Improving Connectivity via Relays Deployment in Wireless Sensor Networks

Ahmed S. Ibrahim, Karim G. Seddik, and K. J. Ray Liu

Department of Electrical and Computer Engineering, and Institute for Systems Research
University of Maryland, College Park, MD 20742, USA.
{asalah, kseddik, kjrliu}@umd.edu

***Abstract*— Enhancing the connectivity of wireless sensor networks is necessary to avoid the occurrence of coverage gaps. In this paper, we aim at improving the network connectivity of a given network by adding a set of relays to it. We characterize the network connectivity by the Fiedler value, which is the second smallest eigenvalue of the Laplacian matrix representing the network graph. We propose a network-maintenance algorithm, which finds the best locations for a given set of relays. The proposed algorithm obtains the best relays' locations through a multi-level approach. In each level, the search problem can be formulated as a standard semi-definite programming (SDP) optimization problem. We show that the proposed algorithm can increase the average Fiedler value by 35% by adding one relay only.**

I. Introduction

Recently, there have been much interest in wireless sensor networks due to its various application areas such as battlefield surveillance systems and industry monitoring systems [1]. A sensor network consists of a large number of sensor nodes, which are deployed in a particular area to measure certain phenomenon such as temperature and pressure. These sensors send their measured data to a central processing unit, which collects the data and develops a decision. In general, each sensor can transmit its data and relay other sensors' data to the sensors that are in its transmission range.

The network is considered connected if there is a path, possibly a multi-hop path, from each sensor to the central processing unit. Consequently, the network is connected if there is a path between each two sensors in the network. After deploying the sensors for sometime, some sensors lose their available energy, which affects its ability to send its own data as well as relay the other sensors' data. This situation results in having a *coverage gap* in the network area, i.e., some locations will not be successfully sensed. Furthermore, it affects the network connectivity and may result in the network being disconnected. In this work, we are concerned about keeping the network away from being disconnected, which is known in the literature as *network maintenance* [2].

In the literature, there have been some works that considered the connectivity in wireless networks. In [2], the authors considered deploying as few additional nodes as possible to reconnect a disconnected network. In [3], the authors have considered the problem of maximizing a particular utility function by deploying a certain number of relay nodes. In [4], the authors proposed a mathematical approach to positioning and flying an unmanned air vehicle (UAV) over a wireless ad hoc network in order to optimize the network's connectivity for better Quality of Service (QoS) and coverage.

In this paper, the main goal is to fully enhance the connectivity of the wireless sensor networks by adding an available set of relays to the network. We quantify the network connectivity using the Fiedler value [5], [6], [7], [8], which will be defined later as the second smallest eigenvalue of the Laplacian matrix representing the network graph. Hence, we aim at finding the optimum locations for a given set of relays in order to maximize the Fiedler value of the resulting graph. Finding the optimum locations for such relays is a difficult problem due to the continuous nature of this problem, which results in an infinite number of possible solutions. In this paper we overcome this problem and propose a network-maintenance algorithm, which specifies the near-optimum locations for an available number of relays $K \geq 1$ to maximize the Fiedler value of the network.

Our proposed network-maintenance algorithm can be explained as follows. First, we divide the network area into a certain number of equal regions and represent each region by a relay in its center. Second, we choose the best K relays' locations by solving a *semi-definite programming* (SDP) optimization problem. Third, we iteratively refine our solution by dividing each obtained relay's region into a number of smaller regions and repeating the same procedure. Thus, our algorithm consists of a number of stages, which are called *levels*. Finally, we choose the best location after a few number of levels, where each relay can be deployed.

The major *contribution* of this paper is to propose an algorithm, which formulates the network-maintenance problem as a semi-definite programming optimization problem. This formulation significantly simplifies the problem and allows us to utilize one of the available standard SDP solvers. By doing so, we show that we can increase the Fiedler value by 35% after adding one relay only. This result is very close to what we get by exhaustively searching for the optimum location. However, our proposed algorithm requires only $1/20$ of the time taken by the exhaustive search scheme. Finally, we point out that our proposed network-maintenance algorithm can be implemented through utilizing low-altitude UAVs. More precisely, we can utilize one UAV or more, which can fly along the obtained relays' locations to improve the connectivity of the ground network.

