APPROXIMATE DYNAMIC PROGRAMMING POLICIES AND PERFORMANCE BOUNDS FOR AMBULANCE REDEPLOYMENT

A Dissertation

Presented to the Faculty of the Graduate School

of Cornell University

in Partial Fulfillment of the Requirements for the Degree of

Doctor of Philosophy

by

Matthew Scott Maxwell

May 2011

APPROXIMATE DYNAMIC PROGRAMMING POLICIES AND PERFORMANCE BOUNDS FOR AMBULANCE REDEPLOYMENT

Matthew Scott Maxwell, Ph.D.

Cornell University 2011

Ambulance redeployment is the practice of dynamically relocating idle ambulances based upon real-time information to reduce expected response times for future emergency calls. Ambulance redeployment performance is often measured by the fraction of "lost calls" or calls with response times larger than a given threshold time. This dissertation is a collection of four papers detailing results for designing ambulance redeployment policies and bounding the performance of an optimal ambulance redeployment policy.

In the first paper ambulance redeployment is modeled as a Markov decision process, and an approximate dynamic programming (ADP) policy is formulated for this model. Computational results show that the ADP policy is able to outperform benchmark policies in two different case studies based on real-life data. Results of practical concern including how the ADP policy performs with varying call arrival rates and varying ambulance fleet sizes are also included.

In the second paper we discuss ADP tuning procedures, i.e., the process of selecting policy parameters to improve performance. We highlight limitations present in many ADP tuning procedures and propose direct-search tuning methods to overcome these limitations. To facilitate direct-search tuning for ambulance redeployment, we reformulate the ADP policy using the so-called "post-decision state" formulation. This reformulation allows policy decisions to be computed without computationally expensive simulations and makes direct-

search tuning computationally feasible.

In the third paper we prove that many ADP policies are equivalent to a simpler class of policies called nested compliance table (NCT) policies that assign ambulances to bases according to the total number of available ambulances. Furthermore, we show that if ambulances are not assigned to the bases dictated by the NCT policy, the ADP-based policies will restore compliance to the NCT policy without dispatcher intervention.

In the fourth paper we derive a computationally tractable lower bound on the minimum fraction of lost calls and propose a heuristic bound based upon simulation data from a reference policy, i.e., a policy we believe to be close to optimal. In certain circumstances, both bounds can be quite loose so we introduce a stylized model of ambulance redeployment and show empirically that for this model the lower bound is quite tight.

BIOGRAPHICAL SKETCH

Matthew Maxwell was born in St. George, Utah and survived his childhood and teenage years by skimboarding in the Virgin River, making forts near said river, and reading MS-DOS textbooks as his leisure vacation reading. He also attempted an underground miniature golf course with one of his brothers, but the big hole they dug eventually got swallowed up when his parents put in a swimming pool. Matt still can't decide if he'd rather have the unfinished miniature golf course or the swimming pool, but he's leaning towards the pool.

Before going to college, Matt served a two-year mission in Japan for the Church of Jesus Christ of Latter-day Saints. Upon his return, he began his studies at Brigham Young University, graduating with honors in 2006 with a B.S. in Computer Science and minors in Japanese and Mathematics.

Matt and his wife, Caroline, have a two-year-old daughter named Madeline. In June of 2011, after completing his Ph.D. at Cornell University, Matt and his family will be moving to Cary, North Carolina where Matt will begin his job as Operations Research Specialist working on hotel revenue management for SAS.

To my wife Caroline.

And my parents Leon and Elizabeth.

ACKNOWLEDGEMENTS

First and foremost, I acknowledge my advisors Shane Henderson and Huseyin Topaloglu for their unmeasurable support in this research. They have been a pleasure to work with.

Much of my research builds upon the previous work of Mateo Restrepo who deserves credit for laying the foundation of approximate dynamic programming policies for ambulance redeployment. I also appreciate Armann Ingolfsson, Andrew Mason, and Oddo Zhang for advice, insight, and data that they have contributed to my research.

I thank Mark Lewis and David Ruppert for serving on my committee and assisting me in my dissertation work. I also thank the rest of the faculty and staff of the School of Operations Research and Information Engineering, especially Kathy King, who possesses an inexhaustible wealth of information.

Reaching back to my distant undergraduate past, I feel an immense sense of gratitude to my undergraduate advisor Sean Warnick. His inspiration and untiring enthusiasm is intoxicating. It is not an understatement to say that Sean had a large impact on shaping the trajectory of my life. I also acknowledge Roger Hansen at the U.S. Bureau of Reclamation for being a superlative supervisor. My undergraduate research revolved heavily around my work with the Bureau, and Roger was the one who facilitated it.

And most importantly, I acknowledge my wife for marrying me—even after I told her I planned on graduate school—and always supporting me. I also acknowledge my daughter who gave me the most adorable type of motivation imaginable.

This research was partially supported by National Science Foundation grants DMI 0400287, DMI 0422133, and CMMI 0758441 and a U.S. Department

of Homeland Security Graduate Fellowship.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 Background

Emergency medical service (EMS) providers are responsible for administering emergency medical assistance within a given region. Typically this includes responding to emergency calls, treating the patient at the scene, and transporting the patient to a hospital if necessary. One common metric used to evaluate EMS performance, both in practice and in research literature, is the fraction of calls that are not responded to within a given time threshold, i.e., the fraction of calls for which the time between when the emergency call arrived and the time when an ambulance first arrived at the scene exceeds the time threshold. Typically EMS providers are under contract to maintain specified performance levels and face serious consequences if contracted performance levels are not met.

To meet these performance requirements EMS providers seek to reduce response times by properly allocating ambulances to bases throughout the region. Early ambulance location models assume ambulances return to their "home" base, or the base to which they have been assigned, after becoming available and optimize over all possible ambulance allocations. Typically these models are designed as integer programs that maximize the fraction of the demand that is covered, i.e., that can be reached within the time threshold, or some other simplified performance metric. These policies are often called ambulance location policies or static policies. Refinements on the original model formulation recognize that an ambulance may be busy with a call and unable to respond to

another call. One model formulation that incorporates this concept maximizes the area that is covered by multiple ambulances so that backup ambulances will be available if one ambulance is busy. Another formulation encourages multiple ambulance to cover a single area by estimating the probability a given ambulance will be available and weighting the objective function contribution of each ambulance by this amount. A more thorough survey of these models can be found in Swersey (1994) and Brotcorne et al. (2003).

Increasing traffic congestion has made the ambulance allocation problem even more difficult since congestion leads to longer travel times for ambulances responding to emergency calls. To compensate for increased travel times ambulances must be spaced further apart in an attempt meet the performance requirements. The difficulty arises when an ambulance is dispatched to an emergency call. Since all the ambulances are spread out there may be no ambulance close enough to respond to the dispatched ambulance's area in a timely manner. Consequently, emergency calls arriving in the same area while that ambulance remains busy are likely to be "lost," or not responded to within the time threshold.

Ambulance redeployment is one strategy used by EMS providers to improve performance. Ambulance redeployment, also known as relocation, move up, system status management, or dynamic repositioning, is the practice of repositioning idle ambulances according to real-time information to be better prepared to respond to future emergency calls. For example, in response to an ambulance being dispatched to an emergency call, an available ambulance may be assigned to travel to the dispatched ambulance's previous location in preparation for future emergencies that may happen in the vicinity. Since these policies

depend upon current information they are often called dynamic policies.

This research on ambulance redeployment centers around approximate dynamic programming (ADP) policies for ambulance redeployment. Chapter 2 formulates an ADP policy for ambulance redeployment and Chapter 3 investigates methods that can be used to tune the ADP policy efficiently. Chapter 4 shows that a certain class of ADP policies can be expressed in a simple form that is easily understood and implemented by dispatchers. Chapter 5 gives lower bounds on the performance for any redeployment policy and compares these bounds with the performance of known ADP policies. The remainder of this chapter explains the contribution of each chapter in more detail.

1.2 Approximate Dynamic Programming for Ambulance Redeployment

In Chapter 2, we give an introduction to ambulance redeployment and ADP policies for ambulance redeployment. We cover the three main steps in constructing an ADP policy: (1) modeling the problem as a Markov Decision Process (MDP), (2) choosing a framework, e.g., a linear combination of basis functions, used to approximate the dynamic programming (DP) value function, and (3) tuning policy parameters to improve performance within the given framework.

After formulating the ADP policy, we show simulated results for two case studies based on real-life data. These results show that an ADP policy is able to outperform benchmark polices that do not use real-time state information to

make redeployment decisions. Additional results of practical importance are shown such as how ADP policies perform with varying call arrival rates and ambulance fleet sizes, how ADP policies are able to reduce response times, and how turn-out time, i.e., the time it takes an ambulance crew to assemble and leave a base after receiving an emergency call, affects performance.

This chapter is a collaboration of results from myself, Mateo Restrepo, Shane G. Henderson, and Huseyin Topaloglu. Mateo is credited with developing the MDP formulation, designing the basis functions, and setting up the simulation environment for Chapter 2 (Restrepo (2008)).

My contribution in Chapter 2 extends Mateo Restrepo's work in many ways. We modified the simulation environment to increase computational efficiency and remove significant errors that existed. During this modification process, we fundamentally changed the simulation to facilitate the ability of running microsimulations, i.e., simulations within simulations. These microsimulations increased policy performance by providing more information to the ADP when making decisions. All the computational results in this paper were performed with the modified simulation environment. Additionally, we performed a comprehensive evaluation of each of the basis functions to ensure that they were all beneficial to the ADP policy.

The contents of Chapter 2 are contained in the paper "Approximate Dynamic Programming for Ambulance Redeployment" as published in the *INFORMS Journal on Computing* (Maxwell et al. (2010c)). A companion paper that highlights the simulation model used in the computational results was published in the proceedings of the 2009 Winter Simulation Conference (Maxwell et al. (2009)).

1.3 Tuning Approximate Dynamic Programming Policies for Ambulance Redeployment via Direct Search

In Chapter 3, we illustrate some limitations of popular methods used to tune ADP policies. Specifically, we give examples where a properly tuned ADP policy would result in an optimal policy, but ADP policies tuned via standard approaches do not.

The reason behind this result is that most ADP-tuning methods attempt to approximate the DP value function and then use this approximate value function to make decisions as a DP policy would. Thus, policy parameters that do not approximate the DP value function well are unlikely to be chosen—even if they would induce an optimal policy. We suggest the use of simulation-based direct-search methods to tune the ADP policy parameters directly based on the performance resulting from these parameters.

A direct-search method is more computationally expensive than value-function fitting methods. To apply this method to ambulance redeployment, we reformulate the simulation-based ADP policy around the post-decision state. Given a state and a feasible action, the post-decision state is the state the system enters immediately after the action is made—before any time passes or stochastic events occur. Reformulating a problem around the post-decision state can often reduce the size of the state space and reduce computational requirements (Powell, 2007, Chapter 4). For ADP ambulance redeployment policies, this reformulation greatly reduces the computation needed without sacrificing performance. With the reduced computational requirements, we tune an ambulance redeployment ADP policy via direct search and show empirically that the re-

sulting policy is superior to that found through a standard tuning method for the case study considered. The post-decision state formulation also becomes useful in Chapter 4.

The contents of Chapter 3 are contained in the paper "Tuning Approximate Dynamic Programming Policies for Ambulance Redeployment via Direct Search," which has been submitted for publication (Maxwell et al. (2010b)). An initial version of this paper was published in the proceedings of the 2010 Winter Simulation Conference (Maxwell et al. (2010a)).

1.4 Equivalence Results for Approximate Dynamic Programming and Compliance Table Policies for Ambulance Redeployment

In Chapter 4 we define a class of post-decision state ADP policies and show that this class of policies is equivalent to the class of nested compliance table (NCT) policies. For a given number of available ambulances, a NCT policy dictates how many ambulances should be assigned to each ambulance base. Furthermore, to reduce strain on ambulance crews, NCT policies are designed so that at most one ambulance must move locations when an ambulance either becomes busy or becomes free. When ambulances are assigned as a NCT policy dictates, the system is said to be in a state of compliance.

A NCT policy is a simple and intuitive policy that dispatchers are able to easily use and understand. We give a set of conditions for post-decision state ADP ambulance redeployment policies and prove that if these conditions are

satisfied then the ADP policy is actually a NCT policy as well. With this result it is possible to use ADP methodology to find redeployment policies that perform well and transform these policies into NCT policies that can be readily understood.

Furthermore, we show that if the system is in a non-compliant state an ADP-based policy will return the system to a state of compliance after a certain number of redeployment decisions. We give a simple expression for the number of redeployment decisions needed.

1.5 Performance Bounds for Ambulance Redeployment

In Chapter 5 we derive a computationally tractable lower bound on the fraction of calls responded to after a given time threshold in a finite simulated time horizon. We model the ambulance system as a queueing system where servers represent ambulances and arrivals represent emergency calls. We obtain a stochastic lower bound on the service time distribution for this queue using any ambulance redeployment policy, and the lower bound follows from known pathwise bounding results for queueing systems. This service time lower bound distribution can be computed by solving a finite number of integer programs.

Since an available ambulance may respond to an emergency call from any location, the lower bound supposes that the ambulances are optimally located throughout the region. In practice, however, most available ambulances sit idle at bases awaiting future calls. In such situations the lower bound will generate a very conservative estimate.

To refine this lower bound we estimate the distribution of the number of ambulances idle at base from a reference policy. Given these estimates we modify the integer programs used in the lower bound calculation and provide better bounds on the performance of an optimal policy. Since an optimal policy may have a different distribution of ambulances idle at base than the reference policy this new bound is only approximate.

Even after refining the lower bound heuristically the bound may still be overly conservative. To compensate for this we introduce a stylized model of ambulance redeployment that assumes ambulances always respond from bases. For this model we show empirically that the lower bound calculation is quite tight.

CHAPTER 2

APPROXIMATE DYNAMIC PROGRAMMING FOR AMBULANCE REDEPLOYMENT

2.1 Introduction

Rising costs of medical equipment, increasing call volumes and worsening traffic conditions are putting emergency medical service (EMS) managers under pressure to meet performance goals set by regulators or contracts. Ambulance redeployment is one strategy that can potentially help. Ambulance redeployment, also known as relocation, move up, system-status management or dynamic repositioning, refers to any strategy by which a dispatcher repositions idle ambulances to compensate for others that are busy, and hence, unavailable. The increasing availability of geographic information systems and the increasing affordability of computing power have finally created ideal conditions for bringing real-time ambulance redeployment approaches to fruitful implementation.

In this paper, which is an outgrowth of Restrepo (2008), we present an approximate dynamic programming (ADP) approach for making real-time ambulance redeployment decisions. We begin by formulating the ambulance redeployment problem as a dynamic program. This dynamic program involves a high-dimensional and uncountable state space and we address this difficulty by constructing approximations to the value function that are parameterized by a small number of parameters. We tune the parameters through an iterative and simulation-based method. Each iteration of this method consists of two steps. In the first step, we simulate the trajectory of the greedy policy induced by the

current value function approximation and collect cost trajectories of the system. In the second step, we tune the parameters of the value function approximation by solving a regression problem that fits the value function approximation to the collected cost trajectories. This yields a new set of parameters that characterize a new value function approximation, and so, we can go back and repeat the same two steps above. In this respect, the idea we use closely resembles the classical policy iteration algorithm in the Markov decision process literature. In particular, the first and second steps are respectively analogous to the policy evaluation and policy improvement step of the policy iteration algorithm.

There are two streams of literature that are related to our work. The first one is the literature on ADP. A generic approach for ADP involves using value function approximations of the form $\sum_{p=1}^{P} r_p \, \phi_p(\cdot)$, where $\{r_p : p = 1, \ldots, P\}$ are tunable parameters and $\{\phi_p(\cdot) : p = 1, \ldots, P\}$ are fixed basis functions; see Bertsekas and Tsitsiklis (1996) and Powell (2007). There are a number of methods to tune the parameters $\{r_p : p = 1, \ldots, P\}$ so that $\sum_{p=1}^{P} r_p \, \phi_p(\cdot)$ yields a good approximation to the value function. For example, temporal-difference learning and Q-learning use stochastic approximation ideas in conjunction with simulated trajectories of the system to iteratively tune the parameters; see Sutton (1988), Watkins and Dayan (1992), Tsitsiklis (1994), Bertsekas and Tsitsiklis (1996), Tsitsiklis and Van Roy (1997) and Si et al. (2004). The linear programming approach for ADP finds a good set of values for the parameters by solving a large linear program whose decision variables are $\{r_p : p = 1, \ldots, P\}$; see Schweitzer and Seidmann (1985), de Farias and Van Roy (2003) and Adelman and Mersereau (2008). Both classes of approaches are aimed at tuning the parameters $\{r_p : p = 1, \ldots, P\}$ so that $\sum_{p=1}^{P} r_p \, \phi_p(\cdot)$ yields a good approximation to the value function. The choice of the basis functions $\{\phi_p(\cdot) : p = 1, \ldots, P\}$, on

the other hand, is regarded as more of an art form, requiring substantial knowledge of the problem structure. Applications of ADP include inventory control (Van Roy et al., 1997), inventory routing (Adelman, 2004), option pricing (Tsitsiklis and Van Roy, 2001), game playing (Yan et al., 2005; Farias and Van Roy, 2006), dynamic fleet management (Topaloglu and Powell, 2006) and network revenue management (Adelman, 2007; Farias and Van Roy, 2007).

The second stream of literature that is related to our work is the literature on ambulance redeployment. One class of redeployment models involves solving integer programs in real-time whenever an ambulance redeployment decision needs to be made; see Kolesar and Walker (1974), Gendreau et al. (2001), Brotcorne et al. (2003), Gendreau et al. (2006), and Nair and Miller-Hooks (2006). The objective function in these integer programs involves a combination of backup coverage for future calls and relocation cost of ambulances. They are usually computationally intensive, since they require solving an optimization problem every time a decision is made. As a result, a parallel computing environment is sometimes used to implement a working real-time system. A second class of models is based on solving integer programs in a preparatory phase. This approach provides a lookup table describing, for each number of available ambulances, where those ambulances should be deployed. Dispatchers attempt to dispatch so as to keep the ambulance configuration close to the one suggested by the lookup table; see Ingolfsson (2006) and Goldberg (2007). A third class of models attempts to capture the randomness in the system explicitly, either through a dynamic programming formulation or through heuristic approaches. Berman (1981a), Berman (1981b) and Berman (1981c) represent the first papers that provide a dynamic programming approach for the ambulance redeployment problem, and this approach was revisited recently by Zhang et al.

(2008) to attempt to gain insight. However, these papers follow an exact dynamic programming formulation and, as is often the case, this formulation is tractable only in oversimplified versions of the problem with few vehicles and small transportation networks. Andersson (2005) and Andersson and Vaerband (2007) make the ambulance redeployment decision by using a "preparedness function" that essentially measures the capability of a certain ambulance configuration to cover future calls. The preparedness function is similar in spirit to the value function in a dynamic program, measuring the impact of current decisions on the future evolution of the system. However, the way the preparedness function is constructed is heuristic in nature.

When compared with the three classes of models described above, our approach provides a number of advantages. In contrast to the models that are based on integer programs, our approach captures the random evolution of the system over time since it is based on a dynamic programming formulation of the ambulance redeployment problem. Furthermore, the decisions made by our approach in real-time can be computed very quickly as this requires solving a simple optimization problem that minimizes the sum of the immediate cost and the value function approximation. In lookup table approaches, there may be more than one way to redeploy the ambulances so that the ambulance configuration over the transportation network matches the configuration suggested by the lookup table. Therefore, table lookup approaches still leave some aspects of dispatch decisions to subjective interpretation by dispatchers. Our approach, on the other hand, can fully automate the decision making process, while allowing dispatchers to override recommendations if they wish. In traditional dynamic programming approaches, one is usually limited to very small problem instances, whereas ADP can be used on problem instances with realis-

tic dimensions. Our approach allows working with a variety of objective functions, such as the number of calls that are not served within a threshold time standard or the total response time for the calls. Furthermore, our approach allows the possibility of constraining the frequency and destinations of ambulance relocations. This is important since a relocation scheme should balance improvements in service with the additional redeployment burden imposed on ambulance crews.

In summary, we make the following research contributions. 1) We develop a tractable ADP approach for the ambulance redeployment problem. Our approach employs value function approximations of the form $\sum_{p=1}^{P} r_p \, \phi_p(\cdot)$ and uses sampled cost trajectories of the system to tune the parameters $\{r_p : p = 1, \ldots, P\}$. Since it is based on the dynamic programming formulation of the problem, our approach is able to capture the random evolution of the system over time. 2) We develop a set of basis functions $\{\phi_p(\cdot) : p = 1, \ldots, P\}$ that yield good value function approximations for the ambulance redeployment problem. This opens up the possibility of using other ADP approaches, such as temporal-difference learning and the linear programming approach. 3) We provide computational experiments on EMS systems in two metropolitan areas. Our results indicate that ADP has the potential to obtain good redeployment policies in real systems. They also show that our approach compares favorably with benchmark policies that are similar to those used in practice.

The remainder of this paper is organized as follows. In Section 2.2, we present a dynamic programing formulation for the ambulance redeployment problem. In Section 2.3, we describe our ADP approach. In Section 2.4, we discuss the basis functions that we use in our value function approximations.

In Section 2.5, we report computational results for two metropolitan areas. We conclude in Section 2.6.

2.2 Ambulance Redeployment as a Markov Decision Process

This section presents a dynamic programming formulation of the ambulance redeployment problem. As will shortly be clear, our model involves an uncountable state space. For an excellent account of the basic terminology, notation and fundamental results regarding dynamic programming in uncountable state spaces, we refer the reader to Bertsekas and Shreve (1978).

2.2.1 State Space

There are N ambulances in the EMS system. To simplify the presentation, we assume that we do not keep more than M waiting calls, possibly by diverting excess calls to another EMS organization. This is not a restriction from a practical perspective since M can be quite large. The two main components in the state of the system are the vectors $A = (a_1, \ldots, a_N)$ and $C = (c_1, \ldots, c_M)$, where a_i contains information about the state of the ith ambulance and c_j contains information about the jth waiting call. Naturally, the state of the ambulances and the calls in the waiting queue evolve over time, but we omit this dependence for brevity. The state of ambulance i is given by a tuple $a_i = (\sigma_i, \ell_i, d_i, t_i)$, where σ_i is the status of the ambulance, ℓ_i and d_i are respectively the origin and destination locations of the ambulance and t_i is the starting time of any ambulance movement. To serve a call, an ambulance first moves to the call scene and provides

service at the scene for a certain amount of time. Following this, the ambulance transports the patient to a hospital, and after spending some time at the hospital, the ambulance becomes free to serve another call. Therefore, the status of an ambulance σ_i can take the values "idle at base," "going to scene of call," "serving at scene of call," "going to hospital," "transferring patient to hospital" and "returning to base." If ambulance i is stationary, then we have $\ell_i = d_i$. If ambulance i is in motion, then t_i corresponds to the starting time of this movement. Otherwise, t_i corresponds to the starting time of the current phase in the service cycle. For example, if the status of the ambulance is "transferring patient at hospital," then t_i corresponds to the time at which the ambulance arrived at the hospital. This time is kept in the state variable to give a Markov formulation for the non-Markovian elements in the system, such as non exponentially distributed service times and deterministic travel times. Similarly, for the jth call in the waiting queue, we have $c_j = (\delta_j, p_j, \zeta_j, \eta_j)$, where δ_j is the status of the call, p_j is the location of the call, ζ_j is the time at which the call arrived into the system and η_j is the priority level of the call. The status of a call δ_j takes one of the values "assigned to ambulance i" and "queued for service." We take a call off the waiting queue C as soon as an ambulance reaches this call and starts serving it.

We model the dynamics of the system as an event-driven process. Events are triggered by changes in the status of the ambulances or by call arrivals. Therefore, the possible event types in the system are "call arrives and is placed in the jth position," "ambulance i departs for scene of call j," "ambulance i arrives at scene of call j," "ambulance i leaves scene of call for hospital," "ambulance i arrives at hospital," "ambulance i finishes at hospital" and "ambulance i arrives at base." We assume that we can make decisions only at the times of these events.

Events occur at discrete points in time, so our modeling approach precludes the possibility of making a decision at any time point. This naturally comes at the cost of some loss of optimality as it may be desirable to make decisions between the times of the events. For example, our modeling approach, as stated, does not allow the possibility of rerouting an ambulance before it reaches its destination. It may be possible to incorporate such extensions by defining artificial events such as "consider repositioning" and firing these events while an ambulance is in transit. Nevertheless, we do not consider these extensions here and rely on the fact that the events that we work with occur frequently enough to provide ample decision opportunities.

By restricting our attention to the times of events, we visualize the system as jumping from one event time to another. Therefore, we can use the tuple $s = (\tau, e, A, C)$ to represent the state of the system, where τ corresponds to the current time, e corresponds to the current event type, and A and C respectively correspond to the state of the ambulances and the waiting call queue. In this case, the state trajectory of the system can be written as $\{s_k : k = 1, 2, \ldots\}$, where s_k is the state of the system just after the kth event occurs. Time is rolled into our state variable. Throughout the paper, we use $\tau(s)$ and $e(s)$ to respectively denote the time and the event type when the state of the system is s. In other words, $\tau(s)$ and $e(s)$ are the first two components of the tuple $s = (\tau, e, A, C)$.

2.2.2 Controls

We assume that calls are served in decreasing order of priority, and within a given priority level, they are served in first-in-first-out order. Furthermore, the

closest available ambulance is dispatched to a call. This is not an exact representation of reality, but it is close enough for our purposes.

We use $\mathcal{R}(s)$ to denote the set of ambulances that are available for redeployment when the state of the system is s. In our implementation, if the state of the system is s and the event $e(s)$ is of the form "ambulance i finishes at hospital," then we let $\mathcal{R}(s) = \{i\}$. Otherwise, we let $\mathcal{R}(s) = \varnothing$. As a result, we consider an ambulance as available for redeployment only immediately after it finishes transferring a patient to a hospital. Ambulances that are idle at the bases or moving to different locations are not considered for redeployment. The appealing aspect of this redeployment policy is that it minimizes disturbance to the crews, but exploring the benefit of additional redeployments is important and we do so in our computational experiments. To capture the decisions, we let $x_{ib}(s) = 1$ if we redeploy ambulance i to base b when the state of the system is s, and 0 otherwise. Letting $\mathcal{B}$ be the set of ambulance bases and $x(s) = \{x_{ib}(s) : i \in \mathcal{R}(s),\ b \in \mathcal{B}\}$, the set of feasible decisions can be written as

$$\mathcal{X}(s) = \Big\{x(s) \in \{0,1\}^{|\mathcal{R}(s)|\times|\mathcal{B}|} : \sum_{b\in\mathcal{B}} x_{ib}(s) = 1 \quad \forall\, i \in \mathcal{R}(s)\Big\}.$$

The constraints in this definition simply state that the ambulance considered for redeployment has to be redeployed to one base. If the event $e(s)$ is not of the form "ambulance i finishes at hospital," then we have $\mathcal{R}(s) = \varnothing$, which implies that $\mathcal{X}(s) = \varnothing$ as well. In this case, we simply allow the system to evolve naturally without any interference from the decision-maker.

An important implication of our definition of $\mathcal{R}(s)$ is that the cardinality of $\mathcal{X}(s)$ is small so that an optimization problem that takes place over this feasible set can be solved by enumerating over all feasible decisions. In other words,

since we consider only one ambulance at a time for deployment, we can try each base one by one to find out to which base an ambulance should be redeployed. In contrast, if we considered K ambulances simultaneously for redeployment, then the number of possible redeployment decisions would be $|\mathcal{B}|^K$, which can get quite large for moderate values of K and $|\mathcal{B}|$. This is a simplification that we make to avoid the combinatorial aspects of the problem and focus more on its dynamic and stochastic nature. Having said that, $K = 2$ is easily within our computational grasp if we restrict attention to sensible relocations of short range, and that will be the subject of future research.

Not considering the ambulances that are in transit as available for redeployment may have some undesirable affects. For example, we may decide to redeploy an ambulance at the northeast corner of the city to a base at the southwest corner. As soon as this redeployment starts, an ambulance in the southwest corner of the city may be available for deployment and this ambulance may be redeployed to a base at the northeast corner. It may be better to reroute the first ambulance to the northeast base and redeploy the second ambulance to the southwest base, although some results of Zhang et al. (2008) suggest that this is not universally true. The controls that we adopt in this paper miss this kind of opportunity to improve performance. Nevertheless, our computational experiments indicate that significant improvements are possible even when we ignore such opportunities, and one would expect that dispatchers monitoring the EMS system would take advantage of any obvious opportunities.

The set $\mathcal{B}$ may include additional locations, other than ambulance bases, at which ambulance crews can park and rest. As a result, our approach allows parking and resting at arbitrary locations as long as the list of possible locations

is not too large. In our computational experiments, work with as many as 88 locations.

2.2.3 Fundamental Dynamics

Call arrivals are generated across the region $R \subset \mathbb{R}^2$ according to a Poisson point process with a known arrival intensity $\{\Lambda(t, x, y) : t \geq 0,\ (x, y) \in R\}$. As mentioned above, we have a fixed policy for serving calls, but our general approach does not depend on the particular form of this policy. If there are no available ambulances to serve a call, then the call is placed into a waiting queue. An ambulance serving a call proceeds through a sequence of events, including arriving at the scene, treating the patient, transporting and handing over the patient at the hospital. The main source of uncertainty in this call service cycle are the times spent between events. We assume that probability distributions for all of the activity durations are known.

Besides these sources of randomness, the major driver of dynamics is dispatching. As a result, the complete trajectory of the system is given by $\{(s_k, x_k) : k = 1, 2, \ldots\}$, where s_k is the state of the system at the time of the kth event and x_k is the decision (if any) made by the dispatcher when the state of the system is s_k. We capture the dynamics of the system symbolically by

$$s_{k+1} = f(s_k, x_k, \omega(s_k, x_k)),$$

where $\omega(s_k, x_k)$ is a random element of an appropriate space encapsulating all the sources of randomness described above and $f(\cdot, \cdot, \cdot)$ is the transfer function. One way to visualize $\omega(s_k, x_k)$ is that there is a stochastic process corresponding to call arrivals and each ambulance. The stochastic process corresponding to call

arrivals keeps track of when calls arrive, along with the location and priority of each call. The stochastic processes corresponding to ambulances that are in transit keep track of the residual travel times. Similarly, the stochastic processes corresponding to ambulances that are serving calls at call scenes or at a hospital keep track of the residual service times. The state s_k and action x_k contain all of the information necessary to deduce the probability distributions for residual travel and service times. Therefore, $\omega(s_k, x_k)$ captures the time and type of the first event in the superposition of all stochastic processes.

2.2.4 Transition Costs

Along with a transition from state s_k to s_{k+1} through decision x_k, we incur a cost $c(s_k, x_k, s_{k+1})$. In our implementation, letting Δ be a fixed threshold response time that is on the order of 8 minutes, we use the transition cost function

$$c(s_k, x_k, s_{k+1}) = \begin{cases} 1 & \text{if the event } e(s_{k+1}) \text{ is of the form "ambulance } i \text{ arrives} \\ & \quad \text{at scene of call } j\text{," call } j \text{ is urgent and the response} \\ & \quad \text{time exceeds } \Delta \\ 0 & \text{otherwise.} \end{cases} \tag{2.1}$$

This cost function counts the number of high-priority calls whose response times exceed Δ. We are interested in the performance of the system over the finite planning horizon $[0, T]$, so let $c(s, \cdot, \cdot) = 0$ whenever $\tau(s) > T$. In our implementation, T corresponds to two weeks. When the state of the system is s_k and we make the decision x_k, the transition cost $c(s_k, x_k, s_{k+1})$ is still a random variable since its value depends on s_{k+1}. This does not create any difficulty since

the Markov decision process framework allows transition costs that depend on the states before and after the transition.

