Using Partial Queue-Length Information to Improve the Queue Inference Engine's Performance

Susan A. Hall and Richard C. Larson

OR 254-91 May 1991

Using Partial Queue-Length Information to Improve the Queue Inference Engine's Performance

Susan A. Hall Richard C. Larson

May 12, 1991

Abstract

The Queue Inference Engine (QIE) uses queue departure time data over a single congestion period to infer queue statistics. With partial queue-length information, the queue statistics become more accurate and the computational burden is reduced. We first consider the case in which we are given that the queue length never exceeded a given length L. We then consider the more general case in which we are given the times of all L-to-$(L+1)$ and $(L+1)$-to-L queue-length transitions. We present algorithms, parallel to the QIE algorithms, for deriving the queue statistics under the new conditioning information. We also present computational results, comparing both accuracy and computation time, under the QIE and the new algorithms, for several sample runs.

1 Introduction

The "Queue Inference Engine" or QIE was first reported in 1990 [Lars 90]. The purpose of the QIE is to deduce the operational behavior of a Poisson arrival queueing system from only the start and stop times of service for all customers served. Such service start and stop times constitute the transactional data set upon which the

computationally intensive QIE is based. From the time-ordered transactional data one observes that the signature of a queue is a service completion followed virtually immediately by a service commencement. The customer associated with the service commencement must have, with probability near one, been delayed in queue. By examining the transactional data for sequences of queue signatures, one can partition the time line into congestion periods (when all servers are continuously busy) and noncongestion periods (when not all servers are busy). All customers arriving during congestion periods are delayed in queue.

The QIE operates on the transactional data set, one congestion period at a time. Using a recursive procedure that requires no parameter estimation and virtually no assumptions about the service process, the QIE produces estimates of the following quantities for a congestion period:

1. mean wait in queue of a random customer;
2. time average queue length;
3. time-dependent mean number in queue;
4. probability distribution of queue length as experienced by a randomly arriving customer.

The underlying theory is based on order statistics (see, for example, [Barl 72] or [Davi 81]), particularly the fact that the N unordered arrival times of N Poisson-arriving customers over any fixed interval $[0, T]$ are i.i.d. (uniform). Larson has demonstrated an algorithm for a congestion period having N queued customers that requires $O(N^3)$ computations [Lars 90,Lars 91].

Other researchers have recently contributed to the advancement of the QIE. For instance, Bertsimas and Servi have proposed an $O(N^3)$ algorithm based on multidimensional integration [Bert 91]. Daley and Servi developed an $O(N^2(\ln N))$ algorithm that can approximate the QIE calculations with any prespecified level of precision [Dale 91].

In applications there is a need for the QIE in many service industries, including banks (human teller lines and ATM lines), airports, rapid food restaurants, hotels, urban transportation, and telecommunications. In some applications, the underlying arrival rate parameter may be slowly time-varying, thereby creating the possibility of inaccuracy in QIE calculations over lengthy congestion periods. Or in some settings there may be state-dependent balking, in which customers are increasingly less likely to join the queue as it becomes longer. Such state-dependent balking is not currently a solved problem in queue inference, and even if it were, its solution would undoubtedly involve the state-conditioned balking probabilities, which would represent in our view a violation of the parameter-free nature of the QIE. Even if neither of these two complications arise in practice, one might seek (perhaps due to constraints of a desk-top computer, or need to perform calculations in near real time) QIE algorithms that are faster than the current ones. Finally, perhaps one would like queue inferences with less variance in the estimated quantities.

The purpose of this paper is to develop QIE-like algorithms in an extended data-acquisition environment that (1) solves a new queue inference problem exactly and, from an engineering point of view, (2) provides at least partial solutions to the aforementioned complications. In this paper, our new assumption is that we know, in addition to service start and stop times, the times at which the queue undergoes transition from $M-1$ to M customers, or from M to $M-1$ customers. These transitions can be plotted over time as a binary "telegraph wave." The problem at hand is to derive algorithms ($O(N^3)$ or better) that infer queue behavior from the union of two data sets: the routine service start and stop times of the original QIE, and this new threshold-detecting telegraph wave. In practice, the telegraph wave could be generated from a pressure-sensitive mat placed in the queue channel or from one of several types of electronic field sensing devices.

2 QIE Background and Notation

Since this paper is based heavily on [Lars 90] and uses the same notation, we review the basic QIE and the notation that applies to it here.

The QIE model assumes Poisson arrivals to a queueing system, which may comprise a single or multiple servers. The service time distribution may be completely general. The Poisson arrival rate, λ, is assumed to be constant over the duration of a single congestion period. It is also assumed that whenever there is a queue, a service initiation follows immediately after any service completion. The data that are provided to the QIE include N, the total number of customers who waited in queue during a given congestion period; t_0, the start time of the congestion period, which here we assume to be 0; and $\mathbf{t} \equiv (t_1, t_2, \ldots, t_N)$, the times of service commencement for customers 1 through N ($t_1, t_2, \ldots, t_N$ may also be thought of as service completion or departure times), again assuming a start time of 0. (Customer 1 is the first customer to arrive and find all servers busy. Customer N is the last customer to commence service immediately after a departure: i.e., the departure at t_{N+1} creates an idle server.) The unobserved ordered arrival times of customers 1 through N are denoted $X_1, X_2, \ldots, X_N$. In order for customers 1 through N to comprise a congestion period, it must be the case that $X_1 \leq t_1$, $X_2 \leq t_2, \ldots, X_N \leq t_N$. This event is denoted $O(\mathbf{t})$. $N(t)$ denotes the cumulative number of arrivals to the system over the time period $(0, t]$, where $t \leq t_N$; and $N(t_1, t_2)$ denotes the cumulative number of arrivals to the system over the time period $(t_1, t_2]$, where $t_1 < t_2$. Finally, $N_Q(t)$ denotes the number of customers in queue at time t.

The quantities calculated by the QIE include $\overline{N}(t)$, the expected cumulative number of arrivals to the system, up to and including time t, conditional on the two events $O(\mathbf{t})$ and $N(t_N) = N$; $\overline{N}_Q(t)$, the expected number of customers in queue at time t, also conditional on the two events $O(\mathbf{t})$ and $N(t_N) = N$; $\overline{N}_Q$, the time-averaged queue length over a congestion period; $\overline{W}_Q$, the average delay in queue over a congestion period; Π_k, the probability that a randomly arriving customer finds k customers

in queue ($k = 0, 1, \ldots, N-1$); and $\bar{\ell}_Q$, the average queue length experienced by a randomly arriving customer. All of these quantities may be calculated as functions of the following quantities:

$$\beta_{ki}(\mathbf{t}) \equiv \Pr[X_k \leq t_i | O(\mathbf{t}), N(t_N) = N], \quad 1 \leq i \leq N,\ 1 \leq k \leq N$$

(See [Lars 90] for the specifics of calculating the queue statistics from the $\beta_{ki}(\mathbf{t})$'s.) Note that $\beta_{ki}(\mathbf{t}) = 1$ for $k \leq i$. The other values for $\beta_{ki}(\mathbf{t})$ are calculated via the following two equations:

$$\beta_{Ni}(\mathbf{t}) = \frac{\alpha_{Ni}(\mathbf{t})}{\alpha_{NN}(\mathbf{t})}, \quad 1 \leq i \leq N \tag{1}$$

$$\beta_{ki}(\mathbf{t}) = \beta_{(k+1),i}(\mathbf{t}) + \frac{\binom{N}{k} \alpha_{ki}(\mathbf{t}) \eta_{ki}(\mathbf{t})}{\alpha_{NN}(\mathbf{t})}, \quad 1 \leq i < k < N \tag{2}$$

The quantities $\alpha_{ki}(\mathbf{t})$ are defined as follows

$$\alpha_{ki}(\mathbf{t}) \equiv \Pr[X_1 \leq t_1,\ X_2 \leq t_2, \ldots, X_i \leq t_i, \ldots, X_k \leq t_i | N(t_N) = k], \quad k \geq i$$

and are computed recursively via the following two equations:

$$\alpha_{k1}(\mathbf{t}) = \left(\frac{t_1}{t_N}\right)^k, \quad 1 \leq k \leq N \tag{3}$$

$$\alpha_{ki}(\mathbf{t}) = \sum_{j=0}^{k-i+1} \binom{k}{j} \alpha_{(k-j),(i-1)}(\mathbf{t}) \left(\frac{t_i - t_{i-1}}{t_N}\right)^j, \quad 1 < i \leq k \leq N \tag{4}$$

Finally, the quantities $\eta_{ki}(\mathbf{t})$ are defined as

$$\eta_{ki}(\mathbf{t}) \equiv \Pr[t_i < X_{k+1} \leq t_{k+1}, \ldots, t_i < X_N \leq t_N | N(t_N) = N - k], \quad k \geq i$$

and are calculated via

$$\eta_{Ni}(\mathbf{t}) \equiv 1, \quad 1 \leq i \leq N \tag{5}$$

$$\eta_{ki}(\mathbf{t}) \equiv 0, \quad 1 \leq k < i \leq N \tag{6}$$

$$\eta_{ki}(\mathbf{t}) = \sum_{j=0}^{N-k} \binom{N-k}{j} \eta_{(k+j)(i+1)}(\mathbf{t}) \left(\frac{t_{i+1} - t_i}{t_N}\right)^j, \quad 1 \leq i \leq k < N \tag{7}$$

This is the extent of the QIE notation that will be used in this paper. For the details of the definitions and the proofs of the recursion algorithms, please see [Lars 90].

3 Addition of Maximum Queue Length Information to Transactional Data

The calculation of the entire β-matrix (and, hence, of the queue statistics of interest) is $O(N^3)$ [Lars 91]. Partial queue-length information can reduce the computational effort, while simultaneously improving the accuracy of the QIE-calculated statistics. We first consider the addition of information regarding the maximum queue length. The algorithm developed in this section will then be used as a building block for algorithms which use other queue-length data. Specifically, suppose we are told that, during a congestion period of N customers, the queue Length never exceeds L customers, where $0 < L < N$. This will force some of the quantities in the β-matrix to be 0, thereby reducing computational effort. For example, with $N = 10$ and $L = 9$, we are told that the queue length never exceeds 9 customers. Therefore, $\beta_{10,1}(\mathbf{t})$, the probability that all 10 customers arrived before the first service completion (hence causing a queue length of 10), is zero. We proceed with a description of how this information affects calculation of the new beta-matrix (which we call the β^L-matrix).

