A Neural Network approach for Wireless sensor network power management

Ahmad Hosseingholizadeh
Dr. Abdolreza Abhari
Department of Computer Science
Ryerson University
Toronto, Canada
{ahossein, aabhari}@scs.ryerson.ca

***Abstract*—In this paper we introduce a new approach for wireless sensor network power management which is based on Neural Networks. In this new approach an intelligent analysis is used to process the structure of a wireless sensor network (WSN) and produce some information which can be used to improve the performance of WSNs' management application. We applied our intelligent method to our previously proposed management approach which uses the concept of Multi-Agent systems for WSNs' management and observed the improvement of the performance. Wireless sensor networks need to be managed in different ways; e.g. power consumption of each sensor, efficient data routing without redundancy, sensing and data sending interval control, etc. The random distribution of wireless sensors, numerous variables which affect WSN's operation and the uncertainty of different algorithms (such as sensors' self-localization) give a fuzzy nature to WSNs. Considering this fuzzy nature and numerous details, a neural network is an ideal tool to be used to cover these details which are so hard to be explicitly discovered and modeled. In this paper we introduce our Neural Network-based approach which results in a more efficient routing path discovery and sensor power management.**

We define a set of attributes based on sensors' location and neighborhood and we use them as inputs of our neural network and the output of the neural network will be used as a factor in the route path discovery and power management. We designed a simulator based on our approach and observed the effect of our method on wireless sensor network lifetime and sensor power consumption which will be presented in this paper.

Keywords- wireless sensor network; WSN; power management; routing; neural networks; agents.

I. INTRODUCTION

Today the application of Sensor Networks can be seen in different aspects of our lives; fire detectors, security sensors, etc, are widely used. From this category, Wireless Sensor Networks (WSN) seems to be the next generation of sensor networks which is going to be widely used in the near future. In wireless sensor networks there are several wireless sensors which are capable of sensing a special phenomenon in the environment and send the data back to one or several base stations. The main feature of WSN that makes it unique is its flexibility in terms of the shape of the network and mobility of the sensors. Without any wires, WSN can be deployed in areas where regular sensor networks cannot operate. Also the self-shaping feature of WSN, along with the freedom of the wireless sensors movement makes it an ideal tool for the situations where the sensors are mobile. Having these features, WSN is used in medical applications, military purposes, disaster area monitoring, etc [3, 4].

The flexibility of wireless sensor networks comes with a series of challenges. Since wireless sensors are not physically connected to any central source they are completely dependent on their battery to operate; also wireless sensors positions are not determined prior to the network deployment, thus sensors should be able to operate in a way that can automatically generate an optimum routing path and deliver the sensed information back to the base-station. Base-station integrates the received data and applies a process over it and sends the results to be viewed by a user or for further processing.

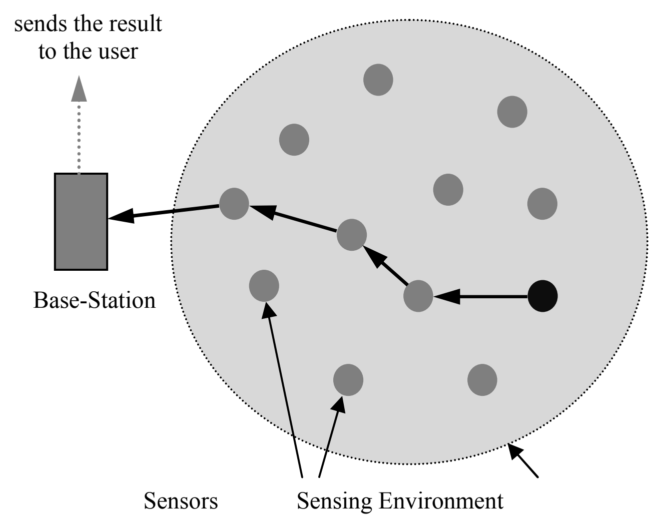

Figure 1. Wireless sensor network general model

Each wireless sensor node is not physically connected to any source of power, thus its own battery is the only reliable power supply for it. Sensor nodes are also constrained on bandwidth. Considering these two limitations, routing and sensing algorithms that use innovative methods to preserve the power of the sensors are required [7]. Since the lifetime of the network is highly dependent on the lifetime of the sensors' batteries [2], preserving the energy in the sensors will increase the lifetime of the network.