An appealing aspect of our transition cost function is its simplicity. Furthermore, most contracts and performance benchmarks in the EMS industry are formulated in terms of the percentage of calls that are reached within a time standard, and our transition cost function is in alignment with this tradition. However, this transition cost function does not distinguish between an urgent call whose response time exceeds Δ by one second or by one hour. As a result, it is conceivable that a call might be left unattended for a long period of time. This is not a huge concern in our approach since high-priority calls are served in first-in-first-out order within a priority level, but there may still be low-priority calls that are left unattended for a long period of time. Recent research by Erkut et al. (2008) incorporates medical outcomes into the objective function, and while we do not take that path here, our framework is general enough to do so.

2.2.5 Objective Function and Optimality Equation

A policy is a mapping from the state space to the action space, prescribing which action to take for each possible state of the system. Throughout the paper, we use $\mu(s) \in \mathcal{X}(s)$ to denote the action prescribed by policy μ when the state of the system is s. In other words, $\mu(s)$ is the action taken in state s given that we use policy μ. If we follow policy μ, then the state trajectory of the system $\{s_k^\mu : k = 1, 2, \ldots\}$ evolves according to $s_{k+1}^\mu = f(s_k^\mu, \mu(s_k^\mu), \omega(s_k^\mu, \mu(s_k^\mu)))$ and the discounted total expected cost incurred by starting from initial state s is given

by

$$J^{\mu}(s) = \mathbb{E}\Big[\sum_{k=1}^{\infty} \alpha^{\tau(s_k^{\mu})}\, c(s_k^{\mu}, \mu(s_k^{\mu}), s_{k+1}^{\mu}) \mid s_1^{\mu} = s\Big], \tag{2.2}$$

where $\alpha \in [0, 1)$ is a fixed discount factor. The expectation in the expression above involves the random variables $\{s_k^{\mu} : k = 1, 2, \ldots\}$ and $\tau(s_k^{\mu})$ is the time at which the system visits state s_k^{μ}. The policy μ^* that minimizes the discounted total expected cost can be found by computing the value function through the optimality equation

$$J(s) = \min_{x \in \mathcal{X}(s)} \left\{ \mathbb{E}\Big[\, c(s, x, f(s, x, \omega(s, x))) + \alpha^{\tau(f(s,x,\omega(s,x))) - \tau(s)}\, J(f(s, x, \omega(s, x)))\Big] \right\} \tag{2.3}$$

and letting $\mu^*(s)$ be the minimizer of the right side above; see Bertsekas and Shreve (1978). The difficulty with the optimality equation above is that the number of possible values for the state variable is uncountable. Even if we are willing to discretize the state variable to obtain a finite state space, the state variable is still a high-dimensional vector and solving the optimality equation (2.3) through classical dynamic programming approaches is computationally very difficult. In the next two sections, we propose a method to construct tractable approximations to the value function.

The discount factor in (2.3) implies that we minimize the total expected *discounted* number of calls that are not reached within the time threshold, though one is more likely to be interested in the total expected *undiscounted* number of calls. The discount factor is an important computational device when we move to the ADP framework in the next section. In particular, our value function approximations invariably have some error, and the role of the discount factor is to put less emphasis on the value of a future state, as predicted by the

value function approximation, when compared with the immediate transition cost. It is quite common in the ADP literature to use a discounted cost formulation when an undiscounted cost formulation may reflect the objectives of the decision-maker more accurately. For example, Farias and Van Roy (2004) and Farias and Van Roy (2006) use ADP to construct strategies for playing Tetris. The authors use a discounted cost formulation, but the objective of the player is clearly to maximize the undiscounted total expected score. Crites and Barto (1996) use a continuous-time discounted cost formulation in an application targeted at minimizing elevator wait times, when the real performance measure of interest is the average wait time per person. Singh and Bertsekas (1996) develop an ADP method for channel allocation in cellular telephone systems. The objective of the decision-maker is to minimize the rate of blocked calls per unit time, whereas the authors use a formulation that minimizes the discounted number of blocked calls.

2.3 Approximate Dynamic Programming

The ADP approach that we use to construct approximations to the value function is closely related to the traditional policy iteration algorithm in the Markov decision processes literature. We begin with a brief description of the policy iteration algorithm. Throughout the rest of the paper, the greedy policy induced by an arbitrary function $\hat{J}(\cdot)$ refers to the policy that takes a decision in the set

$$\underset{x \in \mathcal{X}(s)}{\operatorname{argmin}} \left\{ \mathbb{E}\Big[c(s, x, f(s, x, \omega(s, x))) + \alpha^{\tau(f(s,x,\omega(s,x)))-\tau(s)} \, \hat{J}(f(s, x, \omega(s, x))) \Big] \right\} \quad (2.4)$$

whenever the state of the system is s. Finding the optimal solution to the problem above requires computing potentially difficult expectations. We use Monte

Carlo simulation to compute such expectations and we elaborate on this issue in our discussion of Approximate Policy Iteration. If the state variable took finitely many values, then the optimal policy could be obtained by using the following policy iteration algorithm.

Policy Iteration

Step 1. Initialize the iteration counter n to 1 and initialize $J^1(\cdot)$ arbitrarily.

Step 2. (Policy improvement) Let μ^n be the greedy policy induced by $J^n(\cdot)$.

Step 3. (Policy evaluation) Let $J^{n+1}(\cdot) = J^{\mu^n}(\cdot)$, where $J^{\mu^n}(s)$ denotes the expected discounted cost incurred when starting from state s and using policy μ^n, as given in (2.2).

Step 4. Increase n by 1 and go to Step 2.

Even if we were willing to discretize the state variable to obtain a finite state space, since the state variable is a high-dimensional vector, the number of possible values for the state variable would be far too large to apply the policy iteration algorithm above directly. We try to overcome this difficulty by using value function approximations of the form

$$J(s, r) = \sum_{p=1}^{P} r_p \, \phi_p(s). \tag{2.5}$$

In the expression above, $r = \{r_p : p = 1, \ldots, P\}$ are tunable parameters and $\{\phi_p(\cdot) : p = 1, \ldots, P\}$ are fixed basis functions. The challenge is to construct the basis functions and tune the parameters so that $J(\cdot, r)$ is a good approximation to the solution to the optimality equation (2.3). To achieve this, each basis function $\phi_p(\cdot)$ should capture some essential information about the solution to the optimality equation. In Section 2.4, we describe one set of basis functions that work well for our application. Once a good set of basis functions is available,

we can use the following approximate version of the policy iteration algorithm to tune the parameters $\{r_p : p = 1, \ldots, P\}$.

Approximate Policy Iteration

Step 1. Initialize the iteration counter n to 1 and initialize $r^1 = \{r_p^1 : p = 1, \ldots, P\}$ arbitrarily.

Step 2. (Policy improvement) Let μ^n be the greedy policy induced by $J(\cdot, r^n)$.

Step 3. (Policy evaluation through simulation) Simulate the trajectory of policy μ^n over the planning horizon $[0, T]$ for Q replications. Let $\{s_k^n(q) : k = 1, \ldots, K(q)\}$ be the state trajectory of policy μ^n in replication q and $C_k^n(q)$ be the discounted cost incurred by starting from state $s_k^n(q)$ and following policy μ^n in replication q.

Step 4. (Projection) Compute the tunable parameters at the next iteration as

$$r^{n+1} = \underset{r \in \mathbb{R}^P}{\operatorname{argmin}} \left\{ \sum_{q=1}^{Q} \sum_{k=1}^{K(q)} \left[C_k^n(q) - J(s_k^n(q), r) \right]^2 \right\}.$$

Step 5. Increase n by 1 and go to Step 2.

In Step 3 of approximate policy iteration, we use simulation to evaluate the expected discounted cost incurred by policy μ^n. Therefore, $\{C_k^n(q) : k = 1, \ldots, K(q),\ q = 1, \ldots, Q\}$ are the sampled cost trajectories of the system under policy μ^n. In Step 4, we tune the parameters $r = \{r_p : p = 1, \ldots, P\}$ so that the value function approximation $J(\cdot, r)$ provides a good fit to the sampled cost trajectories.

There is still one computational difficulty in the approximate policy iteration algorithm. When simulating the trajectory of policy μ^n in Step 3, we need to solve an optimization problem of the form (2.4) to find the action taken by

the greedy policy induced by $J(\cdot, r^n)$. This optimization problem involves an expectation that is difficult to compute. As mentioned earlier, we use Monte Carlo simulation to overcome this difficulty. In particular, if the state of the system is s and we want to find the action taken by the greedy policy induced by $J(\cdot, r^n)$ in this state, then we enumerate over all decisions in the feasible set $\mathcal{X}(s)$. Enumerating over all feasible decisions in the set $\mathcal{X}(s)$ is possible since the cardinality of this set is equal to the number of ambulance bases, which is at most 88 in our experiments. Starting from state s and taking decision x, we simulate the trajectory of the system until the next event and this provides a sample of $f(s, x, \omega(s, x))$. Since this simulation is only until the time of the next event, it is very quick to run. In particular, sampling a realization of $\omega(s, x)$ involves sampling the residual interarrival time for the next call, and at most N residual travel times and N residual service times. In this case, we obtain a sample of $f(s, x, \omega(s, x))$ by generating the state of the system at the next event time. By simulating multiple samples, we estimate the expectation

$$\mathbb{E}\Big[c(s, x, f(s, x, \omega(s, x))) + \alpha^{\tau(f(s,x,\omega(s,x)))-\tau(s)}\, J(f(s, x, \omega(s, x)), r^n)\Big]$$

through a sample average. Once we estimate the expectation above for all $x \in \mathcal{X}(s)$, we choose the decision that yields the smallest value and use it as the decision taken by the greedy policy induced by $J(\cdot, r^n)$ when the state of the system is s. This approach is naturally subject to sampling error, but it provides good performance in practice. In our computational experiments, we use 10 to 25 replications to estimate the expectation above. This is a small number of replications, but we use common random numbers when estimating the expectation for different actions. This allows us to quickly identify whether the expectation corresponding to a particular action is smaller than the expectation corresponding to another action.

Proposition 6.2 in Bertsekas and Tsitsiklis (1996) provides a performance guarantee for the approximate policy iteration algorithm. Their result is for finite state spaces, but it is easily extended to infinite state spaces. This result provides theoretical support for the approximate policy iteration algorithm, but its conditions are difficult to verify in practice. In particular, the result assumes that we precisely know the error induced by using regression to estimate the discounted total expected cost of a policy, and it assumes that expectations are computed exactly rather than via sampling as in our case. For this reason, we do not go into the details of this result and refer the reader to Bertsekas and Tsitsiklis (1996) for further details.

2.4 Basis Functions

In this section, we describe the basis functions $\{\phi_p(\cdot) : p = 1, \ldots, P\}$ that we use in our value function approximations. We use six basis functions, some of which are based on the queueing insights developed in Restrepo et al. (2009).

1. Baseline

The first basis function is $\phi_1(s) = 1$. When multiplied by r_1 in (2.5), this basis function shifts the value function approximation to any desired level.

2. Unreachable Calls

The second basis function computes the number of calls in the call waiting queue for which an ambulance assignment has been made, but the ambulance

will not reach the call within the threshold response time. This quantity is easily computable when the travel times are deterministic, which is the case in our implementation. To see this, let $\mathbf{1}(\cdot)$ be the indicator function and $d(\ell_1, \ell_2)$ be the travel time between locations ℓ_1 and ℓ_2. Given that the state of the system is s, we can count the number of ambulances that are assigned to the jth call in the call waiting queue but will not reach the call within the threshold response time via

$$G_j(s) = \sum_{i=1}^{N} \mathbf{1}(\sigma_i = \text{“going to scene of call } j\text{”})\, \mathbf{1}(t_i + d(\ell_i, d_i) - \zeta_j \geq \Delta).$$

The expression above first checks each ambulance to see if there is one whose status is “going to scene of call j.” If there is one such ambulance, then it checks whether the time at which this ambulance arrives at its destination exceeds the arrival time of the jth call by more than Δ time units. The second basis function can now be written as

$$\phi_2(s) = \sum_{j=1}^{M} G_j(s).$$

3. Uncovered Call Rate

The third basis function captures the rate of call arrivals that cannot be reached on time by any available ambulance. To define this basis function precisely, we need additional notation. Recall that calls arrive across the region $R \subset \mathbb{R}^2$ according to a Poisson point process with arrival intensity $\{\Lambda(t, x, y) : t \geq 0,\ (x, y) \in R\}$. Partition the region R into L subregions and associate a representative point or center of mass location ρ_l with each subregion l.

The coverage of subregion l is the number of available ambulances that can reach the center of mass within the threshold time standard. We let $\mathcal{A}(s)$ be

the set of available ambulances when the state of the system is s. This set includes the ambulances whose status is either "idle at base" or "returning to base." Given that the system is in state s, we let $\hat{\ell}_i(s)$ be the location of ambulance i at time $\tau(s)$. In this case, the coverage of subregion l can be written as

$$N_l(s) = \sum_{i \in \mathcal{A}(s)} \mathbf{1}(d(\hat{\ell}_i(s), \rho_l) \leq \Delta).$$

Using $\Lambda_l(t)$ to denote the total rate of call arrivals in subregion l at time t, we can compute the rate of call arrivals that are not covered by any available ambulance by

$$\phi_3(s) = \sum_{l=1}^{L} \Lambda_l(\tau(s))\, \mathbf{1}(N_l(s) = 0).$$

The values $\{\Lambda_l(t) : t \geq 0\}$ in the expression above are part of the problem data and can be computed in advance.

4. Missed Call Rate

The previous two basis functions respectively capture calls already received that we know we cannot reach on time and the rate of arriving calls that cannot be reached on time because they are too far from any available ambulance. We could also fail to reach a call on time due to queueing effects from ambulances being busy with other calls. The fourth basis function represents an attempt to capture this effect. This basis function is of the form

$$\phi_4(s) = \sum_{l=1}^{L} \Lambda_l(\tau(s))\, P_l(s),$$

where $P_l(s)$ is the probability that all ambulances that could reach a call in subregion l on time are busy with other calls. We estimate $\{P_l(s) : l = 1, \ldots, L\}$ by

treating the call service processes in different subregions as Erlang loss systems. In Erlang loss systems, calls arriving when all servers are busy are lost. In particular, in an Erlang loss system with arrival rate λ, service rate μ and n servers, the steady state probability of losing a call is given by

$$\mathcal{L}(\lambda, \mu, n) = \frac{(\lambda/\mu)^n/n!}{\sum_{k=0}^{n}(\lambda/\mu)^k/k!}.$$

In our EMS system, a call that arrives when all ambulances are busy is queued and served as ambulances become free, but the time threshold is almost always missed for such calls, so counting them as lost seems reasonable. The issue that such calls impose some load on the true system, but are discarded in an Erlang loss system creates a slight mismatch between our EMS system and the Erlang loss system, but our computational experiments show that this basis function is still highly effective.

To characterize the Erlang loss system for subregion l, given that the state of the system is s, we need to specify the number of servers, along with the arrival and service rates. Let $\mathcal{N}_l(s)$ be the set of available ambulances that can serve a call in subregion l within the threshold response time so that

$$\mathcal{N}_l(s) = \{i \in \mathcal{A}(s) : d(\hat{\ell}_i(s), \rho_l) \leq \Delta\}.$$

We use $|\mathcal{N}_l(s)|$ as the number of servers in the Erlang loss system for subregion l. Let $\mu_l(t)$ be the service rate in the loss system. This is the rate at which an ambulance can serve a call at time t in subregion l. It is difficult to come up with a precise value for $\mu_l(t)$. It primarily depends on the time spent at the scene of a call and any transfer time at the hospital, since the travel times are usually small relative to these quantities. In our implementation, we use historical data to estimate the time spent at the call scenes and the hospital, and add a small padding factor to capture travel times. Finally, let $\lambda_l(s)$ be the rate of call arrivals

that should be served by ambulances in the set $\mathcal{N}_l(s)$. Coming up with a value for $\lambda_l(s)$ is even more difficult than devising a value for $\mu_l(t)$. One option is to let $\lambda_l(s) = \Lambda_l(\tau(s))$, which is the rate of call arrivals at time $\tau(s)$ in subregion l. However, ambulances in the set $\mathcal{N}_l(s)$ serve calls other than those in subregion l. To attempt to capture this, let

$$\lambda_l(s) = \sum_{i \in \mathcal{N}_l(s)} \sum_{k=1}^{L} \Lambda_k(\tau(s)) \, \mathbf{1}(d(\hat{\ell}_i(s), \rho_k) \leq \Delta), \tag{2.6}$$

so that $\lambda_l(s)$ reflects the total call arrival rate in subregions that are close to any of the ambulances in the set $\mathcal{N}_l(s)$. We then use the approximation $P_l(s) \approx \mathcal{L}(\lambda_l(s), \mu_l(\tau(s)), |\mathcal{N}_l(s)|)$.

There are at least two shortcomings in our estimate of $\lambda_l(s)$ and the approximation that we use for $P_l(s)$. First, there is double counting in the estimate of $\lambda_l(s)$. In particular, if two ambulances $i_1, i_2 \in \mathcal{N}_l(s)$ can both reach subregion l' within the time threshold, then the summation for $\lambda_l(s)$ counts $\Lambda_{l'}(\tau(s))$ twice. Second, $\Lambda_{l'}(\tau(s))$ could be counted in the demand rates for multiple subregions. To be more precise, if there are three subregions l_1, l_2, l' and two ambulances $i_1 \in \mathcal{N}_{l_1}(s)$, $i_2 \in \mathcal{N}_{l_2}(s)$ such that both i_1 and i_2 can reach subregion l' within the time threshold, then the summations for $\lambda_{l_1}(s)$ and $\lambda_{l_2}(s)$ both count $\Lambda_{l'}(\tau(s))$. These two effects typically cause $\sum_{l=1}^{L} \lambda_l(s) > \sum_{l=1}^{L} \Lambda_l(\tau(s))$. However, any ambulance i may also be required to serve calls in subregions l where $d(\hat{l}_i(s), \rho_l) > \Delta$. This effect causes our approximation to underestimate the arrival rate $\lambda_l(s)$ and hence can cause $\sum_{l=1}^{L} \lambda_l(s) < \sum_{l=1}^{L} \Lambda_l(\tau(s))$. To overcome these problems, we modify the call arrival rates by a factor κ. In particular, we use the call arrival rate $\kappa \, \Lambda_l(\tau(s))$ in (2.6) instead of $\Lambda_l(\tau(s))$ so that we may roughly have $\sum_{l=1}^{L} \lambda_l(s) = \sum_{l=1}^{L} \kappa \, \Lambda_l(\tau(s))$. We find a good value for κ through preliminary experimentation. As we demonstrate in our computational experi-

ments, it is not necessarily the case that the best choice of κ lies in $(0, 1)$.

5. Future Uncovered Call Rate

We do not consider ambulances that are already in transit as available for redeployment. Therefore, from the perspective of covering future calls, the destinations of moving ambulances are as important as their current locations. This is the motivation underlying the fifth and sixth basis functions.

Our fifth basis function parallels the third one, but it replaces the current locations of ambulances by their destinations. In other words, the definition of this basis function is identical to that of $\phi_3(\cdot)$, but the configuration of the ambulances that we use to compute $N_l(s)$ is not the current one, but an estimated future configuration that is obtained by letting all ambulances in transit reach their destinations and all stationary ambulances remain at their current locations. Given that the system is in state $s = (\tau, e, A, C)$ with $A = (a_1, \ldots, a_N)$ and $a_i = (\sigma_i, \ell_i, d_i, t_i)$, we define a new state $\vec{s}(s) = (\tau + 1/\sum_{l=1}^{L} \Lambda_l(\tau), e, \vec{A}, C)$ with $\vec{A} = (\vec{a}_1, \ldots, \vec{a}_N)$ and $\vec{a}_i = (\vec{\sigma}_i, d_i, d_i, \tau + 1/\sum_{l=1}^{L} \Lambda_l(\tau))$, where $\vec{\sigma}_i$ is the status of ambulance i when it reaches its destination d_i. In this case, the fifth basis function is

$$\phi_5(s) = \sum_{l=1}^{L} \Lambda_l(\tau(\vec{s}(s)))\, \mathbf{1}(N_l(\vec{s}(s)) = 0).$$

In the expression above, the time $\tau(\vec{s}(s)) = \tau + 1/\sum_{l=1}^{L} \Lambda_l(\tau)$ is used as an approximation for the expected time of the next call arrival. The next call may arrive before or after the ambulances actually reach their destinations, but we heuristically use the time $\tau(\vec{s}(s))$ simply to look into the future. The idea is that the estimated future configuration of the ambulances $\vec{A}$ is more likely to hold at

the future time $\tau(\vec{s}(s))$ than at the current time $\tau(s)$.

We plug $J(\cdot, r)$ into the right side of (2.4) to find the action taken by the greedy policy induced by $J(\cdot, r)$. Since $J(\cdot, r)$ is computed at the future state $f(s, x, \omega(s, x))$, we need to compute $\phi_5(\cdot)$ at this future state as well. To compute $\phi_5(\cdot)$ at $f(s, x, \omega(s, x))$, we need to compute the state $\vec{s}(f(s, x, \omega(s, x)))$ and this state peeks even further ahead of the future state $f(s, x, \omega(s, x))$.

6. Future Missed Call Rate

The sixth basis function parallels $\phi_4(\cdot)$ in that it captures the rate of calls that will likely be lost due to queueing congestion. As with the fifth basis function, it uses an estimated future configuration of the ambulances. It is defined as

$$\phi_6(s) = \sum_{l=1}^{L} \Lambda_l(\tau(\vec{s}(s)))\, P_l(\vec{s}(s)).$$

Our basis functions are relatively complex and this complexity, coupled with the expectation in problem (2.4), could make computing the greedy policy induced by the value function approximation $J(\cdot, r)$ quite difficult. Nevertheless, since the cardinality of the feasible set $\mathcal{X}(s)$ is small, we can solve an optimization problem that takes place over this feasible set simply by enumerating over all possible decisions. For very large feasible sets, this approach would not be practical. We close this section with a brief note on the complexity of computing our basis functions at a particular state s. The effort required to compute $\phi_1(s)$ is clearly $O(1)$. We can compute $G_j(s)$ in $O(N)$ time, which implies that the effort required to compute $\phi_2(s)$ is $O(MN)$. It is possible to compute $N_l(s)$ in $O(N)$ time so that we can compute $\phi_3(s)$ in $O(NL)$ time. We can compute $\mathcal{N}_l(s)$ in $O(N)$ time. We always have $|\mathcal{N}_l(s)| \leq N$, so we can compute $\lambda_l(s)$

in $O(NL)$ time. Since the computation of the Erlang loss function with $|\mathcal{N}_l(s)|$ servers can be done in $O(N)$ time, we can compute $P_l(s)$ in $O(N+NL) = O(NL)$ time. Therefore, the effort required to compute $\phi_4(s)$ is $O(NL^2)$. The effort for computing the future state $\vec{s}(s)$ is small when compared to the basis functions. Therefore, we can compute $\phi_5(s)$ and $\phi_6(s)$ in essentially the same time as $\phi_3(s)$ and $\phi_4(s)$. As a result, all of our basis functions can be computed in $O(MN + NL^2)$ time. We give specific timing results in our computational results.

2.5 Computational Results

We now present computational results for two EMS systems. The first EMS system belongs to the city of Edmonton, Alberta in Canada and was also studied in Ingolfsson et al. (2003). We are not able to disclose the identity of the second EMS system due to confidentiality agreements.

2.5.1 Experimental Setup

In this section, we give a description of the data sets along with our assumptions.

1. The City of Edmonton

The data we use for the city of Edmonton corresponds to the data set used in the computational experiments in Ingolfsson et al. (2003). The city has a population

of 730,000 and covers an area of around 40×30 km^2. The EMS system has 16 ambulances, 11 bases and 5 hospitals. We assume that all ambulances are of the same type and operate all day. An ambulance, upon arriving at a call scene, treats the patient for an exponentially distributed amount of time with mean 12 minutes. After treating the patient at the call scene, the ambulance transports the patient to a hospital with probability 0.75. The probability distribution for the hospital chosen is inferred from historical data. The time an ambulance spends at the hospital has a Weibull distribution with mean 30 minutes and standard deviation 13 minutes. The turn-out time is assumed to be 45 seconds, i.e., if an ambulance crew is at a base when notified of a call, then it takes 45 seconds to get on the road. An ambulance crew already on the road does not incur the turn-out time. A call that is not served within 8 minutes is interpreted as missed. The road network that we use models the actual road network on the avenue level. There are 252 nodes and 934 arcs in the road network. Travel times are deterministic and do not depend on the time of day.

The data we had access to were not sufficient to develop a detailed model of call arrivals, so we proceed with a representative model that does not exactly correspond to the actual distribution of calls over the city of Edmonton. The model maintains a constant overall arrival rate, but the distribution of the location of calls changes in time. We divide the city into 20×17 subregions and assume that the rate of call arrivals in subregion l at time t is given by

$$\Lambda_l(t) = \Lambda\big[\gamma_l + \beta_l \sin(2\pi t/24)\big],$$

where t is measured in hours. In the expression above, Λ, γ_l and β_l are fixed parameters that satisfy $\Lambda \geq 0$, $\gamma_l \geq |\beta_l|$, $\sum_{l=1}^{340} \gamma_l = 1$ and $\sum_{l=1}^{340} \beta_l = 0$. We have $\sum_{l=1}^{340} \gamma_l + \sum_{l=1}^{340} \beta_l \sin(2\pi t/24) = 1$ so that we can interpret Λ as the total call arrival rate into the system and $\gamma_l + \beta_l \sin(2\pi t/24)$ as the probability that a call arriving

at time t falls in subregion l. If $\beta_l > 0$, then the peak call arrival rate in subregion l occurs at hours $\{6, 30, 54, \ldots\}$, whereas if $\beta_l < 0$, then the peak call arrival rate in subregion l occurs at hours $\{18, 42, 66, \ldots\}$. The average call arrival rate over a day in subregion l is $\Lambda\gamma_l$. We estimated Λ and γ_l using historical data and Λ came out to be about 4 calls per hour. We chose appropriate values of β_l so that we have higher call arrival rates in the business subregions early in the day and higher call arrival rates in the residential subregions later in the day.

2. The Second City

The population of the second city is more than five times that of the city of Edmonton and its size is around 180×100 km^2. The EMS system includes up to 97 ambulances operating during peak times, 88 bases and 22 hospitals. The turnout times, call-scene times and hospital-transfer times are comparable to those in Edmonton, but are chosen to be representative rather than realistic to protect confidentiality. The destination hospital for a call depends on the location of the call. Calls originating at a given location are transported to any of a small set of hospitals, usually no more than two or three out of the 22 hospitals in the system. The corresponding probabilities are inferred from historical data. The road network that we use models the actual network on the avenue level and there are 4,955 nodes and 11,876 arcs.

Our call arrival model is quite realistic. The data were collected from one year of operations of the EMS system and consist of aggregated counts of calls for each hour of the week and for each of 100×100 geographic zones in the city. Due to the irregular shape of the metropolitan area, roughly 80% of these zones have zero total call counts and do not intervene in the dynamics. From

the remaining 20%, a significant number of zones have very low hourly counts of at most five calls. Therefore, it was necessary to apply a smoothing procedure for the lowest intensity zones so as to reduce the sampling noise. We classified the zones into a few groups according to their average intensity over the week. For the lowest intensity groups, we computed a total intensity for each hour and then distributed this total intensity uniformly among the zones in this group. In this way, we were able to obtain an intensity model that combined a uniform low intensity background with accurate counts on the highest intensity zones. In the end, the average call arrival rate is 570 calls per day and fluctuates on any day of the week from a low of around 300 calls per day to a high of around 750 calls per day. These figures represent a modest departure from the true figures to protect confidentiality.

For both data sets, the simulation horizon is 14 days. We use a discount factor of $\alpha = 0.8$, but our results are relatively insensitive to the choice of the discount factor as long as it is not too close to one. We initialize r^1 to zero in Step 1 of the approximate policy iteration algorithm and use $Q = 30$ replications in Step 3. A few setup runs indicated that setting $Q = 30$ provides a reasonable balance between computational burden and stable performance. We also tried using different values for κ in the fourth and sixth basis functions. We varied κ over the interval $[0, 3]$ and setting $\kappa = 0.1$ gave the best results for the city of Edmonton, whereas setting $\kappa = 1.6$ gave the best results for the second city. These setup runs also indicated that tuning the value of κ is quite important for obtaining good performance from our ADP approach.

2.5.2 Baseline Performance

The goal of our first set of computational experiments is to give a feel for how our ADP approach compares with several benchmark strategies. In Figure 2.1, we begin by showing the performance of our ADP approach on the city of Edmonton over 25 iterations. The horizontal axis in this figure gives the iteration number in the approximate policy iteration algorithm, whereas the vertical axis gives the expected percentage of calls not reached within the threshold response time. In other words, each data point in Figure 2.1 gives the expected percentage of calls missed by the greedy policy induced by the value function approximation at a particular iteration. We compute the expected percentage of missed calls by using the undiscounted numbers of calls. Since each iteration of the approximate policy iteration algorithm requires simulating the performance of the greedy policy for 30 replications, it is straightforward to estimate the expected percentage of missed calls at each iteration by using a sample average. From Figure 2.1, we observe that the apparent-best policy is obtained at the third iteration and this policy yields an expected percentage of missed calls of about 25.6%. To test the performance of this policy more carefully, we simulate its performance for an independent set of 400 replications. From these 400 replications, the expected percentage of missed calls is estimated to be $25.5\% \mp 0.1\%$, where $\mp 0.1\%$ is a 95% confidence interval.

We use two benchmark strategies. The first benchmark strategy is the myopic policy, which is obtained by letting $r_p = 0$ for all $p = 1, \ldots, P$ in (2.5) and using the greedy policy induced by this value function approximation. Step 1 of the approximate policy iteration algorithm initializes r^1 to zero, so the first data point Figure 2.1 naturally gives the performance of the myopic policy. The

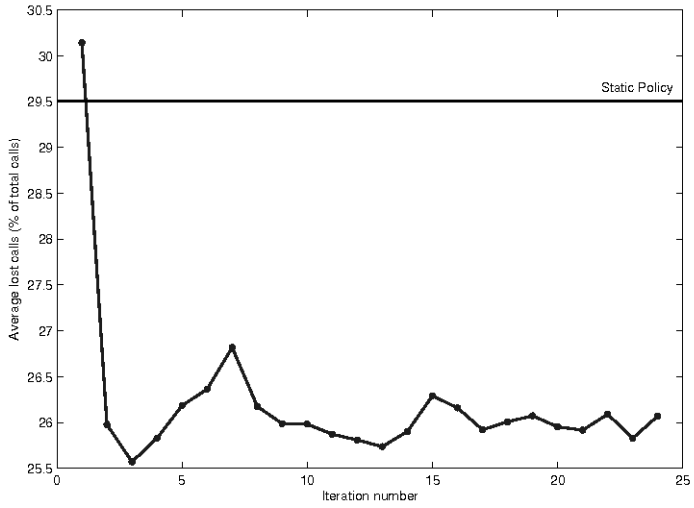

Figure 2.1: Performance of our ADP approach on the city of Edmonton.

expected percentage of calls missed by the myopic policy is about 30.2%.

The second benchmark strategy that we use is the static policy, which preassigns a base to each ambulance and redeploys an ambulance back to its preassigned base whenever it becomes free after serving a call. We find a good static policy by simulating the performance of the system under a large number of possible base assignments and choosing the base assignment that gives the best performance. The horizontal line in Figure 2.1 shows the performance of the best static policy we found. The expected percentage of calls missed by the best static policy is $29.5\% \mp 0.1\%$.

