Keeping Mobile Robot Swarms Connected

Alejandro Cornejo[1], Fabian Kuhn[1], Ruy Ley-Wild[2], and Nancy Lynch[1]

[1] MIT, Cambridge MA 02139, USA
{acornejo,fkuhn,lynch}@csail.mit.edu
[2] CMU, Pittsburgh PA 15213, USA
rleywild@cs.cmu.edu

Abstract. Designing robust algorithms for mobile agents with reliable communication is difficult due to the distributed nature of computation, in mobile ad hoc networks (MANETs) the matter is exacerbated by the need to ensure connectivity. Existing distributed algorithms provide coordination but typically assume connectivity is ensured by other means. We present a connectivity service that encapsulates an arbitrary motion planner and can refine any plan to preserve connectivity (the graph of agents remains connected) and ensure progress (the agents advance towards their goal). The service is realized by a distributed algorithm that is *modular* in that it makes no assumptions of the motion-planning mechanism except the ability for an agent to query its position and intended goal position, *local* in that it uses 1-hop broadcast to communicate with nearby agents but doesn't need any network routing infrastructure, and *oblivious* in that it does not depend on previous computations.

We prove the progress of the algorithm in one round is at least $\Omega(\min(d, r))$, where d is the minimum distance between an agent and its target and r is the communication radius. We characterize the worst case configuration and show that when $d \geq r$ this bound is tight and the algorithm is optimal, since no algorithm can guarantee greater progress. Finally we show all agents get ε-close to their targets within $O(D_0/r+n^2/\varepsilon)$ rounds where n is the number of agents and D_0 is the sum of the initial distances to the targets.

1 Introduction

Motivation. Designing robust algorithms for mobile agents with reliable communication is difficult due to the distributed nature of computation. If the agents form a mobile ad hoc network (MANET) there is an additional tension because communication is necessary for motion-planning, but agent movement may destabilize the communication infrastructure. As connectivity is the core property of a communication graph that makes distributed computation possible, algorithms for MANETs must reconcile the interaction between communication and motion planning in order to preserve connectivity.

Existing distributed algorithms for MANETs provide coordination but typically sidestep the issue of connectivity by assuming it is ensured by other means. For example, algorithms on routing [1,2], leader election [3], and mutual exclusion [4] for MANETs assume they run on top of a mobility layer that controls the trajectories of the agents. Those algorithms deal with connectivity by assuming the mobility layer guarantees that every pair of nodes that need to exchange a message are connected at some instant or transitively through time, otherwise they work on each independent connected cluster. On the other hand, work on flocking [5,6], pattern formation [7], and leader following [8] provides a mobility layer for a MANET that determines how agents will move. Again connectivity is sidestepped by assuming coordination runs atop a network layer that ensures it is always possible to exchange information between every pair of agents. The service we present would thus enable to execute the flocking algorithm of [5] using the routing algorithm of [2], or running the leader follower algorithm of [8] using the leader election service of [3], with the formal guarantee that connectivity is maintained and progress is made. The connectivity service allows an algorithm designer to focus on the problems which are specific to the application (i.e. search and rescue, demining fields, space exploration, etc.) without having to deal with the additional issues that arise when there is no fixed communication infrastructure. We expect algorithms designed on top of this service will be easier to prove correct because the safety and progress properties are maintained orthogonally by the guarantees of the service.

Related work. The problem of preserving connectivity has been addressed before, mainly in the control theory community. However, most proposed solutions are either centralized or preserve connectivity only while performing specific tasks (i.e. converging to a point). For example [9] models connectivity as a constrained optimization problem, but as a result the solution is centralized and does not exploit the locality of distributed computation. Another centralized algorithm for second-order agents is proposed in [10], however it conservatively preserves all edges in the graph. The problem of gathering (rendezvous) all agents to a single point while preserving connectivity is studied in [11,12,13,14]. In [15,16] the authors evaluate through simulations the problem of connected deployment, but do not prove in which configurations the algorithms achieve deployment or preserve connectivity. In contrast in this paper we present a local algorithm that preserves connectivity while performing an arbitrary task, we focus on providing formal safety and progress guarantees. A preliminary version of the algorithm without progress guarantees appeared in [17].

Communication Model. We assume each agent is equipped with a communication device that permits reliable broadcasting to all other agents within some communication radius r. Without loss of generality we suppose $r = 1$ throughout. The service operates in synchronous rounds, it assumes access to a positioning device; relative position between neighboring agents is sufficient, but for ease of exposition we assume absolute position is available. Finally the service assumes the existence of a motion planner which is queried at each round for the desired target position, the service produces a trajectory which preserves connectedness and, when possible, gets closer to the target.

Contributions. We present a distributed connectivity service that modifies an existing motion plan to ensure connectivity using only local information and without making any assumptions of the current and goal configurations. In particular, even if the goal configuration is disconnected, the service guarantees connectivity while trying to get each agent as close as possible to its target. Furthermore, the connectivity service only requires the immediate intended trajectory and the current position, but it is stateless, and hence *oblivious*. The service is also *robust* to the motion of each agent in that the refined plan preserves connectivity irrespective of the agents' speed changes. Therefore agents remain connected throughout their motion even if they only travel a fraction (possibly none) of their trajectory.

Connectivity is a global property, so determining whether an edge can be removed without disconnecting the graph may require traversing the whole graph. However, exploiting the distributed nature of a team of agents requires allowing each agent to perform tasks with a certain degree of independence, so communicating with every agent in the graph before performing each motion is prohibitive. To solve this we parametrize the service with a filtering method that determines which edges must be preserved and which can be removed, we also suggest several local algorithms that can be used to implement this filtering step.

