Performance Optimization for Unmanned Vehicle Systems

by

Jerome Le Ny

Ingénieur de l'Ecole Polytechnique (2001)
M.S., Electrical Engineering, University of Michigan (2003)

Submitted to the Department of Aeronautics and Astronautics
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 2008

Author . . . . . .
Department of Aeronautics and Astronautics
August 29, 2008

Certified by . . . . . .
Emilio Frazzoli
Associate Professor of Aeronautics and Astronautics
Thesis Supervisor

Certified by . . . . . .
Munther A. Dahleh
Professor of Electrical Engineering
Thesis Supervisor

Certified by . . . . . .
Eric Feron
Professor of Aerospace Engineering
Thesis Supervisor

Certified by . . . . . .
John N. Tsitsiklis
Professor of Electrical Engineering
Thesis Advisor

Accepted by . . . . . .
David L. Darmofal
Professor of Aeronautics and Astronautics, Associate Department Head
Chair, Committee on Graduate Students

Performance Optimization for Unmanned Vehicle Systems

by

Jerome Le Ny

Submitted to the Department of Aeronautics and Astronautics
on August 29, 2008, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

Abstract

Technological advances in the area of unmanned vehicles are opening new possibilities for creating teams of vehicles performing complex missions with some degree of autonomy. Perhaps the most spectacular example of these advances concerns the increasing deployment of unmanned aerial vehicles (UAVs) in military operations. Unmanned Vehicle Systems (UVS) are mainly used in Information, Surveillance and Reconnaissance missions (ISR). In this context, the vehicles typically move about a low-threat environment which is sufficiently simple to be modeled successfully. This thesis develops tools for optimizing the performance of UVS performing ISR missions, assuming such a model.

First, in a static environment, the UVS operator typically requires that a vehicle visit a set of waypoints once or repetitively, with no a priori specified order. Minimizing the length of the tour traveled by the vehicle through these waypoints requires solving a Traveling Salesman Problem (TSP). We study the TSP for the Dubins' vehicle, which models the limited turning radius of fixed wing UAVs. In contrast to previously proposed approaches, our algorithms determine an ordering of the waypoints that depends on the model of the vehicle dynamics. We evaluate the performance gains obtained by incorporating such a model in the mission planner.

With a dynamic model of the environment the decision making level of the UVS also needs to solve a sensor scheduling problem. We consider M UAVs monitoring $N > M$ sites with independent Markovian dynamics, and treat two important examples arising in this and other contexts, such as wireless channel or radar waveform selection. In the first example, the sensors must detect events arising at sites modeled as two-state Markov chains. In the second example, the sites are assumed to be Gaussian linear time invariant (LTI) systems and the sensors must keep the best possible estimate of the state of each site. We first present a bound on the achievable performance which can be computed efficiently by a convex program, involving linear matrix inequalities in the LTI case. We give closed-form formulas for a feedback index policy proposed by Whittle. Comparing the performance of this policy to the bound, it is seen to perform very well in simulations. For the LTI example, we propose new open-loop periodic switching policies whose performance matches the bound.

Ultimately, we need to solve the task scheduling and motion planning problems simultaneously. We first extend the approach developed for the sensor scheduling problems to the case where switching penalties model the path planning component. Finally, we propose a new modeling approach, based on fluid models for stochastic networks, to obtain insight into more complex spatiotemporal resource allocation problems. In particular, we give a necessary and sufficient stabilizability condition for the fluid approximation of the problem of harvesting data from a set of spatially distributed queues with spatially varying transmission rates using a mobile server.

Thesis Supervisor: Emilio Frazzoli
Title: Associate Professor of Aeronautics and Astronautics

Thesis Supervisor: Munther A. Dahleh
Title: Professor of Electrical Engineering

Thesis Supervisor: Eric Feron
Title: Professor of Aerospace Engineering

Thesis Advisor: John N. Tsitsiklis
Title: Professor of Electrical Engineering

Acknowledgments

I was very fortunate to have three remarkable advisors, Prof. Eric Feron, Prof. Munther Dahleh, and Prof. Emilio Frazzoli, during my Ph.D. program. I thank them for their encouragement and advice, in research and academic life, which have been invaluable. They are a constant source of ideas and inspiration, and have shaped the directions taken by this thesis.

I thank Prof. John Tsitsiklis for his helpful comments as member of my committee. His questions and insight during our meetings have led me to interesting paths. I am grateful to Prof. Hamsa Balakrishnan for her willingness to serve as my thesis reader and for her teaching. It is also a good opportunity to thank Prof. Sean Meyn for his help and interest in chapter 7.

LIDS has been an wonderful place to work for the past four years, and I would like to thank all the professors and students, in particular from the control systems group, with whom I interacted. I remember vividly the courses of Profs. Alex Megretski and Pablo Parrilo. Prof. Asu Ozdaglar was a great source of encouragements. Over the years, my labmates Alessandro Arsie, Ola Ayaso, Erin Aylward, Selcuk Bayraktar, Amit Bhatia, Animesh Chakravarthy, Han-Lim Choi, Emily Craparo, Jan de Mot, Sleiman Itani, Damien Jourdan, Sertac Karaman, Yola Katsargyri, Georgios Kotsalis, Patrick Kreidl, Ilan Lobel, Paul Njoroge, Mesrob Ohannessian, Mitra Osqui, Marco Pavone, Mardavij Roozbehani, Michael Rinehart, Phil Root, Navid Sabbaghi, Keith Santarelli, Ketan Savla, Tom Schouwenaars, Parikshit Shah, Danielle Tarraf, Olivier Toupet, Holly Waisanen, Theo Weber, certainly made the experience a lot more fun! The LIDS and Aero-Astro staff was also very helpful, and in particular Jennifer Donovan, Lisa Gaumond, Doris Inslee, Fifa Monserrate and Marie Stuppard made my life at MIT much easier.

Tracing back the influences which brought me to MIT, I wish to thank my teachers at the Ecole Polytechnique and the University of Michigan. In particular, I thank Profs. Nick Triantafyllidis and Linda Katehi for accepting me as their student and introducing me to research.

I wish to express my deepest gratitude to my parents and brothers for their love and support. Finally, I dedicate this thesis to my wife, Bettina, whose love and patience were my source of motivation. I look forward to starting our new journey together.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Unmanned Vehicle Systems

Unmanned vehicles, which have been used for decades for space exploration, are becoming important assets in military operations. The traditional role of unmanned aerial vehicles (UAVs) in Intelligence, Surveillance and Reconnaissance missions (ISR) is now completed by developments of their capacities for secure communications and data distribution, and combat missions, among others [128]. For these tasks, unmanned vehicles offer significant advantages over standard aircraft: the possibility of cheaper machines that can be deployed in greater number, with better capabilities (e.g. endurance), and without risk to aircrews. In 2002, the goal of the Air Force was that, "by 2010, one third of the operational deep strike aircraft of the Armed Forces be unmanned" [9]. Although important challenges still have to be overcome to allow their seamless operations in the civilian airspace, they could be used in areas such as environmental research, weather forecasting, border patrol, traffic monitoring, and petroleum exploration. The Darpa Grand Challenge is perhaps the best example of the attention that autonomous ground vehicles are receiving as well [42].

The main motivation for this thesis comes from the development of Unmanned Aerial Systems (UAS). Compared to the parallel research efforts on sensor networks, we can characterize these mobile robotic networks, at least for a reasonably close future, as being composed of tens or perhaps hundreds of vehicles at most, each possessing important computational and communication capabilities. The current architecture of these systems is currently composed of a ground station collecting data from the vehicles and supervising their operation. Especially for ISR missions in low threat environments, an important research topic is to increase the level of autonomy of UAS and their integration with human operators. Ideally, a single operator should be able to monitor several UAVs which would automatically optimize their behavior based on the environment encountered and the data gathered by the overall system. Compared to UAV flight technology, the autonomy technology is fairly immature and will probably remain the primary bottleneck for future developments.

Increasing autonomy in UAS will require integrating advances in diverse areas such as sensor fusion, cooperation over communication networks, path and motion planning, trajectory regulation, task allocation and scheduling. In this work, we adopt an approach based on performance optimization to design trajectories and allocate tasks in UAS. We consider several scenarios, with a particular emphasis on dynamic environments, which is a less developed area in the available literature on UAS mission planning.

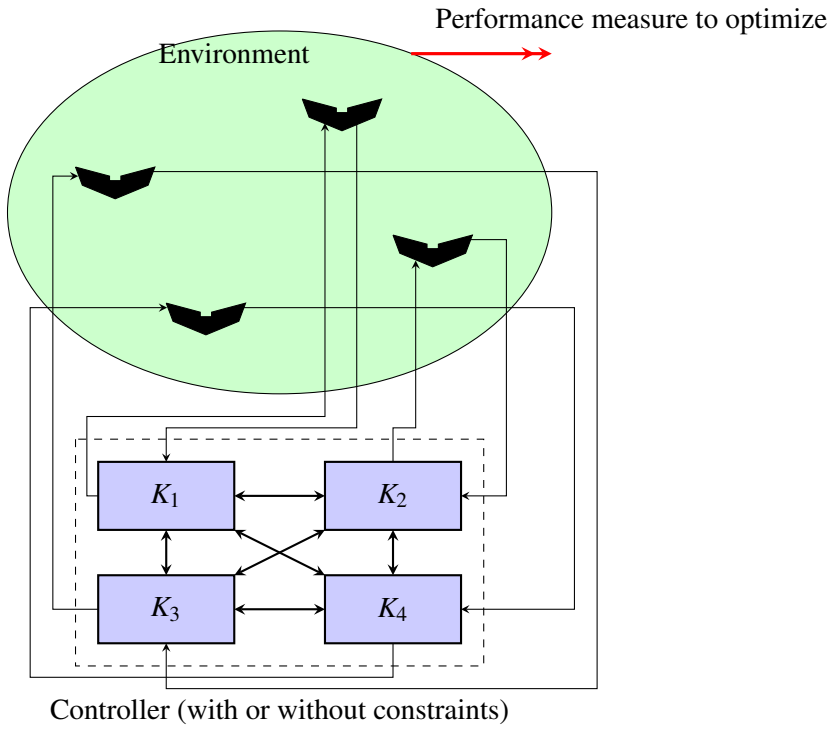


Figure 1-1: UAS Mission Performance Optimization

1.2 General Framework

Figure (1-1) describes a general framework in which the controller design problem for unmanned vehicle systems (UVS) can be cast. The aggregate state of the environment and the vehicles constitutes the "plant", and we can only influence its dynamics by controllers acting on the state of the vehicles. These vehicles typically provide only local observations of the state of the environment. The goal is to design controllers so that the vehicles behave as well as possible in the environment, according to some specified performance measure. There is already a large and fast growing literature on designing trajectories for UVS and mobile sensors. This could be for an end-point control problem, when the sensors must reach a final configuration achieving a good area coverage [32], or a more dynamical task such as adaptive sampling [86, 129]. For other task allocation problems for UVS, most of the current literature focuses on static problems, usually variations of the weapon-target assignment problem [115]. Under this approach, an optimization problem is solved periodically to account for changes in the environment. An exception is the adaptation of the work on the dynamic traveling repairman problem [18] in the context of mobile robotic networks [110].

A large part of the literature on mobile robotic networks adopts a "bottom-up" approach, focusing first on a set of constraints that must be satisfied by the controller, such as being distributed and allowing only local communications. The focus, at least currently, is then more on the tactical level planning, for example understanding how the group can agree on adopting a given spatial configuration, and synthesizing interesting collective behaviors from local rules, see e.g. [98, 21]. Since we assume that a central controller is available to the UAS at the ground station, or that all vehicles have the capacity to communicate with each other, these tactical level issues do not arise in this thesis. Instead, we focus on mission planning at the strategic level, deciding what action the system should take at each time and using which vehicle. We do not consider in much detail the

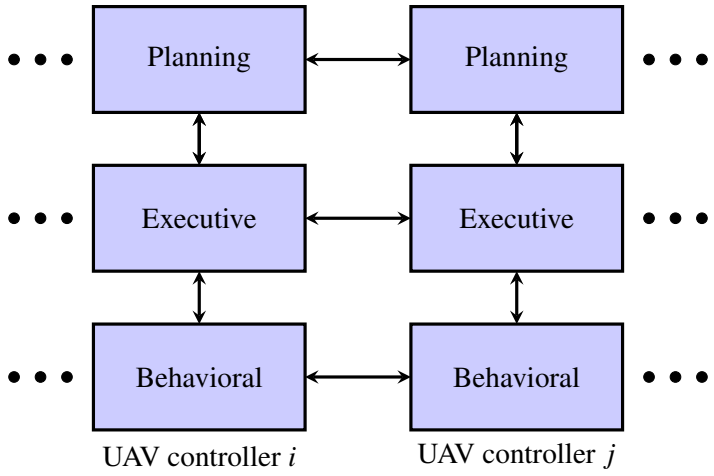


Figure 1-2: Hierarchical Control

questions related to the potential distributed implementation of the control laws developed. However, the computational complexity of the proposed algorithms still plays an important role in our study.

Planning for unmanned systems is a broad area that has been studied by the computer science and control communities in particular, see e.g. [25, 78, 29] for a very small sample of the existing research with a focus similar to ours. We review the more relevant literature in the next section along with the specific problems considered in this thesis.

1.3 Contributions and Literature Overview

1.3.1 Static Path Planning: The Traveling Salesman Problem for a Dubins Vehicle

We start by considering a static environment with a given set of waypoints to be visited by a single vehicle, with no a priori specified order. In order to increase its autonomy and minimize its cycle time through the waypoints, this vehicle must then solve a traveling salesman problem (TSP), i.e., find the shortest tour visiting these waypoints. This problem is especially important since often the waypoints have to be visited in a periodic fashion and any improvement on the length of the tour can have large performance benefits over the long term. Good tours can also be used to design open-loop (polling) scheduling strategies in a dynamic environment, by additionally specifying the time to stay at each waypoint location.

To cope with the difficulty of designing controllers for autonomous robots, a layered architecture such as the one shown on fig. 1-2 is usually employed. A mission planning level typically solves combinatorial problems such a the traveling salesman problem, and the executive layer is in charge of designing a trajectory following the commands sent by the mission planner. However, in the most naive implementation, the decision making level does not have a model of the dynamics of the vehicle, and the motion planner might have great difficulties executing the plan on the physical system. Considering the traveling salesman problem for a Dubins vehicle, we show that important performance gains can be expected by improving the interaction between the decision making level and the motion planner.

The Dubins vehicle is a useful approximation of the dynamics of fixed wing UAVs. This vehicle can only move forward with constant velocity in the plane, and has a limited turning radius. At the

same time, the model is simple enough to have a complete characterization of the optimal point-to-point trajectories. The Dubins TSP (DTSP) was recently considered implicitly in [124], and then more thoroughly in [109, 108, 110, 104, 88, 59]. The methods recently proposed to solve this problem with a deterministic set of waypoints all build on a waypoint ordering computed without taking into account the dynamics of the vehicle, enforcing in effect the separation of the controller into distinct layers. Typically this ordering is obtained by solving the TSP for the Euclidean metric (ETSP). The methods then focus on the design of an admissible trajectory following this ordering. Note that if the ordering is fixed, designing a shortest curve through the waypoints has a 5.03-approximation described earlier in [84] and running in linear time.

In chapter 2, we show that the DTSP is NP-hard, justifying the work on approximation algorithms and heuristics. Moreover, we show that choosing the ordering of points optimal for the Euclidean TSP cannot lead to an approximation ratio better than $\Omega(n)$. We describe an algorithm based on choosing the headings or a set of possible headings at each point first, and then solving a generalized asymmetric TSP to find the ordering. This approach follows naturally from the previous contributions to non-holonomic motion planning for the shortest path problem, see e.g. [60]. Numerical simulations confirm the improved practical performance of this approach over ETSP-based algorithms, when the turning radius of the vehicle is of the same order of magnitude as the inter-waypoint distances.

In fact, there is currently no algorithm that returns, for any instance of the DTSP, a tour within a factor $f(n)$ of the optimum, for f a function independent of the geometry of the instance. For all proposed algorithms so far, f scales as the inverse of the minimum Euclidean distance between any two waypoints of the instance. A complete enumeration of all permutations of the waypoints followed by the determination of the headings using the linear algorithm of [84] provides a constant factor approximation, which however runs in time at least $(n \cdot n!)$. We describe another algorithm achieving a $O(\log n)$ approximation ratio, which runs in time $O((n \log n)^2 \cdot 2^n)$ in the worst case. Obtaining an approximation algorithm running in polynomial time is still an open problem.

1.3.2 Dynamic Scheduling and Bandit Problems

Suppose now that the environment is dynamic and that as time passes, task requests appear at the location of the waypoints. Several ways of handling the dynamic nature of the associated performance optimization problems are possible. A solution to the traveling salesman problem can be used as a building block for open-loop policies scheduling the activities of the vehicles, in association with periodic re-optimization. The TSP tour is executed periodically and the policy only decides how much time should be spent at each location. Optimizing the length of these visit times is performed at the beginning of each cycle, assuming that the state of the environment remains fixed for the length of the cycle, or even only once at the beginning of the mission based on the available information such as task mean inter-arrival and service times. Such analysis can be found in the abundant literature on polling models [123, 138].

In this thesis, we focus instead on closed-loop policies, which continuously determine the behavior of the vehicles based on the knowledge acquired so far about the environment. In chapters 4 to 6, we make the assumption that a probabilistic description of the environment dynamics is available and consider the problem of allocating tasks through time to a set of vehicles performing an ISR mission. We consider a set of N sites that must be inspected by M vehicles. We assume that the states of these sites evolve independently as Markov processes with known transition probabilities, and that rewards can be collected by the vehicles visiting certain sites in certain states. We take as a starting point the multi-armed bandit problem (MABP), which is one of the few instances of a dynamic stochastic scheduling problem with an efficiently computable optimal solution [47]. The

MABP corresponds to the case where $M = 1$ and the states of the sites that are not inspected remain idle. A more interesting model for our purpose is the restless bandit problem (RBP) [136], which relaxes these assumptions, but quickly leads to intractable computational problems [101]. Chapter 3 provides background material on the MABP and the RBP.

The thesis contributes to the literature on stochastic scheduling and restless bandit problems, and the application of these tools to sensor scheduling, in chapters 4, 5 and 6. We consider two main models relevant to unmanned vehicle routing:

- *Dynamic scheduling for problems with partial observations* (chapters 4 and 5). The information available to an UAS is usually limited to the information gathered by the vehicles themselves. To develop our understanding of this situation, we consider first in chapter 4 a discrete-time model where the sites evolve as independent two-state Markov chains. This evolution is independent of the action of the vehicles. A vehicle visiting a site in the first state receives a fixed reward, and no reward if the site is in the second state. Moreover, at each period the UAS can only observe the state of the sites where a vehicle is present, and estimate the state of the remaining sites. Hence there is trade-off between visiting a site to collect a reward present with high probability and visiting sites to update the state estimates. For this problem, which is a particular case of the RBP, we can compute in closed form an index policy proposed by Whittle, and compute efficiently an upper bound on the achievable performance. Numerical simulations seem to show that the index policy performs very well, and is possibly asymptotically optimal in the limit $N \to \infty$ with the ratio M/N constant. Note that recently the same model was considered by Guha et al. [53]. They showed that a slightly more general version of this problem is NP-hard, and gave a 2-approximation for the average-cost version of the problem, using a different policy, which requires more computations. More references on the use of the MABP and RBP for other sensor scheduling problems are given in the introduction of chapter 4 and 5.

  This problem is a type of sequential detection problem. In chapter 5, we consider an estimation problem where the N sites evolve in continuous time as N independent Gaussian linear time invariant (LTI) systems. The M mobile sensors are used to keep the best possible estimate of the states of the sites, neglecting the time it takes for them to travel between sites. It is well known that this problem reduces to a deterministic optimal control problem, which however is not easy to solve in general [91, 8]. We apply the ideas developed for the RBP to this problem. In the case of one-dimensional systems with identical sensors, Whittle's index policy is computable in closed form and experimentally its performance matches the standard restless bandit bound on achievable performance. We then consider the multidimensional case, and a generalization to non-identical sensors. We show how the generalization of the performance bound can be computed by a convex program involving linear matrix inequalities whose size grows as a polynomial of the size of the instance of the problem. We also build from the computation of this bound a family of new switching policies and prove that their steady-state performance approaches the bound arbitrarily closely.

- *Restless bandits with switching costs* (chapter 6). In chapters 4 and 5 we do not take the path planning problem into account, neglecting the time it takes for a vehicle to switch from one site to another. This is done in order to obtain clean results; the policies could then provide heuristics for more realistic scenarios involving switching times or costs. In chapter 6, we take a closer look at the problem and add switching costs penalizing the travel of vehicles between the sites to inspect. Adding switching costs even to the MABP destroys the optimality of the classical index based solution. In fact, the basic problem of optimally

allocating a server between two queues with exponential service times and a switching cost is still open. Chapter 6 adopts a computational approach to the restless bandit problem with switching costs (RBPSC). Extending a technique previously proposed by Bertsimas and Niño-Mora for the RBP [17], we develop a linear programming relaxation for the RBPSC. We then propose a heuristic built on the information extracted from the relaxation, and provide computational simulations comparing the performance of the heuristic to the performance bound given by the relaxation. Note that we do not expect to obtain an a priori approximation factor for such a general problem, since it is PSPACE-hard to approximate the RBP within any factor, unless some restriction on the rewards is imposed.

1.3.3 Continuous Path Planning for a Data Harvesting Mobile Server

When we approach the UAS performance optimization problems using the discrete models above, the considerations about the design of the vehicle trajectories are limited to specifying which sites should be visited at each period. Moreover, we assume in chapters 4, 5 and 6, as is typically done in the vehicle routing literature, that a task requested at a site requires a vehicle to be positioned exactly at that site. For typical missions performed by UAS, such as gathering data, performing remote measurements, or relaying communications, this requirement should be relaxed. We would like to refine the models by superposing to the scheduling problems a trajectory optimization problem in continuous space.

In chapter 7, we focus on the trajectory optimization problem for an UAV acting as a communication relay or performing measurements remotely. There are N sites transmitting information to the vehicle, either actively in a communication scenario or passively in a reconnaissance scenario. The transmission rate for each bit of information depends on the position of the vehicle with respect to the sites, see fig. 1-3. In a more classical setting, such as in Klimov model or in polling systems, we would have restricted the server to move between the site locations only. This does not necessarily provide the best performance in our situation, where any intermediate position is available.

Given the complexity of the computations in chapter 6 for the approach based on Markov decision processes, we resort to model approximations from the start. We consider a deterministic fluid approximation to obtain insight into this problem. We provide a necessary and sufficient stabilizability condition for the fluid model, that is valid for any set of spatial service rate functions. For the case of two sites, we consider the design of server trajectories to drain optimally a given set of jobs present at the sites initially when no further job arrivals occur. A feedback form solution is given for some piecewise linear service rate functions, and we test the policy with good results on a more realistic model assuming stochastic job arrivals.

1.4 Outline

In chapter 2 we consider the traveling salesman problem for one Dubins vehicle. This is an example where an automated planner can optimize the performance of the vehicle in an environment assumed to be static. Starting with chapter 3, the environment is assumed to be dynamic and our goal is to design mission planners that incorporate the information gathered with time. Chapter 3 itself provides some background on Bandit problems and the associated computational tools. Chapter 4 and 5 consider the dynamic planning problem for mobile sensors using the restless bandit approach, in a sequential detection scenario and an estimation scenario respectively. In chapter 6 we treat a RBP with additional switching costs penalizing switching between the sites. This is intended to capture the path-planning component and avoid excessive site switching in the sequential allocation

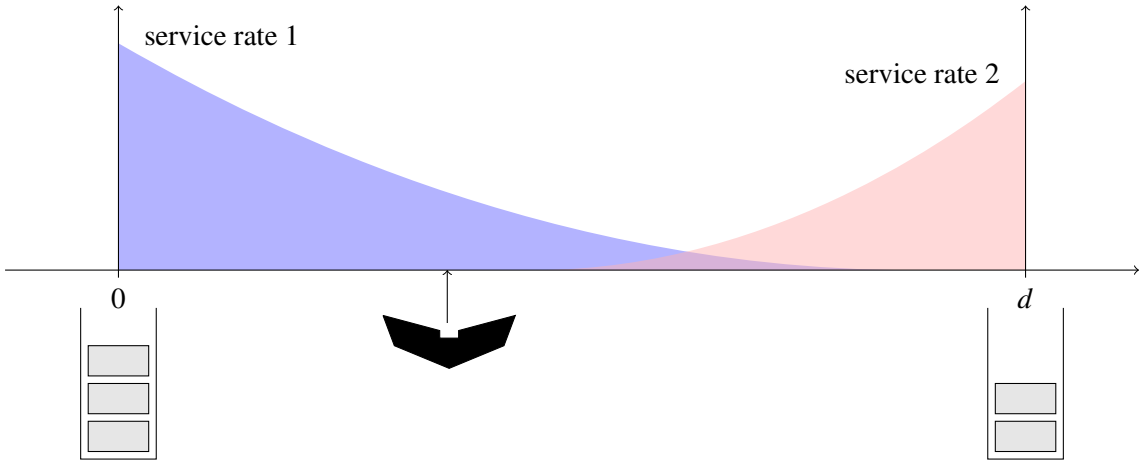


Figure 1-3: Uplink from two fixed stations to a mobile receiver, with spatially decaying service rates.

policies obtained previously. Finally in chapter 7 we account for a continuous environment where the typical surveillance tasks of an UAV can be performed remotely and for several sites simultaneously. The fluid model approximation that we propose is a flexible tool that could be useful to obtain insight into complicated UAS mission planning problem. Chapter 8 concludes the thesis and outlines some research topics for future work.

Chapter 2

Static Path Planning: The Curvature-Constrained Traveling Salesman Problem

A basic tool needed by an UVS operator is a way of generating the trajectory of a vehicle automatically once he has specified a set of target locations that the vehicle must inspect. Often, the order in which the sites are to be inspected is not specified a priori and we should design the tour of minimum length visiting the waypoints, or equivalently the minimum time tour if the velocity of the vehicle is constant. Gains in performance on this problem can have a large impact since the tour might have to be executed repetitively during the duration of a surveillance mission. It is also a building block for static scheduling policies where, once an order of the waypoints has been chosen to minimize the total travel time, we can specify the time to spend at each site for inspection. Later chapters will consider closed-loop scheduling policies that decide dynamically which target to visit next based on the information gathered about the environment during the mission.

The static waypoint ordering problem minimizing the travel distance is an instance of the well known traveling salesman problem (TSP). However, existing algorithms for the TSP must be adapted to include the dynamics of the vehicles if good performance is to be expected. We consider the TSP for a type of vehicle subject to differential constraints, namely the Dubins vehicle, which is a good model for fixed wing UAVs. We first provide a proof of the NP-hardness of the TSP for the Dubins vehicle, justifying our subsequent work on approximate solutions. We also provide a lower bound of $\Omega(n)$ on the approximation ratio achievable by any algorithm following the ordering optimal for the Euclidean metric, which is an often used heuristic.

2.1 Introduction

In an instance of the traveling salesman problem (TSP) we are given the distances d_{ij} between any pair of n points. The problem is to find the shortest tour visiting every point exactly once. We also call this problem the tour-TSP to distinguish it from the path-TSP, where the requirement that the vehicle must start and end at the same point is removed. This famously intractable problem is often encountered in robotics and typically solved by the higher decision-making levels in the common layered controller architectures. The dynamics of the robot are usually not taken into account at this stage and the mission planner might typically chose to solve the TSP for the Euclidean metric (ETSP), i.e., the distances d_{ij} represent the Euclidean distances between waypoints. This can result in poor overall performance, since the sequence of points chosen by the algorithm can be very hard

to follow for the physical system.

In order to improve the performance of unmanned aerial systems in particular, we need to integrate the high level problem of mission planning and the low level problem of path planning [124]. We obtain a motion planning problem with differential constraints, for which most of the existing work concerns the shortest path problem, see, e.g. [79]. In theory, the TSP with differential constraints could be formulated as an optimal control problem and solved numerically, for example using a mixed-integer linear program [113]. However this procedure is already complex if the sequence of waypoints is specified, and so directly adding a hard combinatorial problem to it is practical only for a very small number of points.

There has been in the past few years some interest in designing tractable algorithms for the traveling salesman problem using simplified dynamic models. In this chapter we consider the traveling salesman problem for the Dubins vehicle (DTSP). The Dubins vehicle can only move forward in the plane, at constant speed, and has a limited turning radius. This model can generate paths that are almost feasible for fixed-wing aircraft, and therefore provides a waypoint sequence that is relevant for the low-level controller of these vehicles tracking the designed trajectories. At the same time, for this model we have for example an analytical characterization of the shortest path between any two configurations, giving hope that efficient algorithms with good performance can be designed to solve the TSP.

A stochastic version of the DTSP for which the points are distributed randomly and uniformly in the plane was considered in [108, 110, 41, 59]. Here however, we concentrate on algorithms and worst-case bounds for the more standard problem where no waypoint input distribution is given. In that case, most of the existing algorithms seem to build on a preliminary solution obtained for the ETSP [109, 104, 88]. We refer the reader to our more detailed survey of the existing algorithms in [82]. Note that we do not impose a lower bound on the minimum Euclidean distance between waypoints. Since the turning radius of an aircraft is proportional to the square of its velocity and high-speed operating conditions can be required, in particular in adversarial environments, situations where the points are densely distributed compared to the turning radius are clearly of interest. For example, an UAV flying at a velocity of 200 km/h and engaging a turn with a 40° bank angle has a turning radius of almost 1 km. In this case the dynamics of the vehicle become really relevant and require new solution methods not developed for the ETSP.

Contributions of this chapter. We first prove that the DTSP is NP-hard, thus justifying the work on heuristics and algorithms that approximate the optimal solution. On the negative side, we give some lower bounds on the approximation ratio achievable by recently proposed heuristics. Following a tour based on the ETSP ordering or the ordering of Tang and Özgüner [124] cannot achieve an approximation ratio better than $\Omega(n)$[1]. The same is true for the nearest neighbor heuristic, in contrast to the ETSP where it achieves a $O(\log n)$ approximation [106]. Then we propose an algorithm based on heading discretization, which emerges naturally from the previous work on the curvature-constrained shortest path. Its theoretical performance is of the same type as for these shortest-path algorithms and does not improve on the previously mentioned heuristics. However numerical simulations show a significant performance improvement in randomly generated instances over other heuristics when the inter waypoint distances are smaller than the turning radius of the vehicle.

The rest of the chapter is organized as follows. We recall some facts about the Dubins paths in section 2.2 and reduce the DTSP to a finite dimensional optimization problem. This section also contains the lower bound on approximation ratios for various heuristics. In Section 2.3 we show that the DTSP is NP-hard. Section 2.4 describes two algorithms that are not based on a

[1] We say $f(n) = O(g(n))$ if there exists $c > 0$ such that $f(n) \leq cg(n)$ for all n, and $f(n) = \Omega(g(n))$ if there exists $c > 0$ such that $f(n) \geq cg(n)$ for all n.

solution for the ETSP. The first, based on heading discretization, returns in time $O(n^3)$ a tour within $O\left(\min\left(\left(1+\frac{\rho}{\varepsilon}\right)\log n, \left(1+\frac{\rho}{\varepsilon}\right)^2\right)\right)$ of the optimum, where ρ is the minimum turning radius of the vehicle and ε is the minimum Euclidean distance between any two waypoints. The second returns a tour within $O(\log n)$ of the optimum, but is not guaranteed to run in polynomial time. Finally, section 2.5 presents some numerical simulations.

2.2 Heuristics Performance

A Dubins vehicle in the plane has its configuration described by its position and heading $(x, y, \theta) \in \mathbb{R}^2 \times (-\pi, \pi]$. Its equations of motion are

$$\dot{x} = v_0 \cos(\theta), \quad \dot{y} = v_0 \sin(\theta), \quad \dot{\theta} = \frac{v_0}{\rho} u, \quad \text{with } u \in [-1, 1].$$

Without loss of generality, we can assume that the speed v_0 of the vehicle is normalized to 1. ρ is the minimum turning radius of the vehicle. u is the available control. The DTSP asks, for a given set of points in the plane, to find the shortest tour through these points that is feasible for a Dubins vehicle. Since we show below that this problem is NP-hard, we will focus on the design of approximation algorithms. An α-approximation algorithm (with $\alpha \geq 1$) for a minimization problem is an algorithm that produces *in polynomial time*, on any instance of the problem with optimum OPT, a feasible solution whose value Z is within a factor α of the optimum, i.e., such that $OPT \leq Z \leq \alpha\, OPT$.

Dubins [38] characterized curvature constrained shortest paths between an initial and a final configuration. Let P be a feasible path. We call a nonempty subpath of P a C-segment or an S-segment if it is a circular arc of radius ρ or a straight line segment, respectively. We paraphrase the following result from Dubins:

Theorem 2.2.1 ([38])**.** *An optimal path between any two configurations is of type CCC or CSC, or a subpath of a path of either of these two types. Moreover the length of the middle arc of a CCC path must be of length greater than* $\pi\rho$.

In the following, we will refer to these minimal-length paths as Dubins paths. When a subpath is a C-segment, it can be a left or a right hand turn: denote these two types of C-segments by L and R respectively. Then we see from Theorem 2.2.1 that to find the minimum length path between an initial and a final configuration, it is enough to find the minimum length path among six paths, namely among $\{LSL, RSR, RSL, LSR, RLR, LRL\}$. The length of these paths can be computed in closed form and, therefore, finding the optimum path and length between any two configurations can be done in constant time [118].

Given this result, solving the DTSP is reduced to choosing a permutation of the points specifying in which order to visit them, as well as choosing a heading for the vehicle at each of these points. If the Euclidean distance between the waypoints to visit is large with respect to ρ, the choice of headings has a limited impact and the DTSP and ETSP behave similarly. This has motivated work using for the DTSP the waypoint ordering optimal for the ETSP [109, 104, 88], and concentrating on the choice of headings. Theorem 2.2.2 provides a limit on the performance one can achieve using such a technique, which becomes particularly significant in the case where the points are densely distributed with respect to ρ.

Before stating the theorem, we describe another simple heuristic, called the nearest neighbor heuristic. We start with an arbitrary point, and choose its heading arbitrarily, fixing an initial configuration. At each step, we find the point not yet on the path which is closest for the Dubins metric to

the last added configuration (i.e., position and heading). This is possible since we also have a complete characterization of the Dubins distance between an initial configuration and a final point with free heading [22]. We add this closest point with the associated optimal arrival heading and repeat the procedure. When all nodes have been added to the path, we add a Dubins path connecting the last configuration and the initial configuration. It is known that for the ETSP, which is a particular case of the symmetric TSP, the nearest neighbor heuristic achieves a $O(\log n)$ approximation ratio [106].

Theorem 2.2.2. *Any algorithm for the DTSP which always follows the ordering of points that is optimal for the ETSP has an approximation ratio R_n which is $\Omega(n)$. If we impose a lower bound ε sufficiently small on the minimum Euclidean distance between any two waypoints, then there exist constants C, C', such that the approximation ratio is not better than $\frac{C}{\varepsilon+(C'/n)}$. These statements are also true for the nearest neighbor heuristic.*

Proof. Consider the configuration of points shown in fig. 2-1(a). Let n be the number of points, and suppose $n = 4m$, m an integer. For clarity we focus on the path-TSP problem but extension to the tour-TSP case is easy, by adding a similar path in the reverse direction. The optimal Euclidean path-TSP is shown on the figure as well. Suppose now that a Dubins vehicle tries to follow the points in this order, and suppose ε is sufficiently small. Then for each sequence of 4 consecutive points, two on the upper line and two on the lower line, the Dubins vehicle will have to execute a maneuver of length at least C, where C is a constant of order $2\pi\rho$. For instance, if the vehicle tries to go through the two top points without a large maneuver, it will exit the second point with a heading almost horizontal and will have to make a large turn to catch the third point on the lower line. This discussion can be formalized using the results on the final headings of direct Dubins paths from [84]. Hence the length of the Dubins path following the Euclidean TSP ordering will be greater than mC.

On the other hand, a Dubins vehicle can simply go through all the points on the top line, execute a U-turn of length C' of order $2\pi\rho$, and then go through the points on the lower line, providing an upper bound of $2n\varepsilon + C'$ for the optimal solution. So we deduce that the worst case approximation ratio of the algorithm is at least:

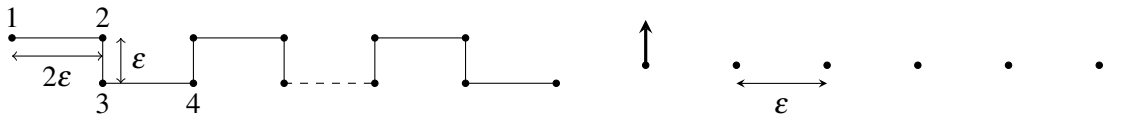
$$R_n \geq \frac{\frac{n}{4}C}{2n\varepsilon + C'}.$$

But we can choose ε as small as we want, in particular we can choose $\varepsilon \leq 1/n$ for problem instances with n points, and thus $R_n = \Omega(n)$.

For the nearest-neighbor heuristic, we use the configuration shown on fig. 2-1(b), for ε small enough. The figure shows the first point and heading chosen by the algorithm. The proof is similar to the argument above and left to the reader. It essentially uses the fact that points inside the circles of radius ρ tangent to the initial heading of a Dubins path starting at a point P are at Dubins distance greater than $\pi\rho$ from P [22]. □

Remark 2.2.3. In [124], Tang and Özgüner do not use the ETSP ordering, but instead, construct the geometric center of the waypoints, calculate the orientations of the waypoints with respect to that center, and traverse the points in order of increasing orientations. It is easy to adapt Theorem 2.2.2 to this case, closing the path of fig. 2-1(a) into a cycle for example.

Remark 2.2.4. Using the notation of the theorem, the "Alternating Algorithm" proposed in [109] achieves an approximation ratio of approximately $1 + 3.48\frac{\rho}{\varepsilon}$ [110].



(a) Waypoint configuration used to analyze the performance of ETSP based heuristics.

(b) Configuration used to analyze the performance of the nearest-neighbor heuristic.

Figure 2-1: Waypoint configurations used as counter-examples in theorem 2.2.2.

Remark 2.2.5. If the waypoint ordering is decided a priori, for example using the solution for the ETSP, [84] describes a linear time algorithm to choose the headings which provides a tour with length at most 5.03 times the length of the tour achievable by a choice of headings optimal for this waypoint ordering.

2.3 Complexity of the DTSP

It is usually accepted that the DTSP is NP-hard and the goal of this section is to prove this claim rigorously. Note that adding the curvature constraint to the Euclidean TSP could well make the problem easier, as in the bitonic TSP [2] [31, p. 364], and so the statement does not follow trivially from the NP-hardness of the Euclidean TSP, which was shown by Papadimitriou [100] and by Garey, Graham and Johnson [46]. In the proof of theorem 2.3.1, we consider, without loss of generality, the decision version of the problem, which we also call DTSP. That is, given a set of points in the plane and a number $L > 0$, DTSP asks if there exists a tour for the Dubins vehicle visiting all these points exactly once, of length at most L.

Theorem 2.3.1. *Tour-DTSP and path-DTSP are NP-hard.*

Proof. This is a corollary of Papadimitriou's proof of the NP-hardness of ETSP, to which we refer [100]. First recall the Exact Cover Problem: given a family F of subsets of the finite set U, is there a subfamily F' of F, consisting of disjoint sets, such that F' covers U? This problem is known to be NP-complete [68]. Papadimitriou described a polynomial-time reduction of Exact Cover to Euclidean TSP. That is, given an instance of the Exact Cover problem, we can construct an instance of the Euclidean Traveling Salesman Problem and a number L such that the Exact Cover problem has a solution if and only if the ETSP has an optimal tour of length less than or equal to L. The important fact to observe however, is that if Exact Cover does not have a solution, Papadimitriou's construction gives an instance of the ETSP which has an optimal tour of length $\geq (L+\delta)$, for some $\delta > 0$, and not just $> L$. More precisely, letting $a = 20$ exactly as in his proof, we can take $0 < \delta < \sqrt{a^2+1} - a$.

Now from [109], there is a constant C such that for any instance $\mathscr{P}$ of ETSP with n points and length $ETSP(\mathscr{P})$, the optimal DTSP tour for this instance has length less than or equal to $ETSP(\mathscr{P}) + Cn$. Then if we have n points in the instance of the ETSP constructed as in Papadimitriou's proof, we simply rescale all the distances by a factor $2Cn/\delta$. If Exact Cover has a solution, the ETSP instance has an optimal tour of length no more than $2CnL/\delta$ and so the curvature constrained tour has a length of no more than $2CnL/\delta + Cn$. If Exact Cover does not have a solution, the ETSP instance has an optimal tour of length at least $2CnL/\delta + 2Cn$, and the curvature constrained tour as well. So Papadimitriou's construction, rescaled by $2Cn/\delta$ and using $2CnL/\delta + Cn$ instead of

[2] I would like to acknowledge Vincent Blondel for suggesting this example

L, where n is the number of points used in the construction, provides a reduction from Exact Cover to DTSP. □

Remark 2.3.2. It is not clear that DTSP is in NP. In fact, there is a difficulty even in the case of ETSP, because evaluating the length of a tour involves computing many square roots. In the case of DTSP, given a permutation of the points and a set of headings, deciding whether the tour has length less than a bound L might require computing trigonometric functions and square roots accurately.

2.4 Algorithms

2.4.1 An Algorithm Based on Heading Discretization

A very natural algorithm for the DTSP is based on a procedure similar to the one used for the shortest path problem [60]. It consists in choosing a priori a finite set of possible headings at each point. Suppose for simplicity that we choose K headings for each point. We construct a graph with n clusters corresponding to the n waypoints, and each cluster containing K nodes corresponding to the choice of headings. Then, we compute the Dubins distances between configurations corresponding to pairs of nodes in distinct clusters. Finally, we would like to compute a tour through the n clusters which contains exactly one point in each cluster. This problem is called the generalized asymmetric traveling salesman problem, and can be reduced to a standard asymmetric traveling salesman problem (ATSP) over nK nodes [11]. This ATSP can in turn be solved directly using available software such as Helsgaun's implementation of the Lin-Kernighan heuristic [57], using the $\log n$ approximation algorithm of Frieze et al. [45], or the $0.842 \log n$ approximation algorithm of Kaplan et al. [66].

We proposed this method in [80] and in practice we found it to perform very well, see section 2.5. The type of approximation result one can expect with an a priori discretization is the same as in the shortest path case, that is, the path obtained is close to the optimum if the optimum is "robust". This means that the discretization must be sufficiently fine so that the algorithm can find a Dubins path close to the optimum which is of the same type, as defined in section 2.2 (see [3]). A limitation of the algorithm is that we must solve a TSP over nK points instead of n.

2.4.2 Performance Bound for $K = 1$.

Suppose that we choose $K = 1$ in the previous paragraph. Our algorithm can then be described as follows:

1. Fix the headings at all points, say to 0, or by choosing them randomly uniformly in $[-\pi, \pi)$, independently for each point.
2. Compute the $n(n-1)$ Dubins distances between all pairs of points.
3. Construct a complete graph with one node for each point and edge weights given by the Dubins distances.
4. We obtain a *directed* graph where the edges satisfy the triangle inequality. Compute an exact or approximate solution for this asymmetric TSP.

Next we derive an upper bound on the approximation ratio provided by this algorithm. If we look for a polynomial time algorithm, we must be satisfied with an approximate solution to the ATSP in step 4. There is however some additional information that we can exploit once the headings have been fixed, which differentiates the problem from a general ATSP. We have the following:

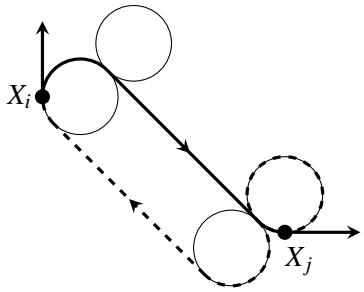


Figure 2-2: A feasible return path for an RSL Dubins path.

Lemma 2.4.1. *Let $\hat{d}_{ij}$ denote the Dubins distance from point $X_i = (x_i, y_i)$ to point $X_j = (x_j, y_j)$ in the plane after the headings $\{\theta_k\}_{k=1}^n$ have been fixed. Then:*

$$\max_{i,j} \frac{\hat{d}_{ij}}{\hat{d}_{ji}} \leq 1 + \frac{4\pi\rho}{\varepsilon},$$

where ε is, for the rest of this section, the minimum Euclidean distance between any two waypoints.

Proof. This result follows from the easy inequality $\hat{d}_{ij} \geq \varepsilon$ and from the following fact:

$$\hat{d}_{ij} \leq \hat{d}_{ji} + 4\pi\rho.$$

To show this, consider the Dubins path from the configuration (X_i, θ_i) to (X_j, θ_j). We construct a feasible path (not necessarily optimal) for the Dubins vehicle, to return from (X_j, θ_j) to (X_i, θ_i). On the forward path, the Dubins vehicle moves along an initial circle (of radius ρ), a straight line or a middle circle, and a final circle. For a CCC path, it is easy to find a return path on the same 3 circles, using the small arc on the middle circle (hence such a return path is not optimal). For the CSC paths, the line is tangent to the following 4 circles: two circles are used as turning circles in the forward path, and the other two circles are symmetric with respect to the line. We consider return paths for the different types of Dubins paths, which use these 4 circles and either the straight line of the forward path or a parallel segment of the same length. The bound is shown to hold in all cases. An example for an RSL path is provided on Fig. 2-2. □

With this bound on the arc distances, we know that there is a modified version of Christofides' algorithm, also due to Frieze et al. [45], which provides a $\frac{3}{2}\left(1 + \frac{4\pi\rho}{\varepsilon}\right)$ approximation for the ATSP in step 4. So to solve the ATSP, we can run the two different algorithms of Frieze et al. and choose the tour with minimum length, thus obtaining an approximation ratio of $\min\left(\log n, \frac{3}{2}\left(1 + \frac{4\pi\rho}{\varepsilon}\right)\right)$. The time complexity of the first three steps of our algorithm is $O(n^2)$. The algorithm for solving the ATSP runs in time $O(n^3)$, so overall the running time of our algorithm is $O(n^3)$. To analyze the approximation performance of this algorithm, we will use the following bound.

Lemma 2.4.2. *Let d_{ij} be the length of the Dubins path between two configurations (X_i, θ_i) and (X_j, θ_j) and let ε_{ij} be the Euclidean distance between X_i and X_j. Let $\hat{d}_{ij}$ be the Dubins distance between the two points after fixing the headings in the first step of the algorithm. If the headings are*

chosen deterministically, we have

$$\hat{d_{ij}} \leq \left(1 + 2\pi\rho \max\left\{\frac{4}{\varepsilon_{ij}}, \frac{7}{3\max(\varepsilon_{ij}, \pi\rho)}\right\}\right) d_{ij} := C_1\, d_{ij}, \tag{2.1}$$

whereas if they are chosen randomly, we obtain

$$E[\hat{d_{ij}}] \leq \left(1 + \frac{13.58\rho}{\varepsilon_{ij}}\right) d_{ij} =: C_2\, d_{ij}. \tag{2.2}$$

Proof. The bound follows from the discussion in section 4.8 of [60][3], which we briefly summarize. The headings at X_i and X_j, as fixed by the algorithm, can differ from the given heading θ_i and θ_j by $\delta_i, \delta_j \in [-\pi, \pi)$. This translates into a path difference

$$|d_{ij} - \hat{d}_{ij}| \leq \rho \left[\max\{A_i, B_i\} + \max\left\{A_j, B_j\right\}\right],$$

where

$$A_i = 3|\delta_i| + \pi|\sin(\frac{\delta_i}{2})|,\ B_i = |\delta| + 4\arccos\left(1 - \frac{|\sin(\delta/2)|}{2}\right),$$

and we substitute δ_i to δ_j to get A_j, B_j. The A and B terms come from the analysis of perturbations of CSC paths with opposite initial and final turning directions and CCC paths respectively. Perturbations of a CSC path with identical initial and final turning directions are dominated by the A terms. To obtain (2.1), we take the worst possible values, $\delta_i = \delta_j = -\pi$. Moreover, for the second part of the max term, the fact that a CCC optimal path must have minimum length $\pi\rho$ was used. The bound (2.2) follows from the fact that in the randomized heading assignment, the perturbations δ_i, δ_j are uniformly distributed in $[-\pi, \pi)$, and we can then compute

$$E\left[\max\{A_i, B_i\} + \max\left\{A_j, B_j\right\}\right] = 13.58.$$

□

The following theorem then describes the approximation ratio of our algorithm.

Theorem 2.4.3. *Given a set of n points in the plane, the algorithm described above with $K = 1$, returns a Dubins traveling salesman tour with length within a factor $O\left(\min\left(\left(1 + \frac{\rho}{\varepsilon}\right)\log n, \left(1 + \frac{\rho}{\varepsilon}\right)^2\right)\right)$ of the length of the optimal tour. More precisely, the deterministic choice of headings guarantees an approximation ratio of*

$$\left(1 + 2\pi\rho \max\left\{\frac{4}{\varepsilon_{ij}}, \frac{7}{3\max(\varepsilon_{ij}, \pi\rho)}\right\}\right) \min\left(\log n, \frac{3}{2}\left(1 + \frac{4\pi\rho}{\varepsilon}\right)\right),$$

whereas the randomized choice of headings provides an approximation ratio of

$$\left(1 + \frac{13.58\rho}{\varepsilon}\right) \min\left(\log n, \frac{3}{2}\left(1 + \frac{4\pi\rho}{\varepsilon}\right)\right)$$

in expectation. The running time of this algorithm is $O(n^3)$.

Proof. Call *OPT* the optimal value of the DTSP and σ^* the corresponding optimal permutation specifying the order of the waypoints. We have $OPT = \sum_{i=1}^{n-1} d_{\sigma^*(i)\sigma^*(i+1)} + d_{\sigma^*(n)\sigma^*(1)} := L(\{d_{ij}\}, \sigma^*)$.

[3] It appears that eq. (4.38) and (4.48) in [60] contain a typographical error, namely a ρ is missing.

Considering the permutation σ^* for the graph problem (where the edge weights are the distances $\{\hat{d}_{ij}\}$) and $\hat{\sigma}^*$ the optimal permutation for the graph problem, we have

$$L(\{\hat{d}_{ij}\}, \hat{\sigma}^*) \leq L(\{\hat{d}_{ij}\}, \sigma^*) \leq C\, L(\{d_{ij}\}, \sigma^*),$$

with C equal to C_1 or C_2 as defined in lemma 2.4.2. We do not obtain the optimal permutation for the ATSP on the graph in general, instead we use the approximation algorithm mentioned above. Calling $\hat{\sigma}$ the permutation obtained, we have:

$$\begin{aligned} L(\{\hat{d}_{ij}\}, \hat{\sigma}) &\leq \min\left(\log n, \frac{3}{2}\left(1+\frac{4\pi\rho}{\varepsilon}\right)\right) L(\{\hat{d}_{ij}\}, \hat{\sigma}^*) \\ &\leq \left(C \min\left(\log n, \frac{3}{2}\left(1+\frac{4\pi\rho}{\varepsilon}\right)\right)\right) OPT. \end{aligned}$$

□

Remark 2.4.4. For better performance in practice, one should try to run the randomized algorithm several times and choose the shortest tour obtained.

We note that the bound provided is in fact worse than the bound available for the "Alternating Algorithm" (AA), see remark 2.2.4. However, this bound does not seem to be tight, and moreover, experiments suggest that the randomized heading assignment in fact performs significantly better in general than the AA when the waypoint distances are smaller than the turning radius (see section 2.5).

2.4.3 On the ρ/ε Term in Approximation Ratios

Every approximation ratio mentioned so far contains a term $(1+\rho/\varepsilon)$, where ε is the minimum Euclidean distance between any two waypoints. As seen in theorem 2.2.2, this term is unavoidable for any algorithm based on the ETSP optimal ordering. It is particularly problematic when waypoints are distributed densely with respect to the turning radius, although it seems in general too pessimistic for randomly distributed waypoints.

An algorithm that returns tours within a constant factor of the optimum certainly exists. A naive algorithm can enumerate all permutations of waypoints, and optimize the headings for each permutation. More precisely, one can use the linear time, 5.03-approximation algorithm of Lee et al. [84] to choose the headings for a given permutation. Taking the best tour, one obtains a constant factor approximation, but of course the algorithm runs in time at least $n \cdot n!$ (with the method described in [73, chap. 1] to enumerate the permutations, we can have an algorithm that runs in time $n^2 \cdot n!$).

We would like to understand if one can design better algorithms with approximation ratios that are independent of ε. In this paragraph, we describe an algorithm that improves on the time complexity of the naive algorithm, although it does not achieve polynomial running time and provides only a $O(\log n)$ approximation ratio. The technique combines ideas from [84, 4] to start by finding chains of points which are almost aligned, as described in lemma 2.4.6.

Assume without loss of generality that $\rho = 1$, by rescaling distances. For two points X and Y such that $\mathrm{dist}(X,Y) < 2$, we can draw two unit circles passing through X and Y, see fig. 2-3. Following [84], a path from X to Y included in the intersection of the two circles is called a *direct path*. We call this region the region of direct paths between X and Y. Otherwise it is called a *detour path*. It is also shown in [84] that the length of any detour path is at least π, and that given an

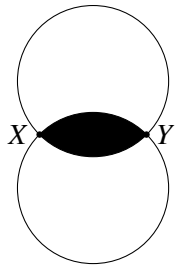


Figure 2-3: A path included in the dark region is called a direct path.

interval I of initial headings for a direct path from X to Y, the set of possible final headings at Y is also an interval (of size less than π), and can be computed from I in constant time. We extend these definitions to paths going through multiple points. A path going through points $p_1, p_2, \ldots, p_m$ will be denoted $(p_1, p_2, \ldots, p_m)$. A path $(p_1, p_2, \ldots, p_m)$ is called a direct path if the Euclidean distance $d(p_1, p_m)$ is strictly less than 2, and the Dubins length of the path is at most π. Otherwise it is called a detour path. It is clear that all the Dubins paths between two points p_i, p_j, $i < j$, in a direct path $(p_1, \ldots, p_m)$ are then direct paths.

Let us fix a point s. We define a relation $\prec_s$ between points distinct from s by $p \prec_s q$ if there exists a direct path (s, p, q). Let us start with the following preliminary result.

