ANCHOR - Self-Configuring Robotic Network

Harris Chi Ho Chiu and Wei-Min Shen

***Abstract*— A challenging task for a robotic radio network is to establish the connectivity among a set of entities (humans or radio nodes) in an unknown environment with minimum number of robots. The main difficulty is that the locations of the entities and the radio connectivities between the physical locations are not known in advance. We represent the problem by a topological graph of locations with known access links but unknown radio links (to be discovered dynamically) and develop a novel *ANCHOR* algorithm for a set of autonomous robots to self-organize a radio network to connect the given entities with the least number of robots. We show in simulation analysis the algorithm scale near-linearly with maximum number of connected hops away from the terminals. Experiment also shows performance improvement with increasing number of robots.**

I. Introduction

Communication between rescuers and outdoor ground units are critical for search and rescue operations. Trapped mine workers have to communicate with the ground station for their needs and situation through wireless devices. Firemen can continuously be acknowledged if the building is likely to collapse. Due to the unknown radio propagation characteristics in indoor environment, reconfigurable robotic wireless network is essential for such operations. This network consists of mobile robots with the self-organizing ability to establish, maintain and self-heal radio connectivity between critical entities, like victims, rescuers and ground stations with *intelligent radio* - radio devices with certain computational power. Often, the locations of these entities are dynamic and unknown to the robots prior to rescue operations. One critical goal is to constantly connect to as many rescuers as possible to deliver critical information. This problem can be divided to two subproblems: *map discovery problem* and *radio link discovery problem*. *Map discovery problem* has been heavily studied in the field of simultaneous localization and mapping (SLAM) [2][8]. Figure 2 shows the floor plan of 9th floor of Information Sciences Institute mapped into a topological map with nodes represent a detectable feature like doors.

This paper focuses on the *radio link discovery problem*: *Given n robots, the topological map - graph G_1 and a set of terminals T with one specified gateway, how can asynchronous radio-equipped robots position themselves to establish a minimal network connecting T with only the ability of self-localization?*

This problem arises challenges in robot placement and coordination as pre-computation of motion plans requires information about radio linkages between locations and the locations of the terminals and other robots. This paper contributes an algorithm to deploy initial robotic network without any information about radio characteristics. It provides a centralized approach to discover all essential radio links for connections of all terminals and forms a network with minimum number of robots. The algorithm returns no optimal solutions but partial solutions for cases with insufficient number of robots.

Harris C.H. Chiu and Wei-Min Shen are with Information Sciences Institute, The University of Southern California, Los Angeles, U.S.A. chichiu@usc.edu and shen@isi.edu

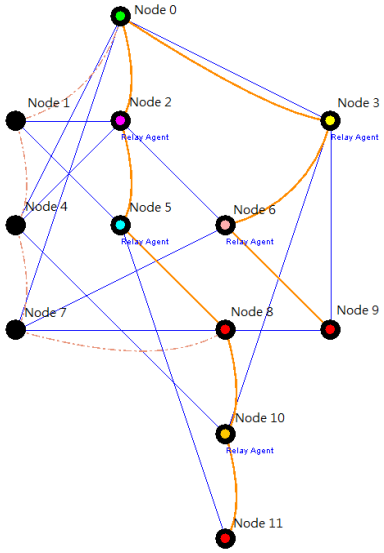

Fig. 1. Graph notation of a robotic radio network problem. Black: nodes in V; Green: Stationary radio at N_{T1} node; Red: Stationary radio at a terminal node N_{Ti}; Other colors excluding black: robots. Blue: traversability edges in E_1; Orange: radio link edges (solid: currently used by robots, dotted: potential links)

