Discovering Near-Optimal Communication Network for Modular Robots

Harris Chi Ho Chiu and Wei-Min Shen

***Abstract*— Neighbor-to-Neighbor (Inter-Module) communication between modules using methods such as infrared, radio and bus-type connection are critical component for global behavior coordination. However, this type of communication method does not necessarily provide the shortest possible routes between modules and not making use of *Hidden Communication Shortcuts (HCS)*. A HCS is defined as a communication channel between two modules without involving any inter-module (non-HCS) communication. For example, instead of sending messages through all modules in a snake configuration, the head module can directly talk to the tail module through infra-red if they are physically close. This paper proposes a novel way to discover existing communication shortcuts distributedly and form a near-optimal network using these shortcuts to connect a set of given *terminal modules* based on inter-module communication. The size of the discovered network is at most two times larger than the optimal (minimal) ones. To make this approach more realistic, we assume that not all communication devices on the modules are turned on simultaneously and in fact no more than r modules are allowed to turn on devices for non inter-module communication at the same time to avoid interferences.**

Derived from a centralized network deployment approach of mobile robot network - ANCHOR [7], this paper provides (1) a distributed algorithm for HCS discovery among modular robots (2) with adaptation to heterogeneity as long as the modular robots are connected at least with neighbor-to-neighbor communication (3) a distributed approach to form a near-optimal network with at most two times the minimum number of modules (2-OPT) to activate their devices for *HCS*. We show that the convergence time on a balanced configuration is of a factor t, $O(\frac{n^2}{t})$, where n and t are the number of modules and terminal modules respectively, but it remains $O(n^2)$ for the total number of activation and deactivation of devices for HCS. In simulation, we also show the improvement of convergence time as r increases.

I. Introduction

Inter-module communication, for example, neighbor-to-neighbor communication through infra-red[11][20][4][14] and radio [15], bus-type connection [16][24] are commonly used to exchange sensor information and coordinate motions and actions among modular robots. However, due to the limitation of some communication methods and uncertainty of surrounding environment, the network relying on inter-module communication may involve extra robots resulting in slower transmission rate and higher energy consumption. For example, in Figure 1, the head module of the snake configuration tries to communicate to the tail module and the modules are equipped with radio. In open space, the head can directly communicate with the tail without involving other modules. However, in an unpredictable environment, such as a tunnel as shown in Figure 1, radio might suffer from multi-path effect and the communication path cannot be well-established. In this case, it could require one or two more modules in the snake to help out shortcutting the communication path without involving every module. This raises a question on how can these *Hidden Communication Shortcuts (HCS)* be discovered to establish a smaller network for specified terminal modules. A HCS is defined as a communication channel between two modules without involving any inter-module communication. The communication devices involved are known as *HCS devices*. A HCS device could be any communication device that does not involve in inter-module communication, for example, infra-red and radio devices. To tackle the possibility of crosstalk, we assume that no more than r devices can be activated at the same time to prevent inference and cross talk can be detected through checksum and source/destination address validation. For example, the most conservative setting for infrared is $r = 2$ and this number could be higher for radio communication. In this paper, this parameter is a fixed input. In the future work, this could be determined dynamically with crosstalk detection information.

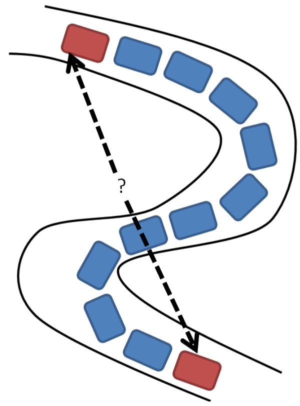


Fig. 1. A scenario for snake configuration: The head module tries to communication with the tail module in a tunnel environment. How many modules should be involved?