Lemma 2.4.5. *The relation $\prec_s$ is a strict order relation.*

Proof. If $p \prec_s q$, then p is in the region of direct paths between s and q, and we see from figure 2-3 that the region of direct paths between s and p is contained strictly in the region of direct paths between s and q (just let $X := s$ on the figure and rotate the circles around s until they both pass through p). The property

$$p \prec_s q \Rightarrow \neg(q \prec_s p)$$

easily follows. For the transitivity property

$$(p \prec_s q) \wedge (q \prec_s r) \Rightarrow (p \prec_s r),$$

we use the same fact to see that p belongs to the region of direct paths between s and r. Then one can see from elementary geometry that there exists a Dubins path (s, p, r).

□

Lemma 2.4.6. *There is an algorithm that finds a maximum cardinality direct Dubins path through a given a set of waypoints, in time $O(n^2 \log n\, 2^n)$.*

Proof. Let us fix a point s. We then choose k other points $p_{i_1}, p_{i_2}, \ldots, p_{i_k}$, starting with $k = n-1$, until the following procedure succeeds. We want to check if there is a direct path starting at s and going through these k points. Consider the order relation $\prec_s$. This relation can be verified between two points in constant time since it requires only verifying that a point belongs to the region of fig. 2-3. When verifying the relation between two points in the following, if $p \prec_s q$ fails, we always test if $q \prec_s p$. We use this order relation to sort the points $p_{i_1}, \ldots, p_{i_k}$, which can be done in time $k \log k$. During this process, we stop immediately if we find two points that are not comparable, since we then know that there is not direct path starting at s through the k points. If the sort succeeds, this produces a direct path starting at s through the k points. Continuing likewise for each set of waypoints and all k, we can find a maximum cardinality direct path starting from s in time at most

$n \log n \, 2^n$. Repeating the procedure for each s, we can find a maximum cardinality direct path of the instance in time at most $n^2 \log n \, 2^n$.

□

The next lemma is as in the proof of theorem 2.3 in [4].

Lemma 2.4.7. *Given a set of n points $\mathscr{P}$, with optimal Dubins tour τ^* with length L^* such that $L^* \leq \pi K$, we can partition $\mathscr{P}$ into r groups of points $\{C_1, \ldots, C_r\}$, where the points in each C_i can be linked by a direct Dubins path, and such that $r \leq K \log n$. This partition can be found in time $O((n \log n)^2 \, 2^n)$.*

Proof. By induction on n. The case $n = 1$ is trivial. Now for the induction step, notice that τ^* contains a subpath of cardinality at least n/K with length at most π. So such a subpath must be a direct path. Using lemma 2.4.6, we can find such a subpath, and we let the points on this path form C_1. Consider now the instance $\mathscr{P}'$ with n' points obtained by deleting the points of C_1 from $\mathscr{P}$. Clearly the length of the optimal tour for $\mathscr{P}'$ is less than πK. Also, $K \log n' \leq K \log \left[n \left(1 - \frac{1}{K} \right) \right] \leq K \log n - 1$. By the induction hypothesis, we can find a partition for $\mathscr{P}'$ consisting of at most $r' \leq K \log n'$ sets C_i, $i = 2, \ldots, r+1$. Then we add C_1. □

Proposition 2.4.8. *There is an algorithm which gives a $O(\log n)$ approximation for the DTSP in time $O((n \log n)^2 \cdot 2^n)$.*

Proof. Given a set of n waypoints, let us call L^* the minimum length of a Dubins tour through these points. Define $K = \lceil L^* / \pi \rceil$. By lemma 2.4.7, we can construct a partition of the points into r direct Dubins paths $\{C_1, \ldots, C_r\}$, with $r \leq K \log n$. We can close the Dubins paths on the sets C_i to obtain r cycles of length $L_i \leq 2\pi + 2\pi =: \kappa_1$, which we also call C_i. If we can find two waypoints $p_k \in C_k$, $p_l \in C_l$, $k \neq l$, with Euclidean distance $\|p_k - p_l\|_2 \leq 2$, then we can merge C_k and C_l to obtain a new cycle for the Dubins vehicle, of length at most $L_k + L_l + \kappa_2$, for some fixed constant κ_2. A value of κ_2 can be obtained from the discussion in [109] for example. We can thus merge all the cycles which have close waypoints. Finally, we are left with cycles such that points on distinct cycles are at Euclidean distance at least 2 apart. Choose arbitrarily a point on each cycle and for this subset of points, compute a Dubins tour using the alternating algorithm. We get a cycle of length less than $2.74 \, L^*$ joining the previous cycles (take $\varepsilon = 2$ in remark 2.2.4). From this configuration, we can finally create a Dubins tour through all the points using the same cycle merging procedure as in the algorithm of Frieze et al. for the ATSP [45]: as long as a point has two incoming and outgoing Dubins paths belonging to 2 different cycles, we can short-circuit one incoming-outgoing pair of paths, incurring each time at most a fixed penalty κ_3. At the intermediate step before the alternating algorithm, the total length of the cycles is at most $(\kappa_1 + \kappa_2) r$, and there are at most r of these cycles. We join them via a cycle of length $2.74 L^*$, and the merging procedure adds a penalty of at most $\kappa_3 r$. So the length of the resulting Dubins tour is bounded by

$$(\kappa_1 + \kappa_2 + \kappa_3) r + 2.74 L^* \leq \kappa_4 L^* \log n.$$

Hence overall the procedure produces a tour with length $O(L^* \log n)$. □

2.5 Numerical Simulations

Fig. 2-4 presents simulation results comparing the practical performance of different algorithms proposed for the DTSP. Points are generated randomly in a 10×10 square. All the TSP tours (symmetric and asymmetric) are computed using the software LKH [57]. We compare the performance

of the Alternating Algorithm (AA) [109], the nearest neighbor heuristic (NN) of section 2.2, and the algorithm described in section 2.4.1. AA is based on the optimal Euclidean ordering. It computes the optimal ETSP, and keeps every other edge in the loop. These edges are straight lines followed by the vehicle, which must then connect the end of a straight path with the start of the next one by a Dubins path. Compared to our randomized heading algorithm, we see that the performance of AA is similar for low point densities. In fact, AA clearly outperforms the randomized heading algorithm if the points are very sparsely distributed and the optimal tour tends to become the same as the optimal Euclidean tour. However, in scenarios with waypoints densely distributed with respect to the vehicle turning radius, which arise for example for UAVs in urban environments, or loitering weapons flying at high speed [70], large performance gains can be obtained by our algorithms over the AA.

Moreover for the choice of heading, even with just one discretization level, we can use the same heading as AA, and compute the solution of the ATSP (as in step 4 of paragraph 2.4.2) using this heading. This clearly always performs at least as well as AA (at least if the TSP are solved exactly), and the figure shows the significant performance improvement due to only changing the ordering, as the number of points increases. Also shown on the figure are the performance curves for an increasing number of discretization levels. With 5 discretization levels, a tour through 100 points can be computed on a standard laptop in about one minute, requiring the solution of an ATSP with 500 points, well below the limits of state-of-the-art TSP software.

In [41], the authors derive the following asymptotic lower bound on the length of the DTSP with waypoints uniformly distributed in a rectangle:

$$\lim_{n\to\infty} \frac{E[L_{DTSP}(n)]}{n^{2/3}} \geq \frac{3}{4}(3\rho WH)^{1/3},$$

where W and H are the dimensions of the rectangular region, and $L_{DTSP}(n)$ is the random length of the optimal Dubins tour over n points. We plot this lower bound on figure 2-4 as well. It is then interesting to evaluate the rate of growth of the tour length with n for the random input case. The $n^{2/3}$ rate of the lower bound is known to be optimal since the algorithm proposed in [110] achieves it. If we perform a linear regression using the points of figure 2-4, disregarding the instances with less than 20 waypoints which are too optimistic, we obtain the experimental growths given in table 2.1. We remark that as the number of discretization levels increases, the algorithm seems to approach the theoretical optimal growth rate. On the other hand, if we evaluate the bound of theorem 2.4.3, then $\varepsilon \approx 1/\sqrt{n}$, and so we obtain a growth rate for the randomized algorithm upper bounded by $n^{\frac{2}{3}+\frac{1}{2}} = n^{7/6}$, which seems to be overly conservative in the case of random waypoints. Note that we do not include the $\log n$ factor here because the ATSP was solved exactly in the experiments.

With 10 discretization levels, the rate is close to optimal on such random inputs. Note also the good asymptotic performance of the nearest neighbor heuristic, which moreover is much easier to compute than the other heuristics. Although not done for the comparisons on Fig. 2-4, in practice it is worth including the headings of NN as part of the set of headings used in the discretization of section 2.4. Finally, Fig. 2-5 shows an example of tours through 50 points computed using the Alternating Algorithm and our algorithm using 10 discretization levels.

2.6 Conclusion

In this chapter, we have presented new contributions to the study of the curvature-constrained traveling salesman problem. These include a proof that the problem is NP-hard and new algorithms that are not based on the waypoint ordering optimal for the Euclidean metric. An important remaining

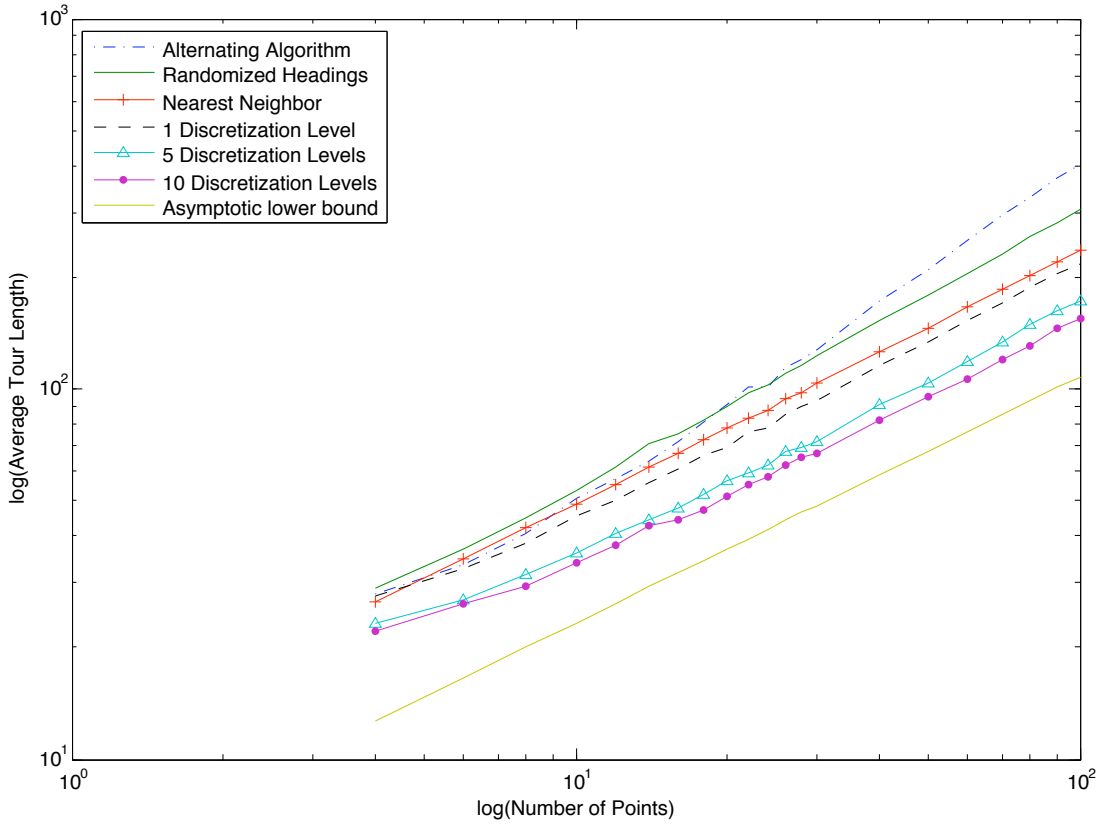
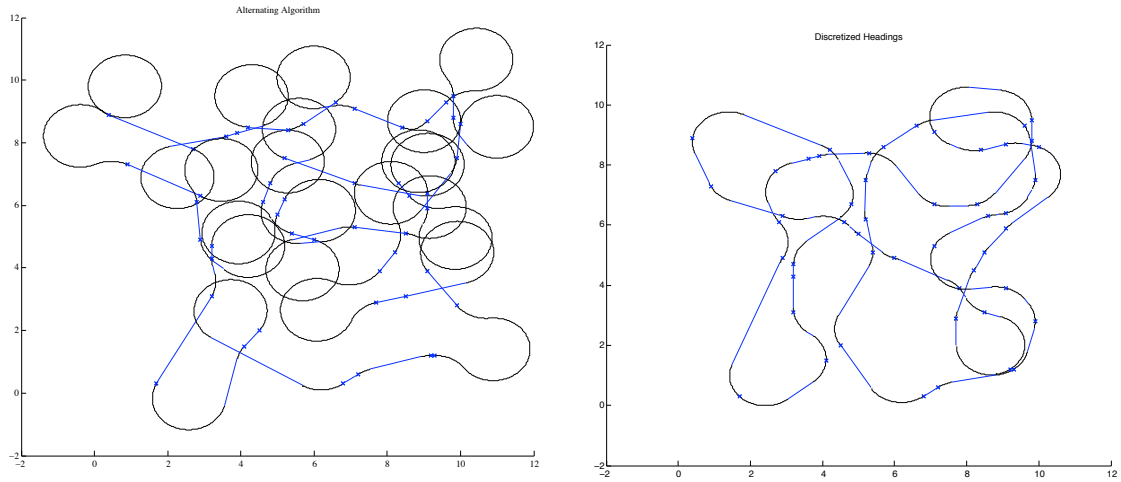


Figure 2-4: Average tour length vs. number of points in a 10×10 square, on a log-log scale. The average is taken over 30 experiments for each given number of points. Except for the randomized headings, the angle of the AA is always included in the set of discrete headings. Hence the difference between the AA curve and the 1 discretization level curve shows the performance improvement obtained by only changing the waypoint ordering.

Algorithm	Dubins tour average length
Alternating Algorithm	$1.6 \times n^{0.96}$
Nearest Neighbor Heuristic	$2.3 \times n^{0.69}$
Randomized Headings	$2.2 \times n^{0.76}$
AA + ATSP with 1 discretization level	$2.1 \times n^{0.71}$
5 discretization levels	$1.9 \times n^{0.7}$
10 discretization levels	$1.9 \times n^{0.68}$

Table 2.1: Experimental growth of the average Dubins tour lengths for the different algorithms, when the waypoints are distributed randomly and uniformly in a 10-by-10 square.



(a) Alternating Algorithm

(b) 10 discretization levels.

Figure 2-5: Dubins tours through 50 points randomly distributed in a 10×10 square. The turning radius of the vehicle is 1. The tour on the right is computed using 10 discretization levels.

direction of research is to improve on the algorithms in section 2.4 to obtain a $\log(n)$ or possibly constant factor approximation algorithm running in polynomial time (or to prove that this is not possible).

Designing good tours for UAVs performing surveillance missions is a basic task that an automated system should handle for an UAV operator. However, in a dynamic environment, the tours should be modified as more information is gathered during the mission, in order to improve the performance of these systems. Starting in the next chapter, we will focus on the design of such reactive systems helping the operator schedule dynamically the tasks of the vehicles.

Chapter 3

Dynamic Stochastic Scheduling and Bandit Problems

For the remainder of this thesis we study some dynamic scheduling and path planning problems, with a focus towards scheduling surveillance tasks for automated vehicles. The environment where these vehicles evolve is typically uncertain and in chapter 4, 5 and 6 we assume a stochastic model of the uncertainty.

A model of fundamental importance in stochastic scheduling and appearing in these chapters is the multi-armed bandit problem (MABP). In particular, the crucial assumption made is that a task to be performed by the vehicles at a site can be associated to a state that evolves in time independently of the states of the other tasks at different locations. In the context of aerial surveillance and sensor management, this means typically assuming that targets evolve independently of each other. As we will see, this assumption forms the basis of computational simplifications to obtain exact or approximate policies.

In this chapter, we provide some background material on the MABP and on an interesting extension, the restless bandit problem (RBP). This material is well known, but we gather results from various sources, with an emphasis on the computational techniques used later.

3.1 The Multi-Armed Bandit Problem (MABP)

Bandit problems can be posed in various forms, but invariably capture the conflict between taking actions which yield immediate reward and taking actions whose benefit will come only later. The terminology "multi-armed bandit problem" comes from the situation where one wants to play optimally on several slot machines (also called one-armed bandits). The basic stochastic version of the multi-armed bandit problem (MABP) in discrete-time can be described as follows. We consider N projects, of which only one can be worked on at any time period. Project i is characterized at time t by its state $x_i(t)$. If project i is worked on at time t, one receives a reward $\alpha^t r_i(x_i(t))$, where $\alpha \in (0,1)$ is a discount factor. The state $x_i(t)$ then evolves to a new state according to given transition probabilities. The states of all idle projects are unaffected. The goal is to find a policy which decides at each time period which project to work on in order to maximize the expected sum of the discounted rewards over an infinite horizon. The MABP was first solved efficiently by Gittins [49, 47]. He showed that it is possible to attach to each project an index that is a function of the project and of its state only, and that the optimal policy is simply characterized by operating at each period the project with the greatest current index. If the state space of each project is finite, these indices can be calculated efficiently [131, 64, 69, 16]. In fact, the best known algorithms to com-

pute the Gittins index function for one project with n states runs in time $O(n^3)$. There are also some closed form solutions available for a few specific MABPs. Among them:

- if a project is deteriorating, i.e., if a project in state x evolves with probability 1 to states y with reward $r(y) \leq r(x)$, then the Gittins' index is just the immediate reward $r(x(t))$. Hence if all projects are deteriorating, the simple greedy policy which maximizes the immediate reward is optimal.
- some deterministic scheduling problems which can be solved by a simple interchange argument [47].
- some problems with continuous time stochastic dynamics and certain reward structures, see in particular [67].

Remark 3.1.1. Varaiya et al. [131] showed that it is not necessary to assume Markovian dynamics for the optimality of an index policy in the MABP, but this assumption allows the efficient computation of the Gittins indices.

Intuitively, the Gittins index of a project in a given state summarizes the "value" of activating the project in this state, taking into account the immediate reward as well as the future evolution of the project. Then if several independent projects are available, we simply activate the one with highest value. There are several possible equivalent definitions of the Gittins index which correspond to this intuition. Consider a single project at time $t = 0$, in state x. Gittins [49, 48] proposed to compare the project to a standard project, whose activation corresponds to permanently retiring with a final reward $\lambda/(1-\alpha)$. Equivalently, the standard project has a single state and produces a constant reward λ at each period. The equivalence is due to the fact that if we activate the standard project in the second formulation, the initial project stops evolving. Hence, at least as far as optimal policies are concerned, once we engage the standard project we should engage it forever (see [39, lemma 4.5 p.23] for a more formal proof). The Gittins index $\nu(x)$ is then defined as the level of reward λ for the standard project which makes the optimal policy indifferent between choosing the initial project in state x or the standard project. Hence as λ varies we obtain a parametrized family of optimal stopping problems, and λ is the value of the parameter pitched just at the right level to make the optimal policy indifferent between continuing and stopping the initial project in state x.

From this discussion we obtain some algebraic definitions of the Gittins index. We take the dynamic programming point of view adopted by Whittle [135]. For now, we continue to consider the problem with one project and an additional standard project. Denote by $J^*(x;\lambda)$ the total expected reward achievable by a policy that chooses optimally the time to retire, when the project is in state x and the retirement reward is $\lambda/(1-\alpha)$. Let us suppose that the immediate reward function $r(\cdot)$ is bounded over the state-space $\mathscr{X}$ of the project so that there exist constants A, B such that $A \leq r(y) \leq B$ for all $y \in \mathscr{X}$.

Lemma 3.1.2 ([135])**.** *$J^*(x;\lambda)$ is an non-decreasing convex function of λ for which $J^*(x;\lambda) = \lambda$ for $\lambda \geq B$.*

Proof. The proof of the convexity of $J^*(x;\lambda)$ as a function of λ is the only non-trivial part. Write $J^*(x;\lambda)$ as

$$J^*(x;\lambda) = \sup_{\tau \geq 0} E\left\{ \sum_{t=0}^{\tau-1} \alpha^t r(x(t)) + \alpha^\tau \frac{\lambda}{1-\alpha} \right\}, \tag{3.1}$$

where the sup is over stopping times τ. Hence $J^*(x;\lambda)$, is the supremum of linear functions of λ hence convex in λ. □

The Gittins index $\lambda(x)$, by definition, is obtained as the minimum value of λ for which $J^*(x;\lambda) = \lambda/(1-\alpha)$, i.e., where both stopping and continuing are optimal. From equation (3.1), we then recover the characterizations of the type used by Gittins,

$$\lambda(x) = \sup_{\tau \geq 1} \frac{E\left[\sum_{t=0}^{\tau-1} \alpha^t r(x(t))\right]}{\frac{1-E[\alpha^\tau]}{1-\alpha}} = \sup_{\tau \geq 1} \frac{E\left[\sum_{t=0}^{\tau-1} \alpha^t r(x(t))\right]}{E\left[\sum_{t=0}^{\tau-1} \alpha^\tau\right]}.$$

Here on the right-hand side we take the supremum over strictly positive stopping times, considering policies that continue for at least one more step in state x.

Now going back to the original problem with N projects, suppose we can compute the indices $\lambda^i(x^i)$ for each project. Gittins theorem then says that for the MABP, it is *optimal* to choose at each period the project which corresponds to the maximal index, choosing arbitrarily among projects for which ties might occur.

3.2 The Restless Bandit Problem (RBP)

The strong assumptions made in the MABP inhibit its applicability in most sensor management problems that are of interest in this thesis. For example, if we have more than one vehicle at our disposal in a surveillance mission, we need to consider extensions to the MABP where $M > 1$ projects can be operated at every period. It is known that in general, operating the M projects with greatest Gittins indices is not optimal [47].

More crucially, even with only one mobile sensor platform, modeling a surveillance problem as a MABP requires assuming that targets that are not observed have their state frozen. This assumption applies for example in target identification problems: if no measurement on a target is taken at a given period, the posterior probability for that target does not evolve. However, in most scenarios of interest, such as the ones presented in chapter 4, 5 and 6, this assumption does not hold.

To overcome the shortcomings of the multi-armed bandit model, Whittle [136] introduced the restless bandit problem (RBP). Whittle himself mentioned the scheduling of aircraft in maritime surveillance missions as a potential application of the RBP. In this problem, we now allow for M projects to be simultaneously operated, rewards can be generated for the projects that are not active, and most importantly these projects are also allowed to evolve, possibly according to different transition rules. These less stringent assumptions are very useful for sensor management problems, but unfortunately the RBP is now known to be intractable, in fact PSPACE-hard [101], even if $M = 1$ and we only allow deterministic transition rules. Nonetheless, Whittle investigated an interesting relaxation and index policy for this problem, which extends Gittins' and gives yet another interpretation of the Gittins indices in the case of the MABP.

3.2.1 Whittle's Relaxation for the RBP

We now present the ideas proposed by Whittle to solve approximately the RBP, with a particular focus on the associated computational tools. Whittle's relaxation technique has been used more recently and apparently independently for sensor management problems by Castañón [27, 28], and investigated in a more general context by Adelman and Mersereau [2] and in Hawkins' thesis [56]. It is in fact a classical technique in control which relaxes hard pathwise constraints into soft integral constraints.

The RBP can be viewed as a constrained Markov decision process (MDP), with a sample-path constraint imposing that M out of the N available bandits be activated at each time period. This type of problem can be hard to solve in general, so we relax the resource constraint to enforce it only on

average. We then obtain a constrained MDP in a more classical form [6]. Moreover, an essential additional feature of Whittle's relaxation is that it decouples almost completely by project and can be computed by solving N independent problems, just as in the MABP, followed by a simple convex optimization problem.

To be more specific, consider N projects evolving as controlled Markov chains over state spaces $\mathscr{X}_i, i = 1,\ldots,N$. The global state space is $\mathscr{X} := \mathscr{X}_1 \times \ldots \times \mathscr{X}_N$. At each time, a project can be either active or passive. We denote the action for project i at time t by $A_i(t) = 0$ if the project is passive, $A_i(t) = 1$ if it is active, and let the global action vector be $A(t) = (A_1(t),\ldots,A_N(t))$. Associated to these actions for project i are two probability transition kernels $P^{i,0}$ and $P^{i,1}$ that govern the evolution of the state of project i. We assume that different projects evolve independently. If the N projects are in state $X = (X_1,\ldots,X_N)$ and action A is taken, we get for that period a reward

$$r(X,A) = \sum_{i=1}^{N} r(X_i,A_i).$$

In the discrete-time, discounted reward version of the RBP, we wish to maximize, over the policies π, the total reward

$$J_\alpha(\nu,\pi) = E_\nu^\pi \left\{ \sum_{t=0}^{\infty} \alpha^t r(X(t),A(t)) \right\}, \quad \alpha \in (0,1), \tag{3.2}$$

where α is the discount factor, ν a distribution on the initial states, and E_ν^π the expectation operator under policy π and initial distribution ν. This maximization problem is subject to the constraint

$$\sum_{i=1}^{N} 1_{\{A_i(t)=1\}} = M, \ \forall t, \tag{3.3}$$

where $1_{\{\cdot\}}$ denotes the indicator function. In the average reward version of the problem, we maximize

$$J_{avg}(\nu,\pi) = \limsup_{T\to\infty} \frac{1}{T} E_\nu^\pi \left\{ \sum_{t=0}^{T} r(X(t),A(t)) \right\}, \tag{3.4}$$

$$\tag{3.5}$$

still subject to constraint (3.3). Note that under appropriate conditions on the transition probabilities, the average reward $J_{avg}(\nu,\pi)$ will not depend on the initial distribution ν [103].

The only coupling between the different projects is due to the resource constraint (3.3). Still this problem does not seem easy to solve, so we relax it to enforce the constraint only on average. It is clear that (3.3) implies the new constraints

$$D_\alpha(\nu,\pi) = E_\nu^\pi \left\{ \sum_{t=0}^{\infty} \alpha^t \sum_{i=1}^{N} 1_{\{A^i(t)=0\}} \right\} = \frac{N-M}{1-\alpha}, \tag{3.6}$$

$$D_{avg}(\nu,\pi) = \limsup_{T\to\infty} \frac{1}{T} E_\nu^\pi \left\{ \sum_{t=0}^{T} \sum_{i=1}^{N} 1_{\{A^i(t)=0\}} \right\} = N-M, \tag{3.7}$$

which are useful for the discounted and average reward problems, respectively.

In the following, we treat the discounted reward problem, but the derivation for the average reward problem is entirely similar. The first goal is to solve the relaxed problem with constraint (3.3) now replaced by (3.6), in order to obtain an *upper bound on the achievable performance*. We

form the Lagrangian

$$\begin{aligned} L_\alpha(\nu,\pi;\lambda) &= J_\alpha(\nu,\pi) + \lambda\left(D_\alpha(\nu,\pi) - \frac{N-M}{1-\alpha}\right) \\ &= E_\nu^\pi\left\{\sum_{t=0}^{\infty}\alpha^t\sum_{i=1}^{N} r_i(X_i(t),A_i(t)) + \lambda 1_{\{A_i(t)=0\}}\right\} - \lambda\frac{N-M}{1-\alpha} \\ &= \sum_{i=1}^{N} E_\nu^\pi\left\{\sum_{t=0}^{\infty}\alpha^t\left(r_i(X_i(t),A_i(t)) + \lambda 1_{\{A_i(t)=0\}}\right)\right\} - \lambda\frac{N-M}{1-\alpha}. \end{aligned}$$

Then the optimal reward for the problem with averaged constraint satisfies

$$\hat{J}_\alpha(\nu) = \sup_{\pi\in\Pi}\inf_{\lambda} L_\alpha(\nu,\pi;\lambda) = \sup_{\pi\in\Pi_S}\inf_{\lambda} L_\alpha(\nu,\pi;\lambda), \tag{3.8}$$

where Π is the set of all policies, and Π_S is the set of stationary Markov (randomized) policies. Since we allow for randomized policies, a classical minimax theorem allows us to interchange the sup and the inf to get (see [6])

$$\hat{J}_\alpha(\nu) = \inf_{\lambda}\left\{\tilde{J}_\alpha(\nu;\lambda) - \lambda\frac{N-M}{1-\alpha}\right\}, \tag{3.9}$$

where

$$\tilde{J}_\alpha(\nu;\lambda) = \sup_{\pi\in\Pi_D} E_\nu^\pi\left\{\sum_{t=0}^{\infty}\alpha^t\sum_{i=1}^{N} r_i(X_i(t),A_i(t)) + \lambda 1_{\{A_i(t)=0\}}\right\}, \tag{3.10}$$

and Π_D is now the set of stationary deterministic policies. For a fixed value of λ, $\tilde{J}_\alpha(\nu;\lambda)$ can be computed using dynamic programming, and the possibility to restrict to deterministic policies is a classical result for unconstrained dynamic programming. Moreover, the computation of $\tilde{J}_\alpha(\nu;\lambda)$ has the interesting property of *being separable by site*, if the initial distribution of the states of the bandits is the product of individual independent distributions ν_i for each bandit. In this case we can solve the dynamic programming problem for each site separately since:

$$\tilde{J}_\alpha(\nu;\lambda) = \sum_{i=1}^{N}\tilde{J}_\alpha^i(\nu_i;\lambda),$$

and we have Bellman's equation (under appropriate conditions [14]) for each site separately:

$$\tilde{J}_\alpha^i(x_i;\lambda) = \max\{r_i(x_i,0) + \lambda + \alpha E^{i,0}[\tilde{J}^i(y_i;\lambda)|x_i], r_i(x_i,1) + \alpha E^{i,1}[\tilde{J}^i(y_i;\lambda)|x_i]\}. \tag{3.11}$$

So, for a fixed value of λ, we have decomposed the large problem over the cartesian product $\mathscr{X}$ into N similar dynamic programs providing the value functions $\tilde{J}_\alpha^i(\nu_i;\lambda)$. This is an important reduction in computational complexity, similar to what is achieved in Gittins' solution.

For any value of λ, we obtain an upper bound

$$G_\alpha(\nu;\lambda) = \tilde{J}_\alpha(\nu;\lambda) - \lambda\frac{N-M}{1-\alpha} \tag{3.12}$$

on the achievable performance. We can now finish the computation of the best such upper bound (3.9) using standard convex optimization methods. $G_\alpha(\nu;\lambda)$ is the dual function, which we would

like to minimize over λ. G is a convex function of λ because it is defined as a supremum of linear functions, but in general it is not differentiable. We can solve the minimization problem (3.9) using the subgradient method, although an even simpler method such as a line search would also be possible. We have the following well-known result, see e.g. [13]:

Theorem 3.2.1. *A subgradient of $G_\alpha(\nu;\cdot)$ at λ is*

$$D_\alpha(\nu,\pi^*_\lambda) - \frac{N-M}{1-\alpha} = \sum_{i=1}^{N} D^i_\alpha(\nu_i,\pi^{*,i}_\lambda) - \frac{N-M}{1-\alpha}, \tag{3.13}$$

where π^*_λ is an optimal policy for the problem (3.10) (which can be decomposed into optimal policies $\pi^{*,i}_\lambda$ for each site), and

$$D^i_\alpha(\nu_i,\pi^{*,i}_\lambda) = E^{\pi^{*,i}_\lambda}_{\nu_i}\left\{\sum_{t=0}^{\infty}\alpha^t 1\{a^i_t=0\}\right\}.$$

Note also that the computation of $D^i_\alpha(\nu^i,\pi^{*,i}_\lambda)$ is similar to the one for $\tilde{J}^{*i}_\alpha(\nu_i;\lambda)$: it is a RBP with passive reward 1 and active reward 0. So if we have a closed form expression of the value function verifying (3.11), we obtain a closed form expression for the subgradient at no additional cost. Chapter 4 will provide such an example.

3.2.2 Linear Programming Formulation of the Relaxation

In this section, we assume that the state space $\mathscr{X}_i$ of each project is finite. For bandit i and fixed λ, Bellman's equation is

$$\tilde{J}^i_\alpha(x_i;\lambda) = \max\left\{\lambda + r_i(x_i,0) + \alpha\sum_{y_i\in\mathscr{X}_i} p^{i,0}_{x_iy_i}\tilde{J}^i_\alpha(y_i;\lambda),\ r_i(x_i,1) + \alpha\sum_{y_i\in\mathscr{X}_i} p^{i,1}_{x_iy_i}\tilde{J}^i_\alpha(y_i;\lambda)\right\}. \tag{3.14}$$

This can be solved using linear programming (LP), see e.g. [103]. We obtain, for each i, the following linear program, whose variables are the $\{\lambda^i_{x_i}\}_{x_i\in\mathscr{X}_i}$, representing the values of the optimal reward-to-go:

$$\begin{aligned} &\text{minimize} \quad \sum_{x_i\in\mathscr{X}_i} \nu^i_{x_i}\lambda^i_{x_i} \\ &\text{subject to} \quad \lambda^i_{x_i} - \alpha\sum_{y_i\in\mathscr{X}_i} p^{i,a_i}_{x_iy_i}\lambda^i_{y_i} \geq r_i(x_i,a_i) + \lambda 1_{\{a_i=0\}}, \quad \forall x_i\in\mathscr{X}_i, a_i\in\{0,1\}. \end{aligned} \tag{3.15}$$

So we can solve, for each λ, these N linear programs to obtain the value of the objective in the subgradient algorithm. Alternatively, we can solve the N problems simultaneously and the minimization over λ as well via a single large LP, whose variables are the $\lambda^i_{x_i}$ and λ. The following LP corresponds to the direct optimization of (3.9) and its optimum value provides the upper bound on the achievable performance:

$$\begin{aligned} &\text{minimize}_{\{\{\lambda^i_{x_i}\},\lambda\}} \quad \sum_{i=1}^{N}\sum_{x_i\in\mathscr{X}_i} \nu^i_{x_i}\lambda^i_{x_i} - \lambda\frac{N-M}{1-\alpha} \\ &\text{subject to} \quad \lambda^i_{x_i} - \alpha\sum_{y_i\in\mathscr{X}_i} p^{i,a_i}_{x_iy_i}\lambda^i_{y_i} - \lambda 1_{\{a_i=0\}} \geq r_i(x_i,a_i), \quad \forall i,\forall x_i\in\mathscr{X}_i, a_i\in\{0,1\}. \end{aligned} \tag{3.16}$$

Remark 3.2.2. The LP (3.16) is the same as the LP (14) in [17], except for the constraint $\lambda \geq 0$ there, which is not correct (unless we want to solve instead the problem with *at most M* bandits

active at each period instead of exactly M bandits). The other slight differences are due to the fact that in that paper, the authors use equivalently a constraint on the number of expected *active* bandits, and λ is then a tax for activity instead of a subsidy for passivity.

For the average reward criterion, assuming each bandit is unichain [103], the LP (3.15) and (3.16) respectively become

$$\begin{aligned} \text{minimize}_{\{\gamma^i,\{h^i_{x_i}\}\}} \quad & \gamma^i \\ \text{subject to} \quad & \gamma^i + h^i_{x_i} - \alpha \sum_{y_i \in \mathscr{X}_i} p^{i,a_i}_{x_i y_i} h^i_{y_i} - \lambda 1_{\{a_i=0\}} \geq r_i(x_i,a_i), \quad \forall x_i \in \mathscr{X}_i, a_i \in \{0,1\}, \end{aligned} \tag{3.17}$$

and

$$\begin{aligned} \text{minimize}_{\{\{\gamma^i\},\{h^i_{x_i}\},\lambda\}} \quad & \sum_{i=1}^{N} \gamma^i - \lambda(N-M) \\ \text{subject to} \quad & \gamma^i + h^i_{x_i} - \alpha \sum_{y_i \in \mathscr{X}_i} p^{i,a_i}_{x_i y_i} h^i_{y_i} - \lambda 1_{\{a_i=0\}} \geq r_i(x_i,a_i), \quad \forall i, \forall x_i \in \mathscr{X}_i, a_i \in \{0,1\}. \end{aligned} \tag{3.18}$$

3.2.3 The State-Action Frequency Approach

In chapter 6, we formulate the RBP with switching costs using the state-action frequency approach to MDPs, which results in the linear programs dual to the ones obtained via dynamic programming as in section 3.2.2 [103]. This approach goes back to [89, 36, 37], and is based on the properties of the set of occupation measures achievable by different classes of policies. This material is only referred to in chapter 6.

Let us start with some general definitions. We will come back to the RBP in a moment. A (finite, discrete-time) MDP is defined by a tuple $\{\mathbf{X}, A, \mathscr{P}, r\}$ as follows:

- $\mathbf{X}$ is the finite state space.
- A is the finite set of actions. $A(x) \subset A$ is the subset of actions available at state x. $\mathscr{K} = \{(x,a) : x \in \mathbf{X}, a \in A(x)\}$ is the set of admissible state-action pairs.
- $\mathscr{P}$ are the transition probabilities. $\mathscr{P}_{xay}$ is the probability of moving from state x to state y if action a is chosen.
- $r : \mathscr{K} \to \mathbb{R}$ is an immediate reward.

We define the history at time t to be the sequence of previous states and actions, as well as the current state: $h_t = (x_1, a_1, x_2, a_2, \ldots, x_{t-1}, a_{t-1}, x_t)$. Let $\mathbf{H}_t$ be the set of all possible histories of length t. A policy u in the class of all policies U is a sequence $(u_1, u_2, \ldots)$. If the history h_t is observed at time t, then the controller chooses an action a with probability $u_t(a|h_t)$. A policy is called a Markov policy ($u \in \Pi_M$) if for any t, u_t only depends on the state at time t. A stationary policy ($u \in \Pi_S$) is a Markov policy that does not depend on t. Under a stationary policy, the state process becomes a homogeneous Markov chain with transition probabilities $P_{xy}[u] = \sum_{a \in A(x)} \mathscr{P}_{xay} u(a|x)$. Finally, a stationary policy is a deterministic policy ($u \in \Pi_D$) if it selects an action with probability one. Then u is identified with a map $u : \mathbf{X} \to A$.

We fix an initial distribution ν over the initial states. In other words, the probability that we are at state x at time 1 is $\nu(x)$. If ν is concentrated on a single state z, we use the Dirac notation $\nu(x) = \delta_z(x)$. Kolmogorov's extension theorem guarantees that the initial distribution ν and any

given policy u determine a unique probability measure $\mathbb{P}_\nu^u$ over the space of trajectories of the states X_t and actions A_t. We denote by $\mathbb{E}_\nu^u$ the corresponding expectation operation.

For any policy u and initial distribution ν, and for a discount factor $0 < \alpha < 1$, we define the total expected discounted reward

$$\begin{aligned} R_\alpha(\nu,u) &= \mathbb{E}_\nu^u \sum_{t=1}^{\infty} \alpha^{t-1} r(X_t, A_t) \\ &= \sum_{t=1}^{\infty} \alpha^{t-1} \mathbb{E}_\nu^u r(X_t, A_t), \end{aligned} \tag{3.19}$$

where the exchange of limit and expectation is valid in the case of finitely many states and actions by the dominated convergence theorem.

An occupation measure corresponding to a policy u is the total expected discounted time spent in different state-action pairs. More precisely, we define for any initial distribution ν, any policy u and any pair $x \in \mathbf{X}, a \in A(x)$:

$$f_\alpha(\nu,u;x,a) := (1-\alpha) \sum_{t=1}^{\infty} \alpha^{t-1} \mathbb{P}_\nu^u(X_t = x, A_t = a).$$

The vector $[f_\alpha(\nu,u;x,a)]_{x,a}$ defines a probability measure $f_\alpha(\nu,u)$ on the space of state-action pairs that assigns probability $f_\alpha(\nu,u;x,a)$ to the pair (x,a). $f_\alpha(\nu,u)$ is called an occupation measure and is associated to a stationary policy w defined by:

$$w(a|y) = \frac{f_\alpha(\nu,u;y,a)}{\sum_{a \in A} f_\alpha(\nu,u;y,a)}, \forall y \in \mathbf{X}, a \in A(y), \tag{3.20}$$

whenever the denominator is non-zero, otherwise we can choose $w(a|y) \in [0,1]$ arbitrarily. We can readily check that

$$R_\alpha(\nu,u) = \sum_{x \in \mathbf{X}} \sum_{a \in A} \frac{f_\alpha(\nu,u;x,a)}{1-\alpha} r(x,a). \tag{3.21}$$

Let $Q^\alpha(\nu)$ to be the set of vectors $\rho = [\rho_{x,a}]_{x,a} \in \mathbb{R}^{|\mathscr{K}|}$ satisfying

$$\begin{cases} \sum_{y \in \mathbf{X}} \sum_{a \in A(y)} \rho_{y,a} (\delta_x(y) - \alpha \mathscr{P}_{yax}) = \nu(x), \forall x \in \mathbf{X} \\ \rho_{y,a} \geq 0, \quad \forall y \in \mathbf{X}, a \in A(y). \end{cases} \tag{3.22}$$

$Q^\alpha(\nu)$ is a closed polyhedron. Note that by summing the first constraints over x we obtain $\sum_{y,a} \rho_{y,a} = 1/(1-\alpha)$, so $(1-\alpha)\rho$ with ρ satisfying the above constraints defines a probability measure. It also follows that $Q^\alpha(\nu)$ is bounded, i.e., is a closed polytope. One can check that the scaled occupation measures $f_\alpha(\nu,u)/(1-\alpha)$ belong to this polytope. In fact $Q^\alpha(\nu)$ describes exactly the set of occupation measures achievable by all policies (see [6] for a proof): the extreme points of the polytope $Q_\alpha(\nu)$ correspond to deterministic policies, and each policy can be obtained as a randomization over these deterministic policies. Thus, one can obtain a (scaled) optimal occupation measure corresponding to the maximization of (3.21) as the solution of the following linear program:

$$\begin{aligned} \text{maximize} \quad & \sum_{x \in \mathbf{X}} \sum_{a \in A} \rho_{x,a} r(x,a) \\ \text{subject to} \quad & [\rho_{x,a}]_{x,a} \in Q^\alpha(\nu), \end{aligned} \tag{3.23}$$

where $\rho = [\rho_{x,a}]_{x,a} \in \mathbb{R}^{|\mathscr{K}|}$ is the vector of decision variables.

The corresponding LP in the average reward formulation turns out to be [6]:

$$\begin{aligned} \text{maximize} \quad & \sum_{x\in\mathbf{X}} \sum_{a\in A} \rho_{x,a}\, r(x,a) \\ \text{subject to} \quad & [\rho_{x,a}]_{x,a} \in Q^1, \end{aligned} \tag{3.24}$$

where Q^1 is defined as

$$\begin{cases} \sum_{y\in\mathbf{X}} \sum_{a\in A(y)} \rho_{y,a}(\delta_x(y) - \mathscr{P}_{yax}) = 0, \forall x \in \mathbf{X} \\ \sum_{y\in\mathbf{X}} \sum_{a\in A(y)} \rho_{y,a} = 1 \\ \rho_{y,a} \geq 0, \quad \forall y \in \mathbf{X}, a \in A(y). \end{cases} \tag{3.25}$$

The decision variables $\rho_{x,a}$ have again the interpretation of state-action frequencies, now in the sense of the expected long-run fraction of time that the system is in state x and action a is taken. We skip the mathematical subtleties of the exact definition [6, chap. 4].

Application to the RBP

The relaxed version of the RBP is a MDP with an additional constraint on the average number of projects activated at each period. This constraint is easy to state in the domain of state-action frequencies, being just

$$\sum_{i=1}^{N} \sum_{x_i\in\mathscr{X}_i} \rho^i_{x_i,0} = \frac{N-M}{1-\alpha}$$

where the $(1-\alpha)\rho^i_{x_i,a_i}$ correspond to the state-action frequencies for bandit i. One can then simply add this constraint to the linear program (3.23) or (3.24), to obtain:

$$\text{maximize} \quad \sum_{i=1}^{N} \sum_{x_i\in\mathscr{X}_i} \rho^i_{x_i,0}\, r_i(x_i,0) + \rho^i_{x_i,1}\, r_i(x_i,1) \tag{3.26}$$

$$\text{subject to} \begin{cases} \sum_{a_i\in\{0,1\}} \left[\rho^i_{x_i,a_i} - \alpha \sum_{y_i\in\mathscr{X}_i} P^{i,a_i}_{y_i x_i} \rho^i_{y_i,a_i}\right] = \nu^i_{x_i},\ \forall i, \forall x_i \in \mathscr{X}_i \\ \sum_{i=1}^{N} \sum_{x_i\in\mathscr{X}_i} \rho^i_{x_i,0} = \frac{N-M}{1-\alpha} \\ \rho^i_{x_i,a_i} \geq 0, \quad \forall i, \forall x_i \in \mathscr{X}_i, a_i \in \{0,1\}, \end{cases} \tag{3.27}$$

in the discounted reward case, and

$$\text{maximize} \quad \sum_{i=1}^{N} \sum_{x_i\in\mathscr{X}_i} \rho^i_{x_i,0}\, r_i(x_i,0) + \rho^i_{x_i,1}\, r_i(x_i,1) \tag{3.28}$$

$$\text{subject to} \begin{cases} \sum_{a_i\in\{0,1\}} \left[\rho^i_{x_i,a_i} - \sum_{y_i\in\mathscr{X}_i} P^{i,a_i}_{y_i x_i} \rho^i_{y_i,a_i}\right] = 0,\ \forall i, \forall x_i \in \mathscr{X}_i \\ \sum_{a_i\in\{0,1\}} \sum_{y_i\in\mathscr{X}_i} \rho^i_{x_i,a_i} = 1,\ \forall i, \\ \sum_{i=1}^{N} \sum_{x_i\in\mathscr{X}_i} \rho^i_{x_i,0} = N-M \\ \rho^i_{x_i,a_i} \geq 0, \quad \forall i, \forall x_i \in \mathscr{X}_i, a_i \in \{0,1\}, \end{cases} \tag{3.29}$$

in the average reward case. It is straightforward to verify that these linear programs are the dual of the relaxed LPs (3.16) and (3.18). They can also be directly interpreted as a relaxation for an exact

formulation of the RBP in the space of state action frequencies. This will be treated in chapter 6.

3.2.4 Index Policies for the RBP

Whittle's Indices

Section 3.2.1 explains how to obtain an upper bound on the performance achievable in a restless bandit problem. It remains to find a good policy for the original, path constrained problem. Papadimitriou and Tsitsiklis have shown that the RBP is PSPACE hard [101] and, as remarked in [52], their proof in fact shows that it is PSPACE hard to decide if the optimal reward in an instance of the RBP is positive. Hence in [52], the authors conclude that it is PSPACE hard to approximate the RBP to any factor. Note however that a slight modification of the problem, say allowing only positive rewards, makes this argument impossible, i.e., no inapproximability result is available then.

On the positive side, we have heuristics that have performed well in practice. For the modification of the MABP where $M > 1$, playing the M bandits with highest Gittins indices is suboptimal in general [47], but is often expected to perform well [134]. For the RBP, Whittle proposed an index policy which generalizes Gittins' policy for the multi-armed bandit problem and emerges naturally from the Lagrangian relaxation [136].

To compute Whittle's indices, we consider the bandits (or targets) individually. Hence we isolate bandit i, and consider the problem of computing $\tilde{J}^i_\alpha(x^i;\lambda)$. λ can be viewed as a "subsidy for passivity", which parametrizes a collection of MDPs. Let us denote by $\mathscr{P}^i(\lambda) \subset \mathscr{X}_i$ the set of states x_i of bandit i such that the passive action is optimal, i.e.,

$$\mathscr{P}^i(\lambda) = \left\{ x_i \in \mathscr{X}_i : r_i(x_i,0) + \lambda + \alpha E^{i,0}[\tilde{J}^i(y_i;\lambda)|x_i] \geq r_i(x_i,1) + \alpha E^{i,1}[\tilde{J}^i(y_i;\lambda)|x_i] \right\}.$$

Definition 3.2.3. Bandit i is *indexable* if $\mathscr{P}^i(\lambda)$ is monotonically increasing from $\emptyset$ to $\mathscr{X}_i$ as λ increases from $-\infty$ to $+\infty$, i.e.,

$$\lambda_1 \leq \lambda_2 \Rightarrow \mathscr{P}^i(\lambda_1) \subseteq \mathscr{P}^i(\lambda_2). \tag{3.30}$$

Hence a bandit is indexable if the set of states for which it is optimal to take the passive action increases with the subsidy for passivity. This requirement seems very natural. Yet Whittle provided an example showing that it is not always satisfied, and typically showing the indexability property for particular cases of the RBP is challenging, see e.g. [97, 50]. However, when this property could be established, Whittle's index policy, which we now describe, was found empirically to perform outstandingly well.

Definition 3.2.4. If bandit i is indexable, its *Whittle index* is given, for any $x_i \in \mathscr{X}_i$, by

$$\lambda^i(x_i) = \inf\left\{ \lambda \in \mathbb{R} : x_i \in \mathscr{P}^i(\lambda) \right\}. \tag{3.31}$$

Hence, if the bandit is in state x_i, $\lambda^i(x_i)$ is the value of the subsidy λ which renders the active and passive actions equally attractive. Note that from the discussion in section 3.1, $\lambda^i(x_i)$ is exactly the Gittins' index if the RBP problem is in fact a MABP. Whittle showed that the MABP is always indexable.

Assuming that each bandit is indexable, we obtain for state $(x_1(t),\ldots,x_N(t))$ a set of Whittle indices $\lambda^1(x_1(t)),\ldots,\lambda^N(x_N(t))$. Then Whittle's index heuristic applies at each period t the active action to the M projects with largest index $\lambda^i(x_i(t))$, and the passive action to the remaining $N-M$ projects.

Note that in the relaxation, we obtained a value λ^* for the optimal Lagrange multiplier. Then the optimal policy for the relaxed problem simply considers each bandit independently, and activates bandit i in state x_i if and only if its index $\lambda^i(x_i)$ is greater than λ^*, by definition of the problem for bandit i, parameter value λ^*, and the Whittle index. In this way, the N bandits coordinate their actions by using the same Lagrange multiplier λ^* so that on average, only M of them will be active. Now Whittle's index policy, instead, activates only the M sites with largest indices, respecting the path constraint. Whittle conjectured that as $N \to \infty$ with the ratio M/N fixed, the performance per bandit of the Whittle policy approaches the performance per bandit of the optimal policy for the relaxed problem, hence is asymptotically optimal and the bound on performance is asymptotically tight. This conjecture, which is in general not true without further assumptions, has been studied by Weber and Weiss [133] when all bandits are identical.

3.3 Conclusion

In this chapter we have provided some background material related to certain classical dynamic scheduling models, namely the MABP and the RBP. The computational techniques and ideas presented in this chapter will be used repetitively in the next three chapters and specialized to the context of UVS management.

Chapter 4

Scheduling Observations

In this chapter we investigate a discrete dynamic vehicle routing problem, with a potentially large number of vehicles and sites to visit. This problem arises in intelligence, surveillance and reconnaissance (ISR) missions for UAS. Each site is modeled as an independent two-state Markov chain, whose state is not observed if the site is not visited by some vehicle. The goal for the vehicles is to collect rewards obtained when they visit the targets in a particular state. This problem can be seen as a type of restless bandit problem, as defined in chapter 3. Here we compute Whittle's index policy in closed form and the upper bound on the achievable performance efficiently. Simulation results provide evidence for the outstanding performance of this index heuristic and for the quality of the upper bound. A related estimation problem, scheduling the measurements of M sensors on $N > M$ independent targets modeled as Gaussian linear systems, will be considered in chapter 5.

In this chapter, we do not impose transition costs or delays for switching sensors between targets or locations. This allows for cleaner results, which can potentially be modified heuristically to account for switching effects. Since these effects are important in UAS missions to capture the path-planning component, we address them in more details in chapter 6.

4.1 Introduction

Consider the following scenario. A group of M mobile sensors (also called UAVs or agents in the following) is tracking the states of $N > M$ sites. We discretize time. At each period, each site can be in one of two states s_1 or s_2, but we only know the state of a site with certainty if we actually visit it with a sensor. For $i \in \{1,\dots,N\}$, the state of site i changes from one period to the next according to a Markov chain with known transition probability matrix P^i, independently of the fact that a sensor is present or not, and independently of the other sites. Hence we can estimate the states of the sites that are not visited by any sensor at a given period. To specify P^i, it is sufficient to give P^i_{11} and P^i_{21}, which are the probabilities of transition to state s_1 from state s_1 and s_2 respectively. When a sensor explores site i, it can observe its state *without measurement error*, and collect a reward R^i if the site is in state s_1. No reward is received if the site turns out to be in state s_2, and there is no cost for moving the agents between the sites. The goal is to allocate the sensors at each time period, in order to maximize an expected total discounted reward over an infinite horizon.

Through this model, we capture the following trade-off. It is advantageous to keep a good estimate of the states of the sites in order to take a good decision about which sites to visit next. However, the estimation problem is not the end goal, and so there is a balance between visiting sites to gain more information and collecting rewards. For an application example, one can think of the following environmental monitoring problem. M UAVs are surveilling N ships for possible bilge

water dumping. A reward is associated with the detection of a dumping event (state s_1 for a ship). However, if the UAV is not present during the dumping, the event is missed.

The problem described above is related to various sensor management problems. These problems have a long history [8, 91], but have enjoyed a renewed interest more recently due to the important research effort in sensor networks, see e.g. [43, 27, 28, 54, 125, 137]. Close to the ideas of our work, we mention the use by Krishnamurthy and Evans [74, 75] of Gittins' solution to the multi-armed bandit problem (MABP) to direct a radar beam towards multiple moving targets. See also [132, 112, 122]. La Scala and Moran [111] suggest to use instead (for a filtering problem) the restless bandits model, as we do here. However, in the restricted symmetric cases that [111] considers, the greedy solution is optimal and Whittle's indices and upper bound are not computed. In fact, Whittle already mentioned the potential application of restless bandits to airborne sensor routing in his original paper [136]. Recently, a slightly more general version of our problem was considered independently by Guha et al. [52], in the average-cost setting, to schedule transmissions on wireless communication channels in different states. These authors propose a policy that is different from Whittle's and offers a performance guarantee of 2. They show that their version of the problem is NP-hard. The complexity class of the problem considered here is currently not known.

