Adaptive Path Planning of UAVs for Delivering Delay-Sensitive Information to Ad-hoc Nodes

JinYi Yoon, YeonJin Jin, Narangerelt Batsoyol, and HyungJune Lee†
Department of Computer Science and Engineering
Ewha Womans University, Seoul, South Korea
Email: {yjin3012, yeonjin13, naraa}@ewhain.net, hyungjune.lee@ewha.ac.kr

***Abstract*—We consider the problem of path planning using multiple UAVs as message ferries to deliver delay-sensitive information in a catastrophic disaster scenario. Our main goal is to find the optimal paths of UAVs to maximize the number of nodes that can successfully be serviced within each designated packet deadline. At the same time, we want to reduce total travel time for visiting over a virtual grid topology. We propose a distributed path planning algorithm that determines the next visit grid point based on a weighted sum of travel time and delivery deadline. Together with path planning, we incorporate a task division mechanism that collaboratively distributes the unvisited grid points with other UAVs so that the entire travel time can substantially be reduced. Simulation results demonstrate that our distributed path planning algorithm mixed with task division outperforms all baseline counterpart algorithms in terms of on-time service node rate and total travel time.**

I. Introduction

Unmanned Aerial Vehicles (UAVs) have been considered to be a new form of aviation, playing various roles: aerial surveillance and monitoring [3], transporting packages [15], and communication relays for network reconstruction [7]. Several key features of UAVs are deployment flexibility, re-programmable architecture, and less constrained movement.

In a catastrophic disaster situation, users may be communication-wise isolated due to the collapsed infrastructure network. The usage of UAVs can be a rescue for delivering location-sensitive and time-sensitive information to the nodes in a certain affected area. For example, evacuation plan varies across location, and its door-to-door information delivery from a disaster control center to local nodes via UAVs can be a very promising approach under network outage.

The problem of path planning has been investigated in robotics and operation research communities under the names of vehicle routing problem (VRP) [4], [9] and traveling salesman problem (TSP) [6], [11]. These approaches tackle the problem with optimization techniques by formulating into linear programs with relaxation. Some genetic algorithm-based approaches have been proposed for UAV path planning [1], [12]. Related to the path planning with deadline constraints, some researchers have studied deadline-TSP [2] and VRP with time window [13]. Although the classic TSP and VRP problems have been well explored considering time constraints, they usually suffer from NP-hardness or centralized computation. Further, they have not explicitly co-optimized both deadlines to each node and entire traversal time with multiple vehicles.

This paper presents a distributed path planning algorithm with time constraints using multiple UAVs. We design a practical framework for UAVs to traverse over a virtual grid topology. A UAV can deliver data directly to locally connectable nodes with the one hop air-to-ground transmission at each grid point. By borrowing some ideas from the Minimum Weighted Sum First (MWSF) scheduling and the Earliest Deadline First (EDF) with k-lookahead [14], we propose a path planning scheduling algorithm that can be applicable to this framework under delivery deadline and traversal time constraints.

To leverage multiple UAVs to deliver time-sensitive and location-sensitive data, we provide a collaborative visit task division for the UAVs to perform during the encounter in the middle of their mission. The fair task division can achieve the evenly distributed traversal over UAVs with a full utilization of multiple UAVs, eventually contributing to reducing the entire traversal time over region of interests (RoI).

The remainder of this paper is organized as follows: After presenting the system model in Sec. II, we present our distributed path planning scheduling algorithm in Sec. III and our collaborative task division mechanism for UAVs in Sec. IV. We validate our approaches based on simulations in Sec. V, and finally conclude this paper in Sec. VI.

II. System Model

We consider a path planning problem for UAVs to deliver delay-sensitive information to each different ad-hoc node. In disaster scenarios, the entire communication infrastructure may be collapsed, and some urgent information such as customized evacuation and status progress dependent on location should be delivered to each affected node, respectively, with the help of UAVs. We use multiple UAVs to carry node-specific information obtained from a disaster control center to the nodes. Our goal is to find the optimal paths of UAVs to maximize the number of nodes that successfully receive data that is supposed to be delivered from a UAV within a given distinct time deadline. At the same time, we want to minimize total travel time for traversing all relevant nodes to which UAVs have data to deliver respectively.

