Technical Report

Department of Computer Science
and Engineering
University of Minnesota
4-192 Keller Hall
200 Union Street SE
Minneapolis, MN 55455-0159 USA

TR 18-016

Asynchronous Network Formation in Unknown Unbounded Environments

Selim Engin, Volkan Isler

September 17, 2018

Asynchronous Network Formation in Unknown Unbounded Environments*

Selim Engin[1] and Volkan Isler[1]

***Abstract*— In this paper, we study the Online Network Formation Problem (ONFP) for a mobile multi-robot system. Consider a group of robots with a bounded communication range operating in a large open area. One of the robots has a piece of information which has to be propagated to all other robots. What strategy should the robots pursue to disseminate the information to the rest of the robots as quickly as possible? The initial locations of the robots are unknown to each other, therefore the problem must be solved in an online fashion.**

For this problem, we present an algorithm whose competitive ratio is $O(H \cdot \max\{M, \sqrt{MH}\})$ for arbitrary robot deployments, where M is the largest edge length in the Euclidean minimum spanning tree on the initial robot configuration and H is the height of the tree. We also study the case when the robot initial positions are chosen uniformly at random and improve the ratio to $O(M)$. Finally, we present simulation results to validate the performance in larger scales and demonstrate our algorithm using three robots in a field experiment.

I. Introduction

Consider a group of robots with unknown locations scattered over a large area. The robots can not communicate with each other unless they are within a given communication range. Such scenarios may occur, for example, after aerial deployments in disaster response settings. It might also be the case that many robots operate independently in a search task without coordination and disperse across the environment.

Now imagine that one of the robots wants to convey a piece of information to all other robots. This information can be used for network formation, rendezvous or any other task. For this purpose, the robot starts searching for other robots. Once a robot is found, it can be recruited to propagate the information. What strategy should the robot team follow to convey this information as soon as possible? In this paper, we study this online network formation in unbounded environments.

Our problem is closely related to the Freeze-Tag Problem (FTP), in which there is initially a single *active* (leader) robot and the rest of the robots are *frozen/asleep*, staying put [2]. The goal is to wake up all the asleep robots in the shortest time. An asleep robot awakens when an active robot is in its close proximity. After a robot awakens, it becomes activated and participates in the process of rousing the robots.

The network formation problem can be regarded as the online version of the original FTP, where the robot positions are unknown to each other. Without the knowledge of the locations of their peers, the active robots need to perform a search strategy to discover them as illustrated in Figure 1. Since the robots do not necessarily start executing the algorithm at the same time, our problem is asynchronous. This property of the problem is beneficial especially in real-world systems where exact synchrony is hard to achieve.

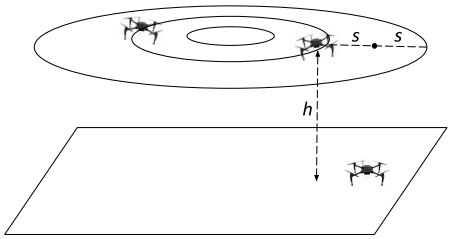


Fig. 1. An example scenario where two active robots are searching for the frozen robot on the ground.

A. Related Work

Algorithm design for mobile network formation has received significant attention over the past years. These approaches can be categorized as centralized and decentralized solutions [16]. In a centralized approach, a base station or a leader robot has access to positions of all the robots and can directly command them [15], [17]. On the other hand, with decentralized algorithms the robots can perform tasks using local information, without explicit instructions from an external source [4], [7].

While centralized approaches are usually efficient and easy to implement, adopting these solutions may not be possible in settings where the communication range of the robots is short. Consequently, there is an increasing interest in distributed communication protocols. Gossip protocols, for instance, are designed to disseminate information in a connected network of nodes [3]. In the single-piece dissemination scenario of a gossip protocol, an arbitrarily chosen node has a piece of information and the objective is to spread this information to all other nodes in the shortest time [14]. In this one-to-all broadcast setting, the network is stationary and a node propagates the information to its neighbors with an associated probability. Contrary to this formulation, in the FTP the nodes are able to move and transfer the piece of information to a neighbor as soon as they are within each other's connectivity range.

In the rendezvous problem, a group of robots need to meet at a point as quickly as possible. The two-player case with

*This work was supported by the NSF grant #1617718 and a Minnesota State LCCMR grant

[1]Selim Engin and Volkan Isler are with Department of Computer Science and Engineering, University of Minnesota, MN 55455, USA {engin003, isler}@umn.edu

initial positions unknown to each other was studied in [1]. Roy and Dudek proposed strategies to achieve rendezvous together with the exploration of an unknown environment [13]. Ozsoyeller et al. studied the rendezvous search problem on the line [10] and in planar environments [11]. Robots in this formulation do not know the positions of each other and they have to execute the same algorithm. Due to this constraint, the strategy is inherently stochastic since otherwise the robots cannot meet by following the same motion.

Poduri and Sukhatme studied the problem of establishing a connected network in a bounded toroidal domain using a decentralized approach [12]. In their model the robots perform random walk, and stop moving when they are near the base station or a robot who is already connected.