The assumptions made in the MABP inhibit its applicability for the sensor management problem. Suppose one has to track the state of N targets evolving independently. First, the MABP solution helps scheduling only one sensor, since only one project can be worked on at each period. Moreover, even if one does not make new measurements on a specific target, its information state still has to be updated using the known dynamics of the true state. This violates the assumption that the projects that are not operated remain idle. Hence in [74] the authors have to assume that the dynamics of the targets are slow and that the propagation step of the filters can be neglected for unobserved targets. This might be a reasonable assumption for scheduling a radar beam, but not necessarily for our purpose of moving a limited number of airborne sensors to different regions. Whittle's extension to the RBP is useful in this context. His relaxation technique in particular has been used more recently and apparently independently for more involved sensor management problems but with similar characteristics by Castañón [27, 28]. Whittle's heuristic however has been rarely mentioned in the literature on sensor scheduling, except in [111]. A reason might be the difficulty of computing Whittle's indices.

The rest of the chapter is organized as follows. In section 4.2, we give a precise formulation of our problem. In section 4.3, we provide a counter example showing that the obvious candidate greedy solution to the problem is not optimal. Our approach is then to compute an upper bound on the achievable performance by using Whittle's relaxation, and a lower bound by computing Whittle's policy. The computation of Whittle's indices is non trivial in general, and the indices may not always exist. An attractive feature of our particular problem however is that most computations can be carried out analytically. Hence in section 4.4, we show the indexability of the problem and obtain simultaneously a closed form expression of Whittle's indices. Finally in section 4.5, we verify experimentally the high performance of the index policy by comparing it to the upper bound for some problems involving a large number of targets and vehicles.

4.2 Problem Formulation

For the dynamic optimization problem described in the introduction, the state of the N sites at time t is $x_t = (x_t^1, \ldots, x_t^N) \in \{s_1, s_2\}^N$, and the control is to decide which M sites to observe. An action at time t can only depend on the information state I_t which consists of the actions $a_0, \ldots, a_{t-1}$ at previous times as well as the observations $y_0, \ldots, y_{t-1}$ and the prior information y_{-1} on the initial

state x_0. We represent an action a_t by the vector $(a_t^1,\ldots,a_t^N)\in\{0,1\}^N$, where $a_t^i=1$ if site i is visited by a sensor at time t, and $a_t^i=0$ otherwise.

Assume the following flow of events. Given our current information state, we make the decision as to which M sites to observe. The rewards are obtained depending on the states observed, and the information state is updated. Once the rewards have been collected, the states of the sites evolve according to the known transition probabilities.

Let p be a given probability distribution on the initial state x_0. We assume independence of the initial distributions, i.e.,

$$\begin{aligned}P(x_0^1=s^1,\ldots,x_0^N=s^N)&=p(s^1,\ldots,s^N)\\&=\prod_{i=1}^N(p_{-1}^i)^{1\{s^i=s_1\}}(1-p_{-1}^i)^{1\{s^i=s_2\}},\end{aligned}$$

for some given numbers $p_{-1}^i\in[0,1]$. We denote by $1\{\cdot\}$ the indicator function. For an admissible policy π, i.e., depending only on the information process, we denote E_p^π the expectation operator. We want to maximize over the set Π of admissible policies the expected infinite-horizon discounted reward (with discount factor α)

$$J(p,\pi)=E_p^\pi\left\{\sum_{t=0}^\infty\alpha^t r(x_t,a_t)\right\},\tag{4.1}$$

where

$$r(x_t,a_t)=\sum_{i=1}^N R^i\,1\{a_t^i=1,x_t^i=s_1\},$$

and subject to the constraint

$$\sum_{i=1}^N 1\{a_t^i=1\}=M,\forall t.\tag{4.2}$$

It is well known that we can reformulate this problem as an equivalent Markov decision process (MDP) with complete information [14]. A sufficient statistic for this problem is given by the conditional probability $P(x_t|I_t)$, so we look for an optimal policy of the form $\pi_t(P(x_t|I_t))$. An additional simplification in our problem comes from the fact that the sites are assumed to evolve independently. Let p_t^i be the probability that site i is in state s_1 at time t, given I_t. A simple sufficient statistic at time t is then $(p_t^1,\ldots,p_t^N)\in[0,1]^N$.

Remark 4.2.1. The state representation chosen here involves a uncountable state space, for which the MDP theory is usually more technical. However, in our case, little additional complexity will be introduced. It is possible to adopt a state representation with a countable state space, by keeping track for each site of the number of periods since last visit as well as the state of the site at that last visit. In addition, we need to treat separately the time periods before we visit a site for the first time. This state representation, although potentially simpler from the theoretical point of view, is notationally more cumbersome and will not be used.

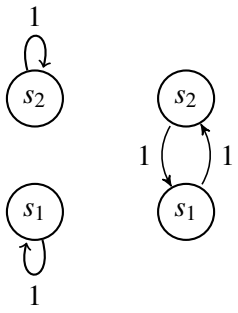


Figure 4-1: Counter Example.

We have the following recursion:

$$p_{t+1}^i = \begin{cases} P_{11}^i, \text{ if site } i \text{ is visited at time } t \text{ and found in state } s_1. \\ P_{21}^i, \text{ if site } i \text{ is visited at time } t \text{ and found in state } s_2. \\ f^i(p_t^i) := p_t^i P_{11}^i + (1-p_t^i)P_{21}^i = P_{21}^i + p_t^i(P_{11}^i - P_{21}^i), \\ \text{if site } i \text{ is not visited at time } t. \end{cases} \tag{4.3}$$

4.3 Non-Optimality of the Greedy Policy

We can first try to solve the problem formulated above with a general purpose solver for partially observable MDPs. However, the computations become quickly intractable, since the size of the underlying state space increases exponentially with the number of sites. Moreover, this approach would not take advantage of the structure of the problem, notably the independent evolution of the sites. We would like to use this structure to design optimal or good suboptimal policies more efficiently.

There is an obvious candidate solution to this problem, which consists in selecting at each period the M sites for which $p_t^i R^i$ is the highest. Let us call this policy the "greedy policy". It is not optimal in general. To see this, it is sufficient to consider a simple example with completely deterministic transition rules but uncertainty on the initial state. This underlines the importance of exploring at the right time.

Consider the example shown on Fig. 4-1, with $N = 2$, $M = 1$. Assume that we know already at the beginning that site 1 is in state s_1, i.e., $p_{-1}^1 = 1$. Hence we know that every time we select site 1, we will receive a reward R^1, and in effect this makes state s_2 of site 1 obsolete. Assume $R^1 > p_{-1}^2 R^2$, but $(1-p_{-1}^2)R^2 > R^1$, i.e., $R^2 - R^1 > p_{-1}^2 R^2$. Let us denote $p_{-1}^2 := p^2$ for simplicity. The greedy policy, with associated reward-to-go J_g, first selects site 1, and we have

$$J_g(1,p^2) = R^1 + \alpha J_g(1,1-p^2).$$

During the second period the greedy policy chooses site 2. Hence

$$J_g(1,1-p^2) = (1-p^2)R^2 + \alpha(1-p^2)J_g(1,0) + \alpha p^2 J_g(1,1).$$

Note that $J_g(1,0)$ and $J_g(1,1)$ are also the optimal values for the reward-to-go at these states, because

the greedy policy is obviously optimal once all uncertainty has been removed. It is easy to compute

$$J_g(1,0) = \frac{R^1 + \alpha R^2}{1-\alpha^2}, \quad J_g(1,1) = \frac{R^2 + \alpha R^1}{1-\alpha^2}.$$

Now suppose we sample first at site 2, removing the uncertainty, and then follow the greedy policy, which is optimal. We get for the associated reward-to-go:

$$J(1,p^2) = p^2 R^2 + \alpha p^2 J_g(1,0) + \alpha(1-p^2) J_g(1,1).$$

Let us take the difference:

$$\begin{aligned}
J - J_g &= p^2 R^2 + \alpha p^2 J_g(1,0) + \alpha(1-p^2) J_g(1,1) - R^1 \\
&\quad - \alpha(1-p^2) R^2 - \alpha^2 (1-p^2) J_g(1,0) - \alpha^2 p^2 J_g(1,1) \\
&= p^2 R^2 - R^1 - \alpha(1-p^2) R^2 \\
&\quad + \alpha J_g(1,0)(p^2 - \alpha + \alpha p^2) + \alpha J_g(1,1)(1 - p^2 - \alpha p^2) \\
&= p^2 R^2 - R^1 - \alpha(1-p^2) R^2 \\
&\quad + \frac{\alpha}{1-\alpha^2}[(R^1 + \alpha R^2)(p^2 - \alpha + \alpha p^2) \\
&\quad + (R^2 + \alpha R^1)(1 - p^2 - \alpha p^2)] \\
&= p^2 R^2 - R^1 - \alpha(1-p^2) R^2 + \\
&\quad \frac{\alpha}{1-\alpha^2}[R^1(p^2 - \alpha + \alpha p^2 + \alpha - \alpha p^2 - \alpha^2 p^2) \\
&\quad + R^2(\alpha p^2 - \alpha^2 + \alpha^2 p^2 + 1 - p^2 - \alpha p^2)] \\
&= p^2 R^2 - R^1 - \alpha(1-p^2) R^2 + \alpha p^2 R^1 + \alpha R^2 (1-p^2) \\
&= p^2 R^2 - R^1 + \alpha p^2 R^1.
\end{aligned}$$

For example, we can take $R^2 = 3R^1$, $p^2 = (1-\varepsilon)/3$, for a small $\varepsilon > 0$. We get $p^2 R^2 = R^1(1-\varepsilon) < R^1$ and $(1-p^2)R^2 = (2-\varepsilon)R^1 > R^1$ so our assumptions are satisfied. Then $J - J_g = \frac{\alpha}{3} R^1 (1 - \varepsilon - \frac{3\varepsilon}{\alpha})$, which can be made positive for ε small enough, and as large as we want by simply scaling the rewards. Hence in this case it is better to first inspect site 2 than to follow the greedy policy from the beginning.

4.4 Indexability and Computation of Whittle's Indices

The optimization problem (4.1) subject to the resource constraint (4.2) seems difficult to solve directly. It is clear however that this is a particular case of the RBP (3.2)-(3.3), and so we will try in the following to make Whittle's ideas more explicit for this specific problem. Our goal is to compute the upper bound (3.8), Whittle's indices (3.31), and to test the performance of Whittle's heuristic against this upper bound.

4.4.1 Preliminaries

In this section we study the indexability property for each site. For the sensor management problem considered in this chapter, we show that the bandits are indeed indexable and compute the Whittle indices in closed form. Since the discussion in this section concerns a single site, we drop the super-

script i indicating the site in the expressions of paragraph 3.2. A site has its dynamics completely specified by P_{21} and P_{11}. We denote the information state by p_t, i.e., p_t is the probability that the site is in state s_1, conditioned on the past. If we visit the site and it is in state s_1, we get a reward $R > 0$, but if it is in state s_2, we get no reward. In the following, we call visiting the site the *active* action. Finally, if we do not visit the site, which is called the *passive* action, we collect a reward λ with probability 1. The indexability property (3.30) that we would like to verify means that as λ increases, the set of information states where it is optimal to choose the passive action grows monotonically (in the sense of inclusion), from $\emptyset$ when $\lambda \to -\infty$ to $[0,1]$ when $\lambda \to +\infty$.

For convenience we rewrite Bellman's equation of optimality for this problem. If J is the optimal value function, then

$$\begin{aligned} &J(p) = \max\{\lambda + \alpha J(fp), pR + \alpha pJ(P_{11}) + \alpha(1-p)J(P_{21})\} \\ &\text{where } fp := pP_{11} + (1-p)P_{21} = P_{21} + p(P_{11} - P_{21}). \end{aligned} \tag{4.4}$$

Note that for simplicity, we redefined $J(p) := \tilde{J}^i(p;\lambda)$ in the notation of paragraph 3.2. First we have

Theorem 4.4.1. *J is a convex function of p, continuous on $[0,1]$.*

Proof. It is well known that we can obtain the value function by value iteration as a uniform limit of cost functions for finite horizon problems, which are continuous, piecewise linear and convex, see e.g. [121]. The uniform convergence follows from the fact that the discounted dynamic programming operator is a contraction mapping. The convexity of J follows, and the continuity on the closed interval $[0,1]$ is a consequence of the uniform convergence. □

Lemma 4.4.1. *1. When $\lambda \leq pR$, it is optimal to take the active action. In particular, if $\lambda \leq 0$, it is always optimal to take the active action and J is affine:*

$$\begin{aligned} J(p) &= \alpha J(P_{21}) + p[R + \alpha(J(P_{11}) - J(P_{21}))] \\ &= \frac{(\alpha P_{21} + p(1-\alpha))R}{(1-\alpha)(1-\alpha(P_{11} - P_{21}))}. \end{aligned} \tag{4.5}$$

2. When $\lambda \geq R$, it is always optimal to take the passive action, and

$$J(p) = \frac{\lambda}{1-\alpha}. \tag{4.6}$$

Proof. By convexity of J, $J(fp) \leq pJ(P_{11}) + (1-p)J(P_{21})$ and so for $\lambda \leq pR$, it is optimal to choose the active action. The rest of 1 follows by easy calculation, solving first for $J(P_{11})$ and $J(P_{21})$. To prove 2, use value iteration, starting from $J_0 = 0$. □

With this lemma, it is sufficient to consider from now on the situation $0 < \lambda < R$.

Lemma 4.4.2. *The set of $p \in [0,1]$ where it is optimal to choose the active action is convex, i.e., an interval in $[0,1]$.*

Proof. In the set where the active action is optimal, we have

$$J(p) = pR + \alpha pJ(P_{11}) + \alpha(1-p)J(P_{21}).$$

Consider p_1 and p_2 in this set. We want to show that for all $\beta \in [0,1]$, it is also optimal to choose the active action at $p = \beta p_1 + (1-\beta)p_2$. We know from Bellman's equation (4.4) that

$$pR + \alpha p J(P_{11}) + \alpha(1-p)J(P_{21}) \leq J(p).$$

By convexity of J, we have

$$\begin{aligned}
J(p) \leq & \beta J(p_1) + (1-\beta)J(p_2) \\
J(p) \leq & \beta\left(p_1 R + \alpha p_1 J(P_{11}) + \alpha(1-p_1)J(P_{21})\right) + (1-\beta)\left(p_2 R + \alpha p_2 J(P_{11}) + \alpha(1-p_2)J(P_{21})\right) \\
J(p) \leq & pR + \alpha p J(P_{11}) + \alpha(1-p)J(P_{21}).
\end{aligned}$$

Combining the two inequalities, we see that the active action is optimal at p. □

Lemma 4.4.3. *The sets of $p \in [0,1]$ where the passive and active actions are optimal are of the form $[0, p^*]$ and $[p^*, 1]$, respectively.*

Proof. This follows from the convexity of the active set and the fact that the active action is optimal for $p \geq \frac{\lambda}{R}$ by lemma 4.4.1. □

In the following, we emphasize the dependence of p^* on λ by writing $p^*(\lambda)$. It is a direct consequence of lemma 4.4.3 and the continuity of J that $p^*(\lambda)$ is the unique value where the passive and the active actions are equally attractive. We also see that *to show the indexability property of definition 3.2.3, it is sufficient to show that $p^*(\lambda)$ is an increasing function of λ*. Then, Whittle's index is obtained by inverting the relation $\lambda \to p^*(\lambda)$, i.e.,

$$\lambda(p) = \inf\{\lambda : p^*(\lambda) = p\}.$$

In the following, we will compute $p^*(\lambda)$ explicitly, distinguishing between various cases depending on the values of the parameters P_{11} and P_{21} of the bandit. In addition we also compute the value function $J(p)$ and the following "discounted passivity measure" for each bandit:

$$D(p, \pi_\lambda^*) = E_p^{\pi_\lambda^*}\left\{\sum_{t=0}^{\infty} \alpha^t 1_{\{a_t=0\}}\right\}.$$

This last quantity is necessary to compute the subgradient (3.13). Its computation is a policy evaluation problem. $D(p, \pi_\lambda^*)$ obeys the equations

$$D(p, \pi_\lambda^*) = \begin{cases} \alpha p D(P_{11}, \pi_\lambda^*) + \alpha(1-p)D(P_{21}, \pi_\lambda^*) & \text{for } p > p^*(\lambda) \\ 1 + \alpha D(fp, \pi_\lambda^*) & \text{for } p \leq p^*(\lambda). \end{cases}$$

These equations can be compared to those verified by $J(p)$ once $p^*(\lambda)$ is known:

$$J(p) = \begin{cases} R + \alpha p J^*(P_{11}, \lambda) + \alpha(1-p)J^*(P_{21}, \lambda) & \text{for } p > p^*(\lambda) \\ \lambda + \alpha J^*(fp, \lambda) & \text{for } p \leq p^*(\lambda). \end{cases}$$

Hence it is sufficient to have a closed form solution for $J(p)$. To compute $D(p, \pi_\lambda^*)$, we simply formally set $R = 0$ and $\lambda = 1$ in the corresponding expression for $J(p)$. For example, starting from expressions (4.5) and (4.6), we recover the (trivial) result that $D(p, \pi_\lambda^*) = 0$ if $\lambda \leq 0$ and $D(p, \pi_\lambda^*) = 1/(1-\alpha)$ if $\lambda \geq R$.

The next paragraphs of this section present the explicit computations. A summary of the results is provided in paragraph 4.4.5.

4.4.2 Case $P_{21} = P_{11}$

This case is very easy. Let $P_{11} = P_{21} = P$. We have in particular $fp = P$, for all $p \in [0,1]$. Bellman's equation gives

$$J(p) = \max\{\lambda + \alpha J(P), pR + \alpha J(P)\},$$

so in this case, we have immediately

$$p^*(\lambda) = \frac{\lambda}{R}$$

and the project is indexable since this is an increasing function of λ.

To give the complete expression of the value function, we only need to determine $J(P)$. There are two cases:

1. Case $P \leq (\lambda/R)$: then $J(P) = \frac{\lambda}{1-\alpha}$, and so

$$J(p) = \begin{cases} \frac{\lambda}{1-\alpha} & \text{if } p \leq \frac{\lambda}{R} \\ pR + \frac{\alpha\lambda}{1-\alpha} & \text{if } p > \frac{\lambda}{R}. \end{cases}$$

2. Case $P > (\lambda/R)$: then $J(P) = \frac{PR}{1-\alpha}$ and so

$$J(p) = \begin{cases} \lambda + \frac{\alpha PR}{1-\alpha} & \text{if } p \leq \frac{\lambda}{R} \\ \left(p + \frac{\alpha P}{1-\alpha}\right) R & \text{if } p > \frac{\lambda}{R}. \end{cases}$$

4.4.3 Case $P_{21} < P_{11}$

This case is more involved. We will need to consider the evolution of the state p_t.

Case $P_{11} = 1$, $P_{21} = 0$

Suppose first $P_{11} = 1$, $P_{21} = 0$, and recall that $0 < \lambda < R$. Then it is clear that $J(0) = \lambda/(1-\alpha)$ and $J(1) = R/(1-\alpha)$. By continuity of J, at p^* we are indifferent between the passive and active actions, so

$$J(p^*) = \lambda + \alpha J(p^*) = p^* R + \alpha p^* J(1) + \alpha(1-p^*)J(0),$$

from which $J(p^*) = \lambda/(1-\alpha)$ follows, and then

$$p^* = \frac{\lambda(1-\alpha)}{R - \alpha\lambda}.$$

Thus $p^*(\lambda)$ an increasing function of λ, since its derivative is

$$\frac{dp^*}{d\lambda} = \frac{(1-\alpha)R}{(R-\alpha\lambda)^2} \geq 0.$$

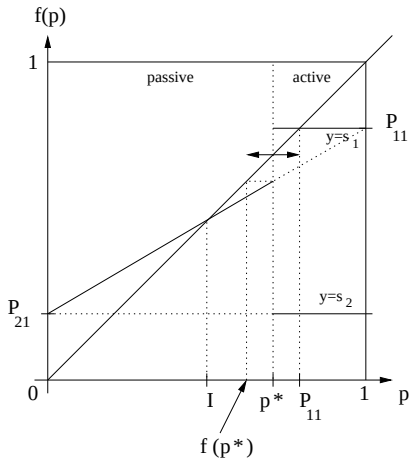


Figure 4-2: Case $I < p^* < P_{11}$. The line joining P_{21} and P_{11} is $p \to f(p)$. In the active region, we have $p_{t+1} = P_{11}$ or P_{21} depending on the observation.

As for the value function, we get

$$J(p) = \begin{cases} \frac{\lambda}{1-\alpha} & \text{if } p \le p^* \\ \frac{\alpha\lambda}{1-\alpha} + p\frac{R-\alpha\lambda}{1-\alpha} & \text{if } p > p^*. \end{cases}$$

Case $0 < P_{11} - P_{21} < 1$

In this case there is a unique point of intersection between the diagonal line and the line which is the graph of $p \to fp$, see Fig. 4-2. We will denote by I the abscissa in $[0,1]$ of this intersection point. Then I is defined by

$$I = fI = P_{21} + I(P_{11} - P_{21}),$$

that is,

$$I = \frac{P_{21}}{1 - (P_{11} - P_{21})},$$

which is well defined as long as we are not in the case of the previous paragraph. Note that $P_{21} \le I \le P_{11}$. Also, we have

$$f^n p_1 - f^n p_2 = (p_1 - p_2)(P_{11} - P_{21})^n, \ \forall\ p_1, p_2 \in [0,1],$$

and in particular

$$f^n p - f^n I = f^n p - I = (p - I)(P_{11} - P_{21})^n. \tag{4.7}$$

So as long as $|P_{11} - P_{21}| < 1$, as in this paragraph or in paragraph 4.4.4, the distance between $f^n p$ and I decreases strictly at each iteration and $f^n p \to I$ as $n \to \infty$.

Case $p^* \ge I$**:**

Assume first that $p^* \ge I$, and consider p belonging to the passive region $[0, p^*]$. From (4.7) and the assumption $0 < P_{11} - P_{21} < 1$, we obtain a sequence of iterates $f^n p$ which remains in the passive region while converging to I. Hence we get, since J is continuous at I,

$$J(p) = \lambda + \alpha J(fp) = \lambda + \alpha\lambda + \alpha^2 J(f^2 p) = \ldots = \frac{\lambda}{1-\alpha}.$$

So for instance, $J(P_{21}) = J(p^*) = \frac{\lambda}{1-\alpha}$, since $P_{21} \leq I \leq p^*$ implies that P_{21} belongs to the passive region.

Now for $p > p^*$, we have

$$J(p) = pR + \alpha pJ(P_{11}) + \alpha(1-p)J(P_{21}) = pR + \alpha pJ(P_{11}) + \alpha(1-p)\frac{\lambda}{1-\alpha}.$$

There are two subcases. First if $P_{11} \leq p^*$, then we get

$$J(p) = pR + \frac{\alpha\lambda}{1-\alpha}.$$

Continuity of J at p^*, gives $p^*R + \frac{\alpha\lambda}{1-\alpha} = \frac{\lambda}{1-\alpha}$ and so

$$p^*(\lambda) = \frac{\lambda}{R}. \tag{4.8}$$

This is the expression in the case $\frac{\lambda}{R} \geq P_{11}$.

It is also possible that $P_{11} > p^*$. Then

$$J(P_{11}) = P_{11}R + \alpha P_{11}J(P_{11}) + \alpha(1-P_{11})J(P_{21}),$$

which gives

$$J(P_{11}) = \frac{P_{11}R + \alpha(1-P_{11})\frac{\lambda}{1-\alpha}}{1-\alpha P_{11}}.$$

Using the continuity of J at p^* and the fact that $fp^* \leq p^*$, we get

$$p^*\left[R + \alpha\frac{P_{11}R + \alpha(1-P_{11})\frac{\lambda}{1-\alpha}}{1-\alpha P_{11}} - \frac{\alpha\lambda}{1-\alpha}\right] = \lambda,$$

which after simplifications gives

$$p^*(\lambda) = \frac{(1-\alpha P_{11})\lambda}{R-\alpha\lambda}. \tag{4.9}$$

Then the condition $p^* < P_{11}$ translates to $\frac{\lambda}{R} < P_{11}$, which is consistent. As before, it is easy to see that p^* is an increasing function of λ in this subcase. This completes the case $p^* \geq I$.

Remark 4.4.4. The expressions (4.8) and (4.9) give a continuous and monotonically increasing function $p^*(\lambda)$ on the interval $[\lambda_I, R] = [\lambda_I, P_{11}R] \cup [P_{11}R, R]$, where λ_I is the point where $p^*(\lambda_I) = I$, with $p^*(\lambda_I)$ given by (4.9). This can be rewritten

$$\lambda_I := \frac{P_{21}R}{1-(P_{11}-P_{21})(1+\alpha-\alpha P_{11})}.$$

Summarizing the results above, we have then

1. If $R > \lambda \geq P_{11}R$, then $p^*(\lambda) = \frac{\lambda}{R}$ and

$$J(p) = \begin{cases} \frac{\lambda}{1-\alpha} & \text{if } p \leq p^* \\ pR + \frac{\alpha\lambda}{1-\alpha} & \text{if } p > p^*. \end{cases}$$

2. If $P_{11}R > \lambda \geq \lambda_I$, then $p^*(\lambda) = \frac{(1-\alpha P_{11})\lambda}{R-\alpha\lambda}$ and

$$J(p) = \begin{cases} \frac{\lambda}{1-\alpha} & \text{if } p \leq p^* \\ \frac{\alpha\lambda}{1-\alpha} + p\frac{R-\alpha\lambda}{1-\alpha P_{11}} & \text{if } p > p^*. \end{cases}$$

Case $p^* < I$:

This subcase is the most involved. We have by continuity of J at p^*:

$$J(p^*) = \lambda + \alpha J(fp^*) = p^*R + \alpha p^* J(P_{11}) + \alpha(1-p^*)J(P_{21}).$$

Since $p^* < I$ we have $fp^* > p^*$, i.e., fp^* is in the active region. So we can rewrite the second equality as:

$$\lambda + \alpha((fp^*)R + \alpha(fp^*)J(P_{11}) + \alpha(1-fp^*)J(P_{21})) = p^*R + \alpha p^* J(P_{11}) + \alpha(1-p^*)J(P_{21}). \quad (4.10)$$

Expanding the left hand side gives

$$J(p^*) = \lambda + \alpha^2 J(P_{21}) + \alpha(P_{21} + p^*(P_{11} - P_{21}))[R + \alpha(J(P_{11}) - J(P_{21}))].$$

It is clear that $P_{11} \geq I$ and since $I > p^*$ we have

$$J(P_{11}) = P_{11}R + \alpha P_{11}J(P_{11}) + \alpha(1-P_{11})J(P_{21}),$$

so

$$J(P_{11}) = \frac{P_{11}R + \alpha(1-P_{11})J(P_{21})}{1-\alpha P_{11}},$$

which gives

$$J(P_{11}) - J(P_{21}) = \frac{P_{11}R - (1-\alpha)J(P_{21})}{1-\alpha P_{11}},$$

and

$$R + \alpha(J(P_{11}) - J(P_{21})) = \frac{R - \alpha(1-\alpha)J(P_{21})}{1-\alpha P_{11}}. \quad (4.11)$$

We now use (4.11) on both sides of the continuity condition (4.10):

$$\lambda + \alpha^2 J(P_{21}) + \alpha(P_{21} + p^*(P_{11} - P_{21}))\left[\frac{R - \alpha(1-\alpha)J(P_{21})}{1-\alpha P_{11}}\right] = \alpha J(P_{21}) + p^*\left[\frac{R - \alpha(1-\alpha)J(P_{21})}{1-\alpha P_{11}}\right].$$

We obtain:

$$p^* = \frac{\lambda - \alpha(1-\alpha)J(P_{21}) + \alpha P_{21}\left[\frac{R-\alpha(1-\alpha)J(P_{21})}{1-\alpha P_{11}}\right]}{(1-\alpha(P_{11}-P_{21}))\left[\frac{R-\alpha(1-\alpha)J(P_{21})}{1-\alpha P_{11}}\right]}.$$

We can rewrite this expression as

$$p^* = \frac{(1-\alpha(P_{11}-P_{21}))\left[\lambda - \alpha(1-\alpha)J(P_{21})\right] + \alpha P_{21}(R-M)}{(1-\alpha(P_{11}-P_{21}))\left[R - \alpha(1-\alpha)J(P_{21})\right]}$$

or

$$p^* = 1 - \frac{(R-\lambda)(1-\alpha P_{11})}{(1-\alpha(P_{11}-P_{21}))\left[R - \alpha(1-\alpha)J(P_{21})\right]}. \quad (4.12)$$

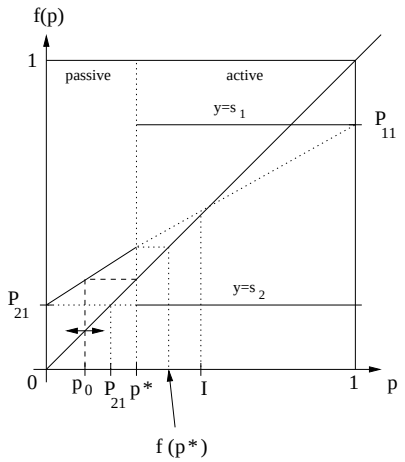


Figure 4-3: Case $P_{21} < p^* < I$.

So essentially the problem is solved if we can have an expression for $J(P_{21})$. The case where $p^* \leq P_{21}$ can be solved the most easily. Then

$$J(P_{21}) = P_{21}R + \alpha P_{21}J(P_{11}) + \alpha(1-P_{21})J(P_{21}),$$

which, combined with the similar equation for $J(P_{11})$, gives:

$$J(P_{11}) = \frac{R(P_{11} - \alpha(P_{11} - P_{21}))}{(1-\alpha)[1-\alpha(P_{11}-P_{21})]} \tag{4.13}$$

$$J(P_{21}) = \frac{P_{21}R}{(1-\alpha)[1-\alpha(P_{11}-P_{21})]}. \tag{4.14}$$

After some calculation, we get:

$$p^*(\lambda) = \frac{\lambda}{R}.$$

and this is valid for the case $p^*(\lambda) \leq P_{21}$, i.e., $\lambda \leq P_{21}R$. Moreover, in this case we obtain after some additional calculation:

$$R + \alpha(J(P_{11}) - J(P_{21})) = \frac{R(1-\alpha)}{(1-\alpha)[1-\alpha(P_{11}-P_{21})]},$$

which will be useful in the expression of $J(p)$.

Now from figure 4-3, if $P_{21} < p^* < I$, it can be seen that the iterates $f^k P_{21}$, initially in the passive region, eventually reach the active region and at that point one can evaluate $J(P_{21})$. The number of iterations however depends on the position of p^*.

It is easy to see that

$$f^n P_{21} = P_{21}\frac{1-(P_{11}-P_{21})^{n+1}}{1-(P_{11}-P_{21})}.$$

The rest of the analysis, computing $J(P_{21})$ for $P_{21} < p^* < I$, will distinguish different cases by the unique integer k such that:

$$f^k P_{21} < p^* \leq f^{k+1} P_{21}.$$

By definition of p^* separating the passive and active regions, we have:

$$J(P_{21}) = \lambda + \alpha\lambda + \alpha^2\lambda + \ldots + \alpha^k\lambda + \alpha^{k+1}J(f^{k+1}P_{21}) = \frac{1-\alpha^{k+1}}{1-\alpha}\lambda + \alpha^{k+1}J(f^{k+1}P_{21}), \tag{4.15}$$

$$\begin{aligned} J(f^{k+1}P_{21}) &= (f^{k+1}P_{21})R + \alpha(f^{k+1}P_{21})J(P_{11}) + \alpha(1-f^{k+1}P_{21})J(P_{21}) \\ &= \alpha J(P_{21}) + (f^{k+1}P_{21})[R + \alpha(J(P_{11}) - J(P_{21}))] \\ &= \alpha J(P_{21}) + (f^{k+1}P_{21})\frac{R-\alpha(1-\alpha)J(P_{21})}{1-\alpha P_{11}}, \end{aligned} \tag{4.16}$$

where the last line was obtained using (4.11). Solving this system of equations, we obtain

$$J(P_{21}) = \frac{1}{1-\alpha}\frac{(1-\alpha^{k+1})(1-\alpha P_{11})\lambda + \alpha^{k+1}(1-\alpha)(f^{k+1}P_{21})R}{(1-\alpha^{k+2})(1-\alpha P_{11}) + \alpha^{k+2}(1-\alpha)(f^{k+1}P_{21})}. \tag{4.17}$$

We can now use this expression in (4.12). As an intermediate step, we compute:

$$R - \alpha(1-\alpha)J(P_{21}) = \frac{(1-\alpha P_{11})[R(1-\alpha^{k+2}) - \lambda(\alpha - \alpha^{k+2})]}{(1-\alpha P_{11})(1-\alpha^{k+2}) + \alpha^{k+2}(1-\alpha)(f^{k+1}P_{21})}.$$

Then we obtain:

$$p^* = 1 - \frac{(R-\lambda)[(1-\alpha P_{11})(1-\alpha^{k+2}) + \alpha^{k+2}(1-\alpha)(f^{k+1}P_{21})]}{(1-\alpha(P_{11}-P_{21}))[R(1-\alpha^{k+2}) - \lambda(\alpha-\alpha^{k+2})]}.$$

We rewrite this expression in a more readable form as:

$$\begin{aligned} p^* &= 1 - A_k\frac{R-\lambda}{B_kR - C_k\lambda} \\ A_k &= \frac{(1-\alpha P_{11})(1-\alpha^{k+2}) + \alpha^{k+2}(1-\alpha)(f^{k+1}P_{21})}{1-\alpha(P_{11}-P_{21})} \\ B_k &= 1-\alpha^{k+2} \\ C_k &= \alpha - \alpha^{k+2}. \end{aligned} \tag{4.18}$$

The condition

$$f^kP_{21} < p^* \leq f^{k+1}P_{21}$$

can then be rewritten as

$$\frac{A_k - (1-f^kP_{21})B_k}{A_k - (1-f^kP_{21})C_k} < \frac{\lambda}{R} \leq \frac{A_k - (1-f^{k+1}P_{21})B_k}{A_k - (1-f^{k+1}P_{21})C_k}. \tag{4.19}$$

As a sanity check, we can verify that the successive thresholds agree:

$$\frac{A_k - (1-f^{k+1}P_{21})B_k}{A_k - (1-f^{k+1}P_{21})C_k} = \frac{A_{k+1} - (1-f^{k+1}P_{21})B_{k+1}}{A_{k+1} - (1-f^{k+1}P_{21})C_{k+1}}.$$

Taking the cross-products, this amounts to verifying:

$$(1-f^{k+1}P_{21})(B_{k+1}C_k - B_kC_{k+1}) + A_k(C_{k+1}-B_{k+1}) + A_{k+1}(B_k - C_k) = 0.$$

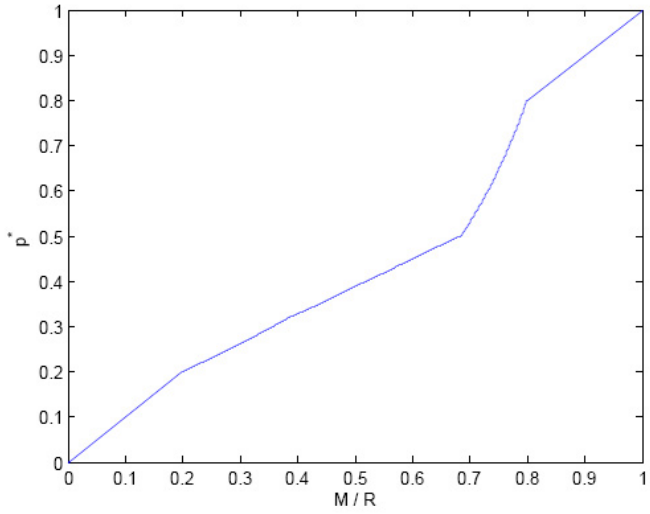


Figure 4-4: Plot of $p^*(\lambda)$ for $P_{21} = 0.2, P_{11} = 0.8, \alpha = 0.9$.

The left hand side gives

$$
\begin{aligned}
&(1-f^{k+1}P_{21})[\alpha(1-\alpha^{k+3})((1-\alpha^{k+1})-\alpha(1-\alpha^{k+2})^2]+(1-\alpha)(A_{k+1}-A_k)\\
=&-\alpha^{k+2}(1-\alpha)^2(1-f^{k+1}P_{21})+(1-\alpha)\frac{(1-\alpha P_{11})\alpha^{k+2}(1-\alpha)+\alpha^{k+2}(1-\alpha)(\alpha f^{k+2}P_{21}-f^{k+1}P_{21})}{1-\alpha(P_{11}-P_{21})}\\
=&\alpha^{k+2}(1-\alpha)^2\left[-(1-f^{k+1}P_{21})+\frac{(1-\alpha P_{11})+\alpha P_{21}+[\alpha(P_{11}-P_{21})-1]f^{k+1}P_{21}}{1-\alpha(P_{11}-P_{21})}\right]\\
=&0.
\end{aligned}
$$

On each interval, we verify that p^* is an increasing function of λ. We just compute

$$
\begin{aligned}
\frac{dp^*}{d\lambda} &= -A_k\frac{-(B_kR-C_k\lambda)+C_k(R-\lambda)}{(B_kR-C_k\lambda)^2}\\
\frac{dp^*}{d\lambda} &= A_k\frac{(1-\alpha)R}{(B_kR-C_k\lambda)^2}\geq 0.
\end{aligned}
$$

Remark 4.4.5. It can be verified that the first threshold also coincides with the previous case studied, that is:

$$\frac{A_0-(1-fP_{21})B_0}{A_0-(1-fP_{21})C_0} = P_{21}$$

Also, as $k\to\infty$, we can easily verify that the thresholds in (4.19) converge to λ_I/R.

A plot of $p^*(\lambda)$ is given on Fig. 4-4 and 4-5. To conclude this case, i.e., $\lambda<\lambda_I$, let us summarize the computational procedure.

1. If $\lambda\leq P_{21}R$, then $p^*(\lambda)=\frac{\lambda}{R}$. $J(P_{11})$ and $J(P_{21})$ are given by (4.13) and (4.14) respectively.
2. If $P_{21}R<\lambda<\lambda_I$, we have first to find the unique k such that the condition (4.19) is verified.

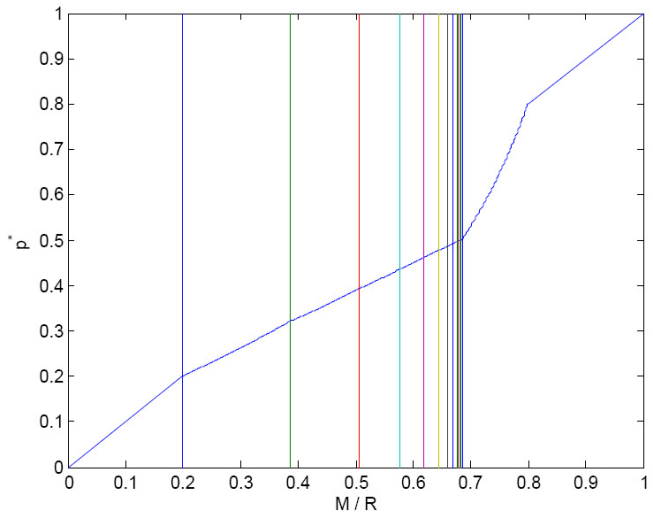


Figure 4-5: Plot of $p^*(\lambda)$ for $P_{21} = 0.2, P_{11} = 0.8, \alpha = 0.9$. The vertical lines show the separation between the regions corresponding to different values of k in the analysis for $p^* < I$. λ_I is an accumulation point, i.e., there are in fact infinitely many such lines converging to λ_I.

Once this is done, p^* is given by (4.18), $J(P_{21})$ is given by (4.17), and $J(P_{11})$, or equivalently $R + \alpha(J(P_{11}) - J(P_{21}))$, is given by (4.11).

With this, we have all the elements to actually compute $J(p)$ for given values of p and λ. When $p \geq p^*$, we have $J(p) = \alpha J(P_{21}) + p[R + \alpha(J(P_{11}) - J(P_{21}))]$ and so we are done. When $p < p^*$ however, to finish the computation we need to proceed as in the computation of $J(P_{21})$. We first find the unique integer l such that $f^l p < p^* \leq f^{l+1} p$ (it exists since here $p^* < I$ and $f^l p \to I$ as $l \to \infty$). Let $s = P_{11} - P_{21}$. Since

$$f^l p = P_{21} \frac{1 - s^l}{1 - s} + p s^l = I(1 - s^l) + p s^l = I - s^l (I - p),$$

we obtain

$$l = \left\lceil \frac{1}{\ln s} \ln \frac{I - p^*}{I - p} \right\rceil - 1.$$

Then we have

$$J(p) = \frac{1 - \alpha^{l+1}}{1 - \alpha} \lambda + \alpha^{l+2} J(P_{21}) + \alpha^{l+1} (f^{l+1} p)[R + \alpha(J(P_{11}) - J(P_{21}))],$$

see (4.15), (4.16).

4.4.4 Case $P_{21} > P_{11}$

Case $P_{11} = 0,\ P_{21} = 1$

As in section 4.4.3, we start the study of the remaining case $P_{21} > P_{11}$ with $P_{11} = 0, P_{21} = 1$. Then, because the active and the passive actions are necessarily optimal at $p = 1$ and $p = 0$ respectively

by lemma 4.4.3, we have

$$J(1) = R + \alpha J(0), \quad J(0) = \lambda + \alpha J(1),$$

which gives

$$J(1) = \frac{R + \alpha\lambda}{1-\alpha^2}, \quad J(0) = \frac{\lambda + \alpha R}{1-\alpha^2}, \tag{4.20}$$

and from which we get

$$R + \alpha(J(0) - J(1)) = \frac{R + \alpha\lambda}{1+\alpha}.$$

Now by continuity of J,

$$J(p^*) = \lambda + \alpha J(1-p^*) = p^* R + \alpha p^* J(0) + \alpha(1-p^*)J(1), \tag{4.21}$$

and using the preceding relations in the right-hand side

$$J(p^*) = \alpha\frac{R + \alpha\lambda}{1-\alpha^2} + p^*\frac{R + \alpha\lambda}{1+\alpha} = \frac{R + \alpha\lambda}{1+\alpha}\left(\frac{\alpha}{1-\alpha} + p^*\right). \tag{4.22}$$

Now if we have $p^* \geq 1/2$, then $1 - p^* \leq 1/2 \leq p^*$ is in the passive region, and so

$$J(1-p^*) = \lambda + \alpha J(p^*).$$

Reporting in the first part of (4.21), we obtain

$$J(p^*) = \frac{\lambda}{1-\alpha},$$

which, with (4.22), gives

$$p^* = \frac{\lambda(1+\alpha-\alpha^2) - \alpha R}{(1-\alpha)(R+\alpha\lambda)}.$$

p^* is an increasing function of λ since one can verify that

$$\frac{dp^*}{d\lambda} = (1-\alpha^2)\frac{R}{(1-\alpha)^2(R+\alpha\lambda)^2} \geq 0.$$

Then the condition $p^* \geq 1/2$ translates to

$$\frac{\lambda}{R} \geq \frac{1+\alpha}{2+\alpha-\alpha^2} = \frac{1}{2-\alpha}.$$

In the case $p^* < 1/2$, $(1-p^*) > 1/2 > p^*$ is in the active region, so we obtain

$$\begin{aligned} J(1-p^*) &= (1-p^*)R + \alpha(1-p^*)J(0) + \alpha p^* J(1) \\ &= R + \alpha J(0) - p^*[R + \alpha(J(0) - J(1))] \end{aligned}$$

Reporting in (4.21), we get

$$\lambda + \alpha(R + \alpha J(0) - J(1)) = p^*(1+\alpha)[R + \alpha(J(0) - J(1))]$$

which gives after easy calculation

$$p^* = \frac{\lambda}{R + \alpha\lambda}.$$

This is again an increasing function of λ and the condition $p^* < 1/2$ can be written

$$\frac{\lambda}{R} < \frac{1}{2-\alpha},$$

which is coherent (the junction between the two cases happens when $p^*(\lambda)$ hits $1/2$).

Now for a given value of p and λ, we also want to compute $J(p)$. Comparing λ/R to $1/(2-\alpha)$, we can deduce the correct formula for p^*. Then if $p \geq p^*$, we are done, using the values of $J(0)$ and $J(1)$ in (4.20). We obtain in this case:

$$J(p) = \frac{R + \alpha\lambda}{1-\alpha^2}(\alpha + p(1-\alpha)).$$

If $p < p^*$, $J(p) = \lambda + \alpha J(1-p)$, and we distinguish between two subcases. If $(1-p) \leq p^*$ (which is only possible if $p^* > 1/2$), i.e., $1 - p^* \leq p < p^*$, then

$$J(p) = \frac{\lambda}{1-\alpha}.$$

Otherwise, if $(1-p) > p^*$, i.e., $p < 1 - p^*$ and $p < p^*$, then

$$J(p) = \lambda + \alpha[(1-p)R + \alpha(1-p)J(0) + \alpha p J(1)].$$

and this gives after simplifications

$$J(p) = \frac{\lambda(1-\alpha^2 p(1-\alpha)) + \alpha R(1-p(1-\alpha))}{1-\alpha^2}.$$

Case $0 < P_{21} - P_{11} < 1$

The discussion related to the fixed point I at the beginning of section 4.4.3 is still valid here, including equation (4.7) showing the convergence of the iterates $f^n p$ to I since we assume in this paragraph that $0 < P_{21} - P_{11} < 1$. These iterations land now alternatively on each side of I.

Case $p^* \geq I$.

In the case $p^* \geq I$, the iterates $f^n p^*$ converge to I while remaining in the passive region. So we immediately get

$$J(p^*) = \frac{\lambda}{1-\alpha}.$$

By continuity of J at p^*, where the active action is also optimal, we have

$$\frac{\lambda}{1-\alpha} = p^* R + \alpha p^* J(P_{11}) + \alpha(1-p^*)J(P_{21}),$$

and this gives us

$$p^* = \frac{\frac{\lambda}{1-\alpha} - \alpha J(P_{21})}{R + \alpha(J(P_{11}) - J(P_{21}))}. \tag{4.23}$$

We now compute $J(P_{11})$ and $J(P_{21})$ in the different cases, depending on the position of p^*. Note

first that $P_{11} < I$ necessarily, so P_{11} is in the passive region by our assumption $p^* \geq I$. Hence

$$J(P_{11}) = \lambda + \alpha J(fP_{11}).$$

fP_{11} is greater than I however and can fall in the passive or the active region. If $fP_{11} \leq p^*$, it is easy to see that the iterates $f^n P_{11}$ will remain in the passive region, and so we obtain

$$J(P_{11}) = \frac{\lambda}{1-\alpha}.$$

If $fP_{11} > p^*$, we get

$$J(fP_{11}) = (fP_{11})R + \alpha(fP_{11})J(P_{11}) + \alpha(1-fP_{11})J(P_{21}).$$

We can now have p^* greater or smaller than P_{21}. However note that necessarily $I < P_{21}$ and so if $P_{21} \leq p^*$, the iterates $f^n P_{21}$ remain in the passive region and

$$J(P_{21}) = \frac{\lambda}{1-\alpha}.$$

Otherwise if $p^* < P_{21}$, then

$$J(P_{21}) = P_{21}R + \alpha P_{21}J(P_{11}) + \alpha(1-P_{21})J(P_{21}),$$

and so

$$J(P_{21}) = P_{21}\frac{R+\alpha J(P_{11})}{1-\alpha+\alpha P_{21}},$$
$$R + \alpha(J(P_{11}) - J(P_{21})) = (1-\alpha)\frac{R+\alpha J(P_{11})}{1-\alpha+\alpha P_{21}}.$$

We can now finish the computation of p^* using (4.23). Note first that

$$fP_{11} = P_{21} - P_{11}(P_{21} - P_{11}) < P_{21}.$$

Hence we will subdivide the interval $[I,1]$ into the union $[I, fP_{11}] \cup [fP_{11}, P_{21}] \cup [P_{21}, 1]$. For $p^* \in [P_{21}, 1]$, $J(P_{11}) = J(P_{21}) = \lambda/(1-\alpha)$ and so

$$p^* = \frac{\lambda}{R}.$$

Clearly then $p^*(\lambda) = P_{21}$ if and only if $\lambda = P_{21}R$.

For $p^* \in [fP_{11}, P_{21}]$,

$$p^* = \frac{\frac{\lambda}{1-\alpha} - \alpha P_{21}\frac{R+\alpha J(P_{11})}{1-\alpha+\alpha P_{21}}}{(1-\alpha)\frac{R+\alpha J(P_{11})}{1-\alpha+\alpha P_{21}}}, \quad J(P_{11}) = \frac{\lambda}{1-\alpha}.$$

This gives after substitution

$$p^* = \frac{\lambda(1+\alpha P_{21}) - \alpha P_{21}R}{R(1-\alpha)+\alpha\lambda}.$$

We verify easily that $\frac{dp^*}{d\lambda}$ is an increasing function of λ and we check that this expression gives again

$p^*(\lambda) = P_{21}$ if and only if $\lambda = P_{21}R$. As for the other side of the interval, we have $p^*(\lambda) = fP_{11}$ if and only if

$$\lambda = \lambda_{fP_{11}} := R\frac{P_{21} - (1-\alpha)P_{11}(P_{21} - P_{11})}{1 + \alpha P_{11}(P_{21} - P_{11})}. \tag{4.24}$$

Finally we consider the case $p^* \in [I, fP_{11}]$. There is a bit more work to get an expression for $J(P_{11})$. We have

$$\begin{aligned}
J(P_{11}) &= \lambda + \alpha J(fP_{11}) \\
J(P_{11}) &= \lambda + \alpha^2 J(P_{21}) + \alpha(fP_{11})[R + \alpha(J(P_{11}) - J(P_{21}))] \\
J(P_{11}) &= \lambda + \frac{R + \alpha J(P_{11})}{1 - \alpha + \alpha P_{21}}[\alpha^2 P_{21} + \alpha(1-\alpha)(fP_{11})].
\end{aligned}$$

This implies immediately

$$\begin{aligned}
R + \alpha J(P_{11}) &= (R + \alpha\lambda) + \frac{R + \alpha J(P_{11})}{1 - \alpha + \alpha P_{21}}[\alpha^3 P_{21} + \alpha^2(1-\alpha)(fP_{11})] \\
R + \alpha J(P_{11}) &= \frac{(R + \alpha\lambda)(1 - \alpha + \alpha P_{21})}{(1-\alpha)(1 + \alpha P_{21} + \alpha^2 P_{11}(P_{21} - P_{11}))}.
\end{aligned}$$

Finally this gives

$$\begin{aligned}
p^* &= \frac{\lambda(1 + \alpha(1+\alpha)P_{21} - \alpha^2(fP_{11})) - \alpha P_{21}(R + \alpha\lambda)}{(1-\alpha)(R + \alpha\lambda)} \\
p^* &= \frac{\lambda(1 + \alpha(1-\alpha)P_{21} + \alpha^2 P_{11}(P_{21} - P_{11})) - \alpha P_{21} R}{(1-\alpha)(R + \alpha\lambda)}.
\end{aligned}$$

It is again straightforward to verify that this is an increasing function of λ by computing the derivative. For the boundary points, we get by direct calculations that $p^*(\lambda) = fP_{11}$ if and only if λ is given by (4.24), verifying the continuity at this point, and $p^*(\lambda) = I$ if and only if

$$\lambda = \lambda_I := \frac{P_{21}R}{1 + (P_{21} - P_{11})(1 - \alpha + \alpha P_{11})}. \tag{4.25}$$

This expression will also be obtained from the analysis below for $P_{11} < p^* < I$.

Case $p^* < I$.

Last, we consider the case $p^* < I$. It is clear graphically or by inspection of the expression for I that $P_{21} > I$, so the active action is optimal at P_{21}, and

$$J(P_{21}) = P_{21}\frac{R + \alpha J(P_{11})}{1 - \alpha(1 - P_{21})}.$$

This gives

$$R + \alpha(J(P_{11}) - J(P_{21})) = (1-\alpha)\frac{R + \alpha J(P_{11})}{1 - \alpha(1 - P_{21})}.$$

We have, again by equation (4.7), that $fp^* > I > p^*$, so, by continuity of J at p^*,

$$\lambda + \alpha J(fp^*) = \lambda + \alpha[(fp^*)R + \alpha(fp^*)J(P_{11}) + \alpha(1 - fp^*)J(P_{21})] = p^*R + \alpha p^* J(P_{11}) + \alpha(1-p^*)J(P_{21}),$$

identical to (4.10). We get

$$\lambda(1-\alpha+\alpha P_{21})+(R+\alpha J(P_{11}))(\alpha^2 P_{21}+\alpha(1-\alpha)(P_{21}+p^*(P_{11}-P_{21})))$$
$$=(R+\alpha J(P_{11}))(\alpha P_{21}+(1-\alpha)p^*),$$

i.e.,

$$\frac{\lambda(1-\alpha+\alpha P_{21})}{R+\alpha J(P_{11})}+\alpha(1-\alpha)p^*(P_{11}-P_{21})=(1-\alpha)p^*,$$

so

$$p^*=\frac{(1+\alpha P_{21}-\alpha)\lambda}{(1-\alpha)(1+\alpha P_{21}-\alpha P_{11})(R+\alpha J(P_{11}))}.$$

Hence the problem is solved if we have an expression for $J(P_{11})$. It is clear graphically that $P_{11}<I$. From equation (4.7), we also see that $fP_{11}>I$ since $P_{11}-P_{21}<0$. Hence this time, fP_{11} is in the active region, and so the analysis is simpler than in paragraph 4.4.3. The only cases to consider are $0\le p^*\le P_{11}$ and $P_{11}<p^*<I$.

In the first subcase $0\le p^*\le P_{11}$, P_{11} is in the active region and so

$$J(P_{11})=P_{11}R+\alpha P_{11}J(P_{11})+\alpha(1-P_{11})J(P_{21})$$
$$J(P_{11})=\alpha\frac{P_{21}R+\alpha P_{21}J(P_{11})}{1-\alpha(1-P_{21})}+P_{11}(1-\alpha)\frac{R+\alpha J(P_{11})}{1-\alpha(1-P_{21})}$$
$$J(P_{11})=\frac{(P_{11}+\alpha(P_{21}-P_{11}))(R+\alpha J(P_{11}))}{1-\alpha(1-P_{21})},$$

and so

$$J(P_{11})=\frac{R(P_{11}+\alpha(P_{21}-P_{11}))}{(1-\alpha)(1+\alpha(P_{21}-P_{11}))},$$
$$R+\alpha J(P_{11})=\frac{R(1-\alpha+\alpha P_{21})}{(1-\alpha)(1+\alpha(P_{21}-P_{11}))},$$
$$R+\alpha[J(P_{11})-J(P_{21})]=\frac{R}{1+\alpha(P_{21}-P_{11})}.$$

This gives

$$p^*=\frac{\lambda}{R},$$

and the condition $p^*\le P_{11}$ is $\lambda\le P_{11}R$.