In our previous work [7], ANCHOR algorithm provides a centralized approach to discover radio links with mobile robots to connect all specified terminal modules. By applying Dreyfus-Wagner algorithm[1] on the discovered radio links to resolve *Minimum Steiner Tree Problem*[1], nodes for minimal robots network for deployment can be calculated and robots are deployed to the expected locations. Same as the robot placement problem, the selection of which modules to activate the communication device in minimal

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

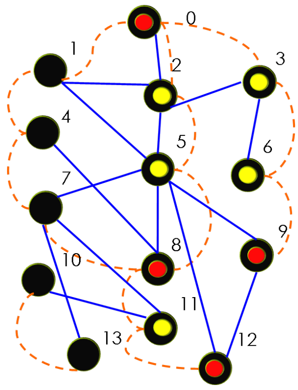


Fig. 2. Graph notation for Hidden Communication Shortcut (HCS) problem for modular robots. Black: nodes in V; Red: terminal modules N_{Ti}; Yellow: modules with device activations; Blue edges: physical connectivity with inter-module communication as in E_1; Orange dotted edges: Hidden Communication shortcuts (undiscovered initially).

number is also the same as *Minimum Steiner Tree Problem*, which is a NP-Complete problem. Instead of focusing on resolving this NP-Complete optimality problem distributedly, we are interested in the near-optimal (2-approximation or good enough) solution in polynomial time.

The problem is defined as: *Given the topology of the modular robots in graph G_1, neighbor-to-neighbor inter-module communication, a set of expected* terminal *modules T, and no more than r modules can simultaneously activate HCS devices, which modules should activate HCS devices to form a network with at most two times (2-OPT) of them?*

The problem inherits similar challenges as in the mobile robot version. Pre-computation of plans requires information about HCS between modules and the locations of the terminal modules. The limitation of activation of HCS device on modules also imposes a challenge in resource management in deciding which pair of HCS devices should be activated for HCS discovery. This paper contributes a distributed algorithm which:

1) discovers HCS between modules despite of heterogeneity.
2) forms a near-optimal network with at most 2-OPT HCS devices activated on the modules.
3) converges in $O(\frac{n^2}{t})$ on a balanced topology, where n and t are the number of modules (nodes) and terminal modules respectively.
4) shows practicality of HCS discovery in simulation and improving performance as r increases.

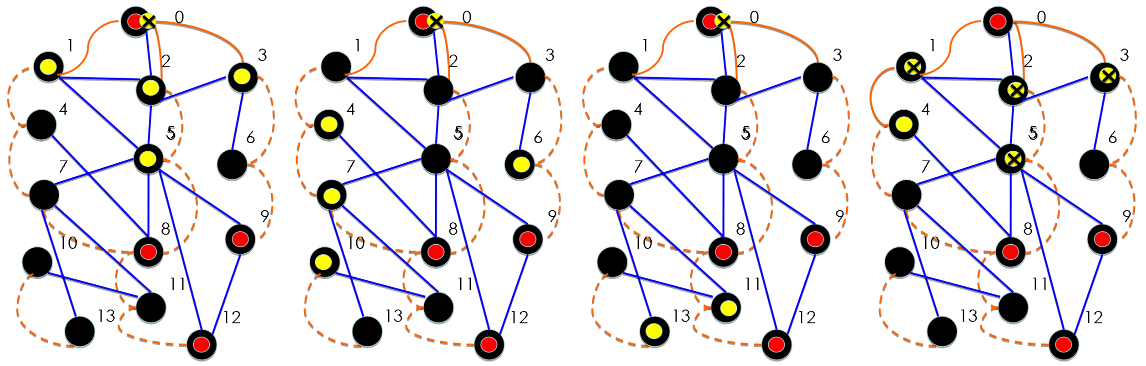
II. Problem Representation

Figure 2 shows an example of the problem presentation. Graph $G_1 = (V, E_1)$ denotes the topology of the modules, where $V = \{N_1, ..., N_k\}$ and k is the identifier of corresponding modules. A node in V represents a module equipped with or without HCS device represented in black node. An edge in E_1 (blue) indicates physical connectivity and their neighbor-to-neighbor communication of the modules. A HCS connectivity graph $G_2 = (V, E_2)$ represents the HCS connectivity among the modules. If two nodes are connected by an edge in E_2 (dotted orange edge), the modules are able to *reliably* communicate to each other through HCS. However, the modular robots do not have any initial knowledge about any HCS in E_2. There are initially i terminal $\{T_1, ..., T_i\}$ at module $\{N_{T_1}, ..., N_{T_i}\} \subset V$ as indicated by red dots. No more than r modules are allowed to activate HCS device at the same time and modules with activated HCS device are represented by yellow dot. The goal is to only activate the HCS devices on the modules to form a 2-OPT network without exceeding the number limitation on HCS device activation r. Figure 2 shows an instance of the graph notation with minimal network connecting four terminals ($Node$ $0, 8, 9$ and 11) with HCS device activations at $Node$ $2, 3, 5, 6$ and 10, which requires 5 robots. Another instance of 2-approximate solution is to activate HCS devices at $Node$ $1, 3, 4, 6, 7$ and 10 requiring 6 modules with HCR device activated $\leq 2 * 5 = 10$.

