A Fast Fault Tolerant Partitioning Algorithm for Wireless Sensor Networks

Dibakar Saha\textsuperscript{1} and Nabanita Das\textsuperscript{2}

Advanced Computing and Microelectronics Unit
Indian Statistical Institute
203 B.T. Road, Kolkata-700108, India
\textsuperscript{1}dibakar.saha10@gmail.com, \textsuperscript{2}ndas@isical.ac.in

Abstract. In this paper, given a random uniform distribution of sensor nodes on a 2-D plane, a fast self-organized distributed algorithm is proposed to find the maximum number of partitions of the nodes such that each partition is connected and covers the area to be monitored. Each connected partition remains active in a round robin fashion to cover the query region individually. In case of a node failure, the proposed distributed fault recovery algorithm reconstructs the affected partition locally utilizing the available free nodes. Simulation studies show significant improvement in performance compared to the earlier works in terms of computation time, the diameter of each partition, message overhead and network lifetime.

Keywords: Wireless Sensor Network (WSN), Self Organization, Coverage, Partition, Network Lifetime, Fault Tolerant

1 Introduction

In an over deployed Wireless Sensor Network (WSN), a large number of sensor nodes are randomly deployed to monitor a large geographical area. Each sensor node is integrated with processing elements, memory, battery power and wireless communication capabilities. Once deployed, they are, in general left unattended. Hence due to power drainage, hardware degradation, environmental hazards etc. sensor nodes are much prone to failures. For better utilization of the over-deployed nodes to save energy and to extend the lifetime of the network, this paper addresses the problem of finding maximum number of partitions of the sensor nodes such that each partition is connected and covers the whole query region. Instead of keeping all the sensors active always, these partitions will remain active one after another in a round robin fashion. Therefore, if there are $K$ such partitions, the network lifetime will be enhanced by at most $K$ times. Here, given a random uniform distribution of sensor nodes over a 2-D plane, a distributed algorithm is developed for finding the maximum number of partitions of connected nodes such that each partition ensures coverage. In case of node failures, a distributed algorithm is developed for fault recovery that rearranges the affected partition locally to tolerate single node faults within a partition. Simulation studies show that compared to the earlier techniques, the proposed
algorithm is faster and results better partition topology with reduced diameter and requires less message overhead. Also, in case of unpredictable node faults the neighboring nodes execute the localized fault recovery algorithm that rearranges the partition locally to make the system fault-tolerant. Simulation results show that it extends the network lifetime significantly.

The rest of the paper is organized as follows. Section 2 presents a brief outline of related works. Section 3 describes the proposed model. Section 4 includes the proposed algorithms. Simulation results are described in Section 5 and Section 6 concludes the paper.

2 Related Works

Extensive research results have been reported so far addressing the problems of sensing coverage and network connectivity in wireless sensor networks. In many works, the authors considered only the coverage issues in wireless sensor networks. In [1,2], authors propose efficient distributed algorithms to optimally solve the coverage problem in WSN. In [3], authors provide an analytical framework for the coverage problem and lifetime maximization of a WSN. In [4] a decentralized and localized node density control algorithm is proposed for network coverage. The work in [5] proposed three approximation algorithms for the set-k Cover problem, where each point of the query region will be covered by at least \( k \) nodes. The work in [6] considers the problem of maximizing the number of disjoint sets of sensor nodes to cover the query region. But unless the coverage and connectedness problems are considered jointly, the data sensed by the nodes covering the region can not be gathered at the sink node in multi hop WSN’s. Authors of [7], [8] focused on both connectivity and coverage problems with the objective of finding a single connected set cover only. The problem of finding a connected set cover of minimum size is itself an NP-hard problem [4]. Some of the papers considered the fault tolerant connected set cover problems. An approximation algorithm is proposed in [9] for fault tolerant connected set cover problem. In [10], a coverage optimization algorithm based on particle swarm optimization technique is developed. In [11–15], authors proposed several dynamic localized algorithms to achieve the coverage and connectivity. But dynamic algorithms, in general, require large message overhead to collect current neighborhood information at some intervals. Also, finding just a single connected cover keeps a large number of sensors unutilized. Hence, the authors in [16], propose a localized algorithm for finding maximum number of connected set covers that is to be executed once during network initialization only. In some papers [13,15,16], it has been assumed that the query area is a dense grid [17], [18] composed of unit cells. The knowledge of exact location of each node is needed here. A sensor node computes the covered area by counting the cells covered by each neighbor that makes the procedure computation intensive. To avoid this, in [19] the DCSP algorithm is proposed where authors assume that the monitoring area is divided into a limited number of square blocks such that a sensor node within a block completely covers it irrespective of its position within the block. Therefore, the coverage problem can be solved easily with much less computation. However, the proposed distributed algorithm was a slow one
requiring \( p \) rounds to achieve a partition with \( p \) nodes. Also, the fault model considers the faults due to energy exhaustion only that assumes that a node can predict its failure and can inform its neighbors in advance.

