Decentralized Drone Swaps for Online Rebalancing of Drone Delivery Tasks

Kamran Vakil and Alyssa Pierson

***Abstract*— Recent research has seen the advancement of drone depot models as a promising way to allocate drones for large-scale task completion. Applications of these drone depot models include data collection, environmental monitoring, package delivery, and more. This paper focuses on sharing agents between static depots for task allocation based on expected demand. We model the problem as a Binary Nonlinear Program, then derive an iterative neighborhood search based on solving a series of Binary Linear Programs to drive towards the optimal configuration of agents for each depot. We show that our method is more tractable than a Branch and Bound approach for this model as problem complexity grows. We also show through simulations that with near optimal allocation between local depots, the overall system performance will outperform greedy and non-sharing approaches.**

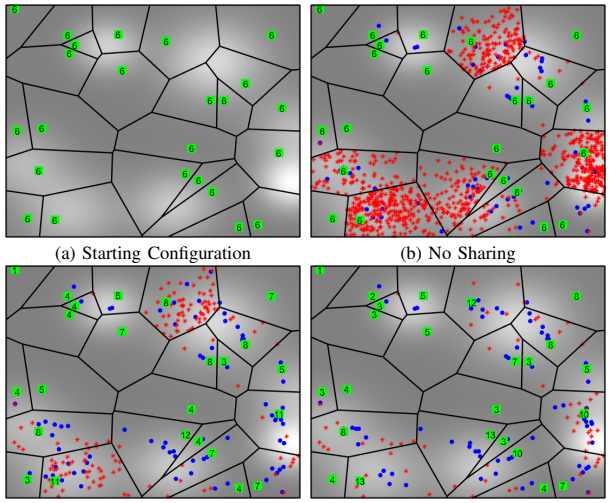
I. INTRODUCTION

Recent robotics advancements enables drones for applications including deliveries, environmental monitoring, event servicing, and more. Often, these drones are combined with depots where drones can recharge, resupply, update new information to a larger network, and more. These depots are often statically placed over the long term, and thus must be sufficiently staffed with drones to deal with varying amounts of tasks over time. As such, sharing drones between depots emerges as a potential solution to increase system efficiency without increasing the total number of drones.

We seek to implement a task assignment planner for multi-agent multi-depot problems with inspiration from coverage control. Multi-Robot Task Allocation (MRTA) is an extensively researched field [1]–[4] with many models and frameworks for assigning agents to a variety of tasks. However, task assignment planners for these systems often focus on linear models, with a deterministic number of tasks to be fulfilled. Coverage control algorithms allow coverage of an expected demand over an area with a team of adaptive agents, where each agent is the generator of its environment partition. We assume the drone depots for recharging and communication uplinks are at fixed locations within an environment, which means they may not be at a locationally-optimal configuration to varying demand. The challenge is to assign drones to depots to best service the events in the environment. The main contributions of this paper are:

1) A Nonlinear Binary Program for decentralized sharing of heterogeneous drones between depots;
2) An iterative neighborhood search method which finds an agent assignment solution for depots, based on satisfying a necessary condition of optimality;
3) Simulations showing that sharing agents between depots using our search method results in better task fulfillment compared to non-sharing and greedy methods.

Kamran Vakil and Alyssa Pierson are with the Department of Mechanical Engineering at Boston University, Boston, MA 02215, USA. [kamranv, pierson]@bu.edu



(a) Starting Configuration (b) No Sharing

(c) Greedy Sharing (d) BLP Sharing

Fig. 1: Simulated Scenario for discrete event satisfaction. Green squares represent static depots with the number of heterogeneous drones assigned to them. Each depot covers the area within their black borders. Drones are represented as blue circles, and are shared between depots to service discrete events. Events are represented as red stars and can be fulfilled by drones. Lighter areas in the environment represent more probable locations for discrete events to appear. We see that better allocation of drones using our BLP method results in less outstanding events over time.

Related Work

Multi-agent systems with depots has been well researched in recent years, due to its adaptability to real world applications. [5] coordinates recharging agents at charging stations while exploring an environment. [6] maintains connectivity between agents and a base station during exploration. [7], [8] provide a field example of using mobile vans to cycle drones during long term missions. [9], [10] offer overviews for utilizing drone deliveries for biological material transportation. [11], [12] both plan paths using a combination of mobile ground vehicles which can recharge mobile aerial vehicles to service tasks. We refer the reader to [13] for further examples which use multi-agent multi-depot models.

Task assignment planners are commonly used to assign agents to a set of tasks based on agent utility, agent constraints, task constraints, and more [1], [2], [14]. These planners often reduce the assignment problem to a centralized linear model, formulate task satisfaction using an Integer Linear Program [6], [10], [15] or a Mixed Integer Linear Program [16] in cases with more complex constraints.

These problems can be solved using optimal methods [4] such as Branch and Bound, or through heuristics such as neighborhood search [16]. These formulations are the basis of our search method, which fulfills a necessary condition of optimality. Other methods of task assignment include coalition formation [17], which partitions agents into groups to maximize overall system utility by increasing task performance. Common approaches include distributed auction-based methods [18], hedonic games [19], or other game-theoretic models.

