Safe Optimistic Path Planning for Autonomous Drones under Dynamic Energy Costs

Giorgos Polychronis[1] and Spyros Lalis[2]
University of Thessaly

***Abstract*— Unmanned aerial vehicles or so-called drones are already used in several applications to perform different sensing and monitoring missions. A central problem is to plan these missions so as to minimize the completion time. This planning must also consider the typically limited autonomy of drones and the need to change their batteries in order to support longer missions. The problem becomes even harder when multiple drones are involved and there is uncertainty about the energy that will be consumed to move between the points of interest in the target area. In this paper, we present a heuristic algorithm for tackling this problem in an online fashion, which takes into account the actual energy costs that occur during the mission in order to adapt the planned paths. The algorithm works in an optimistic way, assuming that the costs will not always be the worst possible. Still, it guarantees that all vehicles will always make it back to the base station. The algorithm is evaluated via simulation experiments for a range of scenarios. Our results show that the proposed heuristic can significantly reduce the mission time of a conservative offline solution by up to 51%, while achieving up to 18% better results compared to a pessimistic online variant that plans the paths of the vehicles assuming the worst possible costs.**

I. Introduction

There is currently a lot interest in using unmanned aerial vehicles (UAVs) or so-called drones in many different applications with a business or purely humanitarian purpose, such as the delivery of goods, search and rescue missions, damage assessment after a physical catastrophe, and regular area patrol/monitoring. The main reason for the steadily increasing popularity of drones is that they can practically fly by themselves, with minimum or no involvement from a pilot, which reduces the room for human error and can further reduce operational costs. In addition, smaller drones like polycopters have become very affordable and have the ability to navigate in a very flexible way, including vertical maneuvers and hovering over a specific location of interest.

However, planning a mission for drones can be quite challenging and complex because of several limitations. Their limited autonomy and takeoff weight may necessitate intermediate stops to change batteries or restore the payload. A mission can also have dynamic aspects, such as changes in the points/sites to be visited or in their relative importance. In addition, weather changes, different local weather conditions, unforeseen manoeuvres or battery degradation may lead to additional delays, increased energy consumption and decreased autonomy, which has to be taken into account to adjust the mission accordingly. Notably, besides the obvious objective of minimizing the mission time, there is also the equally important requirement of providing strong safety guarantees, namely to ensure the drone will not exhaust its energy reserves in the air. This is to avoid undesirable emergency landings, which not only renders the drone unavailable for a potentially long period of time until it is recovered by ground personnel, but can also result in material damages or even human injuries.

In this work, we focus on the uncertainty of the energy that needs to be spent by the drone in order to move between two points of interest in the target area. We formulate the problem as a general dynamic multiple vehicle routing problem, and tackle it using an online heuristic algorithm. Our main contributions are: (i) the development of an online algorithm for this problem, and (ii) the extensive evaluation of the algorithm over a wide range of scenarios, varying the placement of depot (battery changing) stations, the uncertainty of the travel costs and the size of fleet. Our results show that the proposed heuristic can lead to significant optimizations.

The rest of the paper is structured as follows. We start by giving a brief overview of related work, in Section II. Then, in Section III, we formulate the problem and, in Section IV, describe the proposed algorithm. Section V presents our evaluation. Finally, Section VI concludes the paper.

II. Related

The routing and coordination of multiple UAVs has been researched for different application scenarios, objectives and assumptions. Besides routing, some works study the placement of drones in the mission area or on the cooperation between drones and trucks. A recent survey on UAV/drone routing problems can be found in [1]. Next, we briefly outline work that focuses on drones only.

The use of drones to assess the damage due to extreme weather conditions is studied in [2]. Initially, assuming stochastic weather events, the start positions of the drones are decided so as to cover a given target area while minimizing the total setup cost. When an event occurs, the paths to be followed by the drones are planned so as to minimize the operating cost and the mission time. Another placement problem is studied in [3], where a number of provisioning

This research has been co-financed by the European Union and Greek national funds through the Operational Program Competitiveness, Entrepreneurship and Innovation, under the call RESEARCH - CREATE - INNOVATE, project PV-Auto-Scout, code T1EDK-02435.

[1]Giorgos Polychronis is a PhD student at the Electrical and Computer Engineering Department, University of Thessaly, Volos, Greece gpolychronis@uth.gr

[2]Spyros Lalis is with the Faculty of the Electrical and Computer Engineering Department, University of Thessaly, Volos, Greece lalis@uth.gr

©2021 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works. DOI: 10.1109/ITSC48978.2021.9564911

facilities have to be placed in the mission area in order to maximize the delivery of supplies to a set of target locations while respecting the capacity of each facility and the flight range of the drones. During the mission, the drones visit one location at a time, returning each time to a facility, until their battery is exhausted. To cope with energy consumption uncertainty, only a percentage of the drone's battery capacity is utilized. In [4], multiple drones are controlled by different ground stations in order to perform a set of tasks. The objective is to complete the mission under certain constraints, such as fuel tank capacity, flight time, maximum number of drones per ground station, while optimizing various objectives, like the total fuel consumption, the number of drones used, the total travel distance and mission time as well as the risk of the mission. The latter depends on several factors, including having drones that run low on fuel during the mission.

