Target assignment for robotic networks: asymptotic performance under limited communication

Stephen L. Smith Francesco Bullo

***Abstract*— We are given an equal number of mobile robotic agents, and distinct target locations. Each agent has simple integrator dynamics, a limited communication range, and knowledge of the position of every target. We address the problem of designing a distributed algorithm that allows the group of agents to divide the targets among themselves and, simultaneously, leads each agent to reach its unique target. We do not require connectivity of the communication graph at any time. We introduce a novel assignment-based algorithm with the following features: initial assignments and robot motions follow a greedy rule, and distributed refinements of the assignment exploit an implicit circular ordering of the targets. We prove correctness of the algorithm, and give worst-case asymptotic bounds on the time to complete the assignment as the environment grows with the number of agents. We show that among a certain class of distributed algorithms, our algorithm is asymptotically optimal. The analysis utilizes results on the Euclidean traveling salesperson problem.**

I. INTRODUCTION

Consider a group of n mobile robotic agents and n target locations, all lying in $\mathbb{R}^d$, $d \geq 1$. Each agent has a limited communication range, and knows the location of some subset (possibly all) of the n targets through GPS coordinates or a map of the environment. The *target assignment problem* is to design a distributed algorithm that allows the group of agents to divide the n targets among themselves and, simultaneously, that leads each agent to reach its unique target. Such a problem could arise in several applications. For example, one could think of the agents as UAV's on a surveillance mission, and the targets as the centers of their desired loitering patterns. Or, this behavior could be used to stabilize a group of agents to any desired formation.

The first question is; how do we divide the targets among the agents in a centralized fashion? A reasonable strategy would be to minimize the sum of the distances traveled by each agent to arrive at its target. The problem of optimally dividing n persons among n objects, subject to a linear cost function, is a problem in combinatorial optimization [1]. It is referred to as the *assignment problem*, or the *minimum weight perfect matching problem in bipartite graphs*. The assignment problem can be written as an integer linear program. Unlike some integer linear programs, such as the Euclidean traveling salesperson problem (ETSP), optimal solutions for the assignment problem can be computed in polynomial time. In 1955 Kuhn [2] developed the Hungarian method—the first polynomial solution for the assignment problem. Kuhn's method solves the problem in $O(n^3)$ time (see Section II for a definition of the O notation). Since 1955, many other methods have been developed. The most efficient method runs in $O(n^2 \log n)$ time, and is achieved by reducing the problem to a *network flow problem* [1].

Another approach to the assignment problem is the *auction algorithm* [3], [4], [5], first proposed by Bertsekas. This method solves the problem in $O(n^3)$ time, but can be computed in a parallel fashion, with one processor for each person. Recently, Moore and Passino [6] modified the auction algorithm to assign mobile robots to spatially distributed tasks in the presence of communication delays. However, in order to exchange bids on a particular object (task), the auction algorithm, and thus the work in [6], requires that the communication graph between processors (robots) is complete. In addition, the auction algorithm requires the election of a "leader" processor to manage the auction for each of the objects; this potentially leads to more complex and less scalable implementations.

In this paper we address the task assignment problem when each agent has knowledge of all target positions, and a limited communication range $r > 0$. We introduce a class of distributed algorithms, called *assignment-based motion*, which provide a natural approach to the problem. Following the recent interest in determining the time complexity of distributed algorithms for robotic networks, as in [7] and [8], we study the worst-case asymptotic performance of the assignment-based motion class as the environment grows with n. We show that for a d-dimensional cube environment, $[0, \ell(n)]^d$, $d \geq 1$, if the side length $\ell(n)$ grows at a rate of at least $(1+\epsilon)rn^{1/d}$, where $\epsilon > 0$, then the time complexity is in $\Omega(n^{(d-1)/d}\ell(n))$, for all algorithms in this class.

In Section V we introduce a novel control and communication algorithm, called ETSP ASSIGNMENT. In this algorithm, each agent computes an ETSP tour through the n targets, turning the cloud of target points into an ordered ring. Agents then move along the ring, looking for the next available target. When agents communicate, they exchange information on how far it is to the next available target along the ring. In Section V-A, we verify the correctness of this algorithm for any communication graph which contains, as a subgraph, the r-disk graph. In Section V-B, we show that when $\ell(n) \geq (1+\epsilon)rn^{1/d}$, among all algorithms in the assignment-based motion class, the ETSP ASSIGNMENT algorithm is asymptotically optimal (i.e., a constant factor approximation of the optimal). Finally, in Section V-E, we note that ETSP ASSIGNMENT solves the target assignment problem even when there are n agents and m targets, $n \neq m$.

Submitted to 2007 ACC on September 26, 2006. This material is based upon work supported in part by ARO MURI Award W911NF-05-1-0219 and NSF SENSORS Award IIS-0330008.

Stephen L. Smith and Francesco Bullo are with the Department of Mechanical Engineering, and the Center for Control, Dynamical Systems and Computation, University of California, Santa Barbara, CA 93106, USA, {stephen,bullo}@engineering.ucsb.edu

II. Background

In this section we introduce notation and review some relevant results in combinatorial optimization.

A. Notation

We let $\mathbb{R}$ denote the set of real numbers, $\mathbb{R}_{>0}$ denote the set of positive real numbers, and $\mathbb{N}$ denote the set of positive integers. For a set A we let $|A|$ denote the cardinality of the set. For two functions $f, g : \mathbb{N} \to \mathbb{R}_{>0}$, we write $f(n) \in O(g)$ (respectively, $f(n) \in \Omega(g)$) if there exist $N \in \mathbb{N}$ and $c \in \mathbb{R}_{>0}$ such that $f(n) \leq cg(n)$ for all $n \geq N$ (respectively, $f(n) \geq cg(n)$ for all $n \geq N$). If $f(n) \in O(g)$ and $f(n) \in \Omega(g)$ we say $f(n) \in \Theta(g)$. Finally, we use the notation $\pmod n$ to denote arithmetic performed modulo $n \in \mathbb{N}$. Thus, for an integer $n \in \mathbb{N}$ we have $n+1 = 1 \pmod n$ and $0 = n \pmod n$, and $\{n-1, n, n+1\} = \{n-1, n, 1\} \pmod n$.

B. The assignment problem

Following [4], the classical assignment problem can be described as follows. Consider n persons who wish to divide themselves among n objects. For each person i, there is a nonempty set $\mathcal{Q}^{[i]}$ of objects that i can be assigned to, and cost $c_{ij} \geq 0$ associated to each object $j \in \mathcal{Q}^{[i]}$. An *assignment* S is a set of person-object pairs (i,j) such that $j \in \mathcal{Q}^{[i]}$ for all $(i,j) \in S$. For each person i (likewise, object j), there is at most one pair $(i,j) \in S$. We call the assignment *complete* if it contains n pairs. The goal is to find the complete assignment which minimizes $\sum_{(i,j)\in S} c_{ij}$.

