Connected K-Coverage Problem in Sensor Networks

Zongheng Zhou,† Samir Das, Himanshu Gupta
Department of Computer Science, SUNY, Stony Brook, NY 11794
Email: zzhou, samir, hgupta@cs.sunysb.edu

***Abstract*— In overdeployed sensor networks, one approach to conserve energy is to keep only a small subset of sensors active at any instant. In this article, we consider the problem of selecting a minimum size connected K-cover, which is defined as a set of sensors M such that each point in the sensor network is "covered" by at least K different sensors in M, and the communication graph induced by M is connected. For the above optimization problem, we design a centralized approximation algorithm that delivers a near-optimal (within a factor of $O(\lg n)$) solution, and present a distributed version of the algorithm. We also present a communication-efficient localized distributed algorithm which is empirically shown to perform well.**

I. Introduction

Wireless sensor networks are often deployed for passive data-gathering or monitoring in a geographical region. An important issue here is to maintain the fidelity of the sensed data while minimizing energy usage in the network. Energy is spent due to message transmissions among sensor nodes, or due to the sensing activities by the signal processing electronics. Energy can be saved if these activities are used only to the extent absolutely needed, and no more.

In this article, we address the two important characteristics, viz., *coverage* and *connectivity* of a sensor network and design a fault-tolerance scheme for energy conservation. In most deployment scenarios, it is cost-effective to deploy sensors randomly in a redundant fashion ([1, 2]), since the sensor hardware is generally cheap relative to the cost of deployment. In such highly redundant sensor networks, it will be useful to select a "minimally sufficient subset" of sensors to keep active at any given time – thus conserving energy and prolonging the sensor network lifetime. In particular, we can choose to keep active a *minimum K-cover* which is informally defined as a set of sensors such that each point in the given region is "covered" by (i.e., within the sensing radius [3] of) at least K distinct sensors. The set of active nodes must also induce a "connected" communication topology so that they can collectively transmit data to a central node. Here, K is a configurable parameter, and larger value of K could be used when sensors have a higher chance for failure, or when sensor data can be very noisy. In this paper, we design various algorithms to select a minimum number of sensor nodes that form a connected communication graph and also provide K-coverage.

The rest of the paper is organized as follows. In Section II, we discuss previous work done in the context of connectivity and coverage in sensor networks. In the following section, we present various algorithms for the connected K-coverage problem. Simulation results that compare the performance of various algorithms is presented in Section IV. We end with concluding remarks in Section V.

†Partially supported by NSF Grant ANI-0308631.

II. Related Work

Recently, there has been a lot of research done to address the coverage problem in sensor networks. In particular, the authors in [4] design a centralized heuristic to select mutually exclusive sensor covers that independently cover the network region. In [3], the authors investigate linear programming techniques to optimally place a set of sensors on a sensor field (three dimensional grid) for a complete coverage of the field. Meguerdichian et al. ([5]) consider a slightly different definition of coverage and address the problem of finding maximal paths of lowest and highest observabilities in a sensor network. Connectivity is also a fundamental issue in wireless ad hoc environment, and many schemes have been proposed to address the issue of energy efficiency while maintaining connectivity in the network topology ([6], [7], [8]).

Researchers have also considered connectivity and coverage in an integrated platform. In [9], the authors consider an unreliable sensor network, and derive necessary and sufficient conditions for the coverage of the region and connectivity of the network with high probability. The PEAS protocol [1] considers a probing technique that maintains only a necessary set of sensors in working mode to ensure converage and connectivity with high probability under certain assumptions. In our prior work [10], we designed a greedy approximation algorithm that delivers a connected 1-cover within a $O(\lg n)$ factor of the optimal solution. Wang et al. [2] is the first and only work to address the connected K-coverage problem. They present a localized heuristic for the problem, but their heuristic does not provide a guarantee of the solution size returned. In addition, they assume that any two sensors with intersecting sensing disks can communicate directly with each other. In this article, we generalize our prior work in [10] to the connected K-coverage problem and design a greedy algorithm that returns a solution within $O(\lg n)$ factor of the optimal. We also design a localized distributed algorithm based on node priorities that is shown to perform well in practice.

