University of Alberta

Emergency Medical Services Performance Under Dynamic Ambulance Redeployment

by

Ramon Alanis

A thesis submitted to the Faculty of Graduate Studies and Research
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy
in
Operations and Information Systems

Faculty of Business

Fall 2012
Edmonton, Alberta

To my wife Cristina,
to my son Rodrigo and my daughters Daniela and Cristina,
who motivated me to keep going.

To my parents,
for teaching me the principles that make me who I am.

Abstract

We present three articles written to satisfy the requirements for the Ph.D. degree in Operations and Information Systems: The first is focused on the use of a bidimensional Markov model to compute the expected performance of an ambulance system using dynamic ambulance repositioning based on the use of a compliance table. The model is validated against a detailed discrete event simulation model, and we show that the ranking of the results obtained from multiple compliance tables is highly correlated with the ranking obtained from the discrete simulation model. The second paper deals with the problem of finding optimal or near-optimal compliance tables for an ambulance system. We propose a framework to classify optimization models and we use it to put the two models developed into context. The first model is an integer programming formulation that assumes the ambulances are always in compliance, while imposing constraints on the ambulance repositioning required. The second model takes the bidimensional Markov model and uses it in a heuristic search to find near-optimal compliance tables.

The final paper, although not directly related to ambulance operation, resulted from the implementation of a discrete event simulation of an ambulance system. In this paper we consider pre-computed routing information from any node to any other node in a road network. We compress it by taking advantage of the structural properties of the information and by transforming the problem into a traveling salesman problem which can be solved either to optimality via a solver or approximately via an insertion heuristic.

Acknowledgements

The completion of this dissertation and the studies and research that led up to it has been a long journey. It could not have been completed without the help and support of many people: my family, my supervisors Armann Ingolfsson and Bora Kolfal, the members of my dissertation committee, the department personnel, my friends, and many others.

In particular, I would like to thank Armann Ingolfsson for his excellent supervision. He expertly guided my work, and he was patient and understanding when my family required my time. He also provided a model of what an academic and a professional should be.

I am also grateful to Bora Kolfal, with whom I collaborated for a significant part of my research, for his contributions, suggestions, and assistance in bringing the research to completion.

Special thanks also go to Ray Patterson, for his comments and suggestions and particularly for his willingness to help whenever I asked. I am also grateful for the help provided by Fernanda Campello to deal with some of the final challenges.

Thank you also to Jeanette Gosine, Kathy Harvey, Keltie Tolmie, and Louise Hebert for their assistance on countless occasions.

Finally, I am deeply grateful to my wife Ana Cristina, for taking on her shoulders so much of the weight of raising a family, and I thank my son Rodrigo and my daughters Monica Daniela and Ana Cristina for motivating me with their smiles and their love to persevere until the end.

Contents

Appendices

List of Tables

List of Figures

CHAPTER 1

Introduction

This dissertation is composed of three articles written to fulfill the requirements of the Ph.D. degree in Operations and Information Systems at the University of Alberta School of Business. The central area of research is the study of dynamic ambulance repositioning policies used in an emergency medical services (EMS) system, in which ambulance repositioning decisions are based on the use of compliance tables.

An EMS system is defined as an organization or group of organizations that provides emergency medical attention and transportation to medical facilities in a geographical region such as a city. Typically, an EMS system includes a fleet of ambulances or emergency response vehicles; the ambulance stations where ambulances and crews wait until they are needed; a dispatch center that receives emergency calls, evaluates the situations, and dispatches the ambulances; and the hospitals with emergency departments that receive the patients.

EMS systems are under constant pressure to achieve better performance because of the nature of medical emergencies—lives may depend on rapid access to medical treatment—and because population growth leads to an ongoing struggle to meet increasing needs with the limited resources available.

Dynamic ambulance repositioning is a technique used to improve performance; it is typically implemented based on a dispatch system supported by the use of GPS technology. The

dispatch system constantly monitors the locations and statuses of the ambulances and checks this information against a compliance table that specifies, given the number of ambulances available, where they should be located. The goal is to minimize the expected response time to emergency calls.

The first article, presented in Chapter 2, provides an analytical model based on the use of a bidimensional Markov model to approximate the expected performance of an EMS system that uses dynamic repositioning based on compliance tables. Simulation models have often been used to estimate performance, but few attempts have been made to find analytical solutions. Our bidimensional Markov model has in one dimension the number of available ambulances, and in the other an indicator of whether or not the system is in compliance. The model is inspired by Larson's Hypercube model (Larson, 1974, 1975) and shares similar inputs and outputs, but our model takes into account the ambulance repositioning whereas that of Larson does not. We tested the model against a detailed discrete event simulation model under multiple scenarios and found it to be an efficient alternative to the simulation. One important characteristic is that when comparing multiple compliance tables the model accurately preserves the rankings obtained via the simulation, while being significantly faster. This in part motivated the second article.

The second article, presented in Chapter 3, considers the problem of finding an optimal or near-optimal compliance table. The goal is to maximize the percentage of emergency calls responded to within a given time threshold τ. The article defines a framework based on the bidimensional Markov model defined in Chapter 2, that classifies performance models based on the assumption that the ambulances are always in compliance or the impact of being out of compliance. The framework also classifies models according to the use of nested or non-nested compliance tables.

The first approach presented is an integer programming (IP) formulation that assumes the ambulances are always in compliance; it has constraints to limit the ambulance repositioning. This allows for the use of non-nested compliance tables. The model requires as input parameters the probability of each possible number of busy ambulances as well as the transition

rates from a given number of busy ambulances to the following and previous states. These parameters can be obtained as outputs from our original bidimensional Markov model. The article tests the sensitivity of the model to changes in the state probabilities, and proposes an iterative procedure between the Markov model and the IP formulation to find an optimal compliance table.

We improve on the IP model by using the Markov model presented in Chapter 2, which takes out-of-compliance performance into account. We combine it with a heuristic search in order to find near-optimal compliance tables that more closely reflect the actual performance expected of the EMS system. Two heuristics are presented, one without a nesting constraint and the other to search for nested compliance tables. Our heuristic solution combines both approaches by using the IP solution as the starting point for the heuristic search, accelerating the process by starting from a near-optimal solution.

The last article, presented in Chapter 4, is a byproduct of the implementation of a discrete event simulation model as a validation tool for the EMS-performance Markov model. In the development of the simulation, we wanted to use a city's actual road network, with in excess of $15,000$ nodes. This was a requirement since the accurate modeling of travel times is fundamental for modeling an EMS system. We therefore had to either solve a shortest path problem every time that an ambulance needed to move, or pre-compute and store all possible paths between all pairs of nodes. This article proposes a successful solution for the latter approach: we first represent all possible paths using a routing matrix of dimension $|V|^2$ (where V is the set of all vertices or nodes), and then develop a compression scheme based on reorganizing the matrix. It is shown that we can optimize the storage required by transforming the problem into a traveling salesman problem, which can be solved for moderately sized problems. We present an insertion heuristic that can achieve near-optimal storage requirements in near-linear time. Compression ratios of up to 99.75% were achieved for large networks.

CHAPTER 2

A Markov Chain Model for an EMS System with Repositioning [1]

2.1 Introduction

Emergency medical services (EMS) systems are designed and operated with the aim of minimizing response times to emergency calls. In order to reduce the response time most emergency systems adapt to changing conditions using "system status management"—a set of strategies that include dynamic repositioning, by which dispatchers "move" ambulances to provide better coverage. In this paper, we propose, validate, and illustrate the use of a two-dimensional Markov chain model of an EMS system that uses a form of repositioning based on compliance tables.

Repositioning strategies are made possible by computer-aided dispatch (CAD) systems and global positioning system (GPS) technology, which make it possible for dispatchers to keep track of the location and status of every ambulance in the system. Also known as "flexing," repositioning strategies stand in contrast to every ambulance returning to its "home station" at the conclusion of every call, which is how many EMS systems operated in the past, and

[1] R. Alanis, A. Ingolfsson and B. Kolfal. Production & Operations Management, accepted for publication. 2012.

some still do. Surveys of North American EMS operators in 2001 (Cady, 2002) and 2009 (Williams, 2009) showed the percentage of operators who used a static deployment strategy decreasing from 41% to 30%, those using a dynamic strategy increasing from 23% to 37%, and those using a combination of static and dynamic strategies changing from 36% to 33%.

Many operations research models of ambulance operations, for example the Hypercube Queueing Model (HQM, Larson, 1974), assume a static deployment policy, either implicitly or explicitly, where ambulances are assigned to a fixed home station regardless of the system state; while in systems that use repositioning, dispatchers may "move" one or more ambulances when there is a change in system state (a call arrival or a call completion). A well-designed repositioning policy can improve performance through better dynamic matching of ambulance supply and call demand. For example, a simulation study for the Edmonton EMS system (Erkut et al., 2005), which used a compliance table policy at the time, indicated that to achieve equal performance without repositioning would require the addition of eight new ambulances, around the clock. Nair and Miller-Hooks (2009) also compared repositioning strategies to static strategies for various scenarios using an analytical approach and found performance improvements ranging from one to six percent.

Repositioning policies are controversial among EMS workers, because such policies increase the time workers need to spend in their vehicles, which may lead to increased back problems (Morneau and Stothart, 1999). Bledsoe (2003) questions the effectiveness of system status management and summarizes the concerns of EMS staff, whose workloads are increased by repositioning. On the other hand, Stout (1989) argues in favor of system status management and details some misconceptions and common implementation problems which could potentially render it ineffective, or result in excessive stress for paramedics.

Response time for an EMS system is the time interval from the arrival of an emergency call until the ambulance reaches the scene of the incident. The performance goal for most EMS systems is to have the response times of at least a certain percentage of high priority calls below a specific time threshold. For example, having response times of at least 90% of urgent calls below 9 minutes is a common performance goal in North America (Fitch,

Available Ambulances	Stations							
1	42/RAH							
2	42	42/RAH						
3	42	42/RAH	32					
4	42	42/RAH	32	33				
5	42	42/RAH	28/32	33	34/27			
6	42	42/RAH	28/32	33	34/27	7/25/NEC		
7	42	42/RAH	28/32	33	34/27	7/25/NEC	9/19/GNH/MIS	
8	42	42/RAH	32/UAH	33	34/27	7/25/NEC	9/GNH	19/28/MIS

Table 2.1: Compliance table example.

2005). Our Markov chain model can be used to compute such performance measures as the response time distribution, the distribution of the number of busy ambulances, expected travel time, and average ambulance utilization.

The repositioning policy that we focus on uses a so-called "compliance table." Table 2.1 provides an example of a real compliance table that has been used in Edmonton, Alberta. Each row in a compliance table shows, for a given number of available ambulances, the desired ambulance locations. For example, based on Table 2.1, with one available ambulance, it should be at station 42 or at the Royal Alexandra Hospital (RAH); with two available ambulances, one should be at station 42 and the other at station 42 or at RAH, and so on. When the system state changes, the dispatchers decide which ambulances to move in order to reach compliance in the new system state. We view the choice of a compliance table as a tactical decision, while the choice of moves to reach compliance is a real-time operational decision. We focus on the tactical compliance table level but our model can accommodate different algorithms for the choice of moves to reach compliance. We also envision our model being used as part of a larger model to support such strategic decisions as when and where to build new ambulance stations or close existing ones.

In practice (Stout, 1989), dispatchers usually choose the moves. Sometimes, the dispatchers use information that is not captured in the CAD system. For example, with two available ambulances located at stations 42 and 32, according to the compliance table, the ambulance at station 32 should be moved to either station 42 or RAH. However, if a dispatcher knows

that an ambulance crew expects to finish its current call at RAH in the next five minutes, then he might decide that no moves are necessary. In some EMS systems that we are familiar with, dispatchers are evaluated, in part, based on the fraction of time that the system is "in compliance," with the system considered to be in compliance as soon as moves have been initiated that, when completed, will result in ambulance locations that are consistent with the compliance table.

Real-time move choice decisions could be supported by computer systems that suggest moves or the decision could even be completely automated. Commercial systems that provide such functionality have started to appear and we discuss related academic work later in this section.

Our Markov chain model has two state variables: The number $B(t)$ of busy ambulances and an indicator variable $C(t)$ for whether the system is in compliance or not. The model has three transition types: Call arrivals, call completions, and reaching compliance. We assume a constant exogenous arrival rate but completion rates are state-dependent and we determine them by analyzing the components of an EMS call: Evaluation and dispatch, chute, travel, on scene, transport, and hospital times. The rates at which the system reaches compliance are also state-dependent. In order to quantify the likelihood of different out-of-compliance configurations, we find it useful to let the call completion rates depend on whether the last change to the number of busy ambulances was the result of a call arrival or a call completion. This results in transition rates that depend on state probabilities and therefore, to solve the model, we iterate between computing steady state probabilities and computing transition rates. After obtaining the steady state probabilities, we use convolution to approximate the response time distribution. We also discuss a simplified model with a single state variable ($B(t)$), which does not need iterations as it assumes that repositioning occurs instantly.

We focus our review of related work on models for performance evaluation and optimization of fleets of emergency vehicles, particularly ambulances. Green and Kolesar (2004) provide a recent review of the literature on the use of management science to support the design and operation of emergency service systems. Kolesar and Walker (1974) were one of the

first to model repositioning, specifically of fire companies for New York City. This work helped to establish repositioning as an effective technique to improve performance, and the repositioning strategies proposed in Kolesar and Walker (1974) were used during the World Trade Center terrorist attacks of September 11, 2001 to re-balance the remaining units to maintain service for the rest of the city (Green and Kolesar, 2004).

Three distinctions are important in relating our work to past research: (1) Whether repositioning is incorporated or not, (2) whether repositioning is preplanned or dynamic, and (3) the time horizon for dynamic repositioning. We discuss each of these distinctions in turn.

Although HQM does not incorporate repositioning, it is an important point of departure for our model. HQM was initially introduced by Larson (1974) as a Markov chain model of a queueing system with *distinguishable* servers. In an EMS context, what makes a server (an ambulance) distinguishable from other servers is its home station, to which the server is assumed to return at the end of every call. In a system with repositioning, home stations are much less important and it becomes reasonable to consider the servers to be indistinguishable. Viewing the servers as identical, we were able to construct a model that is more tractable than the HQM, even though the operations of a system that uses repositioning are more complicated than the operations of a system where ambulances always return to their home stations. Specifically, for a system with n ambulances, the cardinality of the HQM Markov chain's state space of 2^n grows exponentially with n, while our model's state space cardinality of $2n + 1$ is linear in n. To give an idea of the magnitude of n, the Edmonton EMS system had up to $n = 36$ scheduled ambulances in 2008 and Ahn et al. (2010) report that Seoul, South Korea, had $n = 134$ scheduled ambulances at all times in 2006-7. To address the computational and storage requirements of the exact HQM, researchers have developed a series of approximate versions (Larson, 1975; Jarvis, 1985; Budge et al., 2009) of the HQM, but such approximations are not necessary for our model.

We mention preplanned repositioning only to emphasize that it is different from the dynamic repositioning that we consider. Preplanned repositioning is done in anticipation of predictable shifts in demand and travel speeds, in contrast to dynamic repositioning based

on the current system state. See Rajagopalan et al. (2008) for recent work on preplanned repositioning.

Dynamic repositioning policies can be modeled using Markov decision processes or stochastic dynamic programming. Berman (1981a,b) took this approach and demonstrated how one could find optimal solutions for small systems. More recently, Zhang et al. (2011) followed in the same vein, solving a single-ambulance problem optimally in order to generate insight for more complicated systems, and Maxwell et al. (2009, 2010) used approximate dynamic programming to find solutions for larger systems with more realistic assumptions. These approaches provide policies that prescribe optimal or near-optimal repositioning moves, taking into account the probability distribution of future consequences of the moves. In contrast, Gendreau et al. (2001) proposed a parallel tabu search heuristic for the real-time repositioning of ambulances, given the current locations of all available ambulances and demand rates throughout the region, which one can view as forecasts for the location of the next call to arrive. We view the compliance table policies that we analyze as being at the tactical level. A compliance table provides a high-level and easy-to-understand policy, which must be complemented with a method for real-time decisions about how to reach compliance. That method could, for example, be a heuristic similar to the one proposed by Gendreau et al. (2001). Even though compliance table policies could be suboptimal, they have the advantage of being much easier to use than optimal dynamic programming policies and already being accepted practice among many EMS operators. Compliance tables do have the drawback that, to our knowledge, no tractable analytical models have been available to predict their impact on system performance. Our aim in this paper is to provide such a model. Gendreau et al. (2006) present an integer program to generate compliance tables, but their model involves several approximating assumptions that we relax, including deterministic response times, an exogenously estimated (binomial) probability distribution for the number of busy ambulances, and no consideration of the time needed to complete the moves necessary to achieve compliance.

We make the following contributions:

1. We propose and analyze a tractable analytical model that has the same data requirements and can produce the same outputs as the HQM, but models repositioning policies, which HQM does not do.

2. We develop procedures to estimate the parameters of the analytical model.

3. We validate the model against a realistic simulation model and find that the Markov chain model provides a good approximation to several performance measures.

4. We demonstrate that the Markov chain model can be used to identify solutions that are near-optimal, as measured by a realistic simulation model.

5. Our numerical results show that different compliance tables may lead to large variations in performance, which demonstrates the importance of using a well-designed compliance table.

In Section 2.2, we describe the operations of an EMS system and discuss the components of the service and response time. Section 2.3 presents the two-dimensional Markov chain model, an iterative algorithm to obtain its steady state probabilities, and a simplified one-dimensional model. Section 2.4 discusses estimation of the input parameters for the Markov chain model. Section 2.5 describes how we approximate the response time distribution using convolution. In Section 2.6, we discuss how we validated the results of the Markov chain model by comparing them to simulation results. Section 2.7 presents a numerical study to show that our Markov chain model can be used by EMS system managers to choose the best or a near-best compliance table. Finally, in Section 2.8, we comment on the importance of our model and summarize its performance.

2.2 EMS System with Repositioning

The purpose of this section is to describe how a real EMS system operates. We support the discussion with 2008 empirical data from the Edmonton, Alberta, EMS system. EMS systems differ in their operational practices and our description will not apply in all respects

to all such systems, but we believe that our description captures the main aspects of EMS operations for many EMS operators in medium to large urban areas in North America.

We have developed a discrete event simulation model of an EMS system, for validation purposes. The simulation model follows the description in this section. In the following section, we present our Markov chain model and explain the assumptions that it is based on.

One important assumption that we make is stationarity. Specifically, we assume that calls arrive to the system according to a homogeneous Poisson process with rate λ and the number of ambulances (servers) on duty is fixed at n. In reality, the arrival rate and the number of servers varies by time of day and day of the week. For example, in Edmonton in 2008, the number of scheduled ambulances varied from 19 to 36. We envision our model being used separately for periods (say, hours) over which the arrival rate and number of servers remain at least approximately constant. This is in line with Edmonton EMS practice, which is to use multiple compliance tables. Although the number of ambulances is fixed in the model, we have done extensive sensitivity analysis that suggests that if a compliance table performs well for a particular number of ambulances, then that compliance table will continue to perform relatively well when the number of ambulances is changed.

We aggregate city neighborhoods into demand nodes and we denote the set of demand nodes by D, with cardinality $|D|$. The probability that a particular call arrives to demand node d is f_d, with $\sum_{d\in D} f_d = 1$. We use S and H to denote the sets of stations and hospitals, with cardinalities $|S|$ and $|H|$. For calls that require transport, the probability of transport to hospital h is g_h, with $\sum_{h\in H} g_h = 1$.

We organize the discussion around the response to a typical emergency call, that is, the sequence of steps followed by the EMS system from the moment when a call arrives until the moment when the emergency crew finishes their duties for the call, as illustrated in Figure 2.1. If one views an ambulance as a server and the EMS system as a queueing system, then it is important to distinguish between the ambulance *service time* and the system *response time*. Figure 2.1 illustrates the difference between the two and we discuss

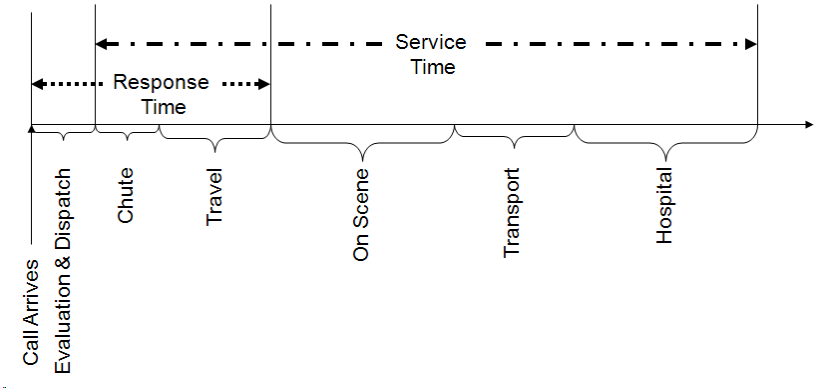


Figure 2.1: Service and Response Times

it throughout this section.

We define random variables to represent the various time intervals. These random variables will be conditional on the system state (b, c) when the call arrives, where b is the number of busy ambulances and c equals one if the system is in compliance and zero otherwise. (We use these state variables in the Markov chain model described later but we recognize that the real system, with only these two state variables, is unlikely to satisfy the Markov property.) The sequence of time instants and corresponding time intervals for a typical call is:

- Call arrival: A medical emergency occurs somewhere in the EMS system's service area and the patient or a bystander calls 911.
- Dispatch: A dispatcher answers the call, evaluates the situation, and dispatches the ambulance closest to the scene of the incident. We define $R_{b,c}^{\text{Eval and Disp}}$ for the evaluation and dispatch time.
- Chute: The crew of the dispatched ambulance receives the notification and boards the ambulance, if they are not already on board, and start driving towards the scene of the incident. We define $S_{b,c}^{\text{Chute}}$ for the chute time.
- Travel: The dispatched ambulance moves from its original location to the scene of the

incident. We define $S_{b,c}^{\text{Travel}}$ for the travel time.

- On-Scene: The crew locates the patient, provides emergency medical attention and, if required, transports the patient to the ambulance. In some cases the call ends at the scene of the incident, whereas in others, the patient requires transportation to a hospital (event *Patient Transportation*, PT). We define $S_{b,c}^{\text{On-Scene}}$ for the on-scene time.

- Transport: If transportation is required, then the ambulance transports the patient to a hospital. We define $S_{b,c}^{\text{Transport}}$ for the transport time.

- Hospital: At the hospital, the crew transfers the patient to the care of the emergency room personnel. This is the end of the call. We define $S_{b,c}^{\text{Hospital}}$ for the hospital time.

