Deploying Sensor Networks with Guaranteed Fault Tolerance

Jonathan L. Bredin,* Erik D. Demaine,† MohammadTaghi Hajiaghayi,‡ Daniela Rus†

***Abstract*—We consider the problem of deploying or repairing a sensor network to guarantee a specified level of multi-path connectivity (k-connectivity) between all nodes. Such a guarantee simultaneously provides fault tolerance against node failures and high overall network capacity (by the max-flow min-cut theorem). We design and analyze the first algorithms that place an almost-minimum number of additional sensors to augment an existing network into a k-connected network, for any desired parameter k. Our algorithms have provable guarantees on the quality of the solution. Specifically, we prove that the number of additional sensors is within a constant factor of the absolute minimum, for any fixed k. We have implemented greedy and distributed versions of this algorithm, and demonstrate in simulation that they produce high-quality placements for the additional sensors.**

I. Introduction

SENSOR-NETWORK applications owe much of their popularity to broad and rapid deployment, frequently into hazardous environments. We use a robotic helicopter to deploy sensors to monitor outdoor environments and provide network connectivity for emergency response scenarios [1], [2]. Such rapid deployment, especially under extreme circumstances, exposes sensors to additional chance of failure and placement errors. Sensors may not be placed in exactly their desired locations because of wind or inaccurate localization. Sensors may fail from impact of deployment, fire or extreme heat, animal or vehicular accidents, malicious activity, or simply from extended use. These failures may occur upon deployment or over time after deployment: extensive operation may drain some of the nodes' power, and external factors may physically damage part of the nodes. Additionally, hazards may change devices' positions over time, possibly disconnecting the network. If any of these initial deployment errors, sensor failures, or change in sensor positions cause the network to be disconnected or lack other desired properties, we need to deploy additional sensors to repair the network.

In an example application, a network of cameras monitors the safety of a building compound. Each sensor does local image analysis to detect events such as motion within its field of view. Upon such events, sensors send images to a base station for more complex analysis such as tracking. This application relies on the network's ability to support a given amount of information flow. The application also illustrates that not all the nodes in the network require the same amount of communication. For example, the nodes along the trajectory of the tracked object will transmit more images and thus use more power to communicate. This means that their communication ability will be diminished and they may become depleted of power. In such a case, the network will have to be extended with new nodes to sustain the desired information flow.

Given the dynamic environment, it is desirable to have procedures to establish network properties, such as connectedness, in the event that multiple devices fail. We are interested in developing an algorithm that can run regularly in the background, to suggest repairs to a deployment mechanism once connectivity properties disappear. Upon the detection of network failures, our algorithm computes the locations where an approximately minimum number of additional nodes need to be deployed in the network using a ground or flying robot. This results in a goal-directed approach for automated maintenance and repair of a network which optimizes the use of the powerful mobile node (e.g., the robot helicopter) tasked to do this operation by deploying additional sensors.

More specifically, to support both fault tolerance and capacity, we focus on the vertex-connectivity of the network. The k-connectivity property has been studied extensively before in the context of wireless networks; see [3], [4] and their many citations. In the worst case, a k-connected network requires k node failures to disconnect the network. Additionally, k-connectivity ensures a high overall transport capacity of the network, by the max-flow min-cut theorem.

Given a desired value of k, we present and analyze a generic algorithm that determines how to establish k-connectivity by placing additional sensors geographically between existing pairs of sensors. Here, in order for the problem to be well-defined, we assume that the sensors' locations are known (either directly by the sensors via GPS, or by an external agent's survey and measurement) and that sensor communication is defined by the unit-disk graph model, so that we can predict how the communication graph changes as we add sensors. Solving this problem with a minimum number of additional sensors is NP-hard [5]. Our approach is to transform the network repair problem to one of selecting the minimum-weight k-vertex connected subgraph of the complete graph underlying the sensor network and then applying existing graph-theoretic approximation methods. Our proposed algorithm provides a

A preliminary version of this paper appeared at MOBIHOC 2005.

*Department of Mathematics and Computer Science, Colorado College, 14 East Cache la Poudre Street, Colorado Springs, CO 80903, USA, jbredin@coloradocollege.edu

†Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology, 32 Vassar Street, Cambridge, MA 02139, USA, {edemaine,rus}@csail.mit.edu

‡ AT&T Labs — Research, 180 Park Ave., Florham Park, NJ 07932, USA. hajiagha@mit.edu. This work was done while the author was at MIT and Microsoft Research.

Manuscript received March 8, 2007; revised May 28, 2009.

bound on the solution quality that is within a constant factor of the optimal solution, for any fixed k. (More precisely, the approximation ratio is $O(k^4\alpha)$, where $\alpha = O(\lg k)$ is the best approximation ratio for a related problem [6], [7].)

Due to limited communication or computational resources available to nodes, our proposed provable approximation algorithm for optimal k-connectivity repair may be difficult to implement on a physical sensor network. This limitation may not be significant because the algorithm can usually be run occasionally and in the background, so there is little need for a "real time" algorithm. Nonetheless, our characterization of the problem complexity shows how to further approximate the problem. We develop alternative methods for determining a low-cost k-connected graph that are simpler, faster, and able to run in the distributed sensor network. The modularity of our base algorithm allows us to trade computational speed for solution accuracy. In an experimental study, we implement in simulation the use of greedy and distributed algorithms and show that, in practice, the solution quality produced by these fast methods is not far from optimal.

Attaining k-connectivity has recently been studied in the context of power assignment, where instead of adding sensors the goal is to assign the sensors' communication power to ensure k-connectivity and minimize overall power consumption. This problem is also NP-hard. Ramanathan and Rosales-Hain [8] consider the special case of 2-connectivity and provide a centralized spanning-tree heuristic for minimizing the maximum transmit power in this case. Bahramgiri et al. [3] generalize the cone-based local heuristic of Wattenhofer et al. [9], [10] in order to solve the general k-connectivity setting. However, both of these works are heuristics and do not have provable bounds on the solution cost [11]. Lloyd et al. [12] present an 8-approximation algorithm for 2-connectivity, but they do not consider general k-connectivity. Hajiaghayi et al. [11] present a constant-factor approximation algorithm for k-connectivity for any fixed k. Recently, different sets of authors (see e.g. [13], [14], [15], [16]) used the notion of k-connectivity and the results of [3], [11] to deal with the fault-tolerance issues for static and dynamic settings.

Our problem has been considered before only in the special case $k = 1$, where the problem has applications in VLSI design and evolutionary/phylogenetic tree constructions in computational biology. See [17] for a description of these and other applications, and [18] for early work on the theory of general graphs. The best approximation algorithm to our knowledge for our context of unit-disk graphs is a $5/2$-approximation algorithm by Du et al. [19], again only for the case $k = 1$.

We proceed by introducing some definitions, notation, and models we will use for our algorithm. Section III presents an algorithm that takes the subgraph-repair problem as a modular black box to establish k-vertex connectivity in a network by adding new nodes geographically between existing ones. Whereas the algorithm is natural, the analysis in Section IV providing an approximation bound is complicated. We present in Section V practical distributed modifications to our algorithm and implement one on computationally limited platforms upon which the ideal approximation algorithms would be difficult to implement. Section VI supports the heuristics through experiments whose simulations compare the performance of our simplified algorithms with our algorithm using optimal subgraph repair. The experiments show that our methods are superior to deployment according to additional random sampling. Finally, we conclude in Section VII with discussion of improvements to our algorithm relying on tighter analysis to derive a polynomial-time approximation scheme.

The conference version of this paper inspired or has been used in several followup papers, e.g., [20], [21], [22], [4], [23], [24], [25], [26], [27], [28], [29], [30], [31], [32], [33], [34], [35], [36], [37], [38], [39], [40]; see also [41], [42], [43].

II. Preliminaries and Models

In this paper we consider static symmetric multi-hop ad-hoc wireless networks with omni-directional fixed-power transmitters that typically arise in the context of sensor networks. This model is considered by Blough et al. [44], Calinescu et al. [45], Kirousis et al. [46], and others in their works on connectivity. The model has many practical consequences; for example, many existing routing protocols can easily be accommodated by this model, in particular because links are bidirectional. Furthermore, many of the restrictions imposed by this model can be relaxed at the cost of additional communication. We summarize the main features of the model here.

An ad-hoc wireless network consists of a set of mobile devices (e.g., sensors) equipped with radio transceivers. In general, each radio transmitter can be assigned a power setting and an orientation that define the reception area of its transmissions. We assume that all transmitters have a common, fixed maximum power setting, and refer to that as the *communication radius*. We also assume that the transceivers are *omni-directional* in the sense that they transmit and receive in all directions equally. Both of these assumptions are satisfied by most wireless networks. We also assume that the signal propagation is uniform in all directions, so in particular there are no radio-opaque obstacles.

We make the further assumptions that our networks are *static* and that all communication links are *bidirectional* or *symmetric*. In a static network, the devices are stationary. If a device moves, the topology of the network can change in ways out of our control. In the symmetric link model, if a device u can receive transmissions from a device v, then u also has enough maximum power to transit to device v. In practice, most wireless networks experience problems from asymmetries, but the symmetric restriction simplifies routing protocols. We assume that the nodes in our networks tune down their effective ranges to the lowest common range.

Given these assumptions, we can model our wireless network as a *unit-disk graph*, $G = (V, E)$, where each vertex represents a device and is assigned two-dimensional coordinates. Two vertices are connected by an edge if and only if their distance is at most the communication radius. For simplicity of exposition, we normalize the coordinate assignment so that the communication radius is 1. The unit-disk graph model is widely used, e.g., in the past theoretical work on connectivity [19] as well as in certain practical scenarios with carefully engineered radios [47].

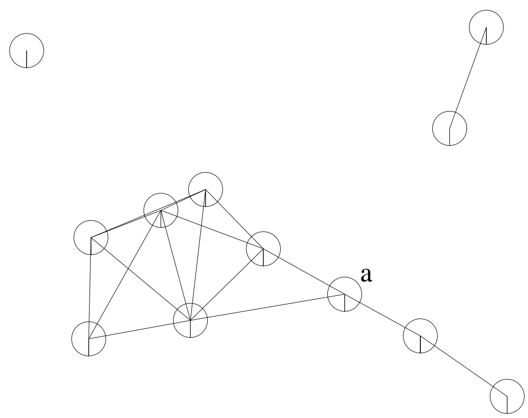


Fig. 1. An example unit-disk graph.

