The International Journal of Parallel, Emergent and Distributed Systems
Vol. 28, Issue. 4, 2013, 289–307

RESEARCH ARTICLE

Energy-Efficient Communication in Wireless Cable Sensor Networks

Xiao Chen[a*] and Neil C. Rowe[b]

[a]*Dept. of Computer Science, Texas State University, San Marcos, TX 78666*; [b]*Dept. of Computer Science, U. S. Naval Postgraduate School, Monterey, CA 93943*

In this paper, we introduce a new type of sensor: Cable sensor. Unlike traditional point sensors, this type of sensor has a rectangular sensing region with a processor installed on it to do processing and communication. The wireless network formed by this kind of sensor is called wireless cable sensor network (WCSN). We study energy-efficient communication algorithms in WCSNs. We address it in two ways: One is through reducing the total transmission power of processors while maintaining the connectivity of the network and the other is through scheduling cable sensors to let them take turns to go to sleep without affecting the coverage and connectivity of the network. In the first approach, we initially develop a distributed algorithm called DTRNG based on the relative neighborhood graph. Later we enhance it to Algorithm DTCYC. Mathematical proofs show that Algorithm DTCYC provides an optimal solution that can not only minimize the total processor transmission power but maintain the connectivity of the network as well. In the second approach, we propose a cable mode transition algorithm (CMT) which determines the minimum number of active sensors to maintain K-coverage as well as K-connectivity required by the application. We discuss the relationship between coverage and connectivity and prove the theorems that lay the foundation for our algorithm. Simulation results demonstrate that our algorithm is efficient in saving energy.

Keywords: cable sensor, communication, energy-efficient, minimum spanning tree, relative neighborhood graph, transmission range

1. Introduction

Nowadays wireless sensor networks (WSNs) have attracted a great deal of study due to the low cost of sensors and their wide-range applications. WSNs provide a new class of computer systems and expand human ability to remotely interact with the physical world. Applications of WSNs include surveillance and target tracking systems in military [1], environment monitoring [7], home health care [13], intelligent home [5], disaster rescuing [8], and self-touring systems [9].

In most sensor network applications so far, different types of sensors including temperature, humidity, magnetic, seismic, acoustic, infrared, etc. sensors have been used. However, no matter how different the sensors are, their sensing model can be represented by a node with a disc sensing region. Therefore they are called *point sensors*. Various topics related to point sensors have been extensively studied.

Recently a new kind of sensor called *cable sensor* has become available for detecting seismic signals [11]. Current technology can detect movement within a distance of 20 meters orthogonal to the cable sensor. The cable sensor can be tens of kilometers long so one cable sensor can provide extensive coverage. Different from point sensors, this type of sensor has a rectangular sensing region. The communication

---

[*] Corresponding author. Email: xc10@txstate.edu

ISSN: 1744-5760 print/ISSN 1744-5779 online

DOI: 10.1080/17445760.2012.664819
http://www.informaworld.com

between cable sensors in the network is done by the processors on them. Each cable sensor has a processor installed on it to collect and process data sensed by the cable and then communicate with other processors on other cable sensors. If the processor is treated as a node, its communication model is the same as that of the point sensor's. The wireless network formed by this type of sensors is called *Wireless Cable Sensor Network* (WCSN).

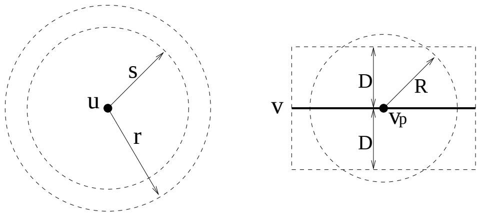
In terms of applications, comparing with the traditional point sensors, this kind of sensor has a long length of coverage area which makes it ideal for borders, bridges and roads. They will be helpful in detecting excavation behavior such as digging tunnels and planting explosive devices on the roads as well as providing a secure border with 24/7 surveillance for illegal immigrants.

In this paper, we discuss energy-efficient communication algorithms in WCSNs. Energy efficiency is very important in WSNs because of the limited power in cable sensors and the inconvenience to recharge their batteries frequently. We will study two ways to minimize communication energy in WCSNs: One is through reducing the total transmission power of the processors but still preserving the connectivity of the network. We initially develop a distributed algorithm called $DTRNG$ (Determine the Transmission power using RNG) based on RNG (Relative Neighborhood Graph). After realizing its deficiency, we enhance it to Algorithm $DTCYC$ (Determine the Transmission power by removing the largest edge in CYCles). Mathematical proofs show that DTCYC provides an optimal solution to our problem and results in an MST (Minimum Spanning Tree), which can not only minimize the total processor transmission power but maintain the connectivity of the network as well. The other way to save energy is through scheduling cable sensors to let them take turns to go to sleep without affecting the K-coverage and K-connectivity of the network required by the application. We develop a distributed algorithm called CMT: A Cable Mode Transition algorithm to let each cable sensor decide whether it should stay active or not based on its neighborhood information. Simulation results demonstrate that our algorithm is efficient in saving energy in WCSNs.

The rest of the paper is organized as follows: Section 2 gives the preliminaries; Section 3 references the related work; Section 4 proposes energy-efficient communication algorithms by reducing the total transmission power of processors; Section 5 puts forward energy-efficient communication algorithms by scheduling cable sensors; Section 6 shows the simulation results; and Section 7 concludes the paper and points out the future work.

2. Preliminaries

In this paper, the cable sensors we use are different from traditional point sensors. They are different in both sensing and communication. The sensing region of a point sensor u can be modeled as a disc with a sensing range s and its communication region can be modeled as a disc with a transmission range r as shown in Fig. 1(a). For a cable sensor v with length L as shown in Fig. 1(b), its sensing region $Rect(v)$ is formed by a rectangle on each side of the cable sensor. The length of the rectangle is L and the width is the sensing range D which is the maximum distance that can be sensed orthogonal to the cable sensor. The communication between cable sensors is done by the processor on each cable sensor. The processor collects data and communicates with other processors on other cable sensors. The processor v_p on cable sensor v can be represented by a black dot and has a communication region of a disc with a transmission range R.



(a) A traditional point sensor (b) A cable sensor

Figure 1. Traditional point sensor vs. cable sensor

3. Related work

In this section, we go over the works done in the wireless point sensor networks regarding energy saving communication algorithms.

3.1 Saving communication energy by reducing transmission power

In the literature, several algorithms have been proposed to reduce communication energy consumption by adjusting sensor transmission power. It is used by [3, 15] to minimize energy in broadcast communication and by [6, 10] to do topology control.