In this paper, a faster distributed algorithm requiring less message overhead is proposed that is executed during network initialization only. It attempts to create maximum number of connected partitions of sensor nodes with reduced diameter such that each partition covers the area under investigation and being active in round robin fashion it enhances the network lifetime manifold. The reduced diameter of the partition keeps the communication latency low. Moreover, a distributed fault recovery algorithm is developed for a stronger fault model that in presence of any unpredictable node faults, can rearrange the affected partition locally, so that it remains operational. Simulation studies show that this fault recovery scheme enhances the network lifetime by more than 50%.

3 Proposed Model and Problem Formulation

Let \( n \) homogeneous sensor nodes be deployed over a 2-D region \( P \) each with same sensing range \( S \) and transmission range \( T \). It is assumed that \( P \) is divided into a finite number of square blocks [19]. Each side of the block is \( \frac{R}{\sqrt{2}} \) as shown in Fig.1, where \( R = \min(S, T) \). Therefore, it is evident that each sensor node

**Fig. 1.** Nodes in \( P \) divided into a grid of square blocks

completely covers the block it belongs to and all nodes within the same block are connected to each other. Hence, activating just a single sensor node from each block is sufficient to cover the region \( P \). But it is not guaranteed that any such set is connected or not. As for example, Fig.2 shows a partition where the selected nodes are connected but a block \( B \) is not covered. Whereas, Fig.3 shows a partition where all blocks are covered but the nodes are not connected, and finally, Fig.4 shows the desired topology where the partition covers all blocks as well as it is connected. Assuming this grid structure of the query region \( P \), this paper addresses the connected set cover partitioning problem introduced in [10]. For completeness, the problem is defined below.

**Definition 1.** Consider a sensor network consisting of a set \( S \) of \( n \) sensors and a query region \( P \). A set of sensors \( N \subseteq S \) is said to be a connected 1-cover for \( P \) if, each point \( p \in P \) is covered by at least one sensor from \( N \) and the communication graph induced by \( N \) is connected.
Connected Set Cover Problem Given $n$ sensor nodes distributed over a query region, the Connected Set Cover Problem is to find a connected 1-cover of minimum size. This problem is known to be an NP-hard problem [7].

Connected Set Cover Partitioning Problem The Connected Set Cover Partitioning Problem is to partition the sensors into connected 1-covers such that the number of covers is maximized [16].

The following section describes the algorithms developed for solving the Connected Set Cover Partitioning Problem.

4 Algorithm for Connected Cover Partitioning

In pervasive computing environments, it is evident that in most of the cases the system captures data in distributed nodes communicating through poorly connected network. Since, in WSN large number of sensor nodes are deployed over a geographical area, to collect information of the whole network at a central node is not feasible in terms of message overhead and energy requirement. Instead, it is better to compute in a distributed fashion based on the local neighborhood information using less communication. Hence the focus of our work is on distributed computation of the connected partitions.

In this section, a distributed algorithm is developed to find the maximum number of connected-1 covers of a WSN. Also, in the presence of a node fault, a localized algorithm is presented to rearrange the affected partition to make the system fault tolerant.