The FTP was originally posed in the graph settings where the robots are placed on the vertices of a graph [2]. The robots can move only along the edges and an asleep robot awakens when its vertex is visited by an active robot. In this formulation, each robot knows the positions of all other robots. The online version of the problem was studied by [6] who proposed a lower bound on the competitive ratio for graph settings.

The online FTP in the Euclidean domain was further explored in [8]. The authors considered the case where the robots are distributed uniformly at random over a rectangular shaped bounded area. In this case, the robots cannot go outside the closed area. For this problem the authors proposed a logarithmic factor approximation algorithm which depends on the area, therefore is not directly applicable.

The results in [8] rely on a bounded environment assumption. Considering the typical operation environments of field robots, including open seas and skies, this constraint may need to be relaxed. In this paper, we generalize the environment by removing the closed boundary assumption. In summary, our contributions can be stated as follows.

Our contribution: We study a scenario where the robots are deployed in an unbounded environment and the robot positions are unknown to each other. We analyze two cases of the problem: the initial robot positions are chosen arbitrarily and uniformly at random. For arbitrary deployments we present a strategy which is $O(H \cdot \max\{M, \sqrt{MH}\})$-competitive. This result improves the competitive ratio of a naive algorithm by a factor of H or $\sqrt{MH}$. The algorithm performs better in random deployments and the ratio becomes $O(M)$. In addition to providing performance guarantees, we demonstrate our algorithm in field experiments as a proof-of-concept implementation.

The remainder of the paper is organized as follows. We formulate our problem and introduce the models used throughout the paper in Section II. In Section III we present our algorithm and analyze its performance in Section IV. Section V contains simulations we conducted. We describe the outdoor experiments demonstrating the algorithm in Section VI. Finally, we conclude with a summary and future research directions in Section VII.

II. Problem Formulation and Models

In the Online Network Formation Problem (ONFP), at the beginning there is a single active robot and the goal is to find a strategy to explore all inactive robots such that the time it takes to tag the last robot is minimized.

In this problem, the robot initial positions are unknown to each other, thus the robots use an online strategy. The performance of online algorithms are usually evaluated in comparison to the optimal offline solution [9]. The optimal offline strategy has the access to the initial positions of all robots beforehand, so the leader robot does not need to perform a search but instead can directly move towards the inactive robots. For an instance σ of the problem, suppose $A(\sigma)$ is the solution produced by our online algorithm, and $OPT(\sigma)$ is the optimal offline solution. The algorithm A is said to be c-competitive if for all instances of the problem the ratio $A(\sigma)/OPT(\sigma) \leq c$ holds.

We proceed by describing the environment and robot models used throughout the paper.

A. Environment and Robot Models

In the ONFP, we have n robots scattered over an unbounded area. Initially, one active robot carries a piece of information to propagate to all other robots. An asleep robot receives this information (or awakens), once an active robot is within its close proximity. The robots are assumed to be point robots that always move at constant speeds.

The initial positions of the robots are $\mathcal{X} = \{x_1, \ldots, x_n\}$, where each $x_i \in \mathbb{R}^2$. Without loss of generality we assume x_1 is the leader's initial position and x_2 to x_n are labeled in non-decreasing order with respect to their distances to x_1. The Euclidean distance between two robots x_i and x_j is denoted by $d(x_i, x_j)$. In our analysis, we use the Minimum Spanning Tree (MST) on the initial robot positions and denote it by $\mathcal{T}$. We denote the longest edge length in $\mathcal{T}$ as M, and its height as H.

Finally, the communication range is denoted by r. We measure the distances in units of r, and a robot displaces by r per time step. In the performance analysis of our algorithm, we consider two cases: sparse and dense configurations. A configuration is called sparse if $n \cdot r \leq M$, and dense otherwise. In the rest of the paper we will ignore the unit distance in the expressions and regard as $r = 1$.

III. The Proposed Algorithm

In this section, we describe Circle-Tag, our strategy for the ONFP.

In essence, the algorithm partitions the unexplored environment into blocks of equal size and assigns an active robot to each of them as shown in Figure 2. The blocks are shaped as cut circle sectors and each active robot performs an arc-shaped back-and-forth motion to cover the block. This motion guarantees that an asleep robot is going to be explored when it is inside a block.

The algorithm includes a set of behaviors which we introduce next. The execution of our algorithm consists of a sequence of **rounds**. A round starts when the leader robot

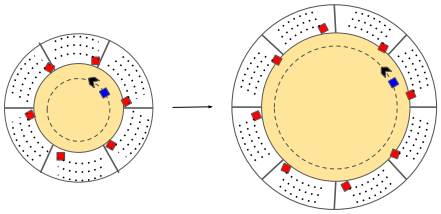


Fig. 2. The CIRCLE-TAG algorithm partitions the area to be covered into blocks of equal size and assigns an active robot (shown in red) to each. The leader robot (shown in blue) serves as a coordinator by collecting the explored robots and assigning blocks in the next round.

begins traveling on its circle and ends when the last active finishes covering its assigned block for that round.

