Jia's algorithm in constructing routing tree mimics Prim's MST algorithm and is in combination with a distributed shortest path algorithm. Here each node has information about the shortest path to every other node in graph. This can be achieved by running the Dijkstra or Distance vector Algorithm.

Furthermore there are no long live loops in the shortest paths. Several loop free shortest path routing algorithm based on Bellman-Fords algorithm have already been proposed. Since dynamic change of routing information may still causes transient loops in shortest path routing, our routing protocol simply terminates when a loop is encountered.

The basic idea of the protocol works as follows, the construction of a routing tree starts with a tree containing only the source S. A destination in D is selected which is the closest to the tree and for which the end to end delay from S to T along the tree in under the bound. The shortest path from the tree to this selected destination is added into the tree. By adding a path to a tree all nodes on the path are included into the tree. At each step, an unused destination, which is the close to the tree under the delay condition, is added to the tree. This operation repeats until all destinations are in the tree.

### 3.2 Algorithm Details

To record the distances from the tree to destination during the construction of the tree a table 4.1 (Tree to destination) is used. The distance from the tree to destination is the shortest distance between the destination and tree node closed to it. Each destination has entry in table 4.1, which is four tuple (destination, distance, tree node, tag), including that the distance from tree to destination is distance and the tree node is the tree node closed to the destination. The tag field indicates whether destination is in tree or not. The table 4.1 is initialized at source node for each request of multicast stop. At this point of time when f is added in the tree, the data in table 4.1 is as follows:

Table 4.1 Tree to Destination table

| Destination | Distance | Tree-node | Tag |
|---|---|---|---|
| C | 2 | A | Yes |
| E | 3 | A | No |
| F | 1 | J | Yes |
| J | 1 | B | Yes |

To check if the delay from the source to destination along the trees exceeds a bound, each tree node records the accumulated delay from root to itself along the tree. Therefore end-to-end delay from source to a detonation via a tree node is accumulated delay to this node plus the delay from this node to that destination the distance from node to any destination can be found from local routing table. Every node in the system execute finite state machine in figure 1.

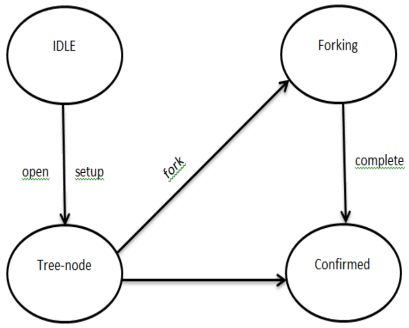


Figure 1: Finite State machine of the protocol

The finite state machine has four states idle, tree-node, forking and confirmed. All the nodes are initially in idle state, waiting for connection setup requests. When a node is included into the routing tree, it entered the tree node state. A tree node enters the forking state when it is requested to fork a branch to another destination. When tree node received a completion message, indicating success of the connection setup, its state changed to confirmed.

When nodes receives an open request for setting up a multicast connection, with parameter such as destination set D and a delay bound Δ, it becomes source node s for this multicast connection. This routing tree initially contains only one tree node s. A table 3.1 initialized for this connection setup only request. Then, a destination which is closest to the tree under delay bound conditions is chosen as first destination to be linked to the tree.

A setup request message which carries the table 4.1 and other information of this connection setup request is send to the next neighbor node via which the chosen destination can be reached. When an intermediate node (a node not in D) receives the setup request, it becomes a tree node (in the tree state of finite state machine). It updates table 4.1, if the destination from the node is shorter than that in the table 4.1 and end to end delay from source via this node is under delay bound Δ. Then it passes the setup request to the next neighbor.

Leading to the chosen destinations, if still the destination is not in the tree, the next destination which closed to the tree under delay bound is selected as next one to be linked to the tree. A fork request is send to the tree node, which is closest to next destination. The selection of the next destination and its fork node is based on the information in the table 4.1.

When the node receives the fork request, it becomes a fork node in the tree. It continues the operation of constructing the routing tree by sending setup request to the neighbor leading to the designed destination in D i.e. all the destination is now in tree. It sends a completion message to source node s, s then propagates this completion request to the entire destination by using the newly constructed routing tree. This real time multicast connection is confirmed as each node in the tree.

