

EMOBSPIN: A Novel Information Dispersal Protocol for Mobile Wireless Sensor Networks

Adway Mitra¹, Arpan Roy¹, Arijit Khan¹ and Debashis Saha²

¹Department of Computer Science & Engineering, Jadavpur University, Kolkata 700032, India

E-mail: hoozoy@yahoo.co.in, royarpan@gmail.com, arijit_arijitkhan@yahoo.co.in.

²MIS Group, Indian Institute of Management (IIM) Calcutta, Joka, D. H. Road, Kolkata 700104, India, E-mail: ds@iimcal.ac.in

Abstract

Sensor Protocol for Information via Negotiation (SPIN) is designed for efficient and cost-effective dissemination of information in an energy-constrained wireless sensor network (WSN). Using this protocol, it is possible to send 60% more data than other less sophisticated protocols, such as flooding, gossiping etc. However, SPIN is designed mainly for static WSNs (i.e., where nodes remain in fixed positions), when request-processing delays in server queues are negligible. This paper proposes two extensions of SPIN, namely MOBSPIN, and EMOBSPIN (Enhanced MOBSPIN) for mobile WSNs, where the topology keeps on changing, and queuing delays are non-negligible. The protocol has two variants: MOBSPIN and EMOBSPIN. MOBSPIN is a simple extension of SPIN to accommodate mobility and queuing delays. Here the sending nodes estimate the probability of success before sending data to some target node, and sends the data if and only if the probability so calculated is higher than some fixed cut-off value. EMOBSPIN is a probabilistic protocol, where the probability that the server sends the data is equal to the probability of success. The protocols try to keep down the number of futile messages, keeping in mind the acute shortage of energy in sensor nodes. Simulation analyses of the three protocols indicate their comparative performance, which, in turn, help us know their respective suitability for differing environments.

Keywords: wireless sensor networks, mobile sensors, information dissemination, SPIN, MOBSPIN, EMOBSPIN, Cut-off Probability, Percentage of Success, Futile Messages, Redundant Messages.

I. INTRODUCTION

A wireless sensor network (WSN) [1] is a network of spatially distributed autonomous tiny devices using sensors to cooperatively monitor physical/ environmental conditions. Each node in a WSN is typically equipped with a radio transceiver or other wireless communications devices, a small microcontroller, and an energy source, usually a battery. The key constraint in all sensor networks is battery power. Hence every protocol involving sensor networks should aim to minimize battery usage. In the MOBSPIN protocols the aim is to minimize battery usage by trying to reduce *futile messages* (i.e. data packets which are sent by the server but not received by the client) and also *redundant messages* (i.e. data packets that are sent by a server node to a client node that has already got it from another server node).

Another constraint that arises in case of mobile sensor networks is topology information. Since every node is moving about, the topology of the network keeps on changing all the time, and so a particular node cannot keep track of other nodes. One method to solve this problem may be that each node should communicate with adjacent nodes to make itself aware of the network topology each time it wants to transmit. But this process will be too slow for a rapidly changing topology, and also the overhead will be too high. Another way out is that, every node can know the network topology all the time if GPS connectivity is available. But this service will not be available everywhere. Also, we have some route-discovery protocols for Mobile Ad-hoc Networks (MANETs) [2] like the well-known

Ad-hoc On-demand-Distance Vector (AODV) [3]. But in case the nodes are continuously moving about fast, this scheme will not be fast enough. Besides, since the sensor nodes are low in processing abilities, it is important to reduce computational jobs as far as possible. So here the simple broadcast mode of communication has been used, to relieve the nodes of the need to discover the topology. In this mode, all the nodes that occur within the broadcast range of a particular node at any instant are the neighbors of that node for that instant.

There are quite a few areas where these novel protocols can be used, such as *Monitoring of Wildlife* (e.g., Zebra Net Project [5] and Sea-bird Monitoring in Maine's Great Duck Island), *Vehicle Data Stream Mining System (VEDAS)* [6] etc.

