International Journal of Scientific & Engineering Research, Volume 3, Issue 10, October-2012
ISSN 2229-5518

Energy Efficient M-SPIN Protocol

Jyoti
Deptt. of Comp. Sc. & Engg
MDU, Rohtak , India
jyotipannu100@gmail.com

Harkesh Sehrawat
Deptt. of Comp. Sc. & Engg
MDU, Rohtak , India
sehrawat_harkesh@yahoo.co.in

Devender Sharma
Deptt. of Comp. Sc. & Engg
H.C.E , Sonipat, India
devsharma115@gmail.com

Abstract-Sensor networks are recently rapidly growing research area in wireless communications and distributed network. Data transmission is one of the major challenges in wireless sensor network (WSN). Different routing protocols have been proposed to save energy during data transmission in WSN. Since the nodes in Wireless Sensor Networks (WSN) are typically very small in size and are powered by irreplaceable battery, efficient use of energy becomes one of the most challenging tasks while designing any protocol for WSN. In this paper, we proposed an algorithm for increasing the energy efficiency of routing protocol M-SPIN belonging to SPIN family Protocol of WSN.

Index Terms- M-SPIN, SPIN, Routing protocols, Sensor Networks, WSN.

1 INTRODUCTION

Wireless Sensor Networks (WSNs) are composed of tiny, inexpensive sensor nodes with several distinguishing characteristics: they have very low processing power and radio ranges, permit very low energy consumption and perform limited and specific monitoring and sensing functions. Wireless networks have offered attractive flexibility to both network operators and users. Ubiquitous network coverage, for both local and wide areas, is provided without the cost of deploying and maintaining the wires. This fact is extremely useful in several situation like network deployment in difficult to wire areas, prohibition of cable deployment and deployment of a temporary network. Mobility support is another salient feature of wireless networks.

1.1 Sensor Networks Vs. Ad-hoc Wireless Networks

Wireless sensor networks share similarities with ad-hoc wireless networks. The dominant communication method in both is multi-hop networking, but several important distinctions can be drawn between the two. Ad-hoc networks typically support routing between any pair of nodes [9],[10],[11],[12] whereas sensor networks have a more specialized communication pattern. Most traffic in sensor networks can be classified into one of three categories: 1) Many-to-one: Multiple sensor nodes send sensor readings to a base station or aggregation point in the network. 2) One-to-many: A single node (typically a base station) multicasts or floods a query or control information to several sensor nodes. 3) Local communication: Neighboring nodes send localized messages to discover and coordinate with each other. A node may broadcast messages intended to be received by all neighboring nodes or unicast messages intended for a only single neighbor. Nodes in ad-hoc networks have generally been considered to have limited resources, sensor nodes are even more constrained. Of all of the resource constraints, limited energy is the most pressing. After deployment, many sensor networks are designed to be unattended for long periods and battery recharging or replacement may be infeasible or impossible. Nodes in sensor networks often exhibit trust relationships beyond those that are typically found in ad-hoc networks. Neighboring nodes in the sensor networks often witness the same or correlated environmental events. If each node sends a packet to the base station in response, precious energy and bandwidth are wasted. To prune these redundant messages to reduce traffic and save energy, sensor networks require in-network processing, aggregation, and duplicate elimination. This often necessitates trust relationships between nodes that are not typically assumed in ad-hoc networks.

1.2 Operation on WSN

A WSN is a large network of resource-constrained sensor nodes with multiple preset functions, such as sensing and processing, to fulfill different application objectives. The major elements of WSN are the sensor nodes and the base stations. In fact, they can be abstracted as the “sensing cells” and the “brain” of the network, respectively. Usually, sensor nodes are deployed in a designated area by an authority and then, automatically form a network through wireless communications. Sensor nodes of homogeneous or heterogeneous type can be deployed randomly or at pre-determined locations using a deterministic scheme. Sensor nodes are static most of the time, whereas mobile nodes can be deployed according to application requirements. One or several, static or mobile [8] base stations (BSs) are deployed together with the network. Sensor nodes keep monitoring the network area after being deployed. After an event of interest occurs, one of the surrounding sensor nodes can detect it, generate a report, and transmit the report to a BS through multi hop wireless links. Collaboration can be carried out if multiple surrounding nodes detect the same event. In this case, one of them generates a final report after collaborating with the other nodes. The BS can process the report and then forward it through either high-quality wireless or wired links to the external world for further processing. The WSN authority can send commands or queries to a BS, which spreads those commands or queries into the network. Hence, a BS acts as a gateway between the WSN and the external world. An example is illustrated in Fig. 1[3].