We define *progress* as the quantification of how much closer each agent gets to its target in a single round. Our algorithm guarantees that the total progress is at least $\min(d, r)$ in configurations where every agent wants to move at least some distance d and the communication radius is r. Furthermore, we exhibit a class of configurations where no local algorithm can do better than this bound, hence under these conditions the bound is tight and the algorithm is asymptotically optimal. In the last section we prove all agents get ε-close to their target within $O(D_0/r + n^2/\varepsilon)$ rounds where D_0 is the total initial distance to the targets and n is the number of agents. Since the motion of the agents occurs in a geometric space and the service deals directly with motion planning, most progress arguments rely on geometrical reasoning.

We introduce some notation and definitions in §2. In §3 we present the intersecting disks connectivity service and discuss its parametrization in a filtering function. We prove the algorithm preserves connectivity and produces robust trajectories (§4). In §5 we prove that any lower-bound on progress for chains also applies to general graphs. We start §8 by giving a lower bound on progress of a very restricted class of chains with only two nodes, and in the rest of the section we show how to extend this lower bound to arbitrary chains. We give the termination bound in §9 and conclude in §10.

2 Preliminary Definitions

The *open disk* centered at p with radius r is the set of points at distance less than r from p: $\mathsf{disk}_r(p) := \{q : \|p-q\| < r\}$. The *circle* centered at p with radius r is the set of points at distance r from p: $\mathsf{circle}_r(p) := \{q : \|p-q\| = r\}$. The *closed disk* centered at p with radius r is the set of points at distance at most

r from p: $\overline{\mathsf{disk}}_r(p) := \mathsf{circle}_r(p) \cup \mathsf{disk}_r(p) = \{q : \|p - q\| \leq r\}$. We abbreviate $\mathsf{disk}(p, q) := \mathsf{disk}_{\|p-q\|}(p)$, $\mathsf{circle}(p, q) := \mathsf{circle}_{\|p-q\|}(p)$, $\overline{\mathsf{disk}}(p, q) := \overline{\mathsf{disk}}_{\|p-q\|}(p)$. The *unit disk* of point p is $\mathsf{disk}_1(p)$.

The *lens* of two points p and q is the intersection of their unit disks: $\mathsf{lens}(p, q) := \mathsf{disk}_1(p) \cap \mathsf{disk}_1(q)$. The *cone* of two points p and q is defined as the locus of all the rays with origin in p that pass through $\mathsf{lens}(p, q)$ (the apex is p and the base is $\mathsf{lens}(p, q)$): $\mathsf{cone}(p, q) := \{r : \exists s \in \mathsf{lens}(p, q). r \in \mathsf{ray}(p, s)\}$, where $\mathsf{ray}(p, q) := \{p + \gamma(q - p) : \gamma \geq 0\}$.

A *configuration* $C = \langle I, F \rangle$ is an undirected graph where an agent $i \in I$ has a *source* coordinate $s_i \in \mathbb{R}^2$, a *target* coordinate $t_i \in \mathbb{R}^2$ at distance $d_i = \|s_i - t_i\|$, and every pair of neighboring agents $(i, j) \in F$ are *source-connected* (*i.e.*, $\|s_i - s_j\| \leq r$) where r is the communication radius. We say a configuration C is a *chain* (resp. *cycle*) if the graph is a simple path (resp. cycle).

3 Distributed Connectivity Service

In this section we present a distributed algorithm for refining an arbitrary motion plan into a plan that moves towards the intended goal and preserves global connectivity. No assumptions are made about trajectories generated by the motion planner, the connectivity service only needs to know the current and target positions and produces a straight line trajectory at each round; the composed trajectory observed over a series of rounds need not be linear. The trajectories output by the service are such that connectivity is preserved even if an adversary is allowed to stop or control the speed of each agent independently.

The algorithm is parameterized by a filtering function that determines a sufficient subset of neighbors such that maintaining 1-hop connectivity between those neighbors preserves global connectivity. The algorithm is *oblivious* because it is stateless and only needs access to the current plan, hence it is resilient to changes in the plan over time.

3.1 The Filtering Function

Assuming the communication graph is connected, we are interested in a FILTER subroutine that determines which edges can be removed while preserving connectivity. Let s be the position of an agent with a set N of 1-hop neighbors, we require a function FILTER(N, s) that returns a subset of neighbors $N' \subseteq N$ such that preserving connectivity with the agents in the subset N' is sufficient to guarantee connectivity.

认为

We will not require for FILTER to be symmetric, hence it may deem necessary for i to preserve j as a neighbor, but not the other way around. However, a FILTER function is *valid* if preserving symmetric edges is sufficient to preserve global connectivity, where an edge (i, j) is symmetric if i should preserve j ($s_j \in N'_i$) and vice versa ($s_i \in N'_j$).

The identity function FILTER$(N, s) := N$ is trivially valid because connectivity is preserved if no edges are removed. However, ideally we want a FILTER function that in some way "minimizes" the number of edges kept. A natural choice is to compute the minimum spanning tree (MST) of the graph, and return for every agent the set of neighbors which are its one hop neighbors in the MST. Although in some sense this would be the ideal filtering function, it cannot be computed locally and thus it is not suited for the connectivity service.

Nevertheless, there are well known local algorithms that compute sparse connected spanning subgraphs, amongst them is the Gabriel graph (GG) [18], the relative neighbor graph (RNG) [19], and the local minimum spanning tree ($LMST$) [20]. All these structures are connected and can be computed using local algorithms. Since we are looking to remove as many neighbors as possible and $MST \subseteq LMST \subseteq RNG \subseteq GG$, from the above $LMST$ is best suited.

Remark. The connected subgraph represented by symmetric filtered neighbors depends on the positions of the agents, which can vary from one round to the next. Hence, the use of a filtering function enables preserving connectivity without preserving a fixed set of edges (topology) throughout the execution; in fact, it is possible that no edge present in the original graph appears in the final graph.

3.2 The Algorithm

We present a three-phase service (cf. Algorithm 1) that consists of a collection phase, a proposal phase, and an adjustment phase. In the collection phase each agent queries the motion planner and the location service to obtain its current and target positions (s_i and t_i respectively). Each agent broadcasts its position and records the position of neighboring agents discovered within its communication radius.