Different techniques have been proposed to increase the lifetime of the wireless sensor network. Since most of the power consumption of each sensor is due to sensing and routing operations, many of the proposed techniques try to optimize these two tasks. Some approaches update the routing path when a sensor node in a path is low in power [10] thus that they would exclude the node from the routing path and preserve its energy. When there are multiple paths to route the data, some methods try to augment the flow on paths whose minimum residual energy after the flow augmentation would be the largest [6, 13]. Many techniques such as MCFA, GBR and Rumor routing use the shortest path method to reduce the communication and energy consumption. Many of WSN management techniques use an agent-based method to manage the wireless sensor network and its power consumption [1, 8, 11, 12, 14].

In this paper we introduce our new approach which uses a neural network-based strategy to preserve the sensors' power and increase the lifetime of the network. Our method can be added and applied to any power efficient algorithm to enhance its efficiency. We'll also propose a method to apply our algorithm to power management approaches which use the shortest path method in their route discovery. As an example, we applied our new neural network method to our previously proposed power efficient algorithm which we introduced in our previous research on wireless sensor networks management [1]. In our previous research we proposed an agent-based model to facilitate the network management and power usage control. We designed a simulator to apply our new approach to a series of randomly generated wireless sensor networks which are created by random scattering of wireless sensors and we observed the results which will be discussed in this paper.

The rest of the paper is organized as follows. First we review our previous agent-based algorithm to set a basis to introduce our neural network approach. After that we introduce our new neural network-based approach which enhances the shortest path-based routing, task association and power management methods. In section IV we discuss the method of applying our neural network algorithm to our agent-based approach and its results. Finally, we'll discuss our conclusion and our plan for future research.

II. AGENT-BASED WSN POWER MANAGEMENT

In this section we review our agent-based wireless sensor network management approach which we introduced in our previous paper on WSN [1]. This approach is based on three main concepts: *the software architecture of the working application on the WSN*, *agents' distribution algorithm* and *sensors' power management*.

A. WSN manager application software architecture

In our proposal we defined 2 architectures for two kinds of applications that work in wireless sensor networks [1]. One of these architectures belongs to the base-station software application and the other one belongs to the software applications which run on wireless sensors.

The *base-station application* is located in the base-station and it is responsible for communicating with the whole network, sending instructions to sensors, integrating received data, etc.

Wireless sensors' applications are located inside each wireless sensor, and they are responsible to control the local operation of each sensor, send the sensed data to the base-station and respond to specific instructions that they receive from the base-station. In our proposal, each sensor that contains an agent would be considered "Fully Functional" (meaning that it participates in both sensing and routing). Other sensors are either Active or Standby. Active sensors participate in routing the data to the base-station but they don't operate as a sensor to sense a phenomenon in the environment. Standby sensors don't participate in routing or sensing tasks.

For both of these software application types we considered a 3-tier architecture which separates the *node-level* operations of the network from the *agent-level* operations: *UI-tier*, *application-tier* and *data-tier*. UI-tier is responsible to receive any inputs from and shows any outputs to the user. Application layer is responsible for *Agent-level* operations which deal with the network as a set of agents; routing paths and inactive nodes are hidden from this level. Data-tier is responsible to translate the Agent-level instructions into *Node-level* operations which deal with nodes localizations, route path update and other low-level tasks.

B. Agents' distribution algorithm

Usually a wireless sensor network is created by randomly scattering a number of wireless sensors in an environment. After deployment these sensors should automatically generate routing paths to the base-station and start sending the sensed data to the base-station. If two of these sensors are located at a same place, they'll perceive the same phenomenon and thus they'll generate redundant data. Since the ratio of power usage in a standby node to a fully-active node is 1:200 [12], sending this redundant data to the base-station will consume sensors power and bandwidth and thus will result in decreasing the networks lifetime.