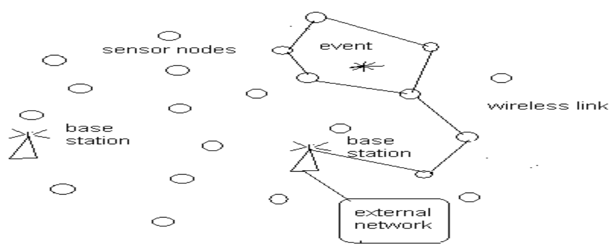

Fig 1 Wireless sensor Network

1.3 Components of Sensor Networks

Sensors can be scaled from micro sensors to larger scale .A sensor network consists sensor nodes which are small, lightweight and portable and these nodes form a network by communicating with each other directly or through other nodes. One or more nodes among them will serve as sink(s) that are responsible of communicating with the user either directly or through the existing wired networks. The main components of a sensor node as seen in the fig. 2[4], are microcontroller, transceiver, external memory, power source and one or more sensors. Every sensor node consist transducer, microcomputer, and transceiver and power source. The transducer (ADC—Analog to digital converter in fig 1) is responsible to generate electrical signals based on sensed phenomena and physical effects. The microcontroller's work is to process and store the sensor output. The transceiver receives command from a central computer or base station and transmits data to the computer or station. Sensor nodes are catered power by a battery. Some sensor nodes include external memory which may be on-chip memory of a microcontroller and Flash memory. Needs of memory of a sensor node are application specific. Each node may also belong to two extra components like: -Location finding system and Mobilizer. First one, location finding system is required since the user may in need of location with high accuracy and mobilizer may be needed to move sensor nodes to carry out the assigned tasks.

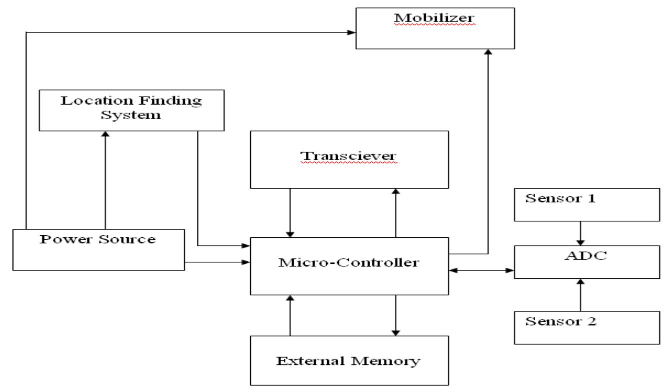

Fig 2 The components of a sensor node

2 WSN ROUTING PROTOCOLS

Routing protocol is created to compromise many aspects such as collision prevention, faster time transmission and energy saving. There are several types of routing protocol in wireless sensor network. However, this paper only discusses SPIN routing protocols. This protocol is from data centric routing.