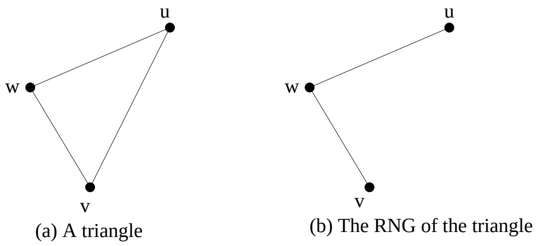
Wieselthier et al. define in [15] a topology control algorithm using nodes' adjustable transmission power based on MST. It is designed in a global manner, meaning that each node needs global network information. The authors also propose in [15] two other globalized algorithms: BLU and BIP to minimize broadcast energy consumption by adjusting nodes' transmission power. The BLU (Broadcast Least-Unicast-cost) applies the Dijkstra's algorithm and the BIP (Broadcast Incremental Power) is a modified version of the Prim's algorithm. Adjustable transmission power is also used by Cartigny et al. in [3] to build RNG locally to solve the minimum-energy broadcast problem.

Several localized solutions exist based on local spanning subgraphs using a node's adjustable transmission power to manage network topology, such as in MST [6] and SPT [10]. In [6], the network topology is constructed by each node building its local MST independently (with the use of information locally collected) and only keeping one-hop on-tree nodes as neighbors. It proves that the topology resulted preserves the network connectivity. Rodoplu et al. [10] introduce the notion of relay region and enclosure for the purpose of power control. It is shown that the network is strongly connected if every node maintains links with the nodes in its enclosure and the resulting topology is a minimum power topology.

Different from these works, in this paper, we will design distributed algorithms to reduce communication energy consumption in WCSNs by minimizing the total transmission power of cable sensors while keeping the connectivity of the network. We start from designing an algorithm using the RNG graph and then extend it to provide an optimal solution to our problem.

3.2 Saving communication energy by scheduling sensors

Another method to save communication power is to schedule cable sensors to let them take turns to go to sleep. The goal is to determine the minimum number of active cable sensors to maintain coverage as well as connectivity. In terms of the relationship between coverage and connectivity, an important result is proved by Zhang and Hou [16], which states that if the transmission range R_c is at least twice the sensing range R_s, a complete coverage of a convex area implies connectivity of the active sensors. Wang et al. [14] generalize the result in [16] by showing that,



(a) A triangle (b) The RNG of the triangle

Figure 2. A triangle and its RNG

when the transmission range R_c is at least twice the sensing range R_s, a K-covered network will result in a K-connected network. Then the authors put forward the coverage configuration protocol (CCP) that can dynamically configure the network to provide different coverage degrees required by applications. Carle and Simplot [?] propose another mechanism for energy-efficient connected area coverage for the case when all sensor nodes have the same sensing range and the transmission range equals the sensing range. The goal of the algorithm is to use one of the existing protocols (e.g. Dai and Wu's algorithm [4]) to select an area-dominating set of nodes of minimum cardinality, such that the selected set covers the given area.

Here we will study the relationship between coverage and connectivity for cable sensors and develop distributed algorithms to let each cable sensor decide whether it should be active based on its local information so as to save energy.

4. Saving communication energy by reducing transmission power

In this section, we work on distributed algorithms to save communication energy by reducing the total transmission power of processors while preserving network connectivity. In this method, only the processors of the cable sensors are involved, so instead of drawing the whole cable sensors in the figures, only the processors are shown in this section. This method can be applied to point sensor networks as well.

4.1 Problem formulation

Given a WCSN, assume the processors on the cable sensors can adjust their transmission powers $p_1, p_2, \cdots, p_n$, minimize the total transmission power $\sum_{i=1}^{n} p_i$ while maintaining the connectivity of the network.

4.2 Determine transmission power using RNG

Our first algorithm was inspired by RNG [12] in graph theory. RNG is an undirected graph defined on a set of points in the Euclidean plane by connecting two points u and v by an edge whenever there does not exist a third point w that is closer to both u and v than they are to each other. In other words, if there exists such a w, points u and v should not be connected. For a simple example, if there is a triangle uvw as shown in Fig. 2 and uv is the largest edge in the triangle, the RNG of the triangle contains only two edges: uw and vw. RNG has a nice property as stated in Theorem 4.1.

Theorem 4.1 . *Given a weighted graph $G = (V, E)$, RNG(G) contains a minimum spanning tree $MST(G)$ as a subgraph.*

Proof. If there is a triangle in a weighted graph G, the triangle can be looked as

Algorithm DTRNG: Determine the Transmission power using RNG to minimize total transmission power

```
Each processor $u$ calls Algorithm DTNBOR to determine the minimum transmission power to reach each of its neighbors.
repeat
    Each processor $u$ checks the following condition:
        $p_{|uw|} \leq p_{|uv|}$ and $p_{|vw|} \leq p_{|uv|}$
        $\forall w \in N(u) \cup N(v)$        (1)
    If it is true, processor $u$ will remove $v$ and the transmission power to reach $v$ from its neighbor table.
until there are no more removals.
Each processor will use $p_{|largest\ edge\ incident\ on\ it|}$ as its transmission power.
```

Figure 3. Algorithm DTRNG

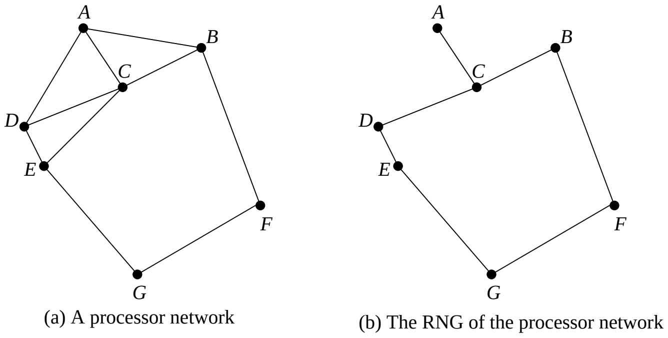
Algorithm DTNBOR: Determine the minimum Transmission power to reach each NeighBOR

```
Each processor needs to build its neighbor table which contains the IDs of its neighbors and the transmission powers to reach them. Initially, the IDs and the transmission powers are empty.
Each processor $u$ starts a timer and sends out a Hello message containing its ID and its neighbor table using its maximum transmission power.
repeat
    If a processor $v$ receives a Hello message from processor $u$, it will add $u$ to its neighbor table and record the transmission power $u$ uses to reach $v$, and send out a REPLY message containing its ID and its neighbor table.
    If processor $u$ receives a REPLY message from a processor $v$, it will add $v$ to its neighbor table and update the transmission power $u$ uses to reach $v$. Then $u$ will reduce its transmission power level to $v$ by $\delta$, start a timer and send out a Hello message to $v$ containing its ID and its neighbor table using the reduced transmission power.
    If processor $u$ does not hear from $v$ before the timer expires, it will use the current transmission power recorded in its neighbor table as the minimum transmission power to reach $v$.
until there are no more changes in each processor's neighbor table.
```

Figure 4. Algorithm DTNBOR

a special case of a cycle. Based on the later Lemma 4.2 in this paper, the largest edge in a cycle cannot be in $MST(G)$. On the other hand, RNG is generated by removing the largest edge in every triangle in G. Therefore, $MST(G)$ is a subgraph of $RNG(G)$. □

Using this property of RNG, the distributed algorithm for each processor to decide its transmission power can be written in Algorithm DTRNG (see Fig 3). This algorithm first calls Algorithm DTNBOR (see Fig.4) to find all the neighbors of a processor and the different minimum transmission powers it uses to reach them. In Algorithm DTNBOR, a processor first sends out a Hello message with its maximum transmission power. If it gets a REPLY from another processor, that means they are neighbors. Next it will repeatedly reduce the transmission power to reach the neighbor until it can no longer hear from it before the timer expires.