Algorithm 1. ConnServ run by agent i

```
                                                        ▷ Collection Phase
  $s_i \leftarrow query\_positioning\_device()$
  $t_i \leftarrow query\_motion\_planner()$
  broadcast $s_i$ to all neighbors
  $N_i \leftarrow \{s_j \mid \text{for each } s_j \text{ received}\}$
                                                        ▷ Proposal Phase
  $N'_i \leftarrow \textsc{Filter}\ (N_i, s_i)$
  $R_i \leftarrow \bigcap_{s_j \in N'_i} \mathsf{disk}_1(s_j)$
  $p_i \leftarrow \text{argmin}_{p \in R_i} \|p - t_i\|$
  broadcast $p_i$ to all neighbors
  $P_i \leftarrow \{p_j \mid \text{for each } p_j \text{ received}\}$
                                                        ▷ Adjustment Phase
  if $\forall s_j \in N'_i . \|p_j - p_i\| \leq r$ then
      return trajectory from $s_i$ to $p_i$
  else
      return trajectory from $s_i$ to $s_i + \frac{1}{2}(p_i - s_i)$
  end if
```

In the proposal phase the service queries the Filter function to determine which neighboring agents are sufficient to preserve connectivity. Using the neighbors returned by Filter the agent optimistically chooses a target p_i. The target is optimistic in the sense that if none of its neighboring agents move, then moving from source s_i to the target p_i would not disconnect the network. The proposed target p_i is broadcast and the proposals of other agents are collected.

Finally in the adjustment phase, each agent checks whether neighbors kept by the Filter function will be reachable after each agent moves to their proposed target. If every neighbor will be reachable, then the agent moves from the current position to its proposed target, otherwise it moves halfway to its proposed target, which ensures connectivity is preserved (proved in the next section).

4 Preserving Connectivity

In this section we prove the algorithm preserves network connectivity with any valid Filter function. Observe that since R_i is the intersection of a set of disks that contain s_i, it follows that R_i is convex and contains s_i. By construction $p_i \in R_i$ and thus by convexity the linear trajectory between s_i and p_i is contained in R_i, so the graph would remain connected if agent i were to move from s_i to p_i and every other agent would remain in place. The following theorems prove a stronger property, namely, the trajectories output guarantee symmetric agents will remain connected, even if they slow down or stop abruptly at any point of their trajectory.

Adjustment Lemma. *The adjusted proposals of symmetric neighbors are connected.*

Proof. The adjusted proposals of symmetric agents i and j are $p'_i = s_i + \frac{1}{2}(p_i - s_i)$ and $p'_j = s_j + \frac{1}{2}(p_j - s_j)$. By construction $\|s_i - p_j\| \leq r$ and $\|s_j - p_i\| \leq r$, so the adjusted proposals are connected:

$$\|p'_i - p'_j\| = \|s_i - s_j + \frac{1}{2}(p_i - p_j + s_j - s_i)\| \leq \frac{1}{2}(\|s_i - p_j\| + \|s_j - p_i\|) \leq r$$

Safety Theorem. *If* Filter *is valid, the service preserves connectivity of the graph.*

Proof. Assuming Filter is valid, it suffices to prove that symmetric neighbors remain connected after one round of the algorithm. Fix symmetric neighbors i and j. If $\|p_i - p_j\| > r$, both adjust their proposals and they remain connected by the Adjustment lemma. If $\|p_i - p_j\| \leq r$ and neither adjust, they trivially remain connected. If $\|p_i - p_j\| \leq r$ but (wlog) i adjusts but j doesn't adjust, then $s_i, p_i \in \mathsf{disk}_1(p_j)$, and by convexity $p'_i \in \mathsf{disk}_1(p_j)$, whence $\|p'_i - p_j\| \leq r$.

Even if two agents are connected and propose connected targets, they might disconnect while following their trajectory to the target. Moreover, agents could stop or slow down unexpectedly (perhaps due to an obstacle) while executing the trajectories. We prove the linear trajectories prescribed by the algorithm for symmetric neighbors are *robust* in that any number of agents can stop or slow down during the execution and connectivity is preserved.

Robustness Theorem. *The linear trajectories followed by symmetric neighbors are robust.*

Proof. Fix symmetric neighbors i and j, we need to prove that all intermediate points on the trajectories are connected. Fix points $q_i := s_i + \gamma_i(p_i - s_i)$ and $q_j := s_j + \gamma_j(p_j - s_j)$ ($\gamma_i, \gamma_j \in [0,1]$) on the trajectory from each source to its proposed target. Since the neighbors are symmetric, $s_i, t_i \in \mathsf{disk}_1(s_j) \cap \mathsf{disk}_1(t_j)$ and by convexity $q_i \in \mathsf{disk}_1(s_j) \cap \mathsf{disk}_1(t_j)$. Similarly $s_j, t_j \in \mathsf{disk}_1(q_i)$ and by convexity $q_j \in \mathsf{disk}_1(q_i)$, whence $\|q_i - q_j\| \leq r$.

5 Ensuring Progress for Graphs

For the algorithm to be useful, besides preserving connectivity (proved in §4) it should also guarantee that agents make progress and eventually reach their intended destination. We start by identifying several subtle conditions without which no local algorithm could both preserve connectivity and guarantee progress.

Cycles. Consider a configuration where nodes are in a cycle, two neighboring nodes want to move apart and break the cycle and every other node wants to remain in place. Clearly no local algorithm can make progress because, without global information, nodes cannot distinguish between being in a cycle or a chain, and in the latter case any movement would violate connectivity. As long as the longest cycle of the graph is bounded by a known constant, say k, using local LMST filtering over $\lfloor k/2 \rfloor$-hops will break all cycles. A way to deal with graphs with arbitrary long cycles without completely sacrificing locality would be to use the algorithm proposed in this paper and switch to a global filtering function to break all cycles when nodes detect no progress has been made for some number of rounds. For proving progress, in the rest of the paper we assume there are no cycles in the filtered graph.

