Multihop Routing In Self-Organizing Wireless Sensor Networks

Rajashree.V.Biradar [1], Dr. S. R. Sawant [2], Dr. R. R. Mudholkar [3], Dr. V.C .Patil [4]

[1] Department of Information Science and Engineering, Ballari Institute of Technology and Management, Bellary-583104, Karnataka, India.

[2] Department of Electronics, Shivaji University, Kolhapur-416004, Maharashtra, India.

[3] Department of Electronics, Shivaji University, Kolhapur-416004, Maharashtra, India.

[4]Department of Electronics and Communication Engineering, Ballari Institute of Technology and Management, Bellary-583104, Karnataka, India.

Abstract

Wireless sensor networks have emerged in the past decade as a result of recent advances in microelectronic system fabrication, wireless communications, integrated circuit technologies, microprocessor hardware and nano-technology, progress in ad-hoc networking routing protocols, distributed signal processing, pervasive computing and embedded systems. As routing protocols are application specific, recent advances in wireless sensor networks have led to many new protocols specifically designed for routing. Efficient routing in a sensor network requires that the routing protocol must minimize energy dissipation and maximize network life time. In this paper we have implemented several multihop flat based routing protocols like Flooding, Gossiping and a cluster based protocol Multihop-LEACH which does inter-cluster and intra-cluster multihopping using TinyOs and TOSSIM simulator. Evaluation and comparison reveals that Multihop-LEACH protocol utilizes less power and least delay compared to other protocols. We further evaluated the Multihop-LEACH protocol with varying probability of clustering to extend the network life time.

Keywords: *Multihop, Flooding, Gossiping, Multihop-LEACH, TinyOS, nesC, TOSSIM and Probability.*

1. Introduction

Sensor networks have emerged as a promising tool for monitoring (and possibly actuating) the physical worlds, utilizing self-organizing networks of battery-powered wireless sensors that can sense, process and communicate. Wireless sensor networks [1,2] consist of small low power nodes with sensing, computational and wireless communications capabilities that can be deployed randomly or deterministically in an area from which the users wish to collect data. Typically, wireless sensor networks contain hundreds or thousands of these sensor nodes that are generally identical. These sensor nodes have the ability to communicate either among each other or directly to a base station (BS). The sensor network is highly distributed and the nodes are lightweight. Intuitively, a greater number of sensors will enable sensing over a larger area. As the manufacturing of small, low-cost sensors become increasingly technically and economically feasible, a large number of these sensors can be networked to operate cooperatively unattended for a variety of applications. The features of sensor networks [3] are as depicted below.

Varying network size: The size of a sensor network can vary from one to thousands of nodes.

Low cost: For the deployment of sensor nodes in large numbers, a sensor node should be inexpensive.

Long lifetime network: An important characteristic of a sensor network is to design and implement efficient protocols so that the network can last as long as possible.

Self-organization: Sensor nodes should be able to form a network automatically without any external configuration.

Query and re-tasking: The user should be able to query for special events in a specific area, or remove obsolete tasks from specific sensors and assign them with new tasks. This saves a lot of energy when the tasks change frequently.

Cooperation/Data aggregation: Sensor nodes should be able to work together and aggregate their data in a meaningful way. This could improve the network efficiency.

Application awareness: A sensor network is not a general purpose network. It only serves specific applications.

Data centric: Data collected by sensor nodes in an area may overlap, which may consume significant energy. To

prevent this, a route should be found in a way that allows in-network consolidation of redundant data.

Recent advances in wireless sensor networks have led to many new protocols specifically designed for sensor networks. Most of the attention, however, has been given to the routing protocols since they might differ depending on the application and network architecture [4]. To prolong the lifetime of the sensor nodes, designing efficient routing protocols is critical. Even though sensor networks are primarily designed for monitoring and reporting events, since they are application dependent, a single routing protocol cannot be efficient for sensor networks across all applications. Multihop routing, clustering and data aggregation are important techniques in minimizing the energy consumption in sensor networks [12, 13, 14].

In this paper we describe and implement several multihop routing protocols for sensor networks and present a critical analysis and evaluation of these protocols. The performance comparison considering all the characteristics that should be possessed by routing protocols reveals the important features that need to be taken into consideration while designing new routing protocols for sensor networks. The remainder of this paper is organized as follows. Section 2 contains classification of routing protocols, section 3 contains description of routing protocols implemented, Section 4 contains implementation and simulation, section 5 contains simulation matrices and results and, finally section 6 contains conclusion and future work.

2. Classification of routing protocols

Broadly speaking, almost all of the routing protocols can be classified according to the network structure; as flat, hierarchical or location-based. Further, these protocols can also be classified according to operation mode; multipath-based, query-based, negotiation-based, QoS-based, and coherent-based [5]. Figure 1 illustrates classification of WSN routing protocols.

