CIRRELT

Centre interuniversitaire de recherche
sur les réseaux d'entreprise, la logistique et le transport

Interuniversity Research Centre
on Enterprise Networks, Logistics and Transportation

Performance Approximation of Emergency Service Systems with Priorities and Partial Backups

Akbar Karimi
Michel Gendreau
Vedat Verter

August 2017

CIRRELT-2017-49

Bureaux de Montréal :
Université de Montréal
Pavillon André-Aisenstadt
C.P. 6128, succursale Centre-ville
Montréal (Québec)
Canada H3C 3J7
Téléphone : 514 343-7575
Télécopie : 514 343-7121

Bureaux de Québec :
Université Laval
Pavillon Palasis-Prince
2325, de la Terrasse, bureau 2642
Québec (Québec)
Canada G1V 0A6
Téléphone : 418 656-2073
Télécopie : 418 656-2624

www.cirrelt.ca

Performance Approximation of Emergency Service Systems with Priorities and Partial Backups

Akbar Karimi[1,2,*], Michel Gendreau[1,2], Vedat Verter[1,3]

[1] Interuniversity Research Centre on Enterprise Networks, Logistics and Transportation (CIRRELT)

[2] Department of Mathematics and Industrial Engineering, Polytechnique Montréal, P.O. Box 6079, Station Centre-Ville, Montréal, Canada H3C 3A7

[3] Desautels Faculty of Management, McGill University, Bronfman Building, 1001 Sherbrooke Street West, Montréal, Canada H3A 1G5

Abstract. An extension of Larson's approximate hypercube procedure is presented for priority emergency service systems with partial backups where requests for service with a given origin and priority can be responded to by a certain subset of response units. We introduce and analyze the family of queuing and loss systems with partial service and use them to approximate the distribution of the number of busy response units which in turn enables us to estimate the average server workloads, immediate and delayed dispatch rates, and waiting delays as system performance measures. We consider systems with zero and infinite queue capacities and let the dispatching policies and service times depend on call priority and customer and server locations. The validity of the approximation model and its computational aspects are studied through numerous tests and a realistic application of locating a fleet of ambulances in downtown Montreal, Canada.

Keywords. Emergency service systems, partial backup, hypercube queuing model, spatially distributed systems, server-to-customer systems, priority queues.

Acknowledgements. This work has been supported by the CREATE program on Health Care Operations and Information Management, by Polytechnique Montréal, and by the Natural Sciences and Engineering Research Council of Canada (NSERC) Discovery Grants and CIRRELT Scholarship programs.Their support is gratefully acknowledged.

---

* Corresponding author:Akbar.Karimi@cirrelt.ca

Dépôt légal – Bibliothèque et Archives nationales du Québec
Bibliothèque et Archives Canada, 2017

1. Introduction

Server-to-Customer systems represent an important class of spatially distributed queuing systems due to their strong presence in modern urban settings ranging from Emergency Service Systems (ESS) including ambulance, fire, police, and repair to non-emergency applications such as non-scheduled home visits, demand-responsive delivery operations, and dial-a-ride transport systems. The nature of ESSs, which involves risks to human lives, has made them a focal point of study by the operations research community over the past few decades as evidenced by the substantial research effort targeted at modelling, designing, and analyzing such systems in different application settings.

An important aspect of the strategic or operational design of ESSs is the ability to accurately predict the equilibrium behavior and hence the expected performance of the system with a given configuration. Discrete-event simulation can be used for this purpose and offers great flexibility in detailed modelling of the system at hand; however, its computational cost can be prohibitive in many practical applications. Fortunately, there exist analytical alternatives to simulation, with reduced computational overhead and reasonable accuracy.

Larson (1974) was the first to view ESS in urban settings as spatially distributed queues, and developed the hypercube queuing model as a descriptive tool for evaluating their performance. For an ESS with N servers, the model sets up a system of 2^N linear equations the solution of which gives the equilibrium probabilities of system states and makes it possible to compute a host of region-wide and server-specific performance measures, such as individual server workloads and the rates at which each server is dispatched to different demand locations. However, the exponential growth of the computational expense of the hypercube model with the fleet size may hinder its application to real life systems with large numbers of response units. This has motivated researchers to develop more tractable approximate alternatives. Larson (1975) was also the first to propose such an approximate procedure where an iterative algorithm is used to solve a system of N nonlinear equations for the individual server workloads (instead of the 2^N state probabilities of the exact model). The development is based on approximating the spatially distributed system with an M/M/N queue or an M/M/N/N loss system (depending on whether queues are allowed or not), and unlike the exact model, assumes identical mean service rates. Server dependency has also been approximately taken into account through a set of *correction factors*.

In this paper, we develop approximation algorithms for both loss and queuing ESSs with two main features. First, we allow multiple call types and priority processing of the waiting requests for the case where queues are allowed. Second, we relax the assumption of full backups, in which any server can travel to any demand location; instead, we allow arbitrary partial backups where

each server can only be dispatched to calls from its own predetermined and priority-specific subset of demand locations. The service times in our models can depend on the customer and server locations and also on the priority level of the request for service. We believe the relaxation of these restricting assumptions represents important steps toward increased realism and applicability of such approximation algorithms.

The assumption of full or total backups, rarely reflects the complex dispatching protocols of the real systems and hence can substantially limit the realism of the descriptive modelling, and as shown by our experiments, may lead to unreliable performance approximations. The paramount importance of the quickness of service delivery in ESSs naturally imposes an upper limit on the customer proximity to the responding server beyond which the quality or outcome of the service delivered can be severely degraded. This is reflected in the notion of coverage thresholds in location science literature and also in the behaviour of real systems where the distance between the customers and servers is a crucial factor in making dispatching decisions. Iannoni and Morabito (2007) present an example of an Emergency Medical Services (EMS) system deployed on highways where only the first and second closest ambulances are allowed to be dispatched to a call, and the request will be transferred to a private service if these two ambulances are busy at the arrival moment even if there are other available servers. Their work is also the only paper incorporating partial backups in the exact hypercube queuing model. To the best of our knowledge, there exists no published approximate method to deal with partial backups in server-to-customer systems or priorities in the waiting lines.

The original hypercube model and its approximation can handle a specific type of priority through the procedure of *layering*, in which demand zones are split into separate sub-nodes each generating a certain type of request and with their own list of preferred servers. However, this basic scheme only considers immediate dispatches and cannot be used to adequately represent the real life ESSs where the waiting customers are processed according to their priority levels. While a recent paper by de Souza et al. (2015) extends the exact hypercube model to account for priorities in the queue of waiting customers, our work is the first to provide the same extension in the approximate model.

The concurrent relaxation of the full backup and FCFS queue discipline assumptions in the approximate hypercube context, not only provides unique analytical and computational challenges, but also paves the way for more sophisticated and realistic modelling and analysis of ESSs for which these simplifying assumptions clearly do not hold. A typical EMS operation with a heterogeneous fleet, consisting of basic life support vehicles and advanced life support ambulances reserved for higher priority patients, is an example of such a system and in fact has been the primary motivation for this work.

The paper is organized as follows: Section 2 gives a brief review of the relevant literature followed by the formulation of our approximation model in Section 3. The approximation algorithm is outlined in Section 4 and in Section 5, we describe our experimental setup and observations. Concluding remarks are given in Section 6 along with possible directions for future development.

2. Literature Review

We first review the notable assumptions underlying Larson's hypercube queuing model and its approximate alternative. Both procedures assume that the demand for service is distributed over a given region broken down into a number of *geographic atoms* from which requests for service arrive at known mean rates following a Poisson process and independently of any other atom. There are N response units that can be dispatched to calls coming from any atom (full backup assumption). The waiting locations of the servers (when they are idle) as well as the mean travel times between each waiting location and atom are assumed to be, at least probabilistically, known. In response to each incoming call for service, exactly one server is dispatched according to a fixed-preference dispatch procedure, where the first available server in a given atom-specific ordered list of all response units is dispatched. If there are no free units, the call is assumed to be added to a waiting line of infinite capacity, which in case of the exact model is depleted according to a queue discipline that does not depend on the expected service time or the call origin, for example First-Come, First-Served (FCFS), Last-Come-First-Served (LCFS), or random. The approximate model is more restricting in this regard as it only allows for a FCFS queue discipline. Negative exponential service times with known average values are assumed that include the travel time, the on-scene time, and the follow-up time. It is also assumed that the service time variation is mostly caused by on-scene and follow-up times rather than travel times. The exact hypercube model allows for server-specific mean service rates whereas the approximate procedure assumes identical mean service rates for all response units, although the contribution of travel times to the service rates can be accounted for through an iterative process called *mean service time calibration*.

Both the exact and approximate hypercube models have been employed, modified, and extended in various ways. Reviews of the relevant developments can be found in Larson (2013) and Galvao and Morabito (2008). Here we briefly mention the notable extensions upon the original approximate procedure. Larson and Mcknew (1982) developed extensions of the original exact and approximate hypercube queuing models in a police patrol context and allowed three server states: patrol, busy with patrol-initiated activity, and busy with a call for service. They allowed the mean service times to depend on both servers and customer types in their exact model and only on customer types in their approximate version. Jarvis (1985) proposed an extended model for loss systems where the mean service times were allowed to depend on the server and customer locations. His formulation

also allowed general service time distributions since these loss systems with distinguishable servers were shown to be relatively insensitive to the shape of the service time distribution beyond its mean. Birge and Pollock (1989) proposed a decomposition procedure in which the large linear system of equations corresponding to state probabilities of service systems is replaced by a set of non-linear equations and then solved iteratively. Their approach is approximate in that the servers are assumed independent. Goldberg and Szidarovszky (1991) presented detailed convergence results for two fixed-point iteration methods assuming independent servers; they considered loss systems with server and customer dependent service times and showed that the independence assumption can lead to biased performance measures especially in high system utilization.

As for the dispatch preference ties, which most often happen in case of co-located servers sharing a waiting station, there are two works to note: the work of Burwell et al. (1993) who explicitly incorporated dispatch ties in Jarvis (1985) and Larson (1975) approximations of loss systems, through their *internal stacking method* and its slightly modified alternative; these algorithms were aimed to alleviate the storage and coding complexity issues involved in the so-called *stacking* procedure of accommodating dispatch ties in the hypercube models through software. Their formulation is based on individual servers and hence employs Larson correction factors to approximate server dependency. The other work by Budge et al. (2009), however, extends the Jarvis (1985) loss model to allow for co-located servers through a formulation organized around stations rather than individual servers and hence a modified set of correction factors that they derived for this purpose.

3. Formulation

In this section, we present our approximation model and the necessary formulation. The problem definition and preliminary assumptions are given first followed by the approximation of the distribution of the number of busy servers and the effects of cooperation and dependency among them. We then estimate the immediate and delayed dispatch rates of servers to demand locations.