The work in [5] considers drones carrying supplies to several target locations. Apart from the energy constraints, drones also have payload limitations and must return to depot stations in order to reload. The goal is to minimize the total distance, time or cost of the routes while supplying the target locations giving priority to the ones with the greater needs. A similar problem is studied in [6], where the drone's energy consumption is a function of its current weight (frame, battery and payload) and the objective is to minimize the operational cost given a delivery deadline, or to minimize the delivery time given a hard budget. [7] tackles the same problem for specific delivery time windows while using a more accurate model to estimate the drone's energy consumption. Yet another energy consumption model is introduced in [8], which is derived based on data taken from a power measurement module on a real drone in conjunction with the flight distance, speed and turn angles. The model's estimates are used in a path planning algorithm that minimizes the maximum energy consumption over all drones. The algorithm initially divides the target area in different regions, one for each drone, and then proceeds to plan the path for each drone separately.

There is also work on more dynamic aspects of drone planning/routing. The authors in [9] focus on drones that perform monitoring missions with a required frequency with the objective to minimize travel costs and avoiding penalties due to not meeting the monitoring requirements. Each path is associated with a different monitoring frequency, which may change dynamically in a deterministic or stochastic way. The placement/scheduling of drones in order to monitor both static and moving targets is studied in [10], so as to minimize the number of drones needed or the total energy consumption. A pickup and delivery problem is studied in [11] for drones that can swap their batteries at specific stations, with the objective to avoid having drones that run out of battery and to minimize delivery time and unnecessary movements. A similar problem is considered in [12] for a heterogeneous fleet of drones, taking into account the capacity, the energy and the maximum speed of each drone.

Although some works, like [4] and [3], acknowledge the issue of uncertainty regarding the energy consumption of drones, they do not deal with this issue during the mission. This problem is tackled in [13], which presents an algorithm that dynamically adjusts the paths of drones based on the actual energy consumption experienced during the mission. To guarantee safety and ensure that the drones will not exhaust their energy budget in flight, the algorithm works in a very pessimistic way, planning the paths to be followed by the drones based on the worst possible energy costs.

Our work tackles the same problem as [13]. However, we propose an optimistic heuristic, which plans paths based on more realistic estimates about the energy costs, while still keeping the risk of energy exhaustion to zero. Thanks to its optimism, the algorithm is able to find consistently better schedules across a very wide range of scenarios.

III. Problem formulation

Our work is motivated by drone-based applications, in particular those using polycopters, such as quadcopters and hexacopters, which have limited autonomy. Still, the problem we study holds for other types of drones and vehicles as well. Therefore, we formulate the problem as a general multi vehicle routing problem.

A. Mission area topology

The target area is modelled as a directed graph $\mathcal{G} = (\mathcal{N}, \mathcal{E})$, where $\mathcal{N}$ is a set of nodes n_i and $\mathcal{E}$ is the set of edges $e_{i,j}$ representing the ability of a vehicle to move directly from node n_i to node n_j. In order for the vehicle to move, it has to spend some of its energy. However, it can also gain energy at so-called depot nodes, which represent battery switching or refueling stations. Without loss of generality, we assume that the set of depot nodes $\mathcal{D}$ is disjoint from the set of nodes $\mathcal{V}$ that have to be visited by the vehicles, $\mathcal{D} \cap \mathcal{V} = \emptyset$. The nodes in $\mathcal{G}$ include the nodes to be visited and the depot nodes, $\mathcal{N} = \mathcal{V} \cup \mathcal{D}$.

B. Energy costs and gains

Each vehicle has a maximum energy capacity B. This represents the capacity of its battery (or the size of its fuel tank). The vehicle's energy reserves change as it moves between nodes. More specifically, when the vehicle traverses an edge $e_{i,j}$, the remaining energy is reduced. Let $c_{i,j}$ be the energy cost for the hop from n_i to n_j. Conversely, when a vehicle is on a depot node $n_i \in \mathcal{D}$ it restores its energy to the full capacity B. We assume that depots can always fully restore the energy reserves of any vehicle.

Let b be the energy/battery reserves of the vehicle at a given point in time. Then, the energy that remains available if the vehicle moves from node n_i to n_j is:

$$rem1(b, n_i, n_j) = b - c_{i,j} \tag{1}$$

Note that if the start node is a depot, $n_i \in \mathcal{D}$, the vehicle will perform this hop starting with the maximum energy reserves $b = B$. If $rem1(b, n_i, n_j) \leq 0$ then the vehicle has exhausted its energy and stops operating without completing the hop. The energy gain at the destination node n_j (if this is a depot) is not taken into account in Equation 1 because

it cannot be used to perform the hop in question. However, the depot can be exploited before attempting the next hop.