(a) A processor network (b) The RNG of the processor network

Figure 5. Decide each processor's transmission power by constructing an RNG

Then the minimum transmission power to reach the neighbor is known. It should be noted that δ in the algorithm decides the precision of the minimum transmission power. The smaller it is, the more accurate the minimum transmission power is, but the longer the algorithm will run before termination and the more energy will be consumed. To reduce the energy consumption of Algorithm DTNBOR, one way is not to make δ too small each time and another way is to piggyback the DTNBOR process with normal message transmission.

After each processor has built its neighbor table, it can generate the RNG of the network by checking condition (1) at Step 3 in Algorithm DTRNG. In condition (1), $p_{|xy|}$ indicates the minimum transmission power to connect processors x and y. $N(x)$ is the neighbor set of x, and w is any other processor in the union of u and v's neighbors. The meaning of condition (1) is that an edge uv will not be included in the topology when there exists a neighbor w such that both $p_{|uw|} \leq p_{|uv|}$ and $p_{|vw|} \leq p_{|uv|}$ are true. In other words, edge uv will be removed if it is the largest edge in triangle uvw. As proved by Theorem 4.1, RNG contains an MST of the network as a subgraph. So it is guaranteed that the resultant topology is connected. Thus, each processor can use transmission power level $p_{|largest\ edge\ incident\ on\ it|}$ to cover the largest edge incident on it in RNG to transfer messages.

We use an example to explain the DTRNG algorithm. Suppose there is a network of processors connected as in Fig. 5(a). Using the DTNBOR algorithm, each processor builds its neighbor table containing the IDs of its neighbors and the minimum transmission powers to reach them. For example, processor A finds three neighbors $\{B, C, D\}$ and the minimum transmission powers to reach them are: $\{p_{|AB|}, p_{|AC|}, p_{|AD|}\}$. Next each processor will remove the largest edge in every triangle having it as a vertex. The resultant graph is an RNG of the original graph (see Fig. 5(b)). And then each processor can decide its transmission power based on the largest edge incident on it in the RNG graph. For example, processor C has three edges AC, BC and DC incident on it, in which, DC is the largest. So processor C will use $p_{|DC|}$ as its transmission power level.

Drawback of the RNG method: The essence of the RNG method is to adjust the transmission powers of processors by removing the largest edge in every triangle in RNG. Even so, it still may not produce the minimum total processor transmission power. For example, in Fig. 6, using the RNG method, the largest edge AC in triangle ABC is removed. Then for processor A, it has AE and AB incident on it, of which AE is longer. So it will use $p_{|AE|}$ as its transmission power. Similar to A, processor E will use $p_{|AE|}$ as its transmission power. However, if AE is also removed, the graph is still connected and A can use $p_{|AB|}$ and E can use $p_{|ED|}$ to further reduce the total transmission power of processors. Therefore, the RNG method can maintain the connectivity of the network, but may not minimize $\sum_{i=1}^{n} p_i$. As we take another look at the figure, AE is the longest edge in the cycle $ABCDEA$. So

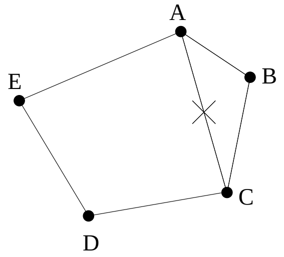


Figure 6. An example of the drawback of the RNG method

Algorithm DTCYC: Determine the Transmission power by removing the largest edge in CYCles to minimize total transmission power

```
1: Each processor $u$ calls Algorithm DTNBOR to find the minimum transmission
   power to reach each of its neighbors.
2: repeat
3:    Each processor $u$ starts a timer and broadcasts a 'Find_cycle' message including
      its ID using the maximum of all the minimum transmission powers to reach its
      neighbors obtained from Step 1.
4:    If a processor $v$ gets a 'Find_cycle' message, it will relay the message by adding
      its ID and the transmission power to reach it from its direct sender to all its neighbors.
5:    Before the timer expires, if $u$ receives its own 'Find_cycle' message and the path
      containing all the relay nodes and the transmission power between each pair, then it
      knows it is involved in a cycle.
6:    Based on the information $u$ gets, $u$ can find the largest edge in the cycle. Then it
      will send a 'Remove_edge' message including the two vertices of that edge to the next
      processor on the cycle path.
7:    If a processor $v$ receives the 'Remove_edge' message and the two vertices for the
      first time and if it is one of the vertices of that edge and it has an edge to the
      other processor $w$ on that edge, it will remove that edge by removing $w$ and the
      transmission power to reach it from its neighbor table; If $v$ has received the same
      'Remove_edge' message before, it will drop the message; Otherwise, it just relays
      the message on.
8: until there are no more changes in each processor's neighbor table.
9: Each processor will use $p_{|largest\ edge\ incident\ on\ it|}$ as its transmission power.
```

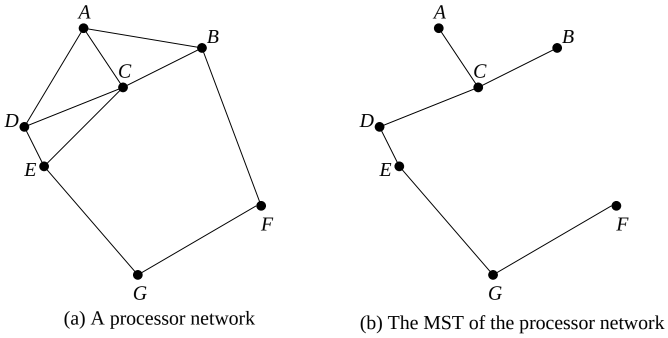
Figure 7. Algorithm DTCYC

if we remove the longest edge in a cycle, or every cycle, can we get better results? This is what we will address in the next section.

4.3 Determine transmission power by removing the largest edge in cycles

As we know, given a random weighted graph $G = (V, E)$, the RNG method can be looked as each processor remove the largest edge of every triangle that includes the processor as a vertex. Now, we can extend this idea to let a processor remove the largest edge in every cycle that includes it as a vertex. Thus, we develop Algorithm DTCYC (see Fig. 7) to determine the transmission power of each processor using this idea.

As in Algorithm DTRNG, Algorithm DTCYC first calls Algorithm DTNBOR to build the neighbor table for each processor. Next, there are two parts in the repeat section. In the first part, each processor needs to find out if there is any cycle involving it as a vertex by broadcasting a 'Find_cycle' message. Multiple



(a) A processor network (b) The MST of the processor network

Figure 8. Determine each processor's transmission power by removing the largest edge in cycles

processors may start the message at the same time and all of these messages will be relayed since they are initiated by different processors. In the second part, a processor removes the largest edge in every cycle involving it as a vertex. Multiple processors may send out the 'Remove_edge' message to remove the same edge. A receiver only deals with the first copy of the message and the rest of the copies will be dropped.

