International Journal of Future Generation Communication and Networking
Vol. 4, No. 1, March 2011

An Energy Conserving Routing Algorithm for Wireless Sensor Networks

Yuping Dong	Hwa Chang	Zhongjian Zou	Sai Tang
Tufts University	*Tufts University*	*Tufts University*	*Tufts University*
yuping.dong@tufts.edu	*hchang@ece.tufts.edu*	*zzou@tufts.edu*	*sai.tang@tufts.edu*

Abstract

Wireless sensor networks (WSN) technologies are widely used in today's world for monitoring purposes. In most applications, the sensors are not plugged in. Instead, they get power from the batteries they carry. To keep the network alive for a long time with such limited power, it is very important to conserve energy while the network is functioning.

In this paper, we present an energy efficient routing algorithm for WSN. In this algorithm, we divide the sensor nodes into several scheduling sets and let them work alternatively. In this way, the sensors do not have to be active all the time which saves a lot of energy. When choosing the next sensor to forward the information to, we consider both the distance from the base station to the sensor and its current energy level. So the network power consumption will be distributed among the sensors. When the network does not have enough sensors that have sufficient energy to run, it generates new scheduling sets automatically. Simulations and comparisons demonstrate that our algorithm outperforms the previous work on energy efficient routing algorithms.

Keywords: *Wireless Sensor Networks, Energy Efficient, Coverage, Connectivity, Network Lifetime.*

1. Introduction

As the technology advances, WSN are used in various areas, like weather monitoring, security surveillance, ambient condition detection, etc. In most applications, the network consists of a vast assembly of tiny sensors. These sensors are able to sense the surrounding conditions and transform them into electronic data which is then routed through other sensor nodes back to the sink node as shown in Figure 1. The sink node maintains a connection to the server where requests and decision are made.

Once the network is deployed, it is expected to last for a long time so as to reduce the initial cost. Due to the small size of the sensors, they do not carry much energy. When distributed randomly in the interested area, it is very hard to locate each sensor correctly. In the case of some sensors using up their energy, it is impossible to find them and recharge or replace their batteries. Thus energy conservation becomes a major problem in WSN.

Recent development of Micro-Electro-Mechanical Systems (MEMS) technology makes it possible to design and maintain energy efficient devices. PicoNode [1] and Smart Dust [2] are two examples of lightweight energy efficient sensor devices. Since the nodes are very small, energy harvesting is very important keep them working. When built in the floor, PicoNode

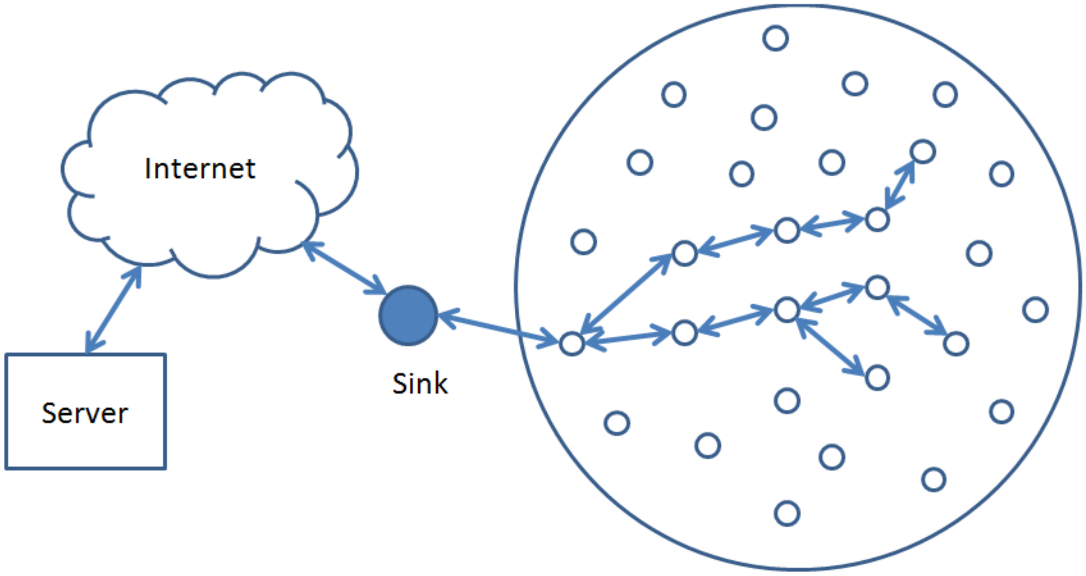


Figure 1. A typical WSN

harvests energy via vibration. Smart Dust is much smaller than PicoNode, which makes low frequency transmission impossible. Passive optical transmission is used by Smart Dust to reduce energy consumption. However, a good routing protocol reduces the number of transmission, spreads power consumption among all sensors which in turn prolongs the network lifetime. There are different definitions of network lifetime [3]. Here we define it as the time period from when the network is deployed until there are not enough sensor nodes that can form a fully functional network.

Coverage and connectivity are the other two fundamental metrics when evaluating the quality and efficiency of WSN. When monitoring, the ideal wireless sensor network covers the entirety of the area, and every bit of detected information is guaranteed successful transmission back to the base station. As for a surveillance system, if and when an intruder is detected within the network, coverage would be expected to increase appropriately so that the track of the intruder is not lost. Meanwhile, higher degree of connectivity ensures that the detected information gets transmitted to the base station correctly.

An existing work on an energy efficient *m*-coverage and *n*-connectivity routing algorithm in [4] insures that each point within the region of interest be covered by at least *m* sensor nodes, with each node having not less than *n* different paths to the sink node. There are flaws in this algorithm, as it allows for the same node to be in as many scheduling sets as possible, dramatically reducing the potential lifetime of the network. The algorithm also lacks the ability to track the energy remaining in each node.