Let x_{ij} be a set of variables for i and j in $\mathcal{I} := \{1,\dots,n\}$. For an assignment S, we write $x_{ij} = 1$ if $(i,j) \in S$, and $x_{ij} = 0$ otherwise. Thus, the problem of determining the optimal assignment can be written as a linear program:

$$\begin{aligned} \text{minimize} \quad & \sum_{i=1}^{n} \sum_{j \in \mathcal{Q}^{[i]}} c_{ij} x_{ij}, \\ \text{subject to} \quad & \sum_{j \in \mathcal{Q}^{[i]}} x_{ij} = 1 \quad \forall\, i \in \mathcal{I}, \\ & \sum_{\{i | j \in \mathcal{Q}^{[i]}\}} x_{ij} = 1 \quad \forall\, j \in \mathcal{I}, \\ & x_{ij} \geq 0. \end{aligned}$$

We cannot use linear inequalities to write the constraint that x_{ij}'s attain only the values zero and one. However, it turns out, [4], that there always exists an optimal solution in which the x_{ij}'s satisfy our integer assumption.

C. The Euclidean traveling salesperson problem

Here we review some relevant results on the Euclidean traveling salesperson problem (ETSP). Let $\mathcal{Q}$ be a set of n points in a compact environment $\mathcal{E} \subset \mathbb{R}^d$, $d \geq 1$, and let $\mathcal{Q}_n$ be the set of all point sets $\mathcal{Q} \subset \mathcal{E}$ with $|\mathcal{Q}| = n$. Let $\mathrm{ETSP}(\mathcal{Q})$ denote the cost of the ETSP tour over the point set $\mathcal{Q}$, i.e., the length of the shortest closed path through all points in $\mathcal{Q}$. An important result, from [9], is that given a compact set $\mathcal{E}$, there exists a finite constant $\alpha(\mathcal{E})$ such that, for all $\mathcal{Q} \in \mathcal{Q}_n$,

$$\mathrm{ETSP}(\mathcal{Q}) \leq \alpha(\mathcal{E}) n^{(d-1)/d}. \tag{1}$$

In fact, we have that in the worst-case setting, the $\mathrm{ETSP}(\mathcal{Q})$ belongs to $\Theta(n^{(d-1)/d})$.

In our application of these results it will be useful to consider the case where the environment grows with the number of points. That is, we are interested in environments which are cubes, $[0, \ell(n)]^d$, $d \geq 1$, where $\ell(n)$ is the side length of the cube. Applying a simple scaling argument to the result in (1), we arrive at the following corollary.

Corollary 2.1 (ETSP tour length): Consider an environment $\mathcal{E} = [0, \ell(n)]^d$, where $d \geq 1$. For every point set $\mathcal{Q} \in \mathcal{Q}_n$,

$$\mathrm{ETSP}(\mathcal{Q}) \in \Theta(n^{(d-1)/d} \ell(n)).$$

The problem of computing an optimal tour is known to be NP-complete. However, there exist heuristics which can be computed efficiently and give a constant factor approximation to the optimal tour. The best known approximation algorithm is due to Christofides [10]. The *Christofides' algorithm* computes a tour that is no more than $3/2$ times longer than the optimal. It runs in time $O(n^3)$. Another method, known as the *double-tree algorithm*, produces tours that are no longer than twice the optimal, in run time $O(n^2)$.

III. Problem formulation

To describe the target assignment problem formally, consider n agents in an environment $\mathcal{E}(n) \subset \mathbb{R}^d$, $d \geq 1$. The environment $\mathcal{E}(n)$ is compact for each n but may grow with the number of agents. For ease of presentation let $\mathcal{E} := [0, \ell(n)]^d$, where $\ell(n) > 0$ (that is, $\mathcal{E}$ is a d-dimensional cube with side length $\ell(n)$). Each agent has a unique identifier (UID) taken from the set $I_{UID} \subseteq \mathbb{N}$. For simplicity, we assume that $I_{UID} := \mathcal{I} = \{1,\dots,n\}$. However, each agent does not know the set of UIDs being used (i.e., agent n does not know it has the largest UID). Agent $i \in \mathcal{I}$ has position $\mathbf{p}^{[i]} \in \mathcal{E}$. Two agents, i and k in $\mathcal{I}$, are able to communicate if and only if $\|\mathbf{p}^{[i]} - \mathbf{p}^{[k]}\| \leq r$, where $r > 0$ is called the communication range. We refer to the graph representing the communication links as the r-disk graph. Agent i's kinematic model is $\dot{\mathbf{p}}^{[i]} = \mathbf{u}^{[i]}$, where $\mathbf{u}^{[i]}$ is a velocity control input bounded by $v > 0$. We assume that the agents move in continuous time and communicate according to a discrete time communication schedule consisting of an increasing sequence of time instants with no accumulation points, $\{t_k\}_{k\in\mathbb{N}}$. We assume that $|t_{k+1} - t_k| \leq t_{max}$, for all $k \in \mathbb{N}$, where $t_{max} \in \mathbb{R}_{>0}$. At each communication round, agents can exchange messages of length $O(\log n)$. [1] We assume that communication round k occurs at time t_k, and that all messages are sent and received instantaneously at t_k. Motion then occurs from t_k until t_{k+1}. It should be noted that in this setup we are emphasizing the time complexity due to the motion of the agents.

Let $\mathcal{Q} := \{\mathbf{q}_1, \dots, \mathbf{q}_n\}$ be a set of distinct target locations, $\mathbf{q}_j \in \mathcal{E}$ for each $j \in \mathcal{I}$. Agent i is equipped with memory

[1] The number of bits required to represent an ID, unique among n agents, grows with the logarithm of n.

$M^{[i]}$, of size $|M^{[i]}|$. In this memory, agent i stores a set of target positions, $\mathcal{Q}^{[i]} \subseteq \mathcal{Q}$. These are the targets to which agent i can be assigned. We let $\mathcal{Q}^{[i]}(0)$ denote agent i's initial target set. These positions may be known through GPS coordinates, or through a map of the environment.

In this paper we assume that each agent knows the position of every target. That is, $\mathcal{Q}^{[i]}(0) = \mathcal{Q}$ for each $i \in \mathcal{I}$. We refer to this as the *full knowledge* assumption. To store this amount of information we must assume that the size of each agents' memory, $|M^{[i]}|$, grows linearly with n. Our goal is to solve the *full knowledge target assignment problem*:

> Determine a control and communication law for $n \in \mathbb{N}$ agents, with attributes as described above, satisfying the following requirement. There exists a time $T > 0$ such that for every agent $i \in \mathcal{I}$, there is a unique target $\mathbf{q}_{j_i} \in \mathcal{Q}^{[i]}(0)$ with $\mathbf{p}^{[i]}(t) = \mathbf{q}_{j_i}$ for all time $t \geq T$, where $j_i = j_k$ if and only if $i = k$.

In the remainder of the paper, we will refer to this as the *target assignment problem*.

Remark 3.1 (Consistent knowledge): A more general assumption on the initial target sets, $\mathcal{Q}^{[i]}(0)$, which still ensures the existence of a complete assignment, is the *consistent knowledge* assumption: For each $\mathcal{K} \subseteq \mathcal{I}$, $\left|\cup_{k\in\mathcal{K}} \mathcal{Q}^{[k]}(0)\right| \geq |\mathcal{K}|$. In fact, it was proved by Frobenius, 1917, and Hall, 1935 that this is the necessary and sufficient condition for the existence of a complete assignment [1]. •

In the full knowledge assumption, each agent knows the position of all targets in $\mathcal{Q}$. These positions will be stored in an array within each agents memory, rather than as an unordered set. To represent this, we replace the target set $\mathcal{Q}$ with the target n-tuple $\mathbf{q} := (\mathbf{q}_1, \dots, \mathbf{q}_n)$, and the local target set $\mathcal{Q}^{[i]}$ with the n-tuple $\mathbf{q}^{[i]}$. Thus, in the full knowledge assumption, $\mathbf{q}^{[i]}(0) := \mathbf{q}$ for each $i \in \mathcal{I}$. (It is possible that the order of the targets in the local sets $\mathbf{q}^{[i]}$ may initially be different. However, given a set of distinct points in $\mathbb{R}^d$, it is always possible to create a unique ordering.)