We use the same example in Algorithm DTRNG to explain this algorithm. The same processor network is shown in Fig. 8(a). After each processor builds its neighbor table using the DTNBOR algorithm, it tries to find out if it is involved in any cycle. If it is, it will remove the largest edge in every cycle. For example, processor B removes edges AB and BF because AB is the largest edge in triangle ABC and BF is the largest edge in cycle $BCEGFB$. All the other processors will do the same thing. Eventually the resultant subgraph generated is an MST of the original topology as is proved by Lemma 4.3 and is shown in Fig. 8(b). After the MST is constructed, each processor can use the transmission power level to cover the largest edge incident on it to transfer messages. Comparing with the DTRNG algorithm, the DTCYC algorithm can further reduce the overall transmission power of processors. In this example, processor B only removes edge AB in the DTRNG algorithm, but it also removes edge BF in the DTCYC algorithm. So instead of using the transmission power level to cover $|BF|$ in the RNG graph, now it can use a smaller power level to cover $|BC|$. The following lemmas and theorems prove that Algorithm DTCYC is an optimal solution to our problem. The MST generated guarantees that the processors are connected and $\sum_{i=1}^{n} p_i$ is minimized. Also note that the order of removing the largest edge in every cycle will not affect the minimization of $\sum_{i=1}^{n} p_i$ since the resultant graph is an MST.

Lemma 4.2. *Given a weighted graph $G = (V, E)$, the largest edge e in any cycle $\notin MST(G)$.*

Proof. This can be proved by contradiction. Without loss of generality, suppose the largest edge e in any cycle $e_1e_2 \cdots e_n$ is in $MST(G)$. To make a tree, one edge in the cycle except e needs to be removed. Since e is the largest edge in the cycle, removing e will result in a spanning tree smaller than the one removing any other edge in the cycle. Therefore, the spanning tree without e is smaller than the one with e. So the spanning tree containing e cannot be the minimum. This conflicts with the assumption and proves the lemma. ☐

Lemma 4.3. *Given a weighted graph $G = (V, E)$, after removing the largest edge e in every cycle, the resultant graph is an $MST(G)$.*

Proof. For a network with n vertices, after removing the largest edge e in every cycle, the resultant graph is a tree and has $n - 1$ edges. Adding a new edge to the resultant graph will create a cycle and that new edge must be the largest edge in this cycle, otherwise it would not have been removed in the first place. Thus replacing an existing edge in the cycle with the new edge will generate a spanning tree bigger than the resultant tree since the largest edge in any cycle cannot be in $MST(G)$ according to Lemma 4.2. This proves that the resultant graph is an $MST(G)$. □

Theorem 4.4. *Given a WCSN, assume the processors can adjust their transmission powers* $p_1, p_2, \cdots, p_n$*, the DTCYC algorithm can minimize* $\sum_{i=1}^{n} p_i$ *while maintaining the connectivity of the network.*

Proof. According to the DTCYC algorithm, the largest edge in every cycle involving any processor vertex is removed and each processor can adjust its transmission power to cover the largest edge incident on it in the resultant graph. From Lemma 4.3, the resultant graph of the WCSN is an MST. The MST is a subgraph of the original graph and is connected, involves all the vertices, and makes $\sum_{i=1}^{n} p_i$ minimum. □

Theorem 4.5. *Given a WCSN with* n *processors, in the worst case, the DTCYC algorithm will terminate after removing* $\frac{n^2}{2} - \frac{3}{2}n + 1$ *edges.*

Proof. In a WCSN with n processors, the maximum number of edges is $\frac{n(n-1)}{2}$ in a complete graph. The number of edges in the resultant MST is $n - 1$. Therefore, in the worse case, DTCYC algorithm will remove $\frac{n(n-1)}{2} - (n-1) = \frac{n^2}{2} - \frac{3}{2}n + 1$ edges before it terminates. □

5. Saving communication energy by scheduling cable sensors

In this part, we study another method to save communication energy in WCSNs. Here, we assume the cable sensors are randomly deployed on a terrain. The idea is to let cable sensors take turns to go to sleep, but at the same time, the terrain should still be K-covered and K-connected. We first explore the relationship between coverage and connectivity and then design a localized distributed algorithm to determine the minimum number of active cable sensors.

5.1 Problem Formulation

We can formulate the problem as follows: Given a convex terrain A, and a coverage degree K specified by the application, find the minimum number of cable sensors to stay active to guarantee that A is K-covered and the backbone formed by the active cable sensors is K-connected.

Here, K-coverage means that every point in the terrain is covered by at least K cable sensors and K-connected means that there does not exist a set of $K-1$ cable sensors whose removal disconnects the graph. In this problem, there are two issues: coverage and connectivity. First we want to know if they are related. If one implies the other, then if we satisfy the stronger one, the other is also satisfied. Obviously connectivity does not imply coverage because if a network on a terrain is connected, it can happen that not every point in the terrain is covered by some cable sensor. On the other hand, if a terrain is fully covered by a sensor network, is the network connected? The following two theorems show the conditions for coverage to imply connectivity with 1-coverage and K-coverage. The conditions are true regardless of the locations of processors on cable sensors.

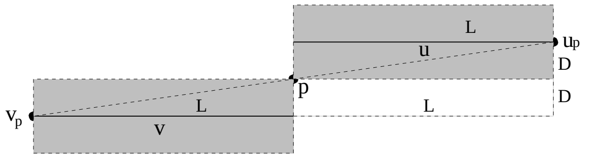


Figure 9. The largest distance of two neighboring cable sensors

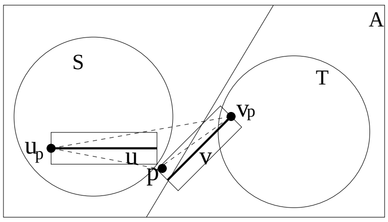


Figure 10. A disconnected network

5.2 Sufficient condition for 1-coverage to imply connectivity

Theorem 5.1. *For a set of cable sensors having sensing range D and length L that 1-cover a 2-dimensional convex terrain A, the communication graph made by processors is connected if the transmission range $R \geq 2\sqrt{L^2 + D^2}$.*

Proof. For any two cable sensors m and n in terrain A, let P_{mn} be the line joining their processors m_p and n_p. Since terrain A is a convex, P_{mn} remains entirely within A. Thus any point q on P_{mn} is at least 1-covered. Therefore, to each point q (starts from m_p and ends with n_p) on P_{mn}, there exist some closest cable sensors. These cable sensors form a set C containing cable sensors $c_1, c_2, \cdots, c_n$, in which c_1 is m and c_n is n. Any two adjacent cable sensors in this set must intersect at some point p. Otherwise, some point in the terrain will not be covered. Suppose two of the adjacent cable sensors are u and v and u_p and v_p are their processors, respectively, as shown in Fig. 9. The three points u_p, v_p and p have the following relationship: $|u_p v_p| \leq |u_p p| + |v_p p|$. The largest value of $|u_p p|$ or $|v_p p|$ is $\sqrt{L^2 + D^2}$. So the maximum value of $|u_p v_p|$ is $2\sqrt{L^2 + D^2}$. Thus, if the transmission range of cable sensors is greater than or equal to this distance, we can construct a path from m_p to n_p through cable sensors in C. Therefore, the communication graph is connected. □