- Relocation: After the call has been completed, either at a hospital or at the scene of the incident, the ambulance travels to a station (possibly different from its original station) to wait for the next call. (At the same time, the dispatcher may ask other ambulances to move, in order to reach compliance.) During this time the ambulance is free, and it can be assigned to a new call.

2.2.1 Service Time vs. Response Time

Service time, $S_{b,c}$, as illustrated in Figure 2.1, is the total time an ambulance and crew remain "busy" with a call, i.e., not available to take other calls. It is composed of the chute, travel, and on-scene time, as well as transport and hospital times for calls that require transport (event PT):

$$S_{b,c} = \begin{cases} S_{b,c}^{\text{Chute}} + S_{b,c}^{\text{Travel}} + S_{b,c}^{\text{On-Scene}} + S_{b,c}^{\text{Transport}} + S_{b,c}^{\text{Hospital}} & \text{if } PT, \\ S_{b,c}^{\text{Chute}} + S_{b,c}^{\text{Travel}} + S_{b,c}^{\text{On-Scene}} & \text{Otherwise.} \end{cases} \tag{2.2.1}$$

The response time $R_{b,c}$ is the basis for most EMS performance measures. It is the time from the moment when a call is received to the moment when the ambulance arrives to the call

address, composed of the evaluation and dispatch, chute, and travel times:

$$R_{b,c} = R_{b,c}^{\text{Eval and Disp}} + S_{b,c}^{\text{Chute}} + S_{b,c}^{\text{Travel}}. \tag{2.2.2}$$

EMS performance targets typically focus on response time for calls that are classified as *urgent* (high priority) by the dispatcher. Evaluation and dispatch time is shorter, on average, for urgent calls, as we illustrate in this section. Travel speeds are also higher, on average, for urgent calls. In the remainder of this section we discuss the components of the service and response time in more detail.

2.2.2 Evaluation and Dispatch Time; Chute Time

Evaluation and dispatch is the process by which the emergency personnel answering the call question the caller to obtain basic information such as the nature of the emergency, and the location where the help is needed and based on this information, decide which unit should be called. Chute time is the time from the moment when an ambulance is dispatched, to the moment when the vehicle starts moving. The left panel of Figure 2.2 shows that the average evaluation and dispatch time was almost constant for urgent calls but increased with the number of busy ambulances for non-urgent calls, possibly because of increased workload on the control center and higher dispatch priority for urgent calls. The decrease in average evaluation and dispatch time when the number of busy ambulances is large might be caused by special protocols that take effect when the number of available ambulances drops below a threshold (a "yellow alert") or to zero (a "red alert"). In contrast, the right panel shows that the average chute time decreased with the number of busy ambulances, likely because the probability that an ambulance responds while traveling (as opposed to being at a station) increases with the number of busy ambulances. Average chute times were similar for urgent and non-urgent calls and therefore we do not show them separately. (Figure 2.2 and all subsequent figures in this section are based on 2008 Edmonton data. These figures show means and 95% confidence intervals (c.i.). The number of busy ambulances varied between 1 and 33 in this data set, but we limit our graphs to the range from 2 to 29, where

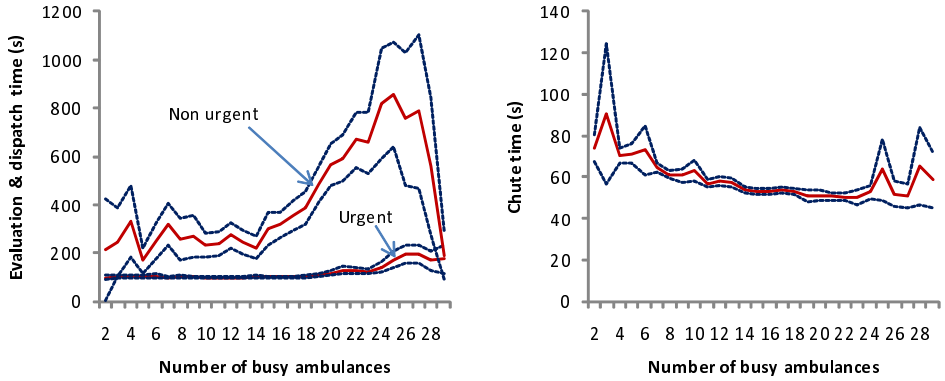


Figure 2.2: Means and 95% c.i. for evaluation and dispatch time and chute time .

we have data on at least 100 calls for each number of busy ambulances. The data included information about the number of busy ambulances (b) when each call arrived, but not about whether the system was in compliance (c).)

2.2.3 Travel Time and Transport Time

Travel time is the largest component of the response time, and it is the component that is most influenced by repositioning. Accordingly, we need to model travel times carefully. Travel time depends not only on the number $n - b$ of available ambulances but also on the locations of those ambulances. The basis for our Markov chain model is the assumption that the *locations* of the available ambulances are determined largely by the *number* of available ambulances, because of the use of a compliance table. The compliance table does not provide the exact locations of the ambulances at all times, but the dispatchers continuously relocate ambulances attempting to match the locations specified by the compliance table.

We define a set Γ of possible origin-destination pairs as:

$$\Gamma = \{(o, d) : o \in S \cup H \cup D \text{ and } d \in D\}, \tag{2.2.3}$$

where the origin o is an ambulance location and the destination d is a demand node. Our

travel time model for a given origin-destination pair is based on the work of Kolesar (1975) and Budge et al. (2010), with two exceptions, which we note in a moment. Let l be the distance from the origin o to the destination d. Parameters a (acceleration) and v_c (cruising speed) determine the median travel time $m(l)$ and parameters b_0, b_1, b_2 determine the travel time coefficient of variation $c(l)$ for the travel time $T(l)$ from o to d. The model is:

$$m(l) = \begin{cases} 2\sqrt{l/a} & l \le v_c^2/a \\ v_c/a + l/v_c & l > v_c^2/a \end{cases} \tag{2.2.4}$$

$$c(l) = \min\left\{ \frac{\sqrt{b_0(b_2+1) + b_1(b_2+1)m(l) + b_2 m(l)^2}}{m(l)}, c_{\max} \right\} \tag{2.2.5}$$

$$T(l) = m(l)\exp(c(l)\epsilon) \tag{2.2.6}$$

where ϵ follows a standard normal distribution.

Budge et al. (2010) found that a Student's t distribution for ϵ fit empirical data better than a standard normal distribution. The heavier tails of the t-distribution make the statistical estimation method used in Budge et al. (2010) more robust against errors in recorded travel times, but unfortunately, if ϵ follows a t distribution, then $T(l)$ has infinite expected value. We assume a standard normal distribution instead, to avoid having to solve a queueing model with a service time whose mean is infinite. The other difference between our travel time model and that of Budge et al. (2010) is that we set an upper bound $c_{\max}$ on the coefficient of variation, to avoid implementation difficulties when the distance l approaches zero.

By using (2.2.6) and the moment generating function for a normal distribution, we obtain the first two moments of T as follows:

$$E[T(l)] = m(l)\exp\left(c(l)^2/2\right), \tag{2.2.7}$$

$$E[(T(l))^2] = (m(l))^2 \exp\left(2c(l)^2\right). \tag{2.2.8}$$

Average travel speeds are faster for urgent calls. When we compute the response time

distribution, we use travel time distributions for urgent calls, since EMS response time targets focus on urgent calls.

We use the same equations ((2.2.4)-(2.2.6)) to model transport time from a given origin to a given destination, but with different parameters to reflect that transport to hospital is often less urgent than reaching the call address. The transport time is zero if the call does not require transport (PT^c), which we denote as follows:

$$S_{b,c}^{\text{Transport}} = \begin{cases} S_{b,c}^{\text{Transport}|PT} & \text{if } PT, \\ 0 & \text{Otherwise.} \end{cases} \tag{2.2.9}$$

2.2.4 On-Scene Time and Hospital Time

On-scene time may vary between calls that require transportation to a hospital and those that do not. We represent this as follows:

$$S_{b,c}^{\text{On-Scene}} = \begin{cases} S_{b,c}^{\text{On-Scene}|PT} & \text{if } PT, \\ S_{b,c}^{\text{On-Scene}|PT^C} & \text{Otherwise.} \end{cases} \tag{2.2.10}$$

The left panel of Figure 2.3 shows that the average on-scene time is approximately constant for calls that require transport but the average is higher and it decreases with the number of busy ambulances for calls that do not require transport. Possibly, ambulance staff provide some discretionary care (that could be performed by other health care providers later) to patients that do not require transport, especially if the EMS system is uncongested.

Hospital time is zero if the call does not require transport, and we represent this as follows:

$$S_{b,c}^{\text{Hospital}} = \begin{cases} S_{b,c}^{\text{Hospital}|PT} & \text{if } PT, \\ 0 & \text{Otherwise.} \end{cases} \tag{2.2.11}$$

In contrast to on-scene time, the right panel of Figure 2.3 shows that the average hospital

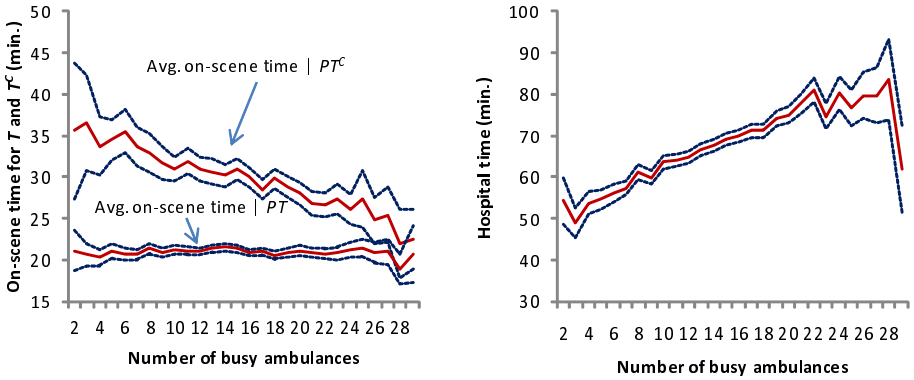


Figure 2.3: Means and 95% c. i. for on-scene time and hospital time.

time increases with the number of busy ambulances, which is what one would expect, because when the EMS system is busy, hospital emergency departments are likely to be busy as well. Long hospital times are a challenge in many cities in Canada (Sinnema, 2010; Segal et al., 2006) and elsewhere (Vandeventer et al., 2011). Some hospitals and health authorities have responded with protocols for expediting patients when emergency department congestion reaches certain thresholds (Alberta Health Services, 2010) and such protocols might explain the irregular pattern for average hospital time when the number of busy ambulances is large.

With this, we have completed our discussion of the components of the service and response times. The discrete event simulation model that we developed for validation purposes follows the assumptions set out in this section. In addition, the simulation model includes a move choice module, based on a greedy heuristic (described in Section 2.4), which attempts to minimize the number of moves needed to reach compliance and uses minimization of total estimated time to reach compliance as a tie breaker. This module could be replaced with another move choice algorithm, such as the tabu search heuristic proposed by Gendreau et al. (2001)—both in our simulation model and in the Markov chain model, which we describe next.

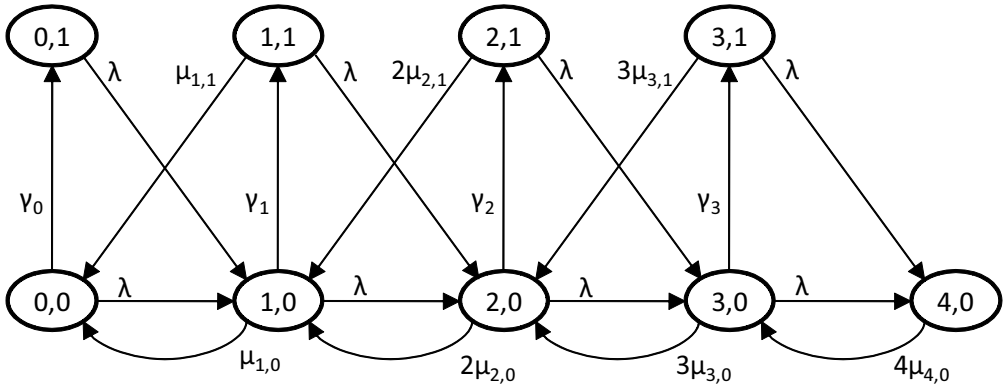


Figure 2.4: Transition rate diagram for $n = 4$

2.3 Markov Chain Model

The model we use to approximate an EMS system with n ambulances that are repositioned according to a compliance table policy is a finite, continuous-time Markov chain, with state variables $B(t)$, the number of busy ambulances, and $C(t)$, an indicator variable for whether the system is in compliance at time t. The state space for this process is $\Omega \equiv \{(b, c) : b = 0, \ldots, n-1, c = 0, 1\} \cup \{(n, 0)\}$. In state $(n, 0)$, all n ambulances are busy and therefore the term compliance is not meaningful. We arbitrarily set $C(t) = 0$ when $B(t) = n$. Figure 2.4 show a transition rate diagram for this process when $n = 4$.

We assume that whenever the number of busy ambulances increases or decreases, the system goes out of compliance. This assumption is made to limit the number of configurations that we need to consider when computing expected travel times.

The Markov chain has three types of transitions:

Call arrival: Call arrivals occur at rate λ. Arrivals are lost when $B(t) = n$. We assume that the system is out of compliance immediately after an arrival, and therefore an arrival to state $(b, 0)$ and an arrival to state $(b, 1)$ both result in a transition to $(b+1, 0)$, for $b = 0, \ldots, n-1$.

Call completion: Call completions occur at rate $b\mu_{b,c}$ when in state (b, c), for $b > 0$. Similar to arrivals, we assume that the system is out of compliance immediately after a call completion and therefore a call completion when in state $(b, 0)$ and a call completion when in state $(b, 1)$ both result in a transition to state $(b-1, 0)$, for $b > 0$. We interpret $\mu_{b,c}$ as the rate at which each individual busy unit completes its call, given that the system is in state (b, c).

Achieve compliance: When the system is out of compliance, in state $(b, 0)$, we assume it reaches compliance at rate γ_b, resulting in a transition to state $(b, 1)$, for $b < n$.

If $\mu_{b,c}$ is independent of b and c and if γ_b approaches infinity for all b, then this model approaches the Erlang B (or $M/M/n/n$) loss model. Given that the steady state probabilities of the Erlang B model are insensitive to the service time distribution beyond its mean (Gross and Harris, 1998, p. 245), the similarity of our model to the Erlang B model suggests that the steady state probabilities for our model might be relatively insensitive to the shape of the service time distribution. We confirm, in Section 2.6, that the steady state probabilities for the Markov chain model are close to those of a simulation model that makes more realistic assumptions about the distributions of service time and time to reach compliance (see Figure 2.6).

If all transition rates are known, then it is straightforward to compute the steady state probabilities, $\pi_{b,c}$, for our model, as we describe next. For convenience in stating our equations, we set $\mu_{0,1} = \mu_{n,1} = \pi_{n,1} = 0$. The local balance equation for state $(b, 1)$ is

$$(\lambda + b\mu_{b,1})\pi_{b,1} = \gamma_b\pi_{b,0} \Rightarrow \pi_{b,1} = \frac{\gamma_b}{\lambda + b\mu_{b,1}}\pi_{b,0}, \text{ for } b = 0, \ldots, n-1, \tag{2.3.1}$$

and this equation allows us to compute $\pi_{b,1}$ when $\pi_{b,0}$ is known. Next, for a given b, if we consider transitions between states in $\{(b', c) \in \Omega : b' \leq b\}$ and states in $\{(b', c) \in \Omega : b' \geq b + 1\}$, we obtain the following balance equation:

$$\lambda(\pi_{b,0} + \pi_{b,1}) = (b+1)\mu_{b+1,0}\pi_{b+1,0} + (b+1)\mu_{b+1,1}\pi_{b+1,1} \text{ for } b = 0, \ldots, n-1. \tag{2.3.2}$$

By combining equations (2.3.1) and (2.3.2), we obtain the following, which allows us to compute $\pi_{b,0}$ once $\pi_{b+1,0}$ and $\pi_{b+1,1}$ are known:

$$\pi_{b,0} = \frac{(b+1)(\lambda + b\mu_{b,1})(\mu_{b+1,0}\pi_{b+1,0} + \mu_{b+1,1}\pi_{b+1,1})}{\lambda(\lambda + \gamma_b + b\mu_{b,1})}, \quad \text{for } b = 0, \ldots, n-1. \tag{2.3.3}$$

We compute the steady state probabilities recursively, starting with state $(n, 0)$, as follows.

Algorithm 1:

1. Set $\pi_{n,0} \leftarrow 1$.
2. Decrement b from $n-1$ to 0 in steps of 1. For each value of b, use (2.3.3) to compute $\pi_{b,0}$ and use (2.3.1) to compute $\pi_{b,1}$.
3. Normalize: Set $\Pi \leftarrow \sum_{(b,c)\in\Omega} \pi_{b,c}$ and $\pi_{b,c} \leftarrow \pi_{b,c}/\Pi$.

In order to solve the model, we need to estimate the call completion rates, $\mu_{b,0}$, and the rates at which compliance is reached, γ_b, from the non-compliant states, using data on the service times $S_{b,c}$ and other relevant information. Equation (2.2.1) expresses the service time $S_{b,c}$ as a sum of components and one might think that the inverse of the call completion rate, $1/\mu_{b,c}$ could be estimated as the sum of historical averages for the service time components, but this procedure is inappropriate for two reasons:

1. The observed values of $S_{b,c}$ were influenced by the repositioning policy that was in effect when the data was collected, whereas we want to use the model to investigate other possible repositioning policies.
2. The expected value of the service time, $\mathrm{E}[S_{b,c}]$, is not equal to $1/\mu_{b,c}$, because the call completion rates are state-dependent.

We discuss how to address the first of these issues in Section 2.4. To avoid complications that result from the second issue, we make the following approximation:

$$\mu_{b,c} = 1/\mathrm{E}[S_{b,c}]. \tag{2.3.4}$$

Similarly, we define random variable E_b as the time it takes to achieve compliance from state $(b, 0)$, by repositioning one or more ambulances, and we use the following approximation:

$$\gamma_b = 1/\mathrm{E}[E_b]. \tag{2.3.5}$$

The rates $\mu_{b,0}$ and γ_b depend on exactly where the $n - b$ available ambulances are located. We view the non-compliant state $(b, 0)$ as an aggregation of a collection of possible location configurations for the available ambulances, with each configuration weighted by its probability of occurrence (see Section 2.4). The collection of configurations can be divided in two subsets that correspond to the non-compliant state $(b, 0)$ having been entered either as a result of a call arrival or a call completion. We need to know the relative likelihood of these two possibilities in order to weigh the two subsets of configurations appropriately. Call arrivals into state $(b, 0)$ occur at rate:

$$\alpha_b \equiv \lambda(\pi_{b-1,0} + \pi_{b-1,1}) \tag{2.3.6}$$

and call completions into state $(b, 0)$ occur at rate:

$$\beta_b \equiv (b + 1)(\mu_{b+1,0}\pi_{b+1,0} + \mu_{b+1,1}\pi_{b+1,1}). \tag{2.3.7}$$

Therefore, if the process is in state $(b, 0)$, the probability that it was entered via a call arrival is $\alpha_b/(\alpha_b + \beta_b)$ and the probability that the state was entered via a call completion is $\beta_b/(\alpha_b + \beta_b)$. We assume that the arrival rate, λ, the call completion rate $\mu_{b,1}$ in the compliant states, and the following parameters are available as input to the Markov chain

model:

$$\begin{aligned}
\tau_{b,0,\text{arrival}} &= \text{E}[S_{b,0}|(b,0) \text{ entered via a call arrival}], \\
\tau_{b,0,\text{completion}} &= \text{E}[S_{b,0}|(b,0) \text{ entered via a call completion}], \\
\phi_{b,0,\text{arrival}} &= \text{E}[E_b|(b,0) \text{ entered via a call arrival}], \\
\phi_{b,0,\text{completion}} &= \text{E}[E_b|(b,0) \text{ entered via a call completion}].
\end{aligned}$$

(We discuss how to estimate the input parameters in Section 2.4.) Using these inputs and the expressions for the probability of entering state $(b,0)$ via a call arrival vs. a call completion, we compute the expected service time and the expected time to reach compliance when in state $(b,0)$:

$$\begin{aligned}
\text{E}[S_{b,0}] &= \frac{\alpha_b \tau_{b,0,\text{arrival}} + \beta_b \tau_{b,0,\text{completion}}}{\alpha_b + \beta_b}, \\
\text{E}[E_b] &= \frac{\alpha_b \phi_{b,0,\text{arrival}} + \beta_b \phi_{b,0,\text{completion}}}{\alpha_b + \beta_b}.
\end{aligned}$$

The resulting expressions for the call completion rates and the rate of reaching compliance are:

$$\mu_{b,0} = \frac{\alpha_b + \beta_b}{\alpha_b \tau_{b,0,\text{arrival}} + \beta_b \tau_{b,0,\text{completion}}}, \tag{2.3.8}$$

$$\gamma_b = \frac{\alpha_b + \beta_b}{\alpha_b \phi_{b,0,\text{arrival}} + \beta_b \phi_{b,0,\text{completion}}}. \tag{2.3.9}$$

To determine the steady state probabilities $\pi_{b,c}$, we need to iterate between solving the balance equations and computing the call completion rates and the rates of reaching compliance. We use the following iterative algorithm. The superscript k is an iteration counter.

Algorithm 2:

1. Initialization: Set $\pi^0_{b,c} \leftarrow 1/(2n+1)$ for all $(b,c) \in \Omega$.

2. Use (2.3.6)-(2.3.9) to compute α_b, β_b, γ_b and $\mu_{b,0}$ for $b = 0, \ldots, n-1$ and set $\mu_{n,0} =$

$1/\tau_{n,0,\text{arrival}}$.

3. Use *Algorithm 1* to solve the balance equations to obtain $\pi_{b,c}^{k+1}$. Set $k \leftarrow k+1$.

4. Iterate steps 2 and 3 until $|\pi_{b,c}^{k+1} - \pi_{b,c}^{k}|/\pi_{b,c}^{k} < \epsilon$, $|\mu_{b,0}^{k+1} - \mu_{b,0}^{k}|/\mu_{b,0}^{k} < \epsilon$, and $|\gamma_{b}^{k+1} - \gamma_{b}^{k}|/\gamma_{b}^{k} < \epsilon$ for all $(b,c) \in \Omega$.

The two-dimensional model can be reduced to a less realistic but simpler one-dimensional birth-death model by assuming that the rates of reaching compliance γ_b approach infinity for all b. The resulting birth-death process has a constant birth rate (λ) and state-dependent death rates ($\mu_{b,1}$) that do not depend on the probabilities α_b and β_b, which eliminates the need for iteration. We compare the accuracy of the one-dimensional and two-dimensional models in Section 2.7.