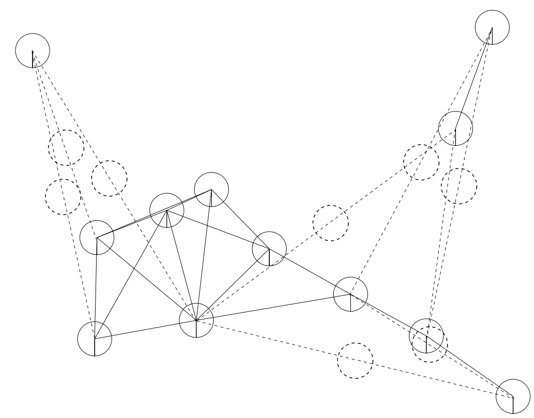


Fig. 2. The example unit-disk graph from Fig. 1 with added vertices, marked with dashed circles, to establish 3-connectivity among the original (solid) vertices. The new vertices lie closely enough to each other to ensure 3-connectedness for the entire graph. We omit drawing edges between the added vertices for clarity.

In this paper we suppose that the network is *multi-hop*, meaning that the mobile devices cooperate to route each others' messages. Thus we are interested in multi-node communication paths between the source and destination of a message. In anticipation of node failures resulting e.g. from power failure or power depletion of a mobile node, we are also interested in finding multiple disjoint communication paths between any source/destination pair.

In this paper we consider the following problem, given a sensor network represented as a unit-disk graph, we wish to compute and deploy the minimum number of additional devices to ensure that the resulting unit-disk graph satisfies the fault-tolerance constraint called vertex k-connectivity. A graph is *vertex k-connected* if there are at least k vertex-disjoint paths connecting every pair of vertices, or equivalently, the graph remains connected when any set of at most $k-1$ vertices is removed. Hence our goal is to make the network resilient to k node failures. In the problem we consider, we are given such a plane unit-disk graph and our goal is to deploy the minimum number of additional sensors to ensure one of two fault-tolerance constraints on the resulting unit-disk graph: either there should be k paths in the new network between every pair of original sensors, or there should be k paths in the new network between every pair of (old or new) sensors. We call the first constraint *partial k-connectivity* and the second constraint *full k-connectivity*.

Fig. 1 shows an example unit-disk graph that is not 3-connected. The largest component of the graph is 1-connected as it can be separated with the removal of the vertex marked a, for example. The graph in Fig. 2 shows the same graph with additional vertices to establish 3-connectivity among vertices from the original graph.

Our problem has been considered before only in the special case $k = 1$, where the problem has also found application in VLSI design and evolutionary/phylogenetic tree constructions in computational biology. See [17] for a description of these and other applications. The problem is NP-hard even for $k = 1$ [5]. The best approximation algorithm to our knowledge is a $5/2$-approximation algorithm by Du et al. [19], again only for the case $k = 1$.

An important related problem is, given a weighted complete graph K and a number $k \geq 1$, to find minimum-weight k-vertex-connected subgraph of K. This problem can be viewed as analogous to our problem of k-fault tolerance but for wired networks. Frank and Tardos [48] and Khuller and Raghavachari [49] were among the first authors who worked on this problem. The problem is NP-hard, and there are by now several polynomial-time approximation algorithms with guaranteed performance ratios. An *α-approximation algorithm* is a polynomial-time algorithm whose solution cost is at most α times the optimal solution cost. Kortsarz and Nutov [6] developed a k-approximation algorithm. At the heart of this algorithm is a combinatorial algorithm of Gabow [50] whose running time is $O(k^2|V|^2|E|)$ (an improvement to an algorithm of Frank and Tardos [48]). They also develop better approximation algorithms for small k, specifically, an approximation ratio of $\lceil (k+1)/2 \rceil$ for $k \leq 7$. Cheriyan et al. [7] improved the $\Theta(k)$ approximation ratio with an $O(\lg k)$-approximation algorithm, provided that the number of vertices in the graph is at least $6k^2$. This algorithm is based on an iterative rounding method and the ellipsoid algorithm applied to a linear-programming relaxation of exponential size, and hence is not very practical.

Our algorithms will use as a subroutine any one of these approximation algorithms for minimum-weight k-connected subgraph. We suppose that the subroutine we use has an approximation ratio of α, and state our own approximation ratio in terms of α. This generality allows us to choose an algorithm according to practicality, or to chose a future improvement to the state-of-the-art for this problem, and understand the impact on the final approximation ratio. Note that a better approximation algorithm, with performance ratio $2 + (k-1)/n$ [6], is known if the graph weights satisfy the triangle inequality, but the weighted complete graphs we consider do not satisfy this property. Algorithm 1 presents the formal operation.

III. Algorithm for Connectivity Repair

In this section we describe our algorithm for minimizing the number of additional sensors to guarantee k-connectivity.

The algorithm is relatively simple, building on approximation algorithms for minimum-weight k-vertex-connected subgraph. This modular design allows us to use several candidate algorithms for finding k-connected subgraphs and achieve a range of trade-offs between quality and performance. In particular, we can use a constant-factor approximation algorithm for k-connected subgraphs and obtain a constant-factor approximation algorithm for k-connectivity repair, for any fixed k. The analysis of this algorithm is complicated because we need to prove that the simplicity of the algorithm does not prevent it from finding more intricate, better solutions; this topic is addressed in the next section.

Algorithm 1 K-CONNECTIVITY-REPAIR

input: k, set V of vertices and their coordinates
if $n \geq k$ **then**
  $E \leftarrow \{(v,w) \mid v,w \in V,\ v \neq w\}$
  $K \leftarrow (V,E)$
  $\omega \leftarrow$ new $V \times V$ array
  for vertices $v,w \in V$ **do**
    $\omega[v,w] \leftarrow \lceil \|v-w\| \rceil - 1$
  end for
  call K-CONNECTED-SUBGRAPH (k,K,ω) to compute α-approximate minimum-weight k-connected spanning subgraph S of (K,ω)
  for edge $(v,w) \in E(S)$ **do**
    for $i = 1,2,\ldots,\omega[v,w]$ **do**
      $t \leftarrow i/(\omega[v,w]+1)$
      place k sensors at position $(1-t)\cdot v + t \cdot w$
      place $k-1$ sensors at position v
      place $k-1$ sensors at position w
    end for
  end for
else
  call K-CONNECTIVITY-REPAIR $(1,\ V)$
  $N \leftarrow \{$newly placed sensors$\}$
  for $x \in V \cup N$ **do**
    place $k-1$ sensors at position x
  end for
end if

The algorithm divides into two cases. Our description concentrates on the main case that the number n of original sensors is at least the desired connectivity k. The second case that $n < k$ is simpler and we consider it later.

First we compute a weighted complete graph K on the same set of vertices as the given graph G. The weight on an edge $\{v,w\}$ is one less than the ceiling of the Euclidean distance between the two points v and w: $\omega(v,w) = \lceil \|v-w\| \rceil - 1$. This weight is zero if v and w are already connected by an edge in the unit-disk graph G, and otherwise it is the number of additional sensors required to connect v and w by a straight path.

Second we run an α-approximation algorithm to find an approximately minimum-weight k-vertex-connected subgraph of this weighted complete graph K. See Section II for a summary of known theoretical approximation algorithms for this problem; see Section V for more practical implementations, including a greedy approach and a fast distributed algorithm. Note that our constructed graph K likely does not satisfy the triangle inequality: if G is connected, then there is a zero-weight path between every pair v,w of vertices, so the triangle inequality would require that all edges $\{v,w\}$ have weight 0, which is rarely the case in K. Therefore we can only use approximation algorithms for general graphs.

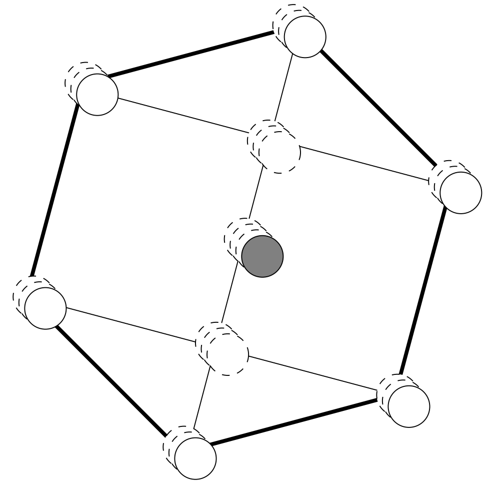


Fig. 3. An illustration of the approximation algorithm's performance in establishing full 3-connectivity. Solid vertices exist in the input at the peripheral and establish the heavy edges for zero cost. The algorithm chooses to add the thinner edges and places additional sensors, marked with dashed circles, along the new edges. The optimal repair adds only the shaded vertex. We omit the optimal solution's edges to avoid clutter.

Third we translate the chosen edges in the k-vertex-connected subgraph of K into a placement of new sensors. This step depends on the desired fault-tolerance constraint. For partial k-connectivity, we simply place ω sensors along the line segment connecting the endpoints of each edge of weight ω, spaced a unit distance apart. For full k-connectivity, we place ω clusters of k collocated sensors along the line segment connecting the endpoints of each edge of weight ω, spacing the clusters a unit distance apart. (Of course, in practice, these clusters can be spread out in a small neighborhood of a point instead of all being placed at the same point, at only a small additional cost.) In addition, for each edge of weight $\omega > 0$, we place $k-1$ additional sensors at each endpoint of the edge.

In the case that $n < k$, the graph K has fewer than k vertices and therefore has no k-connected subgraph. We run the algorithm for $k = 1$ to compute an approximately minimum number of additional sensors that connect the sensors. Then we replicate each original and additional sensor k times by adding $k-1$ more copies.