An outline for the rest of this paper is as follows. In the next section (Section II) discussed with references are the different ideas proposed that are relevant with the contents of this paper. In Section III the protocols MOBSPIN and EMOBSPIN (Enhanced MOBSPIN) are described in details. Section IV discusses the improvements over the original SPIN protocol. In Section V some mathematical calculations used in the protocols are shown briefly. Section VI describes the simulation set-up. In section VII the simulation results are shown and analyzed with the help of graphs. Also the three protocols are compared according to different requirements. Section VIII presents an equation involving three key parameters, obtained by regression analysis of simulation results. Finally a conclusion is provided to the paper and also areas of improvements are identified in Section IX. The references are listed in Section X.

II. RELATED WORK

Several information dissemination protocols have been proposed earlier for wireless sensor networks (assuming the nodes to be static). Some of the simplest are *flooding*, *gossiping* [1] etc. These methods suffer from some problems as described in [1]. Much improved performance is provided by other algorithms like SPIN [9], LEACH [10], and PEGASIS [11]. SPIN (Sensor Protocols for Information via Negotiation) is a 3-way handshaking protocol developed mainly for static sensor networks. The idea is to minimize battery power usage by transmission of data packets. A node (server node) that collects some information first advertises it to other nodes in the network. The interested nodes (client nodes) send requests to the server, which then send the data to these nodes. This paper describes a way to extend this idea to accommodate mobility of the nodes. Another aspect of this work is mobility. Most of the works done on sensor networks so far do not consider mobility, as sensor networks often consist of static nodes. So, **mobility models** are not easily available for sensor networks. However, a survey of different mobility models for MANETs is presented in [12]. Some of the most important ones are the Random Walk Model and also the Random Waypoint Model (RYP). Here the simple random-walk model has been used. Apart from mobility model, the other issue in mobile networks is **mobility management**. In [13], Muneeb Ali et al have considered mobility in sensor networks, and have suggested a centralized database visible to all the layers from which a node can get information about the motion of others. But in the MOBSPIN algorithms this source of overhead has not been used. For these algorithms, a server node does not need to be aware of the motion or position of other nodes at the time of advertising which is done in broadcast mode. In other interactions (like sending of REQ, DATA or INTR messages) which are done by unicasting, the sending node knows the position of the destination node (with some uncertainty).

III. PROTOCOL DESCRIPTIONS

In this section MOBSPIN and EMOBSPIN are described, along with the conditions required for their successful running.

This paper considers the nodes to be moving in a 2-D area of fixed dimensions. At any instant, a node can have a speed from 0 to V_{max} with uniform probability, and any direction with uniform probability. Also, the node has no control on its motion, as it is possibly attached to some moving entity. Each node knows its own position at every instant, but nothing about the positions of the other nodes. This may be possible using fixed “seed” or “anchor” nodes or by using “Mosaic Localization” [15]. Also, each node has a unique identification number. If a node “A” lies within the radio communication range of another node “B”, then A and B are said to be neighbors. So the topology is like an undirected graph, which is assumed to be connected all the time by suitable topology control protocols.

A. MOBSPIN

When a node gathers some information that might be important for other nodes, it broadcasts an ADV message in the meta-data form so that all other nodes which are within its antenna range can receive the message. The message also contains the sender’s (i.e., *global server*’s) serial number, position (X and Y

coordinates corresponding to a fixed common reference) and timestamp (i.e., time of sending). If a neighboring node (positioned within the antenna reach of the server) receives the ADV message, it rebroadcasts the same ADV message to its neighbors after adding its serial number, position etc. This is done to keep a record of the “routing history” of the packet. It acts as *local server*. The neighbors too act similarly on receiving the broadcast. It is assumed the presence of a MAC protocol which ensures that no interference occurs during broadcast. Also, if a particular node is interested about the data, it sends a REQ message to the server (which had sent it the ADV message), informing the server of its serial number, position and time of sending the REQ message.