To address the former issue, in our previous research [1] we proposed a new approach to deactivate redundant sensors and preserve the power of the sensors in the wireless sensor network. We defined a 2D grid over the sensing environment. Each unit of the grid has an equal length and width of $\sqrt{2}R$ in which R is the Sensing Range of each sensor. Using our algorithm, the base-station finds the closest sensors to each one of these grid-points and sends the Agent-Activation command only to them; thus only a part of the wireless sensors would be activated (Figure 2).

Using this distribution technique, it is possible to activate a subset of nodes and keep others in the standby mode to be used as reserved nodes for later usages. Also this technique makes it possible to cover most of the environment (which is possible to cover) and it is compatible with the software-architecture that we discussed in the previous section.

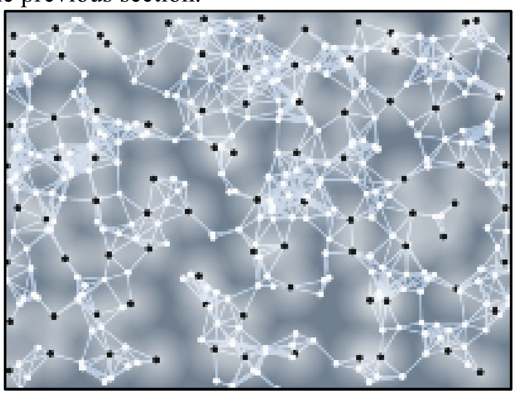

Figure 2. Black dots represent the nodes that contain agents. The glowing area around each agent represents its sensing range.

C. Sensors' power management

One of the most critical issues about WSN is the power issue. Power management techniques in WSN try to preserve network's nodes' powers and increase the lifetime of the network. In our previous research [1] we described a new approach which is based on our agent-based architecture. To represent our approach, we describe a step-by-step scenario of a WSN deployment and setup.

When sensors are scattered in the sensing environment the lifetime of the network begins. The base-station sends a flooding message into the network; as this flooding message passes over different nodes it sets the gradients toward the base-station, thus each sensor that receives the flooding packet can reach the base-station based in its own link-state. After receiving the flooding message each sensor creates a message containing sensor's neighbors' IDs and sensor's position and sends it back to the base-station. Having all the nodes neighbors' state, base-station is able to create the adjacency matrix and find the shortest paths to each one of the sensors. For the weight of all the connections between sensor nodes and we assumed the same equal value of 1. After finding the shortest paths the base-station distributes the agents in the WSN using the algorithm described in the previous section and agents start their sensing operation.

We defined 4 thresholds for the sensor nodes: *power threshold*, *migration threshold*, *sensing threshold* and *routing threshold*. We defined the relation between these thresholds as *power threshold* < *migration threshold* < *sensing threshold* < *routing threshold*. As the sensors operate and use their power, they monitor their own power and whenever their remaining power meets each one of these thresholds the sensor runs a power preserving algorithm. These algorithms are described below.

The highest threshold is the *routing threshold*. As a sensor is using its power this is the first threshold that it meets. When the power of a sensor meets the routing threshold it sends a message to the base-station requesting to remove that sensor from Adjacency Matrix and update the routing paths. This means that the base-station removes it from routing paths and thus this node consumes less energy and continues to operate. If removing this node causes some other nodes to be unreachable, the base-station rejects the request and keep the routing paths as before.

When a node's power reaches the *sensing threshold* it stops the sensing operation. This means that it only keeps on routing the data. Considering that only those nodes which contain an agent are sensing, this is obvious that this threshold only affects the agent-nodes.

When a node's power (which contains an agent) reaches the *migration threshold* it sends a migration message to the base-station, requesting to move the agent to another node. Base station would search the nodes to find the closest node to that agent and relocates the agent. This is operation is also only applicable to those nodes that contain an agent.

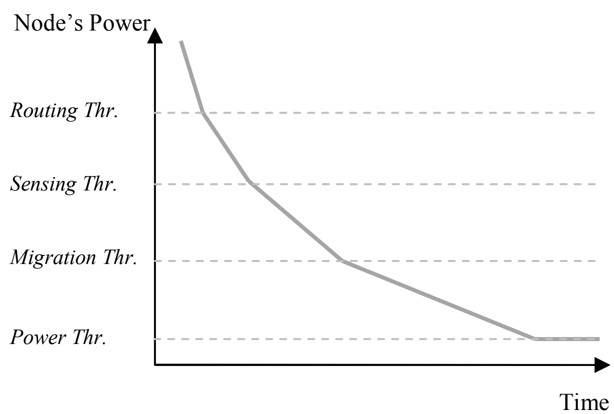

Figure 3. A sample of the effect of Thresholds over the sensors' power usage. As a sensor operates, it meets each of these thresholds and in each level, it will reduce the power usage of the sensor and increases the sensors lifetime.

