Hypercube Model With Multiple-Unit Dispatches and Police Patrol-Initiated Activities

by

Shiow-Hwa Gau and Richard C. Larson

OR 188-88 October 1988

Hypercube Model with Multiple-Unit Dispatches and Police Patrol-Initiated Activities

Shiow-Hwa Gau
Richard C. Larson

Massachusetts Institute of Technology
Cambridge, MA 02139

Sep. 24, 1988

1. Introduction

In recent years, queueing models, which are frequently used to estimate a system's performance and to optimize server management policies or customer profit strategies, have been developed and applied in a wide variety of settings. Urban police operations is one of the now classic application areas.

The police patrol and dispatch system can be described and modeled as a multi-server queueing system in which calls for service (CFS's) and patrol-initiated activities (PIA's) arrive randomly over time and space and require various amounts of service time. A CFS is initiated by a citizen telephoning the police, reporting the need for on-scene police service. A PIA is a discretionary activity undertaken by the mobile police patrol officer as a result of something he or she sees from the patrol car (e.g., traffic violation, building check, car check, pedestrian check, resident check, car chase). The first analytic model to incorporate the spatial queueing aspects of police patrol and dispatch was Larson's hypercube queueing model; he described both an exact model incorporating a 2^N state continuous time Markov process (assuming N police cars in the system) (1974) and an "approximate model" that reduces the number of equations to solve from 2^N to N (1975). In 1978, building on word described by Larson in Chapter 5 of his book Urban Police Patrol Analysis, Chaiken and Dormont designed the Patrol Car Allocation Model (PCAM). The original versions of both models have two major limitations. First, they assume that only a single unit is dispatched to each call, and second, they allow only two activity states for each patrol unit : busy on a CFS, or free on patrol. In applications, these assumptions are not always realistic. For instance, Larson and McKnew reported that up to fifty percent of the busy time of a police patrol unit may be on PIA's (1982). Green and Kolesar (1984), in New York City, reported that thirty percent of dispatches were multiple unit dispatches (i.e., dispatches in which more than one police patrol vehicle was assigned to respond to the scene of the CFS.)

To extend the hypercube models' applicability, Chelst and Barlach (1981) presented two models, one exact and the other approximate, that capture the simultaneous response of one or two identical units dispatched to a single call. Furthermore, to incorporate patrol-initiated activities that consume a sizable fraction of a patrol unit's time, Larson and McKnew (1982) developed a hypercube type queueing model of a police patrol force that allows PIA's. With regard to

PCAM, Green (1984) presented a multi-server, multi-priority queueing model in which the number of servers assigned to each customer is a random variable that depends on the customer's "type" and the availability of servers. Recently, Schaack and Larson (1986 and 1988) considered the differences between high- and low-priority demands for service and developed cutoff priority queues to optimize server management, including the complexity of multiple unit assignments.

Basically, the Chelst and Barlach and the Larson and McKnew methods are built on Larson's hypercube model (1974 and 1975), both exact and approximate forms. In this paper, we present two models, one exact and the other approximate, to allow multiple-unit dispatches and police patrol-initiated activities within a system queueing model, thereby merging into one model the earlier generalization of Chelst / Barlach and Larson / McKnew. This is the first generalization of the hypercube model that incorporates both of these complexities, which are important in implementation environments in such cities as St. Louis, New York City, and Peoria, Illinois.

On the basis of these two models, we then create two computer programs that can be run on any 16-bit microcomputer such as the IBM PC/AT (since most city-supported computer facilities do not have mainframe systems). The approximate version is more feasible to use since the state space of the exact model expands to become computationally infeasible for even medium-sized emergency service problems. Our purpose is to develop more accurate, management-oriented tools to improve patrol dispatch policies. The outcomes of our tests on these two models using simulations and statistics support our work.

Section 2 introduces the models and their assumptions. In Sections 3 and 4, we describe the required notation and discuss the mathematical structure of the exact model and the approximate model. We then implement the models' formulae in two computer programs that are written in FORTRAN and can be run on a 16-bit microcomputer. (The Appendix describes the organization and computational results of these implementation tools). Section 5 lists some statistical results of the approximate model. Finally, in Section 6, we offer some remarks on implementing the results, and briefly discuss the possible applications of the models.

2. An introduction to the models and their assumptions

In order to develop a spatially-oriented, N-patrol-unit system, we consider a continuous-time, discrete-state Markov model that is a generalization of the hypercube model (Larson 1974). In addition to yielding values for all the performance measures of the hypercube model, our version allows multiple-unit dispatches and police-patrol initiated activities. We assume the reader is familiar with Larson (1974, 1975), Chelst and Barlach (1981), and Larson and McKnew (1982).

Suppose we have J geographical atoms and N servers. CFS (calls for service) from each atom j, and PIA (patrol-initiated activities) opportunities for each server n, arrive according to independent Poisson processes. For simplicity, we distinguish between two types of CFS's: type I calls summon only a single unit to service, while type II calls require two units. For any given atom j, type I and type II CFS's arrive independently at the rate $\lambda_1^{(1)}(j)$ and $\lambda_1^{(2)}(j)$, respectively. Similarly, $\lambda_2(n)$ is the arrival rate of PIA opportunities for server n.

We assume that the (on scene) service times for both the CFS's and the PIA's are independently and exponentially distributed for each server n. The same condition is also true for the two units dispatched under type II CFS's. In other words, when service begins, the "completion time" for CFS's (type I or type II) and PIA's for each server, say server n, has a distribution that is exponential, with mean $\mu_1(n)^{-1}$ and $\mu_2(n)^{-1}$, respectively. If $\mu_1(n)^{-1}$ equals $\mu_2(n)^{-1}$ for each server n, two units servicing a type II CFS can be treated as equivalent to two units servicing two separate type I CFS's. In such a case, we can reduce the number of states and simplify the model. In order to save computer storage space and simplify the computation, we make some assumptions about the service times. Considering the case in which the variation in service times between a CFS and a PIA is more than that between servers, we have the following assumption for the approximate model: $\mu_1(n) = \mu_1$ and $\mu_2(n) = \mu_2$, for all servers n.

Upon completion of service, the server immediately returns to one of its possible locations, which depends on the probability that the server is located in one of several possible atoms. Let l_{nj} be the probability that server n, when available, is located in atom j. Associated with each atom k is a list that ranks servers in order of their dispatch preference. The ranking is determined by the relative average travel time, t_{nk}, from the possible locations of each server n to atom k, the atom under consideration. The server that has the smallest t_{nk} occupies first place, followed by the server with the next smallest t_{nk}, and so forth. The determination of t_{nk} depends on

- l_{nj} – the probability that server n is located in atom j, for all j,
- d_{jk} – the distance between atoms j and k, and
- v – the effective average respond speed of each server.

Clearly each atom has a fixed list of servers, independent of the state of the system. This dispatch policy, labeled EMCM (expected modified center-of-mass) by Larson has been shown (Larson 1972) to be a more appropiate dispatch policy compared to the methods SCM (strict center-of-mass), MCM (modified center-of-mass) and ESCM (expected strict center-of-mass).

We consider a finite state system. When a type I CFS arrives at an atom, the highest ranked available server is dispatched. In the case where all N servers are busy, the CFS will be lost. For type II CFS's, the two highest-ranked available servers are dispatched. If only one server is available, we will treat this CFS as a type I CFS and dispatch the available server. In all other cases, this CFS will be lost. PIA opportunities not observed by a free (patrolling) server are also lost. There is no preemption.

In practice, no service system will ever satisfy the mathematical model's assumptions exactly. We can check compatability of the model with real system by means of test results and simulations.

3. The exact model

Before we describe the formulation of the exact model, we need to introduce some new notation. We define

$S(i,n) \equiv$ status of server n when the system is in state i; $i = 0, 1, \cdots, 4^N-1$;
$n = 1, 2, \cdots, N$;

$= 0$ [free]
$= 1$ [busy on type I CFS's]
$= 2$ [busy on type II CFS's]
$= 3$ [busy on PIA's].

Thus we have the following 1-1 mapping from 4^N system states $\{ i; i = 0, 1, 2, \cdots, 4^N-1 \}$ into the status of servers $\{ (S(i, n); n = 1, 2, \cdots, N), i = 0, 1, 2, \cdots, 4^N-1 \}$:

$$i = S(i, N)\cdot 4^{N-1} + \cdots + S(i, m)\cdot 4^{m-1} + \cdots + S(i, 2)\cdot 4^1 + S(i, 1)\cdot 4^0.$$

By the assumptions stated in Section 2, we see that when the system is in state i, and a CFS arrives from atom j, one or two servers n(i, j) that satisfy $S(i, n(i, j)) = 0$ should be dispatched to the CFS. Otherwise, the call will be lost.

Let $n_{i,j}^1$ $\equiv$ identification number of the server to be assigned to the CFS, given a type I CFS from atom j when the system is in state i; $i = 0, 1, 2, \cdots, 4^N-1$; $j = 1, 2, \cdots, J$.

$n_{i,j}^2(1), n_{i,j}^2(2) \equiv$ identification numbers of two servers to be assigned to the CFS, given a type II CFS from atom j when the system is in state i; $i = 0, 1, 2, \cdots, 4^N-1$; $j = 1, 2, \cdots, J$.

$P(i)$ $\equiv$ equilibrium probability that the system is in state i; $i = 0, 1, 2, \cdots, 4^N-1$.

Obviously, these definitions imply the following conditions.

$S(i, n_{i,j}^1) = S(i, n_{i,j}^2(1)) = S(i, n_{i,j}^2(2)) = 0$
and
$P(i) = 0$ for all $i < 0$ and $i \geqq 4N$.

We can now proceed to derive a balance-of-flow formula for the exact model by equating the output flow from and the input flow into each state i :

$$P(i)\left[(\lambda_1^{(1)}+\lambda_1^{(2)})\delta + \sum_{m:\,S(i,m)=0} \lambda_2(m) + \sum_{m:\,S(i,m)=1,2} \mu_1(m) + \sum_{m:\,S(i,m)=3} \mu_2(m) \right]$$

$$= \sum_{m:\,S(i,m)=0} \left[\{P(i+4^{m-1})+P(i+2\cdot\ 4^{m-1})\}\cdot\ \mu_1(m) + P(i+3\cdot\ 4^{m-1})\cdot\ \mu_2(m) \right]$$

$$+ \sum_{m:\,S(i,m)=1} P(i-4^{m-1}) \sum_{j:\, n^{1}_{i-4^{m-1},\,j}=m} \lambda_1^{(1)}(j)$$

$$+ \sum_{m_1,m_2:\,S(i,m_1)=S(i,m_2)=2} P(i-s(m_1,m_2)) \sum_{j\in\{m_1,m_2\}\subseteq\{n^{2}_{i-s(m_1,m_2),\,j^{(1)}},\, n^{2}_{i-s(m_1,m_2),\,j^{(2)}}} \lambda_1^{(2)}(j)$$

$$+ \sum_{m:\,S(i,m)=3} P(i-3\cdot\ 4^{m-1})\,\lambda_2(m)\ , \qquad (1)$$

where $s(m_1,m_2)$ =	$2\cdot4^{m_1-1}+2\cdot4^{m_2-1}$	if there exist m_1,m_2 such that $S(i,m_1)=S(i,m_2)=2$.
	$2\cdot4^{m_1-1}$	if only one m, say m_1, satisfies $S(i,m_1)=2$.
	$i+1$	if for all m, $S(i,m)=2$.