We assume that the area of interest is partitioned into M distinct demand zones, each generating calls that can be classified into K priority levels. Requests for service with priority p from demand zone i arrive as Poisson streams with known average rates λ_{ip}. There are N mobile servers which respond to calls according to a fixed dispatch policy. For any given priority level p and demand zone i, there is a predetermined subset of servers which can be considered for dispatch. Binary variables b_{ijp} specify whether server j can be dispatched to priority p calls from demand zone i ($b_{ijp}=1$) or not ($b_{ijp}=0$). If $b_{ijp}=1$, we may say server j *p-covers* demand zone i, or equivalently, server j covers *sub-queue* (i,p). Letting $\mathcal{I}$, $\mathcal{J}$, and $\mathcal{P}$ respectively be the set of demand locations, servers, and priority classes, and denoting the number of servers covering sub-queue (i,p) by c_{ip} ($\leq N$) we have

$$c_{ip}=\sum_{j\in\mathcal{J}} b_{ijp} \qquad i\in\mathcal{I},\ p\in\mathcal{P}\,.$$

We assume that in response to an incoming call, the dispatcher considers the covering servers according to a fixed preference order and assigns the first available unit. The k-th preferred server to dispatch to a priority p call from demand zone i is denoted by $l(i,k,p)$ with $k = 1, ..., c_{ip}$. Auxiliary variables $r(i,j,p)$ are defined to denote the dispatch preference order of the j-th server when responding to a priority p call coming from demand zone i. By convention, we set $r(i,j,p) = 0$ if server j does not p-cover demand zone i, that is $b_{ijp} = 0$. Similarly, we set $l(i,k,p) = 0$ for $k > c_{ip}$. Thus, we will have $l(i,k,p) = j \Leftrightarrow r(i,j,p) = k$ for $k \leq c_{ip}$. As an example, consider a system with three demand zones ($M = 3$), three servers ($N = 3$), and two priority levels ($K = 2$). If the dispatch preference lists for the priority one calls coming from demand zones 1, 2, and 3 are respectively given as $\{2,3,1\}$, $\{3,2\}$ and $\{1\}$, then we will have $l(1,1,1) = 2$, $l(1,2,1) = 3$, $l(1,3,1) = 1$, $l(2,1,1) = 3$, $l(2,2,1) = 2$, $l(2,3,1) = 0$, $l(3,1,1) = 1$, $l(3,2,1) = 0$, and $l(3,3,1) = 0$, whereas $r(1,1,1) = 3$, $r(1,2,1) = 1$, $r(1,3,1) = 2$, $r(2,1,1) = 0$, $r(2,2,1) = 2$, $r(2,3,1) = 1$, $r(3,1,1) = 1$, $r(3,2,1) = 0$, and $r(3,3,1) = 0$.

We consider both loss and queuing systems; in the former case, calls arriving while all the covering servers are occupied will be lost, and in the latter case, they will join the queue of waiting customers. Because of partial backups, overtaking can happen in this queue when considered as one single line; however, the service discipline within each sub-queue will be strictly FCFS. Upon finishing its current job, a server will be dispatched to the longest waiting among the highest priority queued customers covered by that server, if one exists.

The rates at which priority p calls from demand location i are immediately assigned to a server or queued (lost in the loss system) are respectively denoted by λ^I_{ip} and λ^D_{ip}, with a^I_{ijp} and a^D_{ijp} the corresponding immediate and delayed dispatch rates to server j. We then have $\lambda_{ip} = \lambda^I_{ip} + \lambda^D_{ip}$, $a_{ijp} = a^I_{ijp} + a^D_{ijp}$, and $\lambda^I_{ip} = \sum_{j \in \mathcal{J}} a^I_{ijp}$. For the system with queues, we also have $\lambda^D_{ip} = \sum_{j \in \mathcal{J}} a^D_{ijp}$ and $\lambda_{ip} = \sum_{j \in \mathcal{J}} a_{ijp}$, whereas for loss systems, we interpret λ^D_{ip} as the mean rate at which calls are lost (rather than queued) and we set $a^D_{ijp} = 0$ since no customers will be dispatched from the queue.

3.1. Distribution of the Number of Busy Servers

The approximation will rely on the distribution of the number of busy servers $P_n = \Pr\{S_n\}$, $n = 0, \cdots, N$, with S_n the event of exactly n servers being busy. In the literature, a simplified queuing model with identical servers, that is M/M/N or M/M/N/N is usually employed to approximate the P_n of the spatially distributed system. Our experiments, however, show that these models are not adequate enough to approximate a system with partial backups operating at moderate to high workloads. To overcome this issue, we introduce *queuing (or loss) systems with partial service* defined below to act as a more accurate proxy to the EMS with partial backups.

DEFINITION 1. The queuing (loss) system with *partial service*, denoted by M/M/[N]/∞ (M/M/[N]/LOSS) has N identical servers with i.i.d exponential service times with rate μ and N customer classes arriving as Poisson streams with rates λ_c, $c=1,2,\cdots,N$, where each class c customer can receive service from exactly c servers randomly selected upon arrival, independently of other arrivals or properties of the system.

We give the distribution of the number of busy servers of the queuing and loss systems with partial service in the following theorems which form the basis of our subsequent analysis.

THEOREM 1. *The queuing system with partial service (M/M/[N]/∞) will reach steady state if* $N\mu > \sum_{c=1}^{N}\lambda_c$ *with the distribution of the number of busy servers given by*

$$P_n = P_0 \frac{N!}{(N-n)!n!}\prod_{j=0}^{n-1}\sum_{c=1}^{N}\lambda_c \sum_{\substack{h=max\{0,\\ c+j-N))\}}}^{min\{c-1,j\}} \frac{c!(N-c)!}{(N+h-c-j)!(j-h)!(c-h)!h!} \times \prod_{j\in\mathcal{J}} \frac{j!(N-j)!}{N!j\mu - j!\sum_{c=1}^{j}\lambda_c\frac{N-c}{j-c}} \quad n=1,\cdots,N, \quad (1)$$

where

$$P_0 = \left[1+\sum_{n=1}^{N}\frac{N!}{(N-n)!n!}\prod_{j=0}^{n-1}\sum_{c=1}^{N}\lambda_c \sum_{\substack{h=max\{0,\\ c+j-N))\}}}^{min\{c-1,j\}} \frac{c!(N-c)!}{(N+h-c-j)!(j-h)!(c-h)!h!} \times \prod_{j\in\mathcal{J}} \frac{j!(N-j)!}{N!j\mu - j!\sum_{c=1}^{j}\lambda_c\frac{N-c}{j-c}}\right]^{-1}. \quad (2)$$

THEOREM 2. *The steady state distribution of the number of busy servers in the loss system with partial service is given by*

$$P_n = P_0 \prod_{k=1}^{n}\frac{\sum_{c=1}^{k}\lambda_c(1-\frac{k!(N-c)!}{N!(k-c)!})+\sum_{c=k+1}^{N}\lambda_c}{k\mu}, \quad n=1,\cdots,N \quad (3)$$

with

$$P_0 = \left[1+\sum_{n=1}^{N}\prod_{k=1}^{n}\frac{\sum_{c=1}^{k}\lambda_c(1-\frac{k!(N-c)!}{N!(k-c)!})+\sum_{c=k+1}^{N}\lambda_c}{k\mu}\right]^{-1}. \quad (4)$$

The Proofs of Theorems 1 and 2 will be given in Appendix A.

We approximate the distribution of the number of busy servers in our priority EMS with partial backups using an M/M/[N] model with input parameters

$$\lambda_c = \sum_{i\in\mathcal{I}}\sum_{\substack{p\in\mathcal{P}\\ c_{ip}=c}}\lambda_{ip}, \quad c=1,\cdots,N, \quad (5)$$

and

$$\mu = \frac{\sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}}\sum_{p\in\mathcal{P}} a_{ijp}}{\sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}}\sum_{p\in\mathcal{P}} a_{ijp}t_{ijp}}\,, \tag{6}$$

whenever known or estimated values for the total dispatch rates a_{ijp} and individual service times t_{ijp} are available.

Unlike the M/M/N queue, the state probabilities of the prioritized M/M/[N] can differ from the non-priority version; the difference, however, is negligible and does not affect the accuracy of our calculations in any practical way. To illustrate this, in Figure 1, we compare the state probabilities of a large set of randomly generated M/M/[N] queues obtained by simulation with those of the equivalent non-priority M/M/N and M/M/[N] models. Noting an order of magnitude difference in the vertical axes, it is clear that the non-priority M/M/[N] model approximates the priority M/M/[N] significantly better than the non-priority M/M/N model with average errors often around 0.5% and rarely exceeding 1% which we deem quite negligible. This allows us to confidently use the state probabilities obtained from the M/M/[N] model as an approximation to the distribution of the number of busy servers in the ESS.

Similar to the original approximate hypercube model, the accuracy of P_n estimation decreases when the individual server workloads

$$\rho_j = \sum_{i\in\mathcal{I}}\sum_{p\in\mathcal{P}} a_{ijp}t_{ijp}, \quad j\in\mathcal{J}\,,$$

are not close enough to the average server workload

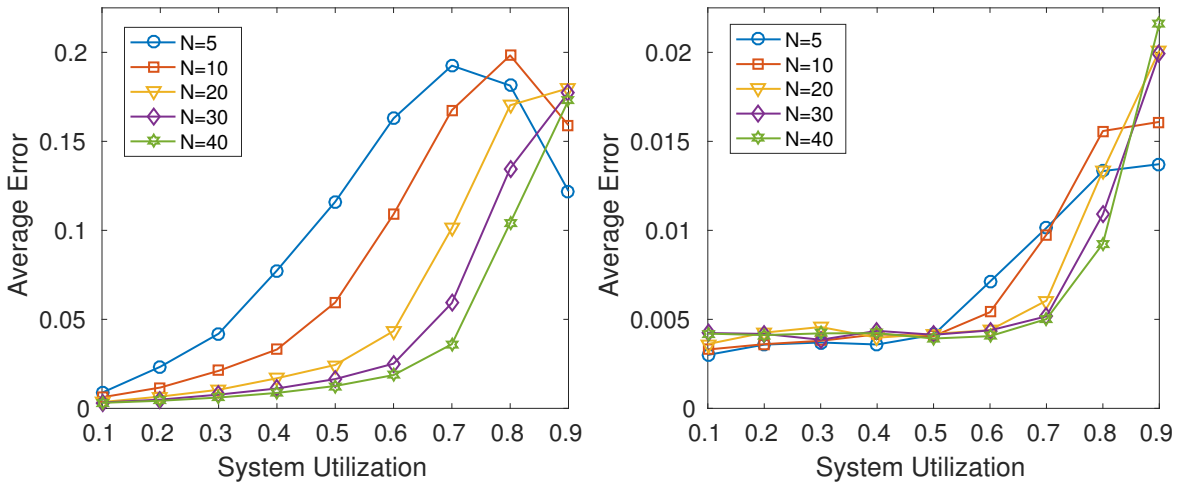
$$\rho = \frac{1}{N}\sum_{j\in\mathcal{J}}\rho_j = \sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}}\sum_{p\in\mathcal{P}} a_{ijp}t_{ijp}\,;$$

therefore, we propose an empirical approach detailed in Appendix B to further improve the accuracy of the estimation by correcting the λ_c and μ_c values obtained from (5) and (6) based on a measure of discrepancy between server workloads.

3.2. Immediate Dispatches

In this section, we consider the calls who find at least one available covering server upon arrival, and hence do not experience any queuing delays. We are interested in the steady-state rates at which these immediately serviced calls are assigned to each of their covering servers.