The motion of the leader robot is called **Concentric-Circles-Search (CCS)**, and CCS(i) refers to the path of the leader in round i. The CCS is composed of concentric circles whose radii increase at each round. These circles are always centered at the initial position of the leader robot. Suppose the leader travels on a circle of radius R_i during a round i. If the number of activated robots is k, then R_i is given by:

$$R_i = \begin{cases} R_{i-1} + 1, & k = 0 \text{ or } k = 1 \\ R_{i-1} + k, & k \geq 2 \end{cases}$$

where the starting radius R_0 is zero. Initially, there is only the leader and no other robot is active, so $k = 0$. Until finding a robot, at each round the leader updates the radius R_i by 1. This motion pattern ensures a full coverage of the leader's surrounding.

At the end of each round, the leader moves in the East direction until it reaches the circle for the next round. Therefore, CCS(i) in a round i consists of traveling on a circle of radius R_i and moving by $R_{i+1} - R_i$ towards East to get to the next circle. An example of this motion is illustrated in Figure 3. Here, the leader robot is shown in blue. The yellow area is the already covered part and gray area is the region explored in the current circle. The white outer part of the environment is unknown to the robot.

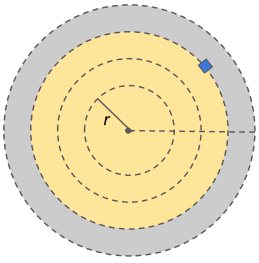


Fig. 3. CCS: The leader robot travels along a circle at each round and goes in the East direction to reach the circle for the next round.

When the leader hits its nearest frozen robot, the activated robot does not immediately participate in the search, but merely follows the leader robot instead. After the second nearest robot is found, the algorithm proceeds by partitioning the area into two blocks and assigns an active robot for each. In each round, the active robots other than the leader cover their blocks in multiple back-and-forth sweeps. This motion pattern is called **Block-Cover** and an instance of it is shown in Figure 4. The actives are reassigned to their new blocks in the beginning of the next round.

Suppose that there are $k_i \geq 2$ active robots, excluding the leader, in the beginning of round i. Then, the number of blocks for that round is also k_i. Let $b_{i,k}$ denote a block in round i. In each block $b_{i,k}$ the active robot travels along k_i concentric arcs and brings all the robots it encounters together, until the round finishes. The task of the leader robot is to circle around the covered area and reassign the active robots to their blocks, taking the newly found robots into account.

In round $i + 1$, the leader robot starts by collecting the robots discovered in round i, as it passes from the active robot locations. An active robot starts sweeping the arcs as soon as it receives the block assignment. The leader robot keeps the collected robots until the end of the round and assigns blocks for them in the next round. Therefore, a robot explored in round i joins the search starting from round $i+2$.

Algorithm 1 CIRCLE-TAG

Input: *status*: either *leader*, *active* or *frozen*

```
for each round i do
    switch status do
        case leader
            Travel on CCS(i)
            Assign blocks and pick frozens while touring
        case active
            Wait until the leader passes
            b_{i,k} ← Block assignment from the leader
            Perform Block-Cover(b_{i,k})
        case frozen
            Wait until found by an active robot
            Follow the active until the end of the round i
            After picked, follow the leader in round i+1
            status ← active
```

The algorithm proceeds in this manner by expanding the coverage area, and terminates when all robots are discovered. A pseudocode for CIRCLE-TAG is presented in Algorithm 1.

IV. PERFORMANCE OF CIRCLE-TAG

We proceed with analyzing the performance of CIRCLE-TAG. We start with the case where the initial positions of the robots are chosen arbitrarily, then continue with the case when they are distributed uniformly at random.

A. Sweeping a Block

The algorithm CIRCLE-TAG begins the process of exploring the frozen robots starting from the leader's initial position. After the leader awakens two robots, each activated robot works on a dedicated zone assigned by the leader.

Suppose there are already explored k_i robots in round i. Thus, the number of blocks is also k_i. An active robot sweeps

its block $b_{i,k}$ in k_i arcs in increasing radius whose apex angle is $2\pi/k_i$ (see Figure 4). The asleep robots encountered during the coverage follow the active robot until the end of the round and become activated in the next round.

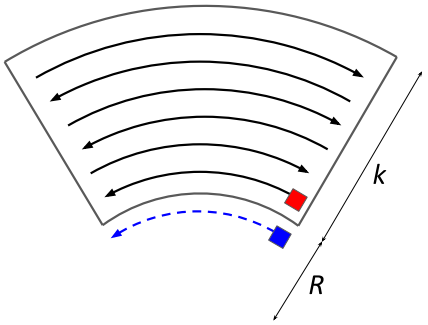


Fig. 4. Block-Cover: Each active robot covers its designated block in multiple back-and-forth sweeps

Proposition 1: Consider a round i with k_i active robots, and let $C(b_{i,k})$ denote the coverage time of a block $b_{i,k}$ using Block-Cover. The time it takes to complete the round is no more than $2C(b_{i,k})$.

Proof: In round i, the leader robot performs CCS(i), and its travel time in this round is $2\pi R_i + k_i$. In the remainder of the proof, we drop the index i in R_i and k_i to simplify the notation.

