

Novel Protocol for Multicast Congestion Control in Ad Hoc Network

Oindrilla Rajaraw Ghosh*, V.S. Shankar Sriram**, G.Sahoo

*mishu_5@rediffmail.com , **sriram@bitmesra.ac.in , ***drgsahoo@yahoo.com

Department of Computer Science & Engineering Birla Institute of Technology, Mesra, Ranchi

Abstract-Multicast routing in wireless ad hoc networks has gain considerable attention in recent years due to its applications in mission critical scenarios, such as military operations and disaster recovery. In this article, a multicast routing protocol for ad hoc network which addresses both robustness and efficiency in one framework is presented. It is a mesh based multicast scheme instead of tree based multicast scheme. In On-Demand Multicast routing protocol (ODMRP), the control infrastructure is coupled with data forwarding infrastructure whereas by Multicast Core Extraction Distributed Ad Hoc Routing (MCEDAR) approach, congestion overhead is achieved. It allows very low overhead control infrastructure with efficient data forwarding. In this, it provides the robustness of the mesh based multicast infrastructure while still approximating the efficiency of a tree based multicast forwarding protocol. ODMRP is then extended to facilitate the congestion control using MCEDAR to combat the performance of the highly congested network.

Keywords: Multicast, Ad hoc Network, Congestion control, Routing.

I. INTRODUCTION

Multicasting has emerged as one of the most focused areas in the field of networking. As the technology and popularity of the Internet have grown, applications that require multicasting (e.g., video conferencing) are becoming more widespread. Another interesting recent development has been the emergence of dynamically reconfigurable wireless ad hoc networks to interconnect mobile users for applications ranging from disaster recovery to distributed collaborative computing. Multicast plays a key role in ad hoc networks because of the notion of terms and the need to show data/images to hold conferences among them. Ad hoc networks are multi-hop wireless networks that are characterized by scarce resources and dynamic topologies. The inherent characteristics of ad hoc networks place two key demands on any routing protocol (unicast and multicast) designed for such environments: *efficiency* and *robustness*. The scarcity of the resources entail a very efficient low overhead protocol while the dynamic topology entails a robust protocol that does not require frequent route recomputations (or does route recomputation in a lightweight manner). The main benefit of multicasting is the significant reduction of network load gained when packets need to be transmitted to a group of nodes. Congestion control (at the network level) is vital in multicast since the most important form of congestion control in the Internet, TCP, is not practical in multicast due to ACK implosion. Moreover,

overload control is essential in wireless networks where scarce bandwidth is the norm.

Some other multicast routing protocol include AMRoute [7], AMRIS [9], CAMP [10], multicast AODV

[8], and the On- Demand Multicast Routing Protocol (ODMRP) [1][2]. ODMRP broadcast packets in a mesh structure instead of tree structure. By using mesh, it introduces redundancy to fight packet loss in ad hoc network where channel noise, collisions and mobility are common. Under low traffic, ODMRP performs well. However, under high traffic it suffers from network congestion. This suggests that some form of congestion control is necessary to support reliable multicast.

In On- Demand Multicast Routing Protocol (ODMRP)[1][2], the data forwarding and the control infrastructure are coupled. But by using Multicast Core-Extraction Distributed Ad hoc routing (MCEDAR) [3], decoupling of the control infrastructure and data forwarding infrastructure are achieved. In ODMRP, the forwarding groups are created and data packets are traversed through the forwarding group. The forwarding groups creates a mesh instead of a tree.

In MCEDAR [3], instead of using the concept of forwarding groups, concept of core node and core broadcast is used. It uses a mesh structure called the *mgraph* as its multicast routing infrastructure. The inherent redundancy present in meshes increases the robustness of the *mgraph*. Hence, recomputation of the *mgraph* may not be necessary for every link breakage. This is a property that is critical to ad hoc environments where link breakages occur often due to node mobility. However, unlike other mesh based approaches [5], it minimizes the number of nodes in the *mgraph* by requiring only core nodes to become members of an *mgraph*. Specifically, an *mgraph* is a subgraph of the core graph and not a subgraph of the underlying network. In this instead of using the forwarding groups, as in ODMRP, concept of *mgraph* is applied and the decoupling is done by separating the data forwarding infrastructure from the control infrastructure.