In the second subcase, $P_{11}<p^*<I$, and fP_{11} is in the active region, so we have

$$J(P_{11})=\lambda+\alpha J(fP_{11})=\lambda+\alpha[fP_{11}R+\alpha fP_{11}J(P_{11})+\alpha(1-fP_{11})J(P_{21})]$$
$$J(P_{11})=\lambda+\frac{R+\alpha J(P_{11})}{1-\alpha(1-P_{21})}(\alpha^2 P_{21}+\alpha(1-\alpha)fP_{11})$$
$$(R+\alpha J(P_{11}))(1-\alpha+\alpha P_{21})=(R+\alpha\lambda)(1-\alpha+\alpha P_{21})+\alpha^2(\alpha P_{21}+(1-\alpha)fP_{11})(R+\alpha J(P_{11}))$$

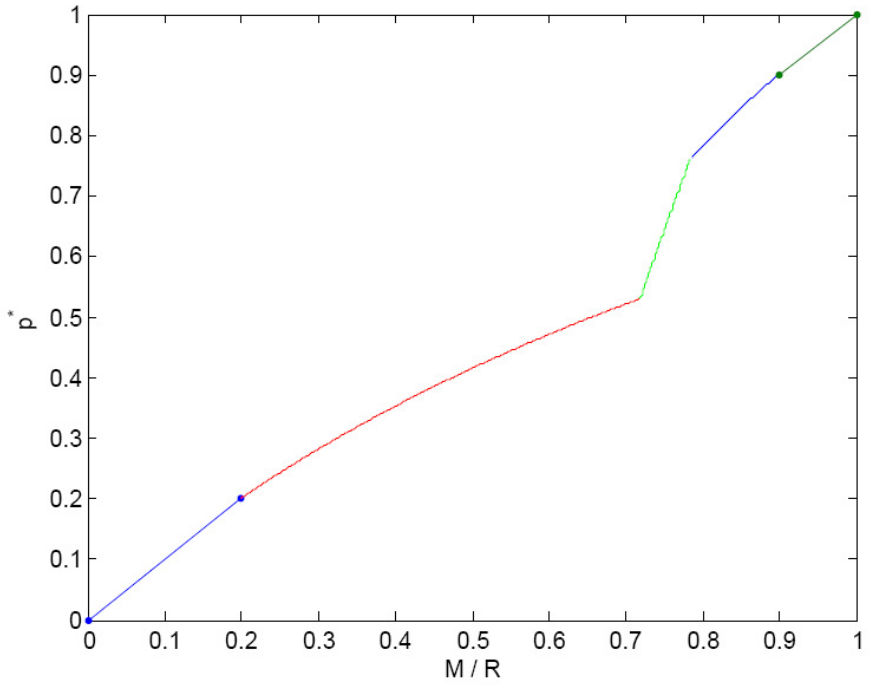


Figure 4-6: Plot of $p^*(\lambda)$ for $P_{21} = 0.9, P_{11} = 0.2, \alpha = 0.9$.

and so we get

$$R + \alpha J(P_{11}) = \frac{(R+\alpha\lambda)(1-\alpha+\alpha P_{21})}{(1-\alpha)[1+\alpha P_{21}+\alpha^2 P_{11}(P_{21}-P_{11})]}$$
$$R + \alpha[J(P_{11}) - J(P_{21})] = \frac{(R+\alpha\lambda)}{[1+\alpha P_{21}+\alpha^2 P_{11}(P_{21}-P_{11})]}.$$

This gives finally

$$p^* = \frac{\lambda[1+\alpha P_{21}+\alpha^2 P_{11}(P_{21}-P_{11})]}{(R+\alpha\lambda)(1+\alpha P_{21}-\alpha P_{11})},$$

which simplifies to (add and substract αP_{11} in the numerator)

$$p^* = \frac{\lambda(1+\alpha P_{11})}{R+\alpha\lambda}.$$

It is easy to see that it is an increasing function of λ, and that the condition $p^* \geq P_{11}$ translates to $\lambda \geq P_{11}R$. We also verify that the condition $p^* < I$ corresponds to $\lambda < \lambda_I$, where λ_I is given by (4.25), which implies the continuity of $p^*(\lambda)$ at λ_I.

A plot of $p^*(\lambda)$ is given on Fig. 4-6. Finally, we summarize the computational procedure for the case $0 < P_{21} - P_{11} < 1$. Given λ, we first check in which subset of the partition $[0, P_{11}R] \cup [P_{11}R, \lambda_I] \cup$

$[\lambda_I, \lambda_{fP_{11}}] \cup [\lambda_{fP_{11}}, P_{21}R] \cup [P_{21}R, R]$ it belongs. We then compute $p^*(\lambda)$ accordingly. That is,

$$p^*(\lambda) = \begin{cases} \frac{\lambda}{R} \text{ if } \lambda \in [0, P_{11}R] \cup [P_{21}R, R] \\ \frac{\lambda(1+\alpha P_{11})}{R+\alpha\lambda} \text{ if } \lambda \in [P_{11}R, \lambda_I] \\ \frac{\lambda(1+\alpha(1-\alpha)P_{21}+\alpha^2 P_{11}(P_{21}-P_{11}))-\alpha P_{21}R}{(1-\alpha)(R+\alpha\lambda)} \text{ if } \lambda \in [\lambda_I, \lambda_{fP_{11}}] \\ \frac{\lambda(1+\alpha P_{21})-\alpha P_{21}R}{R(1-\alpha)+\alpha\lambda} \text{ if } \lambda \in [\lambda_{fP_{11}}, P_{21}R]. \end{cases}$$

With the value of $p^*(\lambda)$, we can finish the computation. For a given $p \in [0,1]$, we can first have $p > p^*$, in which case we are done, using the values of $J(P_{11})$ and $J(P_{21})$ computed in the various cases. If $p < p^*$, as in the previous paragraph, we need to distinguish two subcases. If $fp \leq p^*$ (which can happen only when $p* > I$, i.e., $\lambda > \lambda_I$), then immediately $J(P) = \lambda/(1-\alpha)$ since the iterations remain in the passive region. Otherwise, $fp > p^*$ and we can compute

$$J(p) = \lambda + \alpha(fp)R + \alpha^2(fp)J(P_{11}) + \alpha^2(1-fp)J(P_{21}),$$

using the values of $J(P_{11})$ and $J(P_{11})$ obtained in the various cases.

4.4.5 Summary: Expressions for the Indices and the Relaxed Value Function

The previous paragraphs establish the indexability property for the two-state Markov chain under partial observations modeling a site, for all possible values of the state-transition matrix. Now we obtain Whittle's index by inverting the relation $p^*(\lambda)$ to $\lambda(p)$. We get

Theorem 4.4.2. *A two-state restless bandit under partial observations as considered in this chapter is indexable. The index $\lambda(p)$ can be computed as follows. Let $s = P_{11} - P_{21}$ (then $-1 \leq s \leq 1$), $f^n P_{21} = P_{21}\frac{1-s^{n+1}}{1-s}$, and $I = \frac{P_{21}}{1-s}$.*

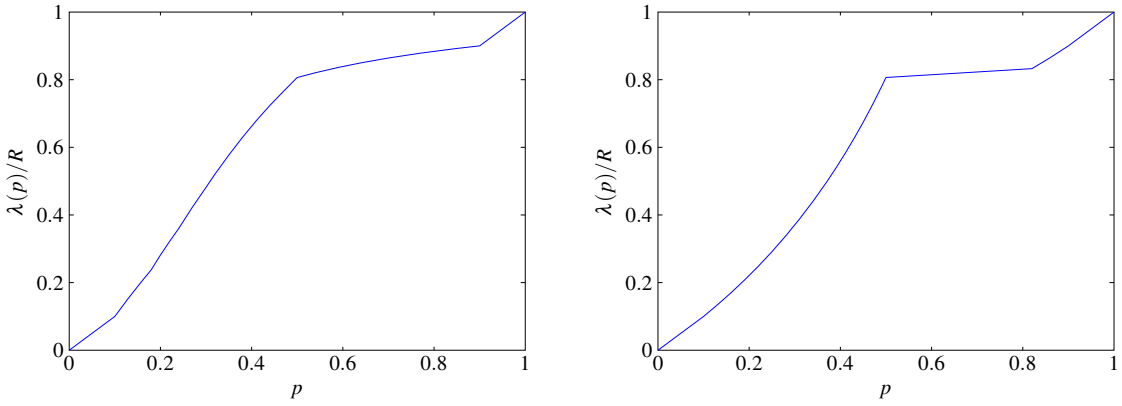
1. *Case $s = 0$ ($P_{11} = P_{21}$):*
$$\lambda(p) = pR$$

2. *Case $s = 1$ ($P_{11} = 1, P_{21} = 0$):*
$$\lambda(p) = \frac{pR}{1-\alpha(1-p)}.$$

3. *Case $0 < s < 1$ ($P_{11} > P_{21}, P_{11} - P_{21} < 1$ and note that $P_{21} < I \leq P_{11}$):*

   - *If $p \geq P_{11}$ or $p \leq P_{21}$: $\lambda(p) = pR$.*
   - *If $I \leq p < P_{11}$: $\lambda(p) = \frac{pR}{1-\alpha(P_{11}-p)}$.*
   - *If $P_{21} < p < I$: Let $k(p) = \lceil \frac{\ln(1-\frac{p}{I})}{\ln s} \rceil - 2$ (i.e., $k(p)$ is the unique integer such that $\frac{\ln(1-\frac{p}{I})}{\ln s} - 2 \leq k(p) < \frac{\ln(1-\frac{p}{I})}{\ln s} - 1$). Then let*

$$A_{k(p)} = \frac{(1-\alpha P_{11})(1-\alpha^{k(p)+2}) + \alpha^{k(p)+2}(1-\alpha)(f^{k(p)+1}P_{21})}{1-\alpha s}$$
$$B_{k(p)} = 1 - \alpha^{k(p)+2}$$
$$C_{k(p)} = \alpha - \alpha^{k(p)+2}.$$



(a) Case $P_{11} > P_{21}$ (b) Case $P_{11} < P_{21}$

Figure 4-7: Whittle indices.

We have

$$\lambda(p) = \frac{A_{k(p)} - (1-p)B_{k(p)}}{A_{k(p)} - (1-p)C_{k(p)}}R.$$

4. *Case* $s = -1$ *(*$P_{11} = 0, P_{21} = 1$*):*
   - *If* $p \geq 1/2$*:* $\lambda(p) = \frac{\alpha + p(1-\alpha)}{1+\alpha(1-\alpha)(1-p)}R$.
   - *If* $p < 1/2$*:* $\lambda(p) = \frac{p}{1-\alpha p}R$.
5. *Case* $-1 < s < 0$ *(*$P_{11} < P_{21}, P_{21} - P_{11} < 1$ *and note that* $P_{21} > I > P_{11}$*):*
   - *If* $p \geq P_{21}$ *or* $p \leq P_{11}$*:* $\lambda(p) = pR$.
   - *If* $fP_{11} \leq p < P_{21}$*:* $\lambda(p) = \frac{p+\alpha(P_{21}-p)}{1+\alpha(P_{21}-p)}R$.
   - *If* $I \leq p < fP_{11}$*:* $\lambda(p) = \frac{p+\alpha(P_{21}-p)}{1+\alpha(1-\alpha)(P_{21}-p)-\alpha^2 P_{11}s}R$.
   - *If* $P_{11} < p < I$*:* $\lambda(p) = \frac{p}{1-\alpha(p-P_{11})}R$.

Summary of the expressions for the Value Function $J(p;\lambda)$

When $\lambda \leq 0$, we have

$$J(p;\lambda) = \frac{(\alpha P_{21} + p(1-\alpha))R}{(1-\alpha)(1-\alpha(P_{11}-P_{21}))}.$$

If $\lambda \geq R$, we have

$$J(p;\lambda) = \frac{\lambda}{1-\alpha}.$$

Finally, consider the case $0 < \lambda < R$. Let $s = P_{11} - P_{21}$ (then $-1 \leq s \leq 1$), $f^n P_{21} = P_{21}\frac{1-s^{n+1}}{1-s}$, and $I = \frac{P_{21}}{1-s}$. Then we have

1. Case $s = 0$ $(i.e., P_{11} = P_{21})$:
   - if $0 < \lambda < P_{11}R$ then

$$J(p) = \begin{cases} \lambda + \frac{\alpha PR}{1-\alpha} & \text{if } p \leq \frac{\lambda}{R} \\ \left(p + \frac{\alpha P}{1-\alpha}\right)R & \text{if } p > \frac{\lambda}{R}. \end{cases}$$

- if $P_{11}R \le \lambda < R$ then
$$J(p) = \begin{cases} \frac{\lambda}{1-\alpha} & \text{if } p \le \frac{\lambda}{R} \\ pR + \frac{\alpha\lambda}{1-\alpha} & \text{if } p > \frac{\lambda}{R}. \end{cases}$$

2. Case $s = 1$ ($i.e., P_{11} = 1, P_{21} = 0$):
$$p^*(\lambda) = \frac{\lambda(1-\alpha)}{R - \alpha\lambda}, \quad J(p) = \begin{cases} \frac{\lambda}{1-\alpha} & \text{if } p \le p^* \\ \frac{\alpha\lambda}{1-\alpha} + p\frac{R-\alpha\lambda}{1-\alpha} & \text{if } p > p^*. \end{cases}$$

3. Case $0 < s < 1$ (i.e., $P_{11} > P_{21}, P_{11} - P_{21} < 1$ and note that $P_{21} < I \le P_{11}$): define
$$\lambda_I := \frac{P_{21}R}{1 - (P_{11} - P_{21})(1 + \alpha - \alpha P_{11})} \quad \text{(note: } P_{21}R < \lambda_I \le P_{11}R).$$

   - If $0 < \lambda \le P_{21}R$, then $p^*(\lambda) = \frac{\lambda}{R}$.

     If $p < p^*$, let $l = \left\lceil \frac{1}{\ln s} \ln \frac{I - p^*}{I - p} \right\rceil - 1$ and
     $$J(p) = \frac{1-\alpha^{l+1}}{1-\alpha}\lambda + \alpha^{l+1}\frac{\alpha P_{21}R}{(1-\alpha)[1-\alpha(P_{11}-P_{21})]} + \alpha^{l+1}(f^{l+1}p)\frac{R}{[1-\alpha(P_{11}-P_{21})]},$$
     if $p \ge p^*$: $J(p) = \frac{\alpha P_{21}R}{(1-\alpha)[1-\alpha(P_{11}-P_{21})]} + p\frac{R(1-\alpha)}{(1-\alpha)[1-\alpha(P_{11}-P_{21})]}$ ($i.e., l = -1$ above).

   - If $P_{21}R < \lambda < \lambda_I$, then we first find the unique integer k such that
     $$\frac{A_k - (1 - f^k P_{21})B_k}{A_k - (1 - f^k P_{21})C_k} < \frac{\lambda}{R} \le \frac{A_k - (1 - f^{k+1} P_{21})B_k}{A_k - (1 - f^{k+1} P_{21})C_k}.$$

     Then $p^*(\lambda)$ is given by (4.18), $J(P_{21})$ is given by (4.17), and $J(P_{11})$, or equivalently $R + \alpha(J(P_{11}) - J(P_{21}))$, is given by (4.11). Finally
     $$J(p) = \frac{1-\alpha^{l+1}}{1-\alpha}\lambda + \alpha^{l+2}J(P_{21}) + \alpha^{l+1}(f^{l+1}p)[R + \alpha(J(P_{11}) - J(P_{21}))],$$
     where l is defined as in the previous case $0 < \lambda \le P_{21}R$, with $l = -1$ if $p \ge p^*(\lambda)$.

   - If $\lambda_I \le \lambda < P_{11}R$, then $p^*(\lambda) = \frac{(1-\alpha P_{11})\lambda}{R - \alpha\lambda}$ and
     $$J(p) = \begin{cases} \frac{\lambda}{1-\alpha} & \text{if } p \le p^* \\ \frac{\alpha\lambda}{1-\alpha} + p\frac{R-\alpha\lambda}{1-\alpha P_{11}} & \text{if } p > p^*. \end{cases}$$

   - If $P_{11}R \le \lambda < R$, then $p^*(\lambda) = \frac{\lambda}{R}$ and
     $$J(p) = \begin{cases} \frac{\lambda}{1-\alpha} & \text{if } p \le p^* \\ pR + \frac{\alpha\lambda}{1-\alpha} & \text{if } p > p^* \end{cases}$$

4. Case $s = -1$ ($P_{11} = 0, P_{21} = 1$):

- If $0 < \lambda \leq \frac{R}{2-\alpha}$, then $p^*(\lambda) = \frac{\lambda}{R+\alpha\lambda}$ and

$$J(p;\lambda) = \begin{cases} \frac{\lambda(1-\alpha^2 p(1-\alpha))+\alpha R(1-p(1-\alpha))}{1-\alpha^2} & \text{if } p < p^* \\ \frac{R+\alpha\lambda}{1-\alpha^2}(\alpha + p(1-\alpha)) & \text{if } p \geq p^*. \end{cases}$$

- If $\frac{R}{2-\alpha} < \lambda < R$, then $p^*(\lambda) = \frac{\lambda(1+\alpha-\alpha^2)-\alpha R}{(1-\alpha)(R+\alpha\lambda)}$ (note: $p^*(\lambda) > 1/2$ then) and

$$J(p;\lambda) = \begin{cases} \frac{\lambda(1-\alpha^2 p(1-\alpha))+\alpha R(1-p(1-\alpha))}{1-\alpha^2} & \text{if } p < 1-p^* \\ \frac{\lambda}{1-\alpha} & \text{if } 1-p^* \leq p < p^* \\ \frac{R+\alpha\lambda}{1-\alpha^2}(\alpha + p(1-\alpha)) & \text{if } p \geq p^*. \end{cases}$$

5. Case $-1 < s < 0$ ($P_{11} < P_{21}, P_{21} - P_{11} < 1$ and note that $P_{21} > I > P_{11}$). Define

$$\lambda_I := \frac{P_{21}R}{1+(P_{21}-P_{11})(1-\alpha+\alpha P_{11})} \quad \lambda_{fP_{11}} := R\frac{P_{21}-(1-\alpha)P_{11}(P_{21}-P_{11})}{1+\alpha P_{11}(P_{21}-P_{11})}.$$

- If $0 < \lambda \leq P_{11}R$, then $p^*(\lambda) = \frac{\lambda}{R}$ and

$$J(p) = \begin{cases} \lambda + \alpha\left(fp + \frac{\alpha P_{21}}{1-\alpha}\right)\frac{R}{1+\alpha(P_{21}-P_{11})}, & \text{if } p < p^*, \\ \left(p + \frac{\alpha P_{21}}{1-\alpha}\right)\frac{R}{1+\alpha(P_{21}-P_{11})}, & \text{if } p \geq p^*. \end{cases}$$

- If $P_{11}R < \lambda \leq \lambda_I$, then $p^*(\lambda) = \frac{\lambda(1+\alpha P_{11})}{R+\alpha\lambda}$ and

$$J(p) = \begin{cases} \lambda + \alpha\left(fp + \frac{\alpha P_{21}}{1-\alpha}\right)\frac{(R+\alpha\lambda)}{[1+\alpha P_{21}+\alpha^2 P_{11}(P_{21}-P_{11})]}, & \text{if } p < p^*, \\ \left(p + \frac{\alpha P_{21}}{1-\alpha}\right)\frac{(R+\alpha\lambda)}{[1+\alpha P_{21}+\alpha^2 P_{11}(P_{21}-P_{11})]}, & \text{if } p \geq p^*. \end{cases}$$

- If $\lambda_I < \lambda \leq \lambda_{fP_{11}}$, then $p^*(\lambda) = \frac{\lambda(1+\alpha(1-\alpha)P_{21}+\alpha^2 P_{11}(P_{21}-P_{11}))-\alpha P_{21}R}{(1-\alpha)(R+\alpha\lambda)}$ and

$$J(p) = \begin{cases} \lambda + \alpha\left(fp + \frac{\alpha P_{21}}{1-\alpha}\right)\frac{(R+\alpha\lambda)}{(1+\alpha P_{21}+\alpha^2 P_{11}(P_{21}-P_{11}))}, & \text{if } p < p^* \text{ and } fp > p^* \\ \frac{\lambda}{1-\alpha}, & \text{if } p < p^* \text{ and } fp \leq p^* \\ \left(p + \frac{\alpha P_{21}}{1-\alpha}\right)\frac{(R+\alpha\lambda)}{(1+\alpha P_{21}+\alpha^2 P_{11}(P_{21}-P_{11}))}, & \text{if } p \geq p^* \end{cases}$$

- If $\lambda_{fP_{11}} < \lambda \leq P_{21}R$, then $p^*(\lambda) = \frac{\lambda(1+\alpha P_{21})-\alpha P_{21}R}{R(1-\alpha)+\alpha\lambda}$ and

$$J(p) = \begin{cases} \lambda + \alpha\left(fp + \frac{\alpha P_{21}}{1-\alpha}\right)\frac{(1-\alpha)R+\alpha\lambda}{1-\alpha+\alpha P_{21}}, & \text{if } p < p^* \text{ and } fp > p^* \\ \frac{\lambda}{1-\alpha}, & \text{if } p < p^* \text{ and } fp \leq p^* \\ \left(p + \frac{\alpha P_{21}}{1-\alpha}\right)\frac{(1-\alpha)R+\alpha\lambda}{1-\alpha+\alpha P_{21}}, & \text{if } p \geq p^* \end{cases}$$

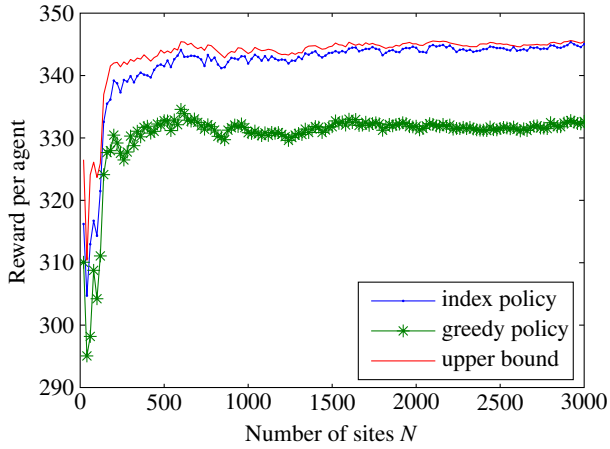


Figure 4-8: Monte-Carlo Simulation for Whittle's index policy and the greedy policy. The upper bound is computed using the subgradient optimization algorithm. We fixed $\alpha = 0.95$.

- If $P_{21}R < \lambda < R$, then $p^*(\lambda) = \frac{\lambda}{R}$ and

$$J(p) = \begin{cases} \frac{(1-\alpha+\alpha^2)\lambda}{1-\alpha} + \alpha(fp)R, & \text{if } p < p^* \text{ and } fp > p^* \\ \frac{\lambda}{1-\alpha}, & \text{if } p < p^* \text{ and } fp \leq p^* \\ pR + \frac{\alpha\lambda}{1-\alpha}, & \text{if } p \geq p^* \end{cases}$$

4.5 Simulation Results

In this section, we briefly present some simulation results illustrating the performance of the index policy and the quality of the upper bound. We generate sites with random rewards R^i within given bounds and random parameters P_{11}, P_{21}. We progressively increase the size of the problem by adding new sites and UAVs to the existing ones. We keep the ratio M/N constant, in this case $M/N = 1/20$. When generating new sites, we only ensure that $|P_{11} - P_{21}|$ is sufficiently far from 0, which is the case where the index policy departs significantly from the simple greedy policy. The upper bound is computed for each value of N using the subgradient optimization algorithm. The expected performance of the index policy and the greedy policy are estimated via Monte-Carlo simulations.

Fig. 4-8 shows the result of simulations for up to $N = 3000$ sites. We plot the reward per agent, dividing the total reward by M, for readability. We can see the consistently stronger performance of the index policy with respect to the simple greedy policy, and in fact its asymptotic quasi-optimality.

4.6 Conclusion

We have proposed the application of Whittle's work on restless bandits in the context of a mobile sensor routing problem with partial information. For given problem parameters, we can compute an upper bound on the achievable performance, and experimental results show that the performance of Whittle's index policy is often very close to the upper bound. This is in agreement with existing work on restless bandit problems for different applications. Some directions for future work include

a better understanding the asymptotic performance of the index policy and the computation of the indices for more general state spaces.

Chapter 5

Scheduling Kalman Filters

In the previous chapter, we modeled the UAV routing problem as a sequential detection problem by assuming that the sites to be observed were independent two-state Markov chains. Suppose now that the sites or targets can be modeled as independent Gaussian linear time invariant systems, and that we would like to use the mobile sensors to keep a good steady-state estimate of the states of the targets. Similarly to the previous chapter, observations can only be made by the sensors if they travel to the location of the target. This is a variation of a problem considered by Meier et al., and Athans [91, 8]. In this chapter we apply the tools and ideas developed for the RBP to this estimation problem. The RBP provides the insight to derive a tractable relaxation of the problem, which provides a bound on the achievable performance. We show how to compute this bound by solving a convex program involving linear matrix inequalities. Exploiting the additional structure of the sites evolving independently, we can decompose this program into coupled smaller dimensional problems. We give the expression of the Whittle indices in the scalar case with identical sensors. In the general case, we develop open-loop switching policies whose performance matches the lower bound arbitrarily closely as the times for the sensors to switch targets tend to zero.

5.1 Introduction

We still have M mobile sensors tracking the state of N sites or targets. We consider here a continuous-time version of the problem. Let us now assume that the sites can be described by N plants with independent linear time invariant dynamics,

$$\dot{x}_i = A_i x_i + B_i u_i + w_i, \quad x_i(0) = x_{i,0}.$$

We assume that the plant controls $u_i(t)$ are deterministic and *known* for $t \geq 0$. Each driving noise $w_i(t)$ is a stationary white Gaussian noise process with zero mean and known power spectral density matrix W_i:

$$\mathrm{Cov}(w_i(t), w_i(t')) = W_i \delta(t - t').$$

The initial conditions are random variables with known mean $\bar{x}_{i,0}$ and covariance matrices $\Sigma_{i,0}$. By independent systems we mean that the noise processes of the different plants are independent, as well as the initial conditions $x_{i,0}$. Moreover the initial conditions are assumed independent of the noise processes. We shall assume that

Assumption 5.1.1. *The matrices $\Sigma_{i,0}$ are positive definite for all $i \in \{1, \ldots, N\}$.*

This can be achieved by adding an arbitrarily small perturbation to a potentially non invertible

matrix $\Sigma_{i,0}$. This assumption is needed in our discussion to be able to use the information filter later on and to use a technical theorem on the convergence of the solutions of a periodic Riccati equation in section 5.4.3.

We assume that we have at our disposal M sensors to observe the N plants. If sensor j is used to observe plant i, we obtain measurements

$$y_{ij} = C_{ij}x_i + v_{ij}.$$

Here v_{ij} is a stationary white Gaussian noise process with power spectral density matrix V_{ij}, assumed positive definite. Also, v_{ij} is independent of the other measurement noises, process noises, and initial states. Finally, to guarantee convergence of the filters later on, we assume throughout that

Assumption 5.1.2. *For all $i \in \{1,\dots,N\}$, there exists a set of indices $j_1, j_2, \dots, j_{n_i} \in \{1,\dots,M\}$ such that the pair $(A_i, \tilde{C}_i)$ is detectable, where $\tilde{C}_i = [C_{ij_1}^T, \dots, C_{ij_{n_i}}^T]^T$.*

Assumption 5.1.3. *The pairs $(A_i, W_i^{1/2})$ are controllable, for all $i \in \{1,\dots,N\}$.*

Remark 5.1.4. It is conjectured that in the following the assumption 5.1.3 can be replaced by the following weaker assumption: the pairs $(A_i, W_i^{1/2})$ are stabilizable, for all $i \in \{1,\dots,N\}$.

Let us define

$$\pi_{ij}(t) = \begin{cases} 1 \text{ if plant } i \text{ is observed at time } t \text{ by sensor } j \\ 0 \text{ otherwise.} \end{cases}$$

We assume that each sensor can observe at most one system at each period, hence we have the constraint

$$\sum_{i=1}^{N} \pi_{ij}(t) \leq 1, \quad \forall t, \ \ j = 1,\dots,M. \tag{5.1}$$

If instead sensor j is required to be always operated, constraint (5.1) should simply be changed to

$$\sum_{i=1}^{N} \pi_{ij}(t) = 1. \tag{5.2}$$

The equality constraint is useful in scenarios involving UAVs, where it might not be possible to withdraw a vehicle from operation during the mission. The performance will of course be worse in general than with an inequality once we introduce operation costs.

We also add the following constraint, similar to the one used by Athans [8]. We suppose that each system can be observed by at most one sensor at each time, so we have

$$\sum_{j=1}^{M} \pi_{ij}(t) \leq 1, \quad \forall t, \ \ i = 1,\dots,N. \tag{5.3}$$

Similarly if system i must always be observed by some sensor, constraint (5.3) can be changed to an equality constraint

$$\sum_{j=1}^{M} \pi_{ij}(t) = 1. \tag{5.4}$$

Note that a sensor in our discussion can correspond to a combination of several physical sensors, and so the constraints above can capture seemingly more general problems where we allow for

example more that one simultaneous measurements per system. Using (5.4) we could also impose a constraint on the total number of allowed observations at each time. Indeed, consider a constraint of the form

$$\sum_{i=1}^{N}\sum_{j=1}^{M}\pi_{ij}(t)\leq p,\quad \text{for some positive integer } p.$$

This constraint means that $M-p$ sensors are required to be idle at each time. So we can create $M-p$ "dummy" systems (we should choose simple scalar stable systems to minimize computations), and associate the constraint (5.4) to each of them. Then we simply do not include the covariance matrix of these systems in the objective function below.

We consider infinite-horizon problems, mostly in the average cost version and briefly in the discounted cost version. The parameters of the models are assumed known. We wish to design an observation policy $\pi(t)=\{\pi_{ij}(t)\}$ satisfying the constraints (5.1), (5.3), and an estimator $\hat{x}_\pi$ of x, *depending at each instant only on the past and current observations produced by the observation policy*, such that the average (or total discounted) error variance is minimized, in addition to some observation costs. The policy π itself can also only depend on the past and current observations. More precisely, we wish to minimize, subject to the constraints (5.1), (5.3),

$$J_{avg}=\min_{\pi,\hat{x}_\pi}\limsup_{T\to\infty}\frac{1}{T}E\left[\int_0^T\sum_{i=1}^{N}\left((x_i-\hat{x}_{\pi,i})'T_i(x_i-\hat{x}_{\pi,i})+\sum_{j=1}^{M}\kappa_{ij}\,\pi_{ij}(t)\right)dt\right], \tag{5.5}$$

for the average cost problem, or

$$J_\beta(\Sigma_0)=\min_{\pi,\hat{x}_\pi}E\left[\int_0^\infty e^{-\beta t}\sum_{i=1}^{N}\left((x_i-\hat{x}_{\pi,i})'T_i(x_i-\hat{x}_{\pi,i})+\sum_{j=1}^{M}\kappa_{ij}\,\pi_{ij}(t)\right)dt\right], \tag{5.6}$$

in the discounted cost version. Here the constants κ_{ij} are a cost paid per unit of time when plant i is observed by sensor j. The T_i's are positive semidefinite weighting matrices.

5.1.1 Literature Review

The sensor scheduling problem presented above, except for minor variations, is an infinite horizon version of the problem considered by Athans in [8]. See also Meier et al. [91] for the discrete-time version. Athans considered only one plant to observe. We include here several plants to show how their independent evolution property can be leveraged in the computations, which is similar to the previous idea demonstrated for bandit problems (called dual decomposition method in optimization).

Discrete-time versions of this sensor selection problem have received a significant amount of attention, see e.g. [43, 99, 126, 125, 54, 111, 117]. All algorithms proposed so far, except for the greedy policy of [111] in the completely symmetric case, either run in exponential time or consist of heuristics with no performance guarantee. However, to the author's knowledge, the problem has not been shown to be intractable. We do not consider the discrete-time problem in this thesis. Finite-horizon continuous-time versions of the problem, besides the presentation of Athans [8], have also been the subject of several papers [119, 107, 10, 83]. For these problems the solutions proposed, usually based on optimal control techniques, also involve computational procedures that scale poorly with the dimension of the problem.

Somewhat surprisingly however, and with the exception of [94], it seems that the infinite-horizon continuous time version of the Kalman filter scheduling problem has not been considered

previously. Mourikis and Roumeliotis [94] consider initially also a discrete time version of the problem for a particular robotic application. However, their discrete model originates from the sampling at high rate of a continuous time system. To cope with the difficulty of determining a sensor schedule, they assume instead a model where each sensor can independently process each of the available measurements at a constant frequency, and seek the optimal measurement frequencies. In fact, they obtain these frequencies by introducing heuristically a continuous time Riccati equation, and show that the frequencies can then be computed by solving a semidefinite program. In contrast, we consider the more standard schedule-based version of the problem in continuous time, which is a priori more constraining. We show that essentially the same convex program provides in fact a *lower bound* on the cost achievable by *any* measurement policy. In addition, we provide additional insight into the decomposition of the computations of this program, which can be useful in the framework of [94] as well.

The rest of the chapter is organized as follows. Section 5.2 briefly recalls that for a fixed policy $\pi(t)$, the optimal estimator is obtained by a type of Kalman-Bucy filter. In contrast to the previous chapter, the properties of the Kalman filter (independence of the error covariance matrix with respect to measurements) imply that the remaining problem of finding the optimal scheduling policy π is a deterministic control problem. In section 5.3 we treat a simplified scalar version of the problem with identical sensors as a restless bandit problem, and provide analytical expressions of the Whittle indices and of the elements necessary to compute the performance bound. Then for the multidimensional case, treated in full generality in section 5.4, we show that the lower bound on performance can be computed as a convex program involving linear matrix inequalities. This lower bound can be approached arbitrarily closely by periodically switching policies described in section 5.4.3.

5.2 Optimal Estimator

For a given observation policy $\pi(t) = \{\pi_{ij}(t)\}_{i,j}$, the minimum variance filter is given by the Kalman-Bucy filter [65], see [8]. The state estimates $\hat{x}_\pi$, where subscripts indicate the dependency on the policy π, are all updated in parallel following the stochastic differential equation

$$\begin{aligned}
&\frac{d}{dt}\hat{x}_{\pi,i}(t) = A_i\hat{x}_{\pi,i}(t) + B_i(t)u_i(t) + \Sigma_{\pi,i}(t)\left(\sum_{j=1}^{M}\pi_{ij}(t)C_{ij}^T V_{ij}^{-1}\left(C_{ij}\hat{x}_{\pi,i}(t) - y_{ij}(t)\right)\right),\\
&\hat{x}_{\pi,i}(0) = \bar{x}_{i,0}.
\end{aligned}$$

The resulting estimator is unbiased and the error covariance matrix $\Sigma_{\pi,i}(t)$ for site i verifies the matrix Riccati differential equation

$$\begin{aligned}
&\frac{d}{dt}\Sigma_{\pi,i}(t) = A_i\Sigma_{\pi,i}(t) + \Sigma_{\pi,i}(t)A_i^T + W_i - \Sigma_{\pi,i}(t)\left(\sum_{j=1}^{M}\pi_{ij}(t)C_{ij}^T V_{ij}^{-1}C_{ij}\right)\Sigma_{\pi,i}(t), \qquad (5.7)\\
&\Sigma_{\pi,i}(0) = \Sigma_{i,0}.
\end{aligned}$$

With this result, we can reformulate the optimization of the observation policy as a *deterministic* optimal control problem. Rewriting

$$E\left((x_i - \hat{x}_i)'T_i(x_i - \hat{x}_i)\right) = \mathrm{Tr}\left(T_i\Sigma_i\right),$$

the problem is to minimize

$$\gamma = \min_{\pi} \limsup_{T\to\infty} \frac{1}{T} \left[\int_0^T \sum_{i=1}^{N} \left(\mathrm{Tr}\left(T_i \Sigma_{\pi,i}(t)\right) + \sum_{j=1}^{M} \kappa_{ij}\, \pi_{ij}(t) \right) dt \right], \tag{5.8}$$

or

$$J_\beta(\Sigma_0) = \min_{\pi} \left[\int_0^\infty e^{-\beta t} \sum_{i=1}^{N} \left(\mathrm{Tr}\left(T_i \Sigma_{\pi,i}(t)\right) + \sum_{j=1}^{M} \kappa_{ij}\, \pi_{ij}(t) \right) dt \right], \tag{5.9}$$

subject to the constraints (5.1), (5.3), or their equality versions, and the dynamics (5.7).

5.3 Sites with One-Dimensional Dynamics and Identical Sensors

We assume in this section that

1. the sites or targets have one-dimensional dynamics, i.e., $x_i \in \mathbb{R}$, $i = 1,\ldots,N$.

2. all the sensors are identical, i.e. $C_{ij} = C_i, V_{ij} = V_i, \kappa_{ij} = \kappa_i,\ j = 1,\ldots,M$.

Because of condition 2, we can simplify the problem formulation introduced above so that it corresponds exactly to the type of bandit problems introduced in chapter 3. For this purpose, we define

$$\pi_i(t) = \begin{cases} 1 \text{ if plant } i \text{ is observed at time } t \text{ by a sensor} \\ 0 \text{ otherwise.} \end{cases}$$

Note that a constraint (5.4) for some system i can be eliminated, by removing one available sensor, which is always measuring the system i. Constraints (5.2) and (5.3) can then be replaced by the single constraint

$$\sum_{i=1}^{N} \pi_i(t) = M,\ \forall t.$$

This constraint means that at each period, exactly M of the N sites are observed. We treat this case in the following, as it is the standard formulation of restless bandit problems, but the equality sign can be changed to an inequality.

To obtain the lower bound, we relax the constraint to

$$\limsup_{T\to\infty} \frac{1}{T} \int_0^T \sum_{i=1}^{N} \pi_i(t) dt = M,$$

for the average cost problem, and

$$\sum_{i=1}^{N} \int_0^\infty e^{-\beta t} \pi_i(t) dt = \frac{M}{\beta},$$

for the discounted cost problem. As explained in chapter 3, we adjoin this constraint using a (scalar) Lagrange multiplier λ and we are led to consider for each site

$$\gamma^i(\lambda) = \min_{\pi_i} \limsup_{T\to\infty} \frac{1}{T} \int_0^T \mathrm{Tr}\left(T_i \Sigma_{\pi,i}(t)\right) + (\kappa_i + \lambda)\, \pi_i(t) dt,$$

for the average cost problem. Here κ_i is the cost per time unit for observing site i. For the discounted cost problem, we need to consider for each site

$$J_\beta^i(\Sigma_{i,0},\lambda) = \min_{\pi_i} \int_0^\infty e^{-\beta t} \left\{ \mathrm{Tr}\left(T_i \Sigma_{\pi,i}(t)\right) + (\kappa_i + \lambda)\, \pi_i(t) \right\} dt.$$

The dynamics of $\Sigma_{\pi,i}$ are now given by

$$\begin{aligned} &\frac{d}{dt}\Sigma_{\pi,i}(t) = A_i\Sigma_{\pi,i}(t) + \Sigma_{\pi,i}(t)A_i' + W_i - \pi_i(t)\,\Sigma_{\pi,i}(t)C_i'V_i^{-1}C_i\Sigma_{\pi,i}(t), \\ &\Sigma_{\pi,i}(0) = \Sigma_{i,0}. \end{aligned} \tag{5.10}$$

If the dynamics of the sites are one dimensional, i.e., $\Sigma_i \in \mathbb{R}$, we can solve this optimal control problem for each site analytically. That is, we obtain an analytical expression of the dual function which provides a lower bound on the cost for each λ. Additionally, we can obtain an analytical expression for the Whittle indices. The expression of the Whittle's indices for first order deterministic systems was given by Whittle [136] and Dusonchet [39], assuming that the smooth-fit principle holds. In fact, their expression is not valid over the whole state-space and we correct this mistake in this section.

We will also change the notation slightly for this one-dimensional problem and define

$$q_i := W_i, \quad r_i := V_i.$$

We can consider the problem for a single site, dropping the index i. The dynamical evolution of the covariance obeys the equation

$$\dot{\Sigma} = 2a\Sigma + q - \pi\frac{c^2}{r}\Sigma^2, \quad \text{with } \pi(t) \in \{0,1\}.$$

The HJB equation is

$$\gamma(\lambda) = \min\left\{ \Sigma + (2a\Sigma+q)h'(\Sigma;\lambda),\, \Sigma + \kappa + \lambda + (2a\Sigma + q - \frac{c^2}{r}\Sigma^2)h'(\Sigma;\lambda) \right\},$$

in the average cost case, and

$$\beta V(\Sigma;\lambda) = \min\left\{ \Sigma + (2a\Sigma+q)V'(\Sigma;\lambda),\, \Sigma + \kappa + \lambda + (2a\Sigma + q - \frac{c^2}{r}\Sigma^2)V'(\Sigma;\lambda) \right\},$$

in the discounted-cost case. We will use the following notation. Consider the algebraic Riccati equation

$$2ax + q - \frac{c^2}{r}x^2 = 0.$$

Note that by the observability assumption 5.1.2, we have $c > 0$. Then, this equation has two roots

$$x_1 = \frac{a - \sqrt{a^2 + c^2 q/r}}{c^2/r}, \quad x_2 = \frac{a + \sqrt{a^2 + c^2 q/r}}{c^2/r}.$$

x_2 is strictly positive, unless $a = q = x_2 = 0$. x_1 is strictly negative, or 0 if $a \geq 0$ and $q = 0$. By assumption 5.1.3, the pair (a,q) is controllable. In fact, stabilizability is sufficient to imply $a < 0$ if $q = 0$ and so x_1 is strictly negative and x_2 is strictly positive.

5.3.1 Average-Cost Criterion

We perform the calculations for the average cost criterion, which leads to simpler formulas. A similar treatment is possible for the discounted cost criterion. First, we can treat the case $\kappa+\lambda \leq 0$ immediately. Then it is obviously optimal to always observe, and we get, letting $\Sigma = x_2$ in the HJB equation:

$$\gamma(\lambda) = x_2 + \kappa + \lambda.$$

So from now on we can assume $\lambda \geq -\kappa$. Note that here λ has the interpretation of a tax for activity instead of a subsidy for passivity as in chapter 3.

Now let us temporarily assume the following result on the form of the optimal policy. Its validity can be verified a posteriori from the formulas obtained below, using the fact that the dynamic programming equation provides a sufficient condition for optimality of a solution.

Conjecture 5.3.1. *The optimal policy is a threshold policy, i.e., it observes the system for $\Sigma \geq T$ and does not observe for $\Sigma < T$, for some $T \in \mathbb{R}_+$.*

Now we would like to obtain the value of the average-cost $\gamma(\lambda)$ and of the threshold $T(\lambda)$ or its inverse $\lambda(T)$. Note that we already know $T(\lambda) = 0$ for $\lambda \leq -\kappa$. Inverting the relation $\lambda \mapsto T(\lambda)$ gives the Whittle index, assuming that $T(\lambda)$ is an increasing function of λ.

Case $T \leq x_2$

In this case, we obtain as before

$$\gamma(\lambda) = x_2 + \kappa + \lambda. \tag{5.11}$$

This is intuitively clear for $T < x_2$: even when observing the system all the time, the variance still converges in finite time to a neighborhood of x_2. Since this neighborhood is in the active region by hypothesis, after potentially a transition period (if the variance started at a value smaller than T), we should observe always, and hence the average cost is the same as for the policy that always observes.

By continuity at the interface between the active and passive regions, we have

$$\begin{aligned}
&T + (2aT+q)h'(T) = T + (\kappa+\lambda) + (2aT+q-\frac{c^2}{r}T^2)h'(T)\\
&\kappa+\lambda = \frac{c^2}{r}T^2 h'(T).
\end{aligned}$$

We have then

$$\begin{aligned}
&\frac{c^2}{r}T^2(x_2+\kappa+\lambda) = \frac{c^2}{r}T^3 + (2aT+q)(\kappa+\lambda)\\
&\left(-\frac{c^2}{r}T^2 + 2aT + q\right)(\kappa+\lambda) = \frac{c^2}{r}T^2(x_2-T)\\
&-(T-x_2)(T-x_1)(\kappa+\lambda) = T^2(x_2-T) \quad \text{(assuming } c \neq 0)\\
&T^2 - T(\kappa+\lambda) + x_1(\kappa+\lambda) = 0\\
\text{so} \quad &\lambda(T) = -\kappa + \frac{T^2}{T-x_1}, \quad T(\lambda) = \frac{\kappa+\lambda+\sqrt{(\kappa+\lambda)(\kappa+\lambda-4x_1)}}{2}.
\end{aligned} \tag{5.12}$$

Expression (5.13) and (5.12) are valid under the condition

$$T(\lambda) = \frac{\kappa+\lambda+\sqrt{(\kappa+\lambda)(\kappa+\lambda-4x_1)}}{2} \leq x_2.$$

Note from (5.12) that $T \to \lambda(T)$ is indeed an increasing function.

Case $T > x_2$

It turns out that in this case we must distinguish between stable and unstable systems. For a stable system ($a < 0$), the equation

$$2ax + q = 0$$

has a strictly positive solution $x_e = -\frac{q}{2a} > x_2$.

Stable System ($a < 0$) **with** $T \geq x_e$:

In this case we know that x_e is in the passive region. Hence, with $\Sigma = x_e$ in the HJB equation, we get

$$\gamma(\lambda) = x_e. \tag{5.13}$$

Then we have again

$$\kappa+\lambda = \frac{c^2}{r}T^2 h'(T),$$

and now

$$T + (2aT+q)h'(T) = x_e.$$

Hence

$$\begin{aligned}
&\frac{c^2}{r}T^2(x_e - T) = (2aT+q)(\kappa+\lambda) = 2a(T-x_e)(\kappa+\lambda) \\
&\lambda(T) = -\kappa + \frac{c^2T^2}{2|a|r}, \quad T(\lambda) = \frac{\sqrt{2|a|r(\lambda+\kappa)}}{|c|}.
\end{aligned} \tag{5.14}$$

Stable System ($a < 0$) **with** $x_2 < T < x_e$**, or non-stable system** ($a \geq 0$):

If the system is marginally stable or unstable, we cannot define x_e. We can think of this case as $x_e \to \infty$ as $a \to 0_-$, and treat it simultaneously with the case where the system is stable and $x_2 < T < x_e$. Then x_2 is in the passive region, and x_e is in the active region, so the prefactors of $h'(x)$ in the HJB equation do not vanish. There is no immediate relation providing the value of $\gamma(\lambda)$. We will use the smooth-fit principle to handle this case and obtain the expression of the Whittle indices, following [136]. Note that this approach alone is insufficient as it misses the cases treated above.

Theorem 5.3.1 ([136],[39])**.** *Consider a continuous time one-dimensional project $x(t) \in \mathbb{R}$ satisfying*

$$\dot{x}(t) = a_k(x), \quad k = 0, 1,$$

with passive and active cost rates $r_k(x)$, $k = 0, 1$. Assume that $a_0(x)$ does not vanish in the optimal passive region, and $a_1(x)$ does not vanish in the optimal active region. Then the Whittle index is given by

$$\lambda(x) = r_0(x) - r_1(x) + \frac{[a_1(x) - a_0(x)][a_0(x)(r_1)'(x) - a_1(x)(r_0)'(x)]}{a_0(x)[(a_1)'(x) - \beta] - a_1(x)[(a_0)'(x) - \beta]},$$

in the discounted case, and

$$\lambda(x) = r_0(x) - r_1(x) + \frac{[a_1(x) - a_0(x)][a_0(x)(r_1)'(x) - a_1(x)(r_0)'(x)]}{a_0(x)(a_1)'(x) - a_1(x)(a_0)'(x)},$$

in the average-cost case.

Remark 5.3.2. The assumption that a_0 and a_1 do not vanish, capturing the cases previously studied, is missing from [136],[39].

Proof. We give for completeness some details skipped in the proofs of [136], [39, p.53], which use the smooth-fit principle. A more rigorous proof should be possible, using variational inequalities. The HJB equation for the discounted-cost case is

$$\beta V(x) = \min\{r_0(x) + a_0 V'(x),\, r_1(x) + \lambda + a_1 V'(x)\}.$$

Hence, under our non vanishing assumption for a_0 and a_1

$$V'(x) = \begin{cases} \frac{\beta V - r_0}{a_0} & \text{in the passive region} \\ \frac{\beta V - r_1 - \lambda}{a_1} & \text{in the active region.} \end{cases} \tag{5.15}$$

Continuity of V on the boundary gives

$$a_0 V' + r_0 = a_1 V' + r_1 + \lambda \quad \text{on the active-passive boundary.}$$

Looking at the continuity of the second derivative of the value function on the boundary (assuming the smooth-fit principle to be valid), we obtain the additional relation:

$$\begin{aligned} \frac{a_0(\beta V' - r_0') - a_0'(\beta V - r_0)}{a_0^2} &= \frac{a_1(\beta V' - r_1') - a_1'(\beta V - r_1 - \lambda)}{a_1^2} \quad \text{on the active-passive boundary} \\ \frac{\beta V' - r_0' - a_0' V'}{a_0} &= \frac{\beta V' - r_1' - a_1' V'}{a_1} \quad \text{using (5.15)} \\ V'(a_1(\beta - a_0') - a_0(\beta - a_1')) &= a_1 r_0' - a_0 r_1' \,. \end{aligned}$$

Hence we obtain,

$$\lambda = r_0 - r_1 + \frac{(a_0 - a_1)(a_1 r_0' - a_0 r_1')}{a_1(\beta - a_0') - a_0(\beta - a_1')}.$$

The average-cost result is obtained for $\beta = 0$ or directly by the same type of argument.

□

Corollary 5.3.3. *The Whittle index for the case $x_2 < T < x_e$ is given by:*

$$\lambda(T) = -\kappa + \frac{c^2}{2r} \frac{T^3}{(a + \beta/2)T + q} \tag{5.16}$$

in the discounted cost case, and

$$\lambda(T) = -\kappa + \frac{c^2}{2r} \frac{T^3}{aT + q} \tag{5.17}$$

in the average cost case.

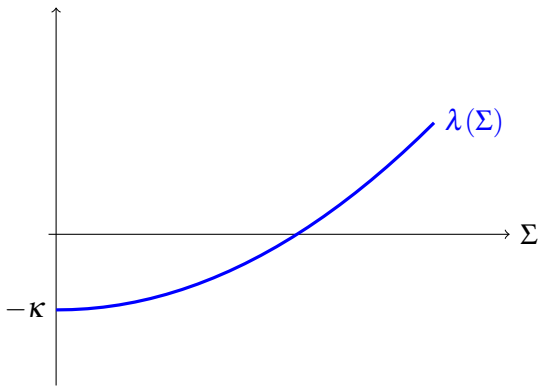


Figure 5-1: Whittle index for one-dimensional dynamics.

Proof. We have

$$a^0(\Sigma) = 2a\Sigma + q, \quad a^1(\Sigma) = 2a\Sigma + q - \frac{c^2}{r}\Sigma^2$$
$$r^0(\Sigma) = \Sigma, \quad r^1(\Sigma) = \Sigma + \kappa$$

We get in theorem 5.3.1

$$\lambda(\Sigma) = -\kappa + \frac{\frac{-c^2\Sigma^2}{r}\frac{c^2\Sigma^2}{r}}{(2a\Sigma+q)(2a-2\frac{c^2\Sigma}{r}-\beta) - (2a\Sigma+q-\frac{c^2}{r}\Sigma^2)(2a-\beta)}$$
$$\lambda(\Sigma) = -\kappa - \frac{c^4}{r^2}\frac{\Sigma^4}{-(2a\Sigma+q)2\frac{c^2}{r}\Sigma + \frac{c^2}{r}\Sigma^2(2a-\beta)}$$
$$\lambda(\Sigma) = -\kappa - \frac{c^4}{r^2}\frac{\Sigma^3}{-2a\frac{c^2}{r}\Sigma - 2\frac{c^2}{r}q - \beta\frac{c^2}{r}\Sigma}$$
$$\lambda(\Sigma) = -\kappa + \frac{c^2}{2r}\frac{\Sigma^3}{(a+\beta/2)\Sigma + q}$$

The average-cost expression is obtained by formally setting $\beta = 0$. □

With the value of the Whittle index, we can finish the computation of the lower bound $\gamma(\lambda)$ for the case $x_2 < T < x_e$. Our task of solving the HJB equation has been simplified by the use of the smooth-fit principle leading to the expression of the Whittle indices. Inverting the relation (5.17), we obtain, for a given value of λ, the boundary $T(\lambda)$ between the passive and active regions. $T(\lambda)$ verifies the depressed cubic equation

$$X^3 - \frac{2r}{c^2}(\lambda+\kappa)aX - \frac{2r}{c^2}(\lambda+\kappa)q = 0. \tag{5.18}$$

For $\lambda + \kappa \geq 0$, by Descartes' rule of signs, this polynomial has exactly one positive root, which is $T(\lambda)$.

