MOTION PLANNING FOR UNMANNED AERIAL VEHICLES WITH RESOURCE CONSTRAINTS

A Thesis

by

KAARTHIK SUNDAR

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

August 2012

Major Subject: Electrical Engineering

MOTION PLANNING FOR UNMANNED AERIAL VEHICLES WITH RESOURCE CONSTRAINTS

A Thesis

by

KAARTHIK SUNDAR

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Approved by:

Co-Chairs of Committee,	Shankar P. Bhattacharyya
	Sivakumar Rathinam
Committee Members,	Jean-François Chamberland-Tremblay
	Alex Sprintson
Head of Department,	Costas Georghaides

August 2012

Major Subject: Electrical Engineering

ABSTRACT

Motion Planning for Unmanned

Aerial Vehicles with Resource Constraints. (August 2012)

Kaarthik Sundar, B.E.,

College of Engineering Guindy, Anna University

Co–Chairs of Advisory Committee: Dr. Shankar P. Bhattacharyya
Dr. Sivakumar Rathinam

Small Unmanned Aerial Vehicles (UAVs) are currently used in several surveillance applications to monitor a set of targets and collect relevant data. One of the main constraints that characterize a small UAV is the maximum amount of fuel the vehicle can carry. In the thesis, we consider a single UAV routing problem where there are multiple depots and the vehicle is allowed to refuel at any depot. The objective of the problem is to find a path for the UAV such that each target is visited at least once by the vehicle, the fuel constraint is never violated along the path for the UAV, and the total length of the path is a minimum. Mixed integer, linear programming formulations are proposed to solve the problem optimally. As solving these formulations to optimality may take a large amount of time, fast and efficient construction and improvement heuristics are developed to find good sub-optimal solutions to the problem. Simulation results are also presented to corroborate the performance of all the algorithms. In addition to the above contributions, this thesis develops an approximation algorithm for a multiple UAV routing problem with fuel constraints.

ACKNOWLEDGMENTS

I owe my deepest gratitude to my advisor Dr Sivakumar Rathinam, for his constant guidance right from start of my graduate studies at Texas A&M. I am also grateful to my other advising committee members Dr Shankar Battacharyya, Dr Jean-François Chamberland and Dr Alex Sprintson for their support. I am indebted to my colleagues Harsha, Srikrishna, Adithya and Jung without whom this thesis would not have been possible. Finally, I thank my parents and my brother for supporting me throughout all my studies at the University.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER I

INTRODUCTION

Motion planning for small Unmanned Aerial Vehicles is one of the research areas that has received a lot of attention from the scientific community in the last decade. Small Unmanned Aerial Vehicles (UAVs) have already been field tested in civilian applications such as wild-fire management [1], weather and hurricane monitoring [2, 3, 4], and pollutant estimation [5] where the vehicles are used to collect relevant sensor information and transmit the information to the ground (control) stations for further processing. As compared to larger UAV platforms, small UAVs are relatively easier to operate and are significantly cheaper. Small UAVs can fly at low altitudes and can avoid obstacles or threats at low altitudes more easily. Even in military applications, smaller vehicles [6] are used frequently for intelligence gathering and damage assessment as they are easier to fly and can be hand launched by an individual without any reliance on a runway or a specific type of terrain.

Even though there are several advantages with using smaller platforms, they also come with other resource constraints due to their size and smaller payload. As small UAVs typically have fuel constraints, it may not be possible for an UAV to complete a surveillance mission before refueling at one of the depots. For example, consider a typical surveillance mission where a vehicle starts at a depot and is required to visit a set of targets. To complete this mission, the vehicle may have to start at the depot, visit a subset of targets and then reach one of the depots for refueling before starting a new path. One can reasonably assume that once the UAV reaches a depot, it will be refueled to full capacity before it leaves again for visiting any remaining targets.

The journal model is *IEEE Transactions on Automatic Control*.

If the goal is to visit each of the given targets at least once, then the UAV may have to repeatedly visit some depots in order to refuel again before visiting all the targets. In this scenario, the following **fuel-constrained, UAV routing problem** naturally arises: Given a set of targets, depots, and an UAV where the vehicle is initially stationed at one of the depots, find a path for the UAV such that each target is visited at least once by the vehicle, the fuel constraint is never violated along the path for the UAV, and the total cost of the edges present in the UAV path is a minimum. Please refer to Fig.1 for an illustration of this problem.

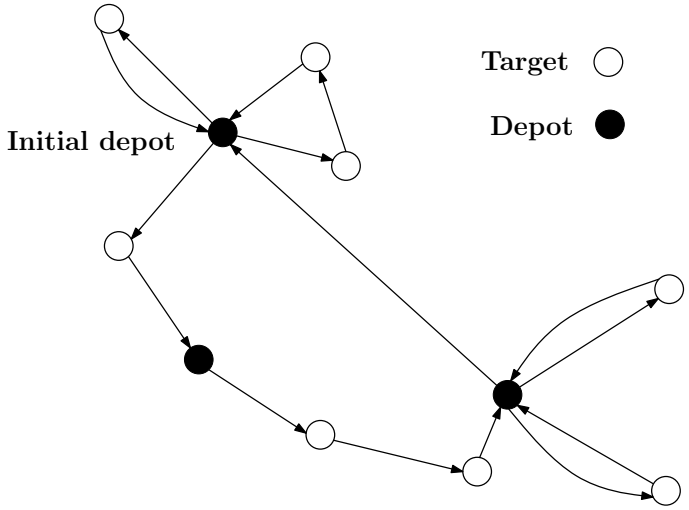


Fig. 1. A feasible plan for the UAV

This problem is quite different and more general compared to the regular fuel constrained TSP addressed in the literature [7] where one is interested in maximizing the number of targets visited by a vehicle subject to its fuel constraints. Apart from the approximation algorithm in [8], we are not aware of any existing formulations or heuristics or any computational results that are available for solving the fuel-constrained, UAV routing problem.

In this article, we will first assume that it is always cheaper to travel directly

from location A to location B than through any other intermediate location. If this assumption is satisfied for the UAV, the distances (or the fuel costs) are said to satisfy the triangle inequality. If this assumption is not satisfied, it is already known that there cannot exist any constant factor approximation algorithm even for a TSP unless $P = NP$. In addition to the above standard assumption, the authors in [8] assume that each target has at least one depot at a distance at most equal to $La/2$ units where a is a constant in the interval $[0, 1)$ and L is the maximum distance the vehicle can travel before it runs out of fuel. This is a reasonable assumption, as in any case, one cannot have a feasible tour if there is a target that cannot be visited from any of the depots. Using these assumptions, Khuller et al. [8] provide a $3(1 + a)/2(1 - a)$-approximation algorithm for the problem. The fuel constrained UAV routing problem addressed in this article is more general than the tour problem considered in [8] in the following way: we allow for the fuel required to travel from location A to location B to be different compared to the fuel required to travel from location B to A. The fuel costs are allowed to be asymmetric because the travel path for the UAV from location A to B may be quite different from the travel path from location B to A.

The following are the contributions of this thesis:

1. We develop two mixed-integer linear programs for the fuel constrained, UAV routing problem based on the single and multi-commodity flow formulations available for standard network synthesis problems in the literature. These formulations are mainly used to find the optimal solutions for the routing problem, and for corroborating the quality of the solutions produced by the heuristics.

2. Fast construction and improvement heuristics are developed to find feasible solutions to the fuel constrained, UAV routing problem. Even though the mixed integer, linear programming formulations can be used to find optimal solutions,

it may be time consuming to solve them. In addition, practical scenarios may only provide approximate input data about the locations of the targets, and as a result, it may be useful to find good, approximate solutions than find optimal solutions that are more difficult to solve. For this reason, we focus on developing several heuristics for the fuel constrained, UAV routing problem.

3. Computational results are presented to compare the performance of all the algorithms with respect to the quality of the solutions produced by the algorithms and their respective computation times. These computational results are presented for two scenarios: one where the vehicle does not have any kinematic constraints and on the other scenario where there is a bound on the maximum yaw rate of the vehicle.

4. We also consider a generalization of the fuel constrained UAV routing problem with multiple vehicles and present an algorithm with an approximation ratio of $\frac{2(1+a)}{(1-a)}$.

A. Organization of the thesis

The second chapter of the thesis defines the single vehicle problem with refueling constraints. After formally defining the problem, it discusses two mixed integer linear programming formulations to solve the problem to optimality. This is followed by heuristics to find sub-optimal solutions to the problem. The third chapter presents the computational results to corroborate the performance of all the algorithms. The final chapter develops an approximation algorithm for the multiple vehicle UAV routing problem with fuel constraints.

