OPTIMAL ROUTING OF UNMANNED VEHICLES IN PERSISTENT MONITORING MISSIONS

A Dissertation

by

SAI KRISHNA KANTH HARI

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

DOCTOR OF PHILOSOPHY

December 2019

Major Subject: Mechanical Engineering

ABSTRACT

Missions such as forest fire monitoring, military surveillance and infrastructure monitoring are referred to as persistent monitoring missions. These missions rely heavily on continual data collection from various locations, referred to as targets. In this dissertation, we consider a framework in which data is collected from the targets with the aid of unmanned aerial vehicles (UAVs). A UAV makes a physical visit to the targets for data collection, and immediately transmits the collected data to a base station for further analysis. Typically, the duration of these monitoring missions is long, and the monitoring vehicles are required to stay in flight for extended periods of time. Therefore, the batteries powering the UAVs must be recharged regularly at a recharging station/depot. From utilitarian and economic points of view, an efficient execution of these missions calls for two requisites: 1) minimizing the time delay between successive data collections at targets; 2) maximizing the total charge/energy drawn from batteries.

The maximum time delay between successive data collections at any target is characterized by a function referred to as the walk revisit time, or simply the revisit time. Given a set of targets and a UAV tasked with monitoring the targets, the charge capacity of the battery powering the UAV can be surrogated by the number of visits the UAV can make to the targets without requiring a recharge. To minimize the wastage of energy resources, a charge penalty is imposed on the visits that are unutilized before each recharge. The aim of this work is to find optimal routes for the UAV(s) to visit the targets such that the sum of the revisit time and the charge penalty is minimized.

The optimal route planning problem is determined by a number of factors such as the number of UAVs used for monitoring, the aerial platform on which the monitoring UAVs are built, the location of their depots, relative importance of the targets being monitored, etc. In this dissertation, we focus on equally weighted targets and address four different variants of the problem, all of which are computationally extremely challenging. The variants considered are the following: 1) single UAV with no motion constraints and the depot located at one of the targets; 2) single UAV with curvature constraints on its path and the depot located at one of the targets; 3) single UAV with no motion constraints and its depot stationed at a location different from that of the targets; 4) multiple UAVs with no motion constraints with their depots located at the targets.

This dissertation builds on the results of Variant 1; specifically, the characterization of the optimal solutions proved in this dissertation is the main contribution of this dissertation; it lends itself to a new formulation of the same problem that results in significant computational savings. The structural characterization also holds for Variant 2. Inspired by this result, conjectures are provided for the structure of optimal solution for variant 3 and is backed up by extensive numerical simulations. Variant 3 can also be perceived as a special case of targets with different weights/priorities, and therefore, the results developed in this dissertation can potentially be extended to solve a few special cases of the general problem involving arbitrarily weighted targets.

DEDICATION

To my father Dr. Hari V.S. Satyanarayana and my mother Hari Manjula.

ACKNOWLEDGMENTS

Firstly, I would like to thank my chair Dr. Swaroop Darbha from the bottom of my heart for giving me a wonderful opportunity of working with him. He always puts his students' interests first and has always allowed me to maintain a good work-life balance. Throughout my graduate studies he has given me a lot of freedom to work on a wide variety of problems ranging from localization of underwater vehicles to optimization of power grids. I owe him a lot for his constant efforts in shaping me as a competent researcher (it is of course a work in progress always) and supporting me financially in the form of research/teaching assistantships.

Another important person in my journey at A & M as a graduate student is my co-chair, Dr. Sivakumar Rathinam. Similar to my advisor, he is very friendly and patient with students. He has always kept me motivated and worked with me closely on a number of research problems. In fact, there were times where I've worked more with him than my advisor. Apart from his invaluable guidance in research, I have also benefited from his valuable suggestions on other fronts such as surviving in the research field, making the right career choices, etc.

In addition to the role they played in building my research career, their contributions to the work presented in this dissertation are significant. The problem considered in this dissertation was first introduced to us by our collaborators: Dr. Krishna Kalayanm, Dr. David Casbeer and Dr. Satyanarayana Manyam. Right from November 2017, when I've started working on the problem, their support and constructive feedback helped us making swift and significant progress in this work. I would specially like to thank Air Force Research Laboratory for supporting this work financially.

Apart from my collaborators, I would also like to thank the rest of my committee members: Dr. Prabhakar Pagilla and Dr. Lewis Ntaimo. Even though I haven't worked directly with them on the current problem, I got a chance to learn a lot from them in the form of courses. I've taken my controls and optimization courses with Dr. Pagilla and Dr. Ntaimo respectively. Their courses have a good mix of theory and implementation parts, which help in preparing students for the current research world. I've always benefited from discussions with them in regards to both courses and the current problem. Their knack of quickly catching subtleties of the problem helped us strengthen our work.

This section wouldn't be complete without mentioning my stint at Los Alamos National Laboratory. After my 7 month co-op at LANL, I could see a significant improvement in my research skills, especially in software implementation and analysis of results. I accredit this to the guidance provided my mentors at the lab: Drs. Russell Bent, Harsha Nagarajan, Kaarthik Sundar and Scott Backhaus. I would also like to thank Kaarthik for familiarizing me the tools to implement optimization algorithms during the initial days of my graduate studies.

Considering how essential it is for a PhD student in engineering to have a reasonable background in mathematics, I would also like to thank Drs. David Larson and Sivakumar Natarajan for teaching me the foundations of mathematical analysis. Numerous discussions I had with them made me better equipped to tackle mathematical proofs.

I would like to thank Dr. Reza Langari for letting me be a part of his research group, mentoring me, and serving as my committee member during my master's program. I would also like to thank Dr. Krishna Narayanan for offering me an office space in his lab for an easy interaction with his students from whom I learnt the basics of communication theory.

Finally, no matter how cliche it might sound, this work would not have been possible without the support of my parents, my sister Sarada Bhanu, all my cousins in the U.S. and their spouses who did not let me miss my family, and my friends who are now like family. I would especially like to thank my maternal cousins Sambhuni Rakesh and Sambhuni Ravi Prakash for the care and support they've provided right from the time I've landed in the U.S. (and of course from my childhood), and my friends Ravi Rohith and Amit Jamadagni for always being there for me.

CONTRIBUTORS AND FUNDING SOURCES

Contributors

This work was supported by a dissertation committee consisting of Professors Swaroop Darbha (chair), Sivakumar Rathinam (co-chair) and Prabhakar Pagilla of the Department of Mechanical Engineering, and Professor Lewis Ntaimo of the Department of Industrial Engineering. External collaborators for the work are Drs. David Casbeer, Satyanarayana Manyam and Krishna Kalyanam.

A rudimentary version of the problem addressed in this disseration was first introduced to the student by the external collaborators, and the subsequent work was conducted by the student under the supervision of the committee members and the external collaborators.

Funding Sources

The work presented in the dissertation was financially supported by the Air Force Research Laboratory (AFRL) and the Mechanical Engineering Department of Texas A & M University.

NOMENCLATURE

UAV	Unmanned Aerial Vehicle
TSP	Traveling Salesman Problem
LP	Linear Program
ILP	Integer Linear Program
MBLP	Mixed Binary Linear Program
PMP	Persistent Monitoring Problem
DPMP	Dubins Persistent Monitoring Problem
GPMP	Generalized Persistent Monitoring Problem

TABLE OF CONTENTS

LIST OF FIGURES

1. INTRODUCTION AND LITERATURE REVIEW

Persistent monitoring missions such as infrastructure monitoring, military surveillance and environmental monitoring [1, 2, 3] predominantly rely on continual data-collection for detecting emergent events such as the formation of cracks in bridges, the onset of forest fires, etc. Based on the mission under consideration and the environment being monitored, certain locations are chosen for data collection; these locations are referred to as *targets*. The type of data collected from the targets could range from visual images to infrared signals.

Ideally, for an instant detection of the events, sensors necessary for data collection could be deployed at the targets. However, the use of stationary sensors is not recommended due to several reasons: 1) the cost required to install sensors at the target locations is usually high, which further increases with the increase in the number of targets; 2) the targets are often inaccessible by humans either due to their remote location, or the safety hazards involved in the mission (for example, military surveillance); 3) even if the targets are accessible, the operational costs could be high due to the need of a human visit to the targets to replace the batteries of sensors; 4) there could be issues related to data transmission due to dense vegetation cover for example. So, there is a need for an effective monitoring technique that is easy, safe, and inexpensive.

For this reason, we consider a framework in which data is collected with the aid of unmanned aerial vehicles (UAVs) that are powered by batteries and equipped with the necessary data collection sensors. These UAVs make physical visits to the targets to collect data and immediately transmit the collected data to a base station. At the base station, the data is processed and presented to a human operator for examination [4], who then makes critical decisions required for the successful execution of the mission. Due to the persistent nature of the mission, the UAVs are required stay in flight for an extended duration and their batteries must be regularly recharged/replaced at a depot [3].

From utilitarian and economic points of view, an efficient execution of these missions calls for two requisites: 1) minimizing the time delay between successive data collections at targets;

2) maximizing the total charge/energy drawn from batteries. *The aim of this work is to plan an optimal sequence of targets to be visited by the UAV* such that the above stipulations are met. The UAVs follow these routes (sequences of visits) repeatedly to persistently monitor the targets. The criteria determining optimal routes will be elaborated further in the subsequent sections.

1.1 Revisit Time

A timely detection of emergent events is key for the success or failure of persistent monitoring missions; for example, an early detection and subsequent regulation of a forest fire could potentially save thousands of lives, millions of acres of land, and billions of dollars. To facilitate such a timely detection, it is of utmost importance to minimize the time between successive visits to the targets. To enable this, we define the maximum time elapsed between successive visits to a target as the *revisit time of the target* or simply the *target revisit time (t.r.t)*, and the maximum of this value over all the targets as the *walk revisit time*, or simply the *revisit time*. One of the objectives of this work is to find routes that yield the minimum revisit time.

In certain applications, some targets may have a higher priority and demand a higher frequency of monitoring. In such cases, the targets are assigned weights based on the information expected to be gained from monitoring the targets or the chances of detecting events at the targets. Then, the target revisit times are multiplied by the weights of their corresponding targets, and the maximum weighted t.r.t over all the targets is referred to as the *weighted revisit time*. The objective in the presence of targets with different weights is to minimize the weighted revisit time.

1.2 Battery Charge Utilization

As the batteries powering the UAVs have finite charge capacities, they must be recharged / replaced regularly at a depot. Therefore, optimal sequences of visits to targets depend directly on the charge capacity of the batteries. Due to the persistent nature of the considered missions, the number of such rechargings[*] required is high. For this reason, it is important to maximize

[*]Here, we use the term recharging to indicate one of the following: replacing the discharged battery with a fully charged one, if the battery is either rechargeable or of the use-and-throw type; recharging the battery at the depot if it is of the rechargeable type.

the charge utilized before every recharging, while ensuring that the lifespan of batteries is not compromised due to restrained or excessive discharging. If the batteries are of use-and-throw type, they must be discharged to the maximum extent before being replaced. If the batteries are of rechargeable type, there is an optimum range of charge-discharge percentage within which they must be operated. Accordingly, to avoid excessive discharging and untimely recharging, we strictly do not allow the batteries to be discharged beyond a certain percentage, and we impose a *penalty* for recharging the batteries earlier than they must be.

1.2.1 Proxy for Battery Charge Capacity

To enforce this penalty, it is important to model the charge utilization of batteries; however, this is not a simple task. While the actual amount of power utilized by a UAV depends on its distance and/or duration of its travel, a number of external factors such as wind and ambient temperature also influence the exact amount of power utilized. Therefore, the task of accurately tracking the battery charge consumption based on the route traversed by a UAV is non-trivial. Even if a comprehensive model considering the above factors is available, it renders the subsequent routing problem intractable. For these reasons, we adopt a simple, yet practical approach to model the battery charge consumption similar to those used in [5, 6, 7]. Suppose a UAV is assigned a set of n targets for monitoring. Then, in the adopted model, the charge capacity of a battery powering the UAV is surrogated by the *number of visits* the vehicle can make to the targets before the battery must be either recharged or replaced. Note that here a visit is defined as the act from traveling between two *different* targets (i.e., each visit has different ‘to’ and ‘from’ targets). If a UAV travels from target i to target j, $j \neq i$, then it is said to have made a visit to j from i. For convenience, the reference to the ‘from’ target is dropped, and the UAV is said to have made a visit to j.

We assume that the batteries have enough fuel capacity to allow the UAV to visit each assigned target at least once before recharging. Let’s say that a UAV can make V visits to the assigned targets before requiring a recharge. Then the assumption is that $V \geq n$. Based on the knowledge of target locations, the value V can be easily estimated. The estimation procedure and the practicality of this approach will be discussed next.

1.2.2 Estimation of the Number of Visits Available for a UAV

To estimate the number of visits V, one can first begin with a rough estimate and compute a corresponding optimal route to the assigned set of targets; the procedure for computing optimal routes will be discussed later in the dissertation. Based on the time/distance required by the UAV to traverse this route, the actual amount of battery charge utilized can be estimated. Depending on whether this estimate exceeds or falls short of the battery charge capacity, one can update the estimated value of V and re-solve the problem. This process can be repeated iteratively till the estimate is either satisfactory or cannot be further updated. Because the number V (corresponding to a given set of targets) obtained by this procedure holds information about when the battery gets discharged, it is a practical/reasonable proxy for battery charge capacity, rather than one whose sole purpose is to simplify the route planning problem.

1.2.3 Charge Penalty on Unutilized Visits & Constraints on UAV Routes

With the number of visits established as the proxy for the battery charge capacity, excessive discharging is strictly disallowed, whereas wastage of resource (charge or battery's lifespan) is discouraged by penalizing the number of visits unutilized before recharging the battery at the depot.

Let us suppose that the UAV is actually recharged at the end of every k ($n \leq k \leq V$) visits. Note that k is different from V; the former is the number of visits after which the *UAV is actually recharged*, whereas the latter indicates the maximum number of visits the *UAV is allowed* to make to the targets by a fully charged battery. Then, the UAV starts the monitoring process by departing from the depot with a fully charged battery. It then makes visits to different targets till the completion of $k-1$ visits. The k^{th} visit of the UAV is made back to the depot for recharging. It is assumed that the UAV visits all the assigned targets in this sequence of k visits; it is therefore tacitly assumed that $k \geq n$. This sequence of k visits is referred to a walk with k visits, and it is repeated a number of times till the completion of the mission to ensure persistent monitoring

Because $V-k$ visits are unutilized after every recharge, and it leads to wastage of resources, a charge penalty is imposed on these visits in the objective function. The charge penalty corresponding to a walk with k visits is given by $m\mu(V - k)$, where m is the number of times the walk is repeated in the mission, and μ indicates the relative cost of the penalty in the objective function.

1.3 Problem of Interest and its Determinants

Given a set of targets to be monitored, a fleet of UAVs to monitor the targets and the fuel capacities of the UAVs, the problem of interest is to find walks for the UAVs such that the sum of the weighted revisit time (the maximum weighted t.r.t over all the targets) and the total charge penalty (sum of the charge penalties over all the UAVs) is minimized. That is, the problem involves finding: an assignment of the targets to the UAVs, the number of visits after which each UAV must be recharged, and sequences in which the assigned targets must be visited by the UAVs to minimize the sum of the revisit time and the charge penalty.

Optimal solutions to this problem are determined by a number of factors such as the number of UAVs used for monitoring, the type of these UAVs, locations of recharging stations (depots) of the UAVs, the relative importance of the targets, etc.

1.3.1 Relative Importance of Targets

As mentioned in Section 1.1, targets could possess different weights/importance, thereby demanding different frequencies of monitoring. Nonetheless, even for the special case where all the targets have equal weights, the problem of interest is computationally very challenging and has a number of rich variants. Throughout this dissertation, we focus on the equally-weighted-targets case and its different variants. The theme of the dissertation is the development of efficient techniques to expeditiously solve these variants. The results developed in this work could potentially be used to systematically include targets with unequal weights and ultimately develop efficient procedures to solve the general problem with arbitrarily weighted targets. We support this claim by considering a variant of the equally-weighted-targets case that can be perceived as a special case of the problem with *unequally* weighted targets.

Note: From here on, whenever the targets under consideration have equal weights, we use the term *revisit time* instead of the term *weighted revisit time*, as they are interchangeable.

1.3.2 Number of UAVs

The minimum revisit time achievable, given a set of targets and the weights of the targets, depends firstly on the number of UAVs used for monitoring. Based on this number, we classify the optimal route planning problem into two classes: 1) persistent monitoring with a single UAV; 2) persistent monitoring using multiple UAVs.

1.3.2.1 Single UAV

When a single UAV is used for monitoring, finding an optimal target assignment is trivial, as the only solution is to assign all the targets that need to be monitored to the UAV. Let's say that the charge capacity of the UAV corresponding to this set of targets is V visits. Then, the problem of interest reduces to determining the optimal number of visits after which the UAV must be recharged, say k visits, where $n \leq k \leq V$, and its corresponding optimal sequence of visits to the targets. It will later be seen that this problem is NP-Hard, and the ability to solve it rests on the solvability of a sub-problem in which k is given and one needs to compute an optimal sequence of k visits. An efficient approach to solve the sub-problem ensures that the master problem of finding the combination of an optimal k and its corresponding optimal sequence of visits can be solved easily. Throughout this dissertation, we focus on developing methods to efficiently solve sub-problems of such kind for different variants of the master problem.

1.3.2.2 Multiple UAVs

When multiple UAVs are used for monitoring, a number of questions need to be addressed: do all the UAVs start from the same depot? are multiple UAVs allowed to visit the same target? if so, are the UAVs allowed to visit the same target either simultaneously or with a small time gap between their individual visits? Even for the simpler case where each UAV starts from a different depot and each target is allowed to be visited by only one UAV, the problem is computationally intractable. This is mainly because of the difficulty involved in finding an optimal assignment of targets to the UAVs. Not only is the number of possible assignments combinatorially large, but also there is coupling involved between different solution components; a target assignment determines the number of visits allowed for a UAV to its assigned set of targets, which in turn determines an optimal sequence of visits for the UAV. Though the availability of a target assignment reduces the problem to individually solving a set of single UAV routing problems, finding an optimal assignment is non-trivial. In this dissertation, we present an efficient formulation to obtain *feasible* solutions to the problem based on the results developed for the single UAV case.

1.3.3 Types of UAVs (Curvature Constraints)

The next factor influencing optimal routes to targets is the aerial platform on which the UAV(s) used for monitoring are built. Different platforms impose different constraints on the UAVs, which impact their minimum travel times between the targets. The type of UAV(s) chosen for monitoring depends on the environment in which the vehicles are required to operate and the nature of the monitoring mission.

Based on their aerial platforms, UAVs can broadly be classified into four types: 1) single-rotor UAVs; 2) multi-rotor UAVs; 3) fixed-wing UAVs; 4) hybrid vertical take-off and landing (VTOL) UAVs. Out of these, multiple-rotor UAVs and fixed-wing UAVs are most commonly used for persistent surveillance missions and are considered here. Single-rotor UAVs are less stable in air (hence, unsuitable for tasks such as high quality photography) and difficult to control compared to multiple rotor UAVs. They are also considered unsafe for certain missions due to their long blades. Hybrid VTOL UAVs are still under development, and there are a very few of them available in market currently.

1.3.3.1 Multi-Rotor UAVs

Multi-rotor UAVs are lifted and propelled by multiple rotors. By changing the relative speeds of these rotors, multi-rotor UAVs can be easily maneuvered and are capable or moving in any direction. Their easy maneuverability allows the usage of simple kinematic models for route planning purposes (especially for planning sequences of visits to targets). Here, we consider one such model in which the assumptions are that the UAV can make sharp turns and can travel at a constant speed for a majority of its flight. We also assume that the travel times of these UAVs between the targets satisfy the triangle inequality, i.e., the time taken to travel from target p to target r via target q is no lesser than the time take for the UAV to travel directly from target p to r.

Multi-rotor UAVs are relatively less expensive compared to their fixed-wing counterparts and are well suited for persistent monitoring over small areas. However, their flight time is relatively low compared to fixed-wing UAVs and must be recharged more often. Moreover, they are more vulnerable (in terms of stability) to winds, making them less suitable for operating in environments with strong winds.

1.3.3.2 Fixed-Wing UAVs

Fixed-wing UAVs are similar to traditional aircrafts in that they have two wings that help in lifting the UAV off the ground. The energy required to keep these vehicles in flight is low. Therefore, they can fly for longer durations without being recharged and are suitable for large scale monitoring missions. Besides, they are more resilient (in terms of stability) to strong winds compared to multi-UAV rotors, making them well-equipped for working in harsh environments.

Generally, for route planning problems, fixed-wing UAVs are modeled using a kinematic model referred to as the Dubins model. Dubins model is known to produce near-feasible paths for fixed-wing UAVs. In this model, the following assumptions are made on UAVs: 1) UAVs can only travel forward; 2) they travel at a constant speed; 3) their minimum turn radii are constrained. These constraints require the UAV to maintain the same heading angle while entering and leaving a target. So, designing a feasible route for these UAVs involves determining the angles at which the UAVs make visits to the targets, in addition to finding sequences in which these visits are made. This makes the optimal-route-planning problem computationally difficult, as the number of allowable heading angles at each target is infinite, and the minimum travel times between a given pair of targets depends on the angles of arrival and departure of the UAV at these targets.

In this work, we present techniques to obtain *near-optimal solutions* and tight lower bounds to the problem, by allowing only a finite number of apriori chosen heading angles at the targets. With the increase in the number heading angles allowed at each target, the upper and lower bounds get closer.

1.3.4 Depot Location

Frequent recharging needs of UAVs make the location of their depots an important factor in determining optimal sequences of their visits to the targets. We categorize the depot locations into two classes: 1) depot is placed at one of the targets; 2) depot is at a location different from that of the targets.

1.3.4.1 Depot at a Target

The rationale behind setting a depot at one of the targets is to save valuable monitoring time by eliminating the need for traveling to a different location just for the sake of refueling. This case is suitable for civilian missions, where at least one of the targets is accessible for the installation of the depot.

Suppose the depot is setup at target d, a target that is assigned to the UAV. Let the UAV be recharged after every k visits. Then, the monitoring process is carried out as the following: The UAV starts from the depot with a fully charged battery and begins the data collection process by collecting data from target d. Then, it makes data collection visits to different targets such that data is collected from every target at least once by the time it completes $k-1$ visits. The k^{th} visit is made back to the depot for recharging. Note that the UAV is allowed to make re-visits to target d for data collection, but it is not recharged at the depot till the completion of k visits. As soon as the UAV is recharged, it collects data from target d and repeats the walk of k visits till the completion of the mission.

For the problem variants considered in this work where depot(s) is (are) placed at a target (targets), *the time required for recharging the UAV is assumed to be negligible compared to the travel times between the targets.*

1.3.4.2 Depot not at a Target

In certain non-civilian missions such as military surveillance, it is not possible to set up a depot at the targets either due to safety concerns or inaccessibility of the targets. In such cases, one is forced to install depots (the depot) at locations (a location) different from the targets. Then, assuming that the UAV is recharged after every k visits, the route taken by the UAV is of the following form: the UAV starts from the depot with a fully charged battery. Then, it makes data collection visits to different targets such that data is collected from every target by the end of $k-1$ visits. The k^{th} visit of the UAV is made back to the depot for recharging. Here, note that the depot is only visited after every k visits. This sequence of k visits, which is referred to as a walk with k visits, is repeated a desired number of times till the completion of the mission. In this case, we consider (do not neglect) the time required for recharging the vehicle.

1.3.5 Problem Variants Addressed

Taking the above discussed determinants into consideration, we address 4 different problem variants in this dissertation. The variants are delineated by the following combination of the determinants:

1. Single UAV, no curvature constraints, depot at a target, equally weighted targets.
2. Single UAV, with curvature constraints, depot at a target, equally weighted targets.
3. Single UAV, no curvature constraints, depot not at a target, equally weighted targets. This variant, can also be perceived as a special case of the weighted-targets version problem, with one of the targets (which is the depot) having a very low priority.
4. Multiple UAVs with depots at targets, no curvature constraints, equally weighted targets.

All the above variants consider targets with equal weights. We also make a common assumption that the travel times between the targets satisfy the triangle inequality, i.e., the time taken to travel from target i to target k is no more than that taken to travel from target i to some other target j and then from target j to target k.

1.3.5.1 Single UAV without Curvature Constraints and Depot at a Target

In this variant, a set of n equally weighted targets are monitored by a UAV that has no curvature constraints. The recharging depot of the UAV is setup at one of the targets. Given that the fuel capacity of the UAV is V visits to these targets, the problem of interest is to find an optimal number of visits k, $k \geq n$, after which the UAV is recharged, and a corresponding optimal walk with k visits. As mentioned in 1.3.2.1, this problem is computationally challenging (NP-Hard), and the ability to solve this problem is dependent on the solvability of a sub-problem, where k is given. For a given k, the charge penalty is fixed, and the problem reduces to finding a walk with k visits that provides the least revisit time. Such a walk is referred to as an optimal walk with k visits, denoted by $\mathcal{W}^*(k)$, and the least revisit time is referred to as the optimal revisit time for k visits, denoted by $\mathcal{R}^*(k)$. This sub-problem is referred to as the *Persistent Monitoring Problem (PMP)*. Developing efficient techniques for solving the PMP will be the focus of Chapter 2. With an efficient method available to solve the PMP, the master problem of determining the combination of an optimal k and its corresponding optimal walk with k visits can be solved easily as will be shown in Section 2.9. The results developed in Chapter 2 are pivotal, and will be used throughout the dissertation for expeditiously solving other variants of the PMP.

1.3.5.2 Single UAV with Curvature Constraints and Depot at a Target

This variants considers the case in which the monitoring vehicles have curvature constraints. We primarily consider the UAV to satisfy the Dubins model; however, the results developed in this variant also hold true for other models such as Reeds-Shepp model. Similar to the above variant, here, n equally weighted targets are given, and the depot is assumed to be installed at one of the targets. This variant is addressed in Chapter 3, where the main focus is to efficiently solve the sub-problem in which the UAV is given to be recharged after every $k(\geq n)$ visits, and the objective is to find a walk that provides the least revisit time. This sub-problem is referred to as the *Dubins Persistent Monitoring Problem (DPMP)*.

1.3.5.3 Single UAV without Curvature Constraints and Depot Not at a Target

While this variant also involves persistent monitoring of n equally weighted targets using a single UAV, the UAV's depot is not restricted to be placed at one of the targets. In the analysis of this variant, we only consider the case where the UAV has no curvature constraints, because the extension to the Dubins model is similar to that considered in Chapter 3. As opposed to the above variants, here we also consider the time required to recharge the UAV at the depot.

Given that the UAV is recharged after every k visits, $k \geq n + 1$ visits, the objective here is to find a walk with the minimum revisit time. It is to be noted that a feasible walk allows the UAV to visit the depot only during its k^{th} visit. It is also to be noted that the objective function is the maximum of the t.r.t of only the targets and not the depot; however, the times taken to travel to and from the depot, and the time taken to recharge the UAV at the depot influence the objective function. In this variant, an optimal walk with k visits is denoted by $\mathcal{W}D^*(k)$, and the optimal revisit time is denoted by $\mathcal{R}D^*(k)$.

As the depot is visited exactly once in a walk, it can also be treated as a target with a very low priority at which the depot is located. Then, the problem can be treated as a special case of the weighted-targets case, and the objective is to minimize the weighted revisit time of the walk.

1.3.5.4 Multiple UAVs without Curvature Constraints and Depots at Targets

This variant considers the chase in which n equally weighted targets are monitored by m UAVs, where $m \leq n$. The monitoring vehicles are assumed to have no curvature constraints and have distinct depots, each of which is located at one of the targets. Due to the modeling and computational difficulties involved in solving the general route planning problem, here we consider a special case of the problem in which each target is visited by exactly one UAV. Given a restriction on the combined fuel capacities of the UAVs in terms of the total number of visits they can make, the objective here is to find walks for multiple UAVs such that the revisit time is minimized. In this work, we develop a formulation to obtain *feasible* solutions to the problem.

1.4 Computational Difficulty

The optimal routing problem considered here falls under the class of challenging combinatorial optimization problems such as the Traveling Salesman Problem (TSP). The problem considered here is computationally at least as difficult to solve as the famous TSP, which is NP-Hard. We establish this difficulty by considering the simplest of the aforementioned problem variants, i.e., the single-UAV variant with no curvature constraints; the argument holds irrespective of the depot location.

In this variant, the problem involves finding both an optimal number of visits after which the UAV must be recharged, say k visits, and its corresponding optimal sequence of visits. As mentioned above, the ability to solve this problem is dependent on that of its sub-problem PMP, in which k is given and one needs to determine an optimal walk with k visits. Upon examination, it can be seen that for the special case when $k = n$, each target is visited exactly once in the walk, and the UAV returns to the depot from which it started the walk. Moreover, it turns out that the revisit time of a walk in this case is equal to its travel time, i.e., the time taken by the UAV to traverse through the sequence specified by the walk. Therefore, when $k = n$, PMP is equivalent to the TSP, making the problem under consideration NP-Hard.

Besides, for a general k, modeling the revisit time is non-trivial; it typically involves nonlinear constraints, thereby adding to the complexity of the problem. We present a preliminary formulation for solving the PMP for a general k in Section 2.2. Numerical simulations performed using this formulation suggest that it is increasingly difficult to solve the problem in its present form with the increase in both the number of targets and the number of visits made by a UAV before recharging. Figure 1.1 shows this increasing trend of the average computation time with the number of visits in the walk for 8-target instances; the average was taken over 50 instances. For instances with higher number of targets, the problem was unsolvable in a practical amount of time; the computation time for a 20-target instance with 21 visits was 165,273 seconds (on a Mac Pro with 8-Core Intel Xeon E5 processor @3 GHz and 32 GB RAM), which is approximately 2 days.

Other variants of the problem are at least as difficult as the PMP. The DPMP involves finding both optimal sequences of visits and the optimal heading angles at which these visits are made. It is a generalization of the Dubins Traveling Salesman Problem (DTSP). The multiple UAV case considered in this work is a generalization of the min-max TSP [8]. Both DTSP and min-max TSP are known to be computationally very challenging.

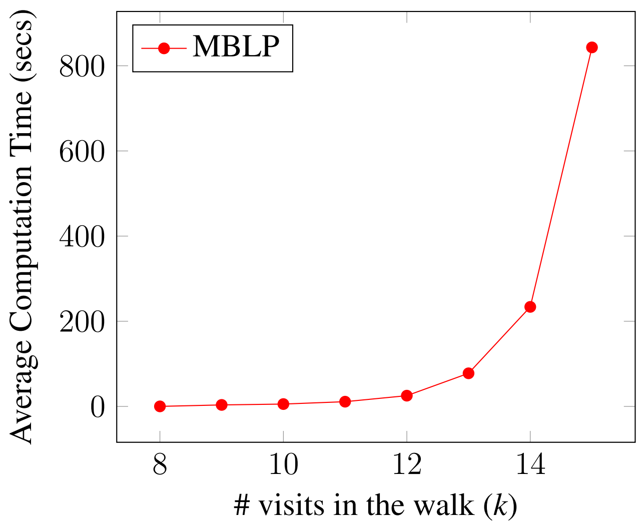


Figure 1.1: Plot showing the average time for computing optimal PMP walks using the MBLP formulation for 50 instances with 8 targets.