An active robot starts the coverage as soon as the leader passes from its location and assigns the block, specifying the apex angle and the number of arcs. Using Block-Cover, the coverage time of a block $b_{i,k}$ is given by:

$$C(b_{i,k}) = k + \sum_{j=1}^{k} 2\pi \frac{R+j}{k} = 2\pi R + k(\pi+1) + \pi. \quad (1)$$

In a round, the last active robot starts sweeping after the leader has traveled $2\pi R - 2\pi R/k = 2\pi(1-1/k)$. The travel time of the leader is $2\pi R + k$ and the last active robot finishes sweeping $C(b_{i,k}) + 2\pi(1-1/k)$ units after the beginning of the round. This means there is no overhead caused by the leader because $2\pi R + k < C(b_{i,k}) + 2\pi(1-1/k)$. Thus, $2C(b_{i,k})$ constitutes an upper bound on the completion time of a round whose blocks can be covered in k arcs. ■

In the analysis of our algorithm, we will bound the execution time of a round by twice the size of the block in that round.

B. Exploration with CIRCLE-TAG

We continue by analyzing the performance of CIRCLE-TAG in unbounded environments. Our analysis consists of two cases for the initial configuration of the robots. We say a configuration is sparse if $n \leq M$, and dense if $n > M$. We start with the sparse case.

1) Sparse configurations: In the beginning of the execution, the leader robot starts by performing a space-filling concentric-circles-search centered at its initial position as shown in Figure 3. Consider the MST $\mathcal{T}$ on the initial configuration of the robots, and let M denote the longest edge length in $\mathcal{T}$. Suppose x_1 is the leader robot, and x_2 and x_3 are the first and second activated robots, respectively. Since $\mathcal{T}$ spans all the robots, we know that the nearest neighbor distance of a robot cannot be more than M. Therefore, $d(x_1, x_2) \leq M$, and from the triangle inequality $d(x_1, x_3) \leq 2M$. Then, the time it takes to explore the first two robots is at most $4\pi M^2$.

To upper bound the solution of our algorithm, that is the time it takes to explore all robots using CIRCLE-TAG, we consider the worst case. The worst case configuration for the ONFP is when the robots are initially in a path configuration with an equal spacing of M, and the leader robot is the leftmost or rightmost robot. This means that the tree $\mathcal{T}$ is a path with all edges of length M. This configuration is the worst case scenario because it has the maximum furthest pairwise distance among all possible configurations.

The analysis of our algorithm comprises multiple **phases**. Each phase is a collection of rounds, and a phase finishes when there is a newly activated robot at the end of a round. Note that the phases are only used in the analysis and they are not explicitly available to the robots.

For any two consecutive phases $t-1$ and t, consider the circles they started and let their radii be R_{t-1} and R_t, respectively (see Figure 5).

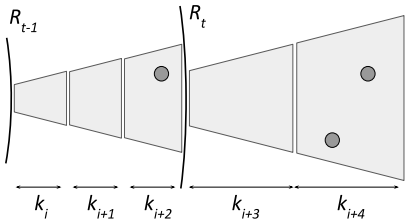


Fig. 5. Two consecutive phases $t-1$ and t are shown. A phase finishes when there is an activated robot at the end of a round in that phase.

Suppose also that the difference of their radii is $\Delta^- := R_t - R_{t-1}$, and sum of them is $\Delta^+ := R_t + R_{t-1}$. We continue by upper bounding Δ^- and Δ^+.

Let k_t be the number of active robots except the leader in phase t. The radius difference Δ^- cannot be more than $2M$. This is because the distance between a robot found in phase t to an already explored robot in earlier phases is at most M, and since there are at most k_t extra sweeps in a phase after finding a robot, the radii difference Δ^- is no more than $M + k_t < M + n \leq 2M$. The last inequality holds because $n \leq M$ in sparse configurations.

The radius sum Δ^+ can be bounded by $2M(t-1) + 2Mt < 4Mt$, since the radius of the starting circle of a phase increases by at most $\Delta^- < 2M$.

In each round of phase t, the blocks have height k_t since the number of activated robots remains same within a phase

and using Block-Cover the active robots sweep k_t arcs. Therefore, until the next robot is explored, in phase t there will be at most $\Delta^-/k_t < 2M/k_t$ rounds in the worst case.

Suppose D is the diameter of the tree $\mathcal{T}$. The distance between the initial position of the leader robot and its furthest neighbor is no more than MD. Then, there are at most $2H$ phases in the worst case, since $MD/M = D \leq 2H$.

We can then bound the solution of CIRCLE-TAG as:

$$\begin{aligned} SOL &\leq 4\pi M^2 + \sum_{t=1}^{2H} 2\pi \frac{(R_t^2 - R_{t-1}^2)}{k_t} \\ &\leq 4\pi M^2 + \sum_{t=1}^{2H} 2\pi \frac{\Delta^- \Delta^+}{k_t} \\ &< 4\pi M^2 + \sum_{t=1}^{2H} 2\pi \frac{8M^2 t}{t} = M^2(32\pi H + 4\pi), \end{aligned} \tag{2}$$

where the radius R_0 is zero, and $k_t \geq t$ since in each phase there is at least one explored robot. For sparse configurations, we then arrive at the following result.

Lemma 1: Suppose the MST on the initial configuration has height H and its maximum edge length is M. Then, the time to explore all robots using CIRCLE-TAG in sparse configurations is $O(M^2 H)$.

