http://dx.doi.org/10.1287/ics.2017.XXXX

15th INFORMS Computing Society Conference
Austin, Texas, January 15–17, 2017
pp. 000–000

An Expected Coverage Model with a Cutoff Priority Queue

Soovin Yoon and Laura Albert McLay
Industrial&Systems Engineering Department, University of Wisconsin-Madison, Madison, WI 53706, USA, {yoon57@wisc.edu, laura@engr.wisc.edu}

Abstract

Keywords coverage model, cutoff priority queue, hypercube approximation

1. Introduction

We specifically consider an emergency medical system with multiple servers and different priority classes of patient calls. In our model, a cutoff level of s_1 is given for admission of low priority customers, so that low priority calls are delayed or rejected whenever s_1 or more servers are busy. As a result we reserve servers available for high priority calls and achieve improved high priority call coverage.

We model the problem as a Mixed Integer Linear Programming embedded with the approximate Hypercube queuing model. The MILP produces deployment and dispatching policies but it needs server availability parameters as inputs. The approximate Hypercube model receives the resulted deploy and dispatch decisions from MILP and provides the server busy probability to MILP. Two procedures alternatively iterate until they converge.

Church and ReVelle [5] proposed a coverage model for the ambulance location problem. Daskin [6] introduced one of the first probabilistic models by assuming that each ambulance has the same busy probability of being unavailable for service. For more extensive review on ambulance location models, we refer the readers to the review article by Brotcorne *et al.* [4].

Such iterative framework was introduced by Batta *et al.* [2]. More recently, Ansari *et al.* [1] develops a maximum expected coverage model that incorporates travel time uncertainty as well as contiguity and equity considerations.

Another stream of research on emergency units are analytical tools that computes server busy probabilities given the location plan. Larson develops an exact Hypercube model to study the behavior of servers([8]), and also provides the way to approximate the procedure by computing correction factors([9]).

Meanwhile, cutoff priority model was first formulated by Benn [3] and detailed in Jaiswal [7] by analyzing many variants and their solutions. Taylor and Templeton [12] improved on Jaiswal's derivation on cutoff priority model and solved balance equations to derive steady-state probabilities and waiting time distributions. They mainly investigated two versions of the model: the SCQQ model where both of high and low priority customers are queued and the SCQL model where high priority customers are lost and only low priority customers are queued. They also applied the model to urban ambulance systems with two classes of emergency calls and routine calls. These models were all concerned about two priority

classes, and then Schaack and Larson [11] took a decomposition approach to solve the system for any number of priority classes.

This paper makes the following contributions. First, we derive the approximate hypercube queuing model with the cutoff priority queue that is computationally easy to implement and solve. Once embedded within the optimization model, this hypercube queuing procedure provides explicit expressions for server busy probabilities and correction factors that assist the model to get better approximations. Alternatively iterating the hypercube queuing model and the optimization model is a way to involve stochasticity of the problem without being intractable due to non-linearity. Next, by solving the iterative procedure with the real-world dataset, it is shown that setting the cutoff can improve the coverage over high priority calls. We also provide the decision maker the tradeoff between the amount of coverage improvement and the probability losing/delaying the calls when cutoff is adjusted.

The remainder of this paper is organized as follows. Section 2 defines the notation and describes the approximate Hypercube model with cutoff priority queue. In section 3 MILP model is proposed. We describe the dataset and present computational results in section 4.

2. Approximate Hypercube Model for Cutoff Priority Queue

In this section, we propose a Hypercube model approximation for cutoff priority queue. The Hypercube model can function as a standalone program for analysis of characteristics of cutoff priority queue. Also, the model iteratively interacts with MILP model described in the next section to assist decision-making for deployment and dispatch of ambulances under uncertainty in their availability. The procedure introduced in this section extends the iterative procedure given by [9] as well as Jarvis(1985) to approximate equilibrium behavior and performance measures for the Hypercube model. Two additional features are implemented; calls are distinguished not only by their origins but also by their priorities, and low priority calls are cut off to reserve servers for future high priority call arrivals.

A system with s distinguishable servers and high(low) priority calls from the set of geographical locations J is considered. High(low) priority calls arrive independent of busy status of servers, at a mean rate $\lambda^H(\lambda^L)$ per hour with exponential inter-arrival times. The average service time for call from j served by server i is τ_{ij}, which includes the travel time from the station to the scene, the time spent at the scene, the time to transport the patient to the hospital and return to the station. Servers are distributed among the set of open stations I^{open}, with one server at a station. Exactly one server is assigned to a call immediately according to a fixed preference list, which is an ordered list of open stations, as long as there is any available server. In the preference list $\{b^H_{jk}\}(\{b^L_{jk}\})$,the kth preferred station by the high(low) priority call from node j is given as $b^H_{jk}(b^L_{jk}) \in I^{open}$(for example, if $b^H_{12} = 3$ and $b^L_{12} = 2$, then the second preferred server by the call from node 1 is 3 if the call is high priority and it is 2 if the call is low priority).

A cutoff level of s_1 is given for admission of low priority customers. This means that low priority calls are placed in a queue or lost whenever s_1 or more servers are busy, instead of being served immediately. High priority calls are immediately served by the most preferred available server and either put in a queue or lost only if all servers are busy. This implies that the cutoff level for high priority customers is set to s by convention([11]). The service is assumed to be nonpreemptive.