2.4 Parameter Estimation for the Markov Chain Model

The Markov chain model requires as input parameters the arrival rate λ, the average service times $\tau_{b,1} = 1/\mu_{b,1}$ in the compliant states, the average service times $\tau_{b,0,\text{arrival}}$ and $\tau_{b,0,\text{completion}}$ in the non-compliant states, and the average times to reach compliance $\phi_{b,0,\text{arrival}}$ and $\phi_{b,0,\text{completion}}$. The arrival rate is straightforward to estimate from empirical data. In this section, we describe how we estimate the state-dependent input parameters.

From (2.2.1) we obtain:

$$\begin{aligned}
\tau_{b,0,\text{arrival}} &= \mathrm{E}[S_{b,0}^{\text{Chute}}] + \mathrm{E}[S_{b,0}^{\text{On-Scene}}] + \Pr\{PT\}(\mathrm{E}[S_{b,0}^{\text{Transport}}] + \mathrm{E}[S_{b,0}^{\text{Hospital}}]) \\
&\quad + \mathrm{E}[S_{b,0}^{\text{Travel}} | (b,0) \text{ entered via a call arrival}], \text{ for } b = 1, \ldots, n, \\
\tau_{b,0,\text{completion}} &= \mathrm{E}[S_{b,0}^{\text{Chute}}] + \mathrm{E}[S_{b,0}^{\text{On-Scene}}] + \Pr\{PT\}(\mathrm{E}[S_{b,0}^{\text{Transport}}] + \mathrm{E}[S_{b,0}^{\text{Hospital}}]) \\
&\quad + \mathrm{E}[S_{b,0}^{\text{Travel}} | (b,0) \text{ entered via a call completion}], \text{ for } b = 0, \ldots, n-1.
\end{aligned}$$

We estimate the means of $S_{b,0}^{\text{Chute}}$, $S_{b,0}^{\text{On-Scene}}$, and $S_{b,0}^{\text{Hospital}}$ and the probability $\Pr\{PT\}$ directly from empirical data. That leaves the estimation of $\mathrm{E}[S_{b,0}^{\text{Transport}}]$, $\mathrm{E}[S_{b,0}^{\text{Travel}} | (b,0)$ entered via a call arrival]

and $\mathrm{E}[S^{\mathrm{Travel}}_{b,0}|(b,0)$ entered via a call completion]. Our focus in this section is on the two travel time means. We used similar procedures to estimate the transport time mean but we omit the details.

For the purpose of estimating $\mathrm{E}[S^{\mathrm{Travel}}_{b,0}|(b,0)$ entered via a call arrival], we assume that at the moment prior to the arrival transition all ambulances were located in compliance for $(b-1)$ busy ambulances. (Note that we make this assumption only for the purpose of estimating the mean travel time. The Markov chain model allows arrival transitions to $(b,0)$ not only from $(b-1,1)$ but also from $(b-1,0)$.) We begin with the ambulance locations specified in the compliance table for $b-1$ busy ambulances and we construct different configurations by removing one of the available ambulances, corresponding to the location of the arriving call. We define the catchment area of an ambulance as the set of demand nodes for which the ambulance is the closest. We define $A_{b,j}, j \neq 0$ as the j-th configuration of the available ambulances in state $(b,0)$ and we use negative values of $j \in J_{b,\mathrm{arrival}} = \{-(n-b+1),\ldots,-1\}$ to index the possible configurations. We assign probabilities $\Pr\{A_{b,j}\}$ to each configuration, equal to the sum of the probabilities f_d for demand nodes d in the catchment area of the ambulance answering the call that generated the configuration.

Similarly, to estimate $\mathrm{E}[S^{\mathrm{Travel}}_{b,0}|(b,0)$ entered via a call completion] we assume that at the moment prior to the call completion transition, all ambulances were located in compliance for $(b+1)$ busy ambulances, and we therefore have $(n-b-1)$ available ambulances with known locations. The ambulance that completed its call could have done so at a hospital $h \in H$, with probability $\Pr\{A_{b,h}\} = \Pr\{PT\} \times g_h$ or it could have completed the call at a demand node $d \in D$, because no transport was required, with probability $\Pr\{A_{b,|H|+d}\} = (1-\Pr\{PT\}) \times f_d$. Note that we index the $|H|$ configurations $A_{b,j}$ for calls completed at a hospital using $j = 1,\ldots,|H|$, we index the $|D|$ configurations for calls completed at a demand node using $j = |H|+1,\ldots,|H|+|D|$, and we define the index set $J_{b,\mathrm{completion}} = \{1,\ldots,|H|+|D|\}$.

We reserve configuration $A_{b,0}$ for the case when the ambulances are located in compliance.

For each configuration $A_{b,j}$, we can enumerate $|D|$ possible trips for the next call arrival, with one trip for each demand node $d \in D$, with probability f_d, and a travel distance l from the closest available ambulance to d based on configuration $A_{b,j}$. We store distances in an $(|S| + |H| + |D|) \times |D|$ matrix L, where $L[o, d]$ is the distance from origin $o \in S \cup H \cup D$ (a station, hospital, or demand node) to destination $d \in D$ (a demand node). For each configuration $A_{b,j}$, we store trip probabilities in a matrix $P_{b,j}$ of the same dimension as L, where $P_{b,j}[o, d]$ is the probability that the next call arrival will be to demand node d and that the closest available ambulance, given by configuration $A_{b,j}$, is at origin o. Note that column d of $P_{b,j}$ will have only one non-zero entry, equal to $f_d \Pr\{A_{b,j}\}$.

In a real system, for each state $(b, 0)$ there is an infinite set of configurations specifying the locations of each available ambulance, but we approximate this infinite set with the finite set indexed by $J_{b,0} = J_{b,\text{arrival}} \cup J_{b,\text{completion}} = \{-(n - b + 1), \ldots, -1, +1, \ldots, |D| + |H|\}$. For a compliant state $(b, 1)$, by definition there is only one configuration, $A_{b,0}$, and therefore $J_{b,1} = \{0\}$.

We add the matrices $P_{b,j}$ to obtain

$$P_{b,\text{arrival}} = \sum_{j \in J_{b,\text{arrival}}} P_{b,j}$$

$$P_{b,\text{completion}} = \sum_{j \in J_{b,\text{completion}}} P_{b,j},$$

where $P_{b,\text{arrival}}[o, d]$ and $P_{b,\text{completion}}[o, d]$ give the probability that the next call arrival will result in an ambulance traveling from ambulance location o to demand node d, when in state $(b, 0)$, given that the last change to the number of busy ambulances was a call arrival or a call completion, respectively. Given parameters a, v_c, b_0, b_1, and b_2 and the distance $L[o, d]$, we can now use (2.2.4)-(2.2.5) and (2.2.7)-(2.2.8) to compute the first and second moment of the travel time from o to d. By aggregating over origin-destination pairs, we can compute the mean travel time to the next call, given that the system is in state $(b, 0)$ and

conditional on the previous state as:

$$\mathrm{E}[S_{b,0}^{\mathrm{Travel}}|(b,0) \text{ entered via a call arrival}] = \sum_{(o,d)\in\Gamma} \mathrm{E}[T(L[o,d])]P_{b,\mathrm{arrival}}[o,d] \quad (2.4.1)$$

$$\mathrm{E}[S_{b,0}^{\mathrm{Travel}}|(b,0) \text{ entered via a call completion}] = \sum_{(o,d)\in\Gamma} \mathrm{E}[T(L[o,d])]P_{b,\mathrm{completion}}[o,d], \quad (2.4.2)$$

where $T(l)$ is the travel time random variable for a trip of distance l. For future reference, we note that we can obtain the second moment and variance of the travel time when in state $(b,0)$ using

$$\begin{aligned}\mathrm{E}\left[(S_{b,c}^{\mathrm{Travel}})^2\right] = {} & \frac{\alpha_b}{\alpha_b+\beta_b} \sum_{(o,d)\in\Gamma} \mathrm{E}[(T(L[o,d]))^2]P_{b,\mathrm{arrival}}[o,d] \\ & + \frac{\beta_b}{\alpha_b+\beta_b} \sum_{(o,d)\in\Gamma} \mathrm{E}[(T(L[o,d]))^2]P_{b,\mathrm{completion}}[o,d] \quad (2.4.3)\end{aligned}$$

$$\mathrm{var}[S_{b,c}^{\mathrm{Travel}}] = \mathrm{E}[(S_{b,c}^{\mathrm{Travel}})^2] - (\mathrm{E}[S_{b,c}^{\mathrm{Travel}}])^2. \quad (2.4.4)$$

We use the same assumption to estimate $\phi_{b,0,\mathrm{arrival}}$ and $\phi_{b,0,\mathrm{completion}}$: That just before the most recent call arrival or a call completion, the system was in compliance, and just after the call arrival or completion, the system is in one of the configurations that we have listed.

In order to estimate the average times to reach compliance, we compare the configuration $A_{b,j}$ just after an event (a call arrival or completion) to the corresponding compliant configuration $A_{b,0}$. We define two sets:

$$\begin{aligned}\text{Non-compliant ambulances: } O_{b,j} &= \text{locations in } A_{b,j} \text{ but not in } A_{b,0} \\ \text{Non-compliant stations: } U_{b,j} &= \text{locations in } A_{b,0} \text{ but not in } A_{b,j}\end{aligned}$$

Both sets are conditional on having just arrived to state $(b,0)$ and the ambulance locations being specified by the configuration $A_{b,j}$. We need to move the non-compliant ambulances to the non-compliant stations. Our models (both the Markov chain model and the sim-

ulation model) can accommodate any method for matching non-compliant ambulances to non-compliant stations. In the computational experiments that we report in this paper, we used the following greedy heuristic:

1. Set $m = 1$ (an iteration counter).
2. Select the ambulance-station pair ($o_i \in O_{b,j}, u_k \in U_{b,j}$) with the shortest distance and record that distance as $r_{b,j,m}$.
3. Eliminate o_i from $O_{b,j}$ and u_k from $U_{b,j}$ and set $m = m + 1$.
4. Repeat Steps 2 and 3 until the sets $O_{b,j}$ and $U_{b,j}$ are empty (that is, until all non-compliant ambulances have been assigned to non-compliant stations).
5. Set $r_{b,j} = r_{b,j,m-1}$.

The relocation distance for the last assignment, $r_{b,j}$, is the maximum distance. It corresponds to the estimated time $\mathrm{E}[T(r_{b,j})]$ to reach compliance, starting from configuration $A_{b,j}$. Using the configuration probabilities, we obtain the following weighted average times to reach compliance:

$$\phi_{b,0,\text{arrival}} = \sum_{j \in J_{b,\text{arrival}}} \mathrm{E}[T(r_{b,j})] \Pr\{A_{b,j}\} \tag{2.4.5}$$

$$\phi_{b,0,\text{completion}} = \sum_{j \in J_{b,\text{completion}}} \mathrm{E}[T(r_{b,j})] \Pr\{A_{b,j}\}. \tag{2.4.6}$$

2.5 Approximating the Response Time Distribution

To compute system performance measures, we require the cumulative distribution function of the response time for urgent calls, $F_R(x)$. We compared three methods to approximate this distribution: (1) moment-matching to a lognormal distribution, (2) the moment generating function based method described in Wu et al. (2005) to approximate the sum of lognormal random variables, and (3) convolution. Our numerical experiments indicated

that moment-matching was the least accurate method and convolution was the most accurate method. Therefore, we describe the convolution method in this section and we use that method in the numerical experiments that follow.

We begin by computing the mean, second moment, and variance of the three components of the response time—evaluation and dispatch; chute; and travel time. For example, we perform the following computations for the evaluation and dispatch time:

$$\begin{aligned}
\mathrm{E}[R^{\text{Eval and Disp}}] &= \sum_{(b,c)\in\Omega} \pi_{b,c}\mathrm{E}[R_{b,c}^{\text{Eval and Disp}}] \\
\mathrm{E}[(R^{\text{Eval and Disp}})^2] &= \sum_{(b,c)\in\Omega} \pi_{b,c}\mathrm{E}[(R_{b,c}^{\text{Eval and Disp}})^2] \\
\mathrm{Var}[R^{\text{Eval and Disp}}] &= \mathrm{E}[(R^{\text{Eval and Disp}})^2] - \mathrm{E}[R^{\text{Eval and Disp}}]^2.
\end{aligned}$$

For the travel time, we compute the first two moments of $S_{b,0}^{\text{Travel}}$ using (2.4.1)-(2.4.3) (and similar but simpler formulas for state $S_{b,1}^{\text{Travel}}$). In all cases, to compute the state probabilities $\pi_{b,c}$, we use parameters that are estimated using data for all calls, but to compute the components of the response time, we use parameters that are estimated using only data for urgent calls.

Next, we approximate $R^{\text{Eval and Disp}}$, S^{Chute}, and S^{Travel} as independent lognormally distributed random variables. The parameters of the lognormal distribution for the evaluation and dispatch time are determined as follows (and similarly for the chute and travel times):

$$\mu = \ln(\mathrm{E}[R^{\text{Eval and Disp}}]) - \frac{1}{2}\ln\left(1 + \frac{\mathrm{Var}[R^{\text{Eval and Disp}}]}{\mathrm{E}[R^{\text{Eval and Disp}}]^2}\right) \tag{2.5.1}$$

$$\sigma^2 = \ln\left(1 + \frac{\mathrm{Var}[R^{\text{Eval and Disp}}]}{\mathrm{E}[R^{\text{Eval and Disp}}]^2}\right) \tag{2.5.2}$$

Finally, we approximate the lognormal distributions with discrete distributions and use discrete convolution to compute the distribution of $R = R^{\text{Eval and Disp}} + S^{\text{Chute}} + S^{\text{Travel}}$. The lognormality assumption is supported by the 2008 Edmonton EMS data, but the convolution approach can accommodate other distributions as well.

2.6 Markov Chain Model Validation

We developed a discrete event simulation model using the Stochastic Simulation in Java (SSJ) simulation library (L'Ecuyer, 2009) to validate the Markov chain model, which we programmed in Matlab. The simulation model uses realistic distributions for the response and service time components, as discussed in Section 2.2, and it models the details of the dispatching and repositioning of ambulances. In the convolution, we used a bin size of 0.01 minutes to discretize the lognormal distributions referred to in the preceding section. In this section, we compare average travel times, state probabilities, and the response time distributions from the two models.

We used the road network and current ambulance station locations for Edmonton, Alberta and we assumed a fleet of 19 ambulances. For the base case, we chose an arrival rate of $\lambda = 6$ calls per hour (average hourly arrival rates in Edmonton varied from 4 to 10 per hour in 2008). We used a compliance table that was adapted from compliance tables that have been used in Edmonton. Since our Edmonton data set does not contain information about the compliance status c, we assumed that the means for evaluation and dispatch, chute, on-scene, travel, and hospital times were independent of c, given b. The mean travel times were estimated using the procedures from Section 2.4 and those means depend on c. The simulation results in this section are based on 100 replications of approximately 20,000 calls each (excluding a warm-up period with approximately 2,000 calls), which took roughly 5 minutes of computation time. The average simulated service time was 77.7 minutes and the average simulated ambulance utilization was 40.9%, which is consistent with typical utilization levels in EMS systems that we are familiar with. The Markov chain calculations took 6.8 seconds and converged in five iterations.

Figure 2.5 shows that mean travel times increase with the number of busy ambulances, as expected. The mean travel time more than doubles, from four to nine minutes, when the number of busy ambulances increases from zero to 18 (corresponding to a single available ambulance.) The figure also shows that mean travel times are shorter when the system is in

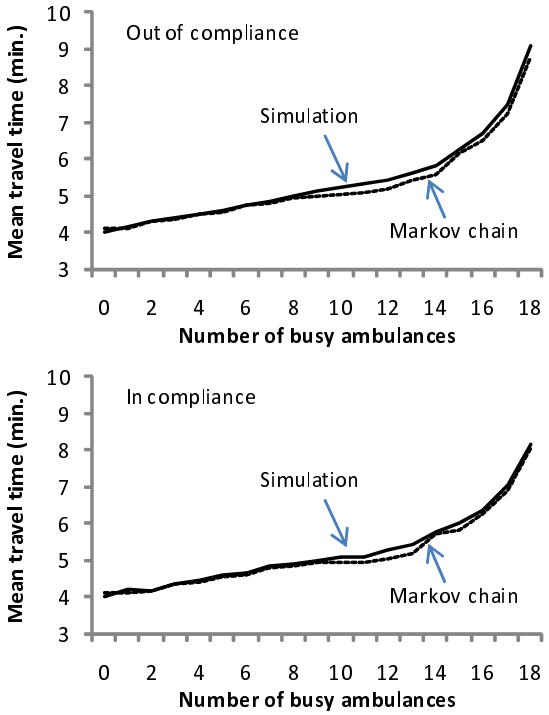


Figure 2.5: Mean travel times, based on the simulation and Markov chain models.

compliance. (Note that this is not guaranteed to happen—poorly designed compliance tables could *increase* mean travel times.) The mean travel times from the Markov chain model are, on average, within 2.0% of the simulated mean travel times. Figure 2.6 shows that the probability distributions for the number of busy ambulances (probability of compliance and out of compliance added) from simulation and from the Markov chain agree closely, differing by at most 0.003. The probabilities of compliance from the two models also agree closely, with 0.332 from the Markov chain model and 0.329 from the simulation. Figure 2.7 shows the response time distribution for urgent calls obtained from the simulation and Markov chain models.

We repeated the computations for four variations from the base case: a *uniform demand* case, a *Euclidean distance* case, a *high load* case, and a *low load* case. The purpose of the uniform demand and Euclidean distance cases was to examine situations whose geographic

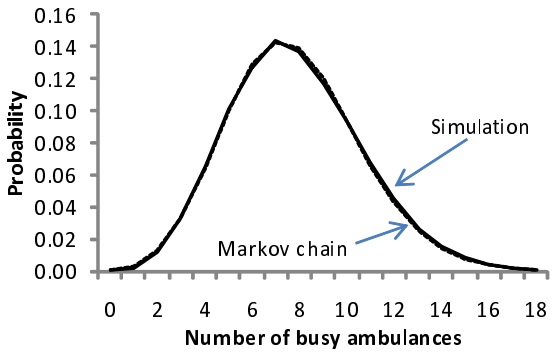


Figure 2.6: Probability distributions for number of busy ambulances

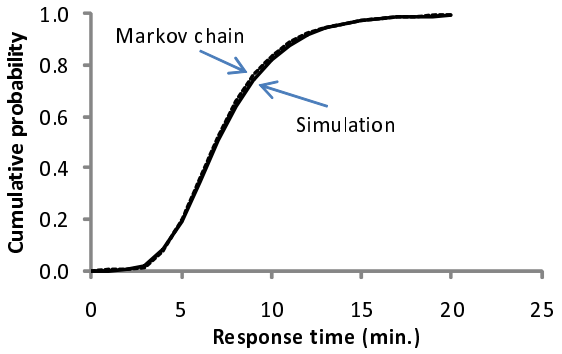


Figure 2.7: Response time distribution for urgent calls

structure differed from that in Edmonton. The spatial distribution of demand for EMS services in Edmonton is heavily concentrated in the city center (as is the case in many other cities). In the uniform demand case, we replaced the historical demand proportions f_d with $f_d = 1/|D|$ for all $d \in D$. The road network in Edmonton is mostly based on a rectangular grid, with the north and south regions separated by a river that runs through the city center. In the Euclidean distance case, we replaced the road network distances with Euclidean distances, which reduces the distance between any two nodes, especially those that are on opposite sides of the river. The accuracy of the average travel times, state probabilities, and the probability of response within nine minutes obtained from the Markov chain model was as good or better in these two cases as for the base case. The utilization and average service time decreased when moving from the base case to the uniform demand case to the Euclidean distance case, while the probability of response within nine minutes increased.

For the high load system, we increased the arrival rate until the probability that all ambulances are busy reached 0.05 and for the low load system, we decreased the arrival rate until the probability that all ambulances are available reached 0.05, keeping all other parameters constant. The resulting arrival rates of 10.74 per hour and 2.28 per hour are outside the range of average hourly arrival rates observed in Edmonton. We kept the number of ambulances fixed at 19 for stress-testing purposes, but in reality, the number of scheduled ambulances varies by time of day, making the high and the low load scenarios unrealistic. Comparing the low load, base, and high load cases, we see that the utilization increases roughly proportionally to the arrival rate, the average service time increases slightly with the load, and performance (measured by the response time probability) decreases with the load. The accuracy of the Markov chain estimates of the utilization, average service time, the state probabilities, and the response time probability was as good or better than in the base case except that the response time probability for the high load case differed by 3% from the simulation estimate (recall that the high load scenario is likely to be less realistic in practice).

In the next section, we investigate the accuracy of the Markov chain model when it is used to determine the best from a set of possible compliance tables.

2.7 Impact of Changing the Compliance Table

One important use that we envision for our model is to screen a large number of potential compliance tables to identify promising ones, that could then be evaluated more carefully via simulation or other means. The results in this section are for three scenarios—the base case from Section 2.6 and two variations (*small* and *large*) of the base case, in which the arrival rate and the number of ambulances are changed, so as to maintain an ambulance utilization of roughly 40%. The (λ, n) values for the three cases are $(4/\text{hr.}, 13)$, $(6/\text{hr.}, 19)$, and $(10/\text{hr.}, 32)$.

In order to assess the ability of the Markov chain model to identify the best compliance table, measured using simulation, we randomly generated 100 nested compliance tables, where "nested" means that the set of desired stations for $n - b$ available ambulances was a subset of the desired stations for $n - b + 1$ available ambulances, for $b = 1, ..., n - 1$. The compliance tables are for a set of 32 distinct ambulance locations, but for the small and base cases, we use only the first 13 and 19 rows of each table, respectively, whereas for the large case we use all 32 rows. We evaluated each compliance table for each of the three cases using both the Markov chain model and the simulation model, using the probability of reaching urgent calls within nine minutes as the performance measure.