Fig. 3 demonstrates an example of how the approximation algorithm establishes 3-connectivity to the network formed by the six peripheral vertices. The dark edges have no cost as the original graph already supports them. The K-CONNECTED-SUBGRAPH routine chooses the light edges to establish 3-connectivity and the approximation places new vertices, marked with dashed circles, along the chosen edges. The optimal solution places a single point, drawn as gray, in the graph center.

IV. Analysis

The main difficulty in the analysis of our algorithm's performance ratio is that the *Steiner points*—i.e., additional points other beyond the input points—can be placed in infinitely many possible locations. In particular, there may be some locations to place a Steiner point that simultaneously interconnect several pairs of original points. Our algorithm does not search for such "hub locations", nor will it notice that it found one if it happens to use one. Another, more subtle problem along the same lines is that it is not always optimal to connect pairs of original points by straight sequences of Steiner points. Rather, it may be beneficial to connect several original points by straight lines to common Steiner points. This issue is precisely what makes Euclidean Steiner tree NP-hard, in contrast to minimum spanning tree. A third challenge is that our objective is to (approximately) minimize the number of added sensors, not the total number of sensors. In particular, if the graph is already k-connected, any approximation algorithm must not add any sensors, because otherwise the ratio to the optimal cost of 0 would be infinite. Thus we need to exploit the existing connectivity among the original sensors, because we cannot afford to pay for it again. This more difficult objective prevents us from using structures whose cost depends on the number of original sensors. (Otherwise we could repeat a minimum spanning tree on the original sensors k times, which would be a trivial $O(k)$ approximation on the total number of sensors.) These issues prevent us from using standard approximation algorithms and analysis based on e.g. minimum spanning trees.

Again we first consider the main case that $n \geq k$.

Lemma 1: For any set of original and Steiner sensors, there is a subgraph G' of the induced unit-disk graph G such that (1) for each edge of G' incident to at least one Steiner sensor, we can assign it to one of its Steiner endpoints such that each Steiner sensor is assigned at most $6k$ edges, and (2) for any set S of less than k vertices, two vertices are connected in $G - S$ if and only if they are connected in $G' - S$.

Proof: We construct G' by considering the Steiner sensors in an arbitrary order, and showing by induction that it suffices to connect each Steiner sensor to at most $6k$ original sensors and/or Steiner sensors that come earlier in the order. Let $s_1, s_2, \dots, s_m$ denote the Steiner sensors, ordered arbitrarily. For each $0 \leq i \leq m$, let G_i denote the graph on all original sensors and just the first i Steiner sensors $\{s_1, s_2, \dots, s_i\}$. Thus $G_m = G$. We will define a subgraph G'_i of G_i for each $0 \leq i \leq m$, such that G'_i satisfies the two properties of the lemma with respect to G_i (i.e., substituting G_i and G'_i for G and G'). Thus G'_m will serve as the desired G'.

In the base case, $G'_0 = G_0$ consists of just the original sensors, and the desired properties hold trivially: the first property because there are no Steiner sensors, and the second property because $G'_0 = G_0$. For the induction step, suppose we have already constructed G'_{i-1} and we wish to construct G'_i. Each G_i differs from the previous G_{i-1} only in that it includes an additional vertex s_i and some incident edges, and we will construct G'_i similarly to differ from G'_{i-1} only around s_i. Thus for G'_i to satisfy the first property we need only that the degree of s_i in G'_i is at most $6k$; then we can assign all of these edges to s_i. We divide the neighbors of s_i in G_i into six groups by dividing the unit disk centered at s_i into six equal pie wedges of angle $60°$. (The neighbors of s_i in G_i are precisely those sensors in the unit disk centered at s_i.) Then we select k arbitrary neighbors from each of the six groups (or we select the entire group if it has size less than k), and make those $6k$ or fewer vertices the neighbors of s_i in G'_i. Certainly s_i has degree at most $6k$ in G'_i, so the first property holds. The key property of this construction is that all vertices in the same group are connected by edges in G_i, because each pie wedge has diameter 1.

Finally we must show that G'_i satisfies the second property that, for any set S of less than k vertices, two vertices are connected in $G_i - S$ if and only if they are connected in $G'_i - S$. Consider some set S of less than k vertices. Because G'_i is a subgraph of G_i, we need to show only that two vertices v and w connected in $G_i - S$ are also connected in $G'_i - S$. (Thus, in particular, the vertices v and w under consideration are not in S.) If S contains s_i, then $G'_i - S = G'_{i-1} - S$ and $G_i - S = G_{i-1} - S$, so by the induction hypothesis on G'_{i-1} v and w are connected in $G'_{i-1} - S = G'_i - S$. Now we consider the case that S does not contain s_i. If v and w are connected by a path in $G_i - S$ that does not visit s_i, then that path also exists in $G_{i-1} - S$, so by induction the vertices are connected in $G'_{i-1} - S$ and thus in the supergraph $G'_i - S$. (In particular, if S contains s_i, then this case applies.) Otherwise, we know that any path connecting v and w in $G_i - S$ visits s_i, and thus in particular any such path visits a vertex in the unit disk centered at s_i immediately before and after s_i the path visits other vertices in the unit disk centered at s_i.

Let v' be the first vertex along a path connecting v and w in $G_i - S$ that is inside the unit disk centered at s_i, and let w' be the last vertex along the same path that is inside the unit disk centered at s_i. (Note that v' might be v, and w' might be w.) Because v and v' are connected by a path that does not visit s_i, by the previous case they are connected in $G'_{i-1} - S$, and similarly w and w' are connected in $G'_{i-1} - S$. We cannot have v' and w' in the same pie wedge of the unit disk, because then they would be connected via an edge in G_{i-1} and thus connected in $G_{i-1} - S$ and by induction connected in $G'_{i-1} - S$, so there would have been a path connecting v and w that does not use s_i.

Now we argue that v' and s_i are connected in $G'_i - S$; by a symmetric argument w' and s_i are connected in $G'_i - S$, and thus v and w are connected in $G'_i - S$. If $v' = s_i$ (which happens precisely when $v = s_i$), then they are trivially connected. Otherwise, v' is in one of the six pie wedges surrounding s_i. If there are at most k vertices in the pie wedge containing v', then s_i has edges to all of them in G'_i, and thus in particular there is an edge between s_i and v'. Otherwise, among the k neighbors of s_i in G'_i in the pie wedge containing v', at least one neighbor v'' must be in $G'_i - S$ because S has size less than k. Because v' and v'' are in the same pie wedge, they are connected by an edge in G_{i-1} so by induction connected in G'_{i-1}. Adding the edge between v'' and s_i to this path, we find that v' and s_i are connected in G'_i. Therefore v and w are connected in G'_i. ■

This lemma shows that G and the constructed subgraph G' are essentially the same in terms of connectivity. The next lemmas show how to remove Steiner points from G' again without losing any connectivity. We consider separately each "Steiner component" defined as follows. The *Steiner component* rooted at a Steiner sensor s is formed by growing a set of vertices and edges in G' starting with $\{s\}$ and stopping after we reach any original sensors. The Steiner component includes the edges connecting pairs of sensors in the component provided at least one of the endpoints is a Steiner sensor. (Equivalently, a Steiner component is a connected component of the induced subgraph of G' on the Steiner sensors, together with the edges connecting these Steiner sensors to original sensors and these original sensors.)

Lemma 2: If G' has at least k original vertices and is vertex k-connected, then the number of original vertices in each Steiner component is at least k.

Proof: The set of original vertices of a Steiner component forms a cut in G' unless that Steiner component is all of G'. In either case we must have at least k original vertices in the Steiner component. ■

Every Steiner component C has a spanning tree $T(C)$ in which the original sensors are leaves of $T(C)$.

Lemma 3: The number of edges in an Eulerian tour of the spanning tree $T(C)$ of a Steiner component C in G' is at most $12k$ times the number of Steiner sensors in C.

Proof: The number of edges in the Eulerian tour is exactly twice the number of edges of $T(C)$. Each of these edges can be assigned to one of the Steiner sensors in C, and by Lemma 1, the number of assignments is at most $6k$ times the number of Steiner sensors in C. Therefore the number of edges in the Eulerian tour is at most $12k$ times the number of Steiner sensors in C. ■

For any integer $n \geq 3$ and any positive integer $k \leq n$, the *Harary graph*[1] $H_{k,n}$ is the k-connected graph on n vertices $v_1, v_2, \ldots, v_n$ where each v_i is connected via an edge to the preceding $\lceil k/2 \rceil$ vertices $v_{i-1}, v_{i-2}, \ldots, v_{i-\lceil k/2 \rceil}$ and the succeeding $\lceil k/2 \rceil$ vertices $v_{i+1}, v_{i+2}, \ldots, v_{i+\lceil k/2 \rceil}$.

We consider the following procedure for *replacement of a Steiner component* C. First we remove the Steiner sensors in C. Second we take an Eulerian tour of the spanning tree $T(C)$. Third we construct a Harary graph $H_{k,m}$ on the m original sensors in C ordered by the order in which the Eulerian tour visits these leaves of $T(C)$. (By Lemma 2, $m \geq k$, so the Harary graph exists.) We view the edges of this graph as edges in the weighted complete graph K, and add these edges to our graph wherever they do not already exist, [2] and translate each edge of weight w into a sequence of w sensors equally spaced along the line segment connecting the endpoints. The resulting graph is no longer a unit disk graph: some edges come from the original unit-distance constraint, and others edges come from K. Once we replace all Steiner components in G', we obtain a subgraph of K with no Steiner sensors.

Lemma 4: The total weight of edges of K that replace a Steiner component C is at most $3k^3 + 12(k^2 + k)$ times the number of Steiner sensors in C.

