A Graph-Based Approach to Multi-Robot Rendezvous for Recharging in Persistent Tasks

Neil Mathew Stephen L. Smith Steven L. Waslander

***Abstract*— This paper addresses the problem of maintaining persistence in coordinated tasks performed by a team of autonomous robots. We introduce a dedicated team of charging robots to service a team of primary working robots. Given that the trajectories of the working robots are known within a planning interval, the objective is to plan routes for the charging robots such that they rendezvous with and recharge all working robots to guarantee their continuous operation. To this end, the working robot trajectories are discretized to form a finite set of recharging points at which rendezvous can occur. The problem is formulated as a directed acyclic graph with vertex partitions containing sets of charging points for each working robot. Solutions consist of paths through the graph for each of the charging robots. The problem is shown to be NP-hard and a mixed integer linear program formulation is presented and solved for small problem instances. Finally, it is shown that while the optimal solution is not computationally feasible for large problem sizes, it is possible to graphically transform the single charging robot problem to a Traveling Salesman Problem, for which existing heuristic and approximation algorithms can be applied. Simulation results are presented for both single and multiple charging robot scenarios.**

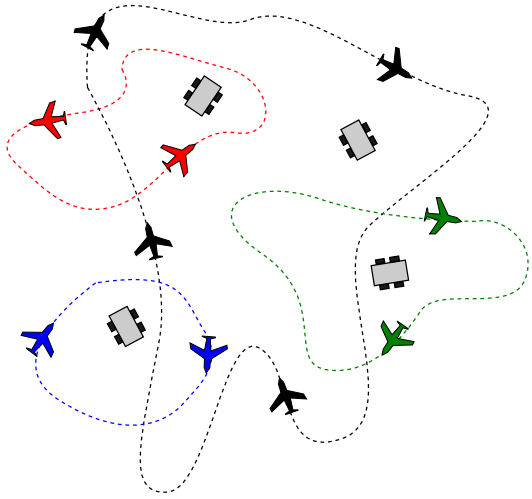


Fig. 1. A persistent monitoring scenario with multiple coordinated UAVs and charging robots. Assuming planar UAV paths, we consider a 2D projection of the environment at ground level.

I. Introduction

Teams of autonomous robots are often proposed as a means to continually monitor changing environments in applications such as oceanographic sampling, forest fire or oil spill monitoring and border security. These surveillance tasks generally require the robot to consistently traverse the environment in trajectories designed to optimize certain performance criteria such as quality or frequency of sensor measurements taken in the region [1], [2], [3]. The challenge with employing autonomous robots in persistent tasks is that mission durations generally exceed the run time of the robots, and in order to maintain functionality they need to be periodically recharged or refueled. As illustrated in Figure 1, this paper presents a recharging[1] strategy for a team of *working robots* performing a surveillance task, using one or more dedicated mobile *charging robots* to keep them operational over extended periods of time.

It is advantageous to use mobile charging robots because of the ease of deployment in unknown environments and a greater flexibility that comes with dynamic charging locations. The charging process could be performed, for example, by automated docking and battery swap systems, as demonstrated in [4] and [5]. The charging robots carry a payload of batteries than can be swapped into docked working robots to replenish their charge.

The problem of persistent coverage and surveillance with mobile robots has been previously investigated in a variety of contexts. Cortes et al. [1] employ Lloyd's algorithm to develop a centroidal Voronoi tessellation-based controller that optimally covers a convex area with a team of mobile robots. Smith et al. [2] design optimal velocity controllers along precomputed paths to persistently cover a set of discrete points with varying desired frequencies of observation, taking advantage of the ability of mobile robots to sense while in motion. While both these works present persistent surveillance scenarios, neither tackle the problem of limited energy resources in the robotic agents.

Derenick et al. [6] propose a modification to [1] which introduces a combined coverage and energy dependent control law to drive each robot toward a fixed docking station as their energy levels become critical. Their work considers only the static coverage case without any notion of charge scheduling as each agent is assigned a dedicated static charging station. In the worst case scenario all agents will move towards their charging locations simultaneously leaving the environment unmonitored.

Litus et al. [7], [8] find a set of meeting points for a team worker robots and a single charging robot, given static robot

This research is partially supported by the Natural Sciences and Engineering Research Council of Canada (NSERC).