II. Problem Representation

The environment is modeled into graph notation for its generality. Graph $G_1 = (V, E_1)$ denotes the map of topological features in the environment and each node has its corresponding identifier, where $V = \{N_1, ..., N_k\}$ and k is the identifier of corresponding node. A node in V represents a grid in a grid map or a feature in the environment. An edge in E_1 indicates robots traversability between two featured locations. A radio connectivity graph $G_2 = (V, E_2)$ represents the radio connectivity between locations. If two nodes are connected by an edge in E_2, robots and terminals situated on both end can *reliably* communicate to each other. There are initially i terminals with radio at node $\{N_{T_1}, ..., N_{T_i}\} \subset V$. There are also n robots $\{R_1, ..., R_i\}$ in random starting locations. The problem definition becomes:

Given each robot R_i: (1) An ability to move from its current node to a neighbor node through an edge in E_1 (2) A radio that sends and receives messages at the current node to and from its 1-edge neighbor through edges in E_2 (3) G_1 and a self localizing function $getCurrentNode()$ returns an identifier for the current node, a robot terminates when it realizes its node in a minimal subgraph of G_2 connecting N_{Ti} for all i. This would result in a global objective: *A network with the use of minimal number of robots or a partial network connecting the terminals for insufficient robots*. Figure 1 shows an instance of the graph notation with minimal network connecting four terminals ($Node$ $0, 8, 9$ and 11) with robots at $Node$ $2, 3, 5$ and 10.

III. Related Work

Robotic wireless network has been a popular research field. Researches have been carried out in the area of self-healing[19], connectivity maintenance[10][1], connectedness optimization[16][7],area coverage[6][9] and formation control for exploration [14][17][12]. However, most of the approaches cannot be directly applied to our problem with the constraints of unknown radio model. Self-Healing networks by Zhang et al.[19] requires robots to stay connected throughout the self-healing process in an open area. The connectivity requirement is also true for connectivity maintenance and connectedness optimization. Area coverage aims at maximizing robots connectivity coverage, but it does not optimize for the use of minimum number of robots. Howard et al.[9] establish connectivity between any points in the area by covering the unknown area incrementally, which requires certain number of robots.

In dealing with unknown radio model, Hsieh et al. [18] generates minimum movement plans for robots to discover radio connectivity for user-specified locations. However, to obtain a minimal network, it requires to specify every combination of locations. Also, due to the large number of possible connections, algorithms for two and three robots are demonstrated. Our previous work[4] addresses the network deployment problem without any map information and localization capability of the robots by forming *"tentacle"* - {*a series of stationary robots*} from the terminals and the ground station with directional radio guidance. However, a lower-bound of the convergence time for tentacles to meet cannot be guaranteed. Also, it uses greedy approach to commit robots to currently found radio connectivity. Therefore, the feasibility of using optimal number of robots for network deployment is uncertain. The paper is organized as follows: In Section IV, ANCHOR algorithm is introduced to discover essential radio links and connectivity paths on the graph representation. In Section V, we shows the algorithm is complete and analyze with different problem parameters in effect of the convergence time through simulation experiments. We conclude our paper and provide possible future work in Section VI.

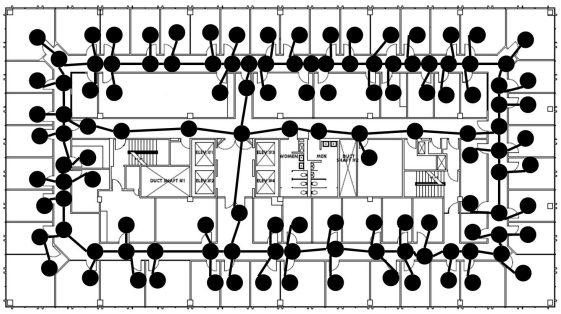

Fig. 2. Graph representation of the floor plan of 9th floor of Information Sciences Institute, Marina Del Rey

Algorithm 1: The ANCHOR algorithm for Robots

Data: G_1 , *getCurrentNode()*
Result: Idle at locations of minimal network or gateway if network does not exist

```
InitPhase();
if role is COOR then
    SearchPhaseCoor();
    Deploy();
else
    SearchAndDeployNonCoor();
```

