Profit Maximization for UAV Trajectory Planning in Time-Constrained Data Collection

Hung-An Kuo†, Jang-Ping Sheu†, and Nguyen Van Cuong ‡
†Department of Computer Science, National Tsing Hua University, Taiwan
‡Insititute of Communications Engineering, National Tsing Hua University, Taiwan
Emails: andykuo8766@gapp.nthu.edu.tw, sheujp@cs.nthu.edu.tw, cuongnv@gapp.nthu.edu.tw

***Abstract*—In this work, we use unmanned aerial vehicles (UAVs) to collect data from IoT devices on the ground. Each device has an amount of data that can be sent to the UAV during a specific time window. Our objective is to maximize the total profit that the UAV can collect the data from the IoT devices. Since the problem is NP-hard, we propose a heuristic algorithm in three stages to solve the problem. We solve the traveling-salesman problem (TSP) in the first stage to find the UAV's flying trajectory. In the second stage, we propose an algorithm to change the visiting order or remove IoT devices from the flying trajectory if we cannot satisfy their time constraints. In the third stage, we improve the UAV's flying distance established in the second stage. The simulation results show that the proposed algorithms outperform some baselines in terms of total profit and execution time.**

***Index Terms*—UAV trajectory, time-constrained data collection, profit maximization.**

I. Introduction

Unmanned aerial vehicles (UAVs) have been widely adopted in aerial delivery, surveillance, monitoring, disaster rescue, and remote sensing due to their inherent advantages, such as mobility, agility, flexibility, and adjustable altitudes [1], [2]. The UAVs can be used as aerial base stations (BSs) to provide reliable, cost-effective, and on-demand wireless communication service to desired areas [1]. Especially using the UAVs to harvest sensing data from distributed sensor nodes (SNs) is promoting the future Internet of Things (IoTs) [2]. However, IoT devices are made up of small and low energy consumption battery-limited sensors to periodically collect environmental information such as wind speed, temperature, and particulate matter 2.5 micrometers (PM2.5) [3]. To save energy, IoT devices often use low power to transmit data in a short period. On the other hand, IoT devices are typically deployed in remote areas where terrestrial communications networks are unavailable. In such a scenario, using the UAV as a data collector appears as a promising solution. The UAV may periodically fly close to the IoT devices and receive the data over a single hop communications [4].

Trajectory planning is essential while employing the UAV as a data collector in IoT applications [5]–[10]. For example, optimizing resource allocation and trajectory scheduling in UAV-assisted mobile edge computing (MEC) networks can provide computation or energy-efficient mobile services [5]. In [6], the authors studied the deployment of UAVs for IoT computation offloading. The objective was to minimize the completion time and energy consumption by optimizing the computation offloading and resource allocation, and UAV trajectory. In [7], the author considered the sum power minimization problem by optimizing user association, power control, computation capacity allocation, and trajectory planning in a MEC network assisted by multiple UAVs. Aiming to mobile edge computing, [8] maximized the energy efficiency by optimizing allocation of computation load, the trajectory of IAV, and the transmit power of users. In [10], the UAV roamed around the target area to offload the computation tasks sent from user equipment (UEs) or relay to the access point (AP) for further offloading. The weighted sum energy consumption of the UAV and UEs was optimized by jointly determining the task constraints, the bandwidth allocation, and the trajectory of the UAV. Note that the above works did not consider the deadline requirement of the tasks (data collection/computation offloading).

Trajectory optimization of UAV for time-constrained data collection has been investigated in several works [11], [12]. In [11], the authors optimized the trajectory of UAV and the radio resource allocation while imposing a deadline on data packets that need to be collected from the IoT devices. The objective was to maximize the number of devices that can be served within a given mission time. They proposed a solution with the branch, reduce, and bound (BRB) algorithm to solve the problem in small-scale networks and a solution based on successive convex approximation method in large-scale networks. In [12], they optimized the data collection deployment costs for energy consumption. In [13], they aim to minimize the number of UAVs and the total operation time by optimizing the UAV trajectory and hovering location. In [14], they address a UAV trajectory design problem for a single UAV with a limited battery and is allowed to swap its battery during its mission in a UAV-aided IoT network that contains multiple ground sensors that need to upload data within random time windows. Notice that most of the above works proposed high computation complexity solutions that may be inappropriate in time-sensitive applications.

This paper considers a single UAV as a data aggregator in an IoT network. The UAV is dispatched to collect data from IoT devices. The UAV can collect data from each IoT device within a particular time window. The UAV can get a

This work was supported in part by the National Science and Technology Council, Taiwan, under grant MOST 109-2221-E-007-079-MY3 and Qualcomm Technologies, Inc. under grant SOW NAT-487844.