We continue with the dense configurations.

2) Dense configurations: In dense configurations, n is larger than M so we cannot bound the completion time of a phase as before. However, since the robots use the same strategy in both cases, for the part when the number of activated robots k_t in phase t is smaller than M, the performance analysis will be the same.

Suppose p denotes the number of phases where $k_t \leq M$, and q is the remaining number of phases. The execution time of CIRCLE-TAG in dense configurations is expressed as $SOL = SOL_p + SOL_q$, where SOL_p and SOL_q denote the time to complete the first p and the remaining q phases, respectively. For the first p phases, the execution time of the algorithm SOL_p is $O(M^2 H)$ since in these phases $k_t \leq n \leq M$ and the performance is the same with the case of sparse configurations.

In the rest of the q phases, the number of activated robots is larger than M. Then, we know that each round at least one frozen robot is going to be activated. SOL_q is bounded as follows.

$$\begin{aligned} SOL_q &\leq \sum_{t=p+1}^{p+q} 2\pi \frac{(R_t^2 - R_{t-1}^2)}{k_t} \\ &= \sum_{t=p+1}^{p+q} 2\pi \frac{(R_t - R_{t-1})(R_t + R_{t-1})}{k_t} \\ &= \sum_{t=p+1}^{p+q} 2\pi (R_t + R_{t-1}) < \sum_{t=p+1}^{p+q} 4\pi R_t. \end{aligned} \tag{3}$$

The radius of the covered area, $R_t \leq MD$, since by that time all the robots will be explored.

In the worst case, at each phase only one frozen robot is tagged and the rest of the robots are found in the last phase. This is because the frozen robots can participate in the search sooner if they were explored early. Suppose k'_t denotes the number of activated robots up to the phase $(p+t)$. Since at each round the coverage area radius increases by k'_t, we have $\sum_{t=p+1}^{p+q} k'_t < MD$.

In the beginning of the phase $p+1$, $k'_t > M$ and there is at least one robot explored in each phase. Then,

$$\sum_{t=p+1}^{p+q} k'_t < \sum_{i=0}^{q-1} (M+i) < qM + q^2/2 < MD.$$

For the values of $M > 0$ and $D > 0$, the range of q satisfying the quadratic inequality above is $q < \sqrt{2MD}$. Then, we can bound SOL_q by $\sum_{t=p+1}^{p+q} 4\pi R_t < 4\pi\sqrt{2}(MD)^{1.5} = O((MH)^{1.5})$.

The execution time in dense configurations $SOL = SOL_p + SOL_q = O(\max\{M^2 H, (MH)^{1.5}\})$, and we obtain the following result.

Lemma 2: The time it takes to explore all robots using CIRCLE-TAG in dense configurations is $O(\max\{M^2 H, (MH)^{1.5}\})$, if the initial configuration MST has height H and maximum edge length M.

We proceed with lower bounding the optimal offline strategy. If a strategy is offline, then it has the access to observe all the inputs in the beginning of the execution. This means that the active robots do not need to search the environment, but instead go directly to the frozen robot locations using the shortest paths.

Consider $\mathcal{T}$, the MST on the initial configuration of the robots. The longest edge length M in $\mathcal{T}$ is a lower bound for the optimal solution, because no strategy with execution time less than M can tag all the robots.

The following is the main result of our paper for arbitrary deployments.

Theorem 1: There exists an $O(H \cdot \max\{M, \sqrt{MH}\})$-competitive algorithm to the Online Network Formation Problem for arbitrary deployments in unbounded environments, where H is the height of the MST on the initial configuration and M is the maximum edge length in the tree.

Proof: In Lemmas 1 and 2 we have shown that the time it takes to explore all the robots with CIRCLE-TAG is $O(\max\{M^2 H, (MH)^{1.5}\})$. Lower bounding the optimal solution by M concludes the proof. ■

Theorem 1 establishes our result for arbitrary robot deployments.

C. CIRCLE-TAG *in Random Deployments*

In this section, we present the performance of CIRCLE-TAG for the case when the initial robot positions are chosen uniformly at random.

Suppose that the robots are distributed uniformly at random over an open circular area of radius L. Neither the environment dimensions nor the robot positions are available to the robots.

Because the environment radius is L, the furthest pairwise distance between the robots is no more than $2L$. We have a loose upper bound by noting that all robots can be tagged by the first robot in time $4\pi L^2$. Using CIRCLE-TAG, however, we show that we can significantly improve the performance to $O(M^2)$.

We analyze the execution time of the algorithm in phases as in the case of arbitrary deployments.

Proposition 2: Consider a partition of the environment surrounding the leader's initial position x_1 with N circular rings of increasing radius iM, for $i = 1, \ldots, N$, until all robots are included. Then, each ring has at least one robot in it and the number of rings is bounded as $N \leq 2L/M$.

Proof: We prove this proposition by contradiction. Suppose there is a ring without any robots in it. This means that between the robots inside and outside this ring, all pairwise distances are larger than M. Then, the tree $\mathcal{T}$ has an edge longer than M, which is a contradiction. $NM \leq 2L$ since the maximum pairwise distance is at most $2L$. ■

In our analysis we use the expected number of robots in the rings. Let S_i denote the number of robots in ring i, and $S_{1:i}$ be the number of robots up to and including the ring i. To compute the expected values of S_i and $S_{1:i}$, we first assume that the leader position is at the center of the area.

