2020 IEEE International Conference on Robotics and Automation (ICRA)
31 May - 31 August, 2020. Paris, France

Efficient Large-Scale Multi-Drone Delivery Using Transit Networks

Shushman Choudhury, Kiril Solovey, Mykel J. Kochenderfer, and Marco Pavone

***Abstract*— We consider the problem of controlling a large fleet of drones to deliver packages simultaneously across broad urban areas. To conserve energy, drones hop between public transit vehicles (e.g., buses and trams). We design a comprehensive algorithmic framework that strives to minimize the maximum time to complete any delivery. We address the multifaceted complexity of the problem through a two-layer approach. First, the upper layer assigns drones to package delivery sequences with a near-optimal polynomial-time task allocation algorithm. Then, the lower layer executes the allocation by periodically routing the fleet over the transit network while employing efficient bounded-suboptimal multi-agent pathfinding techniques tailored to our setting. Experiments demonstrate the efficiency of our approach on settings with up to 200 drones, 5000 packages, and transit networks with up to 8000 stops in San Francisco and Washington DC. Our results show that the framework computes solutions within a few seconds (up to 2 minutes at most) on commodity hardware, and that drones travel up to 450% of their flight range with public transit.**

I. Introduction

Rapidly growing e-commerce demands have greatly strained dense urban communities by increasing delivery truck traffic and slowing operations and impacting travel times for public and private vehicles [23, 25]. Further congestion is being induced by newer services relying on ride-sharing vehicles. There is a clear need to redesign the current method of package distribution in cities [28]. The agility and aerial reach of drones, the flexibility and ease of establishing drone networks, and recent advances in drone capabilities make them highly promising for logistics networks [27]. However, drones have limited travel range and carrying capacity [14, 43]. On the other hand, ground-based transit networks have less flexibility but greater coverage and throughput. By combining the strengths of both, we can achieve significant commercial benefits and social impact (e.g., reducing ground congestion and delivering essentials).

We address the problem of operating a large number of drones to deliver multiple packages simultaneously in an area. The drones can use one or more vehicles in a public-transit network as modes of transportation, thereby saving their limited battery energy stored onboard and increasing their effective travel range. We are required to decide which deliveries each drone should make and in what order, which modes of transit to use, and for what duration (Figure 1).

Our approach must contend with the multiple significant challenges of our problem. It must plan over large time-dependent transit networks, while accounting for energy constraints that limit the drones' flight ranges. It must avoid inter-drone conflicts, such as where more than one drone attempts to board the same vehicle at the same time, or when the maximum carrying capacity of a vehicle is exceeded.

The authors are with Stanford University, CA, USA.

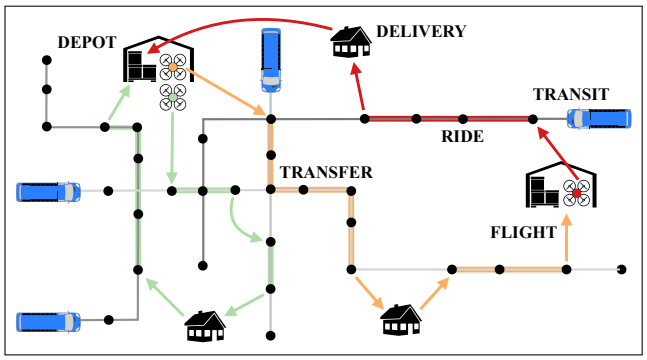


Fig. 1: Our multi-drone delivery framework plans for drones to piggyback on public transit vehicles while delivering packages from depots to the requested locations. Our framework is scalable and efficient, and minimizes the amount of time for any individual delivery.

We seek not just feasible multi-agent plans but high-quality solutions in terms of a cumulative objective over all drones, the makespan, i.e., the maximum individual delivery time for any drone. Additionally, our approach must also solve the task allocation problem of determining which drones deliver which packages, and from which distribution centers.

A. Related work

Some individual aspects of our problem have already been studied. Choudhury et al. [10] investigated the single-agent setting of controlling a drone to use multiple modes of transit en route to its destination. Recent work has considered pairing a drone with a delivery truck, which does not exploit public transit [2, 18, 34]. The multi-agent issues of task allocation and inter-agent conflicts were not addressed either. Our problem is closely related to routing a fleet of autonomous vehicles providing mobility-on-demand services [26, 42, 45]. Specifically, the task is to compute routes for the vehicles (both customer-carrying and empty) so that travel demand is fulfilled and operational cost is minimized. In particular, recent works study the combination of such service with public transit, where passengers can use several modes of transportation in the same trip [38, 48]. However, such works abstract away inter-agent constraints or dynamics and are not suited for autonomous pathfinding. The task-allocation setting we consider in our problem can be viewed as an instance of the vehicle routing problem [8, 35, 44], variants of which are typically solved by mixed integer linear programming (MILP) formulations that scale poorly, or by heuristics without optimality guarantees.

We must contend with the challenges of planning for multiple agents. Accordingly, the second layer of our approach is a multi-agent path finding (MAPF) problem [16, 46]. Since the drones are on the same team, we have a centralized or cooperative pathfinding setting [40]. The MAPF problem is NP-hard to solve optimally [47]. A number

of efficient solvers have been developed that work well in practice [17]. The MAPF formulation and algorithms have been extended to several relevant scenarios such as lifelong pickup-and-delivery [31] and joint task assignment and pathfinding [24, 30], though for different task settings and constraints than ours. Also, a MAPF formulation was applied for UAV traffic management in cities [22]. However, none of the approaches considered pathfinding over large time-dependent transit networks. We use models, algorithms and techniques from transportation planning [6, 13, 37].

B. Statement of contributions

We present a comprehensive algorithmic framework for large-scale multi-drone delivery in synergy with a ground transit network. Our approach strives to minimize the maximum time to complete any delivery. We decompose the highly challenging problem and solve it stage-wise with a two-layer approach. First, the upper layer assigns drones to package-delivery sequences with a task allocation algorithm. Then, the lower layer executes the allocation by periodically routing the fleet over the transit network.