If an agent reaches the *power threshold* it will stop all its operations and go to the standby mode to reserve its power for critical operations forced by base-station.

Aside from these thresholds we also defined an operation called *Neighbors Power Comparison* which helps the node to preserve its power. On specific intervals called *power broadcast intervals* each node broadcasts its remaining power to all its neighbors. Each node that receives these remaining powers will use them to compare its own power-level to its neighbors and update its own sensing interval using the following formula:

$$New\ Interval = \left\lfloor \left(\frac{Neighbors'Average\ Power}{Sensor's\ Power}\right) \times Old\ Interval \right\rfloor \quad (1)$$

The *New Interval* will be applied to the sensor node only if its value is between the *Upper Limit* and *Lower Limit* values which are pre-defined in the sensors.

Using these techniques, we're able to increase the lifetime of the network. There are different ways that we can measure the lifetime of the network; some algorithms define the end of the networks lifetime as the time when the first sensor runs out of power, some other define it as the time when a fraction of sensors run out of power [5]. In our approach we defined the end of the networks lifetime as the time when the first sensor node goes below the Power Threshold, or the time when an agent is not able to reach the base-station. Since having an agent which is not able to reach the base-station means that a critical sensor node in the way to the base-station has run out of power, the second definition is basically equal to the first one.

III. EFFICIENT POWER MANAGEMENT USING NEURAL NETWORKS

Usually a wireless sensor network life-time ends by having a single sensor node which uses all its power while other sensors have a significant amount of remaining power. This sensor (which is the cause of the networks end of lifetime) is most likely located in a very critical hotspot which is in the routing path of many nodes to the base-station. By predicting these hotspot nodes, it is possible to allocate tasks to the nodes in a more efficient way and thus increase the lifetime of the network.

In order to predict WSN's hotspots, we propose a method based on Neural Networks. Using our proposed method, we can answer to this question that "what would the power levels of sensor nodes be at the end of a WSN's lifetime?", and thus we can predict which nodes will consume more power and are hotspots of the WSN. We discuss our method in three sections:

- Hotspots prediction
- Using hotspots in Agent-based WSN route discovery.
- Using hotspots in Agent-based WSN task management.

A. Hotspots prediction using neural networks

In order to predict hotspots in a WSN we defined a set of attributes for each sensor which will be used as the inputs of our 3-layered neural network. These attributes belong to one wireless sensor node and by using them as the inputs of the neural network we can predict the power-level of the sensor at the end of WSN's lifetime. These attributes are as follows:

- Sensor's distance from sink
- Sensor's distance from the closest border
- Sensor's number of neighbors (regional density)
- Sensor's agent accessibility (number of neighbors which initially route their data through this sensor)

After deploying sensor nodes, base-station receives sensor nodes positions and neighbors' information (as discussed in section II), thus it can easily calculate these attributes for each sensor and run the neural network to predict their final power-level. To train the neural network numerous simulations of the neural network should be executed and the final power-level of each node, along with its attributes should be used to train the neural network. A well-trained neural network would be able to receive each sensor's attributes as the inputs and predict its final power, thus if the neural network be executed for each one of wireless sensors at the start of the WSN's lifetime it would be possible to predict the hotspots of the WSN.