The rest of the paper is organised as follows: Section 2 provides the overview of ODMRP[1][2] approach, Section 3 provides the overview of MCEDAR[3] approach, Section 4 describes ODMRP approach with MCEDAR for congestion control and Section 5 presents the conclusion.

II. ON DEMAND MULTICAST ROUTING PROTOCOL OVERVIEW

The basic concept behind ODMRP [1][2] is that it creates a mesh structure to route multicast packets. The source initiates the creation of mesh by periodically probing the members to join the network. Upon receiving the probes, the members respond and forwarding groups are formed. The forwarding groups create a mesh and data packets traverse through the forwarding groups to the multicast members. It initiates the route discovery process. The route discovery process involves two phases: *request phase* and *reply phase*. The path from sender (S) and receiver (R) is shown in figure 1.

A. REQUEST PHASE

During this phase, the source floods the network with member advertisement packets called JOIN QUERY packet. JOIN QUERY packets are periodically broadcasted to the network to refresh membership information and reestablish new multicast routes. Upon receiving a non-duplicate JOIN QUERY packet, the node inserts and updates its ROUTING TABLE with the address of the upstream node as the next node to the source node.

B. REPLY PHASE

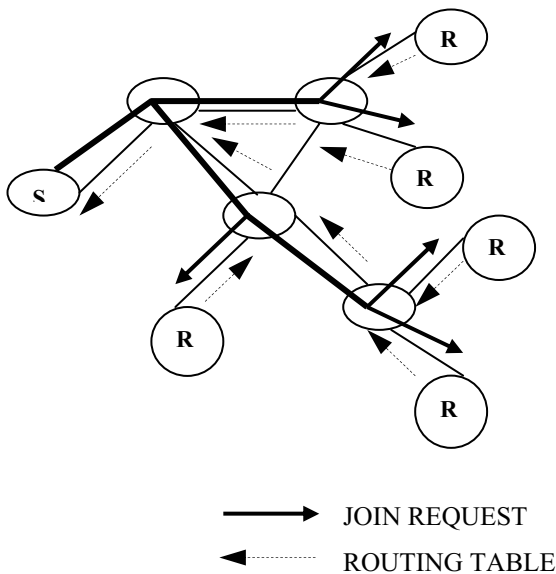


Fig.1. On demand procedure for member set-up

Once the non-duplicate packet is received, the reply phase begins. During this phase, the multicast member creates and broadcasts a JOIN REPLY packet with the address of the node the member received the JOIN QUERY from stamped in the JOIN REPLY. Once the JOIN REPLY. Upon receiving the JOIN REPLY, a node determines if its address is stamped in the JOIN REPLY. If it is, the node realizes it is on the path to

the source. The node then sets FG_FLAG and becomes part of the forwarding group. Afterwards, the node rebroadcasts JOIN REPLY with the upstream node address obtained from the ROUTING TABLE that is stamped in the JOIN REPLY packet. This process continues until the packet reaches the source. Once the source receives JOIN REPLY, a mesh of nodes, or forwarding groups, is formed and packets can be delivered to the members.

One of the way to achieve congestion control using ODMRP is to couple it with BMW (Broadcast Medium Window) which has been done in one of the research [6]. In that paper, they have used the concept of mean aggregate queue length and used the BMW as the underlying MAC layer.

III. MULTICAST CORE-EXTRACTION DISTRIBUTED AD-HOC ROUTING OVERVIEW

In MCEDAR approach, instead of using the forwarding group, core node concept has been used. MCEDAR multicast routing protocol [3] addresses both the issues of robustness and efficiency in one framework.

Essentially, MCEDAR [3] uses two of CEDAR's[4] components, namely, core and core broadcast. The infrastructure for a multicast group resides entirely within the core and the core broadcast mechanism is used to perform data forwarding on the infrastructure.