Proof: Each edge in the Harary graph connects two original sensors that are within distance at most $\lceil k/2 \rceil$ in the order defined by the Eulerian tour. We charge the weight of this edge to the path of the Eulerian tour that connects these two original sensors. The weight of the edge is at most the number of edges in the path of the Eulerian tour because the edge in K represents a shortcutting of the path taken by the Eulerian tour. We distribute the charge on the path of the Eulerian tour to the subpaths connecting consecutive original sensors in the Eulerian tour. Each subpath is charged at most $\lceil k/2 \rceil^2$ times, one for each edge of the Harary graph that spans that subpath. By Lemma 3, the number of edges in the Eulerian tour is at most $12k$ times the number of Steiner sensors in C. Therefore the total weight of the replacement is at most $12k\lceil k/2 \rceil^2 \leq 12k(k/2+1)^2 = 3k^3 + 12(k^2 + k)$ times the number of Steiner sensors in C. ■

Lemma 5: If G' is vertex k-connected, then replacement of all Steiner components in G' results in a vertex k-connected subgraph of K.

Proof: We claim that replacement of a Steiner component preserves vertex k-connectivity. Let $C_1, C_2, \ldots, C_r$ denote the Steiner components in G'. For each $0 \leq i \leq r$, let G'_i denote G' after replacement of the first i Steiner components. Thus $G'_0 = G'$ is k-connected.

Assume by induction that G'_{i-1} is k-connected. Consider any set S of less than k vertices in G'_i whose removal disconnects two vertices v and w in $G'_i - S$. By the induction hypothesis, there is a path connecting v and w in $G'_{i-1} - S$. Let v' and w' be the first and last vertex along that path that belong to Steiner component C_i. Thus v' and w' are both original vertices and therefore also present in G'_i. Also, v and v' are connected by the same subpath in $G'_i - S$, as are w and w'. Because the replacement Harary graph is k-connected, removal of S cannot disconnect it, so v' and w' are connected in the Harary graph and thus in $G'_i - S$. Therefore v and w are connected in $G'_i - S$. The result follows by induction. ■

Theorem 6: In the case $n \geq k$, the algorithm is a polynomial-time $O(k^4\alpha)$-approximation on the minimum number of added sensors to attain vertex k-connectivity of the entire unit-disk graph.

Proof: Consider the optimal set of added sensors that results in a vertex k-connected unit-disk graph G. We construct G' as in Lemma 1, and then perform a replacement of each Steiner component. By Lemmas 1 and 5, the resulting subgraph of K is vertex k-connected. By Lemma 4, the total weight of the subgraph of K is at most $3k^3+12(k^2+k)$ times the number of Steiner sensors in G. The minimum-weight k-connected subgraph of K can have only smaller weight, so is also at most $3k^3 + 12(k^2 + k)$ times the number of Steiner sensors in G. Our algorithm finds an α-approximation to the minimum-weight k-connected subgraph, and then for every edge of weight $w \geq 1$, adds $kw+2k-2 < 3kw$ sensors. (This replication guarantees vertex k-connectivity of the entire graph because the removal of less than k vertices cannot disconnect

[1] In fact, Harary graphs are normally defined differently when k is odd. We round up to the Harary graph for the next even value of k in order to obtain the desired approximation bound in this paper.

[2] We avoid the addition of multiple edges, but in fact single edges and multiple edges are equivalent for our purposes of vertex k-connectivity.

the subgraph of K, nor can it disconnect the sensors that constitute an edge of K.) Therefore the number of sensors added by the algorithm is less than $(9k^4\alpha + 36(k^3 + k^2)\alpha)$ times the optimal number of added sensors used in G. ■

In the second case that $n < k$, the replication guarantees vertex k-connectivity of the entire graph because the removal of less than k vertices still leaves at least one copy of the original spanning tree which is connected. We apply the $k = 1$ analysis to show that the number of added sensors for 1-connectivity is $O(1)$ times the optimal. The optimal number of added sensors for 1-connectivity is certainly a lower bound on the optimal number of added sensors for k-connectivity. The replication of these sensors costs an additional factor of k. The replication of the original sensors uses $k(k-1)$ additional sensors, which can be charged to the optimal number of added sensors for k-connectivity, which is at least 1 because the original graph cannot be k-connected. Therefore we obtain an approximation ratio of at most $k^2 + O(k)$.

Corollary 7: For any fixed $k \geq 1$, our algorithm is an $O(1)$-approximation on the minimum number of added sensors to attain vertex k-connectivity of the entire unit-disk graph.

The analysis bounds the number of vertices used to establish k-connectivity in the network through considering only points lying on edges in the complete weighted graph representing the network. Thus it leads us to simpler algorithms to achieve k-connectivity that do not have to consider Steiner points. We discuss such an algorithm next.

V. Practical Implementations

In this section we present practical implementations of the k-connectivity repair algorithm analyzed in the previous section. All the algorithms that we consider are based on the K-CONNECTIVITY-REPAIR outline given in Algorithm 1, but they use different subroutines for K-CONNECTED-SUBGRAPH (line 9). The guaranteed α-approximation algorithms for k-connected subgraphs [6], [7] are based upon solving linear programs, and implementation may be difficult on computationally and communication-constrained sensor network nodes. Furthermore, because those algorithms are designed to optimize worst-case behavior, it is unclear how they compare to other algorithms on average-case instances. We implement simple greedy and distributed algorithms for K-CONNECTED-SUBGRAPH, and show that the resulting approximation factors in practice are much smaller than the worst-case bounds.

Our two practical algorithms are both based upon greedy approaches that choose inexpensive repairs. While the straightforward greedy approach is conceptually simple, it is difficult to implement in a distributed environment. To address such difficulty, we present and implement a distributed algorithm, similar in spirit to Garcia-Molina's invitation leader-election algorithm [51], where network nodes elect leaders to make decisions how to merge k-connected subnetworks.

A. Greedy Approach

Algorithm 2 illustrates a greedy solution to K-CONNECTED-SUBGRAPH. The algorithm consists of two phases. We begin with an empty subgraph of the specified graph G. In the first phase, the algorithm repeatedly adds edges (v, w) from the graph G in increasing order by weight $\omega(v, w)$. The first phase ends once the subgraph is k-connected. In the second phase, the algorithm repeatedly attempts to remove edges (v, w) from the subgraph, in decreasing order by weight $\omega(v, w)$, but putting the edge back if it was necessary for k-connectivity. Experimentally, this pruning stage can remove 58–85% of the added edges. The resulting subgraph is therefore a k-connected subgraph of G, and we expect that the weight of the chosen edges is reasonably small.

Algorithm 2 Greedy K-CONNECTED-SUBGRAPH

input: k, $G = (V, E)$, ω
$G' \leftarrow (V, \emptyset)$
$E' \leftarrow \{(v, w) \mid v, w \in V\}$
for $(v, w) \in E'$ in increasing order of $\omega[v, w]$ **do**
  $E(G') \leftarrow E(G) \cup \{(v, w)\}$
  if G' is k-connected **then**
    break
  end if
end for
for $(v, w) \in E(G')$ in decreasing order of $\omega[v, w]$ **do**
  $G'' \leftarrow (V, E(G') \setminus \{(v, w)\})$
  if G'' is k-connected **then**
    $G' \leftarrow G''$
  end if
end for
return G'

This greedy algorithm produces the same subgraph as the following less-efficient algorithm: start from the complete graph G as the subgraph, and repeatedly remove edges (v, w) that do not destroy k-connectivity, in decreasing order by weight $\omega(v, w)$. For $k = 1$, this algorithm (and hence the original greedy algorithm) finds the minimum spanning tree; it is essentially dual to Prim's algorithm. Thus the greedy algorithm generalizes a minimum-spanning-tree construction to $k \geq 1$, so we expect that it does well.

The greedy algorithm, however, can have arbitrarily poor performance, as depicted in Figure 4 when the graph forms a long narrow U-shaped chain to be repaired to 2-connectivity. The greedy algorithm chooses to reinforce the existing links by placing new nodes on top of existing ones. The optimal repair places the two gray nodes to create a loop. Because the chain can be arbitrarily long, but still require only a fixed number of nodes to repair, the ratio of the worst-case greedy cost to optimal is unbounded.

On the other hand, for the Steiner version where you only have to k-connect designated vertices, the algorithm can be $\Omega(n)$ away from optimal. Consider, for example, two vertices x and y that are connected by two paths, one path with $n-1$ edges of weight 0 and one edge of weight $1+\epsilon$, and the other path with n edges of weight 1. The greedy algorithm for $k = 1$ removes the weight-$(1+\epsilon)$ edge and destroys the path of total weight $1+\epsilon$, leaving the path of total weight n, for a ratio arbitrarily close to n.

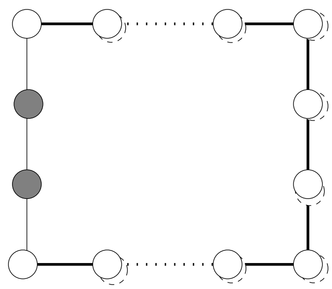


Fig. 4. An example of poor performance by the greedy algorithm to establish 2-connectivity. The greedy algorithm places the dashed nodes at the same position as the existing nodes, whereas the optimal repair places the gray nodes to construct a circuit.

B. Distributed Implementation

Algorithm 2 repeatedly tests for k-connectivity—a time consuming operation.[3] The algorithm also requires global knowledge of the candidate edges.

To address both problems, we distribute a locally greedy algorithm that grows and merges k-connected components. The distributed approach relies on a synchronous message passing only in that all messages are either delivered or lost forever within a fixed time limit. Each component elects a leader to compute the cost of joining neighboring components. Two disjoint k-connected components form a larger k-connected component if bridged with k vertex-disjoint edges. Members of a k-connected component report the cost of joining some number of the closest other components. The leader chooses k vertex-disjoint paths used to merge with each component. The assignment can quickly be computed with an auction-based assignment implementation [52], or using more-easily implemented heuristics. Each leader then proposes merging with its lowest-cost counterpart. A propositioned leader accepts the first offer that matches its lowest computed cost.

When components are smaller then k nodes, leaders merge components by greedily creating k-cliques.

The distributed algorithm relaxes the need for synchronization, complete centralized knowledge of node distances, and k-connectivity testing. Each node, however, still requires an upper bound of the distance to each other node, obtained either by localization [53] or through bounding the distance through inspecting a routing table. Additionally, the algorithm has no pruning stage similar to the second stage of Algorithm 2, so we expect it to generate heavier subgraphs.

1) Merging Subnetworks: We now show how the merging of subnetworks produces a k-connected networks.