First, we define $Q^L(t_1, t_2)$ to be the event $N_Q(t) \leq L, t_1 < t \leq t_2$. Then we define the following quantities:

$$\tilde{\alpha}^L_{ki}(\mathbf{t}) \equiv \Pr[X_1 \leq t_1, \ldots, X_i \leq t_i, \ldots, X_k \leq t_i, Q^L(0, t_i) | N(t_i) = k], \quad k \geq i \tag{8}$$

$$\tilde{\eta}^L_{ki}(\mathbf{t}) \equiv \Pr[t_i < X_{k+1} \leq t_{k+1}, \ldots, t_i < X_N \leq t_N, Q^L(t_i, t_N) | N_Q(t_i^+) = k - i, \; N(t_i, t_N) = N - k], \quad k \geq i \tag{9}$$

$$\beta^L_{ki}(\mathbf{t}) \equiv \Pr[X_k \leq t_i | Q^L(0, t_N), O(\mathbf{t}), N(t_N) = N], \quad 1 \leq i \leq N, \; 1 \leq k \leq N \tag{10}$$

Note that we use tilde's here to indicate that, for example, $\tilde{\alpha}^L_{ki}(\mathbf{t})$ is conditioned on $N(t_i) = k$, rather than on $N(t_N) = k$, and similarly for $\tilde{\eta}^L_{ki}(\mathbf{t})$. Also note that in the case of $\tilde{\eta}^L_{ki}(\mathbf{t})$, we have made explicit the queue length at time t_i^+, since maintaining a queue length less than L between t_i and t_N depends on that queue length. (Note that, in the case of $\tilde{\alpha}^L_{ki}(\mathbf{t})$, there is the implicit condition that $N_Q(0^+) = 0$.) All of

the queue statistics we obtain will be conditioned on the event $Q^L(0,t_N)$, and so they may be calculated similarly to the method described in Section 2, via the $\beta^L_{ki}(\mathbf{t})$'s. We proceed in an approach parallel to that used in [Lars 90] and begin by noting the following two lemmata:

Lemma 3.1 $\tilde{\alpha}^L_{ki}(\mathbf{t}) = 0, \quad k-i \geq L.$

Proof: Assume $k-i \geq L$. In order that we may have $X_k \leq t_i$, it must be the case that just prior to t_i, there will have been k arrivals and $i-1$ departures to a system all of whose servers were busy at $t = 0^+$, and hence $N_Q(t_i^-) = k-i+1 \geq L+1$ (since $k-i \geq L$). Therefore, we cannot simultaneously have $X_k \leq t_i$ and $N_Q(t) \leq L, 0 < t \leq t_i$, when $k-i \geq L$. ∎

Lemma 3.2 $\tilde{\eta}^L_{ki}(\mathbf{t}) = 0, \quad k-i > L.$

Proof: Assume $k-i > L$. Then $N_Q(t_i^+) > L$ and $\Pr[Q^L(t_i,t_N)|N_Q(t_i^+) > L] = 0$, and hence $\tilde{\eta}^L_{ki}(\mathbf{t}) = 0$ when $k-i > L$. ∎

Now, the first step is to calculate the values $\tilde{\alpha}^L_{ki}(\mathbf{t})$, for $k \geq i$, thereby also determining $\tilde{\alpha}^L_{NN}(\mathbf{t}) = \Pr[O(\mathbf{t}), Q^L(0,t_N)|N(t_N) = N]$. First, it should be obvious from the definition of $\tilde{\alpha}^L_{ki}(\mathbf{t})$ that:

$$\tilde{\alpha}^L_{k1}(\mathbf{t}) = 1, \quad 0 \leq k-1 < L \tag{11}$$

The following lemma is also easily derived from the definition of $\tilde{\alpha}^L_{ki}(\mathbf{t})$.

Lemma 3.3

$$\tilde{\alpha}^L_{ki}(\mathbf{t}) = \sum_{j=0}^{k-i+1} \binom{k}{j} \left(\frac{t_{i-1}}{t_i}\right)^{k-j} \left(\frac{t_i - t_{i-1}}{t_i}\right)^j \tilde{\alpha}^L_{(k-j),(i-1)}(\mathbf{t}),$$
$$2 \leq i \leq N, \ 2 \leq k \leq N, \ 0 \leq k-i < L$$

Proof: This lemma results from an expansion of the definition of $\tilde{\alpha}^L_{ki}(\mathbf{t})$. We have:

$$\tilde{\alpha}^L_{ki}(\mathbf{t}) \equiv \Pr[X_1 \leq t_1, \ldots, X_i \leq t_i, \ldots, X_k \leq t_i, Q^L(0,t_i)|N(t_i) = k]$$

$$
\begin{aligned}
&= \sum_{j=0}^{k-i+1} \Pr[X_1 \le t_1, \ldots, X_{i-1} \le t_{i-1}, \ldots, X_{k-j} \le t_{i-1}, \\
&\quad t_{i-1} < X_{k-j+1} \le t_i, \ldots, t_{i-1} < X_k \le t_i, Q^L(0, t_{i-1}), Q^L(t_{i-1}, t_i), \\
&\quad N(t_{i-1}) = k-j, N(t_{i-1}, t_i) = j | N(t_i) = k] \qquad (12) \\
&= \sum_{j=0}^{k-i+1} \Pr[X_1 \le t_1, \ldots, X_{i-1} \le t_{i-1}, \ldots, X_{k-j} \le t_{i-1}, Q^L(0, t_{i-1}), \\
&\quad Q^L(t_{i-1}, t_i), N(t_{i-1}) = k-j, N(t_{i-1}, t_i) = j | N(t_i) = k]
\end{aligned}
$$

where the last equality follows since the events $N(t_{i-1}) = k - j$ and $N(t_{i-1}, t_i) = j$ imply the event $t_{i-1} < X_{k-j+1} \le t_i \ldots, t_{i-1} < X_k \le t_i$. By noting that events in non-overlapping time intervals in a Poisson process are independent, we continue:

$$
\begin{aligned}
\tilde{\alpha}^L_{ki}(\mathbf{t}) &= \sum_{j=0}^{k-i+1} \Pr[N(t_{i-1}) = k-j, N(t_{i-1}, t_i) = j | N(t_i) = k] \times \Pr[X_1 \le t_1, \ldots, \\
&\quad X_{i-1} \le t_{i-1}, \ldots, X_{k-j} \le t_{i-1}, Q^L(0, t_{i-1}) | N(t_{i-1}) = k-j] \\
&\quad \times \Pr[Q^L(t_{i-1}, t_i) | N(t_{i-1}) = k-j, N(t_{i-1}, t_i) = j] \qquad (13)
\end{aligned}
$$

The first probability above is just a binomial term, since a given number of Poisson arrivals over a fixed time interval are distributed uniformly over that interval. The last probability above is 1 as long as $k - j - i + 1$ (the number in queue at t^+_{i-1}) plus j (the number of arrivals in $(t_{i-1}, t_i]$) does not exceed L (i.e. as long as $k - i < L$), and is 0 otherwise. Recognizing the middle probability as $\tilde{\alpha}^L_{(k-j),(i-1)}(\mathbf{t})$, we get the result of Lemma 3.3 above. ∎

Hence, we may first fill in column 1 of the $\tilde{\alpha}^L$-matrix, using Lemma 3.1 and Equation 11 above; then we may fill in the rest of the lower triangular half of the matrix one column at a time, using Lemmata 3.1 and 3.3. In order to determine all of the $\tilde{\alpha}^L_{ki}(\mathbf{t})$ values (including $\tilde{\alpha}^L_{NN}(\mathbf{t})$), we must calculate only a band of the lower triangular $N \times N$ matrix: i.e., in each column, we calculate at most L values. Each of these calculations involves at most L computations from the previous column. Finally, there are a total of N columns: hence, the overall algorithm for determining all of the elements of the $\tilde{\alpha}^L$-matrix is $O(NL^2)$.

Next, we determine the method for calculating the $\tilde{\eta}^L_{ki}(\mathbf{t})$'s. First, the following should be obvious from the definition of $\tilde{\eta}^L_{ki}(\mathbf{t})$, but we highlight it here:

$$\begin{aligned}\tilde{\eta}^L_{Ni}(\mathbf{t}) &\equiv \Pr[Q^L(t_i,t_N)|N_Q(t_i^+)=N-i, N(t_i,t_N)=0] \\ &= \begin{cases} 0, & 1 \le i < N-L \\ 1, & N-L \le i < N \end{cases} \end{aligned} \tag{14}$$

The general recursion for finding $\tilde{\eta}^L_{ki}(\mathbf{t})$ is given by the following lemma:

Lemma 3.4

$$\tilde{\eta}^L_{ki}(\mathbf{t}) = \sum_{j=0}^{\min(N-k,L-k+i)} \binom{N-k}{j} \left(\frac{t_{i+1}-t_i}{t_N-t_i}\right)^j \left(\frac{t_N-t_{i+1}}{t_N-t_i}\right)^{N-k-j} \tilde{\eta}^L_{(k+j),(i+1)}(\mathbf{t}),$$
$$1 \le i \le N-1, \;\; 1 \le k \le N-1, \;\; 0 \le k-i \le L$$

Proof: This proof is very similar to that of Lemma 3.3 and results from an expansion of the definition of $\tilde{\eta}^L_{ki}(\mathbf{t})$. We have:

$$\begin{aligned}\tilde{\eta}^L_{ki}(\mathbf{t}) &\equiv \Pr[t_i < X_{k+1} \le t_{k+1}, \ldots, t_i < X_N \le t_N, Q^L(t_i,t_N)|N_Q(t_i^+)=k-i, \\ &\quad N(t_i,t_N)=N-k] \\ &= \sum_{j=0}^{N-k} \Pr[t_i < X_{k+1} \le t_{i+1}, \ldots, t_i < X_{k+j} \le t_{i+1}, t_{i+1} < X_{k+j+1} \le t_{k+j+1}, \ldots, \\ &\quad t_{i+1} < X_N \le t_N, Q^L(t_i,t_{i+1}), Q^L(t_{i+1},t_N), N(t_i,t_{i+1})=j, \\ &\quad N(t_{i+1},t_N)=N-k-j|N_Q(t_i^+)=k-i, N(t_i,t_N)=N-k] \\ &= \sum_{j=0}^{N-k} \Pr[t_{i+1} < X_{k+j+1} \le t_{k+j+1}, \ldots, t_{i+1} < X_N \le t_N, Q^L(t_i,t_{i+1}), \\ &\quad Q^L(t_{i+1},t_N), N(t_i,t_{i+1})=j, N(t_{i+1},t_N)=N-k-j|N_Q(t_i^+)=k-i, \\ &\quad N(t_i,t_N)=N-k]\end{aligned}$$