A. CORE EXTRACTION

The core extraction process is done same as the CEDAR approach [4]. The core computation algorithm is as follows:

- (i) Choose a node m .
- (i) m , node id.
- (ii) $d(m)$, the degree of m , i.e. direct neighbors.
- (iii) $d^*(m)$ is the effective degree that the number of neighbors selects m as its dominator.
- (iv) $\text{dom}(m)$, the dominator of node m (may be itself).
- (v) m sets $\text{dom}(m)$ to v to the highest value of $\langle d^*(v), d(v) \rangle$, where v is the next node in its neighbor.
- (vi) m then sends v (its dominator) an unicast message containing its id and the ids and the dominators of its immediate neighbors. v then increments $d^*(v)$.
- (vii) if $d^*(m) > 0$, then m joins the core of the network

After extracting the core nodes, the route computation is done for the packets to follow. The route computation algorithm is as follows:

- I. Let t be the end point of the chosen tunnel.
- II. $p(s, t)$ denote the tunnel.
- III. $\text{core}(s)$ sends the following message $\langle s, d, t, b, p(s, t), \text{core}(s), \text{core}(d) \rangle$, where s, d, t are source destination, intermediate node respectively, b is the required bandwidth and $p(s, t)$ is the tunnel from s to t .
- IV. $\text{core}(t)$ perform the route computation and send the packet to $\text{core}(d)$ and there is an admissible path to $\text{dest}(d)$ by using core node only.

After the core extraction, core broadcast is done same as the CEDAR approach [4]. In which, a mechanism based on reliable unicast (using RTS-CTS) is done.

B. MGRAPH INFRASTRUCTURE

MCEDAR uses a mesh structure called the *mgraph* as its multicast routing infrastructure. Unlike other mesh based approaches [5], the number of nodes in the *mgraph* is reduced by making only the core nodes as its member. Basically, *mgraph* is a subgraph of the core node and not the subgraph of the underlying network. So, only core nodes are the members of the *mgraph*. When a new node wants to join the graph, it requests its dominating core node to first join the graph and then the core node performs the join operation.

MCEDAR associates a *robustness factor* R with each *mgraph*. The factor R represents the degree of robustness supported by the *mgraph*. The factor R signifies how “strongly” each member of a multicast group is connected to the rest of the corresponding *mgraph*. Each member of the *mgraph* maintains a notion of which of its nearby core nodes are the member of the same *mgraph*. This is used in the data forwarding where incoming data at a member is forwarded onto all other members except the one from which the data arrived.

C. JOIN PROTOCOL

As only core nodes are the member of the *mgraph*, only a core node is allowed to perform the join operation. A core node performs the join operation by issuing the *join request* JOIN (MA, *join ID*) where MA is the address of the joining group and *join ID* is the id of the node. Note that, the join ID of the freshly new joining node is infinity. When a non-member node receives the join request, it simply broadcasts the message to its nearby core node according to the core broadcast mechanism as explained in the CEDAR[4]. On the other hand, when a member of the group receives the message, it sends a JOIN_ACK (MA, *join ID*) only if its join ID is less than the join ID of the request arrived. It then forwards the message further down. The intermediate node in the reverse path receives the JOIN_ACK message, after receiving, it decides whether to accept or reject it based on the number of JOIN_ACKs it has already accepted for a particular group. An intermediate node accepts the JOIN_ACK, if the number of such accepted messages for that group is less than the *robustness factor* R . Each member of the group maintains two other data structures: *parent set* and *child set*. On the other hand, when an intermediate node reject a JOIN_ACK, it suppresses the JOIN_ACK and upstream node id is removed from its child set.

D. Forwarding Protocol

When a data packet arrives at an *mgraph* member, the member *attempts* to forward the data packet only to those nearby core nodes that is knows are also members of the same *mgraph*.

E. Decoupling Control Infrastructure and data Forwarding Infrastructure

This is an optimization mechanism which is performed by decoupling the control infrastructure-*mgraph*, from the actual data forwarding infrastructure. When a JOIN_ACK is sent back to the newly joined node, intermediate node computes the data path from its domain to its child’s domain. It then adds information about the link from domain to child’s domain to the JOIN_ACK and then it forwards the message to domain. It also updates the multicast routing table in its domain accordingly. Thus, after the join process is completed, there is a parallel data forwarding graph along with the subgraph of the *mgraph*.