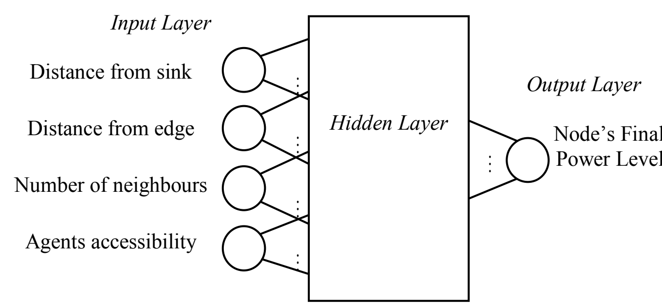

Figure 4. The description of the proposed neural network input and output layers. By entering the inputs related to one sensor node, this neural network (if it is well-trained) would be able to predict the final power-level of that sensor. Running this neural network for all the sensors at the beginning of the WSN's lifetime, it is possible to predict the hotspots of the WSN.

The result of this prediction is dependent of initial power management method which was used in the WSN. As an example if in a WSN management algorithm the power of those nodes which are located at the edge of the sensing field is mostly used, after training the network would be able to understand this behavior of the algorithm and thus it can predict that the final power level of the nodes at the edge of the sensing environment would be the lowest.

B. Using hotspot prediction in agent-based WSN management

Having the predicted power-levels of the sensor nodes, we would be able to use them to define the weight of each connection in the wireless sensor network. The base-station uses these weights and runs the shortest path algorithm (Dijkstra's algorithm). Using these weights, nodes which are in the hot spots will get higher weight values and less routing paths will cross them.

To set the weights, in the first step we normalize and prepare the predicted power-level of each sensor to be used in the shortest path algorithm by using the following formula:

$$Power\ Factor_n = 1 - \frac{Predicted\ Power_n}{Initial\ Power} \quad (2)$$

With this formula we calculate the *power factor* of sensor node number *n* which is a value between 1 and 0. All sensors' powers are initially equal. If the predicted power of a sensor is not different from its initial power then the *power factor* of that sensor would be 0; and if the prediction shows that a sensor node would consume all its power then the *power factor* for that sensor would be 1, thus higher values would be assigned to hot spots and they will get less chance to be chosen in a path.

As we apply the Dijkstra's algorithm to find the shortest paths, we assume that the weight of each edge in the WSN is the power factor of its next node in the path. Those edges that are connected to the base-station would have equal weights of 1.

Using his approach, we would be able to set the efficient paths to the sensor nodes which encounter the nodes' positions and their regional attributes. We also want to mention that our approach can be applied to any WSN management technique which uses the shortest path algorithm.

C. *Using hotspot prediction in agent-based WSN task management*

In this section we describe an approach for task management that can be used in our Agent-Based WSN management technique which is presented in [1]. As mentioned in section II, to distribute agents in the sensing environment, first a set of ideal points will be identified and after that closest nodes to those ideal points will be selected for agents' allocation. To use the hotspot prediction technique in task management we use the same *Power Factor* for each sensor which was introduced in the previous section. As discussed earlier, each sensor which has lower predicted power would have a power factor closer to 1 and those sensors that have higher predicted power would have a power factor closer to 0. We can use this parameter along with the distance from the *ideal points* to create a *selection factor* for each node. We define this *selection factor* as follows:

$$Selection\ Factor_n = \alpha P_n + \beta d_n \quad (3)$$

P_n stands for *Power Factor of node n*, and d_n stands for *the distance of node n from the ideal point*. α and β are *effectiveness factors* which indicate the effectiveness of P_n and d_n in selecting the agent. By choosing the values of α and β, *Selection Factor$_n$* of all the sensor nodes can be calculated for a specific ideal point and the sensor with the lowest value will be selected.

Again, we also need to point out that our Hotspot Prediction approach is a method that can be applied and added to any WSN Management algorithm which uses Shortest Path or Task Management techniques. In the following section we present the simulation results of applying this approach to a sample WSN management algorithm.

IV. SIMULATION

To train our neural network and run our proposed technique we designed a simulator that simulates our agent-based management technique. Our simulator generates random WSNs and calculates all the mentioned characteristics for each sensor, then it continues to operate until the lifetime of the network ends; at this point our simulator will have all the sensors' final power-levels and thus it can use them as the training output of the neural network. Having all the characteristics and final-power levels of each sensor, the simulator trains the neural network with these inputs and outputs.