III. Related Work

In mobile robots literature, researches have been carried out in the area of self-healing[26], connectivity maintenance[13][2], connectedness optimization[21][9],area coverage[8][12] and formation control for exploration [18][22][17]. Self-Healing networks by Zhang et al.[26] 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.[12] establish connectivity between any points in the area by covering the unknown area incrementally, which requires certain number of robots. Vieira et al. [23] provide an algorithm to deploy minimal robot network with the assumption of known radio network model and sufficient number of robots. However, most of the approaches cannot be directly applied to our problem since prior HCS cannot be always obtained.

In dealing with unknown connectivity between mobile robots, Hsieh et al. [25] generate 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, only the case of two and three robots are demonstrated. Our previous work[6] addresses the network deployment problem without any map information, radio model information and localization capability of the robots by forming *"tentacle"* - $\{$*a series of stationary robots*$\}$ from the terminals and the ground station with coarse 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 possibility in obtaining optimal or near-optimal number of robots for network deployment is uncertain. Therefore, it cannot be directly adopted to HCS activation on modules.



(a) First Step of the First cycle (b) Second Step of the First Cycle (c) Third Step of the First Cycle (d) First Step of the Second Cycle. Node 1,2,3 and 5 become anchor

Fig. 3. Steps of HCS devices activations. Yellow crossed circle refers to an anchor with HCS device activated. Solid orange edges indicate discovered HCS.

In graph literature, Bruandggemann et. al[3] provided a proof of equivalence between the problem of finding minimum number of nodes with specified terminals and *Minimum Steiner Tree Problem* (MinSteinerT) with unit edge cost. It is well-known that the problem of finding MinSteinerT is NP-Complete. Dreyfus-Wagner algorithm [1] provides a solution by applying dynamic programming to return solution. For approximated solution, Robins et al. [19] provide an algorithm with current best approximation factor of 1.55. Distributed algorithm for 2-approximate Steiner Tree is also presented by Chalermsook et al.[5]. However, the algorithm for MinSteinerT does not always applicable since the knowledge of the edges (HCS) is not always known in advance.

In this paper, we present a distributed algorithm based on the searching approach from ANCHOR and provides parallel searching of HCS leading to faster average convergence time and also allow redundancy of multiple coordination agents to avoid single point of failure. The paper is organized as follows: In Section IV, we describe how ANCHOR algorithm can be modified to apply to modular robots. In Section V, we show how multiple terminal with modified ANCHOR algorithm can be used to discover essential HSC connectivity paths on the graph representation in parallel. In Section VI, we show the algorithm is complete and provide analysis on convergence time and total traversal steps. The HCS discovery phase has been implemented in simulation and is discussed in Section VII. We conclude our paper and provide possible future work in Section VIII.

IV. The HCS Discovery from One Terminal

The coordination of robots for radio link discovery in ANCHOR algorithm is based on a principle - a radio link can only be discovered by two radios at corresponding places. One robot with radio is commanded to stay while the coordination robot travels around places to discover radio links with the radio of the stationary robot.

Similarly, in this problem, we also have *anchored* (temporarily activated HCS device) module and modules with HCS activation to test connectivity. The terminal also known as *root terminal* is the first anchor, then the HCS devices on other modules are activated through neighbor-to-neighbor communication to test the connectivity.

Figure 3 shows an example of the HCS discovery process for one terminal at Node 0, where the maximum of simultaneous activation quota $r = 5$. The optimal solution is as shown in Figure 2. In each cycle, the terminal generates activation order of each node. Currently, there is no preference for this order, which could be optimized in the future. Each module keeps a hop count *distance vector* to each terminal. The distance is based on the lowest hop count from current node to the terminals through *both* E_1 (blue) and E_2 (orange edges).