C. Paths and path feasibility

We model a path p as a node sequence $p[k], 1 \leq k \leq |p|$, where $|p|$ is the number of nodes in p. A path is non-empty if it includes at least one hop, $|p| \geq 2$. Note that in order for $p[k] = n_i \wedge p[k+1] = n_j$ to hold, there has to be a corresponding edge $e_{i,j} \in \mathcal{E}$. We also say that $e_{i,j} \in p$. Also, let $p[k1 : k2]$ denote the part of p starting from node $p[k1]$ and ending at node $p[k2]$.

Based on the above, the remaining energy of the vehicle if it starts with energy reserves b and travels along path p, can be expressed as follows:

$$rem(b,p) = \begin{cases} rem1(b, p[1], p[2]) & |p| = 2 \\ rem(rem1(b, p[1], p[2]), p[2 : |p|]) & |p| > 2 \end{cases} \quad (2)$$

Namely, if p consists of a single hop, the remaining energy is given by Equation 1. If p includes more than one hops, the energy that remains available at the end of p is equal to the remaining energy for the path without the first hop, starting with the energy that remains after taking the first hop of p. In the same spirit as above, this does not include the energy gain at the final destination node even if this is a depot.

We say that path p is feasible for initial energy/battery reserves b, if $rem(b, p[1 : k]) > 0, \forall k : 1 < k \leq |p|$. In other words, p is feasible if the vehicle will not exhaust its energy in any hop along p. Finally, let $nodes(p)$ denote the set of nodes that are part of p.

D. Schedule and time aspects

We assume M vehicles that can be used to follow different paths in parallel. A schedule (or plan) s consists of the paths that are assigned to the vehicles. In the general case, each vehicle will have several paths $paths[k], 1 \leq k \leq |paths|$ where $|paths|$ is the total number of paths assigned to it. Each such path starts from a depot node, $paths[k][1] \in \mathcal{D}$, and ends at a depot $paths[k][|paths[k]|] \in \mathcal{D}$. Further, the end node of a path is the same as the start node of the next path, $paths[k][|paths[k]|] = paths[k+1][1]$. The reason for assigning several paths to a vehicle is to make intermediate stops at depot nodes in order to restore its energy reserves so that it can safely visit additional nodes of interest. Also, the vehicles initially start the mission from a depot and have to return back to a depot after the mission completes.

Let $t_{i,j}$ be the time needed for a vehicle to traverse $e_{i,j}$. We assume that $t_{i,j}$ is proportional to $c_{i,j}$. Then, the completion time of path p is $d(p) = \sum_{e_{i,j} \in p} t_{i,j}$, and the total time needed for vehicle m to complete all the paths assigned to it, is $d(m) = \sum_{k=1}^{|paths_m|} d(paths_m[k])$. We assume that the restoration of the vehicle's energy reserves at a depot is fast compared to the travel time and does not affect $d(m)$ in a significant way. Finally, since vehicles travel in parallel to each other, the makespan (or completion time) of schedule s is $max_{m=1}^{M} d(m)$. This is the time it takes for the last vehicle to complete all the paths that have been assigned to it.

E. Problem statement

The objective is to find a feasible schedule s such that all nodes of interest are visited $\cup_{m=1}^{M} nodes(paths_m) = \mathcal{V}$ while minimizing the makespan $max_{m=1}^{M} d(m)$.

Our work tackles a dynamic version of the problem, where there is some uncertainty about the travel costs that will occur during the mission. More specifically, we assume that the edge costs $c_{i,j}$ follow a known distribution in the range $[c_{i,j}^{min} .. c_{i,j}^{max}]$ but the actual cost that will be incurred when a vehicle performs a hop is not known beforehand. Since $t_{i,j}$ is proportional to $c_{i,j}$, there is also uncertainty regarding the time it takes to perform a hop. An offline solution to this problem can be quite suboptimal. Therefore, we propose an online method that takes planning decisions at runtime.

We study the problem for the case where there is a single depot node, let this be n_d. Thus, every path p assigned to a vehicle starts from this node, $p[1] = n_d$, and ends at this node, $p[|p|] = n_d$. Also, in order for the problem to be always solvable, we assume that B is sufficiently large so that, starting from n_d, a vehicle can visit any node $n_i \in \mathcal{V}$, and then return back to n_d, even if every single hop over edge $e_{i,j}$ incurs the maximum possible cost $c_{i,j}^{max}$.

IV. Algorithm

We propose a heuristic that starts from an initial schedule (path plan) and then adjusts it during the mission time using large neighbourhood search (LNS). The main characteristic of the algorithm is that the paths are planned based on *optimistic estimates* about the travel costs. If the actual costs turn out to be lower, an attempt is made to find a better plan. Else, if a path turns out to be risky based on the current energy reserves, a detour is introduced back to the depot. In the following, we explain the most important elements of the heuristic and provide an algorithmic description for it.