δ =	1	if at least one server is free.
	0	if all servers are busy.

$$\lambda_1^{(1)} \equiv \sum_{j=1}^{J} \lambda_1^{(1)}(j)\ ; \qquad \lambda_1^{(2)} \equiv \sum_{j=1}^{J} \lambda_1^{(2)}(j)\ ; \qquad \lambda_2 \equiv \sum_{n=1}^{N} \lambda_2(n)\ .$$

Each term on the left-hand side of (1) contains the rate the system leaves state i, due respectively, to

① dispatch assignments to CFS's
② initiations of PIA's
③ completions of service on CFS's
④ completions of service on PIA's.

On the right-hand side of (1), each term contains the rate the system enters state i , due respectively, to

① both types of service completions that bring the system into state i

② the rates for dispatch assignments for type I CFS 's

③ the rates for dispatch assignments for type II CFS 's

④ the rates of PIA opportunities that take the system into state i

Since the system equations (1) are linearly dependent, we add the condition that

$$\sum_{i=0}^{4^N-1} P(i) = 1$$

to solve for 4^N unknown probabilities. Once we obtain the P(i)'s, system performance measures and additional key statistics are easy to calculate. For instance,

- The workloads of server n on type I and type II CFS's and on PIA's, respectively, are

$$\rho_n^{CFS\,\mathrm{I}} = \sum_{i:\,S(i,n)=1} P(i) \;; \qquad (2a)$$

$$\rho_n^{CFS\,\mathrm{II}} = \sum_{i:\,S(i,n)=2} P(i) \;; \qquad (2b)$$

$$\rho_n^{PIA} = \sum_{i:\,S(i,n)=3} P(i) \;. \qquad (2c)$$

The total workload for server n is $\rho_n = \rho_n^{\mathrm{CFS\,I}} + \rho_n^{\mathrm{CFS\,II}} + \rho_n^{\mathrm{PIA}}$.

- The saturation probability is

$$P_{SAT} = \sum_{i:\,S(i,n)\neq 0,\,\forall n} P(i) \;. \qquad (3)$$

- Consider type I CFS's from atom j, the fraction of dispatches that are assigned to server n is

$$f_{n,j,1} = \frac{1}{1-P_{SAT}} \sum_{n=n^1_{i,j}} P(i) \quad . \tag{4}$$

- Consider type II CFS's from atom j, the fraction of dispatches that are assigned to server m and server n is

$$f_{(m,n),j,2} = \frac{1}{1-P_{SAT}} \sum_{\{m,n\}=\{n^2_{i,j}(1),\, n^2_{i,j}(2)\}} P(i) \quad . \tag{5}$$

- Consider type II CFS's from atom j, the fraction of dispatches that are assigned to a single server, the server m, is

$$f_{(m,0),j,2} = \frac{1}{1-P_{SAT}} \sum_{m=\{n^2_{i,j}(1),\, n^2_{i,j}(2)\}} P(i) \quad . \tag{6}$$

- The average travel time to atom j is

$$T_j = \frac{\sum_{m=1}^{N} \left[f_{m,j,1} t_{mj} + \sum_{n=1}^{N} f_{(m,n),j,2} \left(\frac{t_{mj}+t_{nj}}{2} \right) + f_{(m,0),j,2} t_{mjj} \right]}{\sum_{m=1}^{N} \left[f_{m,j,1} + \sum_{n=1}^{N} f_{(m,n),j,2} + f_{(m,0),j,2} \right]} \quad . \tag{7}$$

- The average travel time of server n is

$$TU_n = \frac{\sum_{j=1}^{J} \left[\lambda_1^{(1)}(j) f_{n,j,1} t_{nj} + \lambda_1^{(2)}(j) \sum_{m=1}^{N} f_{(m,n),j,2} t_{nj} + \lambda_1^{(2)}(j) f_{(n,0),j,2} t_{nj} \right]}{\sum_{j=1}^{J} \lambda_1^{(1)}(j) f_{n,j,1} + \lambda_1^{(2)}(j) \sum_{m=1}^{N} f_{(m,n),j,2} + \lambda_1^{(2)}(j) f_{(n,0),j,2}} \tag{8}$$

- The average travel time to requests in the primary response area of server n is

$$TRA_n = \frac{\sum_{j \in \text{response area } n} \sum_{m=1}^{N} \left[f_{m,j,1} t_{mj} + \sum_{l=1}^{N} f_{(m,l),j,2} \left(\frac{t_{mj} + t_{lj}}{2} \right) + f_{(m,0),j,2} t_{mj} \right]}{\sum_{j \in \text{response area } n} \sum_{m=1}^{N} \left[f_{m,j,1} + \sum_{l=1}^{N} f_{(m,l),j,2} + f_{(m,0),j,2} \right]} . \qquad (9)$$

4. The approximate model

In many practical applications, the number of servers is greater than 5. This reduces tremendously the applicability of the exact model. The number of simultaneous linear equations that must be solved, 4^N, increases exponentially as the number of servers, N, increases. Calculating the steady-state probabilities thus requires enormous amount of storage and execution time. To handle this drawback, we now present an approximate model that requires only the solution of N simultaneous linear equations (which describe an "aggregate" model) and N simultaneous nonlinear equations to solve the workload performance measures.

To compute system performance measures in the approximate model, we need dispatch probabilities rather than fine-grain Markov system state probabilities. We formulate the aggregate model to estimate "correction" factors in a product form solution, and use the correction factors to account for the false assumption that the servers in the approximate model are mutually independent of each other. Our approximate model is analogous to Larson's (1975) for the hypercube model, and differs mainly in both the details of the aggregate model and in the correction factors required.

To simplify the aggregate model, we assume that all servers are identical. This implies that mean service times are server independent, i. e., $\mu_1(n)=\mu_1$, $\mu_2(n)=\mu_2$ for all $n=1, 2, \dots, N$.

We assume that there is a fixed preference dispatch list for each atom. To derive the correction factors, we define the following terms.

W_n ≡ the nth ranked server is working. (i. e., "busy")
W_n^c ≡ the nth ranked server is not working. (i. e., "not busy")

$P(W_1W_2 \cdots W_k W^c_{k+1})$ ≡ the probability that the (k+1)th server will be dispatched. (10')

$P(W_1W_2 \cdots W_{l-1}W_l^cW_{l+1} \cdots W_kW^c_{k+1})$ ≡ the probability that the lth and (k+1)th ($1 \leqq l \leqq k$) servers will be dispatched.(11')

$P(W_1W_2 \cdots W_kW^c_{k+1}W_{k+2} \cdots W_N)$ ≡ the probability that only the (k+1)th server will be dispatched. (12')

If the servers are independent of each other, we can obtain, by the product of respective servers' utilization factors or availability factors, the probabilities (10'), (11') and (12') that some servers will be dispatched to CFS's. We then introduce the factors Q(N, *k*, 1) and Q(N, *k*, 2) to "correct" this independence argument. Thus we have the following expressions.

Consider a type I CFS from atom j. We approximate (10') as follows :

$$P(W_1 W_2 \ldots\ldots W_k W_{k+1}^c) \simeq Q(N, k, 1) \prod_{r=1}^{k} \rho_{(r)}(1 - \rho_{(k+1)}) \quad . \tag{10}$$

Consider a type II CFS from atom j. We approximate (11') and (12'), respectively, as follows:

$$P(W_1 W_2 \ldots\ldots W_{l-1} W_l^c W_{l+1} \ldots\ldots W_k W_{k+1}^c) \simeq Q(N, k, 2) \prod_{r=1, r\neq l}^{k} \rho_{(r)}(1 - \rho_{(l)})(1 - \rho_{(k+1)}) \tag{11}$$

$$P(W_1 W_2 \ldots\ldots W_k W_{k+1}^c W_{k+2} \ldots\ldots W_N) \simeq Q(N, N-1, 1) \prod_{r=1, r\neq k+1}^{N} \rho_{(r)}(1 - \rho_{(k+1)}) \quad , \tag{12}$$

where $\rho_{(r)}$ = fraction of time that the rth preferred server is busy on CFS or PIA opportunities, and

Q(N, *k*, 1) and Q(N, *k*, 2) are correction factors for dispatching one and two servers, respectively.

Because of the assumption that all servers are identical, the right-hand sides of Equations 10, 11 and 12 reduce, respectively, to

$$Q(N, k, 1)\rho^{k}(1-\rho); \quad Q(N, k, 2)\rho^{k-1}(1-\rho)^{2}; \quad \text{and} \quad Q(N, N-1, 1)\rho^{N-1}(1-\rho).$$

To derive the correction factors, we try to solve the left-hand sides of Equations 10 11 and 12. Following the theory developed by Larson (1975) for sampling servers without replacement, we condition on all states of the system having exactly m servers busy, where m=1, 2, ..., N. Thus the left-hand sides of 10, 11 and 12, respectively, can be obtained as follows.

$$P(W_1W_2\ldots W_kW^c_{k+1}) = \sum_{m=k}^{N-1} P(W_1W_2\ldots W_kW^c_{k+1}|S_m)P(S_m)$$

$$P(W_1W_2\ldots W_{l-1}W^c_lW_{l+1}\ldots W_kW^c_{k+1}) = \sum_{m=k-1}^{N-2} P(W_1W_2\ldots W_{l-1}W^c_lW_{l+1}\ldots W_kW^c_{k+1}|S_m)P(S_m)$$

$$P(W_1W_2\ldots W_kW^c_{k+1}W_{k+2}\ldots W_N) = P(W_1W_2\ldots W_kW^c_{k+1}W_{k+2}\ldots W_N|S_{N-1})P(S_{N-1}),$$

where S_m is the state corresponding to exactly m servers being busy.

By arguments similar to those of Larson and McKnew (1982), we find the following correction factors:

$$Q(N,k,1)= \sum_{m=k}^{N-1} \frac{C^m_k}{C^N_k}\,\frac{(N-m)}{(N-k)}\,\frac{P_m}{r^k(1-r)} \qquad k=1,2,\ldots,N-1.$$

$$Q(N,k,2)= \sum_{m=k-1}^{N-2} \frac{C^m_{k-1}}{C^N_{k-1}}\,\frac{(N-m)(N-m-1)}{(N-k+1)(N-k)}\,\frac{P_m}{r^{k-1}(1-r)^2} \qquad k=1,2,\ldots,N-1.$$

where r is the average utilization factor of servers in the aggregate model, i. e.,

$$r \equiv \frac{1}{N}\sum_{m=1}^{N} mP_m \quad . \tag{13}$$

To solve $P_m=P(S_m)$ and derive the above correction factors, we develop an appropriate aggregate model. Figure 1 is the state-transition-rate diagram for this model. We say that the queueing system " is in state n " in the aggregate model whenever n servers are busy (n = 0, 1, 2, ···, N).