The first cycle is shown in Figure 3(a), 3(b) and 3(c). Module 1,2,3 and 5 are first activated with terminal module T_0 (Node 0) being an anchor. In the first step, the HCS connections 0-1, 0-2 and 0-3 are discovered. The distance vector to module 1,2, 3 and 5 are then updated to 1,1,1 and 2 respectively. In the following step, module 4,6,7 and 10 are activated. Since there is no direct HCS connection from these modules to the achored module, no HCS connections is revealed.

In the next cycle as shown in Figure 3(d), the modules with lowest distance vector to the root terminal T_0 become an anchored module. The terminal would activate the HCS devices of the remaining modules to check the possibility of HCS connections. At the end of the second cycle, module 4,6,7,8 and 9 would have their distance vector for T_0 as 2,2,3,3 and 3. Similarly, at the third cycle, module 4,6,7 and 8 becomes the anchored modules with current lowest distance vector to T_0. The process terminates when no more modules requires to be anchored. In the single terminal case, it terminates when all modules have been anchored. To handle some modules without a HCS device (heterogeneous modules), the coordination agent can still continue the discovery by recording no HCS connections to the module.

V. The Proposed Algorithm

A. Overview

The general idea of the distributed algorithm is to perform the modified ANCHOR algorithm from previous section (Section IV) *for each terminal* in parallel and yield 2-approximate minimal robotic network by having the terminals to activate the HCS devices at the desired modules. Since the problem restricts no more than r HCS devices can be activated at the same time, this value named as *HCS activation quota (HCS-AQ)* are first distributed among the terminals. Each terminal initially performs a depth first search algorithm through inter-module communication to obtain current number of terminals in the configuration. Each terminal is assigned with at least $\lfloor \frac{r}{t} \rfloor$ HCS-AQ, where t is the number of terminals. In addition, as the minimum number of identifying a HCS connection is two, we assume that $\lfloor \frac{r}{t} \rfloor \geq 2$ so that each terminal is able to perform the HCS discovery. In the following description, *module* and *node* are used interchangeably.

1) The HCS Discovery Phase: In this phase, the modified algorithm from Section IV is executed by each terminal to discover HCS connections in parallel. The goal is to let each module to know their distance vector to various terminal and hence decide which terminal is the nearest. For example, in Figure 2, at the end of the phase, the terminals at Node $0, 8, 9$ and 12 should have a *nearestNodeSet* of $\{$Node $1, 2$ and $3\}$, $\{$Node $4, 5, 7, 10, 11$ and $13\}$, $\{$Node $6\}$ and $\{$null$\}$ respectively. If there is tie in distance, the node always goes with a terminal with a smaller identifier.

To coordinate among the parallel searches, each module stores a variable named*rootTerminal*. *rootTerminal* contains the identifier of a terminal is currently the nearest, that is with the lowest hop count. If a node has not been anchored before,*rootTerminal* is equal to the terminal requesting the module to be an anchor. Each search is independent unless (1) a terminal requests a node to be anchored has a smaller distance vector compared to the distance vector of the *rootTerminal* of the node. Then, the *rootTerminal* of the node is reassigned to the requesting terminal. (2) The terminal T_j activates a node N_k and connects to an anchor N_a with root terminal T_j. Denote the distance vector of N_k for terminal T_j is $N_k.dv[T_j]$. The distance vector is updated in the following situations:

1) *If N_k does not have a root terminal:* $N_k.dv[T_j] = N_a.dv[T_j] + 1$
2) *If N_k has a root terminal T_i:* $N_a.dv[T_i] = N_k.dv[T_i] + 1$ and $N_k.dv[T_j] = N_a.dv[T_j] + 1$

At the end of the phase, each terminal broadcasts a *Search_Complete* message. In reply to the message, if the sending terminal has the lowest distance vector for a module, the module sends a *Join message* containing distance vector and the neighbor information (Module ID can be connected through either E_1 or E_2) to the broadcasting terminals. The terminal received *Join message* would add the module ID in the *nearestNodeSet*. In case, a module received a change of *rootTerminal*, the module would send the *Join* message to the new root terminal and an update *Unjoin* to the previous terminal module. Each terminal determines the discovery phase ends until it receives all *Search_Complete* from other terminals and a timeout for the last *Search_Complete* message received for as long as the time necessary for a message traveling the number of hops equal to the number of modules.