5.3 Sufficient condition for K-coverage to imply K-connectivity

Theorem 5.2. *A set of cable sensors that K-cover a 2-dimensional convex terrain A forms a K-connected communication graph if $R \geq 2\sqrt{L^2 + D^2}$.*

Proof. We can prove by contradiction. Suppose $K - 1$ cable sensors are removed and the network is disconnected. Then the network is divided at least into two parts S and T as shown in Fig. 10. The cable sensors inside S and T are connected, but there is no connection between any processor in S and any processor in T. Now suppose there is a point p which is outside S but very close to some cable sensor u in S. This point p must exist because if it does not, part S will cover the whole terrain and the network is connected because the cable sensors inside S are connected. After removing $K - 1$ cable sensors, any point in the terrain is still covered by at least one cable sensor. Since p is outside part S, it must be covered by some cable sensor v in part T. The largest distance between p and the processor v_p on cable sensor v is $\sqrt{L^2 + D^2}$. Similarly the largest distance between p and the

Figure 11. w is ineligible if $K = 1$

processor u_p on cable sensor u is also $\sqrt{L^2 + D^2}$. Again the three points p, v_p, u_p satisfy: $|u_p v_p| \leq |u_p p| + |v_p p|$. For $|u_p v_p|$ to reach maximum, it should be equal to: $2\sqrt{L^2 + D^2}$. Now that $R \geq 2\sqrt{L^2 + D^2}$, u_p and v_p can communicate with each other. So there is a connection between a processor in S and a processor in T. In other words, the network is connected. It contradicts with the assumption that the network is not connected. Therefore, the theorem is true. $\square$

5.4 The CMT algorithm

From the above theorems, we know that if a terrain is K-covered and if the transmission range $R \geq 2\sqrt{L^2 + D^2}$, then the network is K-connected. In this section, we present the cable mode transition (CMT) algorithm to determine the minimum number of active cable sensors to maintain K-coverage specified by the application as well as K-connectivity. The idea was inspired by [14]. The difference is that their approach applies to traditional point sensors with a disc-shaped sensing region and our approach applies to cable sensors with a rectangular sensing region.

A cable sensor can be in one of the three modes with the energy consumption from the highest to the lowest: ACTIVE, SNOOPY and SLEEP. In the ACTIVE mode, a cable sensor actively senses and communicates with other cable sensors; in the SNOOPY mode, each cable sensor collects HELLO messages from its neighboring sensors and checks its eligibility to determine its new mode; and in the SLEEP mode, a cable sensor sleeps to save energy. The CMT algorithm describes the rules of cable mode transition and generates the minimum number of active cable sensors to maintain K-coverage and connectivity. Before presenting CMT, we introduce the eligibility algorithm it calls whose role is to make each cable sensor check its eligibility to stay active.

5.5 K-coverage eligibility algorithm

Each cable sensor executes an eligibility algorithm to determine whether it is necessary to stay active. Given a requested coverage degree K, a cable sensor v is ineligible to stay active if every location within its coverage region is already K-covered by other active cable sensors in its neighborhood. For example, in Fig. 11, cable sensors u and v are active and cable sensor w is ineligible if $K = 1$, but eligible if $K > 1$.

Before presenting the eligibility algorithm, we define the following concepts:

- The *sensing region* of cable sensor v: Contains all the points p such that $|pv| < D$.
- An *intersection point* p of two cable sensors u and v: Denoted by $p \in u \wedge v$, is an intersection point of the sensing rectangles of u and v.
- An *intersection point* p of a cable sensor v and terrain A: Denoted by $p \in v \wedge A$, is an intersection point of the sensing rectangle of cable sensor v and terrain A.

Note that the intersection points of two cable sensors and between a cable sensor and a terrain A are different from regular definitions. Here they are formed by the

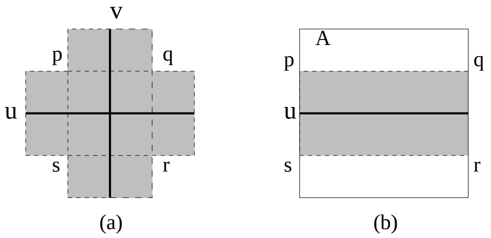


Figure 12. The intersection points of two cable sensors and between a cable sensor and a terrain

sensing rectangles of cable sensors, not cable sensors themselves. As shown in Fig. 12(a)(b), the intersection points of two cable sensors u and v and between cable sensor u and terrain A are p, q, r and s.

Also note that when deployed in a terrain, two cable sensors may not be parallel or perpendicular to each other. They can intersect with any angle. But for the sake of convenience and without affecting the results, Fig. 12 and later Figs. 13 and 14 draw cable sensors parallel or perpendicular to each other.

Theorem 5.3. *A convex terrain A is K-covered by a set of cable sensors C if 1) there exist in terrain A intersection points between cable sensors or between cable sensors and A's boundary; 2) all intersection points between any cable sensors are at least K-covered; and 3) all intersection points between any cable sensor and A's boundary are at least K-covered.*

Proof. We prove by contradiction. Let p be the point that has the lowest coverage degree k in terrain A and $k < K$. Also we assume that there is no intersection point in A which is covered to a degree less than K. The set of sensing rectangles partition A into a collection of *coverage patches*. Each of them is bounded by edges of sensing rectangles and/or the boundary of A, and all points in coverage patch have the same coverage degree. Suppose point p is located in coverage patch S. First we prove that the interior edge of any sensing rectangle cannot serve as the boundary of S. We prove by contradiction. Assume there exists an interior edge of sensing rectangle $Rect(u)$ serving as the boundary of S, crossing this edge (*i.e.* leaving the coverage region of cable sensor u) would reach an area that is lower covered than point p. This contradicts with the assumption that point p has the lowest coverage degree in terrain A. So patch S cannot be bounded by an interior edge of any sensing rectangle. Then the boundary of S can fall into the following three cases:

(1) Point p lies in a coverage region S whose boundary is only composed of exterior edges of a collection of sensing rectangles. For example, in Fig. 13, patch S is bounded by the external edges of sensing rectangles of cable sensors u, v, w and x. Furthermore, since the edges of sensing rectangles themselves are outside the sensing region of the cable sensors that define them, the entire boundary of this coverage patch, including the intersection points of the sensing rectangles defining the boundary, has the same coverage degree as point p. This contradicts with the assertion that p is covered to a degree less than K and no intersection point in A has a coverage degree less than K.