CHAPTER II

SINGLE VEHICLE WITH REFUELING CONSTRAINTS

A. Problem statement

Let T denote the set of targets and D represent the set of depots. Let $s \in D$ be the depot where the UAV is initially located. The Fuel Constrained Routing Problem (FCRP) is formulated on the complete directed graph $G = (V, E)$ with $V = T \bigcup D$. The cost of traveling from vertex $i \in V$ to vertex $j \in V$ is denoted by c_{ij}. Let f_{ij} represent the amount of fuel required by the vehicle to travel from vertex $i \in V$ to vertex $j \in V$. It is assumed that both the fuel spent and the travel costs satisfy the triangle inequality *i.e.*, for all distinct $i, j, k \in V$, $c_{ij} + c_{jk} \geq c_{ik}$ and $f_{ij} + f_{jk} \geq f_{ik}$.

Let L denote the maximum fuel capacity of the vehicle. We will assume that there is at least one depot d such that $f_{id} + f_{di} \leq L$ for any target $i \in T$. We will also assume that it is always possible to travel from one depot to any another depot (either directly or by passing through some intermediate depots) without violating the fuel constraints. The objective of the problem is to find a tour such that

- the tour starts and terminates at the initial depot,
- each target is visited at least once by the UAV,
- the fuel required to travel any segment of the tour which joins two consecutive depots in the tour must be at most equal to L, and,
- the sum of the cost of all the edges present in the tour is a minimum.

B. Problem formulation

In this section, we provide two problem formulations for the FCRP based on the single and multi commodity flow formulations [9] available for network synthesis problems. We first formulate the problem using the multi-commodity flow constraints. In this formulation, we pose the FCRP as a problem of synthesizing a network of edges such that a unit of commodity can be shipped from the depot to each of the targets using the vehicle. Essentially, the depot is the source of all the commodities and each target receives a distinct commodity from the depot. A commodity destined for a particular target may not accumulate at any of the intermediate locations. Let x_{ij} denote the binary decision variable which determines the presence of the edge (i, j) in the network; that is, x_{ij} is equal to one if the edge (i, j) is present in the network and is equal to zero otherwise. The vehicle must use the edges in the network to ship the commodities form the depot to the respective targets. For any target $k \in T$, let p_{ij}^k denote the k^{th} commodity flowing from vertex i to vertex j. Also, let r_i represent the fuel left in the vehicle when the i^{th} target is visited. Now, FCRP can be formulated as a mixed integer linear program as follows:

$$\min \sum_{(i,j)\in E} c_{ij}\ x_{ij},$$

subject to

Degree constraints:

$$\sum_{i\in V\setminus\{k\}} x_{ik} = \sum_{i\in V\setminus\{k\}} x_{ki} \quad \forall k \in V, \tag{2.1}$$

$$\sum_{i\in V\setminus\{k\}} x_{ik} = 1 \quad \forall k \in T, \tag{2.2}$$

Capacity and flow constraints (Multiple-commodity):

$$\sum_{j \in V \setminus \{s\}} (p_{ij}^k - p_{ji}^k) = 1 \quad \forall k \in T \text{ and } i = s, \tag{2.3}$$

$$\sum_{j \in V \setminus \{i\}} (p_{ij}^k - p_{ji}^k) = 0 \quad \forall i, k \in T \text{ and } i \neq k, \tag{2.4}$$

$$\sum_{j \in V \setminus \{j\}} (p_{kj}^k - p_{jk}^k) = -1 \quad \forall k \in T, \tag{2.5}$$

$$0 \leq \ p_{ij}^k \ \leq x_{ij} \quad \forall i, j \in V, \ \forall k \in T, \tag{2.6}$$

Fuel constraints:

$$r_j - r_i + f_{ij} \leq M(1 - x_{ij}) \quad \forall i, j \in T, \tag{2.7}$$

$$r_j - r_i + f_{ij} \geq -M(1 - x_{ij}) \quad \forall i, j \in T, \tag{2.8}$$

$$r_j - L + f_{ij} \geq -M(1 - x_{ij}) \quad \forall i \in D \text{ and } j \in T, \tag{2.9}$$

$$r_j - L + f_{ij} \leq M(1 - x_{ij}) \quad \forall i \in D \text{ and } j \in T, \tag{2.10}$$

$$r_i - f_{ij} \geq -M(1 - x_{ij}) \quad \forall i \in T \text{ and } j \in D, \tag{2.11}$$

$$\sum_{i \in V} \sum_{j \in V} f_{ij} \ x_{ij} \leq L \quad \sum_{k \in D} \sum_{i \in V} x_{ki}, \tag{2.12}$$

$$\begin{aligned} 0 \leq r_i \leq L \quad \forall i \in T, \\ x_{ij} \in \{0, 1\} \quad \forall (i, j) \in E, \\ p_{ij}^k \in \mathbb{R}^+ \quad \forall i, j \in V \text{ and } k \in T. \end{aligned} \tag{2.13}$$

Equation (2.1) states that the in degree of a vertex is equal to the out degree of a vertex, and equation (2.2) ensures that each target is visited once by the vehicle. Note that these equations imply that a depot can be visited any number of times for refueling. Equations (2.3), (2.4), and (2.5) formulate the requirement that each

target must receive one unit of commodity and a commodity destined to a target cannot accumulate at any intermediate locations. Constraint (2.6) states that any commodity can only flow through edges that are present in the network. These flow constraints ensure that there exists a path from the source depot s to each of the targets.

If the UAV is traveling from target i to target j, equations (2.7) and (2.8) ensures that the fuel left in the vehicle after reaching target j is $r_j = d_i - f_{ij}$. In equations, (2.7)-(2.12), M denotes a large constant and can be chosen to be equal to $L + \max_{i,j \in V} f_{i,j}$. If the UAV is traveling from a depot i to a target j, equations (2.9), (2.10) ensures that the fuel left in the vehicle after reaching target j is $r_j = L - f_{ij}$. If the UAV is directly traveling from any target to a depot, constraint (2.11) states that the fuel remaining at the target must be at least equal to the amount required to reach the depot. Equation (2.12) states that the total fuel consumed by the UAV must be at most equal to L times the total number of visits to all the depots.

In the above formulation, the capacity and the flow constraints mainly ensure that each target is connected to the depot. There are also other ways of expressing this connectedness. Specifically, instead of shipping a distinct unit of commodity to each target, one can also ship $|T|$ units of the same commodity from the source depot to all the targets. In this method, the vehicle will deliver one unit of commodity at each target as it travels along its path. Suppose p_{ij} denotes the amount of commodity flowing from vertex i to vertex j. Then the multi-commodity flow constraints in (2.3-2.6) can be replaced with the following single-commodity flow constraints as follows:

Capacity and flow constraints (Single-commodity):

$$\sum_{i \in V \setminus \{s\}} (p_{si} - p_{is}) = |T|, \tag{2.14}$$

$$\sum_{j \in V \setminus \{i\}} (p_{ji} - p_{ij}) = 1 \quad \forall i \in T, \tag{2.15}$$

$$\sum_{j \in V \setminus \{i\}} (p_{ji} - p_{ij}) = 0 \quad \forall i \in D \setminus \{s\}, \tag{2.16}$$

$$0 \leq \ p_{ij} \ \leq |T| x_{ij} \quad \forall i, j \in V. \tag{2.17}$$

The constraints in (2.14)-(2.17) ensure that there are $|T|$ units of commodity shipped from the depot and the vehicle delivers exactly one of commodity at each target. In summary, the single-commodity flow formulation of the FCRP aims to minimize $\sum_{(i,j) \in E} c_{ij} \ x_{ij}$ subject to the degree constraints in (2.1)-(2.2), flow constraints in (2.14)-(2.17), fuel constraints in (2.7)-(2.12) and the constraints in (2.13). An advantage of the single-commodity formulation is that it has a fewer number of flow variables. However, it is also known that the multi-commodity flow formulation provides better lower bounds [9] as compared with the single-commodity flow formulation. The effectiveness of these formulations will also depend on the specific application and the size of the problem. Later, in the next chapter, we will present some numerical results to compare the performance of both these formulations.

C. Heuristics for the FCRP

In the following sections, we first provide a tour construction heuristic which finds an initial feasible solution to the problem. This heuristic is a generalization of the approximation algorithm presented by Khuller et al. [8] for the symmetric version of the problem. Then, we provide improvement heuristics based on the well known $k-$opt methods available for sequencing problems.