In an attempt to address the issues mentioned above, we propose an energy efficient routing algorithm which introduces multiple sink nodes and divides the sensor nodes into different scheduling sets. The members of each set together satisfy *m*-coverage and *n*-connectivity for the whole monitored region. The network switches between different scheduling sets when collecting data. The algorithm keeps track of each node's energy level, considers the distance to sink nodes and the remaining energy when selecting next hop nodes. In this way, the network power consumption is balanced among all sensor nodes.

The rest of this paper is organized as follows. In Section 2, we analyze the design challenges posed in WSN. In Section 3, current solutions to energy-efficient WSN are carefully studied. Section 4 explains the operation mechanism of the proposed algorithm. Simulation results and discussion are shown in Section 5. Section 6 concludes this paper and points out future works.

2. WSN Design Challenges

Based on the characteristics of WSN, many researchers have pointed out numerous factors that affect the design of the WSN. As suggested in [5] and [6], fault tolerance, scalability, power consumption, coverage and connectivity are some of the design factors of WSN.

Fault Tolerance: During data collection, some paths may fail due to the limited power of sensor nodes or congestion. A reasonable network should be able to find alternative paths with more energy or less traffic instead of dropping the data. As a result, redundancy is inevitable in fault tolerant WSN.

Scalability: In many applications, for example, border surveillance and disaster recovery, there are hundreds and thousands randomly deployed sensor nodes. The designed network must be able to work with such a huge number of sensor nodes.

Power Consumption: The sensor nodes are usually small. When used in border surveillance, they have to be unnoticeable. Due to the tiny size, the energy carried by the sensor node’s battery is very little. Since the sensors are deployed in massive amount, it is very hard to correctly locate each of them. Battery recharge or replacement of the failed sensors is impossible. Sensors collect data, they also route data. Failure of several sensors will affect the network topology significantly. Reducing power consumption is of extreme importance to a long lasting wireless sensor network.

Coverage: Each sensor has a limited sensing range which confines the area each sensor can monitor. Accurate decision depends on complete information of the whole monitored area, so full coverage is another important parameter in WSN.

Connectivity: Once an event is sensed, the information must be guaranteed to be transmitted correctly to the sink node. Hence, each sensor node should be able to connect with the sink node via multi hops.

There are other design challenges for WSN. For example, node deployment can be deterministic or randomized. With deterministic deployment, data are routed through predetermined paths. When deployed randomly, paths are automatically determined by the network. In this paper, we focus on energy conservation, coverage and connectivity.

3. Energy Efficient WSN

Energy is an essential issue in WSN. The great achievements people made in electronics and wireless communication in last couple years made the development of low cost energy efficient WSN possible. There are mainly three different methods available, designing energy efficient devices for sensor nodes, designing energy saving MAC (Media Access Control) layer scheme, and designing energy efficient routing protocols for the Network layer.

As mentioned previously in Section 1, “PicoNode” and “Smart Dust” are two typical examples of energy efficient sensor devices.

Since the sensor nodes are deployed in a high density, and sometimes the data rate is quite low, not all nodes need to be active all the time. It saves huge amount of energy when we schedule the unused nodes to sleep. Wu et al. [7] proposed an energy efficient wake up scheduling for data collection and data aggregation in WSNs. TDMA is used as MAC layer protocol, sensor nodes are scheduled with consecutive time slots in different states. Using this scheme, sensor nodes can stay in sleep mode for longer time, and the number of state transitions is reduced significantly. Researchers developed many different network layer protocols to efficiently route information with low power consumption.

Equipped with tiny antennas, the sensors can only communicate in high frequency which makes data transmission very power consuming. A good routing protocol reduces collision, and thus reduces number of retransmission. A good routing protocol also spreads power consumption among the whole network to maintain longer network lifetime.

There are mainly three types of energy efficient routing protocols, data centric, hierarchical and location-based protocols [8].

Data centric protocols [9] specify data with attribute-based names, data are transmitted as requested. Sensor Protocols for Information via Negotiation (SPIN) uses data negotiation between neighboring nodes. Sensor nodes only transmit data to neighbors that are interested in the data so that redundant data transmissions can be eliminated. Direct diffusion is another data centric protocol in which sensor nodes choose empirically best paths to transmit data. The query from the sink node is transformed to an interest message and diffused to a specified region through the relaying nodes between them. Each node remembers which neighbor is the interest from for different queries. When the corresponding data are sensed at the interested region, the sensor nodes relay the data through the path according to their records of the neighbors for each query.

Hierarchical protocols divide the sensor nodes into many clusters with each cluster having a cluster head. The cluster head gathers information from other nodes inside the cluster and transmits the aggregation to the sink node by relaying through other cluster heads. Clustering eliminates redundant communication between sensor nodes that are in the same cluster. It also greatly reduces the number of relayed packets by only using cluster heads as intermediate nodes [10]. In [11], an energy-efficient integrated-LEACH algorithm for clustered WSN is proposed. In this algorithm, the sensor nodes whose remaining energy is above the threshold level are on while others are off. This algorithm does not take coverage into consideration.

In some applications, the location of each sensor node can be obtained by GPS technology. If the interested region is known at a priori, data queries can be propagated to that particular region. By using this location-based routing protocol, the number of data transmission is reduced significantly. The geographic location-based routing in [12] chooses the ideal minimum energy consumption path to route data to minimize the end to end energy consumption. Each node in the network makes a decision on next hop in the path based on the geographic information of the destination, the neighbor nodes and itself. This algorithm does not track the energy level of each node, therefore, it does not balance energy consumption among all sensor nodes.

