Annals of Operations Research
https://doi.org/10.1007/s10479-023-05714-7

ORIGINAL RESEARCH

The urban air mobility problem

Bruce Golden[1] · **Eric Oden**[2] · **S. Raghavan**[3]

Received: 12 April 2023 / Accepted: 6 November 2023

Abstract

Over the next two decades, Urban Air Mobility (UAM) Systems are anticipated to revolutionize the mass transportation industry. As envisioned presently, these systems will consist of electric vertical take-off and landing aircraft (eVTOLs) that operate from specially designed ports dispersed throughout a city. We consider the network logistics associated with the operation of a UAM system in its early phases, and focus on a problem of 'routing and scheduling eVTOLs to maximize passenger throughput'. Key challenges for providers are the temporal nature of the demand, time windows for customers, and battery management constraints of the eVTOLs. We develop a three-index, arc-based formulation, routing eVTOLs over a time-expanded network. Due to the computational limitations of the arc-based formulation, we develop an alternate path-based formulation, and design a corresponding column generation procedure, identifying charge-feasible routes by way of a resource-constrained shortest path problem. The path-based approach is computationally robust, and can be applied in a heuristic manner by (i) sparsifying the time-expanded network, (ii) limiting column generation to the root node in a branch-and-bound scheme, and (iii) applying early termination criteria in the column generation procedure. Our computational experience on a large set of test instances indicates that the path-based approach identifies high-quality solutions for large instances. We conduct a case study using Washington D.C. taxi data, to demonstrate the viability of the column generation based heuristic procedure on real-world data.

Keywords Urban air mobility · Routing and scheduling · Time-expanded network · Mixed integer program · Smart cities

✉ S. Raghavan
raghavan@umd.edu

Bruce Golden
bgolden@umd.edu

Eric Oden
ericjoden94@gmail.com

[1] Robert H. Smith School of Business, University of Maryland, College Park, Maryland 20742, USA

[2] College of Computer, Mathematical, and Natural Sciences University of Maryland, College Park, Maryland 20742, USA

[3] Robert H. Smith School of Business and Institute for Systems Research, University of Maryland, College Park, Maryland 20742, USA

Published online: 26 December 2023

1 Introduction

Cities around the globe are growing at a rate that no one could have ever predicted in the past. However, with increasing urbanization, travel congestion in urban areas has become one of the greatest detriments to urban living. A key challenge is to produce clean scalable cost-efficient technologies to improve mobility in cities of the future. Looking ahead, many aerospace companies have focused on electric vertical takeoff and landing (eVTOLs) aircraft that can provide safe, rapid transport in an urban environment without the need of significant additional infrastructure (i.e., no need for a runway, etc.) as a technology to address this problem. Some estimates predict a reduction of an average of three hours of daily travel time for a commuter in New Delhi (Holden & Goel, 2016) when opting to fly rather than drive. This burgeoning area of research, encompassing a large variety of logistical, engineering, and regulatory concerns, is referred to by the umbrella term Urban Air Mobility (UAM), or 'Flying Taxis' by the popular press.

An early form of UAM service already exists in the form of helicopter flights (Stone, 2018), which are often used to bypass traffic on time-sensitive trips to the airport. As the service exists currently, it is considered a luxury service rather than a regular commuting option. However, the promise of eVTOL technology with its order of magnitude advantages on noise, manufacturing cost, and fuel efficiency has resulted in a belief that eVTOLs can deliver as a mass market technology for rapid intracity travel, and encouraged many aerospace companies to venture into this industry. Several manufacturers including Airbus, Airspace Experience Technologies, Aurora Flight Sciences, Bell Helicopter, The Boeing Company, EHANG, Embraer, Karem Aircraft, Kitty Hawk, Lilium (they already have a prototype capable of carrying five passengers, achieving speeds of 186 miles per hour and landing in narrow urban areas, see Hawkins, 2019), Neva Aerospace, Opener, Pipistrel, Volocopter, and Workhorse Group are seeking eVTOL certification.

The white papers put out by the eVTOL industry (see Hendrik, 2020; Holden & Goel, 2016; Xu, 2020) envision the market to be largely an airborne taxi service, integrated into a ridesharing platform. The eVTOLs will operate out of specially designed transportation hubs referred to as 'ports'. Presently, the industry is largely focused on regulatory and engineering aspects in making eVTOLs a reality. While these are key in the development of the technology, network logistics associated with the operation of the UAM system (once certified and commercialized) are likely to be another key factor in the success of this market. Specific challenges operators face include the temporal nature of demand, time windows for customers, and battery management constraints for the eVTOLs.

In this paper, we take the perspective of a company in the early stages of development of an urban air transportation network. This manifests itself with the assumption that demand exceeds supply, due to a limited fleet size. In a more typical setting, the objective could be to minimize travel costs. However, the priority of a new company (using a new technology) intending to rapidly grow its service is to maximize market share. For the UAM service provider, this means maximizing the throughput of their service, i.e., getting as many passengers aboard their eVTOLs as possible.

The rest of this paper is organized as follows. Section 2 provides relevant background, modeling assumptions, and a description of the problem studied in this paper. Section 3 discusses relevant literature. Section 4 develops an arc-based mixed integer program formulation of the problem. However, this approach is not scalable and has computational difficulties dealing with battery management constraints. Consequently, Sect. 5 develops a path-based approach that can adequately address the scale of problems anticipated in practice (at least, initially).

First, Sect. 5.1 introduces the path-based formulation, while Sects. 5.2 and 5.3 describe the associated column generation procedure and pricing problem (which is a shortest path problem with resource constraints), respectively. Section 5.4 provides details on the initialization and Sect. 5.5 explains our branching scheme. Section 6 provides a detailed computational study, which includes a case study involving the Washington D.C. metropolitan area that demonstrates the ability of our approach to solve (anticipated) real-world instances. Section 7 discusses future areas of research in the UAM arena.

2 Problem background

Our knowledge of the network logistics problems has been gained by industry white papers (Hendrik, 2020; Holden & Goel, 2016; Xu, 2020), participation in industry sponsored conferences (e.g., Uber Elevate Summit 2019, Washington D.C.), and contacts in the industry. Based on these, network logistics for the UAM provider (in the early phases) can be separated into three distinct and interrelated problems.

First, where should eVTOL ports be located? The success of this service will largely be dictated by the convenience of the ports for the users. Therefore, ports will have to be constructed to accommodate the expected travel demand (both existing and latent). It is expected that the eVTOLs will be used as one leg in a multi-modal transportation trip. Thus, passengers are likely to assess the tradeoffs of a direct road trip against a multi-modal trip with up to 3 legs (road trip to eVTOL port, eVTOL trip to destination port, road trip from eVTOL port to destination). Wu and Zhang (2021) describe an integer programming formulation for the port placement problem that accounts for these tradeoffs. Second, they need to set a daily schedule for the eVTOLs. Although it is envisaged that UAM service at maturity will be a fully on-demand service, it is expected that in the multi-modal ridesharing setting where the service will be deployed—to reduce complexities of the system—the eVTOL schedule will be fixed (while the road legs will be scheduled dynamically when a request comes in), customers will be offered an eVTOL option if there is capacity available on their trip. Thus, given the location of ports and daily (potential) trip demands (i.e., origin port, destination port, and time window at origin) the goal is to develop an a priori eVTOL schedule to maximize customer throughput. Third, since day-to-day demand patterns may vary, when required (e.g., if it becomes clear that changing the schedule will result in a significant improvement in throughput), the provider should be able to modify the a priori schedule in real time to improve throughput.

In this paper, we concentrate on the second problem—determining an a priori schedule for the eVTOLs based on the anticipated demand—for several reasons. The placement of ports is a one time decision as they are capital intensive and require several years of planning and construction (currently UAM system providers are planning on retrofitting roofs of large garages as ports). Although their location could be optimized, there are other factors that play a larger role in determining their location (e.g., tax incentives, partnerships with garage operators, etc). Once port locations are fixed, the success of the UAM system depends on its ability to route as much demand as possible. Thus, the a priori routing and scheduling problem is critical. Further, as demand changes and grows, it will be necessary to repeatedly solve this problem. With eVTOL service several years away from being ready for commercialization (several cities including Dallas, Los Angeles, and Melbourne (Australia) have announced plans and hope to pilot ride sharing transportation networks that use eVTOLs, but this could be delayed as eVTOLs continue to be designed and await certification, a lengthy process) the

parameters necessary to formulate the third problem are unclear at this juncture. Finally, we note that a solution procedure for the second problem can be a key tool for strategic planning for the UAM service provider (i.e., it could be used right away as it plans ahead for service and allows one to answer many strategic what-if questions and explore scenarios). With this background, we now elaborate and explicitly describe the second problem.

Suppose a provider possesses a fleet of eVTOLs, each of a finite, uniform capacity and a finite, uniform charge capacity. There is a set of ports dispersed throughout the area at which the eVTOLs are limited to take off and land. The charge level of each eVTOL is reduced as it travels, but may be recharged when stationed (either by swapping the battery or by plugging the battery into a power source). The provider has a set of customers who wish to use the eVTOLs over some time horizon (e.g., this could be the entire day, or the day could be broken up into pieces like an am peak, pm peak, and off-peak periods). Each customer is characterized by an origin port, destination port, and a time window in which the departure is requested. The customer may only be routed on a nonstop flight from her origin port to her destination port (given that the eVTOL leg is one of multiple legs in a multi-modal trip and the eVTOLs have a limited capacity it is expected to be more efficient operationally and profitable in terms of maximizing revenue to only allow direct legs for the eVTOL portion, at least, initially).

At the beginning of the time horizon, the eVTOLs may be placed at any of the ports (perhaps by relocating overnight). The charge level of each of the eVTOLs is initialized to a given value (typically full charge), and must remain above the minimum charge level. An eVTOL can pick up a passenger if the eVTOL is currently located at the customer's origin port, the current time is within the passenger's time window, and the eVTOL immediately flies directly to the passenger's destination port. The number of passengers the eVTOL can carry cannot exceed the eVTOL's capacity. Each eVTOL's charge level decreases as it flies from one port to another, but can increase while the eVTOL is stationary at a port (by recharging). We seek to maximize the total number of customers served. We call this the Urban Air Mobility Problem (UAMP).

We note that the UAMP (as described) is deterministic (significant variations from the planned for demand that necessitate schedule changes would be addressed on a day-to-day basis in real-time). In this paper, we assume a linear charge/discharge behavior for the batteries. However, nonlinear charge/discharge behavior (which is more typical as factors including payload and current charge level influence the recharge/drainage rates) is easily incorporated into the path-based approach described in Sect. 5. We should clarify one issue that might have caught the attention of the alert reader. It may be the case that in a multi-modal trip there are multiple possibilities for the eVTOL leg, with differing origins and destinations (this could be the case in a large network with a very large number of eVTOL ports). However, in our modeling, we assume that a given customer's demand will only be associated with a single origin–destination port pair (this is reasonable as, in the early phases, the possibility of having multiple possible origin–destination pairs for the eVTOL leg is low due to the fact that there will be few ports). We note, however, with minor adaptations (as described in the appendix to this paper) our model easily accounts for this.

3 Related literature

To the best of our knowledge, there is no published work in the operations research literature on the UAMP. There are a handful of papers (all of these research efforts are largely in parallel

to ours) that deal with various aspects of an urban air mobility system. Chen et al. (2022), Lim and Hwang (2019), Wang et al. (2022), and Wu and Zhang (2021) focus on the port placement problem. Specifically Chen et al. (2022) describe a variable neighborhood search heuristic that is able to solve 144 node instances (of the port placement problem) rapidly. Lim and Hwang (2019) used a k-means algorithm to cluster the commuting data and select ports for Seoul, Korea. Wang et al. (2022) describe a sophisticated adaptive discretization approach to approximately solve (typically within 1% of optimality) a port placement problem that accounts for tradeoffs between the strategic aspects of the port placement problem, tactical aspects of UAM operations, and customer adoption. Wu and Zhang (2021) build an integer programming model for the port placement problem that accounts for customer's mode choice between ground transportation and multimodal UAM service. They apply their model to a case study in the Tampa Bay (Florida, USA) area. Pelegrin and D'Ambrosio (2022) and Tang et al. (2021) focus on automated route planning and collision avoidance systems in UAM systems. This is an important issue since it is envisaged that eVTOLs will be 'pilotless'. Garrow et al. (2020) and Hill and Garrow (2021) focus on survey data and market segmentation analysis to estimate demand for air taxi service.

We note that the UAMP shares a few similarities with the well-studied Dial-a-Ride Problem (DARP), first introduced in the 1970s, which seeks the minimum-cost routing of a fleet of vehicles such that each customer is picked up and dropped off at nodes within passenger-set time windows. Additionally, the ride time of each customer must not exceed some maximum length. For a thorough account of the state of the art, we refer the reader to the survey by Ho et al. (2018). According to the survey, the most successful exact method for solving the DARP is the branch-and-cut-and-price approach developed by Gschwind and Irnich (2014).

With the increasing presence of electrically powered automobiles on urban road networks, many vehicle routing problems have been adapted to incorporate battery life constraints. Energy-consumption models have been developed for this purpose (e.g., Genikomsakis & Mitrentsis, 2017; Goeke & Schneider, 2015; Pelletier et al. 2018). Such models can be used to predict charging and discharging rates for use in transportation modeling. In particular, Masmoudi et al. (2018) solve an electric DARP using an Evolutionary Variable Neighborhood Search approach to solve a mixed-integer program, using the model presented in Genikomsakis and Mitrentsis (2017) to determine energy consumption. In the work of Masmoudi et al. (2018), recharging is accomplished by battery swapping stations dispersed throughout the network. In the work of Bongiovanni et al. (2019), a branch-and-cut approach is used to solve the electric DARP.

We develop a discrete-time model of the UAMP, which maps routing/scheduling decision points onto a time-expanded network. This technique for representing a problem involving temporal constraints has been applied in many contexts. For example, see Marshall et al. (2021) and Bsaybes et al. (2019).

There are two important differences between the UAMP and the above problems. First, rather than minimizing the routing costs while guaranteeing that each customer is served, we seek to maximize the number of customers served. This stems from our perspective of a UAM provider in early development, where the priority is to maximize market share. Second, in each of the earlier problems, routes were not required to travel immediately to a customer's destination node after picking them up. Instead, quality of service considerations are incorporated by including time windows on arrival times and maximum trip lengths. As stated in Sect. 2, our assumption of non-stop passenger flights is more appropriate here.

4 Arc-based approach

In our work, we have focused on a discrete-time model of the problem, where the time horizon is partitioned into time steps. We will describe both an arc-based and path-based approach in this discrete time setting. It has been noted that there is a price to pay for the quantization of time in some models (Boland et al., 2018). However, this approximation can be justified as an operational reality. Though improved routing may be possible if flight departures could be scheduled down to the second, it is perhaps unreasonable to expect such precision in reality. If flight times are rounded up to the nearest time step (say, five minutes), we automatically incorporate temporal 'padding' for any proposed schedule. We now describe an arc-based mathematical formulation of the problem. We first define notation and then present the formulation.