We consider two types of model that differ by the queue capacity. Under the assumption of Loss model with $0-$ capacity queue $(M/M/s/s)$, high priority calls arriving when all servers are busy or low priority calls arriving when there are more than or equal to s_1 busy servers are "lost to the system", as the queue has zero capacity. Lost calls are considered as being served by external resources. It is common in the EMS literature to assume the system with no queue, and it is also consistent with the real-world operation(Budge et al. 2009). On the other hand, in the Queued model where the queue is assumed to have $\infty-$

capacity($M/M/s/\infty$), calls are put in a queue when they cannot be served immediately, so that they can be served later on a first-come-first-served basis. As the queue is infinitely large, backlogged calls are never lost as they are eventually served in the long run.

Table 1 summarizes symbols used frequently. The Hypercube queuing model requires deployment and dispatch policies as inputs, which can be generated either externally(for example, send-the-closest) or by solving the MILP in the next section. Other parameters needed such as service times and arrival rates can be collected from the data. The outputs of the model are correction factors Q_k for $k = 0, \cdots, s-1$ and server busy probabilities r_i for $i = 1, \cdots, s$. Then the dispatch probability f^p_{jk}, which is the probability that a priority p call from node j is served by its kth preferred server, can be approximated as

$$f^p_{jk} = Q^p_{k-1} \Pi^{k-1}_{l=1} r_{b^p_{jl}} (1 - r_{b^p_{jk}}). \tag{1}$$

for all $j \in J$ and $k = 1, \cdots, s$. If servers are independent, the probability that the kth preferred server is dispatched is simply $\Pi^{k-1}_{l=1} r_{b^p_{jl}} (1 - r_{b^p_{jk}})$, but the independence assumption is erroneous. Therefore correction factors Q_ks are needed to improve this approximation.

STEP 0 Given: λ^p_j, τ_{ij} from inputs, open stations I^{open} and preference list b_{jk}
Initialize:

$$\tau^0 = \sum_{j \in J} \sum_{i \in I^{open}} \sum_{p \in \{H,L\}} \tau_{ij} \frac{\lambda^p_j}{\lambda^p} \tag{2}$$

$$\rho = \frac{\lambda \tau}{s} \tag{3}$$

STEP 1 Update steady-state probabilities P_i.
For Loss model, use

$$\begin{aligned} P_i &= P_0 \frac{s^i \rho^i}{i!}, \qquad 0 \le i \le s_1 - 1 \\ &= P_0 \frac{s^i \rho^i}{i!} \left(\frac{\lambda^H}{\lambda}\right)^{i - s_1}, \qquad s_1 \le i \le s \end{aligned} \tag{4}$$

TABLE 1. Summary of symbols.

symbol	description
$j \in J$	set of call nodes
$I \in I$	set of potential stations
$i \in I^{open}$	set of open stations, $I^{open} \subset I$
$p \in \{H, L\}$	set of call priorities
s	total number of servers
s_1	cutoff level for low priority calls
ρ	server utilization
r_i	busy probability of server at station i
r	system-wide server busy probability, $r = \sum_i r_i / s$
P_i	steady-state probability
q_k	correction factor
f_{jk}	probability of dispatching kth preferred server to call nodes j
b_{jk}	the station where the server kth preferred by call node j is located
λ^p_j	arrival rate of priority p call from node j
$\lambda^H (\lambda^L)$	arrival rate of high(low) priority calls
λ	system-wide call arrival rate, $\lambda = \lambda^H + \lambda^L = \sum_{j \in J} \sum_{p \in \{H,L\}} \lambda^p_j$
τ_{ij}	mean service time for calls from node j served by a server from station i
τ	system-wide mean service time

where

$$P_0 = \Big(\sum_{i=0}^{i=s1} \frac{s^i\rho^i}{i!} + \sum_{i=s1+1}^{s} \frac{s^i\rho^i}{i!}\big(\frac{\lambda^H}{\lambda}\big)^{i-s_1}\Big)^{-1}. \tag{5}$$

For Queued model, denoting $\xi_1 = \lambda_H\tau$, $\xi_2 = \lambda_L\tau$, and $\xi = \lambda\tau$, use

$$P_i = P_0\frac{\xi^s}{s!}, \qquad 0 \le i \le s_1 - 1 \tag{6}$$

$$= P_0(\xi^{s_1}\xi_1^{i-s_1}/i!)s_1/(s_1 - \xi_2 S_Q(s_1, s)), \qquad s_1 \le i \le s \tag{7}$$

where

$$P_0 = \{\sum_{i=0}^{s_1-1} \xi^i/i! + (\xi^{s_1}/(s_1-1)!)S_Q(s_1,s)/[s_1 - \xi_2 S_Q(s_1,s)]\}^{-1}, \tag{8}$$

$$S_Q(k,s) = \xi_1^{-k}k![\sum_{i=k}^{s}\rho_H^{i-k}/i! + (\xi_1^s/s!)s/(s-\xi_1)]. \tag{9}$$

STEP 2 Update correction factors Q_k^H and Q_k^L by

$$Q_k^H = \frac{\sum_{j=k}^{s-1}\frac{(s-k-1)!(s-j)j!}{(j-k)!s!}P_k}{w^k\rho^k\big(1-(w\rho)\big)}, \tag{10}$$

and

$$Q_k^L = \frac{\sum_{j=k}^{s_1-1}\frac{(s-k-1)!(s-j)j!}{(j-k)!s!}P_k}{w^k\rho^k\big(1-w\rho\big)}. \tag{11}$$