N. Mathew and S. L. Waslander are with the Department of Mechanical and Mechatronics Engineering and S. L. Smith is with the Department of Electrical and Computer Engineering, all at the University of Waterloo, Waterloo ON, N2L 3G1 Canada (nmathew@uwaterloo.ca; stephen.smith@uwaterloo.ca; stevenw@uwaterloo.ca)

[1]We use terminology for electric vehicles (recharging) throughout, but fuel-based vehicles can be treated equivalently.

locations and a predetermined charging order. Obermeyer et al. [9] use a sampling-based roadmap method to design trajectories for a UAV conducting visual reconnaissance. They introduce a number of graph-based formulations for robotic path planning and we will extend these techniques to design charging robot rendezvous paths.

In this work, we present three main contributions. First, we formulate the problem of multi-robot recharging in persistent surveillance tasks. Second, we convert the problem into a one-in-a-set path planning problem on a directed acyclic graph (DAG) with vertex partitions, and prove that the problem is NP-hard. Finally, we formulate the problem as a mixed integer linear program (MILP), and for the single-charging robot case, we present a transformation to the standard traveling salesman problem (TSP), for which efficient algorithms are available.

The organization of this paper is as follows. In Section II, we formulate the continuous time rendezvous problem and in Section III, we convert the problem into a discrete formulation that results in a DAG. In Section IV, we prove that our DAG formulation is an NP-hard problem. Section V describes the MILP formulation of the problem and Section VI proposes an algorithm from literature to solve this problem for the single robot case.

II. Motion Planning For Charging Robots

Given a set of working robots performing a task and a set of charging robots, the recharging problem is to find a path for each charging robot such that the team of chargers meets every worker exactly once while minimizing a certain objective function. The working robots are not allowed to divert from their trajectories in order to minimize hindrances to the persistent mission caused by the recharging process. The assumption is made that charging robots possess sufficient energy resources and need not be refueled or restocked within the planning horizon. The problem can now be formally stated.

A. Problem Formulation

Consider an environment $\mathcal{E} \subset \mathbb{R}^2$ which contains a team of R working robots, performing a persistent task. Each working robot, indexed by $r \in \{1, \ldots, R\}$, is described by its motion along a known trajectory, $p_r(t) \in \mathcal{E}$ within a planning horizon $t \in [0, T_r]$, and a charging time window $[\underline{T}_r, \overline{T}_r] \subseteq [0, T_r]$.

The environment also contains a set of M charging robots that are free to move arbitrarily. Each charging robot, indexed by $m \in \{1, \ldots, M\}$, is described by its initial position $p_m(0)$ and its speed, v. We assume that all charging robots have equal and constant velocities in this formulation. The problem is to find optimal paths for the charging robots, $p_m(t) \in \mathcal{E}$ (where $|\dot{p}_m(t)| \leq v$) such that for each $r \in \{1, \ldots, R\}$, there exists a ground robot $m \in \{1, \ldots, M\}$ and a time $t_r \in [\underline{T}_r, \overline{T}_r]$ for which $p_m(t_r) = p_r(t_r)$.

This constraint states that the team of charging robots must rendezvous at least once with each working robot at a point along its respective path before it runs out of charge. Figure 2 illustrates the problem statement with a team of four working robots and two charging robots following a single path.

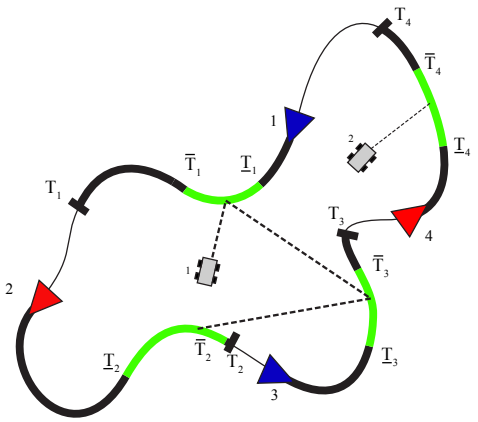


Fig. 2. Four working robots (blue and red triangles) traveling along one path. For each working robot r, $[0, T_r]$ is denoted by a bold black line and $[\underline{T}_r, \overline{T}_r]$, by a bold green line. The two grey charging robots must meet all working robots on their paths within their charging windows to guarantee persistent operation.

III. Discrete Problem Formulation

The continuous-time problem, as stated, requires an optimization over the space of all charging robot trajectories. In this section we present a graph-based formulation of the discretized problem.