IV. The ANCHOR algorithm

A. Overview

The ANCHOR algorithm discovers unknown radio links (named *hidden edges* in the paper) incrementally from a specific terminal named as *gateway*), e.g T_1. The general strategy is to perform Breadth First Search (BFS) [5] from the gateway on graph G_2 with unknown edge set E_2. Recall that a BFS algorithm keeps a priority queue for the candidate nodes to be expanded in ascending depth level from the root node. It expands (dequeues) one candidate node from the priority queue and enqueue its unvisited children to the queue. However, for graph G_2 with unknown E_2, the next breadth level nodes are unavailable. To discover all the next depth level children of one node, one robot must stay temporarily at the expanding candidate node while another robot visits every other untested node of the graph at least once. The stationary robot is said to be in *anchor mode* because this robot anchors itself as if a circle-drawing compass. The moving robot finds all hidden edges that is one-edge away from where the stationary robot situates. The ANCHOR algorithm carries out three phases sequentially - the initial phase, the search phase and the deploy phase. Algorithm 1 and 3 shows the main algorithm algorithm for robots and terminals. Table I gives brief descriptions of the message types and variables used in the pseudo code of ANCHOR algorithm.

1) Initial Phase: Initially, each robot at random locations generates a tour in the physical graph G_1 to visit every node at least once and return to the starting node. As *Hamiltonian*

TABLE I
VARIABLES FOR ANCHOR ROBOTS

Variable \ Function	Description
role	Current role of terminals and gateway - {*GATEWAY,TERMINAL,ROBOT, COOR, ANCHOR*}
step	Current step-*Discover, Migrate and Settle* in the search phase for the coordination robot
isIdle	True if current robot should move with function *goNextNode()*
phase	Current phase of the algorithm
discoverWalk, migrateWalk, settleWalk	A list stores the nodes for coordination robots to walk in *Discover, Migrate and Settle* step respectively
links	A set contains link edges with pairs of the identifier of the nodes, where ($NodeID_a$, $NodeID_b$) is the same as ($NodeID_b$, $NodeID_a$)
candAnchor	A priority queue stores candidate node identifier according to the depth level (hops away from gateway)
prevCandAnchor	A set stores candidate nodes which has allocated a robot to migrate to it
goNextNode(n)	The first node is popped from the queue n and move to it
GenApprox-TSPTour(P)	It returns a approximated tour to visit nodes in set P at least once

Message type	Description
INIT	Contains $< visitedNode >$Initial message that the robot sends to a gateway
INIT_ACK	Contains field $coor$ in reply whether the recv'ing robot should be the coordination robot
PROBE	Field $srcNodeID$ contains the node ID where the sender is
DISCOVERY_START	Indicates the start of $Discover$ step in Search Phase
DISCOVERY _COMPLETE	Contains field $links$ stores all the radio edges discovered in step $Discover$
Settle_COMPLETE	Indicates all migrating robots has settled in its desired position
MIGRATE	Contains field $dest$ for destination
DEPLOY	Activates the terminal to find the MinST based on the collected $links$ information
MoveTo	Contains $phase$ and destination $dest$ information about where robot should move to

cycle does not always exist in the graph, the walk can be generated using a distance metric graph G_3. A distance metric graph G_3 is a fully connected graph $G_3 = (V, E_3)$, where each cost of each edge (u, w) is the shortest distance between node u and w in graph G_1. This graph can be calculated using Dijkstra's algorithm. [5]. To obtain lowest cost tour, solutions to *Traveling Salesman Tour (TST)* on graph G_3 can be applied. Due to its NP-Completeness, approximation algorithms [5] can be used to compute a close to optimal solution. Function *GenApproxTSPTour(P)* in Algorithm 2 returns a 2-approximated path visiting every node in set P at least once from current node. The first arrival robot becomes the coordination robot and starts the *search phase*. Algorithm 2 and line 6 - 9 of Algorithm **??** gives on how robots and gateway are interacted during the initial phase.