For Loss model, use

$$w = (1 - P_s\frac{\lambda^H}{\lambda} - \sum_{i=s_1}^{s} P_i\frac{\lambda^L}{\lambda}) \tag{12}$$

and $w = 1$ for Queued model.
STEP 3 Update server busy probabilities r_i as

$$r_i = \frac{V_i}{V_i + 1}. \tag{13}$$

where

$$V_i = \sum_{i\in I^{Open}}\sum_{k\in K}\sum_{p\in\{H,L\}} \lambda_j^p\tau_{b_{jk}^p j}q_k^p(\Pi_{l=1}^{k-1}r_{b_{jl}^p}) \tag{14}$$

STEP 4 Update system-wide service time τ as

$$\tau = \sum_{i\in I^{open}}\sum_{j\in J}\tau_{ij}\frac{\lambda_j^p}{\lambda}\frac{f^p_{j,a_{ij}^p}}{\sum_{k\in K}f_{jk}^p} \tag{15}$$

where f_{jk}^H and f_{jk}^L follows

$$f_{jk}^p = Q_{k-1}^p(1 - r_{b_{jk}^p})(\Pi_{l=1}^{k-1}r_{b_{jl}^p}) \tag{16}$$

STEP 5 Update $\rho = \frac{\lambda\tau}{s}$ and $r = w\rho$.
Normalize r_i by

$$r_i \leftarrow \frac{r_i}{\sum_i r_i/s}r \tag{17}$$

Check if max change in r_i is below the predetermined threshold. If yes, terminate and return r_i and Q_k^p. If no, go back to step 1.

3. Maximum Expected Coverage Model

In this section, a mixed integer linear program model that simultaneously locate servers at stations and dispatch servers to call nodes is introduced. The goal of the model is to maximize expected high priority coverage over the region. The system of concern is the same as the previous section. A call is considered to be covered if a server is dispatched to the call immediately upon the arrival and it reaches the call in the predetermined time limit(e.g. 9 minutes).

Define the set of binary variable $y_i = 1(0), i \in I$ if a server is (not) located at station i. This variable set represents deployment decisions by defining the set of open stations. Also define the set of Bernoulli variables $x^p_{ijk} = 1(0), i \in I, j \in J, k = 1, \cdots, s$ if it is (not) the kth preferred station for a priority p call from node j. This variable set captures the dispatch policy. Note that $\{x^p_{ijk}\}$ is basically a binary representation of the preference list $\{b^p_{jk}\}$. Once the MILP problem is solved, the preference list b^p_{jk} is created from the optimal value of x^p_{ijk} by Algorithm 1 in the appendix.

The mixed integer linear program is formally stated below.

$$\max \sum_{i=1}^{s} \sum_{j \in J} \sum_{k=1}^{s} c_{ijk} x^H_{ijk} \tag{18}$$

subject to

$$\sum_{i \in I} y_i = s, \quad y_i \in \{0,1\} \tag{19}$$

$$\sum_{k=1}^{s} x^p_{ijk} = y_i \qquad \forall i, j, p \tag{20}$$

$$\sum_{i \in I} x^p_{ijk} = 1 \qquad \forall j, k, p \tag{21}$$

$$\sum_{j \in J} \sum_{k=1}^{s} \sum_{p \in \{H,L\}} \lambda^p_j q_k (1-r) r^{k-1} \tau_{ij} x^p_{ijk} \le (r+\delta) y_i, \qquad \forall i \in I \tag{22}$$

$$\sum_{j \in J} \sum_{k=1}^{s} \sum_{p \in \{H,L\}} \lambda^p_j q_k (1-r) r^{k-1} \tau_{ij} x^p_{ijk} \ge (r-\delta) y_i, \qquad \forall i \in I \tag{23}$$

$$x^p_{ij'1} \ge x^p_{ij1}, \quad \forall p \in \{H,L\}, j \in J, i \in I, j' \in N_{ij} \tag{24}$$

where

$$c_{ijk} = q_{k-1}(1-r) r^{k-1} R_{ij} \frac{\lambda^H_j}{\lambda^H}. \tag{25}$$

The objective function in (18) represents the expected coverage over high priority calls. The coefficient set c_{ijk} used in the objective function is detailed in (25), and they represents explicit considerations for uncertainty in the ambulance unavailability and the travel time. Ambulance unavailability is captured by introducing r, the system-wide busy probability. Correction factor q_k, which is the output from previous Hypercube model, yields approximation to actual queuing probabilities that compensate unrealistic independence assumption between the servers. Non-deterministic travel time is also implemented by adopting a parameter R_{ij}, which represents the probability that an ambulance located at station i may reach the node j within the time limit. R_{ij} is naturally a decreasing function of the distance between the station i and node j. Putting the parameters together, $q_{k-1}(1-r)r^{k-1}R_{ij}$ captures the coverage of the server at station i over high priority calls from node j when i is the kth preferred server by high priority call at j. And then this value is weighted by its relative call volume proportion λ^H_j/λ^H to shape the coefficient c_{ijk}, so that the objective function can represents the expected coverage over the entire region.

A station is considered to be opened if a server is located, and there are exactly s open stations which is guaranteed by (19). (20) ensures that a server from station i is dispatched only when the station i is open. Also, (21) enforces every call node to have an ordered, non-overlapping preference list.