4.1 Notation

Let N be the set of eVTOL ports, and let $T = \{0, 1, \ldots, t_{\max}\} \subset \mathbb{Z}$ be the discrete time horizon. Each time step corresponds to a fixed amount of time, τ. For instance, if the service is to begin at 8:00 am, and the time step, τ, is five minutes, then 0 corresponds to 8:00 am, 1 to 8:05 am, and so on. Let C be the set of customers. Each customer $c \in C$ is associated with an origin port $o_c \in N$, a destination port $d_c \in N$, and a set of times $T_c \subseteq T$ in which the customer can depart from her origin node. T_c must be a contiguous set of times. That is, if $t_c^a = \min\{t | t \in T_c\}$, $t_c^b = \max\{t | t \in T_c\}$, and $t \in T$ is such that $t_c^a \leq t \leq t_c^b$, then $t \in T_c$. Thus, T_c represents the discrete time window for customer c. Let $K = \{1, \ldots, |K|\}$ denote the fleet of eVTOLs, and let $S \in Z^+$ be the capacity (number of seats) of each eVTOL. Let Q_+, Q_-, and Q_0 denote the uniform maximum, minimum, and initial charge levels of each eVTOL, respectively.

Define the complete, directed graph $G = (N, A)$. Each arc $(i, j) \in A$ is associated with a travel time, $t_{ij} \in \mathbb{Z}^+ \cup \{0\}$ (this quantity may include take-off, landing, and some minimum 'turnaround' time allocated for inter-flight processes such as deplaning, boarding, etc.). The units here correspond to time steps, as opposed to minutes. Without loss of generality, we assume travel times satisfy the triangle inequality. Define the directed, time-expanded graph $H = (M \cup \{s, f\}, B)$, where $M = N \times T$, s and f represent fictitious origin and destination nodes, respectively, and B is the set of directed arcs connecting members of $M \cup \{s, f\}$. Each node $i \in M$ corresponds to a port-time pair (p, t), where $p \in N$ and $t \in T$. When the port associated with node $i \in M$ must be specified, it is denoted p_i. Likewise, the time associated with node i is denoted t_i. An arc (i, j) is included in B if and only if $t_j - t_i = t_{p_i, p_j}$, or if $i = s$ and $t_j = 0$, or if $j = f$ and $t_i = t_{\max}$. This includes all possible eVTOL flight connections, as well as links the fictitious origin and destination nodes with each of the ports at the beginning and end of the time horizon, respectively. We allow for an eVTOL to remain stationary at port $i \in N$ for a single time step by defining $t_{ii} = 1$ for all $i \in N$. Note, H is an acyclic network. Denote the change in an eVTOL's charge level associated with arc $(i, j) \in B$ with $e_{ij} \in \mathbb{R}$, which may correspond to an increase or a decrease. Let $\delta^+(i) \subset M \cup \{s, f\}$ and $\delta^-(i) \subset M \cup \{s, f\}$ denote the set of outgoing and incoming nodes for node $i \in M \cup \{s, f\}$, respectively. That is, $\delta^-(i)$ denotes those j such that $(j, i) \in B$, and $\delta^+(i)$ denotes those j such that $(i, j) \in B$. Let $\mathcal{F}(c) \subseteq B$ denote the set of possible flights for customer c. This will be the collection of arcs $(i, j) \in B$, such that $p_i = o_c$, $p_j = d_c$, and $t_i \in T_c$. Denote the set of all possible flights $\mathcal{F} = \bigcup_{c \in C} \mathcal{F}(c)$.

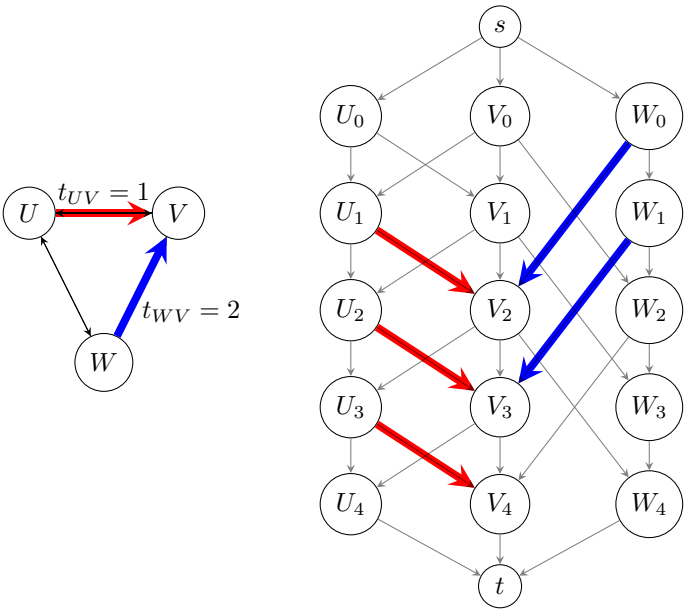

Fig. 1 Example with three ports and two customers to illustrate notation. The network representing pairwise distances between ports is on the left, and the time-expanded network of feasible connections is on the right

We illustrate the above notation with a simple example, depicted in Fig. 1. Here $N = \{U, V, W\}$ is the set of ports, and the time horizon is $T = \{0, 1, 2, 3, 4\}$. The set C consists of two customers, c_1 and c_2, whose associated arcs are depicted in red and blue, respectively. Customer c_1 wishes to travel from port $o_{c_1} = U$ to port $d_{c_1} = V$, while customer c_2 wishes to travel from port $o_{c_2} = W$ to port $d_{c_2} = V$. Customer c_1 wishes to depart from its origin at times 1, 2, or 3, while customer c_2 wishes to depart from its origin at times 0 or 1. That is, $T_{c_1} = \{1, 2, 3\}$ and $T_{c_2} = \{0, 1\}$. Note that T_{c_1} and T_{c_2} are contiguous subsets of T. On the left-hand side of Fig. 1, we show the network $G(N, A)$, where A is the set of directed arcs connecting the ports. It takes 1 unit of travel time to travel from U to V, and 2 units to travel from W to V, hence $t_{UV} = t_{VU} = 1$ and $t_{WV} = t_{VW} = 2$. Suppose $t_{UW} = 5$. On the right-hand side, we show the time-expanded network H. $M = \{U_0, U_1, \ldots, U_4, V_0, \ldots, V_4, W_0, \ldots, W_4\}$ is the set of port-time nodes. The light grey arcs represent B, connecting port-time nodes. The fictitious node s is connected to each of the U_0, V_0, and W_0 nodes, and represents the source of the network. Meanwhile, the fictitious node t is connected to each of the U_4, V_4, and W_4 nodes, and represents the sink of the network. Notice arcs in B connect different port-time nodes if the difference in time from the origination node to the destination node is equal to the travel time between the ports. Since it is impossible to travel from U to W in 5 units of time, there is no arc connecting any U_i node to any W_j node. For port-time node U_1, $\delta^+(U_1) = \{U_2, V_2\}$, and $\delta^-(U_1) = \{U_0, V_0\}$. We have $\mathcal{F}(c_1) = \{(U_1, V_2), (U_2, V_3), (U_3, V_4)\}$, $\mathcal{F}(c_2) = \{(W_0, V_2), (W_1, V_3)\}$, and $\mathcal{F} = \mathcal{F}(c_1) \cup \mathcal{F}(c_2)$.

4.2 Three-index formulation

For each arc $(i, j) \in B$, and for each eVTOL $k \in K$, let x_{ij}^k be a binary variable equal to 1 if and only if eVTOL k traverses arc (i, j). For each customer $c \in C$, and for each $(i, j) \in \mathcal{F}(c)$, let the binary variable y_{ij}^c be equal to 1 if and only if customer c flies along arc (i, j). For each $i \in M \cup \{s, f\}$, and for each eVTOL $k \in K$, let $q_i^k \in \mathbb{R}$ indicate the charge level of eVTOL k upon arrival at node i. If eVTOL k does not visit node i, the quantity is meaningless. With these variables, we can formulate the UAMP with the following mixed-integer program, which we denote MIP3 to reflect the three indices on the decision variable x_{ij}^k.

$$\text{(MIP3)} \quad \text{Max} \sum_{c \in C} \sum_{\{(i,j) \in \mathcal{F}(c)\}} y_{ij}^c \tag{1}$$

$$\text{s.t.} \sum_{j \in \delta^+(i)} x_{ij}^k - \sum_{j \in \delta^-(i)} x_{ji}^k = \begin{cases} 1 & i = s \\ 0 & i \in M \backslash \{s, f\} \\ -1 & i = f \end{cases} \quad \forall i \in M \cup \{s, f\}, k \in K \tag{2}$$

$$\sum_{\{c | (i,j) \in \mathcal{F}(c)\}} y_{ij}^c \leq \sum_{k \in K} S x_{ij}^k \quad \forall (i, j) \in \mathcal{F} \tag{3}$$

$$\sum_{(i,j) \in \mathcal{F}(c)} y_{ij}^c \leq 1 \quad \forall c \in C \tag{4}$$

$$q_0^k = Q_0^k \quad \forall k \in K \tag{5}$$

$$q_j^k \leq (q_i^k + e_{ij}) + M(1 - x_{ij}^k) \quad \forall (i, j) \in H, k \in K \tag{6}$$

$$Q_- \leq q_i^k \leq Q_+ \quad \forall i \in M, k \in K \tag{7}$$

$$x_{ij}^k \in \{0, 1\} \quad \forall (i, j) \in H, k \in K \tag{8}$$

$$y_{ij}^r \in \{0, 1\} \quad \forall (i, j) \in \mathcal{F}(c), c \in R \tag{9}$$

$$q_i^k \in \mathbb{R} \quad \forall i \in M, k \in K \tag{10}$$

The objective function (1) maximizes the number of customers that are served. Constraints (2) ensure that each vehicle k follows a path through the time-expanded network H, beginning at s and ending at f. Constraints (3) ensure that for each possible flight arc, the number of passengers traveling along that arc is bounded from above by the number of eVTOLs taking the same arc, multiplied by the uniform eVTOL capacity. Constraints (4) ensure that each customer is served at most once. Constraints (5) initialize the charge level of each eVTOL. Constraints (6) ensure that the charge level of an eVTOL evolves according to its path. This is achieved with a 'Big M' term, where, e.g., $M = Q_+ + \max_{(i,j) \in B} e_{ij}$. Constraints (7) bound the charge levels. Constraints (8) and (9) ensure binary-valued eVTOL flow and passenger flow variables, respectively, and constraints (10) ensure real-valued charge levels. There are a number of extensions that can be incorporated into this formulation, including customer groups, soft time windows, and battery swaps. We describe these modifications in the appendix to this article.

4.3 Two-index heuristic

The charge constraints significantly complicate the problem, and, therefore, severely limit the problem sizes that can be solved using MIP3 within a reasonable computation time. One idea is to relax the charge constraints, solve the problem, and, then, repair the solution to make it charge-feasible. When the charge constraints are relaxed it suffices to aggregate the arc variables to obtain a two-index formulation, i.e., aggregate the eVTOL flow variables from MIP3, $x_{ij} = \sum_{k \in K} x_{ij}^k$. The integer-valued variable x_{ij} in a two-index formulation thus corresponds to the number of eVTOLs traveling along arc (i, j). The resulting integer program (that we refer to as MIP2) is described in the appendix. The solution to MIP2 provides a set of K throughput-maximizing paths through the network, which may not satisfy charge constraints. To obtain a charge-feasible solution, we use the solution to MIP2 to sparsify the network and then apply MIP3. The process is as follows. We first solve MIP2 and obtain an optimal solution $x^* = \{x_{ij}^*\}_{(i,j) \in B}$. We then construct a sparsified network B^0. An arc $(i, j) \in B$ is included in B^0 if and only if $x_{ij}^* > 0$ or $p_i = p_j$ and $t_j = t_i + 1$. That is, we only keep the arcs from the solution of MIP2 as well as the arcs corresponding to remaining at a port for consecutive time steps. Keeping these latter arcs ensures charge feasibility. MIP3 can be run on this dramatically sparsified network. We refer to the process of sparsifying in this fashion, and then running MIP3 on the sparsified network as MIP2⇒3.

Although this approach dramatically improves upon MIP3 in terms of the size of the problems solved, it's efficacy is limited as the scale of problems grows even larger. The appendix provides a detailed discussion of our computational experience with MIP2⇒3. Instead, to improve our ability to find high-quality solutions to larger instances, we pursue a path-based approach in the next section.

5 Path-based approach

We now describe a path-based approach to solve the UAMP. Path-based approaches generally lend themselves to efficient column generation schemes, as well as lead to stronger LP-relaxations than arc-based counterparts. We present the path-based formulation in Sect. 5.1 and develop the corresponding column generation problem in Sect. 5.2. We then present the subproblem algorithm in Sect. 5.3, the manner in which initial paths can be determined in Sect. 5.4, and the branching rules we employ in the event of fractional solutions in Sect. 5.5. Finally, Sect. 5.6 describes a network sparsification heuristic that we apply in the path-based approach to speed up the procedure.

5.1 Path-based formulation

Let C continue to be the set of customers, and V the fleet of eVTOLs. Let R be the set of all charge-feasible eVTOL paths. That is, the set of all paths through the network H from s to f such that all charge constraints are respected. Each route r is associated with a set of customers it serves. For each route, we define the binary-valued vector $s_r = \{s_r^c\}_{c \in C}$, where $s_r^c \in \{0, 1\}$ is 1 if and only if customer c is served by route r. We refer to s_r as the service vector of route r. For each path $r \in R$, let $x_r \in \{0, 1\}$ be 1 if and only if route r is selected. Let $y_c \in \{0, 1\}$ be 1 if and only if customer c is served. We may reformulate the UAMP as the following integer program.

$$\max \sum_{c \in C} y_c \tag{11}$$

$$\text{s.t. } y_c - \sum_{r \in R} s_r^c x_r \leq 0 \qquad \forall c \in C \tag{12}$$

$$\sum_{r \in R} x_r = |V| \tag{13}$$

$$x_r \in \{0, 1\} \qquad \forall r \in R \tag{14}$$

$$y_c \in \{0, 1\} \qquad \forall c \in C \tag{15}$$

We denote this integer program MP, for master problem. The objective (11) maximizes the number of customers served. Constraints (12) ensure customers are only counted as served if a route which serves them is selected. Constraint (13) ensures the number of eVTOLs is equal to $|V|$. Constraints (14) and (15) ensure binary-valued route selection and customer service variables, respectively. The above formulation may be relaxed as a linear program by replacing constraints (14) and (15) with the constraints below.

$$y_c \leq 1 \qquad \forall c \in C \tag{16}$$

$$y_c \geq 0 \qquad \forall c \in C \tag{17}$$

$$x_r \geq 0 \qquad \forall r \in R \tag{18}$$

Constraints (16) prohibit any customer from being counted more than once, and constraints (17) and (18) ensure nonnegativity. We need not bound the x_r variables by 1, as there is no benefit obtained with solutions that have $x_r > 1$ (i.e., if $x_r > 1$ occurs, we replace x_r by 1, remaining feasible and use the excess above 1 to bring in another route variable(s)). However, as a linear relaxation, fractional solutions are a possibility. We denote this linear relaxation L-MP, for linear master problem.