1.5 Contributions

The main contribution of this dissertation is a rigorous analysis of the structure of optimal solutions to the PMP. This analysis leads to the development of efficient formulations for PMP, resulting in significant savings in the computational effort required to solve the problem. Using the results of this analysis, we develop an ILP formulation that offers noteworthy improvement in the computation time required find optimal PMP solutions. This improvement for 8-target instances discussed in Section 1.4 is shown in Figure 1.2. More so, the average computation time for 20-target instances that were not solvable in reasonable time with the MBLP formulation was 0.14 seconds with the proposed ILP formulation. Additionally, we utilize the aforementioned results to develop a 1.5-approximation algorithm (an approximation algorithm with an approximation factor of 1.5) for the PMP using the aforementioned results.

The results presented for the PMP also extend to the case with curvature constraints. For this case, we develop a formulation that produces near-optimal solutions and tight lower bounds. Considering the computational difficulty in solving the formulation, we develop preliminary heuristics that are based on independently solving the sequencing and selection (of optimal heading angles)

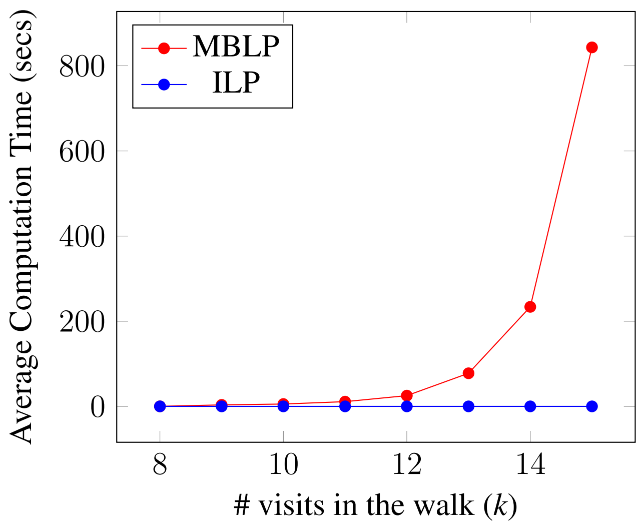


Figure 1.2: Plot showing a comparison of the average computation times using the MBLP and the ILP, for 50 instances containing 8 targets

problems to obtain feasible routes.

For the case with a separate depot, the structure of optimal solutions was much more involved and difficult to be deciphered. For this variant, we present a set of conjectures that were inspired by the results developed for the earlier variants. These conjectures, if were to be true, lead to an expeditious computation of optimal solutions. The conjectures are supported by extensive numerical simulations performed over 350 randomly generated instances.

The multiple UAV variant remains intractable due the difficulty involved in both modeling and solving the problem. In Chapter 5, we consider a special case of the problem and develop a formulation to quickly compute feasible solutions to the problem.

1.6 Literature Review

The revisit time objective was first proposed in [9] for a multi-agent patrolling problem; the author discussed several issues arising in multi-agent systems, proposed various multi-agent architectures, and different efficiency criteria to evaluate them. Since then, the revisit time criteria had been studied in a handful of articles, [10, 11, 12, 13, 14, 15, 16]. The problem of patrolling or persistent monitoring is different from the regular graph exploration problems due to its repetitive nature. Hence, the techniques from coverage or search problems cannot be directly applied here. For the same reason, optimal persistent monitoring routes must account for vehicle's fuel capacity. However, fuel constraints have not received as much attention in the literature as they deserve.

For the ideal case of unlimited fuel capacity, the problems considered in [10, 11, 12] take a form similar to the one considered herein (i.e., walks). In [10], the author considered the problem of monitoring equally weighted targets using multiple UAVs; due to its computational difficulty, the scope of the work was limited to heuristics. In [11], the problem of monitoring targets with different priorities/weights using a single UAV (with unlimited fuel capacity) was considered. Targets with different weights require different frequencies of monitoring, and the routing objective was to minimize the maximum *weighted* target revisit time. In this work, [11], authors proved that *optimal* infinite walks (optimal walks with unlimited fuel capacity) can be constructed by continuously repeating a finite walk (walk with a finite fuel capacity, as considered herein). They also provided two polynomial time approximation algorithms for the problem; the approximation factors were $log\ n$, where n is the number of targets, and $log\ \rho$, where ρ is the ratio of maximum and minimum weights of the targets. In [12], the authors posed the problem as a Mixed Integer Linear Program (MILP) and proposed an iterative sub-optimal scheme. The proposed scheme was observed to provide good quality solutions, at an order of magnitude faster than the optimal scheme.

While the vehicle routes in this work were described using *walks*, [13, 14, 15] considered other graph topologies. In [13, 14], the focus was on developing multi-agent patrolling strategies, based on optimal TSP tours for a single vehicle; while [13] considered the case of equally weighted targets, [14] considered the weighted case. Nevertheless, restricting a vehicle's route to TSP tours has two major drawbacks: 1) for a vehicle to *re*visit a target, it must visit all other targets assigned to it, 2) incomplete fuel usage could lead to sub-optimal vehicle routes (penalty for imprudent refueling). In [15], without reference to fuel constraints, the problem of multi-agent patrolling was considered for various graph topologies. For the case of cyclic roadmaps, the authors proved that the problem is NP-hard, and proposed the first constant factor approximation algorithm for the problem; the approximation ratio is 8. In the current work, we consider fuel constraints into the problem, and provide tightest possible bounds for monitoring equally weighted targets using a single vehicle. These results were used to develop good quality heuristics and approximation algorithms. The complete list of contributions of this work were presented in Section 1.5.

In other related work, [17, 18, 19, 20, 21] adopted an optimal control framework for a problem, where the uncertainty of information at a target increases with the time since the previous visit to it, and reduces with the time spent at the target. In this problem, as opposed to minimizing the maximum uncertainty over the targets, the objective is to minimize the aggregated uncertainty. For the case of one-dimensional spaces, [17] the problem was reduced to a parametric optimization problem that is non-convex. The decisions of the problem involve direction switching locations for the vehicles and the dwell times at these locations. [18] extends this work to a 2-dimensional spaces, by choosing elliptical trajectories for the vehicles. Other heuristic algorithms for this problem were presented in [19, 20, 21]. In [22], the authors considered a multi-agent continuous coverage problem, with severely degraded communication; a global representation of the environment is not available to the vehicles. Here, the number of visits made to each target is decided based on a pre-specified frequency distribution. This is not an appropriate criteria for a persistent monitoring setting, as a high number of visits to a target does not guarantee low target revisit time (duration between those visits could be high). While the aforementioned articles considered the problem of designing vehicle trajectories, [16] considered fixed trajectories; the focus was on developing velocity controllers for the vehicles, to achieve the minimum revisit time.

2. SINGLE VEHICLE PERSISTENT MONITORING PROBLEM*

In this chapter, we consider the case in which a set of equally weighted targets are monitored using a single UAV. The UAV is assumed to have no curvature constraints and can make sharp turns similar to a multi-rotor UAV. We begin the chapter with the statement of the PMP in Section 2.1. Then, we present a preliminary MBLP formulation for solving the PMP and show that the computation time required to solve this formulation is huge. To address the need for efficient techniques to solve the problem, in Section 2.4, we present a set of results characterizing the structure of optimal PMP solutions. These results lead to the development efficient formulations and approximation algorithms for the PMP, which are presented in Sections 2.5 to 2.8. We conclude this chapter in Section 2.9 by demonstrating the utility of the results developed in this chapter in solving the master problem, which involves finding both the optimal number of visits after which the UAV must be recharged, and its corresponding optimal sequence of visits.

2.1 Problem Statement

Let the targets (nodes) to be visited be denoted by the set $\mathcal{T} = \{1, 2, \cdots, n\}$. The travel times between any two distinct targets u, v $(u \neq v)$ in $\mathcal{T}$ be denoted by $c(u, v) > 0$. We assume that the targets are of equal priorities, and the travel times between them satisfy the triangle inequality, i.e., $c(u, v) + c(v, w) \geq c(u, w)$ for all $u, v, w \in \mathcal{T}$.

The UAV tasked with monitoring the targets must be recharged after every k visits. The UAV visits the targets in the order specified by the sequence $\mathcal{W} = (v_1, v_2, \cdots, v_{k+1})$ (such that $v_i \in \mathcal{T}$ for $i = 1, \cdots, k+1$); this sequence is repeated after every k visits. Here, we assume $k \geq n$, which allows each target to be visited at least once and some targets to be visited multiple times in a cycle of k visits. Due to repetition of nodes, this sequence is referred to as a *walk*. The node v_1 is the

*Copyright 2019 IEEE. Parts of this chapter have been reprinted with permissions from Sai Krishna Kanth Hari, Sivaku_mar Rathinam, Swaroop Darbha, Krishna Kalyanam, Satyanarayana G. Manyam, and David Casbeer, "Efficient Com-putation of Optimal UAV Routes for Persistent Monitoring of Targets [23]", ICUAS 2019 and Sai Krishna Kanth Hari, Sivakumar Rathinam, Swaroop Darbha, Krishna Kalyanam, Satyanarayana G. Manyam, and David Casbeer, "Bounding Algorithms for Persistent Monitoring of Targets using Unmanned Vehicles [24]", ICUAS 2019.

first or starting node of the walk from which the vehicle starts its mission, and is not counted in the number of visits made to targets. The node v_{k+1} is the last visited target in the walk.

We assume that the vehicle starts from and returns to the depot (chosen as of one the targets), after every k visits. Consequently, we have $v_1 = v_{k+1}$, and a walk with the same initial and terminal node is referred to as a closed walk. Note that a visit is counted only if a vehicle travels between two *distinct* targets. Therefore, we have $v_i \neq v_{i+1}$ for $i = 1, \cdots, k$. Moreover, the vehicle must visit all the targets in a walk of k visits. In contrast with the graph-theoretic terminology[25], we use the term *closed walk* differently to refer to a walk (a) with the same initial and terminal nodes, (b) in which every target is visited at least once and (c) successive visits correspond to distinct targets. Since we primarily deal with only closed walks in this dissertation, *we use the terms walk and closed walk interchangeably* when the context is clear.

For the purpose of illustration, consider Figure 2.1, which shows a closed walk of eight visits to five targets. Since a closed walk is continuously repeated after every k visits (in Figure 2.1, $k = 8$), it is often depicted using a cyclic representation as shown in Figure 2.2. The total time required to traverse through all the nodes in a walk is referred to as the *travel time of the walk*.

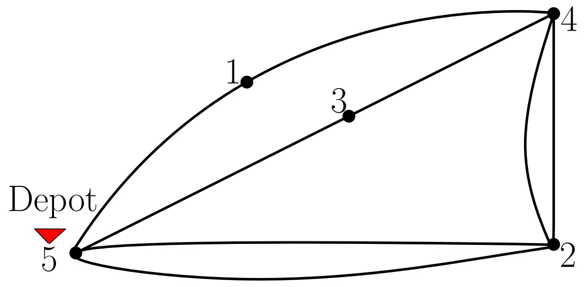


Figure 2.1: Graphical representation of a walk $\mathcal{W}(8) = (5, 1, 4, 2, 5, 3, 4, 2, 5)$ with 8 visits

Given a walk $\mathcal{W}(k)$ of k visits, the revisit time of a target $d \in \mathcal{T}$, denoted by $RT(d, \mathcal{W})$, is the maximum time taken between successive visits to the target, when the walk is repeated. For example, consider the walk shown in Figure 2.3. Target 2 is visited twice in the walk. Times between successive visits to target 2 are $t_1 = c(2, 5) + c(5, 1) + c(1, 4) + c(4, 2)$ and $t_2 = c(2, 5) +$

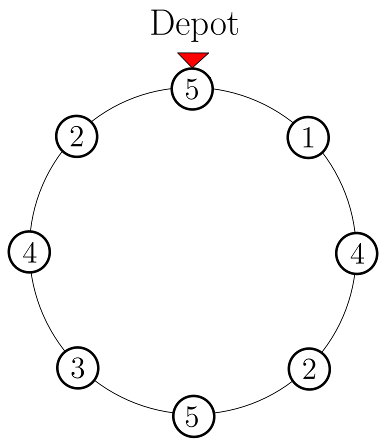


Figure 2.2: A cyclic representation of the walk $\mathcal{W}(8) = (5, 1, 4, 2, 5, 3, 4, 2, 5)$ (shown in Figure 2.1) with node 5 as the depot

$c(5, 3) + c(3, 4) + c(4, 2)$. Hence, the revisit time for target 2 is max $\{t_1, t_2\}$. Similarly, the revisit time of target 4 is max $\{t_3, t_4\}$ where $t_3 = c(4, 2) + c(2, 5) + c(5, 1) + c(1, 4)$ and $t_4 = c(4, 2) + c(2, 5) + c(5, 3) + c(3, 4)$. The maximum revisit time over all the targets is defined as the *revisit time of the walk* or *walk revisit time*, which is denoted by $\mathcal{R}(\mathcal{W}(k))$. Note that *if a target is visited exactly once in a walk, it has the maximum revisit time which is also equal to the travel time of the walk*. Therefore, in the example considered above, $\mathcal{R}(\mathcal{W}(k))$ = $RT(1, \mathcal{W})$ = $RT(3, \mathcal{W})$, as targets 1 and 3 are visited exactly once in the walk.

Using the aforementioned notation, the problem is stated as follows: *Given $k \geq n$ allowed visits for a UAV, find a closed walk of k visits with the minimum walk revisit time.*

2.2 Mixed Binary Linear Program (MBLP) Formulation

In this section, the PMP is modeled as a mixed binary linear program. Suppose the set of n targets to be monitored is given by $\mathcal{N}$ and the following data is available:

Input Data:

d : A target which is also a depot.

c_{ij}: Time taken by the UAV to travel target i to target j.

k : Number of visits after which the UAV must be refueled ($k \geq n$).

Then, a feasible walk and its revisit time are modeled using three sets of decision variables.

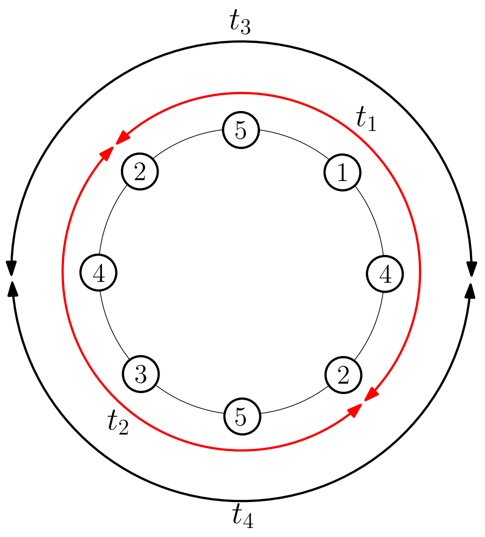


Figure 2.3: Figure depicting the revisit times of targets 2 and 4 in the walk $\mathcal{W}(8)$ considered in Figure 2.1

Decision Variables:

$$x_v^{ij} = \begin{cases} 1, \text{ if the } v^{th} \text{ visit in the walk is from target } i \\ \text{ to target } j; \\ 0, \text{ otherwise.} \end{cases}$$

f_v^i: Continuous variable indicating the time elapsed since the previous visit (in the walk)* to target $i \in \mathcal{N}$, after the completion of v^{th} visit in the walk.

z_v^i: Continuous variable used to reset the value of f_v^i to zero, after every visit to target i.

With the first set of binary variables, i.e., x_{ij}^v, a feasible walk with k visits is modeled using the following degree constraints:

2.2.1 Degree Constraints

The UAV starts from the depot during its first visit, and returns to the depot for its k^{th} visit.

*Here, the walk is considered in a cyclic manner, i.e., the first visit to target i in the walk is preceded by the last visit to target i in the walk

$$\sum_{j \in \mathcal{N}} x_1^{d,j} = \sum_{l \in \mathcal{N}} x_k^{l,d} = 1. \tag{2.1}$$

Every visit of the UAV must be between two distinct targets, i.e., no self-loops are allowed.

$$x_v^{ii} = 0, \quad \forall i \in \mathcal{N}, \;\; v \in \{1, \ldots, k\}. \tag{2.2}$$

Also, for every visit, the UAV has exactly one ’from’ target and one ’to’ target.

$$\sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{N}} x_v^{ij} = 1, \quad \forall v \in \{1, \ldots, k\}. \tag{2.3}$$

If the UAV visits target i for its $(v-1)^{th}$ visit, then it must leave the target for its v^{th} visit.

$$\sum_{l \in \mathcal{N}} x_{v-1}^{li} = \sum_{j \in \mathcal{N}} x_v^{ij}, \quad \forall i \in \mathcal{N}, \;\; v \in \{2, \ldots, k\}. \tag{2.4}$$

Each target must be visited at least once :

$$\sum_{v \in \{1, \ldots, k\}} \sum_{j \in \mathcal{N}} x_v^{ij} \geq 1, \quad \forall i \in \mathcal{N}. \tag{2.5}$$

These constraints model a feasible walk with k visits, which is then repeated for persistence. Next, we model the revisit time of the walk.

2.2.2 Walk Revisit Time

The time between consecutive visits to a target i is modeled with the help of two sets of continuous variables, f_v^i and z_v^i. After every visit v in the walk, f_v^i captures the time elapsed since the previous visit to target i. Unless the $(v-1)^{th}$ visit in the walk is to target i, the value f_v^i is higher than f_{v-1}^i by the amount of time required by the UAV to make its v^{th} visit.

If the $(v-1)^{th}$ visit in the walk is to target i, then the time accumulation restarts from 0, and the value f_v^i is simply equal to the time taken by the UAV to make its v^{th} visit.

The mathematical model capturing this idea takes the form of equations (2.6) and (2.7). Here, to accurately model the time accumulation and resetting, a third set of variables z_v^i is used. z_v^i equals f_{v-1}^i if the $v-1^{th}$ visit of the vehicle is to target i, and equals zero otherwise (see equation (2.6)).

$$z_{v-1}^i = f_{v-1}^i \sum_{l \in \mathcal{N}} x_{v-1}^{li}, \quad \forall i \in \mathcal{N}, \;\; v \in \{2, \ldots, k\}. \tag{2.6}$$

$$f_v^i - f_{v-1}^i + z_{v-1}^i = \sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{N}} x_v^{ij} c(i,j), \qquad \forall i \in \mathcal{N}, \quad v \in \{2, \ldots, k\}. \tag{2.7}$$

In constraint 2.7, if the $(v-1)^{th}$ visit in the walk is to target i, z_{v-1}^i negates the time accumulated till the $v-1^{th}$ visit; else, z_{v-1}^i is zero, and the time accumulation continues.

Recall that this walk of k visits is repeated a number of times, for persistent monitoring. So, the k^{th} visit in the walk is following by the 1^{st} in the next cycle of the walk. Moreover, the k^{th} visit in the walk is to the depot d. So, z_k^i is given by

$$z_k^i = \begin{cases} f_k^i, i = d, \\ 0, i \neq d. \end{cases}$$

Consequently, the time accumulation for the first visit in the sequence takes the following form:

$$f_1^i - f_k^i = \sum_{j \in \mathcal{N}} x_1^{d,j} c(d,j), \quad \forall i \in \mathcal{N} \backslash \{d\}. \tag{2.8}$$

$$f_1^i = \sum_{j \in \mathcal{N}} x_1^{d,j} c(d,j), \quad i = d. \tag{2.9}$$

Once the time accumulation in a walk is modeled, the maximum time between successive revisits to a target is the target revisit time. For target i, the target revisit time is simply given by $\max_{v \in \{1,\dots,k\}} \{f_v^i\}$. Then, the walk revisit time is given by $\max_{i \in \mathcal{N}, v \in \{1,\dots,k\}} \{f_v^i\}$:.

So, the optimization problem modeling the PMP can be posed as

$$(\mathcal{L}_1) : \min \quad \max_{i \in \mathcal{N}, v \in \{1,\dots,k\}} \{f_v^i\}, \tag{2.10}$$

subject to constraints (2.4)-(2.3) and (2.6)-(2.9).

This can be recast as a Mixed Binary Linear Program (MBLP) as follows:

2.2.3 Recasting as MBLP

The objective function can be re-written in terms of a variable mrt that acts as a proxy for the maximum revisit time among all the targets, as follows:

$$f_v^i \le mrt, \quad \forall i \in \mathcal{N}, \;\; v \in \{1,\dots,k\}. \tag{2.11}$$

Correspondingly, the objective (2.10) can be changed to

$$\mathcal{R}^*(k) = \min mrt, \tag{2.12}$$

subject to constraints (2.4)-(2.3), (2.6)-(2.9), and (2.11).

Note that the constraints (2.6) are bilinear as they involve product of f_{v-1}^i and x_{v-1}^{li}. These constraints can be linearized using the standard *big-M* procedure [26].

After linearization of constraints (2.6), they can be replaced with the following constraints:

$$z^i_{v-1} \leq M \sum_{l \in \mathcal{N}} x^{l,i}_{v-1}, \quad \forall i \in \mathcal{N}, \;\; v \in \{2, \ldots, k\}, \tag{2.13}$$

$$z^i_{v-1} \geq f^i_{v-1} - M \left(1 - \sum_{l \in \mathcal{N}} x^{l,i}_{v-1}\right), \forall i \in \mathcal{N}, \;\; v \in \{2, \ldots, k\}, \tag{2.14}$$

$$z^i_{v-1} \leq f^i_{v-1}, \quad \forall i \in \mathcal{N}, \;\; v \in \{2, \ldots, k\}, \tag{2.15}$$

$$z^i_{v-1} \geq 0 \quad \forall i \in \mathcal{N}, \;\; v \in \{2, \ldots, k\}. \tag{2.16}$$

The recast MBLP can be described with the objective given by (2.12), subject to linear constraints given by (2.4)- (2.3), (2.7)-(2.9), (2.11), and (2.13)-(2.16).

2.2.4 Computation Time for Solving the PMP Using the MBLP Formulation

To demonstrate the performance of the MBLP formulation, numerical simulations were performed on a number of randomly generated instances and implemented in CPLEX; the number of targets in the instances varied from $n = 5$ to $n = 20$. These simulations were performed on MacBook Pro with 16 GB RAM and Inter Core i7 processor with a processor speed of 2.5 GHz.

There were 50 instances with $n = 8$. For each instance, starting from $k = n(= 8)$, the time required to compute an optimal walk with k visits was computed; the exercise was repeated for incremental values of k. Then, for each k, the average computation time over all the 50 instances was computed, and is plotted in Figure 2.4. As mentioned in Section 1.4, average computation times were observed to increase with the number of visits. Beyond a certain number of visits, it was practically impossible to compute an optimal walk.

Furthermore, for instances with higher number of targets, say $n = 10$ and 20, computing optimal walks for even small number of visits is extremely difficult. To get a perspective, for an instance with 20 targets, the time expended for computing an optimal walk with 21 visits was 165,273 seconds, which is approximately 2 days. For larger k, the problem is intractable. Henceforth, the formulation is not scalable, and there is a need for better methods to compute optimal PMP walks. Towards this end, we set to analyse the structure of optimal solutions to the problem.

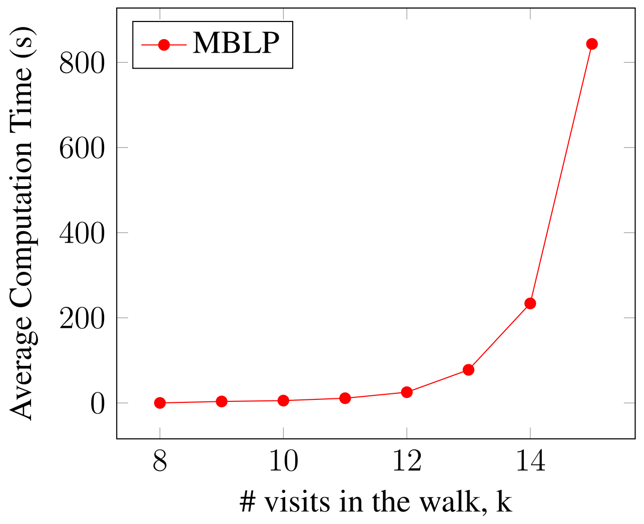


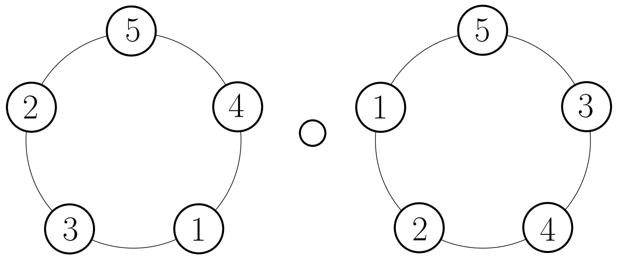
Figure 2.4: Average computation time for the MBLP, over 50 instances containing 8 targets

2.3 Terminology

In this section, we introduce the terminology required for understanding the structure of optimal walks.

Cyclic Permutation: Cyclic permutation of a walk chooses an intermediate target in the sequence of visits as the first element of the sequence, retaining the order of visits. For example, cyclic permutation of a walk $\mathcal{W} = (d, v_1, \dots, v_{r-1}, v_r, v_{r+1}, \dots, v_{k-1}, d)$, with respect to node v_r is $(v_r, v_{r+1}, \dots, v_{k-1}, d, v_1, \dots, v_{r-1}, v_r)$; it is denoted by $\mathcal{C}(\mathcal{W}, v_r)$. Note that performing *a cyclic permutation does not change the revisit time of the walk.*

Concatenation: Concatenation of *closed* walks $\mathcal{W}_1 = (d, v_1, \cdots, v_k, d)$ and $\mathcal{W}_2 = (d, u_1, \cdots, u_l, d)$ is defined as $\mathcal{W}_1 \circ \mathcal{W}_2 := (d, v_1, \cdots, v_k, d, u_1, \cdots, u_l, d)$. Figure 2.5 shows a walk obtained by concatenating two smaller walks. The concept of concatenation is helpful in the construction of feasible walks with larger number of visits. *It is easy to see that concatenating a walk with itself any number of times does not change the revisit time of any target.*


(a) Concatenation of two walks with 5 visits each to form a larger walk

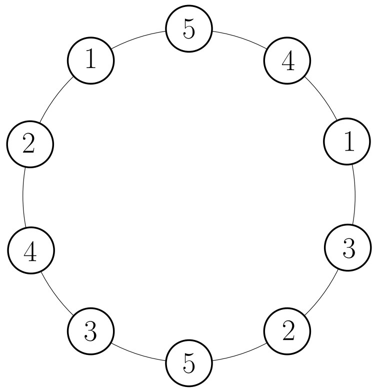

(b) A walk with 10 visits obtained by concatenation the above two walks

Figure 2.5: Concatenation of two walks with 5 visits each to form a walk with 10 visits.

Shortcut Walk: Given a closed walk $\mathcal{W}$, removal of a visit to any target $u \in \mathcal{W}$ is referred to as shortcutting a visit to u from $\mathcal{W}$. In this chapter, we refer to a shortcut walk of $\mathcal{W}$ as a *closed walk* obtained by shortcutting visits from $\mathcal{W}$ while retaining the last visit to each target. For example, $\mathcal{W}(8) = (5, 1, 4, 2, 5, 3, 4, 2, 5)$ (shown in Figure 2.1) is a closed walk with 8 visits with two visits to each of targets 2, 4 & 5, and one visit to each of targets 1 & 3. A shortcut walk $(5, 1, 4, 3, 4, 2, 5)$ of $\mathcal{W}(8)$ is obtained by shortcutting the first visit to targets 2 & 5, and retaining the other visits to all the targets in the order they appear in $\mathcal{W}(8)$, as shown in Figure 2.6. We remind the readers that shortcutting the last visits to targets 2, 4 & 5 or the only visits to targets 1 & 3 is not permitted by the definition.

Subwalk: A contiguous subsequence of a walk is referred to as its subwalk. A subwalk need not span all the targets, and need not have the same initial and terminal nodes. For example, consider the walk $\mathcal{W}(8)$ shown in Figure 2.7. The contiguous subsequences (2,5,1,4), (2,5,3) are subwalks

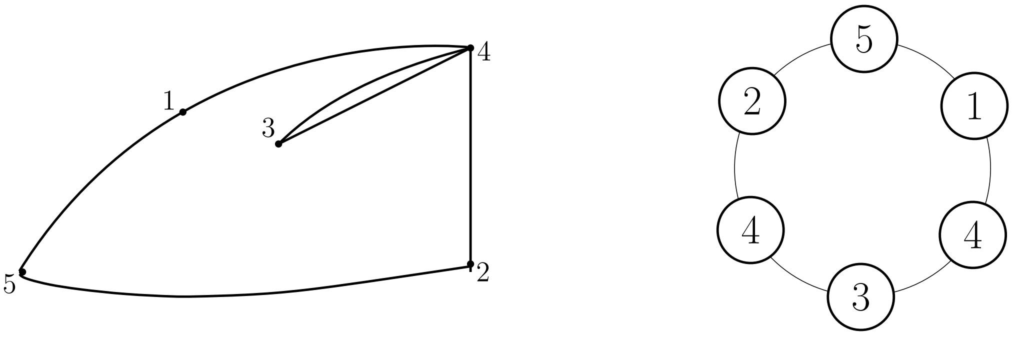


Figure 2.6: A shortcut walk $\mathcal{W}(6)$ obtained by shortcutting visits to targets 2 and 5 from the walk $\mathcal{W} = (5, 1, 4, 2, 5, 3, 4, 2, 5)$

of $\mathcal{W}(8)$.

If the terminal node of a subwalk $\mathcal{S}_1 = (d, v_1, \ldots, v_l)$ is the initial node of another subwalk $\mathcal{S}_2 = (v_l, v_{l+1}, \ldots, v_q)$, they can be concatenated as $\mathcal{S}_1 \circ \mathcal{S}_2 = (v_1, \ldots, v_l, \ldots, v_q)$. Note $\mathcal{S}_2 \circ \mathcal{S}_1$ may not be possible unless $v_q = d$.

Closed Subwalk: A subwalk with the same initial and terminal nodes is referred to as a *closed subwalk*. Similar to subwalks, a closed subwalk need not span all the targets, and hence need not be a walk. Figure 2.7 shows a closed subwalk of $\mathcal{W}(8)$ with target 5 as its initial and terminal nodes.

Travel Time of a Subwalk: Given a subwalk $\mathcal{W}_c := (v_1, \cdots, v_k)$, the time taken for the vehicle to traverse through all the nodes in $\mathcal{W}_c$ is defined as the travel time of the subwalk, and is denoted by $T(\mathcal{W}_c)$. That is, $T(\mathcal{W}_c) = \sum_{i=1}^{k-1} c(v_i, v_{i+1})$.