Lemma 8: For $(n \leq k)$, the k-connected graph G_k and the n-clique K_n form a k-connected supergraph when bridged so that every vertex in K_n has $k-n+1$ new neighbors in G and no two vertices in K_n share neighbors in G_k.

Proof: We prove Lemma 8 by contradiction. Suppose that a graph, G, is constructed in the manner of the lemma such that it not k-connected, so that its vertices can be partitioned into non-empty sets A, B and C so that C serves as a cut set separating the vertices in A and B.

claim: $A \cap V(G_k) = \emptyset$ or $B \cap V(G_k) = \emptyset$.The claim follows from noting that if both A and B contain elements of $V(G_k)$, then G_k could not have been k-connected.

Without loss of generality, let $B \cap V(G_k) = \emptyset$. Thus $A \cap V(K_n) \neq \emptyset$, because A is not empty.

claim: $k < |\mathcal{N}(B)|$. The claim follows from the expansion that each $v \in A$, $\mathcal{N}(\{v\}) = k$ and for each $u, v \in A$, $\mathcal{N}(\{v\})/\mathcal{N}(\{u\}) \neq \emptyset$.

Thus $|C| > k$ and we arrive at a contradiction that G must be k-connected. ■

Corollary 9: Given a k-connected graph G_k, the supergraph G constructed by adding one additional vertex connected to k distinct vertices in G_k is also k-connected.

Corollary 9 is sometimes referred to as the Expansion Lemma.

Corollary 10: The supergraph constructed from two disjoint k-connected graphs bridged with k vertex-disjoint edges is k-connected.

2) Protocol: Leader nodes alternate between two modes: invitation and listening. In the invitation mode, a leader requests from its followers bridges to other subnetworks, and issues a merge invitation to other leaders. An invitation will only be accepted if the recipient is in listening mode and the invitation includes the recipient's current group size. If the invitation is accepted the recipient relinquishes leadership and broadcasts for every group member, the member's new leader. Through a gossip network, a leader can determine whether its invitation has been accepted, or timeout if it hears no such change in the gossip network.

If J is the expected time for a node issuing requests to find a partner, δ the time spent waiting for requests, Δ the time time spent waiting for a replies to a request, and p the probability that a single message is delivered, then an upper-bound for J can be expressed as the recurrence

$$J = \frac{\delta}{\Delta + \delta} p\Delta + \left(1 - \frac{p\delta}{\Delta + \delta}\right)(\Delta + \delta + J),$$

which simplifies to

$$J = \frac{(\Delta + \delta)^2}{p\delta} - \delta.$$

If Δ is fixed, J is minimized when $dJ/d\delta$ to zero, leaving the optimal latent period to be

$$\delta = \frac{\Delta}{\sqrt{1-p}}.$$

A gossip protocol informs the network of group membership and group size. Only leader nodes may insert new gossip into the network which uses a logical timestamp associated with each subject to achieve consistency. A node receiving gossip about a subject compares the message timestamp with the timestamp for the subject. If the received timestamp is greater than the local one, the node updates its local gossip knowledge and broadcasts the gossip to its neighbors. If the received timestamp is less than the local one, the node broadcasts its

[3]The runtime improves if the connectivity test first checks that the minimum degree for each node is at least k. Noting that the minimum degree for each node must match or exceed k, and knowing such failure eliminates the need for many k-connectivity tests.

own gossip to the rest of the network. Each node periodically broadcasts its own state to ensure that changes are eventually recorded.

3) Message Complexity: We next bound the number of messages the algorithm requires to terminate.

Theorem 11: The distributed repair algorithm requires $O(n^2)$ point-to-point messages and $O(n^2)$ broadcast messages to repair a graph to k-connectivity.

Proof: The algorithm proceeds in rounds where each leader requests to merge with another leader. During each round, the algorithm merges at least two subnetworks, so the algorithm terminates after at most $n-1$ rounds.

We first count the messages sent to leaders by followers and the merge proposals sent between leaders. Each round requires that a leader node send one message to its preferred partner, and each follower to send to its leader a constant number of messages representing links to bridge to another subnetwork. No node sends more than a constant number of messages either as a leader or follower, hence no node sends more than $O(n)$ messages, and the algorithm requires $O(n^2)$ messages to terminate.

After each merge operation, the leader must inform the rest of the network of its erstwhile followers new group membership. A node may change its membership no more than $n-1$ times, thus there are $O(n^2)$ broadcast messages required to repair a network to k-connectivity. ■

VI. Experimental Results

We implement in simulation both the centralized greedy-repair and the distributed solutions from the previous section to compare their performance with a sampling-based and the optimal subgraph repair solutions. The sampling-based algorithm scatters new vertices uniformly through the environment until the original nodes are k-connected, similar to the starting point of [54], except that we do not require k-connectivity to the additional nodes. For small graphs, we compare the repairs of each of the three previous algorithms against the optimal placement, restricted by Algorithm 1, that we compute through combinatorial branch-and-bound search using Algorithm 2 as an initial bound.

We test each algorithm on uniformly generated and grid graphs that have had some portion removed to destroy 3-connectivity. The first graph structures we consider have nodes placed on a two-dimensional rectangular grid evenly spaced one unit apart. With the exceptions of the corners, these graphs are 3-connected. The second class of graphs are generated through a Las-Vegas method. The vertices are uniformly placed inside a fixed-size rectangular area until the resulting graph is k-connected.

The generated graphs are damaged by one of two methods, uniform selection and geographic selection. The uniform selection method takes size parameters p and n. Vertices are uniformly selected and removed from a graph originally containing n vertices until there are fewer than pn vertices remaining and the graph is no longer 3-connected. The selection process models network failures representing decay over time.

The geographic selection process selects a graph's two most-distant nodes, s and t. It repeatedly removes the node

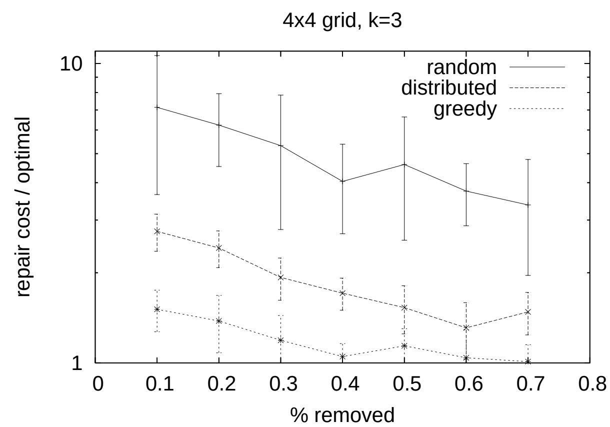


Fig. 5. The cost, relative to optimal, to repair 4×4 uniformly damaged grid graphs to 3-connectivity. The figure plots mean repair cost of groups of ten experiments with error bars denoting the observed (biased) standard deviation.

located mid-way along the shortest remaining s-t path. We test our algorithms on augmented graphs that have had only enough nodes removed so that k-connectivity does not hold, as well as on graphs that the process completely disconnected. The resulting graphs model networks with geographically dependent failures.

We measure the number of vertices required to repair a damaged graph to k-connectivity, forgetting the vertices removed from the original graph. For these experiments, we only guarantee that the original vertices are k-connected, not the additional ones. Multiplying the cost by k (for replication) provides an upper bound on the cost of repairs guaranteeing that the additional vertices are also k-connected. Empirically, however, additional vertices almost always lie closely enough to k-connected without extra duplication.

Fig. 5 plots the mean number of vertices added to uniformly damaged graphs to re-establish 3-connectivity to 4×4 grids for the greedy, distributed, and randomized implementations. Each of the plotted values are normalized first by the number of vertices in the damaged graph and then by the optimal cost to repair the graph. The greedy algorithm frequently finds the optimal repair for graphs with more than 30% of their vertices removed.

It is difficult to measure the optimal cost of larger graphs, but we observe that the repair costs for uniformly damaged 10×10 grid graphs plotted in Fig. 6 scale similarly to the costs of 4×4 before normalization by the optimal.

Figs. 7 and 8 plot results for the same experiments on denser, uniformly generated graphs. For comparison to optimal, we are able only to look at uniformly distributed graphs in a 3×3 area, sustaining at most 35% node removal. The original graphs have on average 48 nodes, but some have as many as twice that. In repairing these graphs, it appears that costs relative to optimal for three repair algorithms are independent to the portion removed. Table I shows for each algorithm the mean cost relative to optimal and χ^2 scores with five degrees-of-freedom. The χ^2 goodness-of-fit measure sums the squared error of the predicted performance, inferred from the mean across the experiments using the same algorithm,

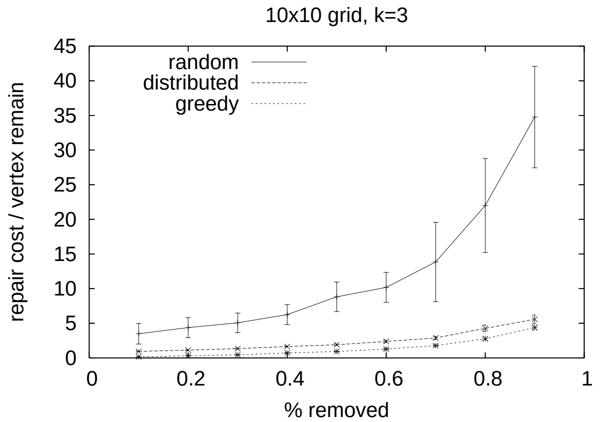


Fig. 6. The cost, relative to the number of vertices remaining, to repair uniformly damaged 10×10 grid graphs to 3-connectivity. The figure plots mean repair cost of groups of ten experiments with error bars denoting the observed (biased) standard deviation.