5.2 Column generation

The main challenge in the path-based approach is the size of R, the set of all charge-feasible paths. This set is generally very large, and, therefore, the MP formulation consists of too many variables to be efficiently solved using a simplex/branch-and-bound based solver. Instead, we employ a column generation approach, by first selecting a small subset of paths, $R^0 \subset R$. We then solve the L-MP, restricted to this subset of paths. We denote this restricted problem L-RMP, for linear restricted master problem. We denote the corresponding problem with binary constraints the RMP. Solving the L-RMP results in a set of dual variables associated with each constraint. We can then use these variables to identify 'useful' paths. That is, those paths which, if included in the formulation, would lead to an increase in objective value. These paths are added to R^0, and the procedure is repeated. This can continue until no more useful paths can be identified, or some other stopping criterion is satisfied.

We now explicitly develop this idea. Let the dual variables α_i, β, and γ_i correspond to constraints (12), (13), and (16), respectively. Taking the dual of the path-based formulation, we have:

$$\min |V|\beta + \sum_{c \in C} \gamma_c \tag{19}$$

$$\text{s.t.} \qquad \alpha_i + \gamma_i \geq 1 \qquad \forall c \in C \tag{20}$$

$$\beta - \sum_{c \in C} s_r^c \alpha_c \geq 0 \qquad \forall r \in R \tag{21}$$

$$\alpha_c \geq 0 \qquad \forall i \in C \tag{22}$$

$$\gamma_c \geq 0 \qquad \forall i \in C. \tag{23}$$

If constraint (21) is violated for any route $r \in R$, it indicates that route has a positive reduced cost, implying the primal solution must be suboptimal. Thus, we seek routes r such that:

$$\beta - \sum_{c \in C} s_r^c \alpha_c < 0 \tag{24}$$

and the subproblem can be stated as:

$$\max_{r \in R} \left\{ \sum_{c \in C} s_r^c \alpha_c \right\}. \tag{25}$$

Thus, we seek the route $r \in R$ such that the inner product of its service vector, s_r, and $\alpha = \{\alpha_c\}_{c \in C}$ is maximized. This can be stated as an elementary shortest path problem with resource constraints, a well-studied problem (see Irnich & Desaulniers, 2006). This is done in the following way. For each customer $c \in C$, each arc $(i, j) \in \mathcal{F}(c)$ is given a weight of $-\alpha_c$. If an arc (i, j) corresponds to a flight arc for multiple customers (i.e., $(i, j) \in \mathcal{F}(c_1) \cap \mathcal{F}(c_2) \cap \cdots \cap \mathcal{F}(c_k)$), the weight is simply the negative of the sum of the corresponding α values (i.e., $-(\alpha_{c_1} + \cdots + \alpha_{c_k})$). If the number, k, of such customers is greater than the uniform eVTOL capacity (i.e., $k > S$), then the arc weight is the negative of the sum of the S greatest corresponding α values. All other arcs in the network are given a weight of 0. We then seek a path from s to t of minimum weight, where the path weight is the sum of the arc weights along the path. Of course, multiple paths can have the same weight, and indeed, many paths can share a service vector despite taking a different route through H. Because our underlying network is acyclic, all feasible paths are acyclic, and, thus, we may instead solve the shortest path problem with resource constraints (SPPRC), a significantly easier problem (the elementary shortest path problem with resource constraints requires that the path be loopless, while the shortest path with resource constraints has no such restriction). We implement a label-setting algorithm to solve this problem, which we describe in Sect. 5.3.

5.3 Subproblem

We seek to identify paths of positive reduced cost to be added to the reduced master problem, which amounts to solving the shortest path problem with resource constraints on the time expanded network $H = (M, B)$. A standard way to solve shortest path problems is to use a label-setting algorithm, which we implement here. For an introduction to the method, see Ahuja et al. (1993). The main data structure in the algorithm is the label, which corresponds to the state of an eVTOL at a particular point on its path, having completed some sub-path through the network. Labels are identified by the current node, current charge level, current score, and a pointer to the previous label. We establish two lists of labels: the unprocessed labels, U, and the processed labels, P. We initialize by adding the starting label, l_0, to U, where:

$$l_0 = [s, Q_0, 0, \emptyset].$$

That is, the starting label is assigned the current node s, the initial charge, a score of zero, and no previous label. While U is nonempty, we choose a label $l \in U$. Let i_l, q_l, and z_l be the node, charge level, and score associated with label l, respectively. We then 'treat' this label. This is done by observing all $j \in \delta^+(i_l)$ (the outgoing nodes of i_l), and creating the new label:

$$l' = [j, \min(Q_+, q_l + e_{ij}), z_l + z_{ij}, l]$$

where z_{ij} is the the weight of the arc (the sum of the reduced costs of the customers that can be served along the arc, or the greatest such sum if the number of customers exceeds the capacity). If $q_l + e_{ij} < Q_-$, the label is thrown away. Otherwise, the label is added to U. Once l is treated, it is added to P. This continues until U is empty. Once U is empty, P is searched for the label $l \in P$ corresponding to a complete path that maximizes z_l. That is, we identify:

$$\arg\max_{z_l} \{l \in P \mid i_l = f\}.$$

This path will be the solution to the SPPRC. We can identify the path corresponding to this label by backtracking through the pointers to previous labels. We can likewise identify the customers served along the path by, for each arc, identifying those customers who can be served along the arc. If the number of such customers exceeds the capacity, we choose the S customers of greatest reduced cost.

There are a couple of considerations that can speed up this algorithm. We can employ a dominance filter to the labels added to U. A new label, l^*, is added to U only if it is not dominated by any label $l \in U \cup P$. A label a dominates another label b if and only if:

$$i_a = i_b \text{ and } q_a \geq q_b \text{ and } z_a \geq z_b.$$

A useful feature of the label-setting algorithm is that many paths of positive reduced cost can be produced in one pass. Rather than only extracting the path of maximum reduced cost, we can instead extract several paths of positive reduced cost at a time.

5.4 Initialization

In the column generation approach, a set of starting paths must be identified. It has been noted (by, for example, Barnhart et al., 1998) that the choice of initial paths is important, as they influence the dual variables identified by L-RMP, and, thus, the new paths produced. Efforts must be made to produce 'good' paths.

We implement a simple greedy initialization procedure. First, we randomly select a starting port and initialize the charge level (typically setting the charge to full). Then, we consider each possible choice of a next node, determined by the arcs in the network, as well as the current charge level. We choose the arc of the greatest score. The score is determined the same way as in the label-setting method, where each customer's α value is set to 1. We then add the arc to the path and move to the corresponding node. In the case of a tie in score, we randomly choose between the tied arcs. We continue until reaching node f. This greedy heuristic can also be used as a heuristic to the subproblem, as an alternative to the labeling algorithm discussed in Sect. 5.3. That is, we can apply the greedy heuristic to find 'good' paths with positive reduced cost, and resort to the exact labeling procedure if the greedy algorithm yields no paths.

5.5 Branching

We branch on the fractional x_r variables, by creating two nodes, one corresponding to $x_r = 0$, the other $x_r = 1$. This corresponds to partitioning solutions into those that prohibit route r (as well as all routes that serve the same customers as route r) and those that include route r. In the latter case, the (pricing) subproblem becomes simpler, as the number of eVTOLs can be reduced by 1, and the customers served by route r may be removed from consideration.

In the former case, when x_r is enforced to be 0, we must prohibit the corresponding variable from entering the solution. Note that other routes may serve the same set of customers as route r. This can happen, for example, by shifting the departure times. We say two routes are equivalent if they serve the same set of customers, i.e., r' is equivalent to r'' if and only if $s_{r'} = s_{r''}$. We remove all routes equivalent to r from the L-RMP. Furthermore, to prevent the entry of an equivalent route into the L-RMP, we add the set of customers served by r to a list of prohibited customer sets. Then, for each route in P (the set of complete routes found by the labeling algorithm), we check if the set of customers served by the route is in the list. If so, the route is removed from consideration.

5.6 Network sparsification

To improve the run times of the above approaches, we implemented a network sparsification procedure to further limit the size of the solution space. For a customer c, the arcs $\mathcal{F}(c)$ are her **passenger flight arcs**. Let $t_c^a = \min\{t | t \in T_c\}$, $t_c^b = \max\{t | t \in T_c\}$ be the first and last time values at which customer c can depart from her origin port, respectively. Let the two arcs $\hat{\mathcal{F}}(c) = \{(i, j) \in \mathcal{F}(c) \mid t_i \in \{t_c^a, t_c^b\}\}$ be the **extreme flight arcs**. For a customer c, the arcs $\mathcal{A}(c) = \{(i, j) \in B \mid p_j = o_c, t_j \in T_c, p_i \neq p_j\}$, are the customer's **anticipatory arcs**. These are those arcs that are traveling from another port to the customer's origin within the customer's time window. Let the arcs $\hat{\mathcal{A}}(c) = \{(i, j) \in \mathcal{A}(c) \mid t_j \in \{t_c^a, t_c^b\}\}$ be the customer's **extreme anticipatory arcs**. Denote the arcs $\{(i, j) \in B \mid p_i = p_j, t_i + 1 = t_j\}$ as the **stay put arcs**. We limited the time expanded networks in our data sets to only include the extreme passenger flight arcs, extreme anticipatory arcs, stay put arcs, and the arcs connecting the fictitious starting and ending nodes to the network. See Fig. 2 for an example of this sparsification.

If there is no upper limit on the eVTOL charge level, and all flight/anticipatory arcs are included (rather than just the extreme arcs), we can prove the sparsification does not cut off the optimal solution (see the appendix). We place an upper bound on the charge level, and only use the extreme flight/anticipatory arcs, and, thus, do not have this guarantee. However, experiments on simulated data based on real-world parameters suggest there is little to no loss in the objective value, but a considerable speed up in run time, when using this procedure. This is particularly the case as the number of customers increases, where there seems to be no loss in objective value.

6 Computational experience

We now describe our computational experience with the arc-based and path-based approaches. These include experiments on simulated data sets and a case study using Washington D.C. taxicab data. Since the instances are difficult to solve to optimality, a major goal in our experiments was to develop a heuristic variant of the path-based approach that

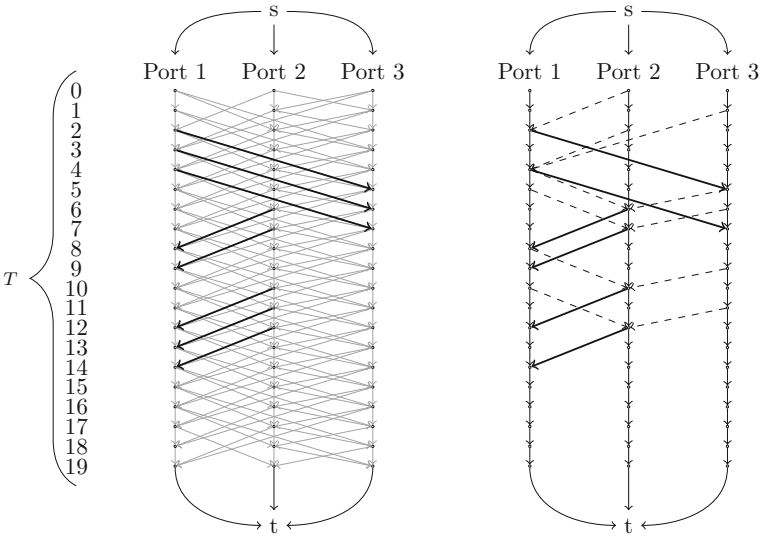

Fig. 2 Comparison of the original network and the sparsified network for a 3-customer, 3 port, 20 time step problem. Left: The original network, which includes every possible eVTOL flight. The passenger flight arcs are in bold. One customer wants to go from Port 1 to Port 3, departing at t = 2, 3, or 4. Another wants to go from Port 2 to Port 1, departing at either t = 6 or 7. A third wants to travel from Port 2 to Port 1, departing at t = 10, 11, or 12. Right: The sparsified network, with only the extreme passenger flight arcs, extreme anticipatory flights, and stay put arcs. Extreme passenger flight arcs are in bold, and extreme anticipatory flights are dashed. In this example, the number of arcs is reduced from 167 to 80

is capable of finding high-quality solutions for the size of instances that may be expected when UAM service is rolled out. Our experiments were run on a Windows computer, using an Intel(R) Core(TM) i5-7400 CPU @ 3.00GHz processor. Our codes were implemented in Python 3.7 and Gurobi v9.0 was the underlying solver.

6.1 Simulated data sets

We created simulated problem sets to evaluate the relative performance of our algorithms. The problems are constructed as follows. We first establish a discrete time horizon, $T = \{1, 2, \ldots, t_{\max}\}$, partitioned into minute-long intervals, so that one time step corresponds to one minute. We then initialize a $\frac{|T|}{10} \times \frac{|T|}{10}$ grid, in which $|P|$ ports are then randomly placed. The distance between ports i and j, d_{ij}, is Euclidean, rounded up to the nearest integer. The corresponding travel time, t_{ij}, between the ports is taken to be equal to d_{ij}. (We start with the assumption that travel time is proportional to distance. Consequently, to simplify the simulated instances further, we let $t_{ij} = d_{ij}$. In the case study discussed in Sect. 6.3, however, we define t_{ij} more precisely.) Each customer is randomly assigned an origin port and a (different) destination port, as well as the uniform time window length $W \in \mathbb{Z}$ minutes. That is, each is willing to wait for W minutes after the beginning of her time window. The beginning of each customer's time window then is randomly drawn from $T' = \{0, 1, \ldots, t_{\max} - W - t^M\}$, where $t^M = \max_{i,j \in N} t_{ij} \in \mathbb{Z}^+$ is the longest of all the pairwise travel times. This ensures all flights arrive at their destination prior to the end of the time horizon. Charge transition values between nodes i and j, e_{ij}, are established as follows:

Table 1 Parameter values

Parameter	Notation	Default value
Time horizon length	$t_{\max}$	60 Time steps
Time step size	τ	1 min
Time window width	W	3 min
Maximum charge level	Q_+	100
Minimum charge level	Q_-	0
Initial charge level	Q_0	100
Charge depletion per time step	ψ	5
Recharge per time step	ϕ	10
eVTOL capacity	S	6

These were the fixed values used in our experiments

$$e_{ij} = \begin{cases} -\psi t_{p_i,p_j} & p_i \neq p_j \\ \phi & p_i = p_j \text{ and } t_j - t_i = 1 \end{cases}$$

where ψ and ϕ are the charge depletion and recharge levels per time step, respectively. Charge transitions are, thus, linear with respect to time.

There are many parameters involved in the UAMP (number of ports, eVTOL speed, eVTOL capacity, number of eVTOLs, number of customers, passenger waiting time, etc.) that at this time vary based on eVTOL manufacturer (since most eVTOLs are still in the early concept phase with prototypes yet to be manufactured and certified), and market assumptions. Consequently, after reviewing industry white papers and studying several papers (see Polaczyk et al., 2019; Bacchini & Cestino, 2019; Uber, 2018) describing specifications of the different eVTOLs in development, we chose a set of reasonable default values that the parameters could assume for testing purposes. Table 1 provides these values, along with their symbolic notation.