Coverage control planners allow agents to equitably cover an environment weighted by an underlying field. Many methods for utilizing coverage controllers with heterogeneous agents have been researched, which inspires our approach. Learning methods such as [20] and [21] allow for complex agent decision making, while more classical approaches such as [22] allow for faster, distributed coverage with large scale teams. Methods combining the two approaches such as [23] allow heterogeneous agents to learn based on past performance and react in a distributed fashion. We draw inspiration from multi-robot coverage control in our decentralized algorithm design, as it provides a framework to model locational cost of expected tasks over our environment.

II. Problem Formulation

In this section, we formulate the model for long term task completion for each depot based on their locational cost in the environment and the utility of their assigned agents. We then represent the assignment of agents to these depots as a Nonlinear Binary Program, where agents contribute to completing tasks based on their own unique utility.

A. Locational Cost for Single Depot Coverage

We start by considering a convex environment $Q \subseteq \mathbb{R}^2$, with points $q \in Q$. Discrete events will appear in the environment, which must be serviced by drones from depots. We assume there are H static depots within Q at positions $p = \{p_1, \dots, p_H\}, p_h \in Q$. Each depot dispatches drones to serve events within a region based on their Voronoi partition,

$$V_h = \{q \in Q \mid \|q - p_h\| \leq \|q - p_k\|\},$$

where $k \neq h$. We represent the probability of a serviceable event occurring at some location with density function $\phi(q)$, and the expected demand of a single depot as its locational cost [22],

$$d_h = \int_{V_h} \|q - p_h\| \phi(q) dq. \tag{1}$$

We choose this cost based on the assumption that drones return to their depot after they service an event. Thus, we assume the distance of an event from a depot is linearly proportional to the difficulty in servicing the event.

B. Depot Cost

We now must relate the expected demand of a depot to its ability to satisfy this demand. We assume there are N drones which are assigned to the depots $D_h, h = \{1, \dots, H\}$. Each drone is assigned a heterogeneous utility value $a_i \in \mathbb{R}, a_i > 0, i = \{1, \dots, N\}$. We define utility as how efficient an agent is at servicing tasks, and assume depots can share drones to each other, where the travel time between depots is negligible compared to time spent servicing tasks. We also assume that agents can independently service tasks in parallel. We define $\mathcal{A}_h$ as the set of M_h drones assigned to depot D_h, and define $\mathcal{A} = \{\mathcal{A}_1, \dots, \mathcal{A}_H\}$ as the set of assignments of drones to all depots. We also define $\mathbf{A}_h = [a_1, \dots, a_M]^\top$ as a vector of the utility of drones within $\mathcal{A}_h$, and $A_h = \sum_{m=1}^{|\mathbf{A}_h|} a_m$ as the total utility of drones assigned to D_h.

We define the depot's locational cost based on the utility of all assigned drones at that depot. Each drone's individual utility is determined by their speed. Note since drones satisfy task independently, we can combine their utility when assigned to a depot (i.e., two drones with utilities of 1 are equivalent to one drone with a utility of 2). We wish to distribute utility to depots based on their expected demand, such that no specific depot is overwhelmed with tasks (see Figure 1b). The new cost at a depot D_h is defined

$$J_h = \frac{d_h}{c_h + A_h}, \tag{2}$$

where c_h is some fixed utility of the depot, i.e. a drone that cannot be reassigned. We assume $c_h \geq 0, c_h + A_h > 0, \forall h = \{1, \dots, H\}$. These constraints mean a depot should always have some ability to service tasks. The intuition behind (2) is that depots with more demand must be given proportionally more utility to lower cost. The system cost across all depots is written

$$J = \sum_{h=1}^{H} \frac{d_h}{c_h + A_h}. \tag{3}$$

Our goal is to find the optimal set of assignments of drones to all depots $\mathcal{A}^* = \{\mathcal{A}_1^*, \dots, \mathcal{A}_H^*\}$ which minimizes (3). To do so, we formulate the problem as a Nonlinear Binary Program, detailed in the next section.

C. Nonlinear Binary Program Formulation

We define a Nonlinear Binary Program (NLBP) to find the best assignment of drones to depots. We define the binary decision variables $z_{ih} \in \{0, 1\}, \forall i = \{1, \dots, N\}, \forall h = \{1, \dots, H\}$. If $z_{ih} = 1$, then drone a^i is assigned to depot D_h. We also assume that each drone cannot be assigned to multiple depots, thus we obtain the constraint $\sum_{h=1}^{H} z_{ih} = 1, \quad \forall i = \{1, \dots, N\}$. We seek to minimize (3) through the assignment of agents. Thus, we formulate the assignment of agents with their specific utilities as part of the NLBP

$$\begin{aligned} &\underset{Z}{\operatorname{argmin}} \sum_{h=1}^{H} \frac{d_h}{c_h + A^\top Z e_h}, \\ &s.t. \quad c_h + A^\top Z e_h > 0, \quad \forall h = \{1, \dots, H\} \\ &z_{ih} \in \{0, 1\}, \quad \sum_{h=1}^{H} z_{ih} = 1, \\ &\forall i = \{1, \dots, N\}, \quad \forall h = \{1, \dots, H\} \end{aligned} \tag{4}$$

where $A = [a_1, ..., a_N]^\top$, Z denotes a matrix of z_{ih} with dimension $N \times H$ such that $Z_{ih} = z_{ih}$, and e_h denotes the standard basis vector. The objective function is a formulation of (3). The term $A^\top Z e_h$ returns A_h, the total utility assigned to a depot. The resulting solution Z^* optimally assigns all N drones to the H depots, and can be converted to $\mathcal{A}^*$.