The last equality follows because the events $N_Q(t_i^+)=k-i$ and $N(t_i,t_{i+1})=j$ imply the event $t_i < X_{k+1} \le t_{i+1}, \ldots, t_i < X_{k+j} \le t_{i+1}$. Again noting the independence of events in non-overlapping time intervals in a Poisson process, we get:

$$\tilde{\eta}^L_{ki}(\mathbf{t}) = \sum_{j=0}^{N-k} \Pr[N(t_i,t_{i+1})=j, N(t_{i+1},t_N)=N-k-j|N(t_i,t_N)=N-k]$$

$$\times \Pr[Q^L(t_i, t_{i+1}) | N_Q(t_i^+) = k - i, N(t_i, t_{i+1}) = j]$$
$$\times \Pr[t_{i+1} < X_{k+j+1} \leq t_{k+j+1}, \ldots, t_{i+1} < X_N \leq t_N, Q^L(t_{i+1}, t_N) |$$
$$N_Q(t_{i+1}^+) = k - i + j - 1, N(t_{i+1}, t_N) = N - k - j]$$

The first probability above is again a binomial term and contributes to the first three terms of Lemma 3.4. The second probability above will be 1 as long as $k - i$ (the number in queue at time t_i^+) plus j (the number of arrivals in $(t_i, t_{i+1}]$) does not exceed L: i.e., as long as $j \leq L - k + i$. This is the reason for the upper limit of the sum in the Lemma. Finally, consider the last probability. Most of the time, this probability is just $\tilde{\eta}^L_{(k+j),(i+1)}(\mathbf{t})$. However, in the special case when $k = i$ and $j = 0$, that probability becomes $\Pr[t_{i+1} < X_{i+1} \leq t_{i+1}, \ldots]$, a probability that should clearly be 0. As long as we make the following definition:

$$\tilde{\eta}^L_{ki}(\mathbf{t}) \equiv 0, \quad k - i < 0 \tag{15}$$

then Lemma 3.4 holds for all cases stipulated. ∎

Hence, we first set $\tilde{\eta}^L_{NN}(\mathbf{t}) = 1$, as given by Equation 14. Then we proceed to fill in columns from right to left, using Equations 14 and 15, and Lemmata 3.2 and 3.4. Again, as in the case of the $\tilde{\alpha}^L$-matrix, the only computations that need be done are in a band of the lower triangular half of the matrix: i.e., in each column, we calculate at most L values, and each of these calculations involves at most L computations from the column to the right. Finally, there are a total of N columns: hence, the overall algorithm for determining all the elements of the $\tilde{\eta}^L$-matrix is $O(NL^2)$.

The next step is to generate the $\beta^L_{ki}(\mathbf{t})$'s using the $\tilde{\alpha}^L_{ki}(\mathbf{t})$'s and the $\tilde{\eta}^L_{ki}(\mathbf{t})$'s. First, $\beta^L_{Ni}(\mathbf{t})$ may be found by the following Lemma, which is proved in a manner similar to the proofs of Lemmata 3.3 and 3.4.

Lemma 3.5

$$\beta^L_{Ni}(\mathbf{t}) = \begin{cases} 0, & 1 \leq i \leq N - L \\ \left(\frac{t_i}{t_N}\right)^N \frac{\tilde{\alpha}^L_{Ni}(\mathbf{t})}{\tilde{\alpha}^L_{NN}(\mathbf{t})}, & N - L < i \leq N \end{cases}$$

Proof:

$$\begin{aligned}
\beta^L_{Ni}(\mathbf{t}) &\equiv \Pr[X_N \le t_i | Q^L(0,t_N), O(\mathbf{t}), N(t_N) = N] \\
&= \frac{\Pr[X_1 \le t_1, \ldots, X_i \le t_i, \ldots, X_N \le t_i, Q^L(0,t_N) | N(t_N) = N]}{\Pr[Q^L(0,t_N), O(\mathbf{t}) | N(t_N) = N]} \\
&= \frac{1}{\tilde{\alpha}^L_{NN}(\mathbf{t})} \Pr[X_1 \le t_1, \ldots, X_i \le t_i, \ldots, X_N \le t_i, Q^L(0,t_i), Q^L(t_i,t_N), \\
&\quad N(t_i) = N, N(t_i,t_N) = 0 | N(t_N) = N] \\
&= \frac{1}{\tilde{\alpha}^L_{NN}(\mathbf{t})} \{\Pr[N(t_i) = N, N(t_i,t_N) = 0 | N(t_N) = N] \\
&\quad \times \Pr[X_1 \le t_1, \ldots, X_i \le t_i, \ldots, X_N \le t_i, Q^L(0,t_i) | N(t_i) = N] \\
&\quad \times \Pr[Q^L(t_i,t_N) | N_Q(t_i^+) = N - i, N(t_i,t_N) = 0]\} \\
&= \begin{cases} 0, & 1 \le i \le N - L \\ \left(\frac{t_i}{t_N}\right)^N \frac{\tilde{\alpha}^L_{Ni}(\mathbf{t})}{\tilde{\alpha}^L_{NN}(\mathbf{t})}, & N - L < i \le N \end{cases}
\end{aligned}$$

Note that the values of i for which $\beta^L_{Ni}(\mathbf{t}) = 0$ are those for which $\tilde{\alpha}^L_{Ni}(\mathbf{t}) = 0$, which are found from Lemma 3.1. ∎

Lemma 3.5 allows us to generate the bottom row of the β^L-matrix. As in the case of the β-matrix of Section 2, $\beta^L_{ki}(\mathbf{t}) = 1$ for $k \le i$. To find $\beta^L_{ki}(\mathbf{t})$ for $k > i$, we use the following Lemma, which is also proved in a manner similar to those preceding:

Lemma 3.6

$$\begin{aligned}
\beta^L_{ki}(\mathbf{t}) &= \beta^L_{(k+1),i}(\mathbf{t}) + \frac{1}{\tilde{\alpha}^L_{NN}(\mathbf{t})} \binom{N}{k} \left(\frac{t_i}{t_N}\right)^k \left(\frac{t_N - t_i}{t_N}\right)^{N-k} \tilde{\alpha}^L_{ki}(\mathbf{t}) \tilde{\eta}^L_{ki}(\mathbf{t}), \\
&\qquad 1 \le i \le N-2, \;\; 1 \le k \le N-1, \;\; 0 < k - i < L \\
\beta^L_{ki}(\mathbf{t}) &= 0, \quad 1 \le i, \;\; k \le N, \;\; L \le k - i \le N - 1
\end{aligned}$$

Proof:

$$\begin{aligned}
\beta^L_{ki}(\mathbf{t}) &\equiv \Pr[X_k \le t_i | Q^L(0,t_N), O(\mathbf{t}), N(t_N) = N] \\
&= \Pr[X_{k+1} \le t_i | Q^L(0,t_N), O(\mathbf{t}), N(t_N) = N] \quad + \\
&\quad \Pr[X_k \le t_i, X_{k+1} > t_i | Q^L(0,t_N), O(\mathbf{t}), N(t_N) = N]
\end{aligned}$$

Clearly the first probability above is just $\beta^L_{(k+1),i}(\mathbf{t})$. We now expand the second term above, calling it "Term 2:"

$$\begin{aligned}
\text{Term 2} &= \frac{\Pr[X_k \le t_i, X_{k+1} > t_i, Q^L(0,t_N), O(\mathbf{t}) | N(t_N) = N]}{\Pr[Q^L(0,t_N), O(\mathbf{t}) | N(t_N) = N]} \\
&= \frac{1}{\tilde{\alpha}^L_{NN}(\mathbf{t})} \Pr[X_1 \le t_1, \ldots, X_i \le t_i, \ldots, X_k \le t_i, \\
&\quad t_i < X_{k+1} \le t_{k+1}, \ldots, t_i < X_N \le t_N, Q^L(0,t_i), \\
&\quad Q^L(t_i,t_N), N(t_i) = k, N(t_i,t_N) = N-k | N(t_N) = N] \\
&= \frac{1}{\tilde{\alpha}^L_{NN}(\mathbf{t})} \{\Pr[N(t_i) = k, N(t_i,t_N) = N-k | N(t_N) = N] \\
&\quad \times \Pr[X_1 \le t_1, \ldots, X_i \le t_i, \ldots, X_k \le t_i, Q^L(0,t_i) | N(t_i) = k] \\
&\quad \times \Pr[t_i < X_{k+1} \le t_{k+1}, \ldots, t_i < X_N \le t_N, Q^L(t_i,t_N) | \\
&\quad N_Q(t_i^+) = k-i, N(t_i,t_N) = N-k]\}
\end{aligned}$$

Again, the last equality above arises from the independence of events in non-overlapping time intervals in a Poisson process. The first probability above gives rise to the second, third, and fourth terms of "Term 2" in the Lemma; the second probability is just $\tilde{\alpha}^L_{ki}(\mathbf{t})$; and the third is $\tilde{\eta}^L_{ki}(\mathbf{t})$. The values of $k-i$ for which $\beta^L_{ki}(\mathbf{t}) = 0$ are the same as those for which $\tilde{\alpha}^L_{ki}(\mathbf{t}) = 0$ and are found from Lemma 3.1. ∎

We now provide the following two definitions, which simplify the computational effort of filling in the β^L-matrix:

$$\alpha^L_{ki}(\mathbf{t}) \equiv \tilde{\alpha}^L_{ki}(\mathbf{t}) \left(\frac{t_i}{t_N}\right)^k \tag{16}$$

$$\eta^L_{ki}(\mathbf{t}) \equiv \tilde{\eta}^L_{ki}(\mathbf{t}) \left(\frac{t_N - t_i}{t_N}\right)^{N-k} \tag{17}$$

With these two definitions, Equation 11 and Lemmata 3.1 and 3.3 become:

$$\alpha^L_{k1}(\mathbf{t}) = \left(\frac{t_1}{t_N}\right)^k, \quad 0 \le k-1 < L \tag{18}$$

$$\alpha^L_{ki}(\mathbf{t}) = 0, \quad k-i \ge L \tag{19}$$

$$\begin{aligned}
\alpha^L_{ki}(\mathbf{t}) &= \sum_{j=0}^{k-i+1} \binom{k}{j} \left(\frac{t_i - t_{i-1}}{t_N}\right)^j \alpha^L_{(k-j),(i-1)}(\mathbf{t}), \\
&\quad 2 \le i \le N, \ 2 \le k \le N, \ 0 \le k-i < L
\end{aligned} \tag{20}$$

Also, Equations 14 and 15 and Lemmata 3.2 and 3.4 become:

$$\eta^L_{Ni}(\mathbf{t}) = \begin{cases} 0, & 1 \le i < N-L \\ 1, & N-L \le i < N \end{cases} \tag{21}$$

$$\eta^L_{ki}(\mathbf{t}) = 0, \quad k-i<0 \text{ or } k-i>L \tag{22}$$

$$\eta^L_{ki}(\mathbf{t}) = \sum_{j=0}^{\min(N-k,L-k+i)} \binom{N-k}{j} \left(\frac{t_{i+1}-t_i}{t_N}\right)^j \eta^L_{(k+j),(i+1)}(\mathbf{t}),$$
$$1 \le i \le N-1, \ 1 \le k \le N-1, \ 0 \le k-i \le L \tag{23}$$

Finally, Lemmata 3.5 and 3.6 become:

$$\beta^L_{Ni}(\mathbf{t}) = \begin{cases} 0, & 1 \le i \le N-L \\ \dfrac{\alpha^L_{Ni}(\mathbf{t})}{\alpha^L_{NN}(\mathbf{t})}, & N-L < i \le N \end{cases} \tag{24}$$

$$\beta^L_{ki}(\mathbf{t}) = \beta^L_{(k+1),i}(\mathbf{t}) + \frac{1}{\alpha^L_{NN}(\mathbf{t})} \binom{N}{k} \alpha^L_{ki}(\mathbf{t}) \eta^L_{ki}(\mathbf{t}),$$
$$1 \le i \le N-2, \ 1 \le k \le N-1, \ 0 < k-i < L \tag{25}$$

$$\beta^L_{ki}(\mathbf{t}) = 0, \quad 1 \le i, \ k \le N-1, \ k-i \ge L \tag{26}$$

We now have a complete method for determining the β^L-matrix, which we call the QIE^L algorithm. We first generate the α^L- and η^L-matrices (each of which is $O(NL^2)$ to compute); and then we multiply elements of those matrices to generate elements of the β^L-matrix. There are N columns of the β^L-matrix to be filled in; each of these columns has at most $L-1$ elements to be calculated; and each of these elements requires a single computation: hence, computation of the β^L-matrix, after computation of the other two matrices, is $O(NL)$, and the computational complexity of the QIE^L algorithm is $O(NL^2)$.