We created a set of smaller test instances, that we refer to as Problem Set A, that the branch-and-price approach was largely able to solve to optimality. This baseline family of problems has 4 ports and 4 eVTOLs, and the number of customers range from 5 to 200 in steps of 5 (a total of forty problems). We then introduce a larger (in terms of the port and eVTOL infrastructure) and more difficult family of problems, which considers 8 ports and 8 eVTOLs, with the number of customers ranging from 50 to 300 in steps of five (a total of fifty problems); we refer to these as Problem Set B. The branch-and-price approach was no longer viable for Problem Set B, necessitating the use of column generation as a heuristic to obtain solutions for these instances. We have largely focused our evaluation of algorithms (in terms of how they scale) by varying the number of customers. This is because (i) the market demand is a big uncertainty and there is great value to UAM operators in understanding how solutions vary as the number of customers varies, and (ii) we found that the performance of our approaches was most sensitive to the number of customers (as compared to the number of eVTOLs or number of ports).

6.2 Results on simulated data sets

We now present our findings with the arc-based and path-based procedures when applied exactly on Problem Set A. We then focus on the larger instances in Problem Set B, and experiment with heuristic versions of the path-based approach (for example, by limiting

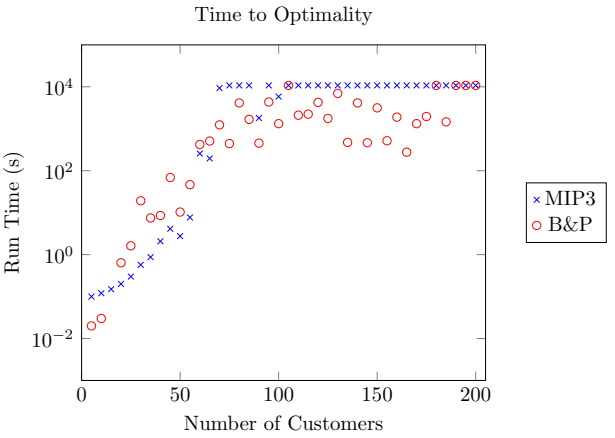

Fig. 3 Time to reach optimality for the UAMP on the Problem Set A, with a three hour time limit imposed. The run time of B&P grows slower than that of MIP3

column generation to the root node). For ease of visualization, we present our results with figures. Tables including run times and objective values for each of the figures are included in the appendix to this paper.

6.2.1 Comparing the arc- and path-based formulations as exact approaches

We compared the Three-Index formulation (MIP3) presented in Sect. 4.2 and the Branch-and-Price method (B&P) presented in Sect. 5 on Problem Set A, with a time limit of three hours. Figure 3 displays the results. While MIP3 appears more efficient for smaller instances, at around 70 customers, B&P outperforms it, and is able to produce and prove the optimal solution for all but five instances (when $|C| = 105, 180, 190, 195$ and 200). Both methods exhibit high variability in run time, with problems of similar size showing run times sometimes differing by an order of magnitude. This reveals a high level of sensitivity to the randomly determined customer data (i.e., origin–destination pairs and departure times).

6.2.2 Column generation heuristics

A popular heuristic version of path-based formulations is to limit column generation to the root node. Generation of columns at each node along the branch and bound tree can be a time consuming process, and it may be the case that such generation is unnecessary if, ultimately, the optimal columns are already in the RMP. In Fig. 4, we present justification for such a heuristic. On the left, we plot the optimality gap (between the upper and lower bounds) for B&P upon completing the root node on Problem Set A. We observe that, for the larger problems, the gap is less than 2%. On the right, with the y-axis shown in logarithmic scale, we show the run times for B&P and the run times to complete solving the RMP at the root node. We observe that the branch-and-price procedure after the root node evaluation can increase the run time by an order of magnitude, while the rewards of such efforts are rather small.

We implemented several column generation based heuristics. First, we only generated columns at the root node. When no more columns of positive reduced cost can be identified,

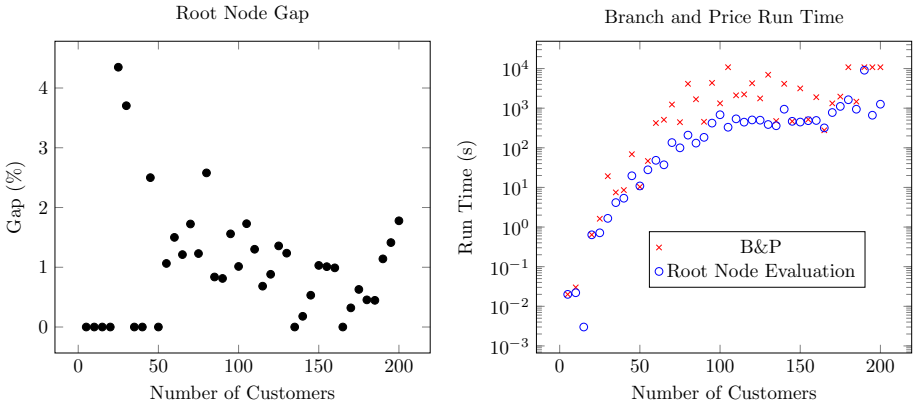

Fig. 4 Left: Percent gap (between the upper and lower bounds) upon finishing the root node for Branch and Price on Problem Set A. For the larger problems, the gap is less than 2%. Right: The run time of the full branch-and-price method and the run time of finishing the root node for the same problems. Together, these plots justify the heuristic that only generates columns at the root node

the program performs traditional branch and bound to find the optimal solution, subject to only having the columns generated at the root available. This heuristic is denoted RCG, for root node column generation. We then considered an extension of RCG, where column generation is not only constrained to the root node, but also stops generation after a certain number of columns have been produced. In Fig. 5, we demonstrate how the marginal improvement of solutions diminishes as more columns are generated. Using ten instances of a 100 customer, 8 port, 8 eVTOL problem (these are distinct from Problem Set B), we record the gap from the best known solution (computed beforehand) as well as the number of columns generated so far. We do the same for ten instances of a 150 customer, 8 port, 8 eVTOL problem (again distinct from Problem Set B). We plot a point for each time the subproblem finishes, reporting the number of columns generated as well as the current gap from the optimal solution (computed beforehand using B&P). We observe that the 100 customer cases find their solutions after fewer columns generated than the 150 customer instances, with medians of 1782.5 and 3495 columns, respectively. Furthermore, we notice the 'tailing off' effect often seen in column generation approaches, with quick early improvement followed by slow incremental progress. We selected the limits of 1000, 3000, and 5000 columns, and denote the resulting heuristics RCG1000, RCG3000, and RCG5000.

Figure 6 presents the optimality gaps and run times for RCG, RCG1000, RCG3000, and RCG5000 on Problem Set A (we use Problem Set A in order to compare with the exact solutions obtained by B&P). We also include run times for B&P. The optimality gaps for the five problem sizes for which B&P could not produce an optimal solution within three hours are calculated with respect to the best known solution. We observe the gap for RCG is very small (mostly zero) while requiring much less time than B&P. Meanwhile, RCG1000, while fast, produces a large optimality gap. As the number of columns produced increases, naturally, the gap goes down, and the run time goes up. Setting the number of columns to 5000 seems to find a good balance.

We noticed that the column generation procedure has a 'tailing off' effect where it generates a large number of columns with reduced cost close to zero (e.g., 10^{-4}) with no improvement in the upper or lower bounds. Consequently, we wanted to experiment with a strategy where we specify a threshold value such that if the maximum reduced cost of the

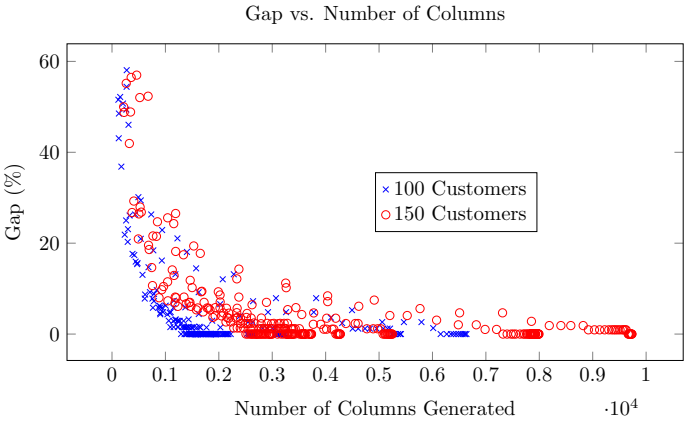

Fig. 5 Gap vs Columns plot. We ran ten instances of an 8 port, 8 eVTOL problem with 100 customers, and ten instances with 150 customers, and, for each instance, after each pass of the subproblem, we recorded the gap from the best known solution as well as the current number of columns generated. The data points from all experiments are plotted here

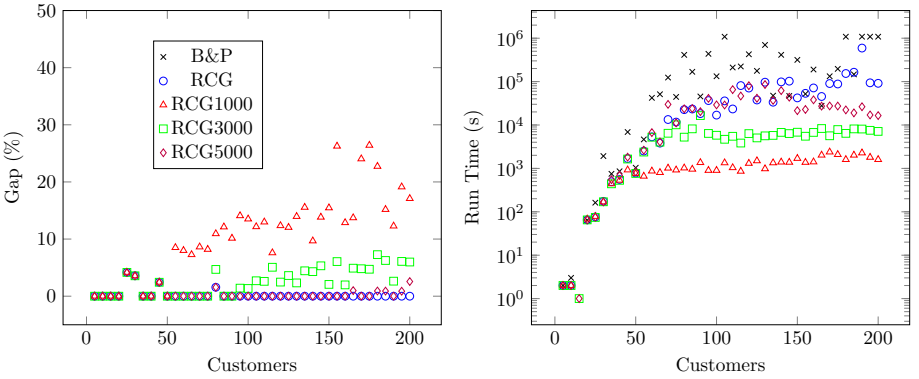

Fig. 6 Left: The gap for each of the column generation based heuristics RCG, RCG1000, RCG3000, and RCG5000 for the 4 port, 4 eVTOL problem (Problem Set A). For problem sizes up to 100 customers, the value is the true optimality gap. For larger problem sizes, as this value is no longer available, the upper bound used to calculate the gap is the best known value (usually found using RCG). Right: Run times for each of the methods, as well as for B&P

latest batch of columns is below the threshold, we stop generating columns. We wanted to apply this both in the exact B&P procedure (effectively determining at what point to treat very small reduced costs as 0), and as a heuristic in RCG. To this end, we experimented with a set of five 150 customer, 8 port, 8 eVTOL problems (again these are distinct from Problem Set B and the instances in Fig. 5). We observed how the value of the maximum reduced cost changed throughout the iterations, as well as observed how the optimality gap changed as a result of the newest set of columns. Figure 7 displays plots of this investigation. The left plot shows the value of the maximum reduced cost of the latest batch of columns found by the subproblem versus the elapsed time. We observe the maximum reduced costs show roughly exponential decay. On the right is a plot of the current optimality gap versus the maximum reduced cost among the latest batch of columns. Together, these plots reveal that, initially,

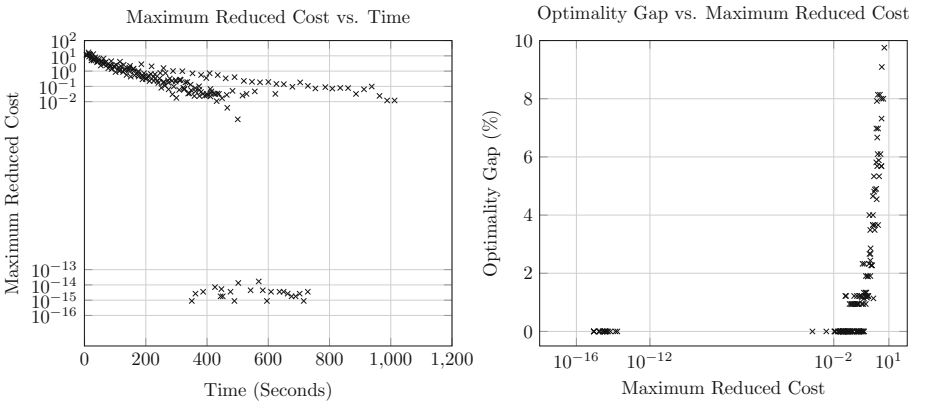

Fig. 7 Maximum reduced cost observations on five instances of an 8 port, 8 eVTOL UAMP with 150 customers. Left: Maximum reduced cost of the paths identified in the most recent iteration of the subproblem vs. the elapsed time. Right: Current gap from the best known solution vs. the maximum reduced cost of the paths identified in the most recent iteration of the subproblem

useful columns are produced, with reduced costs on the order of 10^{-2} to 10^{1}. As time passes, the magnitude of the reduced costs decreases. Our investigation indicates that columns of reduced cost less than 0.01 do not improve the objective value. We also notice that reduced costs can plummet to the order of 10^{-14}, suggesting error due to floating point arithmetic. Two of the five instances spend a significant portion of the computation time identifying these columns of such extremely small reduced cost before terminating (these instances account for the cluster of points on the bottom left of the right plot). Meanwhile, we also observe that once these columns are being produced, the objective value is already at the maximum value it will reach! Thus, we treated reduced costs below 10^{-2}, as being the same as 0. We apply this modification for all procedures involving column generation (including the instances solved using B&P previously).

As a heuristic, we then considered threshold values of 10^{-1} and 10^{0} as a stopping criterion for RCG (rather than setting a limit on the number of columns to generate). We label the heuristics RCG-TH−0.1 and RCG-TH-1, respectively. Figure 8 compares RCG, RCG-TH−0.1 and RCG-TH-1 on Problem Set A. Both RCG-TH−0.1 and RCG-TH-1 show a small optimality gap for all problem sizes. The run times of RCG-TH−0.1 often improve on those of RCG by an order of magnitude, while the run times for RCG-TH-1 are, naturally, even better. The small optimality gap and the significant time savings indicate RCG-TH-1 finds a good balance.

We then investigated the performance of the 'winners' from the previous experimentation (RCG, RCG5000, RCG-TH−0.1, and RCG-TH-1) on the larger data set, Problem Set B. The plot in Fig. 9 reveals that RCG5000 eventually suffers in performance on larger instances. Meanwhile, RCG-TH−0.1 and RCG-TH-1 both provide high-quality solutions remaining close in objective value to the best solution found by RCG across all problem sizes, while yielding significant time savings. In particular, RCG-TH-1 appears to reduce the run time from RCG by an order of magnitude, while remaining within 2% of the best known objective value across all problem sizes.