Target-connectedness. If the proposed targets are disconnected, clearly progress cannot be achieved without violating connectivity, hence its necessary to assume the target graph is connected. For simplicity, in the rest of the paper we assume that the current graph is a subgraph of the target graph, this avoids reasoning about filtering when proving progress and one can check that as a side effect the adjustment phase is never required.

5.1 Dependency Graphs

Fix some node in an execution of the $\mathsf{ConnServ}$ algorithm, on how many other nodes does its trajectory depend on? Let $\mathsf{region}(S) := \bigcap_{s \in S} \mathsf{disk}_1(s)$ and let

$\mathsf{proposal}(S,t) := \operatorname{argmin}_{p\in\mathsf{region}(S)} \|p-t\|$, then a node with filtered neighbor set N' and target t depends on k neighbors (has dependency k) if there exists a subset $S \subseteq N'$ of size $|S| = k$ such that $\mathsf{proposal}(S,t) = \mathsf{proposal}(N',t)$ but $\mathsf{proposal}(S',t) \neq \mathsf{proposal}(N',t)$ for any subset $S' \subseteq N'$ of smaller size $|S'| < k$.

The dependency of a node can be bounded by the size of its filtered neighborhood. If the filtering function is LMST then the number of neighbors is at most 6 or 5 depending on whether the distances to neighbors are unique (*i.e.* breaking ties using unique ID's). The following lemma gives a tighter upper bound on the dependency of a neighbor which is independent of the filtering function.

Lemma 6. *Every agent depends on at most two neighbors.*

Proof. Fix agent i with filtered neighbors N' and target t, let $R = \mathsf{region}(N')$. If $t \in R$ then $\mathsf{proposal}(N',t) = \mathsf{proposal}(\varnothing,t) = t$ and agent i depends on no neighbors. If $t \notin R$ then $\mathsf{proposal}(N',t)$ returns a point p in the boundary of region R. Since R is the intersection of a finite set of disks it follows that p is either in the boundary of a single disk so i depends on a single neighbor, or the intersection of two disks so i depends on at most two neighbors.

Given the above, for any configuration $C = \langle I, F\rangle$ we can consider its dependency graph $D = \langle I, E\rangle$ where there exists a directed edge $(u,v) \in E$ iff node u depends on node v. Hence, D is a directed subgraph of C with maximum out-degree 2. Moreover since graphs with cycles cannot be handled by any local connectivity service, then for the purpose of proving progress we assume C has no undirected cycles. This implies that the only directed cycles in D are simple cycles of length 2, we refer to such dependency graphs as *nice* graphs.

A *prechain* H is a sequence of vertices $\langle v_i\rangle_{i\in 1..n}$ such that there is a simple cycle between v_i, v_{i+1} $(i \in 1..n-1)$. Observe that a vertex v is a singleton prechain. Below we prove that any nice dependency graph D contains a nonempty prechain H with no out-edges.

Theorem 7. *Every finite nice graph $G = \langle V, E\rangle$ contains a nonempty prechain $H \subseteq V$ with no out-edges.*

Proof. Fix a graph $G = \langle V, E\rangle$ and consider the graph G' that results from iteratively contracting the vertices $u, v \in V$ if $(u,v) \in E$ and $(v,u) \in E$. Clearly G' is also a finite nice graph and any vertex v' in G' is a prechain of G, however G' does not contain any directed cycles. 有向循环

We follow a directed path in G' starting at an arbitrary vertex u', since the graph is finite and contains no cycles, we must eventually reach some vertex v' with no outgoing edges, such a vertex is a prechain and has no outgoing edges, which implies the theorem.

Therefore by theorem 7 any lower bound on progress for chains also holds for general configurations. In particular the lower bound of $\Omega(\min(d,r))$ for chains proved in the next section applies for general graphs as well.

8 Ensuring Progress for Chains

In this section we restrict our attention to chain configurations and show that, if agents execute the connectivity service's refined plan, the total progress of the configuration is at least $\min(d, r)$, where d is the minimum distance between any agent and its target and r is the communication radius. We introduce some terminology to classify chains according to their geometric attributes, then we prove the progress bound for a very restricted class of chains. Finally, we establish the result for all chains by showing that the progress of an arbitrary chain is bounded below by the progress of a restricted chain.

Terminology. Each agent has a local coordinate system where the source is the origin ($s_i = \langle 0, 0\rangle$) and the target is directly above it ($t_i = \langle 0, d_i\rangle$). The left side of agent i is defined as $L_i := \{\langle x, y\rangle : x \leq 0\}$ and the right side as $R_i := \{\langle x, y\rangle : x > 0\}$ where points are relative to the local coordinate system. An agent in a chain is *balanced* if it has one neighbor on its left side, and the other on its right side; a configuration is balanced if every agent is balanced.

A configuration is *d-uniform* if every agent is at distance d from its target ($d_i = d$ for every agent i). Given a pair of agents i and j, they are *source-separated* if $\|s_i - s_j\| = 1$; they are *target-separated* if $\|s_i - s_j\| = 1$; and they are *target-parallel* if the rays $\mathsf{ray}(s_i, t_i)$ and $\mathsf{ray}(s_j, t_j)$ are parallel. An agent i with neighbors j and k is *straight* if s_i, s_j and s_k are collinear; a chain configuration is straight if all agents are straight.

Given an agent with source s, target t and a (possibly empty) subset of neighbors $S \subseteq N$, its proposed target w.r.t. S is defined as $t^* = \mathsf{proposal}(S, t)$. The *progress* of the agent would be $\delta(s, t; S) := \|s - t\| - \|t^* - t\|$, which we abbreviate as δ_i for agent i when the s_i, t_i and S_i are clear from context. Observe that since $\mathsf{region}(S \cup S') \subseteq \mathsf{region}(S)$, $\delta(s, t; S \cup S') \leq \delta(s, t; S)$. The *progress* of a configuration C is the sum every agent's progress: $\mathsf{prog}(C) := \sum_i \delta_i$.