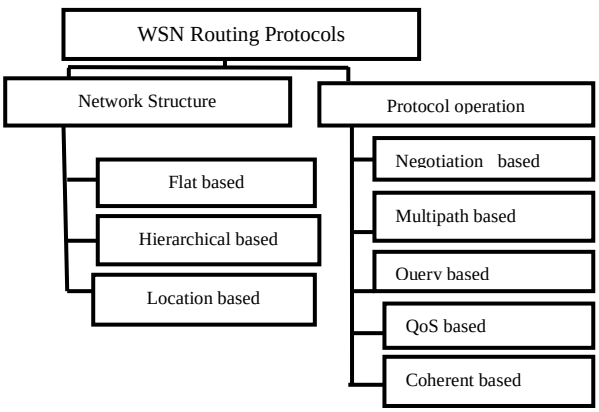

Fig. 1: Classification of WSN Routing Protocols

2.1 Network Structure

Based on the structural orientation of a network, which includes structural orientation of base stations and the structural orientation of sensor nodes we classify routing protocols as flat based, hierarchical based and location based.

Flat based: In these networks, all nodes play the same role and there is absolutely no hierarchy. Flat routing protocols distribute information as needed to any reachable sensor node within the sensor cloud [6]. No effort is made to organize the network or its traffic, only to discover the best route hop by hop to a destination by any path.

Hierarchical based: This class of routing protocols sets out to attempt to conserve energy by arranging the nodes into clusters as shown in Figure 2. Nodes in a cluster transmit to a head node within close proximity which aggregates the collected information and forward this it to the base station [6, 7].

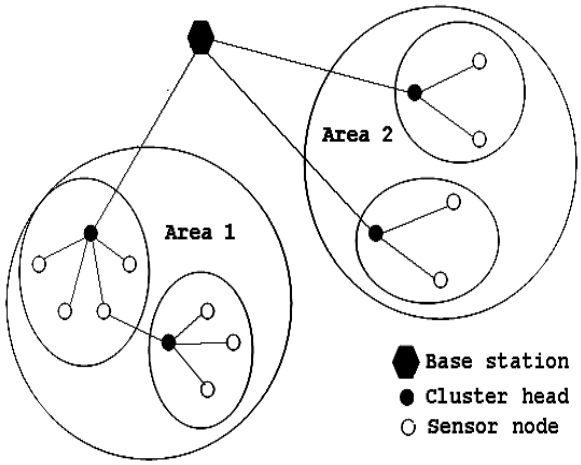

Fig. 2: Clustering Nodes

Good clustering protocols play an important role in network scalability as well as energy efficient communication. On the negative side of it, clusters may lead to a bottleneck. This is because only one head communicate on behalf of the entire cluster. Energy depletion will be strongest in that head.

Location based: Most of the routing protocols for sensor networks require location information for sensor nodes. In most cases location information is needed to calculate the distance between two particular nodes so that energy consumption can be estimated. Since there is no addressing scheme for sensor networks like IP-addresses, location information can be utilized in routing data in an energy efficient way [6].

2.2 Protocol Operation

It describes the main operational characteristics of a routing protocol; in terms of communication pattern, hierarchy, delivery method, computation, next- hop.

Multipath based: In this case, the network derives benefit from the fact that there may be multiple paths between a source node and the destination. Using different paths ensures that energy is depleted uniformly and no single node bears the brunt [12, 13].

Query based: Here the focus lies on propagation of queries throughout the network by the nodes which require some data. Any node which receives a query and also has the requested data, replies with the data to the requesting node. This approach conserves energy by minimizing redundant or non-requested data transmissions [8].

Negotiation based: The nodes here exchange a number of messages between themselves before transmission of data [9, 10]. The benefit of this is that redundant data transmissions are suppressed. It should however be ensured that the negotiation transmissions are not allowed to exceed an extent that the energy saving benefit is offset by the negotiation overhead.

QoS-based: QoS based protocols have to find a trade-off between energy consumption and the quality of service [11]. A high energy consumption path or approach may be adopted if it improves the QoS. So when interested in energy conservation, these types of protocols are usually not very useful.

Coherent-based : Coherence based protocols focus on how much data processing takes place at each node[11]. In coherent protocols, data is sent to an aggregator node after minimum possible processing, and processing is then done at the aggregator. Coherent processing is usually adopted for energy efficient routing because they reduce the computation steps per node. However, the aggregator nodes must have more energy than the other ordinary nodes, or else they will be depleted rapidly.

3. Description of routing protocols implemented

3.1 Flooding

Flooding [1] starts with a source node sending its data to all of its neighbors. Upon receiving a piece of data, each node then stores and sends a copy of the data to all of *its* neighbors. Only packets which are destined for the node itself or packets whose hop count has exceeded a preset limit are not forwarded. This is therefore a straight forward protocol requiring no protocol state at any node, and it disseminates data quickly in a network where bandwidth is not scarce and links are not loss-prone. The main benefit of Flooding is that it requires no costly topology maintenance or route discovery. Once sent a packet will follow all possible routes to its destination. If the network topology changes sent packets will simply follow the new routes added. Flooding does however have several problems. One such problem is implosion. Implosion is where a sensor node receives duplicate packets from its neighbors. Figure 3 illustrates the implosion problem. Node A broadcasts a data packet ([A]) which is received by all nodes in range (nodes B and C in this case). These nodes then forward the packet by broadcasting it to all nodes within range (nodes A and D). This results in node D receiving two copies of the packet originally sent by node A. This can result in problems determining if a packet is new or old due to the large volume of duplicate packets generated when flooding. Overlap is another problem which occurs when using Flooding. If two nodes share the same observation region both nodes will witness an event at the same time and transmit details of this event. This results in nodes receiving several messages containing the same data from different nodes. Figure 4 illustrates the overlap problem. Nodes A and B both monitor geographic region Y. When nodes A and B flood the network with their sensor data node C receives two copies of the data for geographic region Y as it is included in both packets. Another problem with Flooding is that the protocol is blind to available resources. Messages are sent and received by a node regardless of how much power it has available. In addition to this the number of packets generated by the Flooding protocol causes a lot of network traffic and causes a large network wide energy drain across the network. This can shorten the life of the network.