The best policy obtained by our ADP approach improves on the myopic and static policies respectively by 4.7% and 4.0%. These improvements are obtained without adding any extra resources and EMS managers would be very interested in obtaining these kinds of improvements by simply using their existing resources more carefully. To put the improvement figures in perspective, a quick

investigation into the data reveals that 18.6% of the overall call volume occurs in locations that are at least 8 minutes away from the ambulance bases. This implies that 18.6% of the calls would be missed even if there were always at least one ambulance available at every base. This line of reasoning ignores the fact that a call may be served by an ambulance that is on the road, but it provides a sense of a lower bound on what is achievable. Our ADP approach makes a significant step towards achieving this lower bound.

The CPU time for each iteration of our approximate policy iteration algorithm is 22 minutes. Such runtimes are acceptable given that we run the approximate policy iteration algorithm in an offline fashion to search for a good value function approximation. Once we have a good value function approximation, it takes about 45 milliseconds to make one redeployment decision by solving an optimization problem of the form (2.4). This CPU time includes enumerating over all feasible decisions and estimating the expectations through Monte Carlo samples, and is far faster than necessary for real-time operation.

Figure 2.2 shows the empirical cumulative distributions of the response times for the static (solid line) and ADP (dashed line) policies. Figure 2.2 indicates that our ADP approach not only decreases the expected percentage of missed calls, but it also shifts the entire distribution of call response times to the left. It is encouraging that the improvement in the expected percentage of missed calls is not obtained by letting a few calls wait for a very long time.

Figure 2.3 shows the performance of our ADP approach on the second city. The observations from Figure 2.3 are very similar to those from Figure 2.1. The best policy obtained by our ADP approach misses $26.9\% \mp 0.1\%$ of the calls, whereas the myopic and static policies respectively miss 29.3% and $28.8\% \mp 0.1\%$

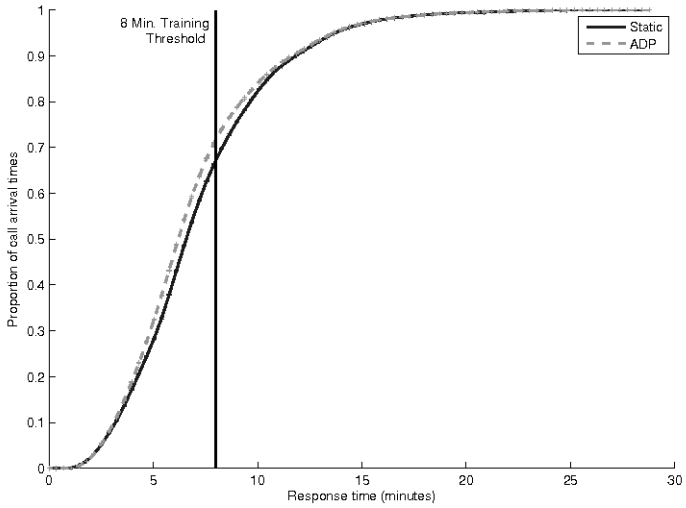

Figure 2.2: Empirical cumulative distribution of the response times for the city of Edmonton.

(as estimated through independent runs of 400 replications). The improvements are smaller for the second city than for Edmonton. Nevertheless, our ADP approach still improves on the static policy by about 2%.

2.5.3 Contributions of Different Basis Functions

Computation time can potentially be reduced if we can satisfactorily use a subset of the basis functions rather than all six of them. Therefore, it is natural to ask whether all six of the basis functions are really needed. We repeated the computational experiments described in the previous subsection by using only subsets of the basis functions. For the city of Edmonton, we are able to drop all but the fifth and sixth basis functions. By using only these two basis functions, our ADP approach identifies a policy that misses $25.4\% \mp 0.3\%$ of the calls. Recall that the best policy obtained by our ADP approach with all of the six basis

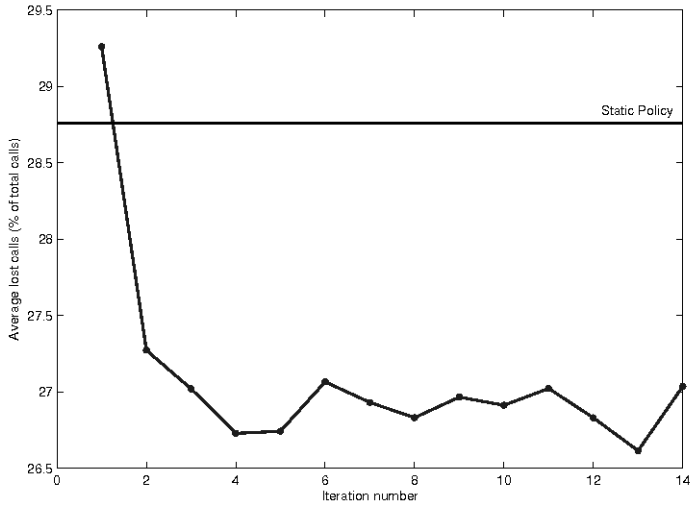

Figure 2.3: Performance of our ADP approach on the second city.

functions misses $25.5\% \mp 0.1\%$ of the calls. Therefore, the performance of our ADP approach with only the fifth and sixth basis functions is pretty close to the performance with all of the six basis functions. Dropping either the fifth or sixth basis function provides policies that perform substantially worse than the myopic policy.

When we tried to carry out a similar set of experiments on the second city, the results were somewhat mixed. For example, when we dropped the first basis function, our ADP approach immediately settled on a sequence of policies that miss about 30.1% of the calls. Noting the computational complexity analysis at the end of Section 2.4, the computational burden for the fifth and sixth basis functions is at least as large as the computational burden for the others. Given that the fifth and sixth basis functions appear to be crucial for the success of our ADP approach, we decided to keep the other basis functions in our approximation architecture as well.

To conserve space, the remainder of this section reports on our computational results only for the city of Edmonton. We carried out similar computational experiments on the second city as well and we obtained very similar results.

2.5.4 Comparison with Random Search

For a fixed set of basis functions $\{\phi_p(\cdot) : p = 1, \ldots, P\}$, a set of values for the tunable parameters $r = \{r_p : p = 1, \ldots, P\}$ characterize a value function approximation $J(\cdot, r)$ and this value function approximation induces a greedy policy. Therefore, a brute-force approach for finding a good set of values for the tunable parameters is to carry out a random search over an appropriate subset of $\mathbb{R}^P$, and use simulation to test the performance of the greedy policies induced by the different sets of values for the tunable parameters.

To implement this idea, we first use our ADP approach to obtain a good set of values for the tunable parameters. Letting $\{\hat{r}_p : p = 1, \ldots, P\}$ be this set of values, we sample $r = \{r_p : p = 1, \ldots, P\}$ uniformly over the box $\left[\hat{r}_1 - \frac{1}{2}\hat{r}_1, \hat{r}_1 + \frac{1}{2}\hat{r}_1\right] \times \ldots \times \left[\hat{r}_P - \frac{1}{2}\hat{r}_P, \hat{r}_P + \frac{1}{2}\hat{r}_P\right]$ and use simulation to test the performance of the greedy policy induced by the value function approximation $J(\cdot, r)$. We sampled 1,000 sets of values for the tunable parameters and this, in turn, provides 1,000 value function approximations. Figure 2.4 gives a histogram for the expected percentage of calls missed by the greedy policies induced by these 1,000 value function approximations. The vertical lines correspond to the expected percentage of calls missed by the best policy obtained by our ADP approach and the static policy. The figure indicates that only 1.2%

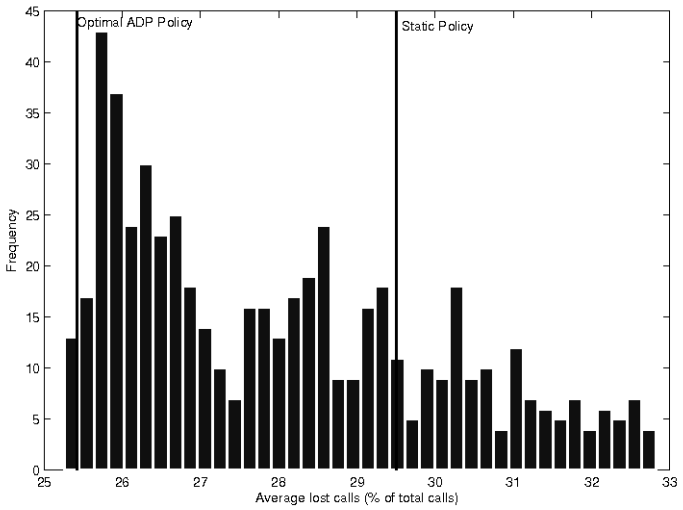

Figure 2.4: Performance of the 1,000 greedy policies obtained through random search.

of the sampled sets of values for the tunable parameters provide better performance than the best policy obtained by our ADP approach. On the other hand, 42.3% of the samples provide better performance than the static policy.

The random search procedure we use is admittedly rudimentary and one could use more sophisticated techniques to focus on the more promising areas of the search space. Nevertheless, our results indicate that when one looks at the broad landscape of the possible values for the tunable parameters, our ADP approach is quite effective in identifying good parameters. Moreover, the computation time required by the random search procedure is on the order of several days, whereas we can carry out our ADP approach in a few hours.

2.5.5 Additional Redeployments

The computational experiments up to this point allow redeployments only when an ambulance becomes free after serving a call. We now explore the possibility of improving performance by allowing additional ambulance redeployments. We define an extra event type "consider redeployment" and schedule an event of this type with a certain frequency that is detailed below. Whenever an event of this type is triggered, we consider redeploying any ambulance that is either at a base or returning to a base, so that $\mathcal{R}(s)$ can contain multiple ambulances at such times. The set $\mathcal{R}(s)$ continues to be a singleton when $e(s)$ corresponds to an ambulance becoming free after serving a call, and at all other events, $\mathcal{R}(s) = \varnothing$.

We use two methods to vary the redeployment frequency. In the first method, we equally space consider-redeployment events to obtain frequencies between 0 and 10 per hour. In the second method, the frequency of consider-redeployment events is fixed at 30 per hour, but we make a redeployment only when the estimated benefit from making the redeployment exceeds the estimated benefit from not making the redeployment by a significant margin. More precisely, letting $\epsilon \in [0, 1)$ be a tolerance margin and using $\bar{0}(s)$ to denote the $|\mathcal{R}(s)| \times |\mathcal{B}|$ dimensional matrix of zeros corresponding to the decision matrix of not making a redeployment, if we have

$$\begin{aligned}\operatorname*{argmin}_{x \in \mathcal{X}(s)} &\left\{ \mathbb{E}\Big[c(s, x, f(s, x, \omega(s, x))) + \alpha^{\tau(f(s,x,\omega(s,u)))-\tau(s)}\, J(f(s, x, \omega(s, x)), r)\Big] \right\} \\ &\leq (1 - \epsilon)\, \mathbb{E}\Big[c(s, \bar{0}, f(s, x, \omega(s, \bar{0}))) + \alpha^{\tau(f(s,\bar{0},\omega(s,\bar{0})))-\tau(s)}\, J(f(s, \bar{0}, \omega(s, \bar{0})), r)\Big],\end{aligned}$$

then we make the redeployment decision indicated by the optimal solution to the problem on the left-hand side. Otherwise, we do not make a redeployment.

Larger values of ϵ decrease the frequency of redeployments. We vary ϵ between 0.1 and 0.001.

Figure 2.5 shows the performance improvement obtained by the additional redeployments. The horizontal axis gives the frequency of the redeployments measured as the number of redeployments per ambulance per day. The vertical axis gives the percentage of missed calls. The solid (dashed) data series corresponds to the first (second) method of varying the redeployment frequency. Recall from Figure 2.1 that we miss 25.5% of the calls without making any additional redeployments. By making about six additional redeployments per ambulance per day, we can decrease the percentage of missed calls to 22.3%. Beyond this range, we reach a plateau and additional redeployments do not provide much improvement. Another important observation is that the second method tends to provide significantly better performance improvements with the same frequency of additional redeployments. For example, the second method reduces the percentage of missed calls to 23.3% with three additional redeployments per ambulance per day, whereas the first method needs eight additional redeployments to reach the same level. Therefore, it appears that making redeployments only when the value function approximation signals a significant benefit is helpful in avoiding pointless redeployments.

2.5.6 Varying Fleet Sizes

In this section, we explore the effect of the fleet size on the performance of our ADP approach. Figure 2.6 summarizes our results. The horizontal axis in this figure gives the number of ambulances in the fleet, whereas the vertical axis

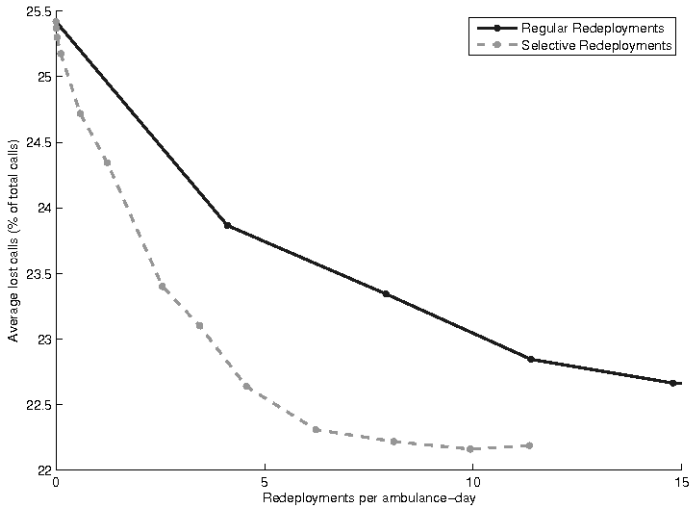

Figure 2.5: Performance of our ADP approach as a function of the frequency of additional redeployments.

gives the expected percentage of missed calls. For each fleet size, we find the best policy obtained by our ADP approach by rerunning the approximate policy iteration algorithm, and the best static policy by enumerating over a large number of possible base assignments. In Figure 2.6, the dashed data series correspond to our ADP approach, whereas the solid data series correspond to the static policy.

Our ADP approach performs consistently better than the static policy. The performance gaps between the two approaches diminish when there are too few or too many ambulances. Intuitively speaking, if the fleet size is small, then ambulances are always busy and there is little opportunity for repositioning. In this case, using a more intelligent repositioning strategy does not make much difference. On the other hand, if the fleet size is large, then there is almost always an available ambulance to respond to a call, and again, using a more intelligent

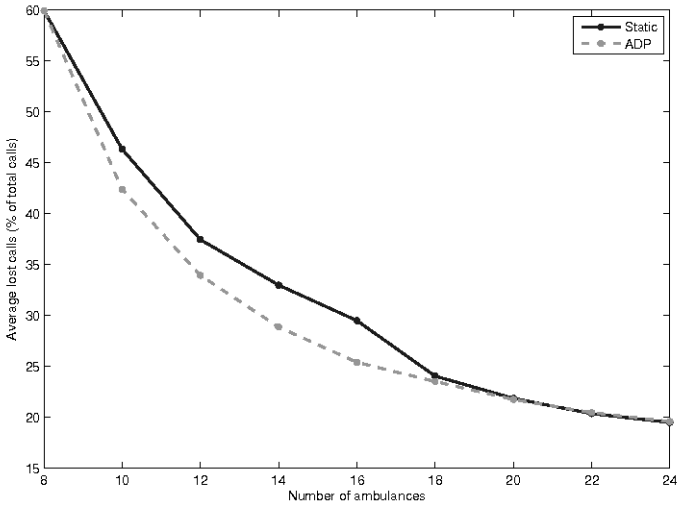

Figure 2.6: Performance of our ADP approach and the static policy for different fleet sizes.

repositioning strategy does not make much difference. Another observation from Figure 2.6 is that if our goal is to keep the percentage of missed calls below a given threshold, say 30%, then our ADP approach allows us to reach this goal with one or two fewer ambulances than the static policy. This translates into significant cost savings in an EMS system.

2.5.7 Varying Call Arrival Rates

In this section, we explore the sensitivity of the policies obtained by our ADP approach to changes in the call arrival rate. Recall that the original call arrival rate we used in Edmonton is about 4 calls per hour. Under this call arrival rate, we run the approximate policy iteration algorithm to find the best policy obtained by our ADP approach. We then vary the call arrival rate over the interval $[3.2, 4.8]$, but continue making the redeployment decisions by using the

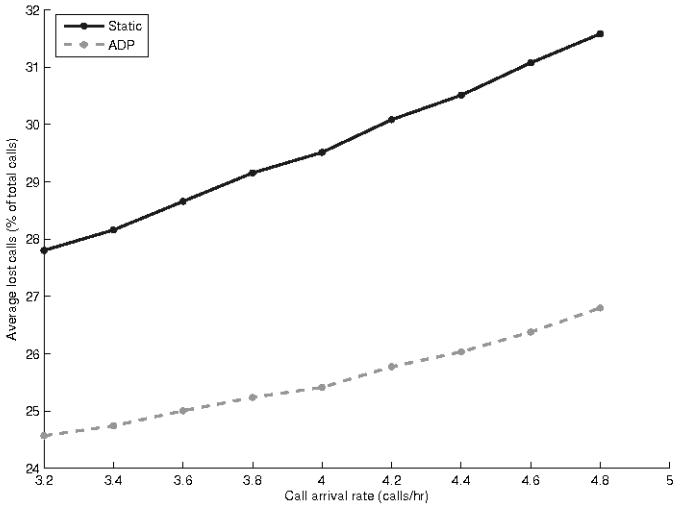

Figure 2.7: Performance of our ADP approach and the static policy for different call arrival rates.

policy that was obtained under the call arrival rate of 4 per hour. Our goal is to show that the policy that was obtained under the call arrival rate of 4 per hour continues to provide good performance when we perturb the call arrival rates. As a benchmark strategy, we find the best static policy under the call arrival rate of 4 per hour and use this static policy as we vary the call arrival rate over the interval $[3.2, 4.8]$.

The results are summarized in Figure 2.7. The horizontal axis in this figure gives the call arrival rate, whereas the vertical axis gives the expected percentage of missed calls. The results indicate that our ADP approach performs consistently better than the benchmark policy. Also, the expected percentage of calls missed by the static policy under the call arrival rate of 3.2 per hour is still larger than the expected percentage of calls missed by our ADP approach under the call arrival rate of 4.8 per hour.

2.5.8 Effect of Turn-out Time

Recall that if the ambulance crew is stationed at a base when it is notified of a call, then it takes 45 seconds to get ready. This duration is referred to as the turn-out time. On the other hand, an ambulance crew that is already on the road does not incur turn-out time. A potential argument against ambulance redeployment is that any gains are simply due to ambulance crews being on the road more often, and therefore incurring less turn-out time delays.

To check the validity of this argument, we applied our ADP approach under the assumption that turn-out time is zero. In this case, the expected percentage of missed calls for our ADP approach turned out to be $21.08\% \mp 0.1\%$, whereas the expected percentage of calls missed by the static policy turned out to be $23.93\% \mp 0.1\%$. As expected both the ADP approach and the static policy performed better when turn-out time is zero; however, our ADP approach continues to provide practically significant improvements over the static policy. This indicates the majority of performance increase from the ADP approach comes from better ambulance allocation rather than from incurring less turn-out time delays.

2.6 Conclusions

In this paper, we formulated the ambulance redeployment problem as a dynamic program and used an approximate version of the policy iteration algorithm to deal with the high-dimensional and uncountable state space. Computational experiments on two realistic problem scenarios show that our ADP

approach can provide high-quality redeployment policies. The basis functions that we construct open up the possibility of using other approaches, such as temporal-difference learning and the linear programming approach, to tune the parameters $\{r_p : p = 1, \ldots, P\}$.

Other future research will incorporate additional degrees of realism into our model. We plan to include stochastic travel times, multiple call priorities, other cost functions and more realistic ambulance dynamics that involve multiple ambulances serving certain calls. Incorporating these complexities may require constructing additional basis functions.

CHAPTER 3

TUNING APPROXIMATE DYNAMIC PROGRAMMING POLICIES FOR AMBULANCE REDEPLOYMENT VIA DIRECT SEARCH

3.1 Introduction

Emergency medical service (EMS) providers are tasked with staffing and positioning emergency vehicles to supply a region with emergency medical care and ensure short emergency response times. Large operating costs and increasing numbers of emergency calls make this a demanding task. One method commonly used to reduce response times to emergency calls is known as ambulance redeployment. Ambulance redeployment, also known as move-up or system-status management, is the strategy of relocating idle ambulances in real-time according to the state of the system to minimize response times for future emergency calls.

The ambulance redeployment literature contains a number of different approaches for computing redeployment policies. The first approach is to formulate an integer program that models ambulance redeployment and solve this integer program in real-time whenever a redeployment decision is required; see Kolesar and Walker (1974), Gendreau et al. (2001), Richards (2007), and Nair and Miller-Hooks (2009). This approach is computationally intensive and often requires a parallel computing environment and/or heuristic solution methods to obtain a redeployment decision within given time constraints. Another approach is to solve an integer program in a preparatory phase which dictates a "lookup table", or the desired ambulance locations given the number of currently available ambulances; see Gendreau et al. (2006) and Alanis et al. (2010).

Real-time redeployment is managed by dispatchers who attempt to position ambulances according to the prescribed locations. Other approaches attempt to directly incorporate the randomness of emergency calls into the model formulation. For example, Berman (1981a), Berman (1981c), and Berman (1981b) formulate the ambulance redeployment problem as a dynamic program (DP). This approach was revisited more recently in Zhang et al. (2010) in an attempt to gain insight into the problem. The difficulty of working with DP formulations is that computing optimal policies is only tractable in simplified situations such as those having only one or two ambulances. The work of Andersson (2005) and Andersson and Vaerband (2007) incorporates random evolution of the system heuristically through the construction of a "preparedness function" that attempts to evaluate future states based on their ability to handle incoming calls.

Another approach, using approximate dynamic programming (ADP), is formulated in Chapter 2 and Maxwell et al. (2009). This approach models ambulance redeployment as a Markov decision process (MDP) and defines a parameterized approximation J_r of the DP value function J. The approximation architecture J_r is constructed as a linear combination of B "basis" functions, which map the MDP state space to the real numbers, and weighting coefficients $r = (r_1, \ldots, r_B)$. For each redeployment decision the ADP policy draws samples of potential future state realizations and evaluates the relative merit of these states with J_r. Because of this need to simulate future state realizations, we call this method a simulation-based ADP approach.

The ADP approach is flexible enough to deal with the random evolution of the system directly and does not have the computational burden of the in-

teger programming methods. Similar to the lookup table approach, ADP algorithms require a preparatory tuning process which may be computationally expensive, but after this initial computation most ADP policies are able to operate in real-time situations without computational concerns. Since the ADP method only approximates the true value function, the resulting ADP policies are not necessarily optimal. Nevertheless, given a suitable approximation architecture, ADP policies have been shown to perform very well in problems that would be intractable otherwise. Examples of ADP applications include inventory control (Van Roy et al. (1997)), inventory routing (Adelman (2004)), option pricing (Tsitsiklis and Van Roy (2001)), backgammon (Tesauro (1994)), dynamic fleet management (Topaloglu and Powell (2006)), and network revenue management (Adelman (2007) and Farias and Van Roy (2007)).

An effective choice of basis functions in a linear approximation architecture usually requires expert opinion, accurate intuition of the MDP dynamics, and significant trial and error. There are no general methods to pick suitable basis functions; they must be chosen manually to represent the key features of the system. On the other hand, there are numerous algorithmic methods for tuning the parameter vector r to obtain good policies given a set of basis functions, e.g., approximate policy iteration (Bertsekas and Tsitsiklis (1996)), temporal-difference learning (Sutton (1988)), least-squares temporal-difference learning (Bradtke et al. (1996) and Boyan (2002)), linear programming (Schweitzer and Seidmann (1985) and de Farias and Van Roy (2003)), and smoothed approximate linear programming (Desai et al. (2009)).

The main weakness of these methods is that they attempt to tune r so that J_r approximates J. In ADP applications it is usually assumed that the true value

function J does not lie within the span of the basis function comprising J_r. Consequently, much of the value of fitting J_r to J may be lost. For many applications the more important metric is how the performance of the ADP policy compares with the optimal performance or with other known policies. Optimization methods such as direct search allow one to optimize over this metric directly rather than through an indirect and potentially less effective method. The main contribution of this paper is to illustrate potential shortcomings in typical value function fitting procedures for ADP and to propose direct search methods as a suitable alternative. Specifically, we build on Maxwell et al. (2010a) by using direct search methods to find ambulance redeployment policies that are superior to previous ADP redeployment policies in terms of both performance and computational effort required for tuning.

One potential drawback of using direct search methods to tune ADP policies is that direct search methods are generally more computationally intensive than value function fitting methods. We mitigate this problem by formulating our ambulance redeployment policy around the post-decision state formulation. The post-decision state formulation is a general-purpose method for (potentially) decreasing the computational burden in high-dimensional problems (see Powell and Roy (2004) and Powell (2007)). We further motivate this formulation by showing that the post-decision state ADP formulation is the limiting policy of a simulation-based ADP policy. This result holds under general conditions not relating to ambulance redeployment.

Section 3.2 introduces ADP and describes the construction of ADP policies. Section 3.3 uses a sample problem to describe theoretical limitations of common parameter tuning methods. Section 3.4 describes the ambulance rede-

ployment problem in detail, formulates ambulance redeployment as an MDP, and defines an ADP policy for ambulance redeployment. Section 3.5 illustrates how the limitations in Section 3.3 can have significant negative effects on policy performance and how direct search methods are able to provide higher-performing policies than traditional tuning approaches for the ambulance redeployment policy. Section 3.6 introduces the post-decision state formulation and shows that, coupled with direct search, the post-decision formulation of ambulance redeployment is an efficient method for tuning ADP parameters in high-dimensional problems. Section 3.7 contains our concluding remarks.

3.2 Approximate Dynamic Programming

MDPs are commonly used to model the evolution and control of stochastic systems. A key reason is that DP techniques provide a computational method to solve for optimal policies in MDPs. The DP optimal policy depends upon the computation of the DP value function J. Unfortunately, for many MDPs computing the value function J is intractable. One common approach to overcome this difficulty is to approximate J with a parameterized formulation that is easy to compute. Section 3.2.1 explains how these approximations are used to create ADP policies and Section 3.2.2 describes two general approaches used to tune the ADP approximation parameters.

3.2.1 ADP Policies

A discrete-time MDP is defined by a state space, a control space, system dynamics, transition costs, and an objective function. A state $s \in \mathcal{S}$ contains enough information such that the future evolution of the MDP is independent of the past evolution given state s. The state space $\mathcal{S}$ is the set of all possible states that may be realized by the MDP. The notation S_k denotes the kth state of the MDP evolution. For each $s \in \mathcal{S}$ we define a control space $\mathcal{X}(s)$ that dictates the available actions in state s. The control space for the MDP is defined as $\mathcal{X} = \bigcup_{s \in \mathcal{S}} \mathcal{X}(s)$ and we assume $|\mathcal{X}(s)| < \infty$ for all $s \in \mathcal{S}$. Let U_{k+1} denote a vector of iid Uniform$(0, 1)$ random variables (with appropriate dimension) used to generate all random behavior between states S_k and S_{k+1}. We denote the MDP system dynamics as $S_{k+1} = f(S_k, x, U_{k+1})$ where the next state S_{k+1} results from being in state S_k, choosing action $x \in \mathcal{X}(S_k)$, and having random effects dictated by U_{k+1}.

Let $c(S_k, x, U_{k+1})$ denote the (possibly random) transition cost associated with being in state S_k and choosing action $x \in \mathcal{X}(S_k)$. One common objective function for MDPs is to minimize the expected sum of the discounted transition costs from a given starting state S_0, i.e.,

$$\min_{\pi} \mathbb{E}\left[\sum_{k=0}^{\infty} \alpha^k c\left(S_k, \pi(S_k), U_{k+1}\right)\right], \tag{3.1}$$

where a policy π is a mapping from $\mathcal{S}$ to $\mathcal{X}$ such that $\pi(s) \in \mathcal{X}(s)$ for all $s \in \mathcal{S}$ and $\alpha \in (0, 1)$ is the discount factor. This expectation is finite if, e.g., c is bounded. The value function for policy π starting in state S_0 is defined as

$$J^{\pi}(S_0) = \mathbb{E}\left[\sum_{k=0}^{\infty} \alpha^k c(S_k, \pi(S_k), U_{k+1})\right].$$

For an optimal policy π^* we define the value function $J = J^{\pi^*}$. From the DP optimality principle we know that

$$J(S_k) = \min_{x \in \mathcal{X}(S_k)} \mathbb{E}\left[c(S_k, x, U_{k+1}) + \alpha J\left(f(S_k, x, U_{k+1})\right)\right] \qquad \forall S_k \in \mathcal{S}, \tag{3.2}$$

and that choosing an action $x \in \mathcal{X}(S_k)$ achieving the minimum in the right-hand side of (3.2) for every state $S_k \in \mathcal{S}$ yields an optimal policy Bertsekas and Shreve (1978). Unfortunately, the complexity of computing J often increases drastically in the size of $\mathcal{S}$; hence most MDPs with large state spaces are intractable to solve via DP.

One approach to overcome these computational difficulties is to approximate J with a simpler function through an approximation architecture and use the approximation in lieu of the value function when computing policy decisions. This approach is known as ADP. For example, given "basis functions" $\phi_1, \ldots, \phi_B$ mapping $\mathcal{S}$ to $\mathbb{R}$ we define a linear approximation architecture $J_r(\cdot) = \sum_{b=1}^{B} r_b \phi_b(\cdot)$ where the subscript r denotes the vector $(r_1, \ldots, r_B)$ of tunable parameters. Given an approximation J_r for J we define the quantity

$$L_r(S_k) = \min_{x \in \mathcal{X}(S_k)} \mathbb{E}\left[c(S_k, x, U_{k+1}) + \alpha J_r\left(f(S_k, x, U_{k+1})\right)\right] \qquad \forall S_k \in \mathcal{S}, \tag{3.3}$$

and the ADP policy with respect to L_r chooses an action $x \in \mathcal{X}(S_k)$ achieving the minimum in the right-hand side. In this manner, L_r defines an ADP policy. This ADP policy is also known as the greedy policy with respect to J_r; however, we use the former terminology to distinguish between ADP policies that use the same approximation architecture J_r but are formulated differently.

One advantage of ADP is that if the basis functions are simple then the approximate value function can be computed quickly and it can be computed for individual states as needed, as opposed to computing the value function for all

states *a priori*. This allows ADP to solve for policies in MDPs with large and even uncountable state spaces, such as those arising in ambulance redeployment.

For any given state $s \in \mathcal{S}$ there may be multiple actions achieving the minimum on the right-hand side of (3.2) or (3.3). In DP any action achieving the minimum may be chosen without loss of optimality. However, in an ADP context these actions may have very different consequences. Nevertheless, it is common to regard all actions achieving $L_r(s)$ as equivalent choices since they are equivalent insofar as the approximation architecture is able to differentiate. Because the vector of tunable coefficients r ultimately affects actions chosen and overall performance these coefficients are referred to as ADP policy parameters. These coefficients are generally tuned in a preparatory phase before an ADP policy is implemented.

For further information, see Puterman (2005) for MDPs, Bertsekas (1995) for DP, and Bertsekas and Tsitsiklis (1996), Powell (2007) for the related ADP concepts.

3.2.2 Tuning ADP policies

Theoretical results show that if $J_r(s) \approx J(s)$ for all $s \in \mathcal{S}$ then the performance of the greedy policy with respect to J_r is not too far from the optimal policy performance, e.g., (Bertsekas and Tsitsiklis, 1996, Proposition 6.1). As a result, the standard method for tuning approximation architectures tries to ensure that $J_r \approx J$. In this sense, value function fitting methods attempt to find the coefficients r which solve

$$\min_r \|J_r - J\|,$$

for some distance measure $\|\cdot\|$. These methods tune the coefficients based upon approximating the true value function with the hope of getting good policy performance as result.