Proof Overview. We first characterize the progress of agents in a balanced and source-separated chain and show the progress bound specifically for chains that are d-uniform, source- and target-separated, balanced, and straight (§8.2). Then we show how to remove each of the requirements of a chain being straight, balanced, source- and target-separated, and d-uniform (§8.3). Ultimately, this means that an *arbitrary* target-connected chain configuration $C = \langle I, F\rangle$ can be transformed into a d-uniform, source- and target-separated, balanced, straight chain configuration C' such that $\mathsf{prog}(C) \geq \mathsf{prog}(C') \geq \min(d, r)$, where d is the minimum distance between each source and its target in the original configuration C ($d := \min_{i \in I} \|s_i - t_i\|$) and the communication radius is r. At each removal step we show that imposing a particular constraint on a more relaxed configuration does not increase progress, so that the lower bound for the final (most constrained) configuration is also a lower bound for the original (unconstrained) configuration. The bound shows that straight chains (the most constrained configurations) are the worst-case configurations since their progress is a lower bound for all chains. We show the lower bound is tight for d-uniform configurations by exhibiting a chain with progress exactly $\min(d, r)$ (§8.3).

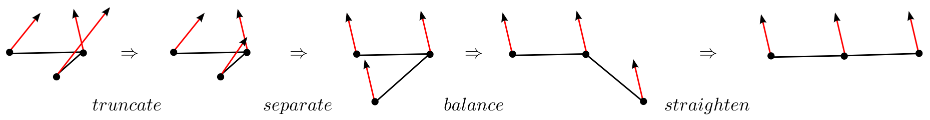

Fig. 1. Transformation overview from arbitrary to restricted chains

8.1 Progress Function for Balanced and Separated Chains

We explicitly characterize the progress of an agent in a balanced, source-separated chain. In such a configuration, if an agent has source s with target t, the source-target distance is $d := \|s-t\|$ and the position of its neighbors s_{-1}, s_{+1} (if any) can be uniquely determined by the angles of the left ($\lambda := \angle t, s, s_{-1}$) and right neighbor ($\rho := \angle t, s, s_{+1}$). Since an agent's progress is determined by it's neighbors, its progress can be defined as a function $\delta^{\angle}(d, \lambda, \rho)$.

If the agent doesn't depend on either neighbor, it can immediately move to its target and its progress is d. If it (partially) depends on a single (left or right) neighbor at angle θ, then progress is $\delta^{\mathsf{single}}(d, \theta) := d + 1 - \sqrt{1 + d^2 - 2d\cos\theta}$. If it (partially) depends on both neighbors at angles ρ and λ, then progress is $\delta^{\mathsf{both}}(d, \lambda, \rho) := d - \sqrt{2 + d^2 - 2d\cos\rho + 2\cos(\rho+\lambda) - 2d\cos\lambda}$. If completely immobilized by one or both of its neighbors, its progress is 0. Therefore the progress of an agent is described by the following piecewise function, parametrized by the source-target distance d and the angle to its neighbors ρ and λ. Observe that the agent i's progress function is monotonically decreasing in ρ and λ.

$$\delta^{\angle}(d, \lambda, \rho) = \begin{cases} d & \rho \le \cos^{-1}\frac{d}{2} \text{ and } \lambda \le \cos^{-1}\frac{d}{2} \\ \delta^{\mathsf{single}}(d, \rho) & \rho > \cos^{-1}\frac{d}{2} \text{ and } \sin(\rho+\lambda) \ge d\sin\lambda \\ \delta^{\mathsf{single}}(d, \lambda) & \lambda > \cos^{-1}\frac{d}{2} \text{ and } \sin(\rho+\lambda) \ge d\sin\rho \\ \delta^{\mathsf{both}}(d, \lambda, \rho) & \rho+\lambda < \pi \text{ and } \sin(\rho+\lambda) < d\sin\rho, d\sin\lambda \\ 0 & \rho+\lambda \ge \pi \end{cases}$$

8.2 Progress for Restricted Chains

We prove a lower bound on progress of $\min(d, r)$ for d-uniform, source- and target-separated, balanced, straight chains with communication radius r. Let $C_k(d, \theta)$ represent a d-uniform, source- and target-separated, straight chain of k nodes, where $\angle t_i, s_i, s_{i+1} = \theta$ for $i \in 1..n-1$. We first establish the progress bound for chains of two nodes and then extend it to more than two nodes.

Progress Theorem for Restricted 2-Chains. *For any $\theta \in [0, \pi]$, the chain $C_2(d, \theta)$ makes progress at least $\min(d, r)$ ($\mathsf{prog}(C_2(d, \theta)) \ge \min(d, r)$).*

Proof. Suppose $\theta \le \arccos\frac{d}{2}$, then if $d \le r$ agent 1 makes progress d, if $d > r$ then agent 1 makes progress at least r. Similarly if $\theta \ge \pi - \arccos\frac{d}{2}$ and $d \le r$

agent 2 makes progress d, if $d > r$ then agent 2 makes progress at least r. Otherwise $\theta \in (\arccos \frac{d}{2}, \pi - \arccos \frac{d}{2})$ and the progress function from §8.1 yields

$$\delta^{\mathsf{single}}(d,\theta) + \delta^{\mathsf{single}}(d,\pi-\theta) = 2 + 2d - \sqrt{1+d^2-2d\cos\theta} - \sqrt{1+d^2+2d\cos\theta}$$

The partial derivative is $d\sin\theta(1/\sqrt{1+d^2+2d\cos\theta} - 1/\sqrt{1+d^2-2d\cos\theta})$, whose only root in $(0,\pi)$ is $\theta = \frac{\pi}{2}$, which is a local minimum. We use the first order Taylor approximation as an upper bound of $\sqrt{1+d^2}$ and since $d^2 < d$:

$$\begin{aligned}\mathsf{prog}(C_2(d,\theta)) &\geq \mathsf{prog}(C_2(d,\frac{\pi}{2})) \geq 2\delta^{\mathsf{single}}(d,\frac{\pi}{2}) \\ &\geq 2 + 2d - 2\sqrt{1+d^2}) \geq 2 + 2d - 2 - d^2 \geq d\end{aligned}$$