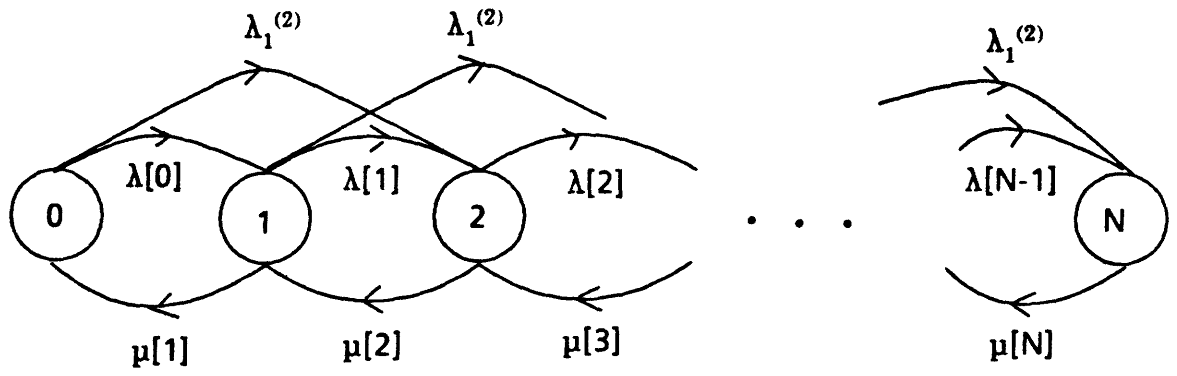


Figure 1. Transition diagram of the aggregate model.

The following global balance equations (14) can be used to compute the correction factors Q(N, *k*, 1) and Q(N, *k*, 2).

$$
\begin{aligned}
\{\lambda[0]+\lambda_1^{(2)}\}P_0 &= \mu[1]\,P_1\\
\{\lambda[1]+\lambda_1^{(2)}+\mu[1]\}P_1 &= \lambda[0]\,P_0+\mu[2]\,P_2\\
\{\lambda[n]+\lambda_1^{(2)}+\mu[n]\}P_n &= \lambda_1^{(2)}P_{n-2}+\lambda[n-1]\,P_{n-1}+\mu[n+1]P_{n+1} \qquad n=2,3,\ldots,N-2\\
\{\lambda[N-1]+\mu[N-1]\}P_{N-1} &= \lambda_1^{(2)}P_{N-3}+\lambda[N-2]\,P_{N-2}+\mu[N]\,P_N\\
\mu[N]\,P_N &= \lambda_1^{(2)}P_{N-2}+\lambda[N-1]\,P_{N-1}\\
\sum_{i=0}^{N}P_i &= 1\,,
\end{aligned}
\tag{14}
$$

where

$$\lambda[n] = \lambda_1^{(1)} + \frac{N-n}{N}\lambda_2 \qquad n=0,1,\dots,N-2,$$

$$\lambda[N-1] = \lambda_1^{(1)} + \frac{1}{N}\lambda_2 + \lambda_1^{(1)}$$

$$\mu[1] = \left\{ \frac{\lambda_1^{(1)}}{\lambda_1^{(1)}+\lambda_2}\frac{1}{\mu_1} + \frac{\lambda_2}{\lambda_1^{(1)}+\lambda_2}\frac{1}{\mu_2} \right\}^{-1}$$

$$\mu[n] = n\left\{ \frac{\lambda_1^{(1)}+\lambda_1^{(2)}}{\lambda_1^{(1)}+\lambda_1^{(2)}+\frac{N-(n-1)}{N}\lambda_2}\frac{1}{\mu_1} + \frac{\frac{N-(n-1)}{N}\lambda_2}{\lambda_1^{(1)}+\lambda_1^{(2)}+\frac{N-(n-1)}{N}\lambda_2}\frac{1}{\mu_2} \right\}^{-1} \qquad n=2,3,\dots,N.$$

Because the approximate model assumes that all servers are identical, each available server initiates PIA opportunities at the same rate, λ_2 / N. In other words, if i servers are busy ($i \leqq N-1$), the transition rate from state i to state $(i+1)$ is $\lambda_1^{(1)}+\lambda_2(N-i)/N$: the first term corresponds to CFS's and the second to PIA's. The other upward transition rate, due to type II CFS's, from state i to state $(i+2)$ is $\lambda_1^{(2)}$.

The downward transition rates from state i to state $(i-1)$ are obtained to approximate the average service rate of mixed work between CFS's and PIA's for each state. Consider the state i. There are three possible dispatches into this state: type I CFS's, type II CFS's and PIA's. Type I CFS's and PIA's are dispatched from state $(i-2)$ into state i, while type II CFS's are dispatched from state $(i-1)$ into state i. We approximate the average fraction of customers in service that correspond to CFS's by

$$f_{\text{CFS}} = [\lambda_1^{(1)}+\lambda_1^{(2)}] / [\lambda_1^{(1)}+\lambda_1^{(2)}+(N-i+1)\lambda_2/N].$$

Of course, the average fraction of customers in service that are PIA customers is $1-f_{\text{CFS}}$. The average service times for CFS's and PIA's are $1/\mu_1$ and $1/\mu_2$, respectively. The average time for any one of the busy servers to complete its service on a current customer would then be $f_{\text{CFS}}(1/\mu_1)+(1-f_{\text{CFS}})(1/\mu_2)$. If we use the inverse of this average time to approximate the rate of service completions for every busy server, the downward transition resulting from the completion of service from state i to state $(i-1)$ is $\mu[n]$, as given above.

By solving N simultaneous linear equations (14), we obtain P_m, the equilibrium probability that exactly i servers are busy. We then compute the correction factors, based on the equilibrium probabilities. After computing the correction factors, we can develop an iterative procedure to estimate the server workloads.

Let $G_n^k \equiv$ set of atoms for which unit n is the kth preferred server.
$v_{j,k} \equiv$ identification number of the kth preferred server for atom j.
$R_n \equiv$ rate at which server n becomes busy.

Following the derivation of Larson and McKnew (1982), we have the following equations :

$$R_n = \lambda_2(n) P\{W_n^c\} + \sum_{j \in G_n^1} \left[\lambda_1^{(1)}(j) + \lambda_1^{(2)}(j)\right] P\{W_n^c\}$$

$$+ \sum_{j \in G_n^2} \left[\lambda_1^{(1)}(j) + \lambda_1^{(2)}(j)\right] P\{W_{v_{j1}} W_n^c\} + \sum_{j \in G_n^2} \lambda_1^{(2)}(j) P\{W_{v_{j1}}^c W_n^c\}$$

$$+ \sum_{j \in G_n^3} \left[\lambda_1^{(1)}(j) + \lambda_1^{(2)}(j)\right] P\{W_{v_{j1}} W_{v_{j2}} W_n^c\} + \sum_{j \in G_n^3} \lambda_1^{(2)}(j) \left[P\{W_{v_{j1}}^c W_{v_{j2}} W_n^c\} + P\{W_{v_{j1}} W_{v_{j2}}^c W_n^c\} \right]$$

$$+ \;\ldots\ldots\ldots\ldots$$

$$+ \sum_{j \in G_n^N} \left[\lambda_1^{(1)}(j) + \lambda_1^{(2)}(j)\right] P\{W_{v_{j1}} W_{v_{j2}} \ldots W_{v_{j(N-1)}} W_n^c\}$$

$$+ \sum_{j \in G_n^N} \lambda_1^{(2)}(j) \sum_{k=1}^{N-1} P\{W_{v_{j1}} \ldots W_{v_{j(k-1)}} W_{v_{jk}}^c W_{v_{j(k+1)}} \ldots W_{v_{j(N-1)}} W_n^c\}$$

$$= \Big\{ \lambda_2(n) + \sum_{j \in G_n^1} \left[\lambda_1^{(1)}(j) + \lambda_1^{(2)}(j) \right]$$

$$+ \sum_{j \in G_n^2} \left[\lambda_1^{(1)}(j) + \lambda_1^{(2)}(j) \right] Q(N,1,1)\, \rho_{v_{j1}} + \sum_{j \in G_n^2} \lambda_1^{(2)}(j) Q(N,1,2)(1 - \rho_{v_{j1}})$$

$$+ \quad \cdots\cdots\cdots$$

$$+ \sum_{j \in G_n^N} \left[\lambda_1^{(1)}(j) + \lambda_1^{(2)}(j) \right] Q(N, N-1, 1) \rho_{v_{j1}} \cdots \rho_{v_{j(N-1)}}$$

$$+ \sum_{j \in G_n^N} \lambda_1^{(2)}(j) Q(N, N-1, 2) \sum_{k=1}^{N-1} \Big(\prod_{s=1, s=k}^{N-1} \rho_{v_{js}} \Big)(1 - \rho_{v_{jk}}) \Big\} \; (1 - \rho_n)$$

$$= F_n \, (1 - \rho_n) \tag{15}$$

$$R_n = \mu_1 \rho_n + \lambda_2(n)\,(1 - \rho_n) \left[1 - \frac{\mu_1}{\mu_2} \right] . \tag{16}$$

By equating (15) and (16), we obtain the following expression for workloads ρ_n.

$$\rho_n = \frac{F_n}{F_n + \mu_1 + \lambda_2(n) \left[\frac{\mu_1}{\mu_2} - 1 \right]} \qquad n = 1, 2, \ldots, N. \tag{17}$$

Based on the expressions for workloads shown in (17), we can solve iteratively each workload ρ_n, n = 1, 2, ..., N, by setting all the initial values equal to r, which is given in (13).

Now we can calculate the workload, ρ_n, for server n, where n = 1, 2, ..., N.

The iterative processes are convergent if the maximum difference in our implementation between any successive corresponding ρ_n is less than ε, where ε is a small number (for instance, ε is set to be 10^{-6}). Experimental observations show that the required number of iterations is roughly between 3 and 6.

As the last step in deriving the approximate model, we can calculate some performance measures based on the correction factors and workloads. For instance, we can determine that

- The saturation probability is

$$P_{SAT} = P_N \qquad \text{(solved by (14)).}$$

- The fraction of type I CFS's from atom j dispatches that are assigned to server n, where $n = v_{j,1}, v_{j,2}, \ldots, v_{j,N}$, is

$$f_{v_{j1},j,1} = \frac{1}{sumj1} f'_{v_{j1},j,1}$$

$$f_{v_{jk},j,1} = \frac{1}{sumj1} f'_{v_{jk},j,1} \qquad k \geq 2\,.$$

- The fraction of type II CFS's from atom j dispatches that are assigned to server m and server n, where $m = v_{j,k}$, $n = v_{j,p}$, $1 \leqq k < p \leqq N$, is

$$f_{(v_{jk},v_{jp}),j,2} = \frac{1}{sumj2} f'_{(v_{jk},v_{jp}),j,2} \qquad 1 \leq k < p \leq N.$$

- The fraction of type II CFS's from atom j dispatches that are assigned only server m, where $m = v_{j,1}, v_{j,2}, \ldots, v_{j,N}$, is

$$f_{(v_{jk},0),j,2} = \frac{1}{sumj2} f'_{(v_{jk},0),j,2} \qquad k = 1, 2, \ldots, N.$$

where

$$f'_{v_{j1},j,1} = 1 - \rho_{v_{j1}}$$

$$f'_{v_{jk},j,1} = Q(N, k-1, 1) \prod_{l=1}^{k-1} \rho_{v_{jl}} (1 - \rho_{v_{jk}}) \qquad k \geq 2.$$

$$f'_{(v_{jk}, v_{jp}), j, 2} = Q(N, p-1, 2) \prod_{l=1}^{p-1} \rho_{v_{jl}} (1 - \rho_{v_{jk}})(1 - \rho_{v_{jp}}) \qquad 1 \leq k < p \leq N.$$

$$f'_{(v_{jk}, 0), j, 2} = Q(N, N-1, 1) \prod_{l=1}^{N} \rho_{v_{jl}} (1 - \rho_{v_{jk}}) \qquad k = 1, 2, ..., N.$$

$$sumj1 = \sum_{k=1}^{N} f'_{v_{jk},j,1} \qquad j = 1, 2, ..., J.$$

$$sumj2 = \sum_{k=1}^{N} \left[f'_{(v_{jk}, 0), j, 2} + \sum_{p=k+1}^{N} f'_{(v_{jk}, v_{jp}), j, 2} \right] \qquad j = 1, 2, ..., J.$$

- The average travel time to atom j is as given in Equation 7.