III. Iterative Neighborhood Search using BLPs

In this section, we present our method of reallocating drones between depots using a decentralized online planner. This method provides a faster near optimal alternative to utilizing a Branch and Bound (B&B) algorithm to solve (4). We first show a necessary condition of optimality which our algorithm fulfills by iteratively solving a series of sub-problems. We then show these sub-problems can be solved optimally using Binary Linear Programs (BLPs) to obtain fast, near optimal sharing solutions.

A. Necessary Condition for Optimal Drone Assignment

While directly solving (4) using a B&B algorithm guarantees an optimal solution, in practice the large solution space and poor bounding given by the objective function makes this method intractable for most non-trivial problems. Figure 2 shows that B&B solution times quickly increase as problem complexity grows. We instead propose using an iterative neighborhood search to reduce runtime. We base our search on the following necessary condition of $\mathcal{A}^*$.

Proposition 1. *If $\mathcal{A} = \mathcal{A}^*$, then $\nexists$ any alternative drone assignments between any 2 depots $\mathcal{A}_k, \mathcal{A}_j$, $\forall k = \{1, \dots, H\}, \forall j = \{1, \dots, H\}, k \neq j$ which lowers the local cost $J(\mathcal{A}_k, \mathcal{A}_j)$ from* (3) *of the two depots.*

Proof. Assume $\mathcal{A}' = A^*$ except assignments $\mathcal{A}'_k, \mathcal{A}'_j$, such that local cost $J(\mathcal{A}'_k, \mathcal{A}'_j) < J(\mathcal{A}^*_k, \mathcal{A}^*_j)$. Depot costs are independent, and by definition $J(\mathcal{A}^*) \leq J(\mathcal{A}')$ for all alternate assignments $\mathcal{A}'$. By contradiction, this new assignment cannot exist without having a lower global cost than $\mathcal{A}^*$. □

Proposition 1 returns a necessary condition of optimality for $\mathcal{A}^*$, it has no drone swaps between any two depots which lowers their local cost. Finding an assignment which fulfills this necessary condition can be computed much faster than finding the optimal assignment for complex problems. In practice, these assignments are still optimal or near optimal.

Now consider swapping discrete utility drones between two depots. We could formulate this sub-problem as in (4). By swapping with every combination of two depots, and repeating until the necessary condition is fulfilled, we would converge to a final arrangement near the optimal drone assignment. However, solving this as a NLBP via B&B can still be very slow as the number of drones increases. Instead, we can format this two depot problem as a BLP. This allows the usage of faster Integer Linear Program (ILP) solvers.

B. Convexity of the relaxed two depot assignment problem

To fulfill the necessary condition from Proposition 1, we must be able to quickly find optimal drone assignments between two depots. We wish to characterize the relaxed two depot objective function as convex. Doing so will help us reformulate the problem as a BLP.

Consider the continuous two depot utility swapping case between depots D_k and D_j, where both depots have initial drone assignments $\mathcal{A}_k$ and $\mathcal{A}_j$. We define the continuous sharing problem between these depots as

$$\underset{x}{\operatorname{argmin}} \quad \frac{d_k}{C_k - x} + \frac{d_j}{C_j + x}, \tag{5}$$

$$s.t. \quad C_k - x > 0, \;\; C_j + x > 0, \tag{6}$$

where $x \in \mathbb{R}$ is a continuous variable representing utility given from D_k to D_j, $C_k = c_k + A_k$, and $C_j = c_j + A_j$. This formulation more directly shows that one depot is sharing to another and ensures that depots can always service demand, though it assumes depots can also share their fixed utilities.

We note that by (1), $d_h \geq 0, \forall h = \{1, \dots, H\}$. If $d_k = d_j = 0$, assignment does not matter, and if one depot demand is 0, the correct assignment is to send all drones to the other depot. Thus, we assume $d_h > 0, \forall h = \{1, \dots, H\}$ for the remainder of the paper.

Proposition 2. *If $d_k, d_j > 0$, the objective function of* (5) *is strictly convex within the constraints* (6) *with a minimum at*

$$x^* = \frac{\sqrt{d_j} C_k - \sqrt{d_k} C_j}{\sqrt{d_k} + \sqrt{d_j}}. \tag{7}$$

Proof. We can easily find the first and second derivative of the objective function of (5) with respect to x as

$$\frac{dJ_{kj}}{dx} = \frac{d_k}{(C_k - x)^2} - \frac{d_j}{(C_j + x)^2} \tag{8}$$

$$\frac{d^2 J_{kj}}{dx^2} = \frac{2d_k}{(C_k - x)^3} + \frac{2d_j}{(C_j + x)^3}, \tag{9}$$

and can see that the constraints (6) force the second derivative to be positive when $d_k, d_j > 0$. The cost goes to infinity as we approach the bounds (6). Setting (8) to 0 and solving within (6) gives one feasible minimum at x^*. □

C. Converting the two depot sharing problem to a BLP

Analyzing (5), we see that for a depot to lower another depots' cost by sharing utility, it must make its own cost increase. Thus, we would like to define the region of sharing where the overall cost between the two depots decreases. To do so, we define the level set bounds of the original assignments $\mathcal{A}_k$, $\mathcal{A}_j$ as

$$J_l = \frac{d_k}{C_k - x^l} + \frac{d_j}{C_j + x^l} = \frac{d_k}{C_k} + \frac{d_j}{C_j}, \tag{10}$$

where x^l is the level set bounds. We note that since the objective function is strictly convex per Proposition 2, the level set is also strictly convex. Thus, we know that any value x within our level set bounds x^l must be less than the cost of the level set bound. We can derive x^l from (10) as

$$x^l_1 = 0, \quad x^l_2 = C_k - C_j + \frac{C_k C_j (d_j - d_k)}{C_j d_k + C_k d_j}, \tag{11}$$

and note that if $x_2^l \leq 0$, we can rearrange which depot shares utility such that $x_2^l \geq 0$.