We assume that a UAV can communicate with terrestrial ad-hoc nodes or with other UAVs using a wireless radio (e.g.,

†HyungJune Lee is the corresponding author.

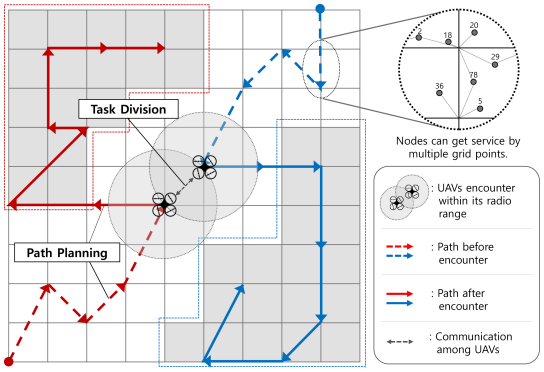


Fig. 1. System overview with distributed path planning and collaborative task division among multiple UAVs.

802.11 or 802.15.4). We let UAVs traverse over a virtual grid topology. When a UAV stays at a virtual grid point on the fly, it starts communicating with terrestrial ad-hoc nodes within its radio range. It is assumed that the virtual grid topology and the communicable ad-hoc node list at each grid point are prior knowledge to UAVs. If there exist multiple grid points that can be communicated with a node, a UAV can deliver data to the node at any of these grid points. A UAV makes movement progress based on its own distributed navigation decision, while not being aware of movements of other UAVs. It should be noted that battery outage, UAV collision, and navigation control issues (e.g., upon encountering static or mobile obstacles) are out-of-scope in this paper.

Each UAV maintains the list of visited grid points, and shares it with another UAV each other when they encounter on the fly within radio range. Our proposed path planning algorithm consists of two parts: 1) distributed path planning (Sec. III) and 2) collaborative task division for unvisited grid points among encountered UAVs (Sec. IV) as in Fig. 1.

III. Distributed Path Planning

In the emergency situation of network outage due to disasters, information imbalance with respect to the urgency level and the location deteriorates the damage condition. It is of great significance to secure an alternative way to communicate from a disaster control center directly to local nodes for distributing both deadline and location sensitive data. Using UAVs as message ferries, a full coverage over all virtual grid points using multiple UAVs for delivering a corresponding data to each node at a designated grid point should be efficient.

UAVs make their own independent navigation decisions among unvisited grid points. If two UAVs become closer within radio range during their navigation, they are allowed to exchange visited grid points so that the duplicated coverage is prohibited as much as possible.

We propose a distributed path planning algorithm for UAVs to traverse virtual grid points. The goal of our algorithm is twofold: 1) maximizing the percentage of nodes that have successfully received data within its designated deadline constraint, and 2) reducing the total traversal time to cover all of grid points with multiple UAVs. Each UAV continues to generate its own optimal path until it serves all of the nodes with the remaining deadline.

We exploit three main factors to control our path planning: 1) travel time from a grid point to another, 2) the number of nodes that can be serviced at a grid point, and 3) delivery deadline for each node. We introduce several notations to denote the distance between grid points a and b as $d(G_a, G_b)$, and the UAV flying speed as $\bar{v}_{uav}$. Also, the packet delivery deadline for node i is denoted as T_{N_i}, and T_{N_i} decreases as time passes.

We adopt a variant of the weighted sum consisting of the travel time and the delivery deadline by referring to [14]. We apply the *urgency* metric not into the level of nodes, but into the level of grid points. This is due to the fact that a UAV makes its movement progress over the grid-based topology. When a UAV is deployed at grid point i, there exist nodes within their radio range from the air, i.e., $\eta_{G_i} = \{N_{n_1}, N_{n_2}, \ldots, N_{n_l}\}$. Also, a single node can receive data from any grid points within radio range, denoting that node j can be communicated with a set of grid points $\psi_{N_j} = \{G_{g_1}, G_{g_2}, \ldots, G_{g_m}\}$.