ICC 2023 - IEEE International Conference on Communications | 978-1-5386-7462-8/23/$31.00 ©2023 IEEE | DOI: 10.1109/ICC45041.2023.10279268

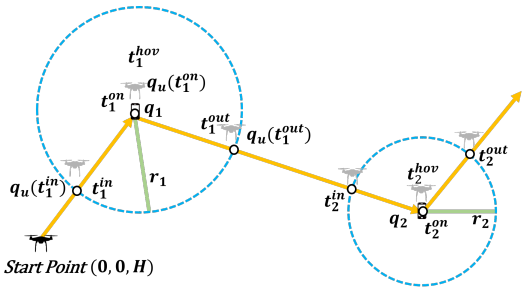

Fig. 1: Illustration of a UAV data collection mission.

specific profit if the data is uploaded on time. We propose a scheme that aims to maximize the total profit the UAV can get from all IoT devices. First, we find the visiting order of devices by solving Travelling Salesman Problem (TSP). Secondly, we design a Find Ending Time Algorithm (FETA) to find the ending time of each IoT device and a Pair Exchange Remove Algorithm (PERA) to meet IoT devices' deadlines by changing visiting orders or removing nodes. Furthermore, we improve the flying distance of the UAV by adjusting the optimal trajectory. Simulation results validate the effectiveness of the proposed strategy.

II. System Model and Problem Formulation

In this work, a single UAV is studied to collect data from N IoT devices on the ground. The UAV takes off from a starting point of $(0, 0, H)$ and flies at the fixed altitude of H while doing the mission, as shown in Fig. 1. The set of IoT devices is denoted as $\mathcal{N} = \{1, \ldots, N\}$. Each IoT device $n, \forall n \in \mathcal{N}$, is located at $q_n \triangleq (x_n, y_n, 0)$ and has a maximum amount of data s_n^{data} that can only be uploaded to the UAV during a specific time window $[t_n^{ready}, t_n^{due}]$.

Let $q_u(t) \triangleq (x(t), y(t), H)$ denote the UAV's location at time t. The three-dimensional (3D) distance between the UAV and IoT device n at time t is defined as

$$d_n(t) = \|q_u(t) - q_n\|, \tag{1}$$

for all $n = 1, \ldots, N$. Since IoT devices are assumed to be outdoors, the communications channel between IoT devices and the UAV is mainly dominated by a line of sight (LoS) link. The maximum link rate between the IoT device n and the UAV at time t can be expressed as [15]

$$R_n(t) = B \log_2 \left(1 + \frac{P_n \beta d_n(t)^{-\alpha}}{\sigma^2}\right), \tag{2}$$

where B is the channel bandwidth, P_n is the transmit power of IoT device n, α is the path loss exponent, σ^2 is noise power at the receiver, and β is the average channel power gain at a reference distance of one meter [15]. Note that the IoT devices have different transmit powers, so their communication ranges are different, as shown in Fig. 1. UAV can collect data within the communication radius $\tilde{r}_n$ of IoT device n. Suppose that the UAV starts and ends data collection from device n at t_n^{start} and t_n^{end}, respectively, where $t_n^{ready} \le t_n^{start} \le t_n^{end} \le t_n^{due}$. The device n is satisfied if its data demand equals the total data which the UAV can collect during the interval $[t_n^{start}, t_n^{end}]$. We have

$$s_n^{data} = \int_{t_n^{start}}^{t_n^{end}} R_n(t) dt. \tag{3}$$

The main objective of this work is to maximize the total profit that the UAV can get by collecting the data from IoT devices by optimizing the UAV's flight path and transmission scheduling of IoT devices. The problem can be formulated as follows

$$\max_{q_u(t), a_n, \forall n} \sum_{n=1}^{N} a_n p_n, \tag{4}$$

where p_n is the profit the UAV can get if device n is satisfied. If device n is satisfied, $a_n = 1$; otherwise, it is set to 0. Solving the problem (4) is even more challenging than the Traveling Salesman Problem with Time Windows (TSPTW) problem, which is an NP-hard problem. In the next section, we propose a heuristic algorithm to solve the problem with polynomial-time complexity.

III. Proposed Algorithm

In this section, we describe our proposed algorithm which is composed of three stages. In the first stage, we aim to find a good initial visiting order of IoT devices with a minimum flight distance. This appears as a variant of the well-known traveling-salesman problem (TSP), except that the UAV is not required to return to the start point. However, adding a dummy node with zero cost to the start point and identical positive costs to other nodes, the problem becomes a regular TSP problem. It then can be solved efficiently by many heuristic algorithms proposed in the literature [16]. For convenience, we relabel the indices of IoT devices according to the visiting order found by solving the TSP problem. The second stage consists of two steps. In the first step, we propose a Find Ending Time Algorithm (FETA), which is used to find the ending time of collecting data for each device. In the second step, we propose a Pair Exchange Remove Algorithm (PERA), which is used to exchange the visiting order of devices or remove the lowest profit device when the device is unsatisfied with its deadline. In the third stage, we optimize the trajectory to reduce the flying distance.