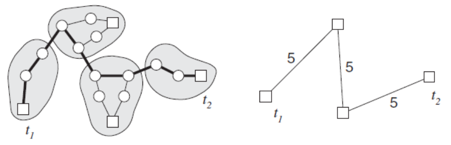


Fig. 4. Graph decomposition by Chalermsook et al.[5]. (a) Original graph with cluster decomposition. (b) A new graph formed through *inter cluster edge*

2) Activation Phase: The *Activation* phase starts right after the end of the HCS discovery phase. In computing a 2-approximate Steiner Tree, for a centralized approach, a all-distance graph of the terminals $D_1 = (T, E, l_e)$ has to be constructed and the approximated Steiner tree is the minimum spanning tree on the graph D_1. The all-distance graph is a complete graph consisting of all terminal nodes T with l_e as the cost of the shortest path between two terminals. However, to compute it distributedly with information from HCS discovery phase, our activation strategy is based on (1) the generation of all-distance graph decomposition proposed by Chalermsook et al [5], and (2) application of finding Minimum Spanning Tree on a all-distance graph distributedly to obtain which modules should activate their HCS devices.

B. Constructing a All-distance Graph distributedly

Chalermsook et al. has proposed a new graph decomposition $D_2 = (T, E_{new}, l_{new})$ by assigning each node in the graph to its nearest terminal (hop-distance). These nodes together with their own assigned terminal form a *cluster*. The edges connects clusters are named inter-cluster edges. Figure 4(a) from their paper indicates the cluster formation in the graph. The edges E_{new} connect every two terminals t_i, t_j if there is inter-cluster edge (u, v) connecting their own cluster. The cost of the edge is the sum of the edge cost from t_i to u, the edge cost inter-cluster edge of (u, v) and the edge cost of v to t_j. Figure 4(b) shows the proposed graph decomposition D_2 with their values. It is proven by Chalermsook et al. that finding Minimum Spanning Tree (MST) on the graph decomposition is the same as the one in the all-distance graph.

From HCS connection discovery phase, *nearestNodeSet* has specified the cluster a module should belong to. Every terminal would know which cluster they are connected to through the information collected from the module. The next step would be finding minimum spanning tree on the graph decomposition and each terminal would decide which modules in their cluster should be activated.

C. Activating HCS devices distributedly

To form our activation strategy, we apply distributed minimum spanning tree algorithm by Gallager et al. [10]. The general idea is that the clusters of the terminals are merged with the shortest inter-cluster edge if both clusters have chosen the same shortest inter-cluster edge. The terminal is not communicating with other terminals about the shortest edge chosen.

Our activation strategy is a greedy approach to activate the HCS devices on modules along the shortest inter-cluster edge, which is a HCS connectivity path. If the modules already have its HCS devices turned on, it would try to activate a HCS device on a module further away from its assigned terminal along the HCS path and the HCS activation quota (HCS-AQ) of the terminal module would not be decremented. If the desired HCS device has been activated, the HCS-AQ of the corresponding terminal is decremented by one. Once two terminals are connected through HCS, they merge into a larger cluster with combined HCS-AQ of the terminals and the shortest inter-cluster edge of this cluster is computed based on the scheme of computing distributed minimum spanning tree proposed by Gallager et al. [10]. Eventually it forms a minimum spanning tree on the graph decomposition which is equivalent to a minimum spanning tree of all-distance graph D_1 of all terminal modules resulting in 2-approximation of minimum Steiner Tree. This activation process can be run in parallel and there are always enough HCS devices (if r is sufficient) for clusters merging based on the minimum of HCS-AQ ($\frac{r}{t}$) to each terminal. Its completeness is discussed in the next section (Section VI-A).

VI. Analysis of the Algorithm

A. Completeness

Lemma 6.1: HCS discovery is complete: All essential HCS connecting among the terminals are discovered.
Proof: The algorithm originates from the ANCHOR algorithm[7], which is able to discover all the HCS connectivity among the terminals, and therefore a single terminal searching is complete. In addition to this termination criteria, the parallel version of the algorithm only does not request anchoring a module that is not anchored another terminal. Therefore, all modules are anchored and all necessary HCS connections can be retrieved.

