Maximum Connectivity and Maximum Lifetime Energy-aware Routing for Wireless Sensor Networks

Charles Pandana and K. J. Ray Liu
Electrical and Computer Engineering Department, University of Maryland, College Park.

Abstract— **In this paper, we consider the energy-aware routing algorithm that explicitly takes into account the connectivity of the sensor networks. In typical sensor network deployment, some nodes may be more important than other nodes because the death of these nodes cause the network disintegration. The network disintegration causes early termination of information delivery. To overcome this problem, we propose a routing algorithm called *keep connect* algorithm, that explicitly considers the connectivity of the network while making the routing decision. The algorithm can be used along with the existing routing algorithms. When doing the routing decision, the *keep connect* algorithm embeds the importance of the nodes in the routing cost. The importance of a node is quantified by how severe the remaining network becomes disconnected/disintegrated when that particular node dies. In particular, the importance of a node is characterized by the algebraic connectivity of the remaining graph. Compared to the existing routing algorithms, the proposed method achieves up to two times improvement in terms of the network lifetime and the number of successfully delivered packets in the special grid network. In random network, the proposed algorithm achieves 20% improvement of network lifetime, 30% less energy per packet, and 33% more successfully delivered packets.**

I. Introduction

Advances in low power integrated circuit devices and communication technologies have enable the deployment of low-cost, low power sensors that can be integrated to form a sensor network. This network has vast important applications, i.e.: from battlefield surveillance system to modern highway and industry monitoring system; from the emergency rescue system to early forest fire detection and the sophisticated earthquake early detection system. Having the broad range of applications, the sensor network is becoming an integral part of human lives. Moreover, it has been identified as one of the most important technologies nowadays [1].

There are many important characteristics of sensor networks. First, the sensor nodes are typically deployed in an area with high redundancy and each of the sensor nodes has very limited energy, therefore is prone to failure. In order to be useful, the sensor nodes are required to collaboratively accomplish a special task. Second, the nodes in the sensor networks typically stay in their original deployed positions for the entire of their lifetime. Hence, it is very important to always keep the remaining network connected, since the disintegrated clusters of nodes are useless for information gathering. Furthermore, because of the immobility of the nodes, it may be difficult to reorganize the remaining nodes to create a new connected network. Therefore, the design of the energy saving routing protocol should take the connectivity of the remaining nodes into account.

Due to the above characteristics of the sensor networks, the design of the routing algorithm becomes very different from the typical ad-hoc networks in the following aspect. Instead of minimizing the hop count and delivery delay in the network, the routing algorithm in the sensor networks focuses more on extending the scarce battery lifetime of the nodes. There are many existing literatures focusing on the routing design of sensor networks. The minimum total energy routing (MTE) [4] algorithm selects the route that minimizes the total transmission energy along the route. The min-max battery cost routing (MMBCR) algorithm [5] tries to minimize the battery cost among the maximum battery cost routes. This algorithm avoids the overuse of nodes along the minimum total energy route. However, the MMBCR selects route that is far from the energy efficient route. To overcome the problem, the conditional min-max battery cost routing (CMMBCR) [4] and the max-min zP_{min} algorithm [6] that trade-off between MTE and MMBCR are proposed. In [2], [3], they proposed a heuristic called flow augmentation (FA) algorithm that gradually makes transition from MTE to MMBCR. They show that their algorithm performs better than CMMBCR and zP_{min} algorithms. However, all the existing energy efficient algorithms do not explicitly consider the connectivity of the network in their routing decisions.

Unlike most of the previous works which use the time until the first node in the network dies as the definition of network lifetime. In this paper, we argue that the network lifetime as the time until the network becomes disconnected should be employed. Using this definition as the network lifetime, the remaining network connectivity becomes a very important criterion to be considered in designing the routing algorithm, especially when the information generation is not known *a priori*. Here, the information generation indicates the source and destination pairs during transmission in the network. To be precise, we define the importance of a node as how many disconnected clusters will be resulted if that particular node becomes dead. The larger the number of disconnected clusters, the more important that node is. By considering the nodes' importance in the routing design, the node with higher importance will not be used unless it is necessary. Therefore, the routing algorithm always maintains the connectivity of the remaining network. We employ the notion of algebraic connectivity of a graph from the spectral graph theory to quantify the importance of the node. We propose an algorithm called *keep connect* to solve the posed problem. Our proposed algorithm is very flexible and can be used along with other existing algorithms such as MTE and FA algorithm. Moreover, we show the effectiveness of our proposed method by extensive simulations.