A. Schedule representation and state information

The schedule is internally captured using M tuples of the form $\langle paths[], rem_{est}[], curr, rem_{curr}, pos, rem_{pos} \rangle$, one for each vehicle. The $paths[]$ field is the list of paths assigned to the vehicle (as discussed in Section III-D), sorted in descending order with respect to their energy costs. Also, $rem_{est}[k]$ is the remaining energy of the vehicle at the end of $paths[k]$ as estimated in the planning phase. This is equal to $rem(B, paths[k])$ since each path starts from the depot.

Each vehicle follows the paths in the order these appear in its $paths[]$ list. The index of the path that is currently being followed is kept in $curr$. The estimate for the energy that will remain available at the end of the current path is updated after each hop, and is kept in rem_{curr}. This is used to detect major deviations vs. the initial estimate $rem_{est}[curr]$. Finally, pos records the position of the vehicle in the current path, while rem_{pos} is the remaining energy of the vehicle at that position. Each time the vehicle begins to pursue a new path, $pos = 1$. Also, $rem_{pos} = B$ since every path starts from the depot node n_d.

B. Cost estimation

Path construction is done so as to minimize the makespan (Section III-D) while ensuring feasibility (Section III-C). Since the actual travel costs are unknown, feasibility checks are based on estimations about the energy that will be required by the vehicle to perform the planned hops.

More specifically, the algorithm uses a plug-in function $est()$ that provides estimates for the travel costs. This function is configurable to be able to experiment with different estimation methods without modifying the algorithm. The most conservative approach is let $est(c_{i,j}) = c_{i,j}^{max}$, i.e., to pessimistically assume that each hop will incur the maximum possible energy cost. In this work, we explore more optimistic estimates which are lower than the worst-case costs.

Notably, the algorithm is designed to be fully safe. In other words, it guarantees that all vehicles are able to return to the depot node, even in the worst-case scenario where the travel costs are the maximum possible. This safety is achieved in an elaborate way, making it possible to find better schedules compared to fully conservative planning.

C. Core logic

Given an initial schedule, each vehicle starts with the first path in its list. Thus, $curr = 1$, $rem_{curr} = rem_{est}[curr]$, $pos = 1$ and $rem_{pos} = B$ (every path starts from the depot). Then, as long as some vehicles are still active, their status is tracked and the plan is adapted accordingly. The core logic of the heuristic is given in Algorithm 1.

After each hop, pos is incremented, rem_{pos} is set to the remaining energy at this position, and rem_{curr} is updated to contain the new estimate for the remaining energy at the end of the current path, based on the actual costs for the hops that have been performed so far. It is then checked whether, based on rem_{pos}, the vehicle can afford to make the next hop, from node $n_j = paths[curr][pos]$ to $n_k = paths[curr][pos + 1]$, and still be able to return to the depot n_d, even under worst-case travel costs. This safety check is expressed as

$$rem_{pos} - c_{j,k}^{max} - c_{k,d}^{max} > 0 \tag{3}$$

If Equation 3 does not hold, a so-called detour is introduced so that the vehicle returns to the depot, and the rest of the path is added as a new path starting from the depot. Also, a replan attempt is made just in case the vehicle may visit some more nodes of interest on its way back to the depot.

If Equation 3 holds, it is checked whether the updated estimate for the remaining energy at the end of the current path is significantly higher than the initial estimate

$$\frac{rem_{curr} - rem_{est}[curr]}{rem_{est}[curr]} \geq Dev_{replan} \tag{4}$$

in which case, a replan attempt is made. The rationale is to exploit the extra available energy to find a better plan. The replan threshold Dev_{replan} is configurable. Note that if the cost estimation is pessimistic, assuming the maximum travel costs, it is guaranteed that Equation 3 will always hold. As a consequence, no detours will ever be produced in this case.

Algorithm 1 Online path planning algorithm.

function PLAN($s, \mathcal{V}, n_d$)
  while $\exists m : s[m].curr \leq |s[m].paths|$ **do**
    $replan \leftarrow false$
    for each m that performs its next hop **do**
      UPDATEVEHICLESTATE($s[m]$)
    end for
    if $replan$ **then** LNS($s, \mathcal{V}, n_d$)
    end if
  end while
end function

function UPDATEVEHICLESTATE($s[m]$)
  if $pos = |paths[curr]|$ **then** ▷ was last hop
    if $curr < |paths|$ **then** ▷ not done
      $curr \leftarrow curr + 1$ ▷ start next path
      $pos, rem_{pos} \leftarrow 1, B$ ▷ restore at the depot
      $rem_{curr} = rem_{est}[curr]$
    else
      $replan \leftarrow true$ ▷ exploit free vehicle
    end if
  else
    $n_i \leftarrow paths[curr][pos]$
    $n_j \leftarrow paths[curr][pos + 1]$
    $pos \leftarrow pos + 1$
    $rem_{pos} \leftarrow rem_{pos} - c_{i,j}$
    $rem_{curr} \leftarrow rem_{curr} + (est(c_{i,j}) - c_{i,j})$
    $n_k \leftarrow paths[curr][pos + 1]$
    if $rem_{pos} - c_{j,k}^{max} - c_{k,d}^{max} \leq 0$ **then**
      ADDDETOUR($paths, curr, pos$)
      $replan \leftarrow true$ ▷ exploit return to depot
    else if $\frac{rem_{curr} - rem_{est}[curr]}{rem_{est}[curr]} \geq Dev_{replan}$ **then**
      $replan \leftarrow true$ ▷ exploit extra energy
    end if
  end if
end function