The selective flooding-based routing protocol [13] saves energy by selectively flooding routing packets to appropriate nodes. Gateway nodes are used to relay data packets between cluster heads when the distance between two cluster heads is larger than the radio transmission range. However, after the cluster formation phase, certain nodes are designated to be the cluster heads. Throughout the data transmission phase, these nodes keep doing more work than other non-cluster head nodes. As a result, the cluster heads will die very soon, which shortens the network lifetime.

In the energy efficient routing protocol [14], each node selects its next hop node based on the energy level of its neighbors. The node will relay data packets to its sibling node instead of its parent node if the sibling node has more energy than the parent node. This algorithm only considers energy balance of the network. It could not guarantee minimum transmission latency.

Zeng et al. [15] proposed an energy efficient geographic routing protocol which makes routing decision locally by jointly considering multiple factors - the realistic wireless channel condition, packets advancement to the destination, and the energy availability on the node with environmental energy supply. This algorithm saves nodes' energy and guarantees short path transmission. But this decision has to be done by each node for every hop of the data transmission. This creates a large amount of overhead. The data transmission latency is significantly increased.

The EECCR routing algorithm proposed in [4] divides sensor nodes into different scheduling sets. The sensor nodes in each set together make the monitored region m-covered and the network n-connected. The authors claim that this algorithm balances the power consumption among the sensor nodes by allowing the network to switch among different scheduling sets. However, this algorithm did not consider nodes' energy level. Nodes choose the neighbors with minimal hop count as their next hop nodes. So the neighbors with minimal hop count will be used more frequently than others, which causes these neighbors die very soon. When most nodes in a particular area die, that area becomes a dead zone. In this algorithm, nodes also prefer the neighbors that are already chosen by other nodes as next hop. The authors wish to reduce energy consumption of unused nodes in this way. However, it actually increases concentration of data flow in some nodes.

4. Proposed Energy Efficient Routing Algorithm

To solve the problems of the EECCR algorithm discussed above in Section 3, we improve it in the following aspects as in [16] and [17].

(1) We monitor the sensor nodes' energy levels, activating only the nodes with sufficient energy to reduce unnecessary communications between useful nodes and dying nodes.

(2) When selecting next hop nodes, we take both the node hop count and energy level into consideration so that we may distribute power consumption amongst each node's neighbors, creating balance between packet transmission latency and network energy efficiency.

(3) We assign random scheduling set numbers to the nodes with highest hop count instead of assigning to all nodes in the whole region at the beginning. Using this method, each sensor node belongs to fewer scheduling sets. So there are fewer nodes active at anytime, and hence the network power consumption is reduced which in turn prolongs the network lifetime.

(4) Multiple sink nodes are used to further distribute power consumption of nodes and to shorten packet's transmission latency.

4.1. Algorithm Basis

We adopt the method in [4] to calculate the minimal number of nodes needed to satisfy m-coverage and n-connectivity for the monitored region.

Assume there are N sensor nodes randomly distributed in a circular region with radius R. As for security purpose, we only consider homogeneous networks, since large heterogeneous sensor networks cause manufacturing problems. Assume the sensor nodes have the sensing range of r_s and transmission range of r_t. For a large region like the border surveillance area, border effects are not so important. We simplify the calculation of the expected value of m-coverage ratio as below.

$$\begin{aligned} E[COV_m] &= 1-\frac{2}{R^2}\int_0^R\left(\sum_{x=0}^{m-1}\frac{\mu^x}{x!}e^{-\mu}\right)rdr \\ &= 1-\sum_{x=0}^{m-1}\frac{\mu^x}{x!}e^{-\mu} \end{aligned} \tag{1}$$

where $\mu = Nr_s^2/R^2$ is the expected number of nodes that cover a point in the region.

The expected n-connectivity probability provided by N nodes is

$$E[CON_n] \geq [1-Y(\frac{Nr_t^2}{R^2})]^N \tag{2}$$

where $Y(k) = e^{-k}\sum_{x=0}^{n-1}\frac{k^x}{x!}$.

When given the required m-coverage ratio ε and n-connectivity probability η, the minimal number of nodes needed for each condition $N_{COV}(m,\varepsilon)$ and $N_{CON}(n,\eta)$ can be calculated from (1) and (2). So the minimal number of nodes needed for the whole network to be m-coverage and n-connectivity is then calculated as below.

$$N_{ac}(m,\varepsilon,n,\eta) = \max\{N_{COV}(m,\varepsilon), N_{CON}(n,\eta)\} \tag{3}$$

Because of the complexity of equations (1) and (2), it is not possible to calculate the number of nodes needed in each set for specific m-coverage and n-connectivity ratios. But we can derive the ratios for specific number of nodes numerically.

As stated in [18], if the transmission range r_t of the sensor nodes is not smaller than twice the sensing range r_s, full network coverage can guarantee network connectivity. The network n-connectivity ratio changes when the ratio r_t/r_s changes, so we keep r_s constant while changing r_t. We plot the expected network m-coverage ratio and n-connectivity ratio in a circular region with radius 200m. In Figure 2, we give a plot of expected m-coverage ratio with different number of nodes in each set. We also plot the expected n-connectivity ratio of different number of nodes with different transmission ranges in Figure 3 and Figure 4.

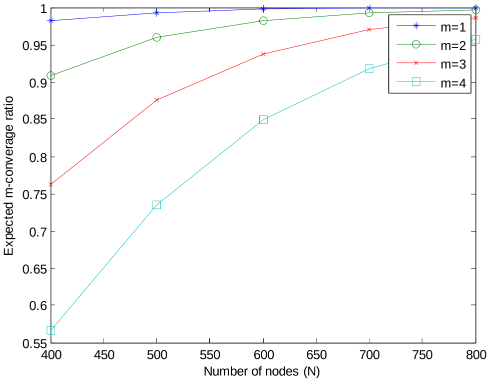


Figure 2. Expected m-coverage ratio vs. number of nodes.