D. Construction heuristic

The **first step** of the construction heuristic aims to find a path for the vehicle to travel from any target $x \in T$ to any other target $y \in T$ such that the path can be a part of a feasible tour for the FCRP, the path satisfies all the refueling constraints and the sum of the cost of traveling all the edges in the path is a minimum. Note that the maximum amount of fuel available for the vehicle when it reaches target x in any tour is $L - \min_d f_{dx}$. Also, in any feasible tour, there must be at least $\min_d f_{yd}$ units of fuel left when the vehicle reaches target y so that the vehicle can continue to visit other vertices along its tour. For any target $x \in T$, let $C_x := \min_d f_{dx}$ and $D_x := \min_d f_{xd}$. The first step of the construction heuristic essentially finds a feasible path of least cost (also referred as the shortest path) such that the vehicle starts at target x with at most $L - C_x$ units of fuel and ends at target y with at least D_y units of fuel. If there is enough fuel available for the vehicle to travel from x to y (or, if $L - C_x - D_y \geq f_{xy}$), the vehicle can directly reach y from x while respecting the fuel constraints. In this case, we say that the vehicle can *directly* travel from x to y and the shortest path is denoted by $PATH_{xy} := (x, y)$. The cost of traveling this shortest path is just c_{xy}.

If the vehicle *cannot directly* travel from x to y (if $L - C_x - D_y < f_{xy}$), the vehicle must visit some of the depots on the way before reaching target y. In this case, we find a shortest path using an auxiliary directed graph, (V', E'), defined on all the depots and the targets x, y, *i.e.*, $V' = \{T \bigcup \{x, y\}\}$ (illustrated in Figure 2). An edge is present in this directed graph only if traveling the edge can satisfy the fuel requirements of the vehicle. For example, as the vehicle has at most $L - C_x$ units of fuel to start with, the vehicle can reach a depot d from x only if $f_{xd} \leq L - C_x$. Therefore, E' contains an edge (x, d) if the constraint $f_{xd} \leq L - C_x$ is satisfied.

Similarly, the vehicle can travel from a depot d to target y only if there are at least D_y units of fuel remaining after the vehicle reaches y. Therefore, E' contains an edge (d, y) if the constraint $f_{dy} \leq L - D_y$ is satisfied. In summary, the following are all the edges that are present in E':

$$E' := \begin{cases} \{(x, d) : \forall d \in D, f_{xd} \leq L - C_x\}, \\ \bigcup\{(d_1, d_2) : \forall d_1, d_2 \in D, f_{d_1 d_2} \leq L\}, \\ \bigcup\{(d, y) : \forall d \in D, f_{dy} \leq L - D_y\}. \end{cases} \tag{2.18}$$

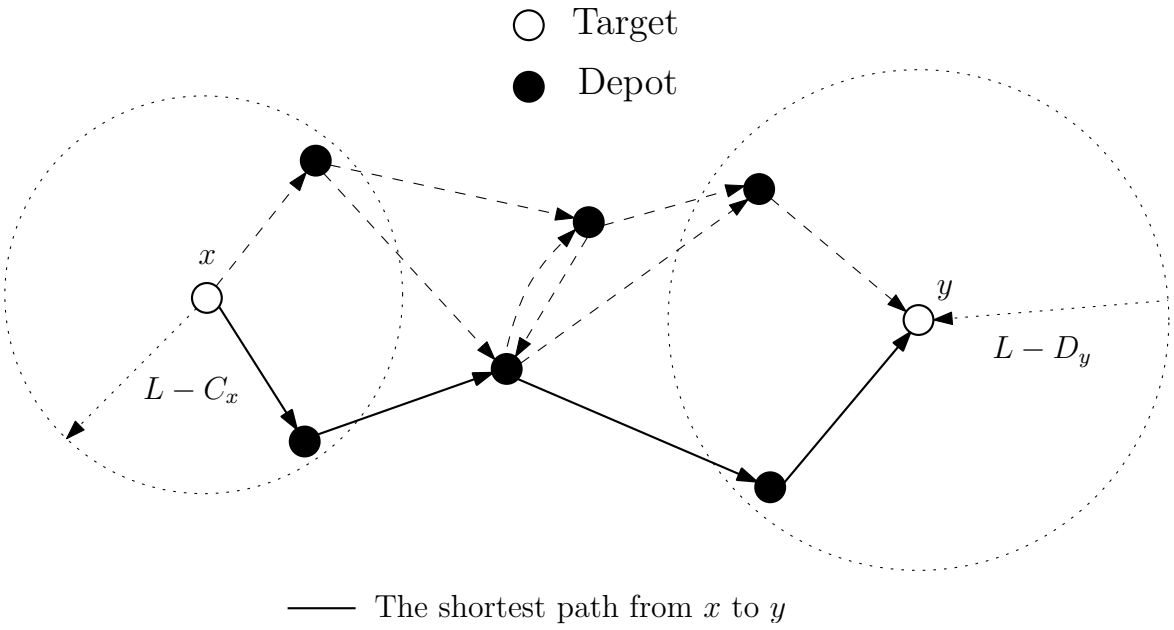


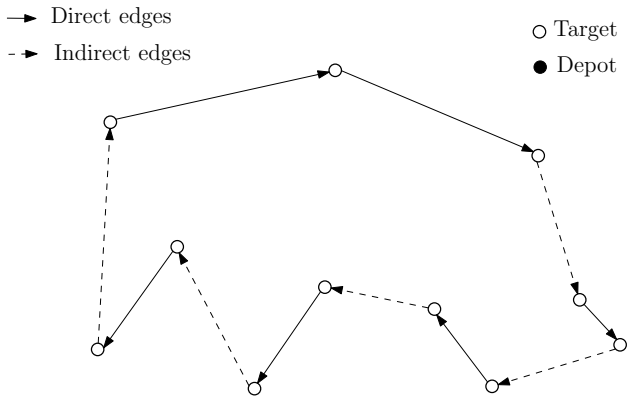
Fig. 2. The first step of the construction heuristic: Computation of l_{xy} for an indirect edge from target x to target y.

In Fig.2 the solid edges represent the shortest path from target x to target y.

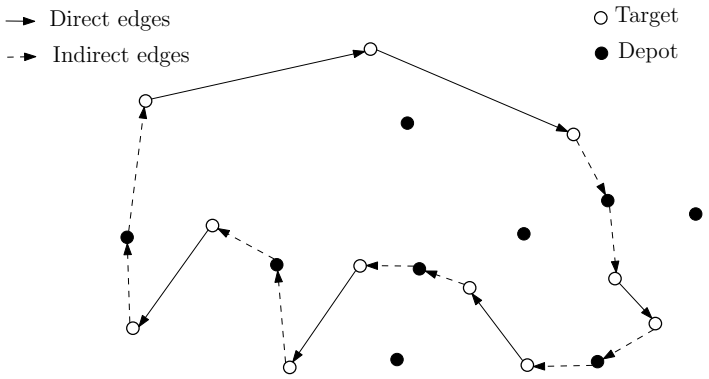
Any path starting at x and ending at y in this auxiliary graph will require the vehicle to carry at most $L - C_x$ units of vehicle at target x, satisfy all the fuel constraints and reach target y with at least D_y units of fuel left. Also, we let the cost

of traveling any edge $(i, j) \in E'$ to be equal to c_{ij} (as defined in section A). Now, we use Dijkstra's algorithm to find a shortest path to travel from x to y. This shortest path can be represented as $PATH_{xy} := (x, d_1, d_2, \cdots, y)$.

In the **second step** (illustrated in Figure 3) of the construction heuristic, we use the shortest path computed between any two targets to find a tour for the vehicle. To do this, let l_{xy} denote the length of the shortest path $PATH_{xy}$ that starts at x and ends at y. Using l_{xy} as the new cost metric, the Lin-Kernighan-Helsgaun (LKH) heuristic [10] is applied to the graph (T, E_T) with $E_T := \{(x, y) : x, y \in T\}$ to obtain a tour which visits each of the targets exactly once. If there is any edge (x, y) in this tour such that the vehicle *cannot directly* travel from x to y, (x, y) is replaced with all the edges present in the shortest path, $PATH_{xy}$, from x to y. After replacing all the relevant edges with the edges from the shortest paths, one obtains a Hamiltonian tour which visits each of the targets exactly once and some of the intermediate depots for refueling. This tour may still be infeasible because there may be a sequence of vertices that starts at a depot and ends at a depot which may not satisfy the fuel constraints for the vehicle. To correct this, we further augment this tour with more visits to the depots as explained in the next step of the heuristic.