Terminus: In some closed subwalks, the end node is *not* visited in between; we refer to such an end node as a terminus. For example, $(5, 1, 4, 2, 5)$, $(5, 3, 4, 2, 5)$ are closed subwalks of $\mathcal{W}(8)$ with node 5 as their terminus.

Decomposition of a walk $\mathcal{W}(k)$ with respect to a node v: Suppose v is visited r times in a walk $\mathcal{W}(k) = (d = v_1, v_2, \ldots, v_{k_1}, \ldots, v_{k_2}, \ldots, v_{k_r}, \ldots, v_1 = d)$, i.e., $v_i \neq v$ except for $v_{k_1} = v_{k_2} =$

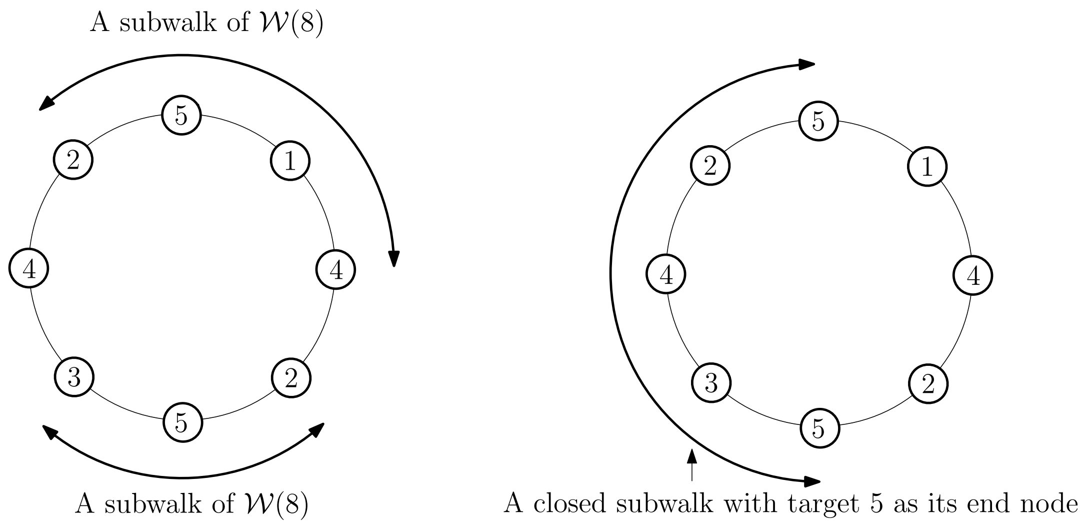

A subwalk of $\mathcal{W}(8)$
5
1
4
2
5
3
4
2
A subwalk of $\mathcal{W}(8)$
A closed subwalk with target 5 as its end node

Figure 2.7: Figure depicting subwalks and closed subwalks of $\mathcal{W}(8) = (5, 1, 4, 2, 5, 3, 4, 2, 5)$, the walk that is shown in Figure 2.1

$\ldots = v_{k_r} = v$; the decomposition of $\mathcal{W}(k)$ with respect to v is defined as:

$$\mathcal{W}(k) = \underbrace{(v_1, \ldots, v_{k_1})}_{\mathcal{S}_1} \circ \underbrace{(v_{k_1}, \ldots, v_{k_2})}_{\mathcal{W}_1} \circ \underbrace{(v_{k_2}, \ldots, v_{k_3})}_{\mathcal{W}_2} \circ$$

$$\ldots \circ \underbrace{(v_{k_{r-1}}, \ldots, v_{k_r})}_{\mathcal{W}_{r-1}} \circ \underbrace{(v_{k_r}, \ldots, v_1)}_{\mathcal{S}_2},$$

where $\mathcal{W}_1, \ldots, \mathcal{W}_{r-1}$ are subwalks with v as terminus and $\mathcal{S}_1, \mathcal{S}_2$ are subwalks with different initial and terminal nodes.

Revisit time of a Walk: To compute the revisit time of a given walk $\mathcal{W}$, choose any node $v \in \mathcal{W}$ and let v be visited r times in $\mathcal{W}$. The cyclic permutation $\mathcal{C}(\mathcal{W}, v)$ can be decomposed into r closed subwalks ($\mathcal{W}_1 \circ \mathcal{W}_2 \cdots \circ \mathcal{W}_r$) with each of them having v as its terminus. The revisit time of v in $\mathcal{W}$ is

$$RT(v, \mathcal{W}) := \max_{1 \le i \le r} \quad T(\mathcal{W}_i).$$

The revisit time $\mathcal{R}(\mathcal{W})$ of a given walk $\mathcal{W}$ is $\max_{1\leq v\leq n} RT(v,\mathcal{W})$.

Binding Subwalk: Let $\mathcal{W}(k)$ denote a feasible solution and $\mathcal{W}^*(k)$ represent an optimal solution to the PMP. A closed subwalk, $\mathcal{W}_b$, of a walk $\mathcal{W}$ is called a binding subwalk, if

- it has a terminus, say target d, and
- $T(\mathcal{W}_b) = \mathcal{R}(\mathcal{W})$.

It is easy to see that $RT(d,\mathcal{W}) = T(\mathcal{W}_b) = \mathcal{R}(\mathcal{W})$.

2.4 Structure of Optimal PMP Walks[*]

First, we show that any binding subwalk of an optimal walk must contain visits to all the targets.

Lemma 2.4.1. *Let $\mathcal{W}_b$ be a binding subwalk of $\mathcal{W}^*(k)$ for $k \geq n$. Each target is visited at least once in $\mathcal{W}_b$.*

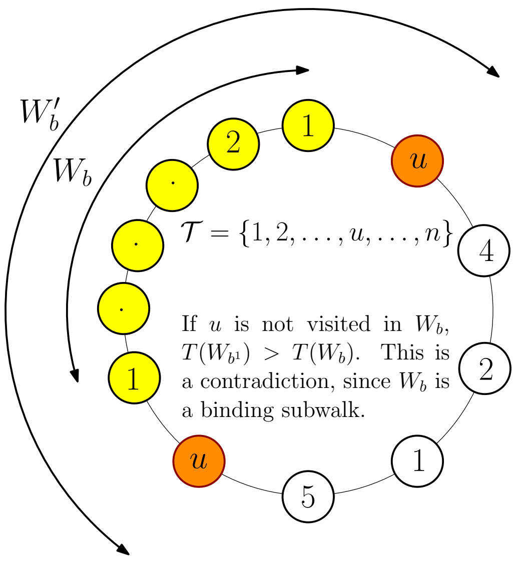
Proof. If a target u is not visited in $\mathcal{W}_b$, then the revisit time for u must be greater than the time required to traverse $\mathcal{W}_b$ as shown in Figure 2.8. This is because we can find a subwalk $\mathcal{W}_b'$ with u as its terminus that contains $\mathcal{W}_b$. However, this contradicts $\mathcal{W}_b$ being a binding subwalk. Therefore, each target must be visited at least once in $\mathcal{W}_b$.

□

This result leads to a lower bound on the optimal revisit time. In the following lemma, we prove that the optimal revisit time is lower bounded by the cost of an optimal TSP tour over the targets.

Lemma 2.4.2. $\mathcal{R}(\mathcal{W}^*(k)) \geq \mathcal{R}(\mathcal{W}^*(n)) = TSP^*$ *where TSP^* denotes the cost of an optimal TSP tour visiting all the n targets.*

[*]The proofs presented in this section also appear in the arxiv preprint [27]



W_b'
W_b
$\mathcal{T} = \{1, 2, \ldots, u, \ldots, n\}$
1
2
·
·
·
1
u
u
4
2
1
5
If u is not visited in W_b, $T(W_{b^1}) > T(W_b)$. This is a contradiction, since W_b is a binding subwalk.

Figure 2.8: A subwalk $\mathcal{W}_b$ (shown in yellow) with a missing node u cannot be binding, since there exists another subwalk $\mathcal{W}_b'$ (with u as its terminus) with a larger travel time

Proof. Any feasible solution to the TSP is also a feasible solution to the persistent surveillance problem with n visits where each target is visited exactly once, and vice versa. Therefore, $\mathcal{R}(\mathcal{W}^*(n)) = TSP^*$.

From Lemma 2.4.1, for $k > n$, each target is visited at least once in a binding subwalk $\mathcal{W}_b$. Shortcut any target that is visited more than once in $\mathcal{W}_b$ to obtain a tour, $\mathcal{W}_{tour}$. As the travel times satisfy the triangle inequality, short cutting any visit will not increase the revisit time, *i.e.*, $T(\mathcal{W}_b) \geq \mathcal{R}(\mathcal{W}_{tour})$. Therefore, $\mathcal{R}(\mathcal{W}^*(k)) = T(\mathcal{W}_b) \geq \mathcal{R}(\mathcal{W}_{tour}) \geq TSP^*$.

□

In some cases, this lower bound can be tightened, especially when the number of visits, k, is not an integral multiple of n.

Lemma 2.4.3. *Consider a walk $\mathcal{W}(k)$ with $k \geq n$ visits. One can express k in the quotient-remainder form as $pn + q$, where $p, q \in Z_+$, $p \geq 1$, and $0 \leq q \leq n-1$. Then, $\mathcal{W}(k)$ contains a closed subwalk $\mathcal{W}_s$ with a terminus, with at least $n + \lceil \frac{q}{p} \rceil$ visits. Consequently, $\mathcal{R}^*(k) \geq \mathcal{R}^*(n + \lceil \frac{q}{p} \rceil)$.*

Proof. Suppose the maximum number of visits in a closed subwalk with a terminus is at most $n + \lceil \frac{q}{p} \rceil - 1$. Then, we show that each target must be visited at least $p+1$ times in $\mathcal{W}(k)$ as follows: In any given walk, the maximum number of visits between consecutive revisits to a target is lower bounded by the average number of visits between consecutive revisits to the same target. If a target v is visited at most p times, the maximum number of visits between consecutive revisits to v is at least $\frac{pn+q}{p}(> n + \lceil \frac{q}{p} \rceil - 1)$. However, if each target is visited at least $p+1$ times, the total number of visits in the walk, k, is at least $(p+1)n > pn+q$, which contradicts the hypothesis.

Hence, there exists a closed subwalk $\mathcal{W}_s$ with a terminus, that has at least $n + \lceil \frac{q}{p} \rceil$ visits. Without loss of generality, one can assume that $\mathcal{W}_s$ spans all the targets. (If not, one can find a target $u \notin \mathcal{W}_s$, such that it is the terminus of another closed subwalk that contains $\mathcal{W}_s$ and spans all the targets). By definition, $\mathcal{R}(\mathcal{W}(k)) \geq T(\mathcal{W}_s)$, and from triangle inequality, it follows that $T(\mathcal{W}_s) \geq \mathcal{R}^*(n + \lceil \frac{q}{p} \rceil)$. Therefore, $\mathcal{R}(\mathcal{W}(k)) \geq R^*(n + \lceil \frac{q}{p} \rceil)$. Since this is true for any walk with k visits, we have $\mathcal{R}^*(k) = \mathcal{R}(\mathcal{W}^*(k)) \geq \mathcal{R}^*(n + \lceil \frac{q}{p} \rceil)$.

□

Remark: If $p \geq n-1$, it is clear that $q \neq 0 \Rightarrow \lceil \frac{q}{p} \rceil = 1$ and consequently, if k is not an integral multiple of n and $p \geq n-1$, we have $\mathcal{R}^*(k) \geq \mathcal{R}^*(n+1)$. We will use this fact in Theorem 2.

In the subsequent sections, we will construct feasible walks by concatenating smaller walks in such a way that the maximum revisit time equals the lower bound found in this section, thereby demonstrating optimality. The key result in the next section is that concatenating a binding subwalk with some of its shortcut walks and subsequently replacing the binding walk with the concatenated walk is not going to increase the maximum revisit time. This result is important in that walks with higher number of visits can be constructed without increasing the maximum revisit time.

2.4.1 Properties of Concatenating Feasible Solutions

The procedure to extend a walk into a larger feasible solution is given by the following lemma, and is depicted in Figure 2.9.

Theorem 2.4.4. *Given a walk $\mathcal{W}$, let $\mathcal{W}_b$ be its binding subwalk, and S_1, S_2 be its subwalks such that $\mathcal{W} = S_1 \circ \mathcal{W}_b \circ S_2$. Let $\mathcal{W}_b'$ be a closed subwalk obtained by shortcutting visits from $\mathcal{W}_b$, but retaining the last visits to targets. Let a closed walk $\bar{\mathcal{W}}$ be formed by concatenating $\mathcal{W}$ with $\mathcal{W}_b'$ as follows: $\bar{\mathcal{W}} = S_1 \circ \mathcal{W}_b \circ \mathcal{W}_b' \circ S_2$. Then, $\mathcal{R}(\bar{\mathcal{W}}) = \mathcal{R}(\mathcal{W})$.*

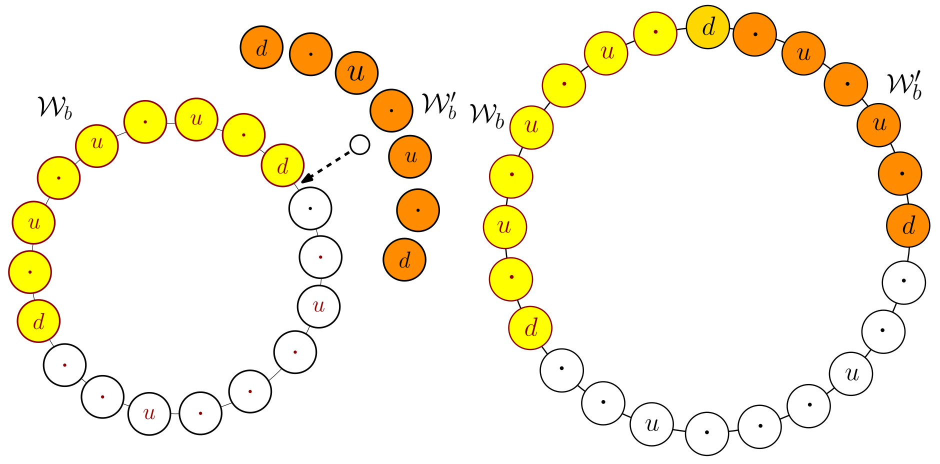


Figure 2.9: A closed walk $\mathcal{W}$ is shown on the left, with its binding subwalk $\mathcal{W}_b$ colored yellow. A subwalk $\mathcal{W}_b'$ (colored orange) is constructed from $\mathcal{W}_b$, by shortcutting visits to target u (retaining the last visit to u). Figure on the right shows the concatenated walk $\bar{\mathcal{W}}$, as discussed in Theorem 2.4.4.

Proof. See Appendix A. □

Next, we present two lemmas involving the cyclic permutation and concatenation of walks. The first of the two deals with the revisit time of a closed walk concatenated with itself an arbitrary number of times.

Lemma 2.4.5. *Let a closed walk $\bar{\mathcal{W}}$ be formed by concatenating $\mathcal{W}(k)$ with itself p times as follows:*

$$\bar{\mathcal{W}} := \underbrace{\mathcal{W}(k) \circ \cdots \circ \mathcal{W}(k)}_{p\ times}.$$

Then, for any node v_r

- $\mathcal{C}(\bar{\mathcal{W}}, v_r) = \underbrace{\mathcal{C}(\mathcal{W}(k), v_r) \circ \cdots \circ \mathcal{C}(\mathcal{W}(k), v_r)}_{p\ times}.$
- $RT(v_r, \bar{\mathcal{W}}) = RT(v_r, \mathcal{W}(k))$,

and thus $\mathcal{R}(\bar{\mathcal{W}}) = \mathcal{R}(\mathcal{W}(k))$.

Proof. See Appendix A.

□

The next lemma involves concatenating a walk with its shortcut walk and is crucial for piecing together smaller feasible walks to build a bigger walk while not increasing the maximum walk revisit time.

Lemma 2.4.6. *Let $\mathcal{W}(k)$ be a closed walk with $n+1 \leq k \leq 2n-1$ visits, and d be a target that is visited exactly once in $\mathcal{W}(k)$. Let $\mathcal{W}(k-1)$ be a walk formed by shortcutting a revisit from $\mathcal{W}(k)$. Consider a closed walk $\bar{\mathcal{W}}$ formed by concatenating $\mathcal{W}(k)$ for q (≥ 1) times and $\mathcal{W}(k-1)$ for p times as follows:*

$$\bar{\mathcal{W}} := \underbrace{\mathcal{W}(k) \circ .. \circ \mathcal{W}(k)}_{q\ times} \circ \underbrace{\mathcal{W}(k-1) \circ .. \circ \mathcal{W}(k-1)}_{p\ times},$$

Then,

$$\mathcal{C}(\bar{\mathcal{W}}, d) = \underbrace{\mathcal{C}(\mathcal{W}(k), d) \circ \cdots \circ \mathcal{C}(\mathcal{W}(k), d)}_{q\ times} \circ \underbrace{\mathcal{C}(\mathcal{W}(k-1), d) \circ \cdots \circ \mathcal{C}(\mathcal{W}(k-1), d)}_{p\ times}.$$

Proof. See Appendix A.

□

Remark: This lemma can be used to construct a feasible walk with K visits as follows: Let $K = rn + s$ for some integers $r > 1, s \geq 0$ and n being the number of targets. If $K \geq n(n-1)$, then $K = rn + s$, where $r \geq n-1, s \in \{0, 1, 2, \ldots, n-1\}$. Set $k = n+1, p = s$ and $q = r - s$ so that $pk + q(k-1) = p(n+1) + q(n) = (p+q)n + p = rn + s = K$; using the above lemma, one can construct a feasible walk for K visits from the feasible walk of $n+1$ visits and its shortcut walk of n. The following lemma shows that building such a bigger walk can be done without increasing the maximum walk revisit time.

Lemma 2.4.7. *Let $\mathcal{W}^*(k)$ be an optimal solution with $n+1 \leq k \leq 2n-1$ visits, and $\mathcal{W}(k-1)$ be a shortcut walk of $\mathcal{W}^*(k)$. Let a closed walk $\bar{\mathcal{W}}$ be formed by concatenating $\mathcal{W}^*(k)$ for q (≥ 1) times and $\mathcal{W}(k-1)$ for p times as follows:*

$$\bar{\mathcal{W}} := \underbrace{\mathcal{W}^*(k) \circ .. \circ \mathcal{W}^*(k)}_{q\ times} \circ \underbrace{\mathcal{W}(k-1) \circ .. \circ \mathcal{W}(k-1)}_{p\ times},$$

Then the maximum revisit time for $\bar{\mathcal{W}}$ is equal to $\mathcal{R}(\mathcal{W}^*(k))$ i.e.,

$$\mathcal{R}(\bar{\mathcal{W}}) = \mathcal{R}(\mathcal{W}^*(k)).$$

Proof. Since $k \leq 2n-1$, there is at least one node, d, in $\mathcal{W}^*(k)$ that is visited exactly once. Consider a cyclic permutation $\mathcal{C}(\mathcal{W}^*(k), d)$ of $\mathcal{W}^*(k)$. Let a closed walk $\mathcal{W}_q$ be formed by concatenating $\mathcal{C}(\mathcal{W}^*(k), d)$ for q times as follows:

$$\mathcal{W}_q = \underbrace{\mathcal{C}(\mathcal{W}^*(k), d) \circ .. \circ \mathcal{C}(\mathcal{W}^*(k), d)}_{q\ times}.$$

From Lemma 2.4.5 it follows that $\mathcal{R}(\mathcal{W}_q) = \mathcal{R}(\mathcal{C}(\mathcal{W}^*(k), d)) = \mathcal{R}(\mathcal{W}^*(k))$. As d is visited exactly once in $\mathcal{W}^*(k)$, we have $T(\mathcal{C}(\mathcal{W}^*(k), d)) = \mathcal{R}(\mathcal{C}(\mathcal{W}^*(k), d))$. Moreover, d is the terminus of $\mathcal{C}(\mathcal{W}^*(k), d)$. Therefore, $\mathcal{C}(\mathcal{W}^*(k), d)$ is a binding subwalk of itself. Because $T(\mathcal{C}(\mathcal{W}^*(k), d)) = \mathcal{R}(\mathcal{C}(\mathcal{W}^*(k), d)) = \mathcal{R}(\mathcal{W}_q)$, $\mathcal{C}(\mathcal{W}^*(k), d)$ is also a binding subwalk of $\mathcal{W}_q$.

Now consider $\mathcal{W}(k-1)$, which is formed by shortcutting a revisit from $\mathcal{W}^*(k)$. Note that d is visited exactly once in $\mathcal{W}(k-1)$. $\mathcal{C}(\mathcal{W}(k-1), d)$, is a shortcut walk of $\mathcal{C}(\mathcal{W}^*(k), d)$, which is a binding subwalk of $\mathcal{W}_q$. It follows from Theorem 2.4.4 that the revisit time of a walk formed by concatenating $\mathcal{W}_q$ with $\mathcal{C}(\mathcal{W}(k-1), d)$, is $\mathcal{R}(\mathcal{W}^*(k))$. Since $\mathcal{R}(\mathcal{W}_q \circ \mathcal{C}(\mathcal{W}(k-1), d)) = \mathcal{R}(\mathcal{W}^*(k))$, $\mathcal{C}(\mathcal{W}^*(k), d)$ is also a binding subwalk of $\mathcal{W}_q \circ \mathcal{C}(\mathcal{W}(k-1), d)$. Therefore, concatenating $\mathcal{W}_q \circ \mathcal{C}(\mathcal{W}(k-1), d)$ with $\mathcal{C}(\mathcal{W}(k-1), d)$ does not change the revisit time, and the process can be repeated for any number of times (say p). Hence, if $\mathcal{W}_p$ is defined as,

$$\mathcal{W}_p = \underbrace{\mathcal{C}(\mathcal{W}(k-1), d) \circ .. \circ \mathcal{C}(\mathcal{W}(k-1), d)}_{p\ times},$$

we have $\mathcal{R}(\mathcal{W}_q \circ \mathcal{W}_p) = \mathcal{R}(\mathcal{W}^*(k))$. From Lemma 2.4.6, $\mathcal{W}_q \circ \mathcal{W}_p$ is a cyclic permutation of $\bar{\mathcal{W}}$, and thus $\mathcal{R}(\bar{\mathcal{W}}) = \mathcal{R}(\mathcal{W}_q \circ \mathcal{W}_p) = \mathcal{R}(\mathcal{W}^*(k))$.

□

With this, we have enough tools to analyze the properties of optimal revisit time. To study the properties of optimal solutions, we split the number of visits into 3 categories: $n \leq k \leq 2n-1$, $2n \leq k < n^2 - n$ and $k \geq n^2 - n$. We begin with the case, $k \geq n^2 - n$, and prove that the optimal revisit time is a periodic function of k with a period n.

2.4.2 Bi-modal Property of Optimal Revisit Time

The focus of this subsection is to prove the main result of this chapter, that after a finite number of visits ($k \geq n^2 - n$), the optimal revisit is bi-modal with a period n. The two values it takes are: TSP^* when k is an integral multiple of n, and $\mathcal{R}(\mathcal{W}^*(n+1))$ when k is not an integral multiple of n.

Lemma 2.4.8. *$\mathcal{R}(\mathcal{W}^*(k)) = TSP^*$ for $k = pn$ where p is any positive integer.*

Proof. Consider the walk $\mathcal{W} := \underbrace{\mathcal{W}^*(n) \circ \cdots \circ \mathcal{W}^*(n)}_{p\ times}$. $\mathcal{W}$ is feasible to the problem with pn visits. Using Lemma 2.4.5, $\mathcal{R}(\mathcal{W}) = \mathcal{R}(\mathcal{W}^*(n)) = TSP^*$. Therefore, from Lemma 2.4.2, it follows that $\mathcal{W}$ is also an optimal solution to the problem with pn visits.

□

Lemma 2.4.9. *$\mathcal{R}(\mathcal{W}^*(k)) \leq \mathcal{R}(\mathcal{W}^*(n+1))$ for $k > n^2 - n$ and k is not an integral multiple of n.*

Proof. Consider the optimal walk $\mathcal{W}^*(n+1)$ with $n+1$ visits. Since exactly one target is visited twice in $\mathcal{W}^*(n+1)$, shortcut one of the visits to this target to get a new walk $\mathcal{W}$. Let $k = np + q$, $p \geq n-1$ and $1 \leq q \leq n-1$. Consider the following feasible solution to the problem with k visits:

$$\bar{\mathcal{W}} = \underbrace{\mathcal{W}^*(n+1) \circ \cdots \circ \mathcal{W}^*(n+1)}_{q\ times} \circ \underbrace{\mathcal{W} \circ \cdots \circ \mathcal{W}}_{(p-q)\ times}.$$

Note that the total number of visits in $\bar{\mathcal{W}}$ is $q(n+1) + (p-q)n = np + q = k$. Using Lemma (2.4.7), we obtain $\mathcal{R}(\bar{\mathcal{W}}) = \mathcal{R}(\mathcal{W}^*(n+1))$. Therefore, $\mathcal{R}(\mathcal{W}^*(k)) \leq \mathcal{R}(\bar{\mathcal{W}}) \leq \mathcal{R}(\mathcal{W}^*(n+1))$.

□

Theorem 2.4.10. *For $k \geq n^2 - n$, the optimal revisit time for the persistent surveillance problem can be only one of two values, i.e.,*

$$\mathcal{R}(\mathcal{W}^*(k)) = \begin{cases} \mathcal{R}(\mathcal{W}^*(n)), k = pn, p \text{ is an integer}, \\ \mathcal{R}(\mathcal{W}^*(n+1)), \text{ otherwise}. \end{cases}$$

Proof. The theorem follows from Lemmas (2.4.8) and (2.4.9), and the remark following Lemma (2.4.3). □

2.4.3 Monotonicity of Optimal Revisit Time

In this subsection, we show that the optimal revisit time is a monotonic function of the number of visits, for $n \leq k \leq 2n-1$.

Lemma 2.4.11. $\mathcal{R}(\mathcal{W}^*(k)) \leq \mathcal{R}(\mathcal{W}^*(k+1))$ *for* $n \leq k \leq 2n-2$.

Proof. Consider the optimal solution $\mathcal{W}^*(k+1)$. As $k+1 \leq 2n-1$, there is at least one node in $\mathcal{W}^*(k+1)$ that is visited exactly once. Let d denote one of these nodes. Since d is visited exactly once, $RT(d, \mathcal{W}^*(k+1)) = \mathcal{R}(\mathcal{W}^*(k+1))$. Also, as $k+1 > n$, there is at least one node (say u) that is visited more than once in $\mathcal{W}^*(k+1)$. Shortcut one of the revisits to u from $\mathcal{W}^*(k+1)$ to obtain a feasible walk $\mathcal{W}$ to the problem with k visits. As the travel times satisfy the triangle inequality, $RT(d, \mathcal{W}) \leq RT(d, \mathcal{W}^*(k+1))$. Therefore, $\mathcal{R}(\mathcal{W}^*(k)) \leq \mathcal{R}(\mathcal{W}) = RT(d, \mathcal{W}) \leq RT(d, \mathcal{W}^*(k+1)) \leq \mathcal{R}(\mathcal{W}^*(k+1))$. □

2.4.4 Bounds on Optimal Revisit Time

In this subsection, we provide a procedure to construct feasible walks for the case $2n < k < n^2 - n$, using the optimal solutions for $n < k \leq 2n-1$. The revisit times of these walks match the lower bounds presented earlier in this section, proving optimality of the constructed solutions.

Lemma 2.4.12. *Let* $k > n$ *be not an integral multiple of* n*, i.e.,* $k = pn + q$ *for some integers* $p \geq 1$, $1 \leq q \leq n-1$. *Then* $\mathcal{R}(\mathcal{W}^*(k)) \leq \mathcal{R}(\mathcal{W}^*(n + \lceil \frac{q}{p} \rceil))$.

Proof. It is sufficient to construct a feasible solution $\bar{\mathcal{W}}$ with k visits, such that $\mathcal{R}(\bar{\mathcal{W}}) = \mathcal{R}(\mathcal{W}^*(n + \lceil \frac{q}{p} \rceil))$. Consider an optimal walk $\mathcal{W}^*(n + \lceil \frac{q}{p} \rceil)$ with $n + \lceil \frac{q}{p} \rceil$ visits, and a corresponding walk $\mathcal{W}(n + \lfloor \frac{q}{p} \rfloor)$ with $n + \lfloor \frac{q}{p} \rfloor$ visits as follows:

1. $\mathcal{W}(n + \lfloor \frac{q}{p} \rfloor)$ is obtained by shortcutting a visit to a repeated target if $q \neq p$.

Let $q = ps + r$ for some integers $s \geq 1$ and $0 \leq r \leq p - 1$. For the sake of convenience, let us refer to $\mathcal{W}^*(n + \lceil \frac{q}{p} \rceil)$ as $\mathcal{W}^*$, and $\mathcal{W}(n + \lfloor \frac{q}{p} \rfloor)$ as $\mathcal{W}$. Then, a feasible solution to the problem can be constructed by concatenating $\mathcal{W}^*(n + \lceil \frac{q}{p} \rceil)$ times and $\mathcal{W}(n + \lfloor \frac{q}{p} \rfloor)$ times as follows:

$$\bar{\mathcal{W}} := \underbrace{\mathcal{W}^* \circ \cdots \circ \mathcal{W}^*}_{r \ times} \circ \underbrace{\mathcal{W} \circ \cdots \circ \mathcal{W}}_{(p-r) \ times}.$$

Note that the total number of visits in $\bar{\mathcal{W}}$ is $r(n + \lceil \frac{q}{p} \rceil) + (p - r)(n + \lfloor \frac{q}{p} \rfloor) = np + q = k$; the equality can be inferred from the following cases:

1. If $q = p$, $\lceil \frac{q}{p} \rceil = \lfloor \frac{q}{p} \rfloor = 1$. Hence, $r(n + \lceil \frac{q}{p} \rceil) + (p - r)(n + \lfloor \frac{q}{p} \rfloor) = pn + p = pn + q$.

2. If $q < p$, we have $r = q$, $\lceil \frac{q}{p} \rceil = 1$ and $\lfloor \frac{q}{p} \rfloor = 0$. Therefore, $r(n + \lceil \frac{q}{p} \rceil) + (p - r)(n + \lfloor \frac{q}{p} \rfloor) = q(n + 1) + (p - q)n = pn + q$.

3. If $q > p$, we have $\lceil \frac{q}{p} \rceil = s + 1$ and $\lfloor \frac{q}{p} \rfloor = s$. Hence, $r(n + \lceil \frac{q}{p} \rceil) + (p - r)(n + \lfloor \frac{q}{p} \rfloor) = r(n + s + 1) + (p - r)(n + s) = pn + ps + r = pn + q$.

Since $n + \lceil \frac{q}{p} \rceil \leq 2n - 1$, from Lemmas 2.4.5 and 2.4.7, we can conclude $\mathcal{R}(\bar{\mathcal{W}}) = \mathcal{R}(\mathcal{W}^*(n + \lceil \frac{q}{p} \rceil))$.