We calculate the weighted sum-based urgency measure for all possible pairs of the unvisited grid point and its communicable node as follows:

$$w(G_i, N_j) = \alpha \cdot T_{N_j} + (1 - \alpha) \cdot d(G_{current}, G_i)/\bar{v}_{uav} \quad (1)$$

where $G_{current}$ is the currently visiting grid point, G_i is a candidate grid point i, $N_j \in \eta_{G_i}$ for grid point i and node j, and α is a weight parameter on the packet delivery deadline.

Each UAV runs the path planning algorithm in a distributed manner to find k grid points to visit for the next traversal at the currently visiting grid point. After obtaining the k future grid points to visit, it finally selects an optimal visiting order over $k!$ permutation cases, similar to [14]. Any nodes with deadline expired during the mission are excluded for the path planning decision. Once a UAV moves to the next grid point, it continues to run the aforementioned procedure, while refreshing the k future grid points to visit.

A. Selecting Next k Grid Point Candidates to Visit

To select the next k grid point candidates to visit, we calculate the weighted sum-based urgency measure for all unvisited grid point-to-node pairs at the currently visiting grid point, and pick up k grid points with the k lowest values.

In case of having more grid points than expected due to multiple points with the same value, we choose the ones that are physically closer to the currently visiting grid point.

If the above second criterion does not still rule out ambiguities, we make the k grid point selection with respect to the expected number of nodes to be serviced within their own packet deadline if visited. For any other remaining cases, we randomly select k grid point candidates out of the remaining candidates.

B. Establishing a Visiting Path over the Selected k Grid Points

Given k grid points to visit for the next traversal, we iterate over k permutations to conjecture all possible paths. We first probe which permutation to have the maximum number of

nodes to be serviced within their packet deadline over the entire selected path. Second, we calculate the expected travel time for each possible visiting order, and obtain the lowest one. We apply the above two criteria in priority.

If two criteria do not result in only one path, we pick up the path that has the closest distance to its first next visiting grid point over the path from the currently visited grid point, and then choose the path with the maximum number of nodes to be serviced at the first next visiting grid point. If we cannot still get the only one visiting path even with the last criterion, we randomly choose one out of them.

The complete path planning procedure is described in Algorithm 1. The computation complexity of the path planning algorithm is given by $\mathcal{O}(k! \cdot nk)$ where n is the number of all communicable nodes in a visiting grid point.

Algorithm 1 Distributed Path Planning Algorithm

Input: *Current location of UAV, allocated grid point list*
Output: *Next grid point to move*
// I. Select next k grid point candidates to visit
calculate weighted sum $w(G, N)$ for all unvisited grid point G and node N pairs;
add at most k grid points of (G, N) with the lowest $w(G, N)$ to candidate list $candi$;
while (# of selected grid points in $candi < k$) **do**
  for (G_i = [grid points of (G, N) with minimum $w(G, N)$ among grid points not in $candi$]) **do**
    if (multiple k grid points of (G_i, N) have minimum $w(G_i, N)$) **then**
      if (multiple G_i have shortest $d(G_{current}, G_i)$) **then**
        if (multiple G_i have same # of communicable nodes) **then**
          add random grid point G_i to $candi$;
        else
          add G_i which has maximum # of communicable nodes to $candi$;
        end if
      else
        add G_i to $candi$ whose $d(G_{current}, G_i)$ is the shortest;
      end if
    end if
  end for
end while
// II. Establish a visiting path over the selected k grid points
for (all permutation paths of k grid points in $candi$) **do**
  calculate # of nodes serviced in deadline and travel time for each path;
end for
if (multiple paths have maximum # of nodes to be serviced) **then**
  if (multiple paths have minimum travel time among paths above) **then**
    if (multiple first next visiting G_i have same $d(G_{current}, G_i)$ from among paths above) **then**
      if (multiple first next visiting G_i have maximum number of N to be serviced) **then**
        select random path;
      else
        select path with the maximum number of N to be serviced at the first next visiting G_i;
      end if
    else
      select path that has the closest $d(G_{current}, G_i)$ to its first next visiting G_i;
    end if
  else
    select path with minimum travel time;
  end if
else
  select path with maximum # of nodes to be serviced;
end if
select first visiting grid point of selected path as G_{next} to move;