This paper is organized as follows, we give the problem formulation in Section II. Several important facts from spectral graph theory are briefly outlined in Section III. In Section IV, our proposed algorithm is explained. The effectiveness of our method is demonstrated in Section V. Finally, conclusions are drawn in Section VI.

II. Problem Formulation

A wireless sensor network is modelled as an undirected simple graph $G(N, A)$, where N is the set of nodes in the network and A is the set of all links/edges. Here, we assume all links in the network are bidirectional, i.e. node i is able to reach node j implies the vice versa. The link (i,j) implies that node $j \in S_i$ can be directly reached from node i with a certain transmit power level in the predefined dynamic range. We denote S_i as the set of nodes that can be directly reached by node i. We assume that every node has the initial battery energy of E_i for $\forall i \in N$. Every packet transmission consumes energy. The energy expenditure for transmission from node i to j is proportional to d_{ij}^{α}, where d_{ij} is the distance between node i and j, α is between 2 and 4 , and it depends on the transmission environment [9]. In this paper, we assume $\alpha = 2$ as the path loss exponent for free space propagation. When the energy in one node is exhausted, we say that the node is dead for the remaining of the network lifetime.

There are many definitions of the network lifetime, depending on the application in the wireless sensor network. In [2]–[4], the network lifetime is defined as the time until the first node/sensor in the network dies. In contrast, the network lifetime is defined as the time until all nodes die [10]. Blough and Santi [11] define the lifetime of sensor networks as the $\min\{t_1, t_2, t_3\}$, where t_1 is the time it takes for the cardinality of the largest connected components to drop below $c_1 \cdot n(t)$, where $n(t)$ is the number of alive nodes at time t; t_2 is the time it takes for $n(t)$ to drop below $c_2 \cdot n(0)$; and t_3 is the time it takes for the area covered to drop below $c_3 \cdot A$, where A is the area covered by the initial deployment of the sensors. It is well-known that in both the ad-hoc and sensor networks, the network connectivity is very important to ensure the maximal delivery of the collected information [11]. In these applications, the time until the first node/sensor dies may not serve as a good definition of the network lifetime. Since, the death of the first node/sensor does not imply the failure of information delivery, and the network disintegration typically causes severe impact in the information delivery; we argue that it is crucial to consider the network connectivity in designing the energy-aware routing algorithm. Precisely, we employ the network lifetime definition by setting $c_1 = 1$, $c_2 = 0$, and $c_3 = 0$. This corresponds to the time before the network becomes disconnected.

In [2], [3], the problem of maximizing the minimum node lifetime in wireless network is posed as linear programming

$$\begin{array}{ll} \text{Max} & T \\ \text{s.t.} & f_{ij}^{(c)} \le 0,\ \forall i \in N,\ \forall j \in S_i,\ \forall c \in C, \\ & \sum_{j\in S_i} e_{ij} \sum_{c\in C} f_{ij}^{(c)} \le E_i,\ \forall i \in N, \\ & \sum_{j:i\in S_j} f_{ji}^{(c)} + TQ_i^{(c)} = \sum_{j\in S_i} f_{ij}^{(c)}, \\ & \forall i \in N \setminus D^{(c)}, \forall c \in C, \end{array} \tag{1}$$

where $f_{ij}^{(c)}$ is the amount of commodity c information that is transmitted from node i to node j, the commodity $c \in C$ indicates different source nodes $O^{(c)}$ and destination nodes $D^{(c)}$, e_{ij} is the energy required to guarantee successful transmission from node i to node j. $Q_i^{(c)}$ denotes the information-generation rates at the set of source nodes $O^{(c)}$. We note that the solution of the above formulation requires the knowledge of information generation rates from all commodities. This implies that to solve the linear programming problem, one requires the *a priori* knowledge of the future information generation. When the information generation is not known *a priori* and the routing decision should be done on the fly, the best one can do is to keep the network connected. Specifically, we want to design the routing algorithm to maximize the time before the remaining network becomes disconnected. When there is no *a priori* knowledge of future information generation, the proposed routing algorithm always keeps the remaining network connected with the hope that any future information generation can still be delivered.