□

Theorem 2.4.13. *For any given number of visits $k \geq n$, $\mathcal{R}^*(k) = \mathcal{R}^*(n + \lceil \frac{q}{p} \rceil)$, where $k = pn + q$, $p \geq 1$, $0 \leq q \leq n - 1$ and $p, q \in Z_+$.*

Proof. The theorem follows from Lemmas 2.4.12, 2.4.3 and 2.4.8. □

A consequence of this result is that one needs to solve at most n distinct problems to construct optimal walks for any given $k \geq n$.

We conclude this section with another general result that holds for any $k \geq n$. Given an optimal walk $\mathcal{W}^*(k)$ with k visits, one can construct a feasible walk with $k + n$ visits without increasing the revisit time; this can be achieved by concatenating a walk with a shortcut version of its binding subwalk (as shown in Figure 2.10). As a consequence, the optimal revisit time for $k + n$ visits is upper bounded by the same for k visits; this provides a monotonically non-increasing subsequence.

Corollary 2.4.13.1 (of Theorem 2.4.4)**.** *The optimal revisit time for a PMP with k visits is an upper bound to the same with $k+n$ visits. That is, $\mathcal{R}(\mathcal{W}^*(k+n)) \leq \mathcal{R}(\mathcal{W}^*(k))$, $\forall k \geq n$.*

Proof. Consider an optimal walk $\mathcal{W}^*(k)$ with $k \geq$ visits, and any of its binding subwalk, say $\mathcal{W}_b$. Figure 2.10 shows $\mathcal{W}_b$ (colored yellow). Let $\mathcal{W}_b'$ be formed by shortcutting all except the last visits to targets from $\mathcal{W}_b$, such that it has exactly n visits. $\mathcal{W}_b'$ is shown in Figure 2.10 (colored orange). Now form a closed walk $\bar{\mathcal{W}}(k+n)$ with $k+n$ visits, by inserting $\mathcal{W}_b'$ in $\mathcal{W}^*(k)$, right next to $\mathcal{W}_b$ as shown in Figure 2.10. From Theorem 2.4.4, we have $\mathcal{R}(\bar{\mathcal{W}}(k+n)) = \mathcal{R}(\mathcal{W}^*(k))$. Since $\mathcal{R}(\bar{\mathcal{W}}(k+n))$ is a feasible solution to the problem with $k+n$ visits, the optimal revisit time for a problem with $k+n$ visits is at most $\mathcal{R}(\mathcal{W}^*(k))$, i.e., $\mathcal{R}(\mathcal{W}^*(k+n)) \leq \mathcal{R}(\mathcal{W}^*(k))$.

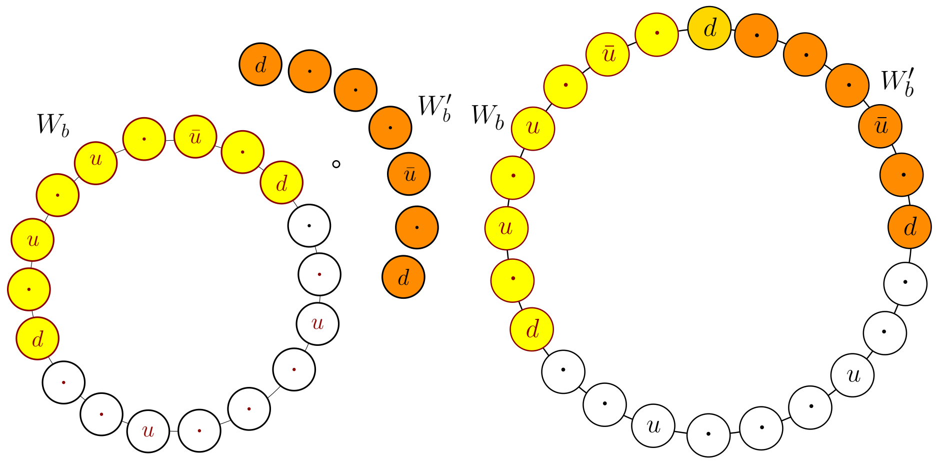


Figure 2.10: Figure showing construction of a solution to the problem with $k+n$ visits, given an optimal solution for the problem with k visits.

□

Based on these results, a typical plot of the optimal revisit time against the number of visit takes the form of Figure 2.11. A key result proved above is that an optimal walk with $k \geq 2n$ visits can be determined by computing only one of the following n solutions: $\mathcal{W}^*(n)$ to $\mathcal{W}^*(2n-1)$. This result is computationally beneficial due to two reasons. Firstly, it is simpler to compute optimal solutions with smaller number of visits. Secondly, for $n \leq k \leq 2n-1$, the revisit time of a walk is equal to its travel time (see the discussion above Figure 2.3). So, a simpler formulation can be employed for the computation of optimal walks. Such a formulation, and its advantages are discussed in the following section.

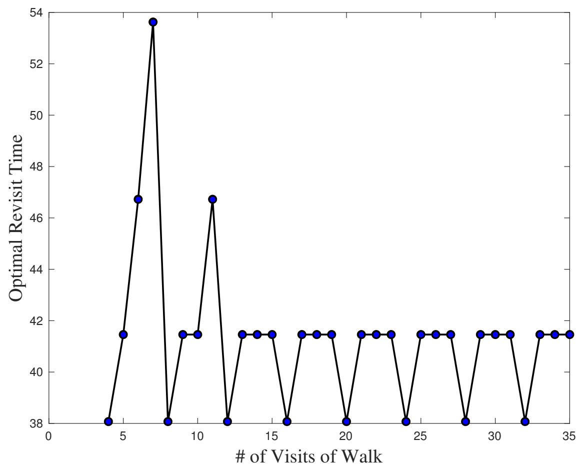
2.5 Integer Linear Program (ILP) Formulation

An Integer Linear Program (ILP) is presented for computing an optimal PMP walk with k visits when $n \leq k \leq 2n-1$. These solutions can be extended to construct optimal walks for any number of visits ($i.e.$, for $k \geq 2n$) using the results from the previous section. The formulation utilizes the fact that the travel time of a walk is equivalent to its revisit time, for the case $n \leq k \leq 2n-1$. Hence, the objective of the formulation is to find a walk with the least revisit time.

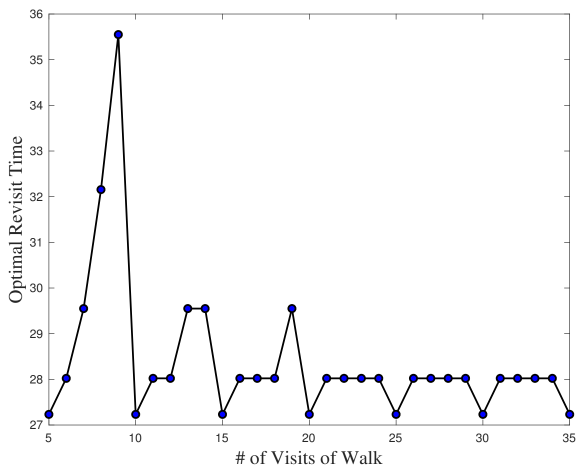
The benefits of this formulation are twofold: 1) constraints modeling the revisit time (refer to the MBLP formulation of [27]) that contribute overly to the computational difficulty can be circumvented; 2) an optimal route with the least travel time can be computed relatively *quickly*, due to the extensive research done on TSPs.

This formulation is interwoven with the graphical representation of the walk, in contrast to the cyclic representation used for the MBLP formulation. Consider a directed graph G with no self loops, such that its vertices represent the targets, edges represent the paths connecting the targets, and edge weights represent the travel times of these paths. Then, the decision variables of the formulation are the number of times each edge is utilized in a walk. A feasible solution is a subgraph satisfying the degree and connectivity requirements of a walk. The number of edges in the subgraph is equal to the number of visits in the walk.

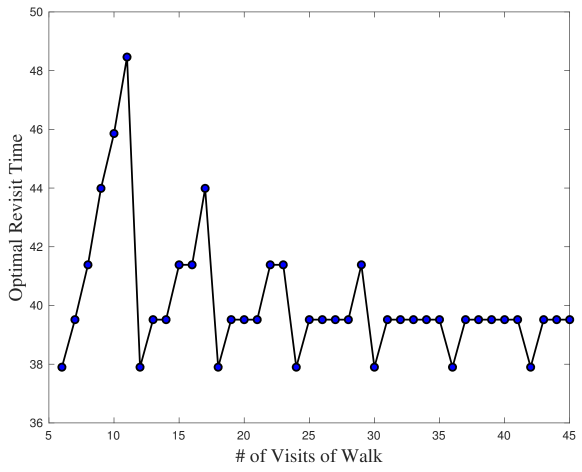
The integer linear program for finding an optimal walk with $n \leq k \leq 2n-1$ visits can be formulated as follows:



(a) Four equally weighted targets



(b) Five equally weighted targets



(c) Six equally weighted targets

Figure 2.11: The optimal revisit time as a function of the number of visits in the walk.

2.5.1 Nomenclature

Sets

$\mathcal{T}$: Set of all the targets to be monitored (set of vertices of G).

$\mathcal{E}$: Set of all edges in the graph G connecting any two targets in $\mathcal{T}$.

$\delta^-(i)$: Set of all incoming edges to target i, $\forall i \in \mathcal{T}$, i.e., $\delta^-(i) = \{(j,i) \in \mathcal{E} : j \in \mathcal{T} \setminus i\}$.

$\delta^+(i)$: Set of all outgoing edges from target i, $\forall i \in \mathcal{T}$, i.e., $\delta^+(i) = \{(i,j) \in \mathcal{E} : j \in \mathcal{T} \setminus i\}$.

Data

c_{ij}: Time taken by the UAV to travel along an edge $(i,j) \in \mathcal{E}$ (i.e., from target i to target j).

Decision Variables

x_{ij}: Integer variable (non-negative) indicating the number of times edge $(i,j) \in \mathcal{E}$ appears in the walk.

2.5.2 Optimization Problem

Objective

The objective here is to minimize the total travel time of the walk, that is,

$$Minimize \sum_{(i,j)\in\mathcal{E}} c_{ij}x_{ij} \tag{2.17}$$

Constraints

To ensure the resulting sub-graph is a feasible walk, degree and connectivity constraints are imposed. First and foremost, the number of edges in the solution must be equal to the number of visits in the walk.

- Total number of edges in the walk is k,

$$\sum_{e \in \mathcal{E}} x_e = k. \tag{2.18}$$

Next, each vehicle must be monitored/visited at least once. Hence, the number of incoming edges to a target $i \in \mathcal{T}$ must be at least 1.

- Each target must be visited at least once,

$$\sum_{e \in \delta^-(i)} x_e \geq 1, \forall i \in \mathcal{T}. \tag{2.19}$$

Whenever the vehicle enters (travels to) a target, it must also leave the target. In other words, the number of incoming and outgoing edges must be equal at every target $i \in \mathcal{T}$.

- In-degree must be equal to the out-degree at every target,

$$\sum_{e \in \delta^-(i)} x_e = \sum_{e \in \delta^+(i)} x_e, \forall i \in \mathcal{T}. \tag{2.20}$$

As the vehicle visits each target at least once, every subset of targets must have at least one outgoing edge. In simple terms, for the resulting sub-graph to be a feasible walk, it must be connected. These constraints can be modeled as follows, for every subset S of $\mathcal{T}$:

- $$\sum_{e \in \{(i,j): i \in S, j \in \mathcal{T} \setminus S\}} x_e \geq 1, \forall S \subset \mathcal{T}, S \neq \phi. \tag{2.21}$$

The objective function (2.17), together with constraints (2.18) - (2.21), forms the Integer Linear Program.

Commercial solvers such as CPLEX and Gurobi can be employed to solve the MIP. However, it is to be noted that the number of connectivity constraints is exponential ($2^n - 2$); including all such constraints renders the optimization problem intractable. To circumvent this issue, we implement these constraints in a "lazy" way. That is, we first solve a relaxed sub-problem by disregarding the connectivity constraints. If the solution to the sub-problem is a connected sub-graph, then it is also the desired optimal walk. If the resulting sub-graph is disconnected, then the sub-problem is re-solved by adding a connectivity constraint that is violated. This process is repeated until the resulting solution is a connected sub-graph.

Here, a violated connectivity constraint is identified by find a minimum weighted cut in the obtained sub-graph. If the weight of the minimum cut is zero, then the sub-graph is disconnected (note that the travel times are positive). The lazy addition of the violated cuts to the problem, can be easily implemented using the 'lazycut' feature in JuMP (a Julia package).

2.5.3 Obtaining a feasible walk from the ILP formulation

As discussed earlier, for $n \leq k \leq 2n-1$, the travel time of a walk is equal to its revisit time. Therefore, any sequence of visits containing exactly the same edges as in the obtained connected sub-graph, is an optimal walk. Moreover, it is known that at least one target is visited exactly once in this walk. Such a target has exactly one incoming and one outgoing edge in the sub-graph. Let target i be visited exactly once. Then, one can always find a sequence of visits such that the vehicle starts from target i, traverses through all the directed edges specified by the sub-graph, and returns to target i only at the end. Such a sequence of visits is an optimal walk with k visits, and can be depicted using the cyclic representation. As the starting target for this walk is i, one can simply perform a cyclic permutation to obtain a desired starting target. As mentioned in section 2.3, revisit time of a walk is not affected by cyclic permutations.

Once an optimal walk with $n \leq k \leq 2n-1$ visits is obtained, it can be used to construct optimal walks with larger number of visits, i.e., $k \geq 2n$.

2.6 Construction of Optimal Solutions for $k \geq 2n$

The construction procedure is elucidated using an illustrative instance consisting of 5 targets. The coordinates of the targets are shown in Figure 2.12.

Here, since $n = 5$, we first compute optimal solutions with 5 to 9 visits, and then utilize these

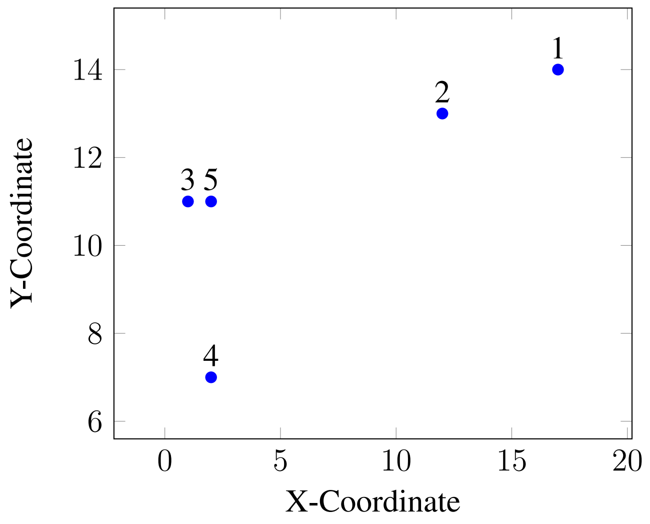


Figure 2.12: Figure showing the coordinates of 5 targets, in the illustrative example considered in Section 2.6

solutions to construct optimal walks with a larger number of visits, i.e., for $k \geq 10$. Optimal walks $\mathcal{W}^*(5)$, $\mathcal{W}^*(6)$, $\mathcal{W}^*(7)$, $\mathcal{W}^*(8)$ and $\mathcal{W}^*(9)$ are computed using the ILP formulation presented in Section 2.5. The obtained optimal walks for this instance are: $\mathcal{W}^*(5) = (1, 2, 5, 3, 4, 1)$, $\mathcal{W}^*(6) = (1, 2, 5, 3, 4, 2, 1)$, $\mathcal{W}^*(7) = (1, 2, 5, 3, 5, 4, 2, 1)$, $\mathcal{W}^*(8) = (1, 2, 5, 3, 5, 3, 4, 2, 1)$ and $\mathcal{W}^*(9) = (1, 2, 5, 3, 5, 3, 5, 4, 2, 1)$.

From each of $\mathcal{W}^*(6)$, $\mathcal{W}^*(7)$, $\mathcal{W}^*(8)$ and $\mathcal{W}^*(9)$, a revisit is shortcut if possible (adhering to the rules discussed in section 2.3) to form its corresponding shortcut walk; let the shortcut walks be $\mathcal{W}_s(5) = (1, 5, 3, 4, 2, 1)$, $\mathcal{W}_s(6) = (1, 2, 3, 5, 4, 2, 1)$, $\mathcal{W}_s(7) = (1, 2, 5, 3, 5, 4, 2, 1)$ and $\mathcal{W}_s(8) = (1, 2, 3, 5, 3, 5, 4, 2, 1)$ respectively. Note that these shortcut walks contain at most one less visit than their original walks, and the visit that is shortcut is arbitrarily chosen. $\mathcal{W}^*(5)$ is not shortcut, as it has no revisits (repeated visits).

Given any k ($\geq n$), it can be expressed as $pn + q$, where $p, q \in Z_+$, $p \geq 1$, and $0 \leq q \leq n - 1$. For ease of understanding, we consider 3 cases: 1) $q = 0$, 2) $0 < q \leq p$ and 3) $q > p$.

First, consider the case when $q = 0$. Notice here that k is an integral multiple of n, i.e., $k = pn$. In this case, an optimal solution with k visits can be simply constructed by concatenating $\mathcal{W}^*(n)$ with itself for p times (i.e., concatenating p copies of $\mathcal{W}^*(n)$). For example, consider

$k = 15$. Then, k can expressed as $3(5)$. Hence, an optimal walk with 15 visits is given by $\mathcal{W}^*(15) = \mathcal{W}^*(5) \circ \mathcal{W}^*(5) \circ \mathcal{W}^*(5)$ (see Figure 2.13).

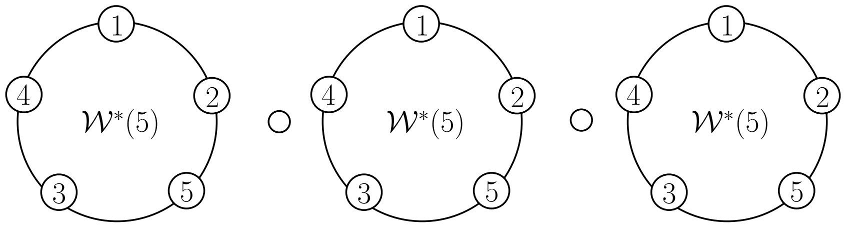


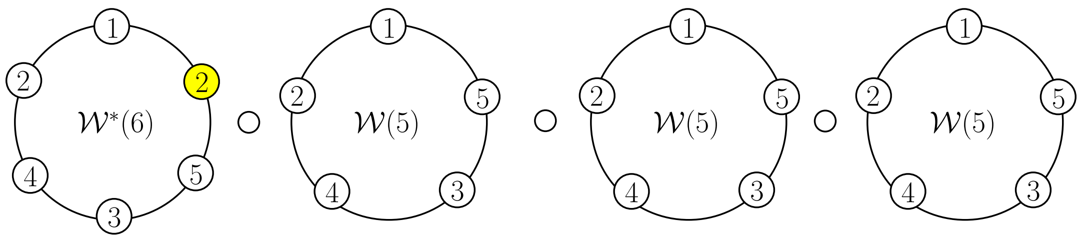
Figure 2.13: Figure showing the construction of optimal walks when k is an integral multiple of n; when $n = 5$ and $k = 15$, an optimal walk is given by $\mathcal{W}^*(15) = \mathcal{W}^*(5) \circ \mathcal{W}^*(5) \circ \mathcal{W}^*(5)$

Next, consider the case $0 < q \leq p$; it can be observed that $\lceil \frac{q}{p} \rceil$ is always 1 (and hence $n + \lceil \frac{q}{p} \rceil = n + 1$). Therefore, the construction procedure requires the computation of only $\mathcal{W}^*(n+1)$. Next, a revisit from $\mathcal{W}^*(n+1)$ is shortcut to form a shortcut walk $\mathcal{W}_s(n)$ with n visits. Subsequently, an optimal walk with k visits is constructed by concatenating $\mathcal{W}^*(n+1)$ for q times, and $\mathcal{W}_s(n)$ for $p - q$ times, as follows:

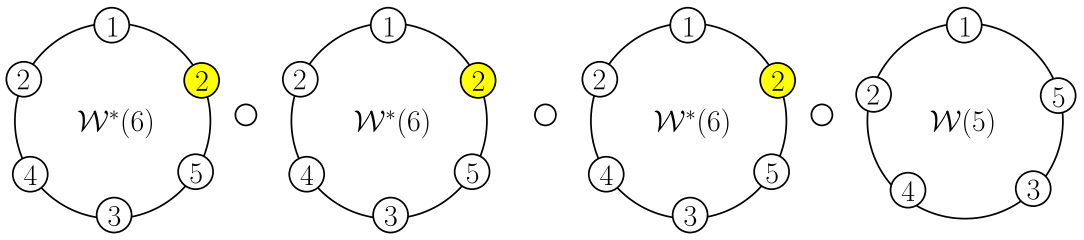
$$\bar{\mathcal{W}} := \underbrace{\mathcal{W}^*(n+1) \circ \cdots \circ \mathcal{W}^*(n+1)}_{q \ times} \circ \underbrace{\mathcal{W}_s(n) \circ \cdots \circ \mathcal{W}_s(n)}_{(p-q) \ times}.$$

For example, consider $k = 21$, which can be expressed as $4(5) + 1$. That is, $p = 4$ and $q = 1$. Hence, optimal walk with 21 visits can be obtained by concatenating $\mathcal{W}^*(6)$ and its shortcut walk $\mathcal{W}_s(5)$ as follows: $\mathcal{W}^*(21) = \mathcal{W}^*(6) \circ \mathcal{W}_s(5) \circ \mathcal{W}_s(5) \circ \mathcal{W}_s(5)$. Note that since $q = 1$ and $p - q = 3$ the concatenation involves 1 copy of $\mathcal{W}^*(6)$ and 3 copies of $\mathcal{W}_s(5)$. Similarly, $k = 23$ can be expressed as $4(5) + 3$. So, an optimal walk with 23 visits is given by $\mathcal{W}^*(23) = \mathcal{W}^*(6) \circ \mathcal{W}^*(6) \circ \mathcal{W}^*(6) \circ \mathcal{W}_s(5)$, as $q = 3$ and $p - q = 1$.

Here, it is worth noticing that when $k \geq n(n-1)$, we have $p \geq n - 1$, and q is no more than p. So, the construction procedure falls in one of the above two cases. One only needs to compute $\mathcal{W}^*(n+1)$ when k is not an integral multiple of n, and $\mathcal{W}^*(n)$ when k is an integral multiple of n.



(a) $\mathcal{W}^*(21) = \mathcal{W}^*(6) \circ \mathcal{W}_s(5) \circ \mathcal{W}_s(5) \circ \mathcal{W}_s(5)$



(b) $\mathcal{W}^*(23) = \mathcal{W}^*(6) \circ \mathcal{W}^*(6) \circ \mathcal{W}^*(6) \circ \mathcal{W}_s(5)$

Figure 2.14: Figures showing the construction of optimal walks for the case $0 < q \leq p$.

This explains the bi-modal property of the revisit time for $k \geq n(n-1)$.

When $q > p$, the construction procedure still involves concatenation of p smaller walks. However, the number of visits in the smaller walks is higher than $n+1$. Given $k = pn + q$ visits, one needs to compute an optimal walk with $n + \lceil \frac{q}{p} \rceil$ visits; let it be $\mathcal{W}^*(n + \lceil \frac{q}{p} \rceil)$. Note that $n + 1 \leq n + \lceil \frac{q}{p} \rceil \leq 2n - 1$.

To make the construction process clearer, we express q as $sp+r$, where $s \geq 1$ and $0 \leq r \leq p-1$ and consider 2 sub-cases.

First, consider the sub-case where $r = 0$; that is, q is a multiple of p. Then, an optimal walk of k visits is simply given by

$$\mathcal{W}^*(k) := \underbrace{\mathcal{W}^*(n + \lceil \frac{q}{p} \rceil) \circ \cdots \circ \mathcal{W}^*(n + \lceil \frac{q}{p} \rceil)}_{p \; times}$$

For example, consider $k = 14$. Then, it can be expressed as $2(5) + 4$. Here, $q(= 4) > p(= 2)$, and q is a multiple of p. So, an optimal walk with 14 visits is given by $\mathcal{W}^*(14) = \mathcal{W}^*(7) \circ \mathcal{W}^*(7)$.

Next, consider the sub-case where $r \neq 0$, i.e., q is not an integral multiple of p. Then, shortcut a revisit from $\mathcal{W}^*(n + \lceil \frac{q}{p} \rceil)$ to form $\mathcal{W}_s(n + \lfloor \frac{q}{p} \rfloor)$. Note that when q is not a multiple of p, $n + \lfloor \frac{q}{p} \rfloor = n + \lceil \frac{q}{p} \rceil - 1$. Then, an optimal walk with k visits is given by

$$\mathcal{W}^*(k) := \underbrace{\mathcal{W}^*(n + \lceil \frac{q}{p} \rceil) \circ \cdots \circ \mathcal{W}^*(n + \lceil \frac{q}{p} \rceil)}_{r\ times}$$

$$\circ \underbrace{\mathcal{W}_s(n + \lfloor \frac{q}{p} \rfloor) \circ \cdots \circ \mathcal{W}_s(n + \lfloor \frac{q}{p} \rfloor)}_{(p-r)\ times}.$$

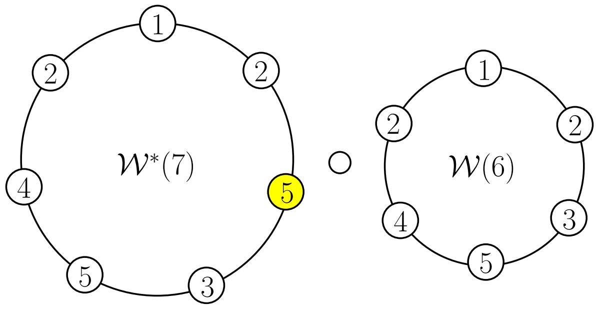
For example, consider $k = 13$. Then, it can be expressed as $2(5) + 3$. Then, $p = 2$ and $q = 3$, $n + \lceil \frac{q}{p} \rceil = 7$ and $n + \lfloor \frac{q}{p} \rfloor = 6$. Also, when q is expressed as $sp + r$ (i.e., $3 = 1(2) + 1$), we have $s = r = 1$. So, an optimal walk with 13 visits is given by $\mathcal{W}^*(13) = \mathcal{W}^*(7) \circ \mathcal{W}_s(6)$.

Using this procedure, optimal walks for any given $k \geq 2n$ can be constructed by computing only one of $\{\mathcal{W}^*(n), \ldots, \mathcal{W}^*(2n - 1)\}$; the ILP formulation can be employed for quick computation of the same.

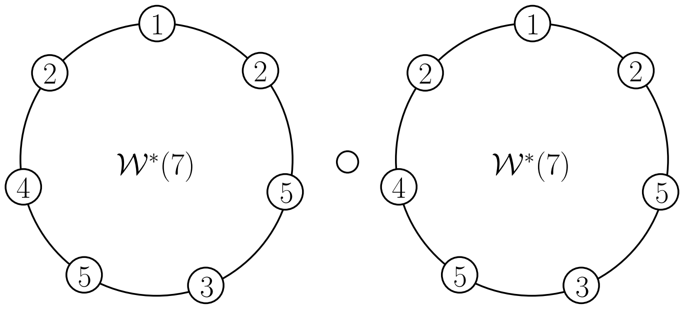
2.7 Computation Time with the ILP formulation

The strength of the theoretical results (presented in Section 2.4) and the ILP formulation (presented in Section 2.5) are demonstrated through numerical simulations performed on 250 instances. The number of targets in the instances varied from 5 to 20. For each of $n = 5, 8, 10, 15$ & 20, fifty instances (i.e., co-ordinates of the targets) were randomly generated. Then, the ILP formulation was employed to compute optimal walks with $n \leq k \leq 2n - 1$ visits.

For each n, the average computation times (average over all the 50 instances) were plotted against the number of visits; these plots for $n = 8, 15$ and 20 are shown in Figures 2.16, 2.17 and 2.18 respectively.



(a) $\mathcal{W}^*(13) = \mathcal{W}^*(7) \circ \mathcal{W}_s(6)$



(b) $\mathcal{W}^*(14) = \mathcal{W}^*(7) \circ \mathcal{W}^*(7)$

Figure 2.15: Figures showing the construction of optimal walks for the case $q > p$

For instances with $n = 8$, optimal solutions were computed using both ILP and MBLP formulations; the computation times are plotted in Figure 2.16. From the figure, it can be clearly seen that the performance of the ILP formulation is significantly better, compared to the MBLP formulation. The maximum and average computation times over $k = n$ to $k = 2n - 1$ visits were merely 0.038 and 0.036 seconds respectively for the ILP, compared to the values of 843.250 and 150.051 seconds for the MBLP.

Decisively, for larger values of n (10, 15 & 20), while the MBLP was interminable, the ILP provided optimal solutions within a fraction of a second.