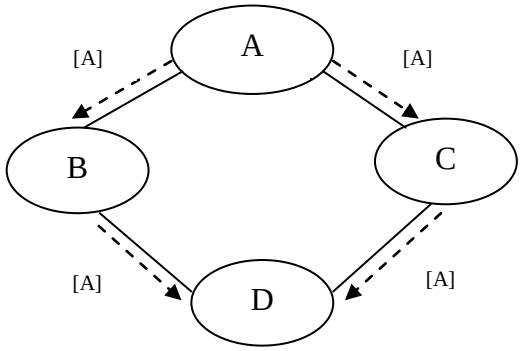

Fig. 3: Implosion problem

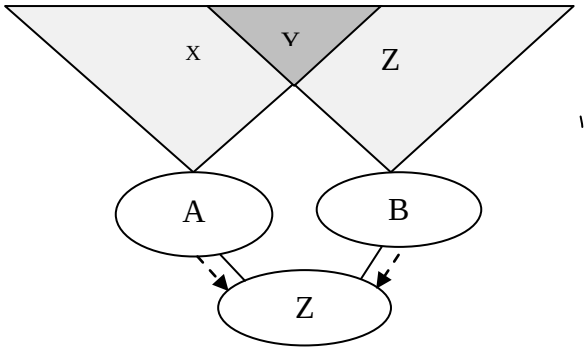

Fig. 4: Overlap problem

3.2 Gossiping

The Gossiping protocol is based on the Flooding protocol. Gossiping is proposed to address some critical problems of the Flooding scheme [1, 2]. Instead of broadcasting each packet to all neighbors the packet is sent to a single neighbor chosen at random from a neighbor table. Having received the packet the neighbor chooses another random node to send to. This can include the node which sent the packet. This continues until the packet reaches its destination or the maximum hop count of the packet is exceeded.

Gossiping avoids the implosion problem experienced by Flooding as only one copy of a packet is in transit at any one time. However, it may cause another problem, the long packet delay. Because the sender randomly selects the subset of the result in a router neighbors to transmit data, the selected sensors may result farther than the shortest path between the sender and the sink. Hence, this may extend the packet delay time. While gossiping distributes information slowly, it dissipates energy at a slow rate as well. Consider the case where a single data source disseminates data using gossiping. Since the source sends to only one of its neighbors, and that neighbor sends to only one of its neighbors, the fastest rate at which gossiping distributes data is 1 node/round. Finally, we note that, although Gossiping largely avoids implosion, it does not solve the overlap problem.

3.3 Multihop Low Energy Adaptive Clustering (Multihop-LEACH)

Multihop-LEACH is a cluster based routing algorithm in which self-elected cluster heads collect data from all the sensor nodes in their cluster, aggregate the collected data by data fusion methods and transmit the data through an optimal path between the cluster head (CH) and the base station(BS) through other intermediate CHs and use these CHs as a relay station to transmit data through them as shown in figure 5.

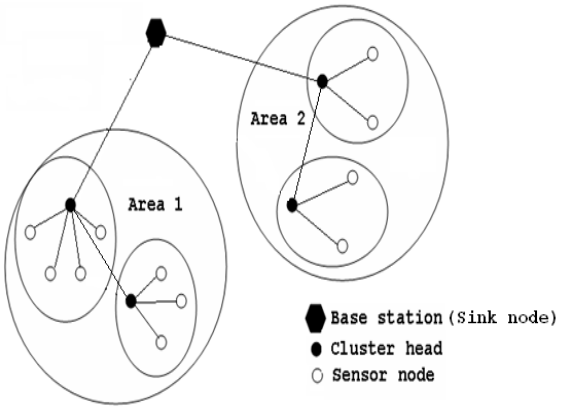

Fig. 5: Nodes communicate to Base Station through an optimal path of Cluster Heads

These self elected cluster heads continue to be cluster heads for a period referred to as a round.At the beginning of each round, every node determines if it can be a cluster head during the current round by the energy left at the node. In this manner, a uniform energy dissipation of the sensor network is obtained. If a node decides to be a cluster head for the current round, it announces its decision to its neighbors. Other nodes which choose not to be cluster heads determine to which cluster they want to belong by choosing the cluster head that requires the minimum communication energy. Multihop-LEACH was mainly proposed for routing data in wireless sensor networks which have a fixed base station to which recorded data needs to be routed. All the sensor nodes are considered static, homogenous and energy constrained. The sensor nodes are expected to sense the environment continuously and thus have data sent at a fixed rate. These assumptions make it unsuitable for sensor networks where a moving source needs to be monitored.