A. Transformation from Continuous-Time

For each working robot, given that $p_r(t)$ is known over the planning horizon, we can discretize its charging time window to generate a set of K_r charging times $\tau_r = \{t_{r,1}, \ldots, t_{r,K_r}\} \subseteq [\underline{T}_r, \overline{T}_r]$ at which it can be reached along its trajectory. The set of charging points that result are defined as,

$$C_r = \{(p_r(t), t) \,|\, t \in \tau_r\}. \tag{1}$$

Each charging point $(p_r(t_{r,i}), t_{r,i})$ is described by its time of occurrence, $t_{r,i}$, and its position along the robot path $p_r(t_{r,i})$.

A charging robot, subject to its speed constraints, will attempt to charge a working robot by arriving at one of its charging points $(p_r(t_{r,i}), t_{r,i}) \in C_r$ at a time $t \leq t_{r,i}$ and remaining there until time $t_{r,i}$ such that $p_m(t_{r,i}) = p_r(t_{r,i})$. This expanded definition satisfies the previously stated definition of a rendezvous in continuous time. Note that for the sake of simplicity, the formulation assumes instantaneous charge, but it can be extended directly to the case of nonzero charging durations, as discussed in Remark III.1.

B. Graph-Based Representation

A weighted directed graph $G = (V, E, c)$ is built, where V is the set of vertices, E is the set of directed edges between them, and $c : E \to \mathbb{R}$ is the set of costs for each edge in E. The elements of the graph are defined as follows.

Vertices: The vertices, are defined by $R+1$ disjoint vertex sets, $V_0, V_1, \ldots, V_R$. The set V_0 is the set of initial locations of the charging robots. The set V_r, for $r \in \{1, \ldots, R\}$ is given by C_r, the set of all charging points for robot r. The complete vertex set is then $V = V_0 \cup V_1 \cup \cdots \cup V_R$.

Edges: We add an edge (v_i, v_j), where $i \in V_{r_1}$ and $j \in V_{r_2}$ for some $r_1, r_2 \in \{1, \ldots, R\}$ with $r_1 \neq r_2$, to E if there exists a feasible traversable path from charging point $(p_{r_1}(t_{r_1,i}), t_{r_1,i})$ to $(p_{r_2}(t_{r_2,j}), t_{r_2,j})$. That is, if

$$\frac{\|p_{r_2}(t_{r_2,j}) - p_{r_1}(t_{r_1,i})\|}{\upsilon} \leq t_{r_2,j} - t_{r_1,i}. \tag{2}$$

Edge Costs: Each edge $e = (v_i, v_j) \in E$ is associated with a non-negative cost $c(e)$ that can be chosen based on the objective of the optimization. As an example, one objective is the following. For an edge $(v_i, v_j) \in E$, where $i \in V_{r_1}$ and $j \in V_{r_2}$ for some $r_1, r_2 \in \{1, \ldots, R\}$ with $r_1 \neq r_2$, we have the corresponding charging points $(p_{r_1}(t_{r_1,i}), t_{r_1,i})$ to $(p_{r_2}(t_{r_2,j}), t_{r_2,j})$ and can define the cost as the time between charges,

$$c(e) = t_{r_2,j} - t_{r_1,i}.$$

Remark III.1 (Nonzero Charging Durations). For simplicity of presentation we have assumed that charging occurs instantaneously. Thus, if a charging robot rendezvous' with a working robot at charging point $(p_r(t_{r,i}), t_{r,i})$, then it can leave that charging point at time $t_{r,i}$. We can extend this formulation to charging points described as triples $(p_r(t_{r,i}), t_{r,i}, \Delta t_{r,i})$, where $\Delta t_{r,i}$ is the time required to charge robot r at the ith charging point. In this case the charging robot can leave the charging point at time $t_{r,i} + \Delta t_{r,i}$. The condition to add an edge in equation 2 then changes slightly to

$$\frac{\|p_{r_2}(t_{r_2,j}) - p_{r_1}(t_{r_1,i})\|}{\upsilon} \leq t_{r_2,j} - (t_{r_1,i} + \Delta t_{r_1,i}).$$