III. Connected K-Cover

In this section, we address the problem of constructing a connected sensor K-cover, wherein each point in the query region is covered by at least K distinct sensors. Each sensor I is stationary and is associated with a fixed *sensing region* which is assumed to be a disk of radius S_I. Also, each sensor can directly communicate with some of the sensors around it based on a fixed communication graph (as defined below).

Definition 1: (Sensor Covering a Point) A sensor I is said to cover a point p, if the distance $d(p, I)$ between p and I is less than S_I, the sensing radius of the sensor. □

Definition 2: (Communication Graph; Communication Distance) Given a sensor network consisting of a set of sensors $\mathcal{I}$, the *communication graph* for the sensor network is the *undirected* graph CG with $\mathcal{I}$ as the set of vertices and an edge between any two sensors if they can communicate directly with each other. The *communication subgraph induced* by a set of sensors $\mathcal{M}$ is the subgraph of CG involving only the vertices/sensors in $\mathcal{M}$. □

Definition 3: (Connected K-Cover) Consider a sensor network consisting of a set $\mathcal{I}$ of n sensors and a query region R_Q. A set of sensors $M \subseteq \mathcal{I}$ is said to be a *connected K-cover* for the query region if the following two conditions hold:

1) each point p in R_Q is covered (as defined above) by at least K distinct sensors in M,
2) the communication graph induced by M is connected.

A set of sensors that satisfies only the first condition is called a *sensor K-cover* for the query region R_Q. □

Connected K-Coverage Problem: Given a sensor network and a query over the network, the connected K-coverage problem is to find a connected K-cover of smallest size. The connected K-coverage problem is NP-hard as it is a generalization of the connected 1-coverage problem which is already known to be NP-hard [10].

A. Greedy Algorithm for Connected K-Cover

In this section, we present the Greedy Algorithm for the connected sensor K-coverage problem. The Greedy Algorithm is a generalization of the centralized approximation algorithm in [10] for the connected 1-coverage problem. Informally, the Greedy Algorithm maintains a set M of selected sensors and at each stage, adds a candidate path of sensors with most "K-Benefit" (defined later). We show that the greedy algorithm based on the above concept of adding a candidate with maximum K-Benefit will deliver a solution that is within $O(r \log Kn)$ factor of the optimal sensor K-cover (not necessarily connected), where r is the *link radius* (defined later) and n is the size of the sensor network.

Definition 4: (Candidate Sensor; Candidate Path) Let M be a set of already selected sensors. A sensor c is called a *candidate sensor* if $c \notin M$ and there is a sensor $m \in M$ such that $d(c, m) < S_c + S_m$, i.e., the sensing region of c intersects with the sensing region of some sensor m in M.

Given a set M of already selected sensors, a *candidate path* is a sequence/path of sensors $<p_0, p_1, \ldots, p_l>$ such that p_0 is a candidate sensor, $p_l \in M$, $p_i \notin M$ for $i < l$, and the sequence of sensors forms a communication path in the communication graph of the sensor network. □

Definition 5: (Subelement; Valid Subelement) A *subelement* is a set of points. Two points belong to same subelement if and only if they are covered by the same set of sensors. If a subelement intersects with a given query region, then it is called a *valid subelement*. □

Definition 6: (K-Value of a Sensor Set) Given a sensor network and a query region, the K-Value of a set of sensors S is denoted as $V(S, K)$ and is defined as the sum of the total number of times (bounded by K) each valid subelement is covered by the sensors in set S. More formally, the K-Value of a set S of sensors, $V(S, K)$, is computed as:

$$V(S, K) = \sum_{e \in E} (max(K, \sum_{s \in S} (\delta(e, s))),$$

where E is the set of valid subelements, and $\delta(e, s)$ is 1 if the subelement e is covered by the sensor s, and else 0. □

Definition 7: (K-Benefit of Candidate Path) Consider a candidate path P and set of already selected sensors M. The K-Benefit of P with respect to M is defined as: $(V(M \cup P, K) - V(M, K))/(|M \cup P| - |M|)$. □