Example

Following example shows how the protocol works and how a delay bound routing tree is constructed in distributed fashion Fig 2 is a network graph, which follows the topology of early stage Arpanet. In this example, node is the, multicast sources, D={c, e, f, j}, and $\underline{\Delta}$=3.

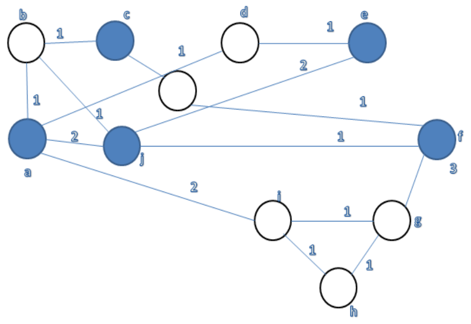


Fig 2. A simple network technology

The construction of the routing tree start from source node a, destination e is chosen to be first linked into the tree. After e is included in the tree, tree now contains path <a, b, c>. Node j is the next destination closed to the tree and the tree node b is the fork node to link j to the tree. Destination j is included in to the tree by path <b, J> followed by destination f in the same way. The routing tree constructed so far is shown in fig3.

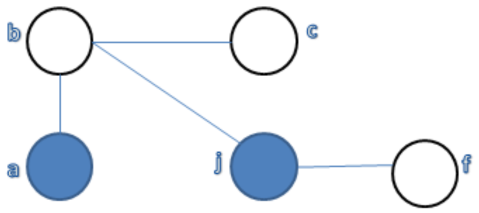


Fig.3 A routing Tree constructed so far

When considering destination e, although it is closed tree node j, and the end-to-end. Delay from a to it via j is 4≤ Δ. The shortest path from the tree to e under the delay bound is the shortest path from source a to it. Thus the destination e is linked to the tree by path <a, b, c>. Fig 4 is the final routing tree, which contains all the destinations.

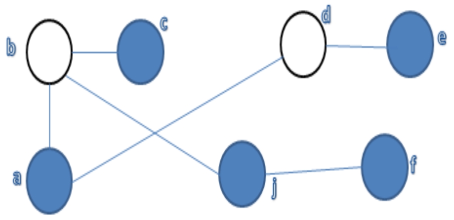


Fig 4. The final routing tree for D={c, e, f, j}

Dynamic multicast membership changes:

In many applications, multicast participants are free to leave or join multicast session dynamically. It is very important to insure that any change of multicast membership will not affect the traffic on the current connection, and the routing tree after the change remains near optimal in terms of network cost.

A widely referenced example of group management protocol is Internet Group Management Protocol (IGMP), which has become a standard of group management for internet multicast. Due to the connectionless nature of internet, there is no logical connection among the group members and thus IGMP does not consider the network cost of a connection. In connection oriented communication, there is a connection for each multicast group and this connection needs to be reconstructed and members dynamically leave or join the group.

In the Project, when a destination requests to leave a multicast group. It is disconnected from the connection. If the destinations a leaf node of routing tree, the disconnection can be easily done. A leave request is sent upward (to the root direction) along the tree, node-by-node, until it reaches a fork node or another destination. At each node, this request passes through, the connection is released. As the result, the destination is disconnected from the routing tree, while rest connection remains intact. If the destination is not a leaf node that is, the destination is also responsible for forwarding multicast messages to another destinations, in this it simply changes the function of this node to perform only the switching function, which switches incoming messages to out ports of this connection. The applications are not affected.

When nodes want to join an existing multicast group, it sends a request to source of the multicast. The source multicast join request to all destinations through the multicast connection. The request message carries a 3-tuple (new-dest, distance, and tree-node) to record the distance from the tree to new destination. At each node this request passes through it modifies the distance and the tree ode information, if it has a shortest path to the new destination under delay bound. When request arrives at leaf node along the tree, the leaf node sends the destination tree node information back to the source. After collecting replies from all the leaf nodes, the source knows the tree node to extend the connection is unaffected and the routing tree still keep near optimal after the new member joins in.

## 4. THE PROGRAM ORGANISATION AND FLOWCHART

The program simulates the Jia's algorithm to construct multicast routing tree under delay bound constrain (real time multicasting). This algorithm is compared against shortest path algorithm in the program. Various parameters such as network topology, delay bound, source and set of destinations are read from a text file. Topology for a delay bound tree and shortest path tree is output in a text file. The topology is stored in the form of 2-d array.