function ADDDETOUR($paths, curr, pos$)
  $p = paths[curr]$
  $p1 \leftarrow p[1 : pos] + n_d$ ▷ detour to depot
  $p2 \leftarrow n_d + p[pos + 1 : |p|]$ ▷ remaining path
  replacePath($paths, curr, p1$)
  insertPath($paths, p2$)
end function

Finally, a replan attempt is made when a vehicle completes the paths assigned to it. This is to correct any imbalance by letting the free vehicle take over some of the nodes assigned to other vehicles, which can further reduce the makespan.

D. Path replanning

The replanning step is performed by running a large neighbourhood search (LNS) algorithm [14]. In a first step, some nodes are removed from the planned paths. We use a proximity-based policy, which randomly picks some nodes and then removes those nodes along with their nearest

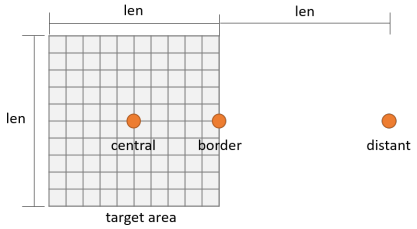


Fig. 1: Depot placements with respect to the target area.

neighbours (based on the Euclidean distance between them). In a second step, these nodes are re-inserted to some (possibly different) paths. This is done using a greedy heuristic, whereby a node is inserted in a path so that the total makespan of the schedule is minimized. Ties are broken by picking the insertion that minimizes the increase in the completion time of the vehicle to which the node is assigned.

The feasibility of the constructed paths is checked based on the estimated travel costs $est(c_{i,j})$. In addition, the safety check as per Equation 3 is performed before changing the next hop in the current path of any vehicle, ensuring that the vehicle will be able to return to the depot even under the worst-case travel costs. If a node can not be inserted in any path under these constraints, a new path is formed, with that node being the only node in the path. This path is assigned to a vehicle so that the schedule's makespan is minimized.

V. Evaluation

We evaluate the algorithm for various scenarios regarding the depot placement, the uncertainty of travel costs, the degree of optimism and the number of vehicles, via simulations.

A. Topology

The target area in our experiments is a square region where the points of interest (nodes) are arranged in a 11×11 grid, as illustrated in Figure 1; the nodes are at the line intersections. We assume a region where vehicles can move in a straight line between any two nodes, which is realistic for drones.

For the depot node, we investigate three placement scenarios. In the *central* placement scenario, the depot is at the center of the target area. In the *border* scenario, it is at the borderline of the area. Finally, in the *distant* scenario, the depot is placed further away from the target area, at a distance equal to the size of the edge of the square area.

B. Energy costs and uncertainty

The edge/travel costs $c_{i,j}$ follow a uniform distribution $[c_{i,j}^{min}..c_{i,j}^{max}]$, with an expected cost $c_{i,j}^{avg}$. We set $c_{i,j}^{avg}$ equal to the Euclidean distance between n_i and n_j. The minimum and maximum costs depend on the degree of uncertainty. In *low* uncertainty, $c_{i,j}^{min} = c_{i,j}^{avg} \times \frac{3}{4}$ and $c_{i,j}^{max} = c_{i,j}^{avg} \times \frac{5}{4}$. In *high* uncertainty, we set $c_{i,j}^{min} = c_{i,j}^{avg} \times \frac{2}{3}$ and $c_{i,j}^{max} = c_{i,j}^{avg} \times \frac{4}{3}$. Without loss of generality, we let travel time be a linear function of edge cost.

To ensure that the problem has a safe solution, we set the maximum capacity B of the vehicles equal to the round trip cost from the depot to the farthest node(s), assuming the worst-case cost $c_{i,j}^{max}$ for each hop. Thus, B has a larger value in scenarios with a higher cost uncertainty and/or where the depot is further away from the center of the target area.

C. Cost estimation

We test the algorithm for two optimistic variants of the cost estimation function $est(c_{i,j})$. In the *aggressive* variant, the function returns $c_{i,j}^{avg}$, whereas in the *moderate* variant it returns $\frac{c_{i,j}^{avg}+c_{i,j}^{max}}{2}$. Note that the latter estimate is more conservative compared to the former. We run all variants with a replan threshold of $Dev_{replan} = 5\%$.