It should be noted that, a particular node may get ADV messages for the same data-item from different sources. If it already has the data (for example, the global server which has the data, will get an ADV from each neighbor as soon as it sends them an ADV) or if it does not need that data item, it simply ignores the broadcast. But *if it needs the data being advertised, it will send a REQ to every node that advertises to it, (provided that node has not received the ADV from it. It can verify this by checking the route history of the ADV packet)*. It also enters the sender’s details (serial number and position) in SERVTAB. This is done because, in a MWSN, it can never be sure whether it will remain within the antenna range of a particular server node when that server node actually sends the DATA. However, *every node should broadcast ADV message for a particular data only once*.

When a server obtains the data in question, it starts processing its REQ stack (REQTAB). Processing of each REQ takes some time. Hence queuing delay can come into play. It is possible that when a node processes a REQ message and sends it the corresponding DATA message, the target node may have moved outside its transmission range. However, since a node sends REQ messages to several nodes at least one is expected to send it the DATA message.

To minimize energy loss by redundant messaging, when a node gets the DATA message, it sends an INTR message to other servers to which it had previously sent REQ messages. It indicates that no further sending of the data is required. A node stores both REQ and INTR messages in separate queues. When a REQ is dequeued for processing, the node scans the entire INTR queue to check if there is

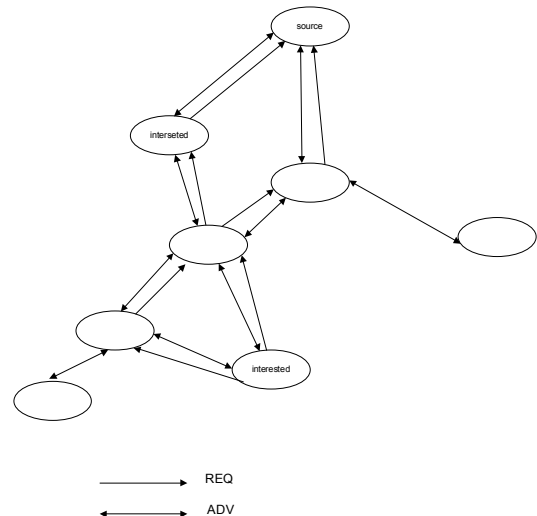


Figure 2: Message Graph

any INTR from the requester. Considering the limited power of the sensor nodes, it will be realistic to consider that while broadcasting or receiving process is ON in a particular node, it cannot collect (i.e. sense) any new data item.

In Figure 2, we have shown the situation. Since a node always rebroadcasts an ADV message after receiving it, ADV messages are shown by bidirectional arrows. The REQ messages are sent by the clients to the servers, so they are shown by unidirectional arrows. The figure also makes it clear that all the nodes that participate in the process of information dissemination do not need the data being dispersed. In fact, only the two nodes marked “interested” need the data. This may be considered a disadvantage, but it is necessary for successful information dissemination since not all the interested nodes can be one-hop neighbors of the global server node.

When a server obtains the data in question, it starts processing its REQ list. Processing of each REQ takes some time. Now, when the server has processed the REQ message from a client the client may have already moved out of its antenna range. The server knows its own current position, the position of the client at the time it had sent the REQ message, and the time that has elapsed since then. Let this time be t . (It is assumed that the clocks of all nodes are synchronized to extremely high degree of accuracy.) So, it knows the geometry of the circle which encloses the area within its antenna range, (this circle has the server’s present co-ordinates as center and its antenna range as radius) and also the circle within which the client must be present now. This circle has, as its center, the client co-ordinates at the time of sending the REQ message (these co-ordinates are known to the server as they were recorded in the REQ message), and the radius r given by $r=v*t$ where v is the maximum speed at which a node (or rather the entity carrying the node) can travel. Four cases may arise:

- a) The two circles do not meet (i.e. the client is definitely beyond the server’s range).
- b) The second circle lies within the first circle i.e., the client surely lies within the server’s transmission range.
- c) The two circles enclose some common area.
- d) The first circle lies within the second circle.