Algorithm 1: **(Greedy Algorithm)**
Input: A set of sensors $\mathcal{I}$ and a query region R_Q.
Output: A connected K-cover M.
BEGIN
  Let M be the set of sensors already selected
  by the algorithm at a given stage.
  $M := \{I\}$, where I is any sensor whose
  sensing region intersects with R_Q.
  while (R_Q is not K-covered by M)
    Find the candidate path $\widehat{P}$ that has the
    maximum K-Benefit with respect to M
    $M = M \cup \widehat{P}$;
  end while
  RETURN M;
END ◇

Definition 8: (Communication Distance; Link Radius) The *communication distance* between a pair of sensors is the distance (minimum number of hops) between the two sensors in the communication graph.

The *link radius* of a sensor network is the maximum communication distance between any two sensors whose sensing regions intersect. □

Theorem 1: The above described Greedy Algorithm returns a connected K-cover set of size at most $2r(\log Kn)|OPT|$, where $|OPT|$ is the size of the optimal sensor K-cover (not necessarily connected), n is the size of the sensor network, and r is the link radius of the sensor network.

Proof: Let M be the set of sensors already selected by the Greedy Algorithm at any stage. A valid subelement is called *active* and is said to contain t *active copies* if it is covered by $K - t$ ($t > 0$) distinct sensors in M. Whenever a candidate path $\widehat{P}$ is added to M, we spread the charge $(|\widehat{P}| - 1)$ uniformly on each of the active copies newly covered by $\widehat{P}$.

Let OPT be an optimal sensor cover for the given query. Let us consider a sensor $I \in OPT$ and try to compute the maximum charge accumulated by its sensing region R_I during the entire course of the algorithm. Let e_j be the number of active copies in R_I after the j^{th} iteration of the algorithm. Note that e_0 is K times the number of valid subelements inside S_I, and the number of active copies of R_I covered during the j^{th} iteration is $e_{j-1} - e_j$.

Let $\widehat{P}_j$ be the candidate path selected for addition at the j^{th} iteration, and $E(\widehat{P}_j)$ be the number of active copies newly covered[1] by $\widehat{P}_j$. The total charge accumulated by R_I is $T = \sum_{j=1}^{j=l}(e_{j-1} - e_j) * (|\widehat{P}_j| - 1)/E(\widehat{P}_j)$, assuming the Greedy Algorithm runs for l iterations. Now, $E(\widehat{P}_1)/(|\widehat{P}_j| - 1) \geq (e_0 - e_1)/(K(r-1))$ and $E(\widehat{P}_j)/(|\widehat{P}_j| - 1) \geq e_{j-1}/(K(r-1))$ for $j \geq 2$, as the sensor I along with a candidate path of K-benefit at least $e_{j-1}/(K(r-1))$ is also eligible for selection. Thus, $T/(K(r-1)) \leq 1 + \sum_{j=2}^{j=l}(e_{j-1} - e_j)/e_{j-1}$. Using some algebra [11], we get $T \leq K(r-1)(1 + \log e_0)$, where $e_0 \leq Kn^2$ ([10]). As the total charge accumulated on all the sensors in OPT is at least $K|M|$, we get $|M| \leq 2r(\log Kn)|OPT|$. ■

Distributed Greedy Algorithm. Now, we briefly describe the distributed version of the above described Greedy Algorithm. The distributed version is along the similar lines as the Distributed Approximation algorithm in [10] for connected 1-cover. As in [10], we make certain optimizations to reduce the overall communication overhead.