Because of the computational savings of the QIE^L algorithm, one might wish to use it, even if no maximum queue length data were available. Specifically, one can view the QIE^L algorithm as an *approximation* to the exact QIE algorithm, an approximation which disregards large-queue events. For long congestion periods, one can choose relatively modest values of L (on the order of 10) and still get results that

are very close to the exact QIE results. Computational data, comparing the exact QIE to the QIE^L algorithm (used as an approximation) are presented in Section 5. Also presented there are comparisons of the QIE to the QIE^L algorithm, when the maximum queue length data are actually available.

4 Addition of Partial Queue-Length Information to Transactional Data

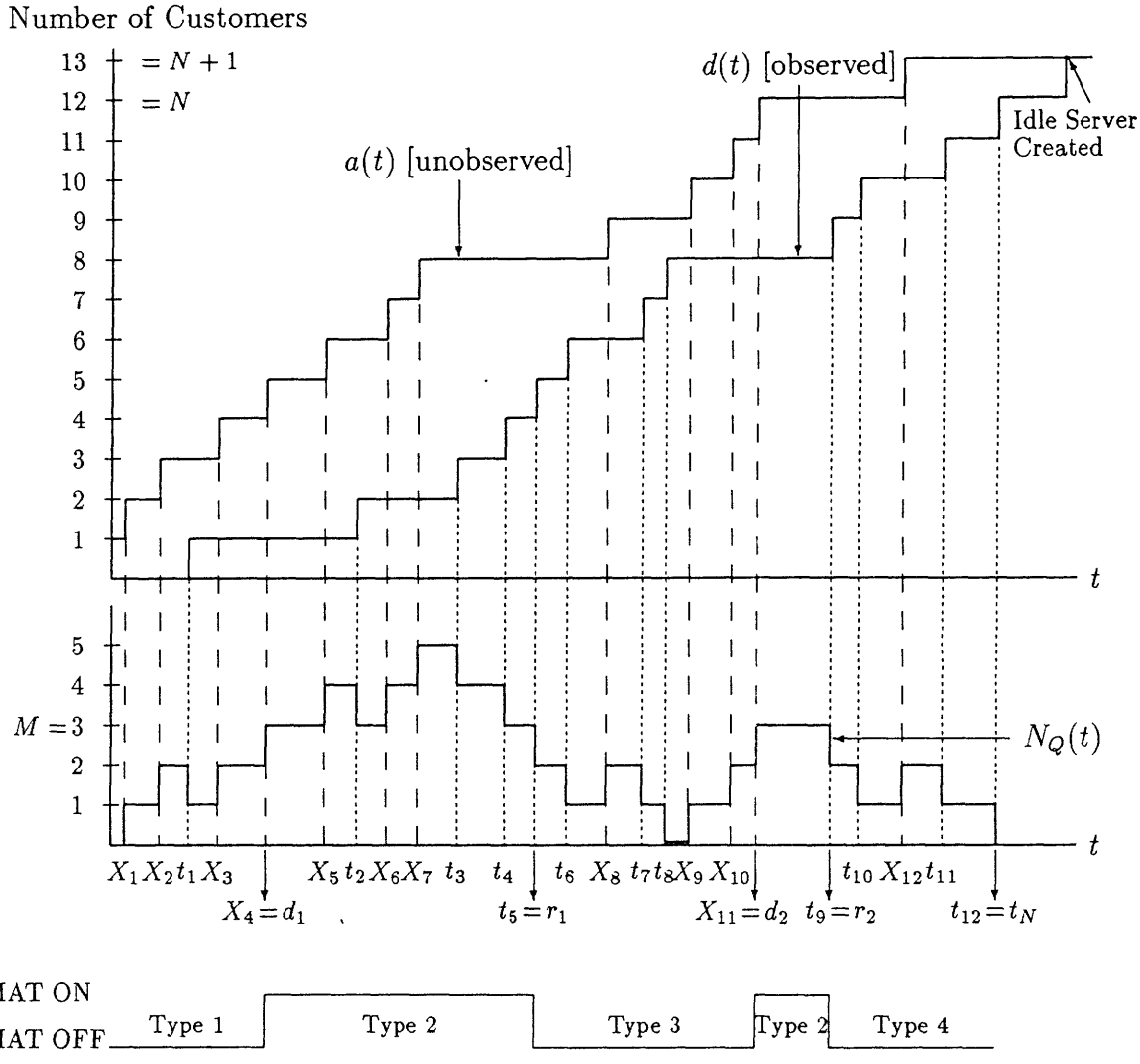
The QIE^L algorithm discussed in the last section raises the following interesting issue. Suppose that the queue actually had some sort of sensing mechanism, for example a pressure-sensitive mat placed at position M in the queue, such that we would be able to detect all queue transitions from $M-1$ to M, as well as all transitions from M to $M-1$. Here, we ignore the transients of customers stepping over the mat just to achieve a position in queue which is less than M. We also assume that there is no queue transit delay: that is, when a customer leaves the queue to enter service, the entire queue immediately shifts forward one position. Then, for a congestion period during which *no* transitions are observed, we have exactly the situation in the previous section, with $M = L + 1$: the mat information allows us to discard the large-queue events which we know did not occur.

For a congestion period during which mat transitions *are* observed, we clearly have new information at the points of transition and so should be able to use this information to improve the QIE performance. In addition, because the state of the queue is known exactly at the points of transition, we may break the congestion period down into "congestion period partitions" and analyze each of these separately, thereby significantly reducing the complexity of the computation. Specifically, assume, as before, that $t_1, t_2, \ldots, t_N$ are the times of service commencement for customers 1 through N. Now define a *mat cycle* as any period of time during which the mat is continuously depressed. Similarly, define a *non-mat cycle* as any period of time within

a single congestion period between two mat cycles. Say there are C mat Cycles during the congestion period, with $C \geq 0$ (hence, there are $C-1$ non-mat cycles whenever $C \geq 1$). Then define $d_1, d_2, \ldots, d_C$ as the times at which the mat is depressed; and define $r_1, r_2, \ldots, r_C$ as the times at which the mat is released. Note that any given mat release time must coincide with one of the t_i's. Figure 1 provides an example of a congestion period which has $N = 12$ customers who must wait in queue, with a mat position at $M = 3$, and two mat cycles ($C = 2$).

Whenever $C > 0$, the congestion period can be broken down into $2C+1$ congestion period partitions, each of which must be one of four distinct types. The first type of congestion period partition comprises the time $(0, d_1]$, i.e. the time from the beginning of the congestion period until the first time that there are M customers in queue (there must be at least $M-1$ arrivals prior to time d_1). The second type of congestion period partition comprises the time $(d_j, r_j]$, $j = 1, \ldots, C$, a single mat cycle. This is the time between any depression of the mat and the subsequent release. Note that the queue can grow to any size greater than or equal to M during this congestion period partition. The third type of congestion period partition comprises the time $(r_j, d_{j+1}]$, $j = 1, \ldots, C-1$, a single non-mat cycle (this type exists only when $C \geq 2$). This is the time within a single congestion period between any mat release and subsequent depression. The queue length can be anything between 0 and $M-1$ during this congestion period partition, although no server may be made idle, as this would cause the end of the congestion period. Finally, the fourth type of congestion period partition comprises the time $(r_C, t_N]$, i.e. the time between the end of the last mat cycle and the end of the congestion period. Including t_N, there must be at least $M-1$ departures during this congestion period partition, to empty out the $M-1$ customers who are in queue at time r_C^+. Again looking at Figure 1, because there are two mat cycles ($C = 2$), there are two type 2 congestion period partitions, and one each of types 1, 3, and 4.

We now proceed to analyze each of these four types of congestion period partitions



$a(t)$ = cumulative number of arrivals from start of congestion period through time t (includes arrival that initiates congestion period)

$d(t)$ = cumulative number of departures from start of congestion period through time t

$N_Q(t)$ = number of customers in queue at time $t = a(t) - d(t) - 1$

X_i = arrival time of i-th customer to arrive during congestion period ($i = 1, 2, \ldots, 12$)

t_i = service commencement time of i-th customer to arrive during congestion period ($i = 1, 2, \ldots, 12$)

d_j = arrival time of j-th customer whose arrival increases queue length from 2 to 3 ($j = 1, 2$)

r_j = departure time of j-th customer whose departure decreases queue length from 3 to 2 ($j = 1, 2$)

Figure 1: Sample Function for a Congestion Period with $N = 12$, $M = 3$, and $C = 2$.

separately. As we analyze each congestion period partition, we will be partially filling in another β-matrix, this one with entries $\beta_{ki}^{\mathcal{M}}(\mathbf{t})$, where

$$\beta_{ki}^{\mathcal{M}}(\mathbf{t}) \equiv \Pr[X_k \leq t_i | O(\mathbf{t}), N(t_N) = N, \mathcal{M}]$$

Here, $\mathcal{M}$ is used to denote the mat data, as contained in the telegraph wave, such as that depicted at the bottom of Figure 1. Finally, we discuss how to complete the $\beta^{\mathcal{M}}$-matrix and how to use it and additional mat information to derive the new queue statistics. We call this new algorithm which uses the mat data the QIE$^{\mathcal{M}}$ algorithm.