IV. Assignment-based algorithms with lower bound analysis

In this section we introduce and analyze a class of deterministic algorithms for the target assignment problem.

A. The assignment-based motion class

The initialization, motion, and communication for each algorithm in the *assignment-based motion* class have the following attributes:

Initialization: In this class of algorithms agent i initially selects the closest target in $\mathbf{q}^{[i]}$, and sets the variable curr$^{[i]}$ (agent i's current target), to the index of that target.

Motion: Agent i moves toward the target curr$^{[i]}$ at speed v:

$$\dot{\mathbf{p}}^{[i]} = \begin{cases} v\frac{\mathbf{q}^{[i]}_{\mathrm{curr}^{[i]}} - \mathbf{p}^{[i]}}{\|\mathbf{q}^{[i]}_{\mathrm{curr}^{[i]}} - \mathbf{p}^{[i]}\|}, & \text{if } \mathbf{q}^{[i]}_{\mathrm{curr}^{[i]}} \neq \mathbf{p}^{[i]}, \\ 0, & \text{otherwise}, \end{cases} \tag{2}$$

where $v > 0$ is a constant.

Communication: As agent i communicates with other agents, it updates the tuple $\mathbf{q}^{[i]}$ "removing" targets which are assigned to other agents. If agent i must change curr$^{[i]}$, it selects a new target in $\mathbf{q}^{[i]}$, that has not been removed. This is described more formally in the following.

Communication round for agent i.
1: Broadcast a message, msg$^{[i]}$, based on $\mathbf{q}^{[i]}$ and containing curr$^{[i]}$ and the UID i.
2: Receive msg$^{[k]}$ from each agent k within communication range.
3: **for all** msg$^{[k]}$ received **do**
4: Based on msg$^{[k]}$, (possibly) remove assigned targets from $\mathbf{q}^{[i]}$.
5: **if** curr$^{[i]}$ = curr$^{[k]}$ **then**
6: If agent i is farther from curr$^{[i]}$ than agent k, or if they are the same distance but $i < k$, remove the target given by curr$^{[i]}$ from $\mathbf{q}^{[i]}$.
7: Set curr$^{[i]}$ to a target in $\mathbf{q}^{[i]}$ (i.e., a target that has not been removed).

B. Lower bound on task complexity

In order to classify the time complexity of the assignment-based motion class of algorithms, we introduce a few useful definitions. We say that agent $i \in \mathcal{I}$ is *assigned* to target $\mathbf{q}^{[i]}_j$, $j \in \mathcal{I}$, when curr$^{[i]} = j$. In this case, we also say target j is assigned to agent i. We say that agent $i \in \mathcal{I}$ *enters a conflict* over the target curr$^{[i]}$, when agent i receives a message, msg$^{[k]}$, with curr$^{[i]}$ = curr$^{[k]}$. Agent i *loses the conflict* if agent i is farther from curr$^{[i]}$ than agent k, and *wins the conflict* if agent i is closer to curr$^{[i]}$ than agent k, where ties are broken by comparing UIDs.

Now we show that if agent i is assigned to the same target as another agent, it will enter a conflict in finite time.

Lemma 4.1 (Conflict in finite time): Consider any communication range $r > 0$, and any fixed number of agents $n \in \mathbb{N}$. If, for two agents i and k, curr$^{[i]}$ = curr$^{[k]}$ at some time $t_1 \geq 0$, then agent i (and likewise, agent k) will enter a conflict over curr$^{[i]}$ in finite time.

Proof: For each n the region $\mathcal{E}$ is compact, and the motion for each agent is given by (2). Hence, agent i will reach curr$^{[i]}$ in no more than $\mathrm{diam}(\mathcal{E})/v$ time units, as will agent k. The condition $\|\mathbf{p}^{[i]} - \mathbf{p}^{[k]}\| \leq r$ will be satisfied within $\mathrm{diam}(\mathcal{E}(n))/v$ time units. At the next communication round, agent i will enter a conflict over curr$^{[i]}$. ■

With these definitions we give a lower bound on the time complexity of the task assignment problem when the environment grows with the number of agents.

Theorem 4.2 (Time complexity of target assignment): Consider n agents, with communication range $r > 0$, in an environment $\mathcal{E} = [0, \ell(n)]^d$, $d \geq 1$. If $\ell(n) \geq (1+\epsilon) r n^{1/d}$, where $\epsilon \in \mathbb{R}_{>0}$, then for all algorithms in the assignment-based motion class, the time complexity of the target assignment problem is in $\Omega(n^{(d-1)/d}\ell(n))$.

Proof: We will construct a set of target positions and a set of initial agent positions for which the bound holds. The environment $\mathcal{E}$ is the d-cube, $[0, \ell(n)]^d$. Divide the cube $\mathcal{E}$ into $(\lceil n^{1/d} \rceil)^d$ cubes, each with side length $\ell(n)/\lceil n^{1/d} \rceil$, and place a target at the center of each of the cubes until you run out. This is shown in Fig. 1. Notice that the distance between any two targets is lower bounded by $\ell(n)/\lceil n^{1/d} \rceil$, and that, for sufficiently large n, $\ell(n)/\lceil n^{1/d} \rceil \geq (1+\epsilon) r n^{1/d}/\lceil n^{1/d} \rceil > r$.

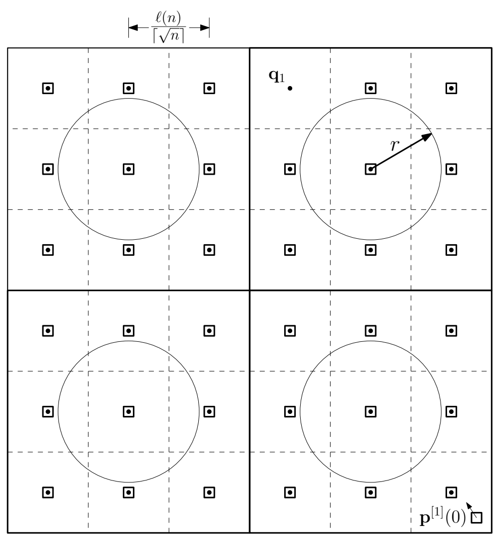


Fig. 1. Targets and agents placed on a lattice for the proof of Theorem 4.2. The lattice is split into 4 blocks, each containing $3^2 = 9$ agents. The center agent of each block is shown along with its communication radius r. The distance between these center agents is lower bounded by $\ell(n)/\lceil n^{1/2}\rceil$.

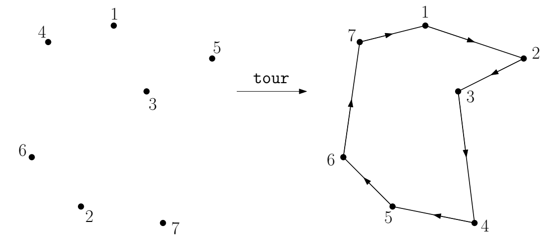


Fig. 2. The map tour, creating an ETSP tour of seven targets.

