Performance Evaluation of Routing Algorithms for Ad Hoc Wireless Sensor Network Protocols

Ms.Rashmi Bichkar
PG Student
Department of Electronics & Telecommunication
Smt.Kashibai Navale College of Engineering, Pune
University, Pune, India

Dr. U.S Sutar
Professor
Department of Electronics & Telecommunication
Smt.Kashibai Navale College of Engineering, Pune
University, Pune, India.

ABSTRACT

For good QoS in Wireless sensor network, which is collection of sensor or mobile nodes used for transforming information in a given area, better protocols should be used. To reduce the routing overhead and improve the efficiency, an enhanced routing algorithm, RE-AODV with a local route enhancement model for AODV can be implemented. Various comparisons based on end-to-end delay and routing load is done between AODV and DSR and DSDV protocols. For even energy load distribution LEACH Protocol is used and is more efficient. For RE-AODV Evaluation, TOSSIM Simulator is used. This paper presents the simulation results in order to choose the best routing protocol to give the highest performance. The simulations have shown that the conventional routing protocols like DSR have a dramatic decrease in performance when mobility is high. However the AODV and DSDV perform well when mobility is high.

General Terms

Routing Algorithms, Performance Comparison, Reliability.

Keywords

Ad Hoc, Load Balancing, Route Stability, AODV protocol, DSR protocol, DSDV, LEACH protocol

1. INTRODUCTION

The wireless technologies have penetrated everyone’s life in various ways in the recent past. Besides the wireless and mobile technologies such as GSM and WLAN unattended and self organizing wireless networks are envisaged. One such network is called the wireless ad-hoc sensor network. Such networks open a plethora of new applications such as disaster relief, community mesh networks, data gathering, monitoring and surveillance. In a wireless ad-hoc sensor network (WASN), mobile nodes with the help of other nodes forward a packet to the destination due to the limited range of each mobile host’s wireless transmission. All nodes in a network are dynamically movable since they are distributed. Due to the mobility of nodes, some pairs of nodes may not be able to communicate directly with each other. Routing protocols for sensor networks are different from other routing protocols, because of the issues such as difficulty in implementing global addressing scheme for the sensor nodes, flow of data from multiple regions to a sink and data redundancy. Ad-hoc On-demand Distance Vector Protocol AODV) is a reactive routing protocol first proposed for mobile ad-hoc networks (MANETS) which is a routing algorithm that uses the Local Route Discovery mechanism and the resulting protocol is named as route enhanced AODV (RE-AODV).There have been many proposals for routing protocols for ad hoc networks, and several protocols have emerged. They can be classified into three main categories: the *proactive*, *reactive*, and *hybrid* protocols. The Dynamic Source Routing protocol (DSR) is a simple and efficient reactive routing protocol allowing the network to be completely self-organizing and self-configuring without the need of infrastructure or administration. Ad-hoc On-demand Distance Vector (AODV) is a reactive routing protocol that utilizes an on-demand technique to establish routes. AODV is an enhancement of DSDV, which is a proactive one. The DSDV protocol requires that each mobile station in the network must constantly advertise to each of its neighbors. Other routing protocols for WASN such as Low-energy Adaptive Clustering Hierarchy (LEACH) can achieve energy efficiency and robustness by using meta-data and randomized local clustering respectively.

2. GOOD SENSOR DEPLOYEMENT

Sensors have size, weight, and cost restrictions, which impact resource availability. For maximizing the network lifetime is an important network design objective [3]. Using a minimum number of sensors is another clear objective, especially in a deterministic \node deployment approach. The coverage algorithms proposed are either centralized or distributed. In distributed algorithms, the decision process is decentralized. The WASN has a dynamic topology and needs to accommodate a large number of sensors, the algorithms and protocols designed should be distributed and localized, in order to better accommodate a scalable architecture. The most discussed coverage problems in literature can be classified in the following types: area coverage, point coverage and barrier coverage. Based on the subject to be covered (area versus discrete points), different problems can be formulated considering the following design choices:

a) Sensor deployment method: deterministic versus random [6]. A deterministic sensor placement may be feasible in friendly and accessible environments. Random sensor distribution is generally considered in remote areas.
b) Sensing and communication ranges: WASN scenarios consider sensor nodes with same or different sensing ranges. Another factor that relates to connectivity is communication range that can be equal or not equal to the sensing range.
c) Additional critical requirements: energy-efficiency and connectivity also referred to as energy-efficient coverage and connected coverage.