A. Find Ending Time Algorithm (FETA)

In FETA, the purpose is to determine the data collection time (and also hovering location and hovering time) of the UAV. We first let the UAV follow the path found in the first stage. We then mark three timestamps for each device visited by the UAV, including when the UAV arrives at the device's communication range, when the UAV arrives at the device's location, and when the UAV starts leaving out of the device's communication range. Let us denote those timestamps by t_n^{in}, t_n^{on}, and t_n^{out}, respectively, as shown in Fig. 1. Next, we relax the deadlines and then compute the hovering time t_n^{hov} and the ending time t_n^{end} of data collection of the UAV by considering the following four cases, as shown in Fig. 2. The red line indicates the duration that starts from the data ready time to

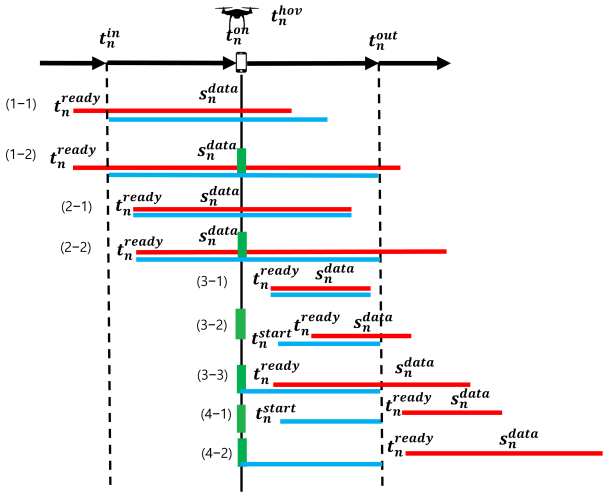

Fig. 2: The cases of t_n^{ready} for FETA.

the moment when the data collection is finished. The blue line is the period in which the device uploads the data to the UAV. The green line is the period the UAV hovers on the location closest to the device. The length of the red line is the length of the blue line plus the length of the green line.

The first case is the ready time $t_n^{ready} \leq t_n^{in}$ as shown in Fig. 2 (1-1). In this case, the UAV can collect the data as soon as it arrives at the device's communication range. We can use the following equation to calculate the data size that collected from the device n during the interval $[t_n^{in}, t_n^{out}]$ without hovering at the device's location.

$$s_n^{full} = \int_{t_n^{in}}^{t_n^{out}} R_n(t)\mathrm{d}t. \tag{5}$$

Note that the integral in (5) can be computed by tools using numerical methods. If s_n^{data} is less than s_n^{full}, the UAV does not need to hover on the device n and t_n^{hov}= 0. In this case, $t_n^{start} = t_n^{in}$ and t_n^{end} can be found by using the formula (3). If s_n^{data} is greater than s_n^{full}, the UAV needs to hover on the location above the device n and the hover time is computed as $t_n^{hov} = \frac{s_n^{data} - s_n^{full}}{R_n(t_n^{on})}$. We add t_n^{hov} to t_n^{out} and then obtain the t_n^{end}, as shown in Fig. 2 (1-2).

The second case is $t_n^{in} < t_n^{ready} \leq t_n^{on}$. Total data the UAV can collect from device n during the interval $[t_n^{ready}, t_n^{out}]$ without hovering is computed by

$$s_n^{fly} = \int_{t_n^{ready}}^{t_n^{out}} R_n(t)\mathrm{d}t. \tag{6}$$

If $s_n^{data} \leq s_n^{fly}$, the UAV does not need to hover, i.e., (t_n^{hov}= 0). We set $t_n^{start} = t_n^{ready}$ and use formula (3) to calculate t_n^{end} as shown in Fig. 2 (2-1). If $s_n^{data} > s_n^{fly}$, the UAV must hover at the device's location for a time $t_n^{hov} = \frac{s_n^{data} - s_n^{fly}}{R_n(t_n^{on})}$. We add t_n^{hov} to t_n^{out} and get the t_n^{end}, as shown in Fig. 2 (2-2).