Since we know the level set bounds x^l and that the objective function is strictly convex, then any x between these values will lower the cost. We create a BLP to lower the cost by considering swaps between the two depots as

$$\underset{z}{\operatorname{argmax}} \begin{bmatrix} \mathbf{A}_k^\top & -\mathbf{A}_j^\top \end{bmatrix} z \tag{12}$$
$$s.t. \begin{bmatrix} \mathbf{A}_k^\top & -\mathbf{A}_j^\top \end{bmatrix} z \leq x_2^l,\ z_m \in \{0,1\},\ \forall m = \{1,\dots,M\}$$

where $x_2^l \geq 0$, M represents the number of drones assigned to D_k and D_j, and z is a $M \times 1$ vector of binary decision variables, where 1 indicates a swap from the drone's initial depot to the other depot. If a solution z results in $x_1^l < \begin{bmatrix} \mathbf{A}_k^\top & -\mathbf{A}_j^\top \end{bmatrix} z < x_2^l$, then there exists a swap of drones whose change in utility lowers the overall depot cost. Thus, the two depot sharing case can be reduced to a BLP.

However, maximizing towards the level set bound in (12) means this formulation might fail or only marginally decrease the cost per swap, rather than finding the optimal drone assignment between the two depots. To find the optimal assignment, we exploit the convexity of (5) and the optimal utility sharing value (7) by instead solving two BLPs,

$$\underset{z}{\operatorname{argmin}} \begin{bmatrix} \mathbf{A}_k^\top & -\mathbf{A}_j^\top \end{bmatrix} z \quad , \tag{13}$$
$$s.t. \begin{bmatrix} \mathbf{A}_k^\top - \mathbf{A}_j^\top \end{bmatrix} z \geq x^*,\ z_m \in \{0,1\},\ \forall m = \{1,\dots,M\}$$

$$\underset{z}{\operatorname{argmax}} \begin{bmatrix} \mathbf{A}_k^\top & -\mathbf{A}_j^\top \end{bmatrix} z \quad , \tag{14}$$
$$s.t. \begin{bmatrix} \mathbf{A}_k^\top & -\mathbf{A}_j^\top \end{bmatrix} z \leq x^*,\ z_m \in \{0,1\},\ \forall m = \{1,\dots,M\}$$

Proposition 3. *At least one of the two results from* (13) *and* (14) *is the optimal two depot swap solution for D_k and D_j.*

Proof. We define x^+ and x^- from (13), (14) as the closest utility values to x^* of all possible solutions which are greater and less than x^*, respectively. Since $d_k, d_j > 0$, the 1-D cost function from (5) is strictly convex per Proposition 2. Thus, $J(x^-) < J(x < x^-)$ and $J(x^+) < J(x > x^+)$. Either $J(x^-) = J(x^+)$, $J(x^-) < J(x^+)$, or $J(x^+) < J(x^-)$. In all cases, the solution with lower cost dominates all other solutions and is optimal. If neither solution has a lower cost than $J(0)$ or is infeasible, then there does not exist any cost lowering swaps between the two depots. □

D. Full Algorithm and Dominant Runtime Analysis

Algorithm 1 shows a non-parallelized version of the iterative search algorithm. Drone assignments are heuristically initialized to depots. Then, the algorithm begins iterating and checks every combination of two depots to see if a drone swap can be made using the BLPs from (13) and (14). If any two depots have a drone swap, that swap is made before continuing. This repeats until there is an iteration where no swaps are found between any two depots. When this occurs, the necessary condition for optimality from Proposition 1 is reached and the current drone assignments are returned. We leave the order of depot pair selections for future work.

The time complexity per iteration for this method becomes $\mathcal{O}(H^2 f(D,A))$, where H is the number of depots being considered and $f(D,A)$ is the runtime for the ILP solver for a two depot swap with their assigned drones, which is usually pseudo-polynomial. This runtime can be reduced through parallelization to $\mathcal{O}(Hf(D,A))$ by comparing different depot swaps at once. Each iteration guarantees cost reduction or convergence to a final solution, and can be stopped anytime.