(22) and (23) are balancing constraints that sets upper and lower limit for server utilization. The server utilization is kept close to average server utilization within the threshold of δ for all servers. This constraint set has two implications. First, the program uses common busy probability r instead of $r_i, i \in I^{open}$ with an implicit assumption that the busy probability is the same for all ambulances. This assumption is reasonable as the load balancing is approximately guaranteed by (22) and (23) to keep the server utilization equal. These constraints are also meaningful in terms of the equity from the service provider's perspective. They ensures that paramedics and emergency medical technicians who staff each server avoid exposures to distress by excessive workload but also have opportunities to practice their skills [10].

(22) and (23) are valid for the Loss model. For Queued model, following constraints may be used instead.

$$\sum_{j\in J}\sum_{k=1}^{s}\sum_{p\in\{H,L\}} \lambda_j^p q_k (1-r) r^{k-1} \tau_{ij} x_{ijk}^p + \frac{\lambda\tau P_D}{s} \leq (r+\delta) y_i \tag{26}$$

$$\sum_{j\in J}\sum_{k=1}^{s}\sum_{p\in\{H,L\}} \lambda_j^p q_k (1-r) r^{k-1} \tau_{ij} x_{ijk}^p + \frac{\lambda\tau P_D}{s} \geq (r-\delta) y_i \tag{27}$$

where the parameter $P_D = P_s \frac{\lambda^H}{\lambda} + \sum_{i=s_1}^{s} P_i \frac{\lambda^L}{\lambda}$ stands for the probability of a call being delayed. The additional term $\frac{\lambda\tau P_D}{s}$ represents the workload of serving delayed calls in the queue. P_i for $i = 1, \cdots, s$ can be achieved from the Hypercube model, along with correction factors and server busy probabilities.

(24) refers to the contiguity restriction, which enforces the program to assign the first priority in a geographically reasonable sense. When the first preferred station for high priority calls from node j is station i, and there is a node j' is a neighborhood of j and closer to i than j, then constraint set (24) enforces i to be the first preferred station for a high priority call from node j.

Finally, the cutoff priority queue scheme is brought to the MILP model through correction factors. q_k^L, which is an output from the Hypercube model, is set to zero for all $k \geq s_1$. In this program, when i is the kth preferred server by priority p call from node j, the probability of dispatching a server i to priority p call from node j is computed as $q_{k-1}^p (1-r) r^{k-1}$. Therefore, the probability of dispatching kth preferred servers to low priority calls are always zero when $k \geq s_1$.

The system-wide server busy probability r is used as a parameter in objective coefficients as well as load balancing constraints. However, this is an endogenous value in that the decision of deployment and dispatching itself affects the busy probability of servers. Nevertheless, in pursuit of problem tractability, we want to avoid expressing r as a function of decision variable y_i and x_{ijk}^p, because doing so would make the mixed integer program non-linear.

In order to overcome this complication, an iterative scheme is introduced. As aforementioned in the previous section, the Hypercube model is utilized to generate the correction factors q_k and server busy probabilities r used as inputs for MILP model. Then the solutions from the MILP model become new inputs for the Hypercube model that updates the value of correction factors and server busy probabilities. Again, those outputs are utilized as inputs for the MILP model. This iterative procedure continues until outputs converges so that the change in server busy probability drops under the threshold.

TABLE 2. Iterative MILP-Hypercube Procedure

STEP 0 **Initialize**:
Generate the initial preference lists b_{jk}^p from the *send-the-closest-available-server* rule.
Set correction factors $q_k^p = 1$ for all $k = 1, \cdots, s$, server busy probability $r = \rho = \frac{\lambda\tau}{s}$, where system-wide mean service time τ set as $\tau = \lambda^{-1} \sum_i \sum_j \sum_p \lambda_j^p \tau_{ij}$. Set imbalance tolerance $\delta = 1.0$.
STEP 1 Solve the MILP model with inputs r, r_i, q_k^p to update preference lists b_{jk}^p and set of open stations I^{open}.
STEP 2 Solve the Hypercube model with inputs b_{jk}^p and I^{open} to update server busy probabilities r,r_i and correction factors q_k^p.
STEP 3 Check termination criteria. Stop if (1) the imbalance measure between r_i is less than the predetermined threshold or if (2) MILP was infeasible. If both termination criteria are not satisfied, go back to **STEP 1**.

Table (2) details the steps in iterative Algorithm that comprehensively describes the iterative use of Hypercube model and MILP model. For more explanation on guaranteed convergence of this procedure, see [1].

4. Computational Results

This section reports analytical and simulation results that illustrates effectiveness of introducing the cutoff priority queue scheme for urban emergency medical service system. The Loss model is considered mainly, but computational results for the Queued model are also introduced for comparison.

4.1. Computational Setting

The model is investigated by using real-world data set from Hanover County, Virginia. The data describes a geographical region with 16 potential stations, 270 aggregated demand nodes with 139 2×2 mile cells and 131 1×1 mile cells. Call arrival rate during the weekdays 12PM to 6PM was selected for demand mean. The demand volume over the region is visualized in Figure 1 as a colored cells on a grid that darker color represents higher call volume. The call volume over the area is not uniform, as it is concentrated in the middle and south-east area of the county while the call is relatively scarce on the west side. The location of 16 potential stations are also marked in the map.

In pursuit of investigating the effect of reserving capacities in the large fleet system, a scaled-up version of Hanover County dataset is also used. The number of server was increased from 5 to 16, and the demand volumn was scaled up by a factor of 5.9. Note that the demand wasn't scaled up by the ratio of fleet size(3.2=16/5), but instead the scale parameter was selected so that the system show the similar coverage performance to the original 5-server case when no server is reserved.