Algorithmically, we develop a new delivery sequence allocation method for the upper layer that obtains a near-optimal solution in polynomial runtime. For the lower layer, we extend techniques for multi-agent path finding that account for time-dependent transit networks and agent energy constraints to perform multi-drone routing. Experimentally, we present results supporting the efficiency of our approach on settings with up to 200 drones, 5000 packages, and transit networks of up to 8000 stops in San Francisco and the Washington DC area. Our framework can compute solutions within a few seconds (up to 2 minutes for the largest settings) on commodity hardware, and in our problem scenarios, drones can travel up to 450% of their flight range using transit.

The following is the paper structure. We present an overall description of the two-layer approach in Section II, and then elaborate on each layer in Sections III and IV. We present experimental results on simulations in Section V, and conclude the paper with Section VI. We will also refer to the appendix of the extended version [11] for additional details, illustrations, results, and discussions.

II. Methodology

We provide a high-level description of our formulation and approach to illustrate the various interacting components.

A. Problem Formulation

We are operating a centralized homogeneous fleet of m drones within a city-scale domain. There are ℓ product *depots* with known geographic locations, denoted by $V_D := \{d_1, \ldots, d_\ell\} \subset \mathbb{R}^2$. The depots are both product dispatch centers and drone-charging stations. At the start of a large time interval (e.g., a day), a batch of delivery request locations for k different *packages*, denoted $V_P := \{p_1, \ldots, p_k\} \subset \mathbb{R}^2$, is received (we assume that $k \gg m$). We assume that any package can be dispatched from any depot; our framework exploits this property to optimize the solution quality in terms of *makespan*, i.e., the maximum execution time for any delivery. In Section III, we mention how our approach can accommodate dispatch constraints.

The drones carry packages from depots to delivery locations. They can extend their effective travel range by using public transit vehicles in the area, which remain unaffected by the drones' actions. Our problem is to route drones to deliver all packages while minimizing makespan. A drone route consists of its current location and the sequence of depot and package locations to visit with a combination of flying and riding on transit. We characterize the drones' limited energy as a maximum flight distance constraint. A feasible solution must satisfy inter-drone constraints such as collision avoidance and transit vehicle capacity limits.

Finally, we make some assumptions for our setting: a drone carries one package at a time, which is reasonable given state-of-the-art drone payloads [14]; drones are recharged upon visiting a depot in negligible time (e.g., a battery replacement); depots have unlimited drone capacity; the transit network is deterministic with respect to locations and vehicle travel times (we mention uncertainty in Section VI). We do account for the time-varying nature of the transit.

B. Approach overview

In principle, we could frame the entire problem as a mixed integer linear program (MILP). However, for real-world problems (hundreds of drones; thousands of packages; large transit networks), even state-of-the-art MILP approaches are unlikely to scale. Moreover, even a simpler problem that ignores the interaction constraints is an instance of the notoriously challenging multi-depot vehicle routing problem [35]. Thus, we decouple the problem into two distinct subproblems that we solve stage-wise in layers.

The upper layer performs *task allocation* to determine which packages are delivered by which drone and in what order. It takes as input the known depot and package locations, and an estimate of the drone travel time between every pair of locations. It then solves a threefold allocation to minimize delivery makespan and assigns to each package (i) the dispatch depot and (ii) the delivery drone, and to each drone (iii) the order of package deliveries. To this end, we develop an efficient polynomial-time task-allocation algorithm that achieves a near-optimal makespan.

The lower layer performs *route planning* for the drone fleet to execute the allocated delivery tasks. It generates detailed routes of drone locations in time and space and the transit vehicles used, while accounting for the time-varying transit network. It also ensures that (i) simultaneous transit boarding by multiple drones is avoided, (ii) no transit vehicle exceeds its drone-carrying capacity, and (iii) drone (battery) energy constraints are respected. We efficiently handle individual and inter-drone constraints by framing the routing problem as an extension of multi-agent path finding (MAPF) to transit networks. We adapt a scalable, bounded sub-optimal variant of a highly effective MAPF solver called Conflict-Based Search (CBS) [39] to solve the one-delivery-per-drone problem. Finally, we obtain routes for the sequence of deliveries in a receding-horizon fashion by replanning for the next task once a drone completes its current one.

Decomposition-based stage-wise optimization approaches typically have an approximation gap compared to the optimal solution of the full problem. For us, this gap manifests in the surrogate cost estimate we use for the drone's travel time in the task-allocation layer (instead of jointly solving for allocation and multi-agent routing over transit networks, which is not feasible at scale). The better the surrogate, the more *coupled* the layers are, i.e., the better is the solution of the first stage for the second one. Such surrogates have a tradeoff between efficiency and approximation quality. An easy-to-compute travel time surrogate, for instance, is the drone's direct flight time between two locations (ignoring transit). However, that can be poor-quality when the drone requires transit for an out-of-range target. We use a surrogate that actually accounts for the transit network, at the expense of some modest preprocessing. We defer details to the appendix, but the idea is to precompute the pairwise shortest travel times between locations spread around the city, over a representative snapshot of the transit network.

III. Task Assignment and Package Allocation

We leverage our problem's structure to design a new algorithm called MERGESPLITTOURS for the task-allocation layer, which guarantees a near-optimal solution in polynomial time. The goal of this layer is to (i) distribute the set of packages V_P among m agents, (ii) assign each package destination $p \in V_P$ to a depot $d \in V_D$, and (iii) assign drones to a sequence of depot pickups and package deliveries. The objective is to **minimize the maximum travel time among all agents** over all three of the above components.

Our problem can be cast as a special version of the m traveling salesman problem [7], which we call the m *minimal visiting paths problem* (m-MVP). We seek a set of m paths such that the makespan, i.e., the maximum travel time for any path, is minimized. We only need *paths* that start and end at (the same or different) depots, not tours. Our formulation is a special case of the the *asymmetric* variant, for a *directed* underlying graph, which is NP-hard even for $m = 1$ on general graphs [4] (although it is not known whether the specific instance of our problem is NP-hard as well). Moreover, the current best polynomial-time approximation [4] yields the fairly large approximation factor $O(\log n / \log\log n)$, for a graph with n vertices. An additional challenge is the inability to assume the triangle inequality on our objective of travel times.