Algorithm 2: InitPhase() for Robots

Data: G_1 , *getCurrentNode()* - ability of self-localization,
Result: Idle at $N_{Gateway}$, first robot's $role$ =COOR

```
1  graphWalk ← GenApproxTSPTour(V);
2  while true do
3      visitedNode ← visitedNode ∪ getCurrentNode();
4      if At N_Gateway and recved Probe msg from Gateway then
5          reply INIT message ;
6          wait for INIT_ACK;
7          if INIT_ACK.coor is 1 then role ← COOR;
8          isIDLE ← true;
9          return // quit Init phase
10     goNextNode(graphWalk);
```

2) Search Phase: Three steps are performed in search phase - *Discover, Migrate and Settle*. The *Discover* step searches for radio links (edges in E_2) connected to current *anchor* by generating route to visits all nodes that has not been radio link tested $\{V \backslash \{msg.visitedNode\}\} \cup T_1$. In other words, nodes that are previously anchored are not included in the list. Line 7 - 9 of Algorithm 4 shows how edges are found and stored. The discovered links information is then reported to the gateway at the end of the walk in line 4.

The *Migrate* step aims at moving idle robots or anchored robots into candidate anchor nodes in the priority queue *candAnchor* for the next depth level link discovery. First, idle robots are moved to the candidate anchoring positions by the coordination robots. If the priority queue is not empty, previously anchored robot are also moved to the candidate positions. Line 20 of Algorithm 3 shows the condition a robot should have migrated to the anchor position. In addition, terminals can become an anchor if its location matched on of the candidate anchor position and the terminal quits its anchor mode after receiving *Migrate* message with non-current node destination. (see line 16 of Algorithm 3)

The *Settle* step is used to synchronize all migrating robots to assigned anchor node if the synchrony of arrival of robots cannot be guaranteed. Due to the reality that robots might not be able to be in its anchor position before the coordination robot starts the next coordination walk, the robot visits each assigned candidate anchor node until the corresponding robot has arrived.

3) Deploy Phase: During the Deploy phase, the gateway computes the minimum set of nodes needed based one all retrieved essential link information for all terminals. In graph theory literature, the Minimum Steiner Tree (MinST) problem [13] is to find a sub-graph of a graph G with minimum total edge cost given specified terminal nodes. The problem of obtaining the final set of nodes for the minimal network is equal to compute MinST in the discovered G_2 with unit edge cost in E_2. Bruandggemann et. al[3] has given

Algorithm 4: SearchPhaseCoor() for Robots

```
Data: G_1 , getCurrentNode()
Result: Robots at N_Gateway
while true do
    switch step do
        case Discover
            if isIdle is true and received DISCOVERY_START msg then
                discoverWalk ← GenApproxTSPTour( {V\{msg.visitedNode}} ∪ T_1);
                isIdle ← false; prevCandAnchor ← assigned node in candAnchor; candAnchor ← ∅;
            if received Probe msg then
                links ← links ∪ <currNode.id,msg.srcNodeID>;
                candAnchor ← candAnchor ∪ msg.srcNodeID;
            goNextNode(discoverWalk);
            if discoverWalk is empty then
                isIdle ← false;
                if candAnchor is ∅ then
                    send DEPLOY to Gateway;
                    return;
                else send DISCOVERY_COMPLETE to Gateway;
                step ← Migrate;
                migrateWalk ← GenApproxTSPTour(prevCandAnchor);
        case Migrate
            if received Probe msg and msg.srcNodeID ∈ prevCandAnchor OR msg.isIdle is true then
                if msg.isTerminal is true then
                    reply MoveTo with dest = msg.srcNodeID;
                else
                    reply MoveTo with dest = unassigned node ∈ {candAnchor\T};
                if all nodes in candAnchor are assigned then
                    reply MoveTo with dest = N_Gateway;
            goNextNode(migrateWalk); if migrateWalk is empty then
                step ← Settle;
                SettleWalk ← GenApproxTSPTour(candAnchor);
        case Settle
            goNextNode(SettleWalk);
            wait for Probe msg with msg.srcNodeID ∈ candAnchor;
            if SettleWalk is empty then
                send Settle_Complete to gateway;
                step ← Discover;
```