The rest of the paper is organized as follows. In the next section, we describe the network model and review some of the definitions related to the algebraic connectivity of a

graph. In Section III, we formulate the network-maintenance problem and describe our proposed algorithm to solve it. We discuss the formulation of our optimization problem as a SDP optimization problem in Section IV. In Section V, we present some simulation results that show the significance of our propose network-maintenance algorithm. Finally, Section VI concludes the paper.

II. Network Model

In this section, we describe the wireless sensor network model. In addition, we review some related concepts related to the algebraic connectivity of a graph [6], [9]. A wireless sensor network can be modeled as an undirected simple finite graph $G(V, E)$, where $V = v_1, v_2, \cdots, v_n$ is the set of all nodes (sensors) and E is the set of all edges (links). An undirected graph implies that all the links in the network are bidirectional, hence, if node v_i can reach node v_j then the opposite is also true. A simple graph means that there is no self loop in each node and there are no multiple edges connecting two nodes. Finally, a finite graph implies that the cardinality of the sets V and E is finite. Let n and m denote the number of nodes and edges in the graph, respectively, i.e., $|V| = n$ and $|E| = m$, where $|.|$ denotes the cardinality of the given set.

We consider a simple topology model, which is denoted by the *disk model*. In this model, two nodes are connected if the distance between them is less than the transmission range. We assume that all the sensors have fixed transmission power, i.e., fixed transmission radius R. Hence, an edge exists between two nodes if the distance between them is less than R. For an edge l, $1 \leq l \leq m$, connecting nodes v_i and v_j, $\{v_i, v_j\} \in V$, define the edge vector $a_l \in \mathbf{R}^n$, where $a_{l,i} = 1$, $a_{l,j} = -1$, and the rest is zero. The *incidence* matrix $A \in \mathbf{R}^{n \times m}$ of the graph G is the matrix with with l-th column given by a_l. We note that the graph G is connected if there exists a series of edges (path) between each two nodes.

The $n \times n$ *Laplacian* matrix L is defined as

$$L = A\, A^T = \sum_{l=1}^{m} a_l\, a_l^T \ . \tag{1}$$

The diagonal entry $L_{i,i}$ is the degree of node i, and $L_{i,j} = -1$ if $(v_i, v_j) \in E$, otherwise $L_{i,j} = 0$. We note that the summation of the elements in each row (column) equals 0. In addition, the Laplacian matrix is positive semi-definite $L \succeq 0$ and its smallest eigenvalue is zero $\lambda_1(L) = 0$. The second smallest eigenvalue of L, $\lambda_2(L)$, is of great importance with respect to algebraic connectivity of the graph G [6], [5]. It is called *Fiedler value* and it measures how connected the graph is because of following reasons. First, $\lambda_2(L) > 0$ if and only if G is connected and the multiplicity of the zero eigenvalue is equal to the number of the connected sub-graphs. Second, $\lambda_2(L)$ is monotone increasing in the edge set, i.e.,

$$\begin{aligned} &\text{if} \quad G_1 = (V, E_1)\, , G_2 = (V, E_2)\, , E_1 \subseteq E_2 \\ &\text{then} \quad \lambda_2(L_1) \leq \lambda_2(L_2)\, , \end{aligned} \tag{2}$$

where L_q denotes the Laplacian matrix of the graph G_q for $q = 1, 2$.

III. Problem Formulation and Proposed Solution

In this section, we formulate the network maintenance problem, then we propose our algorithm to solve it. The connectivity problem can be formulated as follows. Given a base network deployed in a $g \times g$ square area and represented by the graph $G_b = (V_b, E_b)$, as well as a set of K relays, what are the optimum locations for these relays in order to maximize the Fiedler value of the resulting network? Intuitively, adding a relay to the base network may result in connecting two sensors or more, which were not connected together. Because this relay can be within the transmission range of these two sensors, hence it can relay data from one sensor to the other. Therefore, adding a relay may result in adding an edge or more to the original graph.