IV. Collaborative Task Division

Our path planning algorithm is continuously run to determine the next visit grid point whenever a UAV arrives at a grid point. When two or more UAVs encounter during their flight, they exchange their own visited grid point list one another and update the shared visit information to their own visited grid point list. The path planning algorithm itself generates its next suitable path that avoids the duplicate coverage by referring to the updated visited grid point list. However, it does not explicitly consider the following aspects: how many unvisited grid points should be covered by each different UAV, and how the remaining unvisited grid points are geographically located relatively to the current positions of UAVs.

Given that UAVs become aware of the remaining unvisited grid points after sharing their own visit list, they may undertake a more aggressive decision of how the unvisited grid points should be distributed to themselves, respectively. Our goal is to minimize the longest traversal time of a UAV that turns out to be the total coverage time using multiple UAVs. We want to make geographically similar clusters of unvisited grid points such that each UAV is assigned to the most relevant group and performs its path planning algorithm only over the well selected unvisited grid point group.

We strive to utilize several existing clustering techniques including K-means clustering [8] and a balanced K-means clustering [5] to embed into our task division mechanism. We demonstrate how a task division mechanism can contribute to further improving the path planning performance. It should be noted that we do not constrain ourselves to a certain clustering algorithm; any other more suitable clustering technique can be applied to our proposed mechanism.

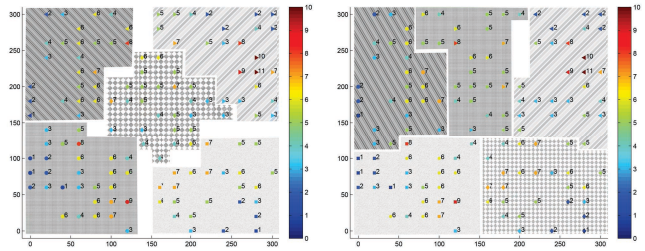
When UAVs encounter in the air within radio range, they share the already-visited grid point list. The UAV that initiates this sharing procedure operates as a master UAV, which performs the task division mechanism and shares the assigned task results with the other UAVs. This task division procedure is continuously executed upon subsequent UAVs' encounter events by running the following clustering algorithm.

A. Naive Clustering

We devise the most naive clustering algorithm called *Closest*. Given the geographical locations of all the unvisited grid points that have nodes to service, shared by the encountered UAVs, we calculate the geographical distance between an unvisited grid point and each UAV candidate and assign the closest UAV for covering the unvisited grid point. After calculating all the unvisited grid points, each UAV is assigned to a subset of unvisited grid points, and all the encountered UAVs collaboratively cover all the unvisited grid points.

B. K-means Clustering

We leverage one of the most popular clustering techniques, K-means clustering technique [8] to our task division mechanism. Assuming that K UAVs need to be assigned to unvisited grid points, we apply the K-means clustering with K number of centroids based on the distance measure in geographical



(a) After K-means clustering (b) After balanced K-means clustering

Fig. 2. Geographical distribution of unvisited grid points belonging to their governing UAV after task division using 5 UAVs. Each grid point is marked with a potential map with respect to the number of available nodes to access.

locations of unvisited grid points. After all the unvisited grid points are assigned to one of K centroids, we find the UAV-to-centroid pairs that provide the minimum deviation in distance. We let all the mapped UAVs cover their assigned unvisited grid points (e.g., as in Fig. 2(a)).