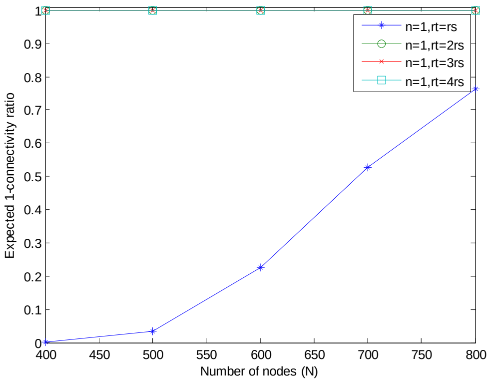


Figure 3. Expected 1-connectivity ratio vs. number of nodes.

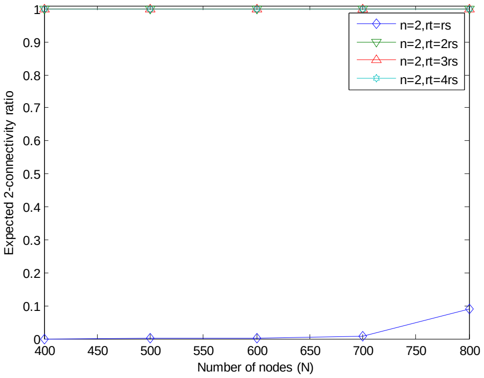


Figure 4. Expected 2-connectivity ratio vs. number of nodes.

Clearly from the above three plots, we can see when there are 400 nodes in each scheduling set, the network's full *1*-coverage ratio is very close to 1. With $r_t / r_s \geq 2$, the network connectivity ratio is very close to 1 as long as there are more than 400 nodes. When $r_t / r_s < 2$, the network connectivity ratios are very low.

4.2. Algorithm Description

Data collection contains the routing setup phase and the data transmission phase. The routing setup phase sets up $s = \left\lfloor \frac{N(t)}{N_{ac}(m,\varepsilon,n,\eta)} \right\rfloor$ scheduling sets, where $N(t)$ is the number of available nodes at time t, $N_{ac}(m,\varepsilon,n,\eta)$ is the minimal number of nodes needed for each scheduling set which is calculated in Equation (3). During the data transmission phase, the scheduling sets are activated periodically and transmit the sensed data to the sink node. Each set checks its member nodes' energy levels right after it finishes one period's sensing and updates each sensor node's routing table.

The routing setup phase is divided into three steps. To distribute sensor nodes' power consumption most efficiently, the sink nodes are evenly distributed in the region of interest.

In the first step, the total number of scheduling sets is calculated from the *m*-coverage ratio ε and *n*-connectivity ratio η.

Step 2 determines the hop count and creates a routing table for each node. We improve this step of EECCR algorithm in [4] with the following modifications.

Each node's routing table contains its neighbors' information including node's ID, hop count and percentage of available energy. The hop count of the sink node is first set to 0 and all other nodes' hop counts are set to infinite. The sink node first broadcasts a hello message with the radio range r_t. When a node *u* receives a hello message from node *v*, it sets node *v* as its neighbor node and records node *v*'s information in its routing table. Here a back-off timer is used to ensure that the nodes do not receive hello messages from higher hop count nodes. When the back-off timer expires, the hop count of node *u* is determined as the minimal hop count of its neighbors increased by 1. After that, node *u* broadcasts a hello message.

Step 3 constructs the *n*-connectivity paths for each node. We modify the method in [4] as follows.

This step starts with the local maximal hop count nodes by assigning each of them a random scheduling set number between 0 and $s-1$ (0 and $s-1$ included). We assign nodes with local maximal hop counts random scheduling set numbers instead of overall maximum to guarantee that we have enough nodes to start with in multiple sink nodes' scenario.

After assigning scheduling set numbers, each of these nodes, say node *u*, calculates a probability *P* of being its next hop node for each neighbor node in its routing table. The *P* value is calculated by the formula below.

$$P = w_h \times \frac{\Delta HC}{CurrHC} + w_e \times EL \tag{4}$$

where *CurrHC* is the hop count of current node, $\Delta HC$is the difference between the hop count of current node and its neighbor node, *EL* is the percentage of remaining energy of the neighbor node, and w_h and w_e are the weights ($0 \le w_h \le 1, 0 \le w_e \le 1$) we give to the two metrics. It then selects *n* neighbor nodes with the maximal value of *P* as its next hop nodes. Then, node *u* sends notifying messages with its ID and scheduling number set $SN_u(t_1)$ (Each node may belong to multiple scheduling sets.) at time t_1 to the selected nodes. If a node *v* receives a notifying message at time t_2, it updates its scheduling number set to $SN_v(t_2) = SN_v(t_1) \bigcup SN_u(t_1)$.

Then node *v* calculates the *P* values for its neighbor nodes and selects *n* neighbors with maximal *P* values and sends the notifying messages to these neighbors. It is possible that a node *u* chooses a neighbor node *v* with the same hop count as its next hop while node *v* also chooses node u as its next hop node using this method. To avoid this infinite loop, we forbid a node *v* from selecting node *u* as its next hop node if it is already selected by node *u* as node *u*'s next hop node.

After these three steps, the network is divided into *s* scheduling sets with both transmission latency and energy consumption taken care of. These s scheduling sets run in turn in the following data transmission phase.