3. LOAD BALANCING

On-demand routing algorithms discover routes only when a node needs to send data packet to a destination and does not have any route to it.

Multipath routing allows the establishment of multiple paths between a single source and single destination node. For typical well suited single path routing protocol, an on-demand multipath distance vector protocol is developed which is called as Load Balancing Ad hoc On-demand Distance Vector (LBAODV) protocol [5]. Primary design goal behind LBAODV is to provide multiple routes which source sends data simultaneously over them.

LBAODV consists of three main phases:

a) Path Discovery Process

When a node has data to send and no route information is known, it initiates path discovery process by sending route request packet (RREQ) to its neighbors. The RREQ packet identifies the host, referred to as the *target* of the route discovery, for which the route is requested. In addition to the address of the original initiator of the request and the target of the request, each route request packet contains a *route record*, in which is accumulated a record of hops taken by the route request packet as it is propagated through the network during this route discovery. Each RREQ packet also contains a unique *request id*, set by the initiator.

To prevent the possibility of forming routing loops, each intermediate node that receives RREQ, propagates it if their address is not already included in RREQ's *Route Record* filed and appends its address to the RREQ's *Route Record* before rebroadcasting it.

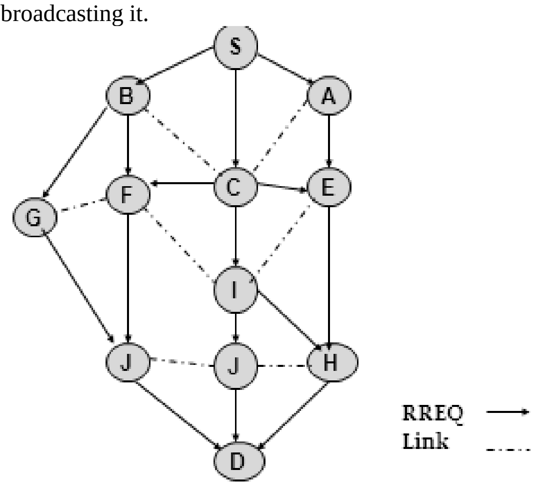

Figure 1. Propagation of RREQs

If a node receives a RREQ with a new *request id* it stores the hop count of this request in the *NumHopCount* variable, appends its address to the *Route Recor* of RREQ, increases the hop count of RREQ and rebroadcast it. To prevent flooding network with too many RREQs, nodes only rebroadcast it if the hop count of received RREQs is less (or equal to) than *NumHopCount*. Rebroadcasting the RREQs is continued until they reach to the destination. By using this method for propagation the RREQs, many RREQs from different routes will be received to the destination. For example in Figure 1, *S* sends RREQ and *A*, *B* and *C* receive this RREQ with *HopCount* sets to 1. When node *B* rebroadcasts RREQ and *C* receives it, since the hop count of received RREQ is greater than the hop count of first RREQ, this RREQ will be dropped. By using this method 7 RREQs from different paths will be received to the destination. Initiating RREQ by the source and handling it by intermediate nodes are shown as pseudo code format in Figure 2 and Figure 3 respectively.

```
Initiating Route Request packet:
If (Source doesn't have any route to D)
{
    rreqPkt = Create RREQ Packet;
    rreqPkt.requestId = get a Unique Id;
    rreqPkt.SourceAddress = Its Address;
    rreqPkt.TargetAddress = D Address;
    rreqPkt.hopCount = 1;
    Append Its Address to the Route Record of rreqPkt;
    Send rreqPkt to its neighbors;
}
Else
    Call Transmit Data function;
```