•

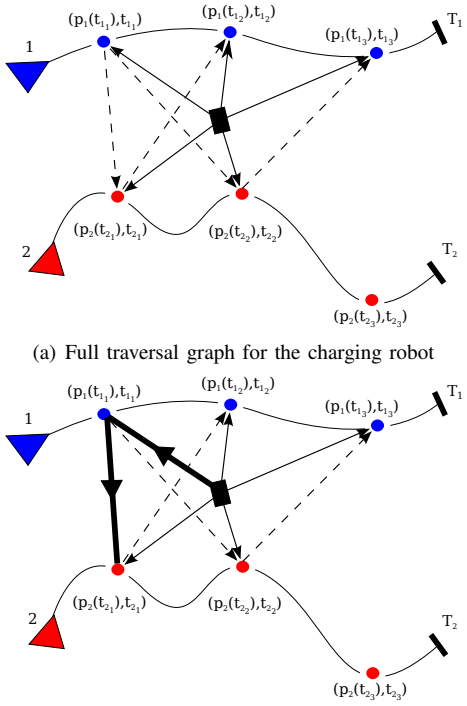
As a simple illustrative example, Figure 3(a) shows two working robots r_1 (blue) and r_2 (red) following arbitrary trajectories and one charging robot m_1 in an environment $\mathcal{E} \subset \mathbb{R}^2$. Each robot path is discretized into three charging points and graph G is constructed through them based on the feasibility conditions.

We can define the constructed graph $G := (V, E, c)$ based on its observed attributes as a *Partitioned DAG*.

Definition III.2 (Partitioned DAG)**.** A partitioned DAG is a graph $G = (V, E, c)$ that consists of directed edges with no directed cycles and has its set of vertices partitioned into R exhaustive and mutually exclusive sets $(V_1, \ldots, V_R)$ such that (i) $V_i \in V$ for all i, (ii) $\cup_i V_i = V$, and (iii) $V_i \cap V_j = \emptyset$ for all $i \neq j$.

C. Problem Formulation on a Partitioned Directed Acyclic Graph

To characterize the complexity of our problem, it will be helpful to state it as a graph optimization. To begin, let us define some notation. For a set V we let $|V|$ denote its cardinality. Similarly, for a path P in a graph, we let



(a) Full traversal graph for the charging robot

(b) Optimal Recharge Path Solution

Fig. 3. Building a traversal Graph for two working robots and one charging robot. The resulting graph is a directed acyclic graph with vertex partitions

$|P|$ denote the number of vertices on the path. For a graph $G = (V, E)$, a path P in G, and a subset of vertices $A \subseteq V$, we let $|P \cap A|$ denote the number of vertices in A on the path P. Similarly, for a set of paths $P = \{P_1, \ldots, P_M\}$, we let $|P \cap A|$ denote the number of vertices in A on that lie on a path $P_m \in P$ for some $m \in \{1, \ldots, M\}$. Given a partitioned DAG, the goal is to find an optimal set of paths that visits each set in the partition once, as shown in Figure 3(b).

It will be helpful for what follows to begin by defining the Multiple *One-in-a-set DAG* decision problem.

Problem III.3 (The Multiple One-in-a-set DAG Decision Problem)**.** Consider a DAG $G = (V, E)$ and a partition $(V_0, V_1, \ldots, V_R)$ of V where $V_0 = \{v_{01}, \ldots, v_{0M}\}$. Does there exist a set of paths $P = \{P_1, \ldots, P_M\}$ in G, where P_m starts at $v_{0m} \in V_0$, such that

$$|V_i \cap P| = 1 \quad \text{for all } i \in \{0, 1, \ldots, R\}?$$

We will say that the DAG G, with partition $(V_0, \ldots, V_R)$ contains One-in-a-set path(s) if and only if the answer to the corresponding decision problem is yes.

The One-in-a-set path problem has been proved to be NP-hard for the case of undirected, complete, or general directed graphs, because they are direct special cases of the Traveling Salesman Problem (TSP). Unlike these TSP problems, the One-in-a-set DAG problem consists of a path through a directed acyclic graph, which is not trivially provable as NP-hard given that the longest path problem for directed acyclic graphs is solvable in polynomial time using dynamic programming [10]. However, in the following section we

prove that the One-in-a-set DAG is in fact an NP-hard Problem.

IV. Proof of NP-hardness

We will prove NP-hardness of the One-in-a-set DAG problem by using a reduction from the well known NP-Complete Hamiltonian path problem [11].