2.1 Sensor Protocol for Information via Negotiation

Heinzelman *et al.* in [6],[14],[15] proposed a family of adaptive protocols called Sensor Protocols for Information via Negotiation (SPIN) that disseminate all the information at each node to every node in the network assuming that all nodes in the network are potential BSs. This enables a user to query any node and get the required information immediately. These protocols make use of the property that nodes in close proximity have similar data, and hence there is a need to only distribute the data other nodes do not posses. SPIN [16] is a negotiation-based information dissemination protocol suitable for WSN. It is based on the concept of metadata. The SPIN family of protocols uses data negotiation and resource-adaptive algorithms. Nodes running SPIN assign a high-level name to completely describe their collected data (called meta-data) and perform metadata negotiations before any data is transmitted. This ensures that there is no redundant data sent throughout the network. The semantics of the meta-data format is application-specific and not specified in SPIN. For example, sensors might use their unique IDs to report meta-data if they cover a certain known region. In addition, SPIN has access to the current energy level of the node and adapts the protocol it is running based on how much energy is remaining. These protocols work in a time-driven fashion and distribute the information all over the network, even when a user does not request any data. The SPIN family is designed to address the deficiencies of classic flooding by negotiation and resource adaptation. The SPIN family of protocols is designed based on two basic ideas: 1) Sensor nodes operate more efficiently and conserve energy by sending data that describe the sensor data instead of sending all the data; for example, image and sensor nodes must monitor the changes in their energy resources. 2) Conventional protocols like flooding or gossiping- based routing protocols [13] waste energy and bandwidth when sending extra and unnecessary copies of data by sensors covering overlapping areas. The drawbacks of flooding include implosion, which is caused by duplicate messages sent to the same node, overlap when two nodes sensing the same region send similar packets to the same neighbor, and resource blindness in consuming large amounts of energy without consideration for energy constraints. Gossiping avoids the problem of implosion by just selecting a random node to which to send the packet rather than broadcasting the packet blindly. However, this causes delays in propagation of data through the nodes. SPIN's meta-data negotiation solves the classic problems of flooding, thus achieving a lot of energy efficiency. SPIN is a three-stage protocol as sensor nodes use three types of messages, ADV, REQ, and DATA, to communicate. ADV is used to advertise new data, REQ to request data, and DATA is the actual message itself. The protocol starts when a SPIN node obtains new data it is willing to share. It does so by broadcasting an ADV message containing Meta data. If a neighbor is interested in the data, it sends a REQ message for the DATA and the DATA is sent to this neighbor node. The neighbor sensor node then repeats this process with its neighbors. As a result, the entire sensor area will receive a copy of the data. One of the advantages of SPIN is that topological changes are localized since each node need know only its single-hop neighbors. SPIN provides more energy savings than flooding, and metadata negotiation almost halves the redundant data. However, SPIN's data advertisement mechanism cannot guarantee delivery of data. The SPIN family of protocol is made of four protocols, SPIN-PP, SPIN-BC, SPIN-RL, SPIN-EC and a modified SPIN (M-SPIN).

2.1.1 SPIN-PP

The first SPIN protocol, SPIN-PP, is optimized for a networks using point-to-point transmission media, where it is possible for nodes *A* and *B* to communicate exclusively with each other without interfering with other nodes. In such a point to point wireless network, the cost of communicating with *n* neighbors in terms of time and energy is *n* times the cost with the data of node A and send advertisements of the aggregated data to all of its neighbors(4). Second, nodes are not required to respond to every message in the protocol. In this example, one neighbor does not send an REQ packet back to node B (5). This would occur if that node already possessed the data being advertised. Although this protocol has been designed for lossless networks with symmetric communication links, it can easily be adapted to work in lossy or mobile networks. In lossy networks, nodes could compensate for lost ADV messages by readvertising these messages periodically, and nodes could compensate for lost REQ and DATA messages by re requesting data items that do not arrive within a fixed time period. Alternatively, the protocol might be augmented to use explicit acknowledgments. For example, whenever a node received an ADV message, it would send a request message (REQ) explicitly stating which advertised data it did and did not want to receive. In this way, the sender could differentiate lost ADV messages and ADV messages that had no corresponding requests for data, and thus re advertise only the lost ADV messages. Finally, for mobile networks, changes in the local topology can trigger updates to a node's neighbor list. If a node notices that its neighbor list has changed, it can spontaneously re advertise all of its data. This protocol's strength is its simplicity. Nodes using the protocol make very simple decisions when they receive new data, and they therefore waste little energy in computation. Furthermore, each node only needs to know about its single hop network neighbors. First, SPIN-PP can be run in a completely unconfigured network with a small startup cost to determine nearest neighbors. Second, if the topology of the network changes frequently, these changes only have to travel one hop before the nodes can continue running the algorithm.

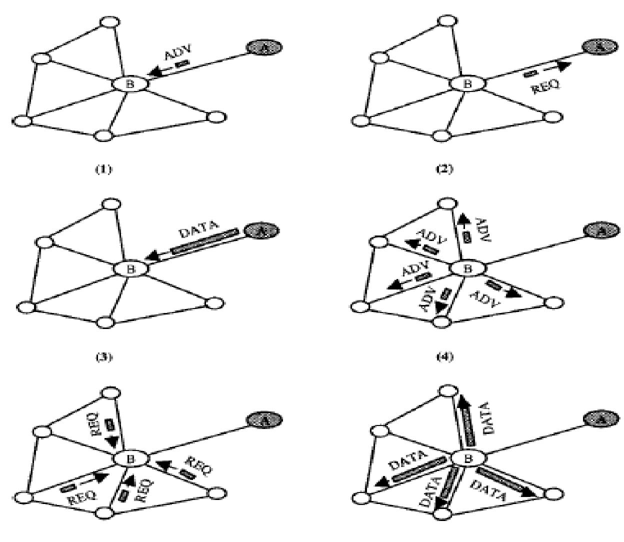

Fig. 3 The SPIN-PP protocol. Node A starts by advertising its data to node B (1). Node B responds by sending a request to node A (2). After receiving the requested data

(3), node B then sends out advertisements to its neighbors (4), who in turn send requests back to B (5, 6).