To approximately account for the effects of server cooperation, we adopt an approach similar to Larson (1975) where P_n is now obtained from an M/M/[N] model. First, assume we randomly sample servers without replacement from the system operating at steady state until we find an idle



(a) M/M/N queue (b) M/M/[N] queue

Figure 1 Comparison of errors in approximating the state probabilities of a priority partial service queue by the corresponding non-priority version (M/M/[N]) and an M/M/N queue. Reported are the mean absolute errors for a system with three priority levels and different numbers of servers.

one. Letting B_j and F_j be the events of the j-th sampled server being respectively busy and free, we write the probability of the $j+1$-th sampled server being the first available one as

$$\Pr\{B_1B_2\cdots B_jF_{j+1}\} = \sum_{n=j}^{N} \Pr\{S_n\}\Pr\{B_1B_2\cdots B_jF_{j+1}|S_n\};$$

moreover,

$$\Pr\{B_1B_2\cdots B_jF_{j+1}|S_n\} = \Pr\{F_{j+1}|B_1B_2\cdots B_jS_k\}\Pr\{B_j|B_1B_2\cdots B_{j-1}S_k\}\cdots\Pr\{B_1|S_n\}.$$

The probabilities on the right-hand side can be expressed as

$$\Pr\{B_i|B_1B_2\cdots B_{i-1}S_n\} = [n-(i-1)]/[N-(i-1)], \quad i=1,2,\cdots,n+1,$$

and

$$\Pr\{F_{j+1}|B_1B_2\cdots B_jS_n\} = (N-n)/(N-j), \quad j=0,1,\cdots,n.$$

Combining the results above we have the desired probability

$$\Pr\{B_1B_2\cdots B_jF_{j+1}\} = \sum_{n=j}^{N-1} \frac{n}{N}\frac{n-1}{N-1}\cdots\frac{n-(j-1)}{N-(j-1)}\frac{N-n}{N-j}P_n, \quad j=0,1,\cdots,N-1,$$

or

$$\Pr\{B_1B_2\cdots B_jF_{j+1}\} = \sum_{n=j}^{N-1} \frac{N-n}{N-j}P_n\prod_{k=0}^{j-1}\frac{n-k}{N-k}, \quad j=0,1,\cdots,N-1.$$

Re-writing the above expression as

$$\Pr\{B_1 B_2 \cdots B_j F_{j+1}\} = Z(N, \mu, [\lambda_c], j)' \rho^j (1-\rho)\,,$$

we obtain our correction factors as

$$Z(N, \mu, [\lambda_c], j) = \sum_{n=j}^{N-1} \frac{n!(N-j-1)!(N-n)}{(n-j)!N!(1-\rho)\rho^j} P_n\,, \quad j = 0, 1, \cdots, N-1\,.$$

For the M/M/[N]/∞ system, we have $\rho = \lambda/(N\mu)$ with $\lambda = \sum_{c=1}^{N} \lambda_c$ and thus the Z factors will be given as

$$Z(N, \mu, [\lambda_c], j) = \sum_{n=j}^{N-1} \frac{n!(N-j-1)!(N-n)}{(n-j)!N!(\frac{\lambda}{N\mu})^j (1-\frac{\lambda}{N\mu})} P_n\,, \quad j = 0, 1, \cdots, N-1\,, \tag{7}$$

with P_n given by (1). Class c customers arriving to the loss system in state S_n will be lost at an average rate of

$$q_c(n) = \frac{\binom{n}{c}}{\binom{N}{c}} = \frac{n!(N-c)!}{N!(n-c)!}\,, \tag{8}$$

resulting in an average server workload of

$$\rho = \frac{\sum_{c=1}^{N} \lambda_c (1 - \sum_{n=0}^{N} q_c(n) P_n)}{N\mu}\,,$$

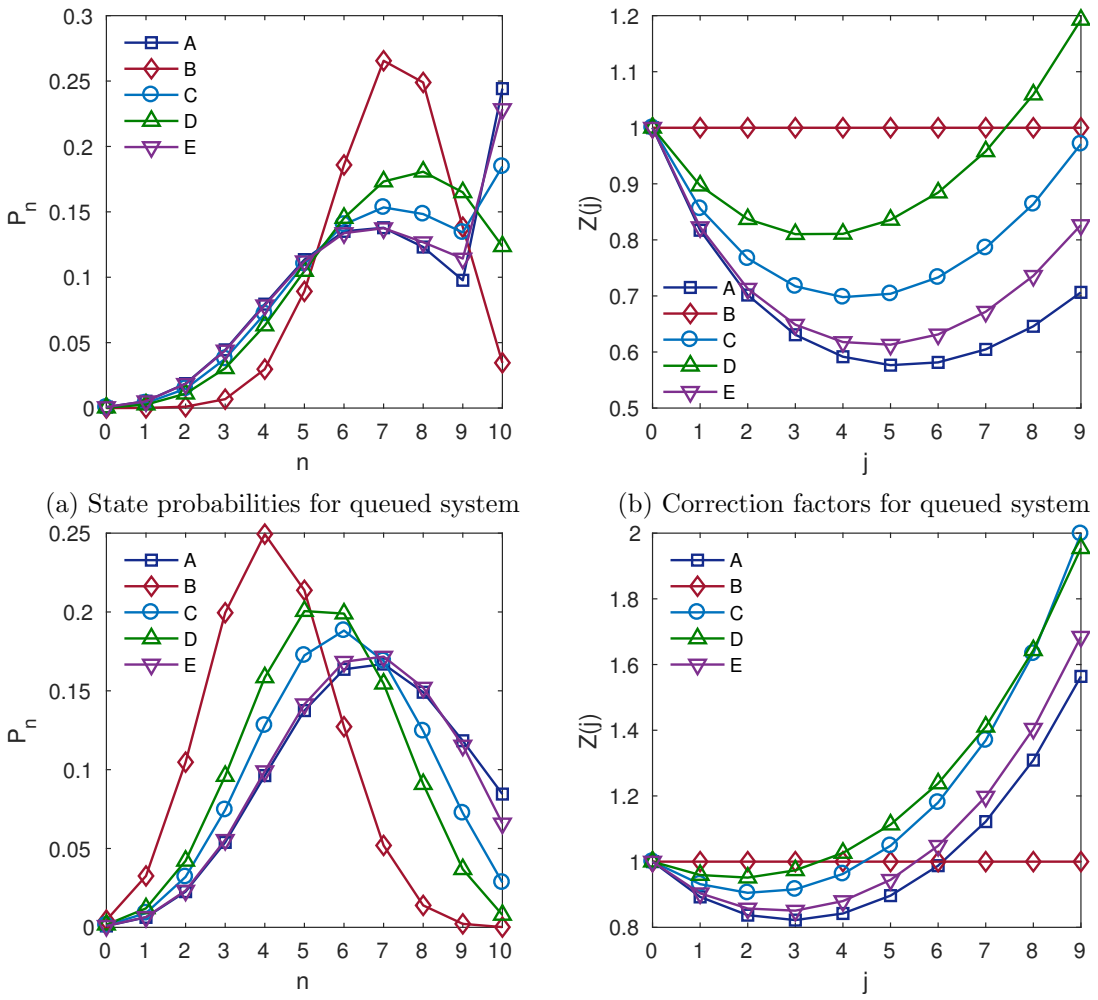
leading to the loss system Z factors of the form

$$Z(N, \mu, [\lambda_c], j) = \left[\frac{1}{N\mu} \sum_{c=1}^{N} \lambda_c \left(1 - \sum_{n=0}^{N} \frac{n!(N-c)!}{N!(n-c)!} P_n\right)\right]^j \left[1 - \frac{1}{N\mu} \sum_{c=1}^{N} \lambda_c \left(1 - \sum_{n=0}^{N} \frac{n!(N-c)!}{N!(n-c)!} P_n\right)\right] \times \sum_{n=j}^{N-1} \frac{n!(N-j-1)!(N-n)}{(n-j)!N!} P_n\,, \quad j = 0, 1, \cdots, N-1\,,$$

with P_n given by (3).

Similar to the Q factors derived by Larson (1975) for M/M/N queues, our Z factors represent the extent to which the term $\rho^j(1-\rho)$, which assumes independent servers, should be *corrected* to reflect the probability of sampling exactly j servers before finding an available server in an M/M/[N] queue operating at steady state.

Values of P_n and $Z(\cdot, j)$ for the queue and loss versions of an M/M/[10] system with different $[\lambda_c]$ but identical total demand are plotted in Figure 2, which clearly shows the strong dependence of both quantities on the distribution of total demand into the customer classes. We note that scenario A in this example is equivalent to a regular M/M/N queue and therefore the corresponding Z factors are identical to the Q factors of Larson (1975). It is particularly interesting to observe how shifting the concentration of the demand intensity towards the more openly received customer classes (higher c) increases the effects of server cooperation; this is highlighted by the most aggressive



(a) State probabilities for queued system

(b) Correction factors for queued system

(c) State probabilities for loss system

(d) Correction factors for loss system

Figure 2 Distribution of the number of busy servers and the Z correction factors for an example queuing system with $N = 10$, $\mu = 1.4$ and different arrival rate scenarios given by: A) $[\lambda_c] = [0,0,0,0,0,0,0,0,0,10]$, B) $[\lambda_c] = [10,0,0,0,0,0,0,0,0,0]$, C) $[\lambda_c] = [1,1,1,1,1,1,1,1,1,1]$, D) $[\lambda_c] = [2,2,2,2,2,0,0,0,0,0]$, and E) $[\lambda_c] = [0,0,0,0,0,2,2,2,2,2]$.

correction factors associated with scenario A which represents the extreme case with the highest server cooperation. Case B is the other extreme where each arrival is covered by a single server and hence we have zero cooperation and perfect independence reflected in correction factors equal to unity.

We also need an expression for λ_{ip}^D the rate at which priority p calls from location i get queued or lost. Denoting the event of having no free servers to cover a priority p call from demand location i by B_{ip}, we have $B_{ip} \equiv \bigcap_{j=1}^{c_{ip}} B_{l(i,p,j)}$. Conditioning on the system state S_n we have $\lambda_{ip}^D = \lambda_{ip} \Pr\{B_{ip}\}$ with

$$\Pr\{B_{ip}\} = \lambda_{ip} \sum_{n=0}^{N} P_n \Pr\{B_{ip} \mid S_n\}\,.$$

We see that $\Pr\{B_{ip} \mid S_n\} = 0$ for $n < c_{ip}$ and

$$\Pr\{B_{ip} \mid S_n\} = \frac{\binom{N-c_{ip}}{n-c_{ip}}}{\binom{N}{n}} = \frac{(N-c_{ip})!n!}{(n-c_{ip})!N!},$$

for $n \geq c_{ip}$. Thus

$$\lambda_{ip}^D = \lambda_{ip} \sum_{n=c_{ip}}^{N} P_n \frac{(N-c_{ip})!n!}{(n-c_{ip})!N!}. \tag{9}$$

Given a set of steady-state server workloads ρ_j, $j \in \mathcal{J}$, one can obtain a set of tentative values for the immediate dispatch rates a_{ijp}^I as

$$a_{ijp}^I = b_{ijp}\lambda_{ip} Z(N, \mu, [\lambda_c], r(i,j,p)-1)(1-\rho_j) \prod_{k=1}^{r(i,j,p)-1} \rho_{l(i,k,p)}, \tag{10}$$

which can then be normalized using any of the normalization schemes discussed in Section 4 to satisfy the balance equation

$$\sum_{j \in \mathcal{J}} a_{ijp}^I = \lambda_{ip} - \lambda_{ip}^D. \tag{11}$$

We can also approximate a_{ijp}^I and λ_{ip}^D using an alternative two-pass scheme where we make a better use of the individual server workloads ρ_j; in the first pass, values for the dispatch rates assuming full backups denoted by $\hat{a}_{ijp}^I$ are calculated as

$$\hat{a}_{ijp}^I = \lambda_{ip} Z(N, \mu, [\lambda_c], r(i,j,p)-1)(1-\rho_j) \prod_{k=1}^{r(i,j,p)-1} \rho_{l(i,k,p)}, \tag{12}$$

and normalized according to the balance equation

$$\sum_{j \in \mathcal{J}} \hat{a}_{ijp}^I = \lambda_{ip} (1 - P_N), \tag{13}$$

where P_N is given by (1) or (3). In the second pass, the a_{ijp}^I values are obtained by recognizing that the normalized full-backup dispatches to non-covering servers would indeed be queued or lost in the actual partial-backup system; that is

$$a_{ijp}^I = b_{ijp}\hat{a}_{ijp}^I. \tag{14}$$

Finally, the rate of calls being queued (or lost) is given by

$$\lambda_{ip}^D = \lambda_{ip} P_N + \sum_{j \in \mathcal{J}} (1 - b_{ijp})\, \hat{a}_{ijp}^I, \tag{15}$$

where the first part of the right-hand side represents the customers who arrive when all servers of the hypothetical full-backup system are busy and the second term adds the contribution of partial

backups by summing up the hypothetical dispatches to the non-covering servers. Our numerical tests show that, on average, using the two-pass scheme defined by (12), (13), (14) and (15) instead of (11), (9) and (10), indeed leads to around 50% reduction in server workload and immediate dispatch estimation errors with an even higher 80% improvement in the estimation of delayed dispatch rates and waiting times discussed in the next section. We hence use this method in our experiments.