The iterative model was implemented in Python 3.4, and Gurobi 6.5.0 is used as an solver for the Mixed Integer Linear Program. The running time for the iterative MILP-Hypercube procedure using Hanover County data were 30 seconds in average including the time spend for running Hypercube models less than 0.26 second for all cases. The runtime for scaled data with large fleet was 39.65 seconds in average with Hypercube model runtime always less than 1.71 seconds. The load imbalance tolerance threshold was set to 2 percent for the original dataset and 15 percent for the scaled dataset, and server busy probability imbalance tolerance was set to 1 percent for both.

FIGURE 1. Demand Distribution and Station Location on the Hanover County Map.

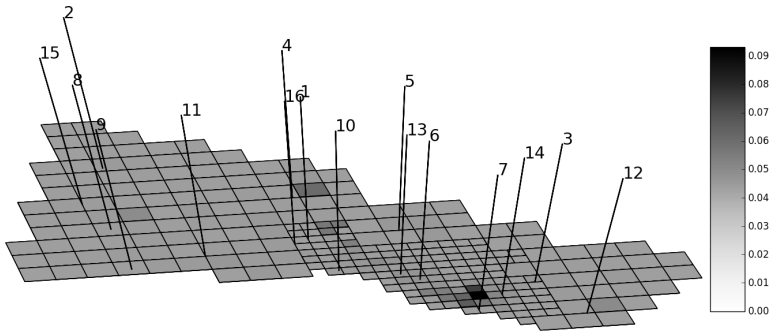


FIGURE 2. Coverage over High Priority Calls Using Different Cutoff Numbers in Hanover County dataset.

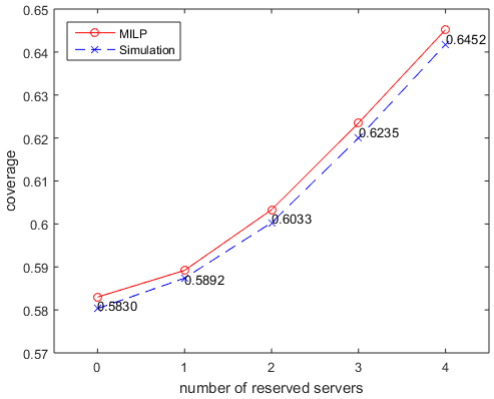
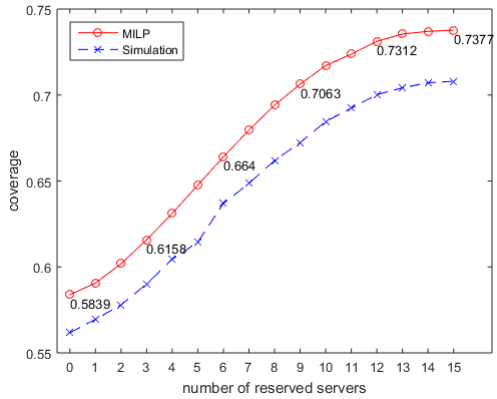


FIGURE 3. Coverage over High Priority Calls Using Different Cutoff Numbers in Scaled Hanover County dataset with Large Fleet.



Throughout the section, we compare analytical results to discrete-event simulation results. The purpose of evaluating the result by simulation is to confirm the validity of cutoff model as well as iterative MILP-Hypercube procedure. The simulation used the same demand distribution and geographical data as the analytical model, and the resulted deployment and dispatch decisions from the analytical model were also used as policy inputs for the simulation. The program was implemented in Python 3.4, ran until 100,000 call arrives and repeated 30 times to generate summary statistics for dispatch probabilities, server busy probabilities, steady-state probabilities and coverage over the high priority calls. The computational time for the simulation were 100.51 seconds in average for the original dataset and 164.85 seconds for the scaled dataset. Note that the runtime of the simulation is significantly longer than the runtime for Hypercube model that serves the similar function, while they both serve the function of evaluating the system given preference lists. In pursuit of precisely getting statistics from rare events, the simulation should be run for a long time, which highlights the another benefit of developing an analytical model like the Hypercube model.

4.2. Coverage Improvement

The resulted coverages over the high priority calls are displayed. Figure 2 show the resulted expected coverage over high priority calls from MILP model and Simulation with 5 servers. The coverage is monotonic increasing in number of reserved servers. Figure 3 was drawn over a scaled-up version of Hanover County dataset. The effect of capacity reservation on

TABLE 3. High Priority Dispatch Probabilities for Different Cutoff Numbers.

	Hypercube					Simulation				
	$s_1=5$	$s_1=4$	$s_1=3$	$s_1=2$	$s_1=1$	$s_1=5$	$s1=4$	$s_1=3$	$s_1=2$	$s_1=1$
k=1	0.6696	0.6774	0.6975	0.7365	0.7909	0.6685	0.6775	0.6994	0.7366	0.7900
k=2	0.2001	0.2022	0.2035	0.1952	0.1678	0.1961	0.1976	0.1973	0.1877	0.1615
k=3	0.0691	0.0698	0.0657	0.0509	0.0308	0.0706	0.0711	0.0661	0.0542	0.0357
k=4	0.0276	0.0279	0.0220	0.0115	0.0070	0.0298	0.0294	0.0244	0.0145	0.0088
k=5	0.0127	0.0128	0.0063	0.0033	0.0020	0.0136	0.0143	0.0080	0.0044	0.0026
P(lost)	0.0209	0.0100	0.0049	0.0026	0.0016	0.0213	0.0101	0.0049	0.0025	0.0015

TABLE 4. Low Priority Dispatch Probabilities for Different Cutoff Numbers.