C. Balanced K-means Clustering

To fully utilize multiple UAVs to reduce the total traversal time, it is important for each UAV to cover the similar number of visiting grid points, while considering the deviation of the travel distance among UAVs. To achieve the fair task division over multiple UAVs, we adopt a classic balanced K-means clustering algorithm [5].

It forces the maximum number of pre-allocated slots per cluster, which is N_{grid}/K where N_{grid} is the total number of unvisited grid points, and K is the number of UAVs and also the number of clusters. We perform a variant of the K-means clustering procedure by using the Hungarian method to calculate the pairwise distance metric and obtain the minimal edge weight pairing. After the continuous iteration procedure converges, we assign each centroid for a cluster to the closest UAV. Finally, all the unvisited grid points are evenly distributed to UAVs for the next traversal (e.g., as in Fig. 2(b)).

V. Evaluation

We validate our path planning algorithm combined with collaborative task division in a simulated network of 300 randomly deployed nodes over a territory area of $300\times300m^2$ as illustrated in Fig. 3. We use a virtual grid topology of 16×16 with the distance of $20m$ between grid points. We randomly generate the packet delivery deadline for each respective ad-hoc node from the uniform distribution on the interval $[30s, 90s]$ for an urgent delivery scenario and on the interval $[370s, 430s]$ for a relatively deadline-relaxed delivery scenario. As in Table I, the flying speed of UAVs is $12m/s$ [10], and the flying height is $7m$. The communication radio range of UAVs and ad-hoc nodes is given by $30m$ (e.g., low-power wireless such as 802.15.4). We let UAVs initiate their mission from fixed grid points nearby the center of the simulated territory area.

We evaluate the efficiency of path planning performance in terms of on-time serviced node percentage and travel time of UAVs. We quantify the on-time serviced node percentage as the total percentage of nodes that have successfully received

Simulation Environment	
Territory area	$300\times300m^2$
# of nodes	300
Grid size	$20m$
Packet delivery deadline	$30 \sim 430s$
UAV speed	$12m/s$
UAV flying height	$7m$
Communication radio range	$30m$
Simulation Parameter	
# of UAV	$1 \sim 4$
Weight parameter α	0.2
Permutation parameter k	2

TABLE I
Simulation environment and parameter

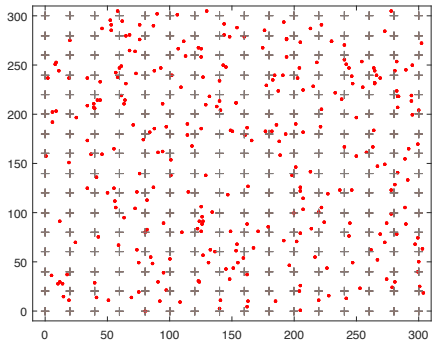


Fig. 3. A distribution map of the simulated network consisting of 300 ad-hoc nodes (red circles) with 16×16 grid points (gray plus signs).

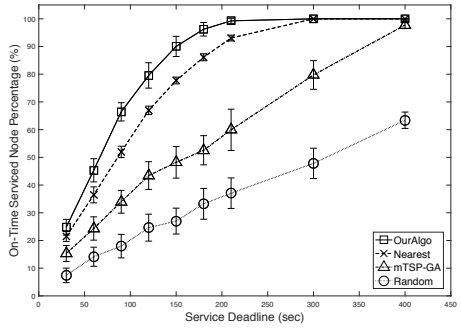
data within each respective packet deadline according to a path planning algorithm. We measure the travel time of UAVs as the longest travel time among UAVs for the entire traversal over grid points with effective ad-hoc nodes in delivery deadline. To obtain statistically meaningful results, we plot the averaged values with standard deviation over 10 experiments. 4 UAVs are used, unless otherwise noted.

After showing each performance our distributed path planning algorithm and our task division mechanism separately, we explore the overall system performance.