Our direct search method instead attempts to find the coefficients r which solve

$$\min_r \mathbb{E}\left[\sum_{k=0}^{\infty} \alpha^k c\left(S_k, \pi_r(S_k), U_{k+1}\right)\right],$$

where π_r denotes the ADP policy with respect to, say, L_r. Thus the direct search method tunes the policy coefficients of the value function approximation J_r based solely upon the performance of the ADP policy using J_r.

This difference with standard tuning methods is significant due to the fact that ADP policies rely upon the *relative* magnitude of J_r for future states as opposed to the *absolute* value for making decisions. Thus, an approximate value function J_r bearing little resemblance to the actual value function J can still induce a good policy. One example of this principle is Szita and Lörincz (2006) where a noisy cross-entropy method was used to improve performance by an order of magnitude over value function fitting methods for ADP policies playing the game Tetris. Additionally, as shown in Section 3.3, value function fitting methods may not always return the coefficients corresponding to the best policy performance.

3.3 Limitations of Common ADP Tuning Approaches

Two general approaches for value function fitting are regression-based methods and linear programming-based methods. Both of these methods share a similar

shortcoming: in situations where there exists a set of coefficients r which induce an optimal policy, the regression- and linear programming-based methods may be unable to select such coefficients in the ADP tuning procedure. Section 3.3.1 illustrates this principle for regression-based approaches, and Section 3.3.2 illustrates this principle for linear programming-based approaches.

3.3.1 Limitations of Regression-Based Approaches

Given an initial policy π_0, regression-based methods take a noisy estimate, $\hat{J}^{\pi_0}(s)$, of the value function for π_0 starting at state s for each $s \in \mathcal{S}$ (or perhaps only a subset of $\mathcal{S}$). A new set of policy parameters $r^*_{\pi_0,p}$ are calculated via regression, i.e.,

$$r^*_{\pi_0,p} = \underset{r}{\text{argmin}} \left\{ \sum_{s \in \mathcal{S}} \left| \hat{J}^{\pi_0}(s) - J_r(s) \right|^p \right\}, \tag{3.4}$$

where $1 \leq p \leq \infty$ indicates the p-norm used in the regression. Commonly in ADP, the least-squares regression is formulated recursively and used in conjunction with simulation to update policy parameters after each sampled state transition; see Bertsekas and Tsitsiklis (1996) and Powell (2007).

Usually the regression-based tuning is iterated with the hope of finding a set of parameters inducing a policy with good performance. For example, one standard tuning approach called approximate policy iteration begins with an initial policy π_0 and then simulates state trajectories using π_0 that are used to compute the coefficients $r^*_{\pi_0,2}$ via (3.4). In the next iteration π_1 is set to be the greedy policy with respect to the approximation architecture using the coefficients from

the initial iteration, i.e., $J_{r^*_{\pi_{0,2}}}$, and π_1 is used to simulate state trajectories and compute the coefficients for the next iteration $r^*_{\pi_1,2}$. This process is repeated until the computational budget is exhausted or a suitable policy is obtained.

Regression-based tuning methods are appealing because they are easily understood, easily implemented, and fast to compute. Although appealing, regression-based tuning methods may have drawbacks. Consider the MDP shown in Figure 3.1 where the objective is to minimize the discounted sum of the transition costs starting from state 3, where $\alpha \in (0, 1)$ is the discounting factor. In this MDP there are only two deterministic policies: π_1 which denotes choosing to transition to State 1 in State 3 and π_2 which denotes choosing to transition to State 2 in State 3. Consider the case where $\alpha > 3/4$. In this case π_2 is the optimal policy.

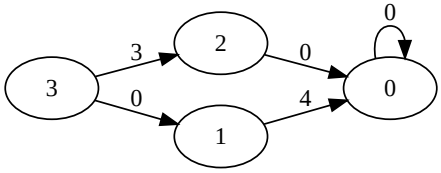

Figure 3.1: Example MDP

Let $\phi_1(s) = s$ and consider the approximation architecture $J_r(s) = r_1\phi_1(s)$. For the approximation architecture to induce the policy π_2 we must have $3 + \alpha J_r(2) = 3 + 2\alpha r_1 < \alpha r_1 = 0 + \alpha J_r(1)$ which implies that $r_1 < -3/\alpha$. Thus we see that the approximation architecture J_r is rich enough to induce an optimal policy provided that r_1 is chosen correctly. Unfortunately, the regression-based formulation in (3.4) is unable to find such a set of parameters.

Proposition 3.1. *Let $\mathcal{R} = \left\{r : r = r^*_{\pi_i,p}$ for $i = 1, 2$ and $1 \leq p \leq \infty\right\}$ denote the set of all possible coefficients from a p-norm regression formulation for the MDP in Figure 3.1 (see* (3.4)*). For all $r \in \mathcal{R}$ the greedy policy with respect to J_r is π_1, i.e., is not optimal.*

Proof. Assume $1 \leq p < \infty$. We have that

$$\sum_{s=0}^{3} |J^{\pi_1}(s) - r_1 s|^p = |0-0|^p + |4-r_1|^p + |0-2r_1|^p + |4\alpha - 3r_1|^p$$

$$= |4-r_1|^p + 2^p |r_1|^p + |4\alpha - 3r_1|^p . \tag{3.5}$$

Note that for any $r_1 < 0$ each term of (3.5) can be decreased by increasing r_1 to 0. Hence we know $r^*_{\pi_1,p} \geq 0$ and the policy induced by $J_{r^*_{\pi_1,p}}$ is π_1 for any $1 \leq p < \infty$. A similar analysis holds for $r^*_{\pi_2,p}$ as well, and the same argument works when $p = \infty$. □

Thus the regression-based tuning methods with respect to any p-norm will always return a sub-optimal policy. Further analysis shows that the same result holds when any subset of $\mathcal{S}$ is used in the regression (with the exception of $\{0\}$ for which any $r_1 \in \mathbb{R}$ is a valid regression solution). Given any $\alpha > 0$ we can create a similar MDP for which Proposition 3.1 holds. Additionally, with minor modifications to the MDP in Figure 3.1, we can create MDPs where the suboptimal policy is arbitrarily worse than the optimal policy. A similar example for an MDP without discounting is given in Maxwell et al. (2010a).

3.3.2 Limitations of LP-Based Approaches

LP-based methods are based on the LP formulation of exact dynamic programs and have the form

$$\max_{r} \quad \nu^T \bar{J}_r \tag{3.6}$$

$$\text{s.t.} \quad \mathbb{E}\left[c(S_k, x, U_{k+1}) + \alpha J_r(f(S_k, x, U_{k+1}))\right] \geq J_r(S_k) \qquad \forall S_k \in \mathcal{S},\ x \in \mathcal{X}(S_k)$$

for minimizing the expected α-discounted transition costs, where ν is a (column) vector with positive components and $\bar{J}_r$ denotes a vector containing

$J_r(S_k)$ for all $S_k \in \mathcal{S}$, see Schweitzer and Seidmann (1985) and de Farias and Van Roy (2003). This LP formulation is most useful when the expectation in the constraints can be computed exactly or estimated easily. Also, due to the number of constraints in this LP, it is often necessary to use techniques such as constraint sampling to make the LP tractable.

Again consider the MDP shown in Figure 3.1 with discounting factor $\alpha > 3/4$. Let $\phi_1(s) = 1$ and $\phi_2(s) = s$ and consider the approximation architecture $J_r(s) = r_1\phi_1(s)+r_2\phi_2(s)$. For the approximation architecture to induce the policy π_2 we must have $3 + \alpha J_r(2) = 3 + \alpha r_1 + 2\alpha r_2 < \alpha r_1 + \alpha r_2 = 0 + \alpha J_r(1)$ which implies that $r_2 < -3/\alpha$. Although the approximation architecture J_r can induce the optimal policy provided r_2 is chosen correctly, the LP-based formulation will not select optimal-policy-inducing coefficients.

Proposition 3.2. *The greedy policy with respect to any optimal solution to the LP-based formulation* (3.6) *is π_1, and hence the LP-based approach results in an ADP policy that is not optimal.*

Proof. The LP formulation for the MDP in Figure 3.1 can be written as

$$\max_r \quad \begin{bmatrix} \nu_1 & \nu_2 & \nu_3 & \nu_4 \end{bmatrix} \begin{bmatrix} r_1 \\ r_1 + r_2 \\ r_1 + 2r_2 \\ r_1 + 3r_2 \end{bmatrix}$$

$$\begin{aligned}
\text{s.t.} \quad (1-\alpha)r_1 &\leq 0 && (3.7)\\
(1-\alpha)r_1 + r_2 &\leq 4 && (3.8)\\
(1-\alpha)r_1 + 2r_2 &\leq 0 && (3.9)\\
(1-\alpha)r_1 + (3-\alpha)r_2 &\leq 0 && (3.10)\\
(1-\alpha)r_1 + (3-2\alpha)r_2 &\leq 3. && (3.11)
\end{aligned}$$

Let $q_\alpha = (1-\alpha)r_1$, and rewrite (3.8)-(3.11) as

$$\begin{aligned} r_2 &\leq 4 - q_\alpha \\ r_2 &\leq \frac{-q_\alpha}{2} \\ r_2 &\leq \frac{-q_\alpha}{3-\alpha} \\ r_2 &\leq \frac{3-q_\alpha}{3-2\alpha}. \end{aligned}$$

Since $q_\alpha \leq 0$ by (3.7) we know that the feasible region for (r_1, r_2) contains at least $\{(r_1, r_2) : r_1 \leq 0, r_2 \leq 0\}$. Given any feasible $r = (r_1, r_2)$ inducing an optimal ADP policy (i.e., $r_1 \leq 0$ and $r_2 < -3/\alpha$) we have that $r' = (r_1, 0)$ is also feasible. The objective function value for r' is greater than that of r by $-(\nu_2+2\nu_3+3\nu_4)r_2 > 0$. Consequently, coefficients inducing an optimal ADP policy will not induce an optimal LP solution, and hence will never be returned via an LP-based tuning approach. □

Given any $\alpha \in (0, 1]$, one can find a value for the transition cost from state 1 to state 0 in the example MDP such that Proposition 3.2 holds for that value of α. Additionally, the difference between the optimal policy performance and that obtained via the LP tuning methods can be made arbitrarily large by increasing this transition cost.

The MDP in Figure 3.1 is of course a simple example that is easily solved exactly, but it is very typical of ADP applications where transition costs must be balanced with value function approximations of future states given an approximation architecture that does not contain the true value function. Furthermore, since the example MDP is deterministic the inability to properly tune the policy coefficients is a direct consequence of the tuning methods themselves, as opposed to sampling-based errors that may arise when dealing with random state

transitions.

In Sections 3.4 and 3.5 we show that similar issues arise in a realistic application of ADP, namely ambulance redeployment.

3.4 Ambulance Redeployment

To model EMS operations we divide the region under consideration into an appropriate-sized grid and assume that calls in each grid cell arrive according to independent time-inhomogeneous Poisson processes and are distributed uniformly throughout the cell. When an emergency call arrives the dispatcher assigns the closest available ambulance to the call. If there are no ambulances available the call is placed on a waiting list, and these calls are served first-come first-served as ambulances become available.

In actual EMS operations there may be special situations where the closest ambulance is not dispatched to an emergency call or when the first call on the waiting list is not assigned to a newly freed ambulance. Additionally, emergency calls are usually categorized depending upon the severity of the emergency by the dispatcher and emergency response is prioritized according to these "levels of care." We do not include this categorization in our model.

If an ambulance assigned to an emergency call is idle at base we assume it takes 45 seconds for the ambulance crew to get situated in their vehicle. This is called the "turn-out" time. If the assigned ambulance is not idle at a base it does not incur turn-out time.

After being assigned to an emergency call, the ambulance travels to the call

scene. We assume deterministic travel times along the shortest path of a representative street network. The distance to emergency calls off the road network are calculated using the Manhattan distance from the nearest node. Although not used in our model, random travel times could be incorporated with little modification.

Paramedics provide preliminary care to the patient at the scene. We model this "on scene" time as an exponentially distributed random variable having a mean of 12 minutes. In approximately 25% of the cases the patient does not need to be transported to a hospital and the ambulance becomes free at the call scene. Otherwise, the patient is transported to a hospital and transferred to the hospital staff. The destination hospital is chosen randomly based on historical data given the location of the emergency, and the transfer time at the hospital is modeled as a Weibull-distributed random variable with a mean of 30 minutes and standard deviation of 13 minutes.

Our ADP policy makes redeployment decisions when an ambulance becomes available after completing care for the patient, either at the call scene or the hospital. Thus, when an ambulance becomes available, the redeployment policy is used to calculate the desired redeployment base for this ambulance. After the redeployment decision is made, the ambulance travels to this destination to wait for future calls; however, if an emergency call arrives before this ambulance has reached its destination it may still be assigned to the call (provided it is closest to the call). Our goal is to tune the ADP policy such that the redeployment decisions position ambulances where they may best respond to future calls. We only consider redeploying ambulances to bases, but other locations such as convenient street intersections and hospitals could be included as

well. If there are calls on the waiting list when an ambulance becomes available, the ambulance is immediately assigned to the first call on the waiting list and no redeployment decision is made.

Section 3.4.1 gives an MDP formulation for this problem and Section 3.4.2 presents the ADP policy for ambulance redeployment including an explanation of the basis functions used within the approximation architecture and the methods used to compute the policy.

3.4.1 Ambulance Redeployment as an MDP

We model the ambulance redeployment problem as a queueing system within the generalized semi-Markov decision process framework Glynn (1989). In this setup emergency calls are customers and ambulances are servers. The service time for a call includes the response time, time at scene, and transport and transfer time to a hospital if needed.

State Space

Let N denote the number of ambulances, and let c_i for $i = 1, \ldots, N$ denote the location of the emergency call being served by ambulance i, with $c_i = \emptyset$ if ambulance i is available. Let a_i denote the location of ambulance i at the time it last responded to or completed serving an emergency call. In this queueing system, the service time distribution for a call depends upon both c_i and a_i. Let r_i denote the redeployment base for ambulance i, i.e., the location of the base to which ambulance i will position itself once ambulance i becomes available. If

ambulance i is serving an emergency call, then the value of r_i does not impact the system dynamics until the ambulance becomes available. Control actions in this MDP will set these redeployment values as ambulances become available. Let w_i for $i = 1, \ldots, M$ denote the location of the ith emergency call on the waiting list, with $w_i = \emptyset$ if there is no ith call waiting. If an emergency call arrives and there are already M calls on the waiting list we assume the emergency call is handled by another agency. This assumption has very little practical implication since one could take M to be quite large to capture EMS operations having no alternative supporting agencies.

The events in the DEDS are e_0 indicating that a call arrival has occurred, and e_i for $i = 1, \ldots, N$ indicating that ambulance i has completed serving its assigned emergency call. Let t denote the current simulation time, t_0 denote the time since the last emergency call arrival, and t_i for $i = 1, \ldots, N$ denote the elapsed time since ambulance i was either assigned to an emergency call or finished serving an emergency call.

Let $C = (c_1, \ldots, c_N)$, $A = (a_1, \ldots, a_N)$, $R = (r_1, \ldots, r_N)$, $W = (w_1, \ldots, w_M)$, and $T = (t, t_0, \ldots, t_N)$. Thus the state of the DEDS can be denoted as $s = (C, A, R, W, e, T)$ where $e \in \{e_0, \ldots, e_N\}$. Let $C(s)$, $A(s)$, $R(s)$, $W(s)$, $e(s)$, and $T(s)$ denote the C, A, R, W, e, and T components of state s. Additionally, let $c_i(s)$, $a_i(s)$, $r_i(s)$, and $w_i(s)$ denote the ith component of $C(s)$, $A(s)$, $R(s)$, and $W(s)$, let $t(s)$ denote the t component of $T(s)$, and let $t_i(s)$ denote the t_i component of $T(s)$.

Ambulance dispatchers typically have more information at their disposal than that contained within this state space representation. For example, dispatchers often know the status of the busy ambulances–whether they are treat-

ing a patient at the call scene, transporting a patient to a hospital, or at a hospital transferring a patient. Dispatchers use this information to supplement redeployment decisions. If an emergency call arrives near a hospital at which some ambulance has almost completed transferring their patient to the hospital staff, the dispatcher may not assign an ambulance to this call and instead wait for the ambulance at the hospital to become available. In this sense, the state space representation is a simplified model of dispatcher information and the simulated dynamics are a simplification of ambulance redeployment dynamics (as is typical for simulation).

Control Space

We call state s a decision state if there is a redeployment decision to be made in this state, i.e., if an ambulance just became available, $e(s) = e_i$ and $t_i(s) = 0$ for some $1 \leq i \leq N$, and there are no calls on the waiting list, $w_1(s) = \emptyset$. For decision states $\mathcal{X}(s)$ is the set of potential locations at which we may position the newly freed ambulance. We consider this set to be a predetermined set of ambulance bases in the proximity of an ambulance's home base. For "non-decision states" $\mathcal{X}(s) = \{\emptyset\}$ indicating a "do-nothing" action. We assume that $|\mathcal{X}(s)|$ is finite (and not too large) for all $s \in \mathcal{S}$ so that the right-hand side of (3.3) can be estimated via Monte Carlo for each $x \in \mathcal{X}(s)$ within real-time computation constraints.

Often in EMS operations more complicated redeployment actions are taken. For example, when one ambulance becomes free multiple idle ambulances may be asked to relocate as well as the newly freed ambulance. Such redeployment policies could also be implemented within our ADP framework; however, we

do not follow this approach for two reasons. First, as the complexity of potential redeployment options increases, the time required to compute the policy decision increases as well. Although more complex redeployment decisions may be able to increase performance over single ambulance redeployments, we believe the potential improvement to be marginal. Second, ambulance crews typically consider relocating from base to base (and perhaps back again later) irritating and even useless. Given high crew turn-over rates there is a desire to avoid frustrating crews via perceived-unnecessary redeployments. As such we only consider redeployment for ambulances that have just finished a call and hence they must travel somewhere anyway. One modification of our ADP policy is to consider redeploying idle ambulances to other bases at set time intervals. The work in Section 2.5 contains empirical data on the performance improvement for such a modification on a similar ADP policy.

System Dynamics

We denote the MDP system dynamics as $S_{k+1} = f(S_k, x, U_{k+1})$ where the next state S_{k+1} is a function of being in state S_k, choosing action $x \in \mathcal{X}(S_k)$, and having random effects dictated by U_{k+1}. Define the post-decision state as the immediate state resulting from being in state S_k and applying the effects of action x to the state (before the passage of time). We denote the post-decision state resulting from state S_k and choosing action $x \in \mathcal{X}(S_k)$ as $S_k^+(x)$. For any decision state S_k and action x the post decision state $S_k^+(x)$ is equal to S_k, except that the ambulance that became available in state S_k has been dispatched to base x. Specifically, if $e(S_k) = e_i$ then $r_i(S_k^+(x)) = x$ and the remaining components of S_k and $S_k^+(x)$ are equal. The pre-decision state and post-decision state are equal

for non-decision states. When the control taken in state S_k is implicitly understood we use the more concise notation S_k^+. A more complete discussion of post-decision states and associated ADP representations can be found in Powell (2007).

Given $S_0^+ \in \mathcal{S}$ with $t(S_0^+) = 0$ we use the following algorithm to construct the DEDS:

1. **(Initialization)** Set the event counter $k = 0$.
2. **(Residual time generation)** For each event $e_i \in E(S_k^+)$ generate a residual time z_i conditional upon the current state S_k^+ where $E(S_k^+) = \{e_0\} \cup \{e_i : c_i(S_k^+) \neq \emptyset \text{ for } i = 1, \ldots, N\}$ denotes the set of active events for state S_k^+, i.e., the set of events that may cause a transition out of state S_k^+. Without loss of generality we assume z_i is generated by inversion from the ith component of U_{k+1} which we denote $U_{k+1}(i)$. Thus $z_i = F_i^{-1}(S_k^+, U_{k+1}(i))$ where $F_i^{-1}(S_k^+, \cdot)$ is the quantile function for the residual event time for event e_i in state S_k^+.
3. **(Select next event)** Let E_{k+1} be an event with minimal residual time from step 2, and let Δ_{k+1} denote that residual time.
4. **(Select next state)** Generate S_{k+1} conditional upon S_k^+, E_{k+1}, and Δ_{k+1} via U_{k+1}.
5. **(Update clocks)** Set $t(S_{k+1}) = t(S_k) + \Delta_{k+1}$. Set $t_i(S_{k+1}) = 0$ where i is the index of event E_{k+1}. Additionally, set $t_i(S_{k+1}) = 0$ if there is an emergency call arrival in state S_{k+1}, i.e., $E_{k+1} = e_0$, and ambulance i is assigned to respond to the arriving call. For all other clocks set $t_i(S_{k+1}) = t_i(S_k) + \Delta_{k+1}$.
6. **(Redeployment Decision)** Let $S_{k+1}^+ = S_{k+1}$. If S_{k+1} is a decision state then choose a desired ambulance base for redeployment, $x \in \mathcal{X}(S_{k+1})$, for

server i and set $R_i(S_{k+1}^+) = x$, where i denotes the server that just became idle, i.e., the index i such that $E_{k+1} = e_i$.

7. **(Repeat)** Set $k = k + 1$ and repeat from Step 2.

Given this DEDS formulation, we define the continuous-time queueing process $S(t)$ of the DEDS for $t \geq 0$ by defining

$$
\begin{aligned}
C(t) &= \sum_{k=0}^{\infty} C(S_k^+) \mathbb{1}_{\{t(S_k) \leq t < t(S_{k+1})\}} \\
A(t) &= \sum_{k=0}^{\infty} A(S_k^+) \mathbb{1}_{\{t(S_k) \leq t < t(S_{k+1})\}} \\
R(t) &= \sum_{k=0}^{\infty} R(S_k^+) \mathbb{1}_{\{t(S_k) \leq t < t(S_{k+1})\}} \\
W(t) &= \sum_{k=0}^{\infty} W(S_k^+) \mathbb{1}_{\{t(S_k) \leq t < t(S_{k+1})\}} \\
e(t) &= \sum_{k=0}^{\infty} e(S_k^+) \mathbb{1}_{\{t(S_k) \leq t < t(S_{k+1})\}}, \text{ and} \\
T(t) &= \left(t, \sum_{k=0}^{\infty} \left(t_0(S_k^+) + t - t(S_k) \right) \mathbb{1}_{\{t(S_k) \leq t < t(S_{k+1})\}}, \ \cdots \ , \right. \\
&\qquad \left. \sum_{k=0}^{\infty} \left(t_N(S_k^+) + t - t(S_k) \right) \mathbb{1}_{\{t(S_k) \leq t < t(S_{k+1})\}} \right).
\end{aligned}
$$

By this construction $S(t)$ is a piecewise constant function in the C, A, R, W, and e components and piecewise linearly increasing in the T component with jumps occurring only at event times. Additionally, $S(t)$ is right continuous with left limits.

Transition Costs and Objective Function

Given a state S_k, an action $x \in \mathcal{X}(S_k)$, and U_{k+1} we incur a transition cost of $c(S_k, x, U_{k+1})$. Let D be a given threshold time of 8 minutes for satisfactory re-

sponse times. The transition cost is 1 if an ambulance is assigned to a call which it cannot reach within D minutes and 0 otherwise. In other words,

$$c(S_k, x, U_{k+1}) = \begin{cases} 1 & \text{if } \exists\ i \text{ s.t. } t_i(S_{k+1}) = 0,\ c_i(S_{k+1}) \neq \emptyset, \text{ and} \\ & \qquad d(a_i(S_{k+1}), c_i(S_{k+1})) > D \\ 0 & \text{otherwise} \end{cases}$$

where $t_i(S_{k+1}) = 0$ and $c_i(S_{k+1}) \neq \emptyset$ together indicate that ambulance i has just started responding to an emergency call and $d(a_i(S_{k+1}), c_i(S_{k+1}))$ is the travel time between ambulance i's location at the time of the arrival, $a_i(S_{k+1})$, and the call location $c_i(S_{k+1})$ (including any turn-out time).

We are interested in the expected number of "lost calls," or calls not responded to within the time threshold D, over a finite planning horizon T which is usually between one and two weeks. Thus, given the initial state S_0, we use the objective function in (3.1) where $c\left(S_k, \pi(S_k), U_{k+1}\right) = 0$ for all S_k such that $t(S_k) > T$. Discounting future lost calls has no clear significance or meaning in an ambulance redeployment context, so α would typically be chosen to be very close to one.

The transition cost we use is chosen both for simplicity and because EMS provider contracts are usually written in terms of the percentage of calls not responded to within the given threshold. One consequence of this transition cost definition is that emergency calls responded to just after the threshold D and those responded to much later than D are weighted equally for optimization purposes although the medical outcome of such response times are likely to be quite different. The work in Erkut et al. (2008) acknowledges this discrepancy and optimizes for "maximum survivability." Although we do not incorporate this into our work, the ADP framework we use is versatile enough to allow

these modifications. Regardless, Maxwell et al. (2009) shows that ADP policies using the given transition costs are able to reduce overall response times, and not just for emergency calls with response times near or below the threshold.

3.4.2 ADP Policy for Ambulance Redeployment

To define an ADP policy for ambulance redeployment using the form of (3.3) we must first define the basis functions $\phi_1, \ldots, \phi_B$ and the policy parameters r that form the approximation architecture J_r (Section 3.4.2). By assumption $|\mathcal{X}(s)|$ is not too large for any $s \in \mathcal{S}$, so we can compute the minimization in (3.3) by calculating the value of the right-hand side for each $x \in \mathcal{X}(s)$ and taking the minimum. We cannot, however, compute the expectation on the right-hand side in closed form, so we must estimate it by using Monte Carlo simulation (Section 3.4.2).

Erlang Basis Function

The approximation architecture used in our ADP policy decouples the region into smaller, independent regions each containing only a single ambulance base. Each of these regions is modeled as an Erlang loss system having exponential interarrival time distributions and general service time distributions. The basis functions in our approximation architecture represent the Erlang loss for each of these smaller regions. Thus B represents both the number of basis functions in our approximation architecture and the number of ambulance bases.

Let Γ denote the collection of grid cells into which the call arrival pro-

cess is discretized. Define $l(\gamma)$ for $\gamma \in \Gamma$ to be the centroid of cell γ and $l(b)$ for $b = 1, \ldots, B$ to be the location of base b. Let $\Gamma_b = \{\gamma \in \Gamma : d(l(\gamma), l(b)) \leq d(l(\gamma), l(b')) \text{ for all } b' = 1, \ldots, B\}$ denote the set of grid cells that have centroids closer to base b than to any other base.

Define $\lambda(\gamma, t)$ to be the call arrival rate in cell γ at time t and $\Lambda(t) = \sum_{\gamma \in \Gamma} \lambda(\gamma, t)$ to be the total call arrival rate at time t. Let $\lambda_b(s) = \sum_{\gamma \in \Gamma_b} \lambda(\gamma, t(s))$ denote the arrival rate to Γ_b, and let $n_b(s) = \sum_{i=1}^{N} \mathbb{1}_{\{c_i(s)=\emptyset\}} \mathbb{1}_{\{r_i(s)=l(b)\}}$ denote the number of ambulances either idle at base b or redeploying to base b in state s.

Thus for $b = 1, \ldots, B$, we define the basis function for base b in state s to be the Erlang loss for an $M/G/n_b(s)/n_b(s)$ queue with arrival rate $\lambda_b(s)$ and service rate μ_b weighted according to how likely call arrivals are within Γ_b:

$$\phi_b(s) = \frac{\lambda_b(s)}{\Lambda(t(s))} \frac{(\lambda_b(s)/\mu_b)^{n_b(s)}/n_b(s)!}{\sum_{k=0}^{n_b(s)} (\lambda_b(s)/\mu_b)^k/k!}.$$

The average service time $1/\mu_b$ is a sum of the average response time, scene time, hospital transport time, and hospital transfer time for calls arriving in Γ_b. The scene time and hospital transfer time are generated from distributions with known means. The average response time and hospital transport time are estimated via simulation as the average response time from base b to calls arriving in Γ_b and the average transport time from calls arriving in Γ_b to their destination hospitals respectively. Repeating this procedure for $b = 1, \ldots, B$ we approximate the service rates $\mu_1, \ldots, \mu_b$ prior to running the DEDS and include them as input to the simulation.

The Erlang loss is the steady state proportion of arrivals refused service in a queue having limited queueing capacity. This quantity is relevant for ambulance redeployment because it can be viewed as an approximation for the

steady state proportion of arrivals that cannot be served by the ambulance base nearest their call location. As a consequence, these emergency calls must either be placed on a waiting list or served by an ambulance assigned to a base further away. In either situation the response time for the call is likely to increase significantly and the majority of such calls will not be served within the given threshold. For this reason the Erlang loss calculation is closely related to the value function for a particular state, i.e., the proportion of lost calls resulting from being in a given state.

As is common in ADP applications, the basis functions $\phi_1, \ldots, \phi_B$ are not intended to represent the value function exactly. Our approximation architecture ignores the state dependent service rates as well as the more complex dynamics involved when ambulances serve emergency calls outside their respective area. Nevertheless, the basis functions in combination with the tuning coefficients $r_1, \ldots, r_B$ are effective for designing policies that perform well. For example, this approximation architecture is able to significantly improve upon the ambulance redeployment policies in Chapter 2 and Maxwell et al. (2009).

Simulation-Based ADP Policy

For a given decision state $S_k \in \mathcal{S}$, the ADP policy selects a redeployment base by choosing the redeployment base $x \in \mathcal{X}(S_k)$ that minimizes the right-hand side of (3.3). Because we cannot compute the expectation in the right-hand side of (3.3) analytically, we estimate it through Monte Carlo simulation. Let $U_{k+1}^{(1)}, \ldots, U_{k+1}^{(G)}$ denote G iid uniform random vectors of appropriate dimension.

We approximate the ADP policy as

$$\underset{x \in \mathcal{X}(S_k)}{\operatorname{argmin}} \frac{1}{G} \sum_{g=1}^{G} \left(c(S_k, x, U_{k+1}^{(g)}) + \alpha J_r \left(f(S_k, x, U_{k+1}^{(g)}) \right) \right). \tag{3.12}$$

We call this approach simulation-based ADP due to the use of Monte Carlo simulation, as opposed to other methods that do not need to estimate the expectation via simulation (e.g., the post-decision state policy in Section 3.6). Techniques such as ranking and selection and common random numbers can also be used to select the minimum instead of using naïve Monte Carlo sampling for each $x \in \mathcal{X}(S_k)$.

3.5 Simulation Optimization Tuning Results

The limitations of value function fitting methods as illustrated in Section 3.3 extend beyond sample problems such as that depicted in Figure 3.1. We consider the problem of tuning the simulation-based ambulance redeployment ADP policy for use in Edmonton, Alberta, Canada. Edmonton is the 5th largest city in Canada with a population over 700,000. We model the ambulance operations of Edmonton using a discrete event simulation having 16 ambulances, 11 ambulance bases, and 5 hospitals. The emergency call arrival model has a flat arrival rate of 6 calls/hour with a higher density of calls in the metro areas during the day and a higher density of calls in rural areas in the evening and early morning. The travel model used in the simulation is a deterministic shortest-path calculation on a network consisting of major roads in the Edmonton area.

Before proceeding with simulation results we wish to stress that since the emergency call arrival process, ambulance redeployment policies, and travel

network are all stylized, our simulation results should not be interpreted as indicative of actual performance in Edmonton, nor should the results in Section 3.6.3 be interpreted in that way. Rather, these computational results showcase the performance of different ADP policies in realistic but not real scenarios with realistic but not real dynamics.