Algorithm 1 Neighborhood Search for Drone Sharing

Input: Depot demands d_h, depot utilities c_h, drone utilities a_i , $\forall h = \{1,\dots,H\}$, $\forall i = \{1,\dots,M\}$
Output: Predicted Drone Assignments Z

```
Z ← initialDroneAssignment(d_h,c_h,a_i)
flagSearch ← True
while flagSearch do        ▷ Prop 1 Condition Unsatisfied
    flagSearch ← False
    for k from 1 to H − 1 do
        for j from k + 1 to H do
            A_k,A_j ← getDepotDrones(k,j,Z,a_i)
            Z,flagSwapped ← BLPswap(d_k,d_j,c_k,c_j,A_k,A_j,Z) (13),(14)
            flagSearch ← flagSearch ∨ flagSwapped
        end for
    end for
end while
```

IV. Simulations

In this section, we rebalance agents through depots to show the effectiveness of sharing. We start by showing the BLP swap method's performance and runtime versus a typical Branch and Bound algorithm. We then show that local rebalancing across a network of depots allows for faster task satisfaction, and that the BLP swap method outperforms greedy sharing. We utilize Matlab's *intlinprog()* solver, which allows for fast solving of ILPs using Branch and Bound combined with a variety of heuristics. All simulations were run on an Intel Core i7-13700KF at 3.4 GHz.

A. Comparison to Branch and Bound for local sharing

We seek to compare our BLP results to optimal solutions of (4). Theoretically, solving (4) for the optimal drone assignment could be guaranteed using a Branch and Bound (B&B) approach. However, this approach becomes intractable as the problem space grows due to poor bounding of the given objective function. To create a comparison of online performance between B&B and the BLP method, we initialize both planners with the same starting assignments and run them on the same trials until the planners converge, or until a cutoff time of one minute. If the cutoff time is reached, we take the best-known solution. To create our random trials, we create an ego depot which has control of N drones with H neighboring depots. We choose the demand and utility of all depots with uniform randomness. Then, our source depot randomly splits its utility among its drones, keeping a small ϵ of utility for numerical stability.

BLP $\leq$ B&B %	3 drones	6 drones	9 drones	12 drones
3 depots	99.3	97.1	92.0	83.0
6 depots	99.5	93.3	88.6	88.8
9 depots	98.9	93.5	90.9	89.5
12 depots	99.2	94.5	93.0	95.2

TABLE I: Percentage comparison of final assignment cost for drone assignments between BLP method and B&B for $N = 1000$ randomized trials with cutoff time of 1 minute and various depot and drone numbers.

B&B Converge %	3 drones	6 drones	9 drones	12 drones
3 depots	100	100	97.0	79.6
6 depots	100	95.8	79.2	43.0
9 depots	100	74.5	49.5	33.0
12 depots	100	48.1	29.4	16.9
BLP Optimal %	3 drones	6 drones	9 drones	12 drones
3 depots	99.3	97.1	91.8	79.2
6 depots	99.5	93.2	86.2	76.9
9 depots	98.9	93.0	84.1	72.3
12 depots	99.2	91.8	81.9	77.0

TABLE II: Optimality Statistics for N=1000 randomized sharing problems with various depot and drone numbers. Row 1 denotes percentage of problems where B&B finds the optimal solution. Row 2 only considers the subset of test cases with known optimal solutions from B&B. It denotes the percentage of matching BLP solutions for those test cases.

Figure 2 illustrates the solution times for randomized trials of 6 depots with varying numbers of drones. As we see in the figure, the computation time for the B&B algorithm quickly grows with the number of agents, such that it reaches the maximum cutoff time for many of the trials. Comparatively, the BLP method has a more consistent runtime with fast convergence for more complex problems.

Table I provides the cost performance of our algorithm against all trials of B&B. We see that in many cases our BLP solution found a better result due to the B&B failing to converge before the cutoff time. Table II presents the percentage of trials for which B&B converges. Recall when B&B converges, we know this is a globally optimal solution. The second row of Table II shows the percent of BLP trials found to be optimal within the trials for which B&B converged. We include these numbers to illustrate that the BLP often achieves optimal performance. Anecdotally, we note that the BLP planner still returns assignments very close to optimal even when it does not find the optimal solution.

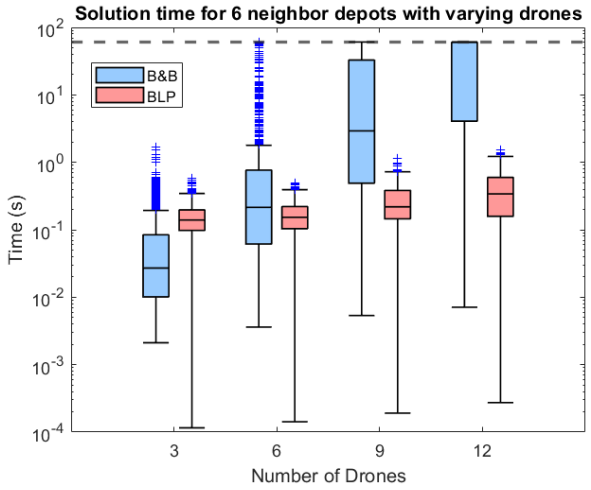


Fig. 2: Box Plot depicting solution times for N=1000 random trials with 6 neighboring depots and varying number of drones. Dotted line depicts maximum allowable solve time before termination.