Progress Theorem for Restricted n-Chains. *Configurations $C_n(d,\theta)$ $(n > 2)$ and $C_2(d,\theta)$ have the same progress ($\mathsf{prog}(C_n(d,\theta)) = \mathsf{prog}(C_2(d,\theta))$).*

Proof. Since C_n is straight and separated, internal nodes make no progress ($\delta_i = 0$ for $i \in 2..n-1$). The first node in C_n (and C_2) has a single neighbor at angle θ, so $\delta_1^n = \delta_1^2$. Similarly the last node in C_n (and C_2) has a single neighbor at angle $\pi - \theta$, so $\delta_n^n = \delta_2^2$. Therefore $\mathsf{prog}(C_n(d,\theta)) = \mathsf{prog}(C_2(d,\theta))$.

8.3 Progress for Arbitrary Chains

We prove that the progress of an *arbitrary* chain is bounded by below by the progress of a restricted chain, hence the progress bound proved in the previous section for restricted chains extends to all chains. Furthermore, we show the bound is tight for d-uniform configurations by exhibiting a class of chains for which progress is exactly $\min(d,r)$.

To extend the progress result from restricted to arbitrary chains, we exhibit a sequence of transformations (cf. Fig 1) that show how to transform an arbitrary chain to be d-uniform, source-separated, target-separated, balanced and straight. Each transformation doesn't increase progress and preserves the configuration's properties. The proofs rely heavily on geometric reasoning and are the most technical part of the progress bound. Due to space restrictions we list the lemmas without proof, see [21] for the detailed proofs.

Truncation Lemma. *Suppose a source s with target t and neighbors S. Let $t^T = s + \gamma(t-s)$ with $\gamma \in [0,1]$ be its truncated target, then $\delta(s,t;S) \geq \delta(s,t^T;S)$.*

Separation Lemma. *A d-uniform configuration C can be transformed into a d-uniform, source- and target-separated configuration C' with $\mathsf{prog}(C) \geq \mathsf{prog}(C')$.*

Balancing Lemma. *Fix a configuration C where agent i has neighbors $i-1$ and $i+1$ on the same side. Let C' be the configuration obtained by reflecting every s_j and t_j for $j > i$ (or $j < i$) around agent i's y-axis. Then $\mathsf{prog}(C) \geq \mathsf{prog}(C')$.*

Straightening Lemma. *Fix a configuration C described by $\{\theta_i\}_{i\in 1..n-1}$ and a straight configuration C' described by $\{\theta'_i\}_{i\in 1..n-1}$ where every angle is θ_{n-1} ($\theta'_i := \theta_{n-1}$ for $i \in 1..n$). Then $\mathsf{prog}(C) \geq \mathsf{prog}(C')$.*

With these transformations in place we are ready to prove a bound on the progress of an arbitrary chain.

Progress Theorem for Chains. *The progress of a chain $C = \langle I, F\rangle$ is $\mathsf{prog}(C) \geq \min(\min_{i\in I} d_i, r)$.*

Proof. By the Truncation lemma we can set all the source-target distances to $d = \min(\min_{i\in I} d_i, r)$ to obtain a d-uniform chain. Using the Separation, Balancing, and Straightening lemmas there exists an angle $\theta \in [0, \pi]$ such that the straight chain $C_n(d, \theta)$ has less progress than C ($\mathsf{prog}(C) \geq \mathsf{prog}(C_n(d, \theta))$).

Finally by the Progress theorem for straight n-chains we have $\mathsf{prog}(C_n(d, \theta)) = \mathsf{prog}(C_2(d, \theta))$, and by the progress lemma progress for 2-chains we have $\mathsf{prog}(C_2(d, \theta)) \geq d$ for any θ. Hence, $\mathsf{prog}(C) \geq \mathsf{prog}(C_n(d, \theta)) = \mathsf{prog}(C_2(d, \theta)) \geq d$.

Optimality Theorem. *The lower bound on progress is tight for d-uniform configurations: there are chains that cannot make more than $\min(d, r)$ progress under any local service, and $\mathsf{ConnServ}$ achieves exactly that much progress.*

Proof. For any n, we exhibit a chain of n agents with progress exactly $\min(d, r)$. Fix n and consider the straight chain $C_n(d, 0)$, the first agent has progress $\min(d, r)$ ($\delta_1^n = \min(d, r)$) while every other agent has no progress ($\delta_i^n = 0$ for $i > 1$), therefore $\mathsf{prog}(C_n(0, d)) = \min(d, r)$. This class of of chains cannot make more than $\min(d, r)$ progress under any local service and $\mathsf{ConnServ}$ achieves exactly that much progress.

9 Termination

Consider an arbitrary chain of agents running the connectivity service. How many rounds does it take the agents to get (arbitrarily close) to their target? Let $d_i[k]$ be the source-target distance of agent i after round k, we say an agent is ε-close to its target at round k iff $d_i[k] \leq \varepsilon$. Given the initial source-target distance $d_i[0]$ of each agent, we will give an upper bound on k to guarantee every agent is ε-close.

So far we proved that while the target of every agent is outside its communication radius r, the collective distance traveled is r; moreover this is tight up to a constant factor. However, once an agent has its target within its communication radius, we can only argue that collective progress is proportional to the smallest source-target distance (since we truncate to the smallest distance). Unfortunately this is not enough to give an upper bound on k.

Let $D_k = \sum_i d_i[k]$ and $d_{\min}[k] = \min_i d_i[k]$, then $D_{k+1} \leq D_k - \min(d_{\min}[k], r)$. However, if $d_{\min}[k] = 0$ this yields $D_{k+1} \leq D_k$ and we cannot prove termination. The following lemma allows us to sidestep this limitation. We call a chain *almost*

d-uniform if all the inner nodes are d-uniform and the outermost nodes have source-target distance 0.