The results are shown in Figure 2.8 and Table 2.2. The performance estimates from the simulation and Markov chain models are highly correlated (Spearman's rank order correlation coefficient ranges from 0.97 to 0.98). For two of the three cases, the compliance table ranked highest by the Markov chain model is also the one ranked highest by the simulation. The Markov chain performance estimates are slightly biased, compared to the simulation estimates but this does not impact the model's ability to identify the best-performing table. The difference in simulated performance between the best and the worst compliance table

Scenario:	Spearman correlation	Avg. estimation error	Max. estimation error	Optimality error	Performance range
Small	0.98	1.05%	5.1%	0.6%	15.2%
Base	0.97	1.03%	2.7%	0%	8.6%
Large	0.98	0.59%	1.4%	0%	6.6%

Table 2.2: Summary of compliance table comparisons.

ranges from 15.2 percentage points (for the small system) to 6.6 percentage points (for the large system). Increasing performance even by one percentage point for an EMS system for a city of this size typically requires considerable resources—on the order of adding one ambulance to the system, 24 hours a day. The wide range in performance across compliance tables indicates the importance of choosing compliance tables carefully.

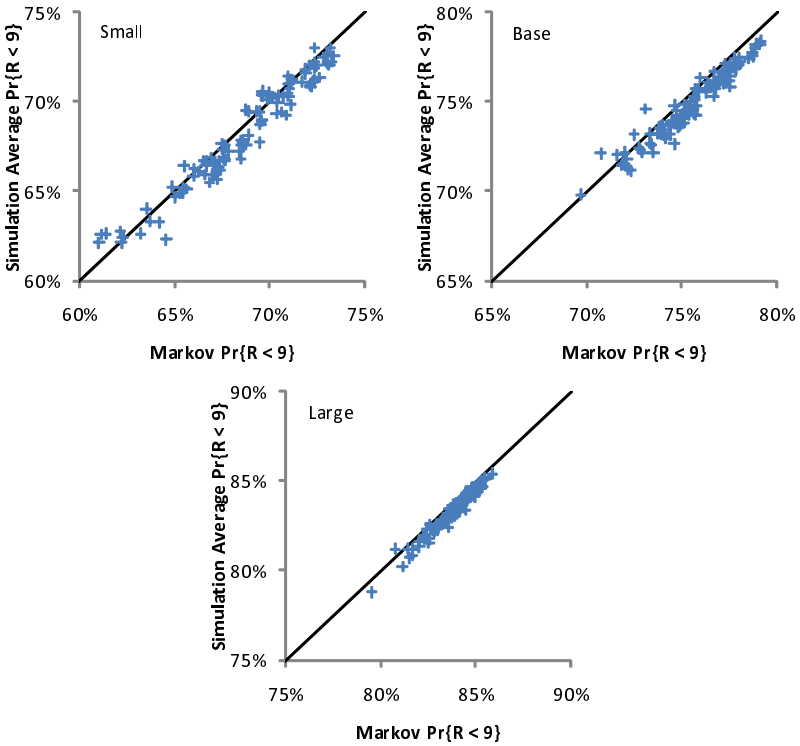


Figure 2.8: Response time probabilities from simulation and Markov chain model.

To quantify the differences between the performance estimates from the two models, we

define two measures: *Estimation error* and *optimality error*. We define the estimation error, Δ_{est}, for each compliance table as

$$\Delta_{\text{est}} = \frac{|M_{\text{Model}} - S_{\text{Sim}}|}{S_{\text{Sim}}} \times 100\%, \tag{2.7.1}$$

where S_{Sim} is the response time probability obtained from the simulation and M_{Model} is the response time probability obtained from the Markov chain model. From Table 2.2, we see that the maximum overall estimation error is 5.1% for one of the compliance tables for the small case, and that the estimation errors appear to decrease with the system size. (For comparison, the simulation estimates of the response time probabilities are accurate to about $\pm 0.1\%$ with 95% confidence.)

The optimality error is a measure of the ability of the Markov chain model to select the optimal or a near-optimal compliance table from a given set of tables, as measured by simulation. We define the optimality error, Δ_{opt}, as the percentage difference between the best response time probability according to the simulation results and the response time probability obtained from simulation for the compliance table chosen as the best by the Markov chain model, that is:

$$\Delta_{\text{opt}} = \frac{|S_{\text{Model}} - S_{\text{Sim}}|}{S_{\text{Sim}}} \times 100\%, \tag{2.7.2}$$

where S_{Sim} is the best response time probability obtained from the simulation and S_{Model} is the response time probability obtained from the simulation for the compliance table chosen by the Markov chain model as the best. The simulation and the Markov chain model picked the same compliance table as the best one for the base case and the large case, resulting in zero optimality error, and the optimality error was 0.63% for the small case. The optimality error is our best estimate for the relative performance loss in the actual system, if the system planners decide to use the results of our Markov chain model, instead of the actual best compliance table in the test suite. If the system planners instead chose, say, the ten compliance tables that were ranked highest by the Markov chain model and evaluated each

Scenario:	Spearman correlation	Avg. estimation error	Max. estimation error	Optimality error
Small	0.96	1.75%	7.1%	0.6%
Base	0.96	1.62%	3.4%	0.3%
Large	0.97	0.88%	1.9%	0%

Table 2.3: Compliance tables using one-dimensional Markov model.

of those tables using simulation, then there would be even less performance loss (for the small case, the compliance table that was ranked highest by the Markov chain model was ranked fourth-highest by simulation).

We see from Figure 2.8 that the system performance increases from the small to the base to the large system (both on average and for almost all individual tables), even though the ambulance utilization is approximately constant for the three systems. We attribute the improvement to the economies of scale typically seen in multi-server queueing systems when the number of servers increases and to the increase in geographical coverage that results from having more ambulance locations. The results for the performance range suggest that the potential for performance improvement using dynamic repositioning is greater for small systems. For Edmonton, this suggests that the benefits of repositioning are greatest when the arrival rate and the number of ambulances is proportionally smallest, as one would expect.

We repeated the calculations described in this section using the one-dimensional Markov chain model described at the end of Section 2.3. The results are shown in Table 2.3. The one-dimensional model has estimation errors that are 50 to 70% higher on average and an optimality error that is worse for the base case (but identical for the small and large cases).

As discussed in Section 2.3, the one-dimensional model assumes that repositioning occurs instantly and therefore does not require iteration. A further simplification is that one only needs to generate configurations for ambulance locations that correspond to the ones in the compliance table and not for the many possible out-of-compliance configurations. Let us stress that although the two-dimensional model increases the complexity of the description of the procedure, it does not have much impact on computation times—the

two-dimensional model converges in five iterations or less for all scenarios that we have investigated. Furthermore, the two-dimensional model does not increase the amount of input data that is required—rather, it does more with the input data. Nevertheless, the simplified one-dimensional model could be useful. For example, we can imagine a three-stage optimization heuristic that initially uses the one-dimensional model, then the two-dimensional model, and finally simulation, as the search moves towards higher performing compliance tables and requires more accuracy to rank tables appropriately.

2.8 Conclusions

We have developed and validated a Markov chain model of an EMS system with repositioning. The model requires iteration between computing transition rates and computing steady state probabilities, but the computations converged in five iterations or less in all examples that we have tried (several hundred cases). The cardinality of the model's state space scales linearly with the number of ambulances and solving the model for a realistic system takes only a few seconds, making it suitable for use in decision support systems that allow EMS planners to experiment with compliance tables and to quickly see the predicted consequences. The model provides accurate estimates of the system response time distribution, the distribution of the number of busy ambulances, and various other performance measures.

We found that the Markov chain model provided estimates that were typically within 2% of estimates found using a more realistic simulation model. When we used the Markov chain model to choose the best from a set of 100 random compliance tables, under realistic conditions, it selected either the one that performed best or very close to the best, as measured by simulation. This, together with short computation times, bodes well for using the Markov chain model as part of a heuristic to search for optimal compliance tables.

Acknowledgements

This work was partially supported by Grant 203534-07 from the Canadian Natural Sciences and Engineering Research Council. The authors thank Alberta Health Services for access to data, Dan Haight for useful discussions, seminar participants at the University of Toronto, University of Ottawa, and two anonymous referees and an associate editor for comments that led to considerable improvement in this chapter.

CHAPTER 3

Optimizing Compliance Tables for Ambulance Repositioning: Optimization Methods and Performance Bounds [1]

3.1 Introduction

As are most components of the health system, ambulance systems are under increasing pressure to improve performance. As a result, most EMS systems are constantly striving to obtain the maximum performance from the existing resources. One approach for this is system status management, which simply means adapting dynamically to changes in the status of the system. This translates into relocating the available ambulances to compensate for unbalanced coverage resulting from ambulances becoming busy or becoming available. System status management is in contrast to the practice, still in use in many small cities, of ambulances simply returning to their home stations after completing a call. System status management usually involves continuously monitoring the availability and location of ambulances, using GPS technology and modern vehicle dispatch systems. It also requires

[1] R. Alanis, A. Ingolfsson, B. Kolfal, and F. Campello. Unpublished. 2012. F. Campello assisted with the proof of Theorem 1.

Available Ambulances	Stations							
1	42/RAH							
2	42	42/RAH						
3	42	42/RAH	32					
4	42	42/RAH	32	33				
5	42	42/RAH	28/32	33	34/27			
6	42	42/RAH	28/32	33	34/27	7/25/NEC		
7	42	42/RAH	28/32	33	34/27	7/25/NEC	9/19/GNH/MIS	
8	42	42/RAH	32/UAH	33	34/27	7/25/NEC	9/GNH	19/28/MIS

Table 3.1: Example of compliance table.

a policy on when ambulance relocation is needed, which ambulance to move, and where to move it. This policy can be implemented at two different levels:

1. At the tactical level, a compliance table is created beforehand. For each possible number of available ambulances, it indicates where they should be located to maximize coverage. Table 3.1 presents an example of a compliance table used in Edmonton, Alberta. In the example, the first row indicates that if we have a single ambulance it should ideally be located either at ambulance station 42 or at the Royal Alexandra Hospital (RAH). If there are two ambulances, one should be at station 42 and the other at station 42 or at RAH, and so forth. Compliance tables can be developed by external consultants or local personnel; developing a good compliance table is the most difficult aspect of this approach. Once the table is available, the operation is relatively simple. The target locations are predefined, but the decision on which ambulance(s) to move is made at the operational level.

2. At the operational level, the situation is analyzed in real time, and when the ambulance locations differ from those in the compliance table a decision is made on which ambulance(s) to move. Bringing the system back into compliance results in a higher expected performance.

Compliance tables offer good performance and provide a solution that is easy to understand and to implement. We will focus on finding optimal or near-optimal compliance tables.

Available Ambulances	Stations			
1	42			
2	42	RAH		
3	42	RAH	32	
4	42	RAH	32	33

Table 3.2: Example of nested compliance table.

Available Ambulances	Stations			
1	42			
2	RAH	32		
3	42	RAH	33	
4	42	RAH	32	33

Table 3.3: Example of non-nested compliance table.

Most compliance tables used in real life are nested. This means that the row for i available ambulances duplicates the row for $i-1$ available ambulances and adds an extra station. The reasoning is that because the rows are nearly identical, the number of ambulance relocations will be minimized. Tables 3.2 and 3.3 show a nested and a non-nested compliance table. Non-nested tables have the potential to achieve higher performance, but the extra relocations are viewed negatively by ambulance crews and lead to additional costs for gas and maintenance. Therefore, non-nested tables are usually avoided.

In practice, finding a good compliance table is challenging. Some EMS organizations depend on external consultants, while others develop the tables in-house using simple empirical approaches. For example, one can rank the stations according to the number of calls that each station has responded to historically and use this ranking to develop a nested compliance table. This method has the drawback that the number of calls responded to was influenced by the deployment policies in use in the past. To avoid this problem, one can rank the stations according to the number of calls that originated in each station's catchment area, regardless of whether the station actually responded to the call. The use of empirical methods is usually suboptimal and in some cases can produce poor results, as we will see on the Results section.

Since our goal is to improve performance, we will start by defining this concept. Let the response time be the time from the moment that a call arrives to the moment that the ambulance arrives at the corresponding destination. Most EMS organizations measure performance by the percentage of high-priority calls for which the response time is below a certain threshold τ. Most organizations have specific performance targets, e.g., to respond to 90% of high-priority calls in less than nine minutes. The targets vary from city to city, and are different for urban and rural areas, but most follow the same pattern.

We make the following contributions:

1. We propose a framework to classify performance-evaluation models for EMS systems with repositioning. The classification depends on whether or not the model assumes that ambulances are always in compliance. It also differentiates between models with nested and non-nested compliance tables. We define a sequence of models from a Markov bidimensional model with out-of-compliance performance and nested compliance tables, to a Markov bidimensional model with in-compliance performance, to a birth-death model with in-compliance performance, to a birth-death model with in-compliance performance and non-nested compliance tables.

2. We use stochastic ordering to prove the ranking of the performance. We establish the performance of in-compliance models as an upper bound on the performance of models with out-of-compliance performance, and we establish the performance of models with non-nested tables as an upper bound on the performance of models with nested tables.

3. We propose an integer program (IP) to find optimal compliance tables under the assumption that ambulances are always in compliance, with a constraint on the expected frequency of ambulance relocations.

4. We use the previously implemented bidimensional Markov model in combination with heuristic search methodologies to find near-optimal solutions with a more realistic performance for both nested and non-nested compliance tables.

The remainder of this chapter is organized as follows: Section 3.2 surveys the related literature. Section 3.3 defines the optimization problem that we wish to solve. Section 3.4 presents our previously developed bidimensional Markov model, and in Section 3.5 we develop a framework for EMS models. Section 3.6 presents an IP that finds an optimal solution assuming that the ambulances are always in compliance. Section 3.7 presents heuristic searches based on our bidimensional Markov model to find both nested and non-nested compliance tables. Section 3.8 presents numerical results for the different models and heuristics under multiple scenarios, and in Section 3.9 we present our conclusions.

3.2 Literature Review

Until recently, most EMS models assumed that, after attending a call, an ambulance would return to its home station. It has now become common practice to use dynamic relocation, or system status management, where ambulances are relocated as conditions change, to improve the level of service. One of the first works to support the use of repositioning was by Kolesar and Walker (1974), who studied fire engine relocation. This work helped to establish relocation as an effective way to improve performance. Before analyzing how to relocate ambulances in a dynamic system we present the initial models that established the basics of ambulance location and provided the foundations for more advanced models.

One of the first well-known ambulance station location models, the location set covering model (LSCM), was proposed by Toregas et al. (1971). In the LSCM a set of candidate ambulance stations and a set of demand nodes are given, together with the distances or average travel times from all stations to all nodes. Coverage is defined in terms of a threshold on the distance or average travel time: a demand is covered if there is an ambulance station within the coverage threshold. All the demand nodes are equally weighted, and the goal is to find the minimum number of stations such that all the nodes are covered. LSCM was implemented as an integer programming model, and it is a good tool for finding the minimum number of ambulance stations to provide a minimum level of service to a geographical area.

However, it fails to take into account the fact that some demand nodes, such as those in the downtown core, have higher demands. For LSCM, high-density and low-density areas have the same importance.

A second model is the maximal covering location problem (MCLP). It was proposed by Church and ReVelle (1974). MCLP is also an IP model and it incorporates nodes with different demands. Instead of attempting to provide full coverage, it maximizes the demand covered with a limited number of ambulances. LSCM and MCLP illustrate a common tradeoff faced by EMS planners: on the one hand they must provide a minimum level of service to everyone, no matter how isolated their neighborhood; on the other hand, they must maximize the expected level of service of the whole system. The first requirement leads to an even distribution of stations, while the second leads to a concentration of stations in high-demand areas.

These two models share a fundamental limitation: they assume that the stations have ambulances available at all times. They fail to consider the stochastic nature of EMS and the fact that ambulances called into service leave their areas uncovered. Some of the first solutions that took this into account were based on maximizing the demand covered by two or more stations, to reduce the probability of having a region uncovered. An example is the work of Gendreau et al. (1997), which uses two coverage thresholds r_1 and r_2, with $r_1 < r_2$. Every demand node has to be within r_2 of a station, and at least a percentage α of the demand has to be within r_1 of a station. An integer programming model was presented and a tabu search heuristic was proposed for its solution. Daskin and Stern (1981) use a hierarchical objective to maximize the number of demand points with multiple coverage. Hogan and ReVelle (1986) maximize the total demand with double coverage.

The use of a multiple coverage model is certainly an improvement over LSCM, MCLP, and similar static location models. However, double coverage or multiple coverage is just a proxy for the actual probability that a demand area will be covered. The maximum expected covering location problem (MEXCLP) proposed by Daskin (1983) represents a significant improvement. The model assumes that there is a probability that an ambulance

will be busy, obtained from the average utilization of the ambulance time. It assumes that this probability is the same for all ambulances and allows more than one ambulance to be located at a station. It estimates the probability that a demand node will be covered based on the number of ambulances that are covering it. Daskin (1987) uses stochastic travel times rather than expected travel times. He also introduces the possibility of multi-vehicle calls, in which more than one vehicle can be dispatched in response to an emergency call.

There is a significant literature on EMS systems and ambulance location, but there has been limited research into the use of compliance tables. Some researchers focus on issues unrelated to performance, such as back pain associated with the time that paramedics spend in their vehicles (Morneau and Stothart, 1999). Others question the effectiveness of relocation (Bledsoe, 2003). Stout (1989) argues in favor of system status management and discusses common misconceptions and implementation mistakes that could affect its effectiveness. Erkut et al. (2005), in a report on a simulation study for the Edmonton EMS system, justifies system status management by concluding that returning to a home-station policy would require the addition of eight new ambulances to compensate for the reduction in performance. Similarly, Nair and Miller-Hooks (2009) analyzed the performance of repositioning strategies and static strategies for multiple scenarios, finding performance improvements of 1% to 6%.

In the area of compliance-table optimization, the main focus of this chapter, Gendreau et al. (2006) proposed an integer programming model to find the optimal compliance table. For every possible number of available ambulances it solves the equivalent of an MCLP, with additional constraints to limit changes in the number of stations between consecutive rows of the compliance table. The model also takes into account the percentage of time that each number of ambulances is available. We will discuss this model in detail in Section 3.6.

Another approach for dynamic relocation uses a real-time analysis of the current situation to decide on the best ambulance moves to maximize the performance of the system. Compared with compliance tables, this approach has the potential to provide better performance, but it requires tight integration with the ambulance dispatch system and uses more

computer resources. Zhang et al. (2009) found optimal or near-optimal redeployment policies for single-ambulance problems using dynamic programming. For multiple vehicles, this dynamic relocation problem has been solved using Markov decision processes and stochastic dynamic programming (Berman, 1981a,b); optimal solutions were found for small idealized systems. Similarly, Maxwell et al. (2009, 2010) used approximate dynamic programming to find solutions for larger and more realistic systems. Gendreau et al. (2001) used a parallel tabu search heuristic based on the locations of available ambulances and the geographical distribution of the demand.

As previously mentioned, our model focuses on the optimization of an EMS system with system status management based on compliance tables.

A final note related to the general topic of optimization: When we formulate an optimization problem, there is often a tradeoff between an accurate objective function and being able to solve the problem to provable optimality. At one extreme, we can simplify the objective function so that it can be expressed as a linear function of the decision variables, and the problem becomes a linear program or integer linear program. At the other extreme, we can use simulation or the numerical solution of a stochastic model to evaluate the objective function and search for better solutions using a heuristic that calls the stochastic model as a subroutine. As an example of this tradeoff we can mention the work of Erkut et al. (2008) where some of the models are integer linear programs, and one is a heuristic that calls the hypercube model to evaluate solutions. We will investigate both of these extremes in this chapter.

3.3 Problem Description

We define a compliance table as an array of variables $x_{j,k}$ for $k \in \{1, \ldots, n\}$ and $j \in \{1, \ldots, k\}$. The value of $x_{j,k}$ is 1 when an available ambulance is located at station j, given that there are k available ambulances at the time, and n is the total number of ambulances on duty.

We define a bidimensional Markov model as $(B^{(1)}(t), C^{(1)}(t))$, with $B^{(1)}(t)$ representing the number of busy ambulances at time t and $C^{(1)}(t)$ indicating the compliance (1) or lack of compliance (0) of the system at time t. The model will be detailed in full in the following section.

Let the performance function $f^{(1)}(b) \equiv \Pr\{R^{(1)} \leq \tau | B^{(1)}(t) = b\}$ be the probability that the response time $R^{(1)}$ is less than or equal to time τ, given b busy ambulances.

We define $\mathrm{E}[f^{(1)}(B^{(1)})]$ to be the expected performance of the system, and we can use it to define our problem:

$$\text{P: Maximize} \qquad \mathrm{E}[f^{(1)}(B^{(1)})] \tag{3.3.1}$$

$$\text{by changing} \qquad x_{j,k}, \qquad j \in \{1, \ldots, k\}; k \in \{1, \ldots, n\} \tag{3.3.2}$$

$$x_{j,k} \in \{0, 1\} \text{ for } j \in \{1, \ldots, k\}; k \in \{1, \ldots, n\} \tag{3.3.3}$$

In the following sections we will use multiple strategies to solve P approximately, including:

- Approximating the objective function via a linear function, thus formulating an approximate version of P as an integer linear program.
- Adding constraints on the frequency of relocations to the IP, and solving it parametrically, to obtain multiple solutions that can be evaluated using the original bidimensional Markov model.
- Solving the ILP without any nesting constraints to obtain an upper bound on the expected performance.
- Using a heuristic to solve P, with a more realistic objective function based on our bidimensional Markov model.

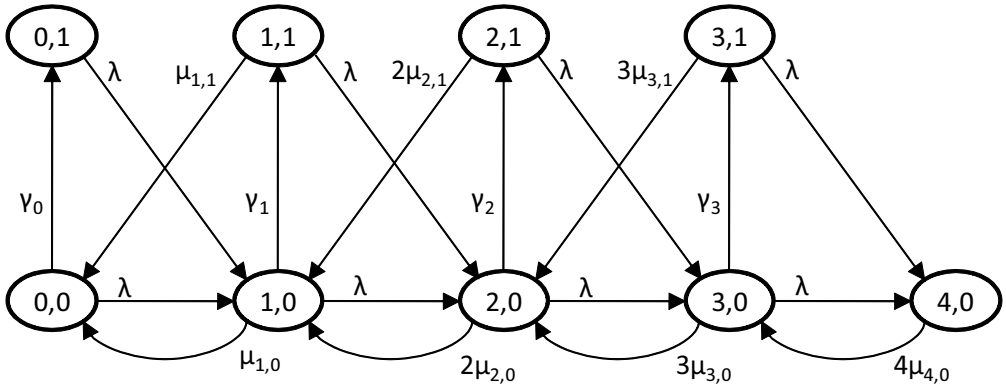


Figure 3.1: Transition rate diagram for $n = 4$

3.4 A Markovian EMS Performance Model

A Markov model was developed as part of previous research into estimating the performance of an EMS system using dynamic relocation based on a compliance table. This model was initially proposed as an analytical alternative to a simulation model. It is based on a bidimensional state space where in one dimension we have the number of available ambulances, $B(t)$, and in the second dimension we have the state of compliance of the system, $C(t)$. Being in compliance, defined as $C(t) = 1$, indicates that all the ambulances are in the positions specified by the compliance table; $C(t) = 0$ indicates that the system is out of compliance. This conceptualization that the system state is determined by the number of available ambulances and the state of compliance is fundamental for understanding the models presented.