(a) A sample tour after performing LKH with the new cost matrix



(b) The tour with the indirect edges replaced with the corresponding shortest paths

Fig. 3. The second step of the construction heuristic

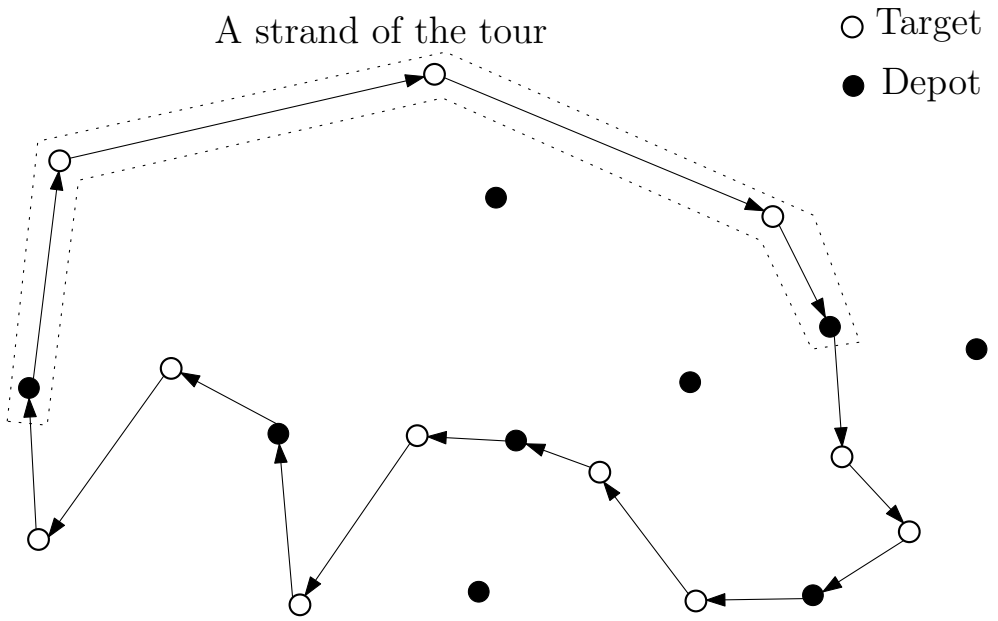


Fig. 4. A strand in a tour

In the **last step** of the construction heuristic (illustrated in Figure 4), the entire tour obtained from the second step is decomposed into a series of strands. A strand is a sequence of consecutive vertices in the tour that starts at a depot, visits a set of targets and ends at a depot. The tour must be infeasible if the total fuel required to travel any one of these strands is greater than the fuel capacity of the vehicle (L). Hence, in this step, all the infeasible strands are identified, and a greedy algorithm is applied to each infeasible strand to transform it to a feasible strand (refer to Fig.5). We present some definitions before we outline the greedy algorithm. A depot, m_x, is referred as a *nearest starting* depot for x if $f_{m_x x} = \min_d f_{dx}$. Similarly, a depot n_x is referred as a *nearest terminal depot* for x if $f_{x n_x} = \min_d f_{xd}$. As in the second step of the construction heuristic, given any two depots $d_1, d_2 \in D$, one can find a path of least cost that starts from d_1 and ends at d_2 while satisfying all the fuel constraints [1].

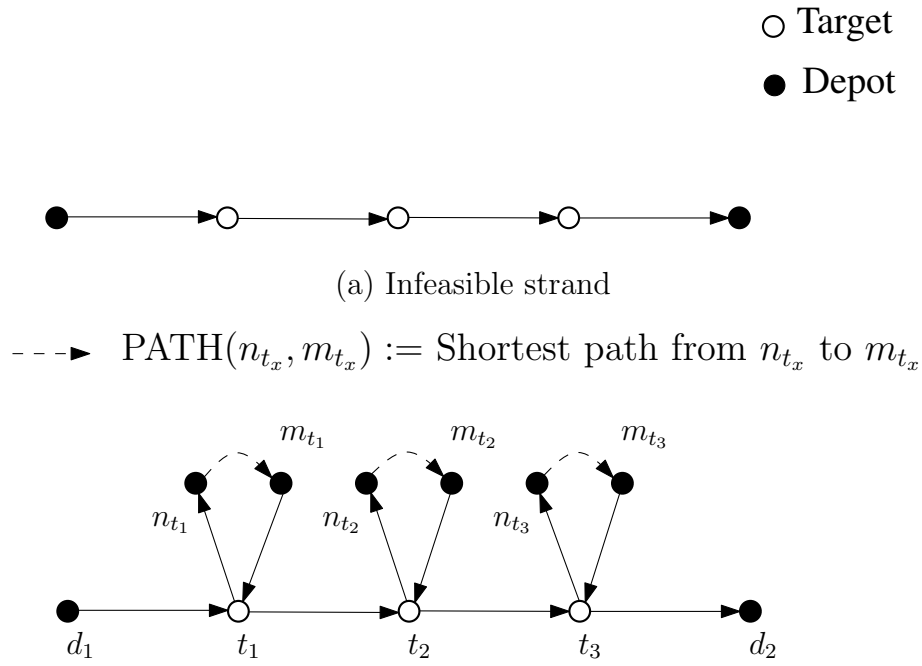
[1]Apply Dijkstra's algorithm on the graph (D, E_d) where $E := \{(i,j) : i,j \in D, f_{ij} \leq L\}$ and cost of traveling from a vertex $i \in D$ to vertex $j \in D$ is c_{ij}.

Let the sequence of all the intermediate depots in this path be denoted by $PATH_{d_1,d_2}$.

The greedy algorithm works as follows: Consider an infeasible strand represented as $(d_1, t_1, \cdots, t_k, d_2)$ where d_1 and d_2 are the two depots of the strand and $t_1, \cdots, t_k$ are the targets. For each target t in this infeasible strand, we add a refueling trip such that

- The vehicle visits a nearest terminal depot n_t after leaving t.
- The vehicle uses the sequence of depots specified in $PATH_{n_t,m_t}$ to travel from n_t to m_t where m_t is a nearest starting depot for t, and finally returns to t after refueling.

After adding all the refueling trips, the modified strand can be denoted as $(d_1, t_1, n_{t_1}, PATH(n_{t_1}, m_{t_1}), t_1, t_2, n_{t_2}, PATH(n_{t_2}, m_{t_2}), t_2, \ldots, PATH(n_{t_k}, m_{t_k}), t_k, d_2)$. This new modified strand must be feasible because the vehicle is allowed to refuel after visiting each of the targets. Now, each of the refueling trips is chosen sequentially in the order they are added and is shortcut if the strand that results after removing the refuel trip still satisfies the fuel constraint.



(a) Infeasible strand

(b) Strand with refuel trips to all targets

(c) Each refueling trip is sequentially shortcut if the resulting strand is feasible. In the above example refuel trips for t_1 and t_3 have been removed as the resulting strand turns out to be feasible

Fig. 5. The greedy procedure to correct infeasibility

After the greedy procedure is applied, one obtains a feasible tour which visits each of the targets exactly once. We now use this feasible tour produced by the construction heuristic as an initial solution for the heuristics discussed in the next section.

E. Improvement heuristics

Now, we develop a combination of a $k-$opt heuristic and a depot exchange heuristic to improve the quality of the tour obtained by the construction heuristic. A $k-$opt heuristic is a local search method which iteratively attempts to improve the quality of a solution until some termination criteria are met. We will first give some basic definitions involved in a $k-$opt heuristic, and then see how it is applicable to the fuel-constrained TSP. A tour S_2 is defined to be in the $k-$exchange neighborhood of the tour S_1 if S_2 can be obtained from S_1 by replacing k edges in S_1 with k new edges. A tour S_2 is said to be obtained from a feasible tour S_1 by an improving $k'-$exchange if S_2 is in the $k'-$exchange neighborhood of S_1, is feasible and has a travel cost lower than S_1. The $k-$opt heuristic starts with a feasible tour and iteratively improves on this tour making successive improving $k'-$exchanges for any $2 \leq k' \leq k$ until no such exchanges can be made. A critical part of developing a $k-$opt heuristic deals with choosing an appropriate $k'-$exchange neighborhood for a tour. In the following sections, we discuss these selections for 2-opt and 3-opt. We also present a depot-exchange heuristic which when combined with the $k-$opt heuristics produce very good solutions for the fuel constrained problem.