Proposition 3: Suppose the i-th innermost ring has radius iM and there are S_i robots inside. Then, the expected value of S_i is lower bounded as $\mathbb{E}[S_i] \geq n(2i-1)M^2/4L^2$.

Proof: The expected number of robots within a single ring is proportional to the size of the ring. If there are N rings, then the outermost ring has radius NM and the region that includes all the robots has area $\pi(NM)^2$. The i-th ring has radius iM, so its area is $\pi(iM)^2 - \pi((i-1)M)^2 = \pi M^2(2i-1)$. Thus, the expected number of robots in ring i is bounded as $\mathbb{E}[S_i] \geq n(2i-1)/N^2 \geq n(2i-1)M^2/4L^2$, since $N \leq 2L/M$. ■

Proposition 4: The expected number of robots in up to and including the i-th innermost ring is lower bounded as $\mathbb{E}[S_{1:i}] \geq n(iM)^2/4L^2$.

Proof: The proof follows by noting that $\mathbb{E}[S_{1:i}] = \sum_{j=1}^{i} \mathbb{E}[S_j]$ and $\sum_{j=1}^{i}(2j-1) = i^2$. ■

From Proposition 1, we know that the completion time of a round is no more than twice the size of a block in that round. The time it takes to finish a phase is the sum of the completion time of the rounds within the phase. Since the i-th innermost ring has area $\pi M^2(2i-1)$, we bound the expected execution time of CIRCLE-TAG as follows.

$$\begin{aligned}\mathbb{E}[SOL] &\leq \sum_{i=1}^{2L/M} \frac{2\pi M^2(2i-1)}{n\frac{i^2M^2}{4L^2}} \\ &\leq \sum_{i=1}^{2L/M} 16\pi\frac{L^2}{ni} \approx 16\pi L^2\frac{\log(2L/M)}{n} \\ &\leq 16\pi L^2\frac{\log n}{n},\end{aligned} \tag{4}$$

where the expectation is taken over all instances of the problem that are equally likely. The last inequality holds since $2L$ is the maximum possible pairwise distance and $Mn \geq 2L$. We can see that the exploration time is in $O(L^2 \log n/n)$. Note that this bound does not change even when the leader is not at the center because the expected value of $S_{1:i}$ would be at least one fourth of the result in Proposition 4.

Lemma 3: Suppose n robots are distributed uniformly at random over a circular area of radius L. The expected time to explore all robots is $O(L^2 \log n/n)$ using CIRCLE-TAG.

Next, we continue by analyzing the lower bound for the optimal solution when the robot positions are chosen uniformly at random.

The longest edge length M of the MST $\mathcal{T}$ determines an important property in initial configurations. Suppose $G(n, r)$ denotes a Random Geometric Graph (RGG) of n nodes each with communication radius r. A two dimensional RGG is constructed by uniformly distributing n points over a unit ball, and placing an edge between two nodes if they are within r of each other.

An MST on a set of points is also a Minimum Bottleneck Spanning Tree, that is no other spanning tree can have a maximum edge length smaller than M, on the same set of points. Then, we can observe that M is the critical distance to make $G(n, M)$ connected, because otherwise the MST would have a smaller longest edge.

The critical distance to make $G(n, r)$ connected is well-studied in the literature. The following result establishes a bound for the connectivity of an RGG.

Lemma 4: ([3], [5]) A two dimensional geometric random graph $G(n, r)$ is connected with probability at least $1 - 1/n^2$, if $r \geq \sqrt{2 \log n/n}$.

Lemma 4 is remarkable since it tells us that the critical distance to make an RGG connected is $O(L\sqrt{\log n/n})$, when the points are distributed over a circle of radius L. We also investigated this bound for connectivity in simulations. Figure 6 shows the average values of M for 20 to 1000 robots over 1000 trials. The robots are distributed uniformly at random over a circle of radius $L = 20$. The expressions $L\sqrt{\log n/n}$ and $L\sqrt{2 \log n/n}$ act as lower and upper bounds for the mean values of M. Then, we can conclude that $M = O(L\sqrt{\log n/n})$, if the robot positions are chosen uniformly at random.

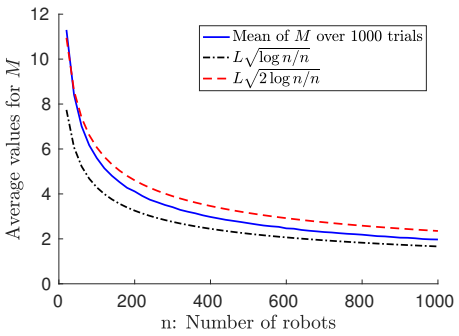


Fig. 6. Comparison of the average values of M to the bounds for $L = 20$ and varying $n = 20$ to 1000.

Theorem 2: There exists an $O(M)$-competitive algorithm to the Online Network Formation Problem in unbounded environments for the case when robot initial positions are chosen uniformly at random.