The HJB equation then reduces to

$$\gamma(\lambda) = x + h'(x)(2ax+q), \quad \text{for } x < T(\lambda) \tag{5.19}$$

$$\gamma(\lambda) = x + \kappa + \lambda + h'(x)(2ax+q-\frac{c^2}{r}x^2), \quad \text{for } x \geq T(\lambda), \tag{5.20}$$

with h, h' and h'' assumed continuous on the boundary between the active and passive regions. Now for $x_2 < T(\lambda) < x_e$, with $x = T(\lambda) > 0$ in the HJB equation, eliminating $h'(T(\lambda);\lambda)$, we get

$$\gamma(\lambda) = T(\lambda) + \kappa + \lambda + (\gamma - T(\lambda))\left(1 - \frac{c^2}{r}\frac{(T(\lambda))^2}{2aT(\lambda)+q}\right)$$

$$(\gamma(\lambda) - T(\lambda))\left(\frac{c^2}{r}\frac{(T(\lambda))^2}{2aT(\lambda)+q}\right) = \kappa + \lambda$$

$$\gamma(\lambda) = T(\lambda) + \frac{r(\kappa+\lambda)(2aT(\lambda)+q)}{c^2(T(\lambda))^2}, \quad \text{for } x_2 < T(\lambda) < x_e.$$

Summary:

Theorem 5.3.2. *The bandits in the Kalman filter scheduling problem are indexable. For bandit i, the Whittle index $\lambda_i(\Sigma_i)$ is given as follows.*

$$\lambda_i(\Sigma_i) = \begin{cases} -\kappa_i + \frac{\Sigma_i^2}{\Sigma_i - x_{1,i}} & \text{if } \Sigma_i \leq x_{2,i}, \\ -\kappa_i + \frac{c_i^2}{2r_i}\frac{\Sigma_i^3}{a_i\Sigma_i + q_i} & \text{if } x_{2,i} < \Sigma_i < x_{e,i}, \\ -\kappa_i + \frac{c_i^2\Sigma_i^2}{2|a_i|r_i} & \text{if } x_{e,i} \leq \Sigma_i, \end{cases}$$

with the convention $x_{e,i} = +\infty$ if $a_i \geq 0$. The lower bound on the achievable performance is obtained by maximizing the concave function

$$\gamma(\lambda) = \sum_{i=1}^{N} \gamma^i(\lambda) - \lambda M$$

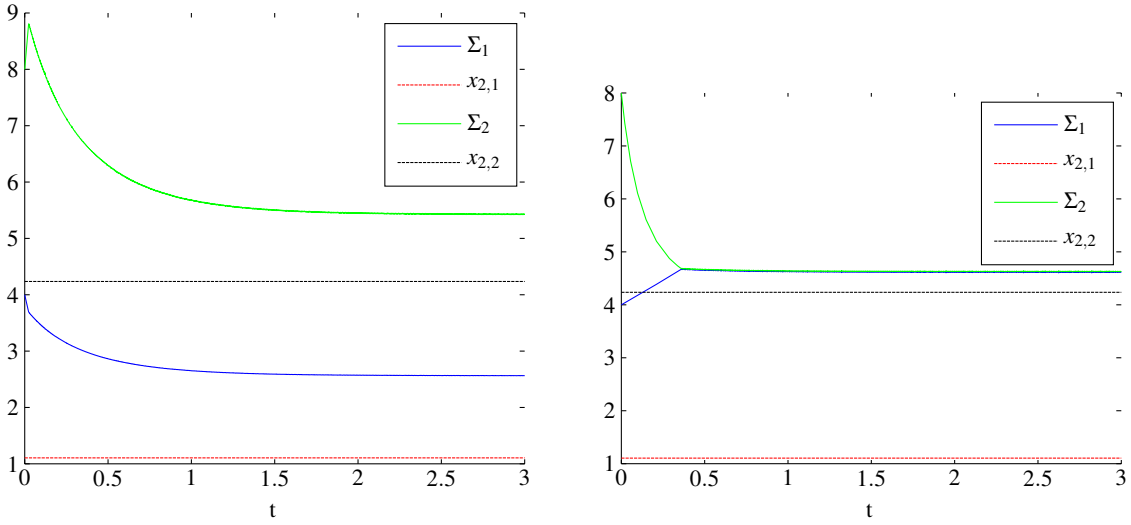
over λ, where the term $\gamma^i(\lambda)$ is given by

$$\gamma^i(\lambda) = \begin{cases} x_{2,i} + \kappa_i + \lambda & \text{if } \lambda \leq \lambda_i(x_{2,i}), \\ T_i(\lambda) + \frac{r_i(\kappa_i+\lambda)(2a_iT_i(\lambda)+q_i)}{c_i^2(T_i(\lambda))^2} & \text{if } \lambda_i(x_{2,i}) < \lambda < \lambda_i(x_{e,i}), \text{ with } T_i(\lambda) \text{ positive root of (5.18),} \\ x_{e,i} & \text{if } \lambda_i(x_{e,i}) \leq \lambda. \end{cases}$$

Proof. The indexability comes from the fact that the indices $\lambda(\Sigma)$ are verified to be monotonically increasing functions of Σ. λ is a tax for activity in the formulation of this chapter. Inverting the relation we obtain $T(\lambda)$ as the variance for which we are indifferent between the active and passive actions. As we increase λ, $T(\lambda)$ increases and the passive region (the interval $[0, T(\lambda)]$) increases. □

5.3.2 Numerical Simulations

Figure 5-2 provides an example of sample covariance trajectories obtained using the feedback control based on the restless bandit indices. There are two (unstable) systems and one sensor, with



(a) Whittle index policy.

(b) Greedy policy.

Figure 5-2: Comparison of the Whittle index policy and the greedy policy for a problem with two systems and one sensor. Also shown on the figure are the steady state covariances $x_{2,i}$ obtained if system i could be always observed. The cost of the Whittle policy is 8 and matches the lower bound, whereas the performance of the greedy policy is 9.2.

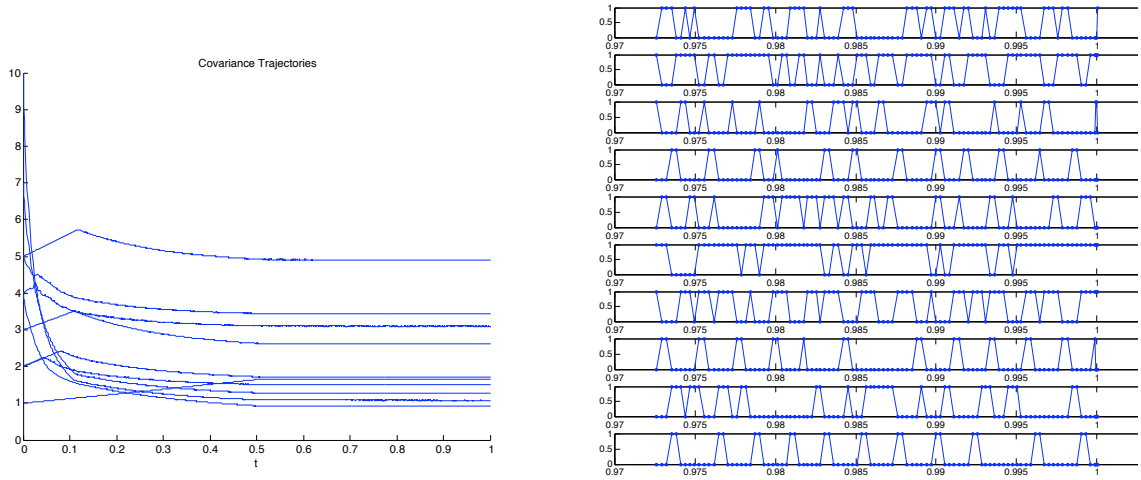
identical parameters c_i, q_i, r_i equal to 1, for $i = 1,2$, and $a_1 = 0.1$, $a_2 = 2$. We set $\kappa_i = 0$, $i = 1,2$ in this example. We compare the Whittle index policy to the greedy policy, which simply measures at each time the system with the largest covariance. As can be seen from the figure, the effect of this greedy policy is to make the steady state values of the covariances of the two systems roughly equal. In contrast, the policy based on restless bandit indices tends to make the steady state Whittle indices converge to the same value. Note that in terms of computational cost, the bandit index policy, using the analytical formulas derived above, scales as well as the greedy policy with the number of sites, since it simply requires ordering N numbers.

Figure 5-3 shows an example of covariance trajectories and a sample of the controls for a problem with 10 sites and 4 sensors. In the simulations performed so far, the performance of the Whittle index policy has matched closely the lower bound, with a gap of the order of the numerical precision involved in the performance simulation. The question of determining if this policy is optimal or of providing an a priori performance bound remains open.

5.4 Multidimensional Systems

Generalizing the computations of the previous section to multidimensional systems requires solving the corresponding optimal problem in higher dimensions, for which it is not clear that a closed form solution exist. Moreover we have considered in section 5.3 a particular case of the sensor scheduling problem where all sensors are identical. Here we return to the general multidimensional problem and sensors with possibly distinct characteristics, as described in the introduction.

Treating the average cost problem, we show that computing the value of a lower bound similar to the one used in section 5.3 reduces to a convex optimization problem involving, at worst, Linear Matrix Inequalities (LMI) whose size grows polynomially with the problem essential parameters. Moreover, one can further decompose the computation of this convex program into N coupled sub-



(a) Covariance Trajectories.

(b) Index policy for each site.

Figure 5-3: N=10 independent systems, unstable, with randomly generated parameters. Here M=4. The lower bound on the achievable performance is 41.08. The simulation evaluation of the performance gives a cost of approximately 41.

problems as in the standard restless bandit case.

5.4.1 Performance Bound

Let us repeat the deterministic optimal control problem under consideration:

$$\min_{\pi} \limsup_{T\to\infty} \frac{1}{T}\int_0^T \sum_{i=1}^{N}\left\{\mathbf{Tr}(T_i\Sigma_i(t)) + \sum_{j=1}^{M}\kappa_{ij}\pi_{ij}(t)\right\}dt, \tag{5.21}$$

$$\text{s.t. } \dot{\Sigma}_i(t) = A_i\Sigma_i + \Sigma_i A_i^T + W_i - \Sigma_i\left(\sum_{j=1}^{M}\pi_{ij}(t)C_{ij}^T V_{ij}^{-1}C_{ij}\right)\Sigma_i,\ i=1\ldots,N, \tag{5.22}$$

$$\pi_{ij}(t)\in\{0,1\},\ \forall t, i=1\ldots,N,\ j=1,\ldots,M,$$

$$\sum_{i=1}^{N}\pi_{ij}(t) = 1,\ \forall t,\ \ j=1,\ldots,M, \tag{5.23}$$

$$\sum_{j=1}^{M}\pi_{ij}(t) \le 1,\ \forall t,\ \ i=1,\ldots,N, \tag{5.24}$$

$$\Sigma_i(0) = \Sigma_{i,0},\ i=1,\ldots,N.$$

Here we consider the constraints (5.2) and (5.3), but any combination of inequality and equality constraints from (5.1)-(5.4) can be used without change in the argument for the derivation of the performance bound.

We define the following quantities:

$$\tilde{\pi}_{ij}(T) = \frac{1}{T}\int_0^T \pi_{ij}(t)dt, \forall T. \tag{5.25}$$

Since $\pi_{ij}(t)\in\{0,1\}$ we must have $0\le\tilde{\pi}_{ij}(T)\le 1$. Our first goal, following the now standard idea

exploited in the restless bandit problem, is to obtain a lower bound on the cost of the optimal control problem in terms of the $\tilde{\pi}_{ij}(T)$ instead of the functions $\pi_{ij}(t)$.

It will be easier to work with the information matrices

$$Q_i(t) = \Sigma_i^{-1}(t).$$

Note that invertibility of $\Sigma_i(t)$ is guaranteed by our assumption 5.1.1, as a consequence of [24, theorem 21.1]. Hence we replace the dynamics (5.22) by the equivalent

$$\dot{Q}_i = -Q_i A_i - A_i^T Q_i - Q_i W_i Q_i + \sum_{j=1}^{M} \pi_{ij}(t) C_{ij}^T V_{ij}^{-1} C_{ij}, \quad i = 1, \ldots, N. \tag{5.26}$$

Let us define, for all T,

$$\tilde{\Sigma}_i(T) := \frac{1}{T}\int_0^T \Sigma_i(t)dt, \quad \tilde{Q}_i(T) := \frac{1}{T}\int_0^T Q_i(t)dt.$$

By linearity of the trace operator, we can rewrite the objective function

$$\limsup_{T\to\infty} \sum_{i=1}^{N} \left\{ \mathbf{Tr}(T_i \tilde{\Sigma}_i(T)) + \sum_{j=1}^{M} \kappa_{ij} \tilde{\pi}_{ij}(T) \right\}.$$

Let $\mathbb{S}^n, \mathbb{S}^n_+, \mathbb{S}^n_{++}$ denote the set of symmetric, symmetric positive semidefinite and symmetric positive definite matrices respectively. A function $f : \mathbb{R}^m \to \mathbb{S}^n$ is called *matrix convex* if and only if for all $x, y \in \mathbb{R}^m$ and $\alpha \in [0,1]$, we have

$$f(\alpha x + (1-\alpha)y) \preceq \alpha f(x) + (1-\alpha) f(y),$$

where $\preceq$ refers to the usual partial order on S^n, i.e., $A \preceq B$ if and only if $B - A \in S^n_+$. Equivalently, f is matrix convex if the scalar function $x \mapsto z^T f(x) z$ is convex for all vectors z. The following lemma will be useful

Lemma 5.4.1. *The functions*

$$\begin{array}{ll} \mathbb{S}^n_{++} \to \mathbb{S}^n_{++} & \mathbb{S}^n \to \mathbb{S}^n \\ X \mapsto X^{-1} & X \mapsto XWX \end{array}$$

for $W \in \mathbb{S}^n_{++}$, are matrix convex.

Proof. See [23, p.76, p.110]. □

A consequence of this lemma is that Jensen's inequality is valid for these functions. We use it first as follows

$$\forall T, \ \left(\frac{1}{T}\int_0^T \Sigma_i(t)dt\right)^{-1} \preceq \frac{1}{T}\int_0^T Q_i(t)dt = \tilde{Q}_i(T),$$

hence

$$\forall T, \ \tilde{\Sigma}_i(T) \succeq (\tilde{Q}_i(T))^{-1}.$$

and so

$$\mathbf{Tr}(T_i \tilde{\Sigma}_i(T)) \geq \mathbf{Tr}(T_i (\tilde{Q}_i(T))^{-1}).$$

Next, integrating (5.26), we have

$$\frac{1}{T}(Q_i(T)-Q_{i,0}) = -\tilde{Q}_i(T)A_i - A_i^T\tilde{Q}_i(T) - \frac{1}{T}\int_0^T Q_i(t)W_iQ_i(t)dt + \sum_{j=1}^{M}\left(\frac{1}{T}\int_0^T \pi_{ij}(t)\right)C_{ij}^TV_{ij}^{-1}C_{ij}$$

$$\frac{1}{T}(Q_i(T)-Q_{i,0}) = -\tilde{Q}_i(T)A_i - A_i^T\tilde{Q}_i(T) - \frac{1}{T}\int_0^T Q_i(t)W_iQ_i(t)dt + \sum_{j=1}^{M}\tilde{\pi}_{ij}(T)C_{ij}^TV_{ij}^{-1}C_{ij}.$$

Using Jensen's inequality and lemma 5.4.1 again, we have

$$\frac{1}{T}\int_0^T Q_i(t)W_iQ_i(t) \succeq \tilde{Q}_i(T)W_i\tilde{Q}_i(T),$$

and so we obtain

$$\frac{1}{T}(Q_i(T)-Q_{i,0}) \preceq -\tilde{Q}_i(T)A_i - A_i^T\tilde{Q}_i(T) - \tilde{Q}_i(T)W_i\tilde{Q}_i(T) + \sum_{j=1}^{M}\tilde{\pi}_{ij}(T)C_{ij}^TV_{ij}^{-1}C_{ij}. \tag{5.27}$$

Last, since $Q_i(T) \succeq 0$, this implies, for all T,

$$\tilde{Q}_i(T)A_i + A_i^T\tilde{Q}_i(T) + \tilde{Q}_i(T)W_i\tilde{Q}_i(T) - \sum_{j=1}^{M}\tilde{\pi}_{ij}(T)C_{ij}^TV_{ij}^{-1}C_{ij} \preceq \frac{Q_{i,0}}{T}. \tag{5.28}$$

So we see that for a fixed policy π and any time T, the quantity

$$\sum_{i=1}^{N}\left\{\mathbf{Tr}(T_i\tilde{\Sigma}_i(T)) + \sum_{j=1}^{M}\kappa_{ij}\tilde{\pi}_{ij}(T)\right\} \tag{5.29}$$

is lower bounded by the quantity

$$\sum_{i=1}^{N}\left\{\mathbf{Tr}(T_i(\tilde{Q}_i(T))^{-1}) + \sum_{j=1}^{M}\kappa_{ij}\tilde{\pi}_{ij}(T)\right\},$$

where the matrices $\tilde{Q}_i(T)$ and the number $\tilde{\pi}_{ij}(T)$ are subject to the constraints (5.28) as well as

$$0 \le \tilde{\pi}_{ij}(T) \le 1,\ \sum_{i=1}^{N}\tilde{\pi}_{ij}(T) = 1,\ j = 1,\ldots,M,\ \sum_{j=1}^{M}\tilde{\pi}_{ij}(T) \le 1,\ i = 1,\ldots,N.$$

Hence for any T, the quantity $Z^*(T)$ defined below is a lower bound on the value of (5.29) for any

choice of policy π

$$Z^*(T) = \min_{\tilde{Q}_i, p_{ij}} \quad \sum_{i=1}^{N} \left\{ \mathbf{Tr}(T_i \tilde{Q}_i^{-1}) + \sum_{j=1}^{M} \kappa_{ij} p_{ij} \right\}, \tag{5.30}$$

$$\text{s.t. } \tilde{Q}_i A_i + A_i^T \tilde{Q}_i + \tilde{Q}_i W_i \tilde{Q}_i - \sum_{j=1}^{M} p_{ij} C_{ij}^T V_{ij}^{-1} C_{ij} \preceq \frac{Q_{i,0}}{T},\ i = 1\ldots,N, \tag{5.31}$$

$$\tilde{Q}_i \succ 0,\ i = 1\ldots,N,$$

$$0 \leq p_{ij} \leq 1,\ i = 1\ldots,N,\ j = 1,\ldots,M,$$

$$\sum_{i=1}^{N} p_{ij} = 1,\ j = 1,\ldots,M,$$

$$\sum_{j=1}^{M} p_{ij} \leq 1,\ i = 1,\ldots,N.$$

Consider now the following program, where the right-hand side of (5.31) has been replaced by 0:

$$Z^* = \min_{\tilde{Q}_i, p_{ij}} \quad \sum_{i=1}^{N} \left\{ \mathbf{Tr}(T_i \tilde{Q}_i^{-1}) + \sum_{j=1}^{M} \kappa_{ij} p_{ij} \right\}, \tag{5.32}$$

$$\text{s.t. } \tilde{Q}_i A_i + A_i^T \tilde{Q}_i + \tilde{Q}_i W_i \tilde{Q}_i - \sum_{j=1}^{M} p_{ij} C_{ij}^T V_{ij}^{-1} C_{ij} \preceq 0,\ i = 1\ldots,N, \tag{5.33}$$

$$\tilde{Q}_i \succ 0,\ i = 1\ldots,N,$$

$$0 \leq p_{ij} \leq 1,\ i = 1\ldots,N,\ j = 1,\ldots,M,$$

$$\sum_{i=1}^{N} p_{ij} \leq 1,\ j = 1,\ldots,M,$$

$$\sum_{j=1}^{M} p_{ij} \leq 1,\ i = 1,\ldots,N.$$

Defining $\delta := 1/T$, and rewriting by slight abuse of notation $Z^*(\delta)$ instead of $Z^*(T)$ for δ positive, we also define $Z^*(0) = Z^*$, where Z^* is given by (5.32). Note that $Z^*(0)$ is finite, since we can find a feasible solution as follows. For each i, we choose a set of indices $J_i = \{j_1, \ldots, j_{n_i}\} \subset \{1, \ldots, M\}$ such that $(A_i, \tilde{C}_i)$ is observable, as in assumption 5.1.2. Once a set J_i has been chosen for each i, we form the matrix $\hat{P}$ with elements $\hat{p}_{ij} = 1\{j \in J_i\}$. Finally, we form a matrix P with elements p_{ij} satisfying the constraints and nonzero where the $\hat{p}_{ij}$ are nonzero. Such a matrix P exists as a consequence of Birkhoff theorem, see theorem 5.4.2. Now we consider the quadratic inequality (5.33) for some value of i. From the detectability assumption 5.1.2 and the choice of p_{ij}, we deduce that the pair $(A_i, \hat{C}_i)$, with

$$\hat{C}_i = \begin{bmatrix} \sqrt{p_{i1}} C_{i1}^T V_{i1}^{-1/2} & \cdots & \sqrt{p_{iM}} C_{iM}^T V_{iM}^{-1/2} \end{bmatrix}^T \tag{5.34}$$

is detectable. Also note that

$$\hat{C}_i^T \hat{C}_i = \sum_{j=1}^{M} p_{ij} C_{ij}^T V_{ij}^{-1} C_{ij}.$$

Together with the controllability assumption 5.1.3, we then know that (5.33) has a positive definite solution $\tilde{Q}_i$ [1, theorem 2.4.25]. Hence $Z^*(0)$ is finite.

We can also define $Z^*(\delta)$ for $\delta < 0$, by changing the right-hand side of (5.31) into $\delta Q_{i,0} = -|\delta| Q_{i,0}$. We have that $Z^*(\delta)$ is finite for $\delta < 0$ small enough. Indeed, passing the term $\delta Q_{i,0}$ on the left hand side, this can then be seen as a perturbation of the matrix $\tilde{C}_i$ above, and for δ small enough, detectability, which is an open condition, is preserved.

We will see below that (5.30), (5.32) can be recast as convex programs. It is then a standard result of perturbation analysis (see e.g. [23, p. 250]) that $Z^*(\delta)$is a convex function of δ, hence continuous on the interior of its domain, in particular continuous at $\delta = 0$. So

$$\limsup_{T\to\infty} Z^*(T) = \lim_{T\to\infty} Z^*(T) = Z^*.$$

Finally, for any policy π, we obtain the following lower bound on the achievable cost

$$\limsup_{T\to\infty} \frac{1}{T}\int_0^T \sum_{i=1}^N \left\{ \mathbf{Tr}(T_i \Sigma_i(t)) + \sum_{j=1}^M \kappa_{ij}\pi_{ij}(t) \right\} dt \geq \lim_{T\to\infty} Z^*(T) = Z^*.$$

We now show how to compute Z^* by solving a convex program involving linear matrix inequalities. For each i, introduce a new (slack) matrix variable R_i. Since $Q_i \succ 0, R_i \succ Q_i^{-1}$ is equivalent, by taking the Schur complement, to

$$\begin{bmatrix} R_i & I \\ I & Q_i \end{bmatrix} \succ 0,$$

and the Riccati inequality (5.33) can be rewritten

$$\begin{bmatrix} \tilde{Q}_i A_i + A_i^T \tilde{Q}_i - \sum_{j=1}^M p_{ij} C_{ij}^T V_{ij}^{-1} C_{ij} & \tilde{Q}_i W_i^{1/2} \\ W_i^{1/2} \tilde{Q}_i & -I \end{bmatrix} \preceq 0$$

we finally obtain, dropping the tildes from the notation $\tilde{Q}_i$, the semidefinite program

$$Z^* = \min_{R_i, Q_i, p_{ij}} \quad \sum_{i=1}^N \left\{ \mathbf{Tr}(T_i R_i) + \sum_{j=1}^M \kappa_{ij} p_{ij} \right\}, \tag{5.35}$$

$$\text{s.t.} \begin{bmatrix} R_i & I \\ I & Q_i \end{bmatrix} \succ 0,\ i = 1\ldots,N,$$

$$\begin{bmatrix} Q_i A_i + A_i^T Q_i - \sum_{j=1}^M p_{ij} C_{ij}^T V_{ij}^{-1} C_{ij} & Q_i W_i^{1/2} \\ W_i Q_i^{1/2} & -I \end{bmatrix} \preceq 0,\ i = 1\ldots,N,$$

$$0 \leq p_{ij} \leq 1,\ i = 1\ldots,N,\ j = 1,\ldots,M,$$

$$\sum_{i=1}^N p_{ij} = 1,\ j = 1,\ldots,M, \tag{5.36}$$

$$\sum_{j=1}^M p_{ij} \leq 1,\ i = 1,\ldots,N.$$

Hence solving the program (5.35) provides a lower bound on the achievable cost for the original optimal control problem.

5.4.2 Problem Decomposition

It is well-know that efficient methods exist to solve (5.35) in polynomial time, which implies a computation time polynomial in the number of variables of the original problem. Still, as the number of targets increases, the large LMI (5.35) becomes difficult to solve. Note however that it can be decomposed into N small coupled LMIs, following the standard dual decomposition approach already used for the restless bandit problem. This decomposition is very useful to solve large scale programs with a large number of systems. For completeness, we repeat the argument below.

We first note that (5.36) is the only constraint which links the N subproblems together. So we form the Lagrangian

$$L(R,Q,p;\lambda) = \sum_{i=1}^{N} \left\{ \mathbf{Tr}(T_i R_i) + \sum_{j=1}^{M} (\kappa_{ij} + \lambda_j) p_{ij} \right\} - \sum_{j=1}^{M} \lambda_j,$$

where $\lambda \in \mathbb{R}^M$ is a vector of Lagrange multipliers. We would take $\lambda \in \mathbb{R}^M_+$ if we had the constraint (5.1) instead of (5.2). Now the dual function is

$$G(\lambda) = \sum_{i=1}^{N} G_i(\lambda) - \sum_{j=1}^{M} \lambda_j, \tag{5.37}$$

$$\begin{aligned} \text{with} \quad G_i(\lambda) = \min_{R_i, Q_i, \{p_{ij}\}_{1\le j\le M}} \quad & \mathbf{Tr}(T_i R_i) + \sum_{j=1}^{M} (\kappa_{ij} + \lambda_j) p_{ij}, \\ \text{s.t.} \quad & \begin{bmatrix} R_i & I \\ I & Q_i \end{bmatrix} \succ 0, \\ & \begin{bmatrix} Q_i A_i + A_i^T Q_i - \sum_{j=1}^{M} p_{ij} C_{ij}^T V_{ij}^{-1} C_{ij} & Q_i W_i^{1/2} \\ W_i^{1/2} Q_i & -I \end{bmatrix} \preceq 0, \\ & 0 \le p_{ij} \le 1, \ j = 1, \ldots, M, \\ & \sum_{j=1}^{M} p_{ij} \le 1. \end{aligned} \tag{5.38}$$

The optimization algorithm proceeds then as follows. We choose an initial value λ^1 and set $k = 1$.

1. For $i = 1, \ldots, N$, compute $R_i^k, Q_i^k, \{p_{ij}^k\}_{1\le j\le M}$ optimal solution of (5.38), and the value $G_i(\lambda^k)$.

2. We have the value of the dual function at λ^k given by (5.37). A supergradient of $G(\lambda^k)$ at λ^k is given by
$$\left[\sum_{i=1}^{N} p_{i1}^k - 1, \ldots, \sum_{i=1}^{N} p_{iM}^k - 1 \right].$$

3. Compute λ^{k+1} in order to maximize $G(\lambda)$. For example, we can do this by using a supergradient algorithm, or any preferred nonsmooth optimization algorithm. Let k:=k+1 and go to step 1, or stop if convergence is satisfying.

Because the initial program (5.35) is convex, we know that the optimal value of the dual optimization problem is equal to the optimal value of the primal. Moreover, the optimal variables of the primal are obtained at step 1 of the algorithm above once convergence has been reached.

5.4.3 Open-loop Periodic Policies Achieving the Performance Bound

In this section we describe a sequence of open-loop policies that can approach arbitrarily closely the lower bound computed by (5.35). These policies are periodic switching strategies using a schedule obtained from the optimal parameters p_{ij}. Assuming no switching times or costs, their performance approaches the bound as the length of the switching cycle decreases toward 0.

Let $P = [p_{ij}]_{1 \leq i \leq N, 1 \leq j \leq M}$ be the matrix of optimal parameters obtained in the solution of (5.35). We assume here that constraints (5.1) and (5.3) were enforced, which is the most general case for the discussion in this section. Hence P verifies

$$0 \leq p_{ij} \leq 1,\ i = 1,\ldots,N,\ j = 1,\ldots,M,$$
$$\sum_{i=1}^{N} p_{ij} \leq 1,\ j = 1,\ldots,M, \quad \text{and} \quad \sum_{j=1}^{M} p_{ij} \leq 1,\ i = 1,\ldots,N.$$

A *doubly substochastic matrix* of dimension n is an $n \times n$ matrix $A = [a_{ij}]_{1 \leq i,j \leq n}$ which satisfies

$$0 \leq a_{ij} \leq 1,\ i,j = 1,\ldots,n,$$
$$\sum_{i=1}^{n} a_{ij} \leq 1,\ j = 1,\ldots,n, \quad \text{and} \quad \sum_{j=1}^{n} a_{ij} \leq 1,\ i = 1,\ldots,n.$$

If $M = N$, P is therefore a doubly stochastic matrix. Else if $M < N$, we can add $M - N$ columns of zeros to P to obtain a substochastic matrix. Finally if $N < M$, we can add $N - M$ rows of zeros to P to obtain a substochastic matrix. In any case, we call the resulting doubly substochastic matrix $\tilde{P} = [\tilde{p}_{ij}]$. If rows have been added, this is equivalent to the initial problem with additional "dummy systems". If columns are added, these correspond to using "dummy sensors". Dummy systems are not included in the objective function, and a dummy sensor $j > M$ is associated to the measurement noise covariance matrix $V_{ij}^{-1} = 0$ for all i, in effect producing no measurement. Without loss of generality, in the following we assume that $\tilde{P}$ is an doubly substochastic $N \times N$ matrix.

Doubly substochastic matrices have been intensively studied, and the material used in the following can be found in the book of Marshall and Olkin [90]. In particular, we have the following corollary of a classical theorem of Birkhoff [19], which says that a doubly stochastic matrix is a convex combination of permutation matrices.

Theorem 5.4.2 ([93])**.** *The set of $N \times N$ doubly substochastic matrices is the convex hull of the set $\mathscr{P}_0$ of $N \times N$ matrices which have a most one unit in each row and each column, and all other entries are zero.*

Hence for the doubly substochastic matrix $\tilde{P}$, there exists a set of positive numbers ϕ_k and matrices $P_k \in \mathscr{P}_0$ such that

$$\tilde{P} = \sum_{k=1}^{K} \phi_k P_k, \ \text{ with } \sum_{k=1}^{K} \phi_k = 1, \ \text{ for some integer } K. \tag{5.39}$$

An algorithm to obtain this decomposition is described in [90] and [30] for example, and runs in time $O(N^{4.5})$. Note that any matrix $A \in \mathscr{P}_0$ represents a valid sensor/system assignment (for the

system with additional dummy systems or sensors), where sensor j is measuring system i if and only if $a_{ij} = 1$.

The family of periodic switching policies considered is parametrized by a positive number ε representing a time-increment over which the switching schedule is executed completely. A policy is defined as follows for a fixed value of ε:

1. At time $t = l\varepsilon, l \in \mathbb{N}$, associate sensor j to system i as specified by the matrix P_1 of the representation (5.39). Run the corresponding continuous-time Kalman filters, keeping this sensor/system association for a duration $\phi_1\varepsilon$.

2. At time $t = (l+\phi_1)\varepsilon$, switch to the assignment specified by P_2. Run the corresponding continuous time Kalman filters until $t = (l+\phi_1+\phi_2)\varepsilon$.

3. Repeat the switching procedure, switching to matrix P_{i+1} at time $t = l+\phi_1+\cdots+\phi_i$, for $i = 1,\ldots,K-1$.

4. At time $t = (l+\phi_1+\cdots+\phi_K)\varepsilon = (l+1)\varepsilon$, start the switching sequence again at step 1 with P_1 and repeat the steps above.

Performance of the Periodic Switching Policies

Let us fix $\varepsilon > 0$ in the definition of the switching policy, and consider now the evolution of the covariance matrix $\Sigma_i^\varepsilon(t)$ for the estimate of the state of system i. The superscript indicates the dependence on the period ε of the policy. First we have

Lemma 5.4.3. *For all $i \in \{1,\ldots,N\}$, the estimate covariance $\Sigma_i^\varepsilon(t)$ converges as $t \to \infty$ to a periodic solution $\bar{\Sigma}_i^\varepsilon(t)$ of period ε.*

Proof. Fix $i \in \{1,\ldots,N\}$ (if we had $M > N$, then $\tilde{P}$ would be a $M \times M$ matrix and we would not consider here the additional dummy systems corresponding to $i > N$). Let $\sigma_i(t) \in \{0,1,\ldots,N\}$ be the function specifying which sensor is observing system i at time t under the switching policy. By convention $\sigma_i(t) = 0$ means that no sensor is scheduled to observe system i. Note that we never have $\sigma_i(t) \in \{M+1,\ldots,N\}$, since this means that a dummy sensor is scheduled to observe system i, but this would imply $p_{ij} > 0$ from (5.39), which is not true for $j \geq M+1$. Note also that $\sigma_i(t)$ is a piecewise constant, ε-periodic function. The switching times of $\sigma_i(t)$ are $t = (l+\phi_1+\cdots+\phi_{k-1})\varepsilon$, for $k = 1,\ldots,K$ and $l \in \mathbb{N}$.

Then the filter covariance matrix $\Sigma_i^\varepsilon(t)$ obeys the following periodic Riccati equation (PRE):

$$\begin{aligned} \dot{\Sigma}_i^\varepsilon(t) &= A_i\Sigma_i^\varepsilon(t) + \Sigma_i^\varepsilon(t)A_i^T + W_i - \Sigma_i^\varepsilon(t)(C_i^\varepsilon(t))^T C_i^\varepsilon(t)\Sigma_i^\varepsilon(t) \\ \Sigma_i^\varepsilon(t) &= \Sigma_{i,0}, \end{aligned} \tag{5.40}$$

where $C_i^\varepsilon(t) := V_{i\sigma_i(t)}^{-1/2} C_{i\sigma_i(t)}$ is a piecewise constant, ε-periodic matrix valued function. We now show that $(A_i, C_i^\varepsilon(\cdot))$ is detectable. Indeed, from the definition of detectability for linear periodic systems and it modal characterization [20], we can see that the pair $(A_i, C_i^\varepsilon(\cdot))$ is not detectable if

and only if there exist an eigenpair (λ, x) for A_i such that

$$\begin{aligned} A_i x &= \lambda x, \\ C_{ij_1} x &= 0, \\ C_{ij_2} e^{A_i \phi_1 \varepsilon} x &= e^{\lambda \phi_1 \varepsilon} C_{ij_2} x = 0, \\ &\vdots \\ C_{ij_K} e^{A_i(\phi_1 + \cdots + \phi_{K-1})\varepsilon} x &= e^{\lambda(\phi_1 + \cdots + \phi_{K-1})\varepsilon} C_{ij_K} x = 0. \end{aligned}$$

But this means that $(A, \tilde{C}_i)$ is not observable, with $\tilde{C}_i = [C_{ij_1}^T, \ldots, C_{ij_K}^T]^T$. It is easy to see then that this equivalent to saying that $(A, \hat{C}_i)$ is not observable, with $\hat{C}_i$ defined as in (5.34), but now with the optimal parameters p_{ij}. This would imply that the program (5.35) is not feasible [1, theorem 2.4.25], a contradiction. Finally, $(A_i, C_i^\varepsilon(\cdot))$ detectable together with our assumption 5.1.1 implies by the result of [95, corollary 2] that

$$\lim_{t \to \infty} \left(\Sigma^\varepsilon(t) - \bar{\Sigma}_i^\varepsilon(t) \right) = 0,$$

where $\bar{\Sigma}_i^\varepsilon(t)$ is the strong solution to the PRE, which is ε-periodic.

□

Next, denote by $\tilde{\Sigma}_i(t)$ the solution to the following Riccati differential equation (RDE):

$$\dot{\Sigma}_i = A_i \Sigma_i + \Sigma_i A_i^T + W_i - \Sigma_i \left(\sum_{j=1}^{M} p_{ij} C_{ij}^T V_{ij}^{-1} C_{ij} \right) \Sigma_i, \quad \Sigma_i(0) = \Sigma_{i,0}. \tag{5.41}$$

Assumptions 5.1.2 and 5.1.3 guarantee that $\tilde{\Sigma}_i(t)$ converges to a positive definite limit denoted Σ_i^*. Moreover, Σ_i^* is stabilizing and is the unique positive definite solution to the algebraic Riccati equation (ARE):

$$A_i \Sigma_i + \Sigma_i A_i^T + W_i - \Sigma_i \left(\sum_{j=1}^{M} p_{ij} C_{ij}^T V_{ij}^{-1} C_{ij} \right) \Sigma_i = 0. \tag{5.42}$$

The next lemma says that the periodic function $\bar{\Sigma}_i^\varepsilon(t)$ oscillates in a neighborhood of Σ_i^*.

Lemma 5.4.4. *For all $t \in \mathbb{R}_+$, we have $\bar{\Sigma}_i^\varepsilon(t) - \Sigma_i^* = O(\varepsilon)$ as $\varepsilon \to 0$.*

Proof. The function $t \to \bar{\Sigma}_i^\varepsilon(t)$ is the positive definite periodic solution of the PRE (5.40). It is ε-periodic. From Radon's lemma [1, p.90], which gives a representation of the solution to a Riccati differential equation as the ratio of solutions to a linear ODE, we also know that $\bar{\Sigma}_i^\varepsilon$ is C^∞ on each interval where $\sigma_i(t)$ is constant, where $\sigma_i(t)$ is the switching signal defined in the proof of lemma 5.4.3.

Let $\hat{\Sigma}_i^\varepsilon$ be the average of $t \to \bar{\Sigma}_i^\varepsilon(t)$:

$$\hat{\Sigma}_i^\varepsilon = \frac{1}{\varepsilon} \int_0^\varepsilon \bar{\Sigma}_i^\varepsilon(t) dt.$$

From the preceding remarks, it is easy to deduce that for all t, we have $\bar{\Sigma}_i^\varepsilon(t) - \hat{\Sigma}_i^\varepsilon = O(\varepsilon)$. Now, integrating the PRE (5.40) over the interval $[0, \varepsilon]$, we obtain

$$A_i \hat{\Sigma}_i^\varepsilon + \hat{\Sigma}_i^\varepsilon A^T + W_i - \frac{1}{\varepsilon} \int_0^\varepsilon \bar{\Sigma}_i^\varepsilon(t) (C_i^\varepsilon(t))^T C_i^\varepsilon(t) \bar{\Sigma}_i^\varepsilon(t) dt = \frac{1}{\varepsilon} (\bar{\Sigma}_i^\varepsilon(\varepsilon) - \bar{\Sigma}_i^\varepsilon(0)) = 0.$$

Expanding this equation in powers of ε, we get

$$A_i\hat{\Sigma}_i^\varepsilon + \hat{\Sigma}_i^\varepsilon A^T + W_i - \hat{\Sigma}_i^\varepsilon \left(\frac{1}{\varepsilon} \int_0^\varepsilon (C_i^\varepsilon(t))^T C_i^\varepsilon(t) dt \right) \hat{\Sigma}_i^\varepsilon + R(\varepsilon) = 0,$$

where $R(\varepsilon) = O(\varepsilon)$. Let $j_k := \sigma_i(t)$ for $t \in [(l+\phi_1+\ldots+\phi_{k-1})\varepsilon, (l+\phi_1+\ldots+\phi_k)\varepsilon]$. We can then rewrite

$$\frac{1}{\varepsilon} \int_0^\varepsilon (C_i^\varepsilon(t))^T C_i^\varepsilon(t) dt = \sum_{k=1}^K \phi_k C_{ij_k}^T V_{ij_k}^{-1} C_{ij_k}.$$

Let us denote by $p_{k,ij}$ the $(i,j)^{\text{th}}$ element of the matrix P_k in (5.39). We have $p_{k,ij} = 1_{\{j=j_k\}}$ with the above definition of j_k, including the case $j_k = 0$ (no measurement), which leads to $p_{k,ij} = 0$ for all $j \in \{1,\ldots,N\}$. Then we can write

$$\begin{aligned} \sum_{k=1}^K \phi_k C_{ij_k}^T V_{ij_k}^{-1} C_{ij_k} &= \sum_{k=1}^K \phi_k \left(\sum_{j=1}^N p_{k,ij} C_{ij}^T V_{ij}^{-1} C_{ij} \right) \\ &= \sum_{j=1}^N \left(\sum_{k=1}^K \phi_k p_{k,ij} \right) C_{ij}^T V_{ij}^{-1} C_{ij} \\ &= \sum_{j=1}^N \tilde{p}_{ij} C_{ij}^T V_{ij}^{-1} C_{ij} \\ &= \sum_{j=1}^M p_{ij} C_{ij}^T V_{ij}^{-1} C_{ij}, \end{aligned} \tag{5.43}$$

where the last equality uses our convention $V_{ij}^{-1} = 0$ for $j \geq M+1$, and the fact that $\tilde{p}_{ij} = p_{ij}$ for $j \leq M$. So finally, we obtain

$$A_i\hat{\Sigma}_i^\varepsilon + \hat{\Sigma}_i^\varepsilon A^T + (W_i + R(\varepsilon)) - \hat{\Sigma}_i^\varepsilon \left(\sum_{j=1}^M p_{ij} C_{ij}^T V_{ij}^{-1} C_{ij} \right) \hat{\Sigma}_i^\varepsilon = 0.$$

Note moreover that $\hat{\Sigma}_i^\varepsilon$ is the unique positive definite stabilizing solution of this ARE. Now comparing this ARE to the ARE (5.42), and since the stabilizing solution of an ARE is a real analytic function of the parameters [35], we deduce that $\hat{\Sigma}_i^\varepsilon - \Sigma_i^* = O(\varepsilon)$, and the lemma.

□

Theorem 5.4.5. *Let Z^ε denote the performance of the periodic switching policy with period ε. Then $Z^\varepsilon - Z^* = O(\varepsilon)$ as $\varepsilon \to 0$, where Z^* is the performance bound (5.35). Hence the switching policies approach the lower bound arbitrarily closely as the period tends to* 0.

Proof. We have

$$Z^\varepsilon = \limsup_{T\to\infty} \frac{1}{T} \int_0^T \sum_{i=1}^N \left(\mathbf{Tr}(T_i \Sigma_i^\varepsilon(t)) + \sum_{j=1}^M \kappa_{ij} \pi_{ij}^\varepsilon(t) \right) dt,$$

where π^ε is the sensor/system assignment of the switching policy. First by using a transformation similar to (5.43) and using the convention $\kappa_{ij} = 0$ for $j \in \{0\} \cup \{M+1,\ldots,N\}$ (no measurement or

measurement by a dummy sensor), we have for system i

$$\begin{aligned}\frac{1}{T}\int_0^T \sum_{j=1}^M \kappa_{ij}\pi_{ij}(t)dt &= \frac{1}{T}\sum_{n=0}^{\lfloor \frac{T}{\varepsilon}\rfloor -1}\int_{n\varepsilon}^{(n+1)\varepsilon} + \frac{1}{T}\int_{\lfloor \frac{T}{\varepsilon}\rfloor \varepsilon}^T \sum_{j=1}^M \kappa_{ij}\pi_{ij}(t)dt \\ &= \frac{1}{T}\frac{\lfloor T\rfloor}{\varepsilon}\sum_{k=1}^K (\phi_{j_k}\varepsilon)\kappa_{ij_k} + \frac{1}{T}\int_{\lfloor \frac{T}{\varepsilon}\rfloor \varepsilon}^T \sum_{j=1}^M \kappa_{ij}\pi_{ij}(t)dt \\ &= \frac{\lfloor T\rfloor}{T}\sum_{j=1}^M \kappa_{ij}p_{ij} + \frac{1}{T}\int_{\lfloor \frac{T}{\varepsilon}\rfloor \varepsilon}^T \sum_{j=1}^M \kappa_{ij}\pi_{ij}(t)dt. \end{aligned} \tag{5.44}$$

Hence

$$\limsup_{T\to\infty}\frac{1}{T}\int_0^T \sum_{i=1}^N\sum_{j=1}^M \kappa_{ij}\pi_{ij}^\varepsilon(t)dt \le \sum_{i=1}^N\sum_{j=1}^M \kappa_{ij}p_{ij}.$$

Next, from lemma 5.4.3 and 5.4.4, it follows readily that $\limsup_{t\to\infty}\Sigma_i^\varepsilon(t) - \Sigma_i^* = O(\varepsilon)$. It is well known [127] that Σ_i^* is the minimal (for the partial order on $\mathbb{S}_+^n$) positive definite solution of the quadratic matrix inequality

$$A_i\Sigma_i + \Sigma_i A_i^T + W_i - \Sigma_i\left(\sum_{j=1}^M p_{ij}C_{ij}^T V_{ij}^{-1}C_{ij}\right)\Sigma_i \preceq 0.$$

Hence for the p_{ij} obtained from the computation of the lower bound, these matrices Σ_i^* minimize

$$\begin{aligned} &\sum_{i=1}^N \mathbf{Tr}\,(T_i\Sigma_i) \\ \text{s.t.}\quad & A_i\Sigma_i + \Sigma_i A_i^T + W_i - \Sigma_i\left(\sum_{j=1}^M p_{ij}C_{ij}^T V_{ij}^{-1}C_{ij}\right)\Sigma_i \preceq 0, \quad i=1,\dots,N. \end{aligned} \tag{5.45}$$

Changing variable to $Q_i = \Sigma_i^{-1}$, and multiplying the inequalities (5.45) on the left and right by Q_i yields

$$Q_iA_i + A_i^TQ_i + Q_iW_iQ_i - \sum_{j=1}^M p_{ij}C_{ij}^TV_{ij}^{-1}C_{ij} \preceq 0,$$

and hence we recover (5.33). In conclusion, the covariance matrices resulting from the switching policies approach within $O(\varepsilon)$ as $\varepsilon \to 0$ the covariance matrices which are obtained from the lower bound on the achievable cost. The theorem follows, by bounding the supremum limit of a sum by the sum of the supremum limits. □

Remark 5.4.6. Since the bound computed in (5.35) is tight, and since it is easy to see that the performance bound of section 5.3 is at least as good as the bound (5.35) for the simplified problem of that section, we can conclude that the two bounds coincide and that section 5.3 gives an alternative way of computing the solution of (5.35) in the case of identical sensors and one-dimensional systems.

Transient Behavior of the Switching Policies

Before we conclude, we take a look at the transient behavior of the switching policies. We show that over a finite time interval, $\Sigma_i^\varepsilon(t)$ remains close to $\tilde{\Sigma}_i(t)$, solution of the "averaged" RDE (5.41).

Together with the previous results on the asymptotic behavior, we see then that $\Sigma_i^\varepsilon(t)$ and $\tilde{\Sigma}_i(t)$ remain close for all t.

Lemma 5.4.7. *For all $0 \leq T_0 < \infty$, there exist constants $\varepsilon_0 > 0$ and $M_0 > 0$ such that for all $0 < \varepsilon \leq \varepsilon_0$ and for all $t \in \left[0, \left\lceil \frac{T_0}{\varepsilon} \right\rceil \varepsilon\right]$, we have $\|\Sigma_i^\varepsilon(t) - \tilde{\Sigma}_i(t)\|_\infty \leq M_0\varepsilon$.*

Proof. By Radon's lemma we know that Σ_i^ε is C^∞ on each interval where $\sigma_i(t)$ is constant. We have then, over the interval $t \in [l\varepsilon, (l+\phi_1)\varepsilon]$, for $l \in \mathbb{N}$:

$$\Sigma_i^\varepsilon((l+\phi_1)\varepsilon) = \Sigma_i^\varepsilon(l\varepsilon) + \phi_1\varepsilon[A_i\Sigma_i^\varepsilon(l\varepsilon) + \Sigma_i^\varepsilon(l\varepsilon)A_i^T + W_i - \Sigma_i^\varepsilon(l\varepsilon)C_{ij_1}^T V_{ij_1}^{-1} C_{ij_1}\Sigma_i^\varepsilon(l\varepsilon)] + O(\varepsilon^2), \tag{5.46}$$

where $j_1 := \sigma_i(t)$ is the sensor associated to system i during the first period as specified by P_1. Similarly in the following, we let $j_k := \sigma_i(t)$ for $t \in [(l+\phi_1+\ldots+\phi_{k-1})\varepsilon, (l+\phi_1+\ldots+\phi_k)\varepsilon]$. Now over the period $t \in [(l+\phi_1)\varepsilon, (l+\phi_1+\phi_2)\varepsilon]$, we have:

$$\begin{aligned}\Sigma_i^\varepsilon((l+\phi_1+\phi_2)\varepsilon) = \Sigma_i^\varepsilon((l+\phi_1)\varepsilon) + \phi_2\varepsilon[A_i\Sigma_i^\varepsilon((l+\phi_1)\varepsilon) + \Sigma_i^\varepsilon((l+\phi_1)\varepsilon)A_i^T + W_i \\ - \Sigma_i^\varepsilon((l+\phi_1)\varepsilon)C_{ij_2}^T V_{ij_2}^{-1} C_{ij_2}\Sigma_i^\varepsilon((l+\phi_1)\varepsilon)] + O(\varepsilon^2).\end{aligned}$$

Using (5.46), we deduce that

$$\begin{aligned}\Sigma_i^\varepsilon((l+\phi_1+\phi_2)\varepsilon) =& \Sigma_i^\varepsilon(l\varepsilon) + \varepsilon[\phi_1+\phi_2]\{A_i\Sigma_i^\varepsilon(l\varepsilon) + \Sigma_i^\varepsilon(l\varepsilon)A_i^T + W_i\} \\ & - \varepsilon\Sigma_i^\varepsilon(l\varepsilon)(\phi_1 C_{ij_1}^T V_{ij_1}^{-1} C_{ij_1} + \phi_2 C_{ij_2}^T V_{ij_2}^{-1} C_{ij_2})\Sigma_i^\varepsilon(l\varepsilon) + O(\varepsilon^2).\end{aligned}$$

By immediate recursion, and since $\phi_1 + \cdots + \phi_K = 1$, we then have

$$\Sigma_i^\varepsilon((l+1)\varepsilon) = \Sigma_i^\varepsilon(l\varepsilon) + \varepsilon\{A_i\Sigma_i^\varepsilon(l\varepsilon) + \Sigma_i^\varepsilon(l\varepsilon)A_i^T + W_i\} - \Sigma_i^\varepsilon(l\varepsilon)\left(\sum_{k=1}^K \phi_k C_{ij_k}^T V_{ij_k}^{-1} C_{ij_k}\right)\Sigma_i^\varepsilon(l\varepsilon) + O(\varepsilon^2).$$

Hence by the same transformation as in lemma 5.4.4, Σ_i^ε verifies the relation

$$\Sigma_i^\varepsilon((l+1)\varepsilon) = \Sigma_i^\varepsilon(l\varepsilon) + \varepsilon\{A_i\Sigma_i^\varepsilon(l\varepsilon) + \Sigma_i^\varepsilon(l\varepsilon)A_i^T + W_i\} - \varepsilon\Sigma_i^\varepsilon(l\varepsilon)\left(\sum_{j=1}^M p_{ij} C_{ij}^T V_{ij}^{-1} C_{ij}\right)\Sigma_i^\varepsilon(l\varepsilon) + O(\varepsilon^2). \tag{5.47}$$

But notice now that the approximation (5.47) is also true by definition for $\tilde{\Sigma}_i(t)$ over the interval $t \in [l\varepsilon, (l+1)\varepsilon]$. Next, note the identity for Q, X and $\tilde{X}$ symmetric:

$$\begin{aligned}& A\tilde{X} + \tilde{X}A^T - \tilde{X}Q\tilde{X} - (AX + XA^T - XQX) \\ &= (A - \tilde{X}Q)(\tilde{X} - X) + (\tilde{X} - X)(A - \tilde{X}Q)^T + (\tilde{X} - X)Q(\tilde{X} - X).\end{aligned}$$

Letting $Q = \sum_{j=1}^M p_{ij} C_{ij}^T V_{ij}^{-1} C_{ij}$, $\Delta_i^\varepsilon(l) = \tilde{\Sigma}_i(l\varepsilon) - \Sigma_i^\varepsilon(l\varepsilon)$, we obtain from this identity

$$\Delta_i^\varepsilon(l+1) = \Delta_i^\varepsilon(l) + \varepsilon\{(A - \tilde{\Sigma}_i(l\varepsilon)Q)\Delta_i^\varepsilon(l) + \Delta_i^\varepsilon(l)(A - \tilde{\Sigma}_i(l\varepsilon)Q)^T + \Delta_i^\varepsilon(l)Q\Delta_i^\varepsilon(l)\} + O(\varepsilon^2).$$

Note that $\Delta_i^\varepsilon(0) = 0$ and $\tilde{\Sigma}_i(t)$ is bounded, so by immediate recursion we have

$$\Delta_i^\varepsilon(l) = \sum_{k=1}^l R_k(\varepsilon), \quad \text{where } R_k(\varepsilon) = O(\varepsilon^2) \text{ for all } k.$$

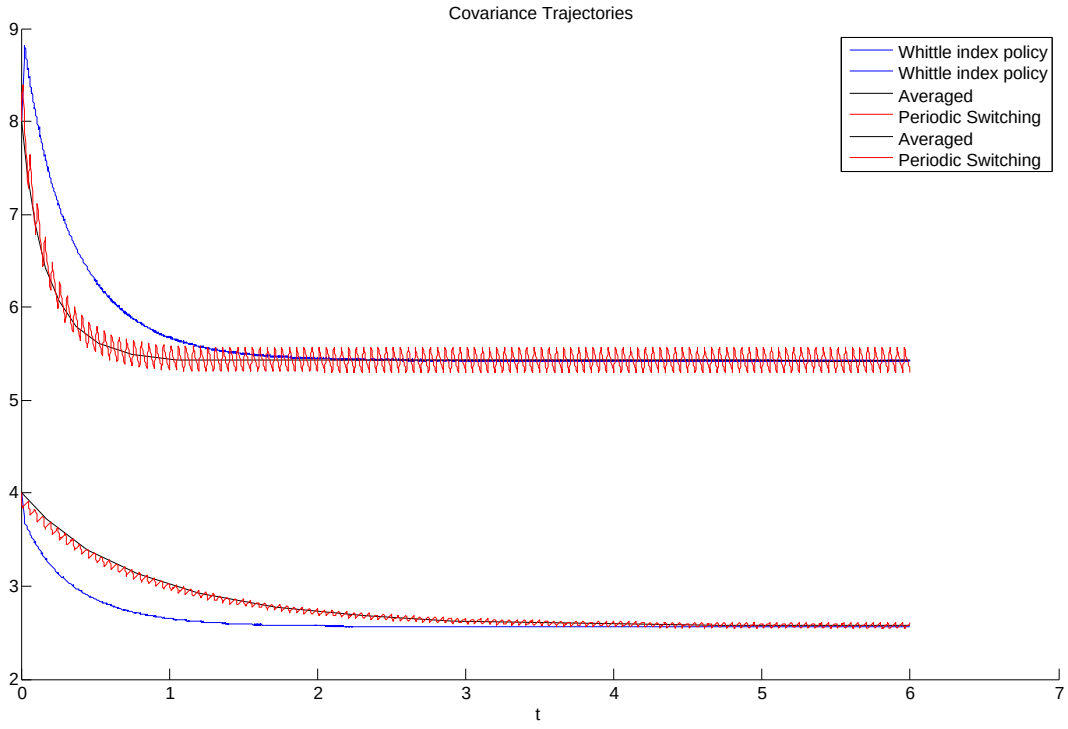


Figure 5-4: Comparison of the covariance trajectories under Whittle's index policy and the periodic switching policy, for the example of figure 5-2. Here the period ε was chosen to be 0.05 (the A coefficients of the systems are 0.1 and 2 for the lower curve and the upper curve respectively). In black we show the solution $\tilde{\Sigma}_i(t)$ of the RDE (5.41).