a proof on equivalence of two problems. It is well-known that the problem of finding MinST is NP-Complete. Dreyfus-Wagner algorithm [13] applies dynamic programming to return solution in worst case time complexity $O(3^k n + 2^k n^2 + n^2 log n + nm)$, where there are k terminals, n nodes and m edges. For approximated solution, Robins et al. [15] provide an algorithm with current best approximation factor of 1.55. Similar to migrating robots to anchor position, the coordination robot commands idle or anchoring robots to the desired nodes through shortest physical path. However, locations in shortest terminal-to-terminal connectivity path first get robot assignments. This allows some terminals to get connected first. If there is insufficient robots, the coordination robot acknowledges to the gateway and other terminals that minimal network is not possible.

V. Analysis of the ANCHOR algorithm

A. Completeness

The algorithm coordinates robots based on Breath First Search mechanism. As BFS is complete, the algorithm is complete in searching all the necessary edges connecting to all terminals. The algorithm is complete as it is able to return

Algorithm 3: ANCHOR algorithm for Terminals

```
Data: Nil
Result: All essential links information for all
        terminals (Gateway only)
isANCHOR ← true;
while true do
    Broadcast PROBE Messages to 1-hop neighbor
    connected in E2;
    if received msg then
        switch msg.type do
            case INIT
                if First received then
                    replyMsg(INIT_ACK, coor=1);
                else
                    replyMsg(INIT_ACK, coor= 0);
            case DISCOVERY_COMPLETE
                links ← links ∪ msg.links;
            case Settle_COMPLETE
                reply DISCOVERY_START;
            case MIGRATE
                if currNode is msg.dest then
                    isANCHOR ← true;
                else
                    isANCHOR ← false;
            case DEPLOY
                minNodeSet ←
                calcMinSteinerTree(links);
                replyMsg(minNodeSet);
                return
```

Algorithm 5: SearchAndDeployNonCoor()

```
Data: G1 , getCurrentNode() - ability of
      self-localization,
Result: Robots idle at N_Gateway, first robot's role
        =COOR
while true do
    if isAnchored is true then
        Broadcast PROBE Messages to 1-hop
        neighbor connected in E2;
        if received MoveTo msg then
            isAnchored ← false;
            phase ← msg.phase;
            walkPath ← GenShortestPath(msg.dest);
    else
        if walkPath is empty then
            if phase is Migrate then
                isAnchored ← true;
            status ← IDLE;
        goNextNode(walkPath);
```

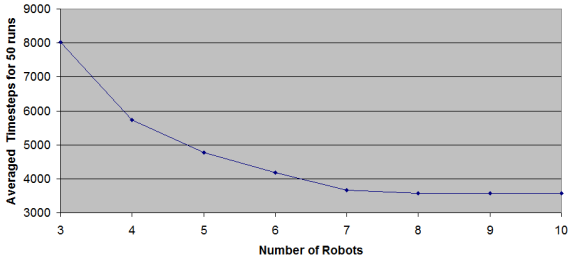

Fig. 3. Performance change with increasing number of robots with 50-Node random graphs and same set of hidden edges

no optimal solutions based on the edges calculated through calculation for the Minimum Steiner Tree.