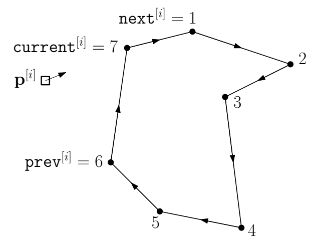


Fig. 3. The initialization for agent i.

Next, place agent 2 at $\mathbf{q}_2$, agent 3 at $\mathbf{q}_3$ and so on so that $\mathbf{p}^{[i]} = \mathbf{q}_i$, for all $i \in \{2, \dots, n\}$. From the initialization, we have that curr$^{[i]} = i$ for each $i \in \{2, \dots, n\}$. Now, if we place agent 1 in $\mathcal{E} \setminus \{\mathbf{q}_1, \dots, \mathbf{q}_n\}$, it will lose a conflict over any of the $n-1$ occupied targets $\mathbf{q}_2, \dots, \mathbf{q}_n$. Thus, the assignment will not be complete until agent 1 reaches target $\mathbf{q}_1$. Since the distance between targets is greater than r, communication between agents i and k is not possible for any $i, k \in \{2, \dots, n\}$. So, agent $i \in \{2, \dots, n\}$ will communicate only with agent 1. Thus, prior to communication with agent 1, each agent $i \in \{2, \dots, n\}$ will have $\mathbf{q}^{[i]} = \mathbf{q}$. The first time agent 1 comes within distance r of a target $j \in \{2, \dots, n\}$, in the best-case, agent 1 will remove target j from $\mathbf{q}^{[i]}$. Now, for any deterministic method of selecting curr$^{[i]}$, we can place agent 1 in $\mathcal{E} \setminus \{\mathbf{q}_1, \dots, \mathbf{q}_n\}$ such that target $\mathbf{q}_1$ is the last target for which agent 1 will come within distance r. Therefore, agent 1 must come within distance r of each of the $n-1$ assigned targets, before finally arriving at q_1.

Now we will lower bound the distance traveled by agent 1. To do this, split the large d-cube into $\lfloor n/3^d \rfloor$ smaller d-cubes, or blocks, where each block contains 3^d targets. An example is shown in Fig. 1. There is one target at the center of each of these blocks, and agent 1 must come within distance r of it. The distance between the center target of each block is lower bounded by the distance between targets, $\ell(n)/\lceil n^{1/d}\rceil$. Agent 1 must travel this distance at least $\lfloor n/3^d \rfloor - 1$ times. So we have

$$\text{Path length} \geq \left(\left\lfloor \frac{n}{3^d} \right\rfloor - 1 \right) \frac{\ell(n)}{\lceil n^{1/d} \rceil} \in \Omega(n^{(d-1)/d}\ell(n)).$$

Hence, the path length is in $\Omega(n^{(d-1)/d}\ell(n))$. Since $v \in \mathbb{R}_{>0}$, the time complexity is also in $\Omega(n^{(d-1)/d}\ell(n))$. ∎

Remark 4.3 ($\ell(n) \leq \ell_{crit}$): We have lower bounded the time complexity when $\ell(n)$ grows faster than some critical value, $\ell_{crit} = rn^{1/d}$. This same type of bound appears in percolation theory and the study of random geometric graphs, where it is referred to the thermodynamic limit [11]. When $\ell(n)$ grows more slowly than this critical value, the performance depends on the particular algorithm in the assignment-based motion class. In addition, when $\ell(n) \leq \ell_{crit}$, congestion issues in both motion and communication become more prevalent, and a more complex communication and motion model would ideally be used. •

In the next section we introduce an asymptotically optimal algorithm in the assignment-based motion class.

V. The ETSP Assignment Algorithm

In this section we introduce the ETSP Assignment algorithm—an algorithm within the assignment-based motion class. We will show that when $\ell(n)$ grows more quickly than a critical value, this algorithm is asymptotically optimal. The algorithm can be described as follows.

For each $i \in \mathcal{I}$, agent i computes a constant factor approximation of the optimal ETSP tour of the n targets in $\mathbf{q}^{[i]}$, denoted $\mathrm{tour}(\mathbf{q}^{[i]})$. We can think of tour as a map which reorders the indices of $\mathbf{q}^{[i]}$; $\mathrm{tour}(\mathbf{q}^{[i]}) = (\mathbf{q}^{[i]}_{\sigma(1)}, \dots, \mathbf{q}^{[i]}_{\sigma(n)})$, where $\sigma : \mathcal{I} \to \mathcal{I}$ is a bijection. Notice that this map is independent of i since all agents use the same method. An example is shown in Fig. 2. Agent i then replaces its n-tuple $\mathbf{q}^{[i]}$ with $\mathrm{tour}(\mathbf{q}^{[i]})$. Next, agent i computes the index of the closest target in $\mathbf{q}^{[i]}$, and calls it curr$^{[i]}$. Agent i also maintains the index of the next target in the tour which may be available, next$^{[i]}$, and first target in the tour before curr$^{[i]}$ which may be available, prev$^{[i]}$. Thus, next$^{[i]}$ is initialized to curr$^{[i]} + 1 \pmod{n}$ and prev$^{[i]}$ to curr$^{[i]} - 1 \pmod{n}$. This is depicted in Fig. 3. In order to "remove" assigned targets from the tuple $\mathbf{q}^{[i]}$, agent i also maintains the n-tuple, status$^{[i]}$. Letting status$^{[i]}(j)$ denote the jth entry in the

n-tuple, the entries are given by

$$\text{status}^{[i]}(j) = \begin{cases} 0, & \text{if agent } i \text{ knows } \mathbf{q}_j^{[i]} \text{ is assigned} \\ & \text{to another agent,} \\ 1, & \text{otherwise.} \end{cases} \tag{3}$$

Thus, status$^{[i]}$ is initialized as the n-tuple $(1, \dots, 1)$. The initialization is summarized in Table I. At each communication

TABLE I

THE INITIALIZATION PROCEDURE FOR AGENT i.

Initialization for agent i.

Assumes: $\mathbf{q}^{[i]} := \mathbf{q}$ for each $i \in \mathcal{I}$.

Compute a TSP tour of $\mathbf{q}^{[i]}$, $\text{tour}(\mathbf{q}^{[i]})$, and set $\mathbf{q}^{[i]} := \text{tour}(\mathbf{q}^{[i]})$.
Compute the closest target in $\mathbf{q}^{[i]}$, and set curr$^{[i]}$ equal to its index: curr$^{[i]} := \arg\min_{j \in \mathcal{I}} \{\|\mathbf{q}_j^{[i]} - \mathbf{p}^{[i]}\|\}$.
Set next$^{[i]} :=$ curr$^{[i]} + 1 \pmod{n}$.
Set prev$^{[i]} :=$ curr$^{[i]} - 1 \pmod{n}$.
Set status$^{[i]} := \mathbf{1}_n$ (i.e., an n-tuple containing n ones).

round agent i executes the algorithm COMM-RD displayed in Table II at the end of the paper. The following is an informal description.

Informal Description of COMM-RD for agent i

Assumes: status$^{[i]}(s) = 0$ for each $s \in \{\text{prev}^{[i]} + 1, \text{prev}^{[i]} + 2, \dots, \text{next}^{[i]} - 1\} \setminus \{\text{curr}^{[i]}\} \pmod{n}$.