1. General framework for improvement heuristics

One way to apply the $k-$opt heuristic is to consider all possible subset of k edges in the tour and try an improving $k-$exchange. Initial implementations showed us that substantial improvements in the quality of the tour were obtained when the $k-$exchanges where performed around the refueling depots in the tour. In view of this observation, we split a given feasible tour into segments. A segment with a span n is defined as a subsequence of $2n+1$ vertices of the tour centered around each depot of

the tour. A segment of span n can be denoted by $(s_1, \ldots, s_n, d, s_{n+1}, \ldots, s_{2n})$, where d is the depot around which the segment is centered. Following the definition of a segment, one can infer that the number of segments in a feasible tour is equal to the number of visits by the UAV to all the depots. When the span is $\lfloor$Tour Length/2$\rfloor$ we get the entire tour as a segment.

For the $k-$opt heuristic, the $k-$exchange neighborhood in each iteration is restricted to one of the segments of the given tour. Therefore, in each iteration of the heuristic, we find all the possible improving k-exchanges with all the deletion and addition of edges restricted to a segment of the tour, and then move to the best possible k-exchange that is feasible. Checking the feasibility of the new tour can be done by keeping track of the fuel remaining in the UAV as it traverses the vertices in the tour.

In the depot exchange heuristic, the depots in the tour are replaced with better refueling depots not present in the tour. In particular, we consider the depots in the order they are visited and try replacing it with other depots. We iteratively do this procedure until we exhaust all the improving depot exchanges. The condition for the depot exchange is explained in the subsequent sections. A flow chart of the overall procedure is given in Figure 6.

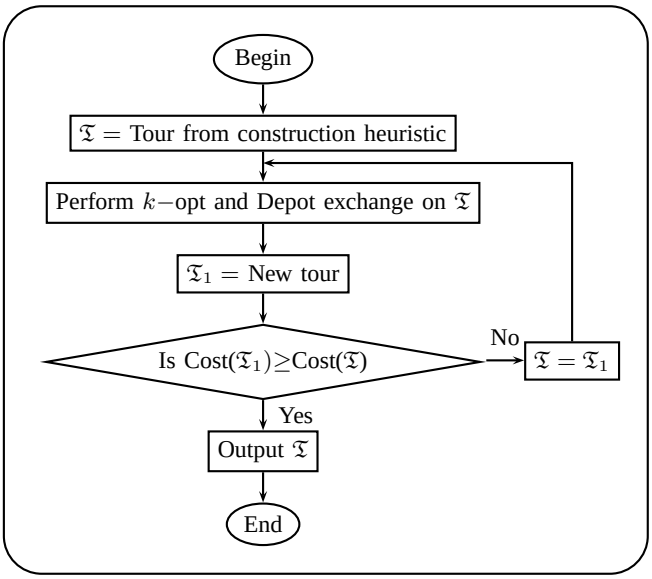


Fig. 6. Flowchart for the improvement heuristics

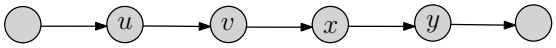
a. $k-$opt

The $k-$opt heuristic requires a feasible tour which in this case is given by the construction heuristic. As explained in the earlier section, the k-opt heuristic starts with a feasible tour and iteratively improves on this tour making successive $k'-$exchanges for any $2 \leq k' \leq k$, until the cost of the tour can no longer be improved. To restrict the neighborhood space, we decompose the tour to segments with span n, and look for improving $k'-$ exchanges within each segment. Given a segment, k' edges are deleted from the segment, and subsequently k' new edges are added to form new segment as shown in Figures 7 and 8. The updated tour is then checked for feasibility, to ensure the UAV never runs out of fuel. The pseudo code for the $k-$opt heuristic is shown in Algorithm 1. An illustration for $2-$opt and $3-$opt is shown in Figures 7 and 8.

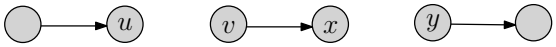
Algorithm 1 : Pseudo code for the k-opt algorithm

Notations: Let $cost(\mathfrak{T})$ denote the sum of all the cost of traveling the edges in the tour $\mathfrak{T}$. Let the segment corresponding to the i^{th} visit to a depot be denoted by $\mathfrak{S}(i, s)$, where s is the *search span* for the segment.

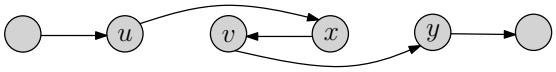
$\mathfrak{T} \leftarrow$ Initial feasible tour
$N \leftarrow$ Number of visits to the depots in $\mathfrak{T}$
$\mathfrak{T}^* \leftarrow \mathfrak{T}$
loop
  for $p = 1, .., N$ **do**
    for each $\mathfrak{S}^* \in k'-$exchange neighborhood of $\mathfrak{S}(p, s)$ and $\forall\ 2 \leq k' \leq k$ **do**
      Find the updated tour $\mathfrak{R}$, with segment $\mathfrak{S}$ replaced with $\mathfrak{S}^*$.
      if $\mathfrak{R}$ is feasible and $cost(\mathfrak{R}) < cost(\mathfrak{T})$ **then**
        $\mathfrak{T} \leftarrow \mathfrak{R}$
      end if
    end for
  end for
  if $cost(\mathfrak{T}) \leq cost(\mathfrak{T}^*)$ **then**
    break
  else
    $\mathfrak{T}^* \leftarrow \mathfrak{T}$
  end if
end loop
Output $\mathfrak{T}^*$ as the solution



(a) Segment considered for 2-opt

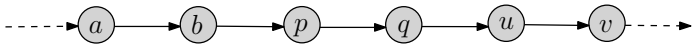


(b) Edges (u, v) and (x, y) are removed



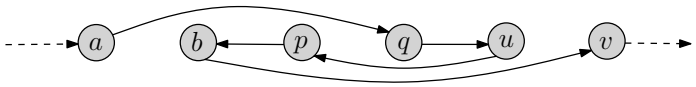
(c) New edges are added to construct a new segment

Fig. 7. Illustration of a 2-opt exchange



(a) Segment considered for 3-opt

(b) Edges $(a, b), (p, q)$ and (u, v) are removed



(c) One possible way of reconnecting edges to construct a new segment

Fig. 8. Illustration of a 3-opt exchange

b. Depot exchange

The depot exchange heuristic works with the depots in a feasible tour. We consider the depots in the order in which they are visited by the UAV and try replacing them with some other depot which can reduce the cost of the tour. Consider a depot d, in the tour. Let t_1 and t_2 be the targets that are visited immediately before and after visiting d. We define a distance function $\mathfrak{D}(d) = c_{t_1 d} + c_{d t_2} \; \forall \; d \in D$. Depot d is then

replaced with d_r where d_r is defined as

$$d_r = \underset{d \in D}{\operatorname{argmin}} \quad \mathfrak{D}(d)$$

The new tour is also checked for feasibility, and is accepted if feasible. The pseudo code for the same is presented in the Algorithm 2.

Algorithm 2 : Pseudo code for the depot exchange

Notations: Let d_i denote the i^{th} depot visited in any given feasible tour. i takes values from $2, 3, ...$ as we do not want to change the starting depot of the tour. D denotes the set of depots.

1: $\mathfrak{T} \leftarrow$ Initial feasible tour
2: $N \leftarrow$ Number of visits to the depots in $\mathfrak{T}$
3: **for** $i = 2 \rightarrow N$ **do**
4: $t_1 \leftarrow$ Target visited immediately before d_i in $\mathfrak{T}$.
5: $t_2 \leftarrow$ Target visited immediately after d_i in $\mathfrak{T}$.
6: $d_r = \underset{d \in D}{\operatorname{argmin}}\ c_{t_1 d} + c_{d t_2}$.
7: Replace d_i with d_r to form the updated tour $\mathfrak{R}$.
8: **if** $\mathfrak{R}$ is feasible and $cost(\mathfrak{R}) < cost(\mathfrak{T})$ **then**
9: $\mathfrak{T} \leftarrow \mathfrak{R}$
10: **end if**
11: **end for**
12: Output $\mathfrak{T}$ as the solution

The next chapter gives a detailed computational study of the all the algorithms explained in this chapter.

CHAPTER III

COMPUTATIONAL STUDY FOR THE SINGLE VEHICLE PROBLEM

A. Comparison of the integer linear programming formulations

The integer linear programming formulations of the FCRP presented in the section B of the previous chapter are solved to optimality using IBM ILOG CPLEX 12 in a Dell Precision T5500 workstation (Intel Xeon E5630 processor @ 2.53GHz, 12GB RAM). The formulations are solved for problem sizes ranging from 15 targets to 40 targets with increments in steps of 5. 50 instances were generated for each problem size and all the targets were chosen randomly from a square area of 5000×5000 units for each instance. In addition, all the instances of the problem had 5 depots chosen at fixed locations in the square area.