During the data transmission phase, the scheduling sets check the energy levels of their member nodes when they finish working for a period. Each node broadcasts its current energy level with the transmission range r_t, and the nodes who have it as a neighbor update their routing tables. If there is any node whose next hop nodes are all with energy levels below a threshold value EL_{thres}, the corresponding scheduling set is eliminated from the list. Periodically, the network constructs new scheduling sets with all the available nodes unless it cannot make even one scheduling set with all the available nodes. The network lifetime is said to be expired if there are not enough nodes that can form even one scheduling set.

4.3. Energy Model

The sensor nodes in a WSN consume energy during sensing, processing and communication. Among these three, communication consumes most energy of all. So here we ignore sensing power and processing power.

As mentioned in [11], the sensor nodes consume energy by transmitting and receiving data packets. Wang *et al.* did research on WSN devices' power consumption model and proposed a realistic model in [19]. In our protocol, we use an energy model that considers power consumption of nodes running in different states. Energy consumed by state transition is also taken into account.

If E_{elec} represents the energy consumption of transmitting or receiving one bit by the transceiver, ε_{amp} represents the parameter for the transmit amplifier to achieve the required signal-to-noise ratio, and d the transmission distance, the energy consumption for transmitting k bits data is

$$E_{TX}(k,d) = E_{elec}k + \varepsilon_{amp}kd^2 \tag{5}$$

and the energy consumption for receiving k bits data is

$$E_{RX}(k) = E_{elec}k \tag{6}$$

The sensor nodes are in idle listening mode when they are not in the running scheduling set. Sensors consume much less energy in this mode than they are active sensing and transmitting data packets. When a sensor node is eliminated from the network for having an energy level lower than the threshold value, it goes into sleep mode and is considered dead in our simulation.

When switching from one set to another, the nodes may transit from active to idle or sleep, or from idle to active. This state transition consumes energy in the nodes. As in [20], the energy consumed in state transition is modeled as

$$E_{ST} = t_{Trans} \times E_{TargetState} \tag{7}$$

where t_{Trans} is the time it spent to transit from one state to another, $E_{TargetState}$ is the power consumption of the node in the target state.

With all the above mentioned energy consumption, in our algorithm, the energy consumed by each sensor node in one period is modeled by the following equation.

$$E_{tot} = E_{TX}(k,d) + E_{RX}(k) + E_{Idle} + E_{ST} + E_{Sleep} \tag{8}$$

Here E_{Idle}, E_{ST} and E_{Sleep} represent the energy consumption of the node being in the idle mode, state transition mode, and the node being in the sleep mode respectively. We consider $E_{Sleep} = 0$ here though. These are all fixed values in our simulation for each sensor working at each state.

4.4. Example Networks

We illustrate our algorithm here with an example network as shown in Figure 5. We have 22 sensor nodes and a sink node. The sensor nodes are labeled from “A” to “V”. The sink node indicated by “SK” is placed in the center of the network.

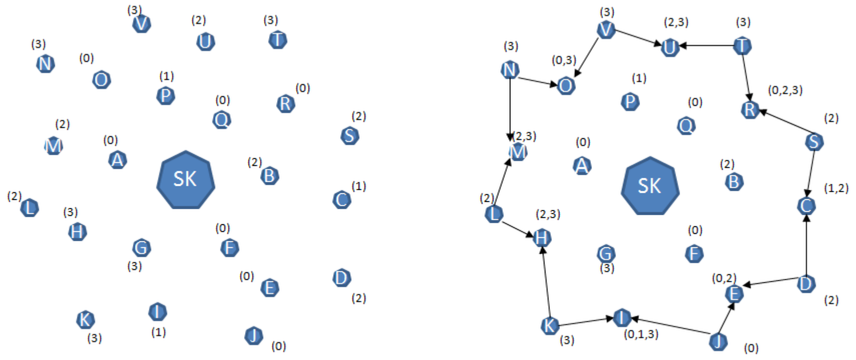
Assume after the first step of routing setup phase, we know the sensor nodes can be divided into 4 scheduling sets.

In the second step, each sensor node determines its own hop count and maintains a routing table with its neighbor nodes’ information. Assume after this step the largest hop count is 3. Then each hop count 3 node assigns itself a scheduling set number between 0 and 3 as shown in Figure 5(a).

At the beginning of the third step, the hop count 3 nodes first start to find their n next hop nodes based on the P values of their neighbor nodes as in Figure 5(b). Assume our application here is to build an *m*-coverage *1*-connectivity network. So each node only seeks for one next hop node. Upon receiving the notifying messages from hop count 3 nodes, the selected nodes update their scheduling number sets. These selected nodes do not necessarily have lower hop count than 3, because of the energy level we take into account when selecting next hop nodes. This process goes on until we reach the sink node. A possible result of the routing setup phase is shown in Figure 5(d).

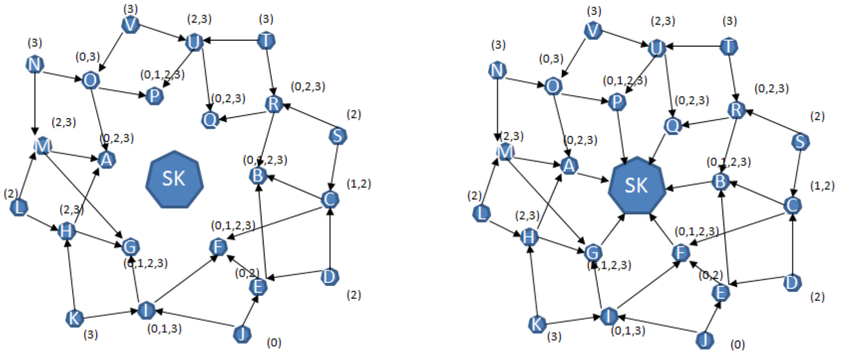
Compare to the EECCR algorithm in [4], our algorithm considers both packet transmission latency and the node’s energy consumption, thus achieves better energy efficiency while maintaining short transmission latency.