The state space is represented in Figure 3.1. The model assumes a stable arrival rate λ and a service rate $\mu_{b,c}$ that depends on the state, in part because of the number of ambulances in service, and also because their locations affect the expected travel time.

The model is inspired by Larson's hypercube model (Larson, 1974, 1975). It has most of the same input parameters and produces most of the same outputs, plus others not available in the hypercube model. We will use some of these outputs, such as the state probabilities

and transition rates, as inputs for our IP formulation.

The model was tested against a simulation model based on the same input parameters for various scenarios. It was found to be extremely accurate at predicting the combined state probabilities (combining the out-of-compliance and in-compliance states), and the predicted performance was on average within 1% of the performance predicted by the simulation model. The performance rankings for multiple compliance tables predicted by the Markov model and the simulation are highly correlated. This model is therefore a good tool for heuristic searches for near-optimal compliance tables. It is also appropriate for decision-support systems that must quickly evaluate the effects of changes to resources and relocation policies.

By analyzing this model we obtain insights that clarify the behavior of the models that will be presented in the following sections. We present four variants of our bidimensional Markov model that help to set a framework of reference.

3.5 Model Hierarchy

Model 1: Our original two-dimensional Markov model: $(B^{(1)}(t), C^{(1)}(t))$, with transition rates λ, $\mu_{b,c}$, and γ_b, where $\mu_{b,1}$ is the in-compliance service rate, and $\mu_{b,0}$ is the out-of-compliance service rate, and γ_b is the rate of transitions from state $(b, 0)$ to state $(b, 1)$. This is our most realistic model, taking into account out-of-compliance performance.

Model 2: This is Model 1 with the out-of-compliance service rates equal to the in-compliance service rates: $(B^{(2)}(t), C^{(2)}(t))$, with transition rates λ, $\mu^{(2)}_{b,c} = \mu_{b,1}$ (for $c = 0$ and 1), and γ_b. We introduce this model as a "bridge" between Models 1 and 3, and we use this model to help prove that Model 3 provides a bound on Model 1.

Model 3: This is a one-dimensional model derived from our Markov model (Model 1) and presented in the same paper (Alanis et al., 2012): $B^{(3)}(t)$, with transition rates λ and $\mu^{(3)}_b = \mu_{b,1}$. This is a birth-death model.

Model 4: This is a one-dimensional birth-death model with maximal service rates: $B^{(4)}(t)$, with transition rates λ and $\mu_b^{\max}$. We assume that $\mu_b^{\max} \geq \mu_b^{(3)} = \mu_{b,1}$. We use this model, together with an integer program that we define in the next section, to obtain an upper bound on the performance of any compliance table, as measured by Model 1.

3.5.1 Assumptions

Service is at least as fast when in compliance: $\mu_{b,1} \geq \mu_{b,0}$ for all b. This is true if we have a well-chosen set of ambulance stations and an optimal or near-optimal compliance table, given that by definition an optimal compliance table minimizes the expected travel time and hence maximizes the service rate $\mu_{b,1}$.

Service is faster when fewer ambulances are busy: $\mu_{b,c} \geq \mu_{b+1,c}$ for all b and c. The reasoning behind this is that when fewer ambulances are busy, more ambulances are available. This should reduce the expected travel time (assuming that the ambulance locations tend toward the locations dictated by a near-optimal compliance table), and as a result the expected service rate is higher.

Service is faster when fewer ambulances are busy in model 4: $\mu_b^{max} \geq \mu_{b+1}^{max}$ for all b. The reasons are the same as in the previous case.

Model 4 has maximal service rates: $\mu_b^{\max} \geq \mu_{b,1}$. In this case the service rate was obtained using an optimal compliance table with no nesting constraints. Thus, the locations selected for state b are those that minimize the travel time, and hence maximize the service rate, without nesting constraints.

3.5.2 Stochastic Comparisons

We say that a random variable X is stochastically larger than a random variable Y, denoted $X \geq_{\text{st}} Y$, if $\Pr\{X > a\} \geq \Pr\{Y > a\}$ for all a (Ross, 1996). This definition can be extended

to say that a stochastic process $\{X(t), t \geq 0\}$ is stochastically larger than another process $\{Y(t), t \geq 0\}$ if $X(t) \geq_{\text{st}} Y(t)$ for all t (Muller and Stoyan, 2002).

Stochastic ordering relations can provide useful information about queuing systems. As an example, consider the following result for comparing birth-death processes $\{X(t), t \geq 0\}$ and $\{Y(t), t \geq 0\}$: if the arrival rate for process $X(t)$ is greater than or equal to that for $Y(t)$ for every state and the service rate for $X(t)$ is less than or equal to that for $Y(t)$ for every state, and the initial state at time 0 is such than $X(0) = k_x$ and $Y(0) = k_y$ and k_x is greater or equal than k_y, then the expected number of customers in the system for $X(t)$ is greater than or equal to that for $Y(t)$ (Bhaskaran, 1986):

$$\text{If} \qquad \lambda_i^{(x)} \geq \lambda_i^{(y)} \qquad i \geq 0 \tag{3.5.1}$$

$$\text{and} \qquad \mu_i^{(x)} \leq \mu_i^{(y)} \qquad i \geq 0 \tag{3.5.2}$$

$$\text{and} \qquad X(0) = k_x, Y(0) = k_y \text{ with } k_x \geq k_y \tag{3.5.3}$$

$$\text{then} \qquad X(t) \geq_{\text{st}} Y(t) \qquad \forall t \tag{3.5.4}$$

We will use this result in the proof of Theorem 1 below. We will use the theorem to obtain bounds on compliance-table performance.

Theorem 1. *Assume the following:*

- *Service is at least as fast when in compliance.*
- *Service is faster when fewer ambulances are busy.*
- *Service is faster when fewer ambulances are busy in model 4.*
- *Model 4 has maximal service rates.*
- *All ambulances are free at time zero in all models:* $B^{(1)}(0) = B^{(2)}(0) = B^{(3)}(0) = B^{(4)}(0) = 0$.

Then $B^{(1)} \geq_{\text{st}} B^{(2)} =_{\text{st}} B^{(3)} \geq_{\text{st}} B^{(4)}$.

Proof. We will use different approaches to prove each of the three relations in Theorem 1.

Proof that $B^{(1)} \geq_{\text{st}} B^{(2)}$

Assuming that both models start with no busy ambulances at $t = 0$, we want to show the following stochastic ordering between the number of busy ambulances in Model 1, $B^{(1)}(t)$, and the number of busy ambulances in Model 2, $B^{(2)}(t)$:

$$B^1(t) \geq_{st} B^2(t), \quad \forall t. \tag{3.5.5}$$

For $t = 0$ it is true that $B^{(1)}(t) \geq_{st} B^{(2)}(t)$. Let the two models have identical arrival streams, and let both models run freely until the first time t' where $B^{(1)}(t') = B^{(2)}(t') = b' > 0$. When this happens, construct all departures in Model 1 from the departures in Model 2. Each time a departure occurs in Model 2, let a departure occur in Model 1 with probability $p = \frac{\mu^{(1)}_{b',c}}{\mu^{(2)}_{b',c}}$. This is possible because $\mu^{(2)}_{b',c} \geq \mu^{(1)}_{b',c}$. This construction will keep the sample paths ordered with probability 1 at all times and ensure the proper distributions. We provide a more detailed proof of this relation in Appendix A.

Proof that $B^{(2)} =_{\text{st}} B^{(3)}$

Assume that models 2 and 3 start with no busy ambulances at $t = 0$. As in the proof of the first relation, let the two models have identical arrival streams. Use the stream of exponential random variables to generate transitions that increase or decrease $B(t)$ in the two models, and use a separate stream of exponential random variables to generate transitions that change $C(t)$ in Model 2. With this construction, the sample paths for $B(t)$ are identical for models 2 and 3 with probability 1.

Proof that $B^{(3)} \geq_{\text{st}} B^{(4)}$

By the definition of Model 4, $\mu^{(3)}_b \leq \mu^{\max}_b$. It follows directly from the results of Bhaskaran (1986) that $B^{(3)} \geq_{\text{st}} B^{(4)}$. □

3.5.3 Performance Evaluation

Let the performance function $f^{(k)}(b) \equiv \Pr\{R^{(k)} \leq \tau\}$ be the probability that the response time $R^{(k)}$ in Model $k, k = 1, 2, 3, 4$, is less than or equal to τ, given b busy ambulances.

We now add the assumption that the percentage of calls with a response time below τ, $f^{(k)}(b)$, is decreasing in b, and therefore $g^{(k)}(b) = 1 - f^{(k)}(b)$ is increasing in b.

Corollary 1. *Given the assumption that $g^{(k)}(b)$ is increasing in b,*

$$E[g^{(1)}(B^{(1)})] \geq E[g^{(2)}(B^{(2)})] = E[g^{(3)}(B^{(3)})] \geq E[g^{(4)}(B^{(4)})]$$

and

$$E[f^{(1)}(B^{(1)})] \leq E[f^{(2)}(B^{(2)})] = E[f^{(3)}(B^{(3)})] \leq E[f^{(4)}(B^{(4)})]$$

Proof. Follows from Theorem 1 above and the fact that $X \geq_{\text{st}} Y$ implies $\mathrm{E}[g(X)] \geq \mathrm{E}[g(Y)]$ for an increasing function g (Ross, 1996). □

We will use the relations in Corollary 1 to compute bounds on the optimal objective function value for problem P.

3.6 Integer Programming Model

The maximal expected coverage relocation problem (MECRP) as described by Gendreau et al. (2006) is closely related to our problem of finding an optimal compliance table. We begin by reviewing the MECRP formulation and then extend it. In its original formulation (Gendreau et al., 2006) the problem is defined on a graph $G = (V \cup W, A)$ where V is the set of demand points and W is the set of potential ambulance stations. We have $n \leq |W|$ emergency vehicles, and A is a set of arcs defined on $(V \cup W)^2$, where each arc has an associated deterministic driving time. Every vertex i in V corresponds to a demand region and has an associated demand value d_i, corresponding to the population of that region. A

station $i \in W$ covers a demand node $j \in V$ if and only if the driving time from i to j is below the coverage radius r. The model defines coverage by a group of sets W_i where W_i is a subset of W that includes all the vertices that cover $i \in V$. The model defines q_k as the probability of having k available ambulances, with k computed assuming that the number of available ambulances follows a binomial distribution with parameters n and p, where p is the average ambulance utilization.

MECRP defines variables $x_{j,k}$ where $x_{j,k}$ is 1 when an available ambulance is located at $j \in W$, given that there are k available ambulances at the time. The model assumes that there is only one ambulance per station; we will also make this assumption. Variable $y_{i,k}$ is 1 if a demand node i is covered by at least one ambulance when there are k available ambulances. Parameter α_k limits the number of differences between consecutive rows to $(\alpha_k + 1)$ to minimize the number of relocations required. Based on this definition the model is:

MECRP Formulation:

$$\text{Maximize:} \sum_{k=1}^{n} \sum_{i \in V} d_i q_k y_{i,k} \tag{3.6.1}$$

subject to:

$$\sum_{j \in W_i} x_{j,k} \geq y_{i,k} \qquad i \in V, k \in \{0, \ldots, n\} \tag{3.6.2}$$

$$\sum_{j \in W} x_{j,k} = k \qquad k \in \{0, \ldots, n\} \tag{3.6.3}$$

$$x_{j,k} - x_{j,k+1} \leq u_{j,k} \qquad j \in W, k \in \{1, \ldots, n-1\} \tag{3.6.4}$$

$$\sum_{j \in W} u_{j,k} \leq \alpha_k \qquad k \in \{1, \ldots, n-1\} \tag{3.6.5}$$

$$x_{j,k} \in \{0,1\} \qquad j \in W, k \in \{0, \ldots, n\} \tag{3.6.6}$$

$$y_{i,k} \in \{0,1\} \qquad i \in V, k \in \{0, \ldots, n\} \tag{3.6.7}$$

$$u_{j,k} \in \{0,1\} \qquad j \in W, k \in \{0, \ldots, n-1\} \tag{3.6.8}$$

The objective function 3.6.1 uses the variables $y_{i,k}$ to indicate which demand nodes i are being covered for each number of available ambulances k. As a result, we have the total of all the demand d_i covered when each number of ambulances k is available. By multiplying by the probability of k happening (q_k) we obtain the weighted average of the covered demand.

Constraint 3.6.2 has the effect of forcing all the variables $y_{i,k}$ to be 0 when the corresponding station j that could cover demand node i has not been assigned. We can cover a demand node only if a station within the covering distance has been assigned.

Constraint 3.6.3 ensures that exactly k stations are selected when there are k ambulances available.

Constraint 3.6.4 is used to enforce the level of nesting in the compliance table. If station j is selected when there are k ambulances available, and we have a nested compliance table, then it must also be selected when we have $k+1$ ambulances available. If it is not, then we have a non-nested table, and we will use $u_{j,k}$ to indicate the station that breaks the nested condition. We can potentially have more than one station breaking the nesting.

Finally, constraint 3.6.5 counts how many breaks in the nesting happen between rows k and $k+1$ in the compliance table, and it uses α_k to set a maximum number of breaks between consecutive rows.

Building on the formulation proposed by Gendreau et al. (2006), we developed a new formulation with the following improvements:

- The model changes from deterministic coverage to stochastic coverage. Previously, each demand node was either covered or not, depending on the distance. We now define $p_{i,j}$ to be the probability that demand node i can be reached in a response time below the coverage time τ by an ambulance dispatched from station j.
- The revised objective function more realistically represents the expected performance of the system in a more traditional way: as the percentage of calls with a response time below a given threshold.

- The handling of nesting is altered to take account of the fact that transitions between some states are more likely than transitions between others. The new model imposes constraints on the expected number of relocations given that a transition happens. This provides a finer control of the effect (expected number of relocations given a transition) rather than the cause (nesting violations). For this to work we assume that the number of nesting violations (plus one) can be used as a proxy for the number of expected relocations on a transition. However, we must be clear that this is just a proxy for the expected number. If for example, we analyze rows 2 and 3 of the compliance table in Table 3.3, we can see that station 32 is in row 2 but not in row 3. Therefore, we have 1 nesting violation and by our assumptions we should expect 2 relocations on a transition from state 2 to 3 or from state 3 to 2. From state 2 to 3 that would be the case if we were in compliance before the transition and we wanted to minimize the number of relocations. However, to minimize the time to reach compliance, it may be more effective to relocate several ambulances (up to the maximum number available) than to make a few long relocations. We may choose to relocate all 3 ambulances at once. In the other direction, from state 3 to 2 and assuming that the system was in compliance before the transition, we would require 2 relocations if the ambulance that became busy was at RAH but only 1 if it was at station 42 or 33. To minimize the relocation time, we may choose to relocate up to 2 ambulances.

To achieve these improvements we change the variables and parameters as follows:

- We change the interpretation of demand (d_i) from a value based on the population of the demand region to the percentage of the demand associated with the given region. This is important for the model to produce a realistic value in the objective function for the expected performance.
- We obtain the probability of a given number of ambulances (q_i) from the complementary Markov model previously discussed, rather than from a binomial distribution.

Based on simulation testing, this approach provides a closer match with the actual probabilities.

- Rather than using a coverage scheme based on sets of stations whose expected travel times are below a given threshold, we define the variable $p_{i,j}$ to be the probability that the response time to a call from demand node i assigned to an ambulance in station j is below the threshold τ.

- Variable $y_{i,j,k}$ is used to indicate if station j covers demand node i when there are k ambulances available.

- We add a new parameter m_k that represents $\Pr\{B(t)$ changes from k to k+1 or from k+1 to $k \mid$ a transition occurs where $B(t)$ changes, for $k = 0, \ldots, n-1\}$. Making use of our Markov model, we can compute its value based on the following equations:

$$\begin{aligned} m_k^* &= \pi_{k,1}\left(\frac{\lambda}{k\mu_{k,1}+\lambda}\right) + \pi_{k+1,1}\left(\frac{(k+1)\mu_{k+1,1}}{(k+1)\mu_{k+1,1}+\lambda}\right) && (3.6.9)\\ &+ \pi_{k,0}\left(\frac{\lambda}{k\mu_{k,0}+\lambda}\right) + \pi_{k+1,0}\left(\frac{(k+1)\mu_{k+1,0}}{(k+1)\mu_{k+1,0}+\lambda}\right) && (3.6.10)\\ m_k &= \frac{m_k^*}{\sum_{j=0}^{n-1} m_j^*} && (3.6.11) \end{aligned}$$

- We no longer need the set of parameters $\alpha_k, k = 1, \ldots, n-1$, representing the number of ambulances breaking the nesting condition between consecutive rows of the compliance table. We instead use the single parameter α to indicate the expected number of relocations once a transition happens. A value of 1 indicates that one ambulance relocation is expected at every transition, and a value of 1.5 indicates that on average 1.5 ambulances must be relocated, or that half the relocations would trigger 1 ambulance relocation and the other half would trigger 2 relocations. For a nested solution we set $\alpha = 1$.

Formulation:

$$\text{Maximize:} \sum_{k=0}^{n} \sum_{j \in W} \sum_{i \in V} q_k d_i p_{i,j} y_{i,j,k} \qquad (3.6.12)$$

subject to:

$$y_{i,j,k} \leq x_{j,k} \qquad i \in V, j \in W, k \in \{0,\ldots,n\} \tag{3.6.13}$$

$$\sum_{j \in W} y_{i,j,k} = 1 \qquad k \in \{1,\ldots,n\}, i \in V \tag{3.6.14}$$

$$\sum_{j \in W} x_{j,k} = k \qquad k \in \{0,\ldots,n\} \tag{3.6.15}$$

$$x_{j,k} - x_{j,k+1} \leq u_{j,k} \qquad j \in W, k \in \{0,\ldots,n-1\} \tag{3.6.16}$$

$$\sum_{k=0}^{n-1}((\sum_{j \in W} u_{j,k}) + 1) * m_k \leq \alpha \tag{3.6.17}$$

$$x_{j,k} \in \{0,1\} \qquad j \in W, k \in \{0,\ldots,n\} \tag{3.6.18}$$

$$y_{i,j,k} \in \{0,1\} \qquad i \in V, j \in W, k \in \{0,\ldots,n\} \tag{3.6.19}$$

$$u_{j,k} \in \{0,1\} \qquad j \in W, k \in \{0,\ldots,n-1\} \tag{3.6.20}$$

In this formulation:

- The objective function is an estimate of the actual performance expected of the system under the assumption that the ambulances are always in compliance, which we previously defined as our Model 3. In most cases, the value obtained is an upper bound on the more realistic performance of Model 1, as we have already established by $\mathrm{E}[f^{(1)}(B^{(1)})] \leq \mathrm{E}[f^{(2)}(B^{(2)})] = \mathrm{E}[f^{(3)}(B^{(3)})]$. If we analyze the components of the objective function, $\sum_{i \in V} d_i = 1$, or 100% of the demand. When we multiply by $p_{i,j}$ and $y_{i,j,k}$ for a single k and for all j we are using $y_{i,j,k}$ to select which ambulance is covering demand node i. Because we are maximizing, the solution will automatically assign a value of 1 to the $y_{i,j,k}$ corresponding to the station with the highest probability of reaching that demand node in time (the highest value of $(p_{i,j}|j)$. The result of this is a weighted average of the probability of reaching a call. When we extend this to all values of k it becomes a weighted average over all values of k: this is equivalent to the expected probability of reaching a call on time for the whole system.

- Constraint 3.6.13 indicates that to cover demand node i from station j when we have

k ambulances available ($y_{i,j,k}$), station j must be selected as part of the compliance table for k available ambulances ($x_{j,k} = 1$).

- Constraint 3.6.14 ensures that for every possible row k of the compliance table, every demand node i is covered by one and only one station.
- Constraints 3.6.15 and 3.6.16 are identical to their counterparts in the MECRP formulation.
- Constraint 3.6.17 establishes a limit on the expected number of relocations for a transition.

If we compare the Markov model and the IP formulation proposed in this chapter, we can make several observations:

- The IP formulation requires as an input the probability of each number of available ambulances, while the Markov model computes those probabilities based on the remaining inputs.
- The IP formulation's objective function, when we restrict the model to nested compliance tables, is equivalent to that of Markov Model 3, which assumes that the ambulances are always in compliance.
- The IP formulation's objective function, when we relax the nesting constraint, is equivalent to that of Markov Model 4.
- Theorem 1 shows that $\mathrm{E}[f^{(1)}(B^{(1)})] \leq \mathrm{E}[f^{(3)}(B^{(3)})] \leq \mathrm{E}[f^{(4)}(B^{(4)})]$, so we can conclude that the optimal solution to our IP model with relaxed nested constraints (Model 4) is an upper bound on the optimal solution to our IP model constrained to nested compliance tables (Model 3), which in turn is an upper bound on the optimal solution of our more realistic bidimensional Markov model (Model 1), using nested compliance tables.

When the number of ambulances n is smaller than the number of candidate stations $|W|$ and we request a nested solution, this model can also be used to select which stations should be in operation.

3.6.1 IP-Markov Iteration

As previously mentioned, both our IP formulation and Gendreau's formulation require as input the probability of each possible number of ambulances. In Gendreau's formulation a binomial distribution is used, while we use our Markov Model 1 to estimate the state probabilities with higher precision.

To do this we use a simple heuristic alternating between the Markov model and the IP formulation:

1. We begin with an initial solution (compliance table), and we solve Markov Model 1 to obtain the initial state probabilities and transition rates.
2. We feed the state probabilities and transition rates into the IP formulation and find an optimal compliance-table solution.
3. We feed the new compliance table to Markov Model 1 and recompute the state probabilities.
4. We repeat from step 2 until we find a compliance table obtained previously. In most cases we converge to a single compliance table, but we have observed situations where we iterate repeatedly between two tables.

3.7 Heuristic Solution

The IP formulations mentioned so far assume that the ambulances are always in the locations designated by the compliance table, as in Model 3. Model 1, on the other hand, can be used to estimate performance in a more realistic way that takes into account the performance

when the system is out of compliance. We propose two heuristics to find near optimal-solutions that more closely match the expected real-life performance.

We assume that all the input parameters previously mentioned: road network, ambulance stations, mean times for evaluation and dispatch, chute, on-site and hospital drop-off, are given, except for the compliance table. We require an arbitrary compliance table as a starting point, or ideally a compliance table obtained from our IP formulation, which already places us in the near-optimal region.