Figure 3.2 shows the performance of three different tuning methods for the simulation-based ADP policy in Section 3.4.2 using the value function approximation architecture given in Section 3.4.2. Each point in the graph represents an unbiased estimate of policy performance for the ADP policy with respect to L_r for a given set of coefficients r. The coefficients used in each iteration are dictated by the tuning method (based upon the results of previous iterations), and policy performance is estimated from 20 two-week simulations of ambulance operations using the specified policy. The policy performance estimation for one set of coefficients is considered to be one function evaluation, and the sequence of these function evaluations are indicated along the x-axis. The estimated policy performance for each policy, as measured by the percent of calls that are lost, is indicated along the y-axis.

The least-squares method is a value function fitting method described in Section 3.3.1, the Nelder-Mead method is a black box unconstrained local optimization heuristic for deterministic functions (Nelder and Mead (1965)), and the Noisy UOBYQA algorithm is a derivative-free unconstrained local optimization algorithm based on quadratic approximation adapted from Powell (2002) for use with noisy data (Deng and Ferris (2006)). Each function evaluation used the same random number seed and simulations were synchronized via the substream functionality of the RngStream random number generator (see L'Ecuyer

et al. (2002)).

We chose the initial policy coefficients used by the tuning methods via a "static" policy, or a policy where every ambulance is assigned a home base to which the ambulance returns each time it becomes available. Specifically, we selected the static policy π_0 yielding the best performance as estimated through Monte Carlo simulation over a large set of static policies. We then used π_0 to generate 30 two-week sample paths and collected the noisy estimates of the value function for π_0, $\hat{J}^{\pi_0}(s)$, for every s in the sample paths. Given the values of $\hat{J}^{\pi_0}(\cdot)$ from the simulation, we used (3.4) to calculate the policy coefficients $r^*_{\pi_0,2}$ (e.g., a single regression-based iteration) and these coefficients were used as the initial policy coefficients for the tuning methods. We also use the static policy π_0 as a benchmark policy to evaluate the potential benefits of using a redeployment policy over a policy which does not redeploy ambulances dynamically.

Each iteration along the x-axis of Figure 3.2 yields a different set of coefficients r, and one can pick the r that gives the best performance. Although each point is an unbiased estimate of performance using the associated coefficients, selecting the coefficients with the best performance introduces a selection bias. Consequently, each of these "minimizing" policies were reevaluated using independent random number streams to estimate their performance. The performance of these policies, expressed as 95% confidence intervals, are 28.6% $\pm$ .1% for least squares, 26.7% $\pm$ .1% for Nelder-Mead, and 24.8% $\pm$ .1% for NUOBYQA. The performance of the inital static policy was 28.4% $\pm$ .1%.

The least squares method found reasonable policies very quickly, but the performance of these policies did not significantly decrease with further tuning. Overall, the least squares method was unable to find a policy with better per-

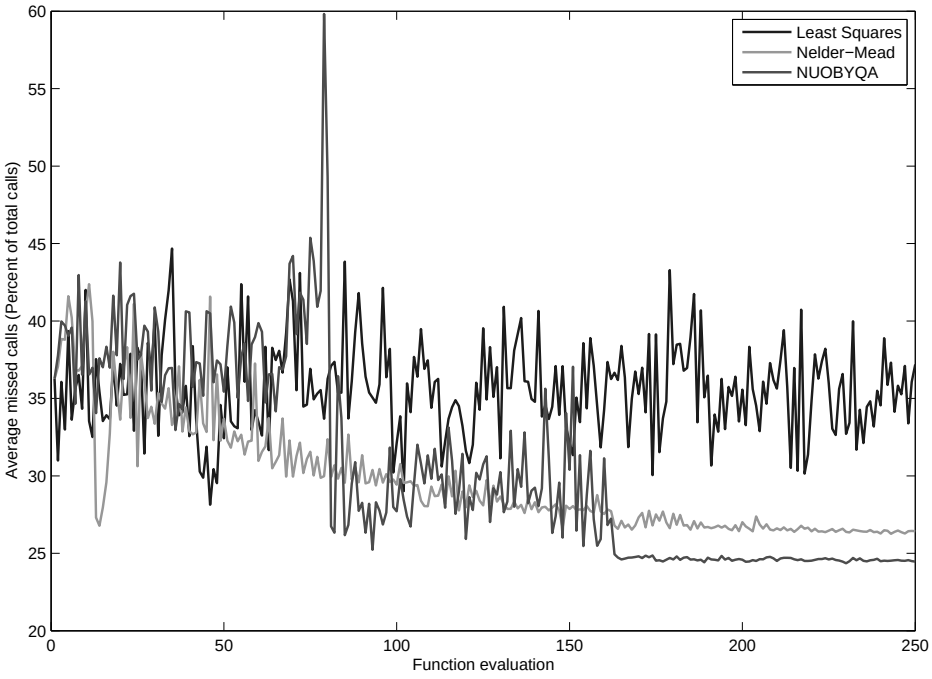

Figure 3.2: ADP Coefficient Tuning Results for Edmonton, Canada

formance than the initial static policy. The Nelder-Mead method also found a policy with good performance early on; however, ultimately this method converges to a local minimum having inferior performance to the policies found by the NUOBYQA method. The best policies were found with the NUOBYQA method and provide a 3.5% decrease in the percentage of lost calls as compared to the initial static policy.

A city-wide decrease of just 1% of lost calls is quite a significant improvement in the context of ambulance redeployment. A city the size of Edmonton would have to purchase, maintain, and staff an additional ambulance at a cost of approximately 1 million dollars a year to sustain such an improvement over baseline performance. Hence a reduction of 3.5% is quite a sizable difference

to EMS providers and indicates the large potential benefits of ambulance redeployment.

These results may appear to contradict those found in Section 2.5, which found policies out-performing the best known static polices via regression-based techniques, but the key difference between the results in Section 2.5 and our results is the approximation architecture used. Repeating the same analysis in this paper with the approximation architecture in Chapter 2 shows that all three methods find policies having about a 2-3% improvement over the static policy consistent with the results of Section 2.5. Furthermore, in this simulation the least squares method is able to identify such a policy well before the other two methods. As indicated by Section 3.3.1, the ability to properly tune an ADP algorithm through regression-based methods is highly dependent upon the approximation architecture used. The approximation architecture in Chapter 2 is perhaps more amenable to regression-based techniques, but ultimately the approximation architecure detailed in Section 3.4.2 is able to provide significant additional improvement to the ADP policy.

The time taken for the simulation-based ADP algorithm to make a single redeployment decision is about .07 seconds on a 2.66 Ghz processor. This time allotment is certainly within real-time operational feasibility for ambulance redeployment purposes. Unfortunately, the computational burden of this tuning procedure is quite extreme. Each function evaluation consists of 20 two-week simulations of ambulance redeployment and requires about 1 hour of computation time. The total tuning process required nearly 12 days of computation time for each method. Since ADP parameters only need to be tuned once before implementing an ADP policy, this is still a feasible method for ambulance rede-

ployment, but these tuning methods for simulation-based ADP are not likely to scale well for larger cities.

There may be situations where the dimension of r or other factors make direct search computationally prohibitive and regression- or LP-based methods must be used. Because of these extreme cases we feel that simulation optimization methods are marginalized and often ignored. This situation persists even though examples such as that in Section 3.3 and Maxwell et al. (2010a) show that there are limitations of standard methods not present in simulation-optimization methods and the gains from using simulation-optimization techniques can be enormous; see e.g., Szita and Lörincz (2006).

The key drawback of simulation-optimization methods is the computation associated with tuning. In Section 3.6 we show how this disadvantage can be overcome in the context of ambulance redeployment in a major metropolitan region.

3.6 Post-Decision State Formulation

Suppose that for any state S_k we can rewrite the immediate cost function $c(S_k, x, U_{k+1})$ and the system dynamics function $f(S_k, x, U_{k+1})$ in terms of the post-decision state $S_k^+(x)$ as $c(S_k^+(x), U_{k+1})$ and $f(S_k^+(x), U_{k+1})$ respectively. Let the post-decision value function be defined as

$$\tilde{J}(S_k^+(x)) = \mathbb{E}\left[c(S_k^+(x), U_{k+1}) + \alpha J(f(S_k^+(x), U_{k+1}))\right]. \tag{3.13}$$

Then we have that $J(S_k) = \min_{x \in \mathcal{X}(S_k)} \tilde{J}(S_k^+(x))$.

Let $\tilde{J}_r$ denote a linear approximation architecture for the post-decision state

where r represents the set of coefficients, and define

$$\tilde{L}_r(s) = \min_{x \in \mathcal{X}(s)} \tilde{J}_r(s^+(x)) \qquad \forall\, s \in \mathcal{S}. \tag{3.14}$$

The ADP policy with respect to $\tilde{L}_r$ is called the post-decision state ADP policy. The computational benefits of (3.14) over (3.3) are that the expectation operator is contained within the approximation $\tilde{J}_r$. Consequently, this formulation trades the computational burden of Monte Carlo simulation with a heavier reliance on the (post-decision) value function approximation. The post-decision state approximation $\tilde{J}_r$ is separate from the pre-decision state approximation J_r, and the two approximation architectures need not be similar. For ambulance redeployment, we have found that the approximation architecture in Section 3.4.2 works well for both approximations although the tuned coefficient parameters are unique to the respective approximation.

In Section 3.6.1 we define a generalization of the ADP policy, in Section 3.6.2 we show that the post-decision state policy is a limiting case of this generalization, and in Section 3.6.3 we show computational results for a direct search tuning method using the post-decision state policy.

3.6.1 Truncated Microsimulation Policy

An ADP policy in state S_k uses (3.3) to choose an action $x \in \mathcal{X}(S_k)$ based upon the expected value of the sum of the transition cost to S_{k+1} and the value function at state S_{k+1} given action x. However, if state S_{k+1} is similar to S_k there may be little new information contained in state S_{k+1}. For example, if S_{k+1} corresponds to an emergency call arrival occurring almost immediately after a service completion, state S_{k+1} does not contain very much information on the

evolution of the system after state S_k. As mentioned in Section 3.4.1, it is often convenient when creating MDP models to include non-decision states s where $\mathcal{X}(s) = \{\emptyset\}$. For situations where S_{k+1} corresponds to a non-decision state, (3.3) can be extended to capture more information on the evolution of the system by evaluating the expectation around the next decision state.

Let $Q = Q(S_k, x)$ denote a random variable indicating the number of non-decision states between state S_k and the next decision state given that we choose decision x is state S_k. Then the value function for S_k can be rewritten as

$$J(S_k) = \min_{x \in \mathcal{X}(S_k)} \mathbb{E}\left[\sum_{j=0}^{Q} \alpha^j c(S_{k+j}, x_j, U_{k+j+1}) + \alpha^{Q+1} J\left(f(S_{k+Q}, x_Q, U_{k+Q+1})\right)\right] \tag{3.15}$$

for all $S_k \in \mathcal{S}$, where $x_0 = x$ and $x_j = \emptyset$ for $1 \leq j < Q$ (see Appendix A.1). Using this formulation we define

$$L_{r,\infty}(S_k) = \min_{x \in \mathcal{X}(S_k)} \mathbb{E}\left[\sum_{j=0}^{Q} \alpha^j c(S_{k+j}, x_j, U_{k+j+1}) + \alpha^{Q+1} J_r\left(f(S_{k+Q}, x_Q, U_{k+Q+1})\right)\right]$$

for all $S_k \in \mathcal{S}$, and denote the ADP policy with respect to $L_{r,\infty}$ as the microsimulation ADP policy. The idea behind the microsimulation ADP policy is that instead of estimating the expectation through Monte Carlo samples of the next state as in (3.12), we use short simulations or "microsimulations" to sample transition costs and future state evolutions up until a decision state is reached. One practical drawback of this policy is that it is not known how long each microsimulation must run before reaching a decision state. When large values of Q are observed the computation time required to compute this policy may be prohibitive for real-time implementation even if the standard ADP policy can be computed rapidly. To overcome this drawback we select a deterministic time $\tau > 0$ and truncate each microsimulation at τ if it has not already stopped due to reaching a decision state.

Since the truncation time may occur at a non-event time we must appeal to the continuous version of the DEDS to express this ADP policy (see Section 3.4.1). Let $S_k(\tau, x)$ denote the random state $S(t(S_k) + \tau)$ given that $S(t(S_k)) = S_k^+(x)$ and let $S_k^+(\tau, x)$ denote the deterministic state $S_k(\tau, x)$ given that $t(S_{k+1}) > t(S_k) + \tau$, i.e., $S(t(S_k) + \tau)$ given that the last event before time $t(S_k) + \tau$ was in state S_k and that action x was chosen in S_k. Thus on the event that there are no decision events within τ time units after state $S_k^+(x)$ the system will be in the random state $S_k(\tau, x)$. If there are no events within τ time units after state $S_k^+(x)$ the system will be in the deterministic state $S_k^+(\tau, x)$.

Let $\gamma_1 = t(S_{k+1}) - t(S_k)$ and $\gamma_{Q+1} = t(S_{Q+1}) - t(S_k)$ denote the time before the next event and the time before the next decision event respectively. Let $Q_\tau = Q_\tau(S_k, x)$ denote a random variable indicating the number of non-decision states between S_k and the next decision state or the threshold time $t(S_k) + \tau$ (whichever comes first) given that we choose decision x is state S_k. Then the truncated microsimulation value function for all $S_k \in \mathcal{S}$ can be expressed as

$$
\begin{aligned}
J(S_k) = \min_{x \in \mathcal{X}(S_k)} & P(\tau \leq \gamma_1) \tilde{J}(S_k^+(\tau, x)) \\
& + P(\gamma_1 < \tau \leq \gamma_{Q+1}) \mathbb{E}\left[\sum_{j=0}^{Q_\tau} \alpha^j c(S_{k+j}, x_j, U_{k+j+1}) + \alpha^{Q_\tau + 1} \tilde{J}(S_k(\tau, x)) \middle| \gamma_1 < \tau \leq \gamma_{Q+1} \right] \\
& + P(\gamma_{Q+1} < \tau) \mathbb{E}\left[\sum_{j=0}^{Q} \alpha^j c(S_{k+j}, x_j, U_{k+j+1}) + \alpha^{Q+1} J(f(S_{k+Q}, x_Q, U_{k+Q+1})) \middle| \gamma_{Q+1} < \tau \right].
\end{aligned}
$$

The derivation of this formulation can be found in Appendix A.2. For all $S_k \in \mathcal{S}$, let

$$
\begin{aligned}
L_{r,r',\tau}(S_k) = \min_{x \in \mathcal{X}(S_k)} & P(\tau \leq \gamma_1) \tilde{J}_r(S_k^+(\tau, x)) \\
& + P(\gamma_1 < \tau \leq \gamma_{Q+1}) \mathbb{E}\left[\sum_{j=0}^{Q_\tau} \alpha^j c(S_{k+j}, x_j, U_{k+j+1}) + \alpha^{Q_\tau + 1} \tilde{J}_r(S_k(\tau, x)) \middle| \gamma_1 < \tau \leq \gamma_{Q+1} \right] \\
& + P(\gamma_{Q+1} < \tau) \mathbb{E}\left[\sum_{j=0}^{Q} \alpha^j c(S_{k+j}, x_j, U_{k+j+1}) + \alpha^{Q+1} J_{r'}(f(S_{k+Q}, x_Q, U_{k+Q+1})) \middle| \gamma_{Q+1} < \tau \right].
\end{aligned}
\tag{3.16}
$$

The ADP policy with respect to $L_{r,r',\tau}$ is called the truncated microsimulation

policy. Two different sets of coefficients r and r' must be used with this policy to account for the approximations of $\tilde{J}$ and J respectively.

3.6.2 Limiting Behavior of the Truncated Microsimulation Value Function Approximation

The truncated microsimulation policy attempts to balance the more precise estimation of longer microsimulations with the computational effort required by such simulations. Since the computational effort required to compute the truncated microsimulation policy generally increases with the threshold time τ it is natural to ask how the policy performs with reduced computational effort, i.e., as τ goes to zero.

The idea behind the limiting behavior argument is that the truncated microsimulation value function approximation $L_{r,r',\tau}$ should be very close to that of the post-decision value function approximation $\tilde{L}_r$ when τ is small because the system has had little time to change between the two evaluation points. This idea holds provided that we can show that the probability of having an event in the interval $[t(S_k), t(S_k) + \tau)$ goes to zero as τ goes to zero. Indeed, if we consider all sample paths starting at S_k^+, then we can always find a $\tau > 0$ such that τ is less than the time of the next state transition $t(S_{k+1})$ provided that $t(S_{k+1}) \neq t(S_k)$.

There are, however, some states where this is not true. If a clock $t_i(S_k)$ has reached the maximal value of the support of the associated distribution, then an immediate event will occur. Let $\mathcal{B}$ denote the set of states in $\mathcal{S}$ where this

happens. Assuming non-atomic interarrival distributions and service distributions, we can show that the probability of getting an immediate event starting from any state not in $\mathcal{B}$ is zero, and we can further show the following result, the proof of which is in Appendix A.3.

Proposition 3.3. *Assume the interarrival distributions and service distributions of the DEDS are non-atomic. If $P(S_0 \notin \mathcal{B}) = 1$ then $P(\exists k : t(S_{k+1}) = t(S_k)) = 0$, i.e., if S_0 is not in $\mathcal{B}$ w.p.1 then the probability of any two events occurring at the same time is zero.*

Proposition 3.3 implies that, assuming $P(S_0 \notin \mathcal{B}) = 1$, $t(S_0) < t(S_1) < t(S_2) < \ldots$ with probability 1. Whenever the event times are strictly increasing we can find a time τ_k s.t. $t(S_k) + \tau_k < t(S_{k+1})$ for any $k \geq 0$. Thus $\lim_{\tau \downarrow 0} P(t(S_k) + \tau < t(S_{k+1})) = 1$ for all $S_k \in \mathcal{S}$.

Assuming J, $\tilde{J}$, and c are bounded, we can bound the two expectations in $L_{r,r',\tau}$ (see (3.16)) by a function of $\mathbb{E}[Q]$, the expected number of non-decision events in $(t(S_k), t(S-k)+\tau)$. We further assume that the rate function of the arrival process is bounded, and hence, the expected number of arrivals on any finite interval is finite. The number of events on any interval can be bounded by twice the number of arrivals (one event for the arrival and one event for the service completion) plus a finite constant depending on the initial state. Thus the two expectations in $L_{r,r',\tau}$ must be finite. This is sufficient to state and prove our final result, the proof of which is given in Appendix A.3.

Theorem 3.4. *Assume that $P(S_0 \notin \mathcal{B}) = 1$, that $\tilde{J}_r$, for a fixed r, and c are bounded, and that $\tilde{J}_r(\cdot)$ is continuous in $T(\cdot)$ (for any fixed remaining state components). Then for any bounded ADP approximation architecture $J_{r'}$*

$$\lim_{\tau \downarrow 0} L_{r,r',\tau}(s) = \tilde{L}_r(s) \qquad \forall s \in \mathcal{S}.$$

Thus the function defining the post-decision state policy is the limit of the function defining the truncated microsimulation policy as the truncation time goes to zero (for any bounded ADP approximation architecture).

The key arguments in the proof of Theorem 3.4 are as follows. Since $L_{r,r',\tau}$ is a minimum over a finite number of convergent sequences we can interchange the order of the limit and the minimum in $L_{r,r',\tau}$. By Proposition 3.3 and the fact that the expectations of $L_{r,r',\tau}$ are finite,

$$\begin{aligned}
\lim_{\tau \downarrow 0} L_{r,r',\tau}(S_k) &= \min_{x \in \mathcal{X}(S_k)} \lim_{\tau \downarrow 0} \tilde{J}_r(S_k^+(\tau, x)) \\
&= \min_{x \in \mathcal{X}(S_k)} \tilde{J}_r(\lim_{\tau \downarrow 0} S_k^+(\tau, x)) \qquad \text{by the continuity assumption on } \tilde{J} \\
&= \min_{x \in \mathcal{X}(S_k)} \tilde{J}_r(S_k^+(x)) \qquad \text{by the construction in Section 3.4.1.}
\end{aligned}$$

Thus as the simulation threshold time goes to zero the function defining the truncated microsimulation policy converges to the function defining the post-decision state policy. This theorem holds under very general conditions that are not specific to the ambulance redeployment problem.

The post-decision state ADP policy has the overwhelming computational advantage of not needing to perform Monte Carlo microsimulations to approximate expectations. Given an approximation architecture $\tilde{J}_r$, the only computation required to make a decision is computing the minimum of $\tilde{J}_r$ over a finite (and reasonably small) set. Consequently, tuning post-decision state policies with direct search methods becomes computationally feasible even with much higher dimensional problems than those of Section 3.5.

3.6.3 Computational Results

Melbourne is the second largest city in Australia with about 4 million residents. We model the ambulance operations of Melbourne using a discrete-event simulation having 97 ambulances, 87 ambulance bases, and 22 hospitals. The emergency call arrival model divides the Melbourne area into 10,000 grid cells and time-dependent arrival rates are based on historical data. Additionally, the ambulances follow shift schedules similar to those found in practice. The travel model used in the simulation is a deterministic shortest-path calculation on a detailed road network. Again, this model is realistic but not real.

The ADP policies used in this section are defined by the post-decision state ADP formulation defined in (3.14). The parameterized approximation architecture $\tilde{J}_r$ used by the post-decision state ADP has the same form as the approximation architecture J_r (as described in Section 3.4.2); however, the coefficients for these two approximation architectures are distinct.

Figure 3.3 shows the results of the least squares and Nelder-Mead tuning methods for Melbourne using the post-decision state ADP formulation. Since this approximation architecture has one basis function per ambulance base there are 87 basis functions used in the approximation architecture, and the ADP policy parameter tuning problem is an 87 dimensional problem. In this high-dimensional space the NUOBYQA method is computationally infeasible and hence it is not included in the results. The static policy used to generate the initial policy for the tuning approach has 26.7% $\pm$ .1% lost calls. The best policy found by the least squares based fitting procedure has 31.3% $\pm$ .1% lost calls which is significantly worse than the inital static policy. The Nelder-Mead search method was able to find a policy with 25.8% $\pm$ .1% lost calls, a practically

significant improvement over the static policy of about 0.9%.

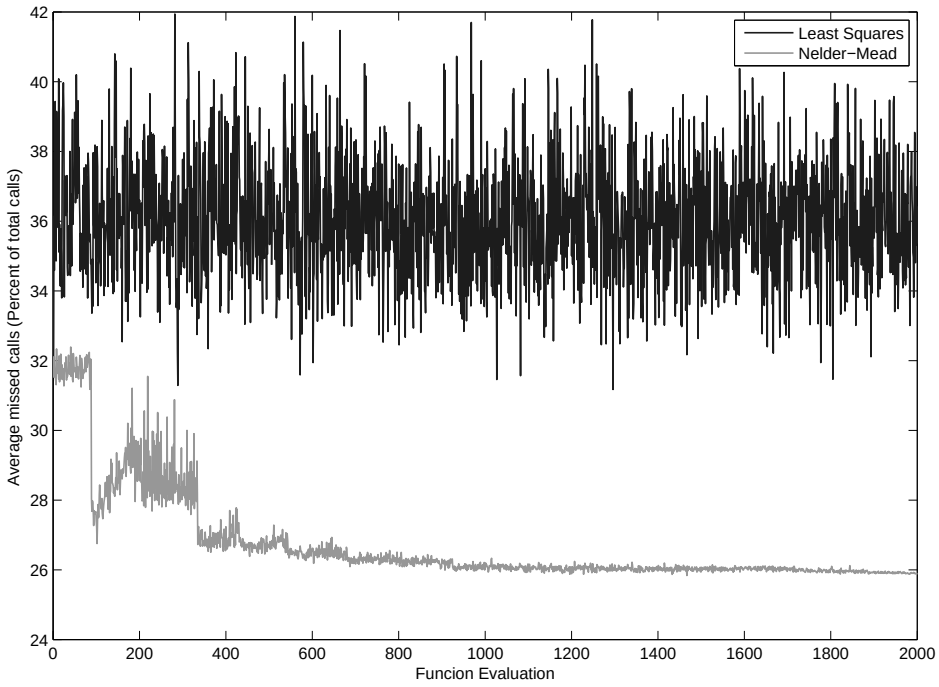

Figure 3.3: ADP Coefficient Tuning Results for Melbourne, Australia

The direct search tuning in Figure 3.3 required approximately 12.5 hours of computation time on a 2.66 Ghz processor whereas one function evaluation of the simulation-based ADP policy on Melbourne requires over 5.5 hours. Without the post-decision state formulation such an extensive tuning process would have required over one year of computation time given the same processing capacities. Thus a direct search tuning method coupled with a post-decision state formulation is able to decrease the computational effort used to tune the ADP policy while simultaneously finding superior policies.

3.7 Conclusion

Ambulance redeployment policies can improve EMS response times within current resources. By altering the positioning of ambulances in real-time EMS providers are better able to respond to gaps in coverage that occur due to inherent uncertainties involved in emergency services. ADP is an appealing framework for implementing ambulance redeployment because it directly accounts for the randomness in the system while maintaining computational feasibility for real-time decision support.

Unfortunately, standard tuning approaches for ADP policies have limitations that may prevent the resulting ADP policies from performing as well as possible within a given approximation architecture. These limitations were shown both theoretically for a sample problem and empirically for two different ambulance redeployment scenarios. In both of these scenarios direct search methods were able to find ADP policies with significantly better performance than the best policies found otherwise.

The benefit of direct-search tuning is that it tunes ADP policy parameters based directly on the performance resulting from those parameters, rather than indirectly through value-function fitting. This benefit, however, comes with the cost of higher computational requirements than typically required for other approaches. Using a post-decision state representation the computational burden for tuning associated ADP policies is within computational limitations–even in high-dimensional tuning problems.

The post-decision state formulation of a problem may seem foreign and perhaps forced at first, but we show that, under general conditions, the post-

decision state ADP policy is actually the limit of a general truncated microsimulation policy which is based on the standard simulation-based ADP formulation. As such, the post-decision state ADP policy can be viewed as the limiting policy of simulation-based ADP policies when the computational budget for simulations goes to zero.

CHAPTER 4

EQUIVALENCE RESULTS FOR APPROXIMATE DYNAMIC PROGRAMMING AND COMPLIANCE TABLE POLICIES FOR AMBULANCE REDEPLOYMENT

4.1 Introduction

Ambulance redeployment is the practice of repositioning ambulances based on real-time information to provide better emergency medical service (EMS) throughout a given region. One example of this practice would be relocating an idle ambulance to an area with no ambulances currently available so that the EMS provider could respond quickly to an additional call that may arrive in that area.

One common method used for ambulance redeployment is a compliance table (CT) policy. A CT policy dictates the number of ambulances that should be assigned to each base given the number of ambulances currently available. The system is said to be *in compliance* (with respect to a given CT policy) when each base has the appropriate number of ambulances assigned to it. In practice, a system is considered compliant when the ambulances are actually *located* at the designated bases, but for simplicity we consider the system to be compliant as soon as ambulances are *assigned* to the designated bases although the ambulances may still be in transit.

The underlying idea behind a CT policy is to ensure ambulances are positioned to achieve the best performance possible given the number of currently available ambulances. One shortcoming of this approach is that each time an

ambulance becomes busy or becomes free all available ambulances may be required to move to reach compliance. Ambulance crews find frequent relocations extremely frustrating. Consequently, nested compliance table (NCT) policies, CT policies designed so that at most one ambulance must relocate to reach compliance when the number of available ambulances changes, are used. NCT policies based upon deterministic models of ambulance redeployment can be obtained using integer programming; however, the integer programs are hard to solve for large EMS operations (Gendreau et al. (2006)). Alanis et al. (2010) gives another approach that approximates the performance of a fixed NCT policy quickly and can be used within a search procedure.

An alternative method uses approximate dynamic programming (ADP) to compute redeployment decisions dynamically as ambulances become busy or free as in Chapters 2 and 3. The ADP method models ambulance redeployment as a Markov decision process and approximates the value function of this model. Similar to a dynamic programming (DP) policy, an ADP policy makes redeployment decisions to minimize the sum of immediate costs and expected future costs, as represented by the approximate value function. ADP policies have the advantage of being able to use full state information, including the location and status of each ambulances (e.g., at call scene, transporting patient to hospital), when making redeployment decisions as opposed to CT policies which only use the number of ambulances assigned to each base when making redeployment decisions. Disadvantages of implementing ADP policies include the necessity of approximating the value function and the complexity of the resulting policies as compared to NCT policies.

Bridging the gap between these two methods, we define a class of ADP poli-

cies for ambulance redeployment called ADP-CT polices and show that they are equivalent to the class of NCT policies. The equivalence of these two classes of policies, and the transformations between them, are important because they provide a means to design NCT policies using standard ADP tuning methods such as approximate linear programming (Schweitzer and Seidmann (1985) and de Farias and Van Roy (2003)), temporal-difference learning (e.g., Sutton (1988)), and others, or direct-search simulation optimization methods on a continuous space (e.g., as in Chapter 3). Once an acceptable ADP-CT policy has been found, the equivalent NCT policy can be readily computed and used as a policy that is intuitive, easy to use, and easy to implement.

It is not uncommon for an EMS system to be out of compliance for a period of time at a dispatcher's discretion due to information that may not be contained within the ambulance redeployment model (e.g., heavy traffic conditions, special events, additional ambulances that will be available soon). NCT policies provide no systematic way to return to a compliant state. Often EMS dispatchers must rely on their own intuition to return the system to compliance. In contrast, if started in a non-compliant state (i.e., there is at least one base that does not have the appropriate number of ambulances assigned to it), we show that an ADP-CT policy will return to a compliant state after at most D redeployments, where D has a simple expression.

In Section 4.2 we formally define the NCT and ADP-CT policies. In Section 4.3 we show that, provided the system is in compliance, the two classes of policies are equivalent, and we define procedures that can be used to transform between an ADP-CT policy and the equivalent NCT policy. In Section 4.4 we examine the conditions required to define an ADP-CT policy and show that if

the system is out of compliance an ADP-CT policy will naturally return to a state of compliance. Section 4.5 concludes.

4.2 Problem formulation

Consider an ambulance redeployment problem having N ambulances and B bases, or locations at which an ambulance may sit idle waiting for future calls.

Although unnecessary for our results, we believe it worthwhile to understand how a typical ambulance redeployment model may be constructed. When a call arrives it is assigned to the closest available ambulance. If a call arrives and there are no ambulances available the call is placed on a waiting list. Calls on the waiting list are served in the order of arrival by ambulances as they become free. After being assigned a call the ambulance travels to the call location and spends a random amount of time treating the patient at the scene. Some patient treatments will be completed at the scene, while others will require transport to a hospital. In the former case the ambulance becomes available to serve other calls. In the latter case, the ambulance transports the patient to a hospital and spends time transferring the patient to the hospital staff. After this is completed the ambulance becomes available to serve other calls. In either case, the newly freed ambulance will travel to a base as dictated by the redeployment policy if there are no calls waiting.

One main goal of EMS providers is to reduce the response time, i.e., the time elapsed from when an emergency call was received to the time when an ambulance arrives on scene.

Chapter 2 provides additional details on an ambulance redeployment model formulation, and Maxwell et al. (2009) gives additional information on simulation procedures.

4.2.1 NCT policy formulation

To formalize the concept of an NCT policy we let the allocation

$$\mathcal{A}_n = (a_n(1), \ldots, a_n(B)) \quad \text{s.t.} \sum_{b=1}^{B} a_n(b) = n$$

denote the number of ambulances, $a_n(b)$, assigned to base b (either idle at b or traveling to b) where n is the number of ambulances available. An NCT is defined as a collection of allocations $\{\mathcal{A}_n\}_{n=0}^{N}$ satisfying

$$\sum_{b=1}^{B} |a_n(b) - a_{n-1}(b)| = 1 \quad \text{for } 0 < n \leq N, \tag{4.1}$$

and an NCT policy redeploys ambulances according to a given NCT. Condition (4.1) ensures that if an ambulance becomes busy or becomes free, then at most one ambulance must move for the system to remain in compliance. For simplicity, we assume that the NCT policy makes redeployment decisions involving as few ambulances as possible (i.e., at most one ambulance is redeployed at each decision point).