III. Facts from Spectral Graph Theory

In this section, we briefly summarize some important facts from spectral graph theory. For simplicity and practicality in our problem, we are only interested in a simple graph (graph that does not contain loops and multiple edges between two nodes). Before we describe some useful lemmas, let us first give several definitions.

Definition 1: In a graph G, let d_v denote the degree of vertex v. Let's define matrix $\mathbf{L}$ as follows:

$$L(u,v) = \begin{cases} d_v & \text{if } u = v, \\ -1 & \text{if } u \text{ and } v \text{ are adjacent,} \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

Equivalently, the matrix $\mathbf{L}$ can be expressed as:

$$\mathbf{L} = T - A, \tag{3}$$

where T denotes the diagonal matrix with the (v,v)-th entry having value d_v, and A is the adjacent matrix.

Definition 2 (Normalized Laplacian matrix): A normalized Laplacian matrix $\mathcal{L}$ associated with a graph is defined as [8]:

$$\mathcal{L}(u,v) = \begin{cases} 1 & \text{if } u = v \text{ and } d_v \neq 0 \text{ ,} \\ -\frac{1}{\sqrt{d_u d_v}} & \text{if } u \text{ and } v \text{ are adjacent,} \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

Equivalently,

$$\mathcal{L} = T^{-1/2} L T^{-1/2}, \tag{5}$$

with the convention $T^{-1}(v,v) = 0$ for $d_v = 0$.

Lemma 1: Let's denote $0 = \lambda_0 \le \lambda_1 \le \cdots \le \lambda_{n-1}$ as the eigenvalues of the normalized Laplacian matrix $\mathcal{L}$ of a graph G. If G is connected, then $\lambda_1 > 0$. Moreover, if $\lambda_i = 0$ and $\lambda_{i+1} \neq 0$, then G has exactly i+1 disjoint connected components.

Proof: See [8]. ■

We note that the set of the λ_i's is usually called the *spectrum* of the Laplacian matrix $\mathcal{L}$. The second smallest eigenvalue of the Laplacian matrix is referred to as the algebraic connectivity of the graph. The above lemma indicates that if G is strongly connected (there exists a simple path from any initial node i to the terminal node j, where $i \neq j$) then the eigenvalue 0 of Laplacian matrix $\mathcal{L}$ has multiplicity 1. Moreover, if the eigenvalue 0 of the Laplacian matrix $\mathcal{L}$ has multiplicity n, then there are n connected components. Given the facts from the spectral graph theory, we are ready to explain our proposed solution called *keep connect* algorithm.

TABLE I: Keep Connect Algorithm 1

```
Let $G(N, A)$ be the original graph. Let's define
  $G_{-i}(\{N-i\}, A_{-i})$ as the graph obtained by
  deleting node $i$ and edges connecting node $i$.
1. Initialization:
     Set nodes' weights as: $W(i) = 0, \forall i \in N$
2. For each node $i$:
     2a. Form the Laplacian matrix $\mathbf{L}_{-i}$ of graph
           $G_{-i}(\{N-i\}, A_{-i})$ as (4).
     2b. Find the multiplicity of eigenvalue 0 of
           matrix $\mathbf{L}_{-i}$. Let's denote this value as $n$.
     2c. Set the weight of node as: $W(i) = n$.
   End for
```

TABLE II: MTE-KC1

```
1. Find the minimum total energy path with edge
   cost $e_{i,j} \cdot W(i)$ for $i \in N$, $j \in S(i)$, where $e_{i,j}$
   is the transmission energy from node $i$ to $j$.
   $W(i)$ is the weight of node $i$.
2. If node dies, recompute the remaining nodes'
   weight using keep connect algorithm.
   Recompute the minimum total energy path.
```

IV. Proposed Solution

The proposed solution can be employed along with any existing routing algorithms. The key idea of our proposed method is that when there is no *a priori* knowledge about the information generation, the best we can do is to design the routing algorithm that does it best to keep the remaining nodes connected. This objective is obvious, since the disintegrated network causes severe performance degradation in terms of the amount of delivered information as discussed in II. In this paper, we demonstrate that our proposed algorithm can be used along with the existing minimum total transmission energy routing (MTE) algorithm [4] and the flow augmentation (FA) algorithm [3]. In fact, our proposed algorithm can be used along with all algorithms that make the routing decision based on minimizing the route cost as the total of edges' costs among all possible routes.