	Hypercube					Simulation				
	$s_1=5$	$s_1=4$	$s_1=3$	$s_1=2$	$s_1=1$	$s_1=5$	$s1=4$	$s_1=3$	$s_1=2$	$s_1=1$
k=1	0.6696	0.6648	0.6289	0.5141	0.2795	0.6697	0.6654	0.6283	0.5150	0.2826
k=2	0.2001	0.1896	0.1504	0.0741	0.0000	0.2003	0.1886	0.1510	0.0746	0.0000
k=3	0.0691	0.0572	0.0280	0.0000	0.0000	0.0682	0.0569	0.0285	0.0000	0.0000
k=4	0.0276	0.0152	0.0000	0.0000	0.0000	0.0278	0.0153	0.0000	0.0000	0.0000
k=5	0.0127	0.0000	0.0000	0.0000	0.0000	0.0125	0.0000	0.0000	0.0000	0.0000
P(lost)	0.0209	0.0729	0.1923	0.4107	0.7203	0.0213	0.0737	0.1926	0.4103	0.7175

TABLE 5. System-wide Server Busy Probability.

	Hypercube					Simulation				
	$s_1=5$	$s_1=4$	$s_1=3$	$s_1=2$	$s_1=1$	$s_1=5$	$s1=4$	$s_1=3$	$s_1=2$	$s_1=1$
r	0.3304	0.3226	0.3025	0.2635	0.2091	0.3300	0.3221	0.3018	0.2624	0.2067

high-priority call coverage is even more clear in this large fleet system. Specifically, the marginal improvement was largest with 1.65 percent when the number of reserved server was increased from 4 to 5. Considering the fact that the maximum achievable coverage of this system is only 81.28 percent (which is gained under the impractical assumption that every call is served immediately by the server located at the most reachable station), the improvement from 63.11 to 64.7 percent by reserving one additional server is significant.

Lastly, in a model validation context, the analytical results were compared to the simulation results. The dotted lines in figure 2 and 3 shows the simulation result that corresponds to the analytical result. The coverage levels for all cutoff numbers were very close between analytical and simulation in the original dataset with differences less than 0.35 percent in the original dataset. The gap was larger in the large fleet system, but the increasing trend appeared to be very similar.

The resulted coverage improvement is a collaboration of two factors. First, by reserving more servers for high priority call arrivals, the dispatch probability of sending highly preferred server increases. Also, with different correction factor and server busy probability given, the MILP program produces different deployment and dispatch decisions that is optimized for given cutoff number.

Dispatch probabilities for the original Hanover County dataset with 5 servers are shown in left half columns of Table 3 and 4. Dispatch probabilities for higher k ($k=1,2$) for high priority calls increase as the system reserve more servers(lower cutoff number s_1) that result in smaller server busy probability as seen in Table 5. Dispatch probabilities for $k > s_1$ for low priority calls are all zero, which is the direct result of cutoff priority queue discipline.

As we set the cutoff s_1 lower, the probability to dispatch 1st preferred server to high priority calls gets larger, which result in coverage improvement. However, this involves a cost of losing more low priority calls. At the base case of reserving no server, the probability of losing a low priority call is as low as 2.09 percent, and by reserving one server it increases into 7.29 percent, while resulting a coverage improvement of 0.65 percent. As there exist this trade-off between improving high priority coverage by reserving server and staying

FIGURE 4. First-Priority District design for High priority calls(left) and Low priority calls(right) in Hanover County dataset.

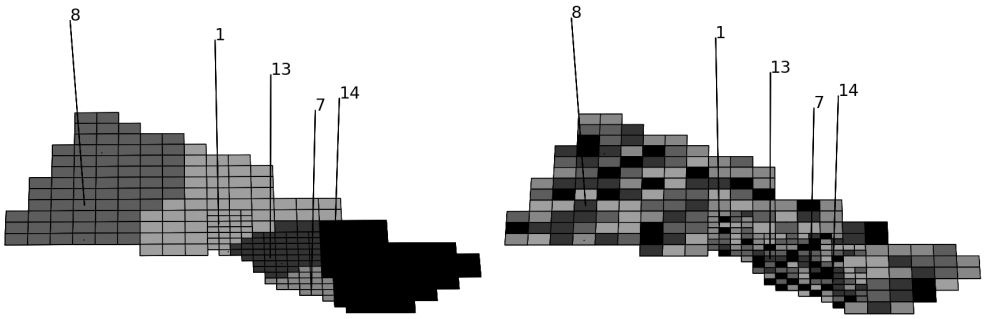
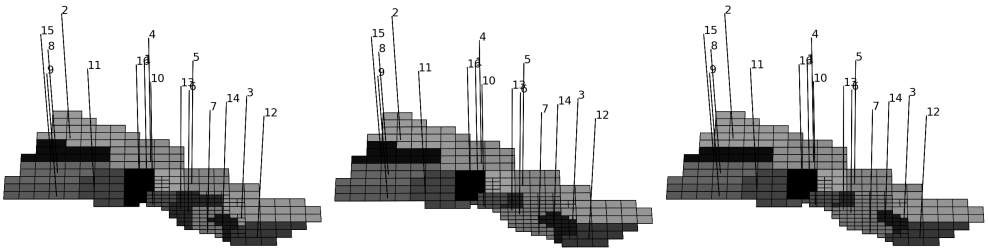


FIGURE 5. First-Priority District design for high priority calls with $s_1 = 2, 8, 15$(from left to right) in the scaled Hanover County dataset.



responsive to low priority calls, it is up to decision maker to choose the level of cutoff with allowable loss for low priority calls.