Broadcast msg$^{[i]}$, consisting of the target indices, prev$^{[i]}$, curr$^{[i]}$, and next$^{[i]}$, the UID i, and the distance to the current target, dist$^{[i]}$.
for all messages, msg$^{[k]}$, received **do**
  Set status$^{[i]}(j)$ to assigned ('0') for each target j from prev$^{[k]} + 1 \pmod{n}$ to next$^{[k]} - 1 \pmod{n}$ not equal to curr$^{[i]}$.
  if prev$^{[k]}$ = next$^{[k]}$ = curr$^{[k]} \neq$ curr$^{[i]}$ **then**
    Set the status of curr$^{[k]}$ to 0 (because it was missed in the previous step).
  if curr$^{[i]}$ = curr$^{[k]}$ but agent i is farther from curr$^{[i]}$ than agent k (ties broken with UIDs) **then**
    Set the status of curr$^{[i]}$ to assigned ('0').
  if curr$^{[i]}$ = curr$^{[k]}$ and agent i is closer than agent k **then**
    Leave curr$^{[i]}$ unchanged. However, agent k will set curr$^{[k]}$ to a new target. This target will be at least as far along the tour as the farther of next$^{[i]}$ and next$^{[k]}$. So, set the status of next$^{[i]}$ and next$^{[k]}$ to assigned ('0').
if the status of every target is assigned ('0') **then**
  Exit ETSP ASSIGNMENT and stop motion. (This can occur only if there are more agents than targets and every target is assigned.)
else
  Update curr$^{[i]}$ to the next target in the tour with status available ('1'), next$^{[i]}$ to the next available target in the tour after curr$^{[i]}$, and prev$^{[i]}$ to the first available target in the tour before curr$^{[i]}$.

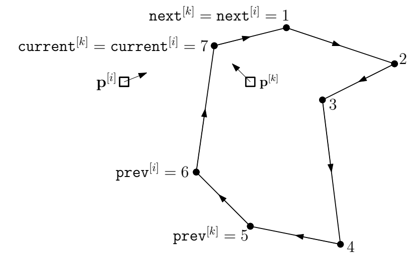
Fig. 4 gives an example of COMM-RD resolving a conflict between agents i and k, over curr$^{[i]}$ = curr$^{[k]}$. In this figure, all other agents are omitted.

We are now ready to define the algorithm ETSP ASSIGNMENT for solving the target assignment problem.

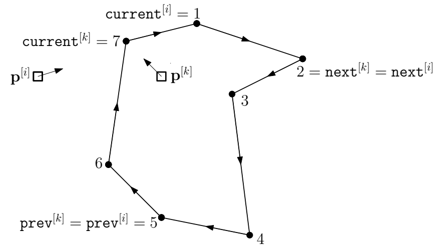
*Definition 5.1 (*ETSP ASSIGNMENT*):* The ETSP ASSIGNMENT algorithm is the triplet consisting of the initialization of each agent (see Table I), the motion law in (2), and COMM-RD (see Table II), which is executed at each communication round.

A. *Correctness of* ETSP ASSIGNMENT

We will now prove the correctness of ETSP ASSIGNMENT. It should be noted that this result is valid for any



(a) Setup before the conflict over target 7.



(b) Setup after resolution of the conflict.

Fig. 4. The resolution of a conflict between agents i and k over target 7. Since agent k is closer to target 7 than agent i, agent k wins the conflict.

communication graph which contains the r-disk graph as a subgraph. In order to prove correctness, let us first present some properties of the algorithm.

*Lemma 5.2 (*ETSP ASSIGNMENT *properties):* During an execution of ETSP ASSIGNMENT the following statements hold:

(i) Once target $j \in \mathcal{I}$, is assigned to some agent, the assignment may change, but target j remains assigned for all time.
(ii) Agent i is assigned to the target curr$^{[i]}$ which satisfies status$^{[i]}$(curr$^{[i]}$) = 1.
(iii) For agent i, status$^{[i]}(j) = 0$, for each $j \in \{\text{prev}^{[i]} + 1, \text{prev}^{[i]} + 2, \dots, \text{next}^{[i]} - 1\} \setminus \{\text{curr}^{[i]}\} \pmod{n}$.
(iv) For agent i, status$^{[i]}(j) = 0$ only if target j is assigned to some agent $k \neq i$.
(v) If, for agent i, status$^{[i]}(j) = 0$ at some time t_1, then status$^{[i]}(j) = 0$ for all $t \geq t_1$.
(vi) If agent i receives msg$^{[k]}$ during a communication round, agent i will set status$^{[i]}(j) = 0$ for each $j \in \{\text{prev}^{[k]} + 1, \dots, \text{next}^{[k]} - 1\} \setminus \{\text{curr}^{[i]}\} \pmod{n}$.

Proof: Statements (ii) and (v) and (vi) follow directly from the initialization and the algorithm COMM-RD.

To see (i), consider an agent i, who is assigned to target j. Agent i's assignment can change only if it loses a conflict over target j. In every conflict there is a winner and the winner remains assigned to target j.

Statement (iii) is initially satisfied since prev$^{[i]} + 1$ = curr$^{[i]}$ = next$^{[i]} - 1$ implies that $\{\text{prev}^{[i]} + 1, \dots, \text{next}^{[i]} - 1\} \setminus \{\text{curr}^{[i]}\} = \emptyset$. Assume that statement (iii) is satisfied before the execution of COMM-RD. At the end of COMM-RD, prev$^{[i]}$ is updated to the first target before curr$^{[i]}$ in the tour with status available ('1'). If status$^{[i]}$(curr$^{[i]}$) = 1 then curr$^{[i]}$ remains unchanged. If status$^{[i]}$(curr$^{[i]}$) = 0 then curr$^{[i]}$ is increased to the first target with status available ('1'). Finally, next$^{[i]}$ is set to the first target after curr$^{[i]}$ which is available. Thus, at the end of COMM-RD the status of prev$^{[i]}$, curr$^{[i]}$ and next$^{[i]}$ are available, and status$^{[i]}(j) = 0$ for each target $j \in \{\text{prev}^{[i]}+1, \dots, \text{next}^{[i]}-1\} \setminus \{\text{curr}^{[i]}\} \pmod{n}$.

Statement (iv) is also initially satisfied since status$^{[i]} = \mathbf{1}_n$ for each $i \in \mathcal{I}$. Assume Statement (iv) is satisfied before the execution of COMM-RD and that during this communication round agent i changes the status of a target j to assigned ('0'). We will show that Statement (iv) is still satisfied upon completion of the execution of COMM-RD. In order for status$^{[i]}(j)$ to be changed, agent i must have received a message, msg$^{[k]}$, for which one of the following cases is satisfied: (1) Target $j \neq$ curr$^{[i]}$ lies between prev$^{[k]}$ and next$^{[k]}$ on the tour; (2) There is a conflict between agents i and k over target j which agent i loses; or, (3) There is a conflict between agents i and k which agent i wins and next$^{[i]} = j$ or next$^{[k]} = j$.