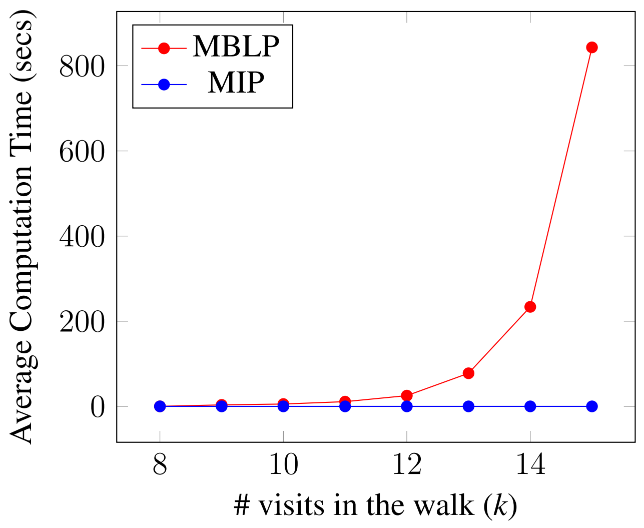


Figure 2.16: Plot showing a comparison of the average computation times using the MBLP and the ILP, for 50 instances containing 8 targets

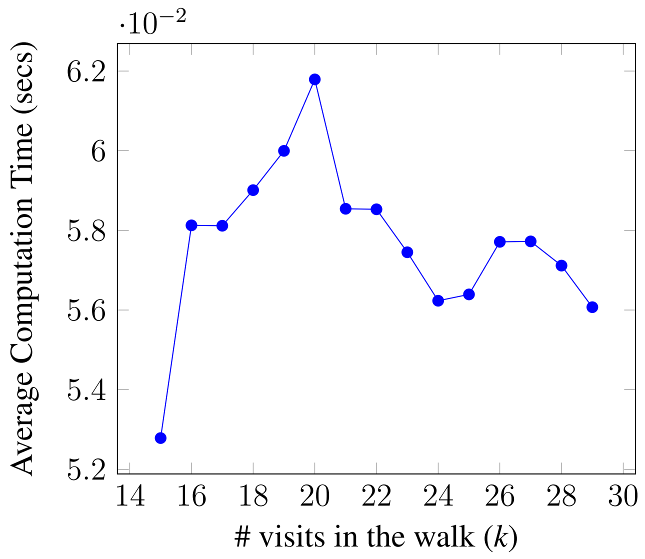


Figure 2.17: Average computation time for the ILP, over 50 instances containing 15 targets

2.8 1.5-Approximation Algorithm

We present a 1.5-approximation algorithm for an optimal PMP walk with k visits, where $n \leq k \leq 2n-1$. The algorithm is inspired by the famous Christofide's algorithm for the symmetric TSP. Consider an undirected complete graph G with: 1) no self loops, 2) vertices of G

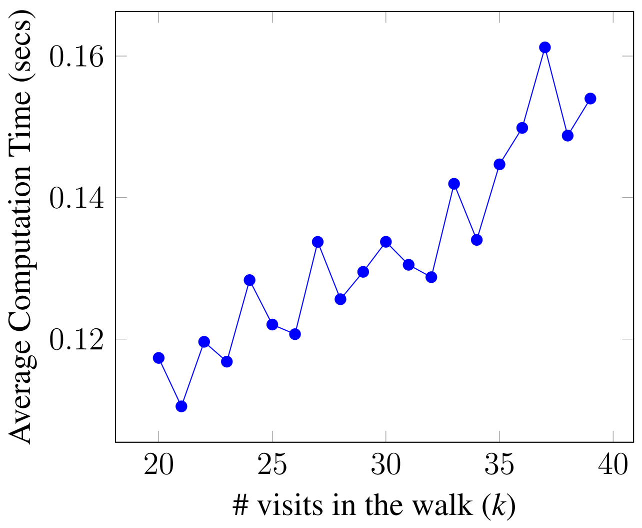


Figure 2.18: Average computation time for the ILP, over 50 instances containing 20 targets

representing the set of all targets $\mathcal{T}$, 3) edges $\mathcal{E}$ of G representing paths connecting the targets, and 4) edge weights representing Euclidean distances (travel times) between the targets. Then, the approximation algorithm involves four steps:

1. Construct an optimal 1-tree Q over $\mathcal{T}$; Q has exactly n edges and spans $\mathcal{T}$.

2. Add $k - n$ copies of the least cost edge (from the set $\mathcal{E}$) to Q, to form Q_k. Let the set of odd degree vertices of Q_k be $\mathcal{O}$.

3. Find a minimum-weight perfect matching, P, on $\mathcal{O}$. Add the edges of P to Q_k to form a walk (Eulerian) $\mathcal{W}_{pc}$.

4. Shortcut the required number of edges from $\mathcal{W}_{pc}$ to form $\mathcal{W}_c(k)$, the desired walk with k edges.

Each of the above steps can be performed in polynomial time. We show that the approximation ratio of this algorithm is 1.5. First, we begin with the fact that the cost (travel time) of an optimal 1-tree serves as a lower bound to the optimal TSP cost [28], i.e., $T(Q) \leq \mathcal{R}^*(n)$. Using this fact, we show that the cost of Q_k is no more than $\mathcal{R}^*(k)$. To prove this, we present two intermediate results.

First, we show that any walk $\mathcal{W}$ (Eulerian and connected) with more than n visits can be shortcut. We consider two cases: 1) all targets are visited more than once in $\mathcal{W}$, 2) at least one target is visited exactly once in $\mathcal{W}$.

<u>All targets are visited more than once in $\mathcal{W}$</u>: Consider an arbitrary visit of $\mathcal{W}$, and label it as v. If the 'from' target of v and the 'to' target of the next visit in the walk are different, then shortcut the visit v. If they are the same, shift the label v to the next visit in the walk, and repeat the process till a visit is shortcut. If all the visits are exhausted, then it implies that only two targets are repeated in $\mathcal{W}$, contradicting the fact that every target is visited at least once in a walk.

<u>At least one target is visited exactly once in $\mathcal{W}$.</u>: Consider a target that is visited exactly once. Label the visit to that target as v. If the 'from' target of v is visited more than once, then shortcut the visit preceding v. If the 'from' target of v is visited exactly once, then shift the label v to the preceding visit in the walk and repeat the process till a visit is shortcut. If none of the visits can be shortcut, it implies that every target is visited exactly once, contradicting the assumption that there are more than n visits in the walk.

Next, we show that $\mathcal{R}^*(n) + (k-n)c_{min} \leq \mathcal{R}^*(k)$, where c_{min} is the cost of the least cost edge of $\mathcal{E}$. We show this by contradiction. Let $\mathcal{W}^*(k)$ be an optimal walk with k visits. Suppose $\mathcal{R}^*(k) < \mathcal{R}^*(n) + (k-n)c_{min}$. If $k = n$, we have $k - n = 0$ and $\mathcal{R}^*(k) < \mathcal{R}^*(n)$, leading to a contradiction. If $k > n$, shortcut $k - n$ edges from $\mathcal{W}^*(k)$ to form $\mathcal{W}_s(n)$, a walk with n visits. Then, $T(\mathcal{W}_s(n)) + (k-n)c_{min} \leq T(\mathcal{W}^*(k)) = \mathcal{R}^*(k) < \mathcal{R}^*(n) + (k-n)c_{min}$. This implies that $T(\mathcal{W}_s(n)) < \mathcal{R}^*(n)$, i.e., there exists a tour/walk $\mathcal{W}_s(n)$ with n visits and a travel time less than the optimal TSP cost. This provides a contradiction. Therefore, $\mathcal{R}^*(n) + (k-n)c_{min} \leq \mathcal{R}^*(k)$.

From the above result it follows that the cost of Q_k is no more than $\mathcal{R}^*(k)$. This is because $T(Q_k) = T(Q) + (k-n)c_{min}$ and $T(Q) \leq \mathcal{R}^*(n)$. So, $T(Q_k) \leq \mathcal{R}^*(n) + (k-n)c_{min} \leq \mathcal{R}^*(k)$. Even though Q_k has k edges, it is not necessarily Eulerian due to the possibility of the presence of odd degree vertices. However, the number of such odd degree vertices is even [29]. So, one can construct a minimum-weight perfect matching P on $\mathcal{O}$ (the set of all odd degree vertices of Q_k) to form an Eulerian walk $\mathcal{W}_{pc}$.

The cost of $\mathcal{W}_{pc}$ is at most $\frac{3}{2}\mathcal{R}^*(k)$, i.e., $T(\mathcal{W}_{pc}) \leq \frac{3}{2}\mathcal{R}^*(k)$. This follows from the following facts: 1) the cost of P is no more than half the cost of an optimal TSP tour [30], i.e., $T(P) \leq \mathcal{R}^*(n)$, 2) $\mathcal{R}^*(n) \leq \mathcal{R}^*(k)$, 3) $T(Q) \leq \mathcal{R}^*(k)$ and 4) $T(\mathcal{W}_{pc}) = T(Q_k) + T(P)$. However, the number of edges in $\mathcal{W}_{pc}$ could be more than k. Nonetheless, because $\mathcal{W}_{pc}$ is Eulerian, it can be shortcut a required number of times to form a feasible walk, $\mathcal{W}_c(k)$, with k edges/visits. For the reason that $\mathcal{W}_c(k)$ is a shortcut walk of $\mathcal{W}_{pc}$, we have $T(\mathcal{W}_c(k)) \leq T(\mathcal{W}_{pc}) \leq \frac{3}{2}\mathcal{R}^*(k)$, and because $n \leq k \leq 2n-1$, the revisit time of $\mathcal{W}_c(k)$ is equal to its travel time. Therefore, $\mathcal{R}(\mathcal{W}_c(k)) \leq \frac{3}{2}\mathcal{R}^*(k)$ and the proposed algorithm has an approximation ratio of 1.5.

2.9 Solving the Master Problem

The main problem of interest in persistent monitoring missions is the following: given that the UAV can be recharged after at most V visits, find: 1) an optimal $n \leq k \leq V$ after which the UAV must be recharged; and 2) a walk with k visits that has the least revisit time. PMP answers the second question. With efficient techniques available (presented above) to solve PMP , the master problem can be solved easily by computing optimal PMP routes for each k in the range $n \leq k \leq V$, and picking the k with the least objective value (sum of the revisit time and charge penalty).

2.9.1 High Charge Penalty

When the relative cost of penalty, μ, high, it is optimal to utilize all the V visits, thereby avoiding a large penalty. In such a case, an optimal solution to the master problem is given by $k = V$ and $\mathcal{W}^*(V)$.

2.9.2 Low Charge Penalty

When $m\mu$ is low, optimum k is the largest $k(\leq V)$ for which $\mathcal{R}^*(k) = \mathcal{R}^*(n)$ (this is because of the initial monotonicity of the optimal revisit time). If $\mathcal{R}^*(n+1) > \mathcal{R}^*(n)$, i.e., if the monotonicity is strict, then optimal k is the largest integral multiple of n that is less than V. As a consequence, it is profitable to merely repeat an optimal TSP solution.

2.9.3 Comparable Charge Penalty

When the charge penalty is comparable to optimal revisit times, one needs to compare objective values for at most n distinct values, i.e., for $V - n + 1 \leq k \leq V$. Optimal k is the one with the least $\mathcal{R}^*(k) + \mu \times m \times (V - k)$. This result follows from Corollary 2.4.13.1.

3. PERSISTENT MONITORING PROBLEM WITH MOTION CONSTRAINTS

Persistent monitoring of vast areas, especially those with harsh environmental conditions, entails greater challenges such as increased recharging needs of the UAVs and the susceptibility of monitoring vehicles to strong winds. This problem can be tackled by the use of Fixed-wing UAVs, which are known for their long flight times and resilience to strong winds. However, these vehicles are not easily maneuverable; they cannot make sharp turns, can move only forward, and have constraints on their minimum turn radii. For route planning purposes, the motion of these vehicles is typically modeled by a set of kinematic constraints referred to as Dubins model. This model is known to produce near-feasible paths for Fixed-wing UAVs. In Dubins model, the following assumptions are made on the UAVs: 1) they travel at a constant speed; 2) they cannot move backwards; 3) their minimum turn radii are constrained. To satisfy this model, the angle at which the UAV enters a target must be the same as the angle at which it leaves the target.

These motion constraints add an additional layer of complexity to the optimal route planning problem, as one needs to choose not only the sequence in which the targets are visited by the UAV, but also the angle at which the targets are visited by the monitoring vehicle. Both target locations and arrival angles influence the travel times between targets. Given the location information of a pair of targets and the angle at which the UAV visits those targets, the problem of finding the shortest path (and hence the minimum travel time in this case) between the targets that satisfies the above described motion constraints was solved by an American mathematician Lester Dubins. According to his work presented in [31], such an optimal path between the targets consists of at most three components: 1) a straight line segment, referred to as S; 2) a left turn with the minimum turn radius, referred to as L; and 3) a right turn with the minimum turn radius, referred to as R. Specifically, an optimal path between a pair of targets is given by at least one of the following combinations of the above mentioned components: LRL, RLR, LSL, LSR, RSL, RSR. For example, LRL indicates that the vehicle must first take a left turn with the minimum turn radius, followed by a right turn with the minimum turn radius, followed again by a left turn with the minimum turn radius. Using this result, for a given UAV, one can calculate the minimum travel time between a given pair of targets, provided the angles of arrival/departure at the targets are known.

Given the additional layer of complexity introduced by motion constraints, the computational difficulty in finding optimal routes for this variant is expected to be higher than that of the primary version considered in Chapter 2. Before discussing the computational aspects of this variant, let us first revisit the notion of a walk and the criteria of optimality for this case.

3.1 Problem Statement

In this variant, similar to Chapter 2, a walk consists of a set of visits. However, in addition to the indices of the targets that are visited, every visit also comprises of the angle at which the UAV visits the target. Accordingly, a walk with k visits takes the form $\mathcal{W}_{db}(k) = \{t_0(h_0), t_1(h_1), t_2(h_2), \ldots, t_k(h_k)\}$, where: t_i, $i = \{1, \ldots, k\}$, represents the target that is visited at the end of the i^{th} visit; h_i indicates the heading angle at which target t_i is visited by the UAV at the end of the i^{th} visit; and $t_0(h_0) = t_k(h_k)$. With this notion of a walk, the definitions of the target revisit time and walk revisit time remain unchanged, and the problem of interest can be stated as the following:

Given that the UAV is recharged at the end of every k visits, find a walk with k visits that satisfies the motion constraints and minimizes the (walk) revisit time.

As we consider the Dubins model to represent the motion constraints, we refer to this problem as the *Dubins Persistent Monitoring Problem (DPMP)*. Nonetheless, the results developed in this Chapter can also be applied to other models such as Reeds-Shepp model that is suitable for ground robots.

3.2 Structure of Optimal DPMP Walks

Next, we examine the applicability of the computation-time-saving results developed in Section 2.4 to the current problem variant. It is to be noted that all the results of Section 2.4 extend directly to the case with asymmetric travel times. For these results to hold true for the current variant, one only needs to show that the Dubins travel times (travel time for optimal Dubins paths between target pairs) satisfy the triangle inequality. From the discussion in the second paragraph of this chapter, it is known that an optima Dubins path from target i to target k and that from target j to l take one of the aforementioned forms, i.e., LRL RSL etc. Let the minimum time required by a Dubins vehicle (a vehicle that satisfies the Dubins model) to travel from a given target i with a heading angle p to another target j with a heading angle q be denoted by $c(i,p,j,q)$. Suppose these times do not satisfy the triangle inequality, i.e., for three distinct targets i, j and l and corresponding heading angles of p, q and r for the vehicles at the targets, we have $c(i,p,j,q) + c(j,q,l,r) < c(i,p,l,r)$. Then, it implies that there exists a curvature constrained path from i to l via target j that is not of the aforementioned form (because the path contains more than three different components; for example, the path can take the form LRL+RSL with components) with a time time less than the minimum Dubins travel time from target i to target l. This is contradictory to the Dubins results presented in [31]. Therefore, the Dubins travel times satisfy the triangle inequality.

As a consequence, one only needs to solve the DPMP for n base cases, i.e.., from n to $2n-1$ visits, to completely determine optimal solutions for any given $k \geq n$. Additionally, for $n \leq k \leq 2n-1$, it is enough to compute walks with the least travel time, which is relatively easier to compute compared to walks with the least revisit time, as observed in Chapter 2.

3.3 Generalized PMP: Near-Optimal DPMP Walks

Despite the computational savings achieved from the above results, finding optimal DPMP walks remains challenging. The main difficulty arises from the availability of infinite number of heading (arrival/departure) angles for the UAV choose from, for every visit to a target. To circumvent this challenge, we consider a simplied version of this problem, in which the UAV is allowed to visit a target at only a finite number of apriori chosen heading angles. We propose a mathematical framework in the form of the *Generalized Persistent Monitoring Problem (GPMP)* [32] to solve this simplified version.

In GPMP, each target is assumed to be made up of several sub-targets, which could represent either different heading angles or other selection decisions available at the target. Given $n \leq k \leq$

$2n-1$ *visits, the objective of GPMP is to find a sequence of* k *visits to these sub-targets such that : 1) the UAV starts from the (sub-target chosen as the) depot for the first visit, and returns to the depot at the end of the* k^{th} *visit; 2) every visit of the UAV is made to exactly one sub-target; 3) no visit is made between sub-targets corresponding to the same target; 4) at least one sub-target corresponding to every target is visited; and 5) the travel time of the obtained sequence is minimized.*

GPMP is useful not only for obtaining feasible walks for DPMP (see Subsection 3.3.1), but also for developing lower bounds to the problem (see Subsection 3.3.2)). Numerical simulations presented at the end of this section (Subsection 3.3.4) suggest that the *feasible DPMP walks obtained by this method are near-optimal.*

3.3.1 Feasible DPMP Walks

To obtain feasible DPMP walks, the UAV is allowed to visit the targets only at apriori chosen heading angles as shown in Figure 3.1; however, the angle at which the UAV arrives at a target must be the same as the angle at which it departs from the target to make its next visit. This simplified version of the problem can be recast as a GPMP, where each heading angle allowed at a target is considered as its sub-target; each sub-target has a unique tuple of the allowed heading angle and the coordinates of its target. The travel times between sub-targets of different targets are given by the Dubins result presented at the beginning of this chapter.

Solving the GPMP using these travel times provides an optimal walk in which the UAV is allowed to arrive at exactly one sub-target at the end of a visit and requires the UAV to depart from the sub-target to make its next visit. This translates to a feasible DPMP walk in which the UAV visits a target at one of the allowed heading angles and departs from the target at the same angle for its next visit. same angle for its next visit.

Remark: As a matter of fact, when the travel times between sub-targets are given by the Dubins-result (and hence satisfy the triangle inequality), solving the GPMP with an objective of minimizing the revisit time yields walks that have a structure similar to that of optimal DPMP walks (refer to Section 3.2. While these results are not presented in this dissertation, the idea of these proofs can be found in [32].

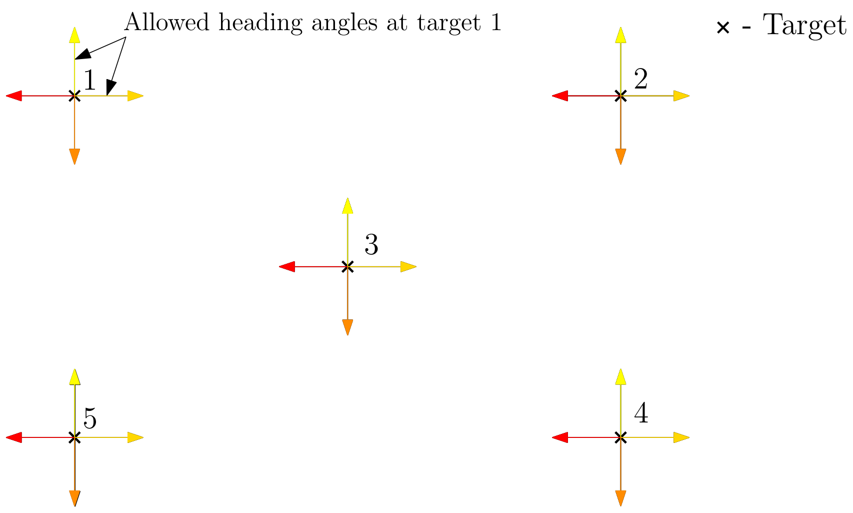


Figure 3.1: To obtained feasible walks, the UAV is allowed to visit the targets only at certain apriori chosen heading angles indicated by the arrows. In the figure, each target is allowed to be visited at one of the indicated heading angles. Once data is collected, the UAV must leave the target at the same heading angle chosen for visiting the target.

3.3.2 Lower Bounds to DPMP

Lower bounds to DPMP are obtained by solving a relaxed version of the problem. The bounds obtained by solving a simple relaxation, in which the arriving and departing angles of the UAV at a target are allowed to take any two values, are weak. Instead, we consider a relaxation in which the set of all heading angles allowed at a target (typically $[0, 2\pi]$) is partitioned into disjoint sectors (of heading angles) as shown in Figure 3.2; the union of all the sectors corresponding to a target span the set of all heading angles allowed at the target.

In the considered relaxation, the angle of arrival of the UAV at a target is allowed to be different from the angle of its departure from the target during its next visit; however, these angles are restricted to lie in the same sector. Therefore, each visit of the UAV reduces to picking exactly one sector of angles at which it can arrive at the target. Nonetheless, the departure angle of the UAV for the next visit must lie in the same sector. Therefore, this problem can be modeled as a GPMP

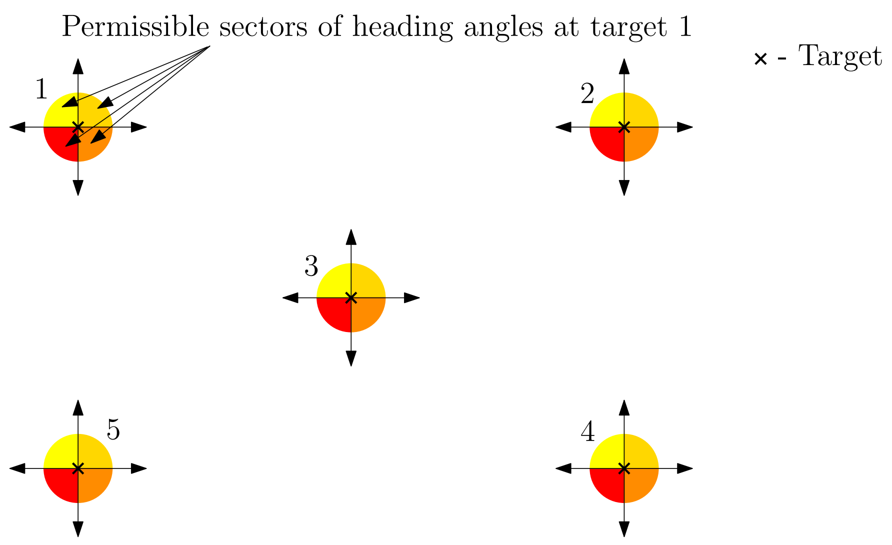


Figure 3.2: To obtain lower bounds to the DPMP, the entering and departing angles of the UAV at a target are reestricted to lie in the same sector. In the figure each target has 4 sectors of allowable heading angles.

with sub-targets representing the sectors of angles allowed at the targets.

Travel times between these sub-targets are obtained as solutions to the Dubins-Interval Problem [33]. Given two distinct targets and a sector of allowable angles at each target, the Dubins-Interval Problem provides the minimum travel time required by the UAV (with motion constraints) to travel between the targets with the angles of arrival/departure restricted to the given sectors.

Note that these travel times need not satisfy the triangle inequality. To see this, consider three three sub-targets i, j and k corresponding to different targets. The UAV traveling directly from i to j must satisfy the motion constraints and is therefore not allowed to make sharp turns along its path. However, if the UAV travels from i to j and then j to k, it is allowed to make a sharp turn after its visit to j. This is because the angles of arrival and departure are allowed to be different (though they lie in the same sector). For this reason, the results of 3.2 do not apply to this relaxation. However, lower bounds are required only for the case where $n \leq k \leq 2n-1$. So, the current procedure can be employed to obtained lower bounds for the DPMP.

3.3.3 Formulations to Solve GPMP

In this subsection, we present two different formulations to solve the GPMP. In the first formulation, the modeling of a feasible GPMP walk is similar to that of a feasible PMP walk in the MBLP formulation. That is, each visit in the walk is modeled using a separate graph, the vertices of which represent the sub-targets, and the edges represent the paths connecting the sub-targets. On the other hand, the modeling of a feasible GPMP walk in the second formulation is analogous to that of a feasible PMP walk in the ILP formulation. All the visits in the walk are modeled using a single graph allowing edges to be picked multiple times. While the connectivity of the walk in the first formulation follows from the flow constraints between successive visits, the connectivity constraints in the second formulation must be explicitly formulated and are exponential in number.

3.3.3.1 Formulation I

Sets:

$\mathcal{T}$: Set of all targets to be monitored.

$\mathcal{H}_i$: Set of all sub-targets corresponding to target $i \in \mathcal{T}$.

$\mathcal{N}$: Set of all sub-targets (corresponding to all the targets), i.e., $\mathcal{N} = \{(i,p) \mid \forall i \in \mathcal{T}, p \in \mathcal{H}_i\}$.

Data:

$c_{i,p,j,q}$: Minimum time required by the UAV to travel from sub-target $p \in \mathcal{H}_i$ of target i to sub-target $q \in \mathcal{H}_j$ of target j, $j \neq i$.

k : Number of visits in the walk ($n \leq k \leq 2n-1$ for this formulation). d: Target at which the depot is located. dp: Sub-target of target d corresponding to the depot.

Decision Variables:

$$x_v^{i,p,j,q} = \begin{cases} 1, & \text{if the } v^{th} \text{ visit in the walk is from sub-target } p \in \mathcal{H}_i \text{ of target } i \in \mathcal{T} \\ & \text{ to sub-target } q \in \mathcal{H}_j \text{ of target } j (\neq i) \in \mathcal{T}; \\ 0, & \text{otherwise.} \end{cases}$$

Objective:

$$Minimize \sum_{i\in\mathcal{T}} \sum_{p\in\mathcal{H}_i} \sum_{j(\neq i)\in\mathcal{T}} \sum_{q\in\mathcal{H}_j} \sum_{v=1}^{k} c_{i,p,j,q}\; x_v^{i,p,j,q}$$

Constraints:

The UAV starts from its depot (sub-target dp of target d) to make its first visit.

$$\sum_{j\in\mathcal{T}\backslash d} \sum_{q\in\mathcal{H}_j} x_1^{d,dp,j,q} = 1. \tag{3.1}$$

The UAV returns to its depot for its k^{th} visit.

$$\sum_{i\in\mathcal{T}\backslash d} \sum_{p\in\mathcal{H}_i} x_k^{i,p,d,dp} = 1. \tag{3.2}$$

If the $v-1^{th}$ visit of the UAV is made *to* sub-target q of target j, then its v^{th} visit must be made *from* the same sub-target (i.e., the UAV must leave the sub-target for its next visit).

$$\sum_{i\in\mathcal{T}\backslash j} \sum_{p\in\mathcal{H}_i} x_{v-1}^{i,p,j,q} = \sum_{l\in\mathcal{T}\backslash j} \sum_{r\in\mathcal{H}_l} x_v^{j,q,l,r}, \quad \forall j \in \mathcal{T},\ q \in \mathcal{H}_j,\ v \in \{2,\dots,k\}. \tag{3.3}$$

Every target must be visited at least once in a walk of k visits.

$$\sum_{v=1}^{k} \sum_{p\in\mathcal{H}_i} \sum_{j\in\mathcal{T}\backslash i} \sum_{q\in\mathcal{H}_j} x_v^{i,p,j,q} \geq 1, \quad \forall i \in \mathcal{T}. \tag{3.4}$$

Every visit has a single 'to' and a single 'from' sub-target. In other words, every visit corresponds to a single edge in the graph.

$$\sum_{i\in\mathcal{T}} \sum_{p\in\mathcal{H}_i} \sum_{j\in\mathcal{T}\backslash i} \sum_{q\in\mathcal{H}_j} x_v^{i,p,j,q} = 1, \quad \forall v \in \{1,\dots,k\}. \tag{3.5}$$

3.3.3.2 Formulation II

Decision Variables:

$x^{i,p,j,q}$: Integer variable indicating the number of times edge (i, p, j, q) appears in the walk, where $i \in \mathcal{T}, p \in \mathcal{H}_i, j(\neq i) \in \mathcal{T}, q \in \mathcal{H}_j$.

$$y^{i,p} = \begin{cases} 1, \text{if the sub-target } p \in \mathcal{H}_i \text{ of target } i \in \mathcal{T} \text{ is picked ;} \\ 0, \text{ otherwise.} \end{cases}$$

Objective:

$$Minimize \sum_{i \in \mathcal{T}} \sum_{p \in \mathcal{H}_i} \sum_{j(\neq i) \in \mathcal{T}} \sum_{q \in \mathcal{H}_j} c_{i,p,j,q} \, x^{i,p,j,q}$$

Constraints:

The sub-target corresponding to the depot is visited at least once.

$$\sum_{j \in \mathcal{T} \setminus d} \sum_{q \in \mathcal{H}_j} x^{d,dp,j,q} \geq 1. \tag{3.6}$$

The number of times the UAV visits a sub-target must be equal to the number of times the UAV leaves the sub-target.

$$\sum_{i \in \mathcal{T} \setminus j} \sum_{p \in \mathcal{H}_i} x^{i,p,j,q} = \sum_{l \in \mathcal{T} \setminus j} \sum_{r \in \mathcal{H}_l} x^{j,q,l,r}, \quad \forall j \in \mathcal{T}, \ q \in \mathcal{H}_j. \tag{3.7}$$

Every target is visited at least once.

$$\sum_{p \in \mathcal{H}_i} \sum_{j \in \mathcal{T} \setminus i} \sum_{q \in \mathcal{H}_j} x^{i,p,j,q} \geq 1, \quad \forall i \in \mathcal{T}. \tag{3.8}$$

The total number of edges in the graph is equal to the number of visits made by the UAV before every recharging, which is equal to k.

$$\sum_{i\in\mathcal{T}}\sum_{p\in\mathcal{H}_i}\sum_{j\in\mathcal{T}\setminus i}\sum_{q\in\mathcal{H}_j} x^{i,p,j,q} = k. \tag{3.9}$$

The purpose of next two constraints is to find out if a sub-target is visited; this is done through a binary variable associated with each sub-target. This information is required for modeling connectivity constraints (3.12), which ensure that the resulting walk is a connected graph.
If the number of outgoing edges from a sub-target p of target i is zero, then the variable $y^{i,p}$ takes the value zero, suggesting that the sub-target is not visited.

$$\sum_{j\in\mathcal{T}\setminus i}\sum_{q\in\mathcal{H}_j} x^{i,p,j,q} \geq y^{i,p}, \quad \forall i \in \mathcal{T}, p \in \mathcal{H}_i. \tag{3.10}$$

If the number of outgoing edges from sub-target p of target i is at least one, then the sub-target is picked, and the variable $y^{i,p}$ takes the value 1. The constant M indicates the maximum number of outgoing edges possible from a sub-target, which is the minimum of $k-n+1$ and $k/2$.

$$\sum_{j\in\mathcal{T}\setminus i}\sum_{q\in\mathcal{H}_j} x^{i,p,j,q} \leq M\, y^{i,p}, \quad \forall i \in \mathcal{T}, p \in \mathcal{H}_i. \tag{3.11}$$

With the help of the above two constraints, the connectivity constraints can be modeled as the following:

$$\sum_{(i,p)\in\mathcal{S}}\ \sum_{(j,q)\in\mathcal{N}\setminus S,\, j\neq i} x^{i,p,j,q} \geq y^{i,p} + y^{j,q} - 1, \quad \forall \mathcal{S} \subset \mathcal{N},\ 2 \leq |\mathcal{S}| \leq k-2. \tag{3.12}$$

The number of constraints given by (3.12) is exponential.

3.3.4 Numerical Simulations

In this section, we evaluate the quality of upper and lower bounds provided by the GPMP through numerical simulations performed on a set of instances with the help of the above two formulations. For each instance, upper and lower bounds to DPMP were obtained by allowing discrete heading angles and sectors of heading angles at the targets respectively. The number of the discrete heading angles or sectors at each target were gradually increased to observe the convergence of these bounds.

The summary of the simulations are illustrated through a sample instance with 5 targets. The number of discretizations at each target was increased from 4 to 32, and for each discretization, bounds were computed by varying the number of visits in the walk from 5 to 9. Figure 3.5 shows the plot of these upper (in blue) and lower bounds (in red) against the number of visits for different number of discretizations. From the plot, it can clearly be seen that the upper bounds decrease and the lower bounds increase with the number of discretizations. Similarly, bounds were computed for 24 other instances with 5 targets, and the average percentage gap between the upper and lower bounds were plotted against the number of visits in the walk for different number of discretizations, as shown in Figure 3.5. From the figure, it is evident that after a certain number of discretizations, the percentage gap between the bounds are very less (within 1% here). This suggests that the feasible DPMP walks obtained by solving the GPMP are near-optimal. Figure 3.3 shows a feasible walk obtained by solving the GPMP. For the rest of this chapter, we focus on the GPMP.