Initially, a random sensor whose sensing region intersects with the query region is chosen to be in M. Then, each stage of the algorithm consists of the following distributed phases:

- **Candidate Path Search(CPS):** The most recently added candidate sensor C broadcasts a Candidate Path Search (CPS) message to all sensors within $2r$-hops, along with the information about the most recently added path.
- **Candidate Path Response(CPR):** Any unselected sensor I that receives a CPS message checks whether it is a *new* candidate sensor, i.e., if its sensing region intersects with the sensing region of any sensor already selected in the most recently added candidate path $\widehat{P}$.[2] If so, the sensor I sends back a CPR message to the sensor C (original sender of the CPS message) along with the path of sensors connecting I to C, which forms the candidate path.
- **Selection of Best Candidate Path/Sensor:** At receiving each CPR message, the node C adds the received candidate path P to the set of candidate paths CP being maintained. After gathering all the CPR messages, the sensor C finds the candidate path P_{new} that has the most K-benefit. Let the associated candidate sensor be C_{new}. The candidate path P_{new} is added to M, and the new M along with the new set of candidate paths is sent to C_{new}.
- **Repeat:** Repeat the above three phases until the query region R_Q is K-covered by M.[3]

[1]Note that $E(\widehat{P}_j) = V(\widehat{P}_j \cup M, K) - V(M, K)$.

[2]If a node's sensing region intersects with previously added nodes in M, then it would have been added as a candidate sensor at an earlier stage.

[3]Since, each candidate path added includes some new sensors, the algorithm is guaranteed to terminate.

B. Distributed Priority Algorithm

While the Distributed Greedy algorithm is based on an approximation algorithm with a performance guarantee, it needs to carry around a central state (the intermediate solution M) via messages. This makes the algorithm prone to message losses and also the size of messages potentially large. In this section, we present an alternate localized approach that uses small and constant size messages. We refer to the alternate approach as the Distributed Priority algorithm, as it uses a notion of node priorities. In the Distributed Priority algorithm, each sensor uses only local neighborhood information, thus more fault tolerant and keeping the number and size of messages small. However, there is no guarantee on the size of the connected K-cover delivered.

Basic Idea. The distributed priority algorithm is based on the following idea. A node I is not needed for connectivity if all pairs of immediate neighbors of I have an alternate communication path not involving I. And, a node is not needed for K-coverage if each point in its sensing region is covered by at least K other sensors. Thus, if a node I satisfies both the above conditions, its deletion would preserve connectivity and K-coverage of the sensor network. Hence, such a node I is marked `deleted`. Node priorities are used to prevent cyclicity of conditions. In particular, only lower priority and non-`deleted` nodes are used for satisfying conditions of a given node. To determine alternate paths between pairs of neighbors without incurring unreasonable communication cost, we limit ourselves to *t-hop neighborhood search*, i.e., among neighbors that are within a communication distance of t. We choose $t = 2$ for our simulations.

Algorithm Description. The distributed Priority algorithm works as follows. First, each sensor node assigns a random number as a priority to itself.[4] Then, each node gathers $l = max(t, r)$-hop information (including node priorities), where t is the constant as described above and r is the link radius of the sensor network. Let $P(I)$ be the priority of a node I. Each node I periodically tests for the following set of conditions and marks itself `deleted` if they are satisfied.[5]

C1: In the communication subgraph induced over the set of non-`deleted` nodes in the l-hop neighborhood of I, each pair of neighbors of I is connected by a communication path with *intermediate* nodes having priorities lower than $P(I)$. This condition ensures that deletion of I will preserve the connectivity of the communication subgraph induced by the set of nodes not marked `deleted`. In [12], Wu et al. use a similar condition for constructing a connected dominating set.

C2: There is a set of sensor nodes H within l-hop neighborhood, such that every sensor node in the set H has a priority lower than $P(I)$ and the sensing

[4]Considering more complicated priority functions based on node degree and/or overlapping area did not result in any performance improvement.

[5]The nodes whose sensing region do not overlap with the query region do not participate in this process.

region of I is K-covered by the sensor nodes in H. Note that H may contain a sensor node that is marked `deleted`.

The `deleted` marking of a node is permanent and at the end, some of the nodes may be left unmarked. If the communication graph of the initial sensor network is connected, the Distributed Priority algorithm guarantees that the set of nodes that have not been marked `deleted` forms a connected K-cover at any intermediate stage of the algorithm.