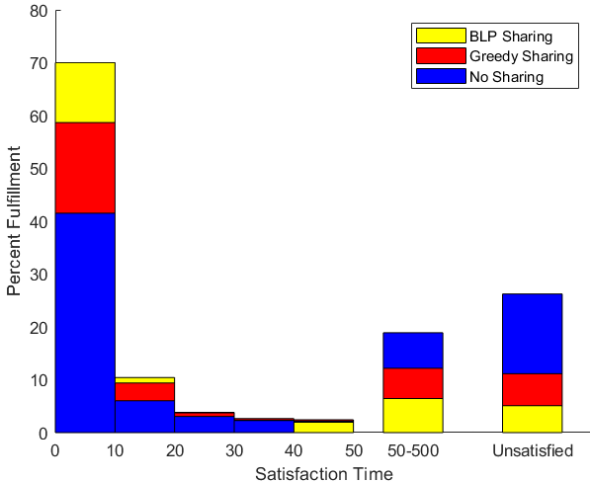
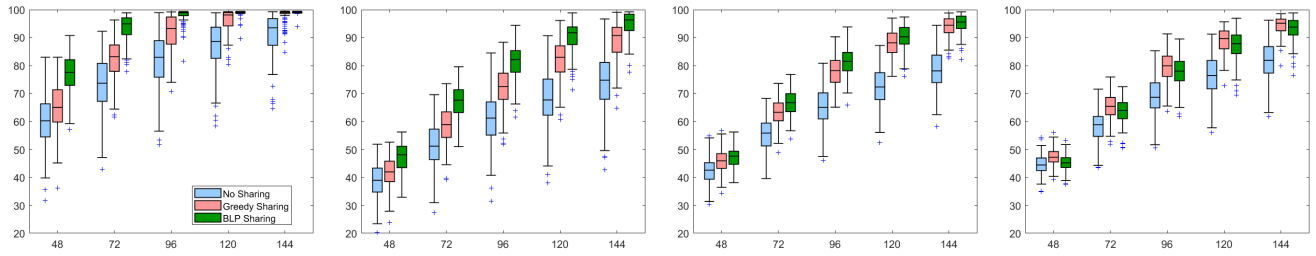


Fig. 3: Histogram depicting satisfaction time for events versus Percent Fulfullment for $N = 100$ trials with local sharing and static demand. Depots, demand, and drone utilities were all randomized, with 24 depots starting with 6 drones each. 10 Discrete tasks were injected every timestep.

B. Local Sharing for multiple depots with static demand

We now implement our approach in simulation as seen in Figure 1. We run 100 randomized simulations with each planner, with 24 depots randomly distributed in the environment. Each depot starts with an equally distributed number of drones, whose utility is set equal to their normally distributed speed. Our underlying density field is assumed to be a Gaussian Mixture Model (GMM), with randomized positions, covariances, and weights. We assume depots can share information and drones with neighbors of their Voronoi partitions. We assume drones can only be shared if they are present at the depot. Each depot only services events within their Voronoi Partition. Events are injected into the environment continuously for the duration of the simulation, and depots prioritize the oldest events with their highest utility drones. We compare our BLP planner to a non-sharing planner and a naive greedy sharing planner. The non-sharing planner maintains the same drones it is originally assigned regardless of neighboring demand. The naive greedy planner computes the optimal continuous utility of the relaxed multi-depot problem for itself and its neighbors. It shares to neighbors if it has any surplus drones while keeping its total utility above its local optimal continuous utility value.

Figures 3 shows the percentage of tasks fulfilled versus satisfaction time for tasks for a 24 depot, 144 drone scenario over $N = 100$ trials. Some fraction of events are always expected to be unsatisfied, as events are continuously injected into the environment. Still, we see that using the BLP planner results in shorter satisfaction times for events compared to the greedy and non-sharing planner, with around 70% of satisfied tasks being serviced in 10 or less timesteps. Further, the BLP planner has less satisfied tasks with large satisfaction times compared to the other planners. The obvious result of this faster satisfaction time is that the BLP planner has less unsatisfied tasks compared to the other methods. These results show that better sharing between depots results in lower satisfaction times and thus more tasks serviced.



(a) Static Demand, 5 injected events per timestep (b) Static Demand, 10 injected events per timestep (c) Slow-moving dynamic demand, 10 injected events per timestep (d) Fast-moving dynamic demand, 10 injected events per timestep

Fig. 4: Simulation Results for $N = 100$ randomized trials with 24 depots and varying drone numbers and settings. X-axis denotes total number of drones in the system, Y-axis denotes final fulfillment percentage with constant event injection. 4a shows static demand with 5 injected events per timestep. 4b shows static demand with 10 injected events per timestep. 4c and 4d show slow and fast dynamic demands with 10 events injected per timestep, respectively.

Figure 4a shows the final percent satisfaction for $N = 100$ trials with varying numbers of drones with 5 events injected per timestep. Not sharing drones results in much worse performance, as some depots are overwhelmed with events. For smaller numbers of drones, more optimal sharing becomes very important. For example, with 48 drones the BLP planner outperforms the greedy planner by as much as 12.6% on average. As drone number increases, more optimal sharing matters less. This can be seen in Figure 4b, which has 10 events injected per timestep to better capture fulfillment ability for larger drone numbers. The BLP planner outperforms the greedy planner by 9.7% with 96 drones, but only by 5.6% with 144 drones. Still, observing the interquartile ranges in Figures 4a and 4b show that the BLP method has more consistent task fulfillment in all cases.