- The average travel time to atom j is as given in Equation 7.

- The average travel time of server n is as given in Equation 8.

- The average travel time to requests in the primary response area of server n is as given in Equation 9.

5. Testing the numerical results of the exact model and the approximate model

For most real-world cases, we cannot apply the exact model due to the huge storage requirements and long execution times of such problems. The approximate model is the more usable and important evaluation tool. In this section, we test the accuracy and the speed of the approximate model and find that, it works extremely well.

As a test for accuracy, we first develop several data sets that are representative of actual police patrol configurations. We then compare the numerical results based on the exact model or the approximate model with those resulting from simulations using the same data set, and use the latter figures as the standard for comparison. We compare two measurements: workload of each unit and the fraction of dispatches of each atom that are assigned to different server(s). **Fig. 1** exhibits the test results using the exact model versus the simulation. **Figs. 2, 3, 4** and **5** exhibit the test results using the approximate model versus the simulation. These results confirm that the output of the approximate model is sufficiently close to the true performance. By comparing the numerical results using the exact model with those using the simulation method, we conclude that the simulation results represent true performance to a great extent.

The testing process involves three comparisons. First, we compare corresponding workload measurements for each unit. The data represented by the bars filled in heavy vertical lines are derived from the exact model (**Fig. 1**) or the approximate model (**Figs. 2, 3, 4** and **5**). The data represented by the bars filled in light horizontal lines are derived from the simulations. Next we compare the saturation probabilities of two comparing models, P^1_{SAT} and P^2_{SAT}, from the exact model or the approximate model and the simulations. In the last step, we calculate the correlation coefficient (Corr. Coe.) of two sets of " fraction of dispatches " and the percentage distribution of corresponding differences, D_i, between them.

The adopted data sets are AS1003, AS1206 AS1506 and AS2010, whose number of atoms and number of units are (10, 3), (12, 6), (15, 6) and (20, 10) respectively. The testing is done by running lengthy simulations on a mainframe to assume accuracy.

The number of CFS's and PIA's generated in a simulation process is approximately 80,000.

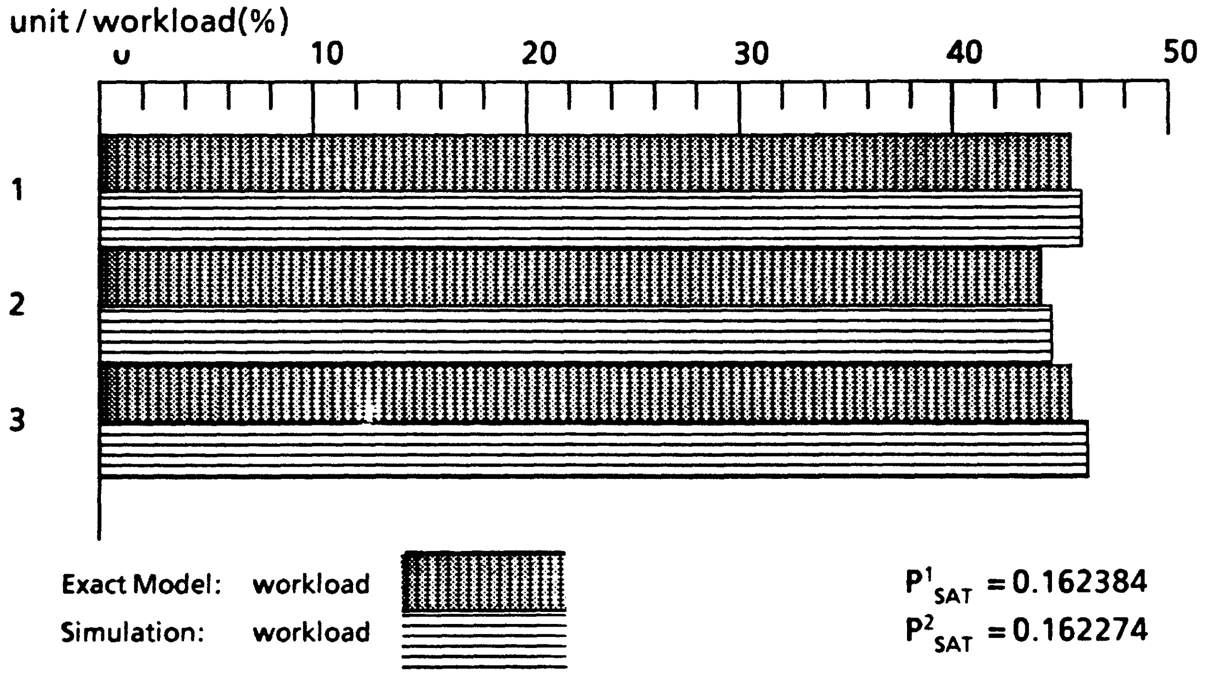
We estimate the workloads for each server, ρ_n, in the simulation process as follows :

$$\rho_n = \frac{\textit{time period during which server n is busy}}{\textit{total time period under consideration}} \qquad n = 1, 2, \ldots, N.$$

In addition, we estimate the saturation probability P_{SAT} in the simulation process in the following way :

$$P_{SAT} = \frac{\textit{no. of type I CFS's that are lost} + \textit{no. of type II CFS's that are lost}}{\textit{no. of type I CFS's} + \textit{no. of type II CFS's}}$$

Evaluating performance measures such as average travel times, workload measurements, and saturation probability is an important step in setting or adjusting dispatch policies. We found that the performance measures of both models, the exact and the approximate, and the simulation yielded very similar results. Comparing the corresponding fraction of dispatches, we observe that the relative differences between these performance measurements are small. The most important criterion for these tests is the quality of numerical results as a function of N (number of servers) and P_{SAT} (saturation probability). From Figures 2, 3, 4 and 5, we observe that the quality of the numerical results of the approximate model improves as N gets large and/or as P_{SAT} gets smaller.



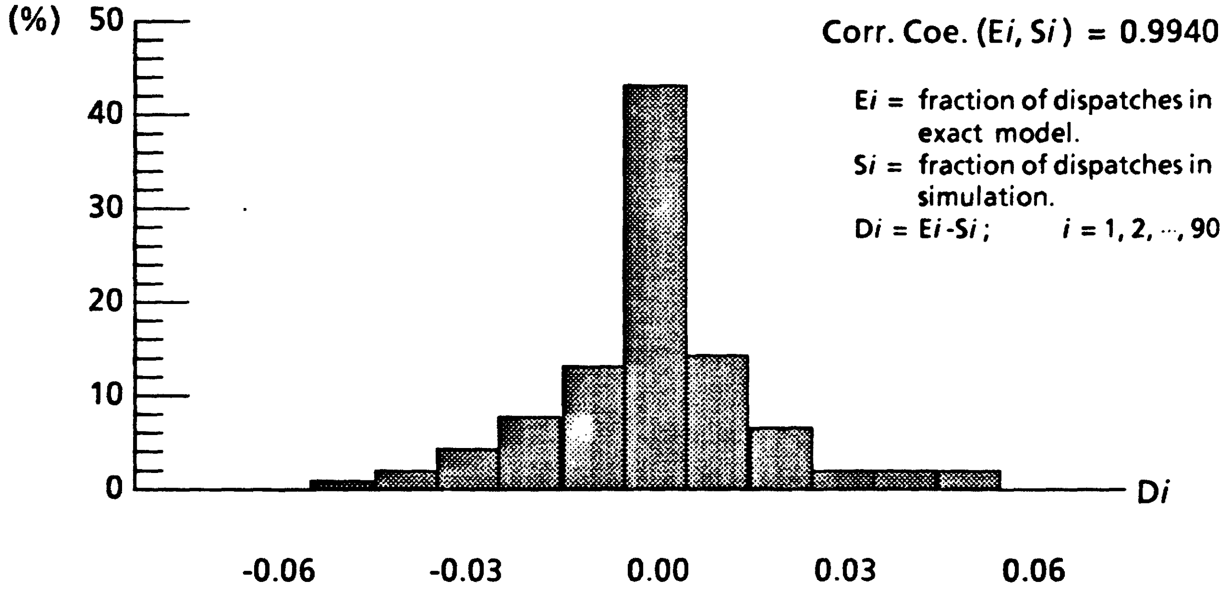
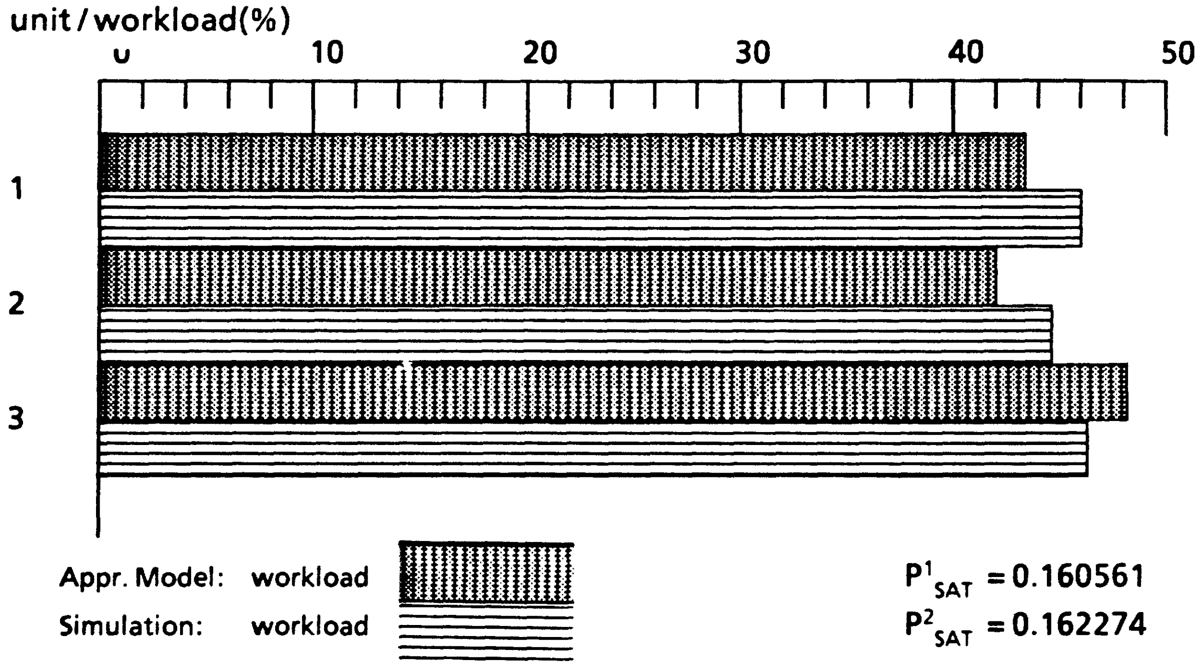