A key element of m-MVP is the *allocation graph* $G_A = (V_A, E_A)$, with vertex set $V_A = V_D \cup V_P$. Each directed edge $(u, v) \in E_A$ is weighted according to an estimated travel time c_{uv} from the location of u to that of v in the city. For every $d \in V_D, p \in V_P$ we exclude the edge (d, p) from E_A if it is impossible to reach p from d while using at most $1/2$ of the flight range allowed (similarly for (p, d) edges). As we flagged in Section II-B, any dispatch constraints are modeled by excluding edges from the corresponding depot. We are now ready for the full definition of m-MVP:

Definition 1. Given allocation graph G_A, the m minimal visiting paths problem (m-MVP) consists of finding m paths $P^*_{1:m}$ on G_A, such that (1) each path P^*_i starts at some depot $d \in V_D$ and terminates at the same or different $d' \in V_D$, (2) exactly one path visits each package $p \in V_P$, and (3) the maximum travel time of any of the paths is minimized.

Algorithm 1: MERGESPLITTOURS(G_A)

Solve MCT(G_A) to get t tours $\mathbb{T} := \{T_1, \ldots, T_t\}$;
while $|\mathbb{T}| > 1$ **do**
  Pick distinct tours $T, T' \in \mathbb{T}$ and depots $d \in T, d' \in T'$ that minimize $c_{dd'} + c_{d'd}$;
  Merge T, T' by adding $(d, d'), (d', d)$ edges ;
Split final tour T into m paths $P_1, \ldots, P_m$, where LENGTH(P_i) is proportional to LENGTH(T)/m for each i (similar to [19]);
Extend each P_i to ensure it begins and ends at a depot;
return $P_1, \ldots, P_m$;

TABLE I: An integer programming formulation of the MCT problem.

Given allocation graph $G_A = (V_A, E_A)$, with $V_A = V_D \cup V_P$,

$$\text{minimize} \sum_{(u,v)\in E_A} x_{uv} \cdot c_{uv} \tag{1}$$

subject to

$$x_{uv} \in \{0,1\}, \quad \forall (u,v) \in E_A, u \in V_P \vee v \in V_P, \tag{2}$$

$$x_{uv} \in \mathbb{N}_{\geqslant 0}, \quad \forall (d,d') \in E_A, d, d' \in V_D, \tag{3}$$

$$\sum_{d\in N_+(p)} x_{dp} = \sum_{d\in N_-(p)} x_{pd} = 1, \quad \forall p \in V_P, \tag{4}$$

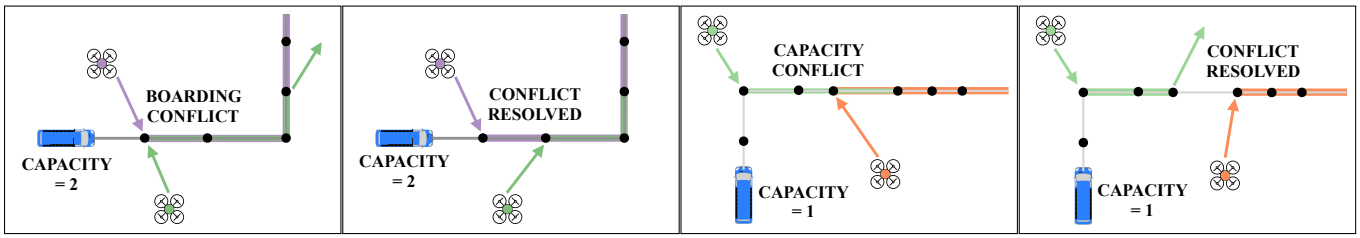
$$\sum_{v\in N_+(d)} x_{vd} - \sum_{v\in N_-(d)} x_{dv} = 0, \quad \forall d \in V_D. \tag{5}$$

where $N_+(v), N_-(v)$ denote the in and out going neighbors of $v \in V_A$.

Let OPT be the optimal makespan, i.e., OPT := $\max_{i\in[m]}$ LENGTH(P^*_i), where LENGTH($\cdot$) denotes the total travel time along a given path or tour. We make three observations. First, if a path contains the sub-path $(d, p), (p, d')$, for some $d, d' \in V_D, p \in V_P$, then p should be dispatched from depot d and the drone delivering p will return to d' after delivery. Second, a package p being found in P^*_i indicates that drone $i \in [m]$ should deliver it. Third, P^*_i fully characterizes the order of packages delivered by drone i.

A. Algorithm Overview

We present our MERGESPLITTOURS algorithm for solving m-MVP (Algorithm 1); see a detailed description in the appendix. A key step is generating an initial set of tours $\mathbb{T}$ by solving the minimal-connecting tours (MCT) problem (see Table I), which attempts to connect packages to depots within tours to minimize the total edge weight in eq. (1). The constraint in eq. (4) is that each package is connected to precisely one incoming and one outgoing edge from and to depots respectively. The final constraint in eq. (5) enforces inflow and outflow equality for every depot. Edges connecting packages can be used at most once, whereas edges connecting depots can be used multiple times. The solution to MCT is the assignment $\{x_{uv}\}_{(u,v)\in E_A}$, i.e., which edges of G_A are used and how many times. This assignment implicitly represents the desired collection of the tours $T_1, \ldots, T_t$; see the appendix.



(a) (b) (c) (d)

Fig. 2: In our formulation of multi-agent path finding with transit networks, conflicts arise from the violation of shared inter-drone constraints: (a) boarding conflicts between two or more drones and (c) capacity conflicts between more drones than the transit vehicle can accommodate. The modified paths after resolving the corresponding conflicts are depicted in (b) and (d), respectively.

B. Theoretical Guarantees

All proofs from this secion are in the appendix. The following theorem states that MERGESPLITTOURS is correct and that its makespan is close to optimal.