The operation of Multihop-LEACH is separated into two phases: the setup phase and the steady state data transfer phase. In the set up phase, the clusters are organized and cluster heads selected. During the setup phase, the cluster heads are selected based on the suggested percentage of probability of clustering for the network and the number of times the node has been a cluster-head so far. This decision is made by each node *n* choosing a random number between 0 and 1. If the number is less than a threshold $T(n)$, the node becomes a cluster-head for the current round. The threshold is set as follows:

$$T(n) = \begin{cases} \dfrac{P}{1-P(r \bmod 1/p)} & \text{if } n \in G \\ 0 & \text{otherwise} \end{cases} \qquad (1)$$

Where P is the desired cluster-head probability, r is the number of the current round and G is the set of nodes that have not been cluster-heads in the last $1/P$ rounds.
Once the nodes have elected themselves to be cluster heads they broadcast an advertisement message (ADV). Each non cluster-head node decides its cluster for this round by choosing the cluster head that requires minimum communication energy, based on the received signal strength of the advertisement from each cluster head.

After each node decides to which cluster it belongs, it informs the cluster head by transmitting a join request message (Join-REQ) back to the cluster head. After receiving all the messages from the nodes that would like to be included into the cluster and based on the number of nodes in the cluster, the cluster head creates and announces a TDMA schedule, assigning each node a time slot when it can transmit. Each cluster communicates using different CDMA codes to reduce interference from

nodes belonging to other clusters. The CDMA code to be used in the current round is transmitted along with the TDMA schedule.

In the steady state phase, the actual data transfer to the base station takes place. Upon receiving all the data, the cluster head node aggregates it before sending it to the other cluster head nodes. After a certain time, determined a priori, the network goes back to the set up phase and enters another round of selecting new cluster heads.

Inter-cluster and intra-cluster multi-hop communication are the two major concepts considered in Multihop-LEACH protocol.

Multihop inter-cluster operation: In this model network is grouped into different clusters. Each cluster is composed of one cluster head (CH) and cluster member nodes. The respective CH gets the sensed data from its cluster member nodes, aggregates the sensed information and then sends it to the Base Station through an optimal multihop tree formed [21] between cluster heads (CHs) with base station as root node as shown in figure 5.

Multihop intra-cluster operation: However, we note that in general using single hop communication within a cluster for communication between the sensor nodes and the cluster heads may not be the optimum choice. When the sensor nodes are deployed in regions of dense vegetation or uneven terrain, it may be beneficial to use multi-hop communication among the nodes in the cluster to reach the cluster head. As it is possible for nodes to remain disconnected from the network due to a cluster head not being in range, each node is able to request another connected node to become a cluster head. This occurs after a timeout period and is done through a normal advertisement message.

4. Implementation and simulation

All routing protocols are implemented with TinyOS [17] using the nesC [16] programming language. A complete application utilizing the library components of TinyOS is developed to test the protocol. The TOSSIM [15] simulator, which builds directly from the TinyOS components is used to simulate implemented protocols. TOSSIM is provided free with TinyOS. It is designed to emulate a sensor network running TinyOS on a PC. TOSSIM also provides a graphical front end to a TinyOS simulation through the TinyViz program written in Java.

4.1 Introduction to TinyOS

TinyOS is an event driven operating system designed for sensor networks, where demands on concurrency and low power consumption are high but the hardware resources are limited [17]. The main strength of TinyOS is that it has a very small footprint – the kernel which occupies approximately 100kb of memory. This means that most of the precious available memory can be allocated to application needs. TinyOS can also be executed on microprocessors that support clock speeds of 5MHz or less as is the case wireless sensor network hardware. Aside from the kernel, TinyOS comes equipped with many support tools, library routines and sample applications and full source is provided. This archive contains the following components:

TinyOS core: Operating System Kernel and Run-time routines.

nesC compiler: An extension to the GNU compiler system.

Sample Applications inc. nesC source code): Applications written in nesC which demonstrate the capabilities of the system and provide a base for extension and adaptation to specific requirements.

Library Routines and System Components (inc. nesC source code): Most importantly, the TinyOS package contains a well-defined hierarchy of system library components. These components provide an abstraction layer for communication and components such as sensors etc.

TOSSIM (TinyOS Simulator): This program is a WSN simulator allowing the simulation of 1000's of motes (discussed in more detail later)

Debugging Tools: There are a number of debugging tools available, including TOSSIM which allow the programs to be interrogated during execution and program states and system calls to be echoed to a PC terminal screen.

Documentation: Documentation is provided for all components although fairly limited. The nesC compiler can also be invoked to produce documentation from source code.

Tutorial: A tutorial in HTML format is also available within the TinyOS downloadable archive and on the web.

4.2 Introduction to nesC

The Network embedded system C (nesC) is an open source programming language is specialized for sensor networks [16]. It is an extension of the C programming language which was designed to facilitate the implementation of the structuring concepts and execution model of TinyOS. nesC was primarily designed for use with embedded systems such as sensor networks. nesC defines a component based model in order to make it possible to split applications into separate parts which communicates with each other using bidirectional interfaces. nesC does not permit separate compilation as C does. This is because nesC uses whole program analysis to

improve the performance and make the source code safer. Because the size of the application often is relatively small the need for separate compilation is not very critical.

In nesC there is a separation of construction and composition. Programs are built out of components which are 'wired' together to form whole programs. In nesC components provide and use bidirectional interfaces which are the only way to access a component. An interface declares a set of commands which must be implemented by the interface provider and a set of events which must be implemented by the user of that interface. If a component wishes to call a command in an interface it must implement the events associated with that interface. The only communication between components is by commands and events. Commands and events are similar to functions and methods in other languages and are used in the same way.