Let $E_c(K)$ denote the set of edges resulting from adding a candidate set of K relays. Thus, the network maintenance problem can be formulated as

$$\max_{E_c(K)} \quad \lambda_2\Big(L\big(E_b \cup E_c(K)\big)\Big)\ . \tag{3}$$

Since each relay can be deployed anywhere in the network, the location of each relay is considered as a continuous variable, which belongs to the interval $([0, g], [0, g])$. It is hard to solve this problem in its current form due to the infinite number of possible solutions.

In the sequel, we explain our proposed algorithm to solve this problem. First, we divide the $g \times g$ network area into n_c equal square regions, each with width w. Thus, $n_c = (\frac{g}{w})^2$. We represent each region by a relay deployed in its center. Thus, we have a set of n_c candidate relays, hence the subscript c, and we want to choose the optimum K relays among these n_c relays. This optimization problem can be formulated as

$$\begin{aligned} &\max \ \ \lambda_2\big(L(x)\big) \\ &\text{s. t. } \ \mathbf{1}^T x = K,\ x \in \{0,1\}^{n_c}\ , \end{aligned} \tag{4}$$

where

$$L(x) = L_b + \sum_{l=1}^{n_c} x_l\, A_l\, A_l^T\ , \tag{5}$$

and $\mathbf{1} \in \mathbf{R}^{n_c}$ is the all-ones vector.

In (5), A_l is the incidence matrix resulting from adding relay l to the original graph. Assuming that adding relay l results in I_l edges between the original n sensors, then the matrix A_l can be formed as $A_l = [a_l^1, a_l^2, \cdots, a_l^{I_l}]$, where $a_l^z \in \mathbf{R}^n$, $z = 1, 2. \cdots, I_l$, represents an edge between two original sensors. We note that the optimization vector in (5) is the vector $x \in \mathbf{R}^{n_c}$. Each element in x is either 1 or 0, which represents whether this relay should be chosen or not, respectively. The optimization problem (4) is close to the one in [6], with the matrix A_l replaced by one edge vector. In Section IV, we will show that the optimization problem in (4) can be formulated as a standard semi-definite programming (SDP) optimization problem and that it can be solved using any SDP standard solver.

Assuming for the time being that the optimization problem in (4) can be solved efficiently and that we can choose K

Step 1 The first level: Divide the network area into n_c equal square regions. Each region is represented by a relay at its center.
Step 2 Solve the optimization problem in (4) and obtain the best $K < n_c$ relays among the n_c relays defined in *Step 1*.
Step 3 Start a new level: For each solution x_k, $k = 1, 2, \cdots, K$, divide the k-th region into n_c equal square regions and obtain the best area for this relay. This can be solved using (4) by setting $K = 1$.
Step 4 Repeat *Step 3* until there is no improvement in the resulting Fiedler value.

TABLE I

Proposed multi-level network-maintenance algorithm.

locations, denoted by x_k, $k = 1, 2, \cdots, K$, among the n_c available ones. We call this stage of the algorithm by *level*. As indicated earlier, each location x_k, $k = 1, 2, \cdots, K$, represents a square region of width w. Choosing $x_k = 1$ implies that the k-th region is more significant, in terms of the connectivity of the whole network, than other ones that have not been chosen. In order to improve the current solution, we repeat the same procedure by dividing each k-th region into n_c smaller areas and representing each one by a relay at its center. Then, we find the best location in these n_c regions to have the relay deployed there. This problem is the same as the one in (4) by setting $K = 1$. We do the same step for each region k, $1 \leq k \leq K$ obtained in (4). The proposed network-maintenance algorithm applies a finite number of levels until there is no more improvement in the connectivity. In Table I, we summarize the implementation of our proposed network-maintenance algorithm.

IV. Semi-definite Programming (SDP) Formulation

The exhaustive search scheme to solve (4) is done by computing $\lambda_2(L)$ for different $\binom{n_c}{K}$ Laplacian matrices, which requires huge computations for large n_c. Hence, we need an efficient and quick way to solve (4). In this section, we describe how the optimization problem in (4) can be relaxed to SDP optimization problem.