Theorem 1. *Suppose G_A is strongly connected and the subgraph $G_A(V_D)$ induced by the vertices V_D is a directed clique. Let $P_1, \ldots, P_m$ be the output of* MERGESPLITTOURS. *Then, every package $p \in V_P$ is contained in exactly one path P_i, and every P_i starts and ends at a depot. Moreover, $\max_{i\in[m]} \text{LENGTH}(P_i) \leqslant \text{OPT} + \alpha + \beta$ holds,*

where $\alpha := \max_{d,d' \in V_D} c_{dd'} + c_{d'd}$, $\beta := \max_{d,d' \in V_D, p \in V_P} c_{dp} + c_{pd'}$.

The key idea is that the total cost of the tours induced by the solution to MCT cannot exceed the total length of $\{P_1^*, \ldots, P_m^*\}$. The MCT solution is then adapted to m paths with an additional overhead of $\alpha + \beta$ per path. When $m \ll |V_P|$ (typically the case), α and β are small compared to OPT, making the bound tight. For instance, in our randomly-generated scenarios in Section V-A, for $m = 5$ and $k = 200$, the approximation ratio $\max_{i\in[m]} \text{LENGTH}(P_i)/\text{OPT} = 1.09$, and for $m = 10, k = 500$, the factor is 1.06.

The computational bottleneck of the algorithm is MCT, while the other components can clearly be implemented polynomially in the input size. However, it suffices to solve a relaxed version of MCT to obtain the same integral solution.

Lemma 1. *The optimal solution to the fractional relaxation of MCT, in which $x_{uv} \in [0, 1]$ for all $u \in V_P \vee v \in V_P$, and $x_{uv} \in \mathbb{R}_+$ otherwise, yields the integer optimal solution.*

The lemma follows from casting MCT as the minimum-cost circulation problem, for which the constraint matrix is totally unimodular [3]. Therefore, MERGESPLITTOURS can be implemented in polynomial time.

IV. MULTI-AGENT PATH FINDING

For each drone $i \in [m]$, the allocation layer yields a sequence of delivery tasks $d_1 p_1 \ldots p_l d_{l+1}$. Each delivery sequence has one or more subsequences of dpd'. The route-planning layer treats each dpd' subsequence *as an individual drone task*, i.e., leaving with the package from depot d, carrying it to package location p and returning to the (same or different) depot d', without exceeding the energy capacity. We seek an efficient and scalable method to obtain high-quality (with respect to travel time) feasible paths, while using transit options to extend range, for m different drone dpd' tasks simultaneously. The full set of delivery sequences can be satisfied by replanning when a drone finishes its current task and begins a new one; we discuss and compare two replanning strategies in the appendix. Thus, we formulate the problem of multi-drone routing to satisfy a set of delivery sequences as receding-horizon multi-agent path finding (MAPF) over transit networks. In this section, we describe the graph representation of our problem and present an efficient bounded sub-optimal algorithm.

A. MAPF with Transit Networks (MAPF-TN)

The problem of Multi-Agent Path Finding with Transit Networks (MAPF-TN) is the extension of standard MAPF to where agents can use one or more modes of transit in addition to moving. The incorporation of transit networks introduces additional challenges and underlying structure. The **input** to MAPF-TN is the set of m tasks $(d_i, p_i, d_i')_{i=1:m}$ and the directed operation graph $G_O = (V_O, E_O)$. In Section III, the allocation graph G_A only considered depots and packages, and edges between them. Here, G_O also includes transit vertices, $V_{TN} = \bigcup_{\tau\in\mathcal{T}} R_\tau$, where $\mathcal{T}$ is the set of trips, and each trip $R_\tau = \{(s_1, t_1) \ldots\}$ is a sequence of time-stamped stop locations (a given stop location may appear as several different nodes with distinct time-stamps). Similarly, we also use time-expanded versions of V_D and V_P [37].

The edges are defined as follows: An edge $e = (u, v) \in E$ is a *transit edge* if $u, v \in V_{TN}$ and are consecutive stops on the same trip R_t. Any other edge is a *flight edge*. An edge is *time-constrained* if $v \in V_{TN}$ and *time-unconstrained* otherwise. Every edge has three attributes: traversal time T, energy expended N, and capacity C. Since each vertex is associated with a location, $\|v - u\|$ denotes the distance between them for a suitable metric. MAPF typically abstracts away agent dynamics; we have a simple model where drones move at constant speed σ, and distance flown represents energy expended. Due to high graph density (drones can fly point-to-point between many stops), we do not explicitly enumerate edges but generate them on-the-fly during search.

We now define the three attributes for E_O. For time-constrained edges, $T(e) = v.t - u.t$ is the difference between corresponding time-stamps (if $u \in V_D \cup V_P$, $u.t$ is the chosen departure time), and for time-unconstrained edges, $T(e) = \|v - u\|/\sigma$ is the time of direct flight. For flight edges, $N(e) = \|v - u\|$ (flight distance), and for transit edges, $N(e) = 0$. For transit edges, $C(e)$ is bounded by the capacity of the vehicle, while for flight edges, $C(e) = \infty$. Here, we

assume that time-unconstrained flight in open space can be accommodated (thoroughly examined in [22]).

We now describe the remaining relevant MAPF-TN details. An individual path π_i for drone i from d_i through p_i to d_i' is **feasible** if the energy constraint $\sum_{e\in\pi_i} N(e) \leqslant \bar{N}$ is satisfied, where $\bar{N}$ is the drone's maximum flight distance. In addition, the drone should be able to traverse the distance of a time-constrained flight edge in time, i.e., $\sigma \times (v.t - u.t) > \|v - u\|$. For simplicity, we abstract away energy expenditure due to hovering in place by flying the drone at reduced speed to reach the transit just in time. Thus, the constraint $\bar{N}$ is only on the traversed distance. The cost of an individual path is the total traversal time, $T(\pi_i) = \sum_{e\in\pi_i} T(e)$. A **feasible solution** $\Pi = \bigcup_{i=1:m} \pi_i$ is a set of m individually feasible paths that does not violate any of the following two *shared constraints* (see Figure 2): (i) *Boarding constraint*, i.e., no two drones may board the same vehicle at the same stop; (ii) *Capacity constraint*, i.e., a transit edge e may not be used by more than $C(e)$ drones. As with the allocation layer, the **global objective** for MAPF-TN is to minimize the solution makespan, $\operatorname{argmin}_{\Pi} \max_{\pi\in\Pi} T(\pi)$, i.e., minimize the worst individual completion time.