(2) Point p lies in a coverage region S that is bounded by the exterior edges of a collection of sensing rectangles and the boundary of A. As shown in Fig. 14, point p is in a region bounded by the exterior edges of sensing rectangles of cable sensors u, v, w and the boundary of terrain A. Similar to case 1), the entire boundary of this coverage patch, including the intersection

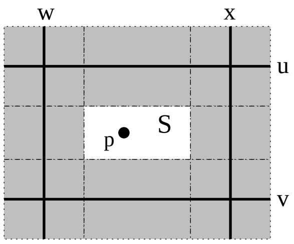


Figure 13. A coverage patch bounded by edges of sensing rectangles of cable sensors

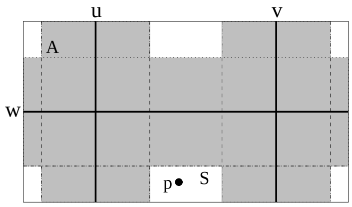


Figure 14. A coverage patch bounded by edges of sensing rectangles and boundary of terrain A

Algorithm Eligibility: determine if a cable sensor is eligible to stay active given a coverage degree K

```
/* find all intersection points within $Rect(v)$ */
$IP = \{p \mid (p \in u \wedge w$ OR $p \in u \wedge A)$ AND $u, w \in SN(v)$ AND $|pv| < D\}$;
/* find all overlapping cable sensors */
$OC = \{u| \ \ |uv| = 0\}$;
if $|IP| = 0$ then
  if $|OC| \geq K$ then
    return INELIGIBLE;
  else
    return ELIGIBLE;
  end if
end if
for each point $p \in IP$ do
  /* compute $p$'s coverage degree */
  $cd(p) = |\{u|u \in SN(v)$  AND  $|pu| < D\}|$
  if $cd(p) < K$ then
    return ELIGIBLE;
  end if
end for
return INELIGIBLE;
```

Figure 15. The K-coverage eligibility algorithm

points of cable sensors u, v, w and intersection points between cable sensors u, v, w and boundary of A, has the same coverage degree as point p. This contradicts with the assertion that p is covered to a degree less than K and no intersection point in A has a coverage degree less than K.

(3) Point p lies in a coverage patch that is bounded only by the boundary of terrain A. If so, terrain A will have the same coverage as point p. This contradicts with the assumption that terrain A is K-covered.

From the above discussion, point p with lower coverage degree than K does not exist. Thus terrain A is K-covered. ☐

Theorem 5.3 converts the problem of finding the coverage degree of a terrain to the simpler problem of finding the coverage degrees of all the intersection points in the terrain. A cable sensor is ineligible to stay active if all the intersection points inside its sensing rectangle are at least K-covered. To find all the intersection points inside its sensing rectangle, a cable sensor v only needs to know the locations of all the cable sensors in its sensing neighbor set, $SN(v)$. $SN(v)$ should include all the active cable sensors that are within the maximum distance between the processor on v and the processor on another cable sensor that attaches to v. As shown in Fig. 9, that distance is the distance between processor v_p on v and u_p on cable sensor u, which is $\sqrt{(2L)^2 + (2D)^2} = 2\sqrt{L^2 + D^2}$. Cable sensors can find their neighbors through exchanging Hello messages. Since the intersection points are decided by the relative positions between cable sensors or between a cable sensor and terrain A, a local coordination system can be used. If the actual transmission range R used by a cable sensor is greater than or equal to $2\sqrt{L^2 + D^2}$, the Hello message from each cable sensor only needs to include its own id and location. If $R < 2\sqrt{L^2 + D^2}$, a cable sensor may not find all its neighbors through such Hello messages in one hop. Then, more cable sensors will stay active because of its limited information. This is proved by the simulation in the next section. Also if $R < 2\sqrt{L^2 + D^2}$, the network is not guaranteed to be connected as indicated by Theorem 5.2.

The resulting algorithm for a cable sensor v to check whether it is eligible to stay active or not is shown in Fig. 15. Algorithm Eligibility has three parts: first, cable sensor v finds all the intersection points (of cable sensors and between cable sensors and terrain A) within its rectangular sensing region and puts these points into the intersection point set IP. Next, cable sensor v tries to find out if there are any *overlapping cable sensors* which are the cable sensors happen to be placed in the same location as itself. If so, these cable sensors are put into the overlapping cable sensor set OC. If there is no intersection point within v's sensing rectangle and the number of overlapping cable sensors $|OC|$ is at least K, cable sensor v is ineligible. Otherwise, it is eligible. Finally, cable sensor v calculates $cd(p)$, the coverage degree of every intersection point p within its sensing region. If the coverage degree is less than K, cable sensor v is eligible; otherwise, it is not eligible. The computation complexity of the eligibility algorithm is $O(N^3)$ where N is the number of cable sensors in the sensing neighbor set.

5.6 Cable sensor mode transition algorithm

Now the CMT algorithm is presented in Fig. 16. A cable sensor can transit among SLEEP, ACTIVE and SNOOPY modes. Initially all cable sensors are ACTIVE. If the terrain coverage goes over the required degree of the application, redundant cable sensors will find themselves ineligible to stay active and go to the SLEEP mode until no more cable sensors can go to sleep without causing insufficient degree of coverage. On the other hand, if the failure of a cable sensor makes the terrain coverage fall below the required degree, some cable sensors will find themselves eligible and go to the ACTIVE mode. The times set by the join and withdraw timers are randomly generated so as to reduce the possibility of collisions among multiple cable sensors wanting to join or withdraw simultaneously.

6. Simulations

In this section, we conduct simulations to evaluate our energy-efficient communication algorithms. We only do simulations for Algorithm CMT since Algorithm DTCYC has been mathematically proved to be an optimal solution to our problem. Our simulator is self-written because there are no available ones for cable sensors.

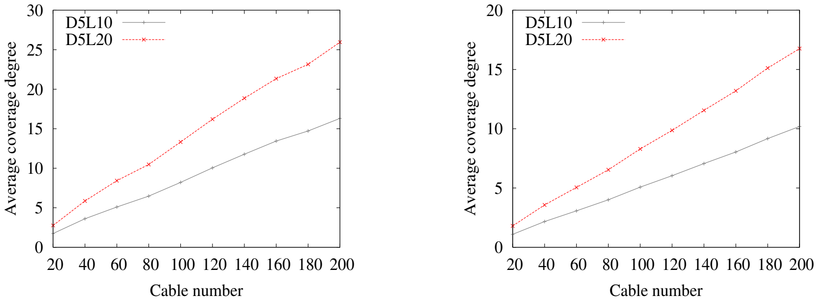
Cable Mode Transition (CMT) Algorithm

1: If a cable sensor is in the SLEEP mode and its sleep timer expires, it turns on, starts a snoopy timer and goes to the SNOOPY mode.