3.3. Delayed dispatches

The delayed dispatches in a system with full backups will be uniformly distributed among the N servers when service times are identical; that is $a^D_{ijp} = \lambda^D_{ijp}/N$ if $t_{ijp} = 1/\mu,\ i \in \mathcal{I},\ j \in \mathcal{J},\ p \in \mathcal{P}$. With location or priority dependent service times, some unevenness will be introduced in this distribution proportional in magnitude to the extent of differences in t_{ijp} values. This, however, will generally be of secondary importance compared to the asymmetry of the case with partial backups where servers covering a given sub-queue will have to finish work on different subsets of waiting customers before being able to receive the next delayed customer from that sub-queue. In this section, we are interested in quantifying this asymmetry and deriving approximate expressions for the steady-state dispatch rates of waiting customers and the average delays incurred in the queue.

Let B^D_{jp} be the event of server j being busy with a delayed priority p call and ρ^D_{ijp} the probability of server j being busy with a delayed call from demand location i with priority p *or higher* (that is with priority $p' \leq p$) given that all servers p-covering the demand location i are also busy (the event $\tilde{B}_{ip}$); that is $\rho^D_{ijp} = \Pr\left\{\cup^p_{p'=1} B^D_{jp'} \mid \tilde{B}_{ip}\right\}$. Although obtaining exact expressions for ρ^D_{ijp} for the general case is mathematically intractable, we propose the following approximation

$$\rho^D_{ijp} = \sum_{p'=1}^{p} \sum_{i' \in \mathcal{I}} \kappa^{ip}_{i'p'} \lambda_{ip} \bar{a}^D_{i'jp'} t_{i'jp'}\,, \tag{16}$$

with $\bar{a}^D_{i'jp'} = a^D_{i'jp'}/\lambda^D_{ip}$ and the *coupling factor* $\kappa^{ip}_{i'p'}$ defined as the probability of all servers p'-covering demand location i' being simultaneously busy given that all servers p-covering demand location i are busy; or

$$\kappa^{ip}_{i'p'} = \Pr\{\tilde{B}_{i'p'} \mid \tilde{B}_{ip}\}\,, \quad i, i' \in \mathcal{I}, p, p' \in \mathcal{P}\,.$$

The coupling factors can be approximated assuming either indistinguishable or independent servers. With the assumption of indistinguishable dependent servers, we readily observe that

$$\kappa^{ip}_{i'p'} = \Pr\{\tilde{B}_{i'p'} \mid \tilde{B}_{ip}\} = \frac{\Pr\{\tilde{B}_{i'p'} \cap \tilde{B}_{ip}\}}{\Pr\{\tilde{B}_{ip}\}}\,,$$

and also that, conditioning on the system state S_n

$$\Pr\{\tilde{B}_{i'p'} \cap \tilde{B}_{ip}\} = \sum_{n=0}^{N} P_n \Pr\{\tilde{B}_{i'p'} \cap \tilde{B}_{ip} \mid S_n\}.$$

Defining $h^{ip}_{i'p'}$ as the number of servers that either p-cover demand location i or p'-cover demand location i', that is $h^{ip}_{i'p'} = \sum_{j\in\mathcal{J}} 1-(1-b_{ijp})(1-b_{i'jp'})$, we have

$$\Pr\{\tilde{B}_{i'p'} \cap \tilde{B}_{ip} \mid S_n\} = \binom{N-h^{ip}_{i'p'}}{n-h^{ip}_{i'p'}}\binom{N}{n}^{-1},$$

if $h^{ip}_{i'p'} \leq n$ and $\Pr\{\tilde{B}_{i'p'} \cap \tilde{B}_{ip} \mid S_n\} = 0$ otherwise. The coupling factors are then obtained as

$$\kappa^{ip}_{i'p'} = \frac{\sum_{n=h^{ip}_{i'p'}}^{N} P_n (N-n)!^{-1} \prod_{k=n+1}^{N} (k-h^{ip}_{i'p'})}{\sum_{n=c_{ip}}^{N} P_n (N-n)!^{-1} \prod_{k=n+1}^{N} (k-c_{ip})}. \tag{17}$$

Equation (17) implies indistinguishable servers assumption as no individual server workloads are used. However, server dependency is taken into account through the conditioning on the system state S_n.

We can also assume independent distinguishable servers and approximate $\kappa^{ip}_{i'p'}$ as

$$\kappa^{ip}_{i'p'} = \prod_{\substack{j\in\mathcal{J} \\ b_{i'jp'}=1,\, b_{ijp}=0}} \rho_j. \tag{18}$$

As discussed in Section EC.4 of the electronic companion, this approach can be a viable alternative to (17) and depending on the problem size and the implementation, may result in better or worse performance in terms of accuracy or efficiency.

Finally, we define *residual service rates* μ^D_{ijp} as the service rate delivered by server j to the end of the waiting line of priority p customers from demand location i and write the approximating expression

$$\mu^D_{ijp} = b_{ijp}\mu_j \left(1-\rho^D_{ijp}\right)\left(1-\rho^D_{ij(p-1)}\right), \tag{19}$$

with μ_j the average service rate of server j defined as

$$\mu_j = \frac{\sum_{p\in\mathcal{P}}\sum_{i\in\mathcal{I}} a_{ijp}}{\sum_{p\in\mathcal{P}}\sum_{i\in\mathcal{I}} a_{ijp}t_{ijp}}.$$

With μ^D_{ijp} known, we compute the normalized delayed dispatch rates as

$$\bar{a}^D_{ijp} = \frac{\mu^D_{ijp}}{\sum_{k=1}^{N}\mu^D_{ikp}}$$

and the average time spent in queue by the delayed customers from sub-queue (i,p) as

$$w_{ip}^D = \left(\sum_{j \in \mathcal{J}} \mu_{ijp}^D \right)^{-1} . \tag{20}$$

The delayed dispatch rates and the waiting times of *all arrivals* from sub-queue (i,p) then follow from

$$a_{ijp}^D = \lambda_{ijp}^D \bar{a}_{ijp}^D , \tag{21}$$

and

$$w_{ip} = \frac{\lambda_{ip}^D}{\lambda_{ip}} w_{ip}^D . \tag{22}$$

Expression (19) has an intuitive interpretation. Imagine a priority p customer from demand location i enters the queue. A covering server j will finish its current task (possibly with an immediately dispatched customer) at rate μ_j; with probability $1-\rho_{ijp}^D$ no other customer will be waiting before this newly arrived one in gaining access to server j; and with probability $1-\rho_{ij(p-1)}^D$ no higher priority customer covered by server j will push this customer back while he is waiting. It can be verified that for the special case with full backups ($b_{ijp}=1$, $i \in \mathcal{I}$, $j \in \mathcal{J}$, $p \in \mathcal{P}$), all coupling factors $\kappa_{i'p'}^{ip}$ will be equal to unity. If we also have identical service times (that is $t_{ijp}=1/\mu$ for all $i \in \mathcal{I}$, $j \in \mathcal{J}$, and $p \in \mathcal{P}$), then relations (16), (19), (21) and (20) will hold exactly and yield the well-known expression for the waiting time of the prioritized M/M/N/∞ queue.

We now have all the ingredients to derive an iterative algorithm to approximate the steady-state values of the desired performance measures, namely the average server workloads, immediate and delayed dispatch rates, and waiting times.

4. Algorithm

In this section, we provide the layout of our approximation algorithm. All the assignments are assumed to be carried out for $i,i' \in I$, $j \in J$, and $p,p' \in P$. Here are the steps to follow:

1. Initialization

   (a) Set the iteration counter: $g=0$;

   (b) Initialize the problem definition parameters N, M, K, λ_{ip}, t_{ijp}, $l(i,j,p)$, $r(i,j,p)$, c_{ip}, b_{ijp}, and $h_{i'p'}^{ip} \leq n$;

   (c) Starting from an empty system, initialize the dispatch rate and server workload variables to zero: $\rho_j^{(g)}=0$, $a_{ijp}^{(g)}=0$, and $\rho^{(g)}=0$;

2. Pre-processing

   (a) Compute the state probabilities of the underlying M/M/[N] queue, $P_0, P_1, \cdots, P_N$ from (1) and (2) for the system with queues and from (3) and (4) for the loss system;

   (b) Compute Z correction factors from (7);

3. Immediate Dispatches
   (a) Compute the full-backup immediate dispatch rates $\hat{a}^I_{ijp}$ from (12);
   (b) Normalize $\hat{a}^I_{ijp}$ values obtained above so that $\sum_{j\in\mathcal{J}} \hat{a}^I_{ijp} = \lambda_{ip}(1-P_N)$;
   (c) Update the immediate dispatch rates as $a^I_{ijp} = b_{ijp}\hat{a}^I_{ijp}$;
   (d) Compute the rates of delayed or lost calls from (15);
4. Delayed Dispatches
   (a) Compute the coupling factors $\kappa^{ip}_{i'p'}$ either from (17) or (18);
   (b) Compute updated values for ρ^D_{ijp} from (16);
   (c) Compute updated values for the residual service rates μ^D_{ijp} from (19);
   (d) Obtain the delayed dispatch rates as in (21): $a^D_{ijp} = \lambda^D_{ijp}\mu^D_{ijp}/\sum_{k=1}^N \mu_{ikp}$;
   (e) Compute the expected waiting times from (22): $w_{ip} = \left(\lambda^D_{ip}/\lambda_{ip}\right)\sum_{j=1}^N \mu^D_{ijp}$;
5. Global Update and Termination
   (a) Define and set auxiliary variables V_j as

$$V_j = \frac{\sum_{i=1}^N \sum_{p\in\mathcal{P}} a_{ijp}t_{ijp}}{1-\rho_j^{(g)}};$$

   (b) Update server workloads as $\rho_j^{(g+1)} = V_j/(1+V_j)$. This update rule ensures that server workloads remain between 0 and 1 and helps with the stability of the algorithm. For details see Larson (1975), Budge et al. (2009) and Goldberg and Szidarovszky (1991).
   (c) Terminate the algorithm if a given convergence condition is satisfied such as

$$\max_{j\in\mathcal{J}} |\rho_j^{(g+1)} - \rho_j^{(g)}| \leq \epsilon,$$

with ϵ a predefined convergence threshold; otherwise, set $g \leftarrow g+1$ and start a new iteration from Step 2.

The normalization procedure in step 3b can be carried out in different ways. The simplest method is to scale all $\hat{a}^I_{ijp}$ for $j \in \mathcal{J}$ and a given sub-queue (i,p) by a single factor so that the balance equation is satisfied. Alternatively, we can retain the element corresponding to the primary (the most preferred) server (that is $\hat{a}^I_{i1p}$) and scale all the remaining values. In our experience this method slightly improves the approximation and is thus employed in our numerical experiments. We found more complicated approaches, such as the third normalization method in Larson (1975), to be generally too computationally costly for the systems and scales we consider here.

Finally, although the algorithm presented above starts from an empty system, based on our experiments, the method performs equally well with arbitrary initial conditions. An illustration of the algorithm is given in Appendix C.

Table 1 Components of the service time (in minutes).