The third case is $t_n^{on} < t_n^{ready} \leq t_n^{out}$. Since the UAV does not receive the data during the interval $[t_n^{in}, t_n^{on}]$, we compare $\frac{s_n^{full}}{2}$ with s_n^{data} to determine the hovering time. If $s_n^{data} \leq \frac{s_n^{full}}{2}$, there are two possible situations that UAV can finish collecting data before t_n^{out}. We will use formula (6) to calculate the total data s_n^{fly} that the UAV can collect from device n during the interval $[t_n^{ready}, t_n^{out}]$ without hovering. If $s_n^{fly} \geq s_n^{data}$, the UAV does not need to hover (t_n^{hov}= 0) and the t_n^{end} will be calculated by formula (3), as shown in Fig. 2 (3-1). If $s_n^{fly} < s_n^{data}$, then we will set t_n^{end} as t_n^{out} and use formula (3) to calculate t_n^{start} under the known data s_n^{data}. So UAV needs to fly to the position of t_n^{start} first and then wait for a duration of $t_n^{hov} = t_n^{ready}$- t_n^{start} until data is ready. After the data is ready, UAV will start collecting data while flying out of the communication radius of the device n. The end time is updated $t_n^{end} = t_n^{out} + t_n^{hov}$, as shown in Fig. 2 (3-2). If $s_n^{data} > \frac{s_n^{full}}{2}$, the UAV needs to hover at the position of t_n^{on}. Besides the hover time, UAV needs to wait for $t_n^{ready} - t_n^{on}$ until data is ready. The time ready to send data is t_n^{ready}. Therefore, the total hover time is $t_n^{hov} = t_n^{ready} - t_n^{on} + \frac{s_n^{data} - \frac{s_n^{full}}{2}}{R_n(t_n^{on})}$. The $t_n^{end} = t_n^{out} + t_n^{hov}$, as shown in (3-3) of Fig. 2.

The last case is ready time $t_n^{ready} > t_n^{out}$. If $s_n^{data} < \frac{s_n^{full}}{2}$, the UAV needs to hover at the position of t_n^{start}. We set $t_n^{end} = t_n^{out}$ and use formula (3) to calculate t_n^{start}. In this case, the UAV flies to the position of t_n^{start} and then wait for $t_n^{hov} = t_n^{ready}$ - t_n^{start} until data is ready. After the data is ready, UAV will start data collection while flying out of the communication radius of the device n. The end time $t_n^{end} = t_n^{out} + t_n^{hov}$, as shown in Fig. 2 (4-1). If $s_n^{data} > \frac{s_n^{full}}{2}$, the UAV needs to hover at the position of t_n^{on} for a duration of $t_n^{ready} - t_n^{on}$ until data is ready. The total hover time is $t_n^{hov} = t_n^{ready} - t_n^{on} + \frac{s_n^{data} - \frac{s_n^{full}}{2}}{R_n(t_n^{on})}$ The end time is updated as $t_n^{end} = t_n^{out} + t_n^{hov}$, as shown in Fig. 2 (4-2). The time complexity of FETA is $\mathcal{O}(n)$.

B. Pair Exchange Remove Algorithm (PERA)

Note that the deadline is excluded in FETA. Once FETA is done, we mark all devices whose $t_n^{end} > t_n^{due}$ as unsatisfied devices, i.e., $a_n = 0$. Next, we use PERA to change the visiting order to check whether the unsatisfied devices can become satisfied devices or not. The PERA is organized into three steps, and the details are described in Algorithm 1. The first step is to check whether unsatisfied devices are in the visiting order. The second step is to exchange pairs of visiting orders to search for satisfying visiting orders. The third step is to remove a device from the visiting order when it is not satisfied after the second step.

In the second step of PERA , we first use FETA to get the end time t_n^{end} and define $r(n) = t_n^{due} - t_n^{end}$ be the remaining time of device n before its deadline. We assume the first unsatisfied device is device u. We will pair exchange the visiting order of device $u-1$ with the device i, $i = u-2, u-3, ..., 1$. Let $V_{u-1,i}$ be the new visiting order that exchanges the device $u-1$ with the device i. We then recalculate the end time of data collection by FETA. Let $r_{u-1,i}(j) = t_{u-1,i}^{due}(j) - t_{u-1,i}^{end}(j)$ denote the remaining time before the deadline of device j, for all $1 \leq j \leq u$ in the visiting order $V_{u-1,i}$. Let

$$r_{u-1,i}^{min} = \min_{1 \leq j \leq u} r_{u-1,i}(j). \tag{7}$$

If $r_{u-1,i}^{min} \geq 0$, the visiting order $V_{u-1,i}$ is satisfied. Otherwise, the visiting order is unsatisfied. If there are multiple satisfied visiting orders, we select the visiting order $V_{u-1,i}$ with the largest value of $r_{u-1,i}^{min}$. Let

$$i^* = \underset{1 \leq i \leq u-2}{\arg\max}\ r_{u-1,i}^{min}. \tag{8}$$

Then, we exchange the orders of device $u-1$ and the device i^* and relabel the devices according to their new visiting order as $n = 1, 2, ..., N$. If no any $V_{u-1,i}$ is satisfied, we execute the third step of PERA.