Lemma 6.2: Activation phase is complete: There must be enough HCS-AQ to activate HCS devices in modules to connect at least 2 groups of terminals if $r > 2OPT$
Proof: From the work by Chalermsook et al[5] as shown in Section V-A.2, the final result of minimum spanning tree of the graph decomposition is equivalent to the minimum spanning tree of all-distance graph among terminals. Suppose the resulting 2-approximate minimal network consists of n modules for t terminals and the active HCR device limitation is r , where $r \geq n$. Therefore, there are $t-1$ edges in the graph decomposition and averaging there are $\frac{n}{t-1}$ nodes in every edge. During the deployment phase, there are two clusters P and Q with p and q terminals respectively identified the same shortest inter-cluster edge and merging is going to perform. Since the two clusters P and Q are with $p-1$ and $q-1$ edges respectively and the algorithm always merge with shortest edge first, the total number of modules

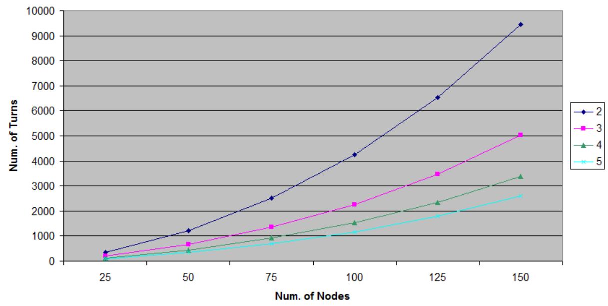


Fig. 5. Simulation Result for HCS Discovery Phase

involved would be below average $c \leq (p+q-1)\frac{n}{t-1}$. Recall that each terminal is assigned with at least $\frac{n}{t}$ HCS-AQ. Consider cluster P and Q, there are at least $(p+q)\frac{n}{t}$ HCS-AQ assigned. Since $(p+q)(\frac{n}{t-1}-\frac{n}{t}) \leq t(\frac{n}{t-1}-\frac{n}{t}) \leq \frac{n}{t-1}$, $c \leq (p+q-1)\frac{n}{t-1} \leq (p+q)\frac{n}{t}$. Therefore, there are always enough HCS-AQ in merging two clusters of terminal modules and deploy a network connecting all terminals with sufficient number of activated HCS devices.

B. Complexity in Total number of activations and Overall time

Since there are $O(n^2)$ possible HCS connections for testing, the worst case of total number of activation and deactivation of HCS device would be $O(n^2)$. However, since the distributed algorithm is a parallel search from each terminal module and the search depth is limited once they encounter a anchored node of another terminal module, the overall running time for a grid configuration (a balanced graph) with every terminal starting synchronously would be $O(\frac{n^2}{t})$.

VII. Simulation

Simulation is implemented to experiment the effect of the increasing number of nodes and allowance of simultaneous activation quota for the HCS discovery phase. In each turn (time step), the HCS device activated module can send and receive control messages through HCS communication or inter-module communication. Since the focus is to test the number of turns for the HCS discovery to converge, we abstract the network layer by assuming terminal modules can directly activate/deactivate the HCS device of one node with reliable data transfer through inter-module communication. Experiment is performed with 50 randomly generated graphs for each size 25, 50, 75, 100,125 and 150. In this test, there are three terminals and the module does not know about the HCS connections. Each graph is tested against different quota with $\frac{r}{t} = 2, 3, 4$ and 5, where each terminal is allowed to activate a maximum of 2,3,4 and 5 HCS devices simultaneously. As shown in Figure 5, the number of turns for convergence of HCS discovery phase has decreased with the increasing number of modules. This is due to more modules require to be anchored as the number of modules increases. With the increase of the quota for each terminal $\frac{r}{t}$, the discovery process is shown to converge faster. This is because more module can be used as anchors in the same cycle to discover HCS. However, we predict the improvement would be stagnant as $\frac{r}{t}$ keep increasing as the number of anchored modules allowed in one cycle depends on how many modules are connected to current anchored modules. For example, a line configuration with two ends as terminals would not benefit from having a larger quota $\frac{r}{t} \geq 2$.