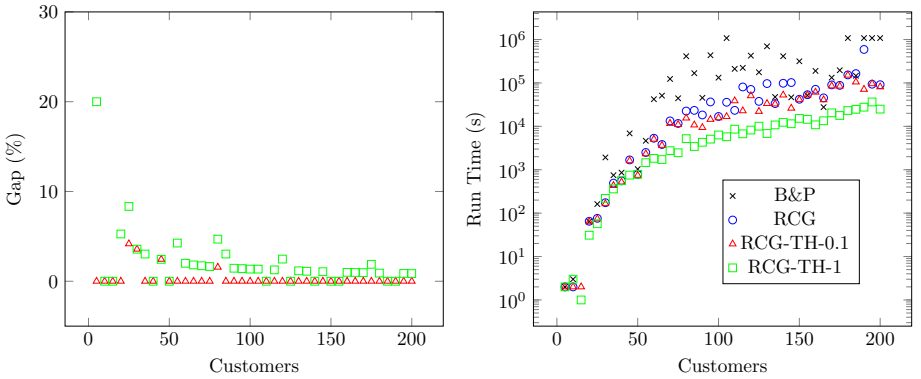

Fig. 8 Left: The gap for each of the column generation based heuristics RCG-TH−0.1 and RCG-TH-1 the 4 port, 4 eVTOL problem (Problem Set A). If the optimal solution is not known, we calculate the gap with respect to the best known value. Right: Run times for each of the methods, as well as for RCG and B&P

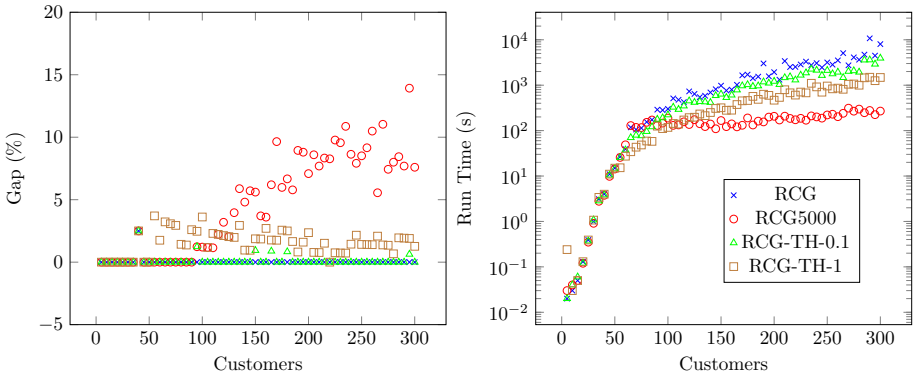

Fig. 9 Left: Gap compared to the best known value for the better performing methods (RCG, RCG5000, RCG-TH−0.1, and RCG-TH-1), on Problem Set B. Right: Run times for the same methods

6.2.3 Sensitivity to time step size

Our model is built on a time-expanded network, where each port is paired with each point in time to create a 'port-time' node. This can be interpreted as modeling the operational practice of eVTOLs leaving only at discrete, regularly spaced times, say every five minutes. Clearly, the finer the discretization of the time horizon (i.e., the shorter the spacing), the larger the resulting network, and, thus, the greater the computational demands. However, the increased flexibility of the denser network can theoretically lead to an improved objective value. In our initial experiments, we chose a time step size of one minute. However, in practice, this level of precision may be too ambitious. It may be unreasonable to expect all parties (eVTOLs and customers) to conform to any schedule right down to the minute, and, thus, in practice, a coarser discretization of time would serve as a buffer for small delays. The computational advantage is clear by observing that merely doubling the time step reduces the number of nodes by half, and similarly at least halves the number of edges in the network.

However, an issue arises when applying customer time window constraints. Because eVTOLs only depart at time steps, if a customer's time window is entirely contained within

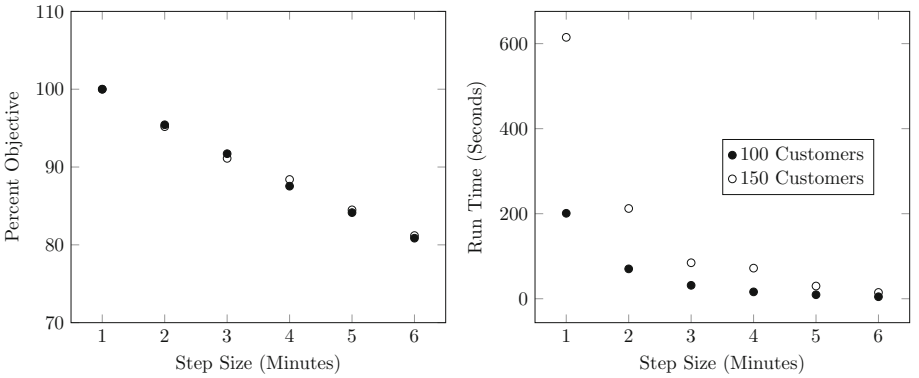

Fig. 10 Left: Average percent objectives when compared to the one-minute time step case over ten instances of a 8 Port, 8 eVTOL problem, for both 100 and 150 customers. We observe a linear loss in objective value as the network loses its flexibility. Right: Average run times for the same experiments. Increasing the time step size appears to decrease the run time exponentially

the width of a time step, the customer is unable to be serviced. For example, if the time horizon is partitioned into five minute intervals, such that possible departure times are 8:00 am, 8:05 am, 8:10 am, etc., a customer whose time window spans 8:06 am to 8:09 am would be impossible to serve. The best way to address this situation depends on the modeling approach.

One approach is to allow the customer to take the first departure after her window ends. For example, we allow the customer in the above example to take the 8:10 am flight. To be consistent, we must allow *all* customers to take the first departure after their windows end. For example, a customer whose window is 8:04 am to 8:09 am will also be allowed to take the 8:05 am or the 8:10 am flights. This artificial extension of time windows is straightforward to implement and seems justifiable from a practical perspective.

We note this approach makes it difficult to compare various discretizations with respect to the objective value. A consequence of allowing for the first departure after a window is that coarser discretizations have longer effective time windows, making the problem easier. Meanwhile, the reduced flexibility limits the space of solutions. The degree to which these effects cancel out one another is difficult to predict.

We conduct tests on the effects of increasing the time step size using this approach. We considered a set of ten 100 and ten 150 customer instances with 8 ports and 8 eVTOLs (these are distinct from the previously discussed instances). For these instances, we varied the time step size from 1 to 6 min, and solved each of the resulting problem instances using RCG. We measured the impact of the network losing flexibility by measuring the percent throughput compared to the one-minute time step case. Figure 10 plots the average percentages and average run times (averages are for the ten instances of a given time step size). We observe a linear decrease in objective percentage as the step size is increased. For example, when the time step is increased to two minutes, the objective value reached is, on average, around 95% of the value reached when the time step is one minute, and that value drops to 90% when the step is increased to three minutes. Meanwhile, the run times appear to decrease exponentially, dropping from 200 s to 70 s when going from one-minute to two-minute time steps. The plots indicate that, if one is willing to tolerate a certain decrease in objective value, the run time can be reduced significantly.

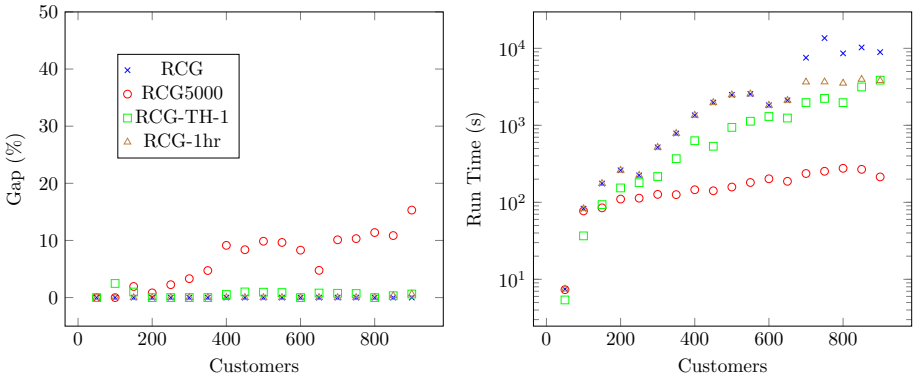

Fig. 11 Left: Gap compared to the best known value for the methods RCG, RCG5000, RCG-TH-1 and RCG-1hr on Problem Set B*, and a time step size of two minutes. Because the time step is larger, more customers can be considered. Right: Run times for the same methods

Increasing the time step substantially reduces computational time, thereby allowing one to address larger problems. To compare the scalability of the different approaches and assess their solution quality we created an augmented Problem Set B*. This data set contains the test instances from Problem Set B with 50, 100, 150, 200, 250, and 300, customers as well as larger 8 port, 8 eVTOL instances starting at 350 customers, going up to 900 customers in increments of 50 customers. Since RCG-TH-1 has a significant advantage over RCG-TH−0.1 in terms of running time without a significant deterioration in objective value, for these large-scale instances in Problem Set B*, we dropped RCG-TH−0.1 from consideration. Instead, we compared RCG, RCG5000, RCG-TH-1, and a new variant of RCG-1hr on Problem Set B*. RCG-1hr limits root node column generation to one hour, and then branches, subsequently. Further, we apply the following heuristic procedure which significantly speeds up the column generation procedures. After each run of the linear master problem, the $|V|$ largest x_r variables are identified. Each of these $|V|$ x_r variables can be set to 1, providing a feasible solution and, thus, a lower bound. We use the best lower bound identified in this way as the starting solution for each run of the linear master problem. We found this provides a significant speed up, especially as the number of columns gets large.

Figure 11 displays the results for RCG, RCG5000, RCG-TH-1, and RCG-1hr on Problem Set B*, using a time step size of two minutes. Notice that we are able to consider larger numbers of customers with the coarsened time step (indicating it could be a productive strategy to scale the approach). For instance, with the one-minute time step, RCG takes over two hours to solve the 300 customer instance, whereas, with the two-minute time step, it takes around 9 min. RCG and RCG-1hr are identical in behavior except for larger instances where RCG generates columns for more than an hour. Since the performance of RCG-1hr dominates RCG-TH-1 with respect to objective value and its run time is limited to (at most) a little over an hour by design, we recommend RCG-1hr as the column generation variant for large scale instances

6.3 Case study: DC metro area

We now transition to exploring the insights possible when applying our algorithms to a real-world problem. In this section, we model the UAMP on the DC metropolitan area and evaluate how various operational parameters affect solution quality.

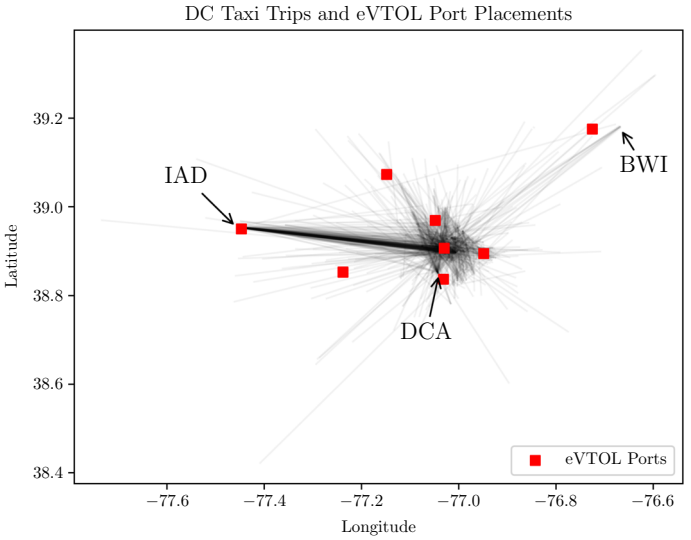

Fig. 12 Visualization of the taxi ridership data provided by the DC Department of For-Hire Vehicles. We filtered the data so as to only include trips longer than 30 min. In the plot, we represent a trip as a thin arc from its origin to its destination. Overlaid on the plot is the resulting placement of eVTOL ports as determined by the k-means clustering algorithm. We see from this plot that airport trips account for a significant portion of long taxi trips, and accordingly, each of the DC area's major airports has an eVTOL port nearby

6.3.1 DC problem set

We introduce a problem set which attempts to model the anticipated real-world setting. The District of Columbia Department of For-Hire Vehicles (DFHV) provides publicly-available taxi ridership records, including origin location, destination location, and trip duration for all taxi trips in a given year. For privacy reasons, departure times are provided only to the nearest hour. We pulled 5000 trips of duration greater than 30 min from this data set from the year 2017. This is because longer trips of this kind will be the most likely to be replaced by eVTOL trips. Then, to place the eVTOL ports, a k-means clustering algorithm (see Steinley, 2006) is run on the resulting set of nodes (that is, all origin and destination nodes). The resulting centers are the port locations. As origins and destinations are given in geographic coordinates, distances between nodes are given by the Haversine great-circle distance formula. A visualization of the taxi data, as well as the resulting port locations, is presented in Fig. 12. The travel time between ports i and j is calculated with $t_{ij} = \lceil d_{ij}/v \rceil + K$, where v is the eVTOL flight speed, and K is a fixed amount of time allotted for takeoff/landing/etc.

To construct a realistic test instance, we create a problem with 8 ports, a fleet of 20 eVTOLs, a set of 1000 customers, over a three hour horizon partitioned into 5 min intervals. Customer origins and destinations are sampled from the 5000 trips from the taxi data set, mapped to their nearest port. Flight speeds are set at 240 kms per hour and time for takeoff/landing set to five minutes. Customer windows are set to be ten minutes long (which corresponds to two time steps, given a step size of five minutes). Requested departure times are spread uniformly over the time horizon (though restricted so that all possible flights arrive at their destination prior to the end of the time horizon). The eVTOL charge depletion rates were set to 5% per five minutes of flight, and recharge rates to 10% per five minutes of charging. At

Table 2 Results on the DC problem, with 8 ports, 20 eVTOLs, and 1000 customers

Problem instance	Greedy solution	RCG solution	Improvement factor
1	440	648	1.47
2	482	636	1.32
3	461	647	1.40
4	491	661	1.35
5	466	640	1.37

Using RCG leads to a 30–50% improvement in solution quality

Table 3 Results on the same DC problems with and without charge constraints

Problem instance	With charge constraints	No charge constraints	Improvement factor
1	648	704	1.09
2	636	690	1.08
3	647	711	1.10
4	661	719	1.09
5	640	687	1.07

If battery constraints could be relaxed, throughput would increase by 7–10%

the beginning of the time horizon, the eVTOLs are at full charge. (Recall we used industry white papers and Polaczyk et al., 2019, Bacchini & Cestino, 2019, Uber, 2018, to help us determine specifications for eVTOLs.)

We ran RCG-1hr on five instances of the above problem.

6.3.2 Results

We now describe several experiments that we conducted on the case study problem to assess the quality of the solutions generated by our path-based approach, and to assess some tradeoffs that managers are likely to be interested in. First, we were interested in the benefit in applying the path-based approach in contrast to a (simpler) greedy strategy. Table 2 presents the objective value found by RCG, with a one hour limit on column generation and a thirty minute limit on branching, compared with the objective values obtained when applying the greedy route generating procedure described in Sect. 5.4. We see that our path-based approach leads to a 30–50% improvement in the solution quality for problems of this size.