Figure 2. Initiating RREQ by the source

```
Handle Route Request Packet:
If (this node is destination)
    Send Back Route Reply to the Source;
Else
{
    If (Its address is not included in rreqPkt's Route Record field)
    {
        If (rreqPkt.hopCount <= the first HopCount received for this rreqPkt.requestId)
        {
            rreqPkt.hopCount++;
            Append Its Address to the Route Record of rreqPkt;
            Rebroadcast rreqPkt;
        }
        Else
            Drop rreqPkt;
    }
    Else
        Drop rreqPkt;
}
```

Figure 3. Handling RREQ by Intermediate nodes

When a destination receives RREQs, reverses the route in the *route record* from the received RREQs, and uses this route to send back the route reply (RREP) packet to the source. As the RREPs travel back to the source each node along the path sets up a forward pointer to the node from which the RREP came, *NextHop*, and updates its timeout information for route entries to the source. When a node receives multiple RREPs from a node, it increments the number of route reply, *CountReply*, that received from this node in its *route table* field which means how many routes from this next hop to the destination are exist. This process is continued until the RREPs reach to the source.

b) Sending Data

When RREPs receive to the source, it can transmit data packets through the discovered routes. Our protocol uses hop-by-hop method for forwarding data. Each node that receives data packets sends them to the next hops according to their *CountReply* values. Each next hop that has greater *CountReply* receives more data than the next hops that have less *CountReply*. This process causes that all of the discovered routes is used and data packets distributed across all of the paths simultaneously. For example *S* sends 4/7 of data packets to *C* that means these packets are distributed to the four different routes later, also 2/7 of data packets transmitted to *B* and 1/7 of them to *A*. Each intermediate node that receives data packets sends them to the next hops according to their *CountReply* in their *Route table.*

c) Route maintenance
If a route is not used for some period of time, a node cannot be sure whether the route is still valid; consequently, the node removes the route from its routing table. If data is flowing and a link break is detected, a Route Error (RERR) packet is sent to the source of the data in a hop-by-hop fashion. As the RERR propagates towards the source, each node decrements *CountReply* by 1 which means one of the routes from this next hop to the destination is broken. When *CountReply* of each next hop in *Route Table* reaches to 0 this next hop is deleted from route table. If no entry for a destination exists in *Route Table* of source, it invalidates the route and reinitiates route discovery process if necessary. For example in Figure 1, if the link between *I* and *J* breaks *I* sends a *RERR* to the *C*, when *C* receives this packet it changes the *CountReply* of next hop *I* in *Route Table* to 1 and forwards this packet to *S*, also when *S* receives this packet changes the *CountReply* of *C* to 3.

4. DSR PROTOCOL

The *Dynamic Source Routing* protocol (DSR) is a simple and efficient reactive Ad hoc unicast routing protocol. It has been designed specifically for use in multi-hop wireless Ad hoc networks of mobile nodes [2]. By using DSR, the network does not need any network infrastructure or administration and it is completely self-organizing and self-configuring. This routing protocol consists of two mechanisms: the *Route Discovery* and *Route Maintenance* that permit to completely maintain and automatically determine routes. These mechanisms are described in the following subsections:

a) Route Discovery
Route Discovery mechanism is in charge of finding routes between the nodes in the network. DSR is a reactive protocol, a source node S starts searching routes only if it needs to send a packet to a destination D and if no routes are enclosed in its cache. To find routes, DSR employs the Route Discovery mechanism by broadcasting Route Request (RReq) Any intermediate node that receives a non-duplicate RReq appends its address to the source route list in the RReq packet and rebroadcast it as shown in Figure 4. When the destination node receives the packet, it sends a Route Reply (RRep) back to the source node. Further, in the network, the nodes may cache routing information obtained from Route Discovery and data packets. Moreover, if an intermediate node has some requested information, that is the route to destination, in its cache, it may send an RRep back to the source node