IV. PROPOSED MODEL MULTI-CORE EXTRACTION IN ODMRP

Our proposed mechanism operates on multi-core extraction based on the principle of MCEDAR [3] in ODMRP[1][2]. The advantage is the congestion due to flooding in on demand protocols is minimized by restricting it to the core nodes.

A. Multi-Core Extraction

The core is extracted in the same way as in CEDAR[4] approach. In ODMRP[1][2], the concept of forwarding group for forwarding the packet is used, but in the MCEDAR[3] approach, core node is used for forwarding the packet. Each node finds its dominator by selecting the highest degree node with the effective degree to be maximum. When a node w joins the core, it sends a broadcast message. In the message, it contains $\langle w, dom, i, path_traversed \rangle$. When a node v receives the message containing $\langle w, dom, I, path_traversed \rangle$, it sends the message $\langle w, dom, i-1, path_traversed+v \rangle$ if $i-1 > 0$. So, the broadcast of the core node makes its presence in its neighborhood. This guarantees that each core node identifies its nearby core nodes and sets up a link among them which is virtual.

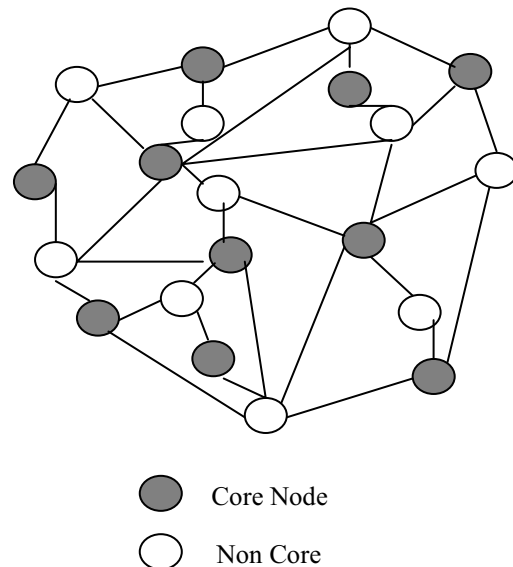


Fig.2. Core Node network

B. Core Broadcast

Core broadcast is needed to route probes to the destination core. Since, flooding causes a repeated broadcasts, it becomes highly unreliable. So, a mechanism for core broadcast based on reliable unicast, i.e. RTS/CTS messages, is provided. This is done by tagging the broadcast RTS/CTS-messages with a message id and a node id. Nodes hearing these link-layer messages cache the tags of the messages for a short period of time in order to avoid duplicate transmission of a message when a message already has reached the destination through another node.

C. Graph Formation

Graph formation from the underlying network is based on the core nodes identified and their associated non-core nodes. It is the subgraph of the core graph and not of the network. In this, each core node and its associated non-core nodes makes a domain of its own. It associates a *robustness factor* R with the graph. R determines how “strongly” each graph member is connected to the rest of the graph. It is an undirected graph $G=(V,E)$, where V are the core nodes and E is the links between them. Each core node keeps the information about its nearby core nodes which is also member of the same graph. This is useful in forwarding the data to all other members except the one from which it arrived. Figure 3 shows the core node graph obtained from core network shown in figure 2.