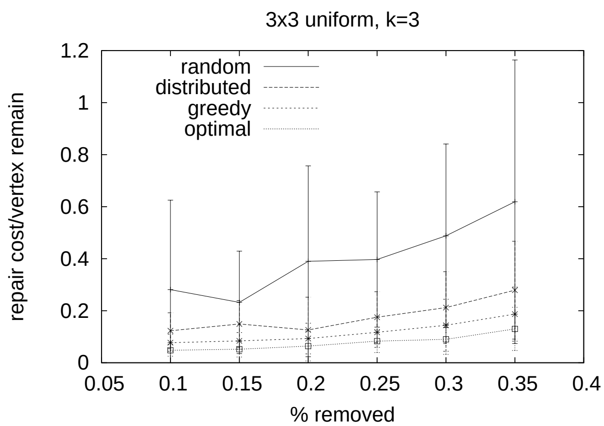


Fig. 7. The cost of re-establishing 3-connectivity to uniformly damaged graphs that were originally uniformly generated in a 3×3 area. We plot the mean repair cost per remaining vertex and (biased) standard deviation for groups of 36 experiments.

and the observed performance. The low χ^2 scores give us little reason to question that algorithm performance, relative to optimal, is independent to the portion removed. We also plot experiments on 7×7 uniformly generated graphs that average 250 vertices in Fig. 8. Again the repair costs increase similarly in both of the two scales. Fig. 9 demonstrates the cost to re-establish 3-connectivity to graphs suffering removal of half of their vertices as a function of the size of the network after removing nodes. Empirically, the cost decreases with the size of the network, shown by the plotted regression lines fitting the logarithm of the cost as a linear function of the network size.

The performance of the greedy repair algorithm is surprising and requires some explanation. Fig. 10 shows an example of a uniformly generated 3-connected graph. Nodes A, B, and C are removed and the greedy repair algorithm takes the resulting graph as input. The algorithm adds node Z. Node A is superfluous to preserving connectivity in the original graph; it resides at the graph's fringe. Because it resides in a densely populated region, node B is redundant. Thus, these two node

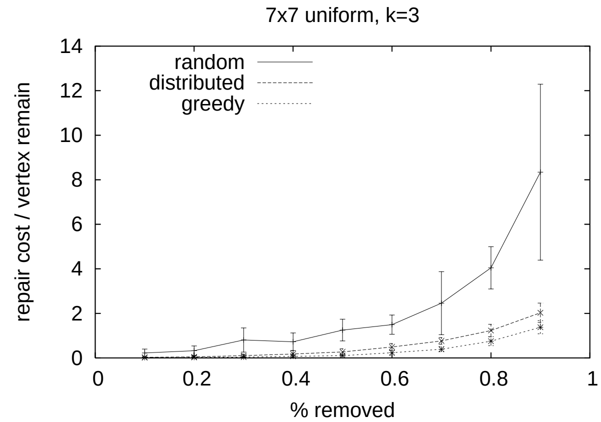


Fig. 8. The cost of re-establishing 3-connectivity to uniformly damaged graphs that were originally uniformly generated in a 7×7 area. We plot the mean repair cost relative to optimal and (biased) standard deviation for groups of ten experiments.

algorithm	mean repair cost / optimal	χ^2 (5 dof)
random	5.3	1.35
distributed	2.3	0.96
greedy	1.5	0.18

TABLE I
THE COST RELATIVE TO OPTIMAL FOR EACH ALGORITHM AND THE χ^2 GOODNESS-OF-FIT SCORES WITH FIVE DEGREES OF FREEDOM INDICATING THE SUM OF SQUARED DIFFERENCES OF THE SAMPLE MEANS AND THE GRAND MEAN.

removals require no attention for repair. The greedy algorithm replaces node C with Z, re-establishing 3-connectivity.

Because many of the vertices uniformly selected to be

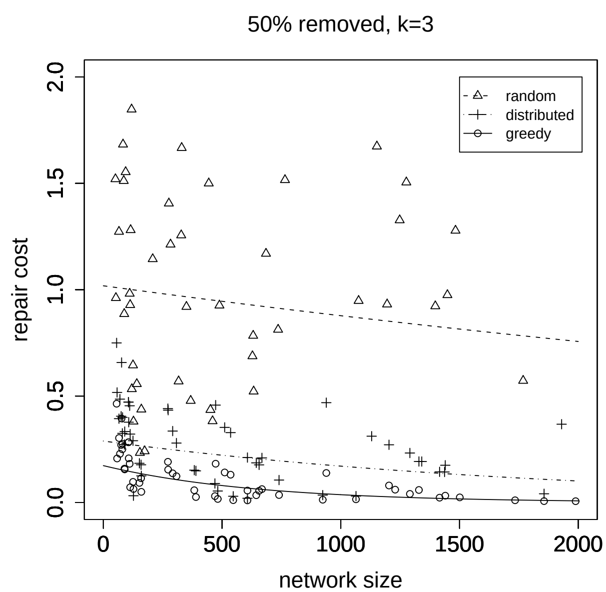


Fig. 9. The proportion of additional nodes required to restore 3-connectivity to previously 3-connected networks that have had 50% of their nodes removed. We also plot the best-fit regression for $\ln(\text{cost}) = \beta_0 + \beta_1\text{size}$ to illustrate the trend.

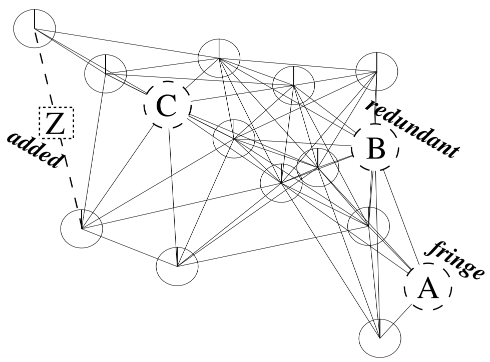


Fig. 10. A typical graph-repair example. The destruction process removes nodes A, B, and C. The greedy repair algorithm adds node Z.

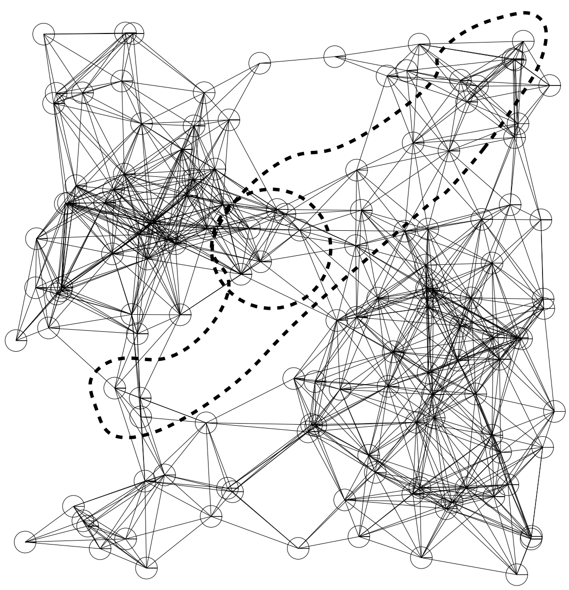


Fig. 11. An example of complete disconnection, represented by the larger dashed region, and substantial vertex removal, denoted by the dashed circle.

removed from the input graphs did not contribute to the graph's k-connectedness, we also test the repair algorithms on geographically damaged graphs. To better ensure that removed vertices contribute to connectivity, we uniformly generate 3-connected 10×10 unit-disk graphs, select the two most-distant vertices s and t, and remove a vertex in the middle of some number the shortest s-t paths. Fig. 11 shows an example 3-connected 10×10 unit-disk graph with two groups of vertices marked for removal. The smaller, circular grouping denotes a substantial disconnection that will affect k-connectivity, whereas removing the larger group completely disconnects the graph.

Table II shows the repair costs of removing enough vertices to completely disconnect s-t pairs and removing only enough edges to destroy k-connectivity. The table shows the increase in the number of vertices required to re-establish 3-connectivity, relative to the optimal repair cost. As the greedy and distributed algorithms can better localize the area to be

	greedy		dist		random	
	μ	σ	μ	σ	μ	σ
complete	1.35	0.72	1.59	0.76	3.94	3.27
substantial	2.13	1.68	2.14	2.77	7.65	9.36

TABLE II
THE REPAIR COST RELATIVE TO OPTIMAL OF REPAIRING UNIFORMLY UNIFORMLY GENERATED 3-CONNECTED 3×3 UNIT-DISK GRAPHS. THE TABLE REPORTS THE MEAN AND OBSERVED STANDARD DEVIATION FOR TEN TRIALS. THE "COMPLETE" ENTRY REPRESENTS GRAPHS THAT WERE COMPLETELY DISCONNECTED, WHEREAS THE "SUBSTANTIAL" ENTRY SHOWS THE REPAIR COST OF REMOVING ONLY ENOUGH VERTICES TO REMOVE k-CONNECTIVITY FROM THE GRAPH.

repaired, they perform about 3 times better than the random repair algorithm.

VII. CONCLUSION

It would be interesting to extend our guaranteed worst-case approximation algorithms to the case of partial k-connectivity, where there should be k vertex-disjoint paths between every pair of original sensors, but not necessarily between pairs involving added sensors. This weaker goal is natural if only the original sensors serve a useful purpose (e.g., carrying information), and the added sensors only serve for additional connectivity between them. We attack the weaker problem in the experiments, but usually the new points are k-connected anyway—not too surprising given that there exist density thresholds that probabilistically guarantee k-connectivity. Even minimum degree k is enough to probably give k-connectivity [55]. But what about the worst case? A variation of our approach may lead to efficient approximation algorithms for this case as well. In the complete graph K, we replicate each edge k times, and assign a weight to each replicated edge equal to the original edge weight if it is positive, or 1 if the original weight is zero. The point of this modification is that, in the partial k-connectivity model, repeating additional sensors can increase connectivity between two vertices, unlike regular k-connectivity. For our approach to work, however, we need a stronger version of the subroutine for finding the approximate minimum-weight k-connected subgraph that supports a multigraph as input. Alternatively, we can subdivide each edge and distribute the weight arbitrarily between the two halves, and use a subroutine that finds the approximate minimum-weight subgraph that is partially k-connected on the specified set of original vertices. There is evidence that this problem is significantly more difficult than regular k-connectivity [56]. If we had either such subroutine, it would seem that the rest of our approach would lead to efficient approximation algorithms for partial k-connectivity.