C. Local Sharing for multiple depots with dynamic demand

We run a similar simulation as Section IV-B using dynamic demand, meaning that the underlying spatial distribution of events changes over time. We change the underlying field $\phi(q)$ to be time-varying as $\phi(q, t)$, and add a randomized velocity to the Gaussians in our GMM. These Gaussians move through the environment at their own velocity, only changing direction when they encounter the border of the environment. We maintain the same parameters as Section IV-B for our depots and drones, and run $N = 100$ trials. We assume that depots update their demand as the field changes over time, and therefore continue to share drones as demand changes. Figures 4c and 4d show our results for fields with slow and fast speeds in their dynamic demand, respectively.

We see in Figure 4c that our expected demand model can still be useful for systems with slow dynamic changes in dynamics. The BLP planner performs approximately the same, while the greedy sharing planner performs better than in the static case. Meanwhile, the non-sharing planner performs slightly better than in the static case. We attribute this to the changes in our density field spreading events out between depots. As a result, there is less concentration of events in depot partitions, leading to less scenarios where a single depot is overwhelmed with events. This is further exemplified in Figure 4d, where the fast changing dynamic field results in the greedy and non-sharing planner performing better on average than in the static cases in Figure 4b.

Figure 4d shows the greedy planner outperforms the BLP planner with fast changing dynamics. We attribute this to two distinct reasons. First, the fast changing dynamics means there is significant model mismatch between the actual demand and expected demand of any depot. Since the BLP planner only uses the current expected demand at the timestep it determines its assignments, this leads to suboptimal sharing choices. Second, the BLP planner is more sensitive to sharing drones compared to our chosen greedy planner. Thus, more drones are in transit between depots instead of servicing events due to the fast demand changes. The greedy planner only shares when it has a local drone utility surplus, meaning there is an inherent lag between sharing drones between depots when demand changes. This lag gives time for depots to fulfill events which appeared while the demand was moving through the depot's partition, thus increasing the greedy planner's performance. In contrast, the BLP planner shares drones when the overall local cost is lowered. This leads to the BLP planner sharing much more on average. For example, in the 24 depot, 144 drone scenario with fast dynamic demand with $N = 100$ trials, the average and standard deviation of drone swaps between depots is 1127.1 ± 199.3 and 582.6 ± 125.8 for the BLP planner and greedy planner, respectively. While these extra swaps aid in reallocation for slowly-evolving demand, we suspect that it was detrimental in for rapidly-evolving demand. This remains an open phenomena to investigate in future work.

V. Conclusions and Future Work

In this work, we introduce a new model for sharing heterogeneous drones between static depots for delivery tasks. We derive an iterative neighborhood search method based on satisfying a necessary condition of optimality. We show that using this method for decentralized local rebalancing allows for better discrete event task fulfillment compared to non-sharing and greedy methods.

Future work will focus on addressing model mismatch for dynamic demand. This can be done using both a future time horizon for expected demand, as well as a corrective demand to fulfill already existing tasks. Using a corrective demand is also beneficial for dealing with unknown or error-prone density fields, as depots can learn the true underlying field over time. Adapting the model to other more complex tasks such as information gathering using learning methods is another interesting research avenue.

REFERENCES

[1] B. P. Gerkey and M. J. Matarić, "A Formal Analysis and Taxonomy of Task Allocation in Multi-Robot Systems," *The International Journal of Robotics Research*, vol. 23, no. 9, pp. 939–954, Sep. 2004, publisher: SAGE Publications Ltd STM. [Online]. Available: https://doi.org/10.1177/0278364904045564

[2] G. A. Korsah, A. Stentz, and M. B. Dias, "A comprehensive taxonomy for multi-robot task allocation," *The International Journal of Robotics Research*, vol. 32, no. 12, pp. 1495–1512, Oct. 2013. [Online]. Available: http://journals.sagepub.com/doi/10.1177/0278364913496484

[3] A. Khamis, A. Hussein, and A. Elmogy, "Multi-robot Task Allocation: A Review of the State-of-the-Art," in *Cooperative Robots and Sensor Networks 2015*, A. Koubâa and J. Martínez-de Dios, Eds. Cham: Springer International Publishing, 2015, pp. 31–51. [Online]. Available: https://doi.org/10.1007/978-3-319-18299-5_2

[4] H. Chakraa, F. Guérin, E. Leclercq, and D. Lefebvre, "Optimization techniques for Multi-Robot Task Allocation problems: Review on the state-of-the-art," *Robotics and Autonomous Systems*, vol. 168, p. 104492, Oct. 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0921889023001318

[5] M. Rappaport and C. Bettstetter, "Coordinated recharging of mobile robots during exploration," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sep. 2017, pp. 6809–6816, iSSN: 2153-0866. [Online]. Available: https://ieeexplore.ieee.org/document/8206600

[6] J. Banfi, A. Quattrini Li, I. Rekleitis, F. Amigoni, and N. Basilico, "Strategies for coordinated multirobot exploration with recurrent connectivity constraints," *Autonomous Robots*, vol. 42, no. 4, pp. 875–894, Apr. 2018. [Online]. Available: https://doi.org/10.1007/s10514-017-9652-y

[7] A. Moortgat-Pick, M. Schwahn, A. Adamczyk, D. A. Duecker, and S. Haddadin, "Autonomous UAV Mission Cycling: A Mobile Hub Approach for Precise Landings and Continuous Operations in Challenging Environments," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*, May 2024, pp. 8450–8456. [Online]. Available: https://ieeexplore.ieee.org/document/10611292