In cases c) and d) there is a probability p ($0 < p < 1$) that the client will still be within the server’s range. In general, this probability is given by:

$p = (\text{area of the second circle which is also within the first circle}) / (\text{area of the second circle})$. Since the server knows all the required parameters for both circles, it can easily calculate the required probability by using an algorithm based on standard formulae on co-ordinate geometry. *If this probability p is greater than some fixed value P (called cut-off probability) it will transmit the DATA message towards the possible position of the client. If $p < P$, then sending will not be done.*

This probabilistic sending may have some negative effects (for example a client that was actually within the antenna range may not receive from that server) but it will certainly reduce the number of futile DATA messages,

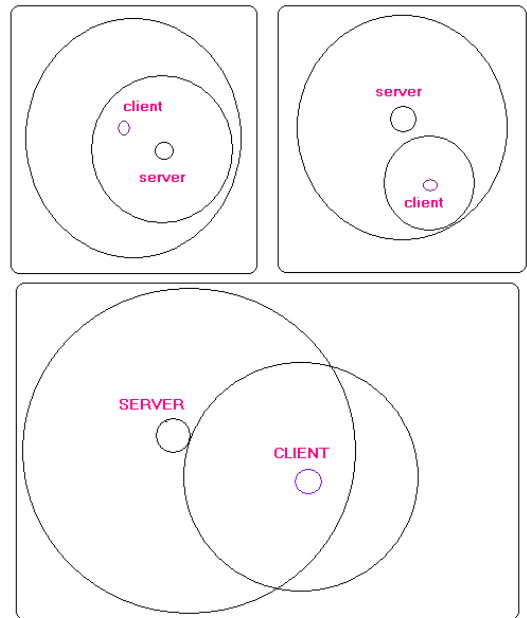


Figure 3: Situations where client is reachable from server

which is very important as the DATA messages are costly. Moreover, as a client requests multiple servers, it is quite likely that it will get the message from another server).

Since a client sends REQ messages to multiple servers it is possible that several (if not all) of these servers will send it DATA, which causes redundant messages, and a waste of precious energy. So, when a client node obtains the desired data it broadcasts an INTR message as earlier so that all the neighbors who might be servers and have this client’s REQ in their REQ-stack, will not send it the DATA message.

A parameter whose value is crucial is the *Cut-off Probability* P . It can be shown mathematically that the expected value of ‘ p ’ (the probability of success) is about 0.27. We can intuitively understand that, if P is very low the number of futile DATA messages will not be significantly reduced. On the other hand, if it is very high, this end will be achieved, but a new problem will arise. The percentage of success will fall drastically as many a client will not get the data they had requested. So by trade-off, an optimal value of P needs to be found. In the computer simulation described below we have found that optimal value for our system.

ADV messages are broadcasted by servers, while REQ, DATA and INTR messages are sent by point-to-point communication (unicasting). To enable point-to-point contacts, ADV and REQ messages always contain the sender’s position at the time of sending the message. However the messages have to be sent after estimating the changed position of the target node while the sender was processing the previous message. So the mode of contact does not remain pure point-to-point, but covers a small angle.

It may be noted that broadcasting in wireless networks require omnidirectional antenna while unicasting requires unidirectional antenna. So the sensor nodes for such a scheme should be provided with both an omnidirectional and an unidirectional antenna.

The approach adopted in MOBSPIN is not completely probabilistic. Rather, when the probability of success for a DATA packet if it is transmitted (given by the ratio of the common areas of the circles to the server circle area) exceeds a predefined value, the packet is sent, but not otherwise. We can extend the ideas of MOBSPIN to a pure probabilistic approach, where the server node first calculates the probability of success (as stated above). If this probability is 'p', then the server transmits with probability 'p'. This approach is somewhat similar to p-persistent CSMA, where the node, on finding the channel silent, transmits with probability 'p' and defers transmission till the next slot with probability q=1-p.