The basic building block of our proposed algorithm is listed in Table I. The *keep connect* algorithm finds the weight of node based on how many connected components will result as this particular node dies. The weight of the node can be thought as the importance of the node in the sense that the most important node is the node that results in large number of disconnected components as it dies. Employing the node importance, we modify the MTE [4] algorithm (called MTE-KC) as in Table II. The flow augmentation algorithm [2], [3] can also be modified by adding the connectivity constraint, we called the modified algorithm as FA-KC.

TABLE III: FA(x_1, x_2, x_3)-KC1

```
In every update time:
1. Find the minimum total energy path with edge
   cost $e_{i,j}^{x_1} \cdot \underline{E}_i^{-x_2} \cdot E_i^{x_3} \cdot W(i)$ for $i \in N$, $j \in S(i)$,
   where $e_{i,j}$ is the transmission energy from node
   $i$ to $j$. $E_i$ is initial energy of node $i$, $\underline{E}_i$ is the
   remaining energy of node $i$ and $W(i)$ is the
   weight of node $i$.
2. If node dies, recompute the remaining nodes'
   weight using keep connect algorithm. Recompute
   the minimum total energy path using FA
   algorithm.
```

TABLE IV: Keep Connect Algorithm 2

```
Let $G(N, A)$ be the original graph. Let's define
$G_{-i}(\{N-i\}, A_{-i})$ as the graph obtained by deleting
node $i$ and edges connecting node $i$. Moreover, let
$G_{(-i,-j)}$ be the graph obtained by deleting node $i$ and
then node $j$, where $j \in \{N-i\}$.
  1. For each node $i$:
     a. Form the $G_{(-i,-j)}$ and find the corresponding
        Laplacian matrix. Let $n_{-j}$ be the multiplicity
        of eigenvalue 0 of the associated Laplacian
        matrix corresponding to $G_{(-i,-j)}$.
     b. Set the weight of node $i$ as
        $W(i) = \prod_j^{\{N-i\}} n_{-j}$.
```

We note that the *keep connect* algorithm can also be extended further by looking ahead to the case when there are more than one node die. The KC algorithm shown in Table I only considers the connectivity of the remaining graph after one node dies. Since it is very common that the death of one node causes the other nodes to become more important, we extend the KC algorithm to consider the case when more than one nodes die. The *keep connect 2* as shown in Table IV can be used to capture the connectivity of the remaining graph after two nodes die. Ideally, this lookahead can be applied to the case when more than 2 nodes die. However, due to the complexity of the algorithm, we only consider up to 2 nodes lookahead.

A. Illustrative Example

Now let us give an illustrative example of how the *keep connect* (KC) algorithm can really improve the performance of the existing routing algorithm. Consider the network shown in Figure 1 (a-c), suppose that there are 10 packets from node 1 to node 3 and another 10 packets from node 3 to node 7, respectively in that order. Also, assume that initially all of the nodes have 10 unit energy and one packet transmission requires one unit energy. Here, we also assume that packet reception energy is negligible. Figure 1 (a-c) show the routing results by using the FA algorithm. After the first 10 packets transmission, the remaining energy of each nodes is shown in Figure 1 (b). The FA algorithm does the load balancing between route $1-2-3$ and route $1-4-3$. However, because of this transmission order, only 5 packets transmission in the second 10 packets transmission will be delivered. Hence, the total throughput is 15 packets. Similar to the FA algorithm, the MTE algorithm chooses one of the route with equal probability. In the worst case, the route $1-4-3$ is chosen and the resulting throughput is 10 packets and in the best case the route $1-2-3$ is chosen first, and the resulting throughput is 20 packets. In contrast, Figure 1 (d-f) illustrate the routing results using the MTE-KC algorithm. The weights of each node (calculated using KC1 algorithm in Table I) are shown below the graph in the figure. In the first 10 packets transmission, the proposed algorithm uses mainly route $1-2-3$, since node 4 is more important and the death of node 4 will cause the remaining network becomes disconnected. The resulting throughput is 20 packets. This solution is the same as the optimal throughput obtained by solving (1). In summary, when the traffic generation is not known *a priori*, it is better to keep the remaining nodes in the network connected.