By relaxing the Boolean constraint $x \in \{0, 1\}^{n_c}$ to be a linear constraint $x \in [0, 1]^{n_c}$, we can represent the optimization problem in (4) as

$$\begin{aligned} &\max \ \lambda_2\big(L(x)\big) \\ &\text{s. t. } \ \mathbf{1}^T x = K, 0 \leq x \leq 1 \ . \end{aligned} \tag{6}$$

We note that the optimal value of the relaxation problem in (6) is an upper bound for the optimal value of the original problem (4), as it has a larger feasible region. As mentioned in Section II, $\lambda_1\big(L(x)\big) = 0$ and its corresponding eigenvector is $\mathbf{1} \in \mathbf{R}^n$. Let $y \in \mathbf{R}^n$ be the eigenvector corresponding to $\lambda_2\big(L(x)\big)$. Thus, $\mathbf{1}^T y = 0$ and $||y|| = 1$. Since, $L(x)\, y = \lambda_2\, y$, hence $y^T\, L(x)\, y = \lambda_2\ y^T\, y = \lambda_2$. Therefor, the Fiedler value can be expressed as

$$\lambda_2\big(L(x)\big) = \inf_y \ \{y^T\, L(x)\, y \mid ||y|| = 1,\ \mathbf{1}^T\, y = 0\} \ . \tag{7}$$

In (7), $\lambda_2\big(L(x)\big)$ is the point-wise infimum of a family of linear functions of x. Hence, it is a concave function in x. In addition, the relaxed constraints are linear in x. Therefore, the optimization problem in (6) is a *convex optimization* problem [10]. Furthermore, the convex optimization problem in (6) is equivalent to the following SDP optimization problem [6], [9]

$$\begin{aligned} &\max \ \ s \\ &\text{s. t. } \ s(\mathbf{I} - \frac{1}{n}\mathbf{1}\,\mathbf{1}^T) \preceq L(x),\ \mathbf{1}^T\, x = K,\ 0 \leq x \leq 1 \ , \end{aligned} \tag{8}$$

where $\mathbf{I} \in \mathbf{R}^{n \times n}$ is the identity matrix and $\preceq$ denotes semi-positive definiteness. In the sequel, we show how the two optimization problems in (6) and (8) are equivalent. Let

$$\tilde{L}(x) = L(x) - s(\mathbf{I} - \frac{1}{n}\mathbf{1}\,\mathbf{1}^T) \ , \tag{9}$$

where s is an unknown scalar. Thus, for any $n \times 1$ vector y, where $||y|| = 1$ and $\mathbf{1}^T\, y = 0$, we get

$$\begin{aligned} y^T\, \tilde{L}(x)\, y &= y^T\, L(x)\, y - s(y^T\, \mathbf{I}\, y - \frac{1}{n}(y^T\, \mathbf{1})\, (\mathbf{1}^T\, y)) \\ &= y^T\, L(x)\, y - s \ . \end{aligned} \tag{10}$$

To ensure that $\tilde{L}(x) \succeq 0$, i.e., $s(\mathbf{I} - \frac{1}{n}\mathbf{1}\,\mathbf{1}^T) \preceq L(x)$, then the maximum possible value of s is

$$\begin{aligned} s &= \inf_y \ \{y^T\, L(x)\, y |\ ||y|| = 1,\ 1^T\, y = 0\} \\ &= \lambda_2\big(L(x)\big) \ , \end{aligned} \tag{11}$$

where the second equality holds from (7). Hence, maximizing s in (8) is equivalent to maximizing $\lambda_2\big(L(x)\big)$ in (6), given that all the other constraints are satisfied.

The optimal solution for (8) is obtained numerically using one of the standard SDP solvers such as the SDPA-M software package [11]. Finally, we use a heuristic to obtain a Boolean vector from the SDP optimal solution, which is the solution for the original problem in (4). In this paper, we consider a simple heuristic, which is to set the largest K x_l to 1 and the rest to 0. In the future, we will consider more sophisticated algorithm to obtain the combinatorial solution.