In Case (1) either status$^{[k]}(j) = 0$ or curr$^{[k]} = j$, and thus target j is assigned. In Case (2) agent k won the conflict implying curr$^{[k]} = j$ entering the communication round. Thus after the communication round, curr$^{[i]} \neq j$ and target j is assigned to another agent. In Case (3), curr$^{[i]} =$ curr$^{[k]} \neq j$, and agent k loses the conflict. In this case, agent k will change curr$^{[k]}$ to the next available target on its tour. All targets from prev$^{[k]}+1$ to next$^{[k]}-1$ have been assigned. Also, during the communication round, agent k will receive msg$^{[i]}$ and determine that all targets from prev$^{[i]}+1$ to next$^{[i]}-1$ are assigned. Thus, the next available target is at least as far along the tour as the farther of next$^{[i]}$ and next$^{[k]}$. Thus, after the communication round, both next$^{[i]}$ and next$^{[k]}$ are assigned. ■

With these properties we are now ready to present the main result of this section.

Theorem 5.3 (Correctness of ETSP ASSIGNMENT*):* For any fixed $n \in \mathbb{N}$, ETSP ASSIGNMENT solves the target assignment problem.

Proof: Assume by way of contradiction that at some time $t_1 \geq 0$ there are $J \in \{1, \dots, n-1\}$ targets unassigned, and for all time $t \geq t_1$, J targets remain unassigned. By Lemma 5.2 (i) the $n-J$ assigned targets remain assigned for all time, and thus it must be the same J targets which remain unassigned for all $t \geq t_1$. Let $\mathcal{J}$ denote the index set of the J unassigned targets. From our assumption, and by Lemma 5.2 (iv), for every $t \geq t_1$ and for every $i \in \mathcal{I}$, status$^{[i]}(j) = 1$ for each $j \in \mathcal{J}$. Now, among the $n-J$ assigned targets, there is at least one target to which two or more agents are assigned. Consider one such target, call it j_1, and consider an agent i_1 with curr$^{[i_1]} = j_1$. By Lemma 4.1, agent i_1 will enter a conflict over j_1 in finite time. Let us follow the loser of this conflict. The losing agent, call it i_2, will set status$^{[i_2]}(j_1) = 0$, and will move to the next target in the tour it believes may be available, call it j_2. Now, we know j_2 is not in $\mathcal{J}$, for if it were $J-1$ targets would be unassigned contradicting our assumption. Moreover, by Lemma 5.2 (ii), $j_2 \neq j_1$. Thus, agent i_2 will enter a conflict over j_2 in finite time. After this conflict, the losing agent, call it i_3, will set status$^{[i_3]}(j_2) = 0$ (because it lost the conflict), and from Lemma 5.2 (vi), status$^{[i_3]}(j_1) = 0$. Again, agent i_3's next target, j_3 must not be in $\mathcal{J}$, for if it were we would have a contradiction. Thus, repeating this argument $n-J$ times we have that agent i_{n-J} loses a conflict over j_{n-J}. After this conflict, we have status$^{[i_{n-J}]}(j_k) = 0$ for each $k \in \{1, \dots, n-J\}$, where $j_{k_1} = j_{k_2}$ if and only if $k_1 = k_2$. In other words, agent i_{n-J} knows that all $n-J$ assigned targets have indeed been assigned. Also, by our initial assumption, status$^{[i_{n-J}]}(j) = 1$ for each $j \in \mathcal{J}$. By Lemma 5.2 (ii), agent i_{n-J}'s new current target must have status available ('1'). Therefore, it must be that agent i_{n-J} will set curr$^{[i_{n-J}]}$ to a target in $\mathcal{J}$. Thus, after a finite amount of time, $J-1$ targets are unassigned, a contradiction. ■

The following remark displays that the ETSP ASSIGNMENT algorithm does not solve the target assignment under the consistent knowledge assumption.

Remark 5.4 (Consistent knowledge: cont'd): Consider as in Remark 3.1 the consistent knowledge assumption for each agent's target set. Specifically, consider two agents, 1 and 2, with initial target sets $\mathcal{Q}^{[1]}(0) = \{\mathbf{q}_2\}$, $\mathcal{Q}^{[2]}(0) = \{\mathbf{q}_1, \mathbf{q}_2\}$, and any initial positions such that $\mathbf{p}^{[1]}(0) = \mathbf{q}_2$, We will have curr$^{[i]} =$ curr$^{[j]} = 2$. However, agent 2 will win the conflict over target 2. Thus, agent 1 will set status$^{[1]}(2) = 0$, and a complete assignment will not be possible. •

B. Time complexity for ETSP ASSIGNMENT

In this section we will give an upper bound on the time complexity for ETSP ASSIGNMENT. We will show that when $\ell(n) \geq (1+\epsilon) r n^{1/d}$, for some $\epsilon \in \mathbb{R}_{>0}$, ETSP ASSIGNMENT is asymptotically optimal among algorithms in the assignment-based motion class. Before doing this, let us first comment on the lower bound when the environment grows at a slower rate.

Remark 5.5 ($\ell(n) \leq \ell_{crit}$ cont'd): When $\ell(n) \leq r n^{1/d}$, and we use the setup shown in Fig. 1, the distance between targets is lower bounded by $\ell(n)/\lceil n^{1/d} \rceil \leq r n^{1/d}/\lceil n^{1/d} \rceil \leq r$. Thus, agents $2, \dots, n$ can communicate to their neighbors on the target grid, and the communication graph is connected. If the agents execute ETSP ASSIGNMENT, from Lemma 5.2 (vi), after one communication round, agent $i \in \{2, \dots, n\}$ will set status$^{[i]}(j) = 0$ for each target $j \neq 1$ within distance r. Thus, after C communication rounds, where C is the diameter of the communication graph, every agent $i \in \{2, \dots, n\}$ will have status$^{[i]}(j) = 0$, for all targets j other than 1 and curr$^{[i]}$. Once C communication rounds are complete, the next time agent 1 enters a conflict, it will set curr$^{[1]} = 1$ and the assignment will be complete. This gives an upper bound of $O(C t_{max} + \ell(n))$ for the specific setup in Fig. 1, but it does not provide a useful lower bound on ETSP ASSIGNMENT. •

In what follows we show that if an agent arrives and remains at its assigned target for sufficiently long time, then it stays there for all subsequent times.

Lemma 5.6: Consider n agents executing ETSP ASSIGNMENT with communication range $r > 0$ and assume the time delay between communication rounds, t_{max}, satisfies $t_{max} < r/v$. If there exists a time t_1 and an agent i such that $\mathbf{p}^{[i]}(t) =$ curr$^{[i]}$ for all $t \in [t_1, t_1 + t_{max}]$, then $\mathbf{p}^{[i]}(t) =$ curr$^{[i]}$ for all $t > t_1 + t_{max}$.

Proof: Consider agent i, who has been at target curr$^{[i]}$ during the entire time interval $[t_1, t_1 + t_{max}]$. By the definition of t_{max} there was a communication round at some time $t_2 \in [t_1, t_1 + t_{max}[$. Agent i must have won any conflicts it entered during this communication round since we have assumed that $\mathbf{p}^{[i]}(t_1 + t_{max}) = \text{curr}^{[i]}$. Thus every agent k within distance r of curr$^{[i]}$ will have set status$^{[k]}(\text{curr}^{[i]}) = 0$. After the communication round at t_2, every agent k with curr$^{[k]}$ = curr$^{[i]}$ must be a distance greater than r from curr$^{[i]}$. Since $t_{max} < r/v$, any agent k that enters a conflict with agent i at time $t > t_2$, will be at a distance dist$^{[k]} \in]0, r[$ from curr$^{[i]}$. Agent k will lose the conflict since dist$^{[k]} > 0 =$ dist$^{[i]}$. Thus, agent i will remain at curr$^{[i]}$ for all $t > t_1 + t_{max}$. ∎

With this lemma we are now able to provide an upper bound on the time complexity of our scheme.