Fig. 1 Exact model vs. Simulation with data file AS1003



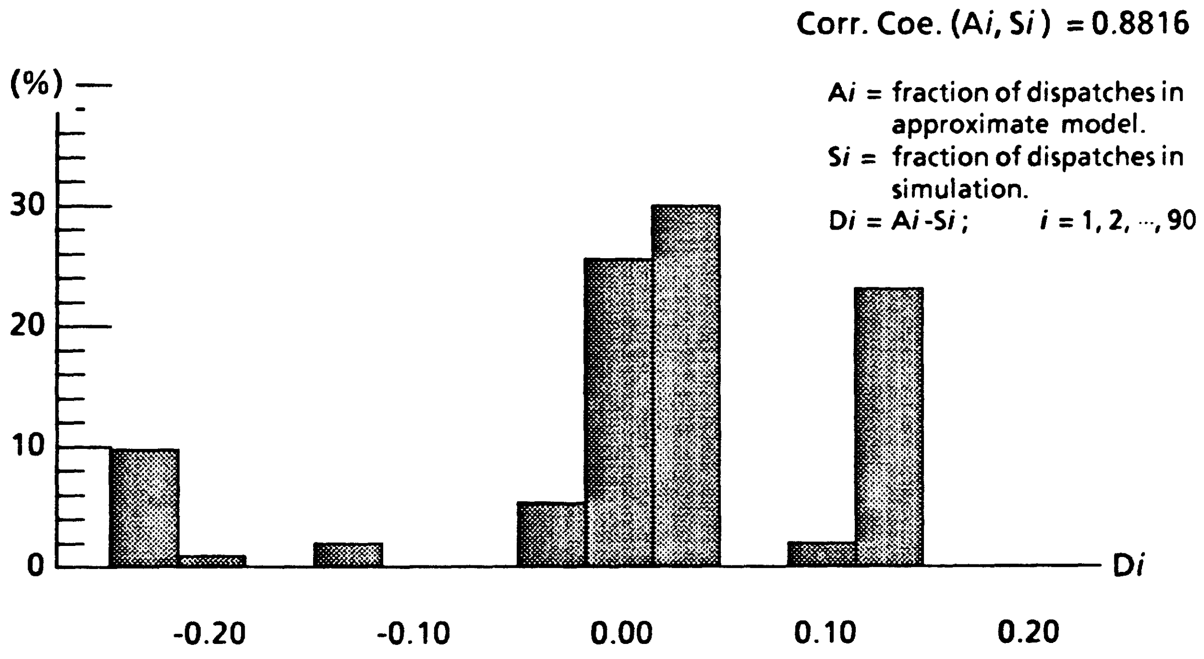
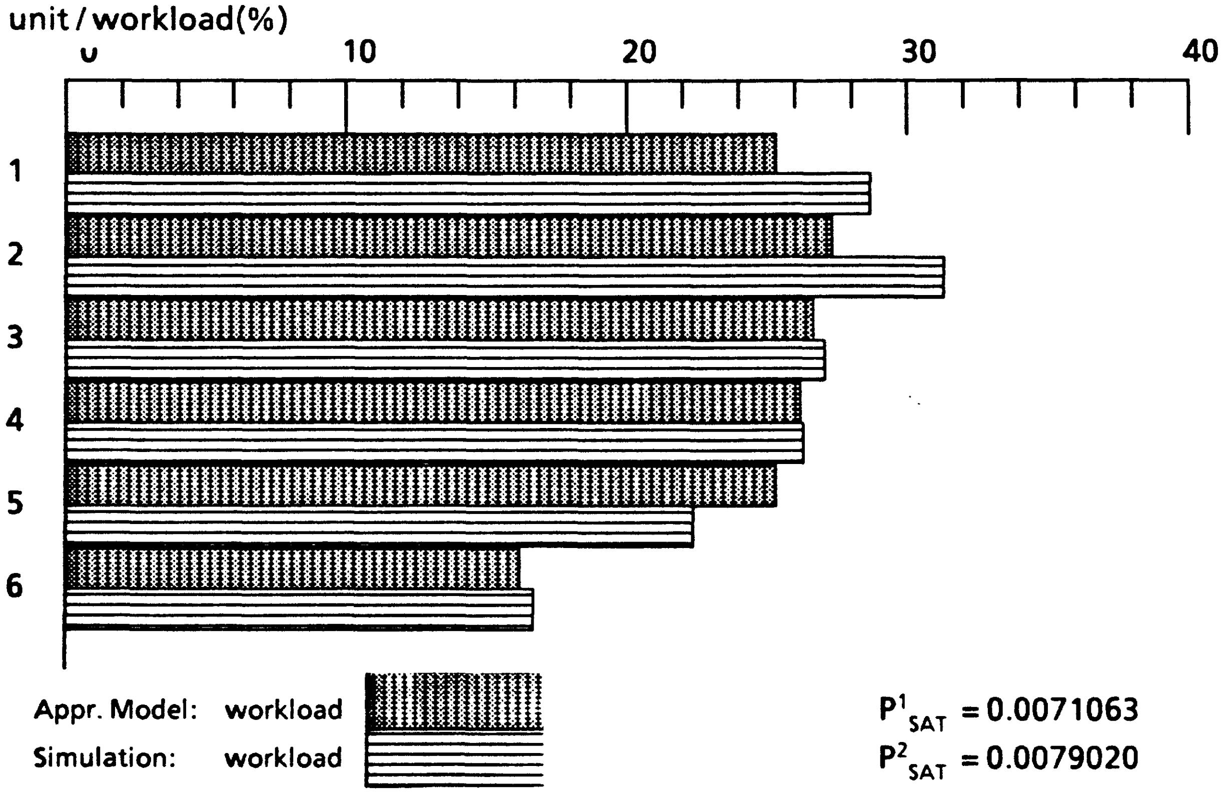


Fig. 2 Approximate model vs. Simulation with data file AS1003



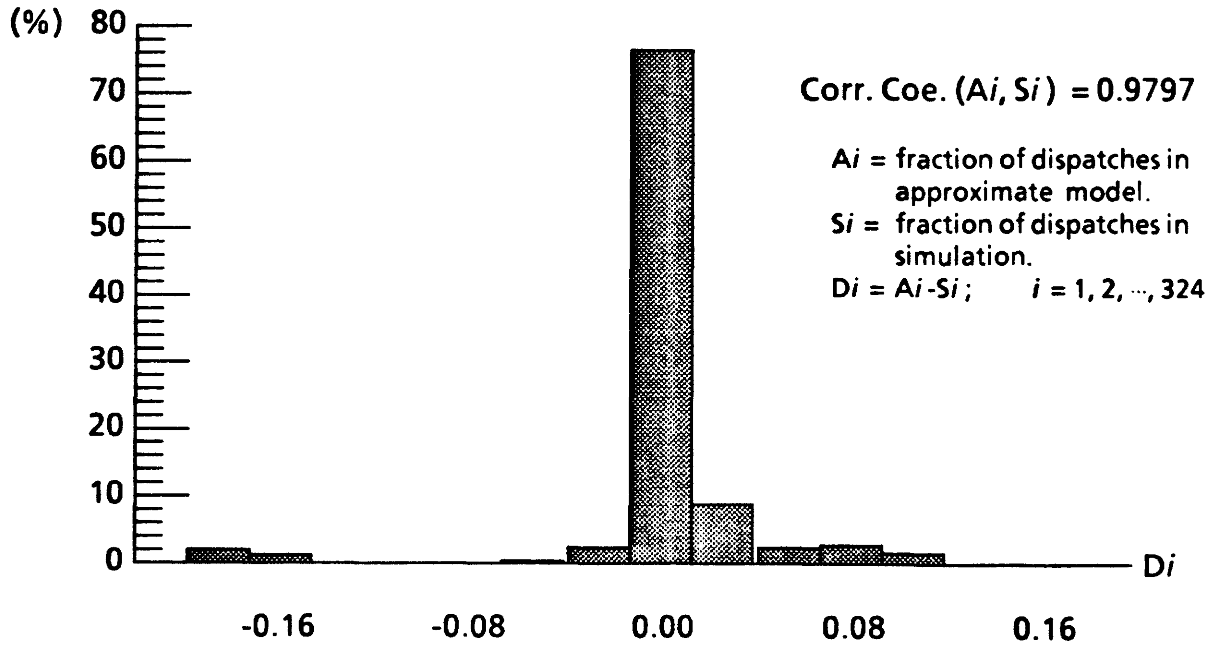
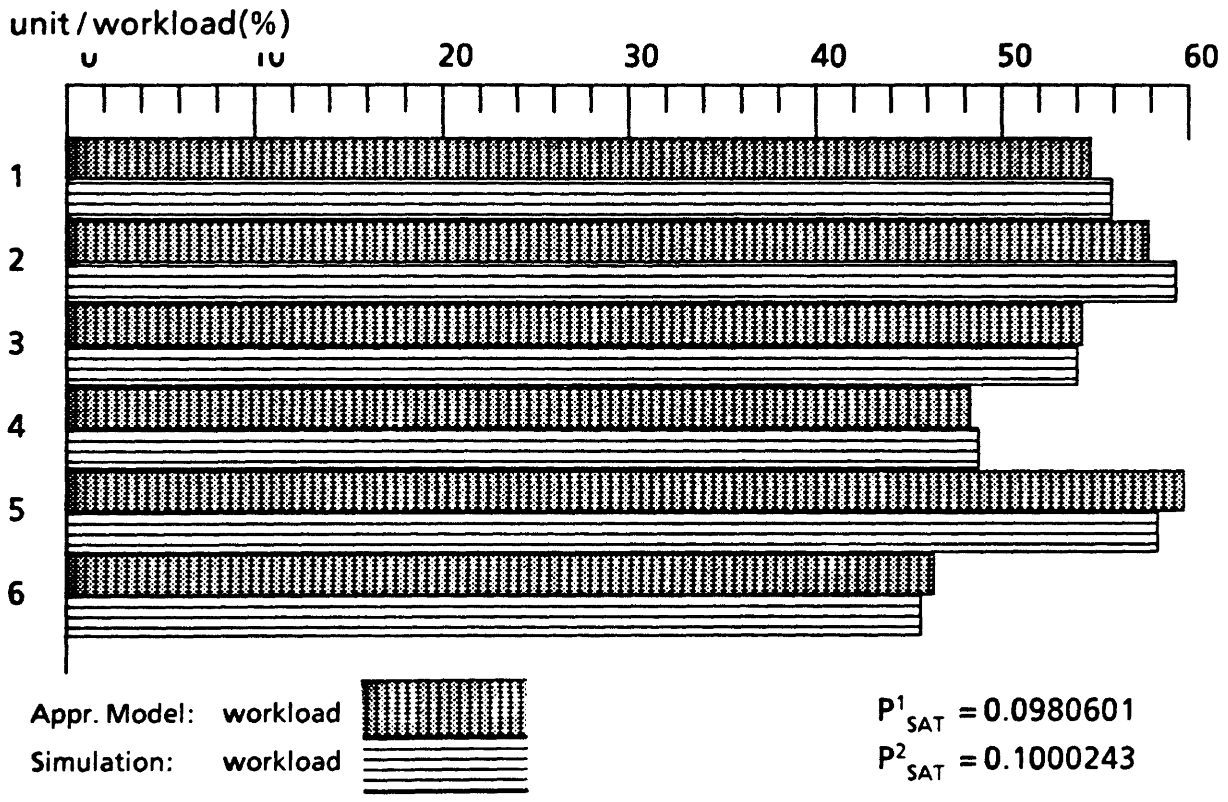