4.1 Type 1 Congestion Period Partition Analysis: from 0 in Queue to M in Queue

During the type 1 congestion period partition, which runs over the time interval $(0, d_1]$, we know that the queue length is less than M until the instant d_1, at which time the queue length increases from $M-1$ to M for the first time. This is similar to the situation we had with the QIE^L algorithm, i.e. we are given that the queue length does not exceed a specified value during a given period of time. We make use of an artificial bulk departure of the $M-1$ customers in queue at time d_1^- to complete the analogy in the following manner. Suppose that, rather than having an arrival at time d_1, we have $M-1$ departures, with the subsequent departure after d_1 causing the creation of an idle server and hence the end of the congestion period. An analysis of this congestion period, using the QIE^L algorithm with $L = M-1$, provides us with the probabilities of the various ways the Poisson arrivals could have occurred over the time interval $(0, d_1]$ while still obeying the queue length constraint and the usual arrival time inequalities, which is exactly what we are looking for.

Specifically, say there have been N_1 departures during the time interval $(0, d_1)$, i.e. $t_{N_1} < d_1$ but $t_{N_1+1} > d_1$. Also, let $L = M-1$. Then we define the vector

$\boldsymbol{\tau} = (\tau_1, \tau_2, \ldots, \tau_{N_1+L})$ as follows:

$$\tau_j \equiv \begin{cases} t_j & j = 1, 2, \ldots, N_1 \\ d_1 & j = N_1 + 1, \ldots, N_1 + L \end{cases}$$

If we now do a QIE^L analysis on a congestion period with total number of customers equal to $N_1 + L$, departure times $\tau_1, \tau_2, \ldots, \tau_{N_1+L}$, and maximum queue length of L, then we have:

Lemma 4.1

$$\beta_{ki}^{\mathcal{M}}(\mathbf{t}) = \beta_{ki}^{L}(\boldsymbol{\tau}) \quad 1 \leq i \leq N_1,\ 1 \leq k \leq N_1 + L$$

Proof: This is most easily seen by writing out the two definitions.

$$\beta_{ki}^{\mathcal{M}}(\mathbf{t}) = \Pr[X_k \leq t_i | X_1 \leq t_1, X_2 \leq t_2, \ldots, X_N \leq t_N, N(t_N) = N, \mathcal{M}]$$

The mat data tell us several things:

- $N(d_1^-) = N_1 + L$
- $N_Q(t) \leq L, \quad 0 \leq t < d_1$
- $X_{N_1+1} < X_{N_1+2} < \cdots < X_{N_1+L} < d_1$
- $X_{N_1+L+1} = d_1$
- $d_1 < X_{N_1+L+2} < \cdots < X_N$

But we are looking for the probability that $X_k \leq t_i$, for $i \leq N_1$ and $k \leq N_1 + L$. Given the exact number of arrivals which occurred prior to time d_1 (the first item of mat data above), the times of subsequent arrivals (the events defined in the last two items of mat data above) are independent of the event $X_k \leq t_i$. Hence we have the following:

$$\begin{aligned} \beta_{ki}^{\mathcal{M}}(\mathbf{t}) &= \Pr[X_k \leq t_i | N_Q(t) \leq L,\ 0 \leq t < d_1,\ X_1 \leq t_1, \ldots, X_{N_1} \leq t_{N_1}, \\ &\qquad X_{N_1+1} < d_1, \ldots, X_{N_1+L} < d_1, N(d_1^-) = N_1 + L] \\ &= \Pr[X_k \leq \tau_i | Q^L(0, \tau_{N_1+L}), O(\boldsymbol{\tau}), N(\tau_{N_1+L}) = N_1 + L] \\ &= \beta_{ki}^{L}(\boldsymbol{\tau}), \quad 1 \leq i \leq N_1,\ 1 \leq k \leq N_1 + L \quad \blacksquare \end{aligned}$$

Although the QIE^L algorithm may be used directly to calculate the $\beta^L_{ki}(\boldsymbol{\tau})$ values, a modified version saves some amount of computational effort. First note that, since we only use values of $\beta^L_{ki}(\boldsymbol{\tau})$ which have $i \leq N_1$, then we need not compute the full $(N_1 + L) \times (N_1 + L)$ matrix. Since computation of one column of a β^L-matrix is independent of computation of any other column, we compute only the first N_1 columns. (Actually, because of the definition of $\boldsymbol{\tau}$, all elements of the last L columns are 1's and so would not require additional computation anyway.)

Second, because $\tau_{N_1+1} = \tau_{N_1+2} = \cdots = \tau_{N_1+L}$, many elements of the α^L-matrix and the η^L-matrices may be filled in without any computation. Specifically, for the case of the α^L-matrix, we have:

Lemma 4.2 $\alpha^L_{ki}(\boldsymbol{\tau}) = \alpha^L_{k,(N_1+1)}(\boldsymbol{\tau}), \quad k > N_1, \quad i > N_1, \quad k \geq i.$

Proof: When $k > N_1$ and $i > N_1$, we have

$$\begin{aligned}
\alpha^L_{ki}(\boldsymbol{\tau}) &\equiv \Pr[X_1 \leq \tau_1, \ldots, X_{N_1+1} \leq \tau_{N_1+1}, \ldots, X_i \leq \tau_i, \ldots, X_k \leq \tau_i, \ldots | \ldots] \\
&= \Pr[X_1 \leq \tau_1, \ldots, X_{N_1+1} \leq d_1, \ldots, X_i \leq d_1, \ldots, X_k \leq d_1, \ldots | \ldots] \\
&= \alpha^L_{k,(N_1+1)}(\boldsymbol{\tau}) \qquad \blacksquare
\end{aligned}$$

Hence we need only compute the first N_1 columns of the α^L-matrix, plus the last element of column $N_1 + 1$, which is $\alpha^L_{(N_1+L),(N_1+1)}(\boldsymbol{\tau}) = \alpha^L_{(N_1+L),(N_1+L)}(\boldsymbol{\tau})$, since those are the only values needed to fill in the first N_1 columns of the β^L-matrix (see Equation 25).

Finally, consider the η^L-matrix. Note that Equations 21 and 22 still hold. Now, however, for the last L columns of the matrix we have:

Lemma 4.3 $\eta^L_{ki}(\boldsymbol{\tau}) = 0, \quad N_1 + 1 \leq i \leq N_1 + L, \quad 1 \leq k \leq N_1 + L - 1$

Proof: We already know that $\eta^L_{ki}(\boldsymbol{\tau}) = 0$ for $k < i$, from Equation 22. When $k \geq i$ and $i > N_1$, we have that:

$$\eta^L_{ki}(\boldsymbol{\tau}) = \Pr[\tau_i < X_{k+1} \leq \tau_{k+1}, \ldots | \ldots]$$

$$= \Pr[d_1 < X_{k+1} \leq d_1, \ldots | \ldots]$$
$$= 0 \quad \blacksquare$$

Hence, we fill in the bottom row of the η^L-matrix using Equation 21, and then we fill in the last L columns with zeroes. Finally, we use Equations 22 and 23 to fill in the first N_1 columns, and we are done.

Note that these modifications to the QIE^L algorithm do indeed save some computational effort. Because we actually do computations in only N_1 columns of the three matrices, the overall computational complexity for this modified type 1 congestion period partition analysis is $O(N_1 L^2)$, rather than $O((N_1 + L)L^2)$ in the unmodified case.

4.2 Type 2 Congestion Period Partition Analysis: A Single Mat Cycle

This type of congestion period partition runs over the time interval $(d_j, r_j]$, $1 \leq j \leq C$. There are no restrictions on queue size during this congestion period partition, except that it be greater than or equal to M, which suggests that some modification of the standard QIE algorithm be applied. Specifically, consider our original queueing system, with a waiting room added for the first M people in queue; additional customers must wait outside. A "congestion period" for this new system begins when the waiting room becomes full (this occurs at time d_j in the original system, when the queue length goes from $M-1$ to M). The congestion period terminates when there is again space in the waiting room (this occurs at time r_j in the original system, when the queue length goes from M to $M-1$). Analysis of this new congestion period, using the original QIE algorithm, gives us the desired arrival time probabilities, as described below.

Say that there have been exactly D_2 departures up to time d_j, and say that there are N_2 departures during the time interval (d_j, r_j), *not including the departure which*

occurs at time r_j. Hence we have $t_{D_2} < d_j, d_j < t_{D_2+1} < t_{D_2+2} < \cdots < t_{D_2+N_2} < r_j$, and $t_{D_2+N_2+1} = r_j$. Now define the following quantities:

$$\begin{aligned} \tau_i &\equiv t_{D_2+i} - d_j, \quad i = 1, 2, \ldots, N_2 \\ X'_k &\equiv X_{k+D_2+M} - d_j, \quad k = 1, 2, \ldots, N_2 \end{aligned}$$

Then we have the following:

Lemma 4.4 $\beta^{\mathcal{M}}_{(k+D_2+M),(D_2+i)}(\mathbf{t}) = \beta'_{ki}(\boldsymbol{\tau}), \quad k = 1, 2, \ldots, N_2, \; i = 1, 2, \ldots, N_2$

where the prime on $\beta'_{ki}(\boldsymbol{\tau})$ indicates that it is defined in terms of X'_k's rather than X_k's.

Proof: Again we write out definitions. We have:

$$\beta^{\mathcal{M}}_{(k+D_2+M),(D_2+i)}(\mathbf{t}) = \Pr[X_{k+D_2+M} \leq t_{D_2+i} | O(\mathbf{t}), N(t_N) = N, \mathcal{M}]$$

The mat data tell us the following:

- $N(d_j^+) = D_2 + M$
- $N(r_j) = D_2 + M + N_2$
- $N_Q(t) \geq M, \quad d_j < t < r_j$
- $X_1 < X_2 < \cdots < X_{D_2+M} = d_j$
- $d_j < X_{D_2+M+i} \leq t_{D_2+i}, \quad 1 \leq i \leq N_2$
- $r_j < X_{D_2+M+N_2+1} < \cdots < X_N$

The second-to-last item above is derived from the queue length constraint: in order to have at least M in queue after the $(D_2 + i)$-th departure, there must have been at least $D_2 + M + i$ arrivals by that time. So we may continue:

$$\begin{aligned} \beta^{\mathcal{M}}_{(k+D_2+M),(D_2+i)}(\mathbf{t}) &= \Pr[X_{k+D_2+M} \leq t_{D_2+i} | d_j < X_{D_2+M+1} \leq t_{D_2+1}, \ldots, \\ &\qquad d_j < X_{D_2+M+N_2} \leq t_{D_2+N_2}, N(d_j, r_j) = N_2] \\ &= \Pr[X'_k \leq \tau_i | X'_1 \leq \tau_1, \ldots, X'_{N_2} \leq \tau_{N_2}, N(\tau_{N_2}) = N_2] \\ &= \beta'_{ki}(\boldsymbol{\tau}), \quad k = 1, 2, \ldots, N_2, \; i = 1, 2, \ldots, N_2 \end{aligned}$$

The first equality above results because the arrival times in question are known to have occurred between d_j and r_j, and hence are independent of events occurring prior to d_j or after r_j. The second equality above results after subtracting d_j from all times. Note that the conditioning data not only tell us that $N(d_j, r_j) = N(d_j, t_{D_2+N_2+1}) = N_2$, but also that $N(d_j, t_{D_2+N_2}) = N_2$: i.e., in order for the queue length to decrease from M to $M-1$ at $t_{D_2+N_2+1}$, given that $N_Q(t^+_{D_2+N_2}) \geq M$, there must have been exactly zero arrivals in $(t_{D_2+N_2}, t_{D_2+N_2+1}]$. Finally, the third equality results from the definition of $\beta'_{ki}(\tau)$. ∎

Hence, application of the original QIE algorithm to the type 2 congestion period partition fills in another portion of the $\beta^{\mathcal{M}}$-matrix. Note that, for $N \gg M$, one of these congestion period partitions could be very long, and hence its concomitant standard QIE analysis, with computational complexity of $O(N_2^3)$ could be computationally burdensome. Of course, it is always possible to apply the QIE^L algorithm (used as an approximation) to any type 2 congestion period partition analysis, which would result in less accuracy but would reduce computational effort.