Our analysis of the approximation ratio of our algorithm is likely not tight; we believe that the same algorithm has an asymptotically smaller approximation ratio than what we prove. Our experimentation measuring worst-case behavior supports this intuition. An example is the case of 1-connectivity, where our algorithm simply short-cuts an Eulerian tour of a minimum spanning tree. Chen et al. [17] proved that this algorithm has a worst-case approximation ratio of precisely 4. In contrast, our analysis for general k proves an upper bound of somewhat more than 4 in the case $k = 1$. The main advantage of our approach and analysis is its generality, applying for arbitrary k.

We conjecture that our approximation results can be further strengthened to find a polynomial-time approximation scheme (PTAS), i.e., an algorithm that attains an approximation ratio of $1 + \epsilon$ for any desired $\epsilon > 0$. We believe that such a PTAS can be obtained using the techniques of Arora [57] and Mitchell [58]. These techniques have been successfully applied to obtain a PTAS for the related problem of finding the minimum-Euclidean-length k-connected subgraph [59]. Our conjecture is wide open even for the case of $k = 1$. To the best of our knowledge, the only related problem known to have a PTAS is the easier problem of approximating the total number of sensors, instead of the number of added sensors, for $k = 1$ [17]. For our problem and $k = 1$, the best known approximation algorithm has an approximation ratio of $5/2$ [19]. For our problem and $k > 1$, our algorithms are the only known approximation algorithms.

We also believe that our theoretical results can be extended to the more general *quasi unit-disk graph* model (see, e.g., [60]). In this model, two parameters $r \leq R$ determine connectivity: two nodes of distance at most r are guaranteed to be connected in the graph, while two nodes of distance more than R are guaranteed to be disconnected in the graph. Nodes whose distance is between r and R may or may not be connected in the graph. This generalized model handles a wide range of fading models; for example, if instead of a unit-disk cutoff, we have a different (e.g., star-shaped) cutoff, we can take the inscribed disk of radius r and circumscribing disk of radius R. Provided $R/r = O(1)$, it seems that our algorithm remains an $O(k^4\alpha)$-approximation for any fixed k.

In our current work, we are implementing the distributed k-connectivity repair algorithm on a mote network that interacts with a mobile robot. We are also developing improved distributed versions of the k-connectivity algorithm. One interesting question to explore in practice is how often k-connectivity repair should be run for a given rate of node failure.

ACKNOWLEDGMENTS

Support for this work has been provided in part by NSF grants IIS-0426838, IIS-0225446 and ITR ANI-0205445, and from Intel, Boeing, MIT Project Oxygen, and by award Number 2000-DT-CX-K001 from the Office for Domestic Preparedness, US Department of Homeland Security. Points of view in this document are those of the authors and do not necessarily represent the official position of the US Department of Homeland Security.

REFERENCES

[1] P. Corke, S. Hrabar, R. Peterson, D. Rus, S. Saripalli, and G. Sukhatme, "Autonomous deployment of a sensor network using an unmanned aerial vehicle," in *Proceedings of the 2004 International Conference on Robotics and Automation*, New Orleans, USA, 2004.

[2] P. Corke, S. Hrabar, R. Peterson, D. Rus, S. Saripalli, and G. Sukhatme, "Deployment and connectivity repair of a sensor net with a flying robot," in *Proceedings of the 9th International Symposium on Experimental Robotics*, Singapore, 2004.

[3] Mohsen Bahramgiri, MohammadTaghi Hajiaghayi, and Vahab S. Mirrokni, "Fault-tolerant and 3-dimensional distributed topology control algorithms in wireless multi-hop networks," *Wireless Networks*, vol. 12, no. 2, pp. 179–188, 2006.

[4] Mohammad Taghi Hajiaghayi, Nicole Immorlica, and Vahab S. Mirrokni, "Power optimization in fault-tolerant topology control algorithms for wireless multi-hop networks," *IEEE/ACM Transactions on Networking*, vol. 15, no. 6, pp. 1345–1358, 2007.

[5] G.H. Lin and G. Xue, "Steiner tree problem with minimum number of Steiner points and bounded edge-length," *Inform. Process. Lett.*, vol. 69, no. 2, pp. 53–57, 1999.

[6] G. Kortsarz and Z. Nutov, "Approximating node connectivity problems via set covers," in *Proceedings of the Third International Workshop on Approximation Algorithms for Combinatorial Optimization (APPROX)*, pp. 194–205. 2000.

[7] J. Cheriyan, S. Vempala, and A. Vetta, "An approximation algorithm for the minimum-cost k-vertex connected subgraph," *SIAM J. Comput.*, vol. 32, no. 4, pp. 1050–1055 (electronic), 2003.

[8] R. Ramanathan and R. Rosales-Hain, "Topology control of multihop radio networks using transmit power adjustment," in *Proceedings of Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, pp. 404–413. March 2000.

[9] R. Wattenhofer, L. Li, V. Bahl, and Y. M. Wang, "Distributed topology control for power efficient operation in multihop wireless ad hoc networks," in *Proceedings of twentieth Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, pp. 1388–1397. 2001.

[10] L. Li, J. Halpern, V. Bahl, Y. M. Wang, and R. Wattenhofer, "Analysis of a cone-based distributed topology control algorithm for wireless multi-hop networks," in *Proceedings of ACM Symposium on Principle of Distributed Computing (PODC)*, pp. 264–273. 2001.

[11] M. Hajiaghayi, N. Immorlica, and V. S. Mirrokni, "Power optimization in fault-tolerant topology control algorithms for w ireless multi-hop networks," in *Proceedings of the 9th Annual International Conference on Mobile Computing and Networking*. 2003, pp. 300–312, ACM Press.

[12] E. Lloyd, R. Liu, M. Marathe, R. Ramanathan, and S. Ravi, "Algorithmic aspects of topology control problems for ad hoc networks," *Proceedings of the Third ACM International Symposium on Mobile Ad Hoc Networking and Computing*, 2002.

[13] D. Blough, M. Leoncini, G. Resta, and P. Santi, "The K-neigh protocol for symmetric topology control in ad hoc networks," in *ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*. June 2003.

[14] X.Y. Li, W.Z. Song, and Y. Wang, "Efficient topology control for wireless ad hoc networks with non-uniform transmission ranges," *ACM Wireless Networks*, vol. 11, no. 3, 2005.

[15] X.Y. Li, Y. Wang, P.J. Wan, and C.W. Yi, "Robust deployment and fault tolerant topology control for wireless ad hoc networks," in *ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*. June 2003.

[16] Ning Li and Jennifer C. Hou, "FLSS: a fault-tolerant topology control algorithm for wireless networks," in *Proceedings of the 10th Annual International Conference on Mobile Computing and Networking*. 2004, pp. 275–286, ACM Press.

[17] D. Chen, D.Z. Du, X.D. Hu, G.H. Lin, L. Wang, and G. Xue, "Approximations for Steiner trees with minimum number of Steiner points," *J. Global Optim.*, vol. 18, no. 1, pp. 17–33, 2000.

[18] Kapali P. Eswaran and R. Endre Tarjan, "Augmentation problems," *SIAM Journal on Computing*, vol. 5, no. 4, pp. 653–665, 1976.

[19] D. Du, L. Wang, and B. Xu, "The Euclidean bottleneck Steiner tree and Steiner tree with minimum number of Steiner points," in *Computing and Combinatorics (Guilin, 2001)*, vol. 2108 of *Lecture Notes in Comput. Sci.*, pp. 509–518. Springer, Berlin, 2001.

[20] Juhua Pu and Zhang Xiong, "Research on the fault tolerance deployment in sensor networks," in *Proceedings of the 4th International Conference on Grid and Cooperative Computing (GCC 2005)*, Beijing, China, 2005, vol. 3795 of *Lecture Notes in Computer Science*, pp. 1179–1184.

[21] Abhishek Kashyap, *Robust Design of Wireless Networks*, Ph.D. thesis, University of Maryland, College Park, 2006.

[22] Abhishek Kashyap, Samir Khuller, and Mark A. Shayman, "Relay placement for higher order connectivity in wireless sensor networks," in *Proceedings of the 25th IEEE International Conference on Computer Communications (INFOCOM 2006)*, 2006.

[23] Xiaofeng Han, Xiang Cao, Errol L. Lloyd, and Chien-Chung Shen, "Fault-tolerant relay node placement in heterogeneous wireless sensor networks," in *Proceedings of the 26th IEEE International Conference*

on Computer Communications (INFOCOM 2007), Anchorage, Alaska, May 2007, pp. 1667–1675.

[24] Weiyi Zhang, Guoliang Xue, and Satyajayant Misra, "Fault-tolerant relay node placement in wireless sensor networks: Problems and algorithms," in *Proceedings of the 26th IEEE International Conference on Computer Communications (INFOCOM 2007)*, Anchorage, Alaska, May 2007, pp. 1649–1657.

[25] Weiping Shang, Pengjun Wan, Frances Yao, and Xiaodong Hu, "Algorithms for minimum m-connected k-tuple dominating set problem," *Theoretical Computer Science*, vol. 381, no. 1–3, pp. 241–247, 2007.

[26] Erik D. Demaine, MohammadTaghi Hajiaghayi, Hamid Mahini, Amin S. Sayedi-Roshkhar, Shayan Oveisgharan, and Morteza Zadimoghaddam, "Minimizing movement," in *Proceedings of the 18th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2007)*, New Orleans, Louisiana, January 2007, pp. 258–267.

[27] Bin Zeng, Jun Wei, and Tao Hu, "Optimal cluster-cluster design for sensor network with guaranteed capacity and fault tolerance," in *Proceedings of the 8th ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD 2007)*, Qingdao, China, 2007, pp. 288–293.

[28] Xuan Thanh Dang, Sergey Frolov, Nirupama Bulusu, Wu chi Feng, and António M. Baptista, "Near optimal sensor selection in the columbia river (corie) observation network for data assimilation using genetic algorithms," in *Proceedings of the 3rd IEEE International Conference on Distributed Computing in Sensor Systems (DCOSS 2007)*, Santa Fe, NM, June 2007, vol. 4549 of *Lecture Notes in Computer Science*, pp. 253–266.