In the third step of PERA, let V_i be the new visiting order after removing device i from visiting order V, for $i = 1, 2, ..., u$, where u is the first unsatisfied device found at the first step of the algorithm PERA. We recalculate the end time of collecting data by FETA. Let $r_i(j) = t_{u-1,i}^{due}(j) - t_{u-1,i}^{end}(j)$ be the remaining deadline time of device j in V_i. We define

$$r_i^{min} = \min_{1 \leq j \leq u, j \neq i} r_i(j). \tag{9}$$

Note that $r_i^{min} \geq 0$ indicates that the visiting order V_i is satisfied and vice versa. If there is no satisfied visiting order, we remove the unsatisfied device, i.e., device u . Otherwise, we remove the device i^* that the resulting visiting order produces the maximum total profit as follows

$$i^* = \underset{1 \leq i \leq u}{\arg\max}\, w_i, \tag{10}$$

where w_i is the total profit of all devices on the visiting order V_i and computed by

$$w_i = \Sigma_{n=1}^{i-1} p_n + \Sigma_{n=i+1}^{u} p_n, \tag{11}$$

for all $i = 1, \ldots, u$. Then, we set $N = N - 1$ and relabel the devices by their visiting order as $n = 1, 2, ..., N$. We iterate the first step of PERA to check the next unsatisfied devices, until no further devices can be served. The time complexity of PERA is $\mathcal{O}(n^2)$.

Algorithm 1: Pair Exchange Remove Algorithm (PERA)

Input: Each device's profit p_n, time window $[t_n^{ready}, t_n^{due}]$, for $1 \leq n \leq N$ and initial visiting order V.
Output: Visiting order V and profit

Use FETA to calculate end time
for $n = 1$ to N **do**
  if $t_n^{end} > t_n^{due}$ **then**
    $u = n$
    break
  else
    if $n == N$ **then**
      Output the visiting order V and profit
    end if
  end if
end for
for $i = u - 2$ to 1 **do**
  Exchange devices $u-1$ and i on the visiting order.
  Get new visiting order $V_{u-1,i}$
  Use FETA to calculate end time
  Use (7) to get $r_{u-1,i}^{min}$
end for
Use (8) to get i^*
if $r_{u-1,i^*}^{min} \geq 0$ **then**
  Relabel the devices $u-1$ to i^* and i^* to $u-1$
  Return to the first step of PERA
else
  for $i = 1$ to u **do**
    Remove device i from visiting order V
    Get new visiting order V_i
    Use FETA to calculate end time
    Use (9) to get r_i^{min}
  end for
  if $r_i^{min} < 0$ for all $i \in \{1, \ldots, u\}$ **then**
    Remove device u.
  else
    Remove device i^* found in (10).
  end if
  $N = N - 1$
  Relabel the visiting order V
  Return to the first step of PERA
end if

C. Flying Distance Refinement

It can be observed that the visiting location of the UAV (location at t_n^{on}) may not need to be the location above the device as initialized in the first stage but can be any location inside the device's communications range. In the third stage, we can reduce the total flying distance of the UAV while relaxing the visiting location $q_u(t_n^{on})$ to be inside the communication range. We first consider the problem that optimizes the visiting locations of the UAV to minimize the total distance of the flying path connecting the visiting locations. Such a problem can be formulated as

$$\min_{q_u(t_n^{on}), \forall n} \quad \sum_{n=2}^{N} \|q_u(t_n^{on}) - q_u(t_{n-1}^{on})\| \tag{12a}$$

$$\text{subject to} \quad \|q_u(t_n^{on}) - q_n\| \leq \tilde{r}_n, \forall n. \tag{12b}$$

The constraint in (12b) implies that the new visiting locations must be inside the communication range of the devices. Note that the problem in (12) is a convex optimization problem that can be solved by convex optimization tools, for example, CVX, with the interior point method. Solving the problem in (12) requires a complexity $\mathcal{O}(M^{3.5}\log(1/\epsilon))$, where M is the number of variables and ϵ is the given accuracy [2]. Once the problem in (12) is solved, we search new visiting locations to construct a shorter trajectory path as shown in Fig. 3. Let $l_n^t, \forall n$, denote the current visiting locations of the UAV and $l_n^c, \forall n$, denote the visiting locations obtained by solving (12). If $l_n^c, \forall n$, satisfies the time constraints of the devices (checked by running FETA), we will use them as the new visiting locations of the UAV. In this case, they are the best solutions. Otherwise, we use a binary search algorithm to find the most suitable visiting locations. First, let $l_n^m = (l_n^t + l_n^c)/2$, $\forall n$, be the middle point between the current visiting location and the visiting location found in (12). Next, we run FETA to check the feasibility of $l_n^m, \forall n$. If all middle points satisfy the devices' time constraints, we will try to find new visiting locations on the line segments connecting