4.3 Type 4 Congestion Period Partition Analysis: from M in Queue to 0 in Queue

We discuss this before the type 3 congestion period partition, because the type 3 analysis is in some sense a combination of the type 1 and the type 4 analyses, and is thus better discussed after a description of the other two. The type 4 congestion period partition runs over the time period $(r_C, t_N]$. We assume that there have been exactly D_4 departures up to *and including* time r_C, i.e. $t_{D_4} = r_C$. We also assume that there are N_4 additional departures in the time interval $(r_C, t_N]$, i.e. $N = D_4 + N_4$. We know that at time r_C, the queue length drops from M to $M-1$, and hence, at time r_C^+, there are exactly $M-1 = L$ customers in queue. We also know that at time t_N^+, there are exactly 0 customers in queue. Hence, there are exactly $N_4 - L$ arrivals during the time interval $(r_C, t_N]$. Now, we know that during this congestion period

partition, the queue length never exceeds L customers: because of this queue-length constraint, we make use of a modified version of the QIE^L algorithm, this time by introducing an artificial bulk arrival of L customers at time r_C.

First we introduce some new definitions. The following two definitions allow us to shift all times back to the origin:

$$\begin{aligned} \tau_i &\equiv t_{D_4+i} - r_C, \quad i = 1, 2, \ldots, N_4 \\ X'_k &\equiv X_{D_4+L+k} - r_C, \quad k = 1, 2, \ldots, N_4 - L \end{aligned}$$

That is, the X'_k's are just time-shifted versions of the arrival times of the last $N_4 - L$ customers to arrive during the congestion period. Now we assume that at time $\tau = 0$, a congestion period was initiated by an arrival, and at time $\tau = 0^+$, we had L additional arrivals, causing the queue length instantaneously to grow to L. Specifically, if $Y_1, Y_2, \ldots, Y_L$ represent the arrival times for these customers, then $Y_1, Y_2, \ldots, Y_L$ are deterministic, with

$$\Pr[Y_k = \tau] = \begin{cases} 1 & \text{for } \tau = 0^+, \quad k = 1, 2, \ldots, L \\ 0 & \text{otherwise} \end{cases} \tag{27}$$

Finally, we define the following:

$$Z_k \equiv \begin{cases} Y_k & k = 1, 2, \ldots, L \\ X'_{k-L} & k = L+1, \ldots, N_4 \end{cases}$$

We now proceed in a manner parallel to that used to derive the β^L-matrix in Section 3, except that the Z_k's represent the arrival times (rather than X_k's), the τ_i's represent the departure times (rather than t_i's), N_4 represents the size of the congestion period (rather than N), and finally, α, η, and β are replaced by their roman equivalents (a, h, and b).

First consider the quantity $\tilde{a}^L_{ki}(\boldsymbol{\tau})$, which is completely analogous to $\tilde{\alpha}^L_{ki}(\mathbf{t})$, but defined in terms of Z's rather than X's:

$$\tilde{a}^L_{ki}(\boldsymbol{\tau}) \equiv \Pr[Z_1 \le \tau_1, \ldots, Z_i \le \tau_i, \ldots, Z_k \le \tau_i, Q^L(0, \tau_i) | N(\tau_i) = k], \quad k \ge i$$

We still have that

$$\tilde{a}_{ki}^{L}(\boldsymbol{\tau}) = 0, \quad k - i \geq L$$

by the identical reasoning used to prove Lemma 3.1. We also have the following:

Lemma 4.5 $\tilde{a}_{ki}^{L}(\boldsymbol{\tau}) = 1, \quad 1 \leq i \leq k \leq L$

Proof: When $k \leq L$, $\tilde{a}_{ki}^{L}(\boldsymbol{\tau}) = Pr[Y_1 \leq \tau_1, \ldots, Y_i \leq \tau_i, \ldots, Y_k \leq \tau_i, Q^L(0, \tau_i) | N(\tau_i) = k]$. But since the Y_k's all occur deterministically at $\tau = 0^+$, then these probabilities will be 1. ∎

Hence, the first column of the $\tilde{a}^L$-matrix consists of L ones followed by $N_4 - L$ zeroes. The recursion formula for finding the remainder of the $\tilde{a}_{ki}^{L}(\boldsymbol{\tau})$ values is given by the following:

Lemma 4.6

$$\tilde{a}_{ki}^{L}(\boldsymbol{\tau}) = \sum_{j=0}^{\min(k-i+1,k-L)} \binom{k-L}{j} \left(\frac{\tau_{i-1}}{\tau_i}\right)^{k-j-L} \left(\frac{\tau_i - \tau_{i-1}}{\tau_i}\right)^{j} \tilde{a}_{(k-j),(i-1)}^{L}(\boldsymbol{\tau})$$
$$L < k \leq N_4, \quad 0 \leq k - i < L$$

Proof: The proof is almost identical to that used in proving Lemma 3.3 and so will not be reproduced here: rather, the differences in the proofs will be highlighted, assuming that we replace the X_k's with Z_k's, the t_i's with τ_i's, and N with N_4. First, consider Equation 12: since for the first L values of k, Z_k is deterministic and occurs at $\tau = 0^+$, then $\Pr[\ldots, \tau_{i-1} < Z_{k-j+1} \leq \tau_i, \ldots, |\ldots]$ will be zero whenever $k - j + 1 \leq L$. Hence, the upper limit of the sum is set to the minimum of $k - i + 1$ and $k - L$. Now consider the first probability in Equation 13. Since L of the k arrivals to occur during $(0, \tau_i]$ are deterministic, then we are looking for the probability that the remaining $k - L$ Poisson arrivals are distributed between the two time intervals such that $k - j - L$ of them occur in $(0, \tau_{i-1}]$ and j of them occur in $(\tau_{i-1}, \tau_i]$. This probability is found by replacing k with $k - L$ in the combination term and in the exponent of the first time ratio. The rest of the proof follows that of Lemma 3.3 exactly. ∎

Finally, we make the following definition which will simplify later computations:

$$a_{ki}^L(\boldsymbol{\tau}) \equiv \tilde{a}_{ki}^L(\boldsymbol{\tau}) \left(\frac{\tau_i}{\tau_{N_4}}\right)^{k-L}, \quad L < k \leq N_4, \quad 0 \leq k-i < L$$

This definition allows us to simplify Lemma 4.6 to the following:

$$\begin{aligned} a_{ki}^L(\boldsymbol{\tau}) &= \sum_{j=0}^{\min(k-i+1,k-L)} \binom{k-L}{j} \left(\frac{\tau_i - \tau_{i-1}}{\tau_{N_4}}\right)^j \tilde{a}_{(k-j),(i-1)}^L(\boldsymbol{\tau}), \\ & \quad L < k \leq N_4, \quad 0 \leq k-i < L \end{aligned}$$

Next we consider the equivalent quantity to $\tilde{\eta}_{ki}^L(\mathbf{t})$, namely $\tilde{h}_{ki}^L(\boldsymbol{\tau})$. (See Equation 9.) Because $\tilde{h}_{ki}^L(\boldsymbol{\tau})$ defines probabilities of arrivals in $(\tau_i, \tau_{N_4}]$, it will be zero whenever $k+1 \leq L$. Other than that, $\tilde{h}_{ki}^L(\boldsymbol{\tau})$ calculates probabilities only of Poisson arrivals and so is identical to $\tilde{\eta}_{ki}^L(\mathbf{t})$. We may convert $\tilde{h}_{ki}^L(\boldsymbol{\tau})$ to $h_{ki}^L(\boldsymbol{\tau})$ in the same manner as that used in Equation 17. Having done so, the following summarizes the calculation of the values of $h_{ki}^L(\boldsymbol{\tau})$:

$$\begin{aligned} h_{N_4,i}^L(\boldsymbol{\tau}) &= \begin{cases} 0, & 1 \leq i < N_4 - L \\ 1, & N_4 - L \leq i \leq N_4 \end{cases} \\ h_{ki}^L(\boldsymbol{\tau}) &= \sum_{j=0}^{\min(N_4-k,L-k+i)} \binom{N_4-k}{j} \left(\frac{\tau_{i+1} - \tau_i}{\tau_{N_4}}\right)^j h_{(k+j),(i+1)}^L(\boldsymbol{\tau}), \\ & \quad L \leq k \leq N_4 - 1, \quad 0 \leq k-i \leq L \\ h_{ki}^L(\boldsymbol{\tau}) &= 0, \quad \text{otherwise} \end{aligned}$$

Finally, we come to the calculation of $b_{ki}^L(\boldsymbol{\tau})$, which we will ultimately use to fill in the bottom right-hand corner of the $\beta^{\mathcal{M}}$-matrix. Recall that $b_{ki}^L(\boldsymbol{\tau})$ is completely analogous to $\beta_{ki}^L(\mathbf{t})$, but defined in terms of Z_k's rather than X_k's:

$$\begin{aligned} b_{ki}^L(\boldsymbol{\tau}) &\equiv \Pr[Z_k \leq \tau_i | Q^L(0, \tau_{N_4}), Z_1 \leq \tau_1, \ldots, Z_{N_4} \leq \tau_{N_4}, N(\tau_{N_4}) = N_4], \\ & \quad 1 \leq i \leq N_4, \quad 1 \leq k \leq N_4 \end{aligned}$$