2: If a cable sensor is in the SNOOPY mode and receives either HELLO, JOIN, or WITHDRAW message, it calls the eligibility algorithm (in Fig. 15) to see if it is eligible to stay active. If it is, it starts a join timer; else it goes to the SLEEP mode. After the join timer starts and if it becomes ineligible (e.g. because of a JOIN message from a communicating neighbor), it cancels the join timer. If the join timer expires, the cable sensor broadcasts a 'Join' message and goes to the ACTIVE mode. If the snoopy timer expires, it starts a sleep timer and goes to the SLEEP mode.

3: If a cable sensor is in the ACTIVE mode and receives a HELLO message, it updates its neighbor table and calls the eligibility algorithm to see if it should remain active. If it should not, it starts a withdraw timer. Before the withdraw timer expires and if it becomes eligible (e.g. because of a WITHDRAW message from a communicating neighbor), it cancels the withdraw timer. If the withdraw timer expires, it broadcasts a 'WithDraw' message, starts a sleep timer and goes to the SLEEP mode.

Figure 16. Cable mode transition algorithm



(a) Average coverage degree achieved using D5L10 cable sensors in a 30 × 30 terrain

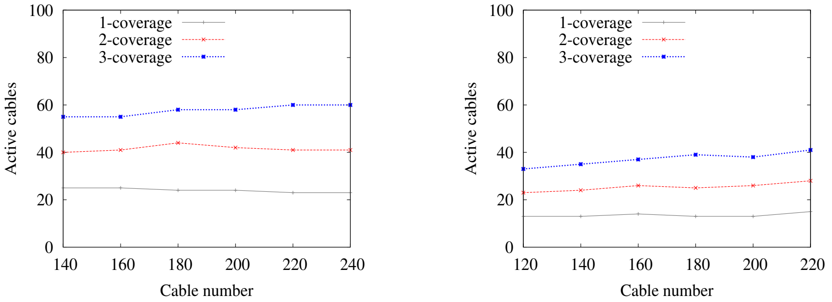
(b) Average coverage degree achieved using D5L20 cable sensors in a 40 × 40 terrain

Figure 17. Average coverage degree achieved using different numbers of cable sensors

It is written in C language. Since this is the first paper on cable sensors, so we just do experiments to evaluate the effectiveness of our algorithm by itself.

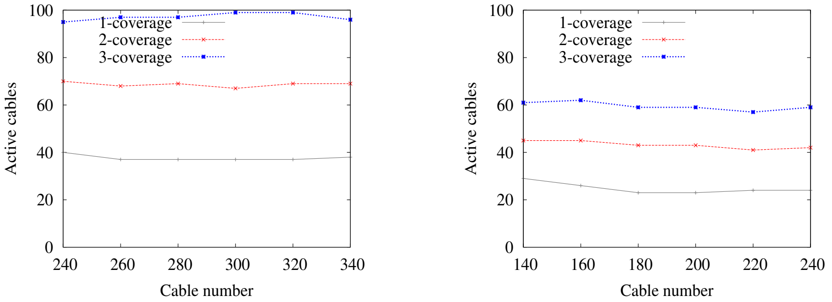
In the first simulation, we want to see the relationship between the number of cable sensors deployed and the average coverage degree achieved. To measure coverage, we divide the entire terrain into 1 × 1 patches. The coverage degree of a patch is approximated by measuring the number of active cable sensors that cover the center of the patch. We use terrains of 30 × 30 and 40 × 40. In each terrain, we try two kinds of cable sensors: D5L10 (sensing range 5 and length 10) and D5L20 (sensing range 5 and length 20). Note that we do not have a unit for these numbers because what matters here is the relative size of the cable sensors to the terrain. The cable sensors are randomly deployed onto the terrain in all directions. We start from 20 cable sensors and go up to 200 with an increment of 20 in each step. The coverage degree of each location is calculated and the final coverage degree of the terrain is the average of all the locations. The results are shown in Figs. 17(a)(b).

From the results, we can see that (1) in a larger terrain, the coverage degree is lower with the same number of cable sensors; (2) in a terrain, with the increase of the number of cable sensors, the coverage degree goes up; (3) with the same number of cable sensors, the D5L20 cable sensor has a higher coverage degree than that of the D5L10 cable sensor because it is longer and thus has a larger coverage



(a) Active D5L10 cable sensors needed in a 30×30 terrain for $1, 2,$ and 3-coverage

(b) Active D5L20 cable sensors needed in a 30×30 terrain for $1, 2,$ and 3-coverage



(c) Active D5L10 cable sensors needed in a 40×40 terrain for $1, 2,$ and 3-coverage

(d) Active D5L20 cable sensors needed in a 40×40 terrain for $1, 2,$ and 3-coverage

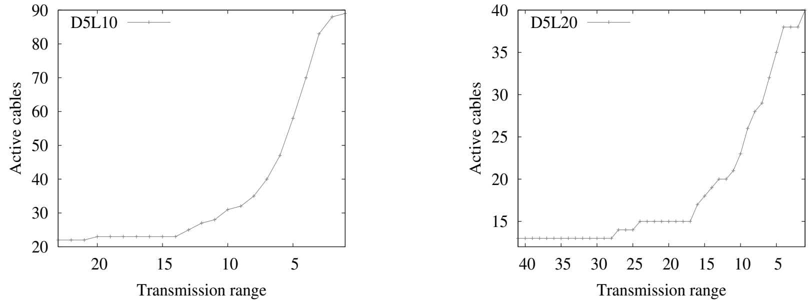
Figure 18. Active cable sensors needed for $1, 2,$ and 3-coverage

region.

In the second simulation, we want to find out how many cable sensors need to be active out of all the cable sensors deployed to achieve coverage degrees $1, 2,$ and 3 using the CMT algorithm. Again we use the D5L10 and D5L20 cable sensors and deploy them on two terrains 30×30 and 40×40. In terrain 30×30 using the D5L10 cable sensors, we start from 140 cable sensors because this is the number to guarantee $1, 2,$ and 3-coverage for the whole terrain. Similarly in terrain 30×30 using the D5L20 cable sensors, the starting number is 120 cable sensors. And in terrain 40×40, the starting cable sensor numbers for D5L10 and D5L20 are 240 and 140, respectively, for the same reason. The results are shown in Figs. 18(a)(b)(c)(d). From the results, we can see that (1) more cable sensors need to be active to have a higher coverage degree; (2) the number of active cable sensors does not increase with the increase of cable sensor numbers. That means, our algorithm does not wake up more cable sensors because the coverage goal has already been achieved.