4.3 Introduction to TOSSIM

TOSSIM is a discrete event simulator for TinyOS sensor networks [15]. Instead of compiling a TinyOS application for a mote, users can compile it into the TOSSIM framework, which runs on a PC. This allows users to debug, test, and analyze algorithms in a controlled and repeatable environment. As TOSSIM runs on a PC, users can examine their TinyOS code using debuggers and other development tools.

The main aim of TOSSIM is to provide a high fidelity simulation of TinyOS applications. In order to achieve this, the focus of TOSSIM is to simulate the execution of TinyOS as opposed to simulating the real world. TOSSIM is very flexible and allows the simulation of thousands of motes with differing behavior in a variety of environments.

The advantage of TOSSIM over alternative simulators is that it is native to TinyOS and nesC source code. TinyViz is a Java visualization and actuation environment for TOSSIM. TinyViz provides an extensible graphical user interface for debugging, visualizing, and interacting with TOSSIM simulations of TinyOS applications. Using TinyViz, we can easily trace the execution of TinyOS apps, set breakpoints when interesting events occur, visualize radio messages, and manipulate the virtual position and radio connectivity of motes. TinyViz supports a simple "plugin" API that allows us to write our own TinyViz modules to visualize data in an application-specific way, or interact with the running simulation.

4.4 Implementation

Implementation is carried out in two stages. In the first stage two flat based multihop routing protocols namely Flooding & Gossiping and one cluster based protocol Multihop-LEACH are implemented, analyzed and compared. The results clearly show that cluster based protocol Multihop-LEACH is more energy efficient than Flooding and Gossiping. In the second stage Multihop-LEACH is further modified and evaluated with varying probability of clustering to improve success rate and to extend network life time. It is proved that increasing the probability of clustering will improve the energy consumption of Multihop-LEACH routing protocol.

As all protocols use multihop routing technique as shown in figure 6, they use MHEngine (multiop engine) module of TinyOS to broadcast and route packets. Selected routing protocol will enable route select module and Path Selection Module (PSM) to select route for data forwarding between sensor nodes. The selected path is sent to MHEngine. The multihop component architecture is shown below.

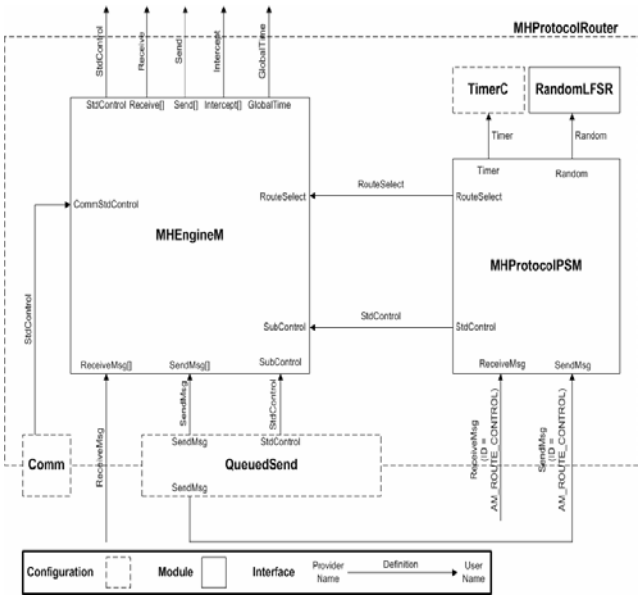

Fig. 6: Multihop component architecture

5. Simulation metrics and results

5.1 Evaluation metrics [18, 19, 20]

Latency: This performance metric is used to measure the average End-to-End delay of data packet transmission. The End-to-End delay implies the average time taken between a packet initially sent by the source, and the time for successfully receiving the message at the destination. Measuring this delay takes into account the queuing and the propagation delay of the packets. The time taken to deliver a packet to the base station from the origin node will be looked at when evaluating the protocols. In addition the per hop time delay will also be looked at. Lower latency is preferable to higher latency.

Battery usage: The power consumption is the sum of used power of all the nodes in the network, where the used

power of a node is the sum of the power used for communication, including transmitting (Pt), receiving (Pr), and idling (Pi). The amount of power used during the simulation will be monitored and used for evaluating the protocols. Batteries have a finite amount of power and nodes die once power runs out. For this reason lower power usage is preferable to higher power usage. In addition the distribution of power usage across the network will be looked at. Uniform drain is preferable.

Success rate: The number of packets received from a node at the base station will be compared with the number of packets sent by a node in order to calculate the Success rate.

Connectivity: The number of nodes that have a route to the base station will be used to assess the node connectivity provided by a particular routing protocol. More connected nodes in a network are preferable to fewer connected nodes.

5.2 Simulation and implementation parameters

The parameters used in simulating the protocols are given below in table 1.

Table 1: Summary of the parameters used in the simulation

parameters	Value
Simulation time	1800 sec
Number of node	10,20,30,40,50
Routing protocols	Flooding, Gossiping, Multihop-LEACH
Nodes distribution	randomly distributed
Network topology	Loss topology generated using LossyBuilder of TOSSIM .
Max. Packet(massage) size	29 bytes
Transmitting power per packet	3 power units
Receiving power per unit	2 power units
Message generation rate	1 message per 5 seconds.
CH probability	10%,25%,40%,50%
Nodes distribution	Nodes are randomly distributed
Operating system	TinyOs
Simulator	TOSSIM
Programming language	nesC

5.3 Simulation test cases

The various simulation test cases used in evaluating the three routing protocols that are implemented in two stages are given below in table 2. Multihop-LEACH with 50 nodes is evaluated by varying the probability of clustering.