Problem IV.1 (The Hamiltonian Path Problem)**.** Given an undirected graph $G = (V, E)$ with $|V| = n$, does there exist a path P in G such that

$$|V \cap P| = n?$$

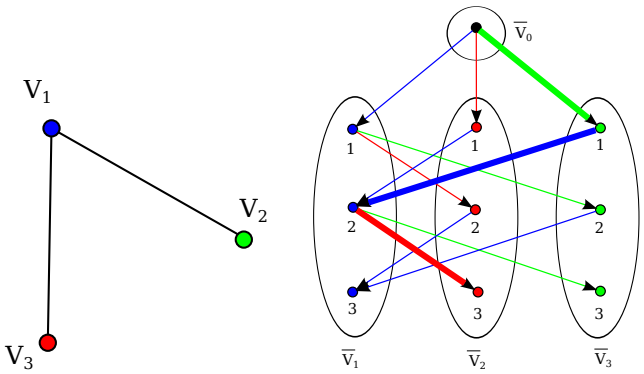
Theorem IV.2 (NP-Completeness of One-in-a-set DAG decision problem)**.** *The One-in-a-set DAG decision problem is NP-Complete.*

Proof. Suppose we have an instance $G = (V, E)$ of the Hamiltonian path problem. We will give a polynomial transformation of G into an input $\overline{G}$ for the One-in-a-set path in a DAG decision problem.

Given the undirected graph $G = (V, E)$, we need to create a DAG $\overline{G} = (\overline{V}, \overline{E})$ along with the vertex partition $(\overline{V}_0, \overline{V}_1, \ldots, \overline{V}_R)$. Our approach will be to encode every possible Hamiltonian path order in $\overline{G}$. The One-in-a-set DAG decision problem will then have a yes answer if and only if the graph G contains a Hamiltonian path.

Let $V = (v_1, \ldots, v_R)$ and for each $r \in \{1, \ldots, R\}$, let $\overline{V}_R$ be given by R copies of v_r, which we will denote by $\overline{V}_r := (v_{r,1}, \ldots, v_{r,R})$. The jth copy of v_r will correspond to all paths in G that have v_r as their jth vertex. Finally, we create a (dummy) vertex $\overline{V}_0$ and define $\overline{V} = \overline{V}_0 \cup \overline{V}_1 \cup \cdots \cup \overline{V}_R$.

Now, we define the edges $\overline{E}$ as follows. We begin by adding an the edge $(\overline{V}_0, v_{r,1})$ to $\overline{E}$ for each $r \in \{1, \ldots, R\}$. Then for any two sets $\overline{V}_i$ and $\overline{V}_j$ and for $k \in \{1, \ldots, R-1\}$ we add the edge $(v_{i,k}, v_{j,k+1})$ if and only if $(v_i, v_j) \in E$. Figure 4 illustrates this reduction and shows that a feasible path is found in the DAG. It is clear that a feasible solution to the described One-in-a-set DAG problem yields a feasible solution to the Hamiltonian path problem.



(a) $G = (V, E)$ (b) $\overline{G} = (\overline{V}, \overline{E})$

Fig. 4. A reduction of the Hamiltonian Path Problem to the One-in-a-set DAG Problem. Each color in graph G represents an individual vertex. Each vertex color in graph G corresponds to a unique vertex set in graph $\overline{G}$

This defines the input $\overline{G}$ to the One-in-a-set DAG decision problem. It is easy to see that $\overline{G}$ is acyclic since it has a topological sort: Define the partial ordering as $v_{i,k} \leq v_{j,\ell}$ if and only if $k \leq \ell$ and note that there is an edge from $v_{i,k}$ to $v_{j,\ell}$ only if $\ell = k+1$. Also, note that $\overline{G}$ has $R^2 + 1$ vertices.

Finally, we just need to show that G contains a Hamiltonian path if and only if $\overline{G}$ contains a One-in-a-set path. Suppose G contains a Hamiltonian path $P = v_{r_1} v_{r_2} \cdots v_{r_R}$, where $(v_{i_j}, v_{i_{j+1}}) \in E$ for each $j \in \{1, \ldots, R-1\}$. Then, the path $\overline{P} = \overline{V}_0, v_{r_1,1} v_{r_2,2} \cdots v_{r_R,R}$ is a One-in-a-set path in $\overline{G}$ since each edge $(v_{r_j,j}, v_{r_{j+1},j+1})$ is in $\overline{E}$.

Conversely, suppose that $\overline{G}$ contains a One-in-a-set path $\overline{P}$. By the definition of the edges $\overline{E}$, the path must be of the form $\overline{V}_0, v_{r_1,1} v_{r_2,2} \cdots v_{r_R,R}$. This implies that $(v_{r_j}, v_{r_{j+1}}) \in E$ for each $j \in \{1, \ldots, R-1\}$ and thus $P = v_{r_1} \cdots v_{r_R}$ is Hamiltonian path in G. □

NP-Completeness of the One-in-a-set DAG decision problem proves that the problem of finding optimal charging robot routes through sets of charging points is NP-hard. Since the single charger route problem is a special case of the multiple charging robot problem, the multi-robot decision problem is also NP-Complete. We approach the solution by first formulating a MILP to solve small cases and then present the application of an existing TSP heuristic algorithm [12], [13] to generate a solution for the single charging robot case.