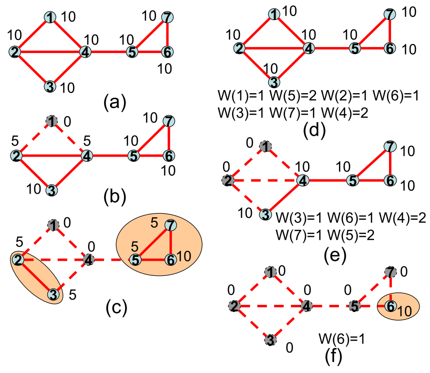

Fig. 1: Illustration

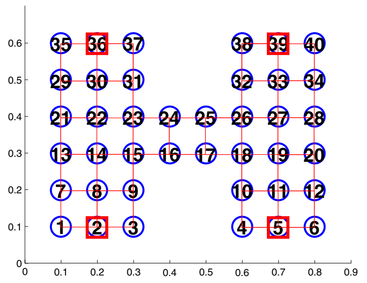

Fig. 2: Grid network

V. Simulation Results

We consider 2 networks in an 1×1 area, shown in Figure 2 and Figure 5. The first one is referred to as grid network. The network consists of 40 nodes. We consider the deterministic source-destination pairs, i.e.: $(S, D) = \{(2, 36)$, $(36, 2)$, $(39, 5)$, $(5, 39)$, $(5, 36)$, $(39, 2)\}$ in this order occurs periodically. We assume the packet arrival follows poisson process. All the arrival packets will be queued in the node and the queued packets will be routed according to the selected routing decision. We consider 4 routing decisions, namely MTE, MTE-KC2, FA$(1, 5, 5)$, FA$(1, 5, 5)$-KC2. We employ the number of packets that are successfully delivered and the time before the network becomes disconnected as our performance metrics. Initially, all the nodes have 120 unit of energy, except nodes 2, 5, 36, and 39, which have infinite energy. The transmission energy follows $P_T = \frac{SNR_{Rx} \cdot d^2}{c}$, SNR_{Rx} is the minimum received SNR to guarantee successful reception, d is the distance between the transmitting node and receiving node, c is the transmission constant. The value of P_T depends on the distance between the communicating nodes and ranges from 0 to 1 unit of energy. The FA algorithm recalculates the all source-destination pairs best route every 20 time instances, based on the residual energy in the nodes at that time. The FA algorithm also recalculates the routes whenever some node dies. In contrast, MTE and MTE-KC2 recalculates the routes only when some node dies. Hence, the MTE-KC2 has much less computational complexity compared to the FA algorithm.

Figure 3 and 4 show the normalized total delivered packets and the normalized network lifetime before the remaining network becomes disconnected with respect to the FA$(1, 5, 5)$ for various mean packet arrival (traffic load). In Figure 3, we see

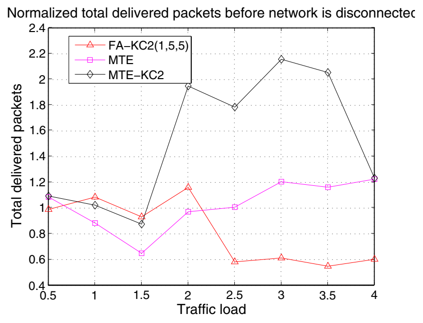

Fig. 3: Normalized total delivered packets before network is disconnected in the grid network