2.1.2 SPIN EC

The SPIN-EC protocol adds a simple energy-conservation heuristic to the SPIN-PP protocol. When energy is plentiful, SPIN-EC nodes communicate using the same three-stage protocol as SPIN-PP nodes. When a SPIN-EC node observes that its energy is approaching a low-energy threshold, it adapts by reducing its participation in the protocol. In general, a node will only participate in a stage of the protocol if it believes that it can complete all the other stages of the protocol without going below the low-energy threshold. This conservative approach implies that if a node receives some new data, it only initiates the three-stage protocol if it believes it has enough energy to participate in the full protocol with all of its neighbors. Similarly, if a node receives an advertisement, it does not send out a request if it does not have enough energy to transmit the request and receive the corresponding data. This approach does not prevent a node from receiving, and therefore expending energy on, ADV or REQ messages below its low-energy threshold. It does, however, prevent the node from ever handling a DATA message below this threshold.

2.1.3 SPIN-BC

In broadcast transmission media, nodes in the network communicate using a single, shared channel. As a result, when a node sends out a message in a lossless, symmetric broadcast network, it is received by every node within a certain range of the sender, regardless of the message's destination. If a node wishes to send a message and senses that the channel is currently in use, it must wait for the channel to become idle before attempting to send the message. The disadvantage of such networks is that whenever a node sends out a message, all nodes within transmission range of that node must pay a price for that transmission, in terms of both time and energy. However, the advantage of such networks is that when a single node sends a message out to a broadcast address, this message can reach all of the node's neighbors using only one transmission. One-to-many communication is therefore $1/n$ times cheaper in a broadcast network than in a point-to-point network, where n is the number of neighbors for each node. SPIN-BC improves upon SPIN-PP for broadcast networks by exclusively using cheap, one-to-many communication. This means that all messages are sent to the broadcast address and thus processed by all nodes that are within transmission range of the sender. We justify this approach by noting that, since broadcast and unicast transmissions use the same amount of network resources in a broadcast network, SPIN-BC does not lose much efficiency by using the broadcast address. Moreover, SPIN-BC nodes can coordinate their resource-conserving efforts more effectively because each node overhears all transactions that occur within its transmission range. For example, if two nodes A and B send requests for a piece of data to node C, C only needs to broadcast the requested data once in order to deliver the data to both A and B. Thus, only one node, either A or B, needs to send a request to C, and all other requests are redundant. If A and B address their requests directly to C, only C will hear the message, though all of the nodes within the transmission range of A and B will pay for two requests. However, if A and B address their requests to the broadcast address, all nodes within range will overhear these requests. Assuming that A and B are not perfectly synchronized, then either A will send its request first or B will. The node that does not send first will overhear the other node's request, realize that its own request is redundant, and suppress its own request. In this example, nodes that use the broadcast address can roughly halve their network resource consumption over nodes that do not. As we will illustrate shortly, this kind of approach, often called *broadcast message-suppression*, can be used to curtail the proliferation of redundant messages in the network. Like the SPIN-PP protocol, the SPIN-BC protocol has an ADV, REQ, and DATA stage, which serve the same purpose as they do in SPIN-PP. There are three central differences between SPIN-PP and SPIN-BC. First, as mentioned above, all SPIN-BC nodes send their messages to the broadcast address, so that all nodes within transmission range will receive the messages. Second, SPIN-BC nodes do not immediately send out requests when they hear advertisements for data they need. Upon receiving an ADV, each node checks to see whether it has already received or requested the advertised data. If not, it sets a random timer to expire, uniformly chosen from a predetermined interval. When the timer expires, the node sends an REQ message out to the broadcast address, specifying the original advertiser in the header of the message. When nodes other than the original advertiser receive the REQ, they cancel their own request timers, and prevent themselves from sending out redundant copies of the same request. The final difference between SPIN-PP and SPIN-BC is that a SPIN-BC node will send out the requested data to the broadcast address once and only once, as this is sufficient to get the data to all its neighbors. It will not respond to multiple requests for the same piece of data. Figure 4 shows an example of the protocol. Upon receiving an ADV packet from node A,

ISSN 2229-5518

A's neighbors check to see whether they have received the advertised data (1). Three of A's neighbors, C, D, and E, do not have A's data, and enter request suppression mode for different, random amounts of time. C's timer expires first, and C broadcasts a request for A's data (2), which in turn suppresses the duplicate request from D. Though several nodes receive the request, only A responds, because it is the originator of the ADV packet (3). After A sends out its data, E's request is suppressed, and C, D, and E all send out advertisements for their new data (4).