V. Mixed Integer Linear Program Formulation

Given the graph $G = (V, E, c)$ and a partition of V into vertex sets $(V_0, V_1, \ldots, V_R)$, the One-in-a-set DAG decision problem can be stated as a MILP as follows.

Each charging robot is represented by an independent route p_m. Thus, a binary decision variable is defined as $x_{ijm} \in \{0, 1\}$ with $x_{ijm} = 1$ if, in route p_m, the vertex v_j is visited after vertex v_i, where $i \in V_{r_1}, j \in V_{r_2}, r_1 \neq r_2$ and $r_1, r_2 \in \{1, \ldots, R\}$. The edge-traversal cost, denoted by c_{ij}, is defined as follows. For the edge $e = (v_i, v_j)$ (with associated decision variable x_{ijm} we define

$$c_{ij} = \begin{cases} c(e), & \text{if } e \in E, \\ \infty, & \text{if } e \notin E. \end{cases} \tag{3}$$

For each charging robot m, index 0 represents the initial position of the charging robot, $p_m(0)$. The solution paths must end at the dummy vertex denoted by index d.

$$\min \sum_{m=1}^{M} \sum_{i \in V} \sum_{j \in V} c_{ij} x_{ijm} \tag{4}$$

subject to

$$\sum_{j \in V \setminus V_0} x_{0jm} = 1 \qquad \forall m \in \{1, \ldots, M\} \tag{5}$$

$$\sum_{i \in V \setminus V_0} x_{idm} = 1 \qquad \forall m \in \{1, \ldots, M\} \tag{6}$$

$$\sum_{m=1}^{M} \sum_{j \in V_r} \sum_{i \in V} x_{ijm} = 1 \qquad \forall r \in \{1, \ldots, R\} \tag{7}$$

$$\sum_{m=1}^{M} \sum_{i \in V_r} \sum_{j \in V} x_{ijm} = 1 \qquad \forall r \in \{1, \ldots, R\} \tag{8}$$

$$\sum\nolimits_{i,j \in V} x_{ikm} - x_{kjm} = 0 \qquad \forall m \in \{1, \ldots, M\}, \; \forall k \in V \setminus V_0 \tag{9}$$

$$x_{ijm} \in \{0, 1\} \qquad \forall i, j \in V, \; \forall m \in \{1, \ldots, K\} \tag{10}$$

The objective function (4) seeks to minimize the total cost of the recharge. Constraint (5) and (6) ensure that each charging robot route starts and ends at the initial charging robot location. Constraints (7) and (8) guarantee that each vertex set is visited just once over the combined set of charger routes. Constraint (9) checks that in each route the in-degree is equal to the out-degree for every vertex. The total number of constraints in this formulation is $(2R + M|V|)$. If $n = |V \setminus V_0|$ is the total number of charging points, then the total number of binary variables is upper bounded by $Mn(n+2)$.

Note that unlike a standard TSP solution on a complete graph, the One-in-a-set TSP introduces a smaller constraint set due to sparse edges in a graph with vertex partitions. Further, the lack of sub-tours within a directed acylic graph removes the need for sub-tour elimination constraints and enables faster MILP solutions. For larger problem sizes, however, the optimal MILP solution is not computationally feasible and Section VI describes the algorithmic approach used to obtain a solution.

VI. Algorithmic Approach

We now present an algorithmic approach to solve the single charging robot problem based on a transformation of the One-in-a-set DAG path problem into a standard TSP for which various heuristic and approximation algorithms exist. One of the best TSP heuristic algorithms currently in use is the Lin-Kernighan heuristic [12], further extended by Helsgaun in [13].

It is observed that the One-in-a-set path in this formulation resembles a Generalized Traveling Salesman Path (GTSP) referred to in [14]. The approach employed to solve the GTSP in this work is a transformation into an Asymmetric TSP (ATSP) using the *Noon-Bean Transformation* [14]. The ATSP can then be solved using the LKH solver which is a freely available implementation of the Lin-Kernighan-Helsgaun heuristic.

A. Noon-Bean Transformation of GTSP to ATSP

The following is a brief description of the graph transformation procedure used. Given a weighted directed graph $G = (V, E, c)$ with a partition of the vertices into R exhaustive and mutually exclusive sets $V = (V_1, \ldots, V_R)$, a new graph $G' = (V', E')$ can be constructed such that an ATSP solution on G' yields the GTSP solution on G.