Message Communications. Initially, each node needs to gather $l = \max(t, r)$-hop neighborhood information. If n is the total number of sensor nodes in the network, l-hop neighborhood information can be gathered using ln messages in l phases as follows. In the first phase, each node broadcasts its neighborhood information to its neighbors. In each of the remaining $(l-1)$ phases, each node collects information transmitted by all its immediate neighbors, and broadcast all the collected information to all its immediate neighbors. At the end of l phases and ln total messages, it is easy to see that each node would have the complete l-hop neighborhood information.

After the initial accumulation of l-hop neighborhood information, whenever a node is marked `deleted`, it informs its immediate neighbors of its `deleted` status, so that they could retest their C1 condition. This can be done using one message transmission for each node that is marked `deleted`. Note that an unsatisfied C1 condition of a node can become `true` only by `deleted` markings of some of its communication neighbors. In addition, it can be shown ([12]) that once the C1 condition is satisfied for a node I, the deletion of I will always preserve connectivity in the communication subgraph induced by the non-`deleted` nodes, even if other nodes get marked `deleted`. Since, message losses only result in some nodes not getting marked `deleted`, the Distributed Priority algorithm always results in a connected K-cover even in case of loss of messages.

Theorem 2: The Distributed Priority algorithm correctly computes a connected sensor K-cover. □

The Distributed Priority algorithm incurs low communication overhead since each sensor makes a decision based only on local information. In addition, the Distributed Priority algorithm generates a connected sensor K-cover of size close to that delivered by the centralized approximation Greedy Algorithm, as shown in the simulation results in Section IV.

C. K-Connectivity and K-Coverage

In the previous section, we addressed the fault-tolerance of connected sensor cover from the perspective of coverage. In this section, we consider fault-tolerance in connectivity as well as coverage. In particular, we address the problem of K-Connected K-Cover, wherein each point in the query region is covered by at least K sensors and there are at least K node-disjoint paths between any pair of sensors in the selected set of sensors.

Below, we show that in a sensor network wherein each sensor has the uniform sensing and transmission radius of S and T respectively where $T \geq 2S$, K-coverage implies K-connectivity. Although this conclusion is also proved in [2] for convex query region, we show using a much simpler proof the implication actually holds in any closed area. Below, we formally define the concept of K-connected set.

Definition 9: (K-connected set) The set of sensors M is a K-connected set if there exists K node-disjoint communication paths between every pairs of nodes in M in the communication graph induced by M. Note that if M is a K-connected set, then to disconnect the communication graph induced by M, at least K nodes must be deleted from M. □

Definition 10: (Coverage graph) Let M be a set of sensors in a sensor network. The coverage graph of M is a graph over the set of nodes M where an edge exists between two nodes if their sensing regions intersect. □

Lemma 1: In a sensor network with uniform sensing and transmission radius of S and T respectively, where $T \geq 2S$, the coverage graph of a set of sensors M is a subgraph of the communication graph of M. □

Lemma 2: Given a sensor network with uniform sensing and transmission radius of S and T, and a closed query region R_Q. If a set of nodes M is a sensor 1-cover, then M's coverage graph is connected. □

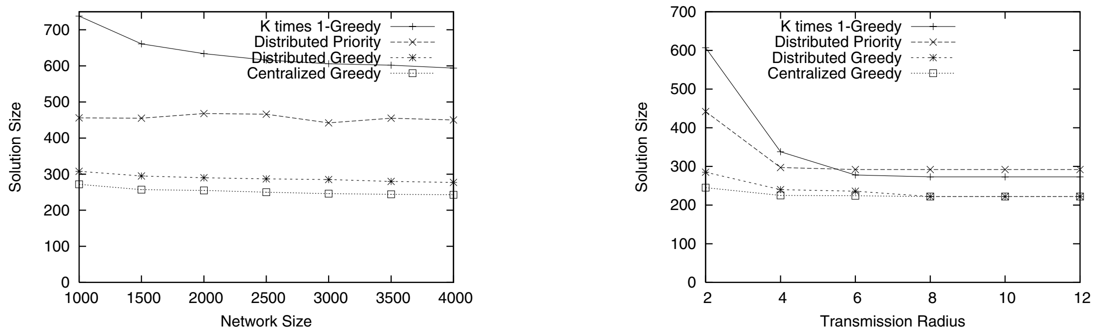
Theorem 3: Given a sensor network with uniform sensing and transmission radius of S and T such that $T \geq 2S$, and a closed query region R_Q. If a set of nodes M is a sensor K-cover, then M is also a K-connected set.