Battery management is an important operational restriction in eVTOL logistics. Operators of eVTOLs have an option (though it can be expensive) to swap batteries as needed, instead of waiting for the battery to be charged (if the next trip cannot be performed with the current charge). We wanted to see how much customer throughput can be increased when eVTOL operations are freed from battery recharging constraints. In effect, we are assuming there will always be a replacement battery available to swap at a port when necessary (we note that swapping is only necessary when the next trip of the eVTOL cannot be performed without recharging). We were then interested in how solutions change when the charge constraints are relaxed (thus, allowing a manager to evaluate the tradeoff between the increased throughput and the increased cost of batteries). We ran the same test instances, but changed the charge depletion per time step to zero (effectively removing battery management from the problem). Table 3 compares the resulting objective values. We observe that the objective value can

Table 4 Results on the DC problem with 20 eVTOLs and 8 ports

Number of customers	Throughput	Percentage (%)
500	404	80.8
600	479	79.8
700	532	76.1
800	579	72.4
900	622	69.1
1000	648	64.8

increase by 7–10% if we relax battery constraints. We should note that without battery management (i.e., when we can swap batteries as needed instead of waiting to recharge), we can solve problem instances that are (at least) an order of magnitude larger to optimality—-indicating that the battery management constraints make it difficult to solve large-scale instances to optimality.

Finally, we wanted to understand how the throughput varied as the demand changed with a fixed fleet size. When there are 1000 customers, only about 62% of the requests can be accommodated. We decreased the total number of customers from 1000 to 500 in steps of 100. Table 4, presents the result of RCG on the first instance of the DC problems, as the total number of customers is varied from 500 to 1000. We see that, though the total throughput decreases as the total number of customers is reduced, the proportion of satisfied requests increases.

7 Conclusions

UAM systems have the promise to improve our daily lives by decreasing the time and cost of moving people and goods in and around cities. They have vast implications for the future of transport, work-life, and urban design. An important ingredient in designing an effective and well-utilized UAM system involves the network logistics.

In this paper, we consider a problem that would be encountered by a UAM provider in the early stages of the company's development. Specifically, we focus on determining an a priori schedule for the eVTOLs based on the anticipated demand, with the goal of maximizing the throughput. The temporal nature of the demand, time windows for customers, and battery management constraints of the eVTOLs make the UAMP particularly challenging. We develop a three-index, arc-based formulation, routing eVTOLs over a time-expanded network. Due to the computational limitations of the arc-based formulation, we also develop a path-based formulation, and apply a column generation procedure to it. The path-based column generation procedure can be applied in a heuristic manner by (i) sparsifying the time-expanded network, (ii) limiting column generation to the root node in a branch-and-bound scheme, and (iii) applying early termination criteria in the column generation procedure. Our computational experience on a large set of test instances indicates that the path-based approach can identify high-quality solutions for small to large instances.

There are several directions for future research. They include the synchronization of air and ground transportation. In other words, when a customer request for transport arises and it can be served in a multi-modal fashion—it is necessary to coordinate the ground transportation mode with the 'scheduled eVTOL service'. Another issue is the online problem of determining real-time changes to the scheduled eVTOL service to address significant

changes or variations in demand for service. Related to the network logistics is the issue of pricing of the eVTOL service.

The UAMP discussed in this paper is an important first step in developing our understanding, and providing methods for the industry to use. As the industry evolves—with changes in market demands and technology—additional network logistics problems will arise. Their solutions could play an important role in the path towards making eVTOL transport a commercial reality.

Acknowledgements Eric Oden was partially supported by the University of Maryland Graduate School Faculty-Student Research Award.

Declarations

Conflict of interest The authors have no competing interests to declare that are relevant to the content of this article.

Ethical approval This article does not contain any studies with human participants or animals performed by any of the authors.

Appendix A: Modeling extensions

The three-index formulation described in the paper can be extended to incorporate a variety of realistic modeling goals. We discuss such considerations in this section and the ways in which the model can be adapted to include them.

A.0.1. Multiple port allocation

In our paper, we assume a single customer-port allocation. That is, we assume each customer has a unique origin and destination port, when in reality there may be a choice of such ports available. For instance, if the customer's final destination is sufficiently close to two ports, a customer may accept flights to either one. We extend our definitions in the following way. Suppose there are L possible origin–destination pairs, each with possibly different time windows. We define, for each customer, a set of origin–destination port pairs $\{(o_c^j, d_c^j)\}_{l=1}^L$ and a corresponding set of discrete time windows $\{T_c^j\}_{l=1}^L$. We extend $\mathcal{F}(c)$ to include each of the possible flight arcs for customer c. This will be the collection of arcs $(i, j) \in B$, for which there exists an $l \in \{1, \ldots, L\}$ such that $p_i = o_c^l$, $p_j = d_c^l$, and $t_i \in T_c^l$. Besides this generalization of $\mathcal{F}(c)$, nothing else needs to be changed in MIP3 to account for multiple port allocation.

A.0.2. Soft time windows

In many vehicle routing applications involving time windows, a popular extension is the inclusion of 'soft' time windows. The idea is to relax each time window constraint and instead penalize its violation in the objective function. This relaxation may be desirable in our setting, where a UAM provider may accept a small loss in quality of service in exchange for increased market share.

Such an extension can be achieved in the following way. For each customer $c \in C$, and for each $(i, j) \in \mathcal{F}(c)$, we define a coefficient $w_{ij}^c \in \mathbb{R}$. This coefficient reflects the value of servicing customer c by flying him along arc (i, j). The coefficients can be determined using any model desired. For instance, if $T_c = \{t_c^a, \ldots, t_c^b\}$ is the contiguous, discrete time window of customer c, one simple such definition is:

$$w_{ij}^c = \frac{t_c^b - t_i}{t_c^b - t_c^a} \qquad \forall (i, j) \in \mathcal{F}(c),$$

which linearly decays from 1 to 0 the later the departure time, t_i. The objective function is then modified to

$$\max \sum_r \sum_{\{(i,j)\in\mathcal{F}(r)\}} w_{ij}^c y_{ij}^c.$$

A.0.3. Customer groups

The three-index formulation assumes each customer is traveling individually, when, in reality, passengers may wish to travel together. Groups of customers can be included in the formulation by simply weighting the objective value of the particular customer (now customer group), c, by the number of customers in the group, n_c, changing the objective to

$$\text{Max} \sum_{c\in C} n_c \sum_{\{(i,j)\in\mathcal{F}(c)\}} y_{ij}^c$$

and changing the passenger flow equation to

$$\sum_{\{c|(i,j)\in\mathcal{F}(c)\}} n_c y_{ij}^c \le \sum_k S x_{ij}^k \quad \forall (i, j) \in \mathcal{F},$$

to account for the increase in load factor.

A.0.4. Battery swaps

The model above assumes charge levels evolve according to e_{ij}, defined for each $(i, j) \in B$. Each arc $(i, j) \in B$ corresponds to a possible transition an eVTOL may make in the time expanded network. If $p_i \neq p_j$, then it must be the case that $t_i + t_{p_i,p_j} = t_j$ (that is, the time at node j must be equal to the time at node i, plus the travel time between the two nodes). We allow for eVTOLs to remain at a port by defining the artificial travel time $t_{ii} = 1$ for each $i \in N$. In the case $p_i \neq p_j$, $e_{ij} < 0$, as an eVTOL is depleting its charge level by flying from one port to another. In the case $p_i = p_j$, $e_{ij} \ge 0$, corresponding to a battery charge increase from recharging. As such arcs correspond to a single time step forward, e_{ij} is specifically the charge increase possible in one time step. This can correspond to a recharging policy or a battery swap policy. In the battery swap policy, $e_{ij} = Q_+$, fully replenishing the charge level.

However, it may be the case that the battery swap process takes more than a single time step. We can account for this by adapting our network in the following way. Suppose a battery swap takes X time steps. For each $i \in M$ such that $t_i + X \in T$, add the arc (i, j) to B, where $p_j = p_i$ and $t_j = t_i + X$. These arcs correspond to remaining at a port for X time steps. Then, for each of these arcs, define $e_{ij} = Q_+$. This will allow the possibility of an eVTOL swapping its battery, with such a choice preventing take-offs during the battery swap period. One can include battery swap and recharging options simultaneously.

Appendix B: Two-index heuristic

The charge constraints significantly complicate the problem, and, therefore, limit the problem sizes that can be solved within a reasonable computation time by an MIP solver. In this section, we present an MIP-based heuristic to identify feasible solutions for large problem sizes. We use a two-index formulation of the UAMP with charge constraints relaxed, and then use the output of such a formulation to dramatically sparsify the network prior to re-solving with the three-index formulation.

The charge-relaxed two-index formulation is constructed by aggregating the eVTOL flow variables from MIP3, $x_{ij} = \sum_{k \in K} x_{ij}^k$. The integer-valued variable x_{ij} thus corresponds to the number of eVTOLs traveling along arc (i, j). We have the resulting integer program (MIP2).

$$\max \sum_{c \in C} \sum_{\{(i,j) \in \mathcal{F}(c)\}} y_{ij}^c \tag{26}$$

$$\text{s.t.} \sum_{j \in \delta^+(i)} x_{ij} - \sum_{j \in \delta^-(i)} x_{ji} = \begin{cases} |K| & i = s \\ 0 & i \in M \backslash \{s, f\} \\ -|K| & i = f \end{cases} \quad \forall i \in M \cup \{s, f\} \tag{27}$$

$$\sum_{\{c | (i,j) \in \mathcal{F}(c)\}} y_{ij}^c \leq S x_{ij} \quad \forall (i, j) \in \mathcal{F} \tag{28}$$

$$\sum_{(i,j) \in \mathcal{F}(c)} y_{ij}^c \leq 1 \quad \forall c \in C \tag{29}$$

$$x_{ij} \in \mathbb{Z} \quad \forall i, j \in M \tag{30}$$

$$y_{ij}^c \in \{0, 1\} \quad \forall (i, j) \in \mathcal{F}(r), c \in C \tag{31}$$

The objective (26) maximizes the number of customers that are served. Constraints (27) ensure continuity of flow throughout the network. Constraints (28) limit the number of customers flying along an arc by the number of eVTOLs flying along the same arc multiplied by the eVTOL capacity. Constraints (29) ensure that each customer is served at most once. Constraints (30) ensure integer-valued eVTOL variables, and (31) ensure binary-valued passenger flow variables. By both relaxing charge-constraints and aggregating eVTOL flow variables, MIP2 is extremely efficient, running within seconds on large (e.g., 1000+ customer) problem sizes.

Let R-MIP3 denote MIP3 without the charge constraints. It is clear that any feasible solution of R-MIP3 can be converted into a feasible solution of MIP2. It is straightforward to show that any feasible solution of MIP2 can be converted into a solution of R-MIP3 by disaggregating the integer-valued flow variables x_{ij}. That is, we can identify $x_{ij}^k \in \{0, 1\}$ for all $(i, j) \in B$ and $k \in K$ such that $\sum_k x_{ij}^k = x_{ij}$ and the flow equations hold. Thus, MIP2 provides a set of K throughput-maximizing paths through the network, which may not satisfy charge constraints.

Our sparsification heuristic appeals to the efficiency of MIP2, as well as the throughput-maximizing quality of the arcs in a given solution. The process is as follows. We first solve MIP2, and obtain an optimal solution $x^* = \{x_{ij}^*\}_{(i,j) \in B}$. We then construct a sparsified network B^0. An arc $(i, j) \in B$ is included in B^0 if and only if $x_{ij}^* > 0$ or $p_i = p_j$ and $t_j = t_i + 1$. That is, we only keep the arcs from the solution of MIP2 as well as the arcs corresponding to remaining at a port for consecutive time steps. Keeping these latter arcs ensures charge feasibility. MIP3 can be run on this dramatically sparsified network. We refer

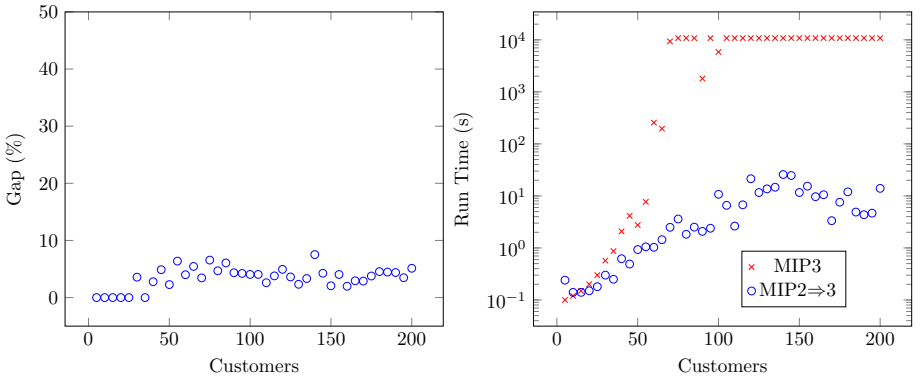

Fig. 13 Left: Optimality gap of MIP2⇒3 for Problem Set A. For sizes greater than 100, where the true optimal solution is unavailable, the best known objective value is used. For all problem sizes, the gap is within 8%. Right: The run times for the exact MIP3 algorithm and the heuristic MIP2⇒3 algorithm are provided. The former explodes, whereas the latter is relatively constant as the number of customers increases

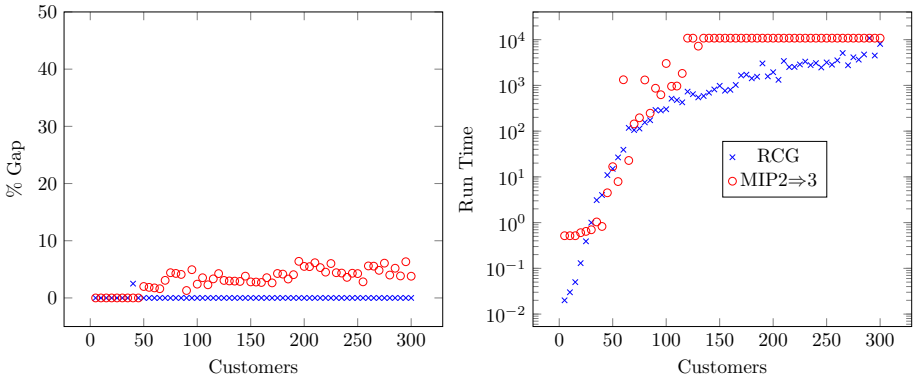

Fig. 14 Left: Optimality gap of MIP2⇒3 on Problem Set B. For all problem sizes, the optimality gap is within 8%. Right: The run times for RCG and the heuristic MIP2⇒3 algorithm

to the process of sparsifying in this fashion, and then running MIP3 on the sparsified network as MIP2⇒3

This heuristic naturally can lead to suboptimal solutions, as the only customer-serving paths in the problem are those produced by MIP2, which may be charge-infeasible. This can be mitigated by, instead of running MIP2 once, running it multiple times, each time cutting off the previously obtained paths. Then, each of the arcs from all the MIP2 runs are included in B, allowing greater system flexibility. However, a balance must be sought, as the more arcs included in B, the longer the computation time.

In Fig. 13, the performance on Problem Set A of MIP2⇒3 is compared to that of the exact MIP3. We observe the solution quality is always within 8% of the optimal solution (or the best known solution, if the optimal solution is unavailable). Meanwhile, the run time of MIP2⇒3 is quite insensitive to increases in the number of customers, whereas MIP3 explodes in run time.

However, MIP2⇒3 falls short when compared to the path-based heuristics. In Fig. 14, MIP2⇒3 is compared to RCG, using Problem Set B. We see that (with one exception), RCG produces better solutions in less time, particularly as the number of customers increases.

Appendix C: Sparsification

We may prove that, with certain assumptions, the sparsification procedure does not cut off the optimal solution.