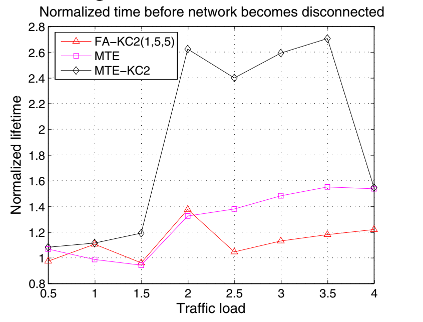

Fig. 4: Normalized lifetime before network is disconnected in the grid network

that when the traffic load is low (< 1.5 packet per simulation time), the FA$(1, 5, 5)$-KC2 algorithm and MTE-KC2 algorithm achieve comparable total successful delivered packets compared to FA$(1, 5, 5)$ algorithm. However, the FA$(1, 5, 5)$ achieves higher throughput compared to MTE algorithm. In the high traffic load, the proposed MTE-KC2 algorithm can achieve up to 2.2 times more packets delivery before the network becomes disconnected. The intuition behind this phenomenon is that when the traffic load is low, distributing evenly the traffic load to the possible routes may not result in overuse some important nodes (nodes which upon their deaths cause disintegrated network). This can be understood considering our example in Figure 1 (a-f), where the traffic consists of the interleaving of 1 packet from node 1 to node 3 and 1 packet from node 3 to node 7, until 20 packets. In this situation, the KC algorithm results in comparable throughput and lifetime as the FA algorithm. However, when the traffic load is high, distributing evenly the traffic load to possible routes wastes the energy of some important nodes, which because of their location and connectivity condition should be used to deliver some particular flow. As the consequence, there is less energy left to deliver that particular flow, when it arrives. Figure 4 shows the normalized lifetime of the network with respect to FA$(1, 5, 5)$ algorithm. Similarly, the MTE-KC2 achieves more than 2.7 times the lifetime of FA$(1, 5, 5)$ algorithm when the traffic load is high. In fact, the MTE-KC2 algorithm is always better than the FA$(1, 5, 5)$ algorithm in term of lifetime. This is due to the fact that MTE is the most energy efficient scheme, however because of not considering the remaining energy in the nodes, MTE will cause unbalanced traffic. This will speed up the death of the first node in the network. We note that due to the definition of network lifetime, the death of the first node may not severely affect the performance of the information delivery. Hence, when MTE is used with our proposed KC2 algorithm, it results in the most energy efficient scheme while keeping the remaining nodes connected.

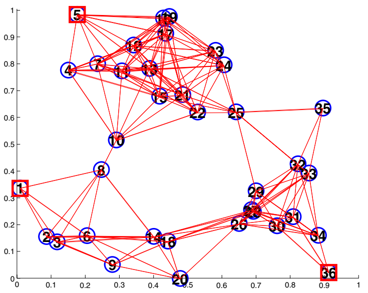

Fig. 5: Network 1: nodes 1, 5 and 36 are the proxies

In our second simulation, we consider a randomly deployed sensor network with configuration as shown in Figure 5. The network consists of 36 nodes. Node 1, 5 and 36 are the proxies that collect the information from other nodes. This implies that the information is randomly generated from the alive nodes to the proxy nodes. The information generation follows the poisson process. We assume that all of the nodes initially have 120 unit of energy. Other simulation parameters are exactly the same as previous one. Figure 6 shows the network lifetime, average routing time, average transmit energy per packet, and total delivered packet before network becomes disconnected for different traffic loads. We can see that the MTE-KC2 algorithm achieves more network lifetime compared to the FA algorithm. The MTE-KC2 algorithm achieves up to 20% more network lifetime for different traffic load. The MTE algorithm itself achieves less network lifetime compared to FA algorithm. In terms of the average routing time per packet, all the algorithms perform comparably. However, the FA algorithm consumes more average energy per packet compared to MTE algorithm, this is obvious since the MTE algorithm is the most energy efficient one. The MTE-KC2 algorithm consumes around 30% less energy per packet compared to the FA algorithm. Finally, the MTE-KC2 algorithm can deliver the most packets before the network becomes disconnected. In particular, the MTE-KC2 delivers 33% more packets compared to the FA algorithm. It is obvious that the MTE-KC2 is more energy efficient, it can deliver more packets before the network becomes disconnected, and it also extend the network lifetime (the time before the network becomes disconnected).