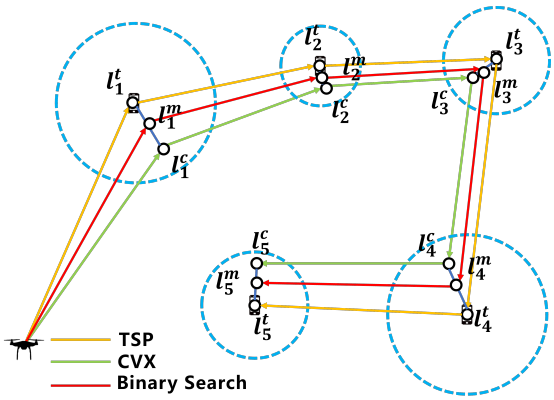

Fig. 3: Illustration of searching for a shorter flying trajectory.

l_n^m and l_n^c by assigning $l_n^t = l_n^m, \forall n$. Otherwise, we will try to find new visiting locations on the line segments connecting l_n^t and l_n^m by assigning $l_n^c = l_n^m, \forall n$. The binary search is repeated until the searching distance is less than one meter, i.e., $\|l_n^t - l_n^c\| \leq 1, \forall n$. According to our simulation, the flying distance with stage three is 25% lower than the flying distance without using the third stage.

IV. Simulation Results

In this section, we demonstrate the effectiveness of the proposed scheme through simulations using Matlab, which is executed in a personal computer with AMD Ryzen 5600x @3.7GHz processor, 16GB RAM. In the following experiments, IoT devices are uniformly deployed in 1000 m × 1000 m. The flying speed of a UAV is 10m/s, and the maximum flying height of the UAV is 70 m. We assume that the device's profit is defined as the data size collected from the IoT devices. We set $B = 10$ MHz, $\alpha = 2$, $\beta = -60$ dB, and $\sigma^2 = -110$ dBm. The transmit power of each device, i.e., P_n, is randomly selected in a range of [0.1, 0.5] watts. The communication radius is in the range of 15 m to 45 m. For comparison, we implement three baselines: the Min-Profit scheme, the Min-Deadline scheme, and the DACO scheme. The Min-Profit scheme/Min-Deadline scheme is implemented following our proposed scheme, except that the PERA is replaced by a removing policy in which minimum profit/minimum deadline devices are removed first until all remaining devices are served. The DACO scheme follows the data-driven ant colony optimization algorithm in [17] to solve TSP with time windows (TSPTW). In ant colony algorithms, the agents simulate the ants in nature that follows the pheromone trail to find the optimal route. The pheromone, in our case, is calculated by a function of the distance, time windows, and profit. Each simulation result is an average of 30 times the simulations.

In Fig. 4, we survey the performance in terms of total profit over the number of IoT devices that is adjusted from $N = 40$ to $N = 60$. The mean of data sizes is set to $M = 600$ Kbits, in which the data size is randomly selected in a range of $[M - 200, M + 200]$ Kbits. We can see that the total profit increases as the number of IoT devices increases in all the algorithms. Our proposed algorithm achieves the highest performance among others. The Min-Profit and Min-Deadline are worse than the proposed algorithm with PERA. The DACO algorithm uses roulette wheel selection to generate data collection path. Although the time window is considered in generating the path, it is still possible to choose a bad path due to the probabilistic decision making.

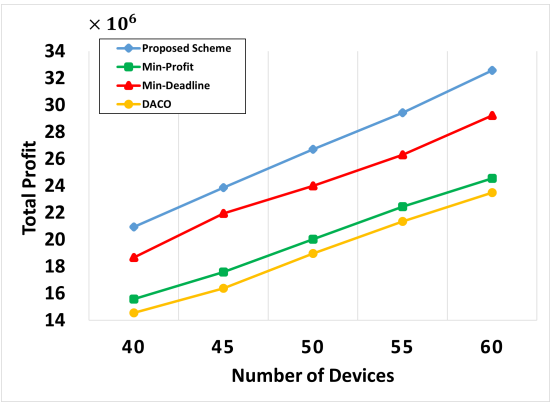

Fig. 4: Total profit versus number of devices.