That is, the only difference is that, in the case of $b_{ki}^L(\boldsymbol{\tau})$, L of the arrivals are deterministic and occur at time $\tau = 0^+$; only the remaining $N_4 - L$ are Poisson arrivals over

the interval $(0, \tau_{N_4}]$. Now we would like to use the $a^L_{ki}(\boldsymbol{\tau})$'s and $h^L_{ki}(\boldsymbol{\tau})$'s to calculate the $b^L_{ki}(\boldsymbol{\tau})$'s. First, note that:

$$b^L_{ki}(\boldsymbol{\tau}) = 1, \quad k \leq L \tag{28}$$

This is because any Z_k with $k \leq L$ is a deterministic arrival at time $\tau = 0^+$: hence the probability that it occurs before any $\tau_i > 0$ is 1. In order to calculate $b^L_{ki}(\boldsymbol{\tau})$ for $k > L$ we follow exactly the technique used to calculate $\beta^L_{ki}(\mathbf{t})$, which begins with Lemma 3.5. We replace t by τ, X by Z, N by N_4, α by a, η by h, and β by b. Otherwise, the following are the only changes made in the derivation:

- in Lemma 3.5, the exponent of the time ratio is $N_4 - L$;
- and in Lemma 3.6, the combination and first time ratio are replaced by:

$$\binom{N_4 - L}{k - L} \left(\frac{\tau_i}{\tau_{N_4}}\right)^{k-L}$$

Both of these items result because we are only choosing among the $N_4 - L$ uniform random variables, to determine the probability that exactly $N_4 - L$ of them (in the case of the first item above) or $k - L$ of them (in the case of the second item) will occur prior to time τ_i. After converting $\tilde{a}$'s to a's and $\tilde{h}$'s to h's, the following is our final form for $b^L_{ki}(\boldsymbol{\tau})$:

$$\begin{aligned}
b^L_{N_4,i}(\boldsymbol{\tau}) &= \frac{a^L_{N_4,i}(\boldsymbol{\tau})}{a^L_{N_4,N_4}(\boldsymbol{\tau})}, \quad N_4 - L < i \leq N_4 \\
b^L_{ki}(\boldsymbol{\tau}) &= 0, \quad 1 \leq i, \ k \leq N_4, \ k - i \geq L \\
b^L_{ki}(\boldsymbol{\tau}) &= b^L_{(k+1),i}(\boldsymbol{\tau}) + \frac{1}{a^L_{N_4,N_4}(\boldsymbol{\tau})} \binom{N_4 - L}{k - L} a^L_{ki}(\boldsymbol{\tau}) h^L_{ki}(\boldsymbol{\tau}), \\
&\quad L < k < N_4, \quad 0 < k - i < L \\
b^L_{ki}(\boldsymbol{\tau}) &= 1, \quad \text{otherwise}
\end{aligned}$$

Finally, we show how to use the $b^L_{ki}(\boldsymbol{\tau})$'s to fill in another portion of the $\beta^{\mathcal{M}}$-matrix, as given in the following Lemma:

Lemma 4.7

$$\beta^{\mathcal{M}}_{(D_4+k),(D_4+i)}(\mathbf{t}) = b^L_{ki}(\boldsymbol{\tau}), \quad L < k \leq N_4,\ 1 \leq i \leq N_4$$

Proof: First write out the definition:

$$\beta^{\mathcal{M}}_{(D_4+k),(D_4+i)}(\mathbf{t}) = \Pr[X_{D_4+k} \leq t_{D_4+i} | O(\mathbf{t}), N(t_N) = N, \mathcal{M}]$$

But the mat data tell us the following:

- $N(r_C) = D_4 + L$
- $N_Q(r_C^+) = L$
- $N_Q(t) \leq L,\ r_C < t \leq t_N$
- $X_1 < X_2 < \cdots < X_{D_4+L} < r_C$
- $r_C < X_{D_4+L+1} < \cdots < X_{D_4+N_4} = X_N$

Since we are only considering cases for which $k > L$, we may rewrite the definition above, incorporating the mat data, as follows:

$$\begin{aligned}\beta^{\mathcal{M}}_{(D_4+k),(D_4+i)}(\mathbf{t}) &= \Pr[X_{D_4+k} \leq t_{D_4+i} | N_Q(r_C^+) = L, Q^L(r_C, t_N), \\ &\quad r_C < X_{D_4+L+1} \leq t_{D_4+L+1}, \ldots, r_C < X_N \leq t_N, \\ &\quad N(r_C, t_N) = N_4 - L]\end{aligned}$$

Now if we time-shift everything by r_C and substitute τ's and Z's for t's and X's, we get:

$$\begin{aligned}\beta^{\mathcal{M}}_{(D_4+k),(D_4+i)}(\mathbf{t}) &= \Pr[Z_k \leq \tau_i | Q^L(0, \tau_{N_4}), Z_1 \leq \tau_1, \ldots, Z_{N_4} \leq \tau_{N_4}, N(\tau_{N_4}) = N_4] \\ &= b^L_{ki}(\boldsymbol{\tau}), \quad L < k \leq N_4, \quad 1 \leq i \leq N_4\end{aligned}$$

Note that in the expression above, we have $N(\tau_{N_4}) = N_4$, since we are counting both the Poisson arrivals and the deterministic arrivals which occurred at $\tau = 0^+$. ∎

Because we calculate $b^L_{ki}(\boldsymbol{\tau})$ for values of i between 1 and N_4, and hence we must also calculate $a^L_{ki}(\boldsymbol{\tau})$ and $h^L_{ki}(\boldsymbol{\tau})$ for these values of i, the overall computational complexity of the type 4 congestion period partition is $O(N_4 L^2)$.

4.4 Type 3 Congestion Period Partition Analysis: A Single Non-Mat Cycle

During the type 3 congestion period partition, which runs over the time interval $(r_j, d_{j+1}], 1 \leq j \leq C-1$, there are many constraints. First, we know that at time r_j^+, there are exactly $M-1=L$ customers in queue. Second, we know that between r_j and d_{j+1}, there are no more than L customers in queue. Also, we know that no server is made idle over the same time period (although we may have 0 in queue). Finally, we know that at time d_{j+1}^-, there are exactly L customers in queue again. Because the maximum queue length constraint applies, we again use a modification of the QIE^L algorithm. This time the modification will comprise both an artificial bulk arrival of L customers at time r_j and an artificial bulk departure of L customers at time d_{j+1}.

Specifically, assume there have been exactly D_3 departures up to *and including* time r_j, i.e., $r_j = t_{D_3}$. Also assume that there are a total of N_3 additional departures during the time interval $(r_j, d_{j+1}]$, i.e. $t_{D_3+N_3} < d_{j+1}$, but $t_{D_3+N_3+1} > d_{j+1}$. Now define $\boldsymbol{\tau}$ to have components as follows:

$$\tau_i = \begin{cases} t_{D_3+i} - r_j, & i = 1, 2, \ldots, N_3 \\ d_{j+1} - r_j, & i = N_3+1, \ldots, N_3+L \end{cases}$$

Finally, define $\mathbf{Z}$ to have the following components:

$$Z_k = \begin{cases} Y_k, & k = 1, 2, \ldots, L \\ X'_{k-L} = X_{D_3+k} - r_j, & k = L+1, \ldots, L+N_3 \end{cases}$$

Here, $Y_1, \ldots, Y_L$ are defined exactly as in Equation 27: i.e., they are deterministic random variables, occurring at time $\tau = 0^+$ with probability 1.

With these definitions, then, we have the following:

Lemma 4.8 $\beta^{\mathcal{M}}_{(D_3+k),(D_3+i)}(\mathbf{t}) = b^L_{ki}(\boldsymbol{\tau}), \quad L < k \leq L+N_3,\ 1 \leq i \leq N_3$

Proof: First, we must rewrite $\beta^{\mathcal{M}}_{(D_3+k),(D_3+i)}(\mathbf{t})$, incorporating the mat data, which include the following information:

- $N(r_j) = D_3 + L$
- $N_Q(t) \leq L, \quad r_j < t < d_{j+1}$
- $X_1 < X_2 < \cdots < X_{D_3+L} < r_j$
- $r_j < X_{D_3+L+1} < \cdots < X_{D_3+L+N_3} < d_{j+1}$
- $X_{D_3+L+N_3+1} = d_{j+1}$
- $d_{j+1} < X_{D_3+L+N_3+2} < \cdots < X_N$

Now we expand $\beta^{\mathcal{M}}_{(D_3+k),(D_3+i)}(\mathbf{t})$, using the above, and then subtract r_j from all times:

$$\begin{aligned}
\beta^{\mathcal{M}}_{(D_3+k),(D_3+i)}(\mathbf{t}) &= \Pr[X_{D_3+k} \leq t_{D_3+i} | N_Q(r_j^+) = L, \\
&\quad N_Q(t) \leq L, r_j < t < d_{j+1},\ r_j < X_{D_3+L+k} < d_{j+1}, 1 \leq k \leq N_3, \\
&\quad O(\mathbf{t}), N(r_j, d_{j+1}) = N_3] \\
&= \Pr[Z_k \leq \tau_i | Q^L(0, \tau_{N_3}), Z_1 \leq \tau_1, \ldots, \\
&\quad Z_{N_3+L} \leq \tau_{N_3+L}, N(\tau_{N_3+L}) = N_3 + L] \\
&= b^L_{ki}(\boldsymbol{\tau}) \qquad \blacksquare
\end{aligned}$$

Because of the bulk departure artifice used here and the resultant set of identical service times at the end of the congestion period partition, we may make simplifications, parallel to those made in Section 4.1, in our calculations of the a^L-matrix and the h^L-matrix. These simplifications follow exactly those given in Lemmata 4.2 and 4.3 and so are not repeated here. The simplifications result in an algorithm for the type 3 congestion period partition which is $O(N_3 L^2)$.

4.5 Completion of the $\beta^{\mathcal{M}}$-Matrix

We have shown how to fill in many parts of the $\beta^{\mathcal{M}}$-matrix by analysis of congestion period partitions. Filling in the rest of the matrix is easily accomplished in the manner described in this section. First, let N_1 and N_4 be the number of departures in

congestion period partition types 1 and 4, as described in sections 4.1 and 4.3. Also, let N_{2i} $(i = 1, 2, \ldots, C)$ and N_{3j} $(j = 1, 2, \ldots, C-1)$ be the number of departures during the i-th type 2 congestion period partition and the j-th type 3 congestion period partition, as described in sections 4.2 and 4.4 respectively. Similarly, define D_{2i}, D_{3j}, and D_4 to be the total number of departures prior to the indicated congestion period partition, as described in sections 4.2 through 4.4.