4.2.2 ADP-CT policy formulation

The ADP-CT policy formulation requires a post-decision state representation of approximate dynamic programming. We keep the amount of detail given below on this formulation brief, but Powell (2007) gives an extensive explanation

of the topic and Chapter 3 gives a formulation specific to ambulance redeployment. The post-decision state $S_t^+(x)$ at a time t is the state the system enters immediately after choosing action $x \in \mathcal{X}(S_t)$ from state S_t. We then choose the action x that achieves the minimum in

$$\min_{x \in \mathcal{X}(S_t)} \sum_{k=1}^{K} \gamma_k f_k(S_t^+(x)). \tag{4.2}$$

Here f_k are basis functions that attempt to capture the essence of the problem, and γ_k are tunable parameters.

We wish to establish a relationship between ADP policies and NCT policies, so we assume that the ADP policy only makes decisions when an ambulance becomes either free or busy. We refer to these two decisions as assignment and redeployment decisions respectively.

Let $(n_1, \ldots, n_B)$ denote the number of ambulances assigned to base $b = 1, \ldots, B$ when the system is in state S_t. Suppose we can rewrite (4.2) so that we have one basis function ϕ_b for each base b, and each basis function only depends upon the number of ambulances assigned to base b in state $S_t^+(x)$. Thus for assignment decisions we can rewrite (4.2) as

$$\min_{r \in \{1,\ldots,B\}} \sum_{b=1}^{B} \alpha_b \phi_b(n_b + \mathbb{1}_{\{b=r\}}), \tag{4.3}$$

where $\phi_b(n_b)$ represents the basis function for base b for any state in which n_b ambulances are assigned to base b, r denotes the base to which the ambulance will be assigned, and α_b are tunable parameters. Similarly, for redeployment decisions we can rewrite (4.2) as

$$\min_{r_o, r_d \in \{1,\ldots,B\}} \sum_{b=1}^{B} \alpha_b \phi_b(n_b - \mathbb{1}_{\{b=r_o\}} + \mathbb{1}_{\{b=r_d\}}), \tag{4.4}$$

where r_o denotes the base from which we redeploy an ambulance (the redeployment origin) and r_d denotes the base to which we redeploy the ambulance (the redeployment destination). Chapter 3 gives an example of a post-decision state ADP policy for ambulance redeployment satisfying these conditions on ϕ_b and discusses a method to tune the α_b parameters.

For the assignment decision, (4.3), n_b denotes the number of available ambulances assigned to base b before the newly freed ambulance has been assigned to a base, and for the redeployment decision, (4.4), n_b denotes the number of available ambulances assigned to base b after the ambulance assigned to the new call becomes busy but before any redeployment decision is made. If there are multiple optimal solutions in (4.3) we assume a lexicographical ordering on n_r then r and in (4.4) we assume a reverse lexicographical order on n_{r_o} then r_o and then a lexicographical ordering on n_{r_d} then r_d. If the optimal solution of (4.4), say, r_o^* and r_d^*, satisfies $r_o^* = r_d^*$ then no redeployment is made (as no improvement in the approximate value function can be attained through redeployment). Let

$$\phi_b^+(n_b) = \alpha_b \left(\phi_b(n_b) - \phi_b(n_b + 1)\right),$$

and assume that $\phi_b^+(n_b)$ is non-increasing in n_b for each b. For convenience we let $\phi_b^+(n_b) = \infty$ for $n_b < 0$. An ADP policy satisfying these assumptions is called an ADP-CT policy.

The assignment and redeployment decisions made by an ADP-CT policy may be rewritten in terms of ϕ_b^+. For the assignment decision, (4.3) can be written as

$$\sum_{b=1}^{B} \alpha_b \phi_b(n_b) - \max_{r \in \{1,\ldots,B\}} \phi_r^+(n_r), \tag{4.5}$$

and for the redeployment decision, (4.4) can be written as

$$\sum_{b=1}^{B} \alpha_b \phi_b(n_b) - \max_{r_o, r_d \in \{1,\ldots,B\}} y(r_o, r_d) \tag{4.6}$$

where

$$y(r_o, r_d) = \begin{cases} 0 & \text{if } r_o = r_d \\ \phi^+_{r_d}(n_{r_d}) - \phi^+_{r_o}(n_{r_o} - 1) & \text{otherwise} \end{cases}.$$

4.3 Equivalence of ADP-CT and NCT policies

To establish the equivalence of ADP-CT and NCT policies we define a new ambulance redeployment policy, the total ordering (TO) policy. The TO policy is not of interest on its own, but will prove useful in showing the equivalence of ADP-CT and NCT policies. Next, we define transformations that convert an NCT policy to a TO policy (and vice versa) and an ADP-CT policy to a TO policy (and vice versa). These transformations allow us to show that when starting in compliance, a policy is an ADP-CT policy if and only if it is an NCT policy.

To define the TO policy, we let the TO matrix $T = [t_{i,j}]$ denote a $B \times N$ matrix such that each row is non-increasing, i.e., $t_{b,n} \geq t_{b,n'}$ for all $n' > n$. Let $g(k) \in \{1, \ldots, B\}$ denote the row containing the kth largest element of T (assume lexicographical ordering on b then n for ties).

Similar to the ADP-CT policy, the TO policy makes an assignment decision when an ambulance becomes free and a redeployment decision when an ambulance becomes busy. For a TO policy these decisions are made as follows.

- **Assignment:** If there are $n < N$ ambulances available and another ambulance becomes free, assign it to base $g(n + 1)$.

- **Redeployment:** If there are $n > 0$ ambulances available and an ambulance assigned to base b becomes busy (leaving $n-1$ available ambulances) and $g(n) \neq b$ then redeploy an ambulance from base $g(n)$ to b.

4.3.1 NCT and TO policy transformations

Let J denote the transformation from a TO policy p to an NCT policy $J(p)$ defined as follows:

1. Set $\mathcal{A}_0 = (0, \ldots, 0)$,
2. For $n = 1, \ldots, N$, set

$$\mathcal{A}_n = (a_{n-1}(1) + e_n(1), \ldots, a_{n-1}(B) + e_n(B))$$

   where $e_n(b) = \mathbb{1}_{\{g(n)=b\}}$.

Since $\sum_{b=1}^{B} a_n(b) = n$ for $0 \leq n \leq N$ and $\sum_{b=1}^{B} |a_n(b) - a_{n-1}(b)| = 1$ for $0 < n \leq N$, $\{\mathcal{A}_n\}_{n=1}^{N}$ defines an NCT policy. Let J^{-1} denote the transformation from an NCT policy p to a TO policy $J^{-1}(p)$ defined as follows:

1. Initialize T as a $B \times N$ matrix of zeros,
2. For $n = 1, \ldots, N$, set $t_{b,k_b} = N - n + 1$ where b satisfies $a_n(b) > a_{n-1}(b)$ and k_b is the smallest integer such that $t_{b,k} = 0$.

Since the rows of T are non-increasing we have that T is a total order matrix which defines a TO policy. The system is said to be in compliance with respect to a TO policy p when it is in compliance with respect to the NCT policy $J(p)$.

Lemma 4.1. *Provided the system starts in compliance, a redeployment policy is an NCT policy if and only if it is a TO policy.*

The key argument in Lemma 4.1 is that the transformations J and J^{-1} produce equivalent policies so that given any NCT or TO policy we can find an equivalent TO or NCT policy respectively by applying the appropriate transformation. The proof of Lemma 4.1 is contained in B.1.

4.3.2 ADP-CT and TO transformations

Let K denote the transformation from an ADP-CT policy p to the TO policy $K(p)$ defined by setting

$$T = \begin{pmatrix} \phi_1^+(0) & \dots & \phi_1^+(N-1) \\ \vdots & & \vdots \\ \phi_B^+(0) & \dots & \phi_B^+(N-1) \end{pmatrix}.$$

Since the ϕ_b^+ are non-increasing we have that the rows of T are also non-increasing, hence T is a total order matrix and defines a TO policy. Let K^{-1} denote the transformation from a TO policy p to the ADP-CT policy $K^{-1}(p)$ defined by setting all $\alpha_b = 1$ and setting $\phi_b(n) = \sum_{i=n+1}^{N} t_{n,i}$ for $b = 1, \dots, B$, $n = 0, \dots, N$. Then

$$\begin{gathered} \phi_b(0) - \phi_b(1) = \phi_b^+(0) = t_{b,1} \\ \vdots \\ \phi_b(N-1) - \phi_b(N) = \phi_b^+(N-1) = t_{b,N} \\ \phi_b(N) = 0. \end{gathered}$$

Since the entries of each row of T are non-increasing we have that ϕ_b^+ is non-increasing. Thus this definition of α_b and ϕ_b define an ADP-CT. The system

is said to be in compliance with respect to a ADP-CT policy p when it is in compliance with respect to the NCT policy $J(K(p))$.

Lemma 4.2. *Provided that the system starts in compliance, a redeployment policy is an ADP-CT policy if and only if it is a TO policy.*

The TO policy ensures that if there are n ambulances available they are assigned to the bases corresponding to the n largest elements of the TO matrix. The ADP-CT policy makes decisions by maximizing specific functions of ϕ_b^+ and n_b. The key concept of Lemma 4.2 is showing that the ADP-CT maximizations result in the same decisions as the TO policy. The proof of Lemma 4.2 is contained in B.1.

4.3.3 Equivalence results

Using the policy transformations in Section 4.3.1 and Section 4.3.2 we can show the following equivalence result for ADP-CT and NCT policies.

Theorem 4.3. *Provided the system starts in compliance, a policy is an ADP-CT policy if and only if it is an NCT policy.*

Proof. Follows directly from Lemma 4.1 and Lemma 4.2. □

4.4 Additional results

4.4.1 Interpreting ϕ_b^+ values

As illustrated by (4.5) and (4.6), decisions in an ADP-CT policy are determined by maximization over ϕ_b^+ values. Intuitively $\phi_b^+(n_b)$ represents the expected benefit achieved by assigning an additional ambulance to base b when there are already n_b ambulances stationed there, where benefit is measured by the basis function ϕ_b and weighted by α_b.

This interpretation of ϕ_b^+ is somewhat limited because any ADP-CT policy with an identical ordering of $\phi_b^+(n_b)$ values for $b = 1, \dots B$ and $n_b = 0, \dots N$ would be an identical policy—even if some or all of the ϕ_b^+ values were negative. On the other hand, it is always possible to add a constant to each ϕ_b function to obtain an equivalent policy where the ϕ_b^+ values are all non-negative.

4.4.2 Necessity of non-increasing ϕ_b^+

Consider relaxing the constraint that $\phi_b^+(n_b)$ be non-increasing for an ADP-CT policy. Then consider an ADP-CT policy defined by the matrix

$$\begin{pmatrix} \phi_1^+(0) & \phi_1^+(1) & \phi_1^+(2) \\ \phi_2^+(0) & \phi_2^+(1) & \phi_2^+(2) \end{pmatrix} = \begin{pmatrix} 2 & 4 & 1 \\ 3 & 1 & 1 \end{pmatrix}.$$

Starting from zero available ambulances we assign the first ambulance to base 2, the second ambulance to base 1, and the third ambulance to base 1 as well. In particular, this implies that the NCT policy $J(K(p))$ has allocations

$\mathcal{A}_1 = (0, 1)$, $\mathcal{A}_2 = (1, 1)$, and $\mathcal{A}_3 = (2, 1)$.

Now consider the case when there are three ambulances available in the compliant allocation $\mathcal{A}_3 = (2, 1)$ and an ambulance from base 1 is assigned to a call. So $n = 2$ and $(n_1, n_2) = (1, 1)$. At this point the ADP-CT policy makes a redeployment decision to move an ambulance from base 2 to base 1 since $\phi_1^+(1) - \phi_2^+(0) = 4 - 3 = 1 > 0$. This results in both ambulances at base 1 which contradicts $\mathcal{A}_2$. Hence this policy does not define an NCT policy.

The key point is that since the ADP-CT policy only maximizes over a subset of the corresponding TO matrix at each decision (rather than considering the total ordering of the elements in the matrix) we must require that each row of the TO matrix be non-increasing.

4.4.3 Convex functions ϕ_b satisfy non-increasing $\phi_b^+(n_b)$

Assume $\alpha_b > 0$. We must show $\phi_b^+(n_b) \geq \phi_b^+(n_b + 1)$, i.e.,

$$\begin{aligned} &\alpha_b(\phi_b(n_b) - \phi_b(n_b + 1)) \geq \alpha_b(\phi_b(n_b + 1) - \phi_b(n_b + 2)) \\ \iff\ &\phi_b(n_b) - \phi_b(n_b + 1) \geq \phi_b(n_b + 1) - \phi_b(n_b + 2) \end{aligned}$$

for $0 \leq n_b \leq N - 2$. If ϕ_b is a convex function (or midpoint convex function) we know that

$$\phi_b(n_b + 1) \leq \frac{\phi_b(n_b) + \phi_b(n_b + 2)}{2},$$

and the desired result holds. Likewise, if $\alpha_b < 0$ a concave ϕ_b will satisfy non-increasing $\phi_b^+(n_b)$.

Messerli (1972) shows that the Erlang loss of a queue $B(n, a)$ with n servers and offered load a satisfies

$$B(n, a) - B(n + 1, a) > B(n + 1, a) - B(n + 2, a).$$

Thus for $\phi_b(n_b) = B(n_b, a_b)$, where a_b denotes the offered load for base b, and non-negative α_b we have that that $\phi_b^+(n_b)$ is non-increasing. In particular, this implies that if we assume emergency calls arrive according to a homogeneous Poisson process and that the call location is randomly chosen according to a fixed probability distribution, then the ADP policy in Maxwell et al. (2010a) is an ADP-CT policy and hence equivalent to an NCT policy.

4.4.4 Out-of-compliance robustness

If started from a compliant state, the NCT and ADP-CT policies, as defined in Section 4.2, will always remain in a compliant state. In practice, however, it is quite common that EMS dispatchers do not strictly follow compliance table recommendations and the system enters a state of non-compliance. After a period of time of non-compliance it is often desirable to return to a compliant state.

Unfortunately, an NCT policy does not prescribe a systematic way of reaching compliance; hence dispatchers are often left to use their own judgment to return to compliance. On the other hand, when initiated out of compliance an ADP-CT policy will return to a state of compliance after at most N redeployments.

Theorem 4.4. *Given a current allocation* $(h_1, \ldots, h_B)$ *where* $\sum_{b=1}^{B} h_b = n$ *and the compliant allocation for* n *free ambulances* $(n_1, \ldots, n_B)$*, an ADP-CT policy returns to*

compliance after at most D redeployments, where $D = \sum_{b=1}^{B} [n_b - h_b]^+$ is the total number of ambulances needed to bring the under-staffed bases into compliance.

Proof. The current allocation can only change through an assignment or redeployment decision.

- **Assignment:** When an ambulance becomes free it will be sent to the base r achieving $\max_r \phi_r^+(h_r)$. If $D > 0$ we know that r corresponds to a base that has fewer ambulances assigned to it than the compliance allocation dictates. By performing the assignment we reduce D by one; however, since there are now $n + 1$ ambulances available a new compliance allocation is used, and this new allocation may cause D to increase by one if $d_{g(n+1)} \leq n_{g(n+1)}$, i.e., if the base to which the $(n + 1)$th ambulance would normally be assigned to under compliance is not currently over-staffed. Hence the number of discrepancies from an assignment either decreases or stays the same.
- **Redeployment:** When an ambulance becomes busy by responding to a call an ambulance will be redeployed to base r_d from r_o where r_d and r_0 achieve the maximum of (4.6). If $D > 0$ we know that $\phi_{r_d}^+(h_{r_d})$ will be maximized by a base r_d with too few ambulances assigned to it (i.e., $h_{r_d} < n_{r_d}$), and $\phi_{r_o}^+(n_{r_o} - 1)$ will be minimized by a base r_o having too many ambulances assigned to it (i.e., $h_{r_o} > n_{r_o}$). With the ambulance becoming busy, the compliance allocation moves from having n ambulances to $n - 1$ ambulances; however, this change cannot cause a base to become more under-staffed than it already is–D either decreases by one or remains the same. Thus, as a result of the redeployment, D is decreased by at least one.

Hence after at most D redeployments we have that $D = 0$ and the system is in compliance. □

4.5 Future Work

The equivalence relationship between ADP-CT and NCT policies opens up a number of possibilities for future work including designing specific search procedures for NCT policies via ADP tuning methods and for adapting NCT policies to time-dependencies inherent in actual systems. For example, when considering two sets of ADP coefficients, one may check a simple system of inequalities to determine if the coefficients induce the same NCT. This verification process could be integrated into simulation optimization routines to guide search procedures and stopping rules. Similarly, given a specified NCT policy, one may solve a system of linear inequalities to find ADP-CT coefficients that induce that NCT policy (if feasible) for a given set of basis functions. One may also choose to optimize over a set of partial NCT policies. For example, local regulations or political concerns may dictate the positions of the first five ambulances in the NCT or specify that a given base must have at least as many ambulances as some other base. These restrictions can be added as additional linear inequalities to test the feasibility of operating such a system as an ADP-CT (for a set of basis functions), and if feasible, the set of solutions to the inequalities dictates a search space for optimization procedures to find the best NCT satisfying the constraints.

Additionally, the result of Theorem 4.4 provides an elegant way to handle the time-dependent call arrival process of actual EMS systems. One common

approach for such systems is to formulate different NCTs for different periods of the day or week. However, as in out-of-compliance situations, dispatchers are left to their best judgment on how to move from one NCT to another at the appropriate time. The behavior of multiple NCTs can be duplicated by changing the coefficients in an ADP-CT value function approximation, and Theorem 4.4 guarantees that the system will return to compliance after a small number of redeployments without dispatcher intervention. Alternatively, one may design time-dependent basis functions and formulate a set of linear inequalities to find a single set of coefficients (if any) that will induce the appropriate NCTs at the appropriate times.

CHAPTER 5

PERFORMANCE BOUNDS FOR AMBULANCE REDEPLOYMENT

5.1 Introduction

Emergency medical service (EMS) providers are responsible for responding to emergency calls, providing medical assistance to patients, and transporting patients to hospitals as necessary. Typically, EMS providers are required by contract to respond to a certain percentage of emergency calls within a specified time threshold. A common measure of performance is the percent of "lost" calls, or calls having response times, i.e., the time between when an emergency call is received and the time an ambulance first arrives at the call scene, greater than the performance threshold, say 8 minutes, in a set time interval, say one month. One method used by EMS providers to reduce response times is that of ambulance redeployment. Ambulance redeployment is the practice of repositioning idle ambulances based on real-time information in an attempt to reduce expected response times for future calls.

The ambulance redeployment literature provides multiple methods to compute ambulance redeployment policies. Gendreau et al. (2001) use a deterministic model to formulate a mixed-integer program that is solved in real-time whenever a redeployment decision is required (see also Brotcorne et al. (2003), Richards (2007), and Nair and Miller-Hooks (2009)). Solving this mixed-integer program can be computationally intense and a parallel computing system or heuristic solution method is typically used to make redeployment decisions within real-time constraints.

Another approach is to calculate recommended locations for ambulances conditional on how many ambulances are available. Dispatchers can then use these pre-computed solutions to assign ambulances in real-time. One disadvantage of this approach is that the entire fleet of available ambulances may be required to move each time an ambulance becomes free or becomes busy. Consequently, the solutions are usually constructed so that no more than one ambulance must move during each redeployment. Policies of this type are called nested compliance table (NCT) policies. Gendreau et al. (2006) uses an integer program formulation to compute optimal NCT policies based upon a deterministic ambulance redeployment model; however, the complexity of the integer program limits the application of this approach to large EMS systems. Alanis et al. (2010) gives a method to approximate the performance of an NCT policy quickly, and this approximation could be used within a search procedure to identify potential policies to evaluate further using highly detailed discrete-event simulation models.

Berman (1981a,c,b) formulates the ambulance redeployment problem as a dynamic program (DP) and solves for an optimal redeployment policy. One advantage of the DP approach is that it models the stochastic nature of the system directly; however, due to computational difficulties arising from computing DP policies in large state spaces this approach is only feasible for very small applications (e.g., one or two ambulances). More recent work further develops the DP formulation in an attempt to gain insight into the structure of an optimal policy (Zhang et al. (2010)) and to extend these results to larger systems (Zhang (2010)).

To construct computationally tractable redeployment policies, Chapter 2 and

Chapter 3 formulate ambulance redeployment as a Markov decision process and use approximate dynamic programming (ADP) to compute a redeployment policy. Andersson (2005) and Andersson and Vaerband (2007) use a similar approach via a heuristic "preparedness" function.

One difficulty common to all these approaches is obtaining a lower bound on an optimal ambulance redeployment policy performance for reasonably sized EMS systems. With the absence of a lower bound, policies are often compared to benchmark policies to evaluate improvement. This is an effective way to compare alternative policies but does not give any indication on how far these policies may be from optimality.

A useful guide when evaluating the performance of a policy is the proportion of "unreachable" calls, or the proportion of call arrivals that are too far from all ambulance bases to be reached within the time threshold. Although useful as a guide, the proportion of unreachable calls does not provide a lower bound because an ambulance may respond to an emergency call while it is on the road (i.e., not located at a base) and hence may respond to an unreachable call within the time threshold.

Apart from using a lower bound as a benchmark to evaluate policies, a lower bound on performance can also be used to calculate the minimal amount of resources required to reach desired performance levels. Typically EMS providers are required to bid on contracts that specify the maximum fraction of lost calls allowed for different call types over a given time interval. To accurately submit a bid, an EMS provider must estimate the resources required, e.g., ambulances and ambulance crews, to meet the specified performance levels. Lower bound calculations on the number of resources needed can then be used to guide the

EMS provider in evaluating their resource requirements. Such lower bounds could also be used by municipalities to evaluate the feasibility of bids submitted by different EMS providers seeking the EMS contract.

In this paper, we formulate a computationally tractable lower bound on the expected fraction of lost calls over a finite simulated time horizon. As far as we are aware, this paper gives the first such lower bound. We derive this bound by modeling the ambulance redeployment system as a multiserver queue where servers represent ambulances and arrivals represent emergency calls. We then construct a stochastic lower bound on the response time distribution of any ambulance redeployment policy and use this "optimistic" response time distribution to estimate the expected fraction of call arrivals when there are a given number of ambulances available. We calculate the ambulance placements that maximize coverage, i.e., the probability of reaching the next call within the time threshold, conditional on the number of available ambulances, and apply known comparison results for multiserver queues to show that the weighted sum of these quantities is a lower bound on the performance of any redeployment policy.

The lower bound calculation supposes that ambulances are always positioned throughout the city to maximize coverage when in reality ambulances are often stationed at only a few locations, e.g., ambulance bases. We tighten the lower bound by incorporating information on the number of ambulances stationed at base. Specifically, we calculate the ambulance placements that maximize coverage for every possible number of available ambulances and available ambulances stationed at bases. Since some of the ambulances are constrained to be at bases the maximum coverage will be less than those in the unconstrained

case. Because we cannot estimate the fraction of time that we have a given number of available ambulances and available ambulances stationed at bases for an optimal policy, we estimate these quantities for a reference policy. As a result we obtain a heuristic lower bound that is tighter than the previous bound.

Both the lower bound and the heuristic bound can be quite loose because the lower bound calculation supposes that all ambulances that are not assigned to a base are optimally located throughout the city to maximize coverage, which is unlikely to be the case in practice. Consequently, we develop a stylized model of ambulance redeployment where emergency calls are always responded to from ambulance bases. This model is justified by empirical computations indicating that the fraction of calls responded to from bases is quite large for the realistic model. For the stylized model we show empirically that the lower bound is quite tight.

The lower bound and heuristic bound may be used to get a lower bound on the number of ambulances necessary to perform at a specified performance level, but since these bounds are frequently quite loose, they are not likely to be useful when trying to evaluate how much additional improvement might be possible with ambulance redeployment policies. However, the performance of ambulance redeployment policies on the stylized model are often very similar to the performance on the more realistic model. Since the lower bound is quite tight for the stylized model, it seems reasonable that this bound can be used as an indication on what performance gains may be possible through improved ambulance redeployment polices.

The rest of the paper is organized as follows. In Section 5.2 we develop a model of ambulance redeployment. In Section 5.3 we construct a stochastic

lower bound on the response time distribution for any redeployment policy, and in Section 5.4 we formulate the lower bound and illustrate its performance on a realistic case study. In Section 5.5 we heuristically improve the lower bound by explicitly considering the proportion of ambulances idle at base conditional on the number of ambulances available. Section 5.6 introduces the stylized model of ambulance redeployment and shows that the lower bound calculation is fairly close to what can be achieved through known policies. Simulation results from this stylized model are also compared with those of the model in Section 5.2.

5.2 Problem Formulation

Consider an EMS region serviced by a fleet of N ambulances. When an emergency call arrives the closest available ambulance is dispatched to the call. Ambulances that are traveling but not currently assisting a patient (e.g., an ambulance has finished at the hospital and is returning to a base) are also considered available and included in the calculation of the closest available ambulance. If there are no ambulances available the call is placed on a waiting list and served in the order they were placed on the list.

Upon arriving at the emergency scene, the paramedics give preliminary care to the patient on scene. If the patient is unable to be successfully treated at the scene the paramedics transport the patient to a nearby hospital and transfer the patient to the hospital staff. The total service time for a patient is calculated as the sum of the response time, on-scene time, transport time, and transfer time. Example distributions for the on-scene time, transport time, transfer time, and

destination hospital (if any) are described in Chapter 2. These times may depend upon the location of the emergency call, but we require that, except for the response time, they are independent of the location from which the ambulance responded. For example, the choice of the destination hospital may be chosen to be near the emergency scene but not the location the ambulance responded from.

When an ambulance finishes assisting the patient, either at the scene or after transferring the patient to a hospital, it becomes available to serve other calls. If there are calls on the waiting list the ambulance is immediately dispatched to serve the call that has been waiting the longest. If there are no calls on the waiting list, an ambulance redeployment policy dictates which base the ambulance should travel to. Upon reaching the destination base, the ambulance will remain idle at that location until it is dispatched to an emergency call.

In addition to assigning destination bases when ambulances become available, an ambulance redeployment policy may also initiate "additional" redeployments. Specifically, a redeployment policy may assign an available ambulance that is idle at base or returning to a base to a new destination. The ambulance will then travel to this new destination and wait for emergency calls there. Although ambulance crews are often frustrated by frequent redeployments, for purposes of constructing a lower bound we allow the ambulance redeployment policy to make redeployment decisions to all ambulances as frequently as desired. As such our lower bound results apply to any ambulance redeployment policy.

We assume travel times are deterministic and non time varying. These assumption are reasonable over short time intervals, say on the order of a few

hours long, but are unlikely to be reasonable for extended periods of time or during exceptional situations such as rush hour traffic. As such our lower bound results apply over time intervals with similar travel times and must be recomputed using different travel time models to incorporate time-varying travel times.

We calculate travel times according to a network-based travel model. Specifically, the path between any two points on the travel network is calculated to have the minimal travel time. We allow call arrivals to occur off the network and calculate the off-network travel time using the Manhattan distance from the closest node on the network and a specified "off-network" driving speed. Every path between two points must go through at least one network node and may only have off-network travel at the beginning and end of the path.

We model ambulance redeployment following the approach in Section 3.4.1 without the requirement of a finite waiting list. Specifically we model the ambulance redeployment as an $M_t/G/N$ queue where the N ambulances are represented by servers and emergency calls are represented by arrivals to the queue. We allow the rate of the arrival process to change over time but require that the distribution governing the location of call arrivals be fixed. The service time distribution depends upon the state of the queue at the time an ambulance is dispatched to a call and the location of the call arrival (see Section 3.4.1 for a detailed description of the state space).

5.3 Stochastic Lower Bound on the Response Time

The motivation for the lower bound stems from the idea that if we could instantaneously relocate all the available ambulances to new locations we could ensure that the ambulances are always positioned to minimize the probability that the next arriving call will be lost (as in Zhang (2010)). The performance of such a policy, however, is not sufficient as a lower bound for any ambulance redeployment policy because optimizing the probability of reaching the next call within the time threshold may lead to ambulance placements having very large response times for certain call locations. Consequently, optimizing the probability of reaching the next call may increase service times which results in fewer ambulances available and degrades performance.

To illustrate this concept consider a situation illustrated in Figure 5.1 where there are two ambulance bases (squares), two demand points (circles) with the first one having a slightly larger arrival rate than the second one, and one ambulance serving the region. Assume that the travel time between the first base and the first demand point is equal to the time threshold for lost calls, Δ, and that the travel time from the second base to the first demand point is slightly larger than the time threshold, say $\Delta + \varepsilon$. Assume that the travel time between the second base and the second demand point is essentially zero and that the travel time between the first base and the second demand point is slightly larger than twice the time threshold, $2\Delta + \varepsilon$.

Given this situation the ambulance will always be positioned in the first base since this allocation maximizes coverage. However the average response time in this allocation is approximately 1.5Δ. If the ambulance was positioned in the

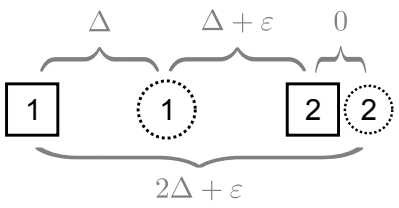

Figure 5.1: Two base maximum coverage example.

second base the average response time would be approximately $.5\Delta$. Thus by allocating the ambulance to maximize coverage we have significantly increased the utilization of the ambulance and have increased the amount of time that there are no ambulances available to respond to emergency calls. If the ambulance were placed in the second base the utilization of the ambulance would be much less and the amount of coverage would only be slightly less than optimal. Consequently the optimal allocation in this situation is likely to be at the second base.

To overcome this limitation we construct an $M_t/G/N$ queue with a service time distribution that is a stochastic lower bound on the service times experienced for any call, any number of ambulances available, and any location of those ambulances at the time the call was received. We obtain this stochastic lower bound by constructing a stochastic lower bound on the portion of the service time that depends upon the ambulance locations, i.e., the response time, and then convolving this lower bound with the remaining components of the service time.

Let the random variable R_n denote an "optimistic" response time that is stochastically smaller than the response time of any redeployment policy when there are n ambulances available. We construct the cumulative distribution function (cdf) of R_n, $F_n(\cdot)$, piecewise such that for any time t, the probability

that a call is responded to within t minutes for any redeployment policy is no larger than $F_n(t)$.

To construct F_n we use an integer program (IP) formulation of ambulance coverage to select ambulance locations that will maximize the region covered, i.e., maximize the probability of reaching the next call within a given time threshold t. To formulate this IP we first divide the EMS service region into a set of disjoint regions $\mathcal{R}$ (e.g., a grid of equally sized cells), and let d_r denote the probability an arriving call will arrive in region $r \in \mathcal{R}$.

Let K denote the total number of nodes in the travel network. Let $c(k, r, t) = \mathbb{1}_{\{\delta(k,r) \leq t\}}$ be a binary parameter that equals one if the travel time between node k and region r, $\delta(k, r)$, is within t minutes (i.e., node k "covers" region r) and 0 if not. To construct a lower bound $\delta(k, r)$ must denote the distance from the node k to the *closest* point in region r; however, for computational ease we use sufficiently small regions and interpret $\delta(k, r)$ to be the distance from node k to the centroid of region r.