[8] A. Moortgat-Pick, A. Adamczyk, D. A. Duecker, and S. Haddadin, "SVan: A Mobile Hub as a Field Robotics Development and Deployment Platform," May 2024, arXiv:2405.03890 [cs] version: 1. [Online]. Available: http://arxiv.org/abs/2405.03890

[9] A. Asadi, S. Nurre Pinkley, and M. Mes, "A Markov decision process approach for managing medical drone deliveries," *Expert Systems with Applications*, vol. 204, p. 117490, Oct. 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0957417422008193

[10] J. Dhote and S. Limbourg, "Designing unmanned aerial vehicle networks for biological material transportation – The case of Brussels," *Computers & Industrial Engineering*, vol. 148, p. 106652, Oct. 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0360835220303867

[11] N. Mathew, S. L. Smith, and S. L. Waslander, "Planning Paths for Package Delivery in Heterogeneous Multirobot Teams," *IEEE Transactions on Automation Science and Engineering*, vol. 12, no. 4, pp. 1298–1308, Oct. 2015, conference Name: IEEE Transactions on Automation Science and Engineering. [Online]. Available: https://ieeexplore.ieee.org/document/7194856

[12] M. S. Mondal, S. Ramasamy, J. D. Humann, J.-P. F. Reddinger, J. M. Dotterweich, M. A. Childers, and P. Bhounsule, "Optimizing Fuel-Constrained UAV-UGV Routes for Large Scale Coverage: Bilevel Planning in Heterogeneous Multi-Agent Systems," in *2023 International Symposium on Multi-Robot and Multi-Agent Systems (MRS)*, Dec. 2023, pp. 114–120. [Online]. Available: https://ieeexplore.ieee.org/document/10416777

[13] A. Otto, N. Agatz, J. Campbell, B. Golden, and E. Pesch, "Optimization approaches for civil applications of unmanned aerial vehicles (UAVs) or aerial drones: A survey," *Networks*, vol. 72, Mar. 2018.

[14] A. Prorok, M. A. Hsieh, and V. Kumar, "Fast Redistribution of a Swarm of Heterogeneous Robots," in *Proceedings of the 9th EAI International Conference on Bio-inspired Information and Communications Technologies (formerly BIONETICS)*. New York City, United States: ACM, 2016. [Online]. Available: http://eudl.eu/doi/10.4108/eai.3-12-2015.2262349

[15] A. Rjeb, J.-P. Gayon, and S. Norre, "Sizing of a heterogeneous fleet of robots in a logistics warehouse," in *2021 IEEE 17th International Conference on Automation Science and Engineering (CASE)*, Aug. 2021, pp. 95–100, iSSN: 2161-8089. [Online]. Available: https://ieeexplore.ieee.org/document/9551422

[16] N. Wilde and J. Alonso-Mora, "Designing Heterogeneous Robot Fleets for Task Allocation and Sequencing," in *2023 International Symposium on Multi-Robot and Multi-Agent Systems (MRS)*, Dec. 2023, pp. 156–162. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/10416795

[17] T. C. Service and J. A. Adams, "Coalition formation for task allocation: theory and algorithms," *Autonomous Agents and Multi-Agent Systems*, vol. 22, no. 2, pp. 225–248, Mar. 2011. [Online]. Available: https://doi.org/10.1007/s10458-010-9123-8

[18] K. Liang and C.-I. Vasile, "Distributed Fair Assignment and Rebalancing for Mobility-on-Demand Systems via an Auction-based Method," *2023 International Symposium on Multi-Robot and Multi-Agent Systems (MRS)*, pp. 128–134, Dec. 2023, conference Name: 2023 International Symposium on Multi-Robot and Multi-Agent Systems (MRS) ISBN: 9798350370768 Place: Boston, MA, USA Publisher: IEEE. [Online]. Available: https://ieeexplore.ieee.org/document/10416781/

[19] G. Diehl and J. A. Adams, "Task Elimination: Faster Coalition Formation for Overtasked Collectives," in *2023 International Symposium on Multi-Robot and Multi-Agent Systems (MRS)*, Dec. 2023, pp. 170–176. [Online]. Available: https://ieeexplore.ieee.org/document/10416770

[20] A. Rao, G. Sartoretti, and H. Choset, "Learning Heterogeneous Multi-Agent Allocations for Ergodic Search," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*, May 2024, pp. 12 345–12 352. [Online]. Available: https://ieeexplore.ieee.org/document/10611297

[21] W. Gosrich, S. Mayya, R. Li, J. Paulos, M. Yim, A. Ribeiro, and V. Kumar, "Coverage Control in Multi-Robot Systems via Graph Neural Networks," in *2022 International Conference on Robotics and Automation (ICRA)*, May 2022, pp. 8787–8793. [Online]. Available: https://ieeexplore.ieee.org/document/9811854

[22] M. Santos, Y. Diaz-Mercado, and M. Egerstedt, "Coverage Control for Multirobot Teams With Heterogeneous Sensing Capabilities," *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 919–925, Apr. 2018, conference Name: IEEE Robotics and Automation Letters. [Online]. Available: https://ieeexplore.ieee.org/document/8255576

[23] M. Coffey and A. Pierson, "Assessing Reputation to Improve Team Performance in Heterogeneous Multi-Robot Coverage," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*, May 2024, pp. 2571–2577. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/10611134