Finally, dispatch probabilities and loss probabilities gained from simulation under the same setting as MILP are displayed in the right half columns of Table 3 and 4. Considering the fact the Hypercube model is an approximate model, the discrepancy between the analytical Hypercube model results and simulation results are relatively small enough to conclude that the Hypercube model is valid.

Figure 4 shows the optimal district design for the original Hanover County dataset with 5 servers, cutoff $s1 = 2$. The district design is contiguous for high priority calls due to constraint 24, while it is not for low priority calls. In the original dataset, district designs for high priority calls were not affected by the number of reserved servers, while they are in scaled dataset. Figure 5 shows the optimal district designs for the scaled Hanover County dataset with 16 servers. Here district designs for high priority calls differ by the cutoff, which contributes to greater increase in the expected coverage by reserving capacities.

4.3. Infinite-line Queue Systems

In this section we provide result from the system with infinite-line queue. As it can be seen from Table 6, dispatch probabilities and the system-wide server busy probability r are not changed much when cutoff is modified. This is straightforward as the servers get busy serving the delayed calls in a queue there is minor improvement from reserving capacity.

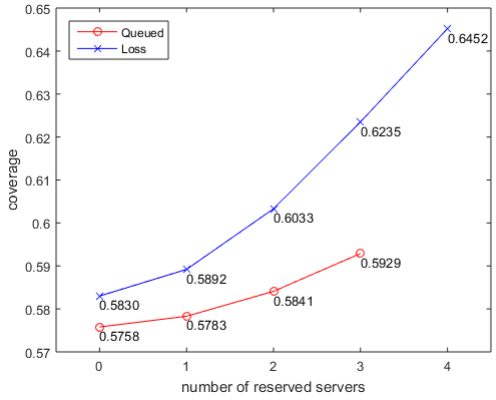
The coverage result from the Queued model is paralleled with the result from the Loss model in Figure 7. The amount of improvement is much smaller in the Queued model than the Loss model. This result is straightforward from the previously stated fact that dispatch probabilities are not changed significantly by reserving capacities in the Queued model. Another thing to note is that the Queued model was infeasible for $s_1 = 1$ because the low priority call queue explodes. Nevertheless, the Queued model has a clear benefit compared

FIGURE 6. High Priority Dispatch Probabilities for Different Cutoff Numbers With Infinite-line Queue.

	Hypercube			
	$s_1 = 5$	$s_1 = 4$	$s_1 = 3$	$s_1 = 2$
k=1	0.6611	0.6606	0.6608	0.6619
k=2	0.1982	0.2014	0.2110	0.2322
k=3	0.0687	0.0728	0.0792	0.0782
k=4	0.0275	0.0319	0.0316	0.0178
k=5	0.0126	0.0171	0.0090	0.0051
P(lost)	0.0317	0.0162	0.0084	0.0047

Note. Result for the cutoff number $s_1 = 1$ is omitted due to infeasibility.

FIGURE 7. Coverage over High Priority Calls.



to the Loss model that it provides improvements in coverage without abandoning calls or resorting to external services to cover them.

5. Discussion

6. References

References

[1] Sardar Ansari, Laura Albert McLay, and Maria E. Mayorga. A Maximum Expected Covering Problem for District Design. *Transportation Science*, November 2015.

[2] Rajan Batta, June M. Dolan, and Nirup N. Krishnamurthy. The maximal expected covering location problem: Revisited. *Transportation Science*, 23(4):277–287, 1989.

[3] B. A. Benn. *Hierarchical Car Pool Systems in Railroad Transportation*. PhD thesis, Case Institute of Technology, Cleveland, Ohaio, 1966.

[4] Luce Brotcorne, Gilbert Laporte, and Frdric Semet. Ambulance location and relocation models. *European Journal of Operational Research*, 147(3):451 – 463, 2003.

[5] Richard Church and Charles ReVelle. The maximal covering location problem. *Papers in Regional Science*, 32(1):101–118, 1974.

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

[7] Jaiswal N. K. *Priority Queues*. Academic Press, New York, 1968.

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

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

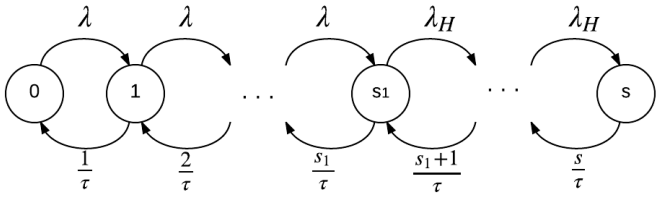
[10] Laura A. McLay and Maria E. Mayorga. A Dispatching Model for Server-to-Customer Systems That Balances Efficiency and Equity. *Manufacturing & Service Operations Management*, 15(2):205–220, May 2013.

[11] Christian Schaack and Richard C. Larson. An n-server cutoff priority queue. *Operations Research*, 34(2):257–266, 1986.

[12] I. D. S. Taylor and J. G. C. Templeton. Waiting time in a multi-server cutoff-priority queue, and its application to an urban ambulance service. *Operations Research*, 28(5):1168–1188, 1980.

Appendix

FIGURE 8. Transition Diagram for 0 Capacity Queue.