B. Conflict-Based Search for MAPF-TN

To tackle MAPF-TN, we modify the Conflict-Based Search (CBS) algorithm [39]. The multi-agent level of CBS identifies shared constraints and imposes corresponding path constraints on the single-agent level. The single-agent level computes optimal individual paths that respect all constraints. If individual paths conflict (i.e., violate a shared constraint), the multi-agent level adds further constraints to resolve the conflict, and invokes the single-agent level again, for the conflicting agents. In MAPF-TN, conflicts arise from boarding and capacity constraints. CBS obtains optimal multi-agent solutions without having to run (potentially significantly expensive) multi-agent searches. However, its performance can degrade heavily with many conflicts in which constraints are violated. Figure 2 illustrates the generation and resolution of conflicts in our MAPF-TN problem.

For scalability, we use a bounded sub-optimal variant of CBS called *Enhanced CBS* (ECBS), which achieves orders of magnitude speedups over CBS [5]. ECBS uses bounded sub-optimal *Focal Search* [36] at both levels, instead of best-first A* [21]. Focal search allows using an inadmissible heuristic that prioritizes efficiency. We now describe a crucial modification to ECBS required for MAPF-TN.

Focal Weight-constrained Search: Unlike typical MAPF, the low-level graph search in MAPF-TN has a path-wide constraint (traversal distance) *in addition to the objective function* of traversal time. For the shortest path problems on graphs, adding a path-wide constraint makes it NP-hard [20]. Several algorithms for constrained search require an explicit enumeration of the edges [9, 15]. We extend the A* for MultiConstraint Shortest Path (A*-MCSP) algorithm [29] (suitable for our implicit graph) to focal search (called Focal-MCSP). Focal-MCSP uses admissible heuristics on both objective and constraint and maintains only non-dominated paths to intermediate nodes. This extensive book-keeping requires a careful implementation for efficiency.

TABLE II: The mean computation time for MERGESPLITTOURS in seconds, over 100 different trials for each setting. MERGESPLITTOURS is polynomial in input size and highly scalable. Here, $k = |V_P|$ is the number of package deliveries and $\ell = |V_D|$ is the number of depots. The Out-of-Memory cases are due to the linear programming step of MCT and could be resolved in practice with a larger machine or a distributed implementation.

k	$\ell = 2$	$\ell = 5$	$\ell = 10$	$\ell = 20$	$\ell = 30$
50	0.006	0.022	0.074	0.326	0.981
100	0.016	0.070	0.268	1.274	2.823
200	0.053	0.272	1.171	4.323	11.09
500	0.311	1.731	6.532	31.15	83.31
1000	1.483	6.811	31.08	OutOfMem	OutOfMem
5000	38.05	OutOfMem	OutOfMem	OutOfMem	OutOfMem

Focal-MCSP inherits the properties of A*-MCSP and Focal Search; therefore, it yields a bounded-suboptimal feasible path to the target. Accordingly, **ECBS with Focal-MCSP yields a bounded sub-optimal solution to MAPF-TN**. The result follows from the analysis of ECBS [5]. Also, note that a dpd' path requires a bounded sub-optimal path from d to p and another from p to d', such that their concatenation is feasible. Since this is even more complicated, in practice, we run Focal-MCSP twice (from d to p and p to d') with half the energy constraint each time and concatenate the paths, guaranteeing feasibility. In the appendix we discuss other required modifications to standard MAPF and important speedup techniques that nonetheless retain the bounded sub-optimality of Enhanced CBS for our MAPF-TN formulation.

V. EXPERIMENTS AND RESULTS

We implemented our approach using the Julia language and tested it on a machine with a 6-core 3.7 GHz 16 GiB RAM CPU.[1] For very large combinatorial optimization problems, solution quality and algorithm efficiency are of interest. We have already shown that the upper and lower layers are near-optimal and bounded-suboptimal respectively in terms of solution quality, i.e., makespan. Therefore, for evaluation we focus on their efficiency and scalability to large real-world settings. We do not attempt to baseline against a MILP approach for the full problem; we estimate that a typical setting of interest will have on the order of 10^7 variables in a MILP formulation, besides exponential constraints.

We ran simulations with two large-scale public transit networks in San Francisco (SFMTA) and the Washington Metropolitan Area (WMATA). We used the open-source General Transit Feed Specification data [1] for each network. We considered only the bus network (by far the most extensive), but our formulation can accommodate multiple modes. We defined a geographical bounding box in each case, of area $150\,\mathrm{km}^2$ for SFMTA and $400\,\mathrm{km}^2$ for WMATA (illustrated in the appendix), within which depots and package locations were randomly generated. For the transit network, we considered all bus trips that operate within the bounding box. The *size* of the time-expanded network, $|V_{TN}|$, is the total number of stops made by all trips; $|V_{TN}| = 4192$ for SFMTA and $|V_{TN}| = 7608$ for WMATA (recall that edges are implicit, so $|E_{TN}|$ varies between queries, but the full graph G_O can be dense). The drone's flight range constraint

[1]The code for our work is available at `https://github.com/sisl/MultiAgentAllocationTransit.jl`.

TABLE III: (All times are in seconds) An extensive analysis of the MAPF-TN layer, on 20 trials for each $\{\ell, m\}$ setting, with different randomly generated depots and delivery locations for each trial. The integer carrying capacity of any transit edge $C(e)$ was randomly chosen from $\{3, 4, 5\}$ (single and double-buses). The sub-optimality factor for ECBS was 1.1. For settings with $m/\ell = 10$, a number of trials timed out (over 180 s) and were discarded.