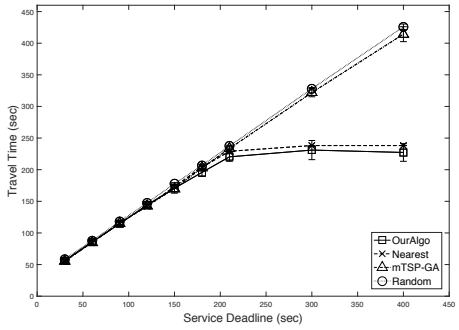
A. Distributed Path Planning

We investigate our distributed path planning algorithm excluding the task division mechanism by varying the average packet deadline with ±30 seconds in the uniform distribution. We compare ours to several baseline counterpart algorithms of *Random*, *Nearest*, and *mTSP-GA* [16] as shown in Fig. 4(a). *Random* algorithm randomly determines the next visiting grid point, while *Nearest* algorithm attempts to go to the nearest grid point from the currently visited grid point. We also compare our algorithm with *mTSP-GA*, which is a multiple traveling salesman problem with genetic algorithm, applying it into our simulation environment. To purely explore the efficiency of path planning algorithms, only one UAV is used.

Regarding on-time serviced node percentage, our algorithm outperforms all the other algorithms. To compare ours with *Nearest* algorithm, *Nearest* algorithm considers only travel time based on geographical distance between the currently visiting grid point and the next candidate point, whereas our algorithm co-optimizes both measures. Although *mTSP-GA* algorithm does not explicitly consider the deadline constraint,

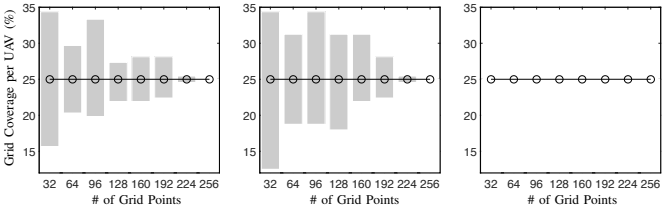


(a) On-time serviced node percentage



(b) Travel time

Fig. 4. On-time serviced node percentage and travel time of path planning algorithms with respect to initial service deadline.



(a) Closest (b) K-means (c) Balanced K-means

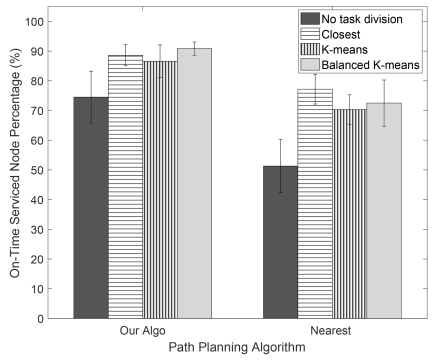
Fig. 5. Grid point coverage rate per UAV using each different clustering algorithm for task division.

and thus the evaluation may not be a fair comparison, we just show how its time-related performance behaves.

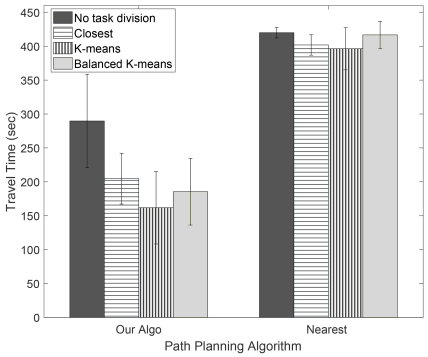
We explore how each path planning algorithm has an effect on the travel time of a UAV in Fig. 4(b). Under the service deadline range from 30 seconds to 200 seconds, all of the algorithms show the similar performance. It should be noted that since we let each path planning algorithm exclude deadline-expired nodes for travel decision, our algorithm has indeed serviced more nodes, while spending similar travel time. Beyond the range corresponding to a deadline-relaxed scenario, our algorithm and *Nearest* algorithm lead to the shortest travel time to cover the whole network. The reason that *mTSP-GA* incurs very high travel time is that there still remain many unvisited nodes to service even with a larger service deadline case due to the inherent lack of timely service feature (as shown in Fig. 4(a)).