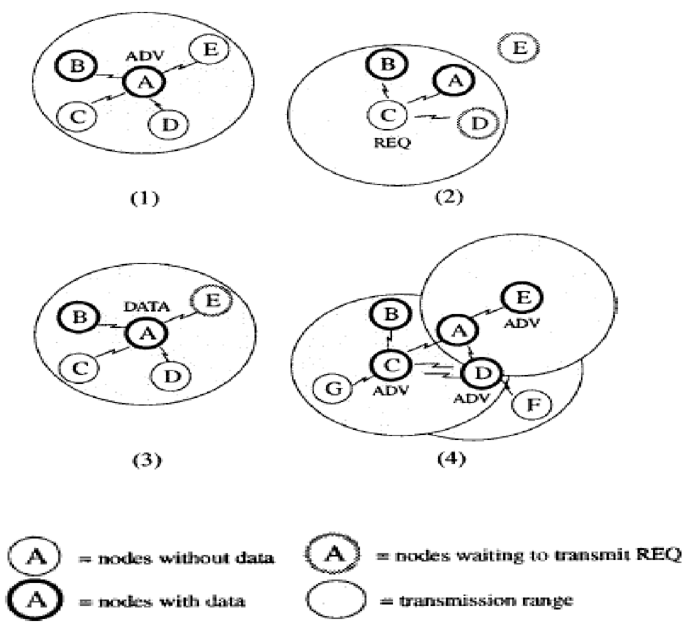

Fig 4. The SPIN-BC protocol. Node A starts by advertising its data to all of its neighbors (1). Node C responds by broadcasting a request, specifying A as the originator of the advertisement (2), and suppressing the request from D. After receiving the requested data (3), E's request is also suppressed, and C, D, and E send advertisements out to their neighbors for the data that they received from A (4).

2.1.4 SPIN-RL

SPIN-RL, a reliable version of SPIN-BC, can disseminate data efficiently through a broadcast network, even if the network loses packets or communication is asymmetric. The SPIN-RL protocol incorporates two adjustments to SPIN-BC to achieve reliability. First, each SPIN-RL node keeps track of which advertisements it hears from which nodes, and if it does not receive the data within a reasonable period of time following a request, the node rerequests the data. It fills out the originating-advertiser field in the header of the REQ message with a destination, randomly picked from the list of neighbors that had advertised that specific piece of data. Second, SPIN-RL nodes limit the frequency with which they will resend data. If a SPIN-RL node sends out a DATA message corresponding to a specific piece of data, it will wait a predetermined amount of time before responding to any more requests for that piece of data.

2.2. Modified SPIN Protocol

Another interesting fact is that energy consumption not only depends on sensing the data but also on processing the sensed data and transmitting or receiving them to or from its neighbor nodes. So if it is possible to control number of transmission and receipt of messages, a significant amount of energy can be saved. Fig. 5 shows an example of a WSN. An event that occurs in the WSN divides the entire network into two regions, A and B. Sensor nodes in region A are on the other side in the network in comparison with the sink node and sensor nodes in region B are on the same side and nearer to the sink node. Sensor nodes of region A can receive data from the event node, however, they will unnecessarily waste their energy in receiving or transmitting the data. In order to reach data to the sink node, data will have to travel more hops if they are sent via the nodes in region A. Thus, when an event occurs, it is always desirable that the data is sent through the nodes in region B. This would save the energy spent for transmission of a piece of data from an event node to the sink node. However, such selective transmission is not supported in the existing SPIN protocols . To overcome this problem, we propose an MSPIN protocol. In few applications such as alarm monitoring applications need quick and reliable responses. Suppose in forest fire warning system, quick response is needed before any disaster occurs. In this case, it is desirable that data must be disseminated towards the sink node very quickly. M-SPIN [7] routing protocol is better approach for such type of applications than SPIN.