Proof: Since M is a K-coverage set, removing less than K nodes maintains the 1-coverage of M. By Lemma 2 and lemma 1, that implies that if less that K nodes are deleted from M, the coverage graph of M is still connected and hence, the communication graph of M remains connected. Thus, M is also a K-connected set. ■

IV. Performance Evaluation

In this section, we present the result of our simulations that we ran to compare the various algorithms described in the previous sections. We ran our algorithms on randomly generated sensor networks wherein a certain number of sensor nodes are placed randomly in an area of 40×40 unit square. We assume that the query region is the entire sensor network region. Each sensor has a uniform sensing radius of 4 units. We vary the network size n from 1000 (which is barely enough to generate a connected 8-cover) to 4000 (which provides substantial redundancy) randomly placed sensors. Also, we vary the transmission radius t of sensor nodes from 2 units to 12 units. Below, we present the comparison of various algorithms presented in this article for connected K-cover, viz., Centralized Greedy, Distributed Greedy, and Distributed Priority. In addition, we also plot the performance of the naive $K\ Times\ 1-Greedy$ algorithm which works by executing the centralized greedy algorithm of [10] for connected 1-cover K times.

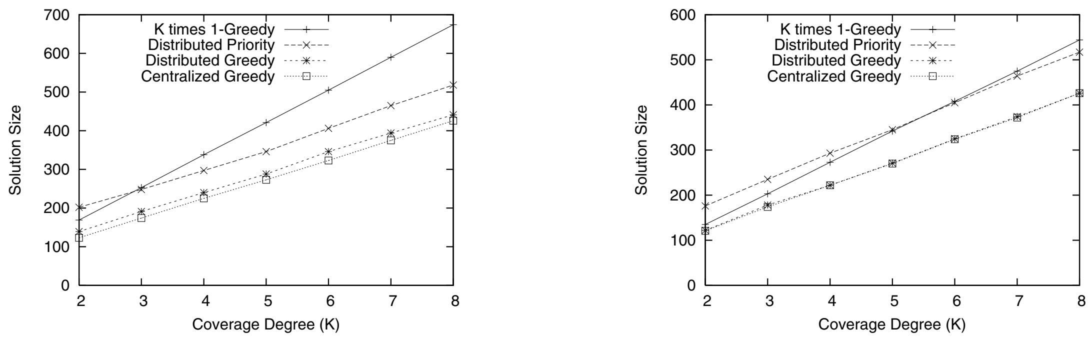
Calculation of the link radius (r). We use the same methodology as in [10] to compute the link radius r of a sensor network. In particular, we define dense networks as networks



(a) Transmission radius = 4 units

(b) Sensor network size = 3000 sensors

Fig. 1. Size of connected 4-cover delivered by various algorithms for various network sizes and transmission radii.



(a) Transmission radius = 4 units

(b) Transmission radius = 8 units

Fig. 2. Size of connected K-cover delivered by various algorithms for various values of K. Here, the network size is 3000 sensors.

with more than $4s/t$ sensors within a distance $2s$. For a 40×40 area, a dense network should have at least $(80/t)^2$ sensors. Thus, for dense networks, we use $r = (2s/t + 1)$. For a non-dense network, we simply use a proportionate density factor to "inflate" the value of r, i.e., for a network with n sensors where $n < (80/t)^2$, we use $r = (2s/t + 1) * ((80/t)^2/n)$.