B. Collaborative Task Division

We evaluate how each different clustering algorithm affects the overall task division performance as in Fig. 5. We quantify what percentage of grid points are covered by a single UAV. To



(a) On-time serviced node percentage under an urgent delivery scenario (with the deadline of $[30s, 90s]$) using 4 UAVs.



(b) Travel time under a deadline-relaxed scenario (with the deadline of $[370s, 430s]$) using 4 UAVs.

Fig. 6. On-time serviced node percentage and travel time performance with respect to path planning and task division types.

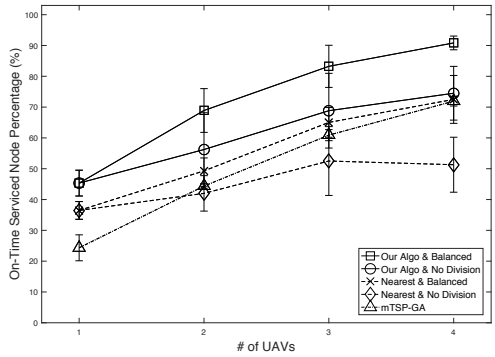
verify the one-shot task division performance in various environments, we control the number of grid points to service out of 16×16 points, which are covered by 4 UAVs. Both *Closest* and K-means clustering algorithms show large deviations in terms of service load per UAV, whereas the balanced K-means clustering algorithm divides the number of grid points evenly with UAVs, achieving the fairness among UAVs.

C. Overall System Performance

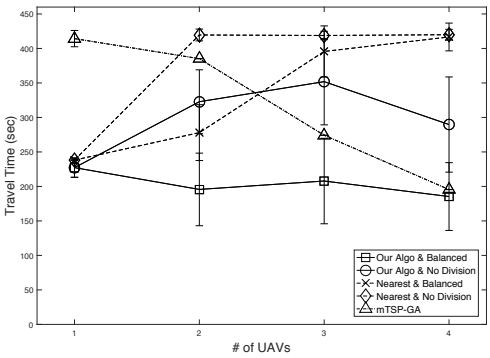
We now evaluate the whole system that incorporates both path planning and task division mechanism. We have chosen the best two path planning algorithms, our algorithm and *Nearest*, based on the evaluation in Sec. V-A. We apply our task division mechanism based on three different clustering algorithms to each path planning scheme, compared to the one without using any task division.

With regard to on-time serviced node percentage under a tight deadline scenario as in Fig. 6(a), a good combination of our path planning algorithm and balanced K-means clustering-based task division achieves the highest on-time service rate of up to 91%. It also demonstrates that any task division procedure further improves on-time service rate for both our path planing algorithm and *Nearest* up to 22% as opposed to no task division case.

Next, we compare travel time for various combinations of path planning and task division types in Fig. 6(b). To make the fair comparison in terms of travel time, we focus on a relatively large deadline scenario over $[370s, 430s]$ by having



(a) On-time serviced node percentage with the deadline of $[30s, 90s]$.



(b) Travel time with the deadline of $[370s, 430s]$.

Fig. 7. On-time serviced node percentage and travel time performance with respect to # of UAVs.

almost 100% on-time service node rate irrespective of path planning. The usage of our path planning with task division constructively reduces the travel time by 44%, compared to the one without task division. On the other hand, the task division mechanism with *Nearest* path planning does not contribute to improving the performance in terms of travel time. This implies that a proper mixture of path planning and task division is a key to improve system performance.

Lastly, we assess system performance by varying the number of UAVs used as shown in Fig. 7. Our path planning algorithm combined with the balanced K-means clustering-based task division achieves the best performance in terms of on-time service node rate and travel time. As the number of UAVs used increases, our algorithm significantly increases on-time service node rate, while lowering the required travel time. This means that the usage of more UAV resource has improved system performance.