The simulator basically works on the assumption that the ratio of power usage in a standby node to a fully-active node is 1:200 [12]. Assuming an equal power level for all the sensor nodes the wireless sensor network starts to operate and use the battery of all the sensor nodes. We also assumed a working cycle for all the nodes, meaning that each node is equipped with an internal clock and operates at specific time periods which give the node enough time to route the gathered data thus the WSN works in discreet points of time. All the implementation is done from scratch in Microsoft Visual Studio C#.Net 2005.

In our simulator we use 1000 randomly generated WSNs with 200 sensor nodes. At the setup time of each simulated WSN these 200 sensors are scattered randomly (using uniform distribution) and our previous agent-based approach [1] starts to operate on the WSN. At the end of each WSN's lifetime our simulator runs a training operation on the neural network and trains the neural network using the information from all 200 sensors. The simulator repeats this operation for each one of the 1000 random WSNs.

After training we tested the neural network with some newly generated WSNs and the results showed a promising amount of precision in the predicted results. If we consider the *maximum possible error* in the prediction as "the value when a sensor's real power is its maximum possible power (initial power of the sensor) and the predicted power is 0 (or vice-versa)", then the simulation shows that the average error percentage in our simulation for 100 networks is 12.7%, which means that in average our neural network could predict the final power-level of each node with 12.7% of error. Considering this 12.7% precision factor, the predicted powers seem to be enough precise to be applied to the shortest paths discovery algorithm. We believe by applying more training iterations, the precision of predictions can be increased even more than that. Figure 5 shows the error diagram for one of the random WSNs which is generated by our simulator.

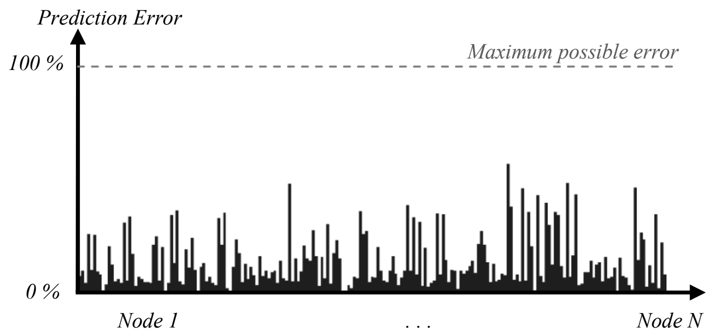

Figure 5. Error diagram of a sample random WSN. The vertical axis represents the error of prediction which is the difference between predicted and the real power values. Each column in the diagram represents the error of one sensor's power prediction. As it can be seen, most of errors are close to zero which means that the neural network could predict the final power of sensors with a high precision.

After training the network we executed our new neural network based algorithm for 1000 simulated WSNs; each WSN is simulated to have 200 randomly scattered sensor nodes. The simulation results showed that in average the lifetime of the network is increased by 3.064%. This value is very much dependent on the neural network precision in predicting the power-levels of the sensor nodes; thus it is possible to increase this average lifetime of the WSN by increasing the training iterations which results in creating a more precise neural network.

We applied different iterations to our neural network and for each one of these iterations, we observed the average lifetime of 400 random networks. Figure 6 shows the result and it can be seen that by increasing the number of iterations and having a more precise neural network the average lifetime of the random WSNs are increasing.

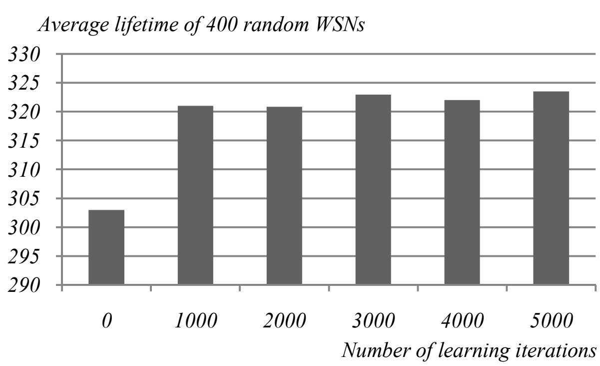

Figure 6. The effect of learning iterations on the performance of the algorithm.