The simulations were run two scenarios, one where the vehicle does not have any kinematic constraints and the other scenario where there is a bound on the maximum yaw rate of the vehicle. The vehicles with a bound on the maximum yaw rate is referred to as the Dubins' vehicle [11]. The problem of finding the minimum distance path a vehicle must take between any two targets on a plane subject to the yaw rate constraints had been solved by Dubins [11]. It is assumed that the minimum turn radius for the vehicle is 100 units and the angle of approach for each target was assigned a random value between 0 ann 2π radians. Now, for the Dubins' vehicles, c_{ij} is length the optimal path between i and j calculated by [11]. We also assume that $c_{ij} = f_{ij} \quad \forall i, j \in V$. It is important to note that the formulations and all our heuristics do allow for the travel distances or the fuel costs to be asymmetric, *i.e.*, the distance to travel from location A to location B may be different from the distance required to travel from location B to location A for the UAV. This is in fact the case

for the simulations that are run on the instances with the Dubins' distances (which include the motion constraints). The average time taken by both the formulations for instance sizes ranging from 15 to 40 are shown for the euclidean case (without the motion constraints) in Table I. It emphasizes that the single-commodity formulation is faster than the multi-commodity formulation for the FCRP with fuel constraints. Since the single-commodity formulation out perfoms the multi-commodity for the euclidean distances which is a special case of the Dubins' case, it is natural to assume that the single-commodity is also faster for the Dubins' vehicle. Hence the table II shows the average time taken to compute the optimal solutions for only the single-commodity formulation for all the Dubins' vehicles for instance sizes ranging from 15 to 35.

Table I. Time taken by MILP formulations to solve euclidean instances

No. of Nodes	Single-commodity(sec)	Multi-commodity(sec)
15	0.62	1.31
20	4.50	28.14
25	11.12	268.57
30	239.72	7051.42
35	3020.37	Optimal not reached in 3 hours
40	18032.45[1]	Optimal not reached in 3 hours

[1] The time is averaged over 10 instances. Other instances did not produce optimal solutions after a 4-hour wait

Table II. Time taken by MILP formulations to solve Dubins' instances

No. of Nodes	Single-commodity(sec)
15	0.9
20	11.02
25	70.08
30	469.20
35	9002.5

B. Computational results for various search spans

All the heuristics were coded using Python 2.7.2 and run on a 2GHz Intel Core 2 Duo, 2GB RAM machine. The quality of a solution produced by applying an heuristic on

an instance I is defined as

$$100.\frac{C_I^{heuristic} - C_I^{optimal}}{C_I^{optimal}} \quad \%$$

The flowchart in Figure 6 indicates that the improvement heuristics are executed repeatedly on the best available feasible tour till no further improvement can be made on the cost of the tour. In practice, the algorithm takes at most 2 passes to produce its best possible feasible tour. To ensure consistency in implementation, the algorithm is allowed only two passes of the feasible tour through the improvement heuristics. Tables III and IV gives the variation of the solution quality and computation times with the search span for a segment for instance sizes 25 to 40 (the smaller instances were not conclusive enough to decide on a value for the search span) for euclidean costs and 15 to 35 for dubins costs respectively. We decided on a search span of 4 for the $k-$opt heuristic by taking into consideration both the average solution quality and computation time.

Table III. Search span study for k-opt (euclidean cost matrix)

Span	25 nodes		30 nodes		35 nodes		40 nodes	
	Sol. Quality	Time	Sol. Quality	Time	Sol. Quality	Time	Sol. Quality	Time
1	4.57	0.02	6.65	0.03	4.91	0.04	6.05	0.05
2	3.85	0.17	5.23	0.23	4.34	0.30	5.15	0.33
3	3.38	0.64	4.73	0.90	3.80	1.10	4.79	1.40
4	3.55	1.63	4.32	2.29	3.43	2.90	4.42	3.44
5	3.54	3.55	3.99	4.36	3.16	5.82	4.38	7.07
6	3.06	4.98	3.81	7.58	2.73	9.95	4.18	13.08

Table IV. Search span study for $k-$opt (Dubins' cost matrix)

Span	20 nodes		25 nodes		30 nodes		35 nodes	
	Sol. Quality	Time	Sol. Quality	Time	Sol. Quality	Time	Sol. Quality	Time
1	4.73	0.02	6.97	0.03	6.81	0.05	11.52	0.06
2	4.69	0.16	6.96	0.23	6.71	0.33	10.87	0.44
3	4.46	0.59	6.13	0.76	6.11	1.30	9.57	1.80
4	4.31	1.50	5.62	2.33	5.74	3.29	9.25	4.45
5	4.04	2.80	5.41	4.58	5.44	6.65	8.86	9.13
6	3.84	4.70	5.40	5.32	5.03	6.68	8.38	14.06

C. Computational study for the heuristics

The approximation algorithm for this problem proposed by Khuller et. al [8] was also implemented and run on the instances with the euclidean distances generated for this problem and its solution quality is compared with that of the heuristics suggested in this paper. A comparison of the heuristics proposed in the thesis is also done for the Dubins' instances in table VI. From the tables I and II, it can be observed that the time taken for the computing the optimal solution increases significantly as the number of nodes of the problem increases. In comparison, the running time of the approximation algorithm by Khuller et al. [8] and all the proposed heuristics was less than 5 seconds for each instance of the problem. It is evident from Table V that the heuristics presented in this paper out perform the approximation algorithm in [8]. Table VI re-emphasizes that the algorithm is applicable and can perform reasonably well for more general variants of the problem.

Table V. Solution quality of the heuristics for euclidean instances

No. of Nodes	Khuller et. al [8]		Construction heuristics		Improvement heuristics	
	Average	Maximum	Average	Maximum	Average	Maximum
15	23.38	47.43	19.47	36.53	**2.14**	11.00
20	27.17	67.03	19.65	42.28	**1.97**	16.23
25	30.67	67.80	21.48	38.49	**3.23**	14.07
30	31.32	55.88	22.97	46.83	**4.32**	13.03
35	28.09	49.32	20.55	38.42	**3.47**	13.09
40	31.43	63.67	21.99	34.07	**4.53**	16.36

Table VI. Solution quality of the heuristics for Dubins' instances

No. of Nodes	Construction heuristics		Improvement heuristics	
	Average	Maximum	Average	Maximum
15	6.72	23.14	**4.59**	23.14
20	7.91	40.79	**3.82**	16.83
25	11.04	23.35	**5.34**	23.35
30	13.38	31.64	**4.92**	18.14
35	14.84	39.63	**7.21**	30.59

D. Effectiveness of using solutions from heuristics in CPLEX

The feasible solution produced by the heuristics was used as an initial feasible solution to the single-commodity formulation. The formulation was now solved in CPLEX with a time bound of 10 seconds. Using the initial feasible solution, CPLEX reduces the size of the branch and bound tree by pruning various branches of the search tree. The best feasible solution that could be obtained was sought after to re-emphasize effectiveness of the heuristics. This procedure led to better solutions which can be observed from the Tables VII and VIII. The single-commodity formulation for the FCRP is also solved on all the instances without using the initial feasible solution and allowed to run for 10 seconds. A comparison of the solution qualities for both the cases is made in Tables VII and VIII.

Table VII. Effect of upper bounds on the MILP formulation (euclidean costs)

No. of Nodes	With initial feasible solution		Without initial feasible solution	
	Average	Maximum	Average	Maximum
25	0.40	2.90	3.95	25.88
30	1.07	7.56	13.66	77.01
35	1.80	13.94	14.03	38.44
40	2.13	6.21	21.73	74.11

Table VIII. Effect of upper bounds on the MILP formulation (Dubins' costs)

No. of Nodes	With initial feasible solution		Without initial feasible solution	
	Average	Maximum	Average	Maximum
20	0.09	2.29	0.18	2.94
25	0.26	3.46	0.86	9.64
30	1.39	7.90	2.58	10.30
35	6.22	28.49	19.17	269.52

CHAPTER IV

APPROXIMATION ALGORITHM FOR MULTIPLE VEHICLE PROBLEM

A. Problem statement

The Multiple Vehicle Fuel Constrained Routing Problem (MVFCRP) can be formally stated as follows. There are n vehicles $v_1, v_2, v_3, \ldots, v_n$ with fuel capacities $L^1, L^2, \ldots, L^n$. Without loss of generality, we can assume $L^1 \geq L^2 \geq \cdots \geq L^n$. Let T denote the set of targets that need to be visited and D denote all the depots. Initially, each vehicle is stationed at $s_i \in D$. Define $S := \{s_1, s_2, \cdots, s_n\}$. The MVFCRP is defined on the graph $G = (V, E)$ with $V := T \bigcup D$ and E representing all the edges joining any two vertices in V. Let f_{ij} denote the fuel required by any vehicle to travel from vertex i to vertex j. It is assumed that the fuel costs are symmetric, *i.e.*, $f_{ij} = f_{ji}$ for all $i, j \in V$. It is also assumed that for every target $t \in T$, there exists at least one depot d such that the fuel consumed to travel from t to d is at most $\frac{L^k \alpha}{2}$, for some vehicle v_k, where $\alpha \in (0, 1]$. This assumption is reasonable, because if a target has no depot which can be reached with $\frac{L^k}{2}$ units of fuel left in the vehicle v_k for some k, then the target cannot be visited by any vehicle.