	Priority 1	Priority 2	Priority 3
Dispatch Time	2	3	3
Chute Time	0.5	0.5	0.5
Scene Time (Patient Transported)	19	19	19
Scene Time (Patient not Transported)	22	22	22
Turnaround Time	40	50	50

5. Numerical Experiments

We have conducted comprehensive numerical experiments to asses the efficacy of the proposed approximation algorithm and also to answer a few questions that may arise while applying the model to practical cases. A model representing the downtown area of the city of Montreal, Canada and its main EMS provider, has been developed and employed in our tests as a realistic example of a typical ESS in an urban setting. This model comprises 447 uniformly distributed demand zones from which requests for service arrive with varying intensities at three different priority levels corresponding to urgent (priority 1), less urgent (priority 2) and non-urgent (priority 3) calls. We used a regular rectangular grid and approximated the total demand intensity of each grid cell based on the population densities of the census dissemination areas intersecting that cell obtained from the latest census data provided by Statistics Canada (2016a,b). Figure 3 shows the distribution of the demand intensity and locations of the hospitals serving the area.

We assume the service times to include six components: (1) Dispatch Time: from the moment the dispatcher received the call until a decision is made to assign an ambulance to the call or put it in the waiting queue; (2) Chute Time: from the moment the ambulance crew is notified of the dispatch until they become en route to the scene; (3) Travel Time: ambulance traveling time from the current location to the scene; (4) Scene Time: time spent on scene; (5) Transport Time: traveling time from the scene to the care center; (6) Turnaround Time: from the arrival of the ambulance to the care center until it is back in service which typically consists of transferring the patient to the care center and possibly ambulance clean-up or replenishing any depleted resources. Based on data we had on ambulance operations in Montreal and other Canadian metropolitan areas we estimated the values of the non-traveling components of the service time as in Table 1.

For the travel and transport times, we use the log-normally distributed stochastic model in Budge et al. (2010) to approximate the random time t to travel a given distance d, that is

$$t = m(d)e^{c(d)z},$$

$$m(d) = \begin{cases} 2\sqrt{d/a} & \text{if } d \le 2d_c \\ v_c/a + d/v_c & \text{if } d > 2d_c \end{cases},$$

Table 2 Values of the travel time model parameters.

Parameter	Priority 1	Priority 2	Priority 3
a (acceleration, km/h/min)	41	25	25
v_c (cruising speed, km/hr)	100	70	70
b_0	0.336	0.336	0.336
b_1	0.000058	0.000116	0.000116
b_2	0.0388	0.0776	0.0776

Table 3 Coverage threshold scenarios considered in the experiments. In each scenario, the maximum coverage threshold for each priority level is given in kilometers.

Scenario	Priority1	Priority 2	Priority 3
C1*	∞	∞	∞
C2	6	6	6
C3	4	4	4
C4	6	4	2
C5	2	4	6
C6	2	2	2

* Full backups

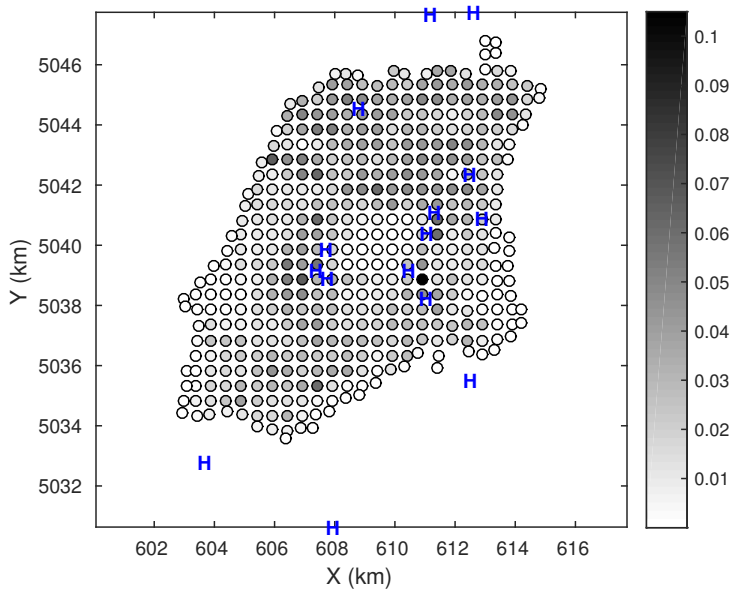
$$c(d) = \frac{\sqrt{b_0(b_2+1) + b_1(b_2+1)m(d) + b_2 m(d)^2}}{m(d)},$$

where $m(d)$ and $c(d)$ are respectively the median and coefficient of variation (CV) of the travel time distribution with z a random variable with a standard normal distribution. The values we selected for the parameters of the model are given in Table 2 for each priority level. For the urgent calls, we used the values reported in Budge et al. (2010) for the Calgary EMS, and for the non-urgent calls, we simply modified some of the parameters to reflect the lower cruising speeds and accelerations and higher variability associated with driving in normal mode and abiding by all traffic laws. Values of d were computed as Manhattan norms tilted at 45 degrees to match the street layout in the region. Finally, the corresponding mean travel times needed by the approximation model can be obtained as

$$\bar{t} = m(d)e^{c(d)^2/2}.$$

We have conducted tests using six different coverage threshold scenarios given in Table 3 with C1 representing the full backup scenario. For a given fleet size (N), we generated 100 random ambulance location sets where no ambulance is located farther than 500 meters from its closest demand zone (to exclude unrealistic deployment patterns). We considered a given test location set in both queuing and loss situations (designated by Queue and Loss in the tables) and in each case compared the outputs of the approximation model with those of a discrete event simulation which

Figure 3 Demand distribution and hospital locations.



we developed to benchmark the performance of our algorithm. We have run the tests for N=20, 30, 40, and 50, but only report the results for N=20 for ease of the exposition.

Denoting simulation outputs by hatted symbols, we use $E_\rho = \sum_{j=1}^{N} |\rho_j - \hat{\rho}_j|/\hat{\rho}_j$ as the measure of error in the estimation of server workloads for a given test problem. For total, immediate, and delayed dispatch rates, we take the average errors per dispatch given by $E_a = \sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}}\sum_{p\in\mathcal{P}} |a_{ijp} - \hat{a}_{ijp}|/\lambda$, $E_a^I = \sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}}\sum_{p\in\mathcal{P}} |a^I_{ijp} - \hat{a}^I_{ijp}|/\lambda$ and $E_a^D = \sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}}\sum_{p\in\mathcal{P}} |a^D_{ijp} - \hat{a}^D_{ijp}|/\lambda$ as error measures, respectively. We capture waiting time estimation errors by $E_w = \sum_{i\in\mathcal{I}}\sum_{p\in\mathcal{P}} |w_{ip} - \hat{w}_{ip}|/\lambda$ and also report the average waiting times from the simulation $\hat{w} = \sum_{i\in\mathcal{I}}\sum_{p\in\mathcal{P}} \hat{w}_{ip}/\lambda$ as a reference for E_w. We emphasize that $\lambda = \sum_{\substack{i\in\mathcal{I},p\in\mathcal{P}\\ c_{ip}>0}} \lambda_{ip}$ is the total covered demand and hence will be different for each test problem in scenarios with finite coverage threshold. The measures computed for a specific priority p are denoted by E_{a^p}, $E^I_{a^p}$, $E^D_{a^p}$, E_{w^p} and $\hat{w}_p$ and of course normalized with the total priority p covered demand. Finally, we report in the subsequent tables the error measures averaged over all test problems in percentage format; that is, for instance $\bar{E}_\rho = (1/T)\sum_{m=1}^{T} E^m_\rho/100\%$ where E^m_ρ is E_ρ corresponding to the m-th out of the T number of test cases (here $T = 100$).

In this section, we test the validity of the full-backup assumption, assess the general efficacy of the algorithm and give a summary of the rest of our computational experiments presented in the electronic companion.

Table 4 Estimation errors assuming full-backups*.

	Loss						Queue					
	C1	C2	C3	C4	C5	C6	C1	C2	C3	C4	C5	C6
$\bar{E}_\rho$	1.68	4.67	17.98	22.64	38.53	111.30	1.69	2.08	9.54	10.60	20.09	78.84
$\bar{E}_a$	5.43	8.95	23.87	26.31	40.10	65.68	5.42	11.97	31.75	31.21	50.15	87.35
$\bar{E}_w$	—	—	—	—	—	—	0.00	0.79	6.70	6.24	13.20	44.97

*$\bar{E}_\rho$ and $\bar{E}_a$ in percentages, $\bar{E}_w$ in minutes

5.1. Validity of the Full-backup Assumption

This paper is focused on the approximation of ESSs with partial backups. It is then naturally interesting to see whether an approximation based on a full-backup assumption will be adequate or not in practical settings. In Table 4 we compare the performance measures obtained from the simulation model for different coverage threshold scenarios with the values predicted by the approximation algorithm assuming full-backups (that is the coverage scenario C1). It is readily observed that as the coverage thresholds become more stringent, the approximation model with a full-backup assumption becomes increasingly incapable of predicting any of the performance measures with reasonable accuracy, with estimation errors up to 100% and 90% for the server workload and dispatch rates. We note that the full-backup assumption yields an average waiting time of nearly zero resulting in 100% waiting time estimation errors for all coverage scenarios except C1. We also observed in our experiments that the discrepancy between the simulation and the full backup model, not surprisingly, grows with increasing system workloads and decreasing fleet sizes.

As mentioned earlier, in systems with partial backups, the values of performance measures corresponding to each server can be largely different and the assumption of full backups will not reveal these imbalances. However, there is another major drawback associated with assuming full backups in applications where the demand that is not covered by any server is considered lost. In these cases, each candidate positioning of the response units will lead to a different total covered demand, λ, and thus a different average server workload. The full backup assumption will also fail to reveal these effects. Therefore, we can conclude that assuming full backups may result in highly inaccurate approximations in many practical applications, especially with fairly moderate to high congestion levels and dispatch protocols not compatible with this assumption.

5.2. Accuracy of the Approximation

Having established the importance of relaxing the full-backup assumption, we now evaluate the predictive accuracy of the algorithm. The average total and priority-specific estimation errors in each performance measure for different coverage scenarios and queue disciplines are given in Tables

Table 5 Estimation errors for the loss system (in %).

	C1	C2	C3	C4	C5	C6
$\bar{E}_\rho$	1.68	1.51	1.50	1.31	1.09	0.70
$\bar{E}_{a^1}$	5.41	4.28	3.30	3.98	1.36	0.96
$\bar{E}_{a^2}$	5.43	4.31	3.30	3.04	2.85	0.94
$\bar{E}_{a^3}$	5.53	4.40	3.36	1.54	3.78	1.02
$\bar{E}_a$	5.43	4.31	3.32	3.23	2.47	0.96

5, 6, and 7. The error margins in prediction of server workloads and dispatch rates are capped at around 1.7% and 5.5% respectively, with considerably lower averages across the test scenarios. The average error in the estimation of waiting times in cases with significant average waiting time values is observed to be at most 10%. The server workload estimation errors are in agreement with the values reported by Larson (1975) and Budge et al. (2009). Overall, we deem the accuracy of the approximation well within the acceptable range in most practical applications. Comparing the error margins with those obtained with a full backup assumption in the previous subsection shows the effectiveness of the method in handling partial backups in the presence of priorities in the queues.

We envisioned to use the example application presented here in developing equitable emergency response models; therefore, we have used a regular grid for demand distribution so that geographical locations are equally represented regardless of the call volumes they generate. This, however, comes at the cost of impractically long simulation run times as the convergence rate of simulation outputs corresponding to demand locations with very low intensities will be extremely slow. This issue is further aggravated by the relatively large scale of the model considered here ($M = 447$), and is naturally more pronounced with larger numbers of non-zero elements of b_{ijp}, that is less stringent coverage threshold scenarios. We consider this a major drawback of using simulation models in studying problems with hugely varying arrival rates; moreover, we assume the reported error margins to be, to some extent, overestimated because of this issue, particularly in less stringent coverage threshold scenarios. The noticeable increase in the error margins with increasing coverage thresholds can be attributed in part to this issue, and in part to the fact that in this application, the total demand faced by the system (λ) increases with increasing coverage thresholds, thus leading to bigger error magnitudes.

5.3. Complementary Computational Results

We now briefly summarize the extra computational experiments presented in the electronic companion to investigate important aspects of the algorithm.