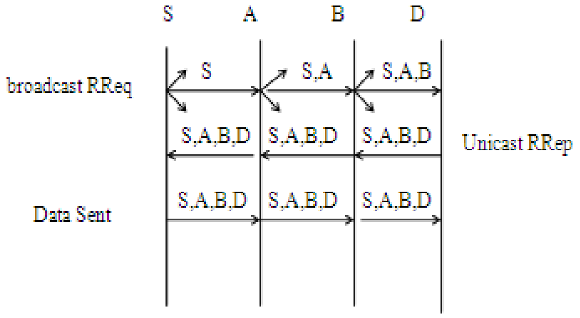

Figure 4. Route Discovery mechanism of DSR protocol

b) Route maintenance
Route Maintenance is a succession of three conditional procedures providing the confirmation that a link may carry data. First, DSR requests from the link layer to insure the maintenance. This may be provided by an existing standard part of the MAC protocol. When this layer cannot insure it, the other procedure consists in listening to every packet in the radio range of the node to determine whether the links are still available. This is called a *passive acknowledgement*. Finally, if the two first conditional procedures fail, the confirmation of receipt may be provided using network layer acknowledgments. To do so, the node inserts an Acknowledgement Request option in the DSR Options header of the packet. Therefore, when the link state is denoted as broken, the corresponding node sends a Route Error to the source node.

5. AODV and RE-AODV PROTOCOL

5.1 AODV Protocol

An Ad-hoc On-Demand Distance Vector (AODV) is one of the reactive routing protocols. It is very simple, efficient, and effective routing protocol for ad-hoc networks. In route discovery phase, source S broadcasts RREQ (Route Request) where destination number of RREQ is the last known number. The destination replies by unicasting RREP (Route Reply). The intermediate nodes which are called as neighbors discard duplicate requests, and reply if they have an active route with higher sequence number. Otherwise, they broadcast the request on all interfaces. There is a two path setup in this process: the first one is reverse path setup where a node records the address of the neighbor sending RREQ, the second is a forward path setup that unicasts RREP back to the reverse path, each node along the path setting up a forward pointer to the node from which the RREP came, and updating its routing table entry.

The node propagates the first RREP or the RREP that contains a greater destination sequence number or the same sequence number with a smaller hop count. The neighboring nodes with active routes periodically exchange hello messages. If a next hop link in the routing table fails, the active neighbors are informed. The source performs a new route request when it receives a RERR (Route Error). AODV maintains a time-based state in each node if node routing entry not recently used is expired. If a route is broken the neighbors can be notified. HELLO messages are used for detecting and monitoring links to neighbors. Although AODV is a reactive protocol, it uses these periodic HELLO messages to inform the neighbors that the link is still alive. When a node receives a HELLO message, it refreshes the corresponding lifetime of the neighbor information in the routing table. Due to hello messages, the control overhead increases linearly with the network size. It is possible that a valid route is expired in AODV and determining the reasonable expiry time is also difficult. The AODV has an evident weakness: its end-to-end delay. The route discovering delay can be a crucial factor in wireless sensor networks. For AODV, the number of control packets steeply increase when traffic load is increased from low to high at perpetual mobility. The standard AODV protocol forms routes using random channels rather than selecting optimal channels

5.2 RE-AODV Protocol

Though AODV is suited for Wireless Sensor Network over DSDV, AODV's high delay from source to destination routing and the 'valid route expiry' are the major disadvantages. A protocol called RE-AODV is presented based on the model for reducing the delay by using hello packets to exchange the local routes using *Local Discovery Algorithm (LDA)* [1].

Local Route Discovery Algorithm:
Step 1: Discover the neighbor node by sending hello packets along with route information.
Step 2: If no route is available, send the hello packet alone.
Step 3: When RREQ is received, check the local route table to know whether any neighbor with route to destination exists.
Step 4: If so, send RREP. If not, broadcast RREQ.