[29] Daniela Tulone and Erik D. Demaine, "Revising quorum systems for energy conservation in sensor networks," in *Proceedings of the International Conference on Wireless Algorithms, Systems and Applications (WASA 2007)*, Chicago, Illinois, August 2007, pp. 147–157.

[30] Yannis Drougas and Vana Kalogeraki, "Distributed, reliable restoration techniques using wireless sensor devices," in *Proceedings of the 21th International Parallel and Distributed Processing Symposium (IPDPS 2007)*, Long Beach, California, March 2007.

[31] Luciana Moreira Sá de Souza, "FT-CoWiseNets: A fault tolerance framework for wireless sensor networks," in *Proceedings of the International Conference on Sensor Technologies and Applications (SensorComm 2007)*, October 2007, pp. 289–294.

[32] Zhijun Yu and Jie Wang, "Fault-tolerant sensor coverage for achieving wanted coverage lifetime with minimum cost," in *Proceedings of the International Conference on Wireless Algorithms, Systems and Applications (WASA 2007)*, 2007, pp. 95–102.

[33] Ai Chen, Ten H. Lai, and Dong Xuan, "Measuring and guaranteeing quality of barrier-coverage in wireless sensor networks," in *Proceedings of the 9th ACM international symposium on Mobile ad hoc networking and computing (MobiHoc 2008)*, Hong Kong, China, 2008, pp. 421–430.

[34] Feng Wang, My T. Thai, and Ding-Zhu Du, "2-connected virtual backbone in wireless networks," *IEEE Transactions on Wireless Communications*, To appear.

[35] Kenan Xu, *Device Deployment Strategies for Large-scale Wireless Sensor Networks*, Ph.D. thesis, Queen's University, Kingston, Ontario, Canada, 2008.

[36] Mohammad Al Hasan, Krishna K. Ramachandran, and John E. Mitchell, "Optimal placement of stereo sensors," *Optimization Letters*, vol. 2, no. 1, pp. 99–111, January 2008.

[37] M. Farukh Munir and F. Filali, "Increasing connectivity in wireless sensor-actuator networks using dynamic actuator cooperation," in *Proceedings of the Vehicular Technology Conference (VTC 2008)*, May 2008, pp. 203–207.

[38] S. Misra, Seung Don Hong, Guoliang Xue, and Jian Tang, "Constrained relay node placement in wireless sensor networks to meet connectivity and survivability requirements," in *Proceedings of the 27th IEEE Conference on Computer Communications (INFOCOM 2008)*, April 2008, pp. 281–285.

[39] Chee-Wei Ang and Chen-Khong Tham, "imst: A bandwidth-guaranteed topology control algorithm for tdma-based ad hoc networks with sectorized antennas," *Computer Networks*, vol. 52, no. 9, pp. 1675–1692, 2008.

[40] Alon Efrat, Sándor P. Fekete, Poornananda R. Gaddehosur, Joseph S. B. Mitchell, Valentin Polishchuk, and Jukka Suomela, "Improved approximation algorithms for relay placement," in *Proceedings of the 16th Annual European Symposium on Algorithms (ESA 2008)*, Karlsruhe, Germany, September 2008, vol. 5193 of *Lecture Notes in Computer Science*, pp. 356–367.

[41] Gary V. Yee, Brian Shucker, Joe Dunn, Anmol Sheth, and Richard Han, "Just-in-time sensor networks," in *Third Workshop on Embedded Networked Sensors (EmNets 2006)*, Cambridge, MA, May 2006.

[42] Mohamed F. Younis and Kemal Akkaya, "Strategies and techniques for node placement in wireless sensor networks: A survey," *Ad Hoc Networks*, vol. 6, no. 4, pp. 621–655, 2008.

[43] Hai Liu, Amiya Nayak, and Ivan Stojmenović, "Fault tolerant algorithms/protocols in wireless sensor networks," in *Handbook of Wireless Ad Hoc and Sensor Networks*. Springer-Verlag, To appear.

[44] D.M. Blough, M. Leoncini, G. Resta, and P. Santi, "On the symmetric range assignment problem in wireless ad hoc networks.," *Proceedings of the 2nd IFIP International Conference on Theoretical Computre Science (TCS)*, pp. 71–82, 2002.

[45] G. Calinescu, I.L. Mandoiu, and A. Zelikovsky, "Symmetric connectivity with minimum power consumption in radio networks," in *Proceedings of 17th IFIP World Computer Congress*, pp. 119–130. 2002.

[46] Lefteris M. Kirousis, Evangelos Kranakis, Danny Krizanc, and Andrzej Pelc, "Power consumption in packet radio networks," *Theoret. Comput. Sci.*, vol. 243, no. 1-2, pp. 289–305, 2000.

[47] James D. McLurkin, *Analysis and Implementation of Distributed Algorithms for Multi-Robot Systems*, Ph.D. thesis, Massachusetts Institute of Technology, 2008.

[48] A. Frank and E. Tardos, "An application of submodular flows," *Linear Algebra Appl.*, vol. 114/115, pp. 329–348, 1989.

[49] S. Khuller and B. Raghavachari, "Improved approximation algorithms for uniform connectivity problems," *J. Algorithms*, vol. 21, no. 2, pp. 434–450, 1996.

[50] Harold N. Gabow, "A representation for crossing set families with applications to submodular flow problems," in *Proceedings of the Fourth Annual ACM-SIAM Symposium on Discrete Algorithms (Austin, TX, 1993)*, New York, 1993, pp. 202–211, ACM.

[51] Hector Garcia-Molina, "Elections in a distributed computing system," *IEEE Transactions on Computers*, vol. 31, no. 1, pp. 48–59, 1982.

[52] Dimitri Bertsekas, *Network Optimization: Continuous and Discrete Models*, Athena Scientific, Nashua, NH, 1998.

[53] David Moore, John Leonard, Daniela Rus, and Seth Teller, "Robust distributed network localization with noisy range measurements," in *SenSys '04: Proceedings of the 2nd international conference on Embedded networked sensor systems*, New York, NY, USA, 2004, pp. 50–61, ACM Press.

[54] Volkan Isler, Sampath Kannan, and Kostas Daniilidis, "Sampling based sensor-network deployment," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, Sendai, Japan, Sept. 2004, pp. 172–177.

[55] Mathew D. Penrose, "On k-connectivity for a geometric random graph," *Random Struct. Algorithms*, vol. 15, no. 2, pp. 145–164, 1999.

[56] G. Kortsarz, R. Kraughgamer, and J. L. Lee, "Hardness of approximation for vertex-connectivity network-design problems," in *Proceedings of 5th International Workshop on Approximation Algorithms for Combinatorial Optimization (APPROX)*, pp. 185–199. 2002.

[57] S. Arora, "Polynomial time approximation schemes for Euclidean traveling salesman and other geometric problems," *J. ACM*, vol. 45, no. 5, pp. 753–782, 1998.

[58] J. S. B. Mitchell, "Guillotine subdivisions approximate polygonal subdivisions: a simple polynomial-time approximation scheme for geometric TSP, k-MST, and related problems," *SIAM J. Comput.*, vol. 28, no. 4, pp. 1298–1309 (electronic), 1999.

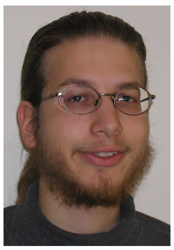
[59] A. Czumaj and A. Lingas, "A polynomial time approximation scheme for Euclidean minimum cost k-connectivity," in *Automata, languages and programming (Aalborg, 1998)*, vol. 1443 of *Lecture Notes in Comput. Sci.*, pp. 682–694. Springer, Berlin, 1998.

[60] Jianer Chen, Anxiao Jiang, Iyad A. Kanj, Ge Xia, and Fenghui Zhang, "Separability and topology control of quasi unit disk graphs," in *Proceedings of the 26th IEEE International Conference on Computer Communications (INFOCOM)*, May 2007, pp. 2225–2233.

Jonathan Bredin received the B.S.E. degree in computer science engineering from the University of Pennsylvania in 1996 and the Ph.D. degree in computer science from Dartmouth College in 2001.

He is an Associate Professor of Mathematics and Computer Science at the Colorado College. His research interests involve electronic commerce and mobile computing.

Prof. Bredin is a member of ACM, SIGCSE, SIGECOM, SIGART, and AAAI.



Erik D. Demaine received the B.Sc. degree in computer science from Dalhousie University, Halifax, Canada, in 1995, and the M.Math. and Ph.D. degrees in computer science from University of Waterloo, Canada, in 1996 and 2001, respectively.

He is an Associate Professor in computer science at the Massachusetts Institute of Technology. His research interests range throughout algorithms, in particular, computational geometry, data structures, graph algorithms, and recreational algorithms.

Prof. Demaine is a member of ACM, AMS, CMS, MAA, and SIAM. He received a MacArthur Fellowship (2003) and Alfred P. Sloan Fellowship (2006).

MohammadTaghi Hajiaghayi received the Bachelor's degree in Computer Engineering from Sharif University of Technology in 2000. He received the Master's degree in Computer Science from the University of Waterloo in 2001. He received the Ph.D. degree in Computer Science from the Massachusetts Institute of Technology in 2005.

He is a Senior Researcher in the Algorithms and Theoretical Computer Science group at AT&T Labs – Research. In addition, he holds a Research Affiliate position in MIT Computer Science and Artificial Intelligence Laboratory (CSAIL). He is also a Permanent Member of the Center for Discrete Mathematics and Theoretical Computer Science (DIMACS) at Rutgers. His research interests include algorithmic graph theory, combinatorial optimizations and approximation algorithms, distributed and mobile computing, computational geometry and embeddings, game theory and combinatorial auctions, and random structures and algorithms.

Daniela Rus is a professor in the EECS Department at MIT. She is the co-director of the CSAIL Center for Robotics. Previously, she was a professor in the computer science department at Dartmouth College. She holds a Ph.D. degree in computer science form Cornell University. Her research interests include distributed robotics, mobile computing, and self-organization. She was the recipient of an NSF CAREER award and an Alfred P. Sloan Foundation Fellowship. She is a class of 2002 MacArthur Fellow.