Table 2: Simulator test cases

The test cases used in the I stage of implementation			
Test	**Protocol**	**Number of Nodes**	**Probability of clustering**
i.	Flooding	10	
ii.	Gossiping	10	
iii.	Multihop-LEACH	10	25%
iv.	Flooding	20	
v.	Gossiping	20	
vi.	Multihop-LEACH	20	25%
vii.	Flooding	30	
viii.	Gossiping	30	
ix.	Multihop-LEACH	30	25%
x.	Flooding	40	
xi.	Gossiping	40	
xii.	MULTIHOP-LEACH	40	25%
xiii.	Flooding	50	
xiv.	Gossiping	50	
xv.	Multihop-LEACH	50	25%
The test cases used in the II stage of implementation			
xvi.	Multihop-LEACH	50	10%
xvii.	Multihop-LEACH	50	25%
xviii.	Multihop-LEACH	50	40%
xix.	Multihop-LEACH	50	50%

5.3 Results

Simulated results obtained using TOSSIM can be viewed and tested in two ways. One way of visualizing the output by using a graphical tool TinyViz and the other way is by storing the results in a output text file.

Output from graphical tool TinyViz: A sample graphical display depicted in figure 7 shows that all the cluster head nodes send a packet to the base station using Multihop-LEACH protocol with a network of 50 nodes.

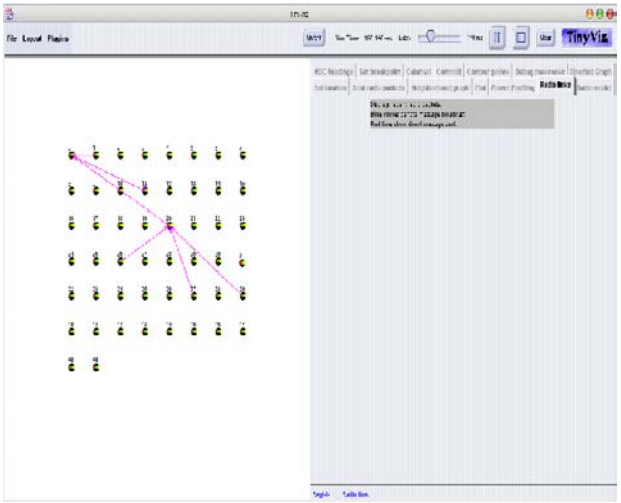

Fig. 7: Cluster head nodes send a packet to the base station.

Results obtained from output file in I stage of implementation: The output from the simulator stored in an output file has been processed in order to evaluate the protocols based on the metrics specified. The processed results are depicted below.