	San Francisco ($\lvert V_{TN}\rvert = 4192$; Area 150 km^2)				**Washington DC** ($\lvert V_{TN}\rvert = 7608$; Area 400 km^2)			
{Depots, Agents} $\{\ell, m\}$	{Median, Avg} Plan Time	{Avg, Max} Range Ext.	{Avg, Max} Transit Used	Avg Soln. Makespan	{Median, Avg} Plan Time	{Avg, Max} Range Ext.	{Avg, Max} Transit Used	Avg Soln. Makespan
$\{5, 10\}$	$\{0.53, 1.07\}$	$\{1.5, 3.0\}$	$\{2.9, 6\}$	2464.8	$\{4.73, 7.17\}$	$\{2.0, 3.5\}$	$\{3.3, 8\}$	5006.5
$\{5, 20\}$	$\{1.11, 2.79\}$	$\{1.6, 2.5\}$	$\{3.1, 6\}$	2555.5	$\{11.5, 12.4\}$	$\{2.3, 3.8\}$	$\{4.5, 7\}$	5819.2
$\{5, 50\}$	$\{3.18, 8.29\}$	$\{1.7, 2.4\}$	$\{4.5, 6\}$	3485.8	$\{50.5, 57.6\}$	$\{2.9, 3.6\}$	$\{5.1, 7\}$	6861.7
$\{10, 20\}$	$\{0.64, 0.68\}$	$\{1.2, 1.8\}$	$\{2.3, 4\}$	2082.7	$\{7.31, 8.78\}$	$\{2.1, 3.2\}$	$\{3.7, 7\}$	5195.6
$\{10, 50\}$	$\{1.73, 2.96\}$	$\{1.6, 3.6\}$	$\{3.2, 5\}$	2671.1	$\{15.6, 19.8\}$	$\{2.5, 4.5\}$	$\{3.9, 6\}$	6316.7
$\{10, 100\}$	$\{2.09, 4.17\}$	$\{1.4, 1.8\}$	$\{3.7, 7\}$	3084.5	$\{35.9, 39.7\}$	$\{2.5, 3.7\}$	$\{4.9, 8\}$	6889.8
$\{20, 50\}$	$\{0.19, 0.26\}$	$\{0.9, 1.2\}$	$\{0.8, 4\}$	1054.8	$\{8.14, 11.2\}$	$\{1.9, 4.1\}$	$\{3.6, 7\}$	5086.9
$\{20, 100\}$	$\{0.41, 0.66\}$	$\{1.1, 1.3\}$	$\{1.4, 4\}$	1457.6	$\{17.5, 19.2\}$	$\{2.2, 3.7\}$	$\{4.1, 6\}$	5400.9
$\{20, 200\}$	$\{0.73, 1.70\}$	$\{1.1, 2.3\}$	$\{2.2, 5\}$	1824.4	$\{22.9, 26.2\}$	$\{2.2, 2.9\}$	$\{4.4, 6\}$	6050.1

is set (conservatively) to 7 km and average speed to 25 kph, based on the DJI Mavic 2 specifications [14]. In this section, we evaluate the two main components — the task allocation and multi-agent path finding layers. In the appendix we compare the performance of two replanning strategies for when a drone finishes its current delivery, and two surrogate travel time estimates for coupling the layers.

A. Task Allocation

The scale of the allocation problem is determined by the number of depots and packages, i.e., $\ell + k$. The runtimes for MERGESPLITTOURS with varying ℓ, k over SFMTA are displayed in Table II. The roughly quadratic increase in runtimes along a specific row or column demonstrate that our provably near-optimal MERGESPLITTOURS algorithm is indeed polynomial in the size of the input. Even for up to 5000 deliveries, the absolute runtimes are quite reasonable. We do not compare with naive MILP even for allocation, as the number of variables would exceed $\ell \cdot k$, in addition to the expensive subtour elimination constraints [32].

B. MAPF with Transit Networks (MAPF-TN)

Solving multi-agent path finding optimally is NP-hard [47]. Previous research has benchmarked CBS variants and shown that Enhanced CBS is most effective [5, 12]. Therefore, we focus on extensively evaluating our own approach rather than redundant baselining. Table III quantifies several aspects of the MAPF-TN layer with varying numbers of depots (ℓ) and agents (m), the two most tunable parameters. Before each trial, we run the allocation layer and collect m *dpd′* tasks, one for each agent. We then run the MAPF-TN solver on this set of tasks to compute a solution.

We discuss broad observations here and provide a detailed analysis in the appendix. The results are very promising; our approach scales to large numbers of agents (200) and large transit networks (nearly 8000 vertices); the highest average makespan for the true delivery time is less than an hour (3485.8 s) for SFMTA and 2 hours (6889.8 s) for WMATA; drones are using up to 8 transit options per route to extend their range by up to 4.5x. As we anticipated, **conflict resolution is a major bottleneck of MAPF-TN**. A *higher ratio of agents to depots* increases conflicts due to shared transit, thereby increasing plan time (compare $\{5, 20\}$ to $\{10, 20\}$). *A higher number of depots* puts more deliveries within flight range of a depot, reducing conflicts, makespan, and the need for transit usage and range extension (compare $\{10, 50\}$ to $\{20, 50\}$). Plan times are much higher for WMATA due to a larger area and a larger and less uniformly distributed bus network, leading to higher single-agent search times and more multi-agent conflicts. Trials taking more than 3 minutes were discarded; two pathological cases with SFMATA and WMATA (each with $\{l = 10, m = 100\}$) took nearly 4 and 8 minutes, due to 30 and 10 conflicts respectively. In any case, a deployed system would have better compute and parallelized implementations. Finally, note that the running times reported here are actually pessimistic, because we consider cases where drones are released simultaneously from the depots, which increases conflicts. However, a gradual release by executing the MAPF solver over a longer horizon (as we discuss in the appendix) results in fewer conflicts, allowing us to cope with an even larger drone fleet.