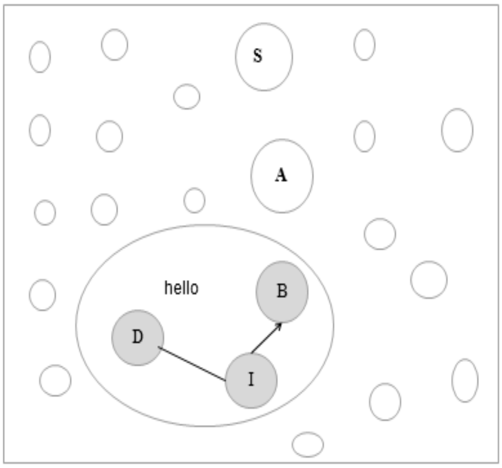

Figure 5. Exchange of hello Packets

The algorithm is better understood with the help of a scenario where n nodes are present in the network in which S is the source and D is the destination. In Figure 5, node I acts as an intermediate node which has route to the destination D. It sends and receives hello packets to their nearby nodes. After node I sends the hello packet to node B, node B updates the information to source S through the return of RREP shown in Figure 6. If route information is not available, the control packets are exchanged as in AODV.

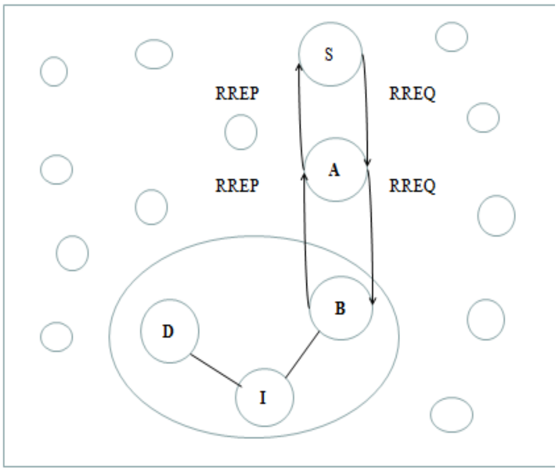

Figure 6. RREQ & RREP Messages

RREQ message is initially sent from source node S to node A, and then to B, since node B has already got hello message from D by using Figure 5. RREQ is only sent up to B. Node B simply sends the RREP message to node A. Finally, the node A sends the RREP message to source node S.

6. DSDV PROTOCOL

DSDV (Destination Sequenced Distance Vector) is one of the most well known table-driven routing algorithms for MANETs. It is a distance vector protocol. In distance vector protocols, every node *i* maintains for each destination *x* a set of distances $\{dij(x)\}$ for each node *j* that is a neighbor of *i*. Node *i* treats neighbor *k* as a next hop for a packet destined to x if $dik(x)$ equals $minj\{dij(x)\}$ as shown in Figure 7 [11]. The succession of next hops chosen in this manner leads to *x* along the shortest path. In order to keep the distance estimates up to date, each node monitors the cost of its outgoing links and periodically broadcasts to all of its neighbors its current estimate of the shortest distance to every other node in the network.

In DSDV, each node maintains a routing table which is constantly and periodically updated (not on-demand) and advertised to each of the node's current neighbors. Each entry in the routing table has the last known destination sequence number. Each node periodically transmits updates, and it does so immediately when significant new information is available. The data broadcasted by each node will contain its new sequence number and the following information for each new route: the destinations address the number of hops to reach the destination and the sequence number of the information received regarding that destination, as originally stamped by the destination. Routes with more recent sequence numbers are always the preferred basis for forwarding decisions. Of the paths with the same sequence number, those with the smallest metric (number of hops to the destination) will be used. The addresses stored in the route tables will correspond to the layer at which the DSDV protocol is operated. Operation at layer 3 will use network layer addresses for the next hop and destination addresses, and operation at layer 2 will use layer-2 MAC addresses

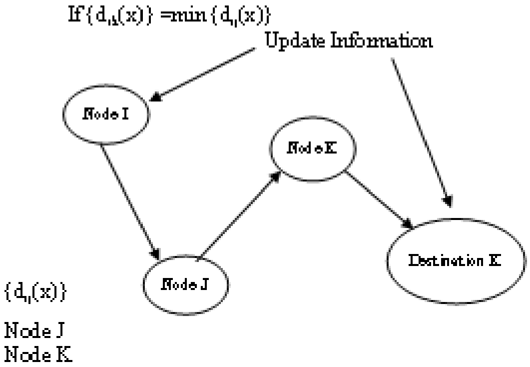

Figure 7. DSDV Illustration

Problems with Distance Vector are that, all routing decisions are taken in a completely distributed fashion. Each node uses its local information for routing messages.