B. Scalability in Number of Robots

To investigate the scalability in number of robots, the ANCHOR algorithm and the graph representation have been implemented using Java-based simulation toolkit MASON [11]. *While* loop is performed once for each time step and messages can be sent and received in every time step. However, robots moves in every two time steps. In other words, *goNextNode(node)* only executes in every other loop. Experiments are performed on randomly generated 50-node graphs and a set of randomly generated radio link edges among terminals. Each experiment is dedicated for a specific number of robots with 50 runs of the simulation on the same graphs from configuration of randomly deployed robots to convergence - with or without minimal network. Eight experiments are performed with the number of robots varying from three to ten. Figure 3 shows a decreasing trend of convergence time with the increase number of robots. This phenomena can be explained by the availability of robots for anchor compared with the length of priority queue *candAnchor*, which is the breadth of the radio edge tree found from the gateway. When more robots are available, more robots can commit to become an anchor and therefore the coordination robot can discover links with several anchors in parallel. However, since there is always a limit of the breadth of the radio connectivity graph. Therefore, the performance improvement is gradually lowered with the increment of robots.

C. Depth of Radio Connectivity Graph E_2

Each *Discover-Migrate-Settle* cycle for coordination robot discovers one depth level increment to current node with anchored robots. It is interesting to investigate on how maximum depth level from the gateway relate to the algorithm performance. To illustrate this parameter, in Figure 1, the depth level of {Node 1,2,3}, {Node 4,5,6}, Node {Node 7} and {Node 10}, , are 1, 2, 3 and 4 respectively. Note that Node 10 belongs to two connectivity paths and the lower value becomes its depth level. 900 random 50-Node graphs are generated and ten robots are involved in

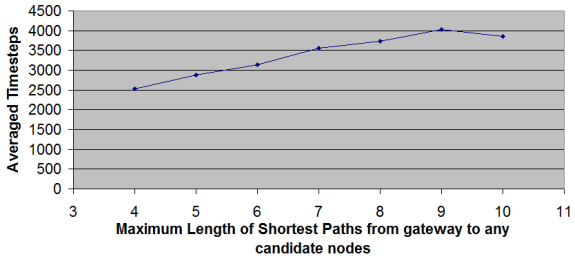

Fig. 4. Performance change with the increase of depth level from the gateway

the each experiment. The maximum depth value of each graph is evaluated and the corresponding convergence time is recorded. As shown in Figure 4, the convergence time (in terms of time steps) increases in a nearly linear way. The reason is that one more *Discover-Migrate-Settle* cycle is needed for the coordination robot when there the depth level is increased by one.

VI. CONCLUSIONS AND FUTURE WORK

This paper introduced a graph representation for multi-robot establishment problem with stationary terminals. This graph representation gives a metric on measuring how well robots self-organize to connect terminal radios to the gateway radio with minimum number of agents. ANCHOR algorithm gives anchor searching mechanism to discover all essential radio links to the gateway radio and obtain minimum number of robots through reconstruction from the discovered paths. Simulation experiments have been performed and the depth level of possible radio route could affect the performance of the algorithm. The experiment also shows multi-robot coordination improves overall convergence time.

Such algorithm serves as a stepping stone to solve the problem without a priori map information by integrating it with SLAM approaches to gather map information and localization. The robustness to SLAM error is our next research issues to investigate. We believe that with proper translation of physical map to graph translation, such as representing large enough area. The effect of localization error is not significant to the result.The rich graph representation also lays a pathway to various future work. For example, values of the visible edges can be used as a metric for total energy consumption minimization. Values of the hidden edges can be treated as bandwidth values for the selection of optimal throughput paths from gateway radio to all terminal radios. Adding relationship between visible edges and hidden edges from a radio signal model as heuristics if radio model can be reliably used in the environment.

VII. ACKNOWLEDGMENTS

We would like to give thanks to the discussion and suggestion of our colleagues at Polymorphic Robotics Laboratory at the Information Sciences Institute of the University of Southern California and also the constructive feedbacks from reviewers from major conferences.

REFERENCES

[1] Stuart Anderson, Reid Simmons, and Dani Goldberg. Maintaining line of sight communications networks between planetary rovers. In *Proceedings of the 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems, (IROS 2003)*, volume 3, pages 2266 – 2272, October 2003.