Assume available ambulances are located at nodes of the travel network. The maximal coverage IP for $n \geq 1$ ambulances and a time threshold $t \geq 0$ is

defined as

$$
\begin{aligned}
L_n(t) = \max & \sum_{r \in \mathcal{R}} d_r w_r \\
\text{s.t.} & \sum_{k=1}^{K} x_k \leq n \\
& w_r \leq \sum_{k=1}^{K} c(k,r,t) x_k \qquad \forall\, r \in \mathcal{R} \\
& x_k \in \{0,1\} \qquad \text{for } k = 1 \ldots K \\
& w_r \in \{0,1\} \qquad \forall\, r \in \mathcal{R}
\end{aligned}
$$

where the decision variable x_k indicates if an ambulances is stationed at node k. The first constraint ensures that no more than n ambulances are used to calculate the coverage probability, and the second constraint allows w_r to be 1 only if region r is covered by an ambulance located at node k with $\delta(k,r) \leq t$. Since $d_r \geq 0$ for all $r \in \mathcal{R}$ we know that there is an optimal solution where $w_r = 1$ if and only if region r is covered by an ambulance. Thus $L_n(t)$ denotes the maximal probability that a call will be responded to within t minutes when there are n ambulances available.

Note that $L_n(t)$, for $n = 1, \ldots N$, satisfies the required condition on the cdf of R_n at time t to ensure that R_n is stochastically smaller than the response time of any ambulance redeployment policy facing any state (in terms of ambulance location and status). Setting $F_n(t)$ to $L_n(t)$ for all $t \geq 0$ would yield a suitable cdf for R_n, but is computationally intractable since there are an uncountable number of response times t that must be considered. Instead assume we calculate $L_n(t)$ at a finite number of points $t \in \{t_1, t_2, \ldots, t_D\}$ and construct a piecewise bound

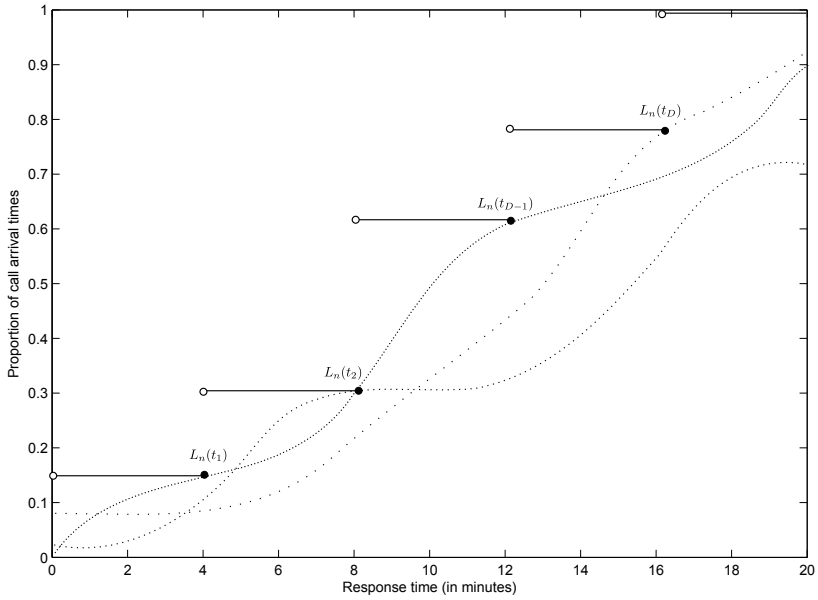

Figure 5.2: Stochastic Bound on Response Time for n ambulances.

on the cdf of R_n as follows:

$$F_n(t) = \begin{cases} L_n(t_1) & \text{if } t \leq t_1 \\ \vdots & \\ L_n(t_D) & \text{if } t \leq t_D \\ 1 & \text{if } t > t_D \end{cases}.$$

Figure 5.2 illustrates this bound. The solid dots, $L_n(t_i)$, correspond to the optimal objective value of the maximal coverage IP given the time threshold t_i and n available ambulances. The piecewise constant lines show the value of $F_n(t)$ for all t, and the dotted lines are representative response time distributions that may occur through various placement of n ambulances.

Since $F_n(t) \geq L_n(t)$ for all t, $R_n \sim F_n$ is stochastically smaller than the response time of any ambulance redeployment policy when there are n ambu-

lances available. Furthermore, if additional computational power is available, this bound can be refined by increasing the number of points at which the IP is solved.

If there are no ambulances available when the call arrives it is placed upon a waiting list and served when an ambulance becomes free and all calls preceding it on the waiting list have had an ambulance dispatched to them. Consequently, the response time distribution for the situation when there are no ambulances available is stochastically larger than the situation when there is one ambulance available. To complete the lower bound response time distribution we set $L_0(t) = L_1(t)$ for all $t \geq 0$ and hence $R_0 = R_1$ in distribution.

Consider an optimal redeployment policy, and let Q denote the ambulance redeployment queueing system under this policy. Let $\tilde{Q}$ denote the ambulance redeployment system when response times are given by R_n. Let p_n and $\tilde{p}_n$ denote the expected fraction of calls that arrived when there were n ambulances available in a finite time horizon of Q and $\tilde{Q}$ respectively. Since the response time of $\tilde{Q}$ is stochastically smaller than the response time of Q we can show that the number of available ambulances in $\tilde{Q}$ is stochastically larger than the number of available ambulances in Q.

Lemma 5.1. *Let the random variables X and $\tilde{X}$ denote the number of available ambulances at the time of a call arrival in a finite simulated time horizon of Q and $\tilde{Q}$ respectively, i.e., X and $\tilde{X}$ take value n with probability p_n and $\tilde{p}_n$ respectively. Then $X \preceq \tilde{X}$.*

Proof. The response time is the only element of the service time that depends on the ambulance's location at the time it was dispatched to a call. Hence the

response time is the only element of the service time that depends upon the redeployment policy used. Since the response time in $\tilde{Q}$ is stochastically smaller than the response time in Q, the service time distribution for $\tilde{Q}$ is stochastically smaller than that of Q. As a consequence, the number of calls in $\tilde{Q}$ is stochastically smaller than the number of calls in Q for all time by Theorem 6.2.3 in Stoyan (1983). Hence the number of available ambulances in $\tilde{Q}$ is stochastically larger than that in Q at all times. Under Poisson arrivals, the probability n ambulances are available when a call arrives in a finite simulated time horizon is equal to the time-average of the number of ambulances available in the queue. Hence the desired result holds. □

5.4 Lower Bound

Consider a finite simulated time horizon with response time threshold Δ. Let r denote the expected fraction of lost calls and r_n denote the expected fraction of lost calls that arrived when there were n available ambulances in this simulation. Let $v_n = 1 - L_n(\Delta)$ denote the minimal expected fraction of lost calls when there are n ambulances available achievable by any redeployment policy.

Theorem 5.2. *The expected fraction of lost calls in a finite simulated time horizon can be bounded below by*

$$\sum_{n=0}^{N} v_n \tilde{p}_n. \tag{5.1}$$

Proof. Let X and $\tilde{X}$ be as defined in Lemma 5.1. Define the function v :

$\{0, 1, \ldots, N\} \to \mathbb{R}$ by $v(i) = v_i$ and note that v is a non-increasing function.

$$
\begin{aligned}
r &= \sum_{n=0}^{N} r_n p_n \\
&\geq \sum_{n=0}^{N} v_n p_n \qquad \text{since } r_n \geq v_n \text{ for all } n \\
&= \mathbb{E}\left[v(X)\right] \\
&\geq \mathbb{E}\left[v(\tilde{X})\right] \qquad \text{see, e.g., Proposition 9.1.2 in Ross (1996)} \\
&= \sum_{n=0}^{N} v_n \tilde{p}_n.
\end{aligned}
$$

□

The values $\{\tilde{p}\}_{n=0}^{N}$ can be estimated from a $M_t/G/N$ queueing simulation where the response time is given by R_n and the on-scene time, transport time, and transfer time distributions are unchanged. Calculation of $\{v_n\}_{n=0}^{N}$ requires solving at most N maximal coverage IPs. (If the threshold time Δ was a value used to compute the optimistic response time cdf then these IPs have already been solved and v_n can be computed from the optimal objective function value of these IPs.) Consequently, the bound in Theorem 5.2 can be calculated by solving multiple IPs to construct a stochastic bound on the response time and simulating a queue to estimate the fraction of calls that arrive for each number of available ambulances.

To illustrate the lower bound we consider the case study of Edmonton, Canada as in Section 2.5. Note that the input parameters for this case study are representative and are no indication of realistic conditions in Edmonton. Figure 5.3 illustrates the lower bound for each call arrival rate (solid curve) and the performance of a sample policy (dashed curve) for a response time threshold of 8 minutes. We fixed the distribution of the call arrival locations and varied the

call arrival rate along the x axis. The y axis represents the average proportion of "lost calls" from 300 two-week simulations. The sample policy is an ADP redeployment policy that was tuned for a call arrival rate of 6 calls/hr (see Chapter 3 for further details on the construction and tuning of the ADP policy). A call arrival rate between 4 and 6 calls/hr is reasonable for this case study, but we include call arrival rates as low as 1 call/hr and as high as 12 calls/hr to illustrate how the lower bound and sample policies perform in extreme situations.

The horizontal dashed line in Figure 5.3 represents the proportion of calls that are "unreachable" given the base locations and the call location distribution, i.e., the proportion of calls that are more than 8 minutes from any base. At a rate of 6 calls/hr empirical results from the ADP policy show that approximately 80% of calls are responded to from bases. Consequently, although the level of unreachable calls does not give a lower bound on the proportion of calls lost, it is unlikely the optimal policy will have performance significantly better than this level.

As seen in the graphic, the gap between the lower bound and the sample performance is quite wide (15% - 25%). The lower bound begins very near zero and, as expected, increases as the call arrival rate increases, but the lower bound does not surpass the level of unreachable calls until the very extreme call arrival rate of 12 calls/hr. This indicates that the lower bound is probably quite conservative.

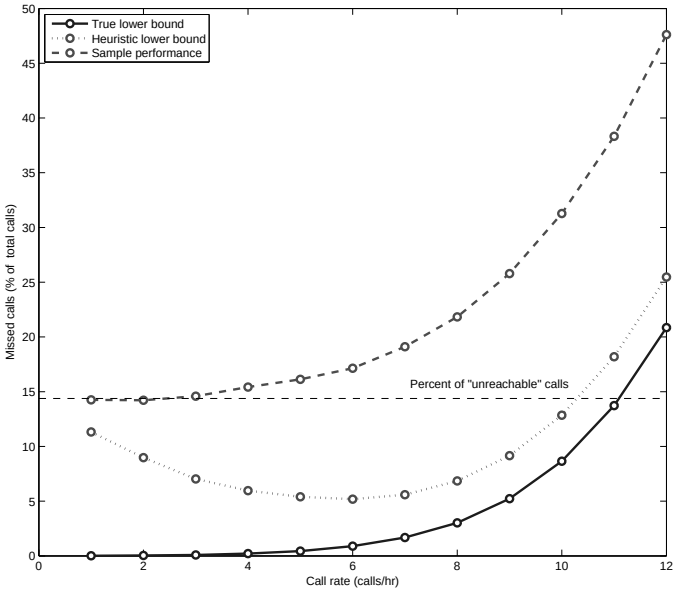

Figure 5.3: Sample Performance and Bounds on a Realistic Simulation Model.

5.5 Heuristic Lower Bound

One reason the lower bound may be overly conservative is that the maximum coverage IP formulation allows ambulances to be placed at any node in the travel network. This is appropriate since ambulances may answer calls while traveling between two locations, and hence an ambulance might respond to an emergency call from any location. In practice, however, ambulances are frequently stationed at only a relatively small number of network nodes corresponding to ambulances bases. According to our simulation results for call arrival rates similar to what would occur in practice, the majority of emergency calls are answered by ambulances idle at base. Restricting the maximum coverage IP to the set of bases would likely tighten the results, but it would no longer bound the actual performance.

One option is to use a reference policy to estimate the distribution of the number of ambulances idle at bases conditional on the number of ambulances available. A modification of the maximal coverage IP allows one to calculate the maximum probability of reaching the next call within a time threshold $t \geq 0$ when there are $n \geq 1$ ambulances available and $b \leq n$ ambulances are idle at base:

$$
\begin{aligned}
L_n(t,b) = \max & \sum_{r \in \mathcal{R}} d_r w_r \\
\text{s.t.} & \sum_{k=1}^{K} x_k \leq n \\
& \sum_{k=1}^{B} x_k = b \\
& w_r \leq \sum_{k=1}^{K} c(k,r,t) x_k \qquad \forall\, r \in \mathcal{R} \\
& x_k \in \{0,1\} \qquad \forall\, k = 1 \ldots K \\
& w_r \in \{0,1\} \qquad \forall\, r \in \mathcal{R}
\end{aligned}
$$

where we assume without loss of generality that the first B nodes in the network correspond to bases. The second constraint is the only modification from the original maximal coverage IP. This constraint requires that b ambulances must be located at bases. Similar to Theorem 5.2 we can construct a bound via,

$$
\sum_{n=0}^{N} \sum_{b=0}^{\min\{B,n\}} (1 - L_n(\Delta, b)) \tilde{p}_{n,b}, \tag{5.2}
$$

where $\tilde{p}_{n,b}$ denotes the expected fraction of calls that arrived when there were n ambulances available of which b were idle at base in a finite simulated time horizon of $\tilde{Q}$ under a reference redeployment policy.

To compute (5.2), we first solve $L_n(t, b)$ for $n = 1, \ldots, N$, $b = 0, \ldots, n$, and for a finite set of times to construct a response time distribution $R_{n,b}$ that is stochastically smaller than the response time distribution for any ambulance redeployment policy when there are n ambulances available of which b are stationed at bases (similar to the construction of the lower bound in Section 5.4). We set $L_0(\cdot, 0) = L_1(\cdot, 0)$ and hence $R_{0,0} = R_{1,0}$. Then $\tilde{p}_{n,b}$ can be estimated (by the fraction of call arrivals that arrived when there were n ambulances available of which b were idle at base) from a finite time horizon simulation in which ambulances are redeployed according to a reference policy and response times are distributed according to $R_{n,b}$.

The bound in (5.2) is only a heuristic lower bound on performance because an optimal redeployment policy may have a different conditional distribution for the number of ambulances idle at base than that of the reference policy. For Edmonton, Canada this bound is shown as the green line in Figure 5.3. The reference policy used to estimate $\tilde{p}_{n,b}$ is the ADP policy shown in the same figure.

For a call arrival rate of 1 call/hour, this heuristic bound is near 12% lost calls–about 3% below the proportion of unreachable calls. As the call arrival rate increases, this bound surprisingly decreases. The bound drops to about 5% at a call arrival rate of 6 calls/hour and then it increases at about the same rate as the lower bound. The reason for the dip in the heuristic bound is that as the call arrival rate increases the number of available ambulances in transit increases as well. Consequently, fewer ambulances are idle at bases and the bound becomes weaker. Although this heuristic bound is greater than the lower bound it is still well below the level of unreachable calls for nearly the entire span of call arrival rates. This indicates that, although it appears to improve on the lower bound,

the heuristic bound is likely too conservative as well.

5.6 Stylized Model Bounds

We developed the heuristic bound by assuming the optimal policy had properties similar to a known policy and then using these properties to tighten the bound. As an alternative approach we assume that all emergency calls are responded to from ambulances stationed at a base. To accommodate this assumption without significantly reducing the number of ambulances available to serve incoming calls we assume that ambulances arrive at their destination base as soon as they are assigned to the base.

As mentioned earlier, when simulating the ADP policy with a call arrival rate of 6 calls/hr about 80% of calls were served from ambulances idle at base. This implies that assigning ambulances to the "right" bases is especially important for good performance. The fact that calls can be responded to by an ambulance already on the road does provide some improvement in performance, but these improvements are likely to be smaller in magnitude than those achieved by proper positioning of ambulances at bases.

Since the majority of calls are served from bases, the assumption that ambulances always respond from bases is likely to still provide a fairly good estimate of policy performance. Additionally, assuming that ambulances arrive at their destination base immediately after being assigned to it is likely to improve the performance of a redeployment policy since ambulances will be positioned in the "right" locations more rapidly than would occur otherwise. Empirical results showing these improvements are illustrated later in this section (Fig-

ure 5.5).

The maximal coverage IP for the stylized model is identical to that of the original simulation model except that ambulances may only be assigned to the first B nodes of the network travel model (i.e., the bases):

$$\begin{aligned}
\Lambda_n(t) = \max & \sum_{r \in \mathcal{R}} d_r w_r \\
\text{s.t.} & \sum_{k=1}^{B} x_k \leq n \\
& w_r \leq \sum_{k=1}^{B} c(k,r,t) x_k \qquad \forall\, r \in \mathcal{R} \\
& x_k \in \{0,1\} \qquad \forall\, k = 1 \ldots B \\
& w_r \in \{0,1\} \qquad \forall\, r \in \mathcal{R},
\end{aligned}$$

and the lower bound for the stylized model is given by

$$\sum_{n=0}^{N} (1 - \Lambda_n(\Delta)) \tilde{\rho}_n,$$

where $\tilde{\rho}_n$ is estimated in an analogous manner as the other two bounds.

Figure 5.4 compares this lower bound with the sample performance of the ADP policy in Figure 5.3 on the stylized model. The bound in this model is much tighter than we observed with the original ambulance redeployment simulation model. In the region of practical importance, about 4 to 6 calls/hr, the bound is only about 1% lower than the ADP policy.

It is worthwhile to understand how the lower bound depends on the two aspects of our bounding technique. Recall that these two assumptions are as follows:

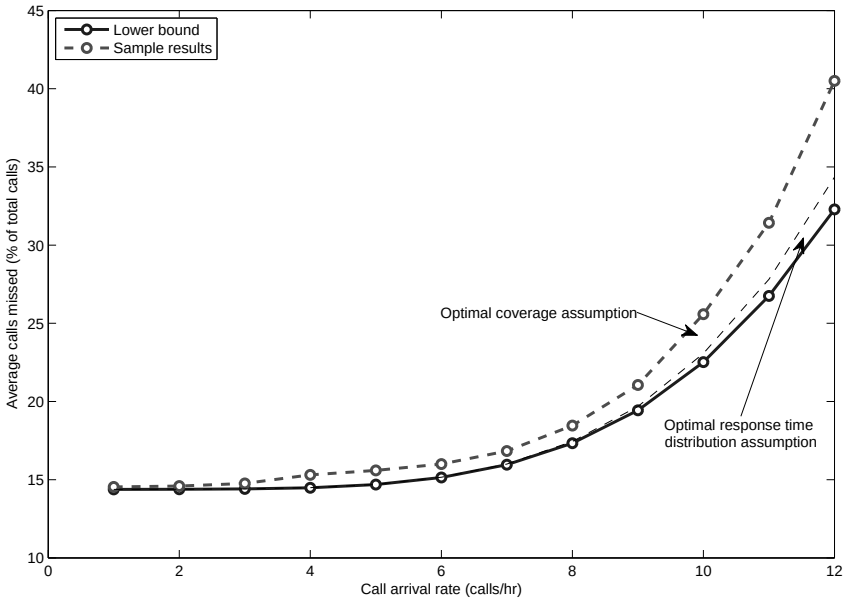

Figure 5.4: Sample Performance and Bounds on a Stylized Simulation Model.

1. The optimal coverage assumption in which we assume ambulances are always placed to maximize the probability of responding to the next call within the time threshold

2. The optimistic response time assumption in which we assume response times are stochastically smaller than the response times of any policy.

We can separate the effects of these two assumptions by calculating the bound using only the first assumption. Specifically, we estimate the expected fraction of call arrivals that occurred when there were n available ambulances in a finite time horizon simulation of the ADP policy on the stylized model, ρ_n, and use these estimates instead of $\tilde{\rho}_n$ when calculating the bound:

$$\sum_{n=0}^{N}(1-\Lambda_n(\Delta))\rho_n. \tag{5.3}$$

The thin dashed curve in Figure 5.4 illustrates the bound decomposition (5.3) over the call arrival range and shows the relative impact of each of these assumptions. The first assumption contributes the most to the discrepancy between the ADP policy and the bound; but, the second assumption becomes increasingly important as the call arrival rate increases. Additionally, as the call arrival rate increases the gap between the ADP policy and the lower bound increases. This is partially attributed to the growing impact of the second assumption. It is likely that some of this discrepancy can also be attributed to the fact that the ADP policy has not been tuned for such high call arrival rates. An appropriately tuned ADP policy would likely perform better in this range of call arrival rates.

Figure 5.5 compares the ADP policy performance and lower bound for the stylized model (dotted and solid curves respectively) versus the observed performance of the ADP policy on the realistic simulation model (dashed curve).

The results in the graph show that the performances of the ADP policy on the realistic model and on the stylized model are very similar across a variety of arrival rates. Based upon this relationship it seems reasonable to expect that the fraction of lost calls in the realistic model are close to, and slightly larger, than those of the stylized model. The performance of the ADP policy on the stylized model is, in turn, bounded by the lower bound computation for the stylized model. In this sense both the ADP policy performance and the lower bound calculation for the stylized model are practically relevant.

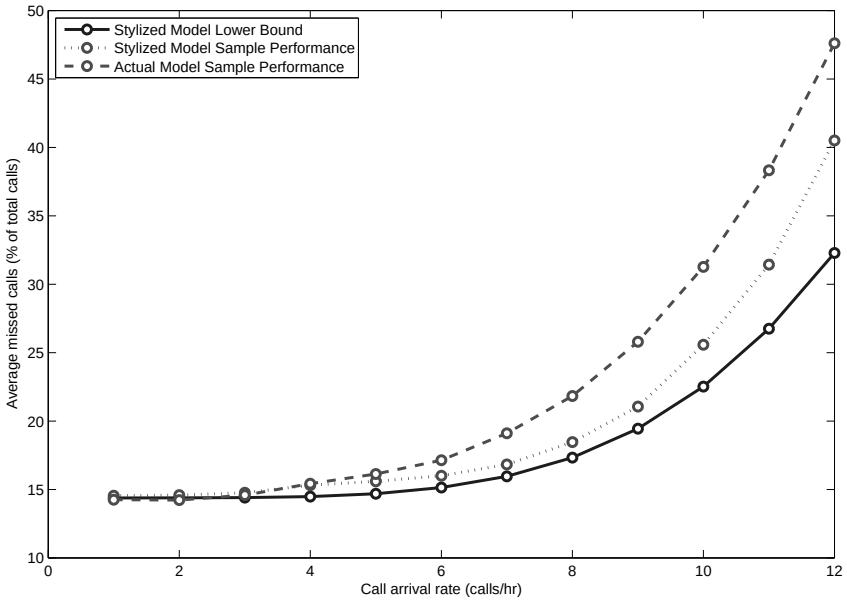

Figure 5.5: Comparison between the Performance in the Realistic and Stylized Models.

Due to the assumption that ambulances can instantaneously move to new locations in the stylized model, there are some policies that work extremely well in the stylized model but would perform very poorly on the realistic model (e.g., policies that never place more than one ambulance at the same base). Consequently, near-optimal performance in the stylized model does not necessarily indicate near-optimal performance in the realistic model. Nevertheless, given a policy designed for the realistic ambulance redeployment model, it is reasonable that the performance of this policy on the stylized model can give some indication on how close to optimal the policy may be for the realistic ambulance redeployment model.

APPENDIX A

APPENDIX FOR CHAPTER 3

A.1 Microsimulation Value Function Derivation

Let $Q = Q(S_k, x)$ denote a random variable indicating the number of non-decision states between state S_k and the next decision state given that we choose decision x is state S_k. For all $S_k \in \mathcal{S}$

$$
\begin{aligned}
J(S_k) &= \min_{x \in \mathcal{X}(S_k)} \mathbb{E}[c(S_k,x,U_{k+1}) + \alpha J(f(S_k,x,U_{k+1}))] \\
&= \min_{x \in \mathcal{X}(S_k)} P(Q=0)\mathbb{E}\left[c(S_k,x,U_{k+1}) + \alpha J(f(S_k,x,U_{k+1})) \middle| Q=0 \right] \\
&\qquad + P(Q=1)\mathbb{E}\left[c(S_k,x,U_{k+1}) + \alpha J(f(S_k,x,U_{k+1})) \middle| Q=1 \right] + \ldots \\
&= \min_{x \in \mathcal{X}(S_k)} P(Q=0)\mathbb{E}\left[c(S_k,x,U_{k+1})) + \alpha J(f(S_k,x,U_{k+1})) \middle| Q=0 \right] \\
&\qquad + P(Q=1)\mathbb{E}\left[c(S_k,x,U_{k+1}) + \alpha c(S_{k+1},\emptyset,U_{k+2}) + \alpha^2 J(f(S_{k+1},\emptyset,U_{k+2})) \middle| Q=1 \right] + \ldots \\
&= \min_{x \in \mathcal{X}(S_k)} \sum_{q=0}^{\infty} P(Q=q)\mathbb{E}\left[\sum_{j=0}^{q} \alpha^j c(S_{k+j},x_j,U_{k+j+1}) + \alpha^{q+1} J(f(S_{k+q},x_q,U_{k+q+1})) \middle| Q=q \right] \\
&= \min_{x \in \mathcal{X}(S_k)} \mathbb{E}\left[\sum_{j=0}^{Q} \alpha^j c(S_{k+j},x_j,U_{k+j+1}) + \alpha^{Q+1} J(f(S_{k+Q},x_Q,U_{k+Q+1})) \right]
\end{aligned}
$$

where $x_0 = x$ and $x_j = \emptyset$ for $j \geq 1$.

A.2 Truncated Microsimulation Value Function Derivation

Let $Q = Q(S_k, x)$ denote a random variable indicating the number of non-decision states between state S_k and the next decision state given that we choose decision x is state S_k. Given a deterministic time $\tau > 0$ let $Q_\tau = Q_\tau(S_k, x)$ denote a random variable indicating the number of non-decision states between S_k and

either the next decision state or the threshold time $t(S_k) + \tau$ (whichever comes first) given that we choose decision x is state S_k.

Let $S_k(\tau, x)$ denote the random state $S(t(S_k)+\tau)$ given that $S(t(S_k)) = S_k^+(x)$ and let $S_k^+(\tau, x)$ denote the deterministic state $S_k(\tau, x)$ given that $t(S_{k+1}) > t(S_k) + \tau$, i.e., $S(t(S_k) + \tau)$ given that the last event before time $t(S_k) + \tau$ was in state S_k and that action x was chosen in S_k. Thus on the event that there are no decision events within τ time after state $S_k^+(x)$ the system will be in the random state $S_k(\tau, x)$. Furthermore, if there are no events at all within τ time after state $S_k^+(x)$ the system will be in the deterministic state $S_k^+(\tau, x)$.

Let $\gamma_1 = t(S_{k+1}) - t(S_k)$ and $\gamma_{Q+1} = t(S_{Q+1}) - t(S_k)$ denote the time before the next event and the time before the next decision event respectively. For all $S_k \in \mathcal{S}$,

$$\begin{aligned}
J(S_k) &= \min_{x \in \mathcal{X}(S_k)} \mathbb{E}\left[c(S_k, x, U_{k+1}) + \alpha J(f(S_k, x, U_{k+1}))\right] \\
&= \min_{x \in \mathcal{X}(S_k)} P(\tau \le \gamma_1)\mathbb{E}\left[c(S_k, x, U_{k+1}) + \alpha J(f(S_k, x, U_{k+1}))\Big|\tau \le \gamma_1\right] \\
&\quad + P(\gamma_1 < \tau \le \gamma_{Q+1})\mathbb{E}\left[c(S_k, x, U_{k+1}) + \alpha J(f(S_k, x, U_{k+1}))\Big|\gamma_1 < \tau \le \gamma_{Q+1}\right] \\
&\quad + P(\gamma_{Q+1} < \tau)\mathbb{E}\left[c(S_k, x, U_{k+1}) + \alpha J(f(S_k, x, U_{k+1}))\Big|\gamma_{Q+1} < \tau\right].
\end{aligned}$$

We see that

$$\begin{aligned}
\mathbb{E}\Big[c(S_k, x, U_{k+1}) &+ \alpha J(f(S_k, x, U_{k+1}))\Big|\tau \le \gamma_1\Big] \\
&= \mathbb{E}\left[0 + c(S_k(\tau, x), U_{k+1}) + \alpha J(f(S_k(\tau, x), U_{k+1}))\Big|\tau \le \gamma_1\right] \\
&= \mathbb{E}\left[\tilde{J}(S_k(\tau, x))\Big|\tau \le \gamma_1\right] \qquad \text{by (3.13)} \\
&= \tilde{J}(S_k^+(\tau, x)).
\end{aligned}$$

The first term of the expectation in right-hand side of the first equation is the immediate cost incurred from state S_k to $S_k(\tau)$ (given decision x was chosen

in S_k), the second term is the immediate cost incurred from state $S_k(\tau)$ to S_{k+1} (by definition of $c(\cdot,\cdot)$), and the third term is the value function at state S_{k+1} (by definition of $f(\cdot,\cdot)$). The last equality holds because on the event $\tau \leq \gamma_1$, $S_k(\tau, x) = S_k^+(\tau, x)$ and $\tilde{J}(S_k^+(\tau, x))$ is a deterministic function of S_k, τ, and x.

Similarly,

$$\begin{aligned}
&\mathbb{E}\left[c(S_k,x,U_{k+1})+\alpha J(f(S_k,x,U_{k+1}))\middle|\gamma_1<\tau\leq\gamma_{Q+1}\right]\\
&\quad=\mathbb{E}\Bigg[\sum_{j=0}^{Q_\tau}\alpha^j c(S_{k+j},x_j,U_{k+j+1})+\alpha^{Q_\tau+1}c(S_{k+Q_\tau+1},x_{Q_\tau+1},U_{k+Q_\tau+2})\\
&\qquad\qquad+\alpha^{Q_\tau+2}J(f(S_{k+Q_\tau+1},x_{Q_\tau+1},U_{k+Q_\tau+2}))\Big|\gamma_1<\tau\leq\gamma_{Q+1}\Bigg] \qquad \text{via Appendix A.1}\\
&\quad=\mathbb{E}\left[\sum_{j=0}^{Q_\tau}\alpha^j c(S_{k+j},x_j,U_{k+j+1})+\alpha^{Q_\tau+1}\tilde{J}(S_k(\tau,x))\middle|\gamma_1<\tau\leq\gamma_{Q+1}\right].
\end{aligned}$$

Thus, for all $S_k \in \mathcal{S}$,

$$\begin{aligned}
J(S_k)=&\min_{x\in\mathcal{X}(S_k)} P(\tau\leq\gamma_1)\tilde{J}(S_k^+(\tau,x))\\
&+P(\gamma_1<\tau\leq\gamma_{Q+1})\mathbb{E}\left[\sum_{j=0}^{Q_\tau}\alpha^j c(S_{k+j},x_j,U_{k+j+1})+\alpha^{Q_\tau+1}\tilde{J}(S_k(\tau,x))\middle|\gamma_1<\tau\leq\gamma_{Q+1}\right]\\
&+P(\gamma_{Q+1}<\tau)\mathbb{E}\left[\sum_{j=0}^{Q}\alpha^j c(S_{k+j},x_j,U_{k+j+1})+\alpha^{Q+1}J(f(S_{k+Q},x_Q,U_{k+Q+1}))\middle|\gamma_{Q+1}<\tau\right]
\end{aligned}$$

where the last line follows from (3.15).