We present two heuristics: one for the case where there are no limits on the number of differences between consecutive rows, or the non-nested case, and another that searches for a strictly nested compliance table.

3.7.1 Greedy Heuristic to Optimize Non-Nested Compliance Tables

We implemented a greedy heuristic that attempts to replace every station in every row of the compliance table by each of the unassigned stations.

We start with the already-defined compliance table $X = (X_1, ..., X_n)$, which we now express as a a collection of sets, for convenience in stating the steps of the heuristic. We define $X_k = \{i \in W | x_{i,k} = 1\}$ to be the set of stations included in row k of the compliance table. We also define $N_k(X)$ to be the neighborhood of X that is obtained by changing one of the stations in row k, or, formally,

$$N_k(X) = \left\{ Y : \begin{array}{ll} Y_i = X_i & \text{for } i \in \{1, \ldots, k-1, k+1, \ldots, n\} \\ Y_k = X_k \cup \{j'\} \backslash \{j''\} & j' \ni X_k, j'' \in X_k \end{array} \right\} \quad (3.7.1)$$

We use Model 1 to define the objective function, which is to be maximized:

$$P^{(1)}(X) = \mathrm{E}[f^{(1)}(B^{(1)}(X))] \tag{3.7.2}$$

We now detail the steps of our basic heuristic:

1. We start with a compliance table X^0, and set the current solution $X^* = X^0$.
2. For every row $k \in \{1, \ldots, n-1\}$:
   (a) For every $Y \in N_k(X^*)$, if $P^{(1)}(Y) > P^{(1)}(X^*)$ then set $X^* = Y$.
3. X^* is our final solution.

In Step 2, we tried the following three possibilities for the order in which the rows are processed. Note that the order matters, because the objective function value depends on the entire compliance table, that is, the contribution of row k to the objective function depends on the other rows.

1. Visit the rows from 1 to $(n-1)$.
2. Visit the rows from $(n-1)$ to 1.
3. Identify the row k' with the highest probability, based on using Model 1 to evaluate X^0. We expect states with a higher probability to have a larger impact on system performance. We visit row k' first, then continue down to row 1, then return to $k'+1$ and continue up to row $n-1$.

3.7.2 Greedy Heuristic to Optimize Nested Compliance Tables

We also implemented a heuristic for nested compliance tables. The problem is the same except that if a station has been selected for the case where k ambulances are available, then that station must remain selected when $k+1$ ambulances are available. This heuristic

starts from the row h with the highest probability, based on X^0, and then we iterate toward the extremes. For this heuristic, we define three neighborhoods:

1. We continue to use $N_k(X)$, defined as for the non-nested case, for the initial row. It interchanges assigned stations and unassigned stations.

2. A neighborhood $N_k^F(X)$ is defined for searching forward from $h+1$ to $n-1$. It takes the stations in the previous row and adds a new station, as follows:

$$N_k^F(X) = \left\{ Y : \begin{array}{ll} Y_i = X_i & \text{for } i \in \{1, \ldots, k-1, k+1, \ldots, n\} \\ Y_k = X_{k-1} \cup \{j'\} & j' \ni X_{k-1} \end{array} \right\} \quad (3.7.3)$$

3. Another neighborhood $N_k^B(X)$ is defined for searching backward from h to 1. It takes the stations in the previous row and removes one station, as follows:

$$N_k^B(X) = \left\{ Y : \begin{array}{ll} Y_i = X_i & \text{for } i \in \{1, \ldots, k-1, k+1, \ldots, n\} \\ Y_k = X_{k+1} \backslash \{j''\} & j'' \in X_{k+1} \end{array} \right\} \quad (3.7.4)$$

We now detail the steps of our heuristic:

1. We start with a compliance table X^0, and set the current solution $X^* = X^0$.

2. For row h:

   (a) For every $Y \in N_k(X^*)$, if $P^{(1)}(Y) > P^{(1)}(X^*)$ then set $X^* = Y$.

3. For every row $k \in \{h+1, \ldots, n-1\}$:

   (a) For every $Y \in N_k^F(X^*)$, if $P^{(1)}(Y) > P^{(1)}(X^*)$ then set $X^* = Y$.

4. For every row $k \in \{h-1, \ldots, 1\}$:

   (a) For every $Y \in N_k^B(X^*)$, if $P^{(1)}(Y) > P^{(1)}(X^*)$ then set $X^* = Y$.

5. X^* is our final solution.

We could repeat steps 2-4 multiple times, saving after each iteration the maximum performance obtained, repeating until the performance obtained at the end of the iteration would not provide any improvement. For our numerical experiments we performed steps 2-4 only once.

Because we are gradually enforcing the nesting constraint in each new row, we may see a decrease in performance from one row to the next.

3.8 Results

3.8.1 Base Example

The results presented are for a sample EMS configuration based on empirical data from Edmonton EMS, in Edmonton, Alberta, Canada. We used the actual Edmonton road network and 19 ambulances to be located in 32 candidate ambulance stations, as well as a division of the city into demand regions based on census dissemination areas (DA). We measure the aggregate performance for high priority calls only. We estimate the response-time probability density function as a convolution of the following probability functions:

- The evaluation and dispatch time, which is the time from the moment a call arrives at the dispatch center to the moment that an ambulance is dispatched.
- The chute time, which is the time from the moment that paramedics receive a dispatch notification to the moment that they start driving.
- The travel time, which is the time from the moment that the ambulance starts moving to the moment that it stops near the site of the incident.

The evaluation and dispatch time and the chute time were approximated by fitting log-normal distributions to empirical data. We based the travel times on a travel-time model derived from that proposed by Budge et al. (2010) and slightly modified as proposed in

Alanis et al. (2012). The response-time distribution was later used to estimate the probability that every demand node could be reached from a station in a time threshold τ, defined as nine minutes for our example.

For our Markov model we also required the call arrival rate, and the average on-scene times, transportation times, and hospital times, all approximated from empirical data, as well as the hospital locations.

In order to obtain a baseline for the performance of our models we generated 100 random nested compliance tables and computed their expected performance using Model 1. We obtained an average expected performance of 75.51% of high priority calls answered in under 9 minutes, with a minimum performance of 69.69%, and a maximum performance of 79.17% out of the 100 random tables.

As an example of an empirical solution we ranked the stations according to the expected performance if each station were the only station available, and we obtained an expected performance of 71.56%. This performance is 11.68% lower than the performance we obtain using our optimization approaches, and it is worse than the average performance we obtained from 100 randomly generated compliance tables. This shows the risks of using simple empirical approaches.

3.8.2 Results: IP formulation

We used the IP model to find optimal compliance tables for our example (EMS system for Edmonton) with 32 ambulance stations and 19 ambulances.

The results shown in Table 3.4 were obtained by using the compliance table found by the IP formulation as input for Markov Model 1, to obtain a more reliable estimate. It is worth noting that the improvement gained by relaxing the nesting constraint is small in this case. Also, by using our constraint on the relocation rate we can reduce this rate significantly while maintaining some of the gains achieved by relaxing the nesting constraint.

The performance obtained directly from the objective function of the IP model is a good

Case	Description	Performance (Model 1)	Rate of Relocations
1	Nested $\alpha = 1.00$	82.845%	100.0%
2	Non-nested $\alpha = 1.10$	83.060%	108.8%
3	Non-nested $\alpha = 1.25$	83.154%	125.0%
4	Non-nested $\alpha = 1.50$	83.160%	146.7%
5	Non-nested $\alpha = 10.00$	83.203%	177.8%

Table 3.4: Sample results using IP formulation.

approximation of the result obtained from the more realistic Model 1. For example: for case 1 in Table 3.4 the objective function returns a performance of 83.827%, while using Model 1 we obtain 82.845%.

3.8.3 Results: IP-Markov Iteration

An important issue was the sensitivity of the optimal solution to small variations in the state probabilities. Specifically, by how much would the state probabilities change with different compliance tables, and how sensitive would the optimal solution be to those changes?

We applied the heuristic with different values of α, including fully nested ($\alpha = 1$) and non-nested ($\alpha = 10$). We also tested uniform state probabilities ($\Pr\{B(t) = i\} = \frac{1}{n}$) and the state probabilities obtained from Model 1.

In Table 3.5 we show only the final results, but for cases 2 and 3 we had to go through a series of iterations to achieve these results. During the iterations we observed that the changes in the combined state probabilities resulting from changing the compliance table were small, in the order of 10^{-5} to 10^{-7} in absolute value. However, even small changes in the state probabilities can affect the selection of stations in the final solution. This is not true for the completely non-nested case since in that case the solution for each row is independent of the state probabilities and of the solutions for other rows.

This shows that the use of realistic state probabilities instead of approximate probabilities can lead to a different optimal solution. Via the iterative process, some changes on expected performance were obtained, as measured by Model 1.

Case	Description	Performance (Model 1)	Improvement	% Improvement	Nesting Violations
1	IP/nested/Uniform	82.887%			0
2	IP/nested/Markov probs	82.968%	0.081%	0.098%	0
3	IP/non-nested/Markov probs	83.229%	0.342%	0.413%	25

Table 3.5: Results after iterative process.

Case	Description	Performance (Model 1)	Improvement	% Improvement	Nesting Violations
4	Non-nested/Order:$1-19$	83.400%	0.171%	0.205%	20
5	Non-nested/Order:$11-19, 10-1$	83.411%	0.182%	0.218%	30
6	Non-nested/Order:$19-1$	83.409%	0.180%	0.216%	18
7	Nested	83.237%	0.269%	0.324%	0

Table 3.6: Summary of heuristic results.

3.8.4 Results: Heuristic Solution

We used our heuristics with the inputs for the non-nested case derived from Edmonton EMS. The results are summarized in Tables 3.5 and 3.6.

The scenarios show the performance achieved via the described optimization model; we use the solution obtained for Model 1 to compute the expected performance. The performance value for example 1 was obtained from the IP formulation, since it is the only model capable of supporting uniform state probabilities. The improvement figures for examples 2 and 3 are relative to example 1 to illustrate the effects of using more realistic state probabilities and to show that non-nested compliance tables can provide better performance. The improvement figures for examples 4, 5, and 6 are relative to example 3, because example 3 is the best non-nested solution based on an IP. The improvement for example 7 is relative to example 2 because example 2 is the best nested solution based on an IP.

3.8.5 Results: Observations

From the results we can make the following observations:

1. Relaxing the nesting constraints increases the expected performance, so we could allow

Case	Description	Performance (Model 1)	Improvement	% Improvement	Nesting Violations
1	IP/nested/Uniform	68.482%			0
2	IP/nested/Markov probs	68.481%	−0.001%	−0.001%	0
3	IP/non-nested/Markov probs	68.649%	0.167%	0.244%	29
4	Heur./non-nested/Order:$1-19$	69.314%	0.665%	0.969%	31
5	Heur./non-nested/Order:$5-19, 4-1$	69.373%	0.724%	1.043%	30
6	Heur./non-nested/Order:$19-1$	69.164%	0.515%	0.750%	17
7	Nested Heur.	68.195%	0.039%	0.057%	0

Table 3.7: Summary of results under high load.

a higher frequency of relocations as a way to gain extra performance. However, the gain obtained was small.

2. The use of the heuristic search can improve the solutions obtained from the IP formulations. This is because the IP formulation ignores out-of-compliance behavior, so its objective function is less realistic.

3. The order used to perform the heuristic searches seems to have an effect.

4. It appears that even our simplest method (solving the IP using uniform state probabilities) finds compliance tables that are close to optimal.

To stress test the model, we performed further tests. We used a high-load arrival rate; the probability of a code red (no ambulances available) is approximately 5%. The results are shown in Table 3.7.

Under the high-load scenario we learn or confirm some properties:

- The average difference in performance between the IP formulation and the heuristic approach is larger for the non-nested cases, but is very small for the nested case.

- The three row orderings used for the heuristic has shown better performance for the cases starting from the extremes, in other examples not reported here, we observed the opposite. Therefore we don't have evidence to claim that one ordering is better than the other.

3.9 Conclusions

We proposed a framework for the models under consideration. We introduced a new IP formulation that builds on the work of Gendreau et al. (2006), adding a more realistic calculation of the state probabilities and the iteration between the IP and Model 1. This also allows finer granularity on the control of the relocation frequency. During the testing we showed that the use of realistic state probabilities is important, and that the solutions are sensitive to changes in the state probabilities.

The research also shows that because the IP formulation models in-compliance ambulances, it is only an upper bound on the actual performance. We used a Markov EMS performance model (Alanis et al., 2012) that takes into account out-of-compliance behavior and provides a more realistic performance estimation. This model is used as an objective function in greedy heuristic searches for both nested and non-nested compliance tables, and the results show that further performance improvements can be obtained via these heuristics. The use of the IP formulation to find the starting solution for the heuristic greatly accelerates the search, since we start in near-optimal territory.

CHAPTER 4

Efficient Storage of Transport Network Routes for Simulation Models [1]

4.1 Introduction

A heuristic approach is proposed which, in combination with basic data structures, allows the efficient storage of travel routes from any node to any other node on a road network. Our motivation is the need to implement a simulation model for a fleet of ambulances and the desire to model vehicle travel realistically. This leads to the use of actual road networks and the need to quickly find the best route to follow when an ambulance receives a call.

A simulation model of a fleet of vehicles in a city-sized road network presents challenges derived from the size of the network. We need to model a vehicle fleet, such as ambulances, police cars, fire engines, or taxis, and at any given moment the vehicles can be directed to move to a new destination. To simulate these vehicle moves we need to know the route that the vehicles should follow, and we can potentially have routes from any node to any other node, on a road network that can easily have tens of thousands of nodes. To model this transportation network we must either precompute and store the routes or compute shortest

[1] R. Alanis. Unpublished. 2012.

paths on demand using Dijkstra's or a similar shortest path algorithm. Since we may have to find tens or hundreds of thousands of routes for a single simulation run, performance is of a high priority. We therefore decided that precomputing and storing all the possible routes would produce a fast simulation model, but the price to pay would be high storage requirements.

Many other applications require the real-time computation or retrieval of optimal routes. With an effective tool, mapping and route-planning web services similar to Google Maps could be easily and efficiently implemented. One example is a local vehicle-dispatch system that manages a fleet of vehicles in a city. The methodology in this paper could potentially simplify the development of route planning services for local areas.

We give a detailed description of the problem in Section 4.2, where we define the data structures used to store routing information and the properties that allow compression as well as how we can alter the level of compression. In Section 4.3 we describe how our basic problem can be transformed into a traveling salesman problem (TSP). Section 4.4 reviews some of the most significant developments in the search for shortest routes and the solution of the TSP. Section 4.5 shows how to find optimal solutions for small problems using a well-known TSP formulation or a publicly available TSP solver. Section 4.6 presents a heuristic approach that allows for efficient storage of information and can be used for large road networks. Finally, Section 4.7 summarizes the results obtained and presents conclusions.

4.2 Problem Description

A road network can be defined as a graph $G = (V, A)$ where V is a set of vertices or nodes, with $n = |V|$, and A is a a set of arcs or road segments. There are distances $d_{i,j}$ associated with each arc, where the word distance is used generically, since it may represent the travel cost or travel time. The small nine-node road network shown in Figure 4.1 will be used to illustrate concepts in the rest of this chapter.

For this road network and any two nodes i and j, the shortest route $p_{i,j} = (p_{i,j,0}, p_{i,j,1}, \ldots, p_{i,j,k})$,

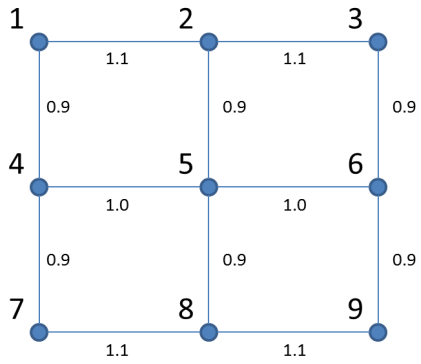


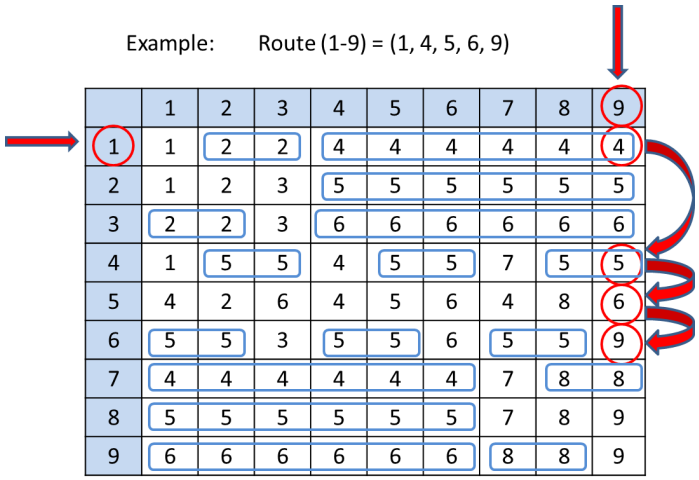
Figure 4.1: Example of Graph or Road Network

where $p_{i,j,0} = i$ and $p_{i,j,k} = j$, is the path for which the sum of the distances along the path is minimized over all possible paths connecting i to j. We assume that the shortest paths have been computed by Dijkstra's or a similar shortest path algorithm.

To represent the shortest paths from any node to any other node we define the matrix $R = \{r_{i,j} = p_{i,j,1}\}$. In this square matrix of dimension $|V| \times |V|$ element $r_{i,j}$ stores the first node after i that would be visited along the shortest path from node i to j. Once at node $l = r_{i,j}$ we proceed to the following node, which is the first node after l on the path from l to j, $r_{l,j}$, and we repeat this until we arrive at $r_{k,j} = j$. For example, in Figure 4.2 we show how to recover the route from node 1 to node 9 for the network in Figure 4.1.

Unfortunately, the road network for a medium to large city can easily have tens of thousands of nodes, and storing a route for all possible pairs of nodes would require storage space of the order of $O(n^2)$. For example, the road network for the city of Edmonton, Alberta, Canada, has 15,227 nodes. This leads to a matrix with 231,861,529 elements of data. This pushes the limit on the amount of main memory allocated to a single program on a personal computer.

Fortunately, there is a simple fact that can be exploited: if you stand on a typical street intersection you can travel in only four possible directions. Therefore, for most intersections or nodes, the corresponding row in our routing matrix R has at most four possible entries, not counting the diagonal entries $r_{i,i} = i$. We observe this in Figure 4.2. In actual road



	1	2	3	4	5	6	7	8	9
1	1	2	2	4	4	4	4	4	4
2	1	2	3	5	5	5	5	5	5
3	2	2	3	6	6	6	6	6	6
4	1	5	5	4	5	5	7	5	5
5	4	2	6	4	5	6	4	8	6
6	5	5	3	5	5	6	5	5	9
7	4	4	4	4	4	4	7	8	8
8	5	5	5	5	5	5	7	8	9
9	6	6	6	6	6	6	8	8	9

Figure 4.2: Routing Matrix

networks, the number of possible directions is not always four, but it is usually small. For four directions, if we alter the order of the columns in R we can obtain four long sequences of repeated values. There are many compression schemes that take advantage of this data replication to store the rows in a more compact way. For example, if there are 10,000 nodes, and the columns are reordered into four sequences of repeated values, they can be stored using four slightly more complex data elements instead of the 10,000 required for the original row.

For example, we could represent our nine-node road network using the routing matrix shown in Figure 4.3, where each data element (a, b) represents a sequence of value a repeated b times. The worst case occurs for row 5 where the number of data elements is 9; for rows $1, 3, 7$, and 9 we require only 3 data elements per row.

Based on this, we define c_i to be the storage cost for row i, equal to the number of sequences of identical values found in row i. Unfortunately, if the columns are reordered to reduce the storage cost for row i, there is no guarantee that this ordering will also minimize the storage cost for other rows.

Based on this explanation, our problem is to find the column ordering in our routing matrix that minimizes the sum of the costs c_i for all rows.

										Cost
1	(1,1)	(2,2)	(4,6)							3
2	(1,1)	(2,1)	(3,1)	(5,6)						4
3	(2,2)	(3,1)	(6,6)							3
4	(1,1)	(5,2)	(4,1)	(5,2)	(7,1)	(5,2)				6
5	(4,1)	(2,1)	(6,1)	(4,1)	(5,1)	(6,1)	(4,1)	(8,1)	(6,1)	9
6	(5,2)	(3,1)	(5,3)	(6,1)	(5,2)	(9,1)				6
7	(4,6)	(7,1)	(8,2)							3
8	(5,6)	(7,1)	(8,1)	(9,1)						4
9	(6,6)	(8,2)	(9,1)							3

Figure 4.3: Compressed Routing Matrix

4.3 Problem Transformation

Since the problem is to find the ordering of the columns that minimizes the total cost, we need a way to compute the cost of having two columns i and j next to each other. By analyzing a single row k we see that if $r_{k,i} = r_{k,j}$ then the existing sequence continues, and the additional cost of having $r_{k,i}$ next to $r_{k,j}$ is 0. On the other hand, if $r_{k,i} \neq r_{k,j}$ then the existing sequence ends at $r_{k,i}$ and a new sequence starts at $r_{k,j}$, which implies the creation of a new data element and therefore an additional storage cost of 1.

To extend the previous analysis to a pair of columns, we define $d^v_{i,j}$ to be the cost of having column i next to column j, such that it is equal to the number of rows for which the two columns have different values. This is also known as the Hamming distance (Hamming, 1950) between the columns, and we will refer to this as the "virtual distance" between columns i and j to distinguish it from physical road-network distances. We represent this by a virtual distance matrix D^v as illustrated in Figure 4.4.

The problem can be transformed by changing our perspective, from a geographical space to a "virtual space" in which the columns in R can be seen as "virtual nodes," with the

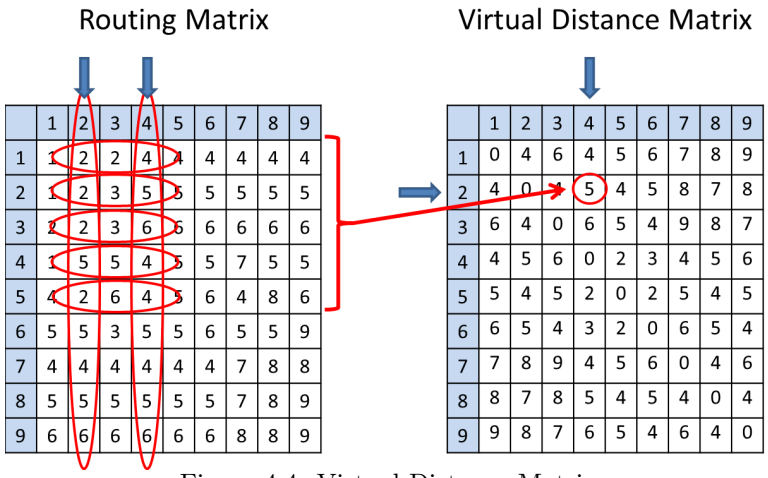


Figure 4.4: Virtual Distance Matrix

distances between each pair of virtual nodes defined by $D^v = \{d^v_{i,j}\}$.