Fig. 3 Approximate model vs.Simulation with data file AS1206



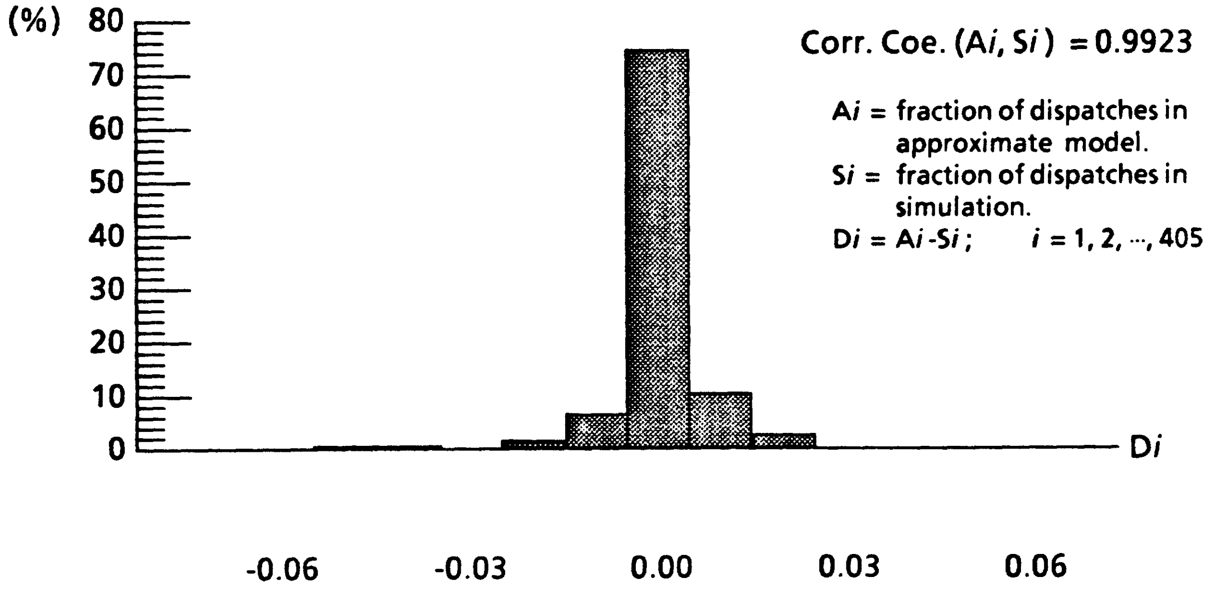
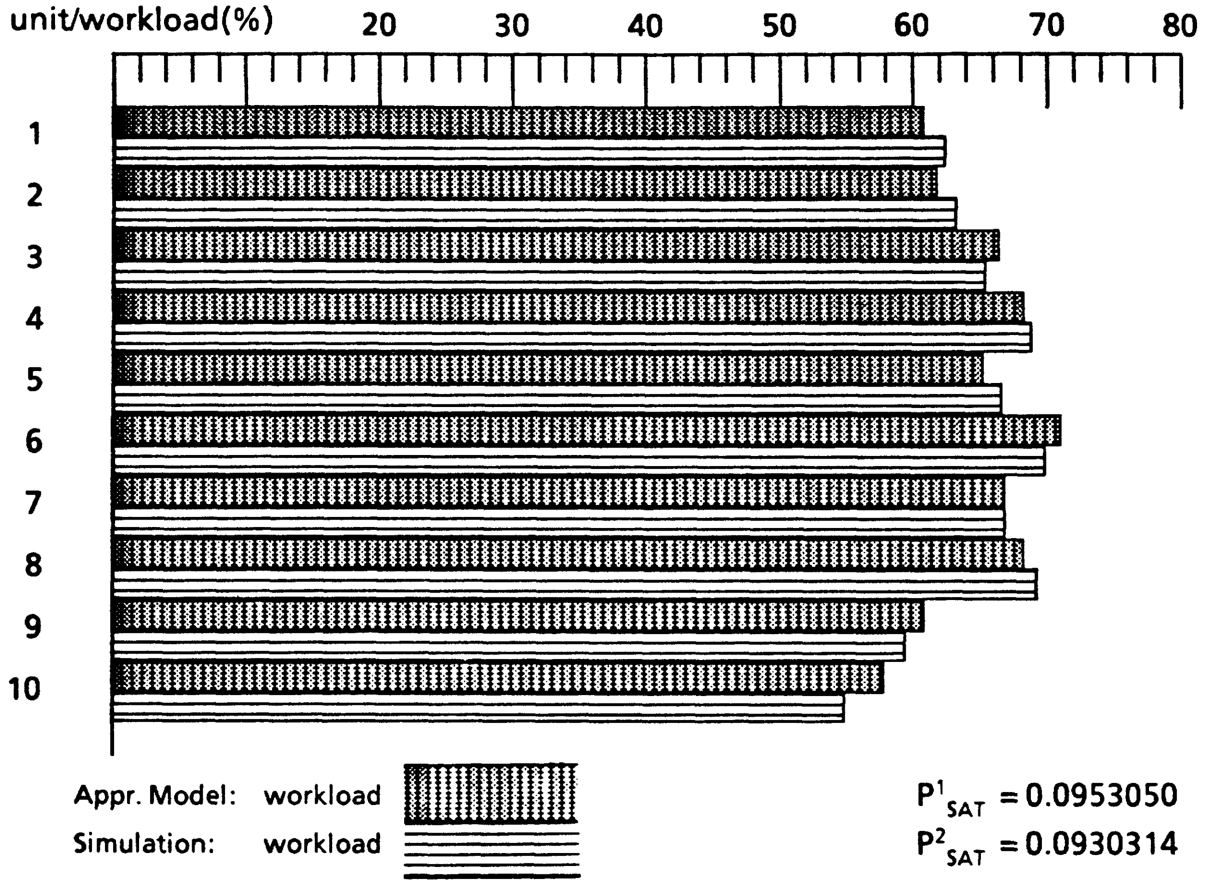


Fig. 4 Approximate model vs. Simulation with data file AS1506



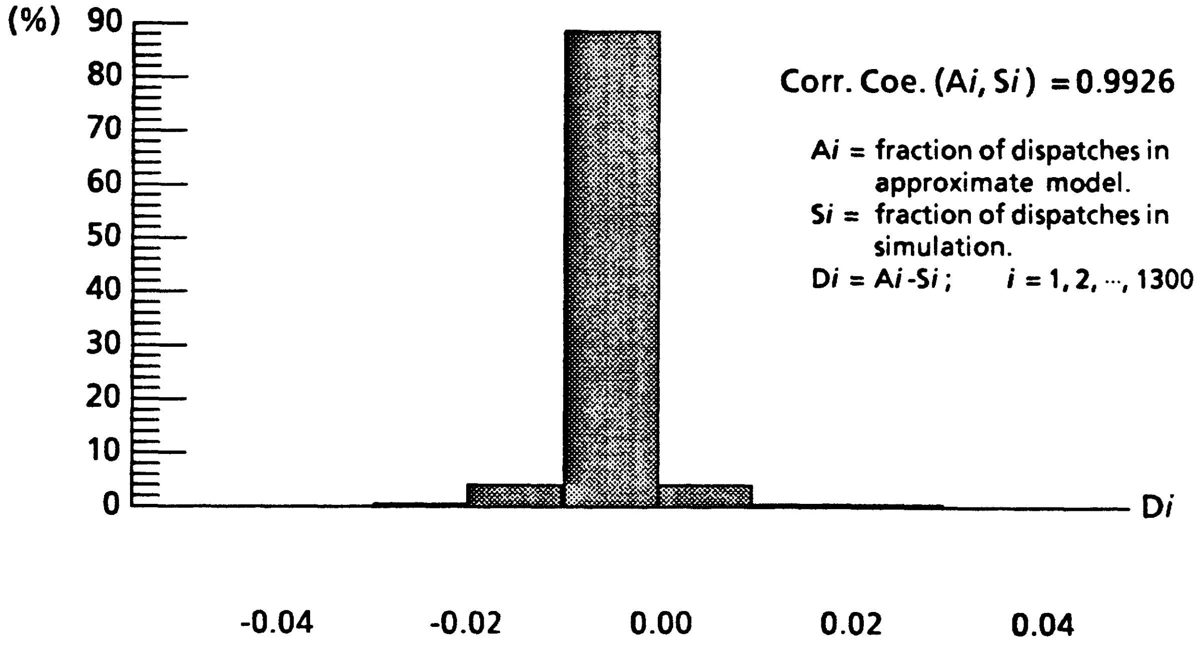


Fig. 5 Approximate model vs.Simulation with data file AS2010

6. Summary

We have presented exact and approximate models of an emergency service that includes PIA opportunities and in which two identical units may be dispatched to one call. We then generalized and created two computer-implemented versions based on these two models. Because of the computational infeasibility of the exact model, we concentrated more on the approximate version.

The programs that accompany this paper are intended to serve as a local police planner's tool. An assignment manager can evaluate a policy by running the program in advance and can then perform appropriate adjustments to arrive at an efficient dispatch policy. After entering the required data, the manager obtains the dispatch list for each atom. Based on each dispatch list, he or she can estimate some performance measures of the whole system. To balance the workload of each server, the manager can adjust the patrol or location probabilities. In addition to the model's usefulness as an aid in determining allocations of service facilities, it has proven to be a valuable tool for evaluating policies by considering fractional dispatch measures and/or average travel time.

The present model could be extended to relax the assumption of identical servers, or to allow CFS's to be queued, or to include the case of random number ($\geqq 3$) of servers. Such extensions would increase even further its applicability to real world problems.

7. Appendix

In Sections 3 and 4, we developed the probabilistic deployment model and its approximate analog. To implement these two models, we have written two computer programs, EXAC.FOR and APPR.FOR. In practice, an assignment manager can choose the appropriate policy by comparing the computational results of either model. In general, the maximum number of atoms and servers allowed in the exact model are 230 and 3, respectively, while in the approximate model, the numbers are 120 and 15, respectively.

In this Appendix, we first outline the data requirements and potential outputs of these two programs. We then explain briefly how to execute them. We illustrate our explanation with two examples, one for the exact model and the other for the approximate model.

7.1 Input and output

The following input data are required to run the programs.

J, N	= number of atoms and servers.		
$\lambda_1^{(1)}(j)$	= arrival rate of type I CFS for jth atom.	j	$= 1, 2, ..., J$
$\lambda_1^{(2)}(j)$	= arrival rate of type II CFS for jth atom.	j	$= 1, 2, ..., J$
$\lambda_2(n)$	= arrival rate of PIA opportunities for server n.	n	$= 1, 2, ..., N$
$\mu_1(n)^{-1}$	= mean service time for server n corresponding to CFS.	n	$= 1, 2, ..., N$
$\mu_2(n)^{-1}$	= mean service time for server n corresponding to PIA.	n	$= 1, 2, ..., N$
(l_{nj})	= the probability that server n is located in atom j when available for dispatch.	n j	$= 1, 2, ..., N$ $= 1, 2, ..., J$
(d_{ij})	= distance between atom i and atom j.	i,j	$= 1, 2, ..., J$

For simplicity, we set v, the speed of each server, to be 1 in both programs.

The output of the exact model contains a preference list for each atom, $P(i)$; equilibrium probabilities, ρ_n; workloads, T_j, TU_n, TRA_n; average travel time, P_{SAT}; saturation probability; and fraction of dispatches, $f_{n,\,j,\,1}$, $f_{(m,n),\,j,\,2}$. In the approximate model, on the other hand, instead of $P(i)$, we have more output on the state probabilities of the aggregate model and the correction factors Q(N, k, 1) and Q(N, k, 2).