VI. CONCLUSION AND FUTURE WORK

We designed a comprehensive algorithmic framework for solving the highly challenging problem of multi-drone package delivery with routing over transit networks. Our two-layer approach is efficient and highly scalable to large problem settings and obtains high-quality solutions that satisfy the many system constraints. We ran extensive simulations with two real-world transit networks that demonstrated the widespread applicability of our framework and how using ground transit judiciously allows drones to significantly extend their effective range.

A key future direction is to perform case studies that estimate the operational cost of our framework, evaluate its impact on road congestion, and consider potential externalities like noise pollution and disparate impact on urban communities. Another direction is to extend our model to overcome its limitations: delays and uncertainty in the travel pattern of transit vehicles [33] and delivery time windows [41]; jointly routing ground vehicles and drones; optimizing for the placements of depots, whose locations are currently randomly generated and given as input.

ACKNOWLEDGMENTS

This work was supported in part by NSF, Award Number: 1830554, the Toyota Research Institute (TRI), and the Ford Motor Company. The authors thank Sarah Laaminach, Nicolas Lanzetti, Mauro Salazar, and Gioele Zardini for fruitful discussions on transit networks.

REFERENCES

[1] General Transit Feed Specification. URL https://developers.google.com/transit/gtfs/. Accessed: August 30, 2019.

[2] Niels Agatz, Paul Bouman, and Marie Schmidt. Optimization Approaches for the Traveling Salesman Problem with Drone. *Transportation Science*, 52(4):965–981, 2018.

[3] Ravindra K. Ahuja, Thomas L. Magnanti, and James B. Orlin. *Network Flows: Theory, Algorithms, and Applications*. Pearson, 1993.

[4] Arash Asadpour, Michel X. Goemans, Aleksander Madry, Shayan Oveis Gharan, and Amin Saberi. An $O(\log n/\log\log n)$-Approximation Algorithm for the Asymmetric Traveling Salesman Problem. *Operations Research*, 65(4):1043–1061, 2017.

[5] Max Barer, Guni Sharon, Roni Stern, and Ariel Felner. Suboptimal Variants of the Conflict-based Search Algorithm for the Multi-agent Pathfinding Problem. In *Symposium on Combinatorial Search*, 2014.

[6] Hannah Bast, Daniel Delling, Andrew Goldberg, Matthias Müller-Hannemann, Thomas Pajor, Peter Sanders, Dorothea Wagner, and Renato F Werneck. Route Planning in Transportation Networks. In *Algorithm Engineering*, pages 19–80. Springer, 2016.

[7] Tolga Bektas. The Multiple Traveling Salesman Problem: an Overview of Formulations and Solution Procedures. *Omega*, 34(3):209–219, 2006.

[8] Jose Caceres-Cruz, Pol Arias, Daniel Guimarans, Daniel Riera, and Angel A. Juan. Rich Vehicle Routing Problem: Survey. *ACM Comput. Surv.*, 47(2):32:1–32:28, 2014.

[9] W Matthew Carlyle, Johannes O Royset, and R Kevin Wood. Lagrangian Relaxation and Enumeration for Solving Constrained Shortest-Path Problems. *Networks*, 52(4):256–270, 2008.

[10] Shushman Choudhury, Jacob P. Knickerbocker, and Mykel J. Kochenderfer. Dynamic Real-time Multimodal Routing with Hierarchical Hybrid Planning. In *IEEE Intelligent Vehicles Symposium (IV)*, pages 2397–2404, 2019.

[11] Shushman Choudhury, Kiril Solovey, Mykel J Kochenderfer, and Marco Pavone. Efficient Large-Scale Multi-Drone Delivery Using Transit Networks. *arXiv preprint arXiv:1909.11840*, 2019.

[12] Liron Cohen, Tansel Uras, TK Satish Kumar, Hong Xu, Nora Ayanian, and Sven Koenig. Improved Solvers for Bounded-Suboptimal Multi-Agent Path Finding. In *International Joint Conference on Artificial Intelligence (IJCAI)*, pages 3067–3074, 2016.

[13] Daniel Delling, Peter Sanders, Dominik Schultes, and Dorothea Wagner. Engineering Route Planning Algorithms. In *Algorithmics of Large and Complex Networks*, pages 117–139. Springer-Verlag, 2009.

[14] DJI. DJI Mavic 2 Specifications Sheet. URL http://bit.ly/2mfCAvz.

[15] Irina Dumitrescu and Natashia Boland. Improved Preprocessing, Labeling and Scaling Algorithms for the Weight-Constrained Shortest Path Problem. *Networks: An International Journal*, 42(3):135–153, 2003.

[16] Michael Erdmann and Tomas Lozano-Perez. On Multiple Moving Objects. *Algorithmica*, 2(1-4):477, 1987.

[17] Ariel Felner, Roni Stern, Solomon Eyal Shimony, Eli Boyarski, Meir Goldenberg, Guni Sharon, Nathan Sturtevant, Glenn Wagner, and Pavel Surynek. Search-based Optimal Solvers for the Multi-Agent Pathfinding Problem: Summary and Challenges. In *Symposium on Combinatorial Search*, 2017.

[18] Sergio Mourelo Ferrandez, Timothy Harbison, Troy Weber, Robert Sturges, and Robert Rich. Optimization of a Truck-Drone in Tandem Delivery Network using k-means and Genetic Algorithm. *Journal of Industrial Engineering and Management*, 9(2):374–388, 2016.

[19] Greg N. Frederickson, Matthew S. Hecht, and Chul E. Kim. Approximation Algorithms for some Routing Problems. In *17th Annual Symposium on Foundations of Computer Science, Houston, Texas, USA, 25-27 October 1976*, pages 216–227, 1976.

[20] Michael R Garey and David S Johnson. *Computers and Intractability; A Guide to the Theory of NP-Completeness*. WH Freeman & Co., 1990.

[21] Peter Hart, Nils Nilsson, and Bertram Raphael. A Formal Basis for the Heuristic Determination of Minimum Cost Paths. *IEEE Transactions on Systems Science and Cybernetics*, 2(4):100–107, 1968.