7. LEACH PROTOCOL

LEACH (Low-Energy Adaptive Clustering Hierarchy) is a self-organizing, adaptive clustering protocol that uses randomization to distribute the energy load evenly among the sensors in the network. In LEACH, the nodes organize themselves into local clusters, with one node acting as the local base station or *cluster-head.* In addition, LEACH performs local data fusion to "compress" the amount of data being sent from the clusters to the base station, further reducing energy dissipation and enhancing system lifetime. Sensors elect themselves to be local cluster-heads at any given time with a certain probability. These cluster-head nodes broadcast their status to the other sensors in the network. Each sensor node determines to which cluster it wants to belong by choosing the cluster-head that requires the minimum communication energy.

Once all the nodes are organized into clusters, each cluster-head creates a schedule for the nodes in its cluster and aggregates the data and then transmits the compressed data to the base station.

Clustering is especially important for sensor network applications [12]. In cluster networks, sensors are partitioned in to smaller clusters and cluster head (CH) for each cluster is elected. Sensor nodes in each cluster transmit their data to the respective cluster head (CH) and CH aggregates data and forward them to a central base station. Clustering concepts deals with the selection of cluster-head, formation of clusters, selection of cluster-head for next round and eligibility for cluster head selection as shown in Figure 8, where clustering module forms six groups along with their respective Cluster-heads,

The operation of LEACH is divided in two phases,

(i) Set phase.

(ii) Steady state phase.

7.1 Set-up Phase

In set-up phase, the cluster head is selected and then it forms a group. The cluster head selection is based on the threshold T(s), which is given by,

T(s) = {Popt/1- Popt (r*mod*1/Popt)} where, T(S) is the threshold, popt is the predetermined percentage of cluster heads, r is the current round.

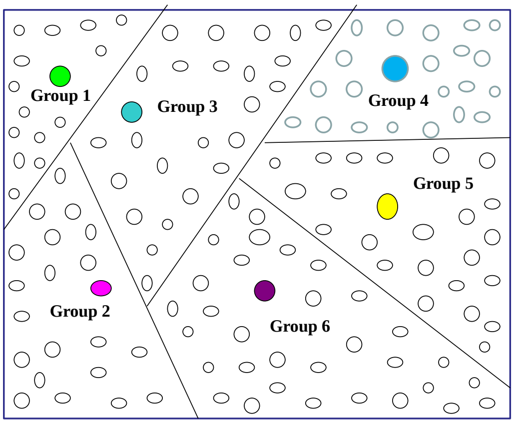

Figure 8. Clustering Form

7.2 Steady State Phase

In steady state phase, the clusters are formed and the corresponding cluster head is selected. After the cluster head receives the data, it can be aggregated and the data can be transmitted to the base station. During the set-up phase, each node sends information about its current location to the sink. In order to determining good clusters, the sink needs to ensure that the energy load is evenly distributed among all the nodes. The sink computes average node energy, and determines which nodes have energy below this average.

After the cluster head and associated cluster (the node which have energy below the average energy) are found, the sink broadcasts a message that obtains the cluster head ID for each node. If a cluster head ID matches its own ID, that node seems to be a cluster head, otherwise the node determines its TDMA slot for data transmission and goes sleep until it's time to transmit data. Once the cluster heads and associated clusters are found, the sink broadcasts a message that obtains the cluster head ID for each node. If a cluster head ID matches its own ID, the node is a cluster head. Otherwise the node determines its TDMA slot for data transmission and goes sleep until it's time to transmit data.

8. PERFORMANCE ANALYSIS

8.1 Simulation Analysis of ADOV & DSDV

The network simulation is implemented using the NS-2 simulation tool. While comparing two protocols, the focus is given on performance measurements such as average Delay.