IV. IMPROVEMENT OVER "SPIN"

The 3 protocols presented above are better than the original SPIN protocol in the following ways:

- 1) Mobility of the nodes is taken into account.
- 2) No knowledge about the network topology is required for a node.
- 3) An inherent problem of SPIN is that, data delivery is not guaranteed. If some nodes far away from the source node (global server) are interested in the data but the intermediate nodes on the route are not, these nodes will not get the data. But in this case, each node, on receiving the ADV is required to make one ADV to advertise to all its neighbors, even if it is not interested in the data. Also, if some neighbor sends it REQ, it has to forward that to its server, and also pass the DATA to the interested neighbor.
- 4) Here processing delays in the server nodes are taken into account.

V. MATHEMATICAL ANALYSIS

The theories mentioned above require some mathematical calculations, of which the most important one is the derivation of a formula for the probability of success when the two circles are known.

Let the server circle S have center (xs, ys) and the client circle have the center (xc, yc). The radius of the server circle is R (radio transmission range) and that of the client circle is 5*t (t is the time delay between the client sending the REQ and the server processing it and the maximum speed of the client is 5 m/sec). So, the equations of the two circles are given by:

$$S: (x-x_s)^2 + (y-y_s)^2 = R^2$$

$$C: (x-x_c)^2 + (y-y_c)^2 = 25*t^2$$

Three cases may arise:

Case 1: S and C do not intersect

$$\text{i.e. } \text{dist}((x_s, y_s), (x_c, y_c)) > R + 5*t$$

In this case, the common area of the two circles=0. Hence the probability that a DATA packet transmitted by the server will reach the client is 0.

Case 2: C lies wholly within S

$$\text{i.e. } \text{dist}((x_s, y_s), (x_c, y_c)) + 5*t > R$$

In this case, the common area of the two circles= area of C= $25*\pi*t^2$. So, in this case, if the server transmits DATA packet, probability of success= 1.

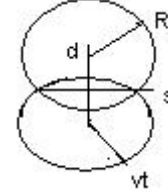
Case 3: C and S partially overlap.

In this case, the common area is given by the common area given by:

$$c = 25*t^2*\text{asin}(\theta) - (1/2)*5*t*s*\cos(\text{asin}(\theta)) + R^2*\text{asin}(\alpha) - (1/2)*R*s*\cos(\text{asin}(\alpha))$$

Where $\alpha = s/(2*R)$, $\theta = s/(10*t)$, $s = \sqrt{4*R^2 - ((R^2) + (d*d) - 25*t^2)/(d*d)}$ and $d = \text{dist}((x_s, y_s), (x_c, y_c))$.

[Here asin means sine inverse]



Successful transmission of DATA can occur only if the client lies within this common area of the two circles, while in general it can be anywhere within the circle C. So, in this case the probability of success= c/(area(C)).

[NOTE: We do not consider the case where S lies completely within C as R is large enough compared to 5*t].

Then the area of overlap A is given by

$$A = A_1 + A_2$$

Where

$$A_1 = R^2*\text{asin}(s/(2*R)) - (s/4)*\sqrt{4*R^2 - s^2}$$

$$A_2 = 5*t^2*\text{asin}(s/(2*5*t)) - (s/4)*\sqrt{4*5*t^2 - s^2}$$

Then the probability of successful transmission= A / ($\pi*R^2$).

So, the server node should be programmed to calculate the probability of success by using the formulae mentioned above, and then it should transmit or abort according to the protocol that it is running. It is extremely difficult to accurately predict the results by mathematics. But we can try to estimate the probability of success in a typical case.