Theorem 5.7 (Time complexity for ETSP ASSIGNMENT*):* Consider an environment $\mathcal{E} = [0, \ell(n)]^d$, $d \geq 1$. If $t_{max} < r/v$, then ETSP ASSIGNMENT solves the target assignment problem with time complexity in $O(n^{(d-1)/d}\ell(n) + n)$. If, in addition, $\ell(n) \geq (1+\epsilon)rn^{1/d}$, where $\epsilon \in \mathbb{R}_{>0}$, the time complexity is in $\Theta(n^{(d-1)/d}\ell(n))$, and ETSP ASSIGNMENT is asymptotically optimal among algorithms in the assignment-based motion class.

Proof: Consider any initial agent positions, $p^{[1]}(0), \dots, p^{[n]}(0)$, and any n-tuple of target positions, $\mathbf{q}$. In the worst-case, some agent must travel around its entire ETSP tour, losing a conflict at each of the first $n-1$ targets in the tour. By Lemma 5.6, this agent can spend no more than t_{max} time units at each of the $n-1$ targets, before losing a conflict. Since each agent's tour is a constant factor approximation of the optimal, the tour length is $O(n^{(d-1)/d}\ell(n))$ (see Theorem 2.1). The agent will not follow the ETSP tour exactly because it will enter a conflict over each of the $n-1$ targets before actually reaching the target. However, the resulting path is no longer than the ETSP tour (since the agent could just follow the ETSP tour exactly if that happened to be the shortest path). Hence, the time complexity is $O(n^{(d-1)/d}\ell(n) + t_{max}(n-1)) \in O(n^{(d-1)/d}\ell(n) + n)$. If $\ell(n) = (2+\epsilon)rn^{1/d}$, with $\epsilon \in \mathbb{R}_{>0}$, we can combine this with Theorem 4.2 to get a time complexity in $\Theta(n^{(d-1)/d}\ell(n))$. ∎

Notice that when $\ell(n)$ satisfies the bound in Theorem 5.7, and $\ell(n) \in O(n^{1/d})$, the time complexity is in $O(n)$.

We have given complexity bounds for the case when r and v are fixed constants, and $\ell(n)$ grows with n. We allow the environment $\mathcal{E}(n)$ to grow with n so that, as more agents are involved in the task, their workspace is larger. An equivalent setup would be to consider ℓ to be fixed, and allow r and v to vary inversely with the n. That is, we can introduce a set of parameters, $\tilde{\ell} = 1$, and $\tilde{r}(n)$ and $\tilde{v}(n)$ such that the time complexity will be the same as for the parameters r, v, $\ell(n)$.

Corollary 5.8 (Scaling radius and speed): Consider n agents in the environment $\mathcal{E} = [0,1]^d$, with speed $\tilde{v}(n) := v/\ell(n)$, and communication radius $\tilde{r}(n) := r/\ell(n)$, where $\ell(n) \geq (1+\epsilon)rn^{1/d}$, and $\epsilon \in \mathbb{R}_{>0}$. Then ETSP ASSIGNMENT solves the target assignment problem with time complexity in $\Theta(n^{(d-1)/d}\ell(n))$.

Scaling the communication radius r inversely with the number of agents arises in the study of wireless networks [12]. In wireless applications there are interference and media access problems between agents in the network. Since the agents are in a compact environment, the only way to limit this interference is to scale the communication radius inversely with the number of agents. Scaling the agent speed inversely with n appears in the study of the vehicle routing problem in [7]. The inverse scaling is required to avoid collisions in the presence of traffic congestion.

C. Communication and computation complexity

In our notion of time complexity we have emphasized the complexity due to the motion of the agents. Here we will briefly classify the complexity of computation and communication for ETSP ASSIGNMENT. (i) *Initialization:* As reviewed in Section II-C, we can compute a constant factor approximation ETSP tour in time $O(n^2)$. This is the most expensive computation and thus the complexity of initialization is in $O(n^2)$. (ii) *Communication complexity per round:* At each round agent i broadcast a message of length $O(\log n)$, msg$^{[i]}$, and we consider this to be one unit of communication. In the worst-case, each agent receives n messages, and so, the worst-case communication complexity is in $O(n)$ [8]. (iii) *Computation complexity per round:* For each message received, agent i sets status$^{[i]}(s) = 0$ for s from prev$^{[k]}+1$ to next$^{[k]}-1$. In the worst-case, this operation is $O(n)$ and must be performed for n messages. This is the dominant computation in COMM-RD and thus the worst-case computation complexity in each round is $O(n^2)$.

It should be noted that in the case when the communication graph is not even connected (let alone complete as is required to achieve these worst-case bounds), the complexity will be considerably lower.

D. Simulations

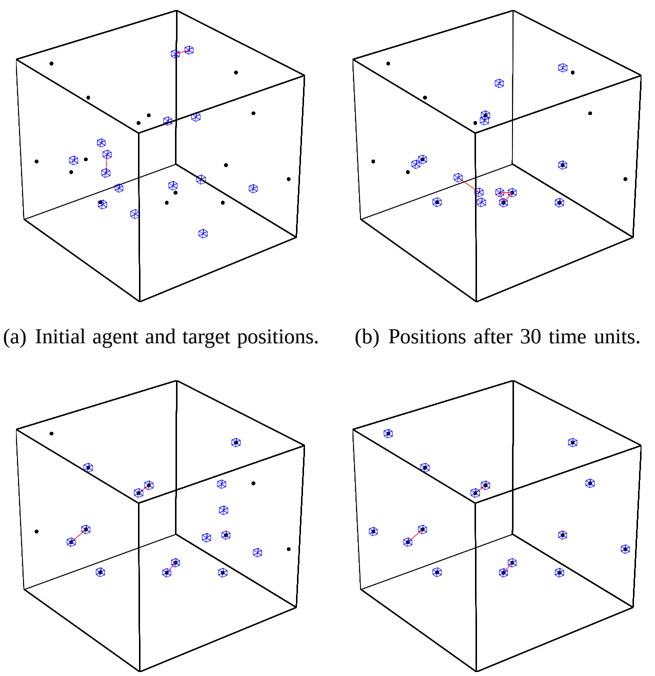
We have simulated ETSP ASSIGNMENT in $\mathbb{R}^2$ and $\mathbb{R}^3$. To compute the ETSP tour we have used the `concorde` TSP solver.[2] A representative simulation for 15 agents in $[0,100]^3 \subset \mathbb{R}^3$ with $r = 15$ and $v = 1$ is shown in Fig. 5. The initial configuration shown in Fig. 5(a) consists of uniformly randomly generated target and agent positions.

E. The case of n agents and m targets

It should be noted that the ETSP ASSIGNMENT algorithm works without any modification when there are n agents and m targets. If $m \geq n$, at completion, n targets are assigned and $m-n$ targets are not. When, $m < n$, at completion, all m targets are assigned, and $n-m$ agents are stationary, after losing a conflict at each of the m targets. The complexity bounds are changed as follows.