Fix $T_0 \geq 0$. We have then

$$\tilde{\Sigma}_i\left(\left\lceil \frac{T_0}{\varepsilon} \right\rceil \varepsilon\right) - \Sigma_i^\varepsilon\left(\left\lceil \frac{T_0}{\varepsilon} \right\rceil \varepsilon\right) = \Delta_i^\varepsilon\left(\left\lceil \frac{T_0}{\varepsilon} \right\rceil\right) = O(\varepsilon).$$

This means that there exist constants $\varepsilon_1, M_1 > 0$ such that

$$\left\| \tilde{\Sigma}_i\left(\left\lceil \frac{T_0}{\varepsilon} \right\rceil \varepsilon\right) - \Sigma_i^\varepsilon\left(\left\lceil \frac{T_0}{\varepsilon} \right\rceil \varepsilon\right) \right\|_\infty \leq M_1 \varepsilon, \quad \text{for all } 0 < \varepsilon < \varepsilon_1.$$

It is clear that this approximation is in fact valid for all t up to time $\left\lceil \frac{T_0}{\varepsilon} \right\rceil \varepsilon$. □

5.5 Conclusion

We have considered in this chapter an attention-control problem in continuous time, which consists in scheduling sensor/target assignments and running the corresponding Kalman filters. In the

context of an UVS surveillance mission, the problem is motivated as in the previous chapter by discretizing the observation space into distinct sites and assuming now that the state of each individual site evolves as an LTI system. Admittedly, and especially for our formulation in continuous time, the absence of switching times or costs when a sensor moves between different sites is potentially problematic for the vehicle routing problem. We will start investigating the addition of switching costs in the next chapter. Still, we hope that this problem, by providing cleaner results than what can be reached if we superimpose a traveling salesman problem, can help build good heuristics for observation missions. Additionally, there are numerous other sensor scheduling applications where the switching costs are not as important. Athans mentions a few: telemetry-data aerospace systems, radar waveform selection for multi-target tracking, data collection and polling. We could also think of applying these results to design an attention-control system for UAV operators. Typically for these systems, the attention switching times are much smaller than the time constants of the dynamics of the processes.

In continuous time, we can prove that the bound obtained from a RBP type relaxation is tight, assuming we allow the sensors to switch arbitrarily fast between the targets. An open question is to characterize the performance of the restless bandit index policy derived in the scalar case. It was found experimentally that the performance of this policy seems to match the bound, but we do not have a proof of this fact. An advantage of the RB index policy over the switching policies is that it is in feedback form. Obtaining optimal feedback policies for the multidimensional case would also be of interest.

An interesting practical aspect of the index based solution is that, as long as the assumption of independent site dynamics is respected, it is not required that the models of the different sites be similar. For example, we could have one site with a discrete state space as in chapter 4 (in a continuous-time version) and another modeled as a linear system as in this chapter. The index solution is still based on comparing the Whittle indices computed for each site, and the performance bound is still computed from the sum of the value functions for relaxed problems at each site. The details regarding problems with heterogeneous dynamics are left for future work.

Chapter 6

Adding Switching Costs

In the approach to mobile sensor scheduling presented in chapters 4 and 5, the path-planning component was not considered. This allows for much simpler computations and the results can provide insight into the final scheduling problem. However, in general in the context of vehicle routing problems the policies obtained require frequent location changes that are costly and even unrealistic in practice. Hence these policies have to be heuristically adapted to the situation at hand. In this chapter we try to incorporate directly some form of switching penalty into the optimization problem. Ideally, we would like to handle scheduling problems featuring a path planning component similar to the Dubins TSP considered in chapter 2. Because of the complexity of the resulting model however, we only consider simplified versions of it in this thesis.

We present here a linear programming relaxation for the discrete-time restless bandit problem with additional costs penalizing switching between the bandits. These switching costs can model travel distances for example. The relaxation is derived in a systematic way, starting from the exact linear programming formulation of the problem, and optimizing over a restricted set of marginals of the occupation measures (see section 3.2.3 for background material on this approach to MDPs). An important part of the derivation is the search for valid constraints on these marginals, improving the quality of the relaxation. This relaxation provides an upper bound on the reward achievable by any policy. A lower bound is obtained by developing a one-step lookahead policy, using the information extracted from the dual of the linear programming relaxation to construct an approximation of the reward-to-go. Moreover, we show how this policy can equivalently be obtained using a technique introduced by Bertsimas and Niño-Mora in their work on the standard restless bandit problem. Experimental results as well as a recently developed bound for approximate dynamic programming provide some empirical support for our heuristic. Throughout we delineate the type of assumptions needed to develop the relaxation, in order to facilitate the adaptation of the methods to different models.

An alternative way, potentially more flexible, of approaching the joint path-planning and scheduling problems arising in real-world missions performed by UAS, and based on a deterministic fluid approximation of the problems, will be described in chapter 7.

6.1 Introduction

In this chapter we study an extension of the MABP and RBP where we add costs for changing the currently active project. We call the resulting problems the multi-armed bandit problem with switching costs (MABPSC) and restless bandit problem with switching costs (RBPSC) respectively. This extension is of great interest to various applications besides mobile sensor routing, as discussed

by [61, 62, 130, 72], in order to model for example set-up and tear-down costs in queuing networks, transition costs in a job search problem or transaction fees in a portfolio optimization problem.

As mentioned by Glazebrook et al. [50], certain MABPSC can be transformed to a restless bandit problem, if we assume that the switching costs have a separable form, that is, the cost for switching from project i to j can be written $c_{ij} = c_i + c_j$. This restriction on the switching costs is not applicable in our typical situations where these costs represent travel distances. Then, an additional level of coupling between the projects is introduced, and it is not clear how to apply the methods developed for restless bandits. A partial characterization of the switching times for the MABPSC with constant switching cost can be found in [7], and some related papers in the statistics literature include [12, 55, 63]. Dusonchet and Hongler [40] solved a two-armed bandit problem with switching costs analytically, in the case of deteriorating rewards. Note also that in the regret minimization formulation of the bandit problem as in Lai and Robbins [76], Agrawal et al. [5] showed that the addition of switching costs does not change the achievable asymptotic performance. We do not consider this formulation of bandit problems in this thesis.

For the RBP, Bertsimas and Niño-Mora [17] used the linear programming approach to MDPs to give refined versions of Whittle's relaxation, and gave empirical results showing that their bounds on the achievable performance are indeed tighter. They also proposed a new heuristic for the RBP. The drawbacks of this heuristic compared to Whittle's index heuristic are that there is less empirical evidence for its quality, it does not reduce to Gittins' optimal policy for the MAPB, and its computation is not decomposable by project as Whittle's. On the other hand, it is more easily applicable than Whittle's heuristic since it does not require indexability of the projects, and it has a natural interpretation from the linear programming point of view. We also provide an equivalent interpretation for this heuristic in terms of approximate dynamic programming.

This chapter essentially extends the technique of Bertsimas and Niño-Mora to the RBPSC. We use the state-action frequency approach to MDPs, as in section 3.2.3. In our context, we found this approach more natural than starting in the dynamic programming domain. Indeed, in the formulation of the relaxation we are naturally led to consider certain resource allocation constraints, taking into account the limited number of bandits that we can activate and their positions. Additional constraints for MDPs are most easily formulated in the state-action frequency domain, and this is done even in works that adopt a dynamic programming approach from the start, such as [136, 27]. These authors then need to use Lagrange multipliers to transfer these constraints back in the dual domain of dynamic programming. It appears clearer to obtain a complete formulation first on the primal side of state-action frequencies, possibly considering the dual later on in order to use the available dynamic programming techniques.

The rest of the chapter is organized as follows. In section 6.2 we formulate the RBPSC using the LP approach of section 3.2.3. In Section 6.3, we show that the special case of the MABPSC is NP-hard. Then, we propose a first-order linear programming relaxation of the general RBPSC problem, which provides an efficiently computable bound on the achievable performance. Section 6.4 describes how the relaxation can be used to motivate a heuristic solving the problem in practice. This heuristic is based on approximate dynamic programming techniques, but we also explain how to recover it from the linear programming theory used by Bertsimas and Niño-Mora to design their primal-dual heuristic for the restless bandit problem. Section 6.5 presents numerical experiments comparing the heuristic to the performance bound. In section 6.6 we provide a result linking the performance of the heuristic to the quality of the relaxation. Together, these last two sections provide some empirical support for our heuristic.

6.2 Exact Formulation of the RBPSC

6.2.1 Exact Formulation of the RBSC Problem

In the RBPSC, N projects are distributed in space at N sites, and $M \leq N$ servers can be allocated to M different projects at each time period $t = 1, 2, \ldots$ In the following, we use the terms project and site interchangeably; likewise, agent and server have the same meaning. At each time period, each server must occupy one site, and different servers must occupy distinct sites. We say that a site is active at time t if it is visited by a server, and is passive otherwise. If a server travels from site k to site l, we incur a cost c_{kl}. Each site can be in one of a finite number of states $x_n \in S_n$, for $n = 1, \ldots, N$, and we denote the Cartesian product of the individual state spaces $\mathscr{S} = S_1 \times \ldots \times S_N$. If site n in state x_n is visited, a reward $r_n^1(x_n)$ is earned, and its state changes to y_n according to the transition probability $p^1_{x_n y_n}$. If the site is not visited, then a reward (potentially negative) $r_n^0(x_n)$ is earned for that site and its state changes according to the transition probabilities $p^0_{x_n y_n}$. *We assume that all sites change their states independently of each other.*

Note that if the switching costs are all 0, we recover the initial formulation of the RBP. If in addition the passive rewards are 0 and the passive transition matrix is the identity matrix, we obtain the MABP. If we just add the switching costs to the basic MABP, we call the resulting model MABPSC.

The RBPSC problem can be cast into the MDP framework as follows. Let us denote the set $\{1, \ldots, N\}$ by $[N]$. We consider that when no agent is present at a given site, there is a fictitious agent called passive agent at that site. We also call the real agents active agents, since they collect active rewards. The transition of a passive agent between sites does not involve any switching cost, and when a passive agent is present at a site, the passive reward is earned. Therefore, we have a total of N agents including both the real and passive agents, and we can describe the positions of all agents by a vector $\mathbf{s} = (s_1, \ldots, s_N)$, which corresponds to a *permutation* of [N] (due to our constraint that different agents must occupy different sites). We denote the set of these permutation vectors by $\Pi_{[N]}$. We let the M first components correspond to the active (i.e., real) agents. For example, with $M = 2$ and $N = 4$, the vector $(s_1 = 2, s_2 = 3, s_3 = 1, s_4 = 4) \in \Pi_{[4]}$ means that agent 1 is in site 2, agent 2 in site 3 and sites 1 and 4 are passive.

For an agent $i \in [N]$, we refer to the set of the other agents by $-i$. If we fix $s_i \in [N]$ for some $1 \leq i \leq N$, then we write $\mathbf{s}_{-i}$ to denote the vector $(s_1, \ldots, s_{i-1}, s_{i+1}, \ldots, s_N)$, and $\Pi_{[N]-s_i}$ to denote the permutations of the set $[N] - \{s_i\}$. In particular, we write $\sum_{\mathbf{s}_{-i} \in \Pi_{[N]-s_i}}$ to denote the sum over all permutations of the positions of the agents $-i$, over the set of sites not occupied by agent i. We also write $\mathscr{S}_{-i}$ to denote the cartesian product $S_1 \times \ldots S_{i-1} \times S_{i+1} \times \ldots \times S_N$.

The state of the system at time t can be described by the state of each site and the positions $\mathbf{s} \in \Pi_{[N]}$ of the servers, including the passive ones. With this state description, we are able to handle any number $M \leq N$ of agents as a parameter within the same framework. We denote the complete state by $(x_1, \ldots, x_N; s_1, \ldots, s_N) := (\mathbf{x}; \mathbf{s})$. We can choose which sites are to be visited next, i.e., the action $\mathbf{a}$ belongs to the set $\Pi_{[N]}$ and corresponds to the assignment of the agents, including the passive ones, to the sites for the next time period. Once the sites to be visited are chosen, there are costs $c_{s_i a_i}$ for moving the active agent i from site s_i to site a_i, including possibly a nonzero cost for staying at the same site. The reward earned is $\sum_{i=1}^{M} r^1_{a_i}(x_{a_i}) + \sum_{i=M+1}^{N} r^0_{a_j}(x_{a_j})$. We are given a distribution ν on the initial state of the system, and *we will assume a product form*

$$\nu(x_1, \ldots, x_N; s_1, \ldots, s_N) = \nu_1(x_1) \ldots \nu_N(x_N) \delta_{d_1}(s_1) \ldots \delta_{d_N}(s_N), \tag{6.1}$$

i.e., the initial states of the sites are independent random variables and server i leaves initially from

site d_i, with $\mathbf{d} \in \Pi_{[N]}$.

The transition matrix has a particular structure, since the sites evolve independently and the transitions of the agents are deterministic:

$$\mathscr{P}_{(\mathbf{x}';\mathbf{s}')\mathbf{a}(\mathbf{x};\mathbf{s})} = \prod_{i=1}^{M} p^1_{x'_{a_i} x_{a_i}} \prod_{i=M+1}^{N} p^0_{x'_{a_i} x_{a_i}} \prod_{i=1}^{N} \delta_{s_i}(a_i). \tag{6.2}$$

With these elements, we can specialize (3.23) to obtain the following proposition.

Proposition 6.2.1. *The optimal occupation measure for the RBSC problem can be obtained by multiplying by* $(1-\alpha)$ *the optimal solution* $[\rho_{(\mathbf{x};\mathbf{s}),\mathbf{a}}]_{\mathbf{x}\in\mathscr{S},(\mathbf{s},\mathbf{a})\in\Pi^2_{[N]}}$ *of the following linear program:*

maximize

$$\sum_{\mathbf{s}\in\Pi_{[N]}} \sum_{\mathbf{a}\in\Pi_{[N]}} \sum_{\mathbf{x}\in\mathscr{S}} \left(\sum_{i=1}^{M} (r^1_{a_i}(x_{a_i}) - c_{s_i a_i}) + \sum_{i=M+1}^{N} r^0_{a_i}(x_{a_i}) \right) \rho_{(\mathbf{x};\mathbf{s}),\mathbf{a}} \tag{6.3}$$

subject to

$$\sum_{\mathbf{a}\in\Pi_{[N]}} \rho_{(\mathbf{x};\mathbf{s}),\mathbf{a}} - \alpha \sum_{\mathbf{s}'\in\Pi_{[N]}} \sum_{\mathbf{x}'\in\mathscr{S}} \rho_{(\mathbf{x}';\mathbf{s}'),\mathbf{s}} \prod_{i=1}^{M} p^1_{x'_{s_i} x_{s_i}} \prod_{i=M+1}^{N} p^0_{x'_{s_i} x_{s_i}} = \prod_{i=1}^{N} \nu_i(x_i) \delta_{d_i}(s_i), \tag{6.4}$$

$\forall\, (\mathbf{x},\mathbf{s}) \in \mathscr{S} \times \Pi_{[N]}$

$\rho_{(\mathbf{x};\mathbf{s}),\mathbf{a}} \geq 0, \quad \forall\, ((\mathbf{x};\mathbf{s}),\mathbf{a}) \in \mathscr{S} \times \Pi^2_{[N]}.$

Remark 6.2.2. The formulation above is of little computational interest since the number of variables and constraints is of the order of $|\mathscr{S}| \times (N!)^2$, that is, exponential in the size of the input. For example, if we consider a problem with $N = 10$ sites and 5 states for each site, we obtain a linear program with more than 6×10^{13} variables, and therefore real world instances of the problem cannot be solved by feeding this formulation directly into an LP solver. Note that for M fixed, we obtained a slightly more compact formulation in [81] (in that paper, for $M = 1$). However, that exact formulation is still intractable, and does not lead to a significantly more compact relaxation. The advantage of the formulation presented here is that M can be given as a parameter without modification of the algorithms.

6.2.2 Dual Linear Program

We can obtain the linear program dual to (6.3) by constructing it directly, or starting from Bellman's equation and using standard dynamic programming arguments [14, vol. 2, p. 53]. The decision variables correspond to the reward-to-go vector $\{\lambda_{\mathbf{x},\mathbf{s}}\}_{\mathbf{x},\mathbf{s}}$. We get

$$\text{minimize } \sum_{\mathbf{x},\mathbf{s}} \lambda_{\mathbf{x},\mathbf{s}} \prod_{i=1}^{N} \nu_i(x_i) \delta_{d_i}(s_i), \tag{6.5}$$

subject to

$$\lambda_{\mathbf{x},\mathbf{s}} - \alpha \sum_{\tilde{\mathbf{x}}\in\mathscr{S}} \left(\prod_{i=1}^{M} p^1_{x_{a_i}\tilde{x}_{a_i}} \prod_{i=M+1}^{N} p^0_{x_{a_i}\tilde{x}_{a_i}} \right) \lambda_{\tilde{\mathbf{x}},\mathbf{a}} \geq \sum_{i=1}^{M} (r^1_{a_i}(x_{a_i}) - c_{s_i a_i}) + \sum_{i=M+1}^{N} r^0_{a_i}(x_{a_i}),$$

$\forall \mathbf{x} \in \mathscr{S}, \forall (\mathbf{s},\mathbf{a}) \in \Pi^2_{[N]}.$

6.3 Linear Programming Relaxation of the RBPSC

6.3.1 Complexity of the RBPSC and MABPSC.

It could well be that the RBPSC appears difficult to solve only because of our own inability to formulate it in an efficient manner. In the most general formulation above, we know however that the problem is likely to be intractable, since [101] showed that the RBP is already PSPACE-hard, even for the case of deterministic transition rules and one server. Here we show that the other special case MABPSC is also difficult. In this Section, we also denote (with a slight abuse of notation) by MABPSC the decision version of the optimization problem considered before; that is, given an instance of the MABPSC and a real number L, is there a policy that achieves a total expected reward greater than or equal to L? The fact that this problem is NP-hard is deduced from the same result for the HAMILTON CIRCUIT problem.

Theorem 6.3.1. *MABPSC is NP-hard.*

Proof. Recall that in the HAMILTON CIRCUIT problem, we are given a graph $G = (V,E)$, and we want to know if there is a circuit in G visiting all the nodes exactly once. HAMILTON CIRCUIT was one of the first combinatorial problems proven to be NP-complete [68]. HAMILTON CIRCUIT is a special case of MABPSC. Indeed, given any graph $G = (V,E)$, we construct an instance of MABPSC with $N = |V|$ sites, travel costs $c_{ij} = 1$ if $\{i, j\} \in E$, and $c_{ij} = 2$ otherwise. We choose arbitrarily one of the sites to be the site where the server is initially present. This site has only one state, and the reward for having the server present at this site at any period is 0. To each of the $(N-1)$ other sites, we associate a Markov chain with two states. To the initial state is associated a reward of 2 (discounted by α at each time period), and after a visit to a site in this state, the site moves with probability 1 to the second state, associated with a reward of (-1) (discounted by α at each time period). Once in this state, the chain remains there with probability 1. We obtain a MABPSC since a site can change state only when it is visited (and in the first state).

Now since $\alpha > 0$, it is clear that a policy can achieve a reward of $(1+\alpha+\ldots+\alpha^{N-2}-\alpha^{N-1})$ if and only if there exists a Hamilton circuit in G. The only possible policy actually just moves the server along the Hamilton circuit without stopping, except when the server is back at the initial site. □

Remark 6.3.2. Note that even the two armed bandit problem with switching costs, considered for example by [7], seems difficult in general, although to the best of our knowledge, no formal lower bound exists on the complexity of this more restricted case.

6.3.2 A Relaxation for the RBPSC

The discussion in the previous paragraph serves as a justification for the introduction of a relaxed formulation of the RBPSC. In this section we will see that in the state-action frequency domain, a relaxation can be easily obtained by optimizing over specific marginals of the occupation measure. Once the relaxation is obtained, we can go back to the value function domain of dynamic programming by taking the dual of the relaxed linear program. In general, the method that we follow to derive a relaxation is the following:

1. identify marginals of the state-action frequencies that are sufficient to express the objective function in (6.3).

2. express the constraints on these marginals by partially summing over the constraints in (6.4).

3. add additional constraints due to the fact that these marginals all derive from the same state-action frequency vector.

Although we consider here in details the RBPSC, this approach is general enough to handle problems with a similar structure. The essential features of the problem that allow the method to work are *the separable structure of the objective function and the independence assumption for the evolution of the sites*. This approach is implicit in the work of [17] on restless bandits. It is relatively systematic once the exact linear program has been formulated. When the coupling between the sites increases (for instance, when we add switching costs to the RBP), the relaxation becomes more complicated and grows in size.

We start by rewriting the objective function and we identify the relevant marginals:

$$\sum_{s=1}^{N}\sum_{a=1}^{N}\sum_{x_a\in S_a}\left[\left(r_a^1(x_a)-c_{sa}\right)\left(\sum_{i=1}^{M}\rho^i_{x_a;s,a}\right)+r_a^0(x_a)\left(\sum_{i=M+1}^{N}\rho^i_{x_a;s,a}\right)\right]$$

where the marginals appearing above are obtained as follows:

$$\rho^i_{(x_a;s),a}=\sum_{\mathbf{x}_{-a}\in\mathscr{S}_{-a}}\ \sum_{\mathbf{s}_{-i}\in\Pi_{[N]-s}}\ \sum_{\mathbf{a}_{-i}\in\Pi_{[N]-a}}\rho_{(\mathbf{x};\mathbf{s}),\mathbf{a}}\ ,\ \forall\ (i,s,a)\in[N]^3,\ x_a\in S_a. \tag{6.6}$$

and the superscripts refer to the agents.

Now to express the constraints, it turns out that we will also need the following variables:

$$\rho^i_{(x_s;s),a}=\sum_{\mathbf{x}_{-s}\in\mathscr{S}_{-s}}\ \sum_{\mathbf{s}_{-i}\in\Pi_{[N]-s}}\ \sum_{\mathbf{a}_{-i}\in\Pi_{[N]-a}}\rho_{(\mathbf{x};\mathbf{s}),\mathbf{a}}\ ,\ \forall\ (i,s,a)\in[N]^3,\ x_s\in S_s. \tag{6.7}$$

The variables in (6.6) can be interpreted as the frequency with which agent i switches from site s to site a and the destination site is in state x_a. The variables in (6.7), up to a factor $(1-\alpha)$, can be interpreted as the frequency with which agent i switches from site s to site a and the origin site is in state x_s. Note that this notation is somewhat redundant, since we can write the variables $\rho^i_{(x_j;j),j}$ as in (6.6) or (6.7).

From the definitions, we have:

$$\sum_{x_s\in S_s}\rho^i_{(x_s;s),a}=\sum_{x_a\in S_a}\rho^i_{(x_a;s),a}\,,\forall\ (i,s,a)\in[N]^3. \tag{6.8}$$

These equality relations are important to obtain a sufficiently strong relaxation. They express compatibility conditions that are clearly true because the marginals are obtained from the same original occupation measure. Another intuitive way to think about this type of constraints in the following is that we will enforce sample path constraints only in average. Indeed, we must have, at each period t,

$$\sum_{x_s\in S_s}\mathbb{1}\{X_s(t)=x_s,S_i(t)=s,A_i(t)=a\}=\sum_{x_a\in S_a}\mathbb{1}\{X_a(t)=x_a,S_i(t)=s,A_i(t)=a\}\,,\forall\ (i,s,a)\in[N]^3,$$

where $\mathbb{1}\{\cdot\}$ denotes the indicator function. Equation (6.8) is obtained by taking expected values on both sides of this relation, discounting and summing over time, and dividing by $(1-\alpha)$.

For agent i, some $\mathbf{x} \in \mathscr{S}$ and $\mathbf{s}$ in $\Pi_{[N]}$, we can sum the constraints in (6.4) over $x_{s_j} \in S_{s_j}$ for $j \neq i$ to get, for all $x_{s_i} \in S_{s_i}$

$$\sum_{\mathbf{a} \in \Pi_{[N]}} \rho_{(x_{s_i};\mathbf{s}),\mathbf{a}} - \alpha \sum_{\mathbf{s}' \in \Pi_{[N]}} \sum_{\tilde{x}_{s_i} \in S_{s_i}} \rho_{(\tilde{x}_{s_i};\mathbf{s}'),\mathbf{s}} \, p_{\tilde{x}_{s_i} x_{s_i}}^{\mathbb{1}\{i \leq M\}} = \nu_{s_i}(x_{s_i}) \delta_{d_1}(s_1) \dots \delta_{d_N}(s_N),$$

where

$$\rho_{(x_{s_i};\mathbf{s}),\mathbf{a}} = \sum_{\mathbf{x}_{-s_i} \in \mathscr{S}_{-s_i}} \rho_{(\mathbf{x};\mathbf{s}),\mathbf{a}}.$$

Then, still for s_i fixed, we can sum over $\mathbf{s}_{-i} \in \Pi_{[N]-s_i}$, rewrite $\sum_{\mathbf{a} \in \Pi_{[N]}} = \sum_{a_i=1}^{N} \sum_{\mathbf{a}_{-\mathbf{i}} \in \Pi_{[N]-a_i}}$ and $\sum_{\mathbf{s}' \in \Pi_{[N]}} = \sum_{s_i'=1}^{N} \sum_{\mathbf{s}'_{-\mathbf{i}} \in \Pi_{[N]-s_i'}}$, to obtain

$$\sum_{a=1}^{N} \rho_{(x_s;s),a}^{i} - \alpha \sum_{\tilde{x}_s \in S_s} \sum_{s'=1}^{N} \rho_{(\tilde{x}_s;s'),s}^{i} \, p_{\tilde{x}_s x_s}^{\mathbb{1}\{i \leq M\}} = \nu_s(x_s) \delta_{d_i}(s) \ , \ \forall \ (i,s) \in [N]^2, \forall x_s \in S_s. \tag{6.9}$$

There are still some compatibility conditions between the marginals that have not been expressed, similar to the constraints in (6.8). Usually, these constraints also have an intuitive interpretation. For instance, in Whittle's relaxation of the RBP, there is only one additional constraint, due to the fact that only M sites can be active at each time period. Finding and enforcing these constraints reduces the size of the feasible polytope, and helps improve the tightness of the relaxation.

For the RBPSC, the constraints below are motivated by the fact that exactly one agent (active or passive) must be at each site at each period. The frequency with which the agents leave site j in state x_j should be equal to the frequency with which the agents move to site j in state x_j. So we expect that the following constraints should hold:

$$\sum_{i=1}^{N} \sum_{a=1}^{N} \rho_{\tilde{x}_j,j,a}^{i} = \sum_{i=1}^{N} \sum_{s=1}^{N} \rho_{\tilde{x}_j,s,j}^{i}, \quad \forall j \in [N], \tilde{x}_j \in S_j. \tag{6.10}$$

We now show that (6.10) is indeed a valid constraint. According to the definition of the marginal (6.7), we have for the left hand side

$$\sum_{i=1}^{N} \sum_{a=1}^{N} \rho_{\tilde{x}_j,j,a}^{i} = \sum_{i=1}^{N} \sum_{a=1}^{N} \sum_{\mathbf{x}_{-\mathbf{j}} \in S_{-j}} \sum_{\mathbf{s}_{-i} \in \Pi_{[N]-j}} \sum_{\mathbf{a}_{-i} \in \Pi_{[N]-a}} \rho_{(\mathbf{x}_{-\mathbf{j}},\tilde{x}_j);(\mathbf{s}_{-\mathbf{i}},j),(\mathbf{a}_{-\mathbf{i}},a)} \ ,$$

using the notation $(\mathbf{x}_{-\mathbf{j}},\tilde{x}_j) = (x_1,\dots,x_{j-1},\tilde{x}_j,x_{j+1},\dots,x_N)$, $(\mathbf{s}_{-\mathbf{i}},j) = (s_1,\dots,s_{i-1},j,s_{i+1},\dots,n)$, and $(\mathbf{a}_{-\mathbf{i}},a) = (a_1,\dots,a_{i-1},a,a_{i+1},\dots,n)$. We get

$$\begin{aligned} \sum_{i=1}^{N} \sum_{a=1}^{N} \rho_{\tilde{x}_j,j,a}^{i} &= \sum_{\mathbf{x}_{-\mathbf{j}} \in S_{-j}} \sum_{\mathbf{a} \in \Pi_{[N]}} \sum_{i=1}^{N} \sum_{\mathbf{s}_{-i} \in \Pi_{[N]-j}} \rho_{(\mathbf{x}_{-\mathbf{j}},\tilde{x}_j);(\mathbf{s}_{-\mathbf{i}},j),\mathbf{a}} \\ &= \sum_{\mathbf{x}_{-\mathbf{j}} \in S_{-j}} \sum_{\mathbf{a} \in \Pi_{[N]}} \sum_{\mathbf{s} \in \Pi_{[N]}} \rho_{(\mathbf{x}_{-\mathbf{j}},\tilde{x}_j);\mathbf{s},\mathbf{a}}. \end{aligned} \tag{6.11}$$

The first equality comes from the fact that we count all the permutation vectors $\mathbf{a}$ by varying first the i^{th} component a_i from 1 to N. The second inequality comes from the fact that we count all the permutations $\mathbf{s}$ by varying the position i where the component s_i is equal to j (exactly one of the components of a permutation vector of $\Pi_{[N]}$ has to be j). The proof that the right hand side of (6.10) is also equal to the quantity in (6.11) is identical.

Here are two additional sets of valid constraints:

$$\sum_{s=1}^{N} \sum_{x_s \in S_s} \sum_{a \in [N]-\tilde{a}} \rho^i_{x_s;s,a} = \sum_{k \in [N]-i} \sum_{s=1}^{N} \sum_{x_s \in S_s} \rho^k_{x_s;s;\tilde{a}}\,, \quad \forall (i,\tilde{a}) \in [N]^2, \tag{6.12}$$

$$\sum_{a=1}^{N} \sum_{x_a \in S_a} \sum_{s \in [N]-\tilde{s}} \rho^i_{x_a;s,a} = \sum_{k \in [N]-i} \sum_{a=1}^{N} \sum_{x_a \in S_a} \rho^k_{x_a;\tilde{s};a}\,, \quad \forall (i,\tilde{s}) \in [N]^2. \tag{6.13}$$

Intuitively, on the left hand side we have the probability that agent i does not go to site $\tilde{a}$ (respectively does not leave from site $\tilde{s}$), which must equal the probability that some other agent k (passive or not) goes to site $\tilde{a}$ (respectively leaves from site $\tilde{s}$). Again, these relations can be verified by inspection of (6.7). Indeed, in (6.12) for $(i,\tilde{a})$ fixed, similarly to (6.10), we have two equivalent ways of summing the occupation measure over all indices $(\mathbf{x},\mathbf{s},\mathbf{a})$ such that none of the permutation vectors $\mathbf{a}$ with $\tilde{a}$ in position i appears. On the left hand side of (6.12), we vary the coefficient a_i in the set $\{1,\ldots,N\} \setminus \{\tilde{a}\}$, whereas on the right hand side, we obtain the same result by forcing the element $\tilde{a}$ to be in a position different from position i. Similarly, in (6.13), we have two ways of summing over all indices such that none of the permutation vectors $\mathbf{s}$ with $\tilde{s}$ in position i appears.

Finally we have obtained a relaxation for the RBPSC:

Theorem 6.3.3. *We can obtain an upper bound on the optimal reward achievable in the RBPSC by solving the following linear program:*

maximize

$$\sum_{s=1}^{N} \sum_{a=1}^{N} \sum_{x_a \in S_a} \left[\left(r^1_a(x_a) - c_{sa}\right) \left(\sum_{i=1}^{M} \rho^i_{x_a;s,a} \right) + r^0_a(x_a) \left(\sum_{i=M+1}^{N} \rho^i_{x_a;s,a} \right) \right] \tag{6.14}$$

subject to

$(6.8),\ (6.9), (6.10), (6.12),\ (6.13),$

$\rho^i_{(x_s;s),a} \geq 0\,, \quad \forall (i,s,a) \in [N]^3, x_s \in S_s$

$\rho^i_{(x_a;s),a} \geq 0\,, \quad \forall (i,s,a) \in [N]^3, x_a \in S_a.$

There are now $O(N^3 \times \max_i |S_i|)$ variables $\rho^i_{(x_s;s)a}$, $\rho^i_{(x_a;s)a}$, and constraints in the relaxed linear program, which is polynomial in the size of the input. From the discussion in paragraph 2.3, it is unlikely that a polynomial number of variables will suffice to formulate the RBPSC exactly. However, the addition of the constraints tying the marginals together helps reduce the size of the feasible region spanned by the decision vectors and improve the quality of the relaxation. Computing the optimal value of this linear program can be done in polynomial time, and provides an upper bound on the performance achievable by any policy for the original problem.

Remark 6.3.4. We could obtain tighter relaxations by considering marginals involving several sites and/or agents at the same time. In the limit case, we obtain the exact formulation when all sites and agents are considered simultaneously. This idea is followed by [17] for the restless bandit problem.

6.3.3 Dual of the Relaxation

It will be useful to consider the dual of the LP relaxation obtained in the previous paragraph, which we derive directly from (6.14). This dual program could also be obtained by dynamic programming arguments, in the spirit of the original work of Whittle, incorporating the constraints $(6.8), (6.10), (6.12), (6.13)$ using Lagrange multipliers. This provides additional insight, and in

the next section, we will give an interpretation of this dual LP in terms of approximate dynamic programming. We obtain:

$$\text{minimize} \sum_{i=1}^{N}\sum_{s=1}^{N}\sum_{x_s\in S_s} v_s(x_s)\delta_{d_i}(s)\lambda^i_{s,x_s} \tag{6.15}$$

subject to

$$\lambda^i_{s,x_s}+\mu^i_{s,a}+\kappa_{s,x_s}-\sum_{i'\neq i}\zeta^{i'}_a+\sum_{a'\neq a}\zeta^i_{a'}\geq 0\ ,\forall(i,s,a)\in[N]^3, s\neq a, \tag{6.16}$$

$$-\alpha\sum_{\tilde{x}_a}p^1_{x_a\tilde{x}_a}\lambda^i_{a,\tilde{x}_a}-\mu^i_{s,a}-\kappa_{a,x_a}-\sum_{i'\neq i}\xi^{i'}_s+\sum_{s'\neq s}\xi^i_{s'}\geq r^1_a(x_a)-c_{sa}\ ,\ \forall\ 1\leq i\leq M,\forall s\neq a, \tag{6.17}$$

$$-\alpha\sum_{\tilde{x}_a}p^0_{x_a\tilde{x}_a}\lambda^i_{a,\tilde{x}_a}-\mu^i_{s,a}-\kappa_{a,x_a}-\sum_{i'\neq i}\xi^{i'}_s+\sum_{s'\neq s}\xi^i_{s'}\geq r^0_a(x_a)\ ,\ \forall\ M+1\leq i\leq N,\forall s\neq a,$$

$$\lambda^i_{s,x_s}-\alpha\sum_{\tilde{x}_s}p^1_{x_s\tilde{x}_s}\lambda^i_{s,\tilde{x}_s}-\sum_{i'\neq i}(\zeta^{i'}_s+\xi^{i'}_s)+\sum_{s'\neq s}(\zeta^i_{s'}+\xi^i_{s'})\geq r^1_s(x_s)-c_{ss}\ ,\ \forall 1\leq i\leq M, \tag{6.18}$$

$$\lambda^i_{s,x_s}-\alpha\sum_{\tilde{x}_s}p^0_{x_s\tilde{x}_s}\lambda^i_{s,\tilde{x}_s}-\sum_{i'\neq i}(\zeta^{i'}_s+\xi^{i'}_s)+\sum_{s'\neq s}(\zeta^i_{s'}+\xi^i_{s'})\geq r^0_s(x_s)\ ,\ \forall M+1\leq i\leq N.$$

The optimal dual variables $\bar{\lambda}^i_{s,x_s}$ are Lagrange multipliers corresponding to the constraints (6.9) and have a natural interpretation in terms of reward-to-go if site s is in state x_s and visited by agent i (see section 6.4). The optimal dual variables $\bar{\mu}^i_{s,a},\bar{\kappa}_{a,x_a},\bar{\zeta}^i_a,\bar{\xi}^i_s$ correspond to the additional constraints (6.8), (6.10), (6.12), and (6.13) respectively. We can obtain the optimal primal and dual variables simultaneously when solving the relaxation. For $j=s$ or a, we also obtain the optimal reduced costs $\bar{\gamma}^i_{x_j,s,a}$: $\bar{\gamma}^i_{x_s,s,a}$ are equal to the left hand side of the constraints (6.16), whereas $\bar{\gamma}^i_{x_a,s,a}$ are equal to the difference between the left hand side and the right hand side of (6.17), or (6.18) if $s=a$. There is one such reduced cost for each variable $\rho^i_{x_j,s,a}$ of the primal, and by complementary slackness, $\bar{\rho}^i_{x_j,s,a}\bar{\gamma}^i_{x_j,s,a}=0$, where $\bar{\rho}^i_{x_j,s,a}$ are the optimal values of the primal.

6.4 Heuristics for the RBPSC

Computing the optimum value of the relaxation presented in the previous section provides a bound on the performance achievable by any assignment policy. The relaxation is also useful to actually design assignment policies for the agents, using approximate dynamic programming techniques. We present here a one-step lookahead policy and its relationship with the primal-dual heuristic of Bertsimas and Ninõ-Mora, developped for the restless bandit problem. Another very simple greedy policy will be used for comparisons in the numerical experiments presented in section 6.5.

6.4.1 One-Step Lookahead Policy

Consider the multi-agent system at a given time, for which the state is

$$(x_1,\ldots,x_N;s_1,\ldots,s_N),$$

with $(s_1,\ldots,s_N)$ a permutation of $[N]$. Given the interpretation of the dual variables $\lambda^i_{s_i,x_{s_i}}$ in terms of reward-to-go mentioned in section 6.3.3, it is natural to try to form an approximation $\tilde{J}(\mathbf{x};\mathbf{s})$ of

the global reward-to-go in state $(\mathbf{x};\mathbf{s})$ as

$$\tilde{J}(x_1,\ldots,x_N;s_1,\ldots,s_N)=\sum_{i=1}^{N}\bar{\lambda}^{\,i}_{s_i,x_{s_i}}, \tag{6.19}$$

where $\bar{\lambda}^{\,i}_{x_{s_i},s_i}$ are the optimal values of the dual variables obtained when solving the LP relaxation. The separable form of this approximate cost function is useful to design an easily computable one-step lookahead policy [14], as follows. At state $(\mathbf{x};\mathbf{s})$, we obtain the assignment of the agents by solving

$$\tilde{u}(\mathbf{x};\mathbf{s})\in \operatorname{argmax}_{\mathbf{a}\in\Pi_{[N]}}\left\{g((\mathbf{x};\mathbf{s}),\mathbf{a})+\alpha\sum_{\mathbf{x}'\in\mathscr{S}}\mathscr{P}_{\mathbf{x}\mathbf{a}\mathbf{x}'}\tilde{J}(\mathbf{x}';\mathbf{a})\right\}, \tag{6.20}$$

where $g((\mathbf{x};\mathbf{s}),\mathbf{a})$ is the immediate reward

$$g((\mathbf{x};\mathbf{s}),\mathbf{a})=\sum_{i=1}^{M}(r^1_{a_i}(x_{a_i})-c_{s_ia_i})+\sum_{i=M+1}^{N}r^0_{a_i}(x_{a_i}) \tag{6.21}$$

and from (6.2)

$$\mathscr{P}_{\mathbf{x}\mathbf{a}\mathbf{x}'}=\mathscr{P}_{(\mathbf{x};\mathbf{s})\mathbf{a}(\mathbf{x}';\mathbf{a})}=\prod_{i=1}^{M}p^1_{x_{a_i}x'_{a_i}}\prod_{i=M+1}^{N}p^0_{x_{a_i}x'_{a_i}}. \tag{6.22}$$

In this computation, we replaced the true optimal cost function, which would provide an optimal policy, by the approximation $\tilde{J}$. Using (6.19), we have the following maximization problem:

$$\begin{aligned}\max_{\mathbf{a}\in\Pi[N]}\Bigg\{&\sum_{i=1}^{M}\left(r^1_{a_i}(x_{a_i})-c_{s_ia_i}+\alpha\sum_{x'_{a_i}\in S_{a_i}}\bar{\lambda}^{\,i}_{a_i,x'_{a_i}}p^1_{x_{a_i}x'_{a_i}}\right)\\&+\sum_{i=M+1}^{N}\left(r^0_{a_i}(x_{a_i})+\alpha\sum_{x'_{a_i}\in S_{a_i}}\bar{\lambda}^{\,i}_{a_i,x'_{a_i}}p^0_{x_{a_i}x'_{a_i}}\right)\Bigg\}.\end{aligned} \tag{6.23}$$

Assuming that the optimal dual variables have been stored in memory, the maximization above is actually easy to perform. The evaluation of one parenthesis involves only the data of the problem for the current state of the system, and one summation over the states of one site, i.e., takes a time $\mathscr{O}(\max_i|S_i|)$. Let us denote the terms in parenthesis m_{i,a_i}. All these possible terms can be computed in time $\mathscr{O}(N^2\max_i|S_i|)$, and (6.23) can be rewritten:

$$\max_{\mathbf{a}\in\Pi_{[N]}}\sum_{i=1}^{N}m_{i,a_i}. \tag{6.24}$$

This is a linear assignment problem, which can be solved by linear programming or in time $O(N^3)$ by the Hungarian method [114]. Thus, the assignment is computed at each time step in time

$$O(N^2\max_i|S_i|+N^3)$$

by a centralized controller, which needs to store the optimal dual variables of the relaxation in addition to the parameters of the problem.

6.4.2 Equivalence with the Primal-Dual Heuristic

Recall from paragraph 6.3.3 that, when solving the linear programming relaxation, we can obtain the optimal primal variables $\{\bar{\rho}^i_{x_j,s,a}\}$, the dual variables $\{\bar{\lambda}^i_{s,x_s}, \bar{\mu}^i_{s,a}, \bar{\kappa}_{a,x_a}, \bar{\zeta}^i_a, \bar{\xi}^i_s\}$, and the reduced costs $\{\bar{\gamma}^i_{x_j,s,a}\}$. These reduced costs are nonnegative. In [17], Bertsimas and Niño-Mora motivated their primal-dual heuristic for the RBP using the following well-known interpretation of the reduced costs: *starting from an optimal solution, $\bar{\gamma}^i_{(x_j;s),a}$ is the rate of decrease in the objective value of the primal linear program (6.14) per unit increase in the value of the variable $\rho^i_{(x_j;s),a}$.*

We use this interpretation and the following intuitive idea: when agent i is in site s in state x_s and we decide to send it to site a in state x_a, in some sense we are increasing the values of $\rho^i_{(x_s;s),a}$ and $\rho^i_{(x_a;s),a}$, which are proportional to the long-term probabilities of such transitions. In particular, we would like to keep the quantities $\bar{\rho}^i_{(x_j;s),a}$ found to be 0 in the relaxation as close to 0 as possible in the final solution. By complementary slackness it is only for these variables that we might have $\bar{\gamma}^i_{(x_j;s),a} > 0$. Hence, when the system is in state $(\mathbf{x};\mathbf{s})$, we associate to each action $\mathbf{a}$ an *index of undesirability*

$$I((\mathbf{x};\mathbf{s}),\mathbf{a}) = \sum_{\{i\in[N]:s_i\neq a_i\}}^{N} (\bar{\gamma}^i_{(x_{s_i};s_i),a_i} + \bar{\gamma}^i_{(x_{a_i};s_i),a_i}) + \sum_{\{i\in[N]:s_i=a_i\}} \bar{\gamma}^i_{(x_{s_i};s_i),s_i}, \tag{6.25}$$

that is, we sum the reduced costs for the N different projects. The separation of the two cases $s_i \neq a_i$ and $s_i = a_i$ in the summation is justified below from the form of the reduced costs. Then we select an action $\mathbf{a_{pd}} \in \Pi_{[N]}$ that minimizes these indices:

$$\mathbf{a_{pd}}(\mathbf{x};\mathbf{s}) \in \operatorname{argmin}_{\mathbf{a}}\{I((\mathbf{x};\mathbf{s}),\mathbf{a})\}. \tag{6.26}$$

We now show that this policy is in fact equivalent to the one-step lookahead policy described earlier. Using the expression for the reduced costs from paragraph 6.3.3, we can rewrite the indices in (6.25) more explicitly. The term $\bar{\gamma}^i_{(x_{s_i};s_i),a_i} + \bar{\gamma}^i_{(x_{a_i};s_i),a_i}$ of the sum (6.25) is equal to

$$\begin{aligned}
&\bar{\lambda}^{\,i}_{s_i,x_{s_i}} + \bar{\mu}^i_{s_i,a_i} + \bar{\kappa}_{s_i,x_{s_i}} - \sum_{i'\neq i} \bar{\zeta}^{i'}_{a_i} + \sum_{a'\neq a_i} \bar{\zeta}^i_{a'} \\
&- \alpha \sum_{\tilde{x}_{a_i}} p^{\mathbb{1}\{i\leq M\}}_{x_{a_i}\tilde{x}_{a_i}} \bar{\lambda}^{\,i}_{a_i,\tilde{x}_{a_i}} - \bar{\mu}^i_{s_i,a_i} - \bar{\kappa}_{a_i,x_{a_i}} - \sum_{i'\neq i} \bar{\xi}^{i'}_{s_i} + \sum_{s'\neq s} \bar{\xi}^{\,i}_{s'} - r^{\mathbb{1}\{i\leq M\}}_{a_i}(x_{a_i}) + c_{s_i a_i}\mathbb{1}\{i\leq M\} \\
&= \bar{\lambda}^{\,i}_{s_i,x_{s_i}} + \bar{\kappa}_{s_i,x_{s_i}} - \sum_{i'=1}^{N} \bar{\xi}^{i'}_{s_i} + \sum_{s'=1}^{N} \bar{\xi}^{\,i}_{s'} - \sum_{i'=1}^{N} \bar{\zeta}^{i'}_{a_i} + \sum_{a'=1}^{N} \bar{\zeta}^i_{a'} - \bar{\kappa}_{a_i,x_{a_i}} \\
&- \alpha \sum_{\tilde{x}_{a_i}} p^{\mathbb{1}\{i\leq M\}}_{x_{a_i}\tilde{x}_{a_i}} \bar{\lambda}^{\,i}_{a_i,\tilde{x}_{a_i}} - r^{\mathbb{1}\{i\leq M\}}_{a_i}(x_{a_i}) + c_{s_i a_i}\mathbb{1}\{i\leq M\},
\end{aligned} \tag{6.27}$$

after cancellation of $\bar{\mu}^i_{s_i,a_i}$, and adding and subtracting $\bar{\zeta}^i_{a_i}$ and $\bar{\xi}^i_{s_i}$. This expression is valid for the terms $\gamma^i_{(x_{s_i};s_i),s_i}$ as well. Now after summation over $i \in [N]$, the first line in expression (6.27) does not play any role in the minimization (6.26). This is obvious for the terms that do not depend on $\mathbf{a}$. For the terms involving the $\zeta^i_{a_j}$, we can write

$$\sum_{i=1}^{N}\sum_{i'=1}^{N} \bar{\zeta}^{i'}_{a_i} = \sum_{i'=1}^{N}\sum_{i=1}^{N} \bar{\zeta}^{i'}_{a_i} = \sum_{i'=1}^{N}\sum_{a=1}^{N} \bar{\zeta}^{i'}_{a},$$

the last equality being true since $\mathbf{a}$ is just a permutation of $\{1,\ldots,N\}$. Hence the sums involving the $\bar{\zeta}^i_{a_i}$ cancel (in fact we even see that each individual sum is independent of the choice of $\mathbf{a}$). As for the term $\sum_{i=1}^N \bar{\kappa}_{a_i,x_{a_i}}$, it is equal to $\sum_{j=1}^N \bar{\kappa}_{j,x_j}$ and so it is independent of the choice of $\mathbf{a} \in \Pi_{[N]}$. We are left with the following optimization problem:

$$\mathbf{a_{pd}}(\mathbf{x};\mathbf{s}) \in \operatorname{argmin}_{\mathbf{a}} \left\{ -\sum_{i=1}^{N} \left(r_{a_i}^{\mathbb{1}\{i\le M\}}(x_{a_i}) - c_{s_i a_i} \mathbb{1}\{i \le M\} + \alpha \sum_{\tilde{x}_{a_i}} p_{x_{a_i}\tilde{x}_{a_i}}^{\mathbb{1}\{i\le M\}} \bar{\lambda}^i_{a_i,\tilde{x}_{a_i}} \right) \right\},$$

which after a sign change is seen to be exactly (6.23). We have shown the following

Theorem 6.4.1. *The primal-dual heuristic (6.26), based on the interpretation of the reduced costs of the LP relaxation, is equivalent to the one-step lookahead policy (6.23) assuming the separable approximation (6.19) for the reward-to-go.*

In view of this result, we obtain an alternative way to compute the one-step lookahead policy. The minimization (6.26) is again a linear assignment problem. If we can store the $\mathscr{O}(N^3 \max_i(|S_i|))$ optimal reduced costs instead of the $\mathscr{O}(N^2 \max_i(|S_i|))$ optimal dual variables, there is just a linear cost involved in computing the parameters m_{i,a_i} of the problem at each period resulting in an overall $\mathscr{O}(N^3)$ computational cost for the on-line problem at each period. This is negligible in practice since N cannot be very large for the LP relaxation to remain computable.

6.4.3 A Simple Greedy Heuristic

Perhaps the simplest policy for the RBSC problem is the greedy policy which chooses the next action $\mathbf{a} \in \Pi_{[N]}$ maximizing the immediate reward. This can be done as before by solving a linear assignment problem, since it is the same as fixing the value of the $\bar{\lambda}^i_{a_i,x'_{a_i}}$ in (6.23) to zero, i.e., approximating the reward-to-go by zero. This policy is actually optimal for the MAB problem with deteriorating active rewards, i.e., such that projects become less profitable as they are worked on [14, vol. 2, p.69]. This policy is easy to implement, it does not require solving a linear program, and we will use it as a benchmark in our numerical simulations.

6.4.4 Additional Computational Considerations

The major bottleneck limiting the use of our method for large-scale problems is the (off-line) computation of the relaxation (6.14) which involves a large number of variables. Solving a problem with 30 sites which can each be in one of two states requires solving a linear program with about 10^5 variables, independently of the number of agents used. Although this is a computation which can still be carried out, as the next section on numerical experiments will demonstrate, the limits of the state-of-the-art linear programming technology are reached for relatively small problems. For these problems of limited size, once the linear program has been solved, the one-step lookahead policy is easily computed in real-time during the experiment.

Since the one-step lookahead policy can be computed quickly for the range of parameters of interest, it is tempting to try to use it as a base policy to obtain a rollout policy, whose performance is known to be at least as good as the base policy, as in [15, 14]. In practice, we noticed interesting improvements when we could indeed implement it, in the single agent formulation presented in [81]. In the multi-agent case however, this policy is difficult to implement since the number of next possible states and actions, from which an approximate value function is to be estimated, grows very fast with the size of the problem.

6.5 Numerical Simulations

Table 6.1 presents numerical experiments on problems whose characteristics differently affect the performance of the heuristics described in section 6.4. In some cases, when the size of the state space allows it, we also compute the optimal performance as the solution of the linear program (6.3). In any case, we give an upper bound on this performance using the relaxation (6.14). Linear programs are implemented in AMPL and solved using CPLEX. Due to the size of the state space, the expected discounted reward of the various heuristics is computed using Monte-Carlo simulations. The computation of each trajectory is terminated after a sufficiently large, but finite horizon: in our case, when α^t times the maximal absolute value of any immediate reward becomes less than 10^{-6}. To reduce the amount of computation in the evaluation of the policies, we assume that the distribution of the initial states of the sites is deterministic.

In table 6.1, we adopt the following nomenclature:

- α: discount factor.
- N: number of sites/projects.
- M: number of agents.
- $|S_i|$: number of states per project (all projects have the same number of states).
- c/r: ratio of the average switching cost divided by the average active reward. This is intended to give an idea of the importance of the switching costs in the particular experiment. The switching costs are always taken to be positive. Hence $c/r = 0$ means that there are no switching costs.
- Z^*: optimal value of the problem (when available).
- Z_r: optimal value of the relaxation.
- Z_g: estimated expected value of the greedy policy.
- Z_{osl}: estimated expected value of the one-step lookahead policy.