4.1 Distributed Algorithm for Partitioning

It is assumed that a set of $n$ sensor nodes $S = \{s_1, s_2, s_3, \ldots, s_n\}$ is deployed on a 2-D plane $P$ divided into say, $m$ square blocks $P = \{p_1, p_2, p_3, \ldots, p_m\}$, as has been described in Section 3. Each square block has unique id. Each sensor node knows the location in terms of its block within which it is located.

We propose the following types of messages to be exchanged among nodes.

- **Selectlist**($C_i, i, \{j\}$) : This message is sent by a node-$i$ that selects a list of neighbors $\{j\}$ for inclusion in its partition with leader $C_i$. 
- **Selected**($C_i, \{j\}$): This message is initiated by the leader node $C_i$ and is sent to node-$\{j\}$ for inclusion in its partition.
- **Confirm**($C_i, j$): Node $j$ sends this message to the leader after joining the partition $C_i$.
- **Include**($C_i, j$): Leader $l$ broadcasts this message within $C_i$ to include node-$j$ in $C_i$.

Depending on the node density, a probability value $0 < l_{prob} < 1$ is determined to select $p$ number of leader nodes randomly. Each node $i \in S$ generates a random number $r$ to check if $r \leq l_{prob}$, the leader probability. If yes, it becomes a leader node and sets its parent as null. If $p$ leader nodes emerge, $L = \{l_1, l_2, l_3, \ldots, l_p\}$, each leader initiates the creation of one partition concurrently to generate disjoint connected set covers. In round 1, each leader $l_i \in L$ initiates a partition $C_i = \{l_i\}$. In each round, each node $i \in C_i$ prepares 'Selectlist' consisting maximum number of neighbors, each one from an uncovered block. In case a node gets more than one node from the same block, it selects the neighbor with minimum degree $D$. Node-$i$ sends a 'Selectlist' message to its parent if it is a leaf node in $C_i$. Else, node $j \in C_i$ selects a list of nodes each belonging to uncovered blocks from its own list and from the received 'Selectlist' messages from its children in same partition. Then it sends the combined 'Selectlist' message to the parent node if it is not a leader node.

The leader node finally from these 'Selectlist' messages selects the nodes to be included and sends the 'Selected' message to them. If a node receives 'Selected' messages, it selects the parent with minimum $D$ and confirms the request by sending a 'Confirm' message to the corresponding leader. The leader includes the node in $C_i$ and broadcasts the 'Include' message to all nodes in $C_i$. On receiving an 'Include($C_i, j$)' message, all nodes $k \in C_i$ include node-$j$ in its

---

**Fig. 5.** steps for making connected set cover

$C_i = \{l_i\}$. In each round, each node $i \in C_i$ prepares 'Selectlist' consisting maximum number of neighbors, each one from an uncovered block. In case a node gets more than one node from the same block, it selects the neighbor with minimum degree $D$. Node-$i$ sends a 'Selectlist' message to its parent if it is a leaf node in $C_i$. Else, node $j \in C_i$ selects a list of nodes each belonging to uncovered blocks from its own list and from the received 'Selectlist' messages from its children in same partition. Then it sends the combined 'Selectlist' message to the parent node if it is not a leader node.

The leader node finally from these 'Selectlist' messages selects the nodes to be included and sends the 'Selected' message to them. If a node receives 'Selected' messages, it selects the parent with minimum $D$ and confirms the request by sending a 'Confirm' message to the corresponding leader. The leader includes the node in $C_i$ and broadcasts the 'Include' message to all nodes in $C_i$. On receiving an 'Include($C_i, j$)' message, all nodes $k \in C_i$ include node-$j$ in its
partition and make necessary updates. In each round, this procedure is repeated until either all blocks are covered by a partition \( C_i \), or no neighbors are left for inclusion.

The formal description of the algorithm is given below.