VIII. CONCLUSIONS AND FUTURE WORK

This paper has presented a distributed algorithm for modular robots to deploy a 2-OPT hop minimum network without any hidden communication shortcut (HCS) information with a specified simultaneous activation quota on the devices. Activation quota are first assigned to terminals and parallel run of connectivity discovery algorithm derived from ANCHOR algorithm are used. With the properties of graph decomposition [5] and group merging in distributed minimum spanning tree algorithm [10], we apply greedy approach in the activation strategy such that with each terminal assigned with a activation quota on HCS devices of $\frac{r}{t}$ is sufficient to deploy a near-optimal network. The convergence time of the hidden communication shortcut discovery can be improved by a factor of t for balanced topology configuration. Simulation also shows a sample set that convergence time decreases along with the increase of quota r increases. We have also identified the possible convergence time improvement stagnancy in some case.

This work further explores the series of research possibilities of network deployment with the possibility of hidden communication shortcuts on modular robots. Future work such as improving the average convergence time by (1) enabling communication between the activating modules belonging to different terminals to avoid revisits and redundant HCS connectivity check, (2) dynamically reassigning quota between terminals to increase efficiency of simultaneous search are also interesting to explore, (3) dynamically determine the quota r based on inference feedback, (4) providing self adaptability to configuration change and (5) fault tolerance to module failure.

IX. ACKNOWLEDGMENTS

We would like to give thanks to Nadeesha Ranasinghe and our colleagues at Polymorphic Robotics Laboratory at the Information Sciences Institute of the University of Southern California for the suggestions and discussions and the constructive feedbacks from reviewers of major conferences.

REFERENCES

[1] *The Steiner Tree Problem.* Vieweg-Verlag, Feb 2002.

[2] 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.

[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] Andres Castano, Alberto Behar, and Peter Will. The Conro modules for reconfigurable robots. *IEEE/ASME Trans. Mechatronics*, 7(4):403–409, December 2002.

[5] Parinya Chalermsook and Jittat Fakcharoenphol. Simple distributed algorithms for approximating minimum steiner trees. In *COCOON*, pages 380–389, 2005.

[6] 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.

[7] Harris Chi Ho Chiu and Wei-Min Shen. Anchor: Self-configuring robotic network. In *IEEE Intl. Conf. on Robotics and Automation*, 2011.

[8] 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.

[9] 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.

[10] R. G. Gallager, P. A. Humblet, and P. M. Spira. A distributed algorithm for minimum-weight spanning trees. *ACM Trans. Program. Lang. Syst.*, 5:66–77, January 1983.

[11] Kyle Gilpin, Keith Kotay, Daniela Rus, and Iuliu Vasilescu. Miche: Modular shape formation by self-disassembly. *Int. J. Rob. Res.*, 27:345–372, March 2008.

[12] 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.

[13] 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.

[14] Morten Winkler Jrgensen, Esben Hallundbk stergaard, and Henrik Hautop Lund. Modular atron: Modules for a self-reconfigurable robot. pages 2068–2073, 2004.

[15] V. Kuo and R. Fitch. A multi-radio architecture for neighbor-to-neighbor communication in modular robots. In *IEEE International Conference on Robotics and Automation*, pages pp. 5387–5394, 2011.

[16] Haruhisa Kurokawa, Kohji Tomita, Akiya Kamimura, Shigeru Kokaji, Takashi Hasuo, and Satoshi Murata. Self-reconfigurable modular robot m-tran: distributed control and communication. pages 21:1–21:7, 2007.

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

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

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

[20] Behnam Salemi, Mark Moll, and Wei-Min Shen. SUPERBOT: A deployable, multi-functional, and modular self-reconfigurable robotic system. October 2006.

[21] 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.

[22] 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.

[23] Marcos A. Vieira, Ramesh Govindan, and Gaurav S. Sukhatme. Towards autonomous wireless backbone deployment in highly-obstructed environments. In *IEEE International Conference on Robotics and Automation*, 2011.

[24] Mark Yim, David Duff, and Kimon Roufas. Polybot: A modular reconfigurable robot. pages 514–520, 2000.

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

[26] 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.