V. Simulation Results

In this section, we present some simulation results to show the performance of our proposed network-maintenance algorithm. In the simulations, we have used the SDPA-M software package [11] to solve the SDP problem in (8) at each level. In order to associate edges with certain relay, first we find all the sensors in the original graph that are within distance R of this relay's location. Second, we construct an edge between each two of these sensors. Obviously, it may happen that two relays result in a common edge. For such case, we choose the relay which results in the maximum number of edges. Intuitively, we need to have as many edges as possible associated with the least number of relays.

In order to illustrate our proposed network-maintenance algorithm, we consider the small network shown in Fig. 1. It consists of a set of $n = 20$ sensors, deployed randomly in a 6×6 area. Each sensor is marked with $+$ and each edge is represented by a solid line connecting two sensors. We assume that an edge exists between each two sensors if the distance between them is less than $R = 2$. We assume that the number

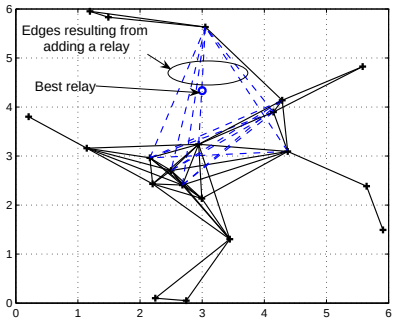

Fig. 1. Sensor network consisting of $n = 20$ nodes in a 6×6 area. Sensors and relay are represented in $+$ and 0, respectively. Original and new edges are represented in solid and dashed lines, respectively.

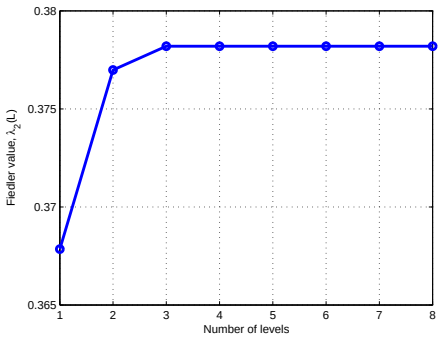

Fig. 2. The Fiedler value versus the number of levels employing $K = 1$ relay for a sensor network consisting of $n = 20$ nodes deployed in a 6×6 area.

of available relays is one, $K = 1$. By calculating the Laplacian matrix of the original network, we find that its Fiedler value is $\lambda_2(0) = \lambda_2(K)|_{K=0} = 0.2359$.

As we explained in Section III, our proposed network-maintenance algorithm finds the optimum location of this relay through a number of levels. In the first level, it divides the network area into $n_c = 9$ square regions, each of width 2, and deploys a relay at the center of each region. We notice that the algorithm picks the region represented by a relay deployed at $(3, 5)$. Deploying a relay at this location $(3, 5)$ results in an increase in the Fiedler value to be $\lambda_2(1) = 0.3679$. In the second level, this small region is divided into $n_c = 9$ smaller regions and the Fiedler value increases slightly to be $\lambda_2(1) = 0.377$. After 4 levels, the algorithm chooses the location $(3, 4.33)$ to be the best location for the relay. Fig. 1 depicts the best location of the relay, represented by 0, as well as the edges resulting from adding this relay shown in dashed lines. For this choice, the resulting final Fiedler value is $\lambda_2(1) = 0.3782$.

Fig. 2 depicts the Fiedler value versus the number of levels for the network shown in Fig. 1. As shown, there is an increase of the Fiedler value in the first few levels until the third one. After that, the Fiedler value is constant and does not increase with more levels. This result signifies the efficiency of our proposed algorithm and its ability to obtain the best relays' locations within a few levels. Based on that result, we set the number of levels to be 3 in the rest of this section.

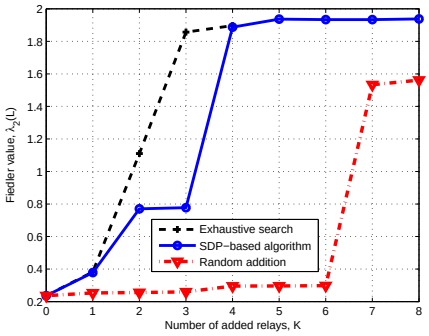

Fig. 3. The Fiedler value versus the number of added relay for a sensor network consisting of $n = 20$ nodes deployed in a 6×6 area.