Theorem 1 *Suppose $Q_+ = \infty$, the distances between ports satisfy the triangle inequality, and that the change in charge level due to recharging is convex with respect to the current charge level. Then, the optimal solution to the UAMP has the same objective value as the optimal solution to the UAMP restricted to the stay-put arcs, anticipatory arcs, and flight arcs.*

Proof Let $\mathbf{r} = \{r_1, \ldots, r_n\}$ be an optimal path for the UAMP, where $r_i \in M \; \forall i \in \{1, \ldots, n\}$, and $(r_i, r_{i+1}) \in B \; \forall i \in \{1, \ldots, n-1\}$. We shall construct a charge-feasible path $\mathbf{r}' = \{r'_1, \ldots, r'_m\}$, where $r'_i \in M \; \forall i \in \{1, \ldots, n\}$, that serves the same customers as $\mathbf{r}$, and is such that $(r'_i, r'_{i+1}) \in B' \; \forall i \in \{1, \ldots, n-1\}$.

The construction is as follows. Initialize $\mathbf{r}' = \{r_1\}$. Starting from $i = 1$, we determine if $(r_i, r_{i+1}) \in B'$. If so, append r_{i+1} to $\mathbf{r}'$. Suppose $(r_i, r_{i+1}) \notin B'$. Since $(r_i, r_{i+1}) \notin B'$, the arc is neither a flight arc, a stay put arc, nor an anticipatory arc. Since the distances between ports satisfy the triangle inequality, we may assume, without loss of generality, that there is some $k \in \mathbb{Z}^+$ such that $\{(r_{i+1}, r_{i+2}), \ldots, (r_{i+k}, r_{i+k+1})\}$ is a sequence of stay put arcs, and that $\{r_{i+k}, r_{i+k+1})$ is a flight arc. Let $r_i^j \in M$ denote the node satisfying $p_{r_i^j} = p_{r_i}$ and $t_{r_i^j} = t_{r_i} + j$. That is, r_i^j is the port-time node at the same port as r_i, j time steps ahead. We may observe that rather than following the sequence $\{r_i, r_{i+1}, r_{i+2}, \ldots, r_{i+k-1}, r_{i+k}\}$, we may follow the sequence $\{r_i, r_i^1, r_i^2, \ldots, r_i^k, r_{i+k}\}$, consisting of only stay put arcs and an anticipatory arc. Since the charge level is convex with respect to the current charge level, and since $Q_+ = \infty$, the charge level upon arrival at r_{i+k} along the second sequence will be no less than along the first sequence. We therefore can append the nodes $r_i^1, \ldots, r_i^k, r_{i+k}, r_{i+k+1}$ to $\mathbf{r}'$, and continue from $i = i + k + 1$. We continue this until reaching $i = n - 1$, in which case $(r_{n-1}, r_n\}$ is, without loss of generality, either a stay put arc or a flight arc, and thus in B', so r_n may be appended to $\mathbf{r}'$. Since $\mathbf{r}$ is charge-feasible, $\mathbf{r}'$ is charge-feasible, and furthermore, $\mathbf{r}'$ serves the same customers as the path $\mathbf{r}$. □

We demonstrate we need all anticipatory arcs, rather than just the 'extreme' anticipatory arcs, in order for the sparsified network to keep the optimal solution by means of an example instance of the UAMP, visualized in Fig. 15. In the instance, there is a single eVTOL, and three customers. Passenger 1 wishes to depart from Port 1 towards Port 2 at time 0. Passenger 2 wishes to depart from Port 3 towards Port 4 at either time 5, 6, or 7. Passenger 3 wishes to depart from Port 4 towards Port 5 at time 8. Assuming it is charge feasible, the optimal solution is to fly Passenger 1 from Port 1 to Port 2 at time 0 (arriving at time 3), then to immediately dead head from Port 2 to Port 3. From there, we immediately bring Passenger 2 from Port 3 to Port 4 (departing at time 6), and then immediately take Passenger 3 from Port 4 to Port 5. We note that this sequence is the only way to serve each of the customers, and that it requires a non-extreme anticipatory arc (the one connecting Port 2 to Port 3, departing at time 3). Thus we cannot delete such arcs from the network and guarantee that we do not cut off optimal solutions.

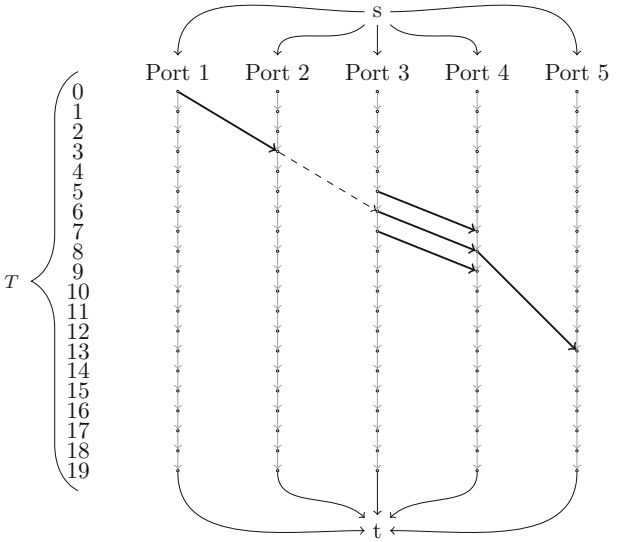

Fig. 15 Example instance of the UAMP demonstrating we must include all anticipatory arcs, rather than just 'extreme' anticipatory arcs

Appendix D: Load factor analysis

Of particular relevance to a UAM provider may be the average flight occupancy, or load factor. In particular, there may be interest in the proportion of 'deadheads' (flights in the solution in which there are no passengers), flights with a single customer, two customers, and so on. We selected parameters to establish a crowded scenario. In particular, the number of ports was set to four, the number of eVTOLs to three, and the number of time steps to 25. We then were able to quickly solve 500-customer problems. Furthermore, by reducing the time horizon in this fashion, we significantly increase the density of customer flight requests. There are roughly $t_{max}\binom{|P|}{2}$ possible flight paths, corresponding to every origin–destination pair and every flight time. Thus, when we restrict the number of ports to four and the number of time steps to 25, there are at most 150 possible paths (fewer if it is infeasible to get from any one particular port to another in one time step). If customer requests are distributed uniformly across the possible requests, each with a wait time of three time steps, then varying the number of customers from 50 to 500 corresponds to an expected number of passengers requesting a particular arc varying from roughly 1 to 10.

In this setting, we observed how the load factor of the eVTOLs and the total system throughput varies as we increase the density of customer requests. In Fig. 16, we present the average load factor for four different parameter settings: setting the number of eVTOLs to 3 and 4 and setting the capacities of the eVTOLs to 5 and 6.

An important metric for a UAM provider might also be the eVTOL flight time per customer (that is, the total time spent flying divided by the number of customers served). One of the principal costs for a UAM provider might be the recharging of its fleet, and providing service might only be profitable if the flight time per customer is below a certain threshold. In Fig. 17, we plot the average flight time per customer for the four different parameter settings versus the number of customers (corresponding to customer request density). We observe a clear delineation between the different capacity levels: When the capacity is increased from 5 to 6,

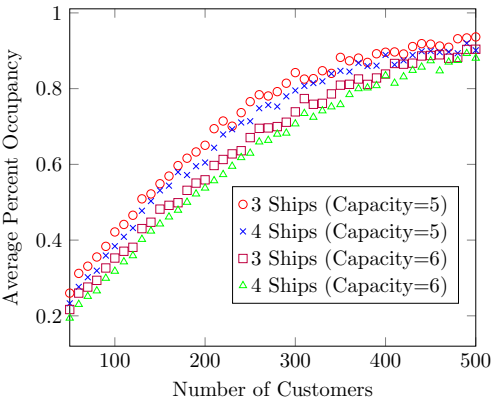

Fig. 16 Average percent load factor for the four different parameter settings. While all parameter settings appear to have similar results in the extreme cases (highly sparse and highly dense customer requests, respectively), the lower capacity configurations climb slightly faster towards full occupancy

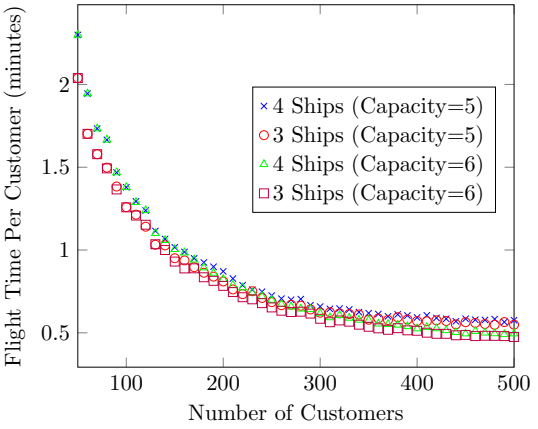

Fig. 17 The average flight time per customer for varying parameter settings across a varying number of customers (corresponding to customer request density). In the higher density setting, a marked improvement (a decrease in flight time per customer) emerges for the higher capacity parameter settings

there is a decrease in flight time per customer. Meanwhile, increasing the fleet size increases the flight time per customer slightly, as the chances of flying at less than full load factor decrease when more eVTOLs are available.

Tables corresponding to computational results in main paper

Tables 5, 6, 7, 8, 9 and 10 present the detailed computational results for the various algorithms discussed in the paper.

Table 5 Run times (in s) for the various methods on Problem Set A (4 ports, 4 eVTOLs)

$\|C\|$	MIP3	B&P	MIP2⇒3	RCG	RCG1000	RCG5000	RCG-TH−0.1	RCG-TH-1
5	0.10	0.02	0.24	0.02	0.03	0.02	0.12	0.01
10	0.12	0.03	0.14	0.03	0.03	0.03	0.02	0.02
15	0.15	0.00	0.14	0.01	0.01	0.01	0.00	0.01
20	0.20	0.64	0.15	0.76	0.76	0.81	0.65	0.24
25	0.30	1.62	0.18	0.83	0.86	0.89	0.75	0.45
30	0.57	19.21	0.30	1.95	2.31	2.07	1.70	1.70
35	0.87	7.49	0.25	6.33	5.76	5.54	4.24	3.51
40	2.08	8.57	0.62	7.14	6.39	6.60	5.48	5.44
45	4.13	68.85	0.49	23.21	12.67	21.60	17.69	6.92
50	2.76	10.34	0.93	9.99	10.84	10.52	8.67	5.26
55	7.69	46.44	1.05	26.65	9.01	28.89	23.23	14.52
60	256.53	421.73	1.03	48.99	12.41	53.12	40.79	28.62
65	196.29	510.12	1.44	59.28	11.09	45.23	34.08	19.37
70	9331.85	1237.34	2.49	213.70	13.13	194.60	95.90	28.63
75	10,800.00	442.87	3.60	152.88	11.39	97.23	93.33	24.22
80	10,800.00	4124.29	1.83	265.44	13.35	183.63	95.45	44.49
85	10,800.00	1678.56	2.51	211.46	13.38	112.65	94.23	33.38
90	1802.00	453.70	2.08	390.62	16.90	155.37	89.80	41.40
95	10,800.00	4350.80	2.39	565.81	11.06	390.67	114.31	49.59
100	5859.01	1324.05	10.74	678.49	11.00	346.09	174.26	63.68

Table 5 continued

$\|C\|$	MIP3	B&P	MIP2⇒3	RCG	RCG1000	RCG5000	RCG-TH−0.1	RCG-TH-1
105	10,800.00	10,800.00	6.59	350.01	16.51	201.85	151.96	65.25
110	10,800.00	2108.75	2.63	758.20	12.54	420.80	171.61	71.99
115	10,800.00	2227.74	6.74	472.44	10.45	432.99	187.21	62.81
120	10,800.00	4251.86	21.37	450.73	16.59	531.97	389.80	82.36
125	10,800.00	1761.92	11.61	1383.51	20.54	591.81	301.59	112.10
130	10,800.00	6979.26	13.65	446.69	13.39	595.59	327.11	87.00
135	10,800.00	473.77	14.68	416.73	17.55	326.19	471.23	82.25
140	10,800.00	4127.90	25.78	1012.70	19.62	411.56	328.45	110.07
145	10,800.00	465.69	24.67	702.39	17.28	981.10	894.33	117.06
150	10,800.00	3154.59	11.63	434.50	23.00	296.18	404.18	198.66
155	10,800.00	518.08	15.39	499.90	17.60	270.52	448.89	184.74
160	10,800.00	1892.63	9.66	479.97	18.90	308.77	437.32	108.63
165	10,800.00	276.66	10.53	298.51	23.13	358.18	279.36	137.36
170	10,800.00	1323.80	3.34	774.97	21.01	381.62	742.45	207.75
175	10,800.00	1958.68	7.57	1151.33	20.38	284.00	1081.95	235.68
180	10,800.00	10,800.00	11.96	1701.53	18.48	200.04	1343.20	425.97
185	10,800.00	1461.04	4.89	998.20	23.01	180.84	935.21	332.32
190	10,800.00	10,800.00	4.35	9308.63	22.61	241.64	8997.49	329.72
195	10,800.00	10,800.00	4.66	668.65	21.79	144.53	757.74	428.56
200	10,800.00	10,800.00	13.96	1248.66	21.78	153.82	733.00	298.59

A three hour time limit is imposed

Table 6 Objective values for each of the algorithms, given the three hour time limit, on Problem Set A

$\|C\|$	MIP3	B&P	MIP2⇒3	RCG	RCG1000	RCG5000	RCG-TH−0.1	RCG-TH-1
5	**5**	**5**	**5**	**5**	**5**	**5**	**5**	4
10	**10**	**10**	**10**	**10**	**10**	**10**	**10**	**10**
15	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**
20	**19**	**19**	**19**	**19**	**19**	**19**	**19**	18
25	**24**	**24**	**24**	23	23	23	23	22
30	**28**	**28**	27	27	27	27	27	27
35	**33**	**33**	**33**	**33**	**33**	**33**	**33**	32
40	**36**	**36**	35	**36**	**36**	**36**	**36**	**36**
45	**41**	**41**	39	40	40	40	40	40
50	**44**	**44**	43	**44**	**44**	**44**	**44**	**44**
55	**47**	**47**	44	**47**	43	**47**	**47**	45
60	**50**	**50**	48	**50**	46	**50**	**50**	49
65	**55**	**55**	52	**55**	51	**55**	**55**	54
70	**58**	**58**	56	**58**	53	**58**	**58**	57
75	(61)[62]	**61**	57	**61**	56	**61**	**61**	60
80	(64)[65]	**64**	61	63	57	63	63	61
85	(66)[67]	**66**	62	**66**	58	**66**	**66**	64
90	69	**69**	66	**69**	62	**69**	**69**	68
95	(71)[72]	**71**	68	**71**	61	**71**	**71**	70
100	74	**74**	71	**74**	64	**74**	**74**	73