The problem is now to find a route or path visiting all the virtual nodes, without any restriction on which node is first or last, such that all the nodes are visited exactly once and the total virtual distance is minimized.

Finding a route visiting all nodes exactly once is also known as the Hamiltonian path problem. Similarly, a Hamiltonian cycle problem involves finding a path visiting each node exactly once, except that the route must be closed into a cycle. If we start from a Hamiltonian path problem there is a simple transformation that converts it into a Hamiltonian cycle problem: we add a dummy virtual node and connect it to every node in our problem, with all the new arcs having the same very small distance, smaller than that of any existing arc. This ensures that a minimum-distance Hamiltonian cycle is also a minimum-distance Hamiltonian path. The solution to the cycle problem can be transformed into a path by eliminating the dummy node and its associated arcs.

Our problem is a minimum-distance Hamiltonian path problem and it is equivalent (via the transformation mentioned above) to a minimum-distance Hamiltonian cycle problem, or in other words, the well-known traveling salesman problem (TSP), one of the most studied problems in graph theory.

4.4 Literature Review

The current literature is reviewed with a focus on two areas: algorithms to find optimal or minimum-cost paths between two nodes and algorithms for the TSP.

4.4.1 Shortest Path algorithms

Even when for our purpose we assume that shortest paths are a pre-computed input, the basic problem is to provide somehow shortest paths between any pair of nodes, and the solutions, as we will see, go from computation of shortest paths on demand, to different levels of pre-computing and storage of partial routing information, with our approach being at the extreme by storing all possible paths.

Finding an optimal or minimum-cost route between two nodes in a graph is a problem whose solution is widely applied today; however, researchers continue to develop improved algorithms. The classical solution was proposed by Dijkstra (1959). The original algorithm stores tentative distances from the source node to each node in the network. We start at the source node and gradually construct a minimum-distance spanning tree from the source to each node by adding, one by one, directly connected nodes to the tree and revisiting the minimum distance to the source, until we finally include the destination in our minimum spanning tree. The execution time for Dijkstra's original algorithm is of the order $O(n^2)$. However, a continental road map for the US or Western Europe has millions of nodes, so this original algorithm is probably not being used in your typical GPS device.

The improvements on Dijkstra's algorithm can be classified into exact algorithms (those that return an optimal solution) and approximate algorithms, such as those used in some GPS units with limited hardware. We will focus on exact algorithms.

The first improvement involves performing a bidirectional search. Instead of starting at the source and gradually constructing a shortest-distance tree, the search starts simultaneously at the source and at the destination. In the original Dijkstra's algorithm the search forms

an approximately circular shape covering the nodes around the source, until the destination is included. In a bidirectional search, two circles start to grow, one around the source and the other around the destination, until some node is visited from both directions (Dantzig, 1962). The number of nodes visited is reduced and a speed-up of approximately a factor of two is observed.

Another improvement can be achieved by implementing Dijkstra's algorithm using $O(n)$ priority queues, as described by Thorup (2004). With this method we can achieve execution times of $O(n \log(n))$ without significantly altering the idea behind the original algorithm.

Another approach is based on the intuitive idea of giving a higher priority to arcs that lead toward the destination, rather than searching equally in all directions. This is the A* algorithm as proposed by Hart et al. (1968). In this algorithm we estimate a lower bound on the distance from every node to the destination, such as the Euclidean distance. Based on this we modify the weight of an arc (u, v) to $w(u, v) - \pi(u) + \pi(v)$ where $\pi(v)$ is the lower bound on the distance to the destination. With this adjustment we shrink arcs leading toward the destination while preserving the selection of the optimal path.

Another step on the search for efficiency comes from the use of hierarchical approaches and the removal or selection of arcs and nodes to simplify the network. Examples include the work of Jing et al. (1998) and Cagigas (2005). Jing et al. (1998) propose a HiTi graph model where a high graph is divided into smaller subgraphs, leaving at the higher level only those nodes on the boundary of the subgraphs. This can be extended to multiple levels. At each subgraph we compute the shortest paths between all possible pairs of boundary nodes, and we replace all the internal nodes by arcs directly connecting the boundary nodes.

A different approach is reach-based routing (Gutman, 2004), based on the concept of "reach." A reach metric is computed for each node based on the smaller of the distance from that node to the source and the distance to the destination on a critical path. These values can be used to eliminate nodes from consideration in a shortest path algorithm when their reach is too short to reach either the source or the destination. This approach requires significant precomputation.

Another hierarchical approach, highway hierarchies (Sanders and Schultes, 2005), creates a hierarchy of highway levels. It uses the concept of neighborhood to classify the nodes on any optimal path into the neighborhood of the source, the neighborhood of the destination, and the other nodes (called highway nodes). Based on this classification a highway graph is constructed and then simplified by contracting and removing edges to create a higher-level graph in a hierarchical network. This process can be repeated to create a multilevel hierarchy.

Transit node routing is a hierarchical approach based on the observation that for long routes there are important (transit) nodes, such as major intersections that are visited by a large number of shortest-path routes. If we identify a set of transit nodes in a graph, most route searches consist of finding a route to a nearby transit node, then finding a route on a higher-level network of transit nodes to a transit node near the destination, and finally finding a local route on to the destination.

These are just a few examples of recent hierarchical approaches, and some algorithms combine several of the techniques described. Most hierarchical approaches have in common the creation of a hierarchical structure to represent the road networks or graphs, with a distinction between local neighborhoods and higher levels of simplified road networks. We must consider not only the complexity of finding a shortest-path route, but also the complexity of performing precomputation and the space required to store the precomputed results that allow for a faster search.

Our approach has a narrower scope than some of the techniques mentioned because we need a simple approach that can be easily incorporated into a simulation, and some of the techniques above are too complex to be easily implemented. Nevertheless, our efficient storage of precomputed shortest paths could be useful in some of the hierarchical methods.

4.4.2 The Traveling Salesman Problem

The TSP is one of the most widely studied problems in graph theory and combinatorial optimization. Consider a graph $G = (V, A)$ where V is a set of vertices and A is a set of arcs

connecting some of those vertices, with costs or distances defined by a matrix $D = (d_{i,j})$. The TSP is defined as the problem of finding a minimum-distance circuit that visits each vertex once and only once. Two common scenarios include the case where D is symmetric ($d_{i,j} = d_{j,i}$), as in our column-ordering problem, and the case where D satisfies the triangle inequality, i.e., $d_{i,j} + d_{j,k} \geq d_{i,k} \forall i, j, k \in V$. The triangle inequality is a property of graphs on a plane with Euclidean distances, but it does not hold for our column-ordering problem. This is a significant drawback since we cannot use the methods that rely on the assumption of Euclidean distances and the triangle inequality.

The TSP is NP-hard (Laporte, 1992), which suggests that finding an optimal solution might be intractable for our target problem size. One of the earliest approaches was an integer linear programming formulation by Dantzig et al. (1954). This original formulation became the foundation for many subsequent models such as those proposed by Miller et al. (1960) and Desrochers and Laporte (1991). Branch-and-bound approaches have been proposed by Carpaneto and Toth (1980) and Miller and Pekny (1991) to solve large TSPs. Some of these methods, in particular the one proposed by Miller and Pekny (1991), have been able to solve instances with thousands of nodes. However, sometimes the same methods are unable to solve even relatively small problems.

Other researchers have investigated heuristics, achieving good empirical performance. Examples include methods for tour construction based on finding a minimum spanning tree and using it to traverse nodes, e.g., Christofides (1976). Other tour-construction procedures include the nearest-neighbor algorithm, insertion algorithms (Rosenkrantz et al., 1977), and the patching algorithm for asymmetric TSPs (Karp, 1979).

Other heuristics start from an initial tour and try to improve on it. Examples include the r-opt algorithm (Lin, 1965; Lin and Kernighan, 1973), simulated annealing (Kirkpatrick et al., 1983), and tabu search (Glover, 1989, 1990). It is worth mentioning that most tour-improvement techniques are based on extensive trials of possible changes that in most cases have computational times of polynomial order. We investigated tour-improvement techniques, but the size of the data required to describe each possible tour, together with the

polynomial computational times, made this approach too slow for large problems.

As already mentioned, this is by no means an exhaustive survey of all the methods available. In our application exact solutions are not easily found for the large problems that we want to analyze, and heuristic methods based on tour improvements are slow for our problems. We therefore consider tour-construction methods, which have the potential to be faster at the cost of suboptimal solutions.

4.5 Exact Solutions

As a first step we propose the use of the well-known TSP integer programming formulation proposed by Dantzig et al. (1959). Unfortunately, it is limited to small problems, with typically tens to hundreds of nodes, much smaller than the problems that we want to solve.

It is worth mentioning the Concorde TSP Solver, a solver freely available for academic use (http://www.tsp.gatech.edu/concorde.html), which uses some of the most advanced exact and heuristic methods to search for optimal solutions. It is considered the state-of-the-art solver for the TSP problem. Benchmark results are available from the same website for a set of problems, most of which are characterized as being neither easy nor hard. From these it is clear that current TSP algorithms cannot be used in real-life applications to consistently solve problems beyond 10,000 nodes. Therefore, heuristic or approximate algorithms are the only practical approach to our problem.

4.6 Approximate Solutions

There are many heuristic algorithms for the TSP; Laporte (1992) in his review mentions many of them. Before selecting one, we analyze some properties of our problem:

1. We have introduced virtual nodes and presented the problem as a TSP, but in reality we do not have geographically located nodes in a two-dimensional space. Therefore, we

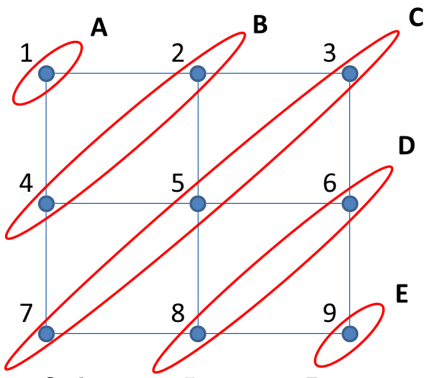


Figure 4.5: Ordering or Layers on Processing of Nodes

must be careful because geometric assumptions, such as Euclidean distances, usually associated with TSP problems can be violated for our virtual distances.

2. A typical road network is sparse, with a low ratio of arcs to nodes. In our virtual space we have a fully connected network, where each virtual node (column) has a direct connection to every other node. This, together with the previous observation, could have an unexpected impact on performance if we attempt to use spanning trees or other heuristics that traverse all possible paths or that take advantage of the geometrical properties of planar networks.

3. If we have two trips that start from the same source but travel to different destinations that are far from the source but close to each other, then there is a high probability that the two routes will initially be identical. As an extreme example consider traveling to two different addresses in a distant town. You are likely to follow the same route for most of the trip, except at the end. Thus, in our routing matrix it would be convenient to have neighboring nodes in adjacent columns.

4. If we could find a Hamiltonian path joining all the original geographical nodes we could satisfy our goal of having neighboring column nodes that correspond to actual geographical neighboring nodes.

Based on the previous observations we decided to implement a heuristic approach loosely based on the family of tour-construction insertion algorithms but taking into account the properties above. The detailed heuristic is as follows:

1. Choose arbitrarily a single geographical node as our starting solution. In Figure 4.5 we could select node 1 at stage A.

2. Define the set of geographical nodes to be added at the next iteration by choosing all the nodes not already in the solution but directly connected to nodes in the solution. We refer to this as the next stage or the next layer to add. In Figure 4.5 the stages are A, B, C, D, and E.

3. Take a geographical node in the new layer but not already added. For demonstration purposes we proceed from left to right.

4. Search in the current solution for the geographical nodes directly connected to the node under consideration and consider adding the corresponding virtual node or column in a position adjacent to any of the directly connected geographical nodes. We insert the new virtual node in such a way that the total cost after the insertion is minimized. If we add node k between nodes i and j, then the cost of the insertion is $(d^v_{i,k}+d^v_{k,j}-d^v_{i,j})$.

5. Repeat the last two steps until all the nodes in the new layer have been added.

6. If there are still nodes that are not in the solution, return to step 2.

Table 4.1 illustrates the sequence of steps required to solve our nine-node example.

With the use of efficient data structures this algorithm is of order $O(n)$ based on the number of operations, since we insert the nodes one by one, and for each node we analyze at most four possible insertion points.

The solution found and displayed in Figure 4.6 is, in this particular case, a Hamiltonian path, which is consistent with the intuition indicating that directly connected nodes should have low "virtual distances." It therefore leads to the intuition that in an optimal solution

Layer	Node to Add	Alternative Paths	Cost of Addition	Selected
A	1	(1)	$n = 9$	(1)
B	4	(4-1) (1-4)	$d^v_{4,1} = 4$ $d^v_{1,4} = 4$	(4-1)
	2	(4-2-1) (4-1-2)	$d^v_{4,2} + d^v_{2,1} - d^v_{4,1} = 5$ $d^v_{1,2} = 4$	(4-1-2)
C	7	(7-4-1-2) (4-7-1-2)	$d^v_{7,4} = 4$ $d^v_{4,7} + d^v_{7,1} - d^v_{4,1} = 7$	(7-4-1-2)
	5	(7-5-4-1-2) (7-4-5-1-2) (7-4-1-5-2) (7-4-1-2-5)	$d^v_{7,5} + d^v_{5,4} - d^v_{7,4} = 3$ $d^v_{4,5} + d^v_{5,1} - d^v_{4,1} = 3$ $d^v_{1,5} + d^v_{5,2} - d^v_{1,2} = 5$ $d^v_{2,5} = 4$	(7-5-4-1-2)
	3	(7-5-4-1-3-2) (7-5-4-1-2-3)	$d^v_{1,3} + d^v_{3,2} - d^v_{1,2} = 6$ $d^v_{2,3} = 4$	(7-5-4-1-2-3)
D	8	(8-7-5-4-1-2-3) (7-8-5-4-1-2-3) (7-5-8-4-1-2-3)	$d^v_{8,7} = 4$ $d^v_{7,8} + d^v_{8,5} - d^v_{7,5} = 3$ $d^v_{5,8} + d^v_{8,4} - d^v_{5,4} = 7$	(7-8-5-4-1-2-3)
	6	(7-8-6-5-4-1-2-3) (7-8-5-6-4-1-2-3) (7-8-5-4-1-2-6-3) (7-8-5-4-1-2-3-6)	$d^v_{8,6} + d^v_{6,5} - d^v_{8,5} = 3$ $d^v_{5,6} + d^v_{6,4} - d^v_{5,4} = 3$ $d^v_{2,6} + d^v_{6,3} - d^v_{2,3} = 5$ $d^v_{3,6} = 4$	(7-8-6-5-4-1-2-3)
E	9	(7-9-8-6-5-4-1-2-3) (7-8-9-6-5-4-1-2-3) (7-8-6-9-5-4-1-2-3)	$d^v_{7,9} + d^v_{9,8} - d^v_{7,8} = 6$ $d^v_{8,9} + d^v_{9,6} - d^v_{8,6} = 3$ $d^v_{6,9} + d^v_{9,5} - d^v_{6,5} = 7$	(7-8-9-6-5-4-1-2-3)

Table 4.1: Heuristic Solution Process.

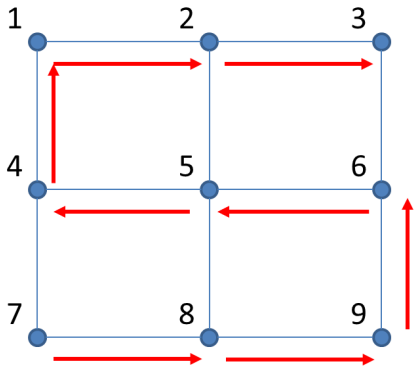


Figure 4.6: Solution Obtained

	1	2	3	4	5	6	7	8	9	10
1	0	4	6	4	5	6	7	8	9	1
2	4	0	4	5	4	5	8	7	8	1
3	6	4	0	6	5	4	9	8	7	1
4	4	5	6	0	2	3	4	5	6	1
5	5	4	5	2	0	2	5	4	5	1
6	6	5	4	3	2	0	6	5	4	1
7	7	8	9	4	5	6	0	4	6	1
8	8	7	8	5	4	5	4	0	4	1
9	9	8	7	6	5	4	6	4	0	1
10	1	1	1	1	1	1	1	1	1	0

Figure 4.7: Modified Virtual Distances for TSP

adjacent columns are likely to correspond to geographical nodes with a direct connection, resulting in the formation of geographic paths. In other examples we have observed optimal solutions composed of several interconnected geographic paths, which shows that the optimal solution is not always a Hamiltonian path in the geographical network.

We used the same data to formulate a TSP. We added the dummy node 10 to our virtual distance matrix as shown in Figure 4.7, and since the shortest node-to-node distance is 2, we assigned a distance of 1 from node 10 to every other node. We solved the problem using the NEOS Concorde Online Solver (http://www.neos-server.org), and we obtained the solution shown in Table 4.2, which matches our heuristic solution.

Because the solution represents a circuit, and node 10 is a dummy, we can remove node 10 and reconnect the start and the end to form a single path: $(7-8-9-6-5-4-1-2-3)$. The total storage cost can be computed via

- The number of rows, 9, for the initial cost of the first sequence in each row.
- Plus the cost of the TSP solution, 30, for each time that a sequence was broken and a new data element was required.
- Minus the cost of connecting the dummy node, $(1+1) = 2$.

From	To	Cost
1	2	4
2	3	4
3	10	1
10	7	1
7	8	4
8	9	4
9	6	4
6	5	2
5	4	2
4	1	4

Table 4.2: TSP Solution.

Therefore, the total number of data elements is 37, compared to the initial cost of 81 elements in the original uncompressed routing matrix. The compression rate usually becomes more significant as the number of nodes grows, as we will see in Section 4.7.

4.7 Results and Conclusions

We created two versions of a 100-node road network, a 9 block by 9 block perfect grid (10×10 nodes). In one version we created an artificially regular routing pattern, such that to travel between any two nodes the rule is to first travel horizontally and then vertically. In the second version we allowed randomness in the length of each block, so that each arc had a base length of 10 plus or minus 1, with an equal probability of $+1$ or -1. This resulted in optimal routes that were less predictable. In both cases the solution was obtained by our heuristic in 0.11 s, while the Concorde NEOS Server took 0.13 s for the regular case and 1.39 s for the random case. Both methods found the optimal cost of 460 data elements for the regular case. For the random case the heuristic method found a cost of $1,058$ elements while Concorde found an optimal cost of 905 elements. Given the original size of the routing matrix (10,000 elements) both solutions provide excellent results: the optimal TSP-based solution requires 9.05% of the original storage requirement while the heuristic approach requires 10.58%. From this experiment we learned that:

1. The level of compression depends on other characteristics of the road network in

Nodes	Original Size	Heuristic			Concorde/IP		
		Cost	Time (s)	Ratio	Cost	Time (s)	Ratio
9	81	37	0.02	45.68%	37	< 0.01	45.68%
12	144	52	0.02	36.11%	52	< 0.01	36.11%
100 Reg	10,000	460	0.11	4.6%	460	0.13	4.6%
100 Rnd	10,000	1,058	0.11	10.58%	905	1.39	10.37%
1,936 Reg	3,748,096	9,504	2.09	0.25%	9,504	4.23	0.25%
2,025 Reg	4,100,625	9,945	2.57	0.24%			
4,900 Reg	24,010,000	24,220	18.01	0.10%			
15,227 Real	231,861,529	574,993	187.68	0.25%			

Table 4.3: Summary of Performance Results.

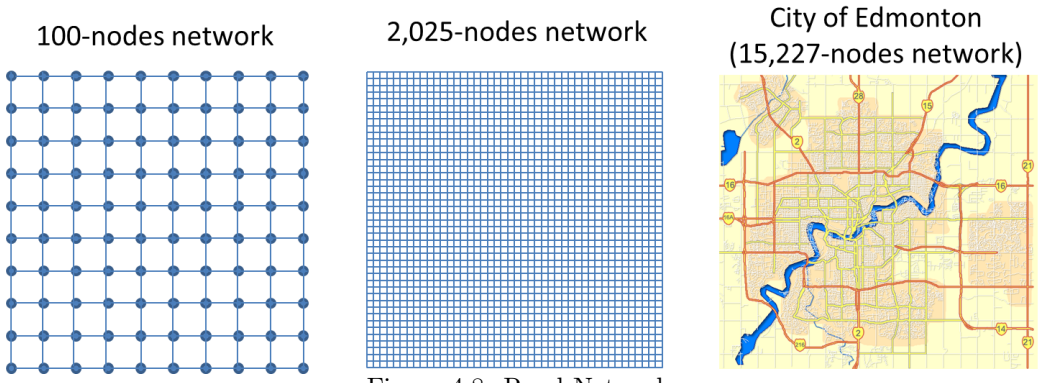


Figure 4.8: Road Networks

addition to its size.

2. The heuristic solution, as expected, can be faster and provide good results.

3. The gains in compression seem to increase with the size of the road network.

The results for different networks are summarized in Table 4.3.

We used eight different road networks to test our models. The networks had 9, 12, 100, 1,936, 2,025, 4,900, and 15,227 nodes. We display 3 of the networks in Figure 4.8. All the networks are artificial, except for the last one, which corresponds to the road network for the city of Edmonton, Alberta, Canada.

The first solution method attempted was the Dantzig et al. (1959) IP formulation already discussed, and it was tested on the models with 9 and 12 nodes. It obtained an optimal solution, but it is suitable only for very small networks. On larger networks (22 nodes) it failed to find an integer solution in 2.5 h. It was deemed inadequate for real road networks. The use of Concorde allows the solution of larger problems. A version of Concorde is available for personal computers, but it is limited to Euclidean distances, which makes it unsuitable for our application. Another version is available online at the NEOS Server (http://www.neos-server.org/neos/), and it supports symmetric distance matrices. This service has a limit on the size of uploaded files that limited our tests to an approximate maximum of 1,936 nodes. This is roughly equivalent to a grid of 43 blocks by 43 blocks. Given the time to solve the problem (4.23 s) we expect that the Concorde solver could deal with larger problems.