A tour for vehicle v_i is defined by a sequence of vertices $(d_i, v_{i1}, v_{i2}, \cdots, v_{i,k_i}, d_i)$ where $d_i \in D$ is the depot where the vehicle is initially stationed, v_{i1} is the first vertex visited by the vehicle, v_{i2} is the second vertex visited by the vehicle and so on. The objective of the problem is to find a tour for each vehicle so that each target is visited at least once, the vehicles never run out of fuel and the sum of the fuel required by all the vehicles to travel their paths is a minimum.

Remark: We have assumed that for any target $x \in T$ there exists a depot d such

that $f_{td} \leq \frac{L^k \alpha}{2}$ for some vehicle v_k. We denote this depot by $h(x)$ and let D_x represent the fuel required to travel from x to $h(x)$ (or vice versa). Now, we say that a target t is reachable for vehicle v_k if there exists a depot which satisfies the above condition. Essentially, the assumption states that each target is reachable by at least one vehicle.

For this problem, we develop an algorithm with an approximation ratio of $\frac{2(1-\alpha)}{(1+\alpha)}$ in the next section.

B. Approximation algorithm

The approximation algorithm can be described by the following steps:

1. For any two vertices $x, y \in V$ and vehicle v_k, find a path for v_k such that the path satisfies all the refueling constraints for v_k and the sum of the fuel required to travel the edges in the path is a minimum. The algorithm used in this step is exactly the same as the one used in the first step of the construction heuristic for the single vehicle problem. Let this path be denoted by $PATH^k_{xy}$ and let the length of this path be represented by $l^k(x, y)$. As in the single vehicle algorithm, a vehicle can either *directly* or *indirectly* travel from x to y. We will later show that this new cost function satisfies the following monotonicity property: $l^1(x, y) \leq l^2(x, y) \cdots \leq l^n(x, y)$ for any two targets $x, y \in T$.

2. Consider the graph $\overline{G} := (\overline{V}, E_{\overline{V}})$ where $\overline{V} = S \bigcup T$ and $E_{\overline{V}}$ denotes all the edges joining any pair of vertices in $\overline{V}$. Given the graph $\overline{G}$ and the new cost function for each vehicle, we aim to solve the multiple vehicle routing problem of finding a tour for each vehicle such that each target is visited at least once by a vehicle and the sum of the cost of all the edges in the tours is a minimum. As this problem is NP-Hard, we use the following algorithm to find a good feasible

solution to the problem.

- Use the primal dual algorithm in Jung et al. [12] to find a collection of edges such that each target is connected to one of the depots in S. These collection of edges would essentially form a forest where each tree will have exactly one depot. Let the tree corresponding to vehicle v_k be denoted by $TREE_k$. Refer to Figure 9 for an illustration of this step. It was shown in Jung et al. that this primal-dual algorithm produces a collection of edges whose cost is at most equal to the optimal cost of the multiple vehicle routing problem.
- Double the edges in each of the trees to obtain an Eulerian graph for each vehicle. Let the Eulerian graph for vehicle v_k be denoted by $\mathfrak{E}_k$.
- For $k = 1, \cdots, n$ do the following:

  Use $\mathfrak{E}_k$ to find an Eulerian tour for vehicle v_k. If there is any edge (x, y) in this tour such that the vehicle cannot directly travel from x to y, (x, y) is replaced with all the edges present in the shortest path, $PATH^k_{xy}$, from x to y. Let the final Eulerian tour after replacing the *indirect* edges with the edges from the shortest path be denoted by $TOUR_k$.

3. The Eulerian tours may still be infeasible for some vehicles as there may be a sequence of vertices that starts at a depot and ends at a depot where a vehicle runs out of fuel. To correct this, we further augment each of the infeasible Eulerian tours with more visits to the depots (similar to the single vehicle algorithm). The Eulerian tour for each vehicle, is decomposed into a series of strands. A strand is a sequence of consecutive vertices in the tour that starts at a depot, visits a set of targets and ends at a depot. The tour for vehicle v_k

must be infeasible if the total fuel required to traverse any one of these strands in $TOUR_k$ is greater than L^k. Hence, for each infeasible Eulerian tour, all the infeasible strands are identified, and a greedy algorithm is applied to each infeasible strand to transform it to a feasible strand. Refer to Figure 10 for an illustration of this step.

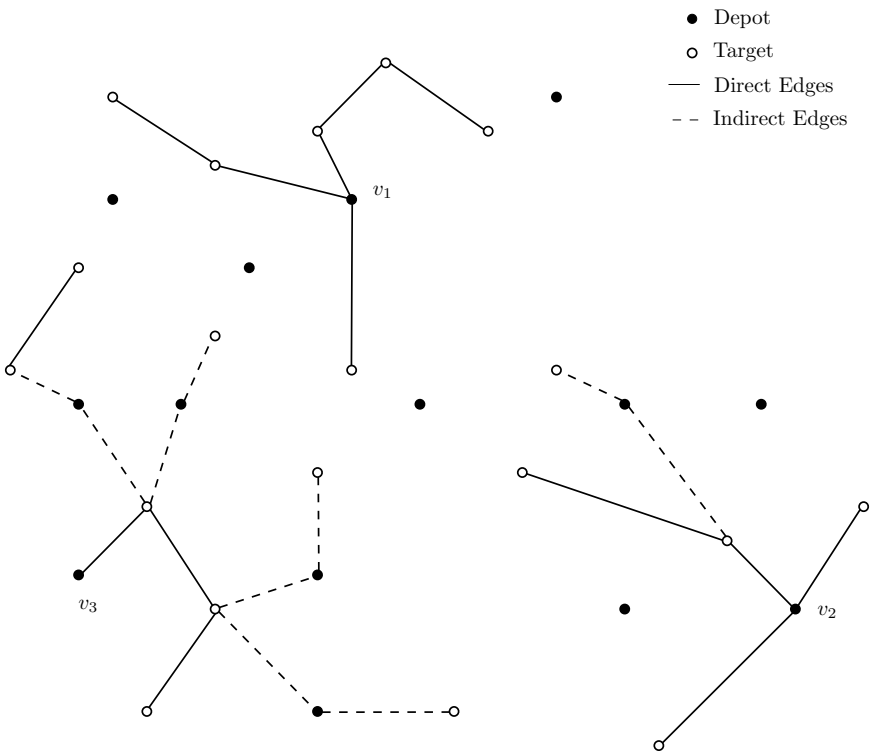


Fig. 9. A forest obtained by the primal dual algorithm for a 3 vehicle problem. The edges of the forst are doubled to get the Eulerian tour corresponding to each vehicle

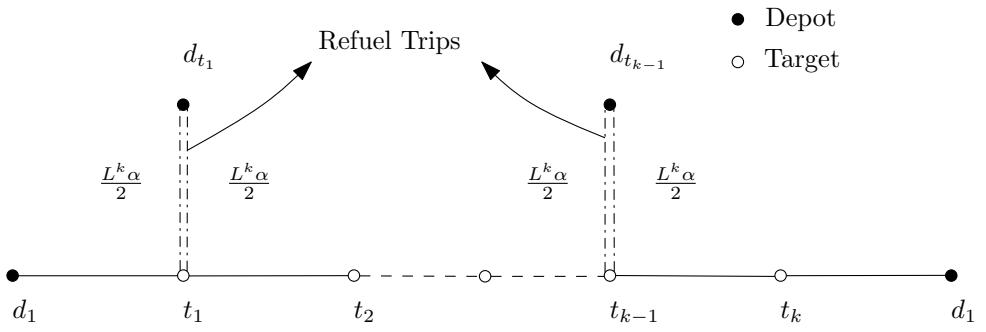


Fig. 10. An infeasible strand of some vehicle v_k, after the addition of a minimal set of refuel trips to make it feasible. Each refuel trip at most consumes $L^k\alpha$ amount of fuel for v_k.

C. Proof of approximation ratio

In this section, we prove that the proposed algorithm has an approximation ratio of $2\left(\frac{1+\alpha}{1-\alpha}\right)$. It is easy to verify that the number of steps required to implement the algorithm is polynomial in the size of the problem. The following lemma proves a claim that was stated in the previous section.