The lower bound on the assignment-based motion class in Theorem 4.2, holds when $m \geq n$, and $\ell(n) \geq (1+\epsilon)rm^{1/d}$ (notice the m instead of n). The bound become $\Omega(\ell(n)m^{-1/d}n)$. If $m = Cn$ where $C \in \mathbb{R}_{\geq 1}$, (i.e., $m \geq n$

[2]The `concorde` TSP solver is available for research use at `http://www.tsp.gatech.edu/concorde/index.html`



(a) Initial agent and target positions. (b) Positions after 30 time units.

(c) Positions after 90 time units. (d) Complete target assignment.

Fig. 5. Simulation for 15 agents, with $v = 1$ and $r = 15$. in an environment $[0, 100]^3$. The targets are spheres and the agents are cubes. An edge is drawn between two agents when they are within communication range.

but they grow at the same rate), then the bound becomes $\Omega(\ell(n)n^{(d-1)/d})$.

The upper bound on ETSP ASSIGNMENT holds for any n and m, and becomes $O(\ell(n)N^{(d-1)/d})$, where $N := \min\{n, m\}$. So our final result would be that if $m = Cn$ where $C \in \mathbb{R}_{\geq 1}$ and when $\ell(n) \geq (1+\epsilon)rm^{1/d}$, then ETSP ASSIGNMENT solves the target assignment problem in $\Theta(\ell(n)n^{(d-1)/d})$. That is, among all algorithms in the assignment-based motion class, ETSP ASSIGNMENT is asymptotically optimal.

TABLE II

COMMUNICATION ROUND (COMM-RD) FOR AGENT i.

Name: COMM-RD
Goal: Obtain information on assigned targets.
Assumes: (i) Knowledge of the n-tuple $\mathbf{q}$, and a method for computing a constant factor TSP tour of the n targets, tour. (ii) A communication range $r > 0$.

Compute $\text{dist}^{[i]} := \|\mathbf{p}^{[i]} - \mathbf{q}^{[i]}_{\text{curr}^{[i]}}\|$.
Broadcast $\text{msg}^{[i]} := (\text{prev}^{[i]}, \text{curr}^{[i]}, \text{next}^{[i]}, i, \text{dist}^{[i]})$.
Receive $\text{msg}^{[k]}$, from each $k \neq i$ satisfying $\|\mathbf{p}^{[i]} - \mathbf{p}^{[k]}\| \leq r$.
for all $\text{msg}^{[k]}$ received **do**
  for $s = \text{prev}^{[k]} + 1$ to $\text{next}^{[k]} - 1 \pmod n$ **do**
    if $s \neq \text{curr}^{[i]}$ **then**
      Set $\text{status}^{[i]}(s) := 0$
  if $\text{prev}^{[k]} = \text{next}^{[k]} = \text{curr}^{[k]} \neq \text{curr}^{[i]}$ **then**
    Set $\text{status}^{[i]}(\text{curr}^{[k]}) := 0$
  if $\text{curr}^{[i]} = \text{curr}^{[k]}$ **then**
    if $(\text{dist}^{[i]} > \text{dist}^{[k]})$ OR $(\text{dist}^{[i]} = \text{dist}^{[k]}$ AND $i < k)$ **then**
      Set $\text{status}^{[i]}(\text{curr}^{[i]}) := 0$.
    else
      if $\text{next}^{[i]} \neq \text{curr}^{[i]}$ **then**
        Set $\text{status}^{[i]}(\text{next}^{[i]}) := 0$.
      if $\text{next}^{[k]} \neq \text{curr}^{[i]}$ **then**
        Set $\text{status}^{[i]}(\text{next}^{[k]}) := 0$.
if $\text{status}^{[i]}(j) = 0$ for every target j **then**
  Exit ETSP ASSIGNMENT and stop motion.
else
  while $\text{status}^{[i]}(\text{curr}^{[i]})$=0 **do**
    $\text{curr}^{[i]} := \text{curr}^{[i]} + 1 \pmod n$.
  Set $\text{next}^{[i]} := \text{curr}^{[i]} + 1 \pmod n$.
  while $\text{status}^{[i]}(\text{next}^{[i]})$=0 **do**
    $\text{next}^{[i]} := \text{next}^{[i]} + 1 \pmod n$.
  while $\text{status}^{[i]}(\text{prev}^{[i]})$=0 **do**
    $\text{prev}^{[i]} := \text{prev}^{[i]} - 1 \pmod n$.

VI. CONCLUSIONS

We have developed the ETSP ASSIGNMENT algorithm for solving the full knowledge target assignment problem. We derived worst-case asymptotic bounds on the time complexity, and we showed that among a certain class of algorithms, ETSP ASSIGNMENT is asymptotically optimal. There are many possible extensions of this work. We have not computed bounds on the time-complexity in the average case. Also, the problem is unsolved under the consistent knowledge assumption. It would be nice to extend the ETSP ASSIGNMENT algorithm to agents with nonholonomic motion constraints. Also, it would be interesting to consider a sensor based version of this problem, where agents acquire target positions through local sensing. Finally, to derive asymptotic time bounds, we made some assumptions on the communication structure at each communication round. An interesting avenue for future study would be to more accurately address the communication issues in robotic networks.

REFERENCES

[1] B. Korte and J. Vygen, *Combinatorial Optimization: Theory and Algorithms*. No. 21 in Algorithmics and Combinatorics, New York: Springer Verlag, 3 ed., 2005.

[2] H. W. Kuhn, "The Hungarian method for the assignment problem," *Naval Research Logistics*, vol. 2, pp. 83–97, 1955.

[3] D. P. Bertsekas and D. A. Castañón, "Parallel synchronous and asynchronous implementations of the auction algorithm," *Parallel Computing*, vol. 17, pp. 707–732, 1991.

[4] D. P. Bertsekas and J. N. Tsitsiklis, *Parallel and Distributed Computation: Numerical Methods*. Belmont, MA: Athena Scientific, 1997.

[5] D. A. Castañón and C. Wu, "Distributed algorithms for dynamic reassignment," in *IEEE Conf. on Decision and Control*, (Maui, HI), pp. 13–18, Dec. 2003.

[6] B. J. Moore and K. M. Passino, "Distributed task assignment for mobile agents," *IEEE Transactions on Automatic Control*, 2006. to appear.

[7] V. Sharma, M. Savchenko, E. Frazzoli, and P. Voulgaris, "Time complexity of sensor-based vehicle routing," in *Robotics: Science and Systems* (S. Thrun, G. Sukhatme, S. Schaal, and O. Brock, eds.), pp. 297–304, Cambridge, MA: MIT Press, 2005.

[8] S. Martínez, F. Bullo, J. Cortés, and E. Frazzoli, "On synchronous robotic networks – Part I: Models, tasks and complexity notions. & Part II: Time complexity of rendezvous and deployment algorithms," *IEEE Transactions on Automatic Control*, Apr. 2005. Submitted.

[9] J. M. Steele, "Probabilistic and worst case analyses of classical problems of combinatorial optimization in Euclidean space," *Mathematics of Operations Research*, vol. 15, no. 4, p. 749, 1990.

[10] N. Christofides, "Worst-case analysis of a new heuristic for the traveling salesman problem," Tech. Rep. 388, Carnegie-Mellon University, Pittsburgh, PA,, Apr. 1976.

[11] M. Penrose, *Random Geometric Graphs*. Oxford Studies in Probability, Oxford, UK: Oxford University Press, 2003.

[12] P. Gupta and P. R. Kumar, "The capacity of wireless networks," *IEEE Transactions on Information Theory*, vol. 46, no. 2, pp. 388–404, 2000.