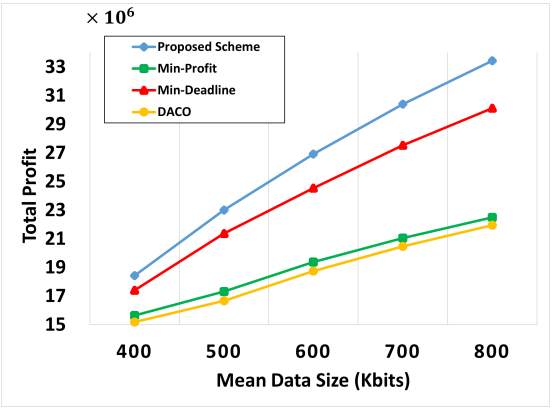

Fig. 5: Total profit versus data size of devices.

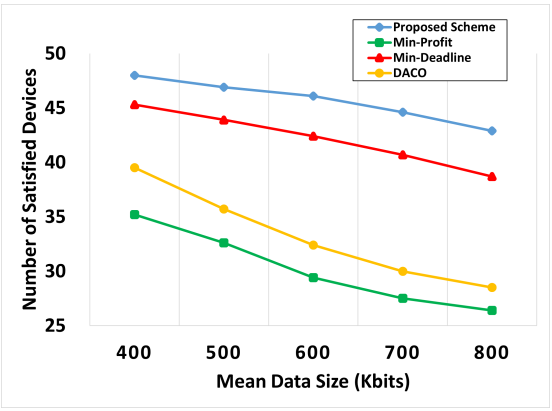

Fig. 6: Total satisfied devices versus data size of devices.

In Fig. 5, we examine the total profit for the data size of the devices. The mean data sizes of the devices are varied from $M = 400$ Kbits to $M = 800$ Kbits and the number of devices is set as $N = 50$. It shows that the total profit increases as the

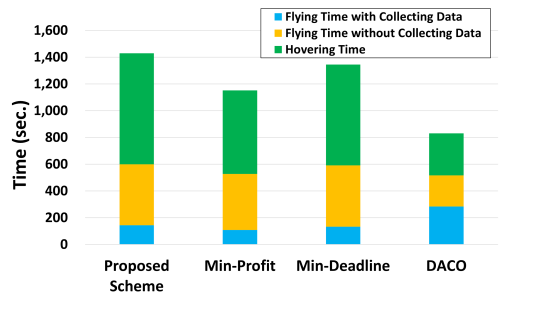

Fig. 7: The flying time and hovering time of algorithms.

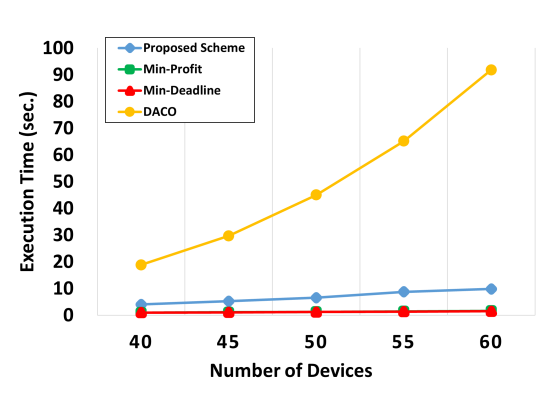

Fig. 8: Execution time versus number of devices.

data sizes of the devices increase in all schemes.

We show the number of satisfied devices in Fig. 6. We can see that the number of satisfied devices decreases in all the algorithms as the data size increases because the UAV needs more time to collect data from IoT devices. Our proposed scheme has the best total profit and device satisfiability performance compared to other schemes. The Min-Profit scheme has the worst performance. Although the device satisfiability of Min-Profit is smaller than DACO, its total profit is better than DACO. This is because Min-Profit removes the unsatisfied device with the lowest profit.

A realization of the total flying time of the UAV in schemes is shown in Fig. 7, where the number of devices is set as $N = 50$ and the mean data size is set to 600 Kbits. We can see that the proposed scheme with PERA spends the longest time since more devices are served in this scheme. Since the DACO prefers hovering at the locations above devices with the highest data rate, it thus has the least hovering time. In Fig. 8, we compare the running time of all schemes. The execution time increases as the number of devices increases in all schemes. The DACO algorithm is a kind of ant colony algorithm that typically takes a lot of time to converge. The Min-Profit and Min-Deadline schemes can be done concisely without exchanging the visiting order of devices.