Figure 6 shows an example network using multiple sink nodes. It is clearly shown that with multiple sink nodes, the transmission latency is even smaller than in Figure 5.



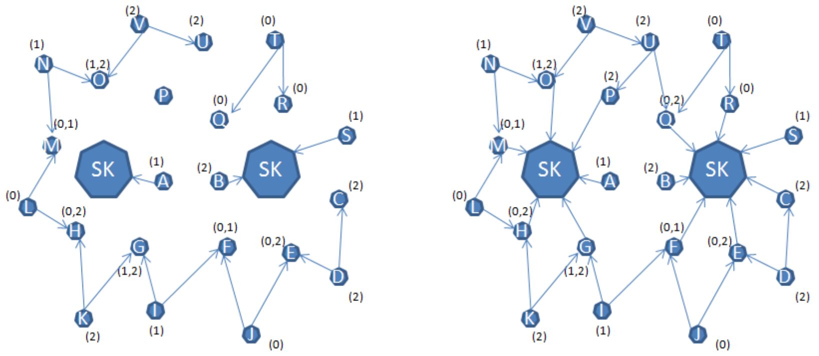
(a)

(b)



(c) (d)

Figure 5. Example network with single sink node.



(a) (b)

Figure 6. Example network with 2 sink nodes.

In the following data transmission phase, the four scheduling sets start to work alternatively.

5. Simulation and Comparison

Simulation is conducted in C# to demonstrate our algorithm and compare it with the EECCR algorithm in [4] and the coverage configuration protocol-CCP algorithm in [21]. Below are the metrics to be considered.

(1) Standard deviation of nodes' usage over hop count. We check the energy level of all nodes periodically and calculate the standard deviation of the energy used among all nodes with the same hop count. We can check the algorithm's efficiency of distributing power consumption among nodes by checking the standard deviation values.

(2) Network lifetime. Here the network lifetime is defined as the time from the sensor network is first established until there are not sufficient available sensor nodes to achieve the *m*-coverage and *n*-connectivity network.

(3) Network *m*-coverage ratio. We loop over the whole region, search in a radius r_s circular area centered at each point to see how many active nodes are in that area for each scheduling set and get the average *m*-coverage ratio among the scheduling sets. We calculate the network *m*-coverage ratio with different number of nodes per scheduling set, and compare the results of our algorithm with the EECCR algorithm, then the CCP algorithm.

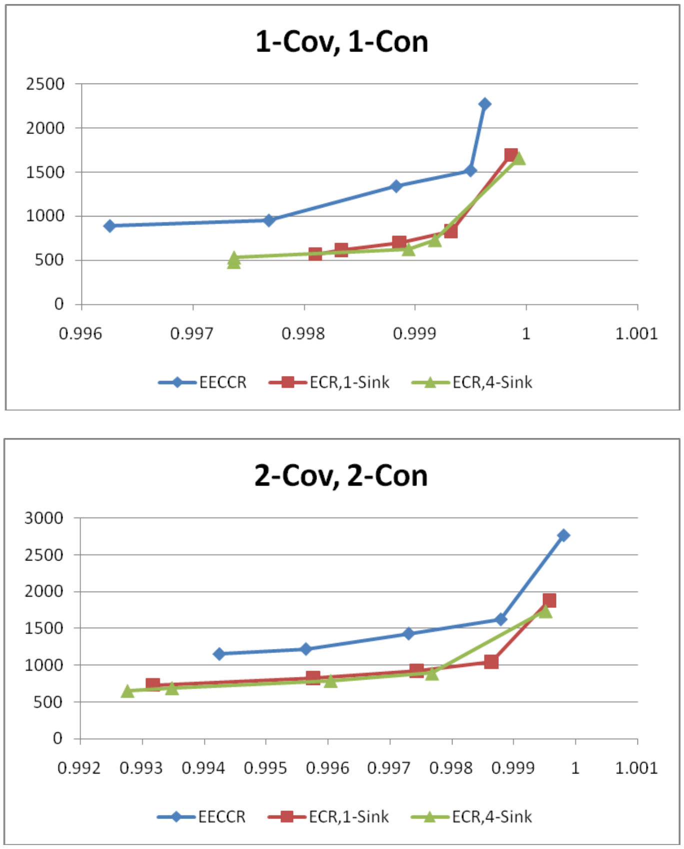


Figure 7(a)(b). Number of Active Nodes vs. Coverage Ratio.

In order to compare with the EECCR algorithm and use the simulation results from [4], the monitored area in our simulation is a circular region with a radius $R=200$m. In a large scale sensor network, homogeneous sensor nodes are usually used. In the simulation, 5000 sensor nodes with a sensing range of 20m and a transmission range of 40m are randomly distributed in the area. We simulated our algorithm with single sink node and 4 sink nodes, and compared the results with EECCR algorithm.

In Figure 7(a) and (b), we plotted the average number of active nodes in the network versus network *m*-coverage ratio of both the EECCR algorithm and the ECR algorithm (Energy Conserving Routing algorithm, which is our algorithm). From the plots we can see that by using our algorithm, there are always fewer nodes that are active in the network than using the EECCR algorithm while keeping the high network *m*-coverage ratio. This is because in our algorithm, the sensor nodes are included in fewer scheduling sets. So there are fewer nodes running at anytime. From the comparison of the EECCR algorithm with the CCP algorithm in [4], the number of active sensor nodes at anytime using the CCP algorithm is always bigger than using the EECCR algorithm. Compared with both algorithms, our algorithm has a big advantage on activating fewer sensor nodes.