Here is a brief description of the scenarios considered in the simulations. Problem 1 is a multi-armed bandit problem (no transition costs, one agent, the passive sites are frozen) with 4 sites and 3 states per site. The index policy is not optimal, so we see that we do not recover Gittins' policy. Hence the policy is also different from Whittle's policy in general, which reduces to Gittins' in the MABP. Problem 2 is the same as problem 1, but the active transition probability matrices are changed so that the problem satisfies the deteriorating reward property. In this case, it is known that the greedy policy is optimal. On the other hand, we see that the one-step lookahead policy does not perform optimally. In problem 3, we add transition costs to problem 2. The greedy policy is not optimal any more, and the one-step lookahead policy performs better in this case. Problem 4 is designed to make the greedy policy underperform: two remote sites have slightly larger initial rewards (taking into account the cost for reaching them), but the active rewards at these sites are rapidly decreasing and the agents are overall better off avoiding these sites. The greedy policy does not take into account the future transition costs incurred when leaving these sites. In this case, it turns out that the one-step lookahead is quasi-optimal. In problem 5 on the other hand, the greedy policy performs better, which means that no guarantee of a better performance from the one-step lookahead policy can be claimed for a general problem. However, from the conducted experiments, it seems that the one-step lookahead policy has often an advantage when the switching costs are

Table 6.1: Numerical Experiments

Problem $(N, M, \lvert S_i\rvert, c/r)$	α	Z^*	Z_r	Z_g	Z_{osl}
Problem 1	0.5	84.69	85.21	84.5	84.3
(4,1,3,0)	0.9	299.6	301.4	276	294
	0.99	2614.2	2614	2324	2611
Problem 2	0.5	84.13	85.14	84.1	84.1
(4,1,3,0)	0.9	231.0	245.1	231	228
	0.99	1337	1339	1337	1336
Problem 3	0.5	57.54	59.32	56.0	57.3
(4,1,3,0.6)	0.9	184.5	185.0	177	183
	0.99	1279	1280	1273	1277
Problem 4	0.5		165.7	115	165
(4,2,5,1.39)	0.9		767.2	661	767
	0.95		1518	1403	1518
Problem 5	0.5		39.25	38.5	36.5
(6,2,4,0)	0.9		214.0	205	198
	0.95		431.6	414	396
Problem 6	0.5		9.727	6.93	8.24
(6,2,4,1.51)	0.9		62.80	38.0	47.0
	0.95		128.7	78.0	99.0
Problem 7	0.5		196.5	189	194
(20,15,3,1.16)	0.9		952.7	877	900
	0.95		1899	1747	1776
Problem 8	0.5		589.4	566	564
(30,15,2,2.18)	0.9		2833	2640	2641
	0.95		5642	5218	5246

significant. Moreover, it is not obvious how to design experiments where it clearly underperforms, in contrast to the greedy policy in problem 4. Problem 7 and 8 are larger scale problems, with up to 30 sites. The relaxations are computed in about 20 minutes on a standard desktop, showing the feasibility of the approach for this range of parameters.

6.6 A "Performance" Bound

In this section, we present a result that offers some insight into why we could expect the one-step lookahead policy to perform well if the linear programming relaxation of the original problem is sufficiently tight. We begin with the following

Lemma 6.6.1. *The approximate reward-to-go (6.19) is a super-harmonic vector, i.e., it is feasible for the original dual linear program (6.5).*

Proof. Consider one constraint in the original dual LP (6.5), for a fixed state-action tuple $(\mathbf{x}, \mathbf{s}, \mathbf{a})$. We consider a situation where $s_i \neq a_i$ for all $i \in \{1, \ldots, N\}$. Summing the constraints (6.16) over i

for the given values of x_{s_i}, s_i, a_i, we get

$$\sum_{i=1}^{N} \lambda^i_{s_i,x_{s_i}} + \sum_{i=1}^{N} \mu^i_{s_i,a_i} + \sum_{i=1}^{N} \kappa_{s_i,x_{s_i}} - \sum_{i=1}^{N}\sum_{i'=1}^{N} \zeta^{i'}_{a_i} + \sum_{i=1}^{N}\sum_{a'=1}^{N} \zeta^{i}_{a'}$$
$$= \sum_{i=1}^{N} \lambda^i_{s_i,x_{s_i}} + \sum_{i=1}^{N} \mu^i_{s_i,a_i} + \sum_{i=1}^{N} \kappa_{s_i,x_{s_i}} \geq 0. \tag{6.28}$$

The cancellation follows from the discussion preceding theorem 6.4.1. Now summing the constraints (6.17) over i, we also get

$$-\alpha \sum_{i=1}^{N} \sum_{\tilde{x}_{a_i}} p^{\mathbb{1}\{i\leq M\}}_{x_{a_i}\tilde{x}_{a_i}} \lambda^i_{a_i,\tilde{x}_{a_i}} - \sum_{i=1}^{N} \mu^i_{s_i,a_i} - \sum_{i=1}^{N} \kappa_{a_i,x_{a_i}} \geq \sum_{i=1}^{N} \left(r^{\mathbb{1}\{i\leq M\}}_{a_i}(x_{a_i}) - c_{s_i a_i} \mathbb{1}\{i \leq M\} \right).$$

Finally, we add these two inequalities. We obtain

$$\sum_{i=1}^{N} \lambda^i_{s_i,x_{s_i}} - \alpha \sum_{i=1}^{N} \sum_{\tilde{x}_{a_i}} p^{\mathbb{1}\{i\leq M\}}_{x_{a_i}\tilde{x}_{a_i}} \lambda^i_{a_i,\tilde{x}_{a_i}} \geq \sum_{i=1}^{N} \left(r^{\mathbb{1}\{i\leq M\}}_{a_i}(x_{a_i}) - c_{s_i a_i} \mathbb{1}\{i \leq M\} \right),$$

which is the inequality obtained by using the vector (6.19) in the constraints of (6.5).

The case where $s_i = a_i$ for some i is almost identical, considering the constraints (6.18) for the corresponding indices. □

In the following theorem, using the notation of section 3.2.3 we will use the occupation measure (state frequency) $F_\alpha(\nu, \tilde{u})$ on the state space, which we define from the state-action frequency simply as

$$F_\alpha(\nu, \tilde{u}; \mathbf{x}, \mathbf{s}) = \sum_{\mathbf{a} \in \Pi_{[N]}} f_\alpha(\nu, \tilde{u}; \mathbf{x}, \mathbf{s}, \mathbf{a}),\ \forall (\mathbf{x}, \mathbf{s}).$$

Theorem 6.6.2. *Let ν be an initial distribution on the states, of the product form (6.1). Let J^* be the optimal reward function, $\tilde{J}$ be an approximation of this reward function formed according to (6.19), and $\tilde{u}$ be the associated one-step lookahead policy. Let $F_\alpha(\nu, \tilde{u})$ and $J_{\tilde{u}}$ be the occupation measure vector and the expected reward associated to the policy $\tilde{u}$. Then*

$$\nu^T (J^* - J_{\tilde{u}}) \leq \frac{1}{1-\alpha} F_\alpha(\nu, \tilde{u})^T (\tilde{J} - J^*), \tag{6.29}$$

or in other words,

$$\sum_{\mathbf{x},\mathbf{s}} \nu(\mathbf{x}, \mathbf{s}) |J^*(\mathbf{x}, \mathbf{s}) - J_{\tilde{u}}(\mathbf{x}, \mathbf{s})| \leq \frac{1}{1-\alpha} \sum_{\mathbf{x},\mathbf{s}} F_\alpha(\nu, \tilde{u}; \mathbf{x}, \mathbf{s}) (\tilde{J}(\mathbf{x}, \mathbf{s}) - J^*(\mathbf{x}, \mathbf{s})).$$

In words, the theorem says that starting with a distribution ν over the states, the difference in expected rewards between the optimal policy and the one-step lookahead policy is bounded by a weighted l^1-distance between the estimate $\tilde{J}$ used in the design of the policy and the optimal value function J^*. The weights are given by the occupation measure of the one-step lookahead policy. This result is true for every one-step lookahead policy that uses a superharmonic vector as an approximation of the cost-to-go. It gives some motivation to obtain a good approximation $\tilde{J}$ and a tight relaxation.

Proof. We follow the analysis presented in [34]. First, by lemma 6.6.1, the vector $\tilde{J}$ in (6.19) is a

feasible vector for the linear program (6.5), and since J^* is the optimal solution we have

$$\nu^T \tilde{J} \geq \nu^T J^*. \tag{6.30}$$

For an action $\mathbf{a} \in \Pi_{[N]}$, we denote by $g_{\mathbf{a}}$ the vector of immediate rewards, i.e., from the definition (6.21), $g_{\mathbf{a}}(\mathbf{x};\mathbf{s}) = g((\mathbf{x};\mathbf{s}),\mathbf{a})$. Recall also the definition (6.22) of the transition matrix $\mathscr{P}_{\mathbf{x}\mathbf{a}\mathbf{x}'}$. Let $T_{\mathbf{a}}$ and T the dynamic programming operators

$$(T_{\mathbf{a}}J)(\mathbf{x};\mathbf{s}) = g_{\mathbf{a}}(\mathbf{x};\mathbf{s}) + \alpha \sum_{\mathbf{x}' \in \mathscr{S}} \mathscr{P}_{\mathbf{x}\mathbf{a}\mathbf{x}'} J(\mathbf{x}';\mathbf{a}),$$

$$(TJ)(\mathbf{x};\mathbf{s}) = \max_{\mathbf{a}} \left(g_{\mathbf{a}}(\mathbf{x};\mathbf{s}) + \alpha \sum_{\mathbf{x}' \in \mathscr{S}} \mathscr{P}_{\mathbf{x}\mathbf{a}\mathbf{x}'} J(\mathbf{x}';\mathbf{a}) \right).$$

For a deterministic stationary policy $u : \mathscr{S} \times \Pi_{[N]} \to \Pi_{[N]}$, we will also simply write u instead of $u(\mathbf{x},\mathbf{s})$ when it appears as a subscript. Then the cost $J_{\tilde{u}}$ of policy $\tilde{u}$ defined in the theorem is the solution of the linear system $J_{\tilde{u}} = T_{\tilde{u}} J_{\tilde{u}}$. If we also denote by $P_{\tilde{u}}$ the stochastic matrix $\mathscr{P}_{\mathbf{x}\tilde{u}(\mathbf{x};\mathbf{s})\mathbf{x}'}$, this can be written as

$$J_{\tilde{u}} = g_{\tilde{u}} + \alpha P_{\tilde{u}} J_{\tilde{u}}.$$

Now $(I - \alpha P_{\tilde{u}})$ has an inverse since $P_{\tilde{u}}$ is a stochastic matrix and $\alpha < 1$, so we get

$$J_{\tilde{u}} = (I - \alpha P_{\tilde{u}})^{-1} g_{\tilde{u}}.$$

Moreover, under policy $\tilde{u}$, the state of the system evolves as a Markov chain, and we have $\mathbb{P}^u_\nu(\mathbf{X}_t = \mathbf{x}, \mathbf{S}_t = \mathbf{s}) = \left[\nu^T P_{\tilde{u}}^t\right]_{\mathbf{x},\mathbf{s}}$. So the occupation measure is

$$F_\alpha(\nu,\tilde{u}) = (1-\alpha) \sum_{t=0}^{\infty} \alpha^t \nu^T P_{\tilde{u}}^t = (1-\alpha)\nu^T (I - \alpha P_{\tilde{u}})^{-1}. \tag{6.31}$$

Now

$$\tilde{J} - J_{\tilde{u}} = (I - \alpha P_{\tilde{u}})^{-1} \left[(I - \alpha P_{\tilde{u}})\tilde{J} - g_{\tilde{u}} \right] = (I - \alpha P_{\tilde{u}})^{-1} \left[\tilde{J} - (g_{\tilde{u}} + \alpha P_{\tilde{u}} \tilde{J}) \right].$$

By the definition of the one-step lookahead policy $g_{\tilde{u}} + \alpha P_{\tilde{u}} \tilde{J} = T\tilde{J}$. So we obtain

$$\tilde{J} - J_{\tilde{u}} = (I - \alpha P_{\tilde{u}})^{-1} \left[\tilde{J} - T\tilde{J} \right].$$

Then starting from (6.30) and using (6.31)

$$\nu^T (J^* - J_{\tilde{u}}) \leq \nu^T (\tilde{J} - J_{\tilde{u}}) \leq \frac{1}{1-\alpha} F_\alpha(\nu,\tilde{u})^T (\tilde{J} - T\tilde{J}).$$

By lemma 6.6.1 we know that $\tilde{J} \geq T\tilde{J}$. Then by the monotonicity of the operator T the fact that $\lim_{N\to\infty} T^N \tilde{J} = J^*$, we obtain $T\tilde{J} \geq T^2\tilde{J} \geq \ldots \geq J^*$. So

$$\nu^T (J^* - J_{\tilde{u}}) \leq \frac{1}{1-\alpha} F_\alpha(\nu,\tilde{u})^T (\tilde{J} - J^*),$$

which is the inequality in the theorem. □

6.7 Conclusion

We have presented a linear programming relaxation for a modified version of the restless bandit problem, adding costs to switch between the bandits. This set-up is quite powerful to model a wide range of dynamic resource allocation problems. Since the problem is computationally intractable and an optimal solution can in general not be directly computed, the relaxation is useful in providing a bound on the performance achievable by any policy. We also presented a heuristic to solve the problem in practice, as well as its performance on specific examples. An interesting part of the analysis showed the equivalence between the approximate dynamic programming point of view and a technique derived from linear programming theory based on the interpretation of the reduced costs. The relaxation can be obtained automatically by first formulating the original problem as a Markov decision process on the whole state space and then optimizing over specific marginals of the occupation measure. The techniques presented only rely on the partially separable structure of the problem and should be useful for other problems with similar structure.

Chapter 7

Continuous Path-Planning for a Data-Harvesting Vehicle

UAS are actively used in various civil and military ISR missions. Another application under consideration mentioned in the introduction is to use UAVs as communication relays, for example between spatially separated groups on a battlefield [128]. Researchers are now developing numerous models targeted towards these data harvesting applications for mobile sensor systems, see e.g. [51, 85, 116, 44, 102]. Usually the path planning problem is simplified by considering discrete models where data is transfered from locations in the environment to the mobile elements only when the locations are physically visited by these vehicles. The path of the vehicles (when it is not assumed to be a random walk) is then determined by solving a combinatorial optimization problem, such as the Mobile Element Scheduling Problem [120] or other variations of vehicle routing problems. This modeling approach discretizing the exploration space was also adopted in the previous chapters on stochastic dynamic scheduling.

When mobile robots are to be used for remote sensing or as communication relays however, it is preferable to optimize their trajectories without assuming a priori a fixed discrete set of possible travel locations. Moreover, measurements and transmission occur during the motion of the sensor, and this motion can even play a crucial role in the quality of the measurements, such as in synthetic aperture radar imaging. These features cannot be captured by the discrete models considered above. In addition, we saw in chapter 6 that the addition of the path planning component to standard stochastic models used in scheduling, via switching costs or times, leads to problems that are very difficult to solve and for which few results are available. For instance, the polling problem with two queues and switching times is still open [105]. This motivates us to introduce in this chapter a more flexible framework for combined path-planning and scheduling, inspired by the deterministic fluid approximations used for controlling stochastic networks. This model, although not as detailed as models based on MDPs, should provide insight for the development of good control policies for complicated spatiotemporal scheduling scenarios.

7.1 Introduction

In this chapter, we assume that the work to be performed at the sites by the vehicles can be modeled as jobs arriving at spatially distributed queues. For example, requests for information transfer between a site and an UAV, via measurement or wireless communication, arrive at the site and must be served by the vehicle. If this problem is approached from a queueing theory perspective, there is an extensive literature under the rubric of polling systems [123], which is mostly concerned with

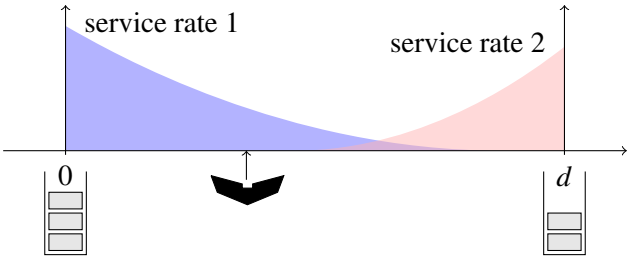


Figure 7-1: Uplink from two fixed stations to a mobile receiver, with spatially decaying service rates.

performance evaluation of open-loop path-planning policies (the queues are served in some fixed cyclic order). The work on polling systems also assumes a discrete model where the service at the queues can be performed only if the server is present at the queue location.

In contrast we consider a continuous polling model where a server can serve spatially separated queues with service rates that depend on the position of the server with respect to the queues. This could model for example the uplink of a wireless transmission system where base stations on the ground transmit their data to a mobile fusion center. Another application concerns planning the trajectory of a mobile measurement station: the quality of the measurements and the time to acquire a bit of information typically depend on the distance to the target. A related integrated model involving path planning and information collection was presented recently in [71].

The goal is to design the trajectory of the server in order to:

- stabilize the system, i.e., keep the number of tasks in the different queues uniformly bounded.
- optimize a given performance measure, such as minimizing the total number of jobs at all queues over a given horizon.

To obtain a tractable model for analysis and control design, we rely on a deterministic fluid approximation of the problem. Fluid models have proved to be very useful in obtaining insight and policies with good performance in the control of queueing networks [92], including discrete polling systems [77], where more traditional approaches based on MDPs are often too detailed and suffer the curse of dimensionality.

The rest of the chapter is organized as follows. Section 7.2 describes the fluid model considered. A necessary and sufficient condition for its stabilization, the main result of this chapter, is presented in section 7.3. Finally in section 7.4, we look at the trajectory optimization problem for an example of a draining problem in which the goal is to collect an initial set of packets from two queues as efficiently as possible, without taking into account new arrivals. This model could be useful to develop batch policies ensuring a degree of fairness in service by periodically freezing the jobs to be served next in the queues. A simple performance bound and numerical simulations briefly discuss the relationship between the fluid model and stochastic models assuming variability.

7.2 Fluid Model

Fig. 7-1 presents a conceptual model of the problem for two base stations or sites. Some information to be transmitted to a mobile server arrives at N queues located at fixed sites, with constant rates α_i, $i = 1,\ldots,N$. We represent this information as a continuous quantity, called fluid hereafter. The fluid present in queue i can be drained from the queue at maximal rate $\mu_i(x(t))$, which is a function

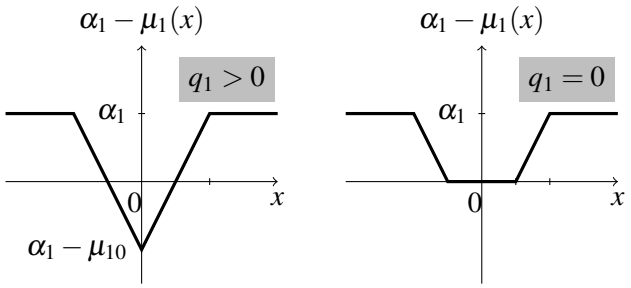


Figure 7-2: Example of a one-dimensional filling rate profile for a site at position $x = 0$.

of the position $x(t)$ of the server at time t. The allocation rate of the server to queue i is denoted ζ_i, with $0 \leq \zeta_i \leq 1$, so that the actual service rate for queue i at time t is $\mu_i(x(t))\zeta_i(t)$. Both the position $x(t)$ and the allocation rate vector $\zeta(t)$ can be controlled by the server, subject to constraints. The server might have dynamics, constraining the choice of trajectories. As we will see, in general the dynamics of the vehicle do not play a role in the stability condition for the system. However, they influence the delay in servicing the jobs in the queues. As an example, we assume in section 7.4 that the server has the trivial dynamics with bounded velocity

$$\dot{x} = u, \;\; |u| \leq V, \;\; x(0) = x_0, \tag{7.1}$$

where u is the velocity control. The allocation rate vector is always subject to the constraint $0 \leq \zeta \leq 1$. Additional constraints, potentially dependent on the position of the server, and modeling for example the communication capacity region in the multiple base station transmission scenario, are assumed to be linear with respect to the rate allocation vector. Hence the constraints on the rate allocation vector can be summarized as

$$\zeta \geq 0, \;\; C(x)\zeta \leq 1,$$

where $C(x)$ is called the *constituency matrix* when the server is at position x. The dynamics of the queues are described by the following differential equation

$$\frac{d^+}{dt} q_i(t) = \Big(\alpha_i - \mu_i(x(t))\zeta_i(t)\Big)^+_{q_i}, \tag{7.2}$$

where $\frac{d^+}{dt}$ denotes the right derivative, and we use the notation

$$(g(x))^+_y = \begin{cases} g(x) \text{ if } y > 0 \\ \max\{g(x), 0\} \text{ if } y = 0. \end{cases}$$

We can also add constraints imposing finite buffer sizes.

Fig. 7-2 shows an example of a filling rate profile for a one-dimensional problem with service rate $\mu_i(x)$ linearly decreasing with the distance $|x|$ to the queue. The vector field in (7.2) enforces the nonnegativity constraint for the fluid levels in the buffers, and is discontinuous on the boundary where $q_i = 0$ and $\alpha_i - \mu_i(x)\zeta_i < 0$.

Example 7.2.1. Two stations transmit a signal to the mobile server (UAV) over a shared additive white Gaussian noise channel, with power $P_i, i = 1, 2$. The server, when at position x, receives the signal from station i with power $p_i(x) = P_i/(C_i + \|x - x_i\|)^{\alpha_i}$, where C_i and α_i are constants, and

x_i is the position of the i^{th} base station. If the server decodes the transmission of the base stations sequentially, it can achieve communication rates $R_i(x)$ that are subject to the following constraints [33]:

$$R_i \leq \log(1+p_i(x)),\ i=1,2, \quad \text{and } R_1+R_2 \leq \log(1+p_1(x)+p_2(x)).$$

To put this example in our framework, we can let $\mu_i(x) = \log(1+p_i(x))$ for $i=1,2$, $R_i = \mu_i(x)\zeta_i$ and rewrite the constraints

$$0 \leq \zeta_i \leq 1,\ i=1,2, \quad \text{and } \frac{\mu_1(x)\zeta_1+\mu_2(x)\zeta_2}{\log(1+p_1(x)+p_2(x))} \leq 1.$$

The model described above can be generalized as follows. The vector of queue sizes $q \in \mathbb{R}_+^N$ obeys the system of differential equations

$$\frac{d^+}{dt}q(t) = \alpha + B(x(t))\zeta(t), \tag{7.3}$$

where α is the vector of external arrival rates, $\zeta \in \mathbb{R}_+^l$ is the vector of allocation rates, and B is an N-by-l matrix. The position $x(t) \in \mathbb{R}^d$ of the server, where d denotes the dimension of the geometry of the problem, obeys the controlled differential equation (7.1). We let $\mathsf{X} \subset \mathbb{R}^d$ denote the set of *reachable* values for the state x. In addition, the state q of the queues is constrained to belong to a polyhedral state-space Q, which includes the non-negativity constraints and the potential finite buffer constraints. When the system is in state (q,x), the rate allocation vector ζ belongs to a convex polyhedron which depends on the state x of the server

$$\zeta \in \mathsf{U}(x) = \{\zeta :\ \zeta \geq 0,\ C(x)\zeta \leq 1\}.$$

In addition, there are additional implicit constraints on ζ when q hits the boundaries of Q, to prevent the queue lengths from violating their positivity or finite buffer size constraints.

In complicated networks, the matrix B might not be square and the components of ζ correspond to different *activities* [92, chapter 6]. In the simple scheduling scenario described here however, $B(x)$ is a diagonal matrix for all x, with $B_{ii}(x) = -\mu_i(x)$. The stability condition presented in section 7.3 is valid for the general model (7.3) however.

Finally, note that models of the vehicle dynamics much more general than (7.1) could be used. In fact, the dynamics of the vehicle do not play a role to obtain our stability condition for the fluid model, except for the fact that (7.1) allows the control law to stop the vehicle at an arbitrary location of the state space X. Hence the stability condition presented below is valid for any vehicle that can stop at any $x \in \mathsf{X}$ (for example, a double integrator, or a Reeds-Shepp vehicle) but would have to be adapted to work for the Dubins vehicle for example.

7.3 Stability of the Fluid Model

In general, it is not possible to drain all queues to zero simultaneously using the mobile server, if the intersection of the draining regions, which are the regions of X where the server can drive the queue level of a particular queue towards 0, is empty. Hence we use the following weaker notion of stability.

Definition 7.3.1. The system q is said to be stabilizable if there exists constants K and T such that, for any initial condition q_0,

$$\|q(t)\|_\infty \leq K,\ \forall t \geq T\|q_0\|_\infty.$$

Remark 7.3.2. The requirement to bring the queue sizes close to 0 in a time proportional to $\|q_0\|$ is without loss of generality, as should be clear from the dynamics (7.3) and the geometric picture presented in the following.

To study the possible trajectories of the vector q in (7.3), we will study the velocity space for the queues, generalizing the corresponding definitions in [92]. Let

$$\mathsf{Y} = \bigcup_{x \in \mathsf{X}} \{\{x\} \times \mathsf{U}(x)\}.$$

First we note that a trajectory $(x(t), \zeta(t))$ defines for each $T \in \mathbb{R}_+$ a probability measure $P_T^{x,\zeta}$ on Y where $P_T^{x,\zeta}(A)$ is the proportion of time that the pair (x,ζ) spends in set $A \subset \mathsf{Y}$ in the interval $[0,T]$. In other words

$$P_T^{x,\zeta}(A) = \frac{1}{T}\int_0^T \delta_{x(t),\zeta(t)}(A)dt.$$

Then, integrating (7.3), we obtain, for all $T \in \mathbb{R}_+$,

$$\begin{aligned} q(T) &= q_0 + \alpha T + \int_0^T B(x(t))\zeta(t)dt \\ q(T) &= q_0 + \alpha T + \int_0^T \int_{\mathsf{Y}} B(x)\zeta\, \delta_{x(t),\zeta(t)}(dx,d\zeta)dt \\ q(T) &= q_0 + \alpha T + T\int_{\mathsf{Y}} B(x)\zeta\, P_T^{x,\zeta}(dx,d\zeta). \end{aligned} \tag{7.4}$$

Define the *velocity space for the queues* as

$$\mathsf{V} = \left\{ \alpha + \int_{\mathsf{Y}} B(x)\zeta\, P(dx,d\zeta) : \ P \text{ a probability measure on } \mathsf{Y} \right\}.$$

Clearly, V is a convex set in $\mathbb{R}^N$. It is the set of directions in which we can steer the vector q, by moving the server and controlling the allocation rates ζ, if we do not take into account the dynamical constraint (7.1) imposed on the vehicle dynamics. In fact, we will see below that this constraint does not play a role as far as stability is concerned.

Now the following is useful for approximating the set V. Note that by definition of the Lebesgue integral, we can approximate

$$\alpha + \int_{\mathsf{Y}} B(x)\zeta dP(x,\zeta),$$

arbitrarily closely by the "Lebesgue sums"

$$\alpha + \sum_{k=1}^{n} B(x^k)\zeta^k p^k, \tag{7.5}$$

where $\sum_{k=1}^n p^k = 1$ (since $P(\mathsf{Y}) = 1$), and $x^k \in \mathsf{X}$, and $\zeta^k \in \mathsf{U}(x^k), i = 1,\ldots,n$. Such a sum correspond to a point in V with associated measure $\sum_{k=1}^n p^k\, \delta_{x^k,\zeta^k}$. Let

$$\hat{\mathsf{V}}(x) = \{\alpha + B(x)\zeta : \zeta \in \mathsf{U}(x)\}.$$

For each position x of the server, $\hat{\mathsf{V}}(x)$ is a convex polyhedron. Note that since $0 \in \mathsf{U}(x)$ for all x, all

these polyhedrons contain the point α. Then we define

$$\hat{\mathsf{V}} = \mathbf{conv}\left\{\bigcup_{x\in\mathsf{X}} \hat{\mathsf{V}}(x)\right\}.$$

Clearly $\hat{\mathsf{V}}$ is the set of points of the form (7.5), and so $\hat{\mathsf{V}}$ is dense in V. Note that since $\hat{\mathsf{V}}$ and V are convex, this implies

$$\text{int}(\hat{\mathsf{V}}) = \text{int}(\mathsf{V}). \tag{7.6}$$

Let C^ε be the l^∞ ε-ball

$$C^\varepsilon = \{v \in \mathbb{R}^N : -\varepsilon < v_i \leq 0,\ i = 1,\ldots,N\}.$$

Then the following theorem provides a necessary and sufficient stabilizability condition for the fluid model.

Theorem 7.3.3. *The following conditions are equivalent*

1. *The fluid model (7.3) is stabilizable.*
2. $C^\varepsilon \subset \mathsf{V}$*, for some* ε*.*
3. $C^\varepsilon \subset \hat{\mathsf{V}}$*, for some* ε*.*

The proof of the theorem is in fact straightforward once the geometric picture is clear. It is thus helpful to consider first a simple low-dimensional example.

A Two-Dimensional Example

Consider a scheduling problem where the server can move on the real line and serve two queues at position 0 and d with service rates $\mu_1(x)\zeta_1$ and $\mu_2(x)\zeta_2$ respectively, with only the trivial constraints $0 \leq \zeta_1, \zeta_2 \leq 1$. Clearly in this case we should always let $\zeta_i = 1$, $i = 1,2$. The arrival rates are α_1 and α_2 respectively. Hence the dynamics are

$$\begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \end{bmatrix} = \begin{bmatrix} \alpha_1 \\ \alpha_2 \end{bmatrix} + \begin{bmatrix} -\mu_1(x) & 0 \\ 0 & -\mu_2(x) \end{bmatrix} \begin{bmatrix} \zeta_1 \\ \zeta_2 \end{bmatrix}, \tag{7.7}$$

$$\mathsf{U}(x) = \mathsf{U} = \left\{\zeta = [\zeta_1\ \zeta_2]^T : 0 \leq \zeta_1, \zeta_2 \leq 1\right\}\ \forall x, \tag{7.8}$$

and in addition, we have the positivity constraints $q_1, q_2 \geq 0$. Figure 7-3 shows the velocity set for an instance of the problem where

$$\alpha_1 = 1, \qquad \alpha_2 = 3, \qquad \mu_1(x) = \frac{3}{1+|x|}, \quad \mu_2(x) = \frac{4}{1+|x-4|}. \tag{7.9}$$

Let us define the *draining region* of queue i as

$$D^i := \{x \in \mathbb{R} \big| \alpha_i - \mu_i(x) < 0\}, \quad i \in \{1,2\},$$

and assume that $D^1 \cap D^2 = \emptyset$. This corresponds to the situation of fig. 7-3, since the curve $\{(\alpha_1 - \mu_1(x), \alpha_2 - \mu_2(x)) : x \in \mathbb{R}\}$ does not intersect the set $\text{int}(\mathbb{R}^2_-)$. The geometric condition for stability of theorem 7.3.3 can then be translated into a more algebraic condition. Indeed we see that the

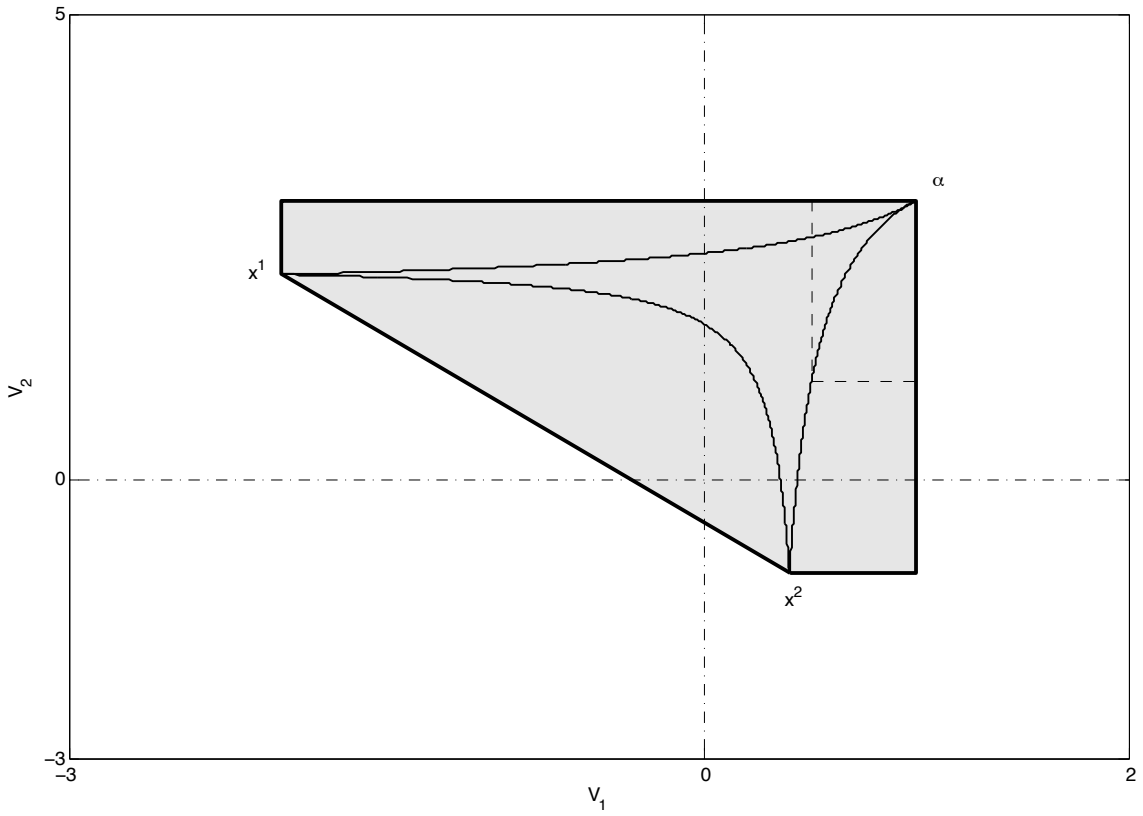


Figure 7-3: Velocity set for the two-dimensional example (7.9). The filled polygon is the velocity set for the queues. The curve is the set of points $\{(\alpha_1 - \mu_1(x), \alpha_2 - \mu_2(x)) : x \in \mathbb{R}\}$. We also show a rectangle which is the set $\mathsf{V}(x)$ for some value of x. From the figure and theorem 7.3.3, we see immediately that the system is stabilizable since $(0,0)$ is an interior point of the velocity set. Moreover, we can approximately steer the vector q in the directions contained in $\mathbb{R}^2_-$ using policies that switch between the points x^1 and x^2 only. The proportion of time spent at each point determines a convex combination of the velocity vectors at x^1 and x^2.

system is stable if and only if there exist two points $x^1 \in D^1$ and $x^2 \in D^2$ and a parameter $0 < \theta < 1$ such that

$$\theta \begin{pmatrix} \alpha_1 - \mu_1(x^1) \\ \alpha_2 - \mu_2(x^1) \end{pmatrix} + (1-\theta) \begin{pmatrix} \alpha_1 - \mu_1(x^2) \\ \alpha_2 - \mu_2(x^2) \end{pmatrix} < \begin{pmatrix} 0 \\ 0 \end{pmatrix}.$$

We rewrite these conditions as

$$\begin{cases} \alpha_1 - \mu_1(x^2) < \theta(\mu_1(x^1) - \mu_1(x^2)) \\ \alpha_2 - \mu_2(x^1) < (1-\theta)(\mu_2(x^2) - \mu_2(x^1)) \end{cases}$$

$$\text{we obtain: } \exists\, \theta \in (0,1) \text{ s.t. } \begin{cases} \frac{\alpha_1 - \mu_1(x^2)}{\mu_1(x^1) - \mu_1(x^2)} < \theta \\ \frac{\alpha_2 - \mu_2(x^1)}{\mu_2(x^2) - \mu_2(x^1)} < 1 - \theta. \end{cases}$$

These last two conditions are equivalent to the following condition

$$\exists x^1 \in D^1, x^2 \in D^2, \text{ s.t. } \frac{\alpha_1 - \mu_1(x^2)}{\mu_1(x^1) - \mu_1(x^2)} + \frac{\alpha_2 - \mu_2(x^1)}{\mu_2(x^2) - \mu_2(x^1)} < 1,$$

which, after expansion and simplifications, can also be rewritten

$$\exists x^1 \in D^1, x^2 \in D^2, \text{ s.t. } \frac{(\alpha_1 - \mu_1(x^2))(\alpha_2 - \mu_2(x^1))}{(\mu_1(x^1) - \alpha_1)(\mu_2(x^2) - \alpha_2)} < 1. \tag{7.10}$$

Example 7.3.4. Let

$$\mu_1(x) = \frac{1}{1+0.1x^2}, \quad \mu_2(x) = \frac{1}{1+0.1|x-8|^3}, \quad \alpha_1 = 0.5, \quad \alpha_2 = 0.55.$$

One can then verify that $\bar{D}_1 \cap \bar{D}_2 = \emptyset$. Condition (7.10) for $x^1 = 0$ and $x^2 = 8$ gives $0.86 < 1$ hence the system is stable. Looking only at the condition $\alpha_1/\mu_1(0) + \alpha_2/\mu_2(0) = 1.05$, we see that the system is easier to stabilize than the traditional discrete two-queue setup, for which $\mu_1(x^2) = \mu_2(x^1) = 0$.

Proof of theorem 7.3.3

$1 \Rightarrow 2$. We have by (7.4)

$$q_i(t) = q_i(0) + v_i^t\, t,\ i = 1, \ldots, N$$

for some $v^t \in \mathsf{V}$. Hence

$$\frac{q_i(t) - q_i(0)}{t} \in \mathsf{V}, \text{ for all } t > 0.$$

Recall the definition of the constants K, T in definition 7.3.1. Let $\delta \in \mathbb{R}_+^N$ with $\|\delta\|_\infty = 1/T$, and let $q^n(t)$ be a stable trajectory with $q^n(0) = n\delta, n \in \mathbb{N}$. Then for all $t \geq Tn\|\delta\|_\infty = n$, we have $\|q^n(t)\|_\infty \leq K$. In particular we deduce that for all $n \in \mathbb{N}$,

$$\frac{q^n(n)}{n} - \delta \in V, \tag{7.11}$$

and so since the sequence $q^n(n)$ is bounded by K, we see that $-\delta \in \mathrm{cl}(\mathsf{V})$. Hence we have the inclusion

$$\{v \in \mathbb{R}^N : -\varepsilon < v_i < 0,\ i = 1, \ldots, N\} \subset \mathsf{V}.$$

Moreover if in (7.11) we take $\delta = \frac{1}{T}e^i$, with e^i the i^{th} standard basis vector, we obtain a sequence $\{(\varepsilon_1^n, \ldots, -\frac{1}{T} + \varepsilon_i^n, \ldots, \varepsilon_N^n)\}_n$ of points of V with $\varepsilon_j^n \geq 0$, $\forall j, n$. Hence by convexity the points $-\frac{1}{T}e^i$ also belong to V. Finally recalling that $\alpha \in \mathsf{V}$ (with $\alpha \in \mathbb{R}_+^N$) and again by convexity of V, we obtain that $C^\varepsilon \subset \mathsf{V}$ for $\varepsilon = 1/T$.

$2 \Rightarrow 3$. Suppose $C^\varepsilon \subset \mathsf{V}$, for some ε. If $C^\varepsilon \setminus \{0\} \subset \text{int}(\mathsf{V})$, then $C^\varepsilon \subset \hat{\mathsf{V}}$ using (7.6), $\alpha \in \hat{\mathsf{V}}$ and the convexity of $\hat{\mathsf{V}}$ (we use this condition since it allows the case $\alpha = 0$ on the boundary of V). The rest of the argument concerns the technical point which arises when facets of C^ε are on the boundary of V. We prove that in this case, these facets are also in $\hat{\mathsf{V}}$. So assume that there is a point $v^0 \in C^\varepsilon \setminus \{0\}$ which is on the boundary of V. We call $\mathscr{P}$ the set of probability measures on Y. We also define, for $P \in \mathscr{P}$,

$$v(P) = \alpha + \int_{\mathsf{Y}} B(x)\zeta \; P(dx, d\zeta).$$

Then let $P^0 \in \mathscr{P}$ be such that $v(P^0) = v^0$.

Since V is convex, there is a supporting hyperplane $\mathsf{H} \subset \mathbb{R}^N$ of V passing through v_0. By a transformation of coordinates, we can assume that H is the plane $x_N = 0$, and that for all $P \in \mathscr{P}$, we have the coordinate $[v(P)]_N \leq 0$. Considering Dirac measures, we have that for all $(x, \zeta) \in \mathsf{Y}$, $(\alpha + B(x)\zeta)_N \leq 0$. In particular, taking $\zeta = 0$ implies that $\alpha_N = 0$, and so

$$\forall (x, \zeta) \in \mathsf{Y}, \; (B(x)\zeta)_N \leq 0. \tag{7.12}$$

Now we have

$$v_N^0 = \int_{\mathsf{Y}} (B(x)\zeta)_N \, P^0(dx, d\zeta) = 0,$$

which, with (7.12), implies that, except possibly for a set of Y of P^0 measure 0, we have $(B(x)\zeta)_N = 0$. Hence we can restrict the problem of approximating the integral defining v^0 by using points $B(x^k)\zeta^k$ which are on the hyperplane H. Repeating the same argument for an intersection of hyperplanes if there is a lower dimensional facet of C^ε (such as an edge) on the boundary of V, we see that we can always restrict the approximation problem for points v of the facet by using points $B(x^k)\zeta^k$ which are on that facet. Hence the facet also belongs to $\hat{\mathsf{V}}$ (by convexity of $\hat{\mathsf{V}}$ again). So $C^\varepsilon \subset \hat{\mathsf{V}}$.

$3 \Rightarrow 1$. The idea is intuitively the following. For T large enough, $-\frac{q_0}{T}$ belongs to $\hat{\mathsf{V}}$, so we can write

$$0 = q_0 + \alpha T + T \sum_{k=1}^{n} B(x^k)\zeta^k p^k.$$

Then a policy which moves the server successively through the points $x^k, k = 1, \ldots, n$, uses at these points the allocation rates ζ^k, and spends time $p^k T$ at point x^k drives the system approximately to 0 if T is much greater than the time spent traveling between the points x^k. Moreover, this policy is approximately time optimal, since on average it drives the vector of queues towards 0 along the direction $-q_0$. Of course, this reasoning is valid only for the case where $q_{0,i}$ is large for all i, for which the positivity constraints and the travel times can indeed be neglected. The argument is also complicated by the fact that the choice of the configurations (x^k, ζ^k), which determines the total travel time and the impact of the positivity constraints, depends on the travel direction $-q_0/T$ and it becomes problematic to obtain an upper bound K on the steady-state sizes of the queues which is independent of q_0 .

Hence we will show stability by driving the queues toward 0 using a single direction (the diagonal), sacrificing time optimality in the process. Let $q_0 = (q_{0,1}, \ldots, q_{0,N})$ be the initial queue sizes.

We start by bringing the queue size vector close to the diagonal $\mathsf{D} := \{q \in \mathbb{R}^N : q_1 = q_2 = \ldots = q_N\}$, as follows. For each $j \in \{1,\ldots,N\}$, there is a configuration $(x^j, \zeta^j) \in \mathsf{Y}$ such that $v^j = B(x^j)\zeta^j \in \hat{\mathsf{V}}$ verifies

$$v^j_j < 0, \text{ and } v^j_i \geq 0, i = 1,\ldots,N. \tag{7.13}$$

This is easily seen from the facts that $C^\varepsilon \subset \hat{\mathsf{V}}$ and $\hat{V}$ is convex. Moreover, we can choose these N vectors in such a way that C^ε is contained in the convex cone that they generate. Then there exist nonnegative constants t^j, $j = 1,\ldots,N$, such that $q_0 + \sum_{j=1}^N t^j v^j =: q_1 \in \mathsf{D}$, and moreover, if the server is spending time t^j in configuration v^j, $j = 1,\ldots,N$, the positivity constraints on q never become active. Now if we call T_1 the time necessary for switching from v^1 to v^2, then to v^3, etc., until v^N, we can fix $\zeta = 0$ while the server travels, and the policy above in fact brings the vector q to the value $q_1 + \alpha T_1$ on the line $\alpha T_1 + \mathsf{D}$, using a trajectory that never hits the boundaries of $\mathbb{R}^N_+$. This first phase can be accomplished in a time upper bounded by $\alpha T_1 + T_2\|q_0\|_\infty$, where T_2 is a constant independent of $\|q_0\|_\infty$ (this is because the distance from q_0 to D is proportional to $\|q_0\|_\infty$, and the coefficients t^j are of the same order).

Once the vector q is at $q_1 + \alpha T_1$, i.e., close to the diagonal, we drive it toward 0. More precisely, there is a constant $T_3 > 0$ such that

$$-\frac{q_1}{T_3} \in \hat{\mathsf{V}}, \text{ so } -q_1 = \alpha T_3 + T_3 \sum_{k=1}^n B(x^k)\zeta^k p^k,$$

for some configurations $(x^k, \zeta^k), k = 1,\ldots,n$ *that are independent of* q_1. Then by adapting the usual the policy that puts the server in the successive configurations $(x^k, \zeta^k), k = 1,\ldots,n$, we can move the queue size vectors by $-q_1 + \delta + \alpha T_4$. Here δ is a nonnegative fixed quantity which takes into account the effect of the positivity constraints, and T_4 is the sum of the switching times between configurations. So at the end of the second step, the vector q is at the point $\delta + \alpha(T_1 + T_4)$. The duration of the second phase is proportional to $\|q_1\|_\infty$, which can be bounded by a constant multiple of $\|q_0\|_\infty$. So the total time can be written $\tilde{T}\|q_0\|_\infty + \alpha T_1$. Using the ideas above, one can see that there are constants K and K_1, with $K \geq K_1 \geq \|\delta + \alpha(T_1 + T_4)\|$, such that if $\|q(t_0)\|_\infty \leq K_1$ for some t, we can insure $\|q(t)\|_\infty \leq K, \forall t \geq t_0$ (we can use the velocity vectors v^j of (7.13) for this purpose). Hence we are done by choosing $T = \tilde{T} + \frac{\alpha}{K_1} T_1$ in the case $\|q_0\|_\infty \geq K_1$.

Time Optimal Trajectories

We conclude this section with a brief and informal discussion of time-optimal trajectories, that is, trajectories which bring the queues in the bounded region of definition 7.3.1 in minimum time. Assume that we start with an initial condition q_0 which is far from the boundaries, see Fig. 7-4. Then time optimal trajectories correspond to driving the queues in the direction $-q_0$. This direction belongs to the velocity set if the system is stabilizable. However, it is possible that one can steer the queues in this direction only be traveling between different points in the space X. As long as we remain far from the boundaries of the state space Q, we can drive the queues approximately in the direction $-q_0$. Problems arise, and in particular we cannot necessarily neglect the traveling time any more, once we approach the boundaries.

7.4 Trajectory Optimization for a Draining Problem

If a system is stable in the sense of definition 7.3.1, then we can implement it using finite buffers. Moreover, the proof of theorem 7.3.3 showed how to bring the queue levels below the upper-bound

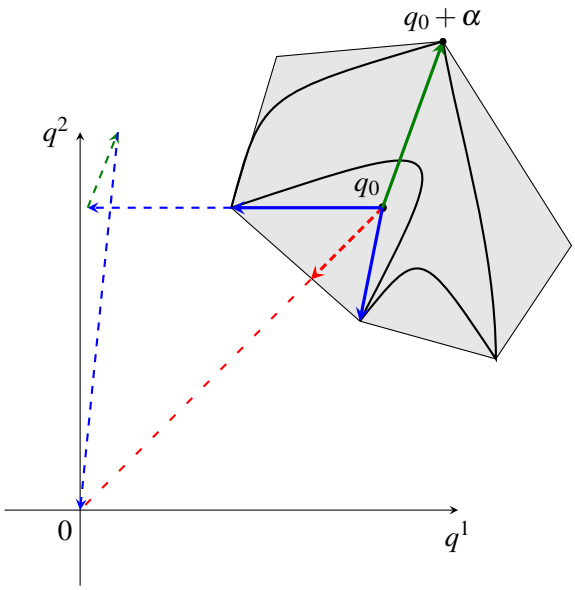


Figure 7-4: Initial condition q_0, velocity set (the gray polygon), and example of trajectory for the queues.

K approximately in minimum time, as least when starting from a large initial condition. The precise dynamics of the vehicle had no influence on the stability condition, but only on the values of K and T.

There are other types of objectives one might want to optimize. If we want to give priority to certain types of jobs at certain locations for example, minimizing a weighted sum of the queue levels might be appropriate. In this section, we touch upon this problem, for which the dynamics of the vehicle play a more important role.

To illustrate the ideas, we consider a simple particular case of the two-queue configuration (7.7), (7.8). We study the problem of *draining optimally the fluid present initially in the system*, when no new arrival occurs (i.e., $\alpha_1 = \alpha_2 = 0$) and we incur a running linear cost $c_1 q_1(t) + c_2 q_2(t)$, with c_1, c_2 strictly positive constants.

7.4.1 Server with Unbounded Velocity

If the server were allowed to travel with infinite velocity, there would be a path-wise optimal solution. Indeed, suppose that additionally to the points x_i^* attaining the maximum service rates μ_i^*, there is a point x^* realizing the maximum

$$x^* \in \arg\max_x \{c_1 \mu_1(x) + c_2 \mu_2(x)\}.$$

Then, since we have

$$\frac{d^+}{dt} c(q(t)) = -c_1 \left[\mu_1(x(t))\right]_{q_1}^+ - c_2 \left[\mu_2(x(t))\right]_{q_2}^+,$$

we obtain immediately

Theorem 7.4.1. *For the draining problem with linear cost function and a server with infinite velocity, the following rule is path-wise optimal, i.e., for given initial fluid levels, it minimizes $c(t)$ for all $t \geq 0$:*

1. *as long as $q_1 > 0$ and $q_2 > 0$, place the server at x^**

2. *if $q_1 = 0$, place the server at x_2^*, otherwise if $q_2 = 0$, place the server at x_1^* (until the total fluid reaches 0).*

Now define the infinite-horizon cost

$$J(q_0;u(\cdot)) = \int_0^\infty c(q(t))dt = \int_0^\infty c_1 q_1(t) + c_2 q_2(t)dt, \tag{7.14}$$

and the draining time

$$T(q_0;u(\cdot)) = \min\{t : q(t) = 0\},$$

where $q_0 = [q_{10}, q_{20}]^T$ is the initial fluid level in the queues. A policy minimizing $J(q_0;\cdot)$ is called infinite-horizon optimal, and it is called time optimal if it drains the system in minimum time. We have then

Corollary 7.4.1. *For a server with infinite velocity, the policy of theorem 7.4.1, being path-wise optimal, is also infinite-horizon optimal and time optimal.*

Remark 7.4.2. If the draining regions do not overlap, the policy of theorem 7.4.1 chooses the position x^* associated to the maximum value of the indices $c_i\mu_i^*, i = 1,2$. Hence, we recover the well-known "$c\mu$ rule" [92].

7.4.2 Necessary Conditions for Optimality

We now consider the optimization of the infinite-horizon cost (7.14) for a server with the simple dynamics (7.1) and bounded velocity. Using the minimum principle, we can derive necessary conditions satisfied by optimal trajectories. For each subset S of $\{1,2\}$, define the Hamiltonian

$$H^S(q,x,u,p) = \sum_{i\in S} c_i q_i - \sum_{i\in S} p_i\mu_i(x) + p_0 u, \tag{7.15}$$

which is the Hamiltonian for the system where S represents the set of non-empty queues, and $p = [p_0, p_1, p_2]$ are the adjoint variables. In contrast to standard optimal control problems for fluid models, where the draining rate is assumed to be under direct control, we cannot enforce the positivity constraint on the fluid levels by a simple state dependent constraint on the control. We must take into account the discontinuity in the dynamics of the system when a queue level reaches zero.

Suppose that both queues are non-empty, that the server is in the draining region D^i of queue i, and that queue i becomes empty at some time τ_i. Suppose that there is only one queue becoming empty at time τ_i. From the results in [26, sections 3.5 and 3.6], at the boundary $q_i = 0$, we have the following relations

$$p_i(\tau_i^-) = p_i(\tau_i^+) + \pi_i, \tag{7.16}$$
$$p_j(\tau_i^-) = p_j(\tau_i^+), \quad j \neq i$$
$$H^{\{1,2\}}(\tau_i^-) = H^{\{j\}}(\tau_i^+), \quad j \neq i \tag{7.17}$$

where π_i is a (constant) Lagrange multiplier. Now using the continuity of the state variables q, x and of the Hamiltonian (7.17), we get

$$p_i(\tau_i^-)\mu_i(x(\tau_i^-)) = 0,$$

and since the server must be in the draining region D^i (where $\mu_i(x(\tau_i^-)) > 0$) when queue i becomes empty this implies

$$p_i(\tau_i^-) = 0.$$

Hence, the adjoint variable corresponding to queue i vanishes at the time when the queue becomes empty. The optimal control is of the bang-bang type

$$u(t) = -\text{sign}(p_0(t))V.$$

When the queues in the set S are non-empty, the adjoint equations are,

$$\dot{p}_i = -c_i, \quad i \in S$$
$$\dot{p}_0 = \sum_{i \in S} p_i \frac{d\mu_i}{dx}(x(t)),$$

where the derivative with respect to x can be taken to mean a subgradient in the case of nonsmooth rate functions. From the first set of equations and the preceding argument, we deduce

$$p_i(t) = c_i(\tau_i - t), \quad i \in \{1,2\},$$

Finally, since the dynamics and cost function are independent of time and this is a free terminal time problem, we know that the Hamiltonian is constant and equal to zero along optimal trajectories.

7.4.3 Draining Problem with Two Distant Sites and Linear Rate Functions

In this section, we consider an example where the server starts initially on the line segment between two base stations located at positions 0 and d. We assume for simplicity that the stations are sufficiently far from the initial position of the server, so that the queue at each base station would become empty before the server can reach it, if it were to always move towards that base station. The objective is to drain the queues while minimizing the infinite-horizon cost function (7.14). We assume that the draining region of each queue covers the whole segment $[0,d]$, so that even a immobile server is draining the queue in finite time. We also assume here that the service rate decreases linearly with the distance to the base station, i.e., $\mu_1(x) = \mu_{10} - s_1|x|$ and $\mu_2(x) = \mu_{20} - s_2|d-x|$, with s_1 and s_2 positive constants. In particular, in the region $[0,d]$ of interest, we have

$$\mu_1(x) = \mu_{10} - s_1 x, \quad \mu_2(x) = \mu_{20} - s_2(d-x), \quad x \in [0,d].$$

Calling the terminal time t_f, we have $t_f = \tau_1$ or $t_f = \tau_2$ and also

$$H(t_f) = -V|p_0(t_f)| = 0 \Rightarrow p_0(t_f) = 0. \tag{7.18}$$

We determine the optimal trajectories by considering different cases based on which queue becomes empty first.

Case $\tau_2 < \tau_1$

Suppose that queue 2 at location d empties first, that is, $\tau_2 < \tau_1 = t_f$. Then for $t > \tau_2$, we have, for x in the region $[0,d]$,

$$\dot{p}_0 = -p_1 s_1 = c_1 s_1 (t - \tau_1).$$

This gives, with the condition (7.18),

$$p_0 = \frac{c_1 s_1}{2}(\tau_1 - t)^2, \text{ for } t \in [\tau_2, \tau_1].$$

Hence p_0 is strictly positive on (τ_2, τ_1) and and so during that phase, we have $u = -V$, i.e., the server moves towards the left base station, as expected.