[2] Tim Bailey and Hugh Durrant-Whyte. Simultaneous localisation and mapping (slam): Part ii the state of the art. *IEEE ROBOTICS AND AUTOMATION MAGAZINE*, 2, 2006.

[3] B. Bruandggemann and D. Schulz. Coordinated navigation of multi-robot systems with binary constraints. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pages 3854 –3859, 2010.

[4] Harris Chi Ho Chiu, Bo Ryu, Hua Zhu, Pedro Szekely, Rajiv Maheswaran, Craig Rogers, Aram Galstyan, Behnam Salemi, Mike Rubenstein, and Wei-Min Shen. Tentacles: Self-configuring robotic radio networks in unknown environments. In *IROS 2009*, St. Louis, MO, October 2009.

[5] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to algorithms*. McGraw-Hill Book Company, Cambridge, London, 2. edition, 2001. 1. editon 1993.

[6] Nikolaus Correll, Jonathan Bachrach, Daniel Vickery, and Daniela Rus. Ad-hoc wireless network coverage with networked robots that cannot localize. In *ICRA'09: Proceedings of the 2009 IEEE international conference on Robotics and Automation*, pages 3554–3561, Piscataway, NJ, USA, 2009. IEEE Press.

[7] M. DeGennaro and A. Jadbabaie. Decentralized control of connectivity for multi-agent systems. In *In 45th IEEE conference on decision and control(pp. 3628-3633)*, 2006.

[8] Hugh Durrant-Whyte and Tim Bailey. Simultaneous localisation and mapping (slam): Part i the essential algorithms. *IEEE ROBOTICS AND AUTOMATION MAGAZINE*, 2:2006, 2006.

[9] Andrew Howard, Maja J Mataric, and Gaurav S Sukhatme. Mobile sensor network deployment using potential fields: A distributed, scalable solution to the area coverage problem. In *Proceedings of the 6th International Symposium on Distributed Autonomous Robotics Systems (DARS02)*, pages 299–308, 2002.

[10] M. Ani Hsieh, Anthony Cowley, Vijay Kumar, and Camillo J. Taylor. Maintaining network connectivity and performance in robot teams: Research articles. *J. Field Robot.*, 25(1-2):111–131, 2008.

[11] Sean Luke, Claudio Cioffi-Revilla, Liviu Panait, Keith Sullivan, and Gabriel Balan. Mason: A multiagent simulation environment. *Simulation*, 81(7):517–527, 2005.

[12] David Payton, Regina Estkowski, and Mike Horward. Progress in pheromone robotics. *Intelligent Autonomous Systems*, 7:256–264, 2002.

[13] Hans Jrgen Prömel and Angelika Steger. *The Steiner Tree Problem*. Vieweg-Verlag, Feb 2002.

[14] Joshua Reich, Vishal Misra, Dan Rubenstein, and Gil Zussman. Spreadable connected autonomic networks (scan). Technical Report CUCS-016-08, Columbia University, 2008.

[15] Gabriel Robins and Alexander Zelikovsky. Tighter bounds for graph steiner tree approximation. *SIAM J. Discret. Math.*, 19:122–134, May 2005.

[16] E. Stump, A. Jadbabaie, and V. Kumar. Connectivity management in mobile robot teams. *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, pages 1525–1530, May 2008.

[17] Jose Vazquez and Chris Malcolm. Distributed multirobot exploration maintaining a mobile network. In *In Proceedings of Second IEEE International Conference on Intelligent Systems*, pages 113–118, 2004.

[18] Mong ying A. Hsieh, Vijay Kumar, and Camillo J. Taylor. Constructing radio signal strength maps with multiple robots. In *ICRA*, pages 4184–4189, 2004.

[19] Fei Zhang and Weidong Chen. Self-healing for mobile robot networks with motion synchronization. *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*, pages 3107–3112, 29 2007-Nov. 2 2007.