Proof: In Lemma 3, we showed that the expected solution of CIRCLE-TAG is $O(L^2 \log n/n)$. The proof follows by lower bounding the optimal solution as $M = O(L\sqrt{\log n/n})$. ■

Theorem 2 is our main result for the case of random deployments. In the next section, we present simulation results to validate the performance analysis.

V. SIMULATIONS

We conducted simulations in MATLAB to confirm our analytical analyses presented in the previous section. The robots are deployed uniformly at random over a circular area of radius L which is not accessible by the robots. In the simulations we used varying number of robots from 20 to 1000.

A snapshot from the execution of the algorithm is shown in Figure 7. Here, the leader robot is shown in blue, the active robots are colored red, and the rest of the robots are gray.

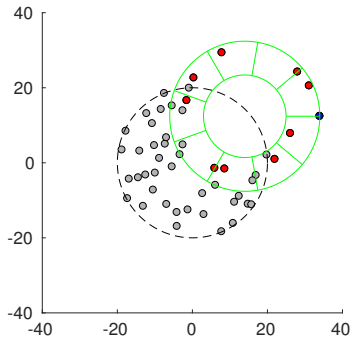


Fig. 7. An instance from the execution of the algorithm where 50 robots are deployed in an area of radius 20.

The robot initial positions are distributed uniformly at random over a circular region of radius $L = 20$ indicated by the dashed black circle. The solid in-circle depicts the already covered area while the out-circle shows the coverage area of a single round. Each partition of the area in between the in and out-circles represent the blocks where the active robots operate.

In Figure 8 we present the simulation results of our algorithm. In this simulation, the robots are distributed uniformly at random over an area of radius $L = 10$ and varying number of robots from 50 up to 1000 is used. For each number of robots we take the average of the algorithm's performance over 1000 trials. We see that the M^2H bound on arbitrary deployments fits nicely on random deployments as well. We note that the arbitrary deployments bound holds even without its coefficient derived in the analysis. The expected bound for random deployments also fits the result especially after the number of robots is significantly large. For fewer number of robots the bound is loose, which suggests that there is a room for improvement in the analysis.

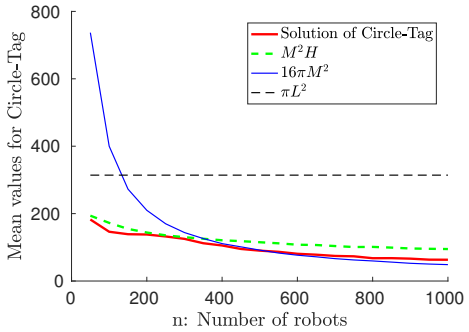


Fig. 8. Comparison between mean values of CIRCLE-TAG performance and upper bounds for $L = 10$ and varying $n = 50$ to 1000. The average values are taken over 1000 trials.

VI. FIELD EXPERIMENTS

In this section, we describe our multi-robot system and the field experiments demonstrating CIRCLE-TAG.

A. System Description

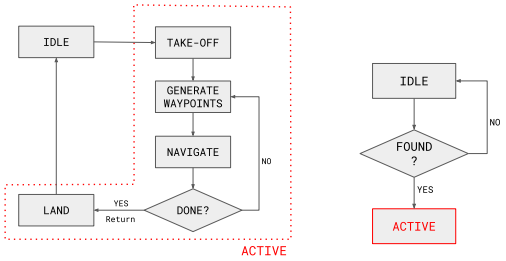
The experimental system is composed of multiple robots, each able to navigate autonomously, communicate and make decisions on the fly. Our system consists of three Uninhabited Aerial Vehicles (UAVs) and a ground station which is used to start and abort the mission. In our experiments, we used the quadrotor DJI Matrice 100 equipped with GPS and compass modules. As an on-board computation unit, each UAV uses the NVIDIA Jetson TX1 running Linux Ubuntu 16.04 operating system. The communication network is based on the XBee modules. Each robot and the base station has an XBee module so that the robot states can be monitored from the ground. This also allows the base station to interrupt the mission in case of emergency situations such as low battery voltage.

We proceed by describing the finite state machine used by the robots during the execution of the algorithm.

B. State Machines for the Mission

In the ONFP we have initially a single active robot and the rest of the robots are staying put. As part of the implementation of CIRCLE-TAG, we designed finite state machines for the active and frozen robots.

Initially, the leader robot runs the state machine shown in 9(a) and all other robots run the one in 9(b). In the beginning of the execution, all robots are in the state `IDLE` waiting for the ground station to start the mission. The inactive robots remain in this state until they are found and became activated. Once the leader hears from the station, it goes into the `TAKE-OFF` state. After the take-off, the leader is in the `GENERATE-WAYPOINTS` state where it computes a path generated by CIRCLE-TAG. While traveling on this path, the robot is in the `NAVIGATE` state. In this state, the active robot checks its vicinity using the XBee module and tags a frozen



(a) Active strategy (b) Frozen strategy

Fig. 9. The finite state machines for the active and frozen robots

robot if it is nearby. If there is a tagged robot by the end of a round, the newly activated robot starts executing the active finite state machine. The robot then takes-off and all active robots generate a new path. This process is repeated until all robots are found and the round is finished. After this point, the robots return to the home locations (a gathering point), and go into the LAND state to finalize the mission.