Table 6 continued

$\|C\|$	MIP3	B&P	MIP2⇒3	RCG	RCG1000	RCG5000	RCG-TH−0.1	RCG-TH-1
105	(**74**)[76]	(**74**)	71	**74**	65	**74**	**74**	73
110	(**77**)[79]	**77**	75	**77**	67	**77**	**77**	**77**
115	(77)[83]	**79**	76	**79**	73	**79**	**79**	78
120	(**81**)[85]	**81**	77	**81**	71	**81**	**81**	79
125	(83)[88]	**84**	80	**84**	73	83	83	83
130	(85)[91]	**86**	84	**86**	74	**86**	**86**	85
135	(89)[94]	**90**	87	**90**	76	**90**	**90**	89
140	(91)[97]	**93**	86	**93**	84	**93**	**93**	**93**
145	(93)[99]	**94**	90	**94**	81	**94**	**94**	93
150	(97)[103]	**98**	95	97	82	97	97	97
155	(97)[104]	**100**	95	99	73	99	99	99
160	(**102**)[106]	**102**	99	101	88	101	101	100
165	(101)[106]	**102**	99	**102**	88	101	**102**	101
170	(103)[108]	**104**	101	**104**	79	**104**	**104**	103
175	(105)[110]	**106**	102	**106**	78	**106**	**106**	104
180	(109)[114]	(**110**)	105	**110**	85	109	**110**	109
185	(**112**)[116]	**112**	107	**112**	95	111	**112**	**112**
190	(113)[118]	(**114**)	109	**114**	100	**114**	**114**	**114**
195	(**116**)[121]	(115)	111	115	93	114	115	114
200	(**117**)[124]	(**117**)	111	**117**	97	114	**117**	116

Best values are bolded. For the exact methods (MIP3 and B&P), values in parenthesis correspond to the best feasible solution found after three hours. For MIP3, the values in brackets correspond to the lowest upper bound after three hours

Table 7 Run times for the better performing heuristics on the 8 port, 8 eVTOL problem (Problem Set B), given a three hour time limit

$\|C\|$	MIP2⇒3	RCG	RCG-TH−0.1	RCG-TH-1	RCG5000
50	16.71	14.99	14.36	14.90	14.26
55	7.90	26.63	25.52	15.22	25.71
60	1329.88	39.16	37.46	27.33	48.65
65	22.83	117.97	70.01	34.27	128.95
70	143.82	105.54	79.23	43.19	117.59
75	195.88	113.87	76.97	48.29	116.46
80	1316.59	155.29	95.66	59.01	152.20
85	246.78	173.16	121.40	57.93	171.38
90	867.10	289.71	177.44	125.28	130.00
95	625.37	282.65	190.96	110.72	146.56
100	3018.94	301.34	222.02	117.55	188.68
105	954.91	512.73	327.45	143.31	130.40
110	968.78	477.10	293.19	139.02	133.44
115	1828.50	426.10	348.62	171.46	156.00
120	10, 800	731.29	450.96	192.95	136.26
125	10, 800	643.29	418.90	197.04	174.08
130	7223.39	546.30	437.25	232.38	141.74
135	10, 800	586.42	428.78	213.42	122.71
140	10, 800	693.43	416.50	250.17	139.83
145	10, 800	814.92	589.67	307.64	109.76
150	10, 800	979.33	634.90	318.43	166.60
155	10, 800	764.67	535.62	280.33	122.35
160	10, 800	800.13	606.47	281.82	140.54
165	10, 800	1021.26	786.12	375.09	121.08
170	10, 800	1649.21	967.40	452.61	131.14
175	10, 800	1716.01	1026.88	460.01	189.94
180	10, 800	1426.21	938.63	566.89	132.92
185	10, 800	1546.70	964.29	487.61	161.67
190	10, 800	3018.32	1179.87	632.71	155.45
195	10, 800	1572.06	1110.85	596.91	198.65
200	10, 800	1941.76	1215.70	467.46	206.68
205	10, 800	1321.91	1035.66	537.96	171.20
210	10, 800	3407.82	1462.72	794.22	207.87
215	10, 800	2511.90	1530.85	679.69	185.31
220	10, 800	2530.22	1304.18	605.49	173.17
225	10, 800	2866.63	1663.74	705.77	186.45
230	10, 800	3312.65	1868.20	677.83	169.47
235	10, 800	2774.25	2252.59	1100.95	210.71
240	10, 800	3065.07	2136.53	917.93	200.28
245	10, 800	2458.76	1640.78	701.38	188.20
250	10, 800	3175.04	2112.46	958.49	221.19

Table 7 continued

$\|C\|$	MIP2⇒3	RCG	RCG-TH−0.1	RCG-TH-1	RCG5000
255	10, 800	2838.78	1777.94	831.14	229.44
260	10, 800	3525.95	1936.10	861.40	202.58
265	10, 800	5087.25	1471.67	812.28	240.54
270	10, 800	2746.22	2016.43	1048.99	312.02
275	10, 800	4108.26	2077.59	1062.55	267.96
280	10, 800	3650.89	1937.69	998.77	297.42
285	10, 800	4728.99	3585.57	1453.53	247.58
290	10, 800	10775.53	3606.87	1495.18	277.19
295	10, 800	4486.52	2906.29	1257.12	224.44
300	10, 800	8045.45	3939.81	1477.68	269.14

Table 8 Objective values for the better-performing heuristics on the 8 port, 8 eVTOL problem (Problem Set B), given a three hour time limit

$\|C\|$	MIP2⇒3	RCG	RCG-TH−0.1	RCG-TH-1	RCG5000
50	49	**50**	**50**	**50**	**50**
55	53	**54**	**54**	52	**54**
60	56	**57**	**57**	56	**57**
65	61	**62**	**62**	60	**62**
70	63	**65**	**65**	63	**65**
75	65	**68**	**68**	66	**68**
80	67	**70**	**70**	69	**70**
85	70	**73**	**73**	72	**73**
90	76	**77**	**77**	75	**77**
95	77	**81**	80	79	80
100	81	**83**	**83**	80	82
105	82	**85**	**85**	84	84
110	85	**87**	**87**	85	86
115	87	**90**	**90**	88	88
120	90	**94**	**94**	92	91
125	95	**98**	**98**	96	96
130	98	**101**	**101**	99	97
135	99	**102**	**102**	99	96
140	101	**104**	**104**	103	99
145	101	**105**	**105**	104	99
150	104	**107**	106	105	101
155	105	**108**	**108**	106	104
160	108	**111**	**111**	108	107
165	109	**113**	112	111	106
170	111	**114**	**114**	112	103
175	112	**117**	**117**	114	110
180	115	**120**	119	117	112

Table 8 continued

$\|C\|$	MIP2⇒3	RCG	RCG-TH−0.1	RCG-TH-1	RCG5000
185	117	**121**	**121**	119	114
190	118	**123**	**123**	122	112
195	117	**125**	**125**	123	114
200	120	**127**	**127**	124	118
205	121	**128**	**128**	127	117
210	122	**130**	**130**	129	120
215	125	**132**	**132**	130	121
220	127	**133**	**133**	133	122
225	125	**133**	**133**	132	120
230	130	**136**	**136**	135	123
235	132	**138**	**138**	137	123
240	134	**139**	**139**	137	127
245	133	**139**	**139**	136	128
250	135	**141**	**141**	139	129
255	138	**142**	**142**	140	129
260	135	**143**	**143**	141	128
265	136	**144**	**144**	141	136
270	138	**145**	**145**	143	129
275	139	**148**	**148**	146	137
280	144	**150**	**150**	149	138
285	146	**154**	**154**	151	141
290	150	**156**	**156**	153	144
295	148	**158**	157	155	136
300	152	**158**	**158**	156	146

Best values are bolded

Table 9 Run times for the better performing heuristics on the 8 port, 8 eVTOL problem (Problem Set B*), using a time step size of two minutes

$\|C\|$	RCG	RCG-TH-1	RCG5000	RCG-1hr
50	7.47	5.40	7.34	7.47
100	82.71	36.68	77.36	82.71
150	175.50	93.45	84.80	175.50
200	260.35	153.83	110.27	260.35
250	223.84	180.04	113.25	223.84
300	517.36	216.42	126.83	517.36
350	783.61	368.97	125.93	783.61
400	1348.88	631.26	145.69	1348.88
450	1977.14	534.88	141.25	1977.14
500	2489.50	939.86	157.97	2489.50
550	2553.20	1129.97	180.98	2553.20
600	1820.28	1301.80	202.19	1820.28
650	2111.24	1242.99	187.52	2111.24

Table 9 continued

$\|C\|$	RCG	RCG-TH-1	RCG5000	RCG-1hr
700	7551.02	1975.04	236.80	3652.00
750	13575.44	2235.51	253.19	3662.51
800	8595.37	1972.04	276.69	3625.09
850	10262.08	3150.56	268.09	3954.77
900	8919.09	3861.25	213.83	3813.95

Table 10 Objective values for the better-performing heuristics on the 8 port, 8 eVTOL problem (Problem Set B*), using a time step size of two minutes

$\|C\|$	RCG	RCG-TH-1	RCG5000	RCG-1hr
50	**48**	**48**	**48**	**48**
100	**81**	79	**81**	**81**
150	**103**	102	101	**103**
200	**121**	**121**	120	**121**
250	**133**	**133**	130	**133**
300	**151**	**151**	146	**151**
350	**169**	**169**	161	**169**
400	**186**	185	169	**186**
450	**203**	201	186	**203**
500	**213**	211	192	**213**
550	**228**	226	206	**228**
600	**241**	**241**	221	**241**
650	**252**	250	242	**252**
700	**267**	265	240	**267**
750	**281**	279	252	**281**
800	**290**	**290**	257	**290**
850	**303**	302	271	**303**
900	**320**	318	271	318

Best values are bolded

References

Ahuja, R,K., Magnanti, T.L., & Orlin, J.B. (1993). *Network flows: Theory, algorithms and applications*. Pearson.

Bacchini, A., & Cestino, E. (2019). Electric VTOL configurations comparison. *Aerospace, 6*(26), 1.

Barnhart, C., Johnson, E. L., Nemhauser, G. L., Savelsbergh, M. W. P., & Vance, P. H. (1998). Branch-and-price: Column generation for solving huge integer programs. *Operations Research, 46*(3), 316–329.

Boland, N., Hewitt, M., Marshall, L., & Savelsbergh, M. (2018). The price of discretizing time: A study in service network design. *EURO Journal on Transportation and Logistics, 8*, 195–216.

Bongiovanni, C., Kaspi, M., & Geroliminis, N. (2019). The electric autonomous dial-a-ride problem. *Transportation Research Part B: Methodological, 122*, 436–456.

Bsaybes, S., Quilliot, A., & Wagler, A. K. (2019). Fleet management for autonomous vehicles using flows in time-expanded networks. *TOP, 27*(2), 288–311.

Chen, L., Wandelt, S., Dai, W., & Sun, X. (2022). Scalable vertiport hub location selection for air taxi operations in a metropolitan region. *INFORMS Journal on Computing, 34*(2), 834–856.

Garrow, L.A., Mokhtarian, P., German. B., & Boddupalli, S. S. (2020). Commuting in the age of the Jetsons: A market segmentation analysis of autonomous ground vehicles and air taxis in five large U.S. cities.

Genikomsakis, K. N., & Mitrentsis, G. (2017). A computationally efficient simulation model for estimating energy consumption of electric vehicles in the context of route planning applications. *Transportation Research Part D: Transport and Environment, 50*, 98–118.
Goeke, D., & Schneider, M. (2015). Routing a mixed fleet of electric and conventional vehicles. *European Journal of Operational Research, 245*(1), 81–99.
Gschwind, T., & Irnich, S. (2014). Effective handling of dynamic time windows and its application to solving the dial-a-ride problem. *Transportation Science, 49*(2), 335–354.
Hawkins, A. J. (2019). Electric air taxi startup Lilium completes first test of its new five-seater aircraft. https://www.theverge.com/2019/5/16/18625088/lilium-jet-test-flight-electric-aircraft-flying-car.
Hendrik, J. (2020). Pioneering the urban air taxi revolution. Technical report, Volocopter. https://press.volocopter.com/images/pdf/Volocopter-WhitePaper-1-0.pdf.
Hill, C., & Garrow, L. A. (2021). A market segmentation analysis for an eVTOL air taxi shuttle.
Ho, S. C., Szeto, W., Kuo, Y. H., Leung, J. M., Petering, M., & Tou, T. W. (2018). A survey of dial-a-ride problems: Literature review and recent developments. *Transportation Research Part B: Methodological, 111*, 395–421.
Holden, J., & Goel, N. (2016). Fast-forwarding to a future of on-demand urban air transportation. Technical report, Uber. https://www.uber.com/elevate.pdf.
Irnich, S., & Desaulniers, G. (2006). Shortest path problems with resource constraints. In Column generation (pp. 33–65). Springer.
Lim, E., & Hwang, H. (2019). The selection of vertiport location for on-demand mobility and its application to Seoul metro area. *International Journal of Aeronautical and Space Sciences, 20*(1), 260–272.
Marshall, L., Boland, N., Savelsbergh, M., & Hewitt, M. (2021). Interval-based dynamic discretization discovery for solving the continuous-time service network design problem. *Transportation Science, 55*(1), 29–51.
Masmoudi, M. A., Hosny, M., Demir, E., Genikomsakis, K. N., & Cheikhrouhou, N. (2018). The dial-a-ride problem with electric vehicles and battery swapping stations. *Transportation Research Part E: Logistics and Transportation Review, 118*, 392–420.
Pelegrin, M., & D'Ambrosio, C. (2022). Aircraft deconfliction via mathematical programming: Review and insights. *Transportation Science, 56*(1), 118–140.
Pelletier, S., Jabali, O., & Laporte, G. (2018). Charge scheduling for electric freight vehicles. *Transportation Research Part B: Methodological, 115*, 246–269.
Polaczyk, N., Trombino, E., Wei, P., & Mitici, M. (2019). A review of current technology and research in urban on-demand air mobility applications. In *Proceedings of the vertical flight society autonomous VTOL technical meeting and electric VTOL symposium*.
Steinley, D. (2006). K-means clustering: A half-century synthesis. *British Journal of Mathematical and Statistical Psychology, 59*(1), 1–34.
Stone, M. (2018). 'Uber-for-helicopters' startup Blade just raised $38 million - here's what it's like to fly to the Hamptons. https://www.businessinsider.com/what-blade-the-uber-for-helicopters-is-like-2017-5.
Tang, H., Zhang, Y., Mohmoodian, V., & Charkhgard, H. (2021). Automated flight planning for high-density urban air mobility. *Transportation Research C, 131*, 103324.
Uber (2018). Uberair vehicle requirements and missions. https://s3.amazonaws.com/uber-static/elevate/Summary+Mission+and+Requirements.pdf.
Wang, K., Jacquillat, A., & Vaze, V. (2022). Vertiport planning for urban aerial mobility: An adaptive discretization approach. *Manufacturing and Service Operations Management, 24*(6), 2797–3306.
Wu, Z., & Zhang, Y. (2021). Integrated network design and demand forecast for on-demand urban air mobility. *Engineering, 7*, 473–487.
Xu, H. (2020). *The future of transportation: White paper on urban air mobility systems*. Technical report, EHang. https://www.ehang.com/app/en/EHang%20White%20Paper%20on%20Urban%20Air%20Mobility%20Systems.pdf.