Progress Theorem for Almost-Uniform Chains. *An* almost *d-uniform chain C_n of size $n \geq 3$ has progress* $\mathsf{prog}(C_n) \geq \delta^{\angle}(d, \frac{\pi}{2}, \arccos\frac{d}{2}) \geq \gamma_0 d$ *where* $\gamma_0 := 1 - \sqrt{2-\sqrt{3}}$.

Proof. Observe that the Balancing and Separation theorems still apply. Moreover, by the independence lemma and the monotonicity of the progress function we can assume the endpoints are at an angle of $\arccos\frac{d}{2}$ to their neighboring source-target vector.

Hence, for $n = 3$ we need to consider one configuration, and by the target-connectedness assumption it's clear that the inner node makes full progress and hence $\mathsf{prog}(C_3) \geq d$. For $n > 3$ there is a family of possible chains determined by the angles between the inner nodes, we proceed by a complete induction on n. Observe that we can assume the progress of the internal nodes depends on both of its neighbors, since otherwise we could argue about a smaller subchain.

Case 1. Base case. Let $n = 4$, clearly only the two internal nodes make progress, therefore we have $\mathsf{prog}(C_4) = \delta^{\mathsf{both}}(d, \arccos\frac{d}{2}, \alpha) + \delta^{\mathsf{both}}(d, \pi-\alpha, \arccos\frac{d}{2})$ where α is the angle between the two internal nodes. If $\alpha \leq \arccos\frac{d}{2}$ or $\pi-\alpha \leq \arccos\frac{d}{2}$, then $\mathsf{prog}(C_4) \geq d$. For $\arccos\frac{d}{2} \leq \alpha \leq \pi - \arccos\frac{d}{2}$ we define the restricted minimization problem $\alpha^* = \operatorname{argmin}_\alpha \mathsf{prog}(C_4)$. There is a unique minimum at $\alpha^* = \frac{\pi}{2}$ and hence $\mathsf{prog}(C_4) \geq 2\delta^{\angle}(d, \frac{\pi}{2}, \arccos\frac{d}{2}) \geq \gamma_0 d$.

Case 2. Inductive step. Consider a chain of length $n > 4$ with $n-2$ interior nodes. Let S be the set of angles between the first $n-3$ interior nodes and let α be the angle between the last interior nodes. The progress of the chain is $\mathsf{prog}(C_n) = p(S,\alpha) + \delta^{\angle}(d, \alpha, \arccos\frac{d}{2})$, where $p(S,\alpha)$ represents the progress of the first $n-3$ interior nodes. Similarly for a chain of length $n+1$ there are $n-1$ interior nodes, and its progress is $\mathsf{prog}(C_{n+1}) = p(S,\alpha) + \delta^{\angle}(d,\alpha,\beta) + \delta^{\angle}(d, \pi-\beta, \arccos\frac{d}{2})$.

We prove the bound by cases on α. If $\alpha \leq \frac{\pi}{2}$, we can minimize the last two terms of $\mathsf{prog}(C_{n+1})$ by solving $\min_{\alpha,\beta} \delta^{\angle}(d,\alpha,\beta) + \delta^{\angle}(d,\pi-\beta,\arccos\frac{d}{2})$, which has a single minimum at $\alpha = \beta = \frac{\pi}{2}$, and thus $\mathsf{prog}(C_{n+1}) = p(S,\alpha) + \delta^{\angle}(d,\alpha,\beta) + \delta^{\angle}(d,\pi-\beta,\arccos\frac{d}{2}) \geq \delta^{\angle}(d,\frac{\pi}{2},\frac{\pi}{2}) + \delta^{\angle}(d,\frac{\pi}{2},\arccos\frac{d}{2}) \geq \gamma_0 d$.

If $\alpha > \frac{\pi}{2}$, by the inductive hypothesis we have $\mathsf{prog}(C_n) \geq \gamma_0 d$ and it suffices to show $\mathsf{prog}(C_{n+1}) \geq \mathsf{prog}(C_n)$. This is equivalent to proving $\delta^{\angle}(d,\alpha,\beta) + \delta^{\angle}(d,\pi-\beta,\arccos\frac{d}{2}) - \delta^{\angle}(a,\alpha,\arccos\frac{d}{2}) \geq 0$ for $\alpha > \frac{\pi}{2}$ and any β, which also holds.

Intuitively, the progress theorem for almost-uniform chains proves that once subset of the agents reach their target, the rest of the agents make almost the same progress as before. Intuitively, it seems reasonable to expect that if a subset of the agents get ε-close to their target (for small enough ε) a similar result should hold. This is at the core of the termination theorem which proves an upper bound on the number of rounds needed for nodes to be ε-close to their targets.

We say the targets of two nodes are ℓ-connected if they are at distance ℓ of each other. So far we have assumed neighboring nodes have connected targets, that is, they are r-connected. To prove the next theorem we require a stronger assumption, namely, that targets are $(r - 2\varepsilon)$-connected.

Termination Theorem. *Under the $(r - 2\varepsilon)$-connected assumption, nodes get ε-close within $O(D_0/r + n^2/\varepsilon)$ rounds.*

Proof. Since targets are $(r - 2\varepsilon)$-close, we can assume each node stops at the first round when they are ε-close to their target and the resulting configuration is connected. Therefore we can consider the source-target distance of a node to be either greater than ε when it is not ε-close, or zero once it is ε-close.

If initially every node i is at distance $d_i \geq r$ from its target, it takes at most D_0/r rounds before there exists some node i with $d_i < r$. If there is a node i with source-target distance $d_i < r$ it follows that $D_k < r\frac{n^2}{2}$, we argue that from this point on we can assume a progress of at least $\gamma_0\varepsilon$ per round until every node reaches its target, therefore the total number of rounds is $O(D_0/r + n^2/\varepsilon)$.