VI. Conclusion

We have presented a distributed path planning algorithm for UAVs to directly deliver data with delivery deadline constraints to local ad-hoc nodes via a virtual grid topology. Upon encountering other UAVs during its mission, we provide a way of effectively distributing visit tasks, reducing the entire traversal time of UAVs.

We have demonstrated that a good mixture of path planning and task division can significantly improve the overall system performance in terms of on-time service rate and travel time.

For future work, we may apply a proximity-based task distribution for UAVs to gradually enlarge the navigation area to obtain more locally optimized paths. It would also be interesting to consider the battery outage issue during the UAV mission to make more realistic refinement on path planning. We may also reduce computation complexity for more scalable path planning.

Acknowledgment

This work was supported by Basic Science Research Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Education(NRF-2015R1D1A1A01057902).

References

[1] M. d. S. Arantes, J. d. S. Arantes, C. F. M. Toledo, and B. C. Williams. A hybrid multi-population genetic algorithm for uav path planning. In *Proceedings of the 2016 on Genetic and Evolutionary Computation Conference*, pages 853–860. ACM, 2016.

[2] N. Bansal, A. Blum, S. Chawla, and A. Meyerson. Approximation algorithms for deadline-tsp and vehicle routing with time-windows. In *Proceedings of ACM STOC*, pages 166–174, 2004.

[3] J. Everaerts et al. The use of unmanned aerial vehicles (UAVs) for remote sensing and mapping. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 37:1187–1192, 2008.

[4] B. L. Golden, S. Raghavan, and E. A. Wasil. *The vehicle routing problem: latest advances and new challenges*, volume 43. Springer Science & Business Media, 2008.

[5] R. He, W. Xu, J. Sun, and B. Zu. Balanced k-means algorithm for partitioning areas in large-scale vehicle routing problem. In *International Symposium on Intelligent Information Technology Application*, volume 3, pages 87–90, 2009.

[6] K. L. Hoffman, M. Padberg, and G. Rinaldi. Traveling salesman problem. In *Encyclopedia of Operations Research and Management Science*, pages 1573–1578. Springer, 2013.

[7] D. Jeong, S.-Y. Park, and H. Lee. DroneNet: Network reconstruction through sparse connectivity probing using distributed UAVs. In *IEEE PIMRC*, pages 1797–1802, 2015.

[8] T. Kanungo, D. M. Mount, N. S. Netanyahu, C. D. Piatko, R. Silverman, and A. Y. Wu. An efficient k-means clustering algorithm: Analysis and implementation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(7):881–892, 2002.

[9] G. Laporte. The vehicle routing problem: An overview of exact and approximate algorithms. *European Journal of Operational Research*, 59(3):345–358, 1992.

[10] L. Lin and M. A. Goodrich. Uav intelligent path planning for wilderness search and rescue. In *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 709–714. IEEE, 2009.

[11] G. Reinelt. Tspliba traveling salesman problem library. *ORSA journal on computing*, 3(4):376–384, 1991.

[12] V. Roberge, M. Tarbouchi, and G. Labonté. Comparison of parallel genetic algorithm and particle swarm optimization for real-time uav path planning. *IEEE Transactions on Industrial Informatics*, 9(1):132–141, 2013.

[13] M. M. Solomon. Algorithms for the vehicle routing and scheduling problems with time window constraints. *Operations research*, 35(2):254–265, 1987.

[14] A. A. Somasundara, A. Ramamoorthy, and M. B. Srivastava. Mobile element scheduling for efficient data collection in wireless sensor networks with dynamic deadlines. In *IEEE Real-Time Systems Symposium*, pages 296–305, 2004.

[15] J. Stolaroff. The need for a life cycle assessment of drone-based commercial package delivery. *Livermore, CA: Lawrence Livermore National Laboratory*, 2014.

[16] L. Tang, J. Liu, A. Rong, and Z. Yang. A multiple traveling salesman problem model for hot rolling scheduling in shanghai baoshan iron & steel complex. *European Journal of Operational Research*, 124(2):267–282, 2000.