7.2 Execution

As a first step in executing these programs, the user enters in the name of the appropriate program, EXAC or APPR, which corresponds to the exact or the approximate model. He or she is then asked to provide the name of the input data file. After the name of data file is input, the output will be displayed on the screen. The user may also print out the result. We will illustrate the execution of these two programs with the following examples.

7.3 Implementation of exact model and computational results

EXAMPLE 1. Implementation of the exact model.
DATA NAME: AS1003 PROGRAM NAME: EXAC.FOR

Before running the program, the user should create a data file which contains the input data, as in the following format.

Data file :	10 3	<-- J, N
(AS1003)	0.4 0.2 0.5 0.2 0.4 0.4 0.2 0.3 0.9 0.4	<-- $\lambda_1^{(1)}(j)$
	0.3 0.1 0.1 0.5 0.1 0.2 0.3 0.1 0.1 0.4	<-- $\lambda_1^{(2)}(j)$
	0.3 0.1 0.6	<-- $\lambda_2(n)$
	5.0 10.0 5.0 10.0 5.0 10.0	<-- $\mu_1(n)^{-1}, \mu_2(n)^{-1}$
	0 0 0 1.0 0 0 0 0 0 0	<-- (l_{nj})
	0 0 0 0 1.0 0 0 0 0 0	
	0 0 0 0 0 0 0.5 0.5 0 0	
	0 3 5 2 5 7 5 5 7 7	<-- (d_{ij})
	3 0 2 5 2 4 8 6 10 8	
	5 2 0 7 4 2 10 8 12 10	
	2 5 7 0 3 5 3 3 5 5	
	5 2 4 3 0 2 6 4 8 6	
	7 4 2 5 2 0 8 6 10 8	
	5 8 10 3 6 8 0 2 2 4	
	5 6 8 3 4 6 2 0 4 2	
	7 10 12 5 8 10 2 4 0 2	
	7 8 10 5 6 8 4 2 2 0	

The following procedures and information are shown on the screen while the program EXAC.FOR is executing.

```
>EXAC    <return>
   Please input the name of data file ?  AS1003     <return>

  Preference list :     atom 1  ->        1 2 3
                        atom 2  ->        2 1 3
                        atom 3  ->        2 1 3
                        atom 4  ->        1 2 3
                        atom 5  ->        2 1 3
                        atom 6  ->        2 1 3
                        atom 7  ->        3 1 2
                        atom 8  ->        3 1 2
                        atom 9  ->        3 1 2
                        atom 10 ->        3 1 2
```

i	0	1	2	3	4	5	6
P(i)	0.2409	0.0490	0.0445	0.0062	0.0692	0.0258	0.0171
i	7	8	9	10	7	12	13
P(i)	0.0020	0.0355	0.0132	0.0372	0.0009	0.0021	0.0006
i	14	15	16	17	18	19	20
P(i)	0.0004	0.0001	0.0777	0.0275	0.0172	0.0021	0.0260
i	21	22	23	24	25	26	27
P(i)	0.0210	0.0130	0.0012	0.0160	0.0118	0.0265	0.0007
i	28	29	30	31	32	33	34
P(i)	0.0007	0.0004	0.0002	0.0001	0.0308	0.0126	0.0296
i	35	36	37	38	39	40	41
P(i)	0.0008	0.0134	0.0108	0.0228	0.0006	0.0139	0.0149
i	42	43	44	45	46	47	48
P(i)	0.0287	0.0011	0.0003	0.0002	0.0005	0.0001	0.0125
i	49	50	51	52	53	54	55
P(i)	0.0036	0.0023	0.0003	0.0037	0.0022	0.0014	0.0001
i	56	57	58	59	60	61	62
P(i)	0.0020	0.0012	0.0030	0.0001	0.0001	0.0001	0.0001
i	63						
P(i)	0.0001						

P(i) = equilibrium probabilities $i = 0,1,2,\ldots,4^N - 1$.

** Saturation probability P_{SAT} = 0.1624

server n	workload	TRAn	TUn
1	0.455	1.515	1.984
2	0.442	2.209	2.074
3	0.455	2.139	1.942

TRAn = the average travel time to requests in primary response area of server n.
TUn = the average travel time of server n.

atom j	1	2	3	4	5	6	7
Tj	2.093	2.217	3.293	1.012	0.920	2.217	1.594
atom j	8	9	10				
Tj	1.226	2.507	2.528				

Tj = the average travel time to atom j.

atom j	server m	$F_{m,j,1}$	$F_{(m,0),j,2}$	$F_{(m,n),j,2}$ n = 1	n = 2	n = 3
1	1	0.650	0.091	0.000	0.432	0.127
1	2	0.234	0.115	0.000	0.000	0.119
1	3	0.116	0.116	0.000	0.000	0.000
2	1	0.218	0.091	0.000	0.000	0.127
2	2	0.666	0.115	0.432	0.000	0.119
2	3	0.116	0.116	0.000	0.000	0.000
3	1	0.218	0.091	0.000	0.000	0.127
3	2	0.666	0.115	0.432	0.000	0.119
3	3	0.116	0.116	0.000	0.000	0.000
4	1	0.650	0.091	0.000	0.432	0.127
4	2	0.234	0.115	0.000	0.000	0.119
4	3	0.116	0.116	0.000	0.000	0.000
5	1	0.218	0.091	0.000	0.000	0.127
5	2	0.666	0.115	0.432	0.000	0.119
5	3	0.116	0.116	0.000	0.000	0.000
6	1	0.218	0.091	0.000	0.000	0.127
6	2	0.666	0.115	0.432	0.000	0.119
6	3	0.116	0.116	0.000	0.000	0.000
7	1	0.235	0.091	0.000	0.144	0.000
7	2	0.115	0.115	0.000	0.000	0.000
7	3	0.650	0.116	0.415	0.119	0.000
8	1	0.235	0.091	0.000	0.144	0.000
8	2	0.115	0.115	0.000	0.000	0.000
8	3	0.650	0.116	0.415	0.119	0.000

(table continued on next page)

atom j	server m	$F_{m,j,1}$	$F_{(m,0),j,2}$	$F_{(m,n),j,2}$ n = 1 n = 2 n = 3
9	1	0.235	0.091	0.000 0.144 0.000
9	2	0.115	0.115	0.000 0.000 0.000
9	3	0.650	0.116	0.415 0.119 0.000
10	1	0.235	0.091	0.000 0.144 0.000
10	2	0.115	0.115	0.000 0.000 0.000
10	3	0.650	0.116	0.415 0.119 0.000

$f_{m,j,1}$ = the fraction of dispatches for type I CFS that assign server n to atom j.
$f_{(m,0),j,2}$ = the fraction of dispatches for type II CFS that only assign server m to atom j.
$f_{(m,n),j,2}$ = the fraction of dispatches for type II CFS that assign server m and n to atom j.

Do you want to print out the results ? (Y/N) -> N

EXAMPLE 2. Implementation of the approximate model.
DATA NAME: AS1506 PROGRAM NAME: APPR.FOR

Before running the program, the user should create a data file which contains the input data, as in the following format.

```
Data file :   15  6                                                   <-- J, N
(AS1506)     0.6 0.5 1.0 0.4 0.3 0.3 0.3 0.6 0.3 0.7 0.7 0.4 0.4 0.6 0.5   <-- $\lambda_1^{(1)}(n)$
             0.3 0.4 0.4 0.3 0.7 0.5 0.7 1.0 0.4 0.5 0.4 0.3 0.3 0.3 0.3   <-- $\lambda_1^{(2)}(n)$
             0.4 0.4 0.6 0.8 1.1 0.7                                   <-- $\lambda_2(n)$
             6.0 12.0 6.0 12.0 6.0 12.0 6.0 12.0 6.0 12.0              <-- $\mu_1(n)^{-1}, \mu_2(n)^{-1}$
             0  0  0  1.0 0  0  0  0  0  0  0  0  0  0  0              <-- $(l_{nj})$
             0  0  0  0  1.0 0  0  0  0  0  0  0  0  0  0
             0  0  0  0  0  0  0.5 0.5 0  0  0  0  0  0  0
             0  0  0  0  0  0  0  0  0  0  1.0 0  0  0  0
             0  0  0  0  0  0  0  0  0  0  0  0.3 0.3 0.4 0
             0  0  0  0  0  0  0  0  0  0  0  0  0  0  1.0
              0  3  5  2  5  7  5  5  7  7  9  7  9  9 12             <-- $(d_{ij})$
              3  0  2  5  2  4  8  6 10  8  6  4  6  6  9
              5  2  0  7  4  2 10  8 12 10  4  6  4  8  7
              2  5  7  0  3  5  3  3  5  5  7  5  7  7 10
              5  2  4  3  0  2  6  4  8  6  4  2  4  4  7
              7  4  2  5  2  0  8  6 10  8  2  4  2  6  5
              5  8 10  3  6  8  0  2  2  4 10  4  6  6  9
              5  6  8  3  4  6  2  0  4  2  8  2  4  4  7
              7 10 12  5  8 10  2  4  0  2 12  6  8  4  7
              7  8 10  5  6  8  4  2  2  0 10  4  6  2  5
              9  6  4  7  4  2 10  8 12 10  0  6  4  8  5
              7  4  6  5  2  4  4  2  6  4  6  0  2  2  5
              9  6  4  7  4  2  6  4  8  6  4  2  0  4  3
              9  6  8  7  4  6  6  4  4  2  8  2  4  0  3
             12  9  7 10  7  5  9  7  7  5  5  5  3  3  0
```

The following procedures and information are shown on the screen while program, APPR.FOR, executes.

```
>APPR  <return>
   Please input the name of data file ?  AS1506   <return>

  Preference list :    atom 1  ->      1 2 3 5 4 6
                       atom 2  ->      2 1 5 4 3 6
                       atom 3  ->      2 4 5 1 6 3
                       atom 4  ->      1 2 3 5 4 6
                       atom 5  ->      2 1 5 4 3 6
                       atom 6  ->      2 4 5 1 6 3
                       atom 7  ->      3 1 5 2 6 4
                       atom 8  ->      3 1 5 2 6 4
                       atom 9  ->      3 1 5 6 2 4
                       atom 10->       3 5 1 6 2 4
                       atom 11->       4 2 6 5 1 3
                       atom 12->       5 2 3 1 6 4
                       atom 13->       5 6 2 4 3 1
                       atom 14->       5 6 2 3 1 4
                       atom 15->       6 5 4 2 3 1
```

k	0	1	2	3	4	5	6
P(k)	0.0458	0.1190	0.1880	0.2117	0.1912	0.1462	0.0981
Q(N,k,1)		0.8396	0.8022	0.8468	0.9706	1.1894	
Q(N,k,2)		0.9727	0.8828	0.7508	0.7040	0.7181	

P(*k*) = equilibrium probability that k servers are busy in aggregate model.
Q(N,k,1) = correction factor for dispatching one server.
Q(N,k,2) = correction factor for dispatching two servers.
** Saturation probability = P(6).

server n	workload	TRAn	TUn
1	0.548	2.397	2.198
2	0.579	2.347	1.985
3	0.543	2.417	1.811
4	0.484	1.808	2.504
5	0.598	1.913	2.328
6	0.463	1.831	2.835