Solution Sizes. We plot the size of connected K-cover delivered by various algorithms in Figure 1 and Figure 2 for various values of sensor network sizes, transmission radii, and coverage degree K. Figure 1 plots the size of the connected 4-cover delivered by the algorithms. Note that the size of the solution selected by the algorithms is much less than the network size. We see that the solution size does not decrease much with increase in n implying that the solution obtained for $n = 1000$ itself is of size quite close to the optimum size. Also, we observed that random sensor network of size even less than 1000 was not redundant enough to yield a connected 4-cover.

From Figures 1 and 2, we can see that the solution size delivered by the Distributed Greedy algorithm is very close to that delivered by the Centralized Greedy algorithm. This observation validates the accuracy of our computation of the link radius and also shows that the optimizations made in the Distributed Greedy does not compromise much on the solution size. Moreover, we can observe that the solution size returned by the Distributed Greedy is noticeably smaller (better) than that returned by the Distributed Priority algorithm, which is expected since the Distributed Priority algorithm is a localized algorithm. However, the "gap" between the Distributed Priority and Distributed Greedy algorithm doesn't increase with the increase in any of the parameter values (coverage degree K, sensor netowrk size n, and transmission radius t). Finally, as expected, we see that the solution returned by the K Times 1-Greedy Algorithm is significantly worse than the other algorithms except for very low K and high transmission radius, and the performance of K Times 1-Greedy Algorithm worsens with the increase in K.

Communication Cost. Let n be the size of the sensor network. The communication cost incurred during the initial phase of Distributed Priority is $max(2, r) * n$, the cost of gathering the $max(t, r)$-hop neighborhood information for each sensor node. As noted before, we chose $t = 2$ for our simulations. The later phases of the Distributed Algorithm incurs communication cost of the order of $(n - m)$ messages, where m is the size of the solution returned. For the Distributed Greedy algorithm the total communication cost is at most $n * b$ where b is the number of sensors in the r-neighborhood of a sensor node. Figure 3 shows the total communication cost incurred by the Distributed Priority and Distributed Greedy algorithms. We observe that the cost incurred by the Distributed Priority

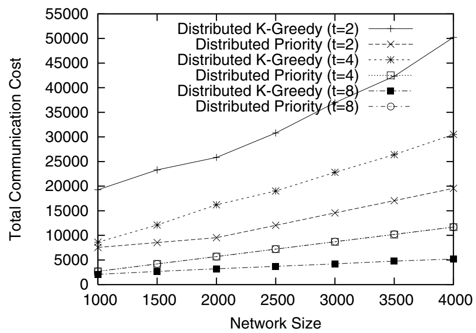


Fig. 3. Message cost incurred by various distributed algorithms for various network sizes. Here, $K = 4$.

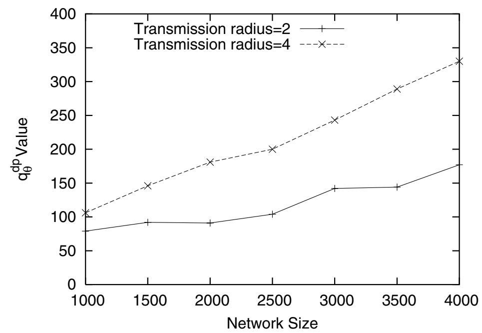


Fig. 4. The threshold value q_θ^{dp}.

algorithm is much less than that incurred by the Distributed Greedy algorithm, except when the transmission radius $t \geq 2s = 8$ where the link radius r is 1. For $t \geq 8$, the total communication cost incurred by both the algorithms is very low and in fact, Distributed Greedy incurs even less communication cost than Distributed Priority.

Summary. From the above observations, we can conclude that the Distributed Greedy algorithm is a very efficient distributed implementation of the Greedy approximation algorithm, and that for transmission radius larger than twice the sensing radius, the Distributed Greedy algorithm outperforms the Distributed Priority algorithm in terms of the solution size as well as the communication cost incurred. Thus, for $t \geq 2s$, the Distributed Greedy algorithm is the best choice for computing connected K-cover.