VI. SIMULATION STUDY

We consider a test-bed of dimension [40 m x 40 m]. There are 20 sensor nodes which move about randomly within this range at speeds up to 5m/sec (which is assumed to follow uniform probability distribution). Also, we assume that the direction of motion is randomly selected, by choosing an angle from 0 to 2π , again assuming uniform distribution. If a node reaches an edge, it reflects back into the field. The antenna range of each node is 20 m. Randomly selected nodes collect the data (i.e. become global servers) at Poisson-distributed times. Since all data items are sent as electromagnetic waves traveling at the speed of light, the message propagation delays are considered to be 0. We have implemented the simulation program in C language under LINUX environment. We also intend to re-evaluate the simulation using NS2. Detailed lists of all the parameters along with their values are presented below:

NUMBER OF NODES	20
DIMENSIONS OF TESTBED	40mX40m
ANTENNA RANGE	20m
SPEED OF PROPAGATION OF PACKETS:	$3 * 10^8$ m/s
TIME OF PROPAGATION	0
INITIAL ENERGY IN EACH NODE	10 J
SPEED OF EACH NODE	0-5m/sec
POWER FOR TRANSMISSION	600mW
RADIO SPEED	1 MBPS
ENERGY TO SEND "REQ"	0.0024J
ENERGY TO SEND "INTR":	0.0006mJ
QUEUING DELAY	0.5 sec(FOR PROCESSING "REQ" MESSAGES)
ENERGY TO SEND "DATA"	108 mJ

Since there are 20 nodes each is assigned a serial number to identify it. The x and y coordinates of a node are also expressed in binary number, approximated to the nearest integer.

ADV MESSAGE	
Metadata size	16 bytes
Serial Number of advertising node: [log 20]	5 bits
Time	15 bits
X coordinate: [log 40]	6 bits
Y coordinate : [log 40]	6 bits
Total	16 bytes + (5+15+6+6) bits = (16 + 4)bytes = 20 bytes

REQ MESSAGE	
Serial Number of requesting node	5 bits
Time	15 bits
X-coordinate	6 bits
Y-coordinate	6 bits
Total	(5+15+6+6) bits = 4 bytes

Due to queuing delays there will be failures in data-sending, as by the time the local /global server processes the REQ and sends it to the requesting node, the requesting node might be outside the server's reach due to the motion of either the sender or the recipient or both. the data. The probability could become higher if we use higher antennae range, but increasing this may not be a viable option given the energy limitations of sensor nodes. In MOBSPIN, the concept of probabilistic decision by the servers has been used. (Each server transmits a DATA message only

after considering the possible position of the client.) As the Cut-off probability is increase, percentage of success decreases, however the number of futile data messages also decreases. While low cut-off probability is expected to enhance success rate, higher values of Cut-off Probability ensure very low (even nil) futile messaging, thereby reducing battery wastage. It is important to appreciate that both high success-rate and low battery-wastage are important in sensor networks. Hence some trade-off is required. Hence we can define a performance metric by giving equal weight to both this metrics.

On the basis of the output data obtained from the program, results are plotted in three graphs, namely Figure 4 (Percentage of Success vs. Cut-off Probability), Figure 5 (Percentage of Futile Data Messages vs. Cut-off Probability) and Figure 6 Utility vs. Cut-off Probability).

VII. RESULTS and DISCUSSION

In this section, the results of the initial simulations in C are shown. The main parameters are Percentage of Success (PoS) and Percentage of Futile Messages. For MOBSPIN, first the behaviors of the number of futile messages and percentage of success (percentage of the number of times the nodes that are interested in a particular data-item actually get it) vs. Cut-off Probability (COP) are shown.