VI. Conclusions

We argue that the time before the first node in the network dies as the definition of network life may be too restrictive. Since, the death of the first node in the network may not cause severe degradation of information delivery in the network. We employ the time until the network becomes disconnected as the network lifetime. Based on this definition, we propose *keep connect* routing algorithm that explicitly considers the connectivity of the network. The proposed algorithm achieves on average up to 20% longer lifetime, 33% more packets

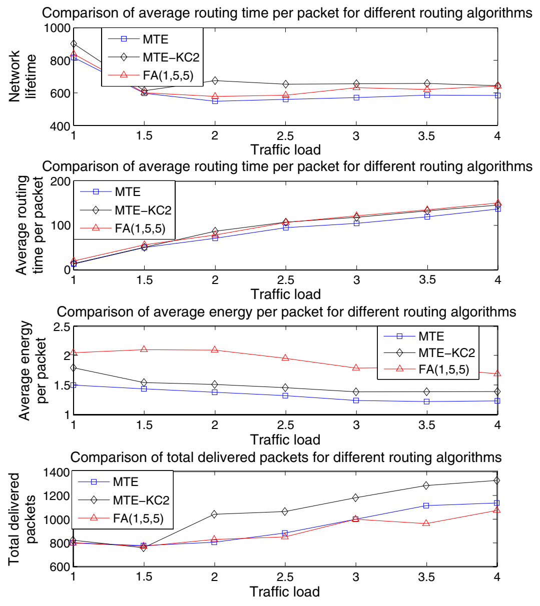

Fig. 6: Comparison of Average routing time per packet, average energy per packet and total delivered packets before network is disconnected in random network

delivery, and 30% less energy per packet compared to the FA algorithm in random network for different traffic load. In special grid network, the MTE-KC2 achieves more than twice longer lifetime and packets delivery compared to the FA algorithm. This is due to the fact that the proposed algorithm finds the most energy efficient route while keeping the remaining network connected. The algorithm uses the most energy efficient route while avoiding the overuse of the important nodes at every time instant in the network.

References

[1] “10 emergying technologies that will change the world”, *Technology Review*, Feb. 2003.
[2] J.-H. Chang and L. Tassiulas, “Energy conserving routing in wireless ad hoc networks”, in *Proc. IEEE INFOCOM*, Tel Aviv, Israel, Mar. 2000, pp. 22-31.
[3] J.-H. Chang and L. Tassiulas, “Maximum lifetime routing in wireless sensor networks”, in *IEEE/ACM Trans. on Networking*, vol. 12, no. 4, Aug. 2004, pp. 609-619.
[4] C.-K. Toh, “Maximum battery life routing to support ubiquitous mobile computing in wireless ad hoc networks”, in *IEEE Communications Magazine*, June 2001, pp. 138-147.
[5] S. Singh, M. Woo and C. S. Raghavendra, “Power-aware routing in mobile ad hoc networks”, in *Proc. 4th Annual IEEE/ACM Int. Conf. Mobile Computing and Networking*, Dallas, TX, Oct. 1998, pp. 181-190.
[6] Q. Li, J. Aslam and D. Rus, “Online power-aware routing in wireless ad hoc networks”, in *MobiCom 2001*, Rome, Italy, July 2001.
[7] Yeling Zhang, Mahalingam Ramkumar and Nasir Memon, “Information flow based routing algorithms for wireless sensor networks”, in *Proc. IEEE GlobeCom 2004*, Dallas, TX, Dec. 2004.
[8] Fan R. K. Chung, *Spectral Graph Theory* (CBMS Regional Conference Series in Mathematics, No. 92), American Mathematical Society 1997.
[9] T. S. Rappaport, *Wireless Communications: Principle and Practice* Prentice Hall, July 1999.
[10] A. Chandrakasan, W. Heinzelman and H. Balakrishnan, “Energy-Efficient communication protocol for wireless microsensor networks”, in *Hawaii ICSS*, 2000.
[11] Douglas M. Blough and P. Santi, “Investigating upper bounds on network lifetime extension for cell-based energy conservation techniques in stationary ad hoc networks”, in *Proc. of ACM Mobicom 2002*.