Now consider the time τ_2, at which we have $p_0(\tau_2) > 0$ and p_0 is continuous. On the interval $[0, \tau_2)$, we have

$$\dot{p}_0 = -p_1 s_1 + p_2 s_2 = -c_1 s_1 (\tau_1 - t) + c_2 s_2 (\tau_2 - t).$$

This equation, together with the continuity condition at τ_2, gives

$$p_0 = \frac{c_1 s_1}{2}(\tau_1 - t)^2 - \frac{c_2 s_2}{2}(\tau_2 - t)^2, \text{ for } t \in [0, \tau_2].$$

If $c_1 s_1 = c_2 s_2$, then $p_0 = c_1 s_1 (\tau_1 - \tau_2)\left(\frac{\tau_1 + \tau_2}{2} - t\right)$ and so p_0 remains positive and the server must always move towards the left base station. This is also true if $c_1 s_1 > c_2 s_2$. Now in the case $c_2 s_2 > c_1 s_1$, there is potentially a time $t_0 > 0$ such that on $[0, t_0)$, $p_0(t)$ is negative and the server must move towards the right base station. We obtain t_0 by solving for $p_0(t_0) = 0$, so that, defining

$$R := \sqrt{\frac{c_1 s_1}{c_2 s_2}} = \frac{\tau_2 - t_0}{\tau_1 - t_0},$$

we get

$$t_0 = \max\left\{\frac{\tau_2 - \tau_1 R}{1 - R}, 0\right\}, \tag{7.19}$$

which is positive if and only if $\tau_2 > \tau_1 R$. In conclusion, for an optimal trajectory to empty queue 2 first, the server must necessarily be always moving to the left if $R \geq 1$, and if $R < 1$ it must be moving right for time t_0 and then left.

Consider the case $R \geq 1$ first. If the server always moves left, we have

$$x(t) = x_0 - Vt.$$

This gives (assuming that the server does not reach station 1 before t_f):

$$\begin{aligned}
\dot{q}_1(t) &= -\mu_{10} + s_1 x(t) = -\mu_{10} + s_1 x_0 - s_1 V t, \\
q_1(t) &= q_{1,0} + (s_1 x_0 - \mu_{10})t - s_1 V \frac{t^2}{2} \\
\dot{q}_2(t) &= -\mu_{20} + s_2(d - x(t)) = -\mu_{20} + s_2(d - x_0) + s_2 V t \\
q_2(t) &= q_{2,0} + (s_2(d - x_0) - \mu_{20})t + s_2 V \frac{t^2}{2}.
\end{aligned}$$

Letting $q_1(\tau_1) = 0$ and $q_2(\tau_2) = 0$, we get

$$\tau_1 := \tau_{1,l} = \frac{(s_1 x_0 - \mu_{10}) + \sqrt{(s_1 x_0 - \mu_{10})^2 + 2 s_1 V q_{1,0}}}{s_1 V} \tag{7.20}$$

$$\tau_2 := \tau_{2,l} = \frac{-(s_2(d - x_0) - \mu_{20}) - \sqrt{(s_2(d - x_0) - \mu_{20})^2 - 2 s_2 V q_{2,0}}}{s_2 V}, \tag{7.21}$$

recalling that $(s_1 x_0 - \mu_{10})$ and $(s_2(d - x_0) - \mu_{20})$ are negative quantities by assumption. The trajec-

tory for the queue level $q_2(t)$ is a convex function of t and the time τ_2 corresponds to the smallest root of the equation $q_2(t) = 0$. With these quantities and $R \geq 1$, for the optimal trajectory to empty queue 2 first, we must have $\tau_{2,l} < \tau_{1,l}$. This gives a condition that depends only on the parameters of the problems. If this condition is not verified and $R \geq 1$, then queue 2 cannot empty strictly before queue 1.

Now suppose $R < 1$. As long as $\tau_2 \leq R\tau_1$, the optimal solution must again move the server always to the left. So for such an extremal solution to exist, we must have $\tau_{2,l} \leq R\tau_{1,l}$. If we find that $\tau_{2,l} > R\tau_{1,l}$, always moving the server to the left cannot be optimal, but we might still have an optimal solution which empties queue 2 first by first moving to the right. Let $\tau_i(t_r)$ be the time at which queue i empties if the server first moves right for a time t_r, and then left. Thus, we are considering the case $R < 1, R\tau_1(0) < \tau_2(0)$. As we increase t_r, $\tau_2(t_r)$ decreases and $\tau_1(t_r)$ increases. The positive time t_0 verifies the equation

$$(1-R)t_0 = \tau_2(t_0) - R\tau_1(t_0), \quad \text{for } t_0 > 0 \tag{7.22}$$

so we have reduced the problem to computing the functions $\tau_i(t_r)$ and solving the fixed point equation (7.22), i.e., finding the intersection of the diagonal with $(\tau_2(t_r) - R\tau_1(t_r))/(1-R)$, a decreasing function of t_r. Once we have obtained t_0, we can verify if the optimal solution indeed empties queue 2 first by verifying $\tau_2(t_0) < \tau_1(t_0)$. If not, again, the optimal solution cannot empty queue 2 strictly before queue 1. Note also that when τ_2 approaches τ_1 from below, i.e., both queues empty at the same time, we get $t_0 \to \tau_1 = t_f$, and so in the limit we should move the server to the right all the time.

Now when the server moves to the right first for time t_r, we have $x = x_0 + Vt$, and we get

$$q_1(t_r) = \max\left\{0, q_{1,0} + (s_1 x_0 - \mu_{10})t_r + s_1 V \frac{t_r^2}{2}\right\}, \tag{7.23}$$

$$q_2(t_r) = \max\left\{0, q_{2,0} + (s_2(d - x_0) - \mu_{20})t_r - s_2 V \frac{t_r^2}{2}\right\}. \tag{7.24}$$

Note that in fact, from the argument above, for $\tau_2 < \tau_1$, t_r must be such that $q_1(t_r)$ and $q_2(t_r)$ are both positive, since when queue 2 empties at τ_2, we must be traveling left and have passed the time t_r. This constrains the set of possible values to consider for t_r. Then we can replace $q_{i,0}$ by $q_i(t_r)$ and x_0 by $x_0 + Vt_r$ in (7.20), (7.21), to obtain

$$\tau_1(t_r) = t_r + \frac{(s_1(x_0 + Vt_r) - \mu_{10})}{s_1 V} + \frac{\sqrt{(s_1(x_0 + Vt_r) - \mu_{10})^2 + 2s_1 V\left(q_{1,0} + (s_1 x_0 - \mu_{10})t_r + s_1 V \frac{t_r^2}{2}\right)}}{s_1 V}$$

$$\tau_2(t_r) = t_r + \frac{-(s_2(d - x_0 - Vt_r) - \mu_{20})}{s_2 V} - \frac{\sqrt{(s_2(d - x_0 - Vt_r) - \mu_{20})^2 - 2s_2 V\left(q_{2,0} + (s_2(d - x_0) - \mu_{20})t_r - s_2 V \frac{t_r^2}{2}\right)}}{s_2 V}$$

From these expression we can compute t_0 by solving the fixed point equation. It reduces to finding

t_r such that

$$\begin{aligned}
&(\mu_{20} - s_2(d - x_0 - Vt_r)) \\
&- \sqrt{(s_2(d - x_0 - Vt_r) - \mu_{20})^2 - 2s_2V\left(q_{2,0} + (s_2(d - x_0) - \mu_{20})t_r - s_2V\frac{t_r^2}{2}\right)} \\
&= \left(R\frac{s_2}{s_1}\right)\Bigg\{(s_1(x_0 + Vt_r) - \mu_{10}) \\
&+ \sqrt{(s_1(x_0 + Vt_r) - \mu_{10})^2 + 2s_1V\left(q_{1,0} + (s_1x_0 - \mu_{10})t_r + s_1V\frac{t_r^2}{2}\right)}\Bigg\}.
\end{aligned}$$

In practice, the feedback solution, discussed later, is more useful than the open-loop control and does not require finding the value of t_0. However, t_0 could be useful for computing performance bounds, or t design an open-loop policy if we don't have access to the updates on the sizes of the queues or the position of the server.

Case $\tau_1 < \tau_2$

The analysis is identical to the previous case, interchanging the indices in the first part, and V with $-V$. Let us repeat the details for completeness. For $t > \tau_1$, we have, for x in the region $[0,d]$,

$$\dot{p}_0 = p_2s_2 = -c_2s_2(t - \tau_2).$$

This gives, with the condition (7.18),

$$p_0 = -\frac{c_2s_2}{2}(\tau_2 - t)^2, \text{ for } t \in [\tau_2, \tau_1].$$

Hence p_0 is strictly negative on (τ_2, τ_1) and and so during that phase, we have $u = +V$, i.e., the server moves towards the right base station, as expected.

Now consider the time τ_1, at which we have $p_0(\tau_1) < 0$ and p_0 is continuous. On the interval $[0, \tau_1)$, we have

$$\dot{p}_0 = -p_1s_1 + p_2s_2 = -c_1s_1(\tau_1 - t) + c_2s_2(\tau_2 - t).$$

This equation, together with the continuity condition at τ_2, gives again

$$p_0 = \frac{c_1s_1}{2}(\tau_1 - t)^2 - \frac{c_2s_2}{2}(\tau_2 - t)^2, \text{ for } t \in [0, \tau_2].$$

If $c_1s_1 = c_2s_2$, then $p_0 = c_1s_1(\tau_1 - \tau_2)\left(\frac{\tau_1+\tau_2}{2} - t\right)$ and so p_0 remains negative and the server must always move towards the right base station. This is also true if $c_1s_1 < c_2s_2$. Now in the case $c_1s_1 > c_2s_2$, there is potentially a time $t_0 > 0$ such that on $[0, t_0)$, $p_0(t)$ is positive and the server must move towards the left base station. We obtain t_0 by solving for $p_0(t_0) = 0$, and defining like before

$$R := \sqrt{\frac{c_1s_1}{c_2s_2}} = \frac{\tau_2 - t_0}{\tau_1 - t_0},$$

we get

$$t_0 = \max\left\{\frac{\tau_1R - \tau_2}{R - 1}, 0\right\}, \tag{7.25}$$

which is positive if and only if $\tau_2 < \tau_1 R$, since we now have $R > 1$. In conclusion, for an optimal trajectory to empty queue 1 first, the server must necessarily be always moving to the right if $R \leq 1$, and if $R > 1$ it must be moving left for time t_0 and then right.

Consider the case $R \leq 1$ first. If the server always moves right, we have

$$x(t) = x_0 + Vt.$$

This gives (assuming that the server does not reach station 2 before t_f):

$$\begin{aligned}
\dot{q}_1(t) &= -\mu_{10} + s_1 x(t) = -\mu_{10} + s_1 x_0 + s_1 V t, \\
q_1(t) &= q_{1,0} + (s_1 x_0 - \mu_{10})t + s_1 V \frac{t^2}{2} \\
\dot{q}_2(t) &= -\mu_{20} + s_2(d - x(t)) = -\mu_{20} + s_2(d - x_0) - s_2 V t \\
q_2(t) &= q_{2,0} + (s_2(d - x_0) - \mu_{20})t - s_2 V \frac{t^2}{2}.
\end{aligned}$$

Letting $q_1(\tau_1) = 0$ and $q_2(\tau_2) = 0$, we get

$$\tau_1 := \tau_{1,r} = \frac{(\mu_{10} - s_1 x_0) - \sqrt{(s_1 x_0 - \mu_{10})^2 - 2 s_1 V q_{1,0}}}{s_1 V} \tag{7.26}$$

$$\tau_2 := \tau_{2,r} = \frac{(s_2(d - x_0) - \mu_{20}) + \sqrt{(s_2(d - x_0) - \mu_{20})^2 + 2 s_2 V q_{2,0}}}{s_2 V}, \tag{7.27}$$

recalling that $(s_1 x_0 - \mu_{10})$ and $(s_2(d - x_0) - \mu_{20})$ are negative quantities by assumption. The trajectory for the queue level $q_1(t)$ is a convex function of t and the time τ_1 corresponds to the smallest root of the equation $q_1(t) = 0$. With these quantities and $R \leq 1$, for the optimal trajectory to empty queue 1 first, we must have $\tau_{1,r} < \tau_{2,r}$. This gives a condition that depends only on the parameters of the problems. If this condition is not verified and $R \leq 1$, then queue 1 cannot empty strictly before queue 2.

Now suppose $R > 1$. As long as $\tau_2 \geq R\tau_1$, the optimal solution must again move the server always to the right. So for such an extremal solution to exist, we must have $\tau_{2,r} \geq R\tau_{1,r}$. If we find that $\tau_{2,r} < R\tau_{1,r}$, always moving the server to the right cannot be optimal, but we might still have an optimal solution which empties queue 1 first by first moving to the left. Let $\tau_i(t_l)$ be the time at which queue i empties if the server first moves left for a time t_l, and then right. Thus, we are considering the case $R > 1$, $R\tau_1(0) > \tau_2(0)$. As we increase t_l, $\tau_1(t_l)$ decreases and $\tau_2(t_l)$ increases. The positive time t_0 verifies the equation

$$(R-1)t_0 = R\tau_1(t_0) - \tau_2(t_0), \quad \text{for } t_0 > 0 \tag{7.28}$$

so we have reduced the problem to computing the functions $\tau_i(t_l)$ and solving the fixed point equation (7.28), i.e., finding the intersection of the diagonal with $(R\tau_1(t_l) - \tau_2(t_l))/(R-1)$, a decreasing function of t_l. Once we have obtained t_0, we can verify if the optimal solution indeed empties queue 1 first by verifying $\tau_1(t_0) < \tau_2(t_0)$. If not, again, the optimal solution cannot empty queue 1 strictly before queue 1. Note also that when τ_1 approaches τ_2 from below, i.e., both queues empty at the same time, we get $t_0 \to \tau_1 = t_f$, and so in the limit we should move the server to the left all the time.

Case $\tau_1 = \tau_2$

In this case, $\tau_1 = \tau_2 - t_f$, and we also have the terminal condition $p_i(t_f) = 0$, $i = 0,1,2$, now simply because it is a free terminal time problem with no terminal cost. Hence we get

$$p_0 = \frac{1}{2}(c_1 s_1 - c_2 s_2)(t_f - t)^2,$$

quantity which is always of the same sign as $R-1$. Hence if $R > 1$, i.e. $c_1 s_1 > c_2 s_2$, an optimal trajectory emptying both queues simultaneously must necessarily always move the server left, whereas if $R < 1$, i.e., $c_1 s_1 < c_2 s_2$, it always moves the server right. Hence in the case $R > 1$, an extremal solution emptying both queues simultaneously is possible only if $\tau_{1,l} = \tau_{2,l}$. If $R < 1$, such an extremal solution is possible only if $\tau_{1,r} = \tau_{2,r}$.

Finally if $R = 1$, i.e., $c_1 s_1 = c_2 s_2$, we get a singular arc ($p_0(t) = 0$, for all t) and higher order conditions are necessary to determine extremal controls. This case is not treated in the following.

Feedback Form Solution

To obtain a feedback control, we simply replace in the previous expressions the quantities x_0, and q_0 by the current state of the system x, q. We still use the same notations for $\tau_{i,r}, \tau_{i,l}$, understood now to be functions of the state x, q. Now for $R < 1$, in a given state, we can compute $\tau_{1,r}(x,q)$ and $\tau_{2,r}(x,q)$. If $\tau_{1,r} > \tau_{2,r}$, we see from the previous paragraphs that the trajectory will empty queue 2 first necessarily. We compute the quantities $\tau_{1,l}$ and $\tau_{2,l}$ for the current state. If $\tau_{2,l} > R\tau_{1,l}$, we have seen that the solution starts by moving the server to the right. Hence the feedback solution commands to move right in the current state. Otherwise, we move left.

If $\tau_{1,r} \leq \tau_{2,r}$, an extremal solution can potentially empty queue 1 before or at the same time as queue 2 by moving the server to the right. Moreover it is clear that this inequality implies $\tau_{1,l} < \tau_{2,l}$ since we obviously have $\tau_{1,l} < \tau_{1,r}$ and $\tau_{2,r} < \tau_{2,l}$. Hence, in particular, $\tau_{2,l} > R\tau_{1,l}$ and a trajectory emptying queue 2 first also necessarily moves right initially. Hence the feedback solution prescribes to move right if $\tau_{1,r} \leq \tau_{2,r}$.

Now consider the case $R > 1$, which is symmetric. In a given state, we first compute $\tau_{1,l}$ and $\tau_{2,l}$. If $\tau_{2,l} > \tau_{1,l}$, we see from the previous paragraphs that the trajectory will empty queue 1 first necessarily. We compute the quantities $\tau_{1,r}$ and $\tau_{2,r}$ for the current state. If $\tau_{2,r} < R\tau_{1,r}$, we have seen that the solution starts by moving the server to the left. Hence the feedback solution commands to move left in the current state. Otherwise, we move right.

If $\tau_{2,l} \leq \tau_{1,l}$, an extremal solution can potentially empty queue 2 before or at the same time as queue 1 by moving the server to the left. Moreover it is clear that this inequality implies $\tau_{2,r} < \tau_{1,r}$ since again we have $\tau_{1,l} < \tau_{1,r}$ and $\tau_{2,r} < \tau_{2,l}$. Hence in particular $\tau_{2,r} < R\tau_{1,r}$ and a trajectory emptying queue 1 first also necessarily moves left initially. Hence the feedback solution prescribes to move left if $\tau_{2,l} \leq \tau_{1,l}$.

Finally in the case $R = 1$, in addition to regular extremals similar to the ones obtained for $R \neq 1$, there are singular arcs emptying both queues simultaneously while leaving the server immobile. We do not provide a full solution for this case, which however can be avoided by a small perturbation of the parameters. Table 7.1 summarizes the optimal feedback control law for the draining problem of this section.

$R<1$	$\tau_{1,r} \le \tau_{2,r}$	$u=+V$
	$\tau_{1,r} > \tau_{2,r} \wedge \tau_{2,l} > R\tau_{1,l}$	$u=+V$
	$\tau_{1,r} > \tau_{2,r} \wedge \tau_{2,l} \le R\tau_{1,l}$	$u=-V$
$R>1$	$\tau_{2,l} \le \tau_{1,l}$	$u=-V$
	$\tau_{2,l} > \tau_{1,l} \wedge \tau_{2,r} < R\tau_{1,r}$	$u=-V$
	$\tau_{2,l} > \tau_{1,l} \wedge \tau_{2,r} \ge R\tau_{1,r}$	$u=+V$

Table 7.1: Optimal feedback control law.

7.5 Simulation

Suppose that when we sample the system dynamics with sampling period T_s, the system dynamics follow in fact the following discrete time stochastic evolution

$$\begin{aligned} Q_i((k+1)T_s) &:= Q_i(k+1) = Q_i(k) + A_i(k+1) - B_i(k+1; X(k)),\ i=1,2. \\ X((k+1)T_s) &:= X(k+1) = X(k) + U(k)\,T_s, \quad |U(k)| \le V, \end{aligned}$$

where the queue lengths Q_i are also subject to the nonnegativity constraints. Assume that the arrival process $\{A(k)\}$ is i.i.d. with $E[A_i(k+1)] = \alpha_i T_s$, and the variables $B_i(k+1, x(k))$ are independent, with distribution depending on the position $x(k)$ of the server, and $E[B_i(k+1; X(k))] = \mu_i(X(k))T_s$. Then, following the proof in [92, section 4.4], we can prove, if $\mu(x)$ is sufficiently smooth, the following lower bound on the achievable performance for the stochastic system, for each fixed $T \in T_s\mathbb{Z}$:

$$\inf E\left[\sum_{k=0}^{T/T_s} T_s c(Q(k); Q(0)=q_0)\right] \ge \inf \int_0^T c(q(t); q(0)=q_0)dt + O(T_s),$$

where the inf is over all admissible control policies, and $\lim_{T_s\to 0} O(T_s) = 0$. For the sampled system, admissible means that we consider control signals constant over the sampling period T_s.

Proof. Take for the continuous fluid system the sampled signal

$$u(t) = u(kT_s) = U(k), \quad kT_s \le t < (k+1)T_s, \forall k.$$

so that we have a server trajectory for the fluid system verifying

$$\begin{aligned} x(t) &= x(kT_s) + (t - kT_s)U(k) \\ &= X(k) + (t-kT_s)U(k), \quad kT_s \le t < (k+1)T_s, \forall k. \end{aligned}$$

Then

$$\begin{aligned} E[Q(k+1)] &= E[Q(k)] + \alpha T_s - \mu(X(k))T_s \\ &= q_0 + \alpha(k+1)T_s - \sum_{i=0}^{k} \mu(X(k))T_s. \end{aligned}$$

The last term is a Riemann sum with left approximation, so (by Taylor expansion)

$$\begin{aligned} E[Q(k+1)] &= q_0 + \alpha(k+1)T_s - \int_0^{(k+1)T_s} \mu(x(t))dt + O((k+1)T_s^2) \\ &= q((k+1)T_s) + O((k+1)T_s^2). \end{aligned}$$

Now we have, for $T = kT_s$, approximating the integral using the trapezoidal rule, and using the linearity of c:

$$\begin{aligned} \int_0^T c_1 q_1(t) + c_2 q_2(t) dt &= T_s \sum_{i=0}^{k-1} \frac{1}{2}[c(q(iT_s)) + c(q((i+1)T_s))] + T^3 \times O(T_s^2) \\ &= T_s \sum_{i=0}^{k-1} \frac{1}{2}\{E[c(Q(iT_s))] + E[c(Q((i+1)T_s))]\} + O(T_s) \\ &= T_s \sum_{i=0}^{k} E[c(Q(iT_s))] - \frac{T_s}{2} c(q_0) - \frac{T_s}{2} E[c(Q(k))] + O(T_s). \end{aligned}$$

The conclusion follows by taking inf first on the fluid model policies, then on the stochastic model policies.

□

Moreover, as the size of the jumps $A(k), B(k)$ becomes small, while keeping the same rates $\alpha_i, \mu_i(x))$, we expect the behavior of the stochastic system to approach the behavior of the fluid system (see, e.g., [87]). Fig. 7-5 gives an example of queue trajectories obtained for a stochastic system and for its fluid approximation, using the optimal feedback policy computed for the fluid model in the previous section.

7.6 Conclusion

We have presented here joint trajectory optimization and scheduling problems for a server serving remotely the jobs arriving at spatially separated queues. The approach using fluid models simplifies some of the computational difficulties we faced in the previous chapter using the more detailed controlled Markov chain approach. The relationship between fluid models and discrete stochastic models are the same as for standard stochastic networks. In particular, stability of the fluid model as in section 7.3 should imply stability of the controlled random walk model. The precise derivation of this result is left for future work. The draining problem has also shown that a feedback solution for the fluid model can provide a useful policy for a stochastic model.

There are numerous possible research directions and extensions, such as completing the trajectory optimization problem for the average and discounted cost criterion when arrivals occur, considering the optimization problem with more queues and more complicated vehicle dynamics, treating models with several vehicles possibly sharing their resources, etc. Fluid models could potentially offer a powerful modeling tool to obtain insight into complicated spatiotemporal task assignment problems, such as the ones arising in UAS control.

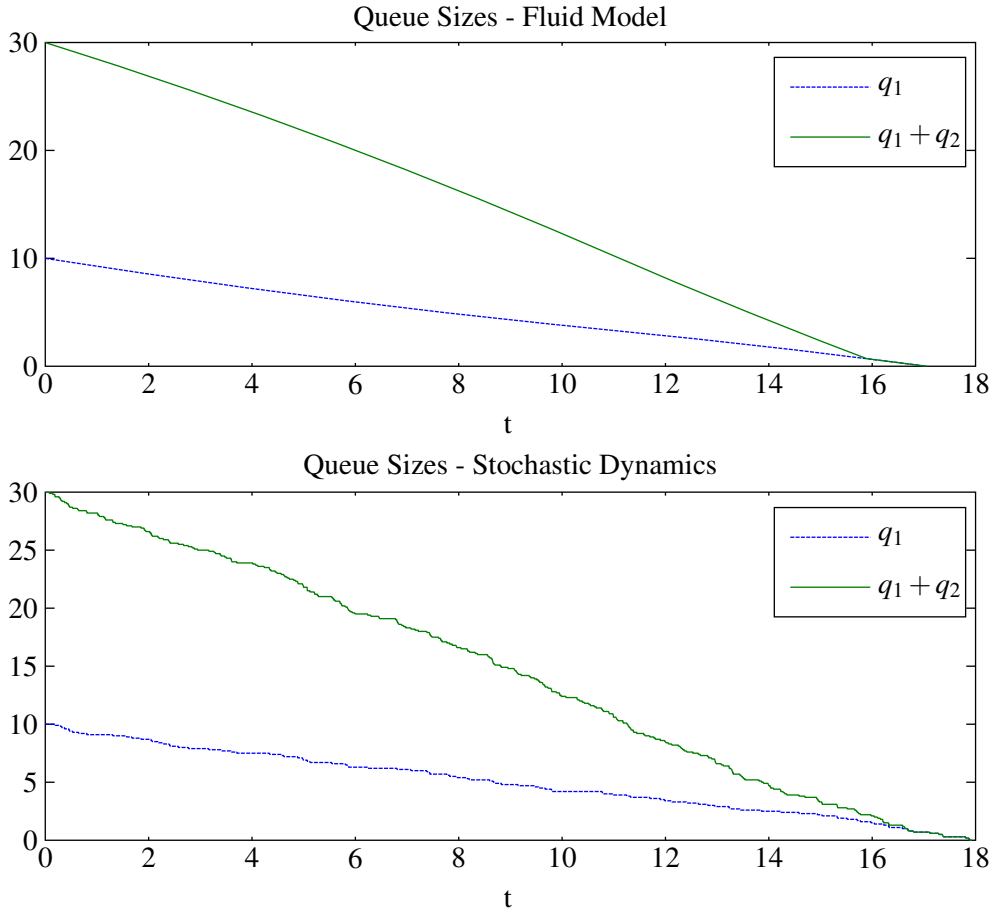


Figure 7-5: Examples of trajectories of the queues for the stochastic draining problem with Bernoulli service variables, and for its deterministic fluid approximation (top). The optimal feedback policy of the fluid model was used in both cases to control the server.

Chapter 8

Conclusion and Future Directions

8.1 Summary

In this thesis we present optimization-based approaches to some high-level problems arising in the control of UVS. Typical problems in this context involve scheduling tasks dynamically while simultaneously solving difficult motion planning problems. This thesis proposes ways of solving these problems while mitigating the computational challenges involved.

In a static environment, the typical problem that the UVS operator faces is to design trajectories for the vehicles through a set of waypoints, which is a variant of the TSP. Chapter 2 proposes new algorithms for the TSP for the Dubins vehicle (DTSP). An operator using these algorithms for UAVs can expect improved performance over the standard practice of solving the Euclidean TSP, in particular when the vehicles are operating at high speed in a restricted space.

In dynamic environments, scheduling problems, that is, deciding where each vehicle should be and when, are superimposed to the path planning problem. In fact, these scheduling problems are already typically intractable, and chapters 4 and 5 first considers them independently of the path planning problem. In particular, we provide new closed form formulas for the Whittle index in a discrete event detection problem as well as a one-dimensional continuous filtering problem, both with multiple sensors and multiple targets. We derive a bound on the achievable performance that allows us to verify the high performance of the Whittle heuristic for these problems. In the multidimensional case of the filtering problem, we show that this bound can be computed by a tractable convex program involving linear matrix inequalities. Finally, we demonstrate that the performance of a new family of periodic switching policies approaches the bound arbitrarily closely.

In the final part of the thesis, we return to our original objective of combining the path planning and scheduling problems. In chapter 6, we start from the scheduling problem based on the restless bandit model, and add switching penalties to model the costs incurred when the vehicles change location. We propose a heuristic and a performance bound that extend a previously proposed solution for the standard RBP. Moreover, we give a new interpretation of this heuristic in terms of approximate dynamic programming methods. We see however that as a consequence of the addition of switching costs, the computational issues increase significantly and the performance is not as satisfactory as for the previous sensor scheduling problems. Moreover, we would like to model more complicated and realistic scenarios, for which the MDP approach seems to lead to intractable formulations.

Hence the goal of the last chapter is to propose a modeling framework that can provide insight into the complicated spatiotemporal scheduling problems involved in controlling UVS. Using a fluid approximation of the dynamics of the environment, and despite the generality of the network

control problem considered, we show that a simple geometric condition allows us to determine if a given system is stabilizable and to construct stabilizing policies, at least for the fluid model. These policies simultaneously describe a trajectory for a vehicle and the allocation of its resources, such as on-board sensors or communication receivers.

8.2 Future Work

Research in motion planning and mission planning have each led to powerful methods for solving problems in their own domain. Separate research has been encouraged by the standard hierarchical structure of robot controllers, which is probably unavoidable to a certain degree. As demonstrated in chapter 2, there is great benefit to gain from a better integration of the motion planner of an UVS with the mission planner. Using a satisfying approximation of the vehicle dynamics such as the Dubins model in the higher layers of the robot controller can lead to improved decisions overall at a reasonable computational cost. The integration of simplified vehicle models in other decision problems solved by UVS controllers besides the TSP should be pursued. In particular, there are few models that integrate simultaneously the motion planning problems and dynamic scheduling problems as they arise in UVS control.

New ways to capture the path planning component of the UVS mission planning problem more realistically than in chapter 6 with reasonable computational cost need to be developed. The point of view of chapter 7 was that the model of the environment used by the decision making level can be simplified as well. Important open research questions were mentioned there, in particular related to trajectory optimization, but the fluid approximation seemed to offer valuable insight.

Different ways to integrate the UVS controller layers exist. Among them are:

1. solving the mission planning problem independently of the vehicle dynamical characteristics and forcing its solution on the system by calling the motion planner only after the plan has been decided;

2. simplifying the motion planning problem and adding it directly to the mission planning problem, possibly calling a more precise motion planner later to execute the plan; and

3. simplifying both the motion planning problem and the model of the environment used by the decision making layer and solving the combined problems simultaneously.

In general, a sound approach to UVS controller design would use in the higher layers a (probably conservative) abstraction of the vehicle model used by the lower layers, which would guarantee certain properties necessary to the correct execution of the plan by these lower layers. For example, if we know that an UAV can always follow the Dubins trajectories for a certain value of the Dubins turning radius larger than the UAV's own turning radius, we can safely use that model at the mission planning level. Hence approach 2 above is preferable to approach 1 in general. More work is needed however to choose the right vehicle model abstractions and solve the more complicated problems arising at the higher level. We expect that approach 3 will be useful here. Indeed, there has apparently been little work done so far on a robust control approach to mission planning. A simplified model of the environment could be used to design a nominal mission plan, and uncertainties and unmodeled characteristics of the problem could be accounted for by appropriate hedging and modification of the nominal controller. For example, we have proposed for the problem discussed in chapter 7 designing the nominal controller using a fluid approximation of the environment. Robust MDPs (see e.g. [96]) could be used in this framework as well if one still wants to keep models

based on controlled Markov chains for control design, but this requires the development of appropriate model reduction techniques for MDPs in order to start with a nominal model of reasonable complexity. Much work remains to be done in this direction, in terms of modeling, so that certain guarantees about the mission objectives can be satisfied, in terms of developing actual robust controller design methods, as well as evaluating the conservatism of this approach.

Finally, UVS are likely to remain supervised by humans. Future work could try to use some of the scheduling techniques presented in this thesis to improve the integration of the operator in the system, in particular to manage the operator's cognitive load and the ranking of tasks requiring his attention.

Bibliography

[1] H. Abou-Kandil, G. Freiling, V. Ionescu, and G. Jank. *Matrix Riccati Equations in Control and Systems Theory*. Systems & Control: Foundations & Applications. Birkhäuser, 2003.

[2] D. Adelman and A. Mersereau. Relaxations of weakly coupled stochastic dynamic programs. *Operations Research*, 2008. Accepted.

[3] P. Agarwal and H. Wang. Approximation algorithms for curvature-contrained shortest paths. *SIAM Journal on Computing*, 30(6):1739–1772, 2001.

[4] A. Aggarwal, D. Coppersmith, S. Khanna, R. Motwani, and B. Schieber. The angular-metric traveling salesman problem. *SIAM Journal on Computing*, 29(3):697–711, 1999.

[5] R. Agrawal, M.V. Hedge, and D. Teneketzis. Asymptotically efficient adaptive allocation rules for the multiarmed bandit problem with switching cost. *IEEE Transactions on Automatic Control*, 33(10):899–905, 1988.

[6] E. Altman. *Constrained Markov Decision Processes*. Chapman and Hall, 1999.

[7] M. Asawa and D. Teneketzis. Multi-armed bandits with switching penalties. *IEEE transactions on automatic control*, 41(3):328–348, 1996.

[8] M. Athans. On the determination of optimal costly measurement strategies for linear stochastic systems. *Automatica*, 8:397–412, 1972.

[9] S. Banda. Future directions in control for unmanned air vehicles. In *AFOSR Workshop on Future Directions in Control*, 2002.

[10] J.S. Baras and A. Bensoussan. Optimal sensor scheduling in nonlinear filtering of diffusion processes. *SIAM Journal on Control and Optimization*, 27(4):786–813, 1989.

[11] A. Behzad and M. Modarres. New efficient transformation of the generalized traveling salesman problem into traveling salesman problem. In *Proceedings of the 15th International Conference of Systems Engineering*, Las Vegas, 2002.

[12] H. Benzing, D. Kalin, and R. Theodorescu. Optimal policies for sequential Bernoulli experiments with switching costs. *Elektronische Informationsverarbeitung und Kybernetik*, 23(12):599–607, 1987.

[13] D.P. Bertsekas. *Nonlinear Programming*. Athena Scientific, 1999.

[14] D.P. Bertsekas. *Dynamic Programming and Optimal Control*, volume 1 and 2. Athena Scientific, 2nd edition, 2001.

[15] D.P. Bertsekas and D.A. Castañón. Rollout algorithms for stochastic scheduling problems. *Journal of Heuristics*, 5(1):89–108, 1999.

[16] D. Bertsimas and J. Niño-Mora. Conservation laws, extended polymatroids and multiarmed bandit problems, a polyhedral approach to indexable systems. *Mathematics of Operations Research*, 21(2):257–306, 1996.

[17] D. Bertsimas and J. Niño-Mora. Restless bandits, linear programming relaxations, and a primal-dual index heuristic. *Operations Research*, 48:80–90, 2000.

[18] D. J. Bertsimas and G. Van Ryzin. A stochastic and dynamic vehicle routing problem in the Euclidean plane. *Operations Research*, 39(4):601–615, July 1991.

[19] G. Birkhoff. Tres observaciones sobre el algebra lineal. *Univ. Nac. Tucuman Rev. Ser. A*, 5:147–151, 1946.

[20] S. Bittanti, P. Colaneri, and G. De Nicolao. The periodic Riccati equation. In S. Bittanti, A. J. Laub, and J. C. Willems, editors, *The Riccati equation*. Springer-Verlag, 1991.

[21] V. D. Blondel, J. M. Hendrickx, A. Olchevsky, and J. N. Tsitsiklis. Convergence in multiagent coordination, consensus and flocking. In *Proceedings of the 44th IEEE Conference on Decision and Control*, December 2005.

[22] J. D. Boissonnat and X. N. Bui. Accessibility region for a car that only moves forward along optimal paths. Technical Report 2181, INRIA, Sophia-Antipolis, France, 1994.

[23] S. P. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2006.

[24] R. W. Brockett. *Finite Dimensional Linear Systems*. John Wiley and Sons, 1970.

[25] B. L. Brumitt and A. Stentz. Dynamic mission planning for multiple mobile robots. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 2396–2401, 1996.

[26] A.E. Bryson and Y.-C. Ho. *Applied Optimal Control: Optimization, Estimation, and Control*. Taylor and Francis, 1975.

[27] D.A. Castañón. Approximate dynamic programming for sensor management. In *Proceedings of the 36th Conference on Decision and Control*, pages 1202–1207, December 1997.

[28] D.A. Castañón. Stochastic control bounds on sensor network performance. In *Proceedings of the 44th IEEE Conference on Decision and Control*, 2005.

[29] P. R. Chandler and S. Rasmussen. UAV cooperative path planning. In *AIAA Guidance, Navigation, and Control Conference and Exhibit*, 2000.

[30] C.S. Chang, W.J. Chen, and H.Y. Huang. Birkhoff-von Neumann input buffered crossbar switches. *INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, 3, 2000.

[31] T.H. Cormen, C.E. Leiserson, R.L. Rivest, and C. Stein. *Introduction to algorithms*. MIT Press, 2001.

[32] J. Cortes, S. Martinez, T. Karatas, and F. Bullo. Coverage control for mobile sensing networks. *IEEE Transactions on Robotics and Automation*, 20(2):243–255, April 2004.

[33] T.M. Cover and J.A. Thomas. *Elements of information theory*. Wiley New York, 1991.

[34] D.P. de Farias and B. Van Roy. The linear programming approach to approximate dynamic programming. *Operations Research*, 51(6):850–856, 2003.

[35] D.F. Delchamps. Analytic feedback control and the algebraic Riccati equation. *IEEE Transactions on Automatic Control*, 29(11):1031–1033, 1984.

[36] F. d'Epenoux. A probabilistic production and inventory problem (English translation). *Management Science*, 10:98–108, 1963.

[37] C. Derman. *Finite State Markovian Decision Processes*. Academic Press, 1970.

[38] 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*, 79(3):497–516, July 1957.

[39] F. Dusonchet. *Dynamic Scheduling for Production Systems Operating in a Random Environment*. PhD thesis, Ecole Polytechnique Fédérale de Lausanne, 2003.

[40] F. Dusonchet and M.-O. Hongler. Optimal hysteresis for a class of deterministic deteriorating two-armed bandit problem with switching costs. *Automatica*, 39(11):1947–1955, 2003.

[41] J.J. Enright and E. Frazzoli. UAV routing in a stochastic, time-varying environment. In *Proceedings of the IFAC World Congress*, 2005.

[42] Sebastian Thrun et al. Stanley: The robot that won the darpa grand challenge. *Journal of Field Robotics*, 23(9):661–692, September 2006.

[43] E. Feron and C. Olivier. Targets, sensors and infinite-horizon tracking optimality. In *Proceedings of the 29th IEEE Conference on Decision and Control*, 1990.

[44] E. Frazzoli and F. Bullo. Decentralized algorithms for vehicle routing in a stochastic time-varying environment. In *43rd IEEE Conference on Decision and Control*, pages 3357–3363, December 2004.

[45] A. Frieze, G. Galbiati, and F. Maffioli. On the worst-case performance of some algorithms for the asymmetric traveling salesman problem. *Networks*, 12:23–39, 1982.

[46] M.R. Garey, R.L. Graham, and D.S. Johnson. Some NP-complete geometric problems. In *Proceedings of the eighth annual ACM symposium on Theory of computing*, 1976.

[47] J. C. Gittins. *Multi-armed Bandit Allocation Indices*. Wiley-Interscience series in Systems and Optimization. John Wiley and sons, New York, 1989.

[48] J.C. Gittins. Bandit processes and dynamic allocation indices. *Journal of the Royal Statistical Society. Series B*, 41(2):148–177, 1979.

[49] J.C. Gittins and D.M. Jones. A dynamic allocation index for the sequential design of experiments. In J. Gani, editor, *Progress in Statistics*, pages 241–266. North-Holland, Amsterdam, 1974.

[50] K.D. Glazebrook, D. Ruiz-Hernandez, and C. Kirkbride. Some indexable families of restless bandit problems. *Advances in Applied Probability*, 38:643–672, 2006.

[51] Y. Gu, D. Bozdag, R.W. Brewer, and E. Ekici. Data harvesting with mobile elements in wireless sensor networks. *Computer Networks*, 50:3449–3465, 2006.

[52] S. Guha, K.Munagala, and P. Shi. On index policies for restless bandit problems. Technical report, 2007. Submitted.

[53] S. Guha, K. Munagala, and P. Shi. On index policies for restless bandit problems. *cond*, 2007.

[54] V. Gupta, T.H. Chung, B. Hassibi, and R.M.Murray. On a stochastic sensor selection algorithm with applications in sensor scheduling and sensor coverage. *Automatica*, 42(2):251–260, 2006.

[55] T. Hamada. A two-armed bandit problem with one arm known including switching costs and terminal rewards. *Journal of the Japan Statistical Society*, 17(1):21–20, 1987.

[56] J. T. Hawkins. *A Lagrangian Decomposition Approach to Weakly Coupled Dynamic Optimization Problems and its Applications*. PhD thesis, Massachusetts Institute of Technology, 2003.

[57] K. Helsgaun. An effective implementation of the Lin-Kernighan traveling salesman heuristic. *European Journal of Operational Research*, 126(1):106–130, 2000. Software available at `http://www.akira.ruc.dk/~keld/research/LKH/`.

[58] B.A. Huberman and F. Wu. The economics of attention: maximizing user value in information-rich environments. *Proceedings of the 1st international workshop on Data mining and audience intelligence for advertising*, pages 16–20, 2007.

[59] S. Itani and M. A. Dahleh. On the stochastic TSP for the dubins vehicle. In *Proceedings of the American Control Conference*, 2007.

[60] P. Jacobs and J. Canny. Planning smooth paths for mobile robots. In Z. Li and J. Canny, editors, *Nonholonomic Motion Planning*, pages 271–342. Kluwer Academic, 1992.

[61] T. Jun. A survey on the bandit problem with switching costs. *De Economist*, 152(4):513–541, 2004.

[62] T. Jun. Costly switching and investment volatility. *Economic Theory*, 25(2):317–332, 2005.

[63] D. Kalin and R. Theodorescu. A uniform two-armed bandit with one arm known and switching costs. *Revue roumaine de mathématiques pures et appliquées*, 47(2):179–189, 2002.

[64] L.C.M. Kallenberg. A note on M. N. Katehakis' and Y.-R. Chen's computation of the Gittins index. *Mathematics of Operations Research*, 11(1):184–186, February 1986.

[65] R. E. Kalman and R. S. Bucy. New results in linear filtering and prediction. *Journal of Basic Engineering (ASME)*, 83D:95–108, 1961.

[66] H. Kaplan, M. Lewenstein, N. Shafir, and M. Sviridenko. Approximation algorithms for asymmetric TSP by decomposing directed regular multidigraphs. In *Proceedings of IEEE FOCS*, pages 56–67, 2003.

[67] I. Karatzas. Gittins indices in the dynamic allocation problem for diffusion processes. *The Annals of Probability*, pages 173–192, 1984.

[68] R.M. Karp. Reducibility among combinatorial problems. In R.E. Miller and J.W. Thatcher, editors, *Complexity of Computer Computations*. Plenum Press, New York, 1972.

[69] M.N. Katehakis and Jr. A.F. Veinott. The multi-armed bandit problem: decomposition and computation. *Mathematics of Operations Research*, 12(2):262–268, February 1987.

[70] Richard J. Kenefic. Finding good Dubins tours for UAVs using particle swarm optimization. *Journal of Aerospace Computing, Information, and Communication*, 5, February 2008.

[71] A. T. Klesh, A. R. Girard, and P. T. Kabamba. Path planning for cooperative time-optimal information collection. In *Proceedings of the American Control Conference*, Seattle, June 2008.

[72] G. Koole. Assigning a single server to inhomogeneous queues with switching costs. *Theoretical Computer Science*, 182:203–216, 1997.

[73] B. Korte and J. Vygen. *Combinatorial Optimization: Theory and Algorithms*. Springer-Verlag, 3rd edition, 2005.

[74] V. Krishnamurthy and R.J. Evans. Hidden markov model multiarm bandits: a methodology for beam scheduling in multitarget tracking. *IEEE Transactions on Signal Processing*, 49(12):2893 – 2908, December 2001.

[75] V. Krishnamurthy and R.J. Evans. Correction to "hidden markov model multiarm bandits: a methodology for beam scheduling in multitarget tracking". *IEEE Transactions on Signal Processing*, 51(6):1662–1663, June 2003.

[76] T.L. Lai and H. Robbins. Asymptotically efficient adaptive allocation rules. *Advances in Applied Mathematics*, 6:4–22, 1985.

[77] W.-M. Lan and T. L. Olsen. Multiproduct systems with both setup times and costs: Fluid bounds and schedules. *Operations Research*, 54(3):505–522, May-June 2006.

[78] S. M. LaValle and S. A. Hutchinson. Optimal motion planning for multiple robots having independent goals. *IEEE Transactions on Robotics and Automation*, 14(6):912–925, December 1998.

[79] S.M. LaValle. *Planning Algorithms*. Cambridge University Press, 2006.

[80] J. Le Ny and E. Feron. An approximation algorithm for the curvature-constrained traveling salesman problem. In *Proceedings of the 43rd Annual Allerton Conference on Communications, Control and Computing*, September 2005.

[81] J. Le Ny and E. Feron. Restless bandits with switching costs: Linear programming relaxations, performance bounds and limited lookahead policies. In *Proceedings of the American Control Conference*, Minneapolis, MN, June 2006.

[82] J. Le Ny, E. Feron, and E. Frazzoli. The curvature-constrained traveling salesman problem for high point densities. In *Proceedings of the 46th IEEE Conference on Decision and Control*, 2007.

[83] H. Lee, K. Teo, and A.E. Lim. Sensor scheduling in continuous time. *Automatica*, 37:2017–2023, 2001.

[84] J.-H. Lee, O. Cheong, W.-C. Kwon, S. Y. Shin, and K.-Y. Chwa. Approximation of curvature-constrained shortest paths through a sequence of points. In *ESA '00: Proceedings of the 8th Annual European Symposium on Algorithms*, pages 314–325, London, UK, 2000. Springer-Verlag.

[85] U. Lee, E. Magistretti, B. Zhou, M. Gerla, P. Bellavista, and A. Corradi. Efficient data harvesting in mobile sensor platforms. In *Fourth IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOMW'06)*, 2006.

[86] N.E. Leonard, D.A. Paley, F. Lekien, R. Sepulchre, D.M. Fratantoni, and R.E. Davis. Collective motion, sensor networks, and ocean sampling. *Proceedings of the IEEE*, 95(1):48–74, January 2007.

[87] P. Liu, R.A. Berry, and M.L. Honig. A fluid analysis of a utility-based wireless scheduling policy. *IEEE Transactions on Information Theory*, 52(7):2872–2889, 2006.

[88] X. Ma and D.A. Castañón. Receding horizon planning for Dubins traveling salesman problems. In *Proceedings of the IEEE Conference on Decision and Control*, 2006.

[89] A.S. Manne. Linear programming and sequential decisions. *Management Science*, 6:259–267, 1960.

[90] A.W. Marshall and I. Olkin. *Inequalities: theory of majorization and its applications*. Academic Press, 1979.

[91] L. Meier, J. Perschon, and R.M. Dressler. Optimal control of measurement systems. *IEEE Transactions on Automatic Control*, 12(5):528–536, 1967.

[92] S. Meyn. *Control Techniques for Complex Networks*. Cambridge University Press, 2008.

[93] L. Mirsky. On a convex set of matrices. *Archiv der Mathematik*, 10:88–92, 1959.

[94] A. I. Mourikis and S. I. Roumeliotis. Optimal sensor scheduling for resource-constrained localization of mobile robot formations. *IEEE Transactions on Robotics*, 22(5):917–931, 2006.

[95] G. De Nicolao. On the convergence to the strong solution of periodic Riccati equations. *International Journal of Control*, 56(1):87–97, 1992.

[96] A. Nilim and L. El Ghaoui. Robust control of Markov decision processes with uncertain transition matrices. *Operations Research*, 53(5):780–798, September-October 2005.

[97] J. Niño-Mora. Restless bandits, partial conservation laws and indexability. *Advances in Applied Probability*, 33:76–98, 2001.

[98] R. Olfati-Saber. Flocking for multi-agent dynamic systems: algorithms and theory. *Automatic Control, IEEE Transactions on*, 51(3):401–420, 2006.

[99] Y. Oshman. Optimal sensor selection strategy for discrete-time state estimators. *IEEE Transactions on Aerospace and Electronic Systems*, 30(2):307–314, 1994.

[100] C.H. Papadimitriou. The Euclidean traveling salesman problem is NP-complete. *Theoretical Computer Science*, 4:237–244, 1977.

[101] C.H. Papadimitriou and J.N. Tsitsiklis. The complexity of optimal queueing network control. *Mathematics of Operations Research*, 24(2):293–305, 1999.

[102] Valentin Polishchuk and Jukka Suomela. Optimal backlog in the plane. arXiv:0804.4819v1, April 2008.

[103] M. L. Puterman. *Markov Decision Processes*. Series in Probability and Statistics. Wiley, 1994.

[104] S. Rathinam, R. Sengupta, and S. Darbha. A resource allocation algorithm for multi-vehicle systems with non-holonomic constraints. *IEEE Transactions on Automation Science and Engineering*, 2006.

[105] M.I. Reiman and L.M. Wein. Dynamic scheduling of a two class queue with setups. *Operations Research*, 46:532–547, 1998.

[106] D.J. Rosenkrantz, R.E. Stearns, and P.M. Lewis II. An analysis of several heuristics for the traveling salesman problem. *SIAM Journal on Computing*, 6:563–581, 1977.

[107] AV Savkin, RJ Evans, and E. Skafidas. The problem of optimal robust sensor scheduling. *Systems and Control Letters*, 43:149–157, 2001.

[108] K. Savla, F. Bullo, and E. Frazzoli. On the stochastic traveling salesperson problem for Dubins' vehicle. In *Proceedings of the IEEE conference on decision and control*, Seville, Spain, December 2005.

[109] K. Savla, E. Frazzoli, and F. Bullo. On the point-to-point and traveling salesperson problems for Dubins' vehicle. In *Proceedings of the American Control Conference*, Portland, OR, June 2005.

[110] K. Savla, E. Frazzoli, and F. Bullo. Traveling salesperson problems for the Dubins vehicle. *IEEE Transactions on Automatic Control*, 53(9), 2008.

[111] B. F. La Scala and B. Moran. Optimal target tracking with restless bandits. *Digital Signal Processing*, 16:479–487, 2006.

[112] M.K. Schneider, G.L. Mealy, and F.M. Pait. Stochastic dynamic programming based approaches to sensor resource management. In *Proceedings of the American Control Conference*, 2004.

[113] T. Schouwenaars, B. De Moor, E. Feron, and J. How. Mixed integer programming for multi-vehicle path planning. In *Proceedings of the European Control Conference*, 2001.

[114] A. Schrijver. *Combinatorial Optimization - Polyhedra and Efficiency*. Springer, 2003.

[115] C. Schumacher, P. R. Chandler, and S. R. Rasmussen. Task allocation for wide area search munitions. In *Proceedings of the American Control Conference*, 2002.

[116] Rahul C. Shah, Sumit Roy, Sushant Jain, and Waylon Brunette. Data mules: Modeling a three-tier architecture for sparse sensor networks. In *Proceedings of the first IEEE international workshop on Sensor Network Protocols and Applications*, 2003.

[117] L. Shi, M. Epstein, B. Sinopoli, and R.M. Murray. Effective sensor scheduling schemes in a sensor network by employing feedback in the communication loop. In *Proceedings of the 16th IEEE International Conference on Control Applications*, 2007.

[118] A.M. Shkel and V.J. Lumelsky. Classification of the Dubins' set. *Robotics and Autonomous Systems*, 34:179–202, 2001.

[119] E. Skafidas and Nerode. Optimal measurement scheduling in linear quadratic gaussian control problems. In *Proceedings of the IEEE International Conference on Control Applications*, pages 1225–1229, Trieste, Italy, 1998.

[120] A. A. Somasundara, A. Ramamoorthy, and M. B. Srivastava. Mobile element scheduling for efficient data collection in wireless sensor networks with dynamic deadlines. In *Proceedings of the 25th IEEE International Real-Time Systems Symposium (RTSS'04)*, 2004.

[121] E.J. Sondik. The optimal control of partially observable markov decision processes over the infinite horizon: Discounted costs. *Operations Research*, 26(2):282–304, March-April 1978.

[122] N.-O. Song and D. Teneketzis. Discrete search with multiple sensors. *Mathematical Methods of Operations Research (ZOR)*, 60(1):1–13, September 2004.

[123] H. Takagi. *Analysis of Polling Systems*. The MIT Press, 1986.

[124] Z. Tang and Ü. Özgüner. Motion planning for multitarget surveillance with mobile sensor agents. *IEEE Transactions on Robotics*, 21:898–908, 2005.

[125] A. Tiwari. *Geometrical analysis of spatio-temporal planning problems*. PhD thesis, California Institute of Technology, 2006.

[126] A. Tiwari, M. Jun, D. E. Jeffcoat, and R. M. Murray. Analysis of dynamic sensor coverage problem using Kalman filters for estimation. In *Proceedings of the 16th IFAC World Congress*, 2005.

[127] H. L. Trentelman and J. C. Willems. The dissipation inequality and the Algebraic Riccati Equation. In S. Bittanti, A. J. Laub, and J. C. Willems, editors, *The Riccati Equation*. Springer-Verlag, 1991.

[128] UAV roadmap. Unmanned aircraft systems roadmap 2005-2030. Technical report, Office of the Secretary of Defense, 2005.

[129] D. Uciński. *Optimal Measurement Methods for Distributed Parameter System Identification*. CRC Press, 2005.

[130] M.P. Van Oyen, D. Pandelis, and D. Teneketzis. Optimality of index policies for stochastic scheduling with switching penalties. *Journal of Applied Probability*, 29:957–966, 1992.

[131] P. Varaiya, J.C. Walrand, and C. Buyukkoc. Extensions of the multiarmed bandit problem: the discounted case. *IEEE transactions on automatic control*, 30(5):426–439, 1985.

[132] R.B. Washburn, M.K. Schneider, and J.J. Fox. Stochastic dynamic programming based approaches to sensor ressource management. In *Proceedings of the International Conference on Information Fusion*, 2002.

[133] R.R. Weber and G. Weiss. On an index policy for restless bandits. *Journal of Applied Probability*, 27:637–648, 1990.

[134] G. Weiss. Approximation results in parallel machines stochastic scheduling. *Annals of Operations Research*, Volume 26(1):195–242, December 1990.

[135] P. Whittle. Multi-armed bandits and the Gittins index. *Journal of the Royal Statistical Society. Series B*, 42(2):143–149, 1980.

[136] P. Whittle. Restless bandits: activity allocation in a changing world. *Journal of Applied Probability*, 25A:287–298, 1988.

[137] J.L. Williams. *Information Theoretic Sensor Management*. PhD thesis, Massachusetts Institute of Technology, February 2007.

[138] H. Xu. *Optimal policies for stochastic and dynamic vehicle routing problems*. PhD thesis, Massachusetts Institute of Technology, 1995.