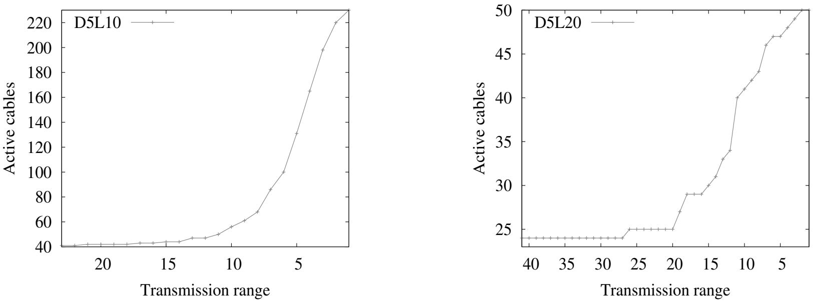
In the third simulation, we want to explore the effect of different transmission ranges to the number of active cable sensors using the CMT algorithm. We still deploy the D5L10 and D5L20 cable sensors on the 30×30 and 40×40 terrains. To guarantee that a terrain is 1-covered, we use different numbers of cable sensors for each setting. For example, for terrain 30×30 with D5L10, we use 90 cable sensors because this is the number to make sure that the terrain is 1-covered. For terrain 30×30 with D5L20, we use 40 cable sensors. For terrain 40×40 with D5L10 and D5L20, the number of cable sensors used is 230 and 50, respectively. To make cases general, we put a processor in a random location on each cable sensor. We start with a transmission range R that is greater than or equal to $2\sqrt{L^2 + D^2}$, then decrease it gradually. The results are shown in Figs. 19(a)(b)(c)(d).

From the results we can see that if the transmission range is greater or equal to $2\sqrt{L^2 + D^2}$, many cable sensors can go to sleep because this is the range that they can fully detect their neighbors and find out if every location in their sensing



(a) Active D5L10 cable sensors needed with different transmission ranges in a 30 × 30 terrain

(b) Active D5L20 cable sensors needed with different transmission ranges in a 30 × 30 terrain



(c) Active D5L10 cable sensors needed with different transmission ranges in a 40 × 40 terrain

(d) Active D5L20 cable sensors needed with different transmission ranges in a 40 × 40 terrain

Figure 19. Active cable sensors needed for different transmission ranges

region is 1-covered by other active cable sensors. For example, in the 30×30 terrain with the D5L10 cable sensors, only 22 cable sensors need to be active out of 90 cable sensors deployed to 1-cover the whole terrain. The saving is 76%. Then we reduce the transmission range. At the beginning, the number of active cable sensors remains the same because the transmission range is still greater than the distance between two processors on cable sensors. Then with the further decrease of the transmission range, each cable sensor cannot fully detect all of its neighbors. From the limited neighbors that a cable sensor can detect, it will more and more likely decide to stay active because not all the locations within its sensing region can be covered by its detected neighbors. After a certain point, for example, in the 40×40 terrain with the D5L20 cable sensors, when the transmission range reduces to 2 or 1, no cable sensor can go to sleep because each cable sensor thinks it is the only one to cover the locations in its sensing region.

In summary, the simulation results show that the CMT algorithm can make as many cable sensors as possible to go to sleep without affecting the connectivity of the network according to the required coverage degree K from the application.

7. Conclusion and future work

In this paper, we introduced a new type of sensor: cable sensor. We discussed the energy-efficient communication algorithms in WCSNs, we addressed it in two ways: One is through reducing the total transmission power of processors and the other is to schedule cable sensors to let them take turns to go to sleep without affecting the coverage and connectivity of the network. In the first approach, we initially developed a distributed algorithm called DTRNG based on RNG. Then we discovered that it may not minimize the total transmission power of processors. So we enhanced it to Algorithm DTCYC. Mathematical proofs showed that the DTCYC algorithm provides an optimal solution to our problem and results in an MST, which can not only minimize the total processor transmission power but maintain the connectivity of the network as well. In the second approach, we proposed the CMT algorithm, which determines the minimum number of active sensors to maintain K-coverage as well as K-connectivity. We first found the relationship between coverage and connectivity and proved the theorems that lay the foundation for our algorithm. Simulation results demonstrated that our algorithm is efficient in saving energy. Our future work will include deploying cable sensors on a real field to measure their sensing ability and communication efficiency. In this paper, we just focused on area coverage of cable sensors. In the future, we will discuss other coverage models such as point coverage and barrier coverage. Also we will address the interference between cable sensors which is critical in multi-hop context after we find out the interference model. Since the cable sensors are different from the traditional point sensors, basically a lot of issues related to the point sensors can be reexamined and studied.

Acknowledgments

This research was supported in part by NSF grant CBET 0729696.

References

[1] I. Akyildiz, W. Su, Y. Sankarasubramanian, and E. Cayicrci, *A survey on sensor networks*, IEEE Communications Magazine, Vol. 40, No. 8, 2002, pp. 102-116.

[2] M. Cardei, D.-Z. Du, *Improving wireless sensor network lifetime through power aware organization*, ACM Wireless Networks, Vol. 11, No. 3, 2005.

[3] J. Cartigny, D. Simplot, and I. Stojmenovic, *Localized Minimum-Energy Broadcasting in Ad-Hoc Networks*, Proc. of Infocom, pp. 2210-2217, 2003.

[4] F. Dai, J. Wu, *Distributed dominant pruning in ad hoc wireless networks*, Proc. of IEEE International Conference on Communications, 2003.

[5] I. A. Essa, *Ubiquitous sensing for smart and aware environments*, IEEE Personal Communications, Vol. 7, 2000, pp. 47-49.

[6] N. Li, J.C. Hou, and L. Sha, *Design and Analysis of an MSTBased Topology Control Algorithm*, Proc. of Infocom, vol. 3, pp. 1702-1712, Mar./Apr. 2003.

[7] A. M. Mainwaring, D. E. Culler, J. Polastre, R. Szewczyk, and J. Anderson, *Wireless sensor networks for habitat monitoring*, Proc. of the 1st ACM INTernational Workshop on Wireless Sensor Networks and Applications (WSNA), Atlanta, Georgia, USA.

[8] D. A. Patterson, *Rescuing our families, our neighbors, and ourselves*, Commun. ACM, Vo. 48, No. 11, 2005, pp. 29-31.

[9] J. M. Rabaey, M. J. Ammer, J. L. da Silva Jr., D. Patel, and S. Roundy, *Picoradio supports ad hoc ultra-low power wireless networking*, IEEE Computer, Vol. 33, No. 7, 2000, pp. 42-48.

[10] V. Rodoplu and T.H. Meng, *Minimum Energy Mobile Wireless Networks*, IEEE J. Selected Areas in Comm., vol. 17, no. 8, pp. 1333-1344, Aug. 1999.

[11] N. Rowe, *Efficient deployment of fiber-optic cable seismic sensors*, Proc. of SIPE, April, 2010.

[12] P. Santi, *Topology control in wireless ad hoc and sensor networks*, Wiley, 2005.

[13] A. Sixsmith and N. Johnson, *A smart sensor to detect the falls of the elderly*, IEEE Pervasive Computing, Vol. 3, 2004, pp. 42-47.

[14] 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. of the 1st ACM SenSys, 2003, pp. 28-39.

[15] J.E. Wieselthier, G.D. Nguyen, and A. Ephremides, *On Constructing Minimum Spanning Trees in k-Dimensional Spaces and Related Problems*, Proc. of Infocom, pp. 585-594, 2000.

[16] H. Zhang, J. C. Hou, *Maintaining sensing coverage and connectivity in large sensor networks*, Technical report UIUC, UIUCDCS-R-2003-2351, 2003.