Numerical simulations suggest that the formulations pressented in 3.3.3 for solving the GPMP do not scale well for larger instances. The computation time required for solving the problem to optimality increases rapidly with the increase in the number of targets and discretizations. Even with the number of discretizations as small as 8, several instances with 15 targets were not solvable within a day [*].

Unlike the trend observed in Chapter 2, here, the computational performance of even formulation II, which was based on the ideas from the ILP formulation for PMP, was poor. In fact, for most of the instances, formulation I, which was based on the MBLP formulation for PMP, was observed to perform better. This can be attributed to the big-M involved in formulation II, which leads to weak LP-relaxations. So, there is a need to develop algorithms to efficiently compute feasible solutions to the GPMP.

[*] All the simulations were performed on a Mac Pro (8-Core Intel Xeon E5 processor @3 GHz, 32 GB RAM) with the help of the Gurobi solver [34]

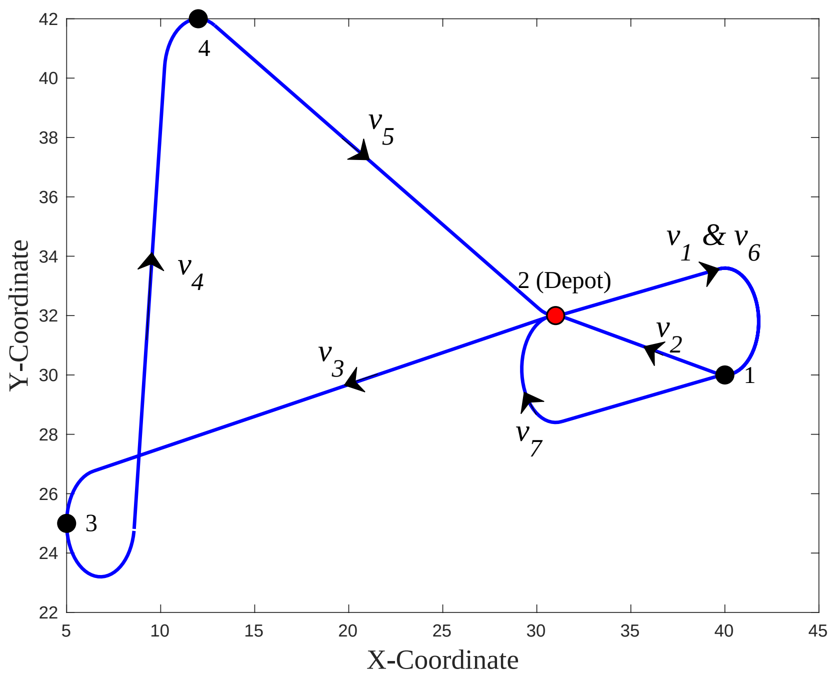


Figure 3.3: A feasible DPMP walk with 7 visits to 4 targets obtained by solving the GPMP

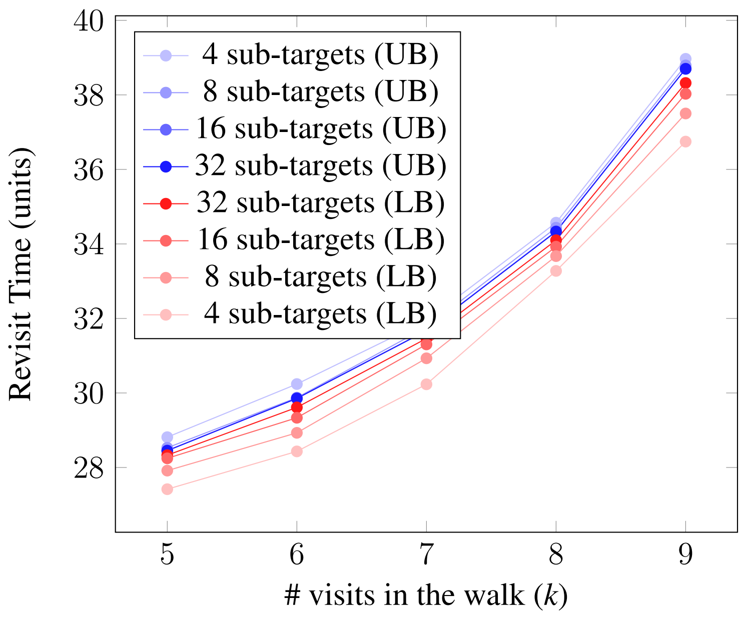


Figure 3.4: Plot showing the upper bounds (UB) and lower bounds (LB) for a 5-target instance of the DPMP. The bounds get closer with the increase in the number of sub-targets.

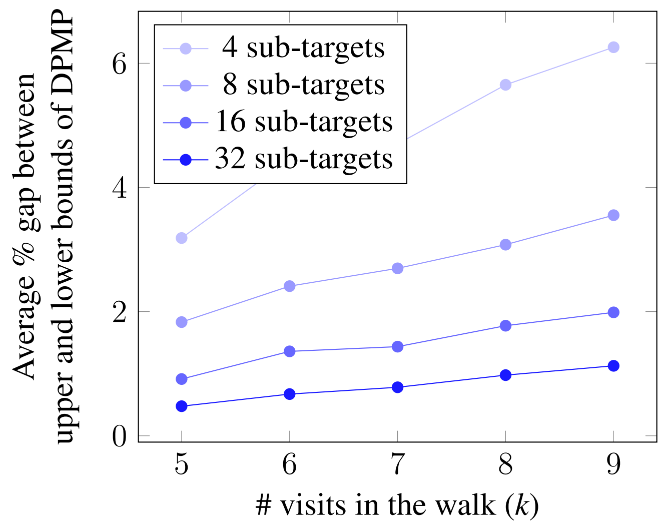


Figure 3.5: Plot illustrating the decrease in the average percentage gap between the upper and lower bounds for DPMP with increase in the number of sub-targets. The plot corresponds to instances with 5 targets, and the bounds are obtained by solving GPMPs.

3.4 Heuristics to Solve GPMP

Here, we present two algorithms to quickly compute feasible solutions to the GPMP. Both the algorithms are based on independently solving the sequencing and the (heading angle) selection problem.

3.4.1 4.56-Approximation Algorithm (AH)

In the first algorithm, an initial sequence of visits to targets is obtained via the 1.5-approximation algorithm for solving the PMP (using Euclidean distances between the targets), which was presented in Section 2.8. Once a sequence of visits is obtained, a set of heading angles corresponding to the sequence is chosen such that the (travel) time required for the UAV to traverse the route specified by the combination of the sequence and heading angles is minimized. The steps of this algorithm can be implemented in polynomial time, and for convenience, this algorithm is referred to as AH.

To provide a guarantee on the quality of solutions obtained using this algorithm, we make a reasonable assumption that the targets are not too close, i.e., the Euclidean distance between every pair of targets is at least twice the minimum turning radius. Under this practical assumption, the Dubins distances (travel time) between any two targets with any combination of arrival and departure angles at the targets is at most $\pi + 1 - \tan^{-1}(2)$ times the Euclidean distance (travel time) between the targets [35]. Therefore, the proposed algorithm has an approximation rato of $\frac{3}{2}(\pi + 1 - \tan^{-1}(2)) \approx 4.56$.

3.4.2 Optimal PMP + Heading angles (OH)

In the second algorithm, an optimal PMP walk with the desired number of visits is used as an elementary sequence of visits. Even though optimal PMP walks are not guaranteed a polynomial computation time, for all practical purposes, they can be computed swiftly using the ILP formulation presented in 2.5. For the obtained sequence of visits, similar to AH, a set of heading angles that provide the least travel time is selected. For convenience, this algorithm is referred to as OH.

3.5 Comparision of the Heuristics

In this section, we compare the above two algorithms against the optimal GPMP values that are obtained by solving the above formulations. For the comparision, only the instances for which optimal GPMP values were computable upto at least 8 discretizations of heading angles were utilized. For each instance, both AH and OH were implemented in Julia [36] with the aid of the NetworkX package [37] for 4 and 8 heading angles and by varying the number of visits in the walk from n to $2n - 1$. The minimum turn radius was varied from 3% to 30% of the maximum Euclidean distance between the targets. For all instances with the same number of targets and the same percentage of minimum turn radius, the average percentage gap of the values obtained from AH and OH with respect to the optimal GPMP values were computed for 4 and 8 discretizations, and plotted against the number of visits in the walk. Sample plots with 3% turn radius are shown in Figures 3.6, 3.7 and 3.8 for 5, 8 and 15 targets respectively.

The plots suggest that when the minimum turn radius is low, the values provided by algorithms that are based on independently solving the sequencing and selection problems are very close to optimal GPMP values. For large number of discretizations, the average percentage gap with OH was observed to be within 5 % of optimal GPMP solutions. The reason for this closeness is that when the minimum turn radius is low, the travel times between the targets are very close to the Euclidean travel times, and optimal sequences obtained by solving the PMP are also optimal for GPMP.

However, when the minimum turn radius is high, the same trend is not expected to repeat. Sample plot for 5-target instances with a turn radius of 30 % of the maximum travel times between the targets is shown in Figure 3.9. From the plot it can be seen that the average percentage gap was around 20 to 30 % away from optimal GPMP solutions. So, in practice, when the minimum turn radius is high, it is not enough to independently solve the sequencing and selection problems once. However, one can explore algorithms that are based on iteratively solving the selection and sequencing problems.

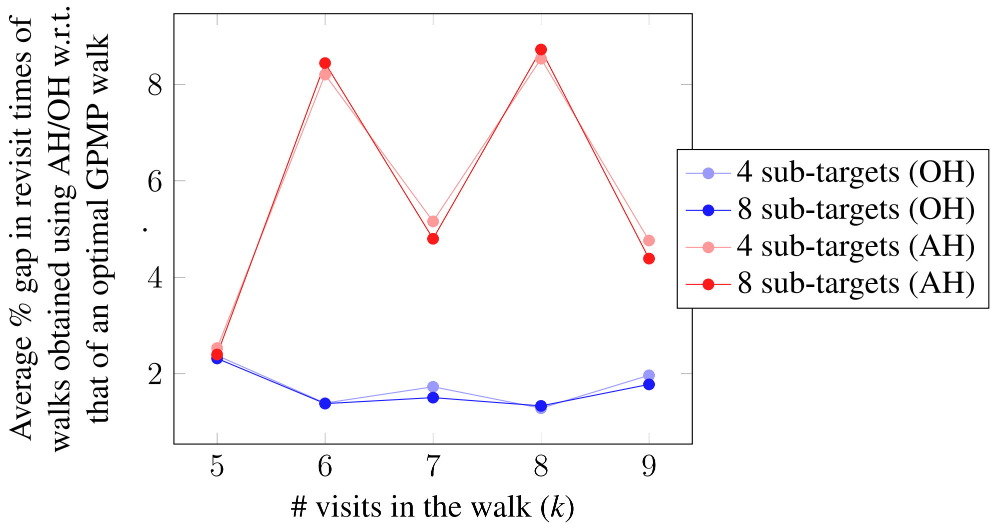


Figure 3.6: Plot showing the average % gap in revisit times of walks obtained using AH and OH algorithms with respect to that of an optimal GPMP walk, for instances with 5 targets 3 % minimum turn radius

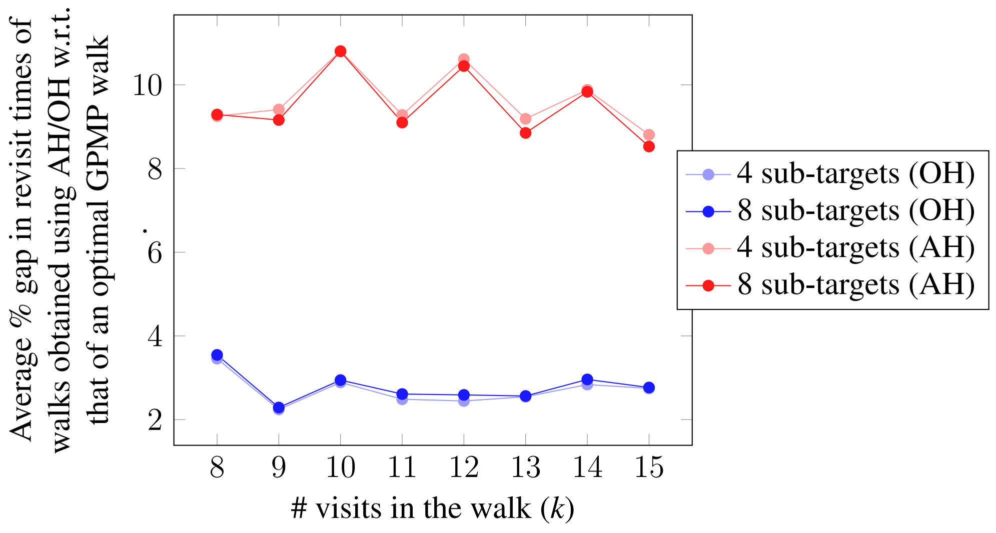


Figure 3.7: Plot showing the average % gap in revisit times of walks obtained using AH and OH algorithms with respect to that of an optimal GPMP walk, for instances with 8 targets 3 % minimum turn radius

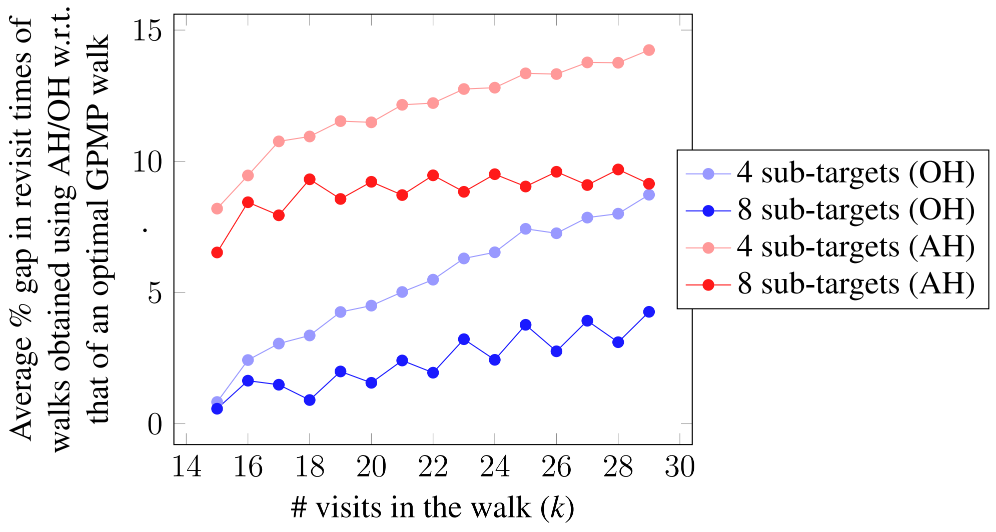


Figure 3.8: Plot showing the average % gap in revisit times of walks obtained using AH and OH algorithms with respect to that of an optimal GPMP walk, for instances with 15 targets and 3 % minimum turn radius

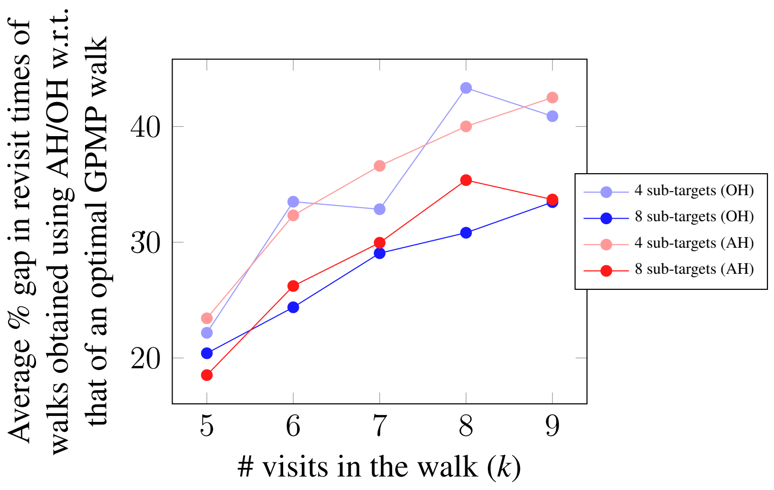


Figure 3.9: Plot showing the average % gap in revisit times of walks obtained using AH and OH algorithms with respect to that of an optimal GPMP walk, for instances with 5 targets and 30 % minimum turn radius

4. PERSISTENT MONITORING PROBLEM WITH A SEPARATE DEPOT

In many applications, the targets are not accessible for humans due to either the safety hazards involved or the remoteness of the targets. In such cases, depot(s) of the monitoring UAV(s) must be stationed at location(s) different from that of the targets. Here, we consider such a case where a set of equally weighted targets are monitored using a single UAV that has no curvature constraints, and the depot of the UAV is not located at the targets.

In this case, the UAV starts the monitoring mission from the depot with a fully charged battery. Suppose it is given that the UAV makes k visits to the given set of targets. Then, the vehicle makes data collection visits to different targets such that every target is visited at least once by the end of $k-1$ visits. The k^{th} visit is then made to the depot for recharging. As soon as the UAV is recharged at the depot, this sequence of k visits is repeated. Similar to the above chapters, this sequence is referred to as a walk with k visits. Note that here a depot can be visited only at the end of every k visits, and cannot be visited in between. To differentiate routes of this case from that of previous cases, we denote a walk with k visits by $\mathcal{WD}(k)$.

The revisit time of $\mathcal{WD}(k)$ is obtained as the maximum t.r.t (target revisit time) over all the targets in the walk (it does not include the t.r.t of the depot), and it is denoted by $\mathcal{RD}(\mathcal{WD}(k))$. Unlike the previous chapters, the problem considered here accounts for the time taken to recharge the UAV at the depot. The optimal route planning problem considered in this chapter is referred to as PMP_d and can be stated formally as the following:

Given n targets and that the UAV monitoring them makes k visits, $k \geq n+1$ (one extra visit for the depot), to the targets before recharging at the depot, find a walk with k visits that provides the least revisit time. Such a walk is referred to as the optimal walk and denoted by $\mathcal{WD}^*(k)$, and the minimum revisit time is referred to as the optimal revisit time and denoted by $\mathcal{RD}^*(k)$. Suppose that the travel time from any target $i \in \mathcal{T}$ to the depot is denoted by $c(i,d)$ and that from the depot to target i is given by $c(d,i)$. Then, rc, the time taken to recharge the UAV at the depot, can be included into the problem by simply replacing $\bar{c}(i,d)$ by $c(i,d) + \frac{rc}{2}$ and $\bar{c}(i,d)$ by $c(i,d) + \frac{rc}{2}$, for all $i \in \mathcal{T}$.

By solving this problem, the master problem of finding an optimal k and its corresponding optimal walk can also be solved easily as discussed in Section 2.9. Besides, by considering the depot as a target with a very low priority, solutions to this problem can also be considered as solutions for the case with unequally weighted targets. This is because, the depot is visited only once in the walk, and its t.r.t, even if considered, is negligible due to its low priority/weight.

The focus of this chapter is to examine the structure of optimal solutions to the PMP_d, as done in Chapter 2 for PMP. We give a first glimpse of this structure through plots of the optimal revisit times versus the number of visits in the walk. Then, we compare these plots with that of the PMP considered in Chapter 2, and present conjectures on the structure of optimal PMP_d solutions. These conjectures are supported by numerical simulations performed over 350 instances.

4.1 Optimal Revisit Time

PMP_d was implemented using a formulation similar to the MBLP for the PMP, with the help of Gurobi solver on several instances. For each instance, the problem was solved for different number of visits starting from $k = n + 1$ upto a value for which an optimal solution was computable in a reasonable time. For these instances, the value of optimal revisit time was plotted against the number of visits in the walk. Based on their asymptotic trends, the plots can be grouped into three categories: Unimodal, Bimodal and Trimodal. Sample plots for each of this category are shown in Figures 4.1, 4.2 & 4.3 for instances with 5 targets and a depot. Before discussing the criteria determining the asymptotic structure of the plot, we present a comparison of the properties of the optimal revisit time of PMP_d with that of the PMP. Here, we refer to an optimal TSP tour over the targets and the depot by TSP_d, and its cost by TSP_d^*.

4.1.1 Similarities with PMP

The optimal revisit time for PMP_d has the following similarities with that of PMP:

- for any k such that $k \geq n + 1$, the optimal revisit time is lower bounded by TSP_d^*, i.e.,

$\mathcal{RD}^*(k) \geq TSP_d^*$. Equality holds for $k = pn + 1$, where p is a positive integer;

- for $n + 1 \leq k \leq 2n$, $\mathcal{RD}^*(k)$ is a monotonically increasing function of k;
- beyond certain value of k, $\mathcal{RD}^*(k)$ is a periodic function of k with a period n.

Note however that the threshold after which the optimal revisit time is periodic is different from that of PMP.

4.1.2 Dissimilarities with PMP

Unlike that in PMP, the structure of optimal revisit time in PMP_d is not independent of the location of the targets. Firstly, $\mathcal{RD}^*(k)$ can take more than $n+1$ distinct values. Secondly, $\mathcal{RD}^*(k)$ is not necessarily bimodal for large values of k. Its asymptotic properties are dependent on the location of the depot relative to that of the targets. Simulations suggest that the optimal revisit time obtained by solving the PMP over only the targets, $\mathcal{R}^*(.)$, aids in determining the asymptotic structure of the optimal revisit time for PMP_d over the targets and the depot. In particular, the value of $\mathcal{R}^*(t+1)$ relative to that of $\mathcal{RD}^*(n+1)$ and $\mathcal{RD}^*(n+2)$ determine the number of distinct values taken by $\mathcal{RD}^*(k)$ asymptotically.

Note that $\mathcal{R}^*(n + 1)$ is the optimal value of a PMP walk with $n + 1$ visits to n targets, that is, one target is visited twice, while the rest are visited exactly once. On the other hand, $\mathcal{RD}^*(k)$ is the optimal value of a PMP_d-walk with $n + 1$ visits to n targets and a depot. Here, every target, in addition to the depot, is visited exactly once. The number of distinct values $\mathcal{RD}^*(k)$ takes asymptotically can be determined as the following:

1. $\mathcal{R}^*(n + 1) \leq \mathcal{RD}^*(n + 1)$: Unimodal (one value);
2. $\mathcal{RD}^*(n + 1) \leq \mathcal{R}^*(n + 1) \leq \mathcal{RD}^*(n + 2)$: Bimodal (two values);
3. $\mathcal{RD}^*(n + 2) \leq \mathcal{R}^*(n + 1)$: Trimodal (three values).

Besides, the minimum number of visits at which the periodic trend of $\mathcal{RD}^*(k)$ beings was observed to be different for different instances. However, a general number that works for all the instances is $n(n+1)+1$ visits, i.e., for $k \geq n(n+1)+1$, $\mathcal{RD}^*(k)$ was observed to be periodic irrespective of its precise asymptotic structure.

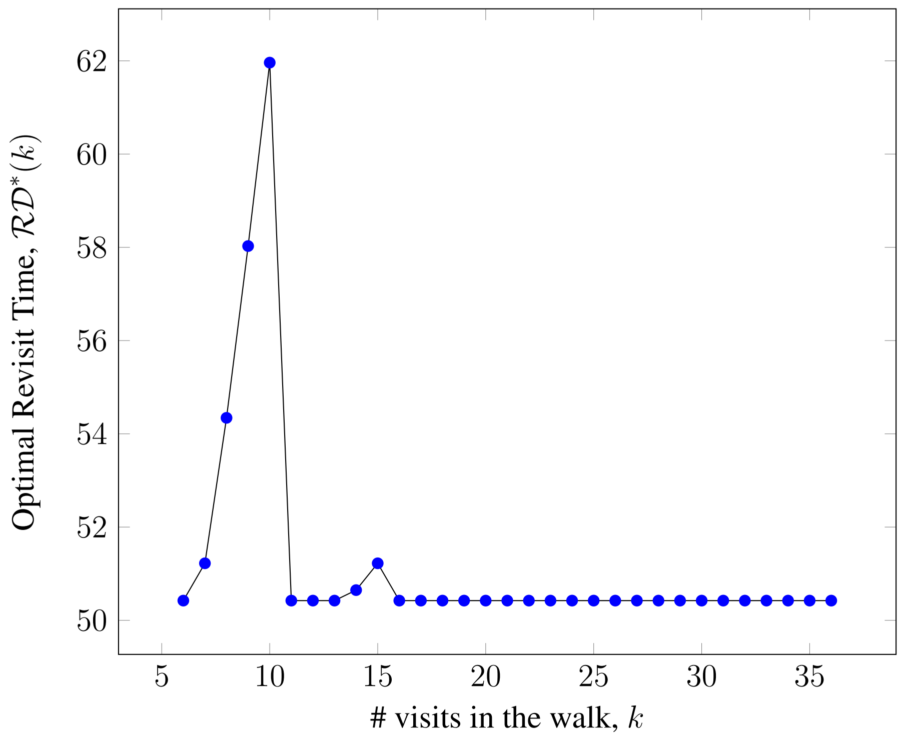


Figure 4.1: Figure illustrating the plot of the optimal revisit time against the number of visits in the walk for an instance with 5 targets and a depot. For this instance, optimal revisit time takes exactly one value asymptotically

Analogous to the utility of optimal revisit time of PMP in determining the structure of that of PMP_d, optimal PMP walks are useful in discerning the structure of optimal PMP_d solutions.

4.2 Structure of Optimal PMP_d walks

Similar to the construction of optimal PMP walks presented in Chapter 2, the construction of optimal PMP_d walks involves operations such as concatenation and cyclic permutation on certain elementary walks. However, the elementary walks used for the construction are not just restricted to optimal PMP_d walks with small number of visits, but also include corresponding optimal PMP walks with small number of visits. Before discussing the construction procedure for the asymptotic

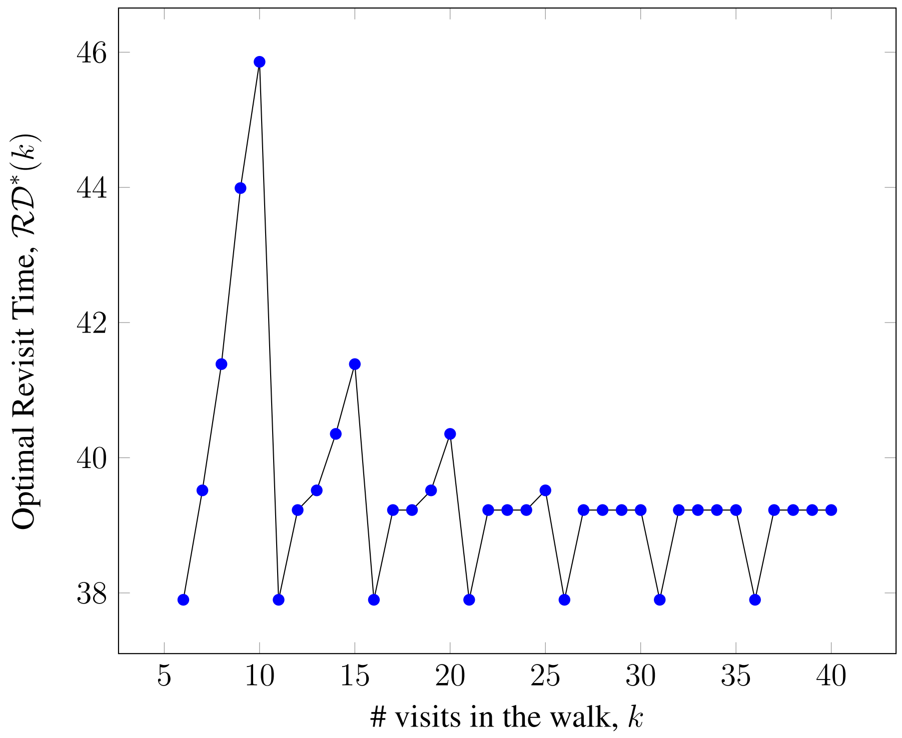


Figure 4.2: Figure illustrating the plot of the optimal revisit time against the number of visits in the walk for an instance with 5 targets and a depot. For this instance, optimal revisit time takes two different values asymptotically

case, here we present the elementary walks required for the same.

4.2.1 Elementary Walks

Three main elements required for the construction of optimal PMP_d walks for the asymptotic case are: $\mathcal{W}^*(n+1)$, an optimal PMP walk with $n+1$ visits to n targets; $\mathcal{WD}^*(n+1)$ or (TSP_d), an optimal PMP_d walk with $n+1$ visits to n targets and a depot, and $\mathcal{WD}^*(n+2)$, an optimal PMP_d walk with $n+2$ visits. These elements can be shortcut in different ways to form walks with smaller number of visits. The elementary walks and their shortcut versions can be concatenated in certain combinations to obtain optimal PMP_d walks with large number of visits. Next, the shortcut walks required for the construction will be presented, followed by the construction procedure.

First, the shortcut walks obtained from $\mathcal{WD}^*(n+2)$ will be presented. In $\mathcal{WD}^*(n+2)$, exactly one target is visited twice. A walk obtained by shortcutting a visit to this target is denoted by

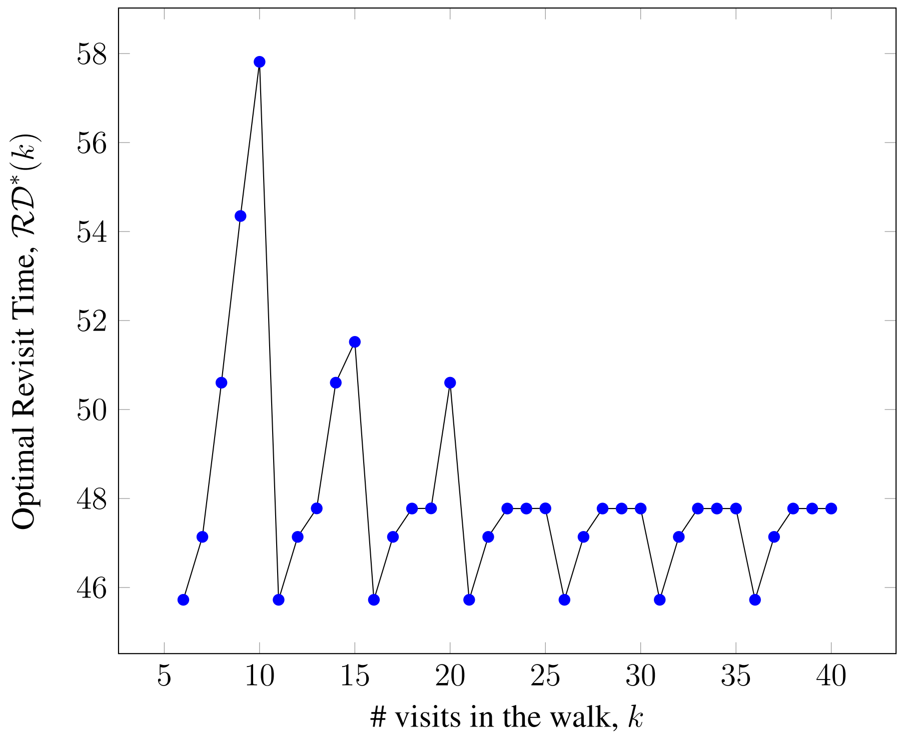


Figure 4.3: Figure illustrating the plot of the optimal revisit time against the number of visits in the walk for an instance with 5 targets and a depot. For this instance, optimal revisit time takes three values asymptotically

$\mathcal{SWD}$. Unlike the convention considered in Chapter 2, here, even though the depot is visited exactly once, we consider shortcutting the visit to the depot. Such a walk obtained by shortcutting (removing) the visit to the depot from $\mathcal{WD}^*(n+2)$ is denoted by $\mathcal{WSD}$. Note however that the depot can only be shortcut if the targets visited before and after the depot in the walk are different. So, it is not always possible to construct $\mathcal{WSD}$ from $\mathcal{WD}^*(n+2)$. We also consider a shortcut walk in which both the depot and the re-visit to a target are shortcut. This walk is denoted by $\mathcal{SWSD}$. Note that unlike $\mathcal{WSD}$, it is always possible to construct $SWSD$ by first shortcutting the repeated visit and then shortcutting the visit to the depot. $\mathcal{WD}^*(n+2)$ and its corresponding modified walks are depicted in Figure 4.4 for a sample instance.