We examine the sensitivity of the algorithm to the service time distribution in section EC.1 and conclude that while the approximations of server workloads and dispatch rates remain fairly

Table 6 Estimation errors for the queuing system (in %).

	C1	C2	C3	C4	C5	C6
$\bar{E}_{\rho}$	1.69	1.59	1.26	1.34	1.14	0.46
$\bar{E}^{I}_{a^1}$	5.38	5.16	4.29	5.24	1.59	1.35
$\bar{E}^{I}_{a^2}$	5.40	5.16	4.29	3.64	3.83	1.38
$\bar{E}^{I}_{a^3}$	5.50	5.22	4.38	1.63	5.37	1.43
$\bar{E}^{D}_{a^1}$	0.09	1.16	2.25	0.91	1.27	1.13
$\bar{E}^{D}_{a^2}$	0.09	1.15	2.37	1.81	1.82	1.17
$\bar{E}^{D}_{a^3}$	0.13	1.17	2.47	1.31	0.83	1.29
$\bar{E}_{a}$	5.42	5.19	4.01	3.97	3.19	1.20

Table 7 Waiting time estimation errors with actual values from simulation in parentheses (in minutes).

	C1	C2	C3	C4	C5	C6
$\bar{E}_{w^1}$	0.00 (0.014)	0.13 (0.69)	0.46 (5.07)	0.10 (0.54)	0.67 (27.22)	0.56 (24.84)
$\bar{E}_{w^2}$	0.00 (0.024)	0.17 (0.85)	0.72 (7.16)	0.46 (4.81)	0.45 (6.03)	1.27 (44.59)
$\bar{E}_{w^3}$	0.01 (0.037)	0.21 (1.03)	1.43 (9.58)	2.33 (30.44)	0.15 (0.83)	2.47 (61.10)
$\bar{E}_{w}$	0.00 (0.021)	0.16 (0.80)	0.69 (6.72)	0.58 (6.26)	0.49 (13.22)	1.18 (40.35)

insensitive to the service time distribution, the waiting time estimations remain valid only if the service time distribution has a CV close to unity.

In Section EC.2, we verify the importance of modeling location and priority dependent service times through test cases in which service times are identical or depend only on location or priority. It is then clearly seen that letting the service times simultaneously depend on customer and server locations and call priority, significantly improves the approximation accuracy and hence is worth the extra modeling effort.

Repeating the numerical tests for varying load factors, we show in Section EC.3 that regardless of the system workload, the approximations provided by the algorithm remain accurate in all performance measures and across all tested scenarios.

Finally, in Section EC.4 we consider the computational expense of the algorithm and suggest optimal intervals to repeat time-consuming calculations so that the computational overhead is reduced without significantly affecting the accuracy.

6. Conclusions

We extended the EMS approximation procedure of Larson (1975) by letting each server be responsible for an arbitrary subset of the demand locations (partial backups) and allowing priorities in the queues. We considered queuing and loss systems and let service times depend on the locations of the customer and server and the call priority. The approximation is based on the queue or loss systems with partial service which we defined and analyzed in steady-state.

We conducted numerical experiments to validate the approximation model and demonstrated that it can accurately predict the performance measures of typical EMS systems with real scales and different operational scenarios.

The proposed method facilitates the more realistic description of emergency response systems where the assumptions of full backups and no priorities in the queues are often too simplistic to represent the dispatching policies of the actual system. In particular, the algorithm paves the way to efficient and reliable analysis and design of systems with multi-tier customers and heterogeneous fleet where dispatch decisions are made based on the customer and server types and only if an accordingly determined minimum response time is not expected to be exceeded.

An interesting line of research to follow will be to extend the current approximation procedure to model EMS systems that intentionally queue or reject lower priority requests to maintain a strategic reserve of available servers for upcoming higher priority calls. A similar extension to the exact hypercube model has been given by Iannoni et al. (2015).

Appendix A: Proof of Theorems

Proof of Theorem 1 We first briefly summarize the results obtained by Visschers et al. (2012) for a class of queues with skill based service and exploit them to derive our expressions of interest.

In a skill based queue with a set of servers $\mathcal{M} = \{m_1, \cdots, m_N\}$, and a set of customer classes $\mathcal{C} = \{\kappa_1, \kappa_2, \cdots\}$, each server $m_k \in \mathcal{M}$ has the skills to only serve a subset $\mathcal{C}(m_k) \subset \mathcal{C}$ of customer types. Servers are memory-less with service rates μ_m, $m \in \mathcal{M}$ and customers of class $\kappa \in \mathcal{C}$ arrive in a Poisson stream with rate λ_κ. The set of customer classes only compatible with servers $\{M_1, \cdots, M_n\} \in \mathcal{M}^n$ is denoted by $\mathcal{U}(\{M_1, \cdots, M_n\})$ where $\mathcal{M}^n$ is the set of all subsets of size n of $\mathcal{M}$. We readily observe that $\mathcal{U}(\{M_1, \cdots, M_n\}) = \mathcal{C} \setminus \bigcup_{m \in \mathcal{M} \setminus \{M_1, \cdots, M_n\}} \mathcal{C}(m)$. For a subset $\mathcal{C}' \subset \mathcal{C}$ of customer types, we denote $\lambda_{\mathcal{C}'} = \sum_{\kappa \in \mathcal{C}'} \lambda_{c'}$; similarly $\mu_{\mathcal{M}' \subset \mathcal{M}} = \sum_{m \in \mathcal{M}'} \mu_m$.

A state of the system, given by the sequence $(M_1, l_1, M_2, l_2, \cdots, M_n, l_n)$, represents a snapshot of the waiting customers and the relative positions of the busy servers in the queue: there are $l_1 + \cdots + l_i$ $(l_i = 0, 1, \cdots)$ customers waiting between the sequence of busy servers $(M_1, \cdots, M_n)$, plus n customers currently in service. Arriving customers start service immediately if they find a compatible server among the set of idle servers $\{M_{n+1}, \cdots, M_N\}$, and push the system to state $(M_1, l_1, M_2, l_2, \cdots, M_n, l_i, M_{n+1}, 0)$; otherwise, they join the right end of the queue, pushing the system into state $(M_1, l_1, M_2, l_2, \cdots, M_n, l_n + 1)$. Customers who find several available and compatible servers upon arrival, choose the receiving server randomly according to given assignment probability distributions. Upon finishing their current job, servers will scan the queue from the left and start servicing the first customer they can handle, if one exists. This results in a new system state with the hosting server relocated to the position of the received customer. If no compatible

customers are found, the newly released servers will join the set of other idle servers, transitioning the system to state $(M_1, l_1, \cdots, M_{k-1}, l_{k-1}+l_k, M_{k+1}, \cdots, M_n, l_n)$ where M_k is the newly idle server $(k \leq n)$. We therefore have that the l_k customers waiting between servers M_k and M_{k+1} must belong to $\mathcal{U}(\{M_1, \cdots, M_k\})$. For a detailed description of this system and its transitions see the original paper.

Within the setup summarized above and with $\Pr(\emptyset)$ the probability of an empty system, Visschers et al. (2012) consider the product form distribution

$$\Pr(M_1, l_1, M_2, l_2, \cdots, M_n, l_n) = \Pr(\emptyset) \frac{\Pi_\lambda(\{M_1, \cdots, M_n\})}{\Pi_\mu(M_1, \cdots, M_n)} \prod_{j=1}^{n} \alpha_j^{l_j}, \tag{23}$$

where

$$\alpha_j = \frac{\lambda_{\mathcal{U}(\{M_1, \cdots, M_j\})}}{\mu_{\{M_1, \cdots, M_j\}}}, \quad \text{for } j = 1, 2, \cdots, n,$$

and

$$\Pi_\lambda(\{M_1, \cdots, M_n\}) = \prod_{j=1}^{n} \lambda_{M_j}(\{M_1, \cdots, M_{j-1}\}), \tag{24}$$

for every subset $\{M_1, \cdots, M_n\} \in \mathcal{M}$ of servers and

$$\Pi_\mu(M_1, \cdots, M_n) = \prod_{j=1}^{n} \mu_{\{M_1, \cdots, M_j\}},$$

for every sequence $(M_1, \cdots, M_n) \in \mathcal{P}^n$ with $\mathcal{P}^n$ the set of all permutations of all subsets of size n of $\mathcal{M}$. While $\Pi_\mu(M_1, \cdots, M_n)$ may depend on the order of servers in the sequence $(M_1, \cdots, M_n)$, for the product form solution to exist, Visschers et al. (2012) show that $\Pi_\lambda(\{M_1, \cdots, M_n\}$ must be independent of the order of servers. It is then shown that there exist unique values for the *activation rates* $\lambda_{M_j}(\{M_1, \cdots, M_{j-1}\})$ which satisfy this *assignment condition* and that these unique values can be calculated recursively. The assignment probability distributions leading to these activation rates are often not unique and can be obtained by solving a maximal flow problem for each set $\{M_1, \cdots, M_{j-1}\}$. Finally, the product form (23) implies the probability of having the sequence of busy servers $(M_1, \cdots, M_n)$ as

$$\Pr(M_1, \cdots, M_n) = \Pr(\emptyset) \frac{\Pi_\lambda(\{M_1, \cdots, M_n\})}{\Pi_\mu(M_1, \cdots, M_n)} \prod_{j=1}^{n} \frac{1}{1-\alpha_j}. \tag{25}$$

The assignment condition is trivially satisfied for the queue with partial service where servers are indistinguishable and hence the assignment of available compatible servers to incoming customers is purely random. This allows us to derive the distribution of the number of busy servers of the queue with partial service (not to be confused with class κ in the skill based queue) by evaluating (25). Distinguishing the expressions obtained for the queue with partial service with tildes we write

$$\tilde{\lambda}_{\mathcal{U}\{M_1, \cdots, M_j\}} = \sum_{c=1}^{j} \lambda_c \frac{\binom{j}{c}}{\binom{N}{c}},$$

and

$$\tilde{\mu}_{\{M_1,\cdots,M_j\}} = j\mu\,,$$

which immediately follow from servers being indistinguishable. We then have

$$\tilde{\alpha}_j = \frac{\tilde{\lambda}_{\mathcal{U}(\{M_1,\cdots,M_j\})}}{\tilde{\mu}_{\{M_1,\cdots,M_j\}}} = (j\mu)^{-1}\sum_{c=1}^{j}\lambda_c\frac{\binom{j}{c}}{\binom{N}{c}}\,,$$

and

$$\tilde{\Pi}_\mu(M_1,\cdots,M_n) = \prod_{j=1}^{n} j\mu = n!\mu^n\,.$$

To derive an expression for $\tilde{\Pi}_\lambda(\{M_1,\cdots,M_n\})$, we consider the situation with $j-1$ busy servers and designate one of the idle servers to act as M_j. We then recognize the probability of a class c customer being simultaneously compatible with the designated server as well as exactly h of the busy servers as

$$\phi_c(h,j) = \frac{\binom{j-1}{h}\binom{N-j}{c-h-1}}{\binom{N}{c}}, \quad \begin{array}{l} j\in\mathcal{M} \\ h=\max\{0,c+j-N-1\},\cdots,\min\{c-1,j-1\} \end{array},$$

and the probability of such a customer being assigned to the designated server as $1/(c-h)$. Conditioning on h and interpreting $\tilde{\lambda}_{M_j}(\{M_1,\cdots,M_{j-1}\})$ as the total activation rate of a designated idle server given the set of busy servers $\{M_1,\cdots,M_{j-1}\}$ we get

$$\tilde{\lambda}_{M_j}(\{M_1,\cdots,M_{j-1}\}) = \sum_{c=1}^{N}\lambda_c \sum_{\substack{h=\max\{0,\\ c-(N-(j-1))\}}}^{\substack{\min\{c-1,\\ j-1\}}} \frac{\phi_c(h,j)}{c-h}\,,$$

and then from (24)