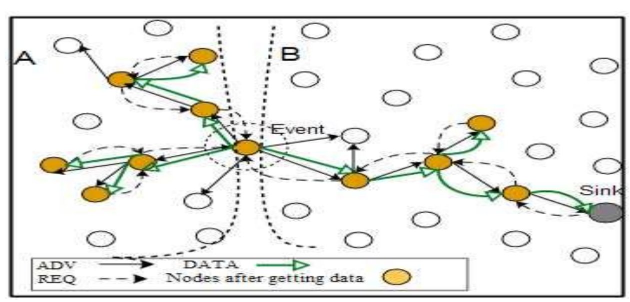

Fig. 5 A data Transmission in WSN

In our proposed protocol, we add a new phase called *Distance discovery* to find distance of each sensor node in the network from the sink node in terms of hops. This means that nodes having higher value of hop distance are far away from the sink node. Other phases of M-SPIN are *Negotiation* and *Data transmission*. On the basis of hop distance, *Negotiation* is done for sending an actual data. Therefore, use of hop value controls dissemination of data in the network. Finally, data is transmitted to the sink node.

2.2.1 Distance discovery

Fig 4 shows the *Distance discovery* phase of M-SPIN. Hop distance is measured from sink nodes. Initially the sink node broadcasts Startup packet in the network with *type, nodeId* and *hop*. Here *type* means type of messages. The nodeId represents id of the sending node and hop represents hop distance from the sink node. Initial value of *hop* is set to 1. When a sensor node receives the *Startup* packet, it stores this hop value as its hop distance from the sink node in memory. After storing the value, the sensor node increases the hop value by 1 and then re-broadcast the *Startup* packet to its neighbor nodes with modified hop value. It may also be possible for a sensor node to receive multiple *Startup* packets from different intermediate nodes. Whenever a sensor node *b* receives *Startup* packets from its neighbors a_i, $1 \le i \le n$, it checks the hop distances and set the distance to the minimum. This process is continued until all nodes in the network get the Startup packets at least once within the Distance discovery phase. After successful completion of this phase, next phase will be started for negotiation. *StartupMsg* structure contains three member variables. *HopTable* structure contains only one member called hop_t to store the hop value at each node.

2.2.2. Negotiation

The source node sends an ADV message. Upon receiving an ADV message, each neighbor node verifies whether it has already received or requested the advertised data. Not only that, receiver node also verifies whether it is nearer to the sink node or not in comparison with the node that has sent the ADV message. If hop distance of the receiving node (own_hop) is less than the hop distance received by it as part of the ADV message (rcev_hop), i.e. *own_hop* < *rcev_ho*p, then the receiving nodes send REQ message to the sending node for current data. The sending node then sends the actual data to the requesting node using DATA message. As soon as a node gets data either from its own application or from other sensor nodes, it stores that data in its memory using the function *storepkt.* Also it uses *setCurrent* function to specify which data is presently residing in its memory. When ADV message is received, then each receiving node first checks its record to ascertain whether it already has seen that data using the function *chkHistory*. Moreover, it calls *setDesired* to indicate which DATA packet it is waiting for.The source nodes which receive the REQ use the function *getCurrent*. It helps to determine whether the received REQ is for the stored data specified by the *setCurrent* function for which the node has sent the ADV. When a requesting node receives any data, it immediately checks whether the data is the same for which it has sent the request using *getDesired* function. The data packet contains the hop distance value along with the information about the event.

2.2.3. Data Transmission

Data transmission phase is same as SPIN-BC protocol. After request is received by the source node, data is immediately sent to the requesting node. If the requesting nodes are intermediate nodes other than the sink node then the *Negotiation* phase repeats. Thus, the intermediate sensor nodes broadcast ADV for the data with modified hop distance value. The sending nodes modify the hop distance field with its own hop distance value and add that in packet format of the ADV message. The process continues till data reaches the sink node. Figure 5 illustrates *Negotiation* and *Data transmission phase*

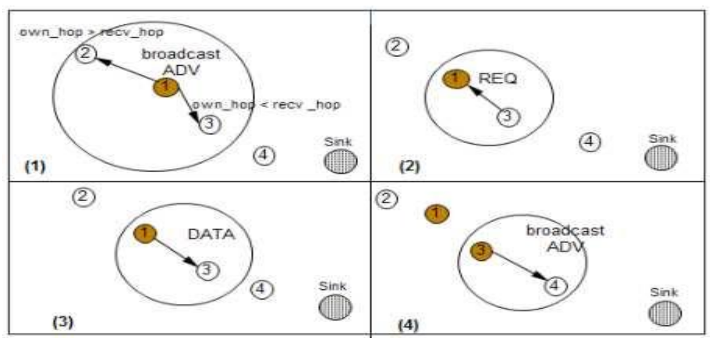

Fig 5. The M-SPIN protocol. (1) Node 1 starts advertising its data to all of its neighbors. (2) Node 3 responds by sending a request to node 1. (3) After receiving the request, node 1 sends the data. (4) Node 3 again sends advertisement out to its neighbors for the data that it received from node 1.