Next, we consider secondary walks obtained from $\mathcal{W}^*(n+1)$. This walk has a repeated visit to exactly one target. A walk obtained by shortcutting a repeated visit to the target is denoted by $\mathcal{SW}$. In the construction procedure to be discussed, one often needs a walk with $n+1$ visits with the least travel time such that n visits are made to n targets and one visit is made to the depot. While TSP_d serves this purpose, in certain cases, one requires a walk with the same sequence of visits as that in $\mathcal{W}^*(n+1)$ except for its revisit. For this reason, we consider a walk that is obtained by adding a depot to $\mathcal{SW}$. This walk is denoted by $\mathcal{SWAD}$, and it is obtained by inserting a visit to the depot in $\mathcal{SW}$ such that the increase in this travel time is the minimum. This can be performed in polynomial time. Simulations suggest that for the cases in which the use of $\mathcal{SWAD}$ is proposed, its cost is no more than that of TSP_d. The walk $\mathcal{W}^*(n+1)$ and its corresponding modified walks are depicted in Figure 4.5 for a sample instance.

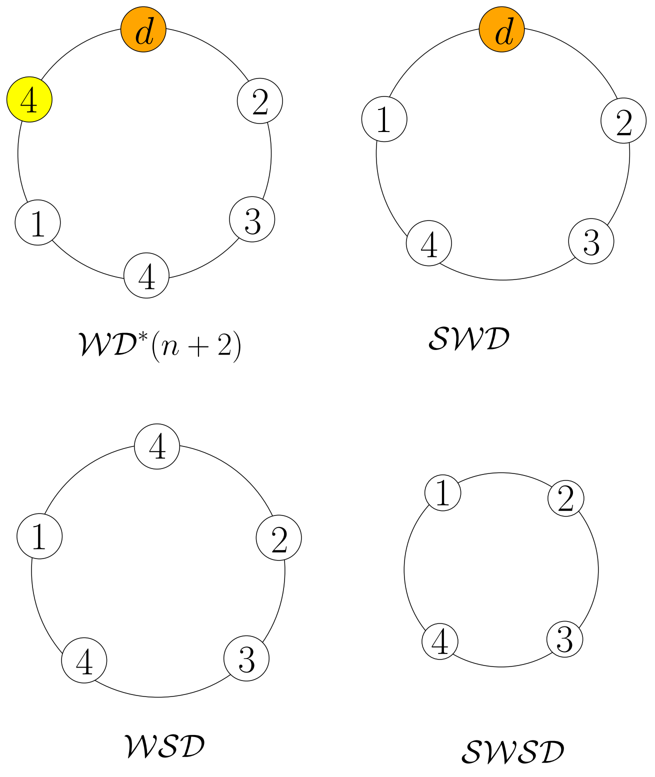


Figure 4.4: Elementary walks obtained by modifying $\mathcal{WD}^*(n+2)$, which are useful in the construction of optimal solutions for PMP_d in the asymptotic case

Lastly, we consider a shortcut walk of TSP_d that is obtained by shortcutting the visit to the depot; we denote this shortcut walk by TSP_{short}. With these components, optimal PMP_d walks for the asymptotic case can be constructed according to the procedure presented in the next subsection.

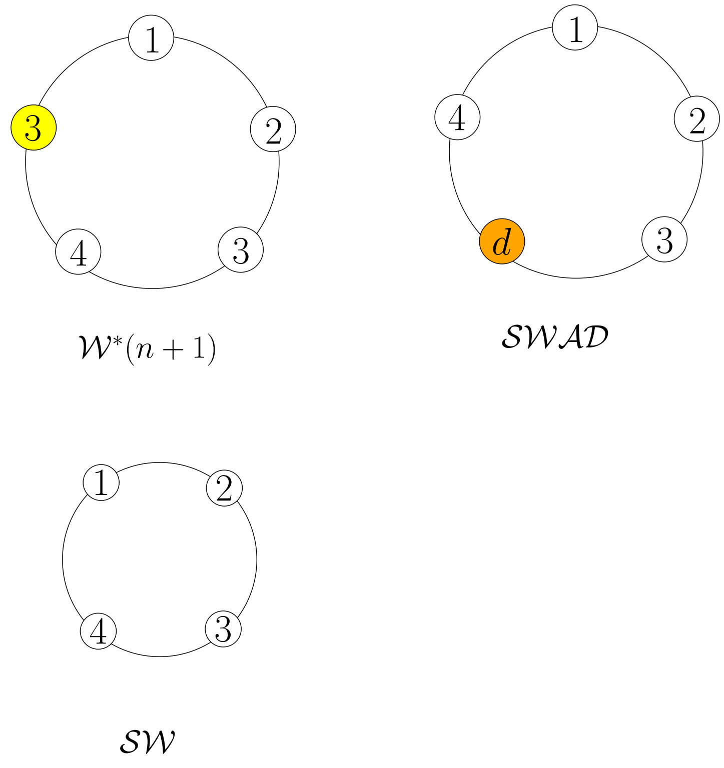


Figure 4.5: Elementary walks obtained by modifying $\mathcal{W}^*(n+1)$, which are useful in the construction of optimal solutions for PMP_d in the asymptotic case

4.2.2 Conjectures on the Construction Procedure

Simulations indicate that the construction of optimal solutions for the asymptotic case can be categorized into 4 cases, and the categorization is based on questions such as: whether $\mathcal{R}^*(n+1) \leq \mathcal{RD}^*(n+1)$, whether one can shortcut the visit to the depot from $\mathcal{WD}^*(n+2)$ to form a feasible walk, and whether $\mathcal{R}^*(n+1) \leq \mathcal{RD}^*(n+2)$.

In all the four cases, when the number of visits k is one more than an (positive) integer multiple of n, an optimal walk can be constructed by concatenating exactly one copy of TSP_d with the required number of copies of TSP_{short} such that the total number of visits is k. The construction procedure for any other k will be discussed in the next few paragraphs.

The first case corresponds to the scenario in which $\mathcal{R}^*(n+1) \leq \mathcal{RD}^*(n+1)$. In this case, the optimal revisit time is asymptotically unimodal, and it takes the value TSP_d^*. An optimal walk when the number of visits is one more than an integer multiple of n follows from the above paragraph. For other values of k, the construction procedure is identical to that of the upcoming cases.

The second case arises when $\mathcal{R}^*(n+1) > \mathcal{RD}^*(n+1)$ and the visit to the depot in $\mathcal{WD}^*(n+2)$ can be shortcut to form a feasible walk. That is, the targets that are visited before and after the visit to the depot are different. In this case, the optimal revisit time is asymptotically bimodal. When $k \neq pn+1$ for some $p \in Z_+$, an optimal walk can be constructed by concatenating exactly one copy of $\mathcal{SWD}$, with the required number of copies of $\mathcal{SWSD}$ and $\mathcal{WSD}$ to complete k visits. The sequence $\mathcal{SWD}$ covers the visit to the depot and has the same cost as TSP_d.

The next case arises when both $\mathcal{R}^*(n+1) > \mathcal{RD}^*(n+1)$ and the visit to the depot in $\mathcal{WD}^*(n+2)$ cannot be shortcut to form a feasible walk. That is, the targets that are visited before and after the visit to the depot are the same. In such a situation, one needs to verify if $\mathcal{R}^*(n+1) \leq \mathcal{RD}^*(n+2)$. If it is true, the optimal revisit time is asymptotically bimodal. When $k \neq pn+1$ for some $p \in Z_+$, an optimal walk can be constructed by concatenating one copy of $\mathcal{SWAD}$ with the required number of copies of $\mathcal{W}^*(n+1)$ and $\mathcal{SW}$.

The final case arises when $\mathcal{RD}^*(n+2) \geq \mathcal{R}^*(n+1) > \mathcal{RD}^*(n+1)$ and the depot in $\mathcal{WD}^*(n+2)$ cannot be shortcut to form a feasible walk. In this case, the optimal revisit time is asymptotically trimodal; that is, it takes three different values. When k is of the form $pn+1$ for some $p \in Z_+$, an optimal walk can be constructed as discussed at the beginning of this subsection. Otherwise, depending on the value of k, an optimal walk can be constructed either by concatenating one copy of $\mathcal{WD}^*(n+2)$ with the required number of copies of $\mathcal{SWSD}$, or by concatenating one copy of $\mathcal{SWAD}$ with the required number of copies of $\mathcal{W}^*(n+1)$ and $\mathcal{SW}$.

Note that in all the cases, the starting targets in all the walks being concatenated must be the same. In order to ensure this, one must start with an appropriate cyclic permutation of an elementary walk and then construct its corresponding secondary walks. Once a walk with the required number of visits is constructed, it can be cyclically permuted such that the concatenated walk starts and ends at the depot. This construction procedure is summarized in the following flowchart (see Figure 4.6) .

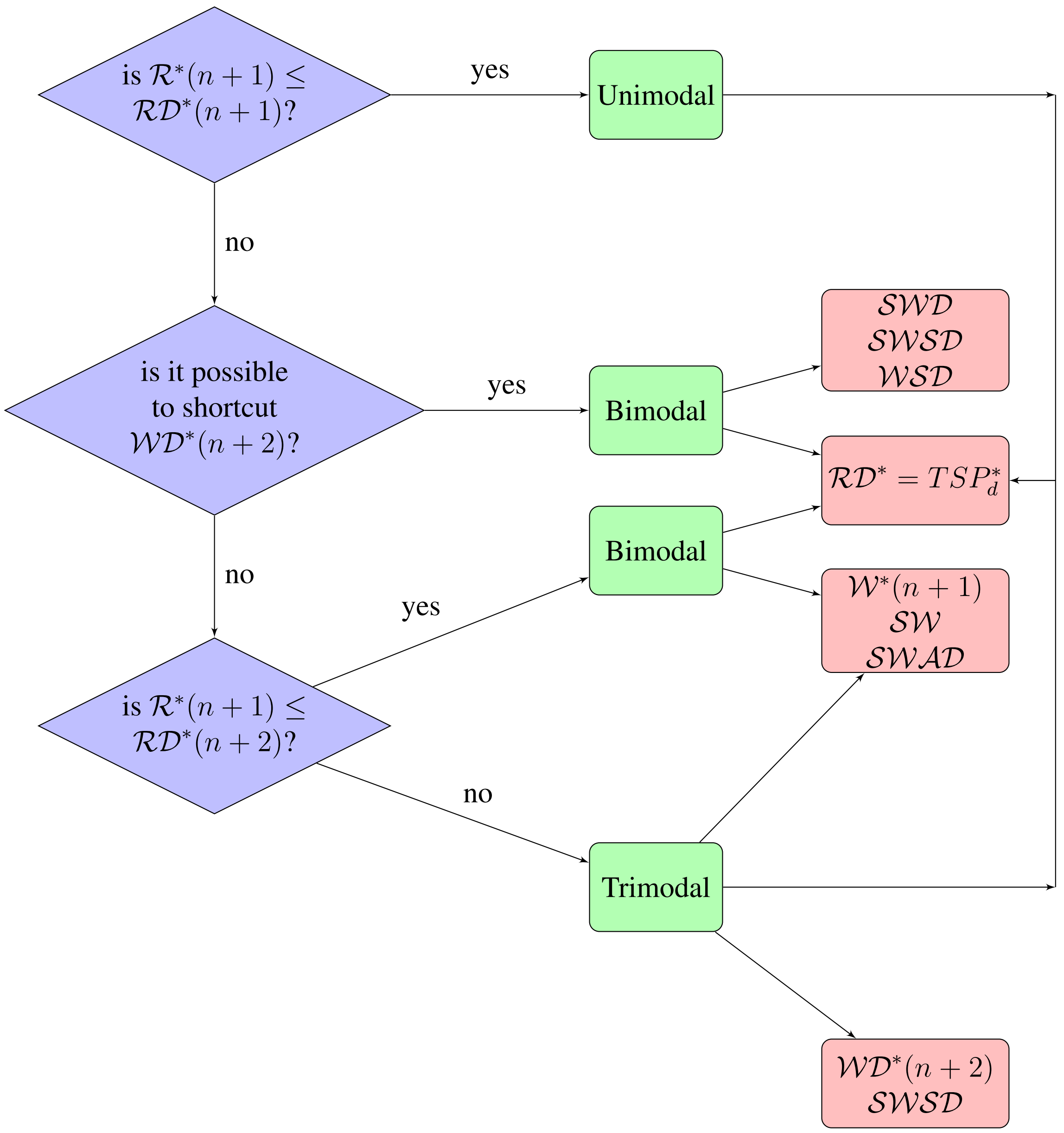


Figure 4.6: Flowchart explaining the construction of optimal walks for PMP_d for large number of visits.

5. MULTI-UAV PERSISTENT MONITORING PROBLEM

In this chapter, we consider the case in which *multiple UAVs* are employed for monitoring the targets. Typically, multiple UAVs are used when either the minimum revisit time attainable with the help of a single UAV is unsatisfactory or the targets necessitate the use of UAVs with different capabilities. While multiple UAVs facilitate an efficient monitoring of targets, they render the route planning problem intractable. Firstly, modeling target revisit times is difficult as one needs to keep track of visits made to a target by different UAVs. Secondly, when multiple UAVs share the same depot, their visits to the depot for recharging must be scheduled such that they do not coincide.

We restrict the scope of this chapter to the case in which the UAVs have separate depots that are setup at the targets, and all the targets have equal weights. In this case, each UAV is required to visit at least two targets, but every target is allowed to be visited by a single UAV. The extension of this work to the case with depots installed at locations different from that of the targets is similar to the extension for the single UAV case considered in 4 and is omitted here. Even for this special case, the problem remains computationally challenging due to the coupling between different components of the solution; a target assignment determines the number of visits allowed for a UAV to its assigned set of targets, which in turn determines an optimal sequence of visits for the UAV. Once a target assignment is available, its corresponding optimal solutions can be computed in a split second using the methods presented in Chapter 2; however, finding an optimal target assignment is non-trivial.

Here, we make use of the techniques presented in earlier chapters to devise an approach to obtain feasible walks for multiple UAVs. This approach is built on two elementary ideas. Firstly, the number of visits allowed for each UAV to its assigned set of targets is not known; however, one can estimate the total number of visits a single UAV can make to the set of *all* targets with a charge capacity that is equivalent to that of all the UAVs combined. Suppose this estimate is k visits. Then, the sum of the number of visits allowed for each UAV to its assigned set of targets (before recharging) is set to k. Secondly, as observed in Chapter 2, constraints modeling the revisit time add to the computational difficulty of the problem. To circumvent this, the number of visits allowed for each UAV to its set of targets is restricted; if a UAV is assigned n_i targets, then the number of visits the UAV is allowed to these targets (before recharging) is at most $2n_i - 1$. With this restriction, the objective reduces to simply minimizing the maximum of the travel times of all the UAVs. Note that this modified problem is a generalization of the min-max TSP, and is referred to as the Multiple Vehicle Persistent Monitoring Problem (MV-PMP). A formulation to obtained optimal solutions to MV-PMP is presented below. The formulation builds on the ILP formulation presented in Chapter 2 for the single vehicle case.

5.1 Formulation

In this formulation, routes of the UAVs are modeled on a complete graph G, the vertices of which represent the targets, and edges represent the paths connecting the targets. This formulation is a generalization of the commonly used formulations for the min-max TSP [38].

Sets:

$\mathcal{T}$: Set of all targets to be monitored.

$\mathcal{E}$: Set of all paths connecting the targets, i.e., $\mathcal{E} = \{(i,j) : i \in \mathcal{T},\ j \in \mathcal{T}\}$.

$\mathcal{V}$: Set of all vehicles, i.e., $\mathcal{V} = \{1, 2, \cdots, m\}$. Note that $|\mathcal{V}| \leq |\mathcal{T}|$.

$\delta^-(S)$: Set of all incoming edges to a set S, $S \subseteq \mathcal{T}$, of vertices,
i.e., $\delta^-(S) = \{(i,j) \in \mathcal{E} : i \notin S, j \in S\}$.

$\delta^+(S)$: Set of all outgoing edges to a set S, $S \subseteq \mathcal{T}$, of vertices,
i.e., $\delta^+(S) = \{(i,j) \in \mathcal{E} : i \in S, j \notin S\}$.

Data:

$c(i,j)$: Time taken by a UAV to travel between two different targets $i \in \mathcal{T}$ and $j \in \mathcal{T}$. All the UAVs are assumed to be homogeneous and have same travel times.

Decision Variables:

$x_{e,v}$: Non-negative integer variable indicating the number of times edge $e \in \mathcal{E}$ appears in the walk of vehicle $v \in \mathcal{V}$.

k_v: Non-negative integer variable indicating the number of visits after which UAV $v \in \mathcal{V}$ is recharged.

$$a_{i,v} = \begin{cases} 1, \text{if target } i \in \mathcal{T} \text{ is assigned to UAV } v \in \mathcal{V}; \\ 0, \text{ otherwise.} \end{cases} .$$

Objective:

The objective is to minimize the maximum travel time over that of all the UAVs.

$$\text{Minimize} \max_{v \in \mathcal{V}} \sum_{e \in \mathcal{E}} c_e * x_{e,v}.$$

Constraints:

The sum of the number of visits made by each UAV to its assigned set of targets before it is recharged at its depot is equal to k, which is the number of visits a single UAV with a charge capacity equal to that of all the UAVs combined can make to the set of all targets.

$$\sum_{v \in \mathcal{V}} k_v = k. \tag{5.1}$$

Every target is assigned to a single UAV.

$$\sum_{v \in \mathcal{V}} a_{i,v} = 1; \ \forall i \in \mathcal{T}. \tag{5.2}$$

The number of visits made by a UAV indexed by v before it is recharged at its depot is k_v

$$\sum_{e \in \mathcal{E}} x_{e,v} = k_v; \ \forall v \in \mathcal{V} \tag{5.3}$$

If a target is assigned to a UAV, then it is visited at least once by the UAV.

$$\sum_{e \in \delta^-(\{i\})} x_{e,v} \geq a_{i,v}; \ \forall i \in \mathcal{T}, \ v \in \mathcal{V}. \tag{5.4}$$

The next two constraints ensure that if a target is not picked, the number of times the UAV enters/leaves the target is zero. However, we require two constraints two obtain a tighter bound on this number, if the target is assigned to the UAV. The maximum number of times a UAV visits a target assigned to it is no more than half the total number of visits made by the UAV before recharging.

$$\sum_{e \in \delta^-(\{i\})} x_{e,v} \leq \frac{1}{2} * k_v * a_{i,v};\ \forall i \in \mathcal{T},\ v \in \mathcal{V}. \tag{5.5}$$

The number of times a UAV indexed v visits a target assigned to it is also no more than $k_v - \sum_{j \in \mathcal{T}} a_{j,v} + 1$. Heree, $\sum_{j \in \mathcal{T}} a_{j,v}$ indicates the is the number of targets assigned to the UAV.

$$\sum_{e \in \delta^-(\{i\})} x_{e,v} \leq (k_v - \sum_{j \in \mathcal{T}} a_{j,v} + 1) * a_{i,v};\ \forall i \in \mathcal{T},\ v \in \mathcal{V}. \tag{5.6}$$

The number of times a UAV enters a target must be equal to the number of times it departs from the target. In other words, the number of outgoing edges at a vertex must be equal to its number of incoming edges.

$$\sum_{e \in \delta^-(\{i\})} x_{e,v} = \sum_{e \in \delta^+(\{i\})} x_{e,v};\ \forall i \in \mathcal{T},\ v \in \mathcal{V}. \tag{5.7}$$

Every target is visited at least once.

$$\sum_{v \in \mathcal{V}} \sum_{e \in \delta^-(\{i\})} x_{e,v} \geq 1;\ \forall i \in \mathcal{T}. \tag{5.8}$$

Route corresponding to any vehicle cannot have sub-tours over any subset of targets.

$$\sum_{e \in \delta^-(S)} x_{e,v} \geq a_{i,v};\ i \in S,\ S \subseteq \mathcal{T},\ \forall v \in \mathcal{V}. \tag{5.9}$$

For simplicity of the formulation, we restrict the number of visits made by a UAV before recharging to be at most one less that twice the number of targets assigned to it.

$$2 * \big(\sum_{i \in \mathcal{T}} a_{i,v}\big) - 1 \geq k_v;\ \forall v \in \mathcal{V}. \tag{5.10}$$

6. SUMMARY AND FUTURE WORK

In this dissertation, we considered an optimal route planning problem that arises in persistent monitoring missions. The problem involves finding a route for UAVs to persistently monitor a given set of targets such that the sum of the maximum weighted revisit time over all the targets and the penalty of wasting the residual charge in the batteries (powering the UAVs) is minimized.

The dissertation begins with the simplest case of monitoring a set of equally weighted targets, one of which is a recharging station (depot), using a single UAV that has no curvature constraints. Even for the simplest case, the problem is computationally challenging, and in fact, it is a generalization of the notorious Traveling Salesman Problem. A naive formulation developed for solving the problem by modeling the revisit time for walks did not scale well with the problem size. The main contribution of this dissertation is a rigorous analysis of the structure of optimal solutions to the problem. The results of this analysis led to the development of efficient formulations and algorithms to solve the problem expeditiously. With the help of these results, several other variants of the problem were addressed.

The first extension was to the case in which the UAV has curvature constraints. In this variant, the problem additionally involves finding the heading angles at which the UAV arrives/departs at the targets for each visit. The aforementioned results characterizing the structure of optimal solutions extend directly to this variant. However, despite the applicability of the above results, finding optimal solutions to this problem remains challenging. For this variant, we provided two formulations that provide both lower bounds and near-optimal feasible solutions to the problem. Due to the computational difficulty involved in solving the formulation for larger instances, we also developed two preliminary algorithms that are based on independently solving the sequencing and the heading angle selection problem.

The next variant considered the case in which the targets were not accessible for the installation of a recharging station (depot). The routes considered in this case are slightly different from the earlier ones, as the UAV is allowed to visit the depot exactly once in a walk, specifically, after exhausting the charge in its batteries. Additionally, this variant also considers the time required for recharging the UAV at the depot. So, the above results do not directly extend to this case. However, based on the insights gained from the above variants and rigorous examination of optimal walks, we presented certain conjectures on the asymptotic structure of optimal walks for this variant. These conjectures were backed by extensive numerical simulations, and if proved to be true, they lead to efficient formulations for solving the problem similar to the ILP for solving the PMP.

This variant can also be considered as a special case of the unequally weighted targets by treating the depot as a target with very low priority. In this manner, the variant stands as a transition point from the case of equally weighted targets to the general case of weighted targets. Thus, the ideas developed in this dissertation can potentially be used to understand the general problem of weighted targets. Towards this end, we performed some simulations with two different weights for the targets, one of them being very high compared to the other. In other words, the simulations contain certain very-high-priority targets and certain very-low-priority targets. The optimal revisit times for an instance with 4 targets are plotted against the number of visits in Figures 6.1, 6.2 and 6.3. Figure 6.1 shows the plot for 1 high priority target, Figure 6.2 shows the plot for 2 high priority targets, and Figure 6.3 shows the plot for 3 high priority targets.

Clearly, the optimal revisit time continues to be asymptotically periodic for the considered weights. A similar pattern was observed for 24 other instances with 4 targets. However, it can be seen that the minimum revisit time achievable is no longer the optimal TSP cost. In fact, for the general case of arbitrary weights, the minimum revisit time can occur after an arbitrarily large number of visits [16]. For this reason, the general case of arbitrary weights is extremely challenging, and as of now, it remains unsolved.

Another challenging problem considered in this work was that of monitoring targets with multiple UAVs. In the presence of multiple UAVs, it is difficult not only to solve the route planning problem, but also to model it. When a target is visited by multiple UAVs, it is difficult to keep track of the time between revisits made by different UAVs. When multiple UAVs compete for recharging at the same depot, their visits must be scheduled such that they do not coincide. Even when every

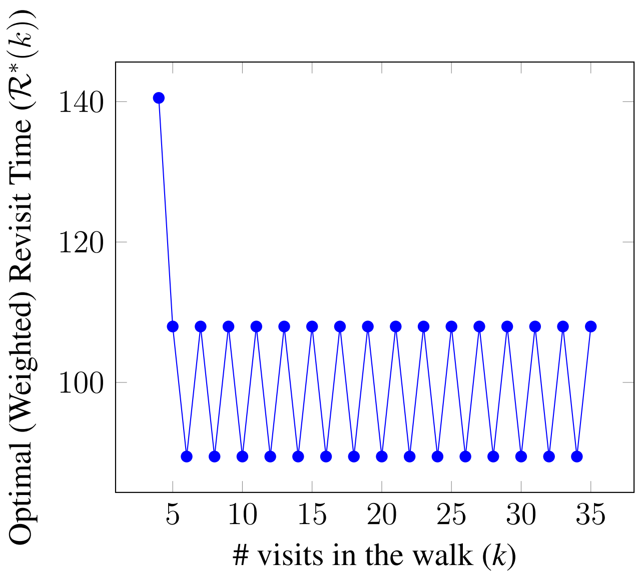


Figure 6.1: Optimal revisit time plotted against the number of visits in the walk for an instance with 4 targets. One of the targets in the instance has a very high priority compared to the others.

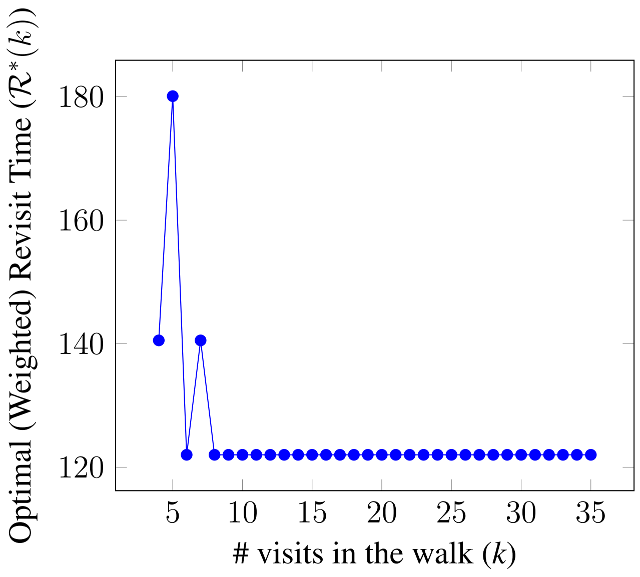


Figure 6.2: Optimal revisit time plotted against the number of visits in the walk for an instance with 4 targets. Two of the targets in the instance have a priority that is very high compared to the other two.

target is allowed to be visited by a single UAV, it is challenging to find an optimal assignment of targets to the UAVs. In this dissertation, based on the insights gained from the earlier cases, we de-

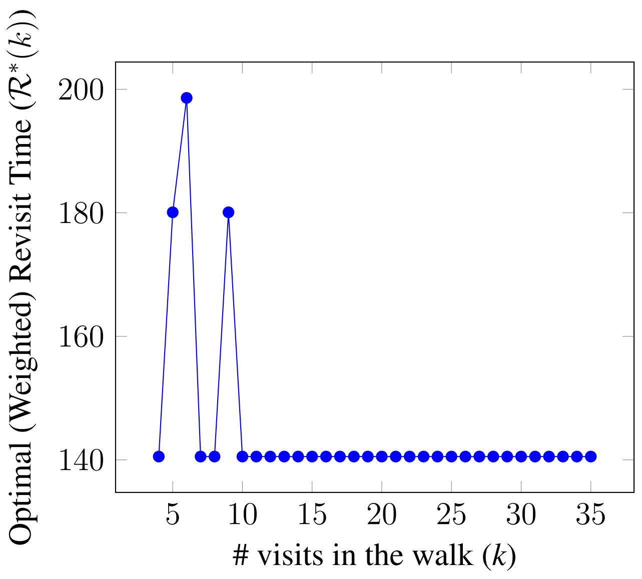


Figure 6.3: Optimal revisit time plotted against the number of visits in the walk for an instance with 4 targets. Three targets in the instance have a priority that is very high compared to the remaining target.

veloped a formulation to obtained feasible walks for multiple UAVs. However, the broad problem of finding optimal routes in the presence of multiple UAVs remains open.

REFERENCES

[1] D. W. Casbeer, R. W. Beard, T. W. McLain, S.-M. Li, and R. K. Mehra, "Forest fire monitoring with multiple small uavs," in *American Control Conference, 2005. Proceedings of the 2005*, pp. 3530–3535, IEEE, 2005.

[2] S. Rathinam, Z. W. Kim, and R. Sengupta, "Vision-based monitoring of locally linear structures using an unmanned aerial vehicle," *Journal of Infrastructure Systems*, vol. 14, no. 1, pp. 52–63, 2008.

[3] N. Nigam, S. Bieniawski, I. Kroo, and J. Vian, "Control of multiple uavs for persistent surveillance: algorithm and flight test results," *IEEE Transactions on Control Systems Technology*, vol. 20, no. 5, pp. 1236–1251, 2012.

[4] S. K. K. Hari, K. Sundar, S. Rathinam, and S. Darbha, "Scheduling tasks for human operators in monitoring and surveillance applications," *IFAC-PapersOnLine*, vol. 49, no. 32, pp. 54–59, 2016.

[5] M. Betke, R. L. Rivest, and M. Singh, "Piecemeal learning of an unknown environment," *Machine Learning*, vol. 18, no. 2-3, pp. 231–254, 1995.

[6] C. A. Duncan, S. G. Kobourov, and V. Kumar, "Optimal constrained graph exploration," *ACM Transactions on Algorithms (TALG)*, vol. 2, no. 3, pp. 380–402, 2006.