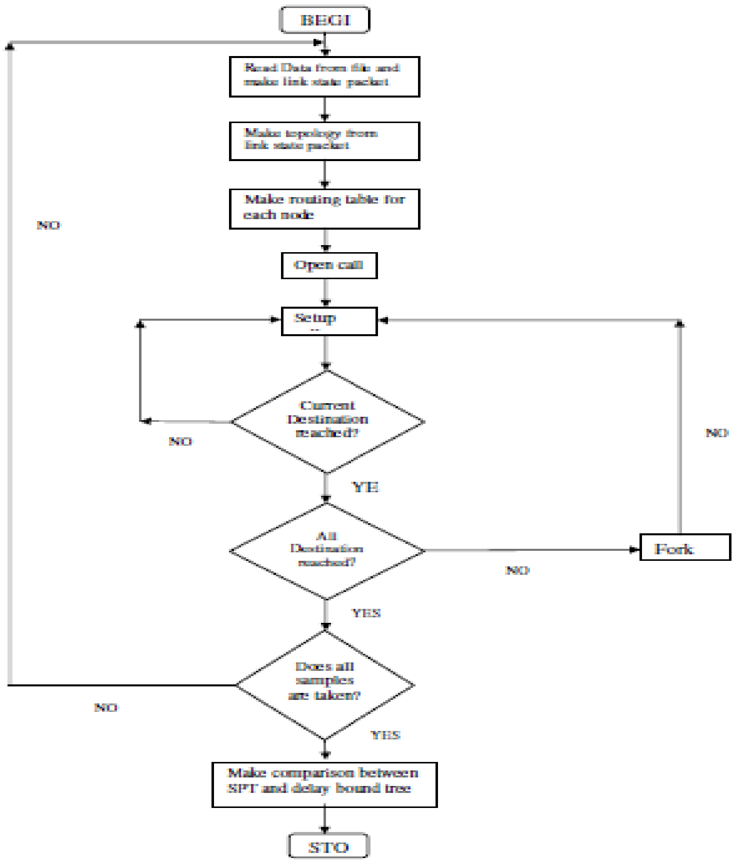
The program is developed using object oriented methodology. The whole network is considered as an object. Each node is also considered as an object. The network object contains number of node object, equal to the number of nodes in the network topology. The variables and objects which are passed across the network, during the construction of tree are taken as global variables in the stimulation, so that each node can access those variables and objects.

To perform various operations on the tree (Delay Bound Tree & Shortest Path Tree), a separate object tree has been created which has basic operations of the tree as its functions. The tree constructions algorithms call these functions to construct the tree. The tree is dynamic in nature. The nodes of tree are created as required from the memory heap. All the sons of a node are linked in a linear list. The node contains a pointer to its first sons; other sons are linked through brother pointer to its father. Various operations of the tree are performed using pointers.

In the main program the number of network objects is created which take their network topology, delay bound, source and destination from their respective file. To stimulate the real network each node makes link state packets and from this link state packets each node can get network topology.

Routing tables for each node in the network is generated using Dijkstra algorithm. Routing table contains distance and next hop for each node.

After this Jia's algorithm for constructing a delay bound tree is called. Jia's algorithm makes Opencall to the source node of multicast. During Opencall the node initializes table 3.1 and first destination is selected from the table and set as a Current destination. It set the value of NextHop as its neighbor leading to Current destination. For this Nexthop for which the setupcall will be called. The Jia's algorithm makes the setupcall for the nexthop until current destination is reached. The setup call update the table 4.1, if the distance from this node to the destination node is shorter that the tree to destination and end-to-end delay from source to destination via this node is under delay bound.


BEGI
Read Data from file and make link state packet
Make topology from link state packet
Make routing table for each node
Open call
Setup
Current Destination reached?
NO
YE
All Destination reached?
NO
Fork
NO
YES
Does all samples are taken?
NO
YES
NO
Make comparison between SPT and delay bound tree
STO

It makes the neighbor leading to currentDestination as nexthop for which setupcall will be called. If current destination is reached and if there are still destination not in tree, the destination which is the closet to the tree under the delay bound is selected as current destination( next node to be linked to the tree). The tree node of this current destination( in table 4.1) becomes fork node and sets the nexthop as its neighbor leading to current destination for which setup will be called.
When the destination are reached completion call is sent to all nodes in the tree. In this way the complete delay bound tree for the given destination is completed. Now for the comparison purpose shortest path tree routed at source is constructed. The tree contains only the multicast

destination node as its leaves. After this comparison is plotted between weight of shortest path tree and weight of delay bound tree.