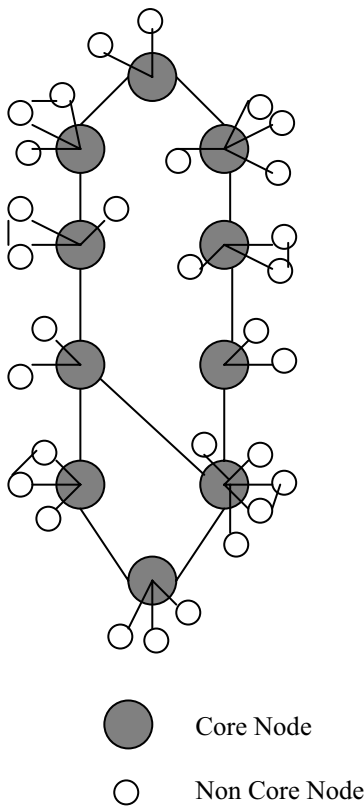
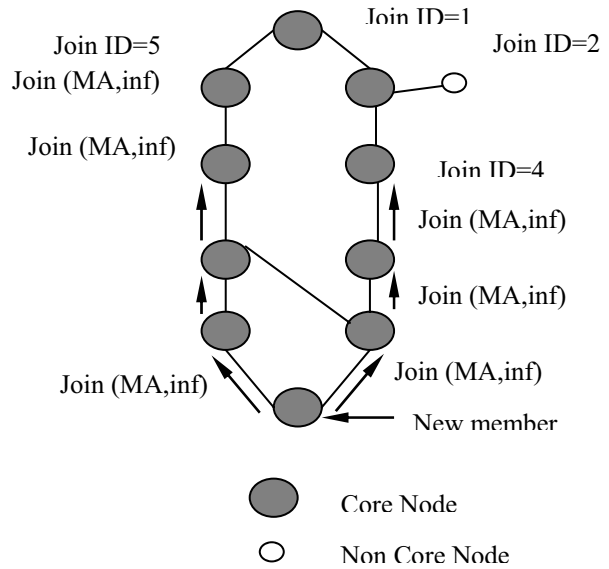
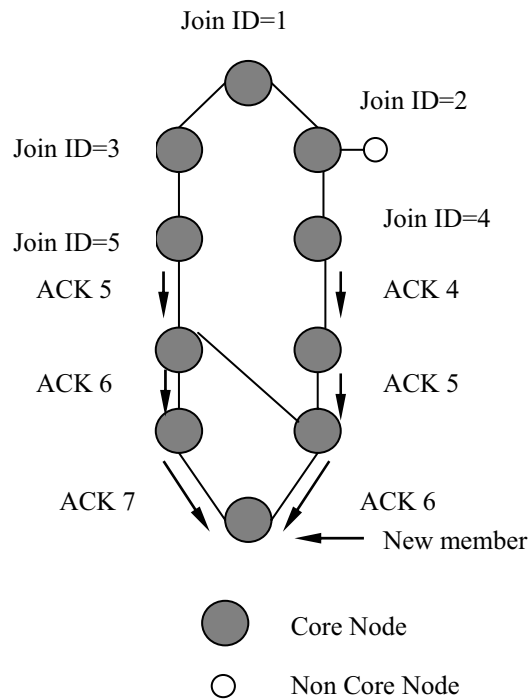


Fig.3. Core Node Graph



(a) Request Phase



(b) Acknowledgement Phase

Fig. 4. Addition of new nodes

D. Addition of new nodes

If a new node wants to join the graph, it sends a message to its dominating core node and request it to perform join operation. A core node performs the operation by sending the request message $JOIN(MA, joinID)$ where MA represents the address of the joining group and $joinID$ is the current ID

of the node. Note that, freshly new node has join ID as infinity. When a non member node receives the message, it sends the message to its nearby core node according to the core broadcast mechanism. If an existing node receives the message, it sends a JOIN_ACK(MA,joinID) only if its joinID is less than the joinID of the arrived message. Each graph member consists of two other data structures: *parent set* and *child set*. *Parent set* consists of the upstream node and *child set* consists of the downstream node. Figure 4(a) shows request phase and 4(b) shows acknowledgement phase in the addition of new nodes.

E. Deletion of existing node

If an existing member wants to leave the graph, it sends a *delete* message to each of its parents which does not have any local members in its domain and its child set is empty. A parent node that receives the *delete* message from one of its child, it removes the corresponding ID from the child set. When a node loses the connectivity with all its parents, it issues a join operation in the same way discussed previously.