[7] P. Sujit and D. Ghose, "Search using multiple uavs with flight time constraints," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 40, no. 2, pp. 491–509, 2004.

[8] T. Bektas, "The multiple traveling salesman problem: an overview of formulations and solution procedures," *Omega*, vol. 34, no. 3, pp. 209–219, 2006.

[9] A. Machado, G. Ramalho, J.-D. Zucker, and A. Drogoul, "Multi-agent patrolling: An empirical analysis of alternative architectures," in *International Workshop on Multi-Agent Systems and Agent-Based Simulation*, pp. 155–170, Springer, 2002.

[10] G. Magesh, "Persistent surveillance using a network of mavs," Master's thesis, Indian Institute of Science, Bangalore, 2011.

[11] S. Alamdari, E. Fata, and S. L. Smith, "Persistent monitoring in discrete environments: Minimizing the maximum weighted latency between observations," *The International Journal of Robotics Research*, vol. 33, no. 1, pp. 138–154, 2014.

[12] K. Kalyanam, S. Manyam, A. V. Moll, D. Casbeer, and M. Pachter, "Scalable and exact milp methods for uav persistent visitation problem," in *2nd IEEE Conference on Control Technology and Applications, CCTA*, 2018.

[13] Y. Chevaleyre, "Theoretical analysis of the multi-agent patrolling problem," in *Intelligent Agent Technology, 2004.(IAT 2004). Proceedings. IEEE/WIC/ACM International Conference on*, pp. 302–308, IEEE, 2004.

[14] F. Pasqualetti, J. W. Durham, and F. Bullo, "Cooperative patrolling via weighted tours: Performance analysis and distributed algorithms," *IEEE Transactions on Robotics*, vol. 28, no. 5, pp. 1181–1188, 2012.

[15] F. Pasqualetti, A. Franchi, and F. Bullo, "On cooperative patrolling: Optimal trajectories, complexity analysis, and approximation algorithms," *IEEE Transactions on Robotics*, vol. 28, no. 3, pp. 592–606, 2012.

[16] S. L. Smith, M. Schwager, and D. Rus, "Persistent robotic tasks: Monitoring and sweeping in changing environments," *IEEE Transactions on Robotics*, vol. 28, no. 2, pp. 410–426, 2012.

[17] C. G. Cassandras, X. C. Ding, and X. Lin, "An optimal control approach for the persistent monitoring problem," in *Decision and Control and European Control Conference (CDC-ECC), 2011 50th IEEE Conference on*, pp. 2907–2912, IEEE, 2011.

[18] X. Lin and C. G. Cassandras, "Trajectory optimization for multi-agent persistent monitoring in two-dimensional spaces," in *Decision and Control (CDC), 2014 IEEE 53rd Annual Conference on*, pp. 3719–3724, IEEE, 2014.

[19] N. Zhou, X. Yu, S. B. Andersson, and C. G. Cassandras, "Optimal event-driven multi-agent persistent monitoring of a finite set of targets," in *Decision and Control (CDC), 2016 IEEE 55th Conference on*, pp. 1814–1819, IEEE, 2016.

[20] N. Zhou, C. G. Cassandras, X. Yu, and S. B. Andersson, "Decentralized event-driven algorithms for multi-agent persistent monitoring tasks," in *Decision and Control (CDC), 2017 IEEE 56th Annual Conference on*, pp. 4064–4069, IEEE, 2017.

[21] N. Zhou, C. G. Cassandras, X. Yu, and S. B. Andersson, "Optimal threshold-based control policies for persistent monitoring on graphs," *arXiv preprint arXiv:1803.02798*, 2018.

[22] G. Cannata and A. Sgorbissa, "A minimalist algorithm for multirobot continuous coverage," *IEEE Transactions on Robotics*, vol. 27, no. 2, pp. 297–312, 2011.

[23] S. K. K. Hari, S. Rathinam, S. Darbha, K. Kalyanam, S. G. Manyam, and D. Casbeer, "Efficient computation of optimal uav routes for persistent monitoring of targets," in *2019 International Conference on Unmanned Aircraft Systems (ICUAS)*, IEEE, 2019.

[24] S. K. K. Hari, S. Rathinam, S. Darbha, K. Kalyanam, S. G. Manyam, and D. Casbeer, "Bounding algorithms for persistent monitoring of targets using unmanned vehicles," in *2019 International Conference on Unmanned Aircraft Systems (ICUAS)*, IEEE, 2019.

[25] C. H. Papadimitriou and K. Steiglitz, *Combinatorial optimization: algorithms and complexity*. New Jersey, 1982.

[26] I. Griva, S. G. Nash, and A. Sofer, *Linear and nonlinear optimization, 2nd Edition, SIAM*. 2009.

[27] S. K. K. Hari, S. Rathinam, S. Darbha, K. Kalyanam, S. G. Manyam, and D. Casbeer, "Persistent monitoring of dynamically changing environments using an unmanned vehicle," *arXiv preprint arXiv:1808.02545*, 2018.

[28] M. Held and R. M. Karp, "The traveling-salesman problem and minimum spanning trees: Part ii," *Mathematical programming*, vol. 1, no. 1, pp. 6–25, 1971.

[29] N. Christofides, *Graph theory: An algorithmic approach (Computer science and applied mathematics)*. Academic Press, Inc., 1975.

[30] N. Christofides, “Worst-case analysis of a new heuristic for the travelling salesman problem,” tech. rep., Carnegie-Mellon Univ Pittsburgh Pa Management Sciences Research Group, 1976.

[31] 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, 1957.

[32] S. K. K. Hari, S. Rathinam, S. Darbha, K. Kalyanam, S. G. Manyam, and D. Casbeer, “The generalized persistent monitoring problem,” in *2019 American Control Conference (ACC)*, IEEE, 2019.

[33] S. G. Manyam, S. Rathinam, D. Casbeer, and E. Garcia, “Tightly bounding the shortest dubins paths through a sequence of points,” *Journal of Intelligent & Robotic Systems*, vol. 88, no. 2-4, pp. 495–511, 2017.

[34] L. Gurobi Optimization, “Gurobi optimizer reference manual,” 2019.

[35] S. Rathinam, R. Sengupta, and S. Darbha, “A resource allocation algorithm for multivehicle systems with nonholonomic constraints,” *IEEE Transactions on Automation Science and Engineering*, vol. 4, no. 1, pp. 98–104, 2007.

[36] J. Bezanson, A. Edelman, S. Karpinski, and V. B. Shah, “Julia: A fresh approach to numerical computing,” *SIAM review*, vol. 59, no. 1, pp. 65–98, 2017.

[37] A. Hagberg, P. Swart, and D. S Chult, “Exploring network structure, dynamics, and function using networkx,” tech. rep., Los Alamos National Lab.(LANL), Los Alamos, NM (United States), 2008.

[38] K. Sundar and S. Rathinam, “Algorithms for heterogeneous, multiple depot, multiple unmanned vehicle path planning problems,” *Journal of Intelligent & Robotic Systems*, vol. 88, no. 2-4, pp. 513–526, 2017.

APPENDIX A

ADDITIONAL PROOFS

Theorem 2.4.4: Given a walk $\mathcal{W}$, let $\mathcal{W}_b$ be its binding subwalk, and S_1, S_2 be its subwalks such that $\mathcal{W} = S_1 \circ \mathcal{W}_b \circ S_2$. Let $\mathcal{W}'_b$ be a closed subwalk obtained by shortcutting visits from $\mathcal{W}_b$, but retaining the last visits to targets. Let a closed walk $\bar{\mathcal{W}}$ be formed by concatenating $\mathcal{W}$ with $\mathcal{W}'_b$ as follows: $\bar{\mathcal{W}} = S_1 \circ \mathcal{W}_b \circ \mathcal{W}'_b \circ S_2$. Then, $\mathcal{R}(\bar{\mathcal{W}}) = \mathcal{R}(\mathcal{W})$.

Proof. Let target d be the terminus of $\mathcal{W}_b$; then d is also the terminus of $\mathcal{W}'_b$, as shown in Figure A.1. By definition, $T(\mathcal{W}_b) = RT(d, \mathcal{W}) = \mathcal{R}(\mathcal{W})$. Since $\mathcal{W}'_b$ is formed by shortcutting visits from $\mathcal{W}_b$, we have $T(\mathcal{W}'_b) \leq T(\mathcal{W}_b) = RT(d, \mathcal{W})$. Now consider the closed walk $\bar{\mathcal{W}} = S_1 \circ \mathcal{W}_b \circ \mathcal{W}'_b \circ S_2$. One can see that $RT(d, \bar{\mathcal{W}}) = \max\{RT(d, \mathcal{W}), T(\mathcal{W}'_b)\} = RT(d, \mathcal{W}) = \mathcal{R}(\mathcal{W})$.

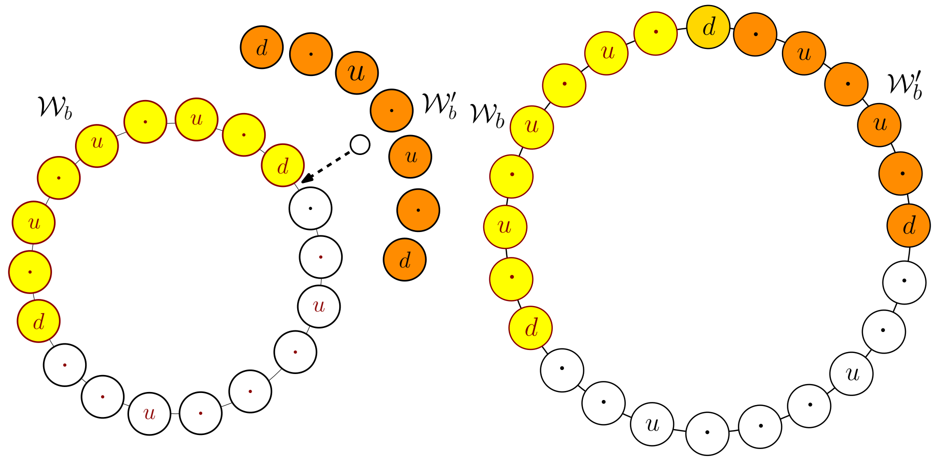


Figure A.1: A closed walk $\mathcal{W}$ is shown on the left, with its binding subwalk $\mathcal{W}_b$ colored yellow. A subwalk $\mathcal{W}'_b$ (colored orange) is constructed from $\mathcal{W}_b$, by shortcutting visits to target u (retaining the last visit to u). Figure on the right shows the concatenated walk $\bar{\mathcal{W}}$, as discussed in Theorem 2.4.4.

It is now sufficient to show that for any target $u \neq d$, $RT(u, \bar{\mathcal{W}}) \leq \mathcal{R}(\mathcal{W})$. Consider a decomposition of $\bar{\mathcal{W}}$ with respect to u. If u is visited r times in $\bar{\mathcal{W}}$, then there exists a decomposition

$$\mathcal{C}(\bar{\mathcal{W}}, u) = \bar{\mathcal{W}}_1 \circ \cdots \circ \bar{\mathcal{W}}_r,$$

such that u is the terminus of $\bar{\mathcal{W}}_1$, ..., $\bar{\mathcal{W}}_r$. According to the above decomposition, the revisit time of u in $\bar{\mathcal{W}}$ can be expressed as $RT(u, \bar{\mathcal{W}}) = \max\{T(\bar{\mathcal{W}}_1), \ldots, T(\bar{\mathcal{W}}_r)\}$, and only one of the following cases hold for every $1 \leq i \leq r$:

1. $\bar{\mathcal{W}}_i$ is a closed subwalk of $\mathcal{W}$,
2. $\bar{\mathcal{W}}_i$ is a closed subwalk of $\mathcal{W}'_b$,
3. $\bar{\mathcal{W}}_i$ is a closed subwalk of $\mathcal{W} \circ \mathcal{W}'_b$, but is not a subwalk of either $\mathcal{W}$ or $\mathcal{W}'_b$,
4. $\bar{\mathcal{W}}_i$ is a closed subwalk of $\mathcal{W}'_b \circ \mathcal{W}$, but is not a subwalk of either $\mathcal{W}$ or $\mathcal{W}'_b$.

In case 1, by definition, $T(\bar{\mathcal{W}}_i) \leq \mathcal{R}(\mathcal{W})$. Similarly, in case 2, we have $T(\bar{\mathcal{W}}_i) \leq T(\bar{\mathcal{W}}'_b) \leq T(\bar{\mathcal{W}}_b) = \mathcal{R}(\mathcal{W})$. To show $T(\bar{\mathcal{W}}_i) \leq \mathcal{R}(\mathcal{W})$ for cases 3 and 4, we consider decompositions of $\mathcal{W}_b$ and $\mathcal{W}'_b$ with respect to their last visists to u as discussed below.

$\mathcal{W}_b$ is decomposed with respect to its last visit to u as:

$$\mathcal{W}_b = \mathcal{S}_{b_1} \circ \mathcal{S}_{b_2},$$

such that the last visit to u in $\mathcal{W}_b$ appears as the last node of $\mathcal{S}_{b_1}$ and the first node of $\mathcal{S}_{b_2}$, and d is the first node of the subwalk $\mathcal{S}_{b_1}$, and the last node of the subwalk $\mathcal{S}_{b_2}$.

Similarly, $\mathcal{W}'_b$ is decomposed with respect to the last visit to u as:

$$\mathcal{W}'_b = \hat{\mathcal{S}}_{b_1} \circ \hat{\mathcal{S}}_{b_2},$$

such that the last visit to u in $\mathcal{W}'_b$ appears as the the last node of $\hat{\mathcal{S}}_{b_1}$, but the first node of $\hat{\mathcal{S}}_{b_1}$, and d is the first node of $\hat{\mathcal{S}}_{b_1}$ and the last node of $\hat{\mathcal{S}}_{b_2}$.

Since $\mathcal{W}'_b$ is formed by shortcutting visits from $\mathcal{W}_b$, but retaining the last visit to u, we have $T(\hat{\mathcal{S}}_{b_1}) \leq T(\mathcal{S}_{b_1})$, and $T(\hat{\mathcal{S}}_{b_2}) \leq T(\mathcal{S}_{b_2})$.

Consider case 3; $\bar{\mathcal{W}}_i$ has its initial node in $\mathcal{W}$, and terminal node in $\mathcal{W}'_b$. By construction, $\mathcal{W}_b$ precedes $\mathcal{W}'_b$ in $\bar{\mathcal{W}}$. Moreover, from Lemma 2.4.1, u is visited in $\mathcal{W}_b$. Therefore, the first node of $\bar{\mathcal{W}}_i$ is in $\mathcal{W}_b$, and the last node of $\bar{\mathcal{W}}_i$ is in $\mathcal{W}'_b$. So, as shown in Figure A.2, $\bar{\mathcal{W}}_i$ can be decomposed with respect to d as:

$$\bar{\mathcal{W}}_i = \bar{\mathcal{S}}_{i_1} \circ \bar{\mathcal{S}}_{i_2},$$

where $\bar{\mathcal{S}}_{i_1}$ is a subwalk of $\mathcal{W}_b$ with u and d as its first and last nodes respectively, and $\bar{\mathcal{S}}_{i_2}$ is a subwalk of $\mathcal{W}'_b$ with d and u as its first and last nodes respectively.

By construction, $\bar{\mathcal{S}}_{i_1} = \mathcal{S}_{b_2}$, and $\bar{\mathcal{S}}_{i_2}$ is a subwalk of $\hat{\mathcal{S}}_{b_1}$. Thereby, we have $T(\bar{\mathcal{S}}_{i_1}) = T(\mathcal{S}_{b_2})$, and $T(\bar{\mathcal{S}}_{i_2}) \leq T(\hat{\mathcal{S}}_{b_1})$ (note that equality holds when $\mathcal{W}'_b$ is formed by shortcutting all visits to u except for the last visit to it). Thereby, it follows that $T(\bar{\mathcal{W}}_i) = T(\bar{\mathcal{S}}_{i_1}) + T(\bar{\mathcal{S}}_{i_2}) = T(\mathcal{S}_{b_2}) + T(\hat{\mathcal{S}}_{b_1}) \leq T(\mathcal{S}_{b_2}) + T(\mathcal{S}_{b_1}) = T(\mathcal{W}_b) = \mathcal{R}(\mathcal{W})$.

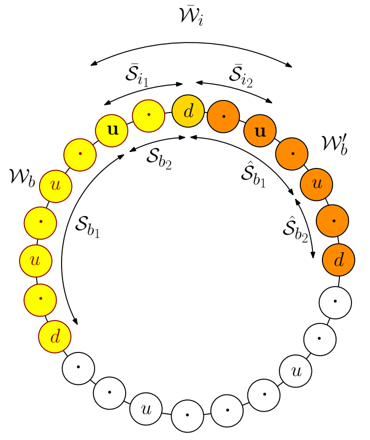
Consider case 4; the first node of $\bar{\mathcal{W}}_i$ is in $\mathcal{W}'_b$, and the last node of $\bar{\mathcal{W}}_i$ is in $\mathcal{W}$. Then, as shown in Figure A.3, $\bar{\mathcal{W}}_i$ can be decomposed with respect to d as:

$$\bar{\mathcal{W}}_i = \bar{\mathcal{S}}_{i_1} \circ \bar{\mathcal{S}}_{i_2},$$

where $\bar{\mathcal{S}}_{i_1}$ is a subwalk of $\mathcal{W}'_b$ with u and d as its first and last nodes respectively, and $\bar{\mathcal{S}}_{i_2}$ is a subwalk of $\mathcal{W}$ (or $\mathcal{S}_2$) with d and u as its first and last nodes respectively.

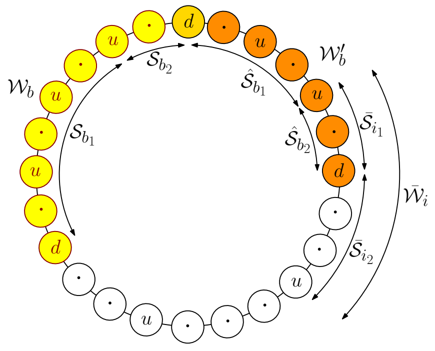
By construction, $\bar{\mathcal{S}}_{i_1} = \hat{\mathcal{S}}_{b_2}$ and $T(\bar{\mathcal{S}}_{i_1}) = T(\hat{\mathcal{S}}_{b_2})$. Therefore, $T(\bar{\mathcal{W}}_i) = T(\bar{\mathcal{S}}_{i_1}) + T(\bar{\mathcal{S}}_{i_2}) = T(\hat{\mathcal{S}}_{b_2}) + T(\bar{\mathcal{S}}_{i_2}) \leq T(\mathcal{S}_{b_2}) + T(\bar{\mathcal{S}}_{i_2}) \leq RT(u, \mathcal{W}) \leq \mathcal{R}(\mathcal{W})$. Note that the second inequality follows from the fact that $\mathcal{S}_{b_1} \circ \bar{\mathcal{S}}_{i_2}$ is a closed subwalk of $\mathcal{W}$ with u as its terminus.

Hence, we proved that for any arbitrary node $u \neq d$, $RT(u, \bar{\mathcal{W}}) \leq \mathcal{R}(\mathcal{W})$, and $RT(d, \bar{\mathcal{W}} = \mathcal{R}(\mathcal{W})$. So, by definition, $\mathcal{R}(\bar{\mathcal{W}}) = RT(d, \bar{\mathcal{W}} = \mathcal{R}(\mathcal{W})$.



$\bar{\mathcal{W}}_i$
$\bar{\mathcal{S}}_{i_1}$
$\bar{\mathcal{S}}_{i_2}$
$\mathcal{W}'_b$
$\mathcal{W}_b$
$\mathcal{S}_{b_2}$
$\hat{\mathcal{S}}_{b_1}$
$\mathcal{S}_{b_1}$
$\hat{\mathcal{S}}_{b_2}$

Figure A.2: A closed walk $\bar{\mathcal{W}}$, and its corresponding subwalks as in case 3 of Theorem 2.4.4.



$\mathcal{W}'_b$
$\mathcal{W}_b$
$\mathcal{S}_{b_2}$
$\hat{\mathcal{S}}_{b_1}$
$\bar{\mathcal{S}}_{i_1}$
$\mathcal{S}_{b_1}$
$\hat{\mathcal{S}}_{b_2}$
$\bar{\mathcal{W}}_i$
$\bar{\mathcal{S}}_{i_2}$

Figure A.3: A closed walk $\bar{\mathcal{W}}$, and its corresponding subwalks as in case 4 of Theorem 2.4.4.

Therefore, we proved that $RT(u, \bar{\mathcal{W}}) \leq \mathcal{R}(\mathcal{W})$. Since u is arbitraty, this is true for all the targets, and we have, $\mathcal{R}(\bar{\mathcal{W}}) \leq \mathcal{R}(\mathcal{W})$. This together with $\mathcal{R}(\bar{\mathcal{W}}) \leq \mathcal{R}(\mathcal{W})$ proved above implies $\mathcal{R}(\bar{\mathcal{W}}) = \mathcal{R}(\mathcal{W})$.

□

Lemma 2.4.5: Let a closed walk $\bar{\mathcal{W}}$ be formed by concatenating $\mathcal{W}(k)$ with itself p times as follows:

$$\bar{\mathcal{W}} := \underbrace{\mathcal{W}(k) \circ \cdots \circ \mathcal{W}(k)}_{p\ times}.$$

Then, for any node v_r

- $\mathcal{C}(\bar{\mathcal{W}}, v_r) = \underbrace{\mathcal{C}(\mathcal{W}(k), v_r) \circ \cdots \circ \mathcal{C}(\mathcal{W}(k), v_r)}_{p\ times}.$
- $RT(v_r, \bar{\mathcal{W}}) = RT(v_r, \mathcal{W}(k)),$

and thus $\mathcal{R}(\bar{\mathcal{W}}) = \mathcal{R}(\mathcal{W}(k))$.

Proof. Let $\mathcal{W} = (v_1, v_2, \dots v_k, v_1)$ with v_r being an intermediate node. Then,

$$\bar{\mathcal{W}} = (\underbrace{v_1, v_2, \dots, v_k}_{1}, \underbrace{v_1, v_2, \dots, v_k}_{2}, \dots, \underbrace{v_1, v_2, \dots, v_k}_{p}, v_1).$$

Clearly, then

$$\begin{aligned}\mathcal{C}(\bar{\mathcal{W}}, v_r) = (&\underbrace{v_r, v_{r+1}, \dots, v_k, v_1, v_2, \dots, v_{r-1}}_{1},\\ &\underbrace{v_r, v_{r+1}, \dots, v_k, v_1, v_2, \dots, v_{r-1}}_{2},\\ &\vdots\\ &\underbrace{v_r, v_{r+1}, \dots, v_k, v_1, v_2, \dots, v_{r-1}}_{p}, v_r)\\ = &\underbrace{\mathcal{C}(\mathcal{W}, v_r) \circ \mathcal{C}(\mathcal{W}, v_r) \circ \cdots \circ \mathcal{C}(\mathcal{W}, v_r)}_{p\ times}.\end{aligned}$$

Let v_r be visited l times in $\mathcal{W}$. Then there exist closed subwalks $\mathcal{W}_1, \mathcal{W}_2, \dots \mathcal{W}_l$ with v_r as a terminus such that

$$\mathcal{C}(\mathcal{W}(k), v_r) = \mathcal{W}_1 \circ \mathcal{W}_2 \circ \dots \circ \mathcal{W}_l.$$

Hence,

$$\mathcal{C}(\bar{\mathcal{W}}, v_r) = \underbrace{\mathcal{W}_1 \circ \mathcal{W}_2 \circ \ldots \circ \mathcal{W}_l}_{1} \circ$$
$$\underbrace{\mathcal{W}_1 \circ \mathcal{W}_2 \circ \ldots \circ \mathcal{W}_l}_{2} \circ$$
$$\vdots$$
$$\underbrace{\mathcal{W}_1 \circ \mathcal{W}_2 \circ \ldots \circ \mathcal{W}_l}_{p}.$$

By definition, we have

$$RT(v_r, \mathcal{C}(\bar{\mathcal{W}}, v_r)) = \max_{1 \le i \le l} T(\mathcal{W}_i), \text{ and}$$

$$RT(v_r, \mathcal{C}(\mathcal{W}(k), v_r)) = \max_{1 \le i \le l} T(\mathcal{W}_i).$$

$$\implies RT(v_r, \mathcal{C}(\bar{\mathcal{W}}, v_r)) = RT(v_r, \mathcal{C}(\mathcal{W}(k), v_r)), \forall v_r \in \mathcal{T}.$$

Hence, by definition,

$RT(v_r, \bar{\mathcal{W}}) = RT(v_r, \mathcal{C}(\bar{\mathcal{W}}, v_r)) = RT(v_r, \mathcal{C}(\mathcal{W}(k), v_r)) = RT(v_r, \mathcal{W}(k)), \quad \forall v_r \in \mathcal{T}.$

Therefore,

$$\mathcal{R}(\bar{\mathcal{W}}) = \mathcal{R}(\mathcal{W}(k)).$$

$\square$

Lemma 2.4.6: Let $\mathcal{W}(k)$ be a closed walk with $n+1 \le k \le 2n-1$ visits, and d be a target that is visited exactly once in $\mathcal{W}(k)$. Let $\mathcal{W}(k-1)$ be a walk formed by shortcutting a revisit from $\mathcal{W}(k)$. Consider a closed walk $\bar{\mathcal{W}}$ formed by concatenating $\mathcal{W}(k)$ for q (≥ 1) times and $\mathcal{W}(k-1)$ for p times as follows:

$$\bar{\mathcal{W}} := \underbrace{\mathcal{W}(k) \circ .. \circ \mathcal{W}(k)}_{q\ times} \circ \underbrace{\mathcal{W}(k-1) \circ .. \circ \mathcal{W}(k-1)}_{p\ times},$$

Then,

$$\mathcal{C}(\bar{\mathcal{W}}, d) \quad = \quad \underbrace{\mathcal{C}(\mathcal{W}(k), d) \circ \cdots \circ \mathcal{C}(\mathcal{W}(k), d)}_{q\ times} \circ \underbrace{\mathcal{C}(\mathcal{W}(k-1), d) \circ \cdots \circ \mathcal{C}(\mathcal{W}(k-1), d)}_{p\ times}.$$

Proof. Let $\mathcal{W}(k) = (v_1, v_2, \dots v_r \dots v_k, v_1)$, with v_1 as the initial node and let $v_r = d$. We may construct $\mathcal{W}(k-1)$ by shortcutting a node v_s from $\mathcal{W}(k)$, where $1 \leq s(\neq r) \leq k$.
Consider the case $s > r$. Then,

$$\mathcal{W}(k) = (v_1, v_2, \dots, v_{r-1}, d, v_{r+1}, \dots, v_s \dots v_k, v_1),$$

$$\text{and } \mathcal{W}(k-1) = (v_1, v_2, \dots, v_{r-1}, d, v_{r+1}, \dots$$

$$, v_{s-1}, v_{s+1}, \dots, v_k, v_1).$$

Their cyclic permutations with d as the terminus are:

$$\mathcal{C}(\mathcal{W}(k), d) = (d, v_{r+1}, \dots, v_s \dots v_k, v_1, \dots, v_{r-1}, d),$$

$$\text{and } \mathcal{C}(\mathcal{W}(k-1), d) = (d, v_{r+1}, \dots, v_{s-1}, v_{s+1}, \dots, v_k, v_1, \dots, v_{r+1}, d).$$

Then,

$$\underbrace{\mathcal{C}(\mathcal{W}(k), d) \circ \cdots \circ \mathcal{C}(\mathcal{W}(k), d)}_{q\ times} \circ \underbrace{\mathcal{C}(\mathcal{W}(k-1), d) \circ \cdots \circ \mathcal{C}(\mathcal{W}(k-1), d)}_{p\ times} \qquad =$$

$$\underbrace{(d, v_{r+1}, \ldots, v_s, \ldots, v_k, v_1, \ldots, v_{r-1},}_{1}$$

$$\vdots$$

$$\underbrace{d, v_{r+1}, \ldots, v_s \ldots v_k, v_1, \ldots, v_{r-1},}_{q}$$

$$\underbrace{d, v_{r+1}, \ldots, v_{s-1}, v_{s+1}, \ldots, v_k, v_1, \ldots, v_{r-1},}_{1}$$

$$\vdots$$

$$\underbrace{d, v_{r+1}, \ldots, v_{s-1}, v_{s+1}, \ldots, v_k, v_1, \ldots, v_{r-1}, d).}_{p}$$

Now consider

$$\bar{\mathcal{W}} := \underbrace{\mathcal{W}(k) \circ .. \circ \mathcal{W}^*(k)}_{q\ times} \circ \underbrace{\mathcal{W}(k-1) \circ .. \circ \mathcal{W}(k-1)}_{p\ times}$$

$$= \underbrace{(v_1, v_2, \ldots, v_{r-1}, \textcircled{d}, v_{r+1}, \ldots v_s \ldots v_k,}_{1}$$

$$\vdots$$

$$\underbrace{(v_1, v_2, \ldots, v_{r-1}, d, v_{r+1}, \ldots v_s \ldots v_k, v_1)}_{p}$$

$$\underbrace{(v_1, v_2, \ldots, v_{r-1}, d, v_{r+1}, \ldots v_{s-1}, v_{s+1} \ldots v_k,)}_{1}$$

$$\vdots$$

$$\underbrace{v_1, v_2, \ldots, v_{r-1}, \boldsymbol{d}, v_{r+1}, \ldots v_{s-1}, v_{s+1} \ldots v_k, v_1)}_{q}.$$

A cyclic permutation of $\bar{\mathcal{W}}$, with its first visit to d (circled) as the initial node can be written as,

$$\mathcal{C}(\bar{\mathcal{W}}, d) =$$

$$\underbrace{(d, v_{r+1}, \ldots, v_s \ldots v_k, v_1, \ldots, v_{r-1},}_{1}$$

$$\vdots$$

$$\underbrace{d, v_{r+1}, \ldots, v_s \ldots v_k, v_1, \ldots, v_{r-1},}_{q}$$

$$\underbrace{d, v_{r+1}, \ldots, v_{s-1}, v_{s+1}, \ldots, v_k, v_1, \ldots, v_{r-1},}_{1}$$

$$\vdots$$

$$\underbrace{d, v_{r+1}, \ldots, v_{s-1}, v_{s+1}, \ldots, v_k, v_1, \ldots, v_{r-1}, d).}_{p}$$

Hence, it can be seen that

$$\mathcal{C}(\bar{\mathcal{W}}, d) = \underbrace{\mathcal{C}(\mathcal{W}(k), d) \circ \cdots \circ \mathcal{C}(\mathcal{W}(k), d)}_{q\ times} \circ \underbrace{\mathcal{C}(\mathcal{W}(k-1), d) \circ \cdots \circ \mathcal{C}(\mathcal{W}(k-1), d)}_{p\ times}.$$

The same can be shown for the case $s < r$, by considering a permutation of $\bar{\mathcal{W}}$ with the last visit to d (shown in bold) as its initial node.

□