[22] Florence Ho, Ana Salta, Ruben Geraldes, Artur Goncalves, Marc Cavazza, and Helmut Prendinger. Multi-agent Path Finding for UAV Traffic Management. In *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 131–139, 2019.

[23] Jose Holguin-Veras, Johanna Amaya Leal, Ivan Sanchez-Diaz, Michael Browne, and Jeffrey Wojtowicz. State of the art and Practice of Urban Freight Management: Part I: Infrastructure, Vehicle-Related, and Traffic Operations. *Transportation Research Part A: Policy and Practice*, 2018.

[24] Wolfgang Hönig, Scott Kiesel, Andrew Tinka, Joseph W Durham, and Nora Ayanian. Conflict-based Search with Optimal Task Assignment. In *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 757–765, 2018.

[25] Edward Humes. Online Shopping Was Supposed to Keep People Out of Traffic. It Only Made Things Worse, 2018. URL http://bit.ly/2HCkAmQ. Accessed: August 30, 2019.

[26] Ramón Iglesias, Federico Rossi, Rick Zhang, and Marco Pavone. A BCMP network approach to modeling and controlling autonomous mobility-on-demand systems. *I. J. Robotics Res.*, 38(2-3), 2019.

[27] Martin Joerss, Florian Neuhaus, and Jurgen Schroder. How Customer Demands are Reshaping Last-Mile Delivery, 2016. URL https://mck.co/2NIRdmE. Accessed: August 30, 2019.

[28] Nabin Kafle, Bo Zou, and Jane Lin. Design and Modeling of a Crowdsource-Enabled System for Urban Parcel Relay and Delivery. *Transportation Research Part B: Methodological*, 99:62 – 82, 2017. ISSN 0191-2615.

[29] Yuxi Li, Janelee Harms, and Robert Holte. Fast Exact Multiconstraint Shortest Path Algorithms. In *IEEE International Conference on Communications*, pages 123–130, 2007.

[30] Minghua Liu, Hang Ma, Jiaoyang Li, and Sven Koenig. Task and Path Planning for Multi-Agent Pickup and Delivery. In *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 1152–1160, 2019.

[31] Hang Ma, Jiaoyang Li, TK Kumar, and Sven Koenig. Lifelong Multi-agent Path Finding for Online Pickup and Delivery Tasks. In *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 837–845, 2017.

[32] Clair E Miller, Albert W Tucker, and Richard A Zemlin. Integer Programming Formulation of Traveling Salesman Problems. *Journal of the ACM (JACM)*, 7(4):326–329, 1960.

[33] Matthias Müller-Hannemann, Frank Schulz, Dorothea Wagner, and Christos Zaroliagis. Timetable Information: Models and Algorithms. In *Algorithmic Methods for Railway Optimization*, pages 67–90. Springer, 2007.

[34] Chase C. Murray and Amanda G. Chu. The Flying Sidekick Traveling Salesman Problem: Optimization of Drone-Assisted Parcel Delivery. *Transportation Research Part C: Emerging Technologies*, 54:86 – 109, 2015.

[35] Alena Otto, Niels Agatz, James Campbell, Bruce Golden, and Erwin Pesch. Optimization Approaches for Civil Applications of Unmanned Aerial Vehicles (uavs) or Aerial Drones: A Survey. *Networks*, 72(4): 411–458, 2018.

[36] Judea Pearl and Jin H Kim. Studies in Semi-Admissible Heuristics. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, (4): 392–399, 1982.

[37] Evangelia Pyrga, Frank Schulz, Dorothea Wagner, and Christos Zaroliagis. Efficient Models for Timetable Information in Public Transportation Systems. *Journal of Experimental Algorithmics (JEA)*, 12: 2–4, 2008.

[38] Mauro Salazar, Federico Rossi, Maximilian Schiffer, Christopher H. Onder, and Marco Pavone. On the interaction between autonomous mobility-on-demand and public transportation systems. In *International Conference on Intelligent Transportation Systems*, pages 2262–2269, 2018.

[39] Guni Sharon, Roni Stern, Ariel Felner, and Nathan Sturtevant. Conflict-based Search for Optimal Multi-Agent Path Finding. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2012.

[40] David Silver. Cooperative Pathfinding. In *AAAI Conference on Artificial Intelligence (AAAI)*, pages 117–122, 2005.

[41] Marius M Solomon. Algorithms for the Vehicle Routing and Scheduling Problems with Time Window Constraints. *Operations Research*, 35(2):254–265, 1987.

[42] Kiril Solovey, Mauro Salazar, and Marco Pavone. Scalable and Congestion-Aware Routing for Autonomous Mobility-On-Demand via Frank-Wolfe Optimization. In *Proceedings of Robotics: Science and Systems*, 2019.

[43] Adrienne Welch Sudbury and E Bruce Hutchinson. A Cost Analysis of Amazon Prime Air (Drone Delivery). *Journal for Economic Educators*, 16(1):1–12, 2016.

[44] P. Toth and D. Vigo. *Vehicle Routing – Problems, Methods, and Applications*. SIAM, 2 edition, 2014.

[45] Alex Wallar, Menno Van Der Zee, Javier Alonso-Mora, and Daniela Rus. Vehicle rebalancing for mobility-on-demand systems with ride-sharing. In *IEEE/RSJ International Conference on Intelligent Robots*

and Systems, pages 4539–4546, 2018.

[46] J. Yu and S. M. LaValle. Optimal Multirobot Path Planning on Graphs: Complete Algorithms and Effective Heuristics. *IEEE Transactions on Robotics*, 32(5):1163–1177, 2016.

[47] Jingjin Yu and Steven M LaValle. Structure and Intractability of Optimal Multi-robot Path Planning on Graphs. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2013.

[48] J. Zgraggen, M. Tsao, M. Salazar, M. Schiffer, and M. Pavone. A Model Predictive Control Scheme for Intermodal Autonomous Mobility-on-Demand. In *IEEE International Conference on Intelligent Transportation Systems*, 2019.