A.3 Proof of Theorem 3.4

We assume that the rate function of the arrival process is bounded, and hence, the expected number of arrivals on any finite interval is finite. We also assume

that the interarrival time distribution and the service time distributions are non-atomic. Let $I(s) = \{i : e_i \in E(s)\}$ denote the index set of active events $E(s)$. For any $s \in \mathcal{S}$ let $M_i(s)$ for $e_i \in E(s)$ denote the (possibly infinite) supremum of the support of the residual time distribution for event e_i. Define

$$\mathcal{B} = \{s \in \mathcal{S} : \exists\, e_i \in E(s) \text{ where } M_i(s) = 0\}$$

as the set of states that have an immediate service or arrival due to a clock equaling the maximum value in the support of its distribution.

Lemma A.1. *If* $P(S_k \notin \mathcal{B}) = 1$ *then*

$$P(t(S_{k+1}) = t(S_k)) = 0,$$

i.e., if S_k *is not in* $\mathcal{B}$ *w.p.1, then the probability of getting an immediate event is zero.*

Proof. Recall that $F_i^{-1}\big(S_k^+, \cdot\big)$ denotes the quantile function for the residual event time for event e_i in state S_k^+. We have

$$\begin{aligned}
P(t(S_{k+1}) = t(S_k)) &= P(t(S_{k+1}) = t(S_k) | S_k \notin \mathcal{B}) P(S_k \notin \mathcal{B}) \\
&\qquad + P(t(S_{k+1}) = t(S_k) | S_k \in \mathcal{B}) P(S_k \in \mathcal{B}) \\
&= P(t(S_{k+1}) = t(S_k) | S_k \notin \mathcal{B}) \qquad \text{by assumption} \\
&= P\left(\min_{i \in I(S_k)} F_i^{-1}\big(S_k, U_{k+1}(i)\big) = 0 \middle| S_k \notin \mathcal{B} \right).
\end{aligned}$$

Conditional upon $I(S_k) = I$, we have

$$\begin{aligned}
&P\Big(\min_{i \in I} F_i^{-1}\big(S_k, U_{k+1}(i)\big) = 0 \Big| S_k \notin \mathcal{B}, I(S_k) = I \Big) \\
&\qquad \leq \sum_{i \in I} P\left(F_i^{-1}\big(S_k, U_{k+1}(i)\big) = 0 \middle| S_k \notin \mathcal{B}, I(S_k) = I \right) \\
&\qquad = \sum_{i \in I} P\left(U_{k+1}(i) = 0 \right) \\
&\qquad = 0
\end{aligned}$$

where the second to last equality holds due to the fact that $S_k \notin \mathcal{B}$ which implies that for each $i \in I$ the residual time distribution is non-atomic and hence $F_i^{-1}\big(S_k, U_{k+1}(i)\big) = 0$ if and only if $U_{k+1}(i) = 0$.

Since there are a finite number of realizations of $I(\cdot)$ we have that

$$P\left(\min_{i \in I(S_k)} F_i^{-1}\big(S_k, U_{k+1}(i)\big) = 0 \Big| S_k \notin \mathcal{B}\right) = 0,$$

and the desired result holds. □

Lemma A.2. *If $P(S_k \notin \mathcal{B}) = 1$ then*

$$P(S_{k+1} \notin \mathcal{B}) = 1,$$

i.e., if S_k is not in $\mathcal{B}$ w.p.1 then S_{k+1} is not in $\mathcal{B}$ w.p.1.

Proof. We have

$$\begin{aligned}
P(S_{k+1} \in \mathcal{B}) &= P(S_{k+1} \in \mathcal{B} | S_k \notin \mathcal{B}) P(S_k \notin \mathcal{B}) + P(S_{k+1} \in \mathcal{B} | S_k \in \mathcal{B}) P(S_k \in \mathcal{B}) \\
&= P(S_{k+1} \in \mathcal{B} | S_k \notin \mathcal{B}) \qquad \text{by assumption} \\
&= P(\exists\, i \in I(S_{k+1}) \text{ such that } M_i(S_{k+1}) = 0 | S_k \notin \mathcal{B}) \\
&\leq P\left(\exists\, i \in I(S_k) \text{ such that } F_i^{-1}\big(S_k, U_{k+1}(i)\big) = M_i(S_k) | S_k \notin \mathcal{B}\right).
\end{aligned}$$

The third equality holds by definition of $\mathcal{B}$ and the first inequality holds because a necessary condition to satisfy $M_i(S_{k+1}) = 0$ is that $F_i^{-1}\big(S_k, U_{k+1}(i)\big)$ be equal to $M_i(S_k)$ (a sufficient condition would also require that $F_i^{-1}\big(S_k, U_{k+1}(i)\big) = F_{i'}^{-1}\big(S_k, U_{k+1}(i')\big)$ where $e_{i'} = e(S_{k+1})$). Conditional upon $I(S_k) = I$, we have

$$\begin{aligned}
&P\left(\exists\, i \in I \text{ such that } F_i^{-1}\big(S_k, U_{k+1}(i)\big) = M_i(S_k) | S_k \notin \mathcal{B}, I(S_k) = I\right) \\
&\qquad \leq \sum_{i \in I} P\left(F_i^{-1}\big(S_k, U_{k+1}(i)\big) = M_i(S_k) | S_k \notin \mathcal{B}, I(S_k) = I\right) \\
&\qquad = \sum_{i \in I} P\left(U_{k+1}(i) = 1\right) \\
&\qquad = 0
\end{aligned}$$

where the second to last equality holds due to the fact that $S_k \notin \mathcal{B}$ which implies that for each $i \in I$ the residual time distribution is non-atomic and hence $F_i^{-1}\big(S_k, U_{k+1}(i)\big) = M_i(S_k)$ if and only if $U_{k+1}(i) = 1$.

Since there are a finite number of realizations of $I(\cdot)$ we have that

$$P\left(\exists\, i \in I(S_k) \text{ such that } F_i^{-1}\big(S_k, U_{k+1}(i)\big) = M_i(S_k) | S_k \notin \mathcal{B}\right) = 0,$$

and the desired result holds. □

Proof of Proposition 3.3. *If $P(S_0 \notin \mathcal{B}) = 1$ then*

$$P\left(\exists k : t(S_{k+1}) = t(S_k)\right) = 0,$$

i.e., if S_0 is not in $\mathcal{B}$ w.p.1 then the probability of any two events occurring at the same time is zero.

Proof. Given $P(S_0 \notin \mathcal{B}) = 1$ and Lemma A.2 we know that $P(S_k \notin \mathcal{B}) = 1$ for all k by induction. Thus by Lemma A.1 we know that $P(t(S_{k+1}) = t(S_k)) = 0$ for all k.

The number of events is bounded by twice the number of arrivals (one for arrival and one for service completion) plus a constant factor depending on the initial state S_0. Since the number of arrivals are countable the number of events are also countable, and we have that

$$\begin{aligned} P\left(\exists k : t(S_{k+1}) = t(S_k)\right) &\leq \sum_{k=0}^{\infty} P\left(t(S_{k+1}) = t(S_k)\right) \\ &= 0. \end{aligned}$$

□

Lemma A.3. *If* $|J_{r'}(s)| \leq H_J < \infty$ *and* $|c(s,x,\cdot)| \leq H_c < \infty$ *for all* $s \in \mathcal{S}$, $x \in \mathcal{X}(s)$ *then*

$$\lim_{\tau \downarrow 0} \mathbb{E}\left[\Big| \sum_{j=0}^{Q} \alpha^j c(S_{k+j}, x_j, U_{k+j+1}) + \alpha^{Q+1} J_{r'}(f(S_{k+Q}, x_Q, U_{k+Q+1})) \Big| \, \Big| \gamma_{Q+1} < \tau \right] < \infty$$

where $x_0 \in \mathcal{X}(S_k)$ *and* $x_j = \emptyset$ *for* $1 \leq j \leq Q$.

Proof. For any $\tau > 0$

$$\begin{aligned}
&\mathbb{E}\Big[\Big| \sum_{j=0}^{Q} \alpha^j c(S_{k+j}, x_j, U_{k+j+1}) + \alpha^{Q+1} J_{r'}(f(S_{k+Q}, x_Q, U_{k+Q+1})) \Big| \, \Big| \gamma_{Q+1} < \tau \Big] \\
&\qquad \leq \mathbb{E}\left[\Big| \sum_{j=0}^{Q} H_c \Big| \, \Big| \gamma_{Q+1} < \tau \right] + H_J \\
&\qquad \leq H_c \mathbb{E}\left[Q+1 \Big| \gamma_{Q+1} < \tau \right] + H_J \\
&\qquad < \infty.
\end{aligned}$$

The quantity Q is bounded by the total number of events in $(t(S_k), t(S_k)+\tau)$ which is bounded by twice the number of arrivals in $(t(S_k), t(S_k)+\tau)$ (one for arrival, one for service completion) plus a constant number of events depending upon the state S_k. By assumption on the arrival rate function, the expected number of arrivals in any finite time period is finite. As $\tau \downarrow 0$ the time period $(t(S_k), t(S_k)+\tau)$ decreases and hence the expected number of arrivals in $(t(S_k), t(S_k)+\tau)$ decreases as well, giving the desired result. □

Corollary A.4. *If* $|\tilde{J}_r(s)| \leq H_{\tilde{J}} < \infty$ *and* $|c(s,x,\cdot)| \leq H_c < \infty$ *for all* $s \in \mathcal{S}$, $x \in \mathcal{X}(s)$ *then*

$$\lim_{\tau \downarrow 0} \mathbb{E}\left[\Big| \sum_{j=0}^{Q_\tau} \alpha^j c(S_{k+j}, x_j, U_{k+j+1}) + \alpha^{Q_\tau+1} \tilde{J}_r(S_k(\tau, x_{Q_\tau})) \Big| \, \Big| \gamma_1 < \tau \leq \gamma_{Q+1} \right] < \infty$$

where $x_0 \in \mathcal{X}(S_k)$ *and* $x_j = \emptyset$ *for* $1 \leq j \leq Q_\tau$.

Proof of Theorem 3.4. *Assume that $P(S_0 \notin \mathcal{B}) = 1$, that $\tilde{J}_r$, for a fixed r, and c are bounded, and that $\tilde{J}_r(\cdot)$ is continuous in $T(\cdot)$ (for any fixed remaining state components). Then for any bounded ADP approximation architecture $J_{r'}$*

$$\lim_{\tau \downarrow 0} L_{r,r',\tau}(s) = \tilde{L}_r(s) \qquad \forall s \in \mathcal{S}.$$

Thus the function defining the post-decision state policy is the limit of the function defining the truncated microsimulation policy as the truncation time goes to zero (for any bounded ADP approximation architecture).

Proof.

$$\begin{aligned}
\lim_{\tau\downarrow 0} L_{r,r',\tau}(S_k) = \lim_{\tau\downarrow 0} \Bigg(& \min_{x\in\mathcal{X}(S_k)} P(\tau \le \gamma_1)\tilde{J}_r(S_k^+(\tau,x)) \\
& + P(\gamma_1 < \tau \le \gamma_{Q+1})\mathbb{E}\left[\sum_{j=0}^{Q_\tau} \alpha^j c(S_{k+j},x_j,U_{k+j+1}) + \alpha^{Q_\tau+1}\tilde{J}_r(S_k(\tau,x)) \,\middle|\, \gamma_1 < \tau \le \gamma_{Q+1} \right] \\
& + P(\gamma_{Q+1} < \tau)\mathbb{E}\left[\sum_{j=0}^{Q} \alpha^j c(S_{k+j},x_j,U_{k+j+1}) + \alpha^{Q+1} J_{r'}(f(S_{k+Q},x_Q,U_{k+Q+1})) \,\middle|\, \gamma_{Q+1} < \tau \right] \Bigg) \\
= \min_{x\in\mathcal{X}(S_k)} \lim_{\tau\downarrow 0} \Bigg(& P(\tau \le \gamma_1)\tilde{J}_r(S_k^+(\tau,x)) \\
& + P(\gamma_1 < \tau \le \gamma_{Q+1})\mathbb{E}\left[\sum_{j=0}^{Q_\tau} \alpha^j c(S_{k+j},x_j,U_{k+j+1}) + \alpha^{Q_\tau+1}\tilde{J}_r(S_k(\tau,x)) \,\middle|\, \gamma_1 < \tau \le \gamma_{Q+1} \right] \\
& + P(\gamma_{Q+1} < \tau)\mathbb{E}\left[\sum_{j=0}^{Q} \alpha^j c(S_{k+j},x_j,U_{k+j+1}) + \alpha^{Q+1} J_{r'}(f(S_{k+Q},x_Q,U_{k+Q+1})) \,\middle|\, \gamma_{Q+1} < \tau \right] \Bigg)
\end{aligned}$$

The interchange of the minimum and limit above is justified because we have a minimization over a finite number of convergent sequences and hence the sequence of the minimums converges uniformly. Interchange of the minimum and limit for uniformly convergent sequences is justified in, for example, (Shapiro, 2003, Proposition 5).

A consequence of Proposition 3.3 is that $\lim_{\tau\downarrow 0} P(\tau \le \gamma_1) = 1$ and $\lim_{\tau\downarrow 0} P(\gamma_1 < \tau \le \gamma_{Q+1}) = \lim_{\tau\downarrow 0} P(\gamma_{Q+1} < \tau) = 0$. From Lemma A.3 and

Corollary A.4 we know that the limits of the expectations above are bounded. Thus we have that

$$\begin{aligned}
\lim_{\tau \downarrow 0} L_{r,r',\tau}(S_k) &= \min_{x \in \mathcal{X}(S_k)} \lim_{\tau \downarrow 0} \tilde{J}_r(S_k^+(\tau, x)) \\
&= \min_{x \in \mathcal{X}(S_k)} \tilde{J}_r \left(\lim_{\tau \downarrow 0} S_k^+(\tau, x) \right) && \text{(A.1)} \\
&= \min_{x \in \mathcal{X}(S_k)} \tilde{J}_r \left(S_k^+(x) \right) && \text{(A.2)} \\
&= \tilde{L}_r(S_k)
\end{aligned}$$

where (A.1) follows because, by definition, the T component of $S_k^+(\tau, x)$ is the only component that varies as $\tau \downarrow 0$ (see Section 3.4.1) and $\tilde{J}_r(\cdot)$ is continuous in $T(\cdot)$ by assumption, and (A.2) follows from the definition of the continuous-time system dynamics in Section 3.4.1. □

APPENDIX B

APPENDIX FOR CHAPTER 4

B.1 Equivalence of ADP-CT and NCT policies

Proof of Lemma 4.1. Let n denote the number of ambulances free. Since we assume the system is in a state of compliance we know the NCT policy is well-defined.

TO $\Rightarrow$ NCT

Let p denote a TO policy (hence $J(p)$ is an NCT policy). The proof is based on showing that p and $J(p)$ make the same decisions at every decision point.

If $n < N$ and an ambulance becomes free, p will assign this ambulance to the base $r = g(n+1)$ and $J(p)$ will assign this ambulance to base r' satisfying $a_{n+1}(r') > a_n(r')$. By the definition of J we know that $a_{n+1}(r') > a_n(r')$ implies that $e_{n+1}(r') = 1$ which implies that $g(n+1) = r'$. Hence $r = r'$ and the two policies make the same assignment decision.

If $n > 0$ and an ambulance from base r_d (the redeployment destination base) becomes busy, p will redeploy an ambulance from base $r_o = g(n)$ to base r_d. If $r_o = r_d$ no redeployment is made. Similarly, $J(p)$ will redeploy an ambulance from base r'_o satisfying $a_n(r'_o) > a_{n-1}(r'_o)$ to r_d provided that $r'_o \neq r_d$, otherwise no redeployment is made. By the definition of J we know that $a_n(r'_o) > a_{n-1}(r'_o)$ implies that $e_n(r'_o) = 1$ which implies that $g(n) = r'_o$. Hence $r_o = r'_o$ and the two policies make the same redeployment decision.

NCT $\Rightarrow$ TO

Let p denote an NCT policy (hence $J^{-1}(p)$ is a TO policy).

If $n < N$ and an ambulance becomes free, p will assign this ambulance to the base r satisfying $a_{n+1}(r) > a_n(r)$ and $J^{-1}(p)$ will assign this ambulance to base $g(n+1) = r'$. By the definition of J^{-1} we know that $(n+1)$st largest element of T is in the row corresponding to the base r' satisfying $a_{n+1}(r') > a_n(r')$. Hence $r = r'$ and the two policies make the same assignment decision.

If $n > 0$ and an ambulance from base r_d becomes busy, p will redeploy an ambulance from the base r_o satisfying $a_n(r_o) > a_{n-1}(r_o)$ to base r_d. If $r_o = r_d$ no redeployment is made. Similarly, $J^{-1}(p)$ will redeploy an ambulance from base $g(n) = r'_o$ to r_d provided that $r'_o \neq r_d$, otherwise no redeployment is made. By the definition of J^{-1} we know that the nth largest element of T is in the row corresponding to the base r'_o satisfying $a_n(r'_o) > a_{n-1}(r'_o)$. Hence $r_o = r'_o$ and the two policies make the same redeployment decision. □

Proof of Lemma 4.2. Let n denote the number of ambulances free.

ADP-CT $\Rightarrow$ TO

Let p denote an ADP-CT policy (hence $K(p)$ is a TO policy).

If $n < N$ and an ambulance becomes free, p will assign this ambulance to the base $\arg\max_r \phi_r^+(n_r)$ and $K(p)$ will assign this ambulance to $r' = g(n+1)$. By the compliance assumption we know that the entries $\bigcup_b \{t_{b,l} \text{ for } l = 1, \ldots, n_b\}$ comprise the largest n entries of T. Since the rows of T are non-increasing we also know $\bigcup_b \{t_{b,n_b+l} \text{ for } l = 2, \ldots, N - n_b\}$ are entries that are no larger than the $(n+1)$th largest entry of T. Hence $g(n+1)$ is the row r' satisfying $\max_{r'} t_{r',n_{r'}+1} = \max_{r'} \phi_{r'}^+(n_{r'})$. Thus $r = r'$ and the two policies make the same assignment decision.

If $n > 0$ and an ambulance from base b becomes busy, p will choose r_o and r_d to achieve the maximum in (4.6). The policy $K(p)$ will redeploy an ambulance from base $g(n) = r'_o$ to b unless $r'_o = b$ in which case no redeployment will be made. Note, $\phi^+_{r_d}(n_{r_d}) \leq \min_{r_o} \phi^+_{r_o}(n_{r_o} - 1)$ for any r_d except potentially $r_d = b$ because otherwise there exists some r_o such that $\sum_{k=1}^{n} \mathbb{1}_{\{g(k)=r_o\}} \neq n_{r_o}$. Thus the maximum in (4.6) can only be positive if $r_d = b$.

We further know that the smallest $\phi^+_{r_o}(n_{r_o} - 1)$ value is achieved when $r_o = g(n)$. Thus, assuming $g(n) \neq b$, p redeploys an ambulance from base $g(n) = r_o = r'_o$ to b and both policies make the same redeployment decision. If $g(n) = b$, by definition of $g(n)$ we know that, $\phi^+_{b'}(n_{b'}) \leq \phi^+_{r_o}(n_{r_o} - 1)$ for all b' thus choosing any other r_o would result in a non-positive value for $y(r_o, b)$. Hence in this situation $r_o = r_d = b$ maximizes $y(r_o, r_d)$ at zero, and no redeployment is made. This too agrees with policy $K(p)$, hence the two policies make the same redeployment decisions.

TO $\Rightarrow$ ADP-CT

By the construction of K^{-1} we know that $t_{i,j} = \phi^+_i(j - 1)$. Hence the same arguments apply, and the two policies make the same decisions. $\square$

BIBLIOGRAPHY

Adelman, D. 2004. A price-directed approach to stochastic inventory routing. *Operations Research* **52**(4) 499–514.

Adelman, D. 2007. Dynamic bid-prices in revenue management. *Operations Research* **55**(4) 647–661.

Adelman, Daniel, Adam J. Mersereau. 2008. Relaxations of weakly coupled stochastic dynamic programs. *Operations Research* **56**(3) 712–727.

Alanis, Ramon, Armann Ingolfsson, Bora Kolfal. 2010. A Markov chain model for an EMS system with repositioning. URL `http://apps.business.ualberta.ca/aingolfsson/documents/PDF/Repositioning.pdf`. Manuscript.

Andersson, T. 2005. Decision support tools for dynamic fleet management. Ph.D. thesis, Department of Science and Technology, Linkoepings Universitet, Norrkoeping, Sweden.

Andersson, T., P. Vaerband. 2007. Decision support tools for ambulance dispatch and relocation. *Journal of the Operational Research Society* **58** 195–201.

Berman, O. 1981a. Dynamic repositioning of indistinguishable service units on transportation networks. *Transportation Science* **15**(2) 115–136. doi:10.1287/trsc.15.2.115.

Berman, O. 1981b. Repositioning of distinguishable urban service units on networks. *Computers and Operations Research* **8** 105–118.

Berman, O. 1981c. Repositioning of two distinguishable service vehicles on networks. *IEEE Transactions on Systems, Man, and Cybernetics* **SMC-11**(3) 187–193.

Bertsekas, D. P. 1995. *Dynamic Programming and Optimal Control*, vol. I and II. Athena Scientific, Belmont, MA.

Bertsekas, D.P., S.E Shreve. 1978. *Stochastic Optimal Control: The Discrete Time Case.*. Academic Press, New York.

Bertsekas, D.P., J.N. Tsitsiklis. 1996. *Neuro-Dynamic Programming*. Athena Scientific, Belmont, Massachusetts.

Boyan, Justin A. 2002. Technical update: Least-squares temporal difference learning. *Machine Learning* **49**(2) 233–246. doi:10.1023/A:1017936530646. URL `http://www.springerlink.com/content/H40XPNJBHXXMN5HV`.

Bradtke, Steven J., Andrew G. Barto, Pack Kaelbling. 1996. Linear least-squares algorithms for temporal difference learning. *Machine Learning*. 22–33.

Brotcorne, L., G. Laporte, F. Semet. 2003. Ambulance location and relocation models. *European Journal of Operations Research* **147**(3) 451–463.

Crites, Robert H., Andrew G. Barto. 1996. Improving elevator performance using reinforcement learning. *Advances in Neural Information Processing Systems 8*. MIT Press, 1017–1023.

de Farias, D. P., B. Van Roy. 2003. The linear programming approach to approximate dynamic programming. *Operations Research* **51**(6) 850–865.

Deng, Geng, Michael C. Ferris. 2006. Adaptation of the UOBYQA algorithm for noisy functions. *WSC '06: Proceedings of the 38th conference on Winter simulation*. Winter Simulation Conference, 312–319.

Desai, Vijay V., Vivek F. Farias, Ciamac C. Moallemi. 2009. The smoothed approximate linear program.

Erkut, E., A. Ingolfsson, G. Erdoğan. 2008. Ambulance deployment for maximum survival. *Naval Research Logistics* **55** 42–58.

Farias, V. F., B. Van Roy. 2004. Tetris: Experiments with the LP approach to approximate DP. Technical report, Stanford University.

Farias, V. F., B. Van Roy. 2006. Tetris: A study of randomized constraint sampling. G. Calafiore, F. Dabbene, eds., *Probabilistic and Randomized Methods for Design Under Uncertainty*, chap. 6. Springer-Verlag, 189–202.

Farias, V. F., B. Van Roy. 2007. An approximate dynamic programming approach to network revenue management. Tech. rep., Stanford University, Department of Electrical Engineering.

Gendreau, M., G. Laporte, S. Semet. 2001. A dynamic model and parallel tabu search heuristic for real time ambulance relocation. *Parallel Computing* **27** 1641–1653.

Gendreau, M., G. Laporte, S. Semet. 2006. The maximal expected coverage relocation problem for emergency vehicles. *Journal of the Operational Research Society* **57** 22–28.

Glynn, Peter W. 1989. A GSMP formalism for discrete event systems. *Proceedings of the IEEE* **77**(1) 14–23.

Goldberg, J. B. 2007. Personal Communication.

Ingolfsson, A. 2006. The impact of ambulance system status management. Presentation at 2006 INFORMS Conference.

Ingolfsson, A., E. Erkut, S. Budge. 2003. Simulation of single start station for Edmonton EMS. *Journal of the Operational Research Society* **54**(7) 736–746.

Kolesar, P., W. E. Walker. 1974. An algorithm for the dynamic relocation of fire companies. *Operations Research* **22**(2) 249–274.

L'Ecuyer, Pierre, Richard Simard, E. Jack Chen, W. David Kelton. 2002. An object-oriented random-number package with many long streams and substreams. *Operations Research* **50**(6) 1073–1075. doi:http://dx.doi.org/10.1287/opre.50.6.1073.358.

Maxwell, Matthew S., Shane G. Henderson, Huseyin Topaloglu. 2009. Ambulance redeployment: An approximate dynamic programming approach. M. D. Rossetti, R. R. Hill, B. Johansson, A. Dunkin, R. G. Ingalls, eds., *Proceedings of the 2009 Winter Simulation Conference*. Institute of Electrical and Electronics Engineers, Inc., Piscataway, New Jersey, 1850–1860.

Maxwell, Matthew S., Shane G. Henderson, Huseyin Topaloglu. 2010a. Identifying effective policies in approximate dynamic programming: Beyond regression. B. Johansson, S. Jain, J. Montoya-Torres, J. Hugan, E. Yücesan, eds., *Winter Simulation Conference*. Institute of Electrical and Electronics Engineers, Inc., WSC, Piscataway, New Jersey, 1079–1087.

Maxwell, Matthew S., Shane G. Henderson, Huseyin Topaloglu. 2010b. Tuning approximate dynamic programming policies for ambulance redeployment via direct search. Submitted for publication.

Maxwell, Matthew S., Mateo Restrepo, Shane G. Henderson, Huseyin Topaloglu. 2010c. Approximate dynamic programming for ambulance redeployment. *INFORMS Journal on Computing* **22**(2) 266–281. doi:10.1287/ijoc.1090.0345. URL `http://joc.journal.informs.org/cgi/content/abstract/ijoc.1090.0345v1`.

Messerli, EJ. 1972. Proof of a convexity property of the Erlang B formula. *The Bell System Technical Journal* **51** 951–953.

Nair, R., E. Miller-Hooks. 2006. A case study of ambulance location and relocation. Presentation at 2006 INFORMS Conference.

Nair, R., E. Miller-Hooks. 2009. Evaluation of relocation strategies for emergency medical service vehicles. *Transportation Research Record* **2137** 63–73.

Nelder, J.A., R. Mead. 1965. A simplex method for function minimization. *The computer journal* **7**(4) 308–313. doi:10.1093/comjnl/7.4.308.

Powell, M.J.D. 2002. UOBYQA: unconstrained optimization by quadratic approximation. *Mathematical Programming* **92**(3) 555–582. doi:10.1007/s101070100290. URL `http://www.springerlink.com/content/213GAGK76H38PW81`.

Powell, Warren B. 2007. *Approximate Dynamic Programming: Solving the Curses of Dimensionality*. John Wiley & Sons, Hoboken, NJ.

Powell, Warren B., Benjamin Van Roy. 2004. Approximate dynamic programming for high dimensional resource allocation problems. W. B. Powell J. Si, A. G. Barto, D. W. II, eds., *Handbook of Learning and Approximate Dynamic Programming*. IEEE Press, 261–284.

Puterman, Martin L. 2005. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, Hoboken, NJ.

Restrepo, M., S. G. Henderson, H. Topaloglu. 2009. Erlang loss models for the static deployment of ambulances. *Health Care Management Science* **12** 67–79.

Restrepo, Mateo. 2008. Computational methods for static allocation and real-time redeployment of ambulances. Ph.D. thesis, Cornell University, Ithaca NY, USA.

Richards, D. P. 2007. Optimised ambulance redeployment strategies. Master's thesis, Department of Engineering Science, University of Auckland, Auckland, New Zealand.

Ross, Sheldon M. 1996. *Stochastic Processes*. 2nd ed. John Wiley & Sons.

Schweitzer, P., A. Seidmann. 1985. Generalized polynomial approximations in Markovian decision processes. *Journal of Mathematical Analysis and Applications* **110** 568–582.

Shapiro, Alexander. 2003. Monte Carlo sampling methods. A. Ruszczynski, A. Shapiro, eds., *Stochastic Programming*, *Handbooks in Operations Research and Management Science*, vol. 10. Elsevier, 353 – 425. doi:DOI:10.1016/S0927-0507(03)10006-0. URL `http://www.sciencedirect.com/science/article/B7P6G-4FM0G3F-8/2/13a777b01ce47e9d75ee1e7c0662fafa`.

Si, Jennie, Andrew G. Barto, Warren B. Powell, Donald Wunsch II, eds. 2004. *Handbook of Learning and Approximate Dynamic Programming*. Wiley-Interscience, Piscataway, NJ.

Singh, S., D. Bertsekas. 1996. Reinforcement learning for dynamic channel allocation in cellular telephone systems. M. C. Mozer, M. I. Jordan, T. Petsche, eds., *Proceedings of Advances in Neural Information Processing Systems*, vol. 9. Advances in Neural Information Processing Systems, MIT Press, Cambridge, MA, 974–980.

Stoyan, Dietrich. 1983. *Comparison Methods for Queues and Other Stochastic Models*. John Wiley & Sons.

Sutton, R. S. 1988. Learning to predict by the methods of temporal differences. *Machine Learning* **3** 9–44.

Swersey, A. J. 1994. The deployment of police, fire and emergency medical units. S. M. Pollock, M. Rothkopf, A. Barnett, eds., *Operations Research and the Public Sector*, *Handbooks in Operations Research and Management Science*, vol. 6. North-Holland, 151–190.

Szita, István, András Lörincz. 2006. Learning tetris using the noisy cross-entropy method. *Neural Computation.* **18**(12) 2936–2941. doi:http://dx.doi.org/10.1162/neco.2006.18.12.2936.

Tesauro, Gerald. 1994. TD-Gammon, a self-teaching backgammon program, achieves master-level play. *Neural Computation* **6**(2) 215–219. doi:10.1162/neco.1994.6.2.215. URL `http://www.mitpressjournals.org/doi/abs/10.1162/neco.1994.6.2.215`.

Topaloglu, H., W. B. Powell. 2006. Dynamic programming approximations for stochastic, time-staged integer multicommodity flow problems. *INFORMS Journal on Computing* **18**(1) 31–42.

Tsitsiklis, J., B. Van Roy. 1997. An analysis of temporal-difference learning with function approximation. *IEEE Transactions on Automatic Control* **42** 674–690.

Tsitsiklis, J. N. 1994. Asynchronous stochastic approximation and Q-learning. *Machine Learning* **16** 185–202.

Tsitsiklis, J.N, B. Van Roy. 2001. Regression methods for pricing complex American-style options. *IEEE Transactions on Neural Networks* **12**(4) 694–703.

Van Roy, Benjamin, Dimitri P. Bertsekas, Yuchun Lee, John N. Tsitsiklis. 1997. A neuro dynamic programming approach to retailer inventory management. *Proceedings of the IEEE Conference on Decision and Control*. IEEE Press, Los Alamitos, CA, 4052–4057.

Watkins, C. J. C. H., P. Dayan. 1992. Q-learning. *Machine Learning* **8** 279–292.

Yan, X., P. Diaconis, P. Rusmevichientong, B. Van Roy. 2005. Solitaire: Man versus machine. *Advances in Neural Information Processing Systems* **17** 1553–1560.

Zhang, Lei. 2010. Optimisation of small-scale ambulance move-up. Matthias Ehrgott, Andrew Mason, eds., *Proceedings of the 45th Annual Conference of the Operations Research Society of New Zealand*. 150–159.

Zhang, Lei, A. J. Mason, A. B. Philpott. 2008. Simulation and optimisation for ambulance logistics and relocation. Presentation at the INFORMS 2008 Conference.

Zhang, Lei, Andrew Mason, Andy Philpott. 2010. Optimization of a single ambulance move up. Tech. rep., University of Auckland Faculty of Engineering. URL `https://researchspace.auckland.ac.nz/handle/2292/6031`.