3 Proposed Routing Technique

M-SPIN has already been implemented but it still has a problem of energy as several nodes are traversed multiple times which results in elimination of those nodes from the network. To solve this problem we will use the energy level of the nodes as a parameter. This kind of technique has already been used in the SPIN-EC protocol. In the SPIN-EC protocol we were using energy

ISSN 2229-5518

factor. In this if a node doesn't have sufficient energy for participating in the data transmission, in that case it only accepts the advertised data but does not forward the message to their neighboring nodes. Only in the case if a node has sufficient energy, it would participate in communication otherwise it saves its energy only for its own transmission. To implement our proposed algorithm, we are performing some changes. Firstly we add an energy level as a parameter. The value of the energy level parameter of the node is equal to the battery life of the node. We also define a Threshold Energy for the node. If the value of the energy level parameter is greater than the threshold energy, that node has active participation in the network. If its energy level is less than threshold energy it does not participate in the transmission of the data. Now the proposed algorithm is

Structures:

typedef struct HopTable
{
uint16_t hop-t; };
typedef struct StartupMsg
{
uint8_t type;
uint16_t originNode;
uint16_t counter;
uint16_t energylevel;
uint16_t thresholdvalue; };

Along with these structure, M-SPIN also uses TinyOS route message
structure like TOS_Msg. It also uses System variables like TOS_LOCAL_ADDRESS and for message broadcasting use TOS_BCAST_ADDR address.

Distance discovery:

Distance discovery(){
If (TOS_LOCAL_ADDRESS = = 0) // node 0 is the sink node
ht.hop_t = 0;
call Startup(1); // sink node send Startup message }

If(Startup_TYPE=3 &&TOS_LOCAL_ADDRESS!=0)

{ if (call chkHop (counter) = = SUCCESS)

{ call updHop (uint16_t hc) // increment the hop value } }

// send the message to next neighbor nodes with updated hop value

call forwardHop(uint16_t hc) }

Startup (uint16_t cn) {

StartupMsg *pSMsg =(startupMsg *)&routeMsg.dat[0];

uint8_t length= sizeof(StartupMsg);

pSMsg -> Msg_type = Startup_TYPE;

pSMsg -> node_id = TOS_LOCAL_ADDRESS;

pSMsg -> hop = cn;

pSMsg -> energylevel = current value;

// value of the energy of the battery used in the node

fwdcount++;

if (fwdcount> MAX_HOP)

return; }

If(call SendMsg.send(TOS_BCAST_ADDR, length, &routeMsg) = = SUCCESS) {

Atomic(sendRouteBusy = = TRUE); } }

Negotiation:

Negotiation( ) {

if(energylevel > thresholdvalue)

{ // advertisement for data broadcasts

call ADV_Msg(origin, seq, sender, type);

call wait_REQ( ) // wait for REQ from neighbor

if(received_ packet_type = =ADV) {

if(checkHistory (ADV.origin, ADV.seq) = = SUCCESS)

if(own_hop < recv_hop) {

call setDesired(ADV.origin, ADV.seq);

// request for desired data

call REQ_Msg(origin, seq, sender, type); }}}

if(received _packet_type = REQ){

if(getCurrent(REQ.origin,REQ.seq) = = SUCCESS)

{ if(fwd =TRUE) {

mForward (storedPacket); }

if(snd = TRUE) {

mSend(storedPacket) } } } } }

else

recieve ADV_Msg(origin, seq, sender, type);

//only receive the advertise data but not forward it to its neighbor }

This is the proposed algorithm for energy efficiency in M-SPIN protocol which is different from SPIN-EC protocol because SPIN-EC only conserve energy but does not support selective transmission while M- SPIN protocol support selective transmission as well as conserve energy by using this proposed algorithm.