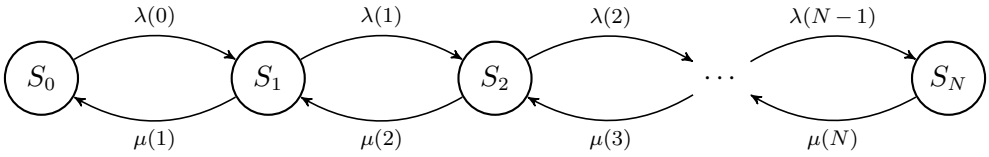
$$\tilde{\Pi}_\lambda(\{M_1,\cdots,M_n\}) = \prod_{j=1}^{n}\sum_{c=1}^{N}\lambda_c \sum_{h=\max\{0,c-(N-(j-1))\}}^{\min\{c-1,j-1\}} \frac{\binom{j-1}{h}\binom{N-j}{c-h-1}}{\binom{N}{c}}\frac{1}{c-h}\,,$$

which of course satisfies the assignment condition. Substituting the above expressions into (25), summing over all members of $\mathcal{P}^n$ for $n=0,\cdots,N$, and simplifying we obtain

$$P_n = P_0\frac{N!}{(N-n)!n!}\prod_{j=0}^{n-1}\sum_{c=1}^{N}\lambda_c \sum_{\substack{h=\max\{0,\\ c+j-N))\}}}^{\min\{c-1,j\}} \frac{c!(N-c)!}{(N+h-c-j)!(j-h)!(c-h)!h!} \times\prod_{j=1}^{n}\frac{j!(N-j)!}{N!j\mu-j!\sum_{c=1}^{j}\lambda_c\frac{N-c}{j-c}}\,, \quad n=0,\cdots,N\,, \quad (26)$$

with the normalizing factor $P_0 = (1+\sum_{j=1}^{N}P_j)^{-1}$. $\square$

Figure 4 The birth-death model to compute state probabilities



REMARK 1. Adan and Weiss (2014) showed that this product form is also valid for the case where the random assignment policy is replaced with the Assign to the Longest Idle Server (ALIS). Consequently, one can derive (26) assuming an FCFS-ALIS queue discipline; however, we find the presented approach more direct and intuitive.

Proof of Theorem 2 A class c customer arriving to the loss system with partial service in state S_n will be lost with probability $q_c(n) = \binom{n}{c}/\binom{N}{c}$ and will enter service immediately with probability $1 - q_c(n)$. The system then can be modeled as a birth-death process shown in Figure 4 with $\mu(n) = n\mu$ for $n = 1, \cdots, N$, and

$$\lambda(n) = \sum_{c=1}^{N} \lambda_c q_c(n) = \sum_{c=1}^{N} \lambda_c \frac{n!(N-c)!}{N!(n-c)!}, \quad n = 0, \cdots, N-1\,.$$

It is then easy to verify that the steady state distribution will be

$$P_n = P_0 \prod_{k=1}^{n} \frac{\sum_{c=1}^{k} \lambda_c (1 - \frac{k!(N-c)!}{N!(k-c)!}) + \sum_{c=k+1}^{N} \lambda_c}{k\mu}, \quad n = 1, \cdots, N\,,$$

with the probability of the empty system $P_0 = (1 + \sum_{n=1}^{N} P_n)^{-1}$ as the normalizing factor. □

REMARK 2. This result also follows from the product form given in Adan et al. (2010) for the loss version of the skill based systems considered in Visschers et al. (2012); the derivation presented here, however, is simpler.

Appendix B: Correction of State Probability Approximations

The accuracy of the distribution of the number of busy servers obtained from the M/M/[N] model with the demand vector $[\lambda_c]$ and service rate μ given by (5) and (6) as parameters can be further improved through the empirical scheme outlined here. We emphasize that although we considered some analytical alternatives which proved to be less effective, the procedure described here is purely empirical and obtained by analyzing a large set of test cases and comparing the outputs of the M/M/[N] and simulation models. The key observation here is that by shifting the distribution of the arrival intensities in the nominal input vector $[\lambda_c]$ towards the lower admission classes (smaller c), we can always find an alternative distribution $[\hat{\lambda}_c]$ for which the state probabilities of the M/M/[N] model will more closely match those of the spatially distributed system. Furthermore,

and as one would expect, the magnitude of the required shift seems to be proportional to the variation of server workloads across the fleet. Therefore, in this method, we capture the variation in the server workloads and construct a suitably altered demand vector $[\hat{\lambda}_c]$ which can then be input to the M/M/[N] model to arrive at a more accurate approximation of the state probabilities of the original spatially distributed system.

We use Algorithm 1 to map any given partial service demand vector $[\lambda_c]$ to a reshaped vector $[\hat{\lambda}_c]$ by moving the demand concentration towards the lower admission classes while keeping the total arrival rate $\lambda = \sum_{c=1}^{N} \lambda_c = \sum_{c=1}^{N} \hat{\lambda}_c$ intact. The parameter $\zeta \in [0,1]$ controls the intensity of the operation; with $\zeta = 0$, we will have $[\hat{\lambda}_c] = [\lambda_c]$ and with $\zeta = 1$, all demand will be moved into the admission class $c = 1$, that is $\hat{\lambda}_1 = \lambda$ and $\hat{\lambda}_c = 0$ for $c = 2, 3, \cdots, N$.

input : $\zeta \in [0,1]$ and λ_c for $c = 1, 2, \cdots, N$
output: $\hat{\lambda}_c$ for $c = 1, 2, \cdots, N$

$\hat{\lambda}_c \leftarrow \lambda_c$, $c = 1, 2, \cdots, N$;
for $i \leftarrow N$ **to** 2 **do**
  $\Delta \leftarrow \zeta \hat{\lambda}_i$;
  for $j \leftarrow 1$ **to** $i - 1$ **do**
    $\hat{\lambda}_j \leftarrow \hat{\lambda}_j + \frac{\Delta}{i-1}$;
  end
  $\hat{\lambda}_i \leftarrow \hat{\lambda}_i - \Delta$;
end

Algorithm 1: Reshaping λ_c with a factor ζ

To measure the variation among server workloads $\rho_1, \rho_2, ..., \rho_N$, we define

$$\eta = \frac{\sum_{j \in \mathcal{J}} |\rho_j - \rho|}{\rho N^{1.3}},$$

where $\rho = \frac{1}{N} \sum_{j \in \mathcal{J}} \rho_j$ is the mean server workload. We have observed through extensive testing that this specific expression for η, makes the formulas proposed here effectively independent of the fleet size N. For a given η, suitable values of ζ can be best predicted using a two-part approximation curve given by

$$\zeta = \begin{cases} b_0 \eta^3 & \text{if } \eta \leq \eta_s \\ \sqrt{\frac{\eta - a_0}{a_2}} & \text{if } \eta \geq \eta_s \end{cases},$$

and plotted in Figure 5 where b_0, a_0, a_2, η_s, and ζ_s are shaping parameters that depend on the nominal loading $\varrho = \lambda / N\mu$ through the following relations:

$$a_0 = \begin{cases} 0.0454\varrho^3 + 0.0510\varrho^2 - 0.3140\varrho + 0.2173 & \text{for systems with queues} \\ 0.1567\varrho^3 - 0.1043\varrho^2 - 0.2442\varrho + 0.2071 & \text{for loss systems} \end{cases}$$

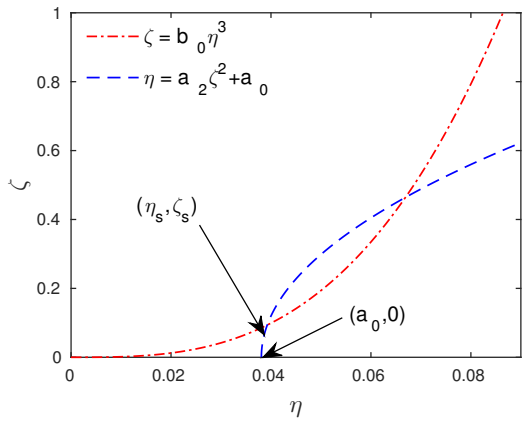


Figure 5 The $\zeta - \eta$ curve

$$\zeta_s = \begin{cases} 0.7262\varrho^2 - 1.761\varrho + 1.341 & \text{for systems with queues} \\ 1.006\varrho^2 - 2.191\varrho + 1.3592 & \text{for loss systems} \end{cases}$$

$$a_2 = \begin{cases} 0.18 & \text{for systems with queues} \\ -0.4419\varrho^3 + 0.3262\varrho^2 - 0.0889\varrho + 0.1878 & \text{for loss systems} \end{cases}$$

$$\eta_s = a_2\zeta_s^2 + a_0$$

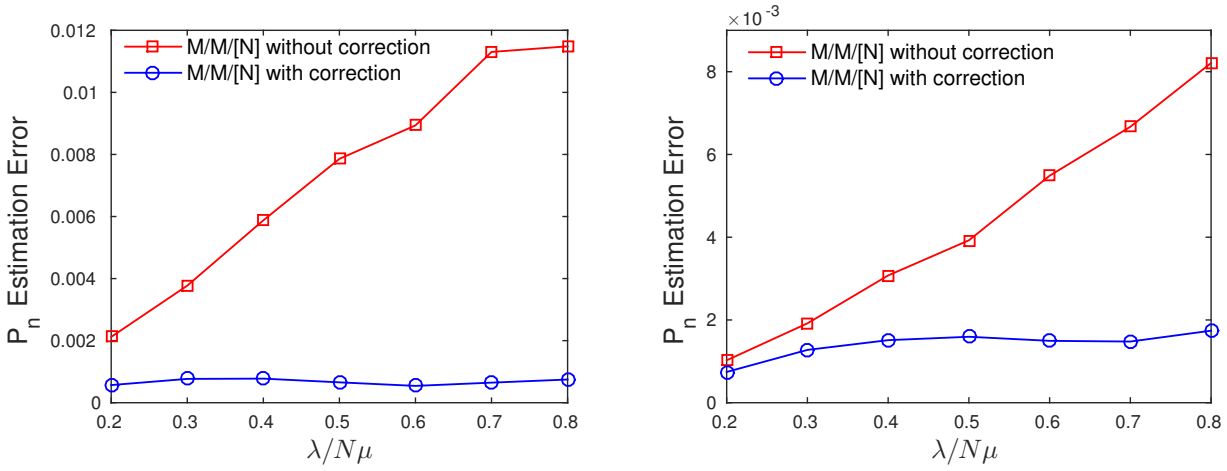
$$b_0 = \zeta_s \eta_s^{-3}.$$

For systems where queues are allowed, this reshaping procedure does not affect the average server utilization ρ as the total demand λ is kept unchanged; however, for loss systems, any alteration of $[\lambda_c]$ through the reshaping algorithm may also change ρ. Therefore, we need to perform a normalization step along with the reshaping procedure to make sure the resulting average server workload matches the current iteration value ρ. This can be achieved by replacing the nominal service rate μ with an adjusted version $\hat{\mu}$ through the following relation

$$\hat{\mu} = \begin{cases} \mu & \text{for systems with queues} \\ (\rho N)^{-1} \sum_{n=0}^{N} P_n \sum_{c=1}^{N} \lambda_c (1 - q_c(n)) & \text{for loss systems} \end{cases},$$

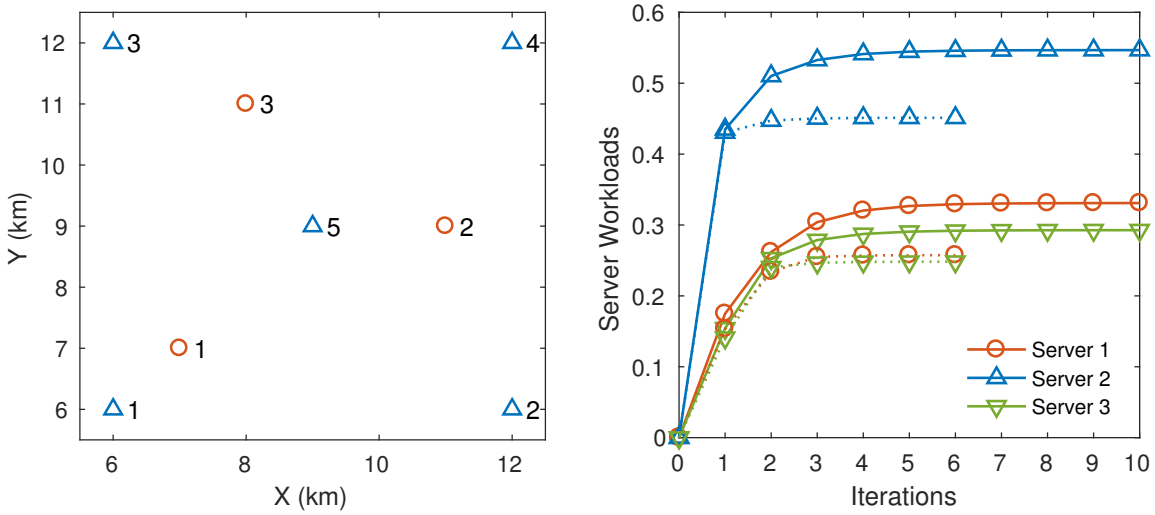
where $q_c(n)$ is given by (8). This normalization step is equivalent to multiplying the input vector $[\lambda_c]$ by a factor of $\mu/\hat{\mu}$.