**Algorithm 1**: Distributed algorithm for connected set cover partitioning

**Input**: 1-hop neighbor list of each node \( NL(i) \) with degree \( D \), Block-Id, Block status, Status, LeaderProbability \( l_{prob} \)

**Output**: Partition \( C_i \) from leader \( l_i \)

*for each node-\( i \) do*

  * if node-\( i \) is a leader then*
    * \( C_i \leftarrow \{i\}, \text{parent}=\phi, \text{status}=1; \)
  *
  * if Status = 1 and not all blocks covered then*
    * Select neighbors from uncovered blocks and received 'Selectlist' messages, selecting nodes with minimum \( D \);*
    * if Parent \( \neq \phi \) then*
      * Send 'Selectlist' to the parent node;*
    *
    * else*
      * if 'Selectlist' = \( \phi \) then*
        * Broadcast success=0 and terminate;*
      *
      * else*
        * Send 'Selected' message to the selected neighbors;*
    *
  *
  * end*

*end*

*if Status = 0 receives 'Selected' message then*

  * Select that partition where the sender have minimum \( D \) and send 'Confirm' message to leader;*
  * Update \( NL(i) \), Block status, Status and include in \( C_i \);*

*end*

*if leader node and receives 'Confirm' message then*

  * Broadcast 'Include' message to all nodes in \( C_i \) and update \( NL(i) \), Block status, Status;*
  *
  * if all blocks are covered then*
    * Broadcast success=1 and terminate;*
  *
  * else*
    * update \( NL(i) \);*

*end*

**Example 1.** In Fig.5, it is shown that in round 1, the leader node (red) selects the neighbors (the green ones) from uncovered blocks. In the next round all black nodes are selected by the red and green nodes. This procedure is repeated to include brown and blue nodes until all blocks are covered. In the last round, all purple nodes are selected and the process is terminated as no uncovered block exists.

It is clear that in each round of the procedure, the nodes already in partition includes several neighbors in the partition so that the partition remains connected with new nodes covering additional blocks. Hence, the procedure terminates faster, each leader either results a successful partition satisfying the condition of connectedness and coverage or it reports a failure when the nodes in the incomplete partition declare them as free nodes.

### 4.2 Distributed Fault Recovery Algorithm

As it has been mentioned in Section 1, once deployed the sensor nodes may fail due to low energy, hardware degradation, inaccurate readings, environmental...
changes etc. This paper focuses on the fault recovery problem in case of a single unpredictable node fault in a partition. It is assumed that when an active node $f$ of a partition $C_i$ fails abruptly, its parent if exists and the children in $C_i$ can detect it. A fast fault tolerant algorithm by which all children and the parent

![Fault detection](image1)

![Disjoint subpartition](image2)

![Fault Recovery](image3)

Fig. 6. Fault Detection and Recovery Example

of the faulty node in the partition after detecting the faulty node rearrange the partition quickly to make the partition connected the full coverage. If the faulty node is the leader node, its children with the minimum node-id becomes the new leader otherwise the parent node becomes the leader and the fault recovery procedure is initiated by the new leader.

The formal description of the algorithm is given below.

**Algorithm 2**: Distributed fault recovery algorithm

**Input**: faulty node-id : $f$, partition: $C_i$, parent & children of $f$: $S_i$

**Output**: Recovered partition $C_i$

If $f$ is a leader node then $leader_{temp}$ ← Minimum ID child of $f$, otherwise $leader_{temp}$ ← parent of $f$;

Make a block list $B_{S_i}$ covered by ($f \cup S_i$) \ $leader_{temp}$ with status ← 0;

include $leader_{temp}$ in $temp$[ ];

for each node-$i$ ∈ $temp$[ ]

\[\]
\[\text{find neighbors in } S_i \text{ or from uncovered block. If none, select the free neighbor with maximum } D;\]
\[\text{Include neighbors in ‘Selectlist’;}\]
\[\text{if node-$i$ not } leader_{temp} \text{ then }\]
\[\text{Send ‘Selectlist’ message to } leader_{temp};\]
\[\text{else }\]
\[\text{if node-$i$ receives ‘Selectlist’ message from all nodes-}j \in temp[|, Include nodes in } temp[|;\]
\[\text{Update } B_{S_i}[ ];\]
\[\text{if } B_{S_i}[] == \phi \text{ then }\]
\[\text{Send ‘FaultRecovered’ message with } temp[| \text{ to all } j \in C_i \text{ and terminate;}\]
\[\text{if } temp[| == \phi \text{ then }\]
\[\text{Broadcast ‘RecoveryFailed’ and terminate;}\]
\[\text{else }\]
\[\text{send } temp[| \text{ and } B_{S_i}[ ] \text{ to all node in } temp[|;\]