(i) **Transforming GTSP to ATSP**.
   (a) Create an arbitrary ordering of vertices within each vertex set V_r and add zero-cost directed edges E' to create a zero-cost cycle that traverses all vertices in the set.
   (b) Shift inter-cluster edges in E so that tail-end of each edge originates from the preceding intra-set vertex in its respective zero-cost cycle. Add these edges to E'.
   (c) Add a large penalty $\beta > \sum_{i,j \in V} c_{ij}$ to the inter-set edges to ensure that any TSP path entering a vertex set will traverse through all its vertices before moving on to the next set.

(ii) **Obtain GTSP solution from ATSP solution**. The ATSP can be solved using the LKH solver to obtain a near-optimal ordering of vertices. Due to the previous edge manipulations, the One-in-a-set path can be extracted by picking the first vertex from each cluster in the ATSP solution.

VII. Simulations

The optimization framework for this paper was implemented in MATLAB® and the mixed integer linear programs were solved using the IBM CPLEX® solver. The solutions were computed on a laptop computer running a 32 bit Ubuntu 12.04 operating system with a 2.53 GHz *Intel Core2 Duo* processor and 4GB of RAM. In this section, we solve the MILP optimization for a multiple charging robot scenario, compare optimal and heuristic solutions for a reasonably small sized single charging robot scenario and demonstrate a large scale single charging robot simulation solved using the LKH TSP Solver.

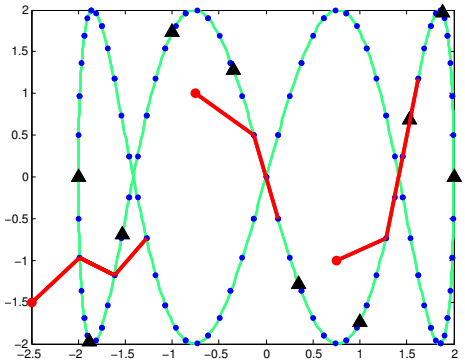


Fig. 5. A MILP solution to the multiple charging robot problem with ten working robots (triangles) distributed over one path and three charging robots (circles).

Figure 5 shows an optimal CPLEX solution for ten working robots (black triangles) distributed along a single path

and three charging robots (red dots) placed at random within the environment. The path is discretized into one hundred charging locations (blue dots) and a set of charging points is generated within a charging window of 50% to 75% voltage depletion for each working robot. The objective of the optimization is chosen to be an equally weighted sum of total path distance and recharge time. The graph built for this problem contained 385 vertices in 10 vertex sets, and the optimization took 1.9 minutes to solve. A simulation for a similar problem instance is illustrated in the video attachment accompanying this paper.

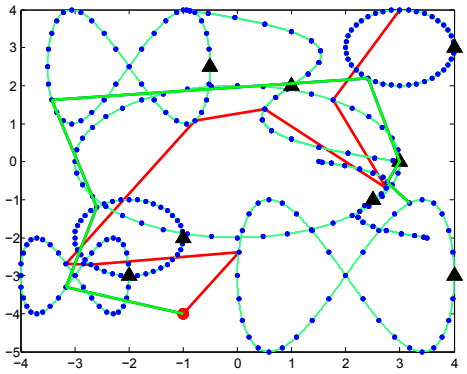


Fig. 6. Comparison of the optimal CPLEX solution (green path) against the heuristic LKH solution (red path). The problem consists of 8 working robots (triangles) on 8 paths.

Figure 6 compares the heuristic LKH solution against the optimal CPLEX MILP solution. The problem instance consists of eight working robots on eight paths. The DAG consists of 8 vertex partitions and 500 vertices. The optimal solution for a single charging robot(green path) was computed by CPLEX in 6.6 minutes. The heuristic solution (red path) was computed by LKH in 0.16 minutes and resulted in a path cost 12.8% higher than the optimal cost.

Since a MILP solution is not computationally feasible for larger problems a realistic large scale problem has been heuristically solved and illustrated in Figure 7. The environment consists of sixteen working robots, evenly distributed among eight paths. The resulting graph for the TSP solution contains 16 vertex sets and 5,761 vertices. The LKH solver obtained a solution in 9.4 minutes.