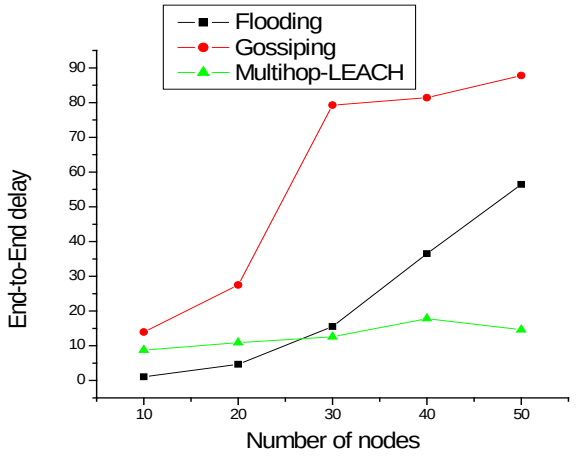

Fig. 8: Number of Nodes Vs End-to-End Delay

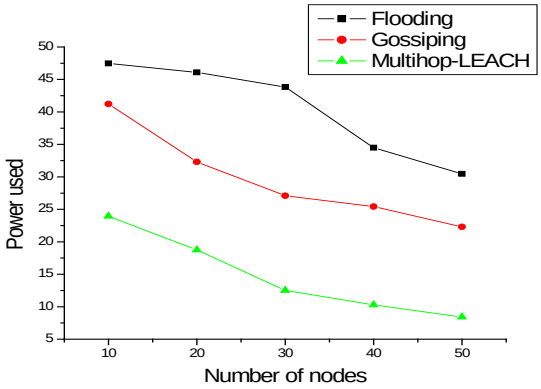

Fig. 9: Number of Nodes Vs Power usage per message

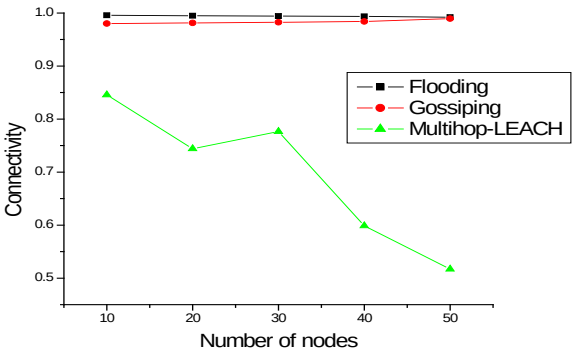

Fig.10: Number of Nodes Vs Connectivity

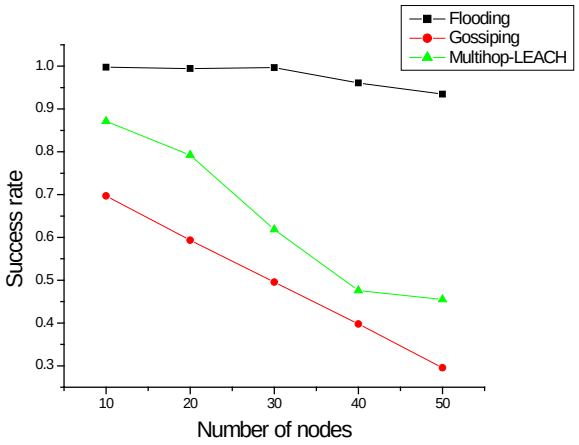

Fig. 11: Number of Nodes Vs Success rate

Results obtained from output file in the II stage of implementation: The probability of becoming Cluster Head in every node is further increased to improve the latency, success rate and connectivity of Multihop-LEACH protocol.

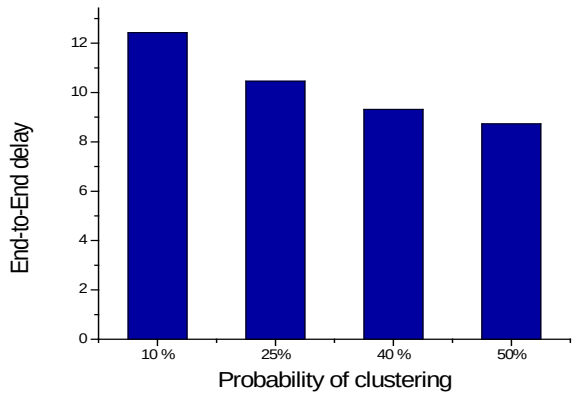

Fig.12: Probability of clustering Vs Latency (End-to-End dealy)

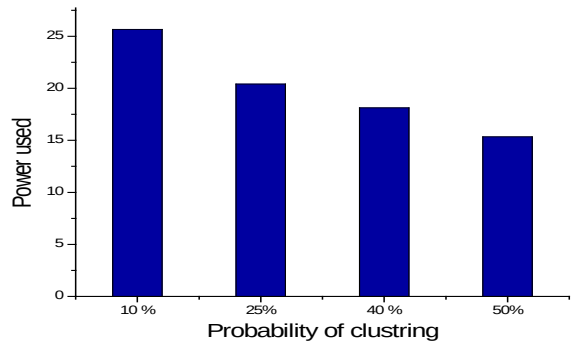

Fig. 13: Probability of clustering Vs Power usage per message

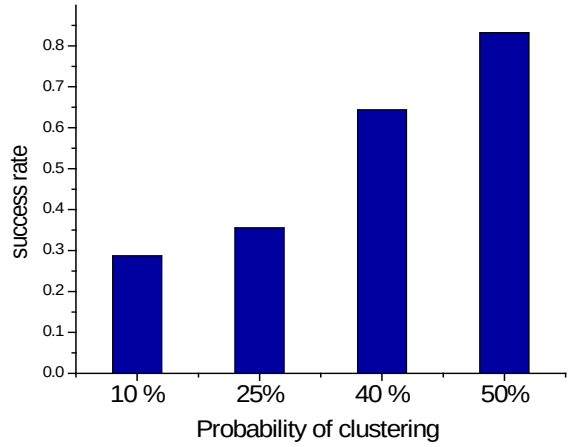

Fig. 14: Probability of clustering Vs Success rate

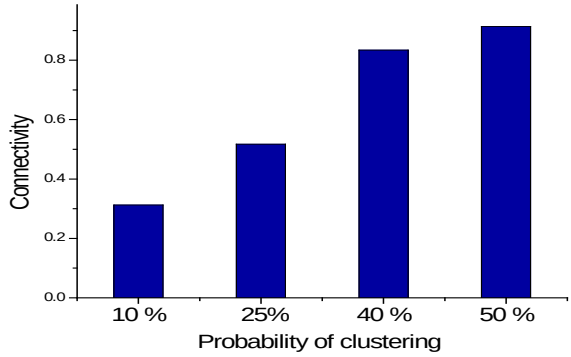

Fig. 15: Probability of clustering Vs Connectivity

Evaluation of results: Result of evaluation clearly indicates that Flooding is the worst in case of power efficiency. Gossiping provides some improvement over Flooding in terms of power usage per message. Power usage per message is less for Multihop-LEACH when compared to other two protocols. Connectivity and End-to-End delay are more in Flooding and gossiping compared Multihop-LEACH as Multihop-LEACH depends on distribution of cluster head nodes around the network. Success rate in gossiping is less because many packets will get dropped when the packets hop count reaches a maximum limit. In Multihop-LEACH success rate and connectivity are improved by increasing the probably of clustering for every nodes.