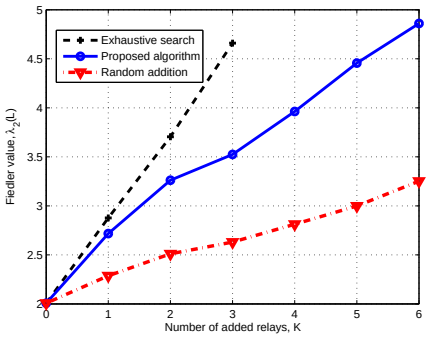

Fig. 4. The average Fiedler value versus the number of added relay for 100 sensor networks, each consists of $n = 100$ nodes and deployed in a 10×10 area.

Next, we explore the effect of increasing the number of possible relays on the Fiedler value. Fig. 3 depicts the Fiedler value as a function of the added number of relays. Obviously, the Fiedler value increases with the number of added relays. We note that there is a large increase in the Fiedler value due to the addition of the first few relays, which is 5 in this example. As we increase the number of added nodes beyond that, we notice that the Fiedler value of the resulting graph is almost constant and does not increase.

We also compare the performance of our proposed algorithm with the exhaustive search and random addition schemes. We have implemented the exhaustive search scheme by dividing the 6×6 network area into many small regions, each region is represented by a relay at its center. The random addition scheme chooses randomly K relays' locations. Fig. 3 depicts the Fiedler value of these two schemes. We have implemented the exhaustive search algorithm up to $K = 4$ nodes due to the huge processing time needed to get the results. As shown, the performance of our proposed algorithm almost coincide with the exhaustive search performance in $K = 1$ and $K = 4$ cases. Finally, we notice that the random addition performs poorly compared to our proposed algorithm.

We also consider a more realistic scenario where we have a large number of sensors, $n = 100$, deployed in a 10×10 area. The transmission radius is $R = 3$. Fig 4 depicts the Fiedler value for various schemes averaged over 100 independent network realizations. In each realization, 100 sensors

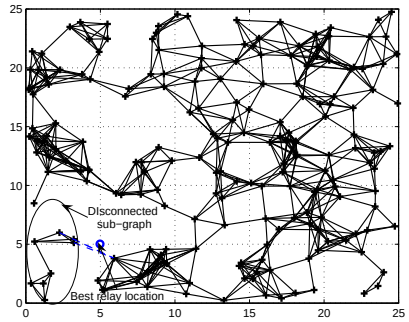

Fig. 5. Sensor network consisting of $n = 200$ nodes in a 25×25 area. The network is reconnected by adding $K = 1$ relay at the location $(4.98, 5.02)$.

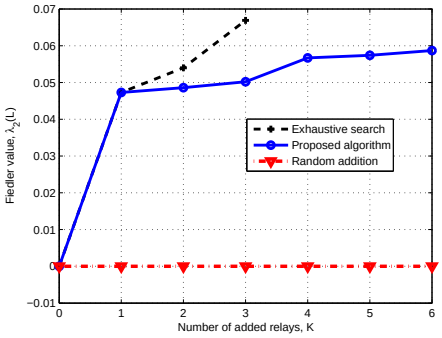

Fig. 6. The Fiedler value versus the number of added relay for a disconnected sensor network consisting of $n = 200$ nodes and deployed in a 25×25 area.

are uniformly deployed in the network area. At $K = 1$, the performance of our proposed algorithm is close to the exhaustive search one. For the $K = 1$ case, our proposed network-maintenance algorithm needs 6.77 seconds to choose the best location, while the exhaustive search scheme needs 138.44 seconds to do the same job. Thus, our proposed algorithm achieves almost the same performance as that of the exhaustive search scheme to allocate a single relay in approximately $1/20$ of the time needed for the exhaustive search scheme. The percentage of the enhancement in the Fiedler value can be calculated as $I(K) = \frac{\lambda_2(K) - \lambda_2(0)}{\lambda_2(0)}\%$. In a sensor network of $n = 100$ sensors deployed in a 10×10 area with transmission radius $R = 3$, the Fiedler value increases by $I(1) = 35\%$ due to the addition of one relay only.