TRAn = the average travel time (Min) to requests in primary response area of server n.
TUn = the average travel time (Min) of server n.

atom j	1	2	3	4	5	6	7
Tj	2.903	2.735	2.995	1.785	1.665	1.920	2.483
atom j	8	9	10	11	12	13	14
Tj	1.927	3.195	2.591	1.808	1.684	1.979	2.049
atom j	15						
Tj	1.831						

Tj = the average travel time (Min) to atom j.

atom j	server m	$F_{m,j,1}$	$F_{(m,0),j,2}$	$F_{(m,n),j,2}$					
				n = 1	n = 2	n = 3	n = 4	n = 5	n = 6
1	1	0.500	0.027	0.000	0.219	0.125	0.036	0.051	0.019
1	2	0.214	0.024	0.000	0.000	0.110	0.032	0.045	0.017
1	3	0.129	0.027	0.000	0.000	0.000	0.037	0.052	0.019
1	4	0.057	0.034	0.000	0.000	0.000	0.000	0.000	0.024
1	5	0.065	0.022	0.000	0.000	0.000	0.030	0.000	0.015
1	6	0.035	0.038	0.000	0.000	0.000	0.000	0.000	0.000
2	1	0.244	0.027	0.000	0.000	0.029	0.072	0.110	0.019
2	2	0.469	0.024	0.219	0.000	0.025	0.063	0.097	0.017
2	3	0.045	0.027	0.000	0.000	0.000	0.000	0.000	0.019
2	4	0.092	0.035	0.000	0.000	0.037	0.000	0.000	0.024
2	5	0.114	0.022	0.000	0.000	0.024	0.058	0.000	0.015
2	6	0.035	0.038	0.000	0.000	0.000	0.000	0.000	0.000
3	1	0.072	0.027	0.000	0.000	0.014	0.000	0.000	0.034
3	2	0.470	0.023	0.049	0.000	0.012	0.248	0.085	0.030
3	3	0.026	0.027	0.000	0.000	0.000	0.000	0.000	0.000
3	4	0.279	0.034	0.071	0.000	0.017	0.000	0.124	0.043
3	5	0.101	0.022	0.045	0.000	0.011	0.000	0.000	0.027
3	6	0.053	0.037	0.000	0.000	0.019	0.000	0.000	0.000
4	1	0.500	0.027	0.000	0.219	0.125	0.036	0.051	0.019
4	2	0.214	0.024	0.000	0.000	0.110	0.032	0.045	0.017
4	3	0.129	0.027	0.000	0.000	0.000	0.037	0.052	0.019
4	4	0.057	0.034	0.000	0.000	0.000	0.000	0.000	0.024
4	5	0.065	0.022	0.000	0.000	0.000	0.030	0.000	0.015
4	6	0.035	0.038	0.000	0.000	0.000	0.000	0.000	0.000
5	1	0.244	0.027	0.000	0.000	0.029	0.072	0.110	0.019
5	2	0.469	0.024	0.219	0.000	0.025	0.063	0.097	0.017
5	3	0.045	0.027	0.000	0.000	0.000	0.000	0.000	0.019
5	4	0.092	0.035	0.000	0.000	0.037	0.000	0.000	0.024
5	5	0.114	0.022	0.000	0.000	0.024	0.058	0.000	0.015
5	6	0.035	0.038	0.000	0.000	0.000	0.000	0.000	0.000

(table continued on next page)

atom j	server m	$F_{m,j,1}$	$F_{(m,0),j,2}$	$F_{(m,n),j,2}$ $n=1$	$n=2$	$n=3$	$n=4$	$n=5$	$n=6$
6	1	0.072	0.027	0.000	0.000	0.014	0.000	0.000	0.034
6	2	0.470	0.023	0.049	0.000	0.012	0.248	0.085	0.030
6	3	0.026	0.027	0.000	0.000	0.000	0.000	0.000	0.000
6	4	0.279	0.034	0.071	0.000	0.017	0.000	0.124	0.043
6	5	0.101	0.022	0.045	0.000	0.011	0.000	0.000	0.027
6	6	0.053	0.037	0.000	0.000	0.019	0.000	0.000	0.000
7	1	0.227	0.027	0.000	0.055	0.000	0.017	0.103	0.038
7	2	0.070	0.024	0.000	0.000	0.000	0.015	0.000	0.033
7	3	0.505	0.027	0.237	0.056	0.000	0.018	0.105	0.039
7	4	0.032	0.034	0.000	0.000	0.000	0.000	0.000	0.000
7	5	0.106	0.022	0.000	0.045	0.000	0.014	0.000	0.031
7	6	0.059	0.038	0.000	0.000	0.000	0.024	0.000	0.000
8	1	0.227	0.027	0.000	0.055	0.000	0.017	0.103	0.038
8	2	0.070	0.024	0.000	0.000	0.000	0.015	0.000	0.033
8	3	0.505	0.027	0.237	0.056	0.000	0.018	0.105	0.039
8	4	0.032	0.034	0.000	0.000	0.000	0.000	0.000	0.000
8	5	0.106	0.022	0.000	0.045	0.000	0.014	0.000	0.031
8	6	0.059	0.038	0.000	0.000	0.000	0.024	0.000	0.000
9	1	0.228	0.027	0.000	0.024	0.000	0.017	0.102	0.070
9	2	0.037	0.023	0.000	0.000	0.000	0.015	0.000	0.000
9	3	0.507	0.027	0.237	0.024	0.000	0.017	0.105	0.071
9	4	0.032	0.034	0.000	0.000	0.000	0.000	0.000	0.000
9	5	0.106	0.022	0.000	0.019	0.000	0.014	0.000	0.057
9	6	0.090	0.037	0.000	0.033	0.000	0.024	0.000	0.000
10	1	0.130	0.027	0.000	0.024	0.000	0.017	0.000	0.070
10	2	0.037	0.024	0.000	0.000	0.000	0.015	0.000	0.000
10	3	0.507	0.027	0.129	0.024	0.000	0.018	0.211	0.071
10	4	0.032	0.034	0.000	0.000	0.000	0.000	0.000	0.000
10	5	0.203	0.022	0.103	0.019	0.000	0.014	0.000	0.057
10	6	0.090	0.038	0.000	0.033	0.000	0.024	0.000	0.000
11	1	0.037	0.026	0.000	0.000	0.013	0.000	0.000	0.000
11	2	0.189	0.023	0.021	0.000	0.012	0.000	0.033	0.112
11	3	0.025	0.027	0.000	0.000	0.000	0.000	0.000	0.000
11	4	0.567	0.034	0.030	0.245	0.017	0.000	0.048	0.164
11	5	0.049	0.021	0.019	0.000	0.011	0.000	0.000	0.000
11	6	0.133	0.037	0.033	0.000	0.019	0.000	0.053	0.000
12	1	0.080	0.027	0.000	0.000	0.000	0.017	0.000	0.038
12	2	0.236	0.024	0.055	0.000	0.121	0.015	0.000	0.034
12	3	0.142	0.027	0.064	0.000	0.000	0.018	0.000	0.039
12	4	0.033	0.035	0.000	0.000	0.000	0.000	0.000	0.000
12	5	0.449	0.022	0.051	0.196	0.111	0.014	0.000	0.031
12	6	0.060	0.038	0.000	0.000	0.000	0.024	0.000	0.000

(table continued on next page)

atom j	server m	$F_{m,j,1}$	$F_{(m,0),j,2}$	$F_{(m,n),j,2}$					
				n = 1	n = 2	n = 3	n = 4	n = 5	n = 6
13	1	0.025	0.026	0.000	0.000	0.000	0.000	0.000	0.000
13	2	0.105	0.023	0.012	0.000	0.021	0.053	0.000	0.000
13	3	0.039	0.027	0.013	0.000	0.000	0.000	0.000	0.000
13	4	0.078	0.034	0.017	0.000	0.031	0.000	0.000	0.000
13	5	0.450	0.022	0.011	0.081	0.020	0.049	0.000	0.246
13	6	0.302	0.037	0.019	0.140	0.034	0.084	0.000	0.000
14	1	0.043	0.027	0.000	0.000	0.000	0.017	0.000	0.000
14	2	0.105	0.023	0.024	0.000	0.047	0.015	0.000	0.000
14	3	0.069	0.027	0.027	0.000	0.000	0.017	0.000	0.000
14	4	0.033	0.034	0.000	0.000	0.000	0.000	0.000	0.000
14	5	0.449	0.022	0.022	0.081	0.043	0.014	0.000	0.246
14	6	0.302	0.037	0.038	0.140	0.075	0.024	0.000	0.000
15	1	0.025	0.026	0.000	0.000	0.000	0.000	0.000	0.000
15	2	0.052	0.023	0.012	0.000	0.021	0.000	0.000	0.000
15	3	0.038	0.027	0.013	0.000	0.000	0.000	0.000	0.000
15	4	0.125	0.034	0.017	0.052	0.031	0.000	0.000	0.000
15	5	0.171	0.021	0.011	0.033	0.020	0.098	0.000	0.000
15	6	0.589	0.037	0.018	0.057	0.034	0.170	0.244	0.000

$f_{m,j,1}$ = the fraction of dispatches for type I CFS from atom j assigned to server n.
$f_{(m,0),j,2}$ = the fraction of dispatches for type II CFS from atom j assigned only to server m.
$f_{(m,n),j,2}$ = the fraction of dispatches for type II CFS fromatom j assigned to server m and server n.

Do you want to print out the results ? (Y/N) -> Y

[Printing is now in process.]

References

Chaiken, Jan M. and Peter Dormont, "A Patrol Car Allocation Model" *Management Science*, Vol. 24 (1978), pp. 1280-1300.

Chelst, Kenneth R. and Zvi Barlach, "Multiple Unit Dispatches in Emergency Services: Models to Estimate System Performance" The Institute of Management Sciences. (1981).

Green, L., "A Multiple Dispatch Queueing Model of Police Patrol Operations" *Management Science*, Vol. 30, No. 6 (1984), pp. 653-664.

Green, L., and P. Kolesar, "A Comparison of the Multiple Dispatch and M/M/C Priority Queueing Models of Police Patrol," *Management Science*, Vol. 30, No. 6, (1984), pp. 665-570.

Larson, R. C., *Urban Police Patrol Analysis*, MIT Press, Cambridge, Mass., 1972.

Larson, R. C., "A Hypercube Queueing Model for Facility Location and Redistricting in Urban Emergency Services" *Computers and Operations Res.*, Vol. 1, No. 1 (1974), pp. 67-94.

Larson, R. C., "Approximating the Performance of an Urban Emergency Service System" *Operations Res.*, Vol. 23, No. 5 (1975), pp. 845-868.

Larson, R. C. and Mark A. McKnew, "Police Patrol-initiated Activities within a Systems Queueing Model," *Management Science*, Vol. 28, No 7 (1982), pp. 759-774.

Schaack, Christian and Richard C. Larson, "An N-Server Cutoff Priority Queue," *Operations Res.*, Vol. 34, No. 2 (1986), pp. 257-266.

Schaack, C., and Richard C. Larson, "An N Server Cutoff Priority Queue Where Arriving Customers Request a Random Number of Servers," to appear in *Management Science*, 1988.

Acknowledgments

We are indebted to the National Institute of Justice (Grant #86-IJ-CX-0005) for supporting this work. The views expressed in this report are those of the authors and do not necessarily reflect the views and opinions of the sponsors.