Consider a chain $C = \langle I, F\rangle$ and let the subset $S_k \subseteq I$ represent the set of agents which are already at their target at round k ($i \in S_k$ iff $d_i[k] = 0$). If $S_k = I$ then we are done, otherwise there exists a subchain $C' \subseteq C$ where all agents except possibly the endpoints have $d_i[k] > \varepsilon$. Hence, by the progress theorem for almost-uniform chains the progress is at least $\gamma_0\varepsilon$.

10 Conclusion

In this paper we present a local, oblivious connectivity service (§3) that encapsulates an arbitrary motion planner and can refine any plan to preserve connectivity (the graph of agents remains connected) and ensure progress (the agents advance towards their goal). We prove the algorithm not only preserves connectivity, but also produces robust trajectories so if an arbitrary number of agents stop or slow down along their trajectories the graph will remain connected (§4).

We also prove a tight lower bound of $\min(d, r)$ on progress for d-uniform configurations (§8). The truncation lemma allows this lower bound to apply to general configurations using the minimum distance between any agent and its goal. Thus, when each agent's target is within a constant multiple of the communication radius, the lower bound implies the configuration will move at a constant speed towards the desired configuration.

As the agents get closer to their goal, this bound no longer implies constant speed convergence. We prove a bound of $O(D_0/r + n^2/\varepsilon)$ on the number of rounds until nodes are ε-close. This bound requires assuming targets are $(r-2\varepsilon)$-connected, though we conjecture that it is possible to remove this assumption. The D_0/r term in the bound is necessary because when the initial source-target distance is large enough, clearly no service can guarantee robust, connected trajectories if agents advance faster than one communication radius per round.

It would be tempting to prove agents advance at a rate proportional to the mean (instead of the minimum) source-target distance, which would imply a termination bound of $O(D_0/r + n \log \frac{n}{\varepsilon})$. However, it is possible to construct an example which shows that the progress is less than $\gamma \cdot mean$, for any constant $\gamma > 0$. An alternative approach we intend to pursue in future work is to directly argue about the number of rounds it takes the agents to reach their target. This may give a tighter bound on the rate of convergence over quantifying the distance traveled by the agents in a single round, which necessarily assumes a worst case configuration at every step.

References

1. Johnson, D., Maltz, D.: Dynamic Source Routing in Ad Hoc Wireless Networks. Computer Communications Review - SIGCOMM (1996)
2. Perkins, C., Royer, E.: Ad-hoc On-Demand Distance Vector Routing. In: Workshop on Mobile Computing Systems and Applications (1999)
3. Malpani, N., Welch, J., Vaidya, N.: Leader Election Algorithms for Mobile Ad-hoc Networks. In: DIAL-M: Workshop in Discrete Algorithms and Methods for Mobile Computing and Communications (2000)
4. Walter, J., Welch, J., Vaidya, N.: A Mutual Exclusion Algorithm for Ad Hoc Mobile Networks. Wireless Networks (2001)
5. Regmi, A., Sandoval, R., Byrne, R., Tanner, H., Abdallah, C.: Experimental Implementation of Flocking Algorithms in Wheeled Mobile Robots. In: Proceedings of the American Control Conference 2005, pp. 4917–4922 (2005)
6. Hayes, A., Dormiani-Tabatabaei, P.: Self-organized flocking with agent failure: Offline optimization and demonstration with real robots. In: ICRA (2002)
7. Fierro, R., Das, A.: A modular architecture for formation control. Robot Motion and Control (2002)
8. Carpin, S., Parker, L.E.: Cooperative Leader Following in a Distributed Multi-Robot System. In: ICRA (2002)
9. Zavlanos, M.M., Pappas, G.J.: Controlling Connectivity of Dynamic Graphs. In: CDC-ECC, pp. 6388–6393 (2005)
10. Savla, K., Notarstefano, G., Bullo, F.: Maintaining limited-range connectivity among second-order agents. SIAM Journal on Control and Optimization (2007)
11. Ando, H., Oasa, Y., Suzuki, I., Yamashita, M.: Distributed memoryless point convergence algorithm for mobilerobots with limited visibility. ICRA 15(5), 818–828 (1999)
12. Flocchini, P., Prencipe, G., Santoro, N., Widmayer, P.: Gathering of asynchronous robots with limited visibility. Theor. Comput. Sci. 337(1-3), 147–168 (2005)
13. Ganguli, A., Cortés, J., Bullo, F.: Multirobot rendezvous with visibility sensors in nonconvex environments. CoRR abs/cs/0611022 (2006)
14. Souissi, S., Défago, X., Yamashita, M.: Using eventually consistent compasses to gather oblivious mobile robots with limited visibility. In: SSS, pp. 484–500 (2006)
15. Lee, G., Chong, N.Y., Defago, X.: Robust Self-Deployment for a Swarm of Autonomous Mobile Robots with Limited Visibility Range. In: Robot and Human interactive Communication (2007)
16. Maja, A.H., Howard, A., Matari, M.J., Sukhatme, G.S.: An Incremental Self-Deployment Algorithm for Mobile Sensor Networks. Autonomous Robots, Special Issue on Intelligent Embedded Systems 13, 113–126 (2001)

17. Cornejo, A., Lynch, N.: Connectivity Service for Mobile Ad-Hoc Networks. In: Spatial Computing Workshop (2008)
18. Gabriel, K., Sokal, R.: A new statistical approach to geographic variation analysis. Systematic Zoology 18(3), 259–278 (1969)
19. Toussaint, G.T.: The relative neighbourhood graph of a finite planar set. Pattern Recognition 12(4), 261–268 (1980)
20. Li, N., Hou, J.C., Sha, L.: Design and analysis of an MST-based topology control algorithm. INFOCOM 3, 1702–1712 (2003)
21. Cornejo, A., Kuhn, F., Lynch, N., Ley-Wild, R.: Keeping mobile robot swarms connected. MIT-CSAIL-TR-2009-027 (2009), `http://hdl.handle.net/1721.1/45568`