Although a standard β-matrix has N rows and N columns, we define the *expanded* $\beta^{\mathcal{M}}$-matrix to include $2C$ additional columns, one each for d_i^- and for d_i^+ $(i = 1, 2, \ldots, C)$. This is done in order to convey all of the information we have about the arrival times $X_1, X_2, \ldots, X_N$. We know explicitly that, for example, $\Pr[X_{N_1+M} \leq d_i^- | \ldots] = 0$, but $\Pr[X_{N_1+M} \leq d_i^+ | \ldots] = 1$. Specifically, we add two columns, d_i^- and d_i^+, between the columns representing $t_{D_{2i}}$ and $t_{D_{2i}+1}$ $(i = 1, 2, \ldots, C)$, thereby maintaining the time order of the columns of the matrix (since we know $t_{D_{2i}} < d_i < t_{D_{2i}+1}$). Now we may proceed to fill in the entire expanded $\beta^{\mathcal{M}}$-matrix.

Lemma 4.1 allows us to fill in the upper $(N_1 + M - 1) \times N_1$ corner of the $\beta^{\mathcal{M}}$-matrix. The next column, representing time d_1^-, has $(N_1 + M - 1)$ 1's and the rest 0's; while the column representing time d_1^+ has $(N_1 + M)$ 1's and the rest 0's (for the reasons argued in the previous paragraph). The following lemma should be obvious:

Lemma 4.9 *A 1 in the expanded $\beta^{\mathcal{M}}$-matrix in position $[k, l]$ generates 1's in positions $[m, n]$ where $m \leq k$ and $n \geq l$. Similarly, a 0 in position $[k, l]$ generates 0's in positions $[m, n]$ where $m \geq k$ and $n \leq l$.*

Proof: If $\Pr[X_k \leq \tau_l | \ldots] = 1$, (where τ_l represents the time corresponding to the l-th column of the matrix, i.e. either a t_i, d_j^-, or d_j^+) then, when $m \leq k$, we have $X_m \leq X_k$, and hence $\Pr[X_m \leq \tau_l | \ldots] = 1$. Then, $\Pr[X_m \leq \tau_n | \ldots] = 1$, too, for $n \geq l$ (i.e. when τ_n is a later time than τ_l). The proof for the 0's is identical to the above. ∎

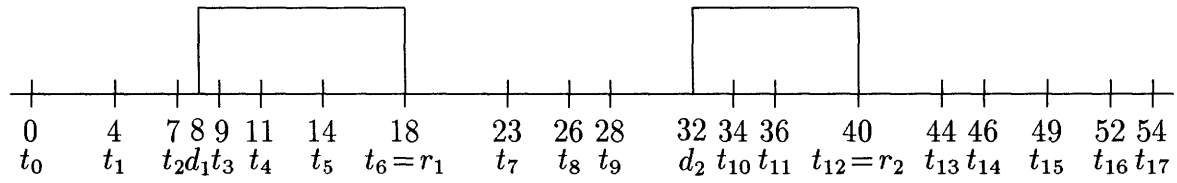
Hence, we fill in with 0's everything in the first N_1 columns which has not already been completed. Then, we fill in with 1's everything in the first $N_1 + M$ rows which

has not already been completed. The upper left-hand corner of what remains is then filled in via Lemma 4.4, specifically the entries corresponding to $\beta^{\mathcal{M}}_{(D_{21}+M+1),(D_{21}+1)}(\mathbf{t})$ through $\beta^{\mathcal{M}}_{(D_{21}+M+N_{21}),(D_{21}+N_{21})}(\mathbf{t})$ in the non-expanded $\beta^{\mathcal{M}}$-matrix. The next column represents $r_1 = t_{D_{21}+N_{21}+1}$. It has $(D_{21} + N_{21} + M)$ 1's and the rest 0's: hence, again we may fill in everything below the sub-matrix that was just filled in with 0's, and everything to the right of that sub-matrix with 1's. Then, we again fill in the upper left-hand corner of what remains, this time via Lemma 4.8, specifically the entries corresponding to $\beta^{\mathcal{M}}_{(D_{31}+M),(D_{31}+1)}(\mathbf{t})$ through $\beta^{\mathcal{M}}_{(D_{31}+M-1+N_{31}),(D_{31}+N_{31})}(\mathbf{t})$ in the non-expanded $\beta^{\mathcal{M}}$-matrix. The next two columns in the expanded matrix represent d_2^- and d_2^+, and are filled in with $(D_{31} + M - 1)$ 1's and the rest 0's, and $(D_{31} + M)$ 1's and the rest 0's, respectively, which again allows us to fill in under and to the right of the sub-matrix just completed. This process continues until the last mat cycle, at which point the remainder of the expanded $\beta^{\mathcal{M}}$-matrix is filled in via Lemma 4.7. An example of this filling-in procedure is provided in Figure 2.

Once the expanded matrix is completely filled in, its values may be used to generate queue statistics. This is done in a method similar (but not identical) to that used in [Lars 90]. As before, $\overline{N}(t_i)$ is calculated by adding up all of the values in the column of the $\beta^{\mathcal{M}}$-matrix corresponding to t_i. We also know that $\overline{N}(d_j^-) = D_{2j} + M - 1$ and that $\overline{N}(d_j^+) = D_{2j} + M$ (this may also be found by adding up the values in the columns corresponding to d_j^- and d_j^+). To find $\overline{N}_Q$, note the following:

$$\begin{aligned}
\overline{N}_Q(t_i^-) &= \overline{N}(t_i) - i + 1 \\
\overline{N}_Q(t_i^+) &= \overline{N}(t_i) - i \\
\overline{N}_Q(d_j^-) &= M - 1 \\
\overline{N}_Q(d_j^+) &= M
\end{aligned}$$

Then let $\boldsymbol{\tau}$ represent both the t_i's and the d_j's, ordered according to time (i.e., $\tau_1 = t_1, \ldots, \tau_{D_{21}} = t_{D_{21}}, \tau_{D_{21}+1} = d_1, \tau_{D_{21}+2} = t_{D_{21}+1}, \ldots$). As in [Lars 90], $\overline{N}_Q(t)$ is linear between the values calculated at the t_i's and d_j's. We may then calculate the time-



	t_1	t_2	d_1^-	d_1^+	t_3	t_4	t_5	t_6 (r_1)	t_7	t_8	t_9	d_2^-	d_2^+	t_{10}	t_{11}	t_{12} (r_2)	t_{13}	t_{14}	t_{15}	t_{16}	t_{17}
X_1	1	1	1	1																	
X_2	0.75	1	1	1																	
X_3	0.32	0.97	1	1									1								
X_4	0	0.74	1	1																	
X_5	0	0	1	1																	
X_6			0	1																	
X_7			0	0	1	1	1	1													
X_8			0	0	0.25	1	1	1							1						
X_9			0	0	0.02	0.30	1	1													
X_{10}			0	0				0	0	0.58	0.87	1	1								
X_{11}			0	0				0	0	0	0.39	1	1								
X_{12}	0		0	0				0	0	0	0	1	1				1				
X_{13}			0	0		0		0				0	1								
X_{14}			0	0				0				0	0	1	1	1		1			
X_{15}			0	0				0		0		0	0	0.33	1	1					
X_{16}			0	0				0				0	0		0	0	0	0.35	0.77	1	1
X_{17}			0	0				0				0	0			0	0	0	0.23	0.65	1

Figure 2: Sample Congestion Period with $M = 4$ and Two Mat Cycles; and Its Expanded $\beta^{\mathcal{M}}$-Matrix

average queue length as the area under $\overline{N}_Q(t)$, divided by the total congestion period time:

$$\overline{N}_Q = \frac{1}{t_N} \sum_{i=1}^{N} (\tau_i - \tau_{i-1}) \left(\frac{\overline{N}_Q(\tau_i^-) + \overline{N}_Q(\tau_{i-1}^+)}{2} \right)$$

where τ_0 is defined to be 0. The average wait in queue is then found to be:

$$\overline{W}_Q = \frac{t_N}{N} \overline{N}_Q$$

Incidence probabilities and $\bar{\ell}_Q$, the average queue length experienced by a randomly arriving customer, are calculated exactly as in [Larson].

5 Computational Results

We include here results from simulation of an M/M/1 queue. These data were generated by a simulation run with Poisson arrivals at rate 10 per hour, a single server, and exponential service time with expected value of 3 minutes (giving a value of $\rho = 0.5$). The first subsection below contains results which pertain to Section 3. We compare the QIE^L algorithm to the standard QIE algorithm, both for the case that the maximum queue length data are available, and for the case that the data are not available and the QIE^L algorithm is used as an approximation to the exact QIE algorithm. The second subsection below contains results which pertain to Section 4. We compare the QIE$^{\mathcal{M}}$ algorithm to the standard QIE algorithm and also consider the effect of moving the mat to different positions. The statistics that are used for comparison of the various algorithms include: $\overline{N}_Q$, the time-averaged number of customers in queue; $\overline{W}_Q$, the average wait in queue; and ϵ, the time-average error, defined to be the absolute area between the actual queue length graph and the QIE expected queue length graph, divided by the total time of the congestion period. The run times to generate the beta-matrix for the different algorithms are also compared. (Runs were on a 386/387-based Northgate Computer Systems PC: 3000 run times were averaged to obtain the run times below.)

5.1 Comparison of the QIE and the QIE^L Algorithms

In this section we compare the QIE^L algorithm to the standard QIE algorithm. Consider a congestion period with $N = 11$, as shown in Figure 3, and suppose we are told that the queue length never exceeded some value L. The figure depicts the exact queue length for the period (which, in fact, never exceeds 3) and, superimposed, depicts the QIE (or QIE^L) expected queue length. As can be seen, the standard QIE overestimates the expected queue length, while the QIE^L estimate with $L = 3$ is quite close to the actual data. With $L = 5$, the QIE^L is actually quite close to the standard QIE output: even though the standard QIE algorithm considers many more possible events (all those with queue length greater than 5 and less than 12), those events are of relatively low probability and so do not have much impact on the final expected queue length. Comparative statistics for these graphs are provided in Table 1.

Algorithm Used	ϵ	$\overline{N}_Q$ (actual = 1.1161)	$\overline{W}_Q$ (minutes) (actual = 3.7160)	Run Time (seconds)
QIE	0.9249	1.9230	6.4025	0.080
QIE^L, $L = 3$	0.5309	1.3654	4.5460	0.027
QIE^L, $L = 4$	0.6947	1.6467	5.4826	0.034
QIE^L, $L = 5$	0.8293	1.8107	6.0286	0.043

Table 1: Comparison of QIE and QIE^L Algorithms (with Max Queue Length Data Given) for a Congestion Period with $N = 11$

When we are not given data regarding the maximum queue length but have a long congestion period to analyze, we may wish to approximate the QIE output by using the QIE^L algorithm. An example of this is presented in Figure 4. These graphs illustrate the mean queue length as estimated by the QIE algorithm, compared with the same quantity as estimated by the QIE^L algorithm, for two congestion periods of 18 (left three graphs) and 21 (right three graphs) customers, respectively. We present these comparisons for values of L of 5, 8, and 10. Note that as L increases,