8.1.1 Average End to end delay of data packets

The delay is affected by high rate of CBR packets as well. The buffers become full much quicker, so the packets have to stay in the buffers a much longer period of time before they are sent. Refer to the graph below, Figure 9 at the 600 pause time the delay is very high for all protocols. This is because at this pause time happen extremely high data rate and the low mobility. The low mobility will mean that already found routes are valid for much longer time period. This means that found routes can be used for more packets. From the graph it would be able to conclude that in **AODV,** since routes are established on demand and destination sequence numbers are used to find the latest route to the destination. The connection setup delay is less. **DSDV,** whenever the topology of the network changes, a new sequence number is necessary before the network re- converges. **DSR** does not have mechanism in knowing which route in the cache is stale.

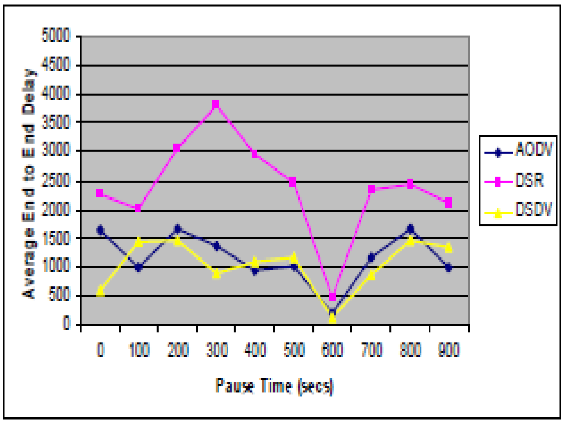

Figure 9. Average End to end delay vs. pause time

8.2 Simulation Analysis of AODV and DSR

Simulation was carried out on using OPNET simulator in a physical topology area of 700m x 500m using a random way point mobility model. At start of the simulation, each node waits for a pause time, and then moves towards a destination with a speed lying between 0 and 20 meter per seconds. On reaching the destination it pauses again and repeats the above procedure till the end of the simulation time. Mobility models were created for the simulations using 40 nodes with pause times of 100, 200, 400,600, 800, and 900 seconds respectively with maximum speed of 10 meter per second.

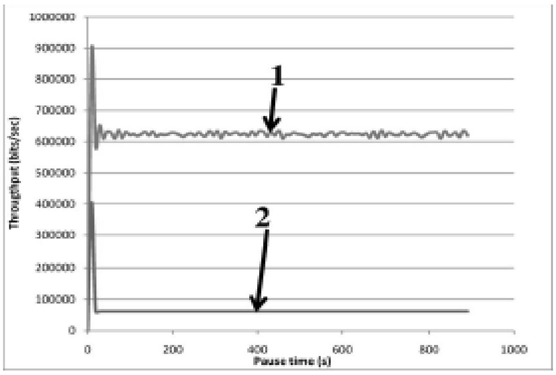

Figure 10. Throughput comparison between DSR & AODV

Constant Bit Rate (CBR) agents and packets size of 1000 bytes are used respectively to generate traffic in the network. The general simulation parameters for the AODV and DSR are summarized in Table I. Figure 10 shows both DSR and AODV throughput in bits per seconds. AODV achieved 400000 bit/sec at 100 sec pause time while DSR recorded 900000 bit/sec. As the pause time increases and more network routes are discovered, AODV throughput drops to 500000 bit/sec while DSR remains constant at 600000 bit with increase in pause time. This shows the effect of variation in pause time of a mobile node. Both protocols deliver a greater percentage of the originated data packet at low node mobility, with AODV routing protocol delivering 99% confidence interval while DSR routing protocol delivers at 98% confidence interval. [2] This shows that DSR routing protocol was very good at all mobility rate and movement speed.