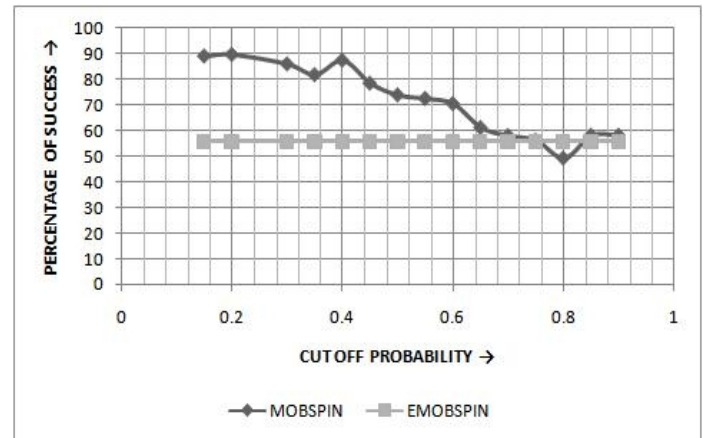


Figure 4): Percentage of Success vs. Cut-off Probability (CoP)

Fig. 4 shows that the percentage of success falls as the CoP is raised for MOBSPIN. As EMOBSPIN has no cut-off probability, the curve is parallel to the X-axis. Fig. 5 shows that the percentage of futile transmissions of DATA packets is dramatically reduced as the CoP decreases. From the graph it is clear that EMOBSPIN performs better than MOBSPIN in this respect.

Now, for MOBSPIN, whether a high or low CoP is to be chosen depends on which is more important- high percentage of success or longer lifetime? The answer is obviously going to be different for different applications. For this type of analysis, a parameter is defined:

$$UTILITY = \infty * PoS + (1 - \infty) * (100 - PoFMsg)$$

where ∞ is a measure of weightage.

Fig.6 displays its behavior with the variation of Cutoff Probability.

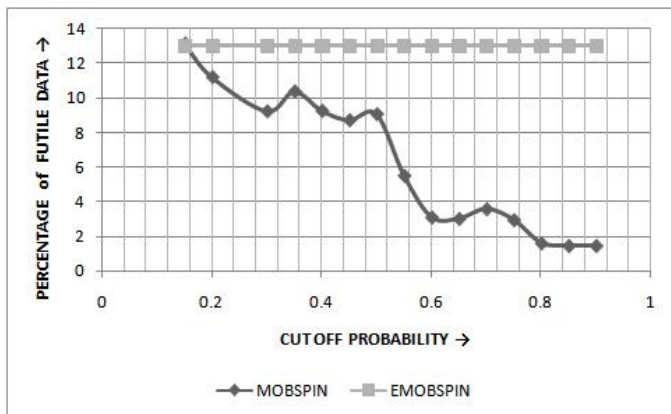


Figure 5) Percentage of Futile Data Messages vs. Cut-off Probability

VIII. RELATION BETWEEN PARAMETERS

The three most important parameters in the schemes MOBSPIN* and MOBSPIN++ are the Cut-off Probability (CoP), the percentage of success (PoS) and the percentage of the Data Messages that are futile (PoFD). It can be easily understood that, higher the CoP, lower the PoS and PoFD. From the data obtained by simulation, the parameters can be fitted into the following equation to relate the three parameters:

$$CoP = (0.017 + 0.038 * (PoS) - 4.075 * 10^{-4} * (PoS)^2) * (-0.054 + 0.336 * (PoFD) - 0.028 * (PoFD)^2)$$