Finally, the corrected state probabilities are obtained from an M/M/[N] model with the reshaped input vector $[\hat{\lambda}_c]$ and the normalized service rate $\hat{\mu}$ as parameters. Figure 6 shows the average improvement of P_n estimations for queuing and loss systems at varying workloads.



(a) Loss System (b) Queuing System

Figure 6 Comparison of the state probability estimation errors with and without the correction scheme averaged over randomly generated cases with 20 servers ($N = 20$) and varying workloads. The estimation error is computed as $\sum_{n=1}^{N} |P_n^{mod} - P_n^{sim}|$.



(a) Locations of the demand zones (squares) and servers (circles) for the example problem.

(b) Convergence of server workloads for the queuing (solid) and loss (dotted) systems.

Figure 7 Illustrative Example

Appendix C: Illustrative Example

In this section, we apply the presented algorithm to a simple example consisting of five demand zones ($M = 5$) and three servers ($N = 3$) located as shown in Figure 7.a. We assume two priority levels ($K = 2$) with arrival rates and dispatch preference orders given as

$$[\lambda_{ip}] = \begin{bmatrix} 1.0 & 1.0 \\ 0.225 & 0.075 \\ 0.075 & 0.1 \\ 0.05 & 0.2 \\ 0.2 & 0.05 \end{bmatrix}, \quad [b_{ij1}] = \begin{bmatrix} 1\ 0\ 0 \\ 1\ 1\ 0 \\ 1\ 0\ 1 \\ 0\ 1\ 1 \\ 1\ 1\ 1 \end{bmatrix}, \quad [b_{ij2}] = \begin{bmatrix} 1\ 1\ 0 \\ 0\ 1\ 1 \\ 1\ 0\ 1 \\ 0\ 1\ 0 \\ 0\ 1\ 1 \end{bmatrix}, \quad [r_{ij1}] = \begin{bmatrix} 1\ 3\ 2 \\ 2\ 1\ 3 \\ 2\ 3\ 1 \\ 3\ 2\ 1 \\ 1\ 2\ 3 \end{bmatrix}, \quad [r_{ij2}] = \begin{bmatrix} 1\ 2\ 3 \\ 3\ 2\ 1 \\ 2\ 3\ 1 \\ 3\ 1\ 2 \\ 3\ 2\ 1 \end{bmatrix},$$

with the matrix $[c_{ip}]$ implied from $[b_{ijp}]$ as

$$[c_{ip}] = \begin{bmatrix} 1\ 2 \\ 2\ 2 \\ 2\ 2 \\ 2\ 1 \\ 3\ 2 \end{bmatrix}.$$

We assume exponentially distributed service times with three components: travel time from the server location to the scene, on-scene time, and follow-up time. We set on-scene and follow-up times respectively to 20 and 30 minutes regardless of the call priority, and use the procedure detailed in Section 5 to approximate the mean travel times between each server and customer location and for each priority. Euclidean norms were used to measure travel distances. The resulting mean service times are

$$[t^{ser}_{ij1}] = \begin{bmatrix} 58.0\ 61.1\ 60.8 \\ 61.4\ 60.2\ 62.2 \\ 60.1\ 60.5\ 58.2 \\ 62.2\ 59.8\ 60.4 \\ 56.9\ 56.2\ 56.4 \end{bmatrix}, \quad [t^{ser}_{ij2}] = \begin{bmatrix} 61.6\ 66.1\ 65.7 \\ 66.6\ 64.8\ 67.8 \\ 64.7\ 65.3\ 61.9 \\ 67.8\ 64.3\ 65.2 \\ 59.9\ 59.0\ 59.3 \end{bmatrix}.$$

The plots in Figure 7.b show how server workloads converge to their final values as the algorithm progresses when applied to the loss and queuing versions of this example problem. As typically observed, the algorithm takes fewer iterations in the case of a loss system.

The outputs of the model when applied to the example problem as a queuing or loss system are compared with the corresponding values obtained from the simulation model in Tables 8, 9, 10 and 11. We observe accurate approximations to all performance measures in each priority and for both queuing and loss systems. We note the significant asymmetry in the distribution of immediate and delayed dispatches. This is caused by differing dispatch preference orders and partial backups in case of the immediate dispatches, and for the delayed dispatches, can be attributed mainly to partial backups and, probably to a much less degree, to varying service times.

Acknowledgments

This work has been supported by the CREATE program on Health Care Operations and Information Management, by Polytechnique Montral, and by the NSERC Discovery Grants and CIRRELT Scholarship programs. Their support is gratefully acknowledged.

Table 8 Comparison of server workloads estimated by the model and simulation.

	Loss			Queue		
	Server 1	Server 2	Server 3	Server 1	Server 2	Server 3
Simulation	0.2589	0.4513	0.2486	0.3323	0.5453	0.2922
Model	0.2574	0.4512	0.2482	0.3309	0.5467	0.2927

Table 9 Comparison of immediate dispatch rates estimated by the model and simulation.

	Priority 1			Priority 2		
Demand Zone	Server 1	Server 2	Server 3	Server 1	Server 2	Server 3
1	1 (1)	—	—	.8462 (.8543)	.1538 (.1457)	—
2	.4187 (.4179)	.5813 (.5821)	—	—	.5674 (.5634)	.4326 (.4366)
3	.1797 (.1892)	—	.8203 (.8108)	.1797 (.1891)	—	.8203 (.8109)
4	—	.5674 (.5659)	.4326 (.4341)	—	1 (1)	—
5	.1105 (.1097)	.5047 (.5018)	.3848 (.3885)	—	.5674 (.5634)	.4326 (.4366)

Table 10 Comparison of delayed dispatch rates estimated by the model and simulation.

	Priority 1			Priority 2		
Demand Zone	Server 1	Server 2	Server 3	Server 1	Server 2	Server 3
1	1 (1)	—	—	.5297 (.5436)	.4703 (.4564)	—
2	.4800 (.4821)	.5200 (.5179)	—	—	.4090 (.4109)	.5910 (.5891)
3	.4667 (.4681)	—	.5333 (.5319)	.4331 (.4460)	—	.5669 (.5540)
4	—	0.4862 (.4877)	.5138 (.5123)	—	1 (1)	—
5	.3032 (.3043)	.3317 (.3297)	.3652 (.3660)	—	.4090 (.4116)	.5910 (.5884)

Table 11 Comparison of average waiting times (in minutes) and fractions of calls queued estimated by the model and simulation for the system with queues.

	Waiting Times		Fraction of Calls Queued	
Demand Zone	Priority 1	Priority 2	Priority 1	Priority 2
1	23.5 (22.8)	12.9 (12.4)	.3309 (.3323)	.2093 (.2192)
2	8.3 (8.0)	9.6 (9.0)	.2201 (.2182)	.2010 (.1939)
3	4.8 (4.5)	6.3 (5.9)	.1378 (.1281)	.1378 (.1271)
4	6.7 (6.4)	54.6 (49.6)	.2010 (.1951)	.5467 (.5446)
5	2.6 (2.4)	9.6 (8.9)	.1017 (.0950)	.2010 (.1929)

References

Adan, Ivo, Cor Hurkens, Gideon Weiss. 2010. A reversible Erlang loss system with multitype customers and multitype servers. *Probability in the Engineering and Informational Sciences* **24**(04) 535–548.

Adan, Ivo, Gideon Weiss. 2014. A skill based parallel service system under FCFS-ALIS—steady state, overloads, and abandonments. *Stochastic Systems* **4**(1) 250–299.

Birge, John R, Stephen M Pollock. 1989. Using parallel iteration for approximate analysis of a multiple server queueing system. *Operations Research* **37**(5) 769–779.

Budge, Susan, Armann Ingolfsson, Erhan Erkut. 2009. 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, Susan, Armann Ingolfsson, Dawit Zerom. 2010. Empirical analysis of ambulance travel times: the case of Calgary emergency medical services. *Management Science* **56**(4) 716–723.

Burwell, Timothy H, James P Jarvis, Mark A McKnew. 1993. Modeling co-located servers and dispatch ties in the hypercube model. *Computers & Operations Research* **20**(2) 113–119.

de Souza, Regiane Máximo, Reinaldo Morabito, Fernando Y Chiyoshi, Ana Paula Iannoni. 2015. Incorporating priorities for waiting customers in the hypercube queuing model with application to an emergency medical service system in Brazil. *European Journal of Operational Research* **242**(1) 274–285.

Galvao, Roberto D, Reinaldo Morabito. 2008. Emergency service systems: The use of the hypercube queueing model in the solution of probabilistic location problems. *International Transactions in Operational Research* **15**(5) 525–549.

Goldberg, Jeffrey, Ferenc Szidarovszky. 1991. Methods for solving nonlinear equations used in evaluating emergency vehicle busy probabilities. *Operations research* **39**(6) 903–916.

Iannoni, Ana Paula, Fernando Chiyoshi, Reinaldo Morabito. 2015. A spatially distributed queuing model considering dispatching policies with server reservation. *Transportation Research Part E: Logistics and Transportation Review* **75** 49–66.

Iannoni, Ana Paula, Reinaldo Morabito. 2007. A multiple dispatch and partial backup hypercube queuing model to analyze emergency medical systems on highways. *Transportation research part E: logistics and transportation review* **43**(6) 755–771.

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

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

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

Larson, Richard C. 2013. Hypercube queueing model. *Encyclopedia of Operations Research and Management Science*. Springer, 733–739.

Larson, Richard C, Mark A Mcknew. 1982. Police patrol-initiated activities within a systems queueing model. *Management Science* **28**(7) 759–774.

Statistics Canada. 2016a. Census profile of dissemination areas for the 2011 census. `http://www12.statcan.gc.ca/census-recensement/2011/dp-pd/prof/index.cfm?Lang=E` accessed on July 4, 2017.

Statistics Canada. 2016b. Dissemination areas cartographic boundary file for the 2011 census. `http://www12.statcan.gc.ca/census-recensement/2011/geo/bound-limit/bound-limit-2011-eng.cfm` accessed on July 4, 2017.

Visschers, Jeremy, Ivo Adan, Gideon Weiss. 2012. A product form solution to a system with multi-type jobs and multi-type servers. *Queueing Systems* **70**(3) 269–298.