Table 1. Simulation Parameters of DSR & AODV

Parameters	Values
Channel Type	Wireless Channel
Physical characteristics	Direct Sequence
Mac type	802.11b
Data rate	11 Mbs
Topology	700m X 500m
Routing Protocol	AODV, DSR
Number of Nodes	40
Transmit Power	0.005 W
Packet Size	1000 bytes
Mobility Model	Random Way Point
Simulation Time	900 sec
Traffic Source	CBR
Speed	0-20 m/sec

9. CONCLUSION AND FUTURE SCOPE

The results showed that AODV and DSR routing protocols were optimized to obtain a higher throughput. Simulation results agree with expected theoretical analysis, hence, AODV performs best considering its ability to maintain connection by periodic exchange of hello messages which is required for CBR traffic source. Throughput showed that AODV performs predictably well delivering all it packets at low mobility but decreases slightly as mobility increases, DSR performs well at both low and high mobility. Interference, high mobility, and high noise level degrades the performance of MANET but this effects can be overcome by adaptively setting the received signal threshold from range of levels in accordance with speed of nodes and topology of its network operating environment. LEACH is completely distributed, requiring no control information from the base station, and the nodes do not require knowledge of the global network in order for LEACH to operate. Future work includes the proper modeling of terrain effect prior to implementation to mitigate the effect of noise on received signal strength of the packet in a multi-hop network & to reduce the energy consumption of nodes within the ad Hoc Network.

10. ACKNOWLEDGEMENTS

The Authors would like to thank all the reviewers for providing related data and analysis of algorithms.

11. REFERENCES

[1] M.Usha, S Jayabharti, R.S.D Wahida Banu, “RE-AODV, An Enhanced Routing algorithm for QoS Support in Wireless Ad Hoc Sensor Network”, IEEE- International Conference on Recent Trends in Information technology (ICRTIT 2011), MIT, Anna University, Chennai, June 3-5, 2011.

[2] Lawal Bello, Panos Bakalis, Samuel J. Manam, Titus I.Eneh, Kwashei A. Anang, “Power Control & Performance Comparison of AODV & DSR Ad Hoc Routing Protocols”, International Conference on Modeling and Simulation, UK, 2011.

[3] M Cardei, J Wu, G.E, “Energy-efficient coverage problems in Wireless ad Hoc Sensor Networks”, Computer Communications 29(4) (2006) 421-420.

[4] Ronan de Renesse, Mona Ghassemianb, Vasilis Friderikos, A. Hamid Aghvani, “QoS Enabled Routing in Wireless Ad Hoc Networks”, Centre for Telecommunication Research, King’s College London, UK, 678 .

[5] Mehdi EffatParvar, MohammadrezaParvar, Amir Darehshoorzadeh, Mahdi Zarei, Naseer Yazdani, “ Load balancing and Route Stability in Mobile Ad Hoc Networks based on AODV Protocol”, International Conference on Electronic Devices, Systems and Applications (ICEDSA2010), 2010.

[6] S. Yang, M. Li, J. Wu, “Scan Based Movement-assisted sensor deployment methods in Wireless Sensor Network”, IEEE Transactions on Parallel and Distributed Systems 18 (8) (2007) 1108–1121.

[7] Ian D. Chakeres, Elizabeth M. Belding-Royer, “AODV Routing Protocol Implementation Design.” Dept of Computer Science University of California, Santa Barbara, 2010.

[8] S. Dhillon, K. Chakrabarty, S. Iyengar, “Sensor Placement for grid coverage under imprecise detections”, Proceedings of International Conference on Information Fusion 2002.

[9] D. Du, F. Hwang, S. Fortune, “Voronoi Diagrams and Delaunay triangulations”, Euclidean Geometry and Computers (1992).

[10] A. Howard, M.J. Mataric, G.S. Sukhatme, “An Incremental Self Deployment Algorithm for mobile sensor network”, Autonomous Robots, Special Issue on Intelligent Embedded Systems, Sept (2002).

[11] V.Ramesh, Dr.P Subbaiah, N Koteshwar Rao, M. Janardhana Raju, “Performance Comparison and Analysis of DSDV and AODV for MANETs”, (IJCSE), International Conference on Computer Science and Engineering, Vol.02, No-02, 2010, 183-185.

[12] Prof. K..Manikandan, Dr.P Purusottamam, “An Efficient Routing Protocol Design for Distributed Wireless Sensor Networks”, International Journal for Computer Applications (0975-8887), Volume-10-N.4, November-2010.