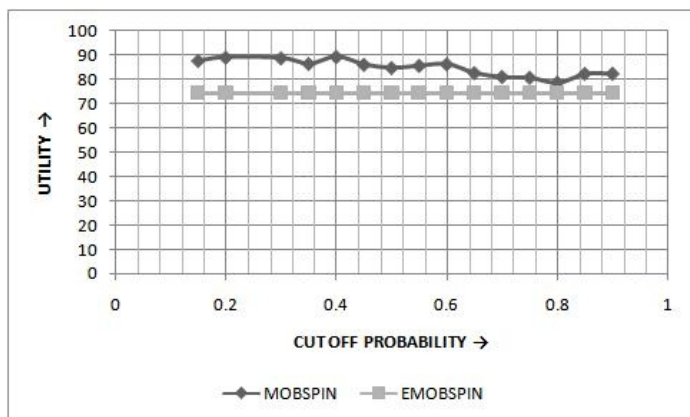


Figure 6): Utility vs. Cut-off Probability

IX. CONCLUSION

To sum up, in this paper, an innovative extension of the well known SPIN protocol is presented. It is named EMOBSPIN for mobile wireless sensor networks. Where the percentage of success is more important than the lifetime of the network (e.g., ROBOCUP, traffic incident detection), MOBSPIN protocol is applied with low or even zero cut-off probability. On the other hand, if the lifetime of the network and costliness of futile data sending is more important than the percentage of success (e.g., Monitoring of Wildlife), EMOBSPIN is implemented. A major scope of improvement in the model is the employment of a

better mobility model, since this model is oversimplified and may be more important than the percentage of success (e.g., Monitoring of Wildlife), EMOBSPIN is implemented. A major not be applicable to real-life mobile sensor networks. Ways of deployment and actual implementation of this kind of network also needs studying. Besides, this protocol is primarily for the Network Layer. It should be complemented with efficient MAC protocols (like CSMA) so that interference can be avoided at the time of broadcasting ADV messages. Finding a suitable MAC protocol is an open research issue.

X. REFERENCES

- [1] I.F. Akyildiz, W. Su, Y.Sankarasubramaniam, E. Cayirci, "A Survey on Sensor Networks", IEEE Communications Magazine, pp 102-114, August 2002.
- [2] J.Liu, S.Singh, "ATCP: TCP for mobile ad hoc networks" IEEE Journal on Volume 19, Issue 7, Jul 2001 Page(s):1300 - 1315
- [3] Ian D. Chakeres and Elizabeth M. Belding Royer. "[AODV Routing Protocol Implementation Design.](#)" Proceedings of the International Workshop on Wireless Ad Hoc Networking (WWAN), Tokyo, Japan, March 2004.
- [4] Sameera Poduri, Gaurav Sukhatme, "Con strained Coverage for Mobile Sensor Networks", IEEE International Conference on Robotics and Automation, pages 165-172, 2004, New Orleans, LA, USA.
- [5] Princeton University, "The ZebraNet Wildlife Tracker", Princeton Weekly Bulletin, November, 2002
- [6] M. M.Gaber, S. Krishnaswamy, and A.Zaslavsky, "A Wireless Data Stream Mining Model", Wireless Information System, 2004
- [7] J. Kulik, W. Heinzelman, and H. Balakrishnan, "Negotiation-based protocols for disseminating information in wireless sensor networks.", *CiteSeer*, 1999, pp 152-160.
- [8] W. Heinzelman, A. P. Chandrakasan and H.Balakrishnan, "An Application-specific Protocol Architecture for Wireless Microsensor Networks", IEEE Trans. Wireless Comm., October 2002, pp 660-670
- [9] S. Lindsey, C. Raghavendra, K.M. Sivalingam, "Data-gathering Algorithms in Sensor Networks using Energy Metrics", IEEE Trans. Parallel and Distr. Systems, September 2002, pp 924-35.
- [10] T. Camp, J. Boleng and V.Davis, "A survey of Mobility Models for Ad-hoc Network Research", Wireless Communication and Mobile Computing (WCMC), vol 2, no.5, pp 483-502, 2002.
- [11] Muneeb Ali, T. Voigt, Z.A. Uzmi, "Mobility Management in Sensor Networks"
- [12] R. Rajagopalan and P.K. Varshney, "Data Aggregation Techniques in Sensor Networks- A Survey", IEEE Comm. Surveys, 2006, pp 49-63.
- [13] J. Kang, D. Kim, S. Ahn, "Mosaic Localization for Wireless Sensor Networks", Information and Communication University", Republic of Korea.