**Example 2.** In Fig. 6 say, the red node is faulty. It will be detected by all its children and parent (colored by blue). Now the blue nodes execute the distributed fault recovery algorithm 2. After the fault, the partition is broken into three disjoint components as shown in Fig. 6(b). The purple node from the faulty
node’s block is chosen next for maintaining the coverage and it is also connected to at least one node from the disjoint components. Therefore, the connectivity is preserved. Now the new partition including the purple node, is ready for monitoring the area.

5 Simulation Results and Discussion

For the simulation studies, we have used network simulator NS 2.34 to evaluate the performance of our proposed algorithm. We have compared our results with [19] that shows significant improvement on the number of rounds for generating the partitions, the network diameter and the number of transmitted messages per node during the procedure. The sensor nodes are deployed over a grid $P$ which is divided into a number of blocks ($2 \times 2$), ($3 \times 3$) to ($7 \times 7$) respectively. Fig.7 shows the variation of the average number of rounds to complete partitioning with the grid size. Obviously, the number of rounds increases with the number of blocks. However, compared to the DCSP algorithm proposed in [19], the present method completes in significantly less number of rounds. Therefore, during initialization, the proposed method will converge faster to achieve the connected covers of the nodes.

In Fig.8 it is shown that the proposed algorithm results significant improvement in terms of network diameter over the DCSP algorithm [19]. In a network with large diameter, the number of steps to route a message from a source node to a destination node will require more delay and more communications between intermediate nodes. Therefore, the low diameter network topologies are preferred for a partition that can aggregate the data using less number of hops, i.e., with less delay and less number of broadcasts.

Also, Fig.9 shows the significant improvement in average number of transmitted messages per node in computing the connected set covers. Since the procedure terminates faster using fewer rounds of computation, the total number of messages exchanged per node is also less here. Finally, Fig.10 shows how the fault recovery algorithm enhances the network lifetime in presence of
faults. Though the proposed fault model includes any unpredicted node faults, in the simulation, only node faults due to energy exhaustion has been taken into account. Simulation results show almost 50% enhancement in network lifetime.

6 Conclusion
In this paper, we have focused on the connected set cover partitioning problem. A self-organized fast distributed algorithm is proposed for finding maximum number of connected cover partitions. Also, distributed fault recovery technique is developed to rearrange connected set covers in presence of unpredictable node faults to satisfy both connectivity and coverage criteria. Minimization of network diameter of the partition and significant improvements in terms of computation rounds and message overhead are also achieved by our proposed method. In summary, the proposed connected set cover partitioning technique along with the localized fault recovery scheme opens up new avenues for setting up self organized wireless sensor networks with enhanced lifetime.

References
1. Li, M., Wan, P.J., Frieder, O.: Coverage in Wireless Ad-hoc Sensor Networks. IEEE Transactions on Computers 52(6) (June) 753–763
2. Meguerdichian, S., Koushanfar, F., Potkonjak, M., Srivastava, M.: Coverage Problems in Wireless Ad-hoc Sensor Networks. In: Proceedings Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies. IEEE INFOCOM 2001. Volume 3. (2001) 1380–1387
3. Wang, D., Xie, B., Agrawal, D.: Coverage and Lifetime Optimization of Wireless Sensor Networks with Gaussian Distribution. IEEE Transactions on Mobile Computing 7(12) (December 2008) 1444–1458
4. Jiang, J., Dou, W.: A Coverage-Preserving Density Control Algorithm for Wireless Sensor Networks. In Nikolaidis, I., Barbeau, M., Kranakis, E., eds.: Ad-Hoc, Mobile, and Wireless Networks. Volume 3158 of Lecture Notes in Computer Science. Springer Berlin Heidelberg (2004) 42–55
5. Huang, Chi-Fu and Tseng, Yu-Chee: The Coverage Problem in a Wireless Sensor Network. In: Proceedings of the 2nd ACM international conference on Wireless
Sensor Networks and Applications. WSNA ’03, New York, NY, USA, ACM (2003) 115–121