As a reference, we also run the algorithm in the most *pessimistic* variant, where the cost function returns the worst-case cost $c_{i,j}^{max}$ (this is equivalent to the algorithm described in [13]). Moreover, we report the results achieved by an offline *oracle* algorithm that has perfect a priori knowledge of the actual cost of each hop during the mission.

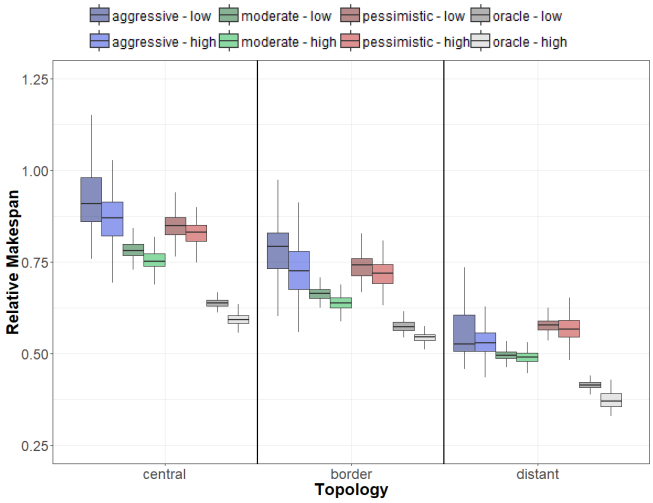
D. Results

For each depot placement and cost uncertainty scenario, we run the variants of the algorithm for 5 different initial schedules and 50 different randomly generated edge cost settings (based on the respective cost distribution). The initial placements are produced using the offline algorithm in [15], which finds good results. The cost estimation function is the same as in the online algorithm, depending on the variant.

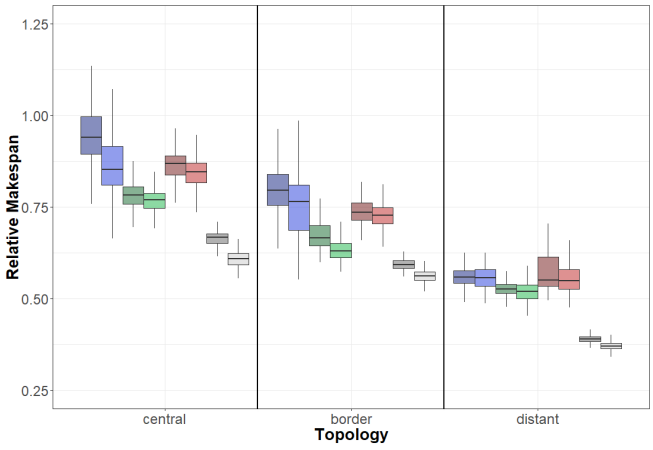
Figure 2 shows the relative makespan achieved in each scenario by the different configurations of the algorithm for a fleet of 1, 2 and 3 vehicles. Each boxplot summarizes the results obtained in the 250 experiments (5 initial schedules, each for 50 travel cost settings). To make a fair comparison, between plans that are safe and ensure that the vehicles do not exhaust their energy, the reference for the online variants is the plan produced offline using pessimistic cost estimates. Note that the absolute values are different for each topology and uncertainty. Also, since different uncertainties lead to different actual travel costs, the respective schedules produced by the oracle differ too in each case.

We observe that all online variants achieve increasingly better results as the depot is placed further away from the target area. This is because the offline plan becomes increasingly suboptimal as the worst-case travel costs increase, and thus there is more room for improvement. Similarly, the results achieved for the high uncertainty are generally better than for low uncertainty. Again, the reason is that higher uncertainty increases the worst-case costs thereby creating more room for improvement vs. the offline solution.

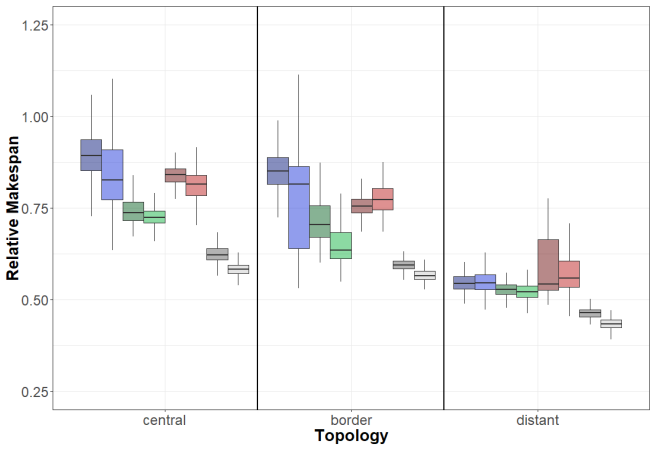
Notably, the aggressive variant achieves worse results than the pessimistic variant in the central and border depot scenarios. It manages to achieve consistently equal or slightly better results only in the distant depot scenario. The reason is that it is overly optimistic and often produces paths which turn out to be risky (paths where at some point Equation 3 no longer holds). This, in turn, leads to a large number of detours (see Figure 3a) which also translate to more trips back to the depot (see Figure 3b).