C. Algorithm Demonstration

We implemented the state machines in Figure 9 as Robot Operating System (ROS) based modules compatible with our hardware. The experiments are conducted in Cedar Creek Ecosystem Science Reserve over a $100m \times 150m$ area.

Figure 10 shows the starting locations and trajectories of the robots used in the experiment. Each solid line with a separate color indicates the trajectory of a robot, and the triangles are the starting positions of the robots. An instance of the experiment is illustrated in Figure 1.

Fig. 10. Starting positions and trajectories of the UAVs

Although our algorithm is designed for planar environments, we generalize it to the UAV setting as follows. The highest UAV altitude is set to $h = 40m$ and the robots increase the radius of their concentric-circles path by $2s = 60m$ at each circle. The communication range of the robots is $r = 50m$. Then, any staying put robot is guaranteed to be explored by the search since if it has a distance d to a hovering active robot, we can bound this distance as $d \leq r = \sqrt{h^2 + s^2}$.

VII. Conclusion

In this paper, we studied the network formation problem for a multi-robot system operating in an unbounded area. Each robot has a limited sensing range, therefore only the robots in the vicinity are communicable. For this problem, we introduced an algorithm whose competitive ratio is $O(H \cdot \max\{M, \sqrt{MH}\})$ for the case where the initial robot positions are chosen arbitrarily. On the other hand, when the robots are distributed uniformly at random we improve the ratio to $O(M)$. Future work includes removing the $O(\sqrt{MH})$ term in the competitive ratio for the case of arbitrary deployments.

Another research direction is to consider the problem for a heterogeneous robots setting. In this case the task assignments of the robots can differ: for example, faster aerial robots can be used for exploring far away regions while the slower ground robots may be more efficient to search the nearby areas.

References

[1] Steve Alpern. The rendezvous search problem. *SIAM Journal on Control and Optimization*, 33(3):673–683, 1995.

[2] Esther M Arkin, Michael A Bender, Sándor P Fekete, Joseph SB Mitchell, and Martin Skutella. The freeze-tag problem: how to wake up a swarm of robots. *Algorithmica*, 46(2):193–221, 2006.

[3] Stephen Boyd, Arpita Ghosh, Balaji Prabhakar, and Devavrat Shah. Randomized gossip algorithms. *IEEE Transactions on Information Theory*, 52(6):2508–2530, 2006.

[4] Jorge Cortés, Sonia Martínez, and Francesco Bullo. Robust rendezvous for mobile autonomous agents via proximity graphs in arbitrary dimensions. *IEEE Transactions on Automatic Control*, 51(8):1289–1298, 2006.

[5] Piyush Gupta and Panganamala R Kumar. Critical power for asymptotic connectivity in wireless networks. In *Stochastic Analysis, Control, Optimization and Applications*, pages 547–566. Springer, 1999.

[6] Mikael Hammar, Bengt J Nilsson, and Mia Persson. The online freeze-tag problem. In *Latin American Symposium on Theoretical Informatics*, pages 569–579. Springer, 2006.

[7] Meng Ji and Magnus Egerstedt. Distributed coordination control of multiagent systems while preserving connectedness. *IEEE Transactions on Robotics*, 23(4):693–703, 2007.

[8] Eric Meisner, Wei Yang, and Volkan Isler. Probabilistic network formation through coverage and freeze-tag. *Intelligent Service Robotics*, 2(4):265–273, 2009.

[9] Rajeev Motwani and Prabhakar Raghavan. *Randomized algorithms*. Chapman & Hall/CRC, 2010.

[10] Deniz Ozsoyeller, Andrew Beveridge, and Volkan Isler. Symmetric rendezvous search on the line with an unknown initial distance. *IEEE Transactions on Robotics*, 29(6):1366–1379, 2013.

[11] Deniz Ozsoyeller, Volkan Isler, and Andrew Beveridge. Symmetric rendezvous in planar environments with and without obstacles. In *AAAI*, 2012.

[12] Sameera Poduri and Gaurav S Sukhatme. Latency analysis of coalescence for robot groups. In *2007 IEEE International Conference on Robotics and Automation*, pages 3295–3300. IEEE, 2007.

[13] Nicholas Roy and Gregory Dudek. Collaborative robot exploration and rendezvous: Algorithms, performance bounds and observations. *Autonomous Robots*, 11(2):117–136, 2001.

[14] Devavrat Shah. Gossip algorithms. *Foundations and Trends in Networking*, 3(1):1–125, 2009.

[15] Kunal Srivastava and Mark W Spong. Multi-agent coordination under connectivity constraints. In *American Control Conference, 2008*, pages 2648–2653. IEEE, 2008.

[16] Michael M Zavlanos, Magnus B Egerstedt, and George J Pappas. Graph-theoretic connectivity control of mobile robot networks. *Proceedings of the IEEE*, 99(9):1525–1540, 2011.

[17] Michael M Zavlanos and George J Pappas. Controlling connectivity of dynamic graphs. In *44th Conference on Decision and Control, European Control Conference CDC-ECC'05*, pages 6388–6393. IEEE, 2005.