6. Slijepcevic, S., Potkonjak, M.: Power Efficient Organization of Wireless Sensor Networks. In: IEEE International Conference on Communications, ICC 2001. Volume 2. (2001) 472–476

7. Gupta, H., Zhou, Z., Das, S., Gu, Q.: Connected sensor cover: self-organization of sensor networks for efficient query execution. Networking, IEEE/ACM Transactions on 14(1) (2006) 55–67

8. Zhou, Z., Das, S., Gupta, H.: Connected K-Coverage Problem in Sensor Networks. In: Proceedings 13th International Conference on Computer Communications and Networks, ICCCN 2004. (October, 2004) 373–378

9. Zhang, Z., Gao, X., Wu, W.: Algorithms for Connected Set Cover Problem and Fault-Tolerant Connected Set Cover Problem. Theoretical Computer Science 410(810) (2009) 812 – 817

10. Peng, L., Liu, K., Liu, G.: Research on Wireless Sensor Networks Fault-Tolerant Coverage Algorithm Based on Particle Swarm Optimization. In: IET International Conference on Wireless Sensor Network, IET-WSN. (November, 2010) 286–290

11. Chow-Sing Lin and Chih-Chung Chen and An-Chi Chen: Partitioning Sensors by Node Coverage Grouping in Wireless Sensor Networks. In: International Symposium on Parallel and Distributed Processing with Applications (ISPA). (September, 2010) 306–312

12. Liu, C., Wu, K., Xiao, Y., Sun, B.: Random Coverage with Guaranteed Connectivity: Joint Scheduling for Wireless Sensor Networks. IEEE Transactions on Parallel and Distributed Systems 17(6) (June 2006) 562–575

13. Tian, Di and Georganas, Nicolas D.: A Coverage-Preserving Node Scheduling Scheme for Large Wireless Sensor Networks. In: Proceedings of the 1st ACM International Workshop on Wireless sensor networks and applications, ACM Press (2002) 32–41

14. Wang, X., Xing, G., Zhang, Y., Lu, C., Pless, R., Gill, C.: Integrated Coverage and Connectivity Configuration in Wireless Sensor Networks. In: Proceedings of the 1st international conference on Embedded networked sensor systems. SenSys ’03, New York, NY, USA, ACM (2003) 28–39

15. Gallais, A. and Carle, J. and Simplot-Ryl, D. and Stejmenovic, I.: Localized Sensor Area Coverage with Low Communication Overhead. In: Fourth Annual IEEE International Conference on Pervasive Computing and Communications, PerCom. (March 2006) 10 pp.–337

16. Pervin, N., Layek, D., Das, N.: Localized Algorithm for Connected Set Cover Partitioning in Wireless Sensor Networks. In: 1st International Conference on Parallel Distributed and Grid Computing (PDGC). (October, 2010 2010) 229–234

17. Ke, W.C., Liu, B.H., Tsai, M.J.: The Critical-square-grid Coverage Problem in Wireless Sensor Networks is NP-Complete. Computer Networks 55(9) (2011) 2209–2220

18. Shakkottai, S., Srikan, R., Shroff, N.: Unreliable Sensor Grids: Coverage, Connectivity and Diameter. In: Twenty-Second Annual Joint Conference of the IEEE Computer and Communications. IEEE Societies, INFOCOM. (March-3 April 2003) 1073–1083

19. Dibakar Saha and Nabanita Das: Distributed Area Coverage by Connected Set Cover Partitioning in Wireless Sensor Networks. In: First International Workshop on Sustainable Monitoring through Cyber-Physical Systems (SuMo-CPS), ACM (2013) 17–22