F. Decoupling of Control infrastructure and Data forwarding

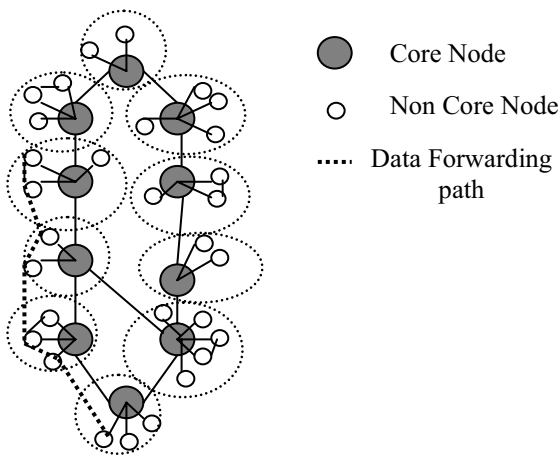


Fig.5. Decoupling of Control and Data paths

infrastructure

The data forwarding occurs in the graph itself, but, it increases the overhead. So, in this mechanism, the data forwarding is decoupled from the control infrastructure which reduces the overhead in the graph and in the entire core network. When a JOIN_ACK is sent back to the newly joined member in its vicinity, it computes the data path from its domain to its child domain and adds it to the JOIN_ACK and sends it down to the lower node. Therefore, after the process of join is completed, there is a different path of the data forwarding which runs parallel to the graph. The domain which is not in its vicinity, the data forwarding path is generated from the core nodes. Figure 5 shows the decoupling of data forwarding and control infrastructure.

CONCLUSION

This new approach provides the robustness of the mesh based multicast infrastructure while still approximating the efficiency of a tree based multicast forwarding protocol. It also uses the notion of join times to eliminate the creation of partitions in the graph when the underlying network is not partitioned. It relies on the core broadcast mechanisms for both control packet forwarding and data forwarding. In this, decoupling of the control infrastructure and the data forwarding infrastructure is done thus decreasing the control overhead and increasing the efficiency of the data forwarding.

REFERENCES

- [1] S.-J. Lee, M. Gerla, C.-C. Chiang, On-demand multicast routing protocol, Proceedings of IEEE WCNC'99, New Orleans, LA, September 1999, pp. 1298–1302.
- [2] S.-J. Lee, W. Su, J. Hsu, M. Gerla, R. Bagrodia, A performance comparison study of ad hoc wireless multicast protocols, Proceedings of IEEE INFOCOM2000, Tel Aviv, Israel, March 2000.
- [3] R.Sivakumar.; P.Sinha ; V.Bharghavan, MCEDAR: multicast core-extraction distributed ad hoc routing; Wireless Communications and Networking Conference, 1999. WCNC. 1999 IEEE 21-24 Sept. 1999 Page(s):1313 - 1317 vol.3.
- [4] R.Sivakumar.; P.Sinha ; V.Bharghavan, CEDAR: a core-extraction distributed ad hoc routing algorithm.; Selected Areas in Communications, IEEE Journal on Volume 17, Issue 8, Aug. 1999 Page(s):1454 – 1465.
- [5] J. J. Garcia-Luna-Aceves and E.L. Madruga, "A multicast routing protocol for ad-hoc networks," in Proceedings of the IEEE INFOCOM, Mar. 1999.
- [6] Ken Tang and Mario Gerla; Congestion control multicast in wireless ad hoc networks Computer Communications, Volume 26, Issue 3, 15 February 2003, Pages 278-288.
- [7] E. Bommaiah, M. Liu, A. McAuley, R. Talpade, AMRoute: ad-hoc multicast routing protocol, Internet-Draft, draft-talpade-manetamroute-00.txt, August 1998, in preparation.
- [8] E.M. Royer, C.E. Perkins, Multicast operation of the ad-hoc ondemand distance vector routing protocol, Proceedings of ACM/IEEE MOBICOM'99, Seattle, WA, August 1999.
- [9] C.W. Wu, Y.C. Tay, C.-K. Toh, Ad hoc multicast routing protocol utilizing increasing id-numberS (AMRIS) functional specification, Internet-Draft, draft-ietf-manet-amris-spec-00.txt, November 1998, in preparation.
- [10] J.J. Garcia-Luna-Aceves, E.L. Madruga, The core-assisted mesh protocol, IEEE Journal on Selected Areas in Communications 17 (8) (1999) 1380–1394.