Finally, we consider the case when the original network is disconnected, i.e., $\lambda_2(0) = 0$. Fig. 5 depicts a 25×25 network of $n = 200$ nodes, which is divided into two sub-networks. By following our network-maintenance algorithm using $n_c = 25$ locations, we find out that the network can be connected using one relay only deployed at $(4.98, 5.02)$. By deploying this relay the Fiedler value jumps to $\lambda_2(1) = 0.0473$. Fig. 6 depicts the Fiedler value of this disconnected network versus the number of added relays. The performances of both exhaustive search and random addition schemes are depicted as well in Fig. 6. For the $K = 1$ case, the optimum solution obtained through our algorithm in 11.6 seconds and the exhaustive search solution obtained in 955.6 seconds result in the same Fiedler value. Hence, our proposed algorithm achieves the same performance in time equal to $1/82$ of that needed by the exhaustive search scheme.

We note that for a disconnected network, our proposed algorithm does not guarantee that it can reconnect this network by adding a predetermined set of relays to it. In the future, we will consider the network rebuilding problem, in which we determine the minimum number of relays along with their optimum locations needed in order to reconnect a disconnected network.

VI. Conclusion

In this paper, we have addressed the problem of improving the connectivity in wireless sensor networks. We have considered the Fiedler value as the network connectivity measure. We have proposed a network maintenance algorithm, which finds the optimum locations for an available set of relays that result in the maximum possible Fiedler value. This algorithm finds the near-optimum location through a small number of levels. In each level, the network maintenance problem is formulated as a semi-definite programming (SDP) optimization problem, which can be solved using the available standard SDP solvers. We showed that adding the first few relays has a more significant effect than adding more relays afterwards. In a sensor network of $n = 100$ sensors deployed in a 10×10 area with transmission radius $R = 3$, the Fiedler value is increased by 35% due to the addition of one relay only. Moreover, our proposed algorithm achieves almost the same performance as that of the exhaustive search scheme to allocate a single relay in approximately $1/20$ of the time needed for the exhaustive search scheme.

References

[1] I.F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," IEEE Communications Magazine, vol. 40, pp. 102-114, Aug 2002.

[2] N. Li and J. C. Hou,"Improving connectivity of wireless ad hoc networks," Proc. The Second Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services (MobiQuitous'05), pp. 314-324, Jul. 2005.

[3] H. Koskinen, J. Karvo, and O. Apilo,"On Improving Connectivity of Static Ad-Hoc Networks by Adding Nodes," Proc. The Fourth annual Mediterranean workshop on Ad Hoc Networks (Med-Hoc-Net'05), pp. 169-178, 2005.

[4] Zhu Han, A. Lee Swindlehurst, and K. J. Ray Liu, "Smart deployment/movement of unmanned air vehicle to improve connectivity in MANET ," Proc. IEEE Wireless Communications and Networking Conference(WCNC'06), vol. 1, pp. 252-257, Apr. 2006.

[5] M. Fiedler,"Algebraic connectivity of graphs," Czechoslovak Lathematics Journal, pp. 298-305, 1973.

[6] A. Ghosh and S. Boyd, "Growing well-connected graphs," Proc. IEEE Conference on Decision and Control (CDC), Dec. 2006.

[7] C. Pandana and K. J. Ray Liu, "Maximum connectivity and maximum lifetime energy-aware routing for wireless sensor networks," Proc. IEEE Global Telecommunications Conference (Globecom'05), vol. 2, pp. 1034-1038, Nov. 2005.

[8] B. Mohar, "Some applications of Laplace eigenvalues of graphs," In G. Hahn and G. Sabidussi, editors, Graph Symmetry: Algebraic Methods and Applications, vol. 497 of NATO ASI Series C, pp. 227-275. Kluwer, Dordrecht, 1997.

[9] S. Boyd, "Convex Optimization of Graph Laplacian Eigenvalues," Proc. International Congress of Mathematicians, 3:1311-1319, 2006.

[10] S. Boyd and L. Vandenberghe, "Convex optimization," Cambridge university press, 2006.

[11] SDPA-M package, Available online at: http://grid.r.dendai.ac.jp/sdpa/