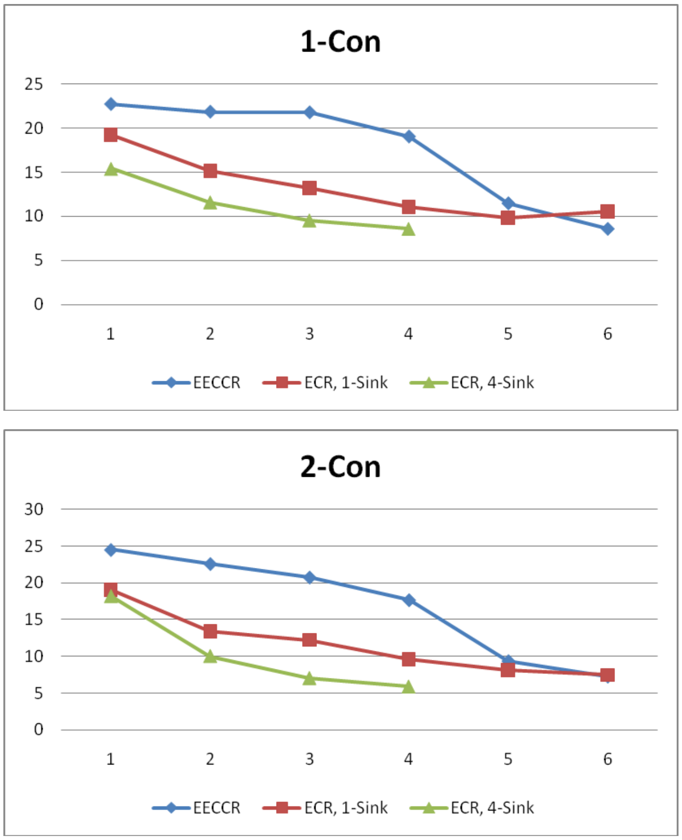


Figure 8(a)(b). Nodes Usage vs. Hop Count

Figure 8(a) and (b) plotted the standard deviation of energy consumption among the nodes with the same hop count. Here we used equal weight for both the energy level and hop count when setting up the *n*-connectivity paths. The standard deviation values of our algorithm are smaller than the EECCR algorithm for most hop counts. This indicates that our algorithm tends to spread the power consumption among the nodes with the same hop count. The ECR algorithm with 4 sink nodes has even smaller standard deviation values, which indicates that with 4 sink nodes, the power consumption is further distributed among all nodes. Note that with more sink nodes, the largest hop count is smaller than with 1 sink node, because it is easier for the sensor nodes to reach the sink nodes.

While if we set $w_e = 1$ and $w_h = 0$, our algorithm is actually a pure energy saving algorithm. In the other hand, the EECCR algorithm only takes hop count into consideration which makes it a greedy algorithm.

To check if our algorithm really prolongs the network lifetime, we did simulations with low initial energy level for all the sensor nodes. With an initial energy level of 120, for simplicity, assume each node's sensing power is 2, packet sending power is 2, and packet receiving power is 1, when using 50% weights, the *2*-connectivity network using our algorithm ran 249 seconds with 1 sink node, 283 seconds with 4 sink nodes. While the *2*-connectivity network using the EECCR algorithm only ran for 174 seconds. That means that our algorithm has more than 40% improvement on energy saving over the EECCR algorithm. Because of the concentrated node usage of the EECCR algorithm, some areas die sooner which causes the shorter network lifetime.

6. Conclusion and Future Work

In large scale wireless sensor networks, the small sensors are randomly distributed in great volume which makes battery recharging or replacement impossible. Energy conservation becomes the only solution to prolonging network lifetime. The EECCR algorithm in [4] divides the whole network to s scheduling sets and lets different sets work alternatively to distribute power consumption among nodes. However, when setting up the scheduling sets, the EECCR algorithm did not take into account nodes' energy level which may cause some nodes deplete very soon. In this paper, we proposed an improved energy aware routing algorithm to distribute data traffic among sensor nodes. When setting up the scheduling sets, we consider both the hop count and the energy level of nodes. Simulation results verified that our algorithm prolongs network lifetime much more than the EECCR algorithm while maintaining better network m-coverage and n-connectivity ratios.

With multiple sink nodes, the network power consumption was further improved. The transmission latency is also shortened because of the smaller distance between each sensor node and the sink node. However, the nodes that are closer to the sink nodes still carry most data traffic. These nodes will be the first nodes that deplete their energy. In order to further distribute power consumption, we could use moving sink nodes to improve energy efficiency.

Acknowledgements

Our thanks to Almir Davis, Nicole Ng, Nick Sullivan, and Clint Epperson for discussing and sharing ideas with us.

References

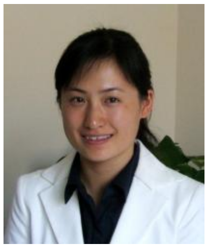
[1] J. M. Rabaey, M. J. Ammer, J. L. da Silva, D. Patel, and S. Roundy, "Pico-Radio Supports Ad Hoc Ultra-Low Power Wireless Networking", IEEE Computer Magazine, 2000, pp. 42-48.

[2] J. M. Kahn, R. H. Katz, and K. S. J. Pister, "Next Century Challenges: Mobile Networking for 'Smart Dust'", Proc. ACM MobiCom'99, Washington, DC, 1999, pp. 271-278.

[3] Y. Chen, Q. Zhao, "On the Lifetime of Wireless Sensor Networks", IEEE Communications Letters, Nov., 2005, vol. 9, no. 11.