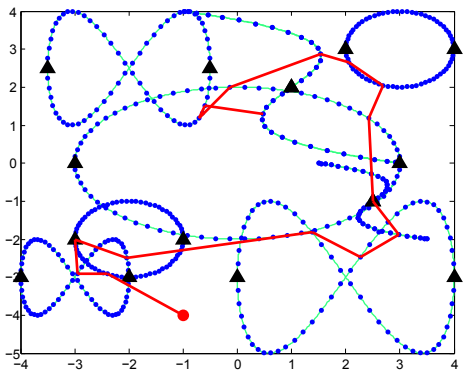


Fig. 7. The heuristic solution(computed using LKH) to the single charging robot problem with sixteen working robots distributed among eight paths.

VIII. Conclusions

In this paper, we present the problem of maintaining continuity in persistent tasks with coordinated teams of autonomous robots. We construct a partitioned directed acyclic graph using pair-wise rendezvous condition checks between charging points generated for each of the working robots. We prove that the One-in-a-set DAG problem is an NP-hard problem and present the MILP formulation for a generalized problem instance. Finally, we apply the Noon-Bean transformation and the LKH Solver on large instances of the single charging robot problem to rapidly obtain reasonable solutions with limited computational resources.

For future work, we are interested in exploring the transformation of the multiple charging robot problem to a TSP. We also plan to consider real environments with charging robots traversing uneven obstacle ridden terrains, as well as dockable aerial robots with time varying battery depletion rates. As coordinated autonomous robots increasingly progress from research to real applications, it is imperative to consider energy-awareness in persistent tasks to overcome the inherent limitations of mobile autonomous robots.

References

[1] J. Cortes, S. Martinez, T. Karatas, and F. Bullo, "Coverage control for mobile sensing networks," *IEEE Transactions on Robotics and Automation*, vol. 20, no. 2, pp. 243 – 255, 2004.

[2] S. L. Smith, M. Schwager, and D. Rus, "Persistent robotic tasks: Monitoring and sweeping in changing environments," *IEEE Transactions on Robotics*, vol. 28, no. 2, pp. 410–426, 2012.

[3] F. Pasqualetti, J. W. Durham, and F. Bullo, "Cooperative patrolling via weighted tours: Performance analysis and distributed algorithms," *IEEE Transactions on Robotics*, vol. 28, no. 5, pp. 1181 –1188, 2012.

[4] K. Swieringa, C. Hanson, J. Richardson, J. White, Z. Hasan, E. Qian, and A. Girard, "Autonomous battery swapping system for small-scale helicopters," in *IEEE International Conference on Robotics and Automation*, May 2010, pp. 3335 –3340.

[5] K. Suzuki, P. Kemper Filho, and J. Morrison, "Automatic battery replacement system for uavs: Analysis and design," *Journal of Intelligent and Robotic Systems*, vol. 65, pp. 563–586, 2012.

[6] J. Derenick, N. Michael, and V. Kumar, "Energy-aware coverage control with docking for robot teams," in *IEEE International Conference on Intelligent Robots and Systems*, 2011, pp. 3667–3672.

[7] Y. Litus, P. Zebrowski, and R. Vaughan, "A distributed heuristic for energy-efficient multirobot multiplace rendezvous," *IEEE Transactions on Robotics*, vol. 25, no. 1, pp. 130 –135, 2009.

[8] Y. Litus, R. T. Vaughan, and P. Zebrowski, "The frugal feeding problem: Energy efficient, multi-robot, multi-place rendezvous," in *IEEE International Conference on Robotics and Automation*, 2007, pp. 27–32.

[9] K. J. Obermeyer, P. Oberlin, and S. Darbha, "Sampling-based path planning for a visual reconnaissance unmanned air vehicle," *Journal of Guidance, Control, and Dynamics*, vol. 35, no. 2, pp. 619–631, 2012.

[10] D. Eppstein, "Finding the k shortest paths," in *Foundations of Computer Science, 1994 Proceedings., 35th Annual Symposium on*, 1994, pp. 154 –165.

[11] B. Korte and J. Vygen, *Combinatorial Optimization: Theory and Algorithms*, 4th ed., ser. Algorithmics and Combinatorics. Springer, 2007, vol. 21.

[12] S. Lin and B. W. Kernighan, "An effective heuristic algorithm for the traveling-salesman problem," *Operations Research*, vol. 21, no. 2, pp. pp. 498–516, 1973.

[13] K. Helsgaun, "General k-opt submoves for the linkernighan tsp heuristic," *Mathematical Programming Computation*, vol. 1, pp. 119–163, 2009.

[14] C. E. Noon and J. C. Bean, "An efficient transformation of the generalized traveling salesman problem," *INFOR*, vol. 31, no. 1, pp. 39 – 44, 1993.