## 5. CONCLUSION

Most of the algorithms available for the multicast routing are centralized and because of which routing takes longer times specially when the network is large. A distributed delay bound ulticast routing protocol has been studied in this paper and a software package has been developed. In addition to the distributed nature and low execution cost of the protocol, it has the following special features:

- It combines routing and connection setup as single phase of operations, which can significantly reduce the time of the connection setup.
- It allows dynamic change of memberships without affecting the existing traffic on the connection, which is very desirable for many multicast applications.
- It generates a good quality of routing trees which have no network cost and the performance is stable

The software developed generated routing trees sequentially, that is , it extends the routing tree to one destination after another. If the multicast group is very large, the connection setup time could be long, the communication cost could be high and quality of produced routing tree could be poor. So far, little work has been done on finding delay bounded Steiner tree in a distributed manner to overcome the above drawbacks in case of large network. The study done in this papers takes into account the above factors and reduces the cost significantly. In multimedia applications and in many other modern network applications, the group size of the multicasts usually are small. In these cases also, this software can be used to produce routing trees in a reasonable setup time, thus saving the cost.

The work embodied in this paper provides guidelines to researchers in the area.

## REFERENCES

[1] Xiaohua Jia, Yanchun Zhang, Niki pissinou, Kia Makki, " A Distibuted multicast routing protocol for real time multicast routing application" IEEE, Tran. COMPUTER NETWORKS (April 1995) 193-203.

[2] F. Bauer, A.Verma, Distributed Algorithm for multicast path in data network, IEEE Trans. Networking 4(2) 1996 181-191.

[3] K.Bharth Kumar, J.M Jaffe, Routing to multiple destinations in computer networks IEEE Trans. Comm. (March 1983) 343-351.

[4] Andrew S. Tenenbaum" Computer Networks" Third Editon

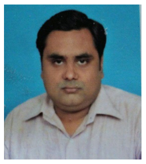
[5] Kelvin C. Almeroth, The Evolution of Multicast IEEE network, Jan-Feb 2000(10-35)

[6] Dimitri Bertseks, Robert Gallager DATA NETWORKS, Second Edition

[7] Stallings, DATA NETWORK, Fifth Edition.

## AUTHORS

**Akhilesh Kumar** graduated from Mahatma Ghandhi Mission's college of Engg. and technology, Noida, Uttar Pradesh in Computer Science & Engineering in 2010. He has been M.Tech in the department of Computer Science & Engineering, Kamla Nehru Institute of Technology, Sultanpur (Uttar Pradesh). SinceAugust2012, he has been with the Department of Department of Information Technology, Rajkiya EngineeringCollege, Ambedkar Nagar, as an Assistant Professor.His area of interests includes Computer Networks and Mobile ad-hoc Nerwork.



**Ashish kumar shrivastava** is lecturer in Computer Science &Engineering Department and become member of different committeesACS, Paper setter, External Examiner And since 2012, he has been with the Department of Information Technology, Rajkiya Engineering college, Ambedkarnagar (UP) as an Assistant professor and hold various post like Dean Academic affairs, Center superintendent, Member of Proctorial board etc. His current research areas are in Biometrics system, Multicast security, Face recognition system, Bio-Medical Multimodal.

**Rakesh Kumar** was born in Bulandshahr (U.P.), India, in 1984. He received the B.Tech. degree in Information Technology from Kamla Nehru Institute of Technology, Sultanpur (U.P.), India, in 2007, and the M.Tech. degrees in ICT Specialization with Software Engineering from the Gautam Buddha University, Greater Noida, Gautam Budh Nagar, Uttar Pradesh, India, in 2012.In 2007, he joined the Quantum Technology, New Delhi as a Software Engineer and Since August 2012, he has been with the Department of Department of Information Technology, Rajkiya Engineering College, Ambedkar Nagar, as an Assistant Professor. His current research interests include Computer Network, Multicast Security, Sensor Network and data mining. He is a Life Member of the Indian Society for Technical Education (ISTE), and he is a Nominee Member of Computer society ofIndia.