Lemma 1. *The new cost functions satisfy the following property:* $l^1(x,y) \leq l^2(x,y) \leq \cdots \leq l^n(x,y)$ *for any pair of targets* $x, y \in T$.

Proof. Recall that $l^k(x,y)$ is defined as the length of a shortest path that starts at x with at most $L^k - D_x$ units of fuel left in the vehicle and ends at y with at least D_y units of fuel left. Note that $D_x := \min_d f_{xd}$ and $D_y := \min_d f_{yd}$ are independent of the capacities of the vehicles. Therefore, a vehicle with a larger fuel capacity will have more units of fuel to start with at target x. For any two vehicles v_{k_1}, v_{k_2} with $k_1 < k_2$, it follows that the length $(l^{k_1}(x,y))$ of the shortest path for the vehicle with a larger fuel capacity would be at most equal to the length $(l^{k_2}(x,y))$ of the shortest path for a vehicle with a lower fuel capacity. Hence proved. □

The next lemma bounds the cost of the refuel trips. This result is proved by Khuller et al for the single vehicle problem [8].

Lemma 2. *(**Khuller et al. [8]**) Let S_i^k be the length of the i^{th} strand corresponding to the vehicle v_k in $TOUR_k$. Then the total cost of the refuel trips of the targets for v_k is at most $\frac{2\alpha}{1-\alpha}S_i^k$.*

Proof. Let us assume that the number of refuel trips in the i^{th} strand be N_i. Label the targets with refuel trips to the nearest depot be $x_i^{j_1}, x_i^{j_2}, \ldots, x_i^{j_{N_i}}$. Also, let the strand be denoted as $(x_i^{j_0} = d_1, \cdots, x_i^{j_1}, \cdots, x_i^{j_2}, \cdots, x_i^{j_{N_i}}, \cdots, x_i^{j_{N_i+1}} = d_2)$ where d_1 and d_2 are the depots at the ends of the strand. The cost of each refuel trip is at most $L^k\alpha$. Hence, the total cost for traversing all the refuel trips in the strand is at most $N_iL^k\alpha$. Also, the cost incurred for v_k when it travels from $x_i^{j_p}$ to $x_i^{j_{p+2}}$, *i.e.*, $Cost(TOUR_k(x_i^{j_p}, x_i^{j_{p+2}})) \geq (1-\alpha)L^k$. If this condition is not satisfied, then the vehicle v_k can directly go from $x_i^{j_p}$ to $x_i^{j_{p+2}}$ without refueling at the target $x_i^{j_{p+1}}$. Therefore,

$$2S_i^k \geq \sum_{0 \leq p \leq N_i - 1} Cost(TOUR_k(x_i^{j_p}, x_i^{j_{p+2}})) \geq N_i(1-\alpha)L^k \implies N_i \leq \frac{2S_i^k}{(1-\alpha)L^k}$$

Hence, the ratio of the cost of the refuel trips to the cost of the strand corresponding to vehicle v_k is at most $\frac{\alpha L^k N_i}{S_i^k}$ which equals $\frac{2\alpha}{1-\alpha}$. □

Let the cost of the Eulerian tour found at the end of step 4 of the algorithm for vehicle v_k be denoted by $Cost(TOUR_k)$. Also, let $Cost(OPT)$ denote the optimal cost of the MVFCRP. Now, the cost of the feasible solution obtained by the algorithm is upper bounded by the sum of the costs of the Eulerian tours and the refueling trips corresponding to all the vehicles. That is, the cost of the feasible solution is bounded by $\sum_k [Cost(TOUR_k) + \sum_{i=1,..,p_k} \alpha L^k N_{ik}$ where p_k denotes the number of strands in

$TOUR_k$ and N_{ik} represents the i^{th} strand in $TOUR_k$. Now,

$$\begin{aligned}
\sum_k [Cost(TOUR_k) + \sum_i \alpha L^k N_{ik}] &\leq \sum_k [Cost(TOUR_k) + \frac{2\alpha}{1-\alpha} Cost(TOUR_k)] \\
&= \left(\frac{1+\alpha}{1-\alpha}\right) \sum_k Cost(TOUR_k) \\
&\leq 2\left(\frac{1+\alpha}{1-\alpha}\right) \sum_k Cost(TREE_k) \\
&\leq 2\left(\frac{1+\alpha}{1-\alpha}\right) Cost(OPT).
\end{aligned}$$

Theorem 1. *There is an algorithm with an approximation ratio of* $2\left(\frac{1+\alpha}{1-\alpha}\right) C(OPT)$ *for the MVFCRP.*

CHAPTER V

CONCLUSION

Fast and efficient heuristics were developed to solve a new generalization of the single vehicle routing problem with refueling constraints. A mixed-integer, linear programming formulations were proposed to find optimal solutions for the problem. In addition, a construction heuristic and few improvement heuristics were presented to find feasible solutions to the fuel constrained TSP. The computational results show that the heuristics produce feasible solution within 3.27% of the optimal, on an average for symmetric instances and 7.21% of optimal, on an average for Dubins's vehicle instances. Future work can be directed towards developing a branch and bound algorithm tailored to the structure of the fuel constrained TSP and it may aid in significantly reducing the computation time for optimal solutions.

Further, a multiple vehicle version of the problem for symmetric instances was addressed. An approximation algorithm for the same was developed. Future work can include formulating the multiple vehicle problem as a combinatorial problem to solve it to optimality. Fast heuristics similar to the ones developed for the single vehicle problem using neighborhood search methods is also another possible direction for future work.

REFERENCES

[1] E. J. Zajkowski T, Dunagan S, "Small UAS communications mission," in *Eleventh Biennial USDA Forest Service Remote Sensing Applications Conference, Salt Lake City, UT*, 2006.

[2] E. W. Frew and T. X. Brown, "Networking issues for small unmanned aircraft systems," *Journal of Intelligent and Robotic Systems*, vol. 54, pp. 21–37, March 2009.

[3] J. A. Curry, J. Maslanik, G. Holland, and J. Pinto, "Applications of aerosondes in the arctic." *Bulletin of the American Meteorological Society*, vol. 85, no. 12, pp. 1855 – 1861, 2004.

[4] NOAA and partners conduct first successful unmanned aircraft hurricane observation by flying through Ophelia. NOAA News Online (2005) URL http://www.noaanews.noaa.gov/stories2005/s2508.htm.

[5] C. E. Corrigan, G. C. Roberts, M. V. Ramana, D. Kim, and V. Ramanathan, "Capturing vertical profiles of aerosols and black carbon over the indian ocean using autonomous unmanned aerial vehicles," *Atmospheric Chemistry and Physics*, vol. 8, no. 3, pp. 737–747, 2008. [Online]. Available: http://www.atmos-chem-phys.net/8/737/2008/

[6] S. T. with Raven UAV's. United States Army, URL http://www.army.mil/article/5644/soldiers-train-with-raven-uavs/.

[7] M. Fischetti, J. J. S. González, and P. Toth, "Solving the orienteering problem through Branch-and-Cut," *INFORMS Journal on Com-*

puting, vol. 10, no. 2, pp. 133 –148, 1998. [Online]. Available: http://joc.journal.informs.org/content/10/2/133.abstract

[8] S. Khuller, A. Malekian, and J. Mestre, "To fill or not to fill: The gas station problem," in *Algorithms – ESA 2007*, L. Arge, M. Hoffmann, and E. Welzl, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, vol. 4698, pp. 534–545. [Online]. Available: http://www.springerlink.com/content/m7k57g0733487383/

[9] T. Magnanti and L. Wolsey, "Optimal trees," Université catholique de Louvain, Center for Operations Research and Econometrics (CORE), Tech. Rep., Jan. 1994. [Online]. Available: http://ideas.repec.org/p/cor/louvco/1994026.html

[10] K. Helsgaun, "An effective implementation of the Lin-Kernighan traveling salesman heuristic," *European Journal of Operational Research*, vol. 126, no. 1, pp. 106–130, Oct. 2000. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0377221799002842

[11] L. E. Dubins, "On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents," *American Journal of Mathematics*, vol. 79, no. 3, pp. 497–516, July 1957, ArticleType: research-article / Full publication date: Jul., 1957 / Copyright © 1957 The Johns Hopkins University Press. [Online]. Available: http://www.jstor.org/stable/2372560

[12] J. Bae and S. Rathinam, "A primal dual algorithm for a multiple depot heterogeneous traveling salesman problem." (accepted) Control and Decision Conference, 2012.