V. CONCLUSION

In this paper we proposed a neural network solution for optimized power management of a wireless sensor network. We discussed that our approach can be added to any other power management algorithm to optimize the task allocation of the sensors, and also it can be used to find more optimized routing paths. We applied our proposed neural network approach to a wireless sensor network management technique which we had proposed in our previous research [1] and we observed that the neural network can predict the final power level of each sensor with approximately 13% of precision. We discussed that by increasing the number iterations in neural network training phase, we can increase the algorithm's precision.

For our future research we intend to study on the inputs of our neural network and modify our set of attributes for each sensor to be able to train the neural network in a more effective way which results in a more precise power prediction. Having a precise prediction, we want to add our approach to different WSN power management techniques and observe the results. We also intend to use neural networks to find the optimized values for the thresholds that we introduced in our agent-based WSN management approach. Defining a set of techniques to apply the predicted power factors to task allocation is another area that we intend to work on.

REFERENCES

[1] Hosseingholizadeh, A.; Abhari, A.; "A new Agent-Based Solution for Wireless Sensor networks Management", 12th Communications and Networking Simulation Symposium (CNS), San Diego, CA, USA, 22-27 March 2009.

[2] Heinzelman, W.; Chandrakasan, A.; and Balakrishnan, H.; "Energy-Efficient Communication Protocol for Wireless Microsensor Networks", Proceedings of the 33rd Hawaii International Conference on System Sciences (HICSS '00), January 2000.

[3] Akyildiz, I.F.; Weilian Su; Sankarasubramaniam, Y.; Cayirci, E., "A survey on sensor networks", Communications Magazine, IEEE, vol.40, no.8, pp. 102-114, Aug 2002.

[4] Chee-Yee Chong; Kumar, S.P., "Sensor networks: evolution, opportunities, and challenges", Proceedings of the IEEE , vol.91, no.8, pp. 1247-1256, Aug. 2003.

[5] Biswas, S.; Morris, R., "Opportunistic Routing in Multi-Hop Wireless Networks", Proceedings of the second workshop on hot topics in networking (Hotnets-II), Massachusetts, Nov. 2003.

[6] Yeling Zhang; Ramkumar, M.; Memon, N., "Information flow based routing algorithms for wireless sensor networks", Global Telecommunications Conference, 2004. GLOBECOM '04. IEEE, vol.2, pp. 742-747, 29 Nov. 2004.

[7] Al-Karaki, J.N.; Kamal, A.E., "Routing techniques in wireless sensor networks: a survey", Wireless Communications, IEEE , vol.11, no.6, pp. 6-28, Dec. 2004.

[8] Tynan, R.; Marsh, D.; O'Kane, D.; O'Hare, G.M.P., "Agents for wireless sensor network power management", International Conference Workshops on Parallel Processing (ICPP 2005), vol., pp. 413-418, June 2005.

[9] Hock Beng Lim; Bang Wang; Cheng Fu; Phull, A.; Di Ma, "A Middleware Services Simulation Platform for Wireless Sensor Networks", 28th International Conference on Distributed Computing Systems (ICDCS 2008) , pp.168-173, 17-20 June 2008.

[10] Li Hu; Yun Li; Qinbin Chen; Jia-yi Liu; Ke-ping Long, "A New Energy-Aware Routing Protocol for Wireless Sensor Networks", International Conference on Wireless Communications, Networking and Mobile Computing (WiCom 2007), pp.2444-2447, 21-25 Sept. 2007.

[11] Fok, C.-L.; Roman, G.-C.; Lu, C., "Mobile agent middleware for sensor networks: an application case study", *Fourth International Symposium on Information Processing in Sensor Network.(IPSN 2005)*, pp. 382-387, April 2005.

[12]Hussain, S.; Shakshuki, E.; Matin, A.W., "Agent-based system architecture for wireless sensor networks", 20th International Conference on Advanced Information Networking and Applications (AINA 2006), vol.2, April 2006.

[13]Jae-Hwan Chang; Tassiulas, L., "Energy conserving routing in wireless ad-hoc networks," *INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, vol.1, pp.22-31, 2000.

[14] Zhou Ying; Xiao Debao, "Mobile agent-based policy management for wireless sensor networks," Proceedings of 2005 International Conference on Wireless Communications, Networking and Mobile Computing, vol.2, pp. 1207-1210, 23-26 Sept. 2005