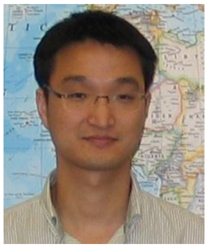
[4] Y.Jin, L. Wang, J. Jo, Y. Kim, M. Yang, and Y. Jiang, "EECCR: An Energy-Efficient m-Coverage and n-Connectivity Routing Algorithm Under Border Effects in Heterogeneous Sensor Networks", IEEE Transactions on Vehicular Technology, March 2009, vol. 58, no. 3.
[5] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A Survey on Sensor Networks", IEEE Communications Magazine, Aug. 2002, vol. 40, no. 8, pp. 102-114.
[6] J. N. Al-Karaki, and A. E. Kamal, "Routing Techniques in Wireless Sensor Networks: A Survey", IEEE Wireless Communication, Dec. 2004, vol. 11, no. 6, pp. 6-28.
[7] Y. Wu, X. Li, Y. Liu, and W. Lou, "Energy-efficient wake-up scheduling for data collection and aggregation", IEEE Transactions on Parallel and Distributed Systems, Feb. 2010, vol. 21, no. 2, pp. 275-287.
[8] K. Akkaya and M. Younis, "A survey on routing protocols for wireless sensor networks," Ad Hoc Networks, May 2005, vol. 3, no. 3, pp. 325-349.
[9] N. Perwaiz, and M. Y. Javed, "A study on distributed diffusion and its variants", Proc. 12th ICCIT International Conference on Computer and Information Technology, Dec. 2009.
[10] H. Lu, J. Li, and G. Wang, "A novel energy efficient routing algorithm for hierarchically clustered wireless sensor networks", International Conference on Frontier of Computer Science and Technology, 2009, pp. 565-570.
[11] D. Wang, H. Tian, and S. Wang, "Energy- efficient routing research for WSN", International Conference on Wireless Communication Networks and Mobile Computing, WiCom, 2007, pp. 2413-2415.
[12] H. Hou, X. Liu, H. Yu, and H. Hu, "GLB-DMECR: Geographic location-based decentralized minimum energy consumption routing in wireless sensor networks", Proc. 6th International Conference on Parallel and Distributed Computing, Applications and Technologies. PDCAT'05.
[13] K. Subbu and X. Li, "SFRP: A selective flooding-based routing protocol for clustered wireless sensor networks", IEEE Radio and Wireless Symposium. RWS'10.
[14] M. M. A. Azim, "MAP: Energy efficient routing protocol for wireless sensor networks", Ubiquitous Information Technologies & Applications, ICUT'09.
[15] K. Zeng, W. Lou, K. Ren, and P. J. Moran, "Energy-efficient geographic routing in environmentally powered wireless sensor networks", IEEE Military Communications Conference, MILCOM'06.
[16] Y. Dong, H. Chang, Z. Zou, and S. Tang, "Energy-balanced m-coverage and n-connectivity routing algorithm for WSN", International Conference on Convergence and Hybrid Information Technology, ICHIT'10.
[17] Y. Dong, H. Chang, Z. Zou, and S. Tang, "Energy aware routing algorithm for WSN applications in border surveillance", Proc. IEEE International Conference on Homeland Security Technologies, 2010.
[18] H. H. Zhang and J. C. Hou, "Maintaining sensing coverage and connectivity in large sensor networks", International Journal on Wireless Ad Hoc Sensor Networks, 2005, vol. 1, no. ½ , pp. 89-124.
[19] Q. Wang, M. Hempstead, and W. Yang, "A realistic power consumption model for wireless sensor network devices", Sensor and Ad Hoc Communications and Networks, 2006.
[20] B. Bougard, F. Catthoor, D.C. Daly, A. Chandrakasan, W. Dehaene, "Energy efficiency of the IEEE 802.15.4 standard in dense wireless microsensor networks: modeling and improvement perspectives", Proc. Design, Automation, and Test in Europe (DATE), March 2005.
[21] X. R. Wang, G. L. Xing, Y. F. Zhang, C. Y. Lu, R. Pless, and C. Gill, "Integrated coverage and connectivity configuration in wireless sensor networks", Proc. 1st International Conference on Embedded Networks, SENSYS'03, pp. 28-39.

Authors



Yuping Dong received her B.S. degree in Electrical Engineering from Zhejiang University, Hangzhou, China, in 2003 and the M.S. degree in System-On-Chip Design from Royal Institute of Technology, Stockholm, Sweden, in 2007. Currently, Ms. Yuping Dong is doing her Ph.D in Computer Engineering at Tufts University, Medford, MA. Her research interests are energy efficient routing protocols for Wireless Sensor Networks (WSNs) and medical applications of WSNs.

Prof. Hwa Chang received his Ph.D degree in Electrical and Computer Engineering from Drexel University in 1987. He is now an associate professor at the Department of Electrical and Computer Engineering, Tufts University. His research interests are Wireless Communications, Parallel Processing, Computer Networking, Distributed Computing, Multimedia Networks, Simulation, Programmable Logic Design, and Engineering Education.



Zhongjian Zou obtained both Bachelor and Master degree from Tufts University. He is an active member of Tufts Wireless Laboratory since 2007. He helped set up and maintained wireless sensor network test bed. He currently works as a software engineer for Delta Search Labs. He also works as a part-time consultant for Nextcea Inc, provides IT supports and software design and maintenance.

Sai (David) Tang is currently (2010) a Tufts undergraduate senior majoring in Computer Science. His past works include iOS/Cocoa app development, game reverse-engineering, and Drupal extension development. His current interests are Android app development, VoiP technologies, and researches related to website scalability and performance (e.g. Cloud Computing).