6. Conclusion and future work

The overall conclusion is that Multihop-LEACH routing protocol is best choice to move towards a network with less energy consumption as it involves energy minimizing techniques like multihop communication, clustering and data aggregation. For applications like military where energy consumption is not much to be bothered and more performance is required, Flooding is the best choice as at is simple to construct. For applications where network subjected to more scalability like environmental monitoring, Gossiping is the best choice as it uses a medium amount of power and no matter how large the network is, each node uses roughly the same amount of power. For applications where energy utilization is more critical like health monitoring, Multihop-LEACH is the best choice. Multihop-LEACH uses both inter cluster as well as intra cluster communication. The power usage, latency and success rate in Multihop-LEACH can further improved by increasing probability of clustering. We can still minimize the energy consumption and extend the network life time by improving the clustering technique. The main limitation of using TinyOs simulator is that multihop engine requires at least 2 sec between two message generations to avoid congestion and hence it requires more simulating time for evaluating the protocols with increase in network size.

References

[1] Jamal N. Al-Karaki Ahmed E. Kamal, “Routing Techniques in Wireless Sensor Networks: A Survey”, IEEE Wireless Communications, 2004, vol.11 pp: 28.

[2] Sarjoun S. Doumit, Dharma P. Agrawal, “Self-Organizing and Energy-Efficient Network of Sensors”, IEEE, 2002, pp. 1-6.

[3] I.F. Akyildiz, W Su, Y. Sankarasubramaniam and E Cayirci, “Wireless Sensor Networks, A Survey,” Communication Magazine, IEEE, August 2002, Vol. 40, Issue 8, pp. 102-114.

[4] W.Heinzelman, “Application specific protocol architectures for wireless networks”, *PhD Thesis*, MIT, 2000.

[5] K. Akkaya, M. Younis,“A Survey on Routing Protocols for Wireless Sensor Networks”, Ad-hoc Networks, May 2005,Vol. 3, No. 3, pp. 325-349.

[6] F. Ye, A. Chen, S. Liu, L. Zhang, “A scalable solution to minimum cost forwarding in large sensor networks”, Proceedings of the tenth International Conference on Computer Communications and Networks (ICCCN) , 2001, pp. 304-309.

[7] M. Chu, H. Haussecker, and F. Zhao, “Scalable Information-Driven Sensor Querying and Routing for ad hoc Heterogeneous Sensor Networks”, in the International Journal of High Performance Computing Applications, Vol. 16, No. 3,, August 2002.

[8] D. Braginsky and D. Estrin, “Rumor Routing Algorithm For Sensor Networks”, International Conference on Distributed Computing Systems (ICDCS’01), November 2001.

[9] W.Heinzelman, J. Kulik,and H. Balakrishnan, “Adaptive Protocols for Information Dissemination in Wireless Sensor Networks”, Proc. 5th ACM/IEEE Mobicom Conference (MobiCom ’99), Seattle, WA, August 1999. pp. 174-85.

[10] J. Kulik, W. R. Heinzelman, and H. Balakrishnan, "Negotiation-based protocols for disseminating information in wireless sensor networks", Wireless Networks, 2002, Vol. 8, pp. 169-185.
[11]K. Sohrabi, J. Pottie, "Protocols for self-organization of a wireless sensor network", IEEE Personal Communications, 2000, Vol. 7, Issue 5, pp 16-27.
[12] J.-H. Chang and L. Tassiulas, "Maximum Lifetime Routing in Wireless Sensor Networks", Proc. Advanced Telecommunications and Information Distribution Research Program (ATIRP2000), College Park, MD, Mar. 2000, pp 334-335.
[13] C. Rahul, J. Rabaey, "Energy Aware Routing for Low Energy Ad Hoc Sensor Networks", IEEE Wireless Communications and Networking Conference (WCNC), vol.1, March 17-21, 2002, Orlando, FL, pp. 350-355.
[14]W.R.Heinzelman ,A.Chandrakasan and H.Balakrishnan, "Energy-Efficient Communication Protocol for Wireless Microsensor Networks", 33rd Hawaii International Conference on System Sciences, Volume 8, January 2000.
[15] P.Levis and N.Lee, "TOSSIM A Simulator for TinyOS Networks", Included with the TinyOS 1.1.0 software, September 2003
[16] D.Gay, P.Levis, D.Culler and E.Brewer, nesC 1.1 Reference Manual, Included with the TinyOS 1.1.0 software, May 2003
[17]TinyOS Home Page, http://webs.cs.berkeley.edu/tos/index.html, April 2004
[18] Rajashree.V.Biradar, V.C Patil, Dr. R. R. Mudholkar , Dr. S. R. Sawant , "Classification And Comparison Of Routing Protocols In Wireless Sensor Networks", Ubiquitous Computing and Communication Journal, 2009, volume 4, pp.704-711.
[19] C. F. Chaisserini, M. Garetto, "Modeling the Performance of Wireless Sensor Networks", IEEE INFOCOM 2004, 23 Annual Joint Conference of the IEEE Computer and Communications Societies, Hong Kong, China, 7-11 March 2004, Vol. 1, pp. 231.
[20] S. Dai, X. Jing, and L. Li, "Research and analysis on routing protocols for wireless sensor networks Communications, Circuits and Systems Proceedings", International Conference on IEEE, May 2005, vol. 1, pp. 407–411.
[21] Dissertation, Hang Zhou, Zhe Jiang and Mo Xiaoyan, "Study and Design on Cluster Routing Protocols of Wireless Sensor Networks", 2006.