A. Derivation of approximate Hypercube model with cutoff priority queue

First, we need to derive closed-form expressions for steady-state probabilities in cutoff priority queue. For 0−line queue, Figure 8 shows the corresponding transition diagram.

Therefore, by solving the following balance equations, we gain the desired probability.

$$P_1 = \lambda\tau P_0 = \rho s P_0, \cdots, P_i = \frac{\rho^i s^i}{i!} P_0 \qquad i \le s_1,$$

$$P_{s_1+1} = \lambda_H \frac{\tau}{s_1+1} P_{s_1} = \frac{\rho s}{s_1+1}\frac{\lambda_H}{\lambda} P_{s_1} = \frac{\rho^{s_1}}{s_1+1!} s^{s_1}\frac{\lambda_H}{\lambda} P_0,$$

$$\cdots, P_i = \frac{\rho^i s^i}{i!}(\frac{\lambda_H}{\lambda})^{i-s_1} P_0 \qquad i > s_1$$

In order to derive the steady-state probabilities formulation for the $\infty-$capacity queue, SCQQ model in Taylor and Templeton(1980) was referred. In their paper they also provide the formula for the case where high priority customers are lost while only low priority customers are queued (SCQL model). Therefore if one wanted to apply the idea of reserving capacity with low priority queue, that model can be directly used in our procedure too, with updates for steady-state probabilities P_i in STEP 1 and setting the term $w = 1 - P_s\frac{\lambda^H}{\lambda} - \sum_{i=s1}^{s} P_i\frac{\lambda^L}{\lambda}$ in STEP 2.

Next, correction factors for low priority in cutoff priority queue is derived. This follows Larson(1975)'s procedure to derive an expression for $P\{B_1 b_2 \cdots B_j F_{j+1}\}$ (p852) closely, be punctuated by some changes due to cutoff assumption.

Let $B_j \equiv$ event that jth selected is busy (not available), $F_j \equiv B_j^c =$ event that jth server selected is free (or available), $S_k \equiv$ state of the queuing system indicating that exactly k servers are busy, and $P\{B_1 B_2 \cdots B_j F_{j+1}\} \equiv$ probability that the first free server is the $j+1$st server selected.

Under the cutoff scheme, if the arriving call is low priority, we have

$$P\{B_1 B_2 \cdots B_j F_{j+1}\} = P\{B_1 b_2 \cdots B_j F_{j+1} | S_k, k < s_1\}$$

because we do not select a server to dispatch to a low priority call, if the number of busy server is equal to or greater than the cutoff.

Therefore we wish to derive an expression for $P\{B_1 b_2 \cdots B_j F_{j+1} | S_k, k < s_1\}$ that will motivate an approximation procedure for the model in which servers are not identical. (Note: In the hypercube model under a fixed-preference dispatching policy, the dispatcher always assigns the most preferred available server. Therefore the desired probability is the probability that the first j preferred servers are busy and the $j+1$st is free.)

By laws of conditional probability we have

$$P\{B_1 b_2 \cdots B_j F_{j+1} | S_k, k < s_1\} = \sum_{k=0}^{k=s_1-1} P\{B_1 B_2 \cdots B_j F_{j+1} | S_k\} P_k$$

Where

$$P\{B_1 B_2 \cdots B_j F_{j+1} | S_k\} = P\{F_{j+1} | B_1 B_2 \cdots B_j S_k\} P\{B_j | B_1 B_2 \cdots B_{j-1} S_k\} \cdots P\{B_1 | S_k\}.$$

Then we have

$$P\{B_i | B_1 B_2 \cdots B_{i-1} S_k\} = \frac{k-(i-1)}{s-(i-1)}, \qquad i = 1, 2, \cdots, k+1$$
$$P\{F_{j+1} | B_1 B_2 \cdots B_j S_k\} = \frac{s-k}{s-j}, \qquad j = 0, 1, \cdots, k$$

Combining above results we have the desired probability

$$P\{B_1 b_2 \cdots B_j F_{j+1} | S_k, k < s_1\} = \sum_{k=j}^{s_1-1} \frac{k}{s}\frac{k-1}{s-1}\cdots\frac{k-(j-1)}{s-(j-1)}\frac{s-k}{s-j}P_k$$
$$= \sum_{k=j}^{s_1-1} \frac{(s-j-1)!(s-k)k!}{(k-j)!s!}P_k$$

with P_k derived in the note above, or,

$$P\{B_1 B_2 \cdots B_j F_{j+1} | S_k, k < s_1\} = Q_j^L r^j (1-r),$$

where

$$Q_j^L = \frac{\sum_{k=j}^{s_1-1} \frac{(s-j-1)!(s-k)k!}{(k-j)!s!}P_j}{(1-P_s\frac{\lambda_H}{\lambda} - \sum_{i=s_1}^{s} P_i\frac{\lambda_L}{\lambda})^j \rho^j \left(1-(1-P_s\frac{\lambda_H}{\lambda} - \sum_{i=s_1}^{s} P_i\frac{\lambda_L}{\lambda})\rho\right)}.$$

Lastly, the derivation of Q_j^H is just analogous to above with s_1 replaced by s.

B. Psuedo-code for generating preference lists

Algorithm 1 Generating Preference Lists

```
for all j ∈ J do
    for all i ∈ I^open do
        k ← 1
        while k ≤ s do
            if x^H_ijk = 1 then
                b^H_jk ← i
            end if
            if x^L_ijk = 1 then
                b^L_jk ← i
            end if
            k ← k + 1
        end while
    end for
end for
```