(a) 1 vehicle

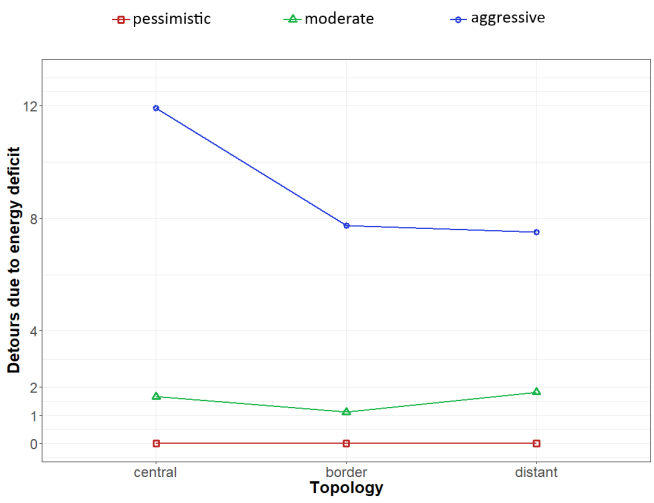


(b) 2 vehicles



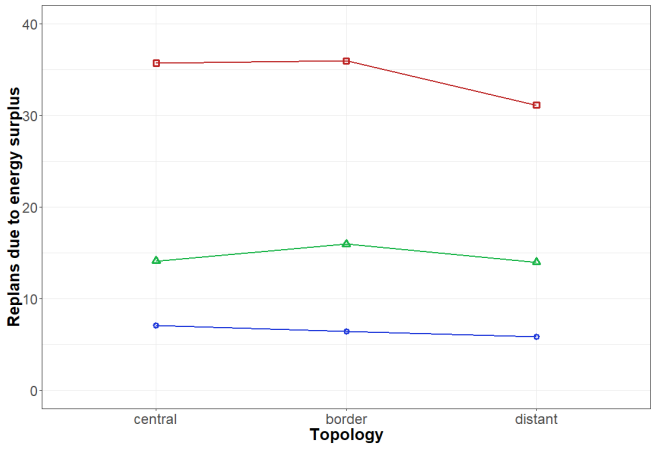
(c) 3 vehicles

Fig. 2: Relative makespan vs. offline for the different depot placements with 1, 2 and 3 vehicles.



(a) Detours due to energy deficit

(b) Depot visits



(c) Replans due to energy surplus

Fig. 3: Features of the plans produced for the different depot placements (average over all uncertainty/vehicle scenarios).

In contrast, the moderate variant outperforms both the pessimistic and the aggressive variant in all cases, reducing the makespen by up to 51% vs. the offline solution and up to 18% vs. the pessimistic variant. This shows that optimistic planning can lead to significantly better results – provided the degree of optimism is not excessive. Note that in the distant depot scenario, the difference with the pessimistic variant becomes quite significant for 1 vehicle, whereas all variants achieve similar results for 2 and 3 vehicles. Also, in the distant depot scenario with 3 vehicles, the results become

quite close to the ones of the oracle. This is because a larger number of vehicles decreases the number of assigned paths per vehicle, which also reduces the opportunities for very significant optimizations.

As can be seen in Figure 3a, the moderate variant introduces significantly fewer detours than the aggressive variant; in fact, the number of detours remains practically constant across all depot placements. This clearly shows that the paths produced by the moderate variant are more realistic compared to those of the aggressive variant. The aggressive variant introduces significantly fewer detours in the border and distant depot scenarios. In these cases, vehicles have larger energy capacity hence any cost deviations that occur when the vehicle is in the target area become less critical with respect to the safety of the planned paths (it becomes more likely for Equation 3 to hold despite cost deviations). Note that the pessimistic variant does not perform any detours because the actual costs cannot possibly be larger than the (worst-case) estimates.

In Figure 3b, we report the number of intermediate depot visits in order for the vehicles to restore their energy reserves during the mission. The moderate variant consistently leads to fewer visits than all other variants. This difference is more significant in the central and border depot scenarios, which also explains the notably lower makespan achieved in those cases (see Figure 2). Note that the aggressive variant leads to almost as many depot visits as the pessimistic variant. Still, the makespan of the aggressive variant is worse or equal to the pessimistic variant in the central and border depot placement scenarios (see Figure 2). We attribute this to the fact that the aggressive variant performs a much larger number of detours. Even though each detour practically also triggers a replan, which may avoid a depot visit (as can be seen by comparing Figure 3a with Figure 3b), replanning out of need, under the pressure of energy deficit, is apparently more disruptive and leads to lower-quality paths compared to replanning under energy surplus (also compared to the initial path planning that is performed offline).