For $t < 2s$, the Distributed Priority algorithm incurs substantially less communication cost than the Distributed Greedy, but delivers a slightly larger connected K-cover. Let D be the number of messages incurred and m be the size of the connected K-cover returned by the Distributed Greedy algorithm, and let D_{dp} be the number of messages incurred and m_{dp} be the size of the connected sensor cover returned by the Distributed Priority algorithm. Figure 4 plots the threshold value q_θ^{dp}, where $q_\theta^{dp} = \frac{D - D_{dp}}{2(m_{dp} - m)}$. If the given spatial query is run q times, then the threshold value q_θ^{dp} is such that for $q > q_\theta^{dp}$ the overall communication cost by using the Distributed Priority is lower than the communication cost using the Distributed Greedy algorithm. From the Figure 4, we observe that q_θ^{dp} is very high (100 to 300). Thus, for applications that do not run a huge number (less than a couple of hundreds) of queries on a given query region, the Distributed Priority algorithm is a better choice for computing the connected K-cover for the given query region.

V. Conclusions

In this article, we have addressed the connected K-coverage problem of selecting a minimum number of sensors that are connected and also cover each point in a given query region with at least K distinct sensors. The idea is to keep *only* these set of sensors active to provide the necessary coverage and connectivity, resulting in a fault-tolerant energy conservation technique. We have designed Centralized Greedy algorithm that provably returns a near-optimal solution, and a communication-efficient distributed version of the Greedy algorithm. In addition, we designed the Distributed Priority which is a localized algorithm, but delivers a slightly larger size solution. Through extensive simulations we show that the designed distributed algorithms indeed deliver a near-optimal connected K-cover. Further analysis shows that, the Distributed Priority algorithm is more efficient in applications where the query is run for less than a few hundred times. For longer running queries, the Distributed Greedy algorithm is more efficient.

References

[1] F. Ye, G. Zhong, S. Lu, and L. Zhang, "PEAS: A robust energy conserving protocol for long-lived sensor networks," in *Proceedings of the International Conference on Distributed Computing Systems*, 2003.

[2] X. Wang, G. Xing, Y. Zhang, C. Lu, R. Pless, and C. Gill, "Integrated coverage and connectivity configuration in wireless sensor networks," in *Proceedings of the ACM SenSys*, 2003.

[3] K. Charkrabarty, S. Iyengar, H. Qi, and E. Cho, "Grid coverage for surveillance and target location in distributed sensor networks," *IEEE Transaction on Computers*, 2002.

[4] S. Slijepcevic and M. Potkonjak, "Power efficient organization of wireless sensor networks," in *Proceedings of the International Conference on Communication (ICC)*, 2001.

[5] S. Meguerdichian, F. Koushanfar, G. Qu, and M. Potkonjak, "Exposure in wireless ad-hoc sensor networks," in *Proceedings of the International Conference on Mobile Computing and Networking (MobiCom)*, 2001.

[6] B. Chen, K. Jamieson, H. Balakrishnan, and R. Morris, "Span: An energy-efficient coordination algorithm for topology maintenance in ad hoc wireless networks," in *Proceedings of the International Conference on Mobile Computing and Networking (MobiCom)*, 2001.

[7] J. E. Wieselther, G. D. Nguyen, and A. Ephremides, "On the construction of energy-efficient broadcast and multicast trees in wireless networks," in *Proceedings of the IEEE INFOCOM*, 2000.

[8] P. Wan, G. Calinescu, X. Li, and O. Frieder, "Minimum-energy broadcast routing in static ad hoc wireless networks," in *Proceedings of the IEEE INFOCOM*, 2001.

[9] S. Shakkottai, R. Srikant, and N. Shroff, "Unreliable sensor grids: Coverage, connectivity and diameter," in *Proceedings of the IEEE INFOCOM*, 2003.

[10] H. Gupta, S. Das, and Q. Gu, "Connected sensor cover: Self-organization of sensor networks for efficient query execution," in *Proceedings of the International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, 2003.

[11] T. Cormen, C. Lieserson, R. Rivest, and C. Stein, *Introduction to Algorithms*. McGraw Hill, 2001.

[12] J. Wu and F. Dai, "Broadcasting in ad hoc networks based on self-pruning," in *Proceedings of the IEEE INFOCOM*, 2003.