We tested our insertion method on the networks with 9, 12, 100, 1,936, 2,025, 4,900, and 15,227 nodes. For 9, 12, 100, and 1,936 nodes we found the optimal solution. For larger problems we found solutions with high compression ratios but we do not know if they are optimal. It is worth mentioning that our computer-generated networks are highly regular. On tests with 100 nodes and randomness in the length of the arcs, and therefore with less predictable routing, our heuristic achieved significant compression but was not able to find an optimal solution. The storage size required was 17% higher than that of the optimal solution, and the compression rate achieved was within 0.21 percentage points of that achieved by the optimal solution.

Our experiments show that the heuristic can handle realistically sized networks in a relatively short time. The level of compression achieved in such networks is impressive: we observed a reduction from 231,861,529 data elements to just 574,864 for the network with 15,227 nodes. This represents a compression of 99.75% and makes storing the full routing information entirely feasible and not demanding at all for a city the size of Edmonton. We used these data in a simulation of the Edmonton EMS system, and the amount of data was easily handled. We achieved fast performance for both the data preparation and the simulation

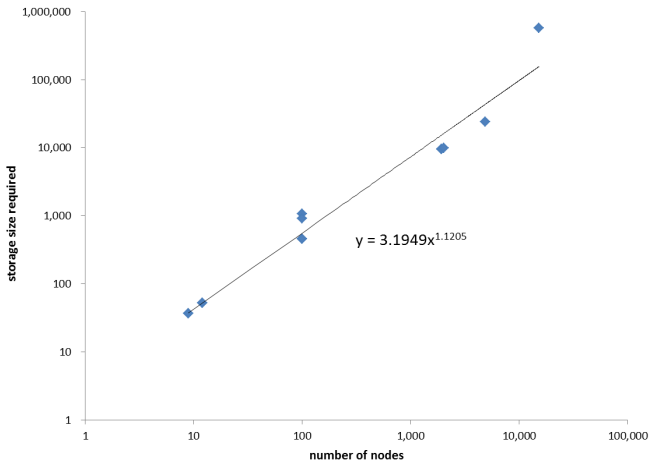


Figure 4.9: Storage Requirements vs Network Size

itself.

The complexity of the storage requirements is not easy to estimate, mostly because, as seen in the two 100-node examples, there is a wide variation depending on the characteristics of the road network. We approximate the complexity by plotting the number of nodes against the observed number of data elements required to store the compressed routing matrix. The plot is shown in Figure 4.9; both axes are transformed under a log_{10} transformation, and the plot displays a power trend line that corresponds to the equation $y = 3.1949n^{1.1205}$.

This storage formula implies that for a continental-size network of 4,000,000 nodes we could require $79,811,503$ data elements. When translated into data structures this indicates that every route in a whole continent could easily fit into a small flash drive or a small portion of the memory available on a smartphone.

To conclude:

- We have developed a simple heuristic that allows a very significant reduction in the storage requirements for routing information for a road network.
- The complexity of the heuristic is linear in the number of operations.

- The level of compression achieved increases with the size of the road network. Empirically, we estimate that it is of order $O(n^{1.1205})$.

CHAPTER 5

Concluding Remarks

The initial goals of this research were to improve the understanding of EMS dynamic relocation policies based on the use of compliance tables, and to provide ways to evaluate the performance of an EMS system. We also planned to provide methods to improve performance through the search for near-optimal compliance tables. Chapters 2 and 3 have provided a substantial discussion of many aspects of EMS systems and dynamic relocation policies. They also discuss system status management implemented through the use of compliance tables, as well as previous research that has led to the current state of operations research for EMS.

Chapter 2 presented a bidimensional Markov model to estimate the performance of an EMS system. This is a tractable analytical model that has the same data requirements and can produce the same outputs as the Hypercube model, but also models repositioning policies. The model was validated against a realistic simulation model and found to provide good approximations of several performance measures in a significantly shorter time, making it a good alternative to simulation.

The Markov model was also shown to provide performance rankings for different compliance tables that were closely correlated with the rankings obtained via the simulation model, which makes it useful in the search for near-optimal compliance tables. We also showed

that by assuming that the ambulances are always in compliance we achieve a simpler model that is easily solved, at the cost of some predictive accuracy in the results.

We extended previous research in Chapter 3 by studying the search for optimal or near-optimal compliance tables. We defined a framework to classify models according to whether they assume that the ambulances are always in compliance or allow for out-of-compliance performance. We also classified models depending on whether they use nested or non-nested compliance tables, and we established stochastic order relations between the different models.

Based on the framework previously defined for EMS models, we presented an IP formulation under the assumption that the ambulances are always in compliance, based on realistic state probabilities and transition rates obtained from our bidimensional Markov model. This is different from similar existing models in that it constrains the expected frequency of relocations rather than the level of nesting in the compliance table. This allows finer control of the frequency of relocations by focusing on the results rather than the structure of the nested table.

The next step was to use the full EMS Markov model in combination with heuristic search approaches to find near-optimal compliance tables, for both nested and non-nested tables. We showed that by taking into account the out-of-compliance behavior our heuristics were able to find solutions that improve on the optimal solutions found by the IP formulation.

In the third paper, presented in Chapter 4, we analyzed the problem of storing routes from any node to any other node in a road network, and we found structural properties that allowed us to store those routes in a compressed format that reduces the storage requirements. We identified properties that allowed us to transform the problem into a traveling salesman problem (TSP), and we showed that optimal solutions can be found for networks of up to a few thousand nodes using publicly available state-of-the-art TSP solvers. An insertion-heuristic approach was also presented that allowed us to find near-optimal solutions for large road networks quickly, achieving compression ratios of up to 99.75%. The approaches were tested with road networks from 9 to $15,227$ nodes, and the

resulting compressed routing information was successfully employed in the implementation of a discrete event simulation for the City of Edmonton EMS system. It required only a small amount of memory while providing very fast retrieval of the routes.

Bibliography

Ahn, Ki Ok, Sang Do Shin, Won Chul Cha, Chulmin Jun, Tae Sik Lee, Ronald G. Pirrallo. 2010. A model for the association of the call volume and the unavailable-for-response interval on the delayed ambulance response for out-of-hospital cardiac arrest using a geographic information system. *Prehospital Emergency Care* **14**(4) 469–476. doi:10.3109/10903127.2010.497895.

Alanis, R., A. Ingolfsson, B. Kolfal. 2012. A Markov chain model for an EMS system with repositioning. *Production and Operations Management* [forthcoming].

Alberta Health Services. 2010. Alberta Health Services launches overcapacity protocols. News Release. http://www.albertahealthservices.ca/3376.asp#, accessed 13 August 2011.

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

Berman, O. 1981b. Repositioning of distinguishable urban service units on networks. *Computers & Operations Research* **8**(2) 105–118.

Bhaskaran, B. G. 1986. Almost sure comparison of birth and death processes with application to M/M/s queueing systems. *Queueing Systems* **1** 103–127.

Bledsoe, B. E. 2003. EMS mythology. EMS myth #7. System status management (SSM) lowers response times and enhances patient care. *Emergency Medical Services* **32**(7) 158–167.

Budge, S., A. Ingolfsson, E. Erkut. 2009. Technical note–Approximating vehicle dispatch probabilities for emergency service systems with location-specific service times and multiple units per location. *Operations Research* **57**(1) 251–255.

Budge, S., A. Ingolfsson, D. Zerom. 2010. Empirical analysis of ambulance travel times: The case of Calgary Emergency Medical Services. *Management Science* **56**(4) 716–723.

Cady, G. 2002. 2001 JEMS 200 city survey, JEMS 2001 annual report on EMS operational & clinical trends in large, urban areas. *JEMS: Journal of Emergency Medical Services* **27**(2) 46–71.

Cagigas, Daniel. 2005. Hierarchical d* algorithm with materialization of costs for robot path planning. *Robotics and Autonomous Systems,* **52**(2–3) 190–208.

Carpaneto, G., P. Toth. 1980. Some new branching and bounding criteria for the asymmetric travelling salesman problem. *Management Science* **26**(7) 736–743.

Christofides, N. 1976. Worst-case analysis of a new heuristic for the travelling salesman problem. Research report, Carnegie-Mellon University, Management Sciences Research Group, Pittsburgh, PA.

Church, R. L., C. S. ReVelle. 1974. The maximal covering location problem. *Papers of the Regional Science Association* **32** 101–118.

Dantzig, G. B. 1962. *Linear Programming and Extensions*. Princeton University Press, Princeton.

Dantzig, G. B., D. R. Fulkerson, S. M. Johnson. 1959. On a linear-programming, combinatorial approach to the traveling-salesman problem. *Operations Research* **7**(1) 58–66.

Dantzig, G. B., R. Fulkerson, S. Johnson. 1954. Solution of a large-scale traveling-salesman problem. *Journal of the Operations Research Society of America* **2**(4) 393–410.

Daskin, M. S. 1983. A maximum expected covering location model: Formulation, properties and heuristic solution. *Transportation Science* **17**(1) 48–70.

Daskin, M. S. 1987. *Spatial analysis and location-allocation models*, chap. Location, dispatching, and routing models for emergency services with stochastic travel times. Van Nostrand Reinhold Company, New York, 224–265.

Daskin, M. S., E. H. Stern. 1981. A hierarchical objective set covering model for emergency medical service vehicle deployment. *Transportation Science* **15**(2) 137–152.

Desrochers, M., G. Laporte. 1991. Improvements and extensions to the Miller-Tucker-Zemlin subtour elimination constraints. *Operations Research Letters* **10**(1) 27–36.

Dijkstra, E. W. 1959. A note on two problems in connexion with graphs. *Numerische Mathematik* **1** 269–271.

Erkut, E., A. Ingolfsson, S. Budge, D. Haight, J. Litchfield, O. Akyol, G. Holmes, J. Cheng. 2005. Final report: The impact of ambulance system status management. Unpublished report, prepared for the Emergency Response Department, City of Edmonton.

Erkut, E., A. Ingolfsson, G. Erdogan. 2008. Ambulance location for maximum survival. *Naval Research Logistics (NRL)* **55**(1) 42–58. doi:10.1002/nav.20267.

Fitch, J. 2005. Response times: Myths, measurement & management. *JEMS: Journal of Emergency Medical Services* **30**(1) 46–56.

Gendreau, M., G. Laporte, F. Semet. 1997. Solving an ambulance location model by tabu search. *Location Science* **5**(2) 75–88. doi:DOI:10.1016/S0966-8349(97)00015-6. URL http://www.sciencedirect.com/science/article/pii/S0966834997000156.

Gendreau, M., G. Laporte, F. Semet. 2001. A dynamic model and parallel tabu search heuristic for real-time ambulance relocation. *Parallel Computing* **27**(12) 1641–1653.

Gendreau, M., G. Laporte, F. Semet. 2006. The maximal expected coverage relocation problem for emergency vehicles. *Journal of the Operational Research Society* **57**(1) 22–28.

Glover, F. 1989. Tabu search– Part i. *ORSA Journal on Computing* **1**(3) 190–206.

Glover, F. 1990. Tabu search– Part ii. *ORSA Journal on Computing* **2**(1) 4–32.

Green, L. V., P. J. Kolesar. 2004. Improving emergency responsiveness with management science. *Management Science* **50**(8) 1001–1014.

Gross, D., C. M. Harris. 1998. *Fundamentals of Queueing Theory*. Wiley, New York.

Gutman, R. J. 2004. Reach-based routing: A new approach to shortest path algorithms optimized for road networks. *ALENEX/ANALC'04*. 100–111.

Hamming, R. W. 1950. Error detection and error correction codes. *Bell System Technical Journal* **29**(2) 147–160. MR0035935.

Hart, P. E., N. J. Nilsson, B. Raphael. 1968. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics* **4**(2) 100–107.

Hogan, K., C. S. ReVelle. 1986. Concepts and applications of backup coverage. *Management Science* **34** 1434–1444.

Jarvis, J. P. 1985. Approximating the equilibrium behavior of multi-server loss systems. *Management Science* **31**(2) 235–239.

Jing, N., Y. W. Huang, E. A. Rundensteiner. 1998. Hierarchical encoded path views for path query processing: An optimal model and its performance evaluation. *IEEE Transactions on Knowledge and Data Engineering* **10**(3) 409–432.

Karp, R. M. 1979. A patching algorithm for the nonsymmetric traveling-salesman problem. *SIAM Journal on Computing* **8**(4) 561–573.

Kirkpatrick, S., C. D. Gelatt, M. P. Vecchi. 1983. Optimization by simulated annealing. *Science* **220**(4598) 671–680.

Kolesar, P. 1975. Model for predicting average fire engine travel times. *Operations Research* **23**(4) 603–613.

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

Laporte, G. 1992. The traveling salesman problem - An overview of exact and approximate algorithms. *European Journal of Operational Research* **59**(2) 231–247.

Larson, R. C. 1974. A hypercube queuing model for facility location and redistricting in urban emergency services. *Computers & Operations Research* **1**(1) 67–95.

Larson, R. C. 1975. Approximating the performance of urban emergency service systems. *Operations Research* **23**(5) 845–868.

L'Ecuyer, P. 2009. *SSJ: Stochastic Simulation in Java*. Département d'Informatique et de Recherche Opérationnelle, Université de Montréal, Montréal, Québec. URL http://www.iro.umontreal.ca/~simardr/ssj/indexe.html. Last accessed 19 December 2009.

Lin, S. 1965. Computer solutions of the traveling salesman problem. *Bell System Technical Journal* **44** 2245–2269.

Lin, S., B. W. Kernighan. 1973. An effective heuristic algorithm for the traveling-salesman problem. *Operations Research* **21**(2) 498–516.

Maxwell, M. S., S. G. Henderson, H. 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*. 1850–1860.

Maxwell, M. S., M. Restrepo, S. G. Henderson, H. Topaloglu. 2010. Approximate dynamic programming for ambulance redeployment. *INFORMS Journal on Computing* **22**(2) 266–281.

Miller, C. E., A. W. Tucker, R. A. Zemlin. 1960. Integer programming formulation of traveling salesman problems. *Journal of the ACM* **7** 326–329.

Miller, D. L., J. F. Pekny. 1991. Exact solution of large asymmetric traveling salesman problems. *Science* **251**(4995) 754–761.

Morneau, P. M., J. P. Stothart. 1999. My aching back. The effects of system status management & ambulance design on EMS personnel. *JEMS: Journal of Emergency Medical Services* **24**(8) 36–50, 78–81.

Muller, A., D. Stoyan. 2002. *Comparison Methods for Stochastic Models and Risks*. 3rd ed. Wiley, New York.

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

Rajagopalan, H. J., C. Saydam, J. Xiao. 2008. A multiperiod set covering location model for dynamic redeployment of ambulances. *Computers & Operations Research* **35**(3) 814–826.

Rosenkrantz, D. J., R. E. Stearns, P. M. Lewis II. 1977. An analysis of several heuristics for the traveling salesman problem. *SIAM Journal on Computing* **6**(3) 563–581.

Ross, S. 1996. *Stochastic Processes*. 2nd ed. Wiley, New York.

Sanders, P., D. Schultes. 2005. Highway hierarchies hasten exact shortest path queries. *ESA'05*. 568–579.

Segal, W., V. Verter, A. Colacone, M. Afilalo. 2006. The inhospital interval: a description of EMT time spent in the emergency department. *Prehospital Emergency Care* **10** 378–382.

Sinnema, J. 2010. ER waits keep paramedics away from calls: Average 12 minutes longer in hallways waiting for patients to be seen. *Edmonton Journal* 28 October.

Stout, J. 1989. System status management. The fact is, its everywhere. *JEMS: Journal of Emergency Medical Services* **14** 65–71.

Thorup, M. 2004. Integer priority queues with decrease key in constant time and the single source shortest paths problem. *Journal of Computer and System Sciences* **69**(3) 330–353. Special Issue on STOC 2003.

Toregas, C., R. Swain, C. ReVelle, L. Bergman. 1971. The location of emergency service facilities. *Operations Research* **19**(6) 1363–1373.

Vandeventer, Steve, Jonathan R. Studnek, John S. Garrett, Steven R. Ward, Kevin Staley, Tom Blackwell. 2011. The association between ambulance hospital turnaround times and

patient acuity, destination hospital, and time of day. *Prehospital Emergency Care* **15**(3) 366–370. doi:10.3109/10903127.2011.561412.

Williams, D. M. 2009. JEMS 2008 200 city survey: The future is your choice. *JEMS: Journal of Emergency Medical Services* **34**(2) 36–51.

Wu, J., N. B. Mehta, J. Zhang. 2005. Flexible lognormal sum approximation method. *Global Telecommunications Conference, 2005. GLOBECOM '05. IEEE*, vol. 6. 3413–3417. doi: 10.1109/GLOCOM.2005.1578407.

Zhang, L., A. Mason, A. Philpott. 2009. Optimization of single ambulance move up. *44th Annual Conference of the Operational Research Soceity of New Zealand, Christchurch, New Zealand*. 225–226.

Zhang, L., A. Mason, A. Philpott. 2011. Optimal repositioning of a single ambulance: Models and insights. Submitted.

APPENDIX A

Detailed Proof that $B^{(1)} \geq_{\text{st}} B^{(2)}$

We will use the terms the terms "birth rate" and "death rate" for the transition rates at which $B(t)$ increases by one and decreases by one, although the process $(B(t), C(t))$ is not a birth-death process (because it has two state variables).

The birth and death rates for Models 1 and 2 are as follows:

- Birth Rates

$$\lambda_b^{(1)} = \lambda_b^{(2)} = \begin{cases} \lambda, & 0 \leq b < n \\ 0, & b = n \end{cases} . \tag{A.0.1}$$

- Death Rates

$$\mu_{b,c}^{(1)} = \begin{cases} b\mu_{b,0}, & 0 \leq b \leq n, c = 0 \\ b\mu_{b,1}, & 0 \leq b \leq n, c = 1 \end{cases} \tag{A.0.2}$$

$$\mu_{b,c}^{(2)} = b\mu_{b,1}, \quad 0 \leq b \leq n, c = 0, 1, \tag{A.0.3}$$

where $\mu_{b,1} \geq \mu_{b,0}$.

Assuming that both models start with no busy ambulances at $t = 0$, we want to show the following stochastic ordering between the number of busy ambulances in Model 1, $B^{(1)}(t)$, and the number of busy ambulances in Model 2, $B^{(2)}(t)$:

$$B^{(1)}(t) \geq_{st} B^{(2)}(t), \quad \forall t \geq 0. \tag{A.0.4}$$

We construct two processes $\tilde{B}^{(1)}(t)$ and $\tilde{B}^{(2)}(t)$ that are equivalent to $B^{(1)}(t)$ and $B^{(2)}(t)$ (respectively) and such that $\tilde{B}^{(1)}(t) \geq \tilde{B}^{(2)}(t)$ with probability 1. We start with $\tilde{B}^{(1)}(0) = \tilde{B}^{(2)}(0) = 0$ and let both processes run freely (according to the birth and death rates in Equations A.1.1–A.1.3) until the first time t' where $\tilde{B}^{(1)}(t') = \tilde{B}^{(2)}(t') = b' > 0$. We construct the time until the next service completion in process $\tilde{B}^{(1)}(t)$ after time t', $d^{(1)}(t')$, as follows:

$$d^{(1)}(t') \sim= \begin{cases} \exp\left\{b'\mu_{b',0}\right\}, & \text{if } C(t') = 0 \\ \exp\left\{b'\mu_{b',1}\right\}, & \text{if } C(t') = 1 \end{cases} \tag{A.0.5}$$

If at $t = t'$ Model 1 is not in compliance ($C(t') = 0$), we construct the time until the next service completion after t' in process $\tilde{B}^{(2)}(t)$, $d^{(2)}(t')$, from $d^{(1)}(t')$ as follows:

$$d^{(2)}(t') = \min\left\{d^{(1)}(t'), \tau\right\}, \tag{A.0.6}$$

where $\tau \sim \exp\left\{b'\mu_{b',1} - b'\mu_{b',0}\right\}$.

If at $t = t'$ Model 1 is in compliance ($C(t') = 1$), we set the time until the next service completion after t' in process $\tilde{B}^{(2)}(t)$, $d^{(2)}(t')$, to be equal to $d^{(1)}(t')$.

This construction guarantees that $d^{(2)}(t') \leq d^{(1)}(t')$ and that $d^{(1)}(t')$ and $d^{(2)}(t')$ follow the appropriate distributions (with $d^{(2)}(t') \sim \exp\left\{b'\mu_{b',1}\right\}$).

We also construct the time until the next arrival in both processes after t' $(a(t'))$ to be identical, following an exponential distribution with parameter $\lambda_{b'}^{(1)} = \lambda_{b'}^{(2)}$.

The time until the first arrival or departure in either process after t' is $m = \min\left\{a(t'), d^{(2)}(t')\right\}$. We know that $\tilde{B}^{(1)}(t) = \tilde{B}^{(2)}(t) = b'$, for $t' \leq t < t' + m$. At $t = t' + m$, we have:

$$\tilde{B}^{1}(t'+m) = \begin{cases} b'+1, & m = a(t') \\ b', & m = d^{(2)}(t') < d^{(1)}(t') \\ b'-1, & m = d^{(2)}(t') = d^{(1)}(t') \end{cases}, \tag{A.0.7}$$

and

$$\tilde{B}^{(2)}(t'+m) = \begin{cases} b'+1, & m = a(t') \\ b'-1, & m = d^{(2)}(t') \leq d^{(1)}(t') \end{cases}. \tag{A.0.8}$$

It follows that $\tilde{B}^{(1)}(t) \geq \tilde{B}^{(2)}(t)$, for $0 \leq t \leq t' + m$. If we repeat this construction whenever $\tilde{B}^{(1)}(t) = \tilde{B}^{(2)}(t) > 0$, we have that $\tilde{B}^{(1)}(t) \geq \tilde{B}^{(2)}(t)$ with probability 1, for all t. Since the interarrival and interdeparture times in $\tilde{B}^{(1)}(t)$ and $\tilde{B}^{(2)}(t)$ follow the same distributions as in $B^{(1)}(t)$ and $B^{(2)}(t)$ (respectively), we have that

$$B^{(1)}(t) \geq_{st} B^{(2)}(t), \quad \forall t. \tag{A.0.9}$$

APPENDIX B

List of Acronyms

CAD: Computer-aided dispatch

DA: Census dissemination area

EMS: Emergency Medical Services

GPS: Global Positioning System

HQM: Hypercube queueing model

ILP: Integer linear program

IP: Integer program

LSCM: Location set covering model

MCLP: Maximal covering location problem

MECRP: Maximal expected coverage relocation problem

MEXCLP: Maximum expected covering location problem

RAH: Royal Alexandra Hospital

TSP: Traveling salesman problem