V. Conclusion

In this work, we study the problem of using UAV to collect time-constrained data from IoT devices and obtain the maximum profit. The resulting problem could be considered as a variant of the TSPTW problem that is challenging to solve. We proposed a solution with FETA and PERA algorithms to solve the problem in polynomial time complexity. In addition, we use a binary search algorithm to reduce the flying distance. The simulation result shows that our proposed solution outperforms the baselines in terms of total profit while requiring a moderate running time.

References

[1] M. Mozaffari, W. Saad, M. Bennis, Y.-H. Nam, and M. Debbah, "A tutorial on UAVs for wireless networks: Applications, challenges, and open problems," *IEEE Communications Surveys Tutorials*, vol. 21, no. 3, pp. 2334–2360, 2019.

[2] C. You and R. Zhang, "3D trajectory optimization in rician fading for UAV-enabled data harvesting," *IEEE Transactions on Wireless Communications*, vol. 18, no. 6, pp. 3192–3207, 2019.

[3] J. Lin, W. Yu, N. Zhang, X. Yang, H. Zhang, and W. Zhao, "A survey on internet of things: Architecture, enabling technologies, security and privacy, and applications," *IEEE Internet of Things Journal*, vol. 4, no. 5, pp. 1125–1142, 2017.

[4] M. Mozaffari, W. Saad, M. Bennis, and M. Debbah, "Mobile unmanned aerial vehicles (UAVs) for energy-efficient internet of things communications," *IEEE Transactions on Wireless Communications*, vol. 16, no. 11, pp. 7574–7589, 2017.

[5] J. Zhang, L. Zhou, F. Zhou, B.-C. Seet, H. Zhang, Z. Cai, and J. Wei, "Computation-efficient offloading and trajectory scheduling for multi-UAV assisted mobile edge computing," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 2, pp. 2114–2125, 2020.

[6] C. Zhan, H. Hu, X. Sui, Z. Liu, and D. Niyato, "Completion time and energy optimization in the UAV-enabled mobile-edge computing system," *IEEE Internet of Things Journal*, vol. 7, no. 8, pp. 7808–7822, 2020.

[7] Z. Yang, C. Pan, K. Wang, and M. Shikh-Bahaei, "Energy efficient resource allocation in UAV-enabled mobile edge computing networks," *IEEE Transactions on Wireless Communications*, vol. 18, no. 9, pp. 4576–4589, 2019.

[8] M. Li, N. Cheng, J. Gao, Y. Wang, L. Zhao, and X. Shen, "Energy-efficient UAV-assisted mobile edge computing: Resource allocation and trajectory optimization," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 3, pp. 3424–3438, 2020.

[9] J. Yang, J. Chen, and Z. Yang, "Energy-efficient UAV communication with trajectory optimization," in *2021 2nd ICBASE*, pp. 508–514, 2021.

[10] X. Hu, K.-K. Wong, K. Yang, and Z. Zheng, "UAV-assisted relaying and edge computing: Scheduling and trajectory optimization," *IEEE Transactions on Wireless Communications*, vol. 18, no. 10, pp. 4738–4752, 2019.

[11] M. Samir, S. Sharafeddine, C. M. Assi, T. M. Nguyen, and A. Ghrayeb, "UAV trajectory planning for data collection from time-constrained IoT devices," *IEEE Transactions on Wireless Communications*, vol. 19, no. 1, pp. 34–46, 2020.

[12] O. Ghdiri, W. Jaafar, S. Alfattani, J. B. Abderrazak, and H. Yanikomeroglu, "Energy-efficient multi-UAV data collection for IoT networks with time deadlines," in *2020 IEEE GLOBECOM*, pp. 1–6, 2020.

[13] S. Shen, K. Yang, K. Wang, G. Zhang, and H. Mei, "Number and operation time minimization for multi-UAV-enabled data collection system with time windows," *IEEE Internet of Things Journal*, vol. 9, no. 12, pp. 10149–10161, 2022.

[14] C. T. Cicek, "A reinforcement learning algorithm for data collection in UAV-aided iot networks with uncertain time windows," in *2021 IEEE ICC Workshops*, pp. 1–6, 2021.

[15] J. Gong, T.-H. Chang, C. Shen, and X. Chen, "Flight time minimization of UAV for data collection over wireless sensor networks," *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 9, pp. 1942–1954, 2018.

[16] Y. Zeng, X. Xu, and R. Zhang, "Trajectory design for completion time minimization in UAV-enabled multicasting," *IEEE Transactions on Wireless Communications*, vol. 17, no. 4, pp. 2233–2246, 2018.

[17] Z. Xu, J. Yu, and W. Su, "Finding the global optimal solution in dynamic multiple TSPTW with data-driven ACO," in *2021 IEEE SmartWorld/SCALCOM/UIC/ATC/IOP/SCI*, pp. 41–48, 2021.