Figure 3c shows the number of replans due to energy surplus. Note that the trend is opposite to that of the detours, namely the number of replans decreases with increasing degree of optimism. The more optimistic the estimate of travel costs, the less likely it is for the actual aggregate cost to be smaller than the estimate, so that the respective deviation becomes not only positive but also larger than the replan threshold. As expected, the pessimistic variant performs the largest number of replans. Since all cost estimates are made based on the maximum possible costs, the actual costs will turn out to be significantly lower with high probability hence Equation 4 holds more often compared to other variants.

Finally, it is worth noting that the moderately optimistic variant performs quite well even compared to the oracle. More specifically, the average makespan over all uncertainty, depot placement and vehicle scenarios, is only about $1.2x$ larger than that of the oracle. Nevertheless, this also shows that there is still some room for improvement, which we intend to explore in future work.

VI. Conclusion

We have presented a new, optimistic algorithm for tackling the dynamic multiple vehicle routing problem under uncertainty about the energy that will be spent for travel. Our evaluation for a wide range of scenarios shows that the algorithm can significantly reduce the makespan of a mission vs offline planning. Also, if the level of optimism is kept moderate, the algorithm consistently outperforms the pessimistic online variant and reduces the number of depot visits during the mission.

Motivated by the room for improvement vs the plans that are produced by the oracle algorithm, we intend to investigate adaptive cost estimation policies, which adjust the level of optimism as a function of the travel costs that are experienced during the mission. We also wish to explore scenarios where the degree of uncertainty for the travel costs changes in time and/or varies between different regions within the target area.

References

[1] D. Rojas Viloria, E. L. Solano-Charris, A. Muñoz-Villamizar, and J. R. Montoya-Torres, "Unmanned aerial vehicles/drones in vehicle routing problems: a literature review," *Intl Transactions in Operational Research*, vol. 28, pp. 1626–1657, 2021.

[2] G. J. Lim, S. Kim, J. Cho, Y. Gong, and A. Khodaei, "Multi-uav pre-positioning and routing for power network damage assessment," *IEEE Transactions on Smart Grid*, vol. 9, no. 4, pp. 3643–3651, 2016.

[3] D. Chauhan, A. Unnikrishnan, and M. Figliozzi, "Maximum coverage capacitated facility location problem with range constrained drones," *Transportation Research Part C: Emerging Technologies*, vol. 99, pp. 1–18, 2019.

[4] C. R. Atencia, J. Del Ser, and D. Camacho, "Weighted strategies to guide a multi-objective evolutionary algorithm for multi-uav mission planning," *Swarm and Evolutionary Computation*, vol. 44, pp. 480–495, 2019.

[5] B. Rabta, C. Wankmüller, and G. Reiner, "A drone fleet model for last-mile distribution in disaster relief operations," *Intl Journal of Disaster Risk Reduction*, vol. 28, pp. 107–112, 2018.

[6] K. Dorling, J. Heinrichs, G. G. Messier, and S. Magierowski, "Vehicle routing problems for drone delivery," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 47, no. 1, pp. 70–85, 2016.

[7] C. Cheng, Y. Adulyasak, and L.-M. Rousseau, *Formulations and exact algorithms for drone routing problem*. CIRRELT Tech. Report, 2018.

[8] J. Modares, F. Ghanei, N. Mastronarde, and K. Dantu, "Ub-anc planner: Energy efficient coverage path planning with multiple drones," in *IEEE Intl Conf. on Robotics and Automation (ICRA)*, 2017, pp. 6182–6189.

[9] J. Y. Chow, "Dynamic uav-based traffic monitoring under uncertainty as a stochastic arc-inventory routing policy," *Intl Journal of transportation science and technology*, vol. 5, no. 3, pp. 167–185, 2016.

[10] D. Zorbas, L. D. P. Pugliese, T. Razafindralambo, and F. Guerriero, "Optimal drone placement and cost-efficient target coverage," *Journal of Network and Computer Applications*, vol. 75, pp. 16–31, 2016.

[11] Y. Liu, "An optimization-driven dynamic vehicle routing algorithm for on-demand meal delivery using drones," *Computers & Operations Research*, vol. 111, pp. 1–20, 2019.

[12] B. N. Coelho, V. N. Coelho, I. M. Coelho, L. S. Ochi, R. Haghnazar, D. Zuidema, M. S. Lima, and A. R. da Costa, "A multi-objective green uav routing problem," *Computers & Operations Research*, vol. 88, pp. 306–315, 2017.

[13] G. Polychronis and S. Lalis, "Dynamic multiple vehicle routing under energy capacity constraints," in *IEEE Intl Conf. on Intelligent Transportation Systems (ITSC)*, 2020, pp. 1–7.

[14] P. Shaw, "Using constraint programming and local search methods to solve vehicle routing problems," in *Intl Conf. on Principles and Practice of Constraint Programming*, 1998, pp. 417–431.

[15] G. Polychronis and S. Lalis, "Tournament selection algorithm for the multiple travelling salesman problem." in *Intl Conf. on Vehicle Technology and Intelligent Transport Systems (VEHITS)*, 2020, pp. 585–594.