4 Conclusion

According to this proposed algorithm Energy efficiency of the nodes is increased because energy level factor previously determine the energy level of the nodes, if they have no sufficient energy to participate in the transmission of data in that case they conserve their energy only for their use and remain exist in the network. Secondly it also help in providing the alternate path for data transmission as some nodes are traversed multiple times in case of the M-SPIN but now by determining the energy level the path is diverted. Along with this it has some limitation like the complex computation means calculating the energy level at each node at every time is a tough task. We have to calculate it at every time if in case it cannot be calculated then it is very difficult to implement it and it works same as that of M-SPIN protocol.

5 References

[1] Frederik Armknecht, Alban Hessler, Joao Girao, Amardeo Sarma and Dirk Westhoff," **Security Solutions for Wireless Sensor Networks",** September 2006.

[2] Chris Karlof, David Wagner,"**Secure Routing in Wireless Sensor Networks: Attacks and Countermeasures"** F33615-01-C-1895.

[3] T.Kavitha, D.Sridharan,"**Security Vulnerabilities In Wireless Sensor Networks: A Survey**" 1554-1010 Dynamic Publishers, Inc. 2009. [4] Kazi Chandrima Rahman,"**A Survey on Sensor Network",** JCIT, ISSN 2078-5828 (PRINT), ISSN 2218-5224 (ONLINE), VOLUME 01, ISSUE 01, 2010.

[5] A. H. Azni, Madihah Mohd Saudi, Azreen Azman, and Ariff Syah Johari, "**Performance Analysis of Routing Protocol for WSN Using Data Centric Approach",** World Academy of Science, Engineering and Technology 53 2009.

[6] Jamal N. Al-Karaki, Ahmed E. Kamal**,"ROUTING TECHNIQUES IN WIRELESS SENSOR NETWORKS: A SURVEY"**1536- 1284/04IEEE Wireless Communications, December 2004.

[7] Zeenat Rehena, Sarbani Roy, Nandini Mukherjee ,"**A Modified SPIN for Wireless Sensor Networks",** 978-1-4244-8953-4/11 2011 IEEE.

[8] Yun Zhou; Yuguang Fang; Yanchao Zhang, **"Securing Wireless Sensor Networks: A Survey",** IEEE Communications Surveys & Tutorials, Vol:10, Issue 3, PP: 6 –28, Third Quarter 2008.

[9] V. D. Park and M. S. Corson, **"A highly adaptive distributed routing algorithm for mobile wireless networks,"** in *IEEE INFOCOM '97*, 1997, pp. 1405–1413.

[10] C. Perkins and E. Royer, **"Ad-hoc on-demand distance vector routing,"** in *MILCOM '97 panel on Ad Hoc Networks*, 1997.

[11]D. B. Johnson and D. A. Maltz, **"Dynamic source routing in ad hoc wireless networks,"** in *Mobile Computing*, Imielinski and Korth, Eds. Kluwer Academic Publishers, 1996, vol. 353.

[12] C. Perkins and P. Bhagwat, **"Highly dynamic destination- sequenced distance-vector routing (DSDV) for mobile computers,"** in *ACM/SIGCOMM'94 Conference on Communications Architectures, Protocols and Applications*, 1994, pp. 234–244.

[13] S. Hedetniemi and A. Liestman, **"A Survey of Gossiping and brocadcasting in Communication Networks,"** *IEEE Network*, vol. 18, no. 4, 1988, pp. 319–49

[14]W. Heinzelman, J. Kulik, and H. Balakrishnan, "Adaptive Protocols for Information Dissemination in Wireless Sensor Networks," *Proc. 5th ACM/IEEE Mobicom*, Seattle, WA, Aug. 1999. pp. 174–85.

[15] J. Kulik, W. R. Heinzelman, and H. Balakrishnan, "Negotiation- Based Protocols for Disseminating Information in Wireless Sensor Networks," *Wireless Networks*, vol. 8, 2002, pp. 169–85.

[16] Kemal Akkaya, Mohamed Younis, **"A Survey on Routing Protocols for Wireless Sensor Networks,"** Department of Computer Sciences and Electrical Engineering University of Maryland, Annual ACM/IEEE, August 2000.

[17] Joanna Kulik, Wendi Heinzelman, Hari Balakrishnan," **Negotiation- Based Protocols for Disseminating Information in Wireless Sensor Networks"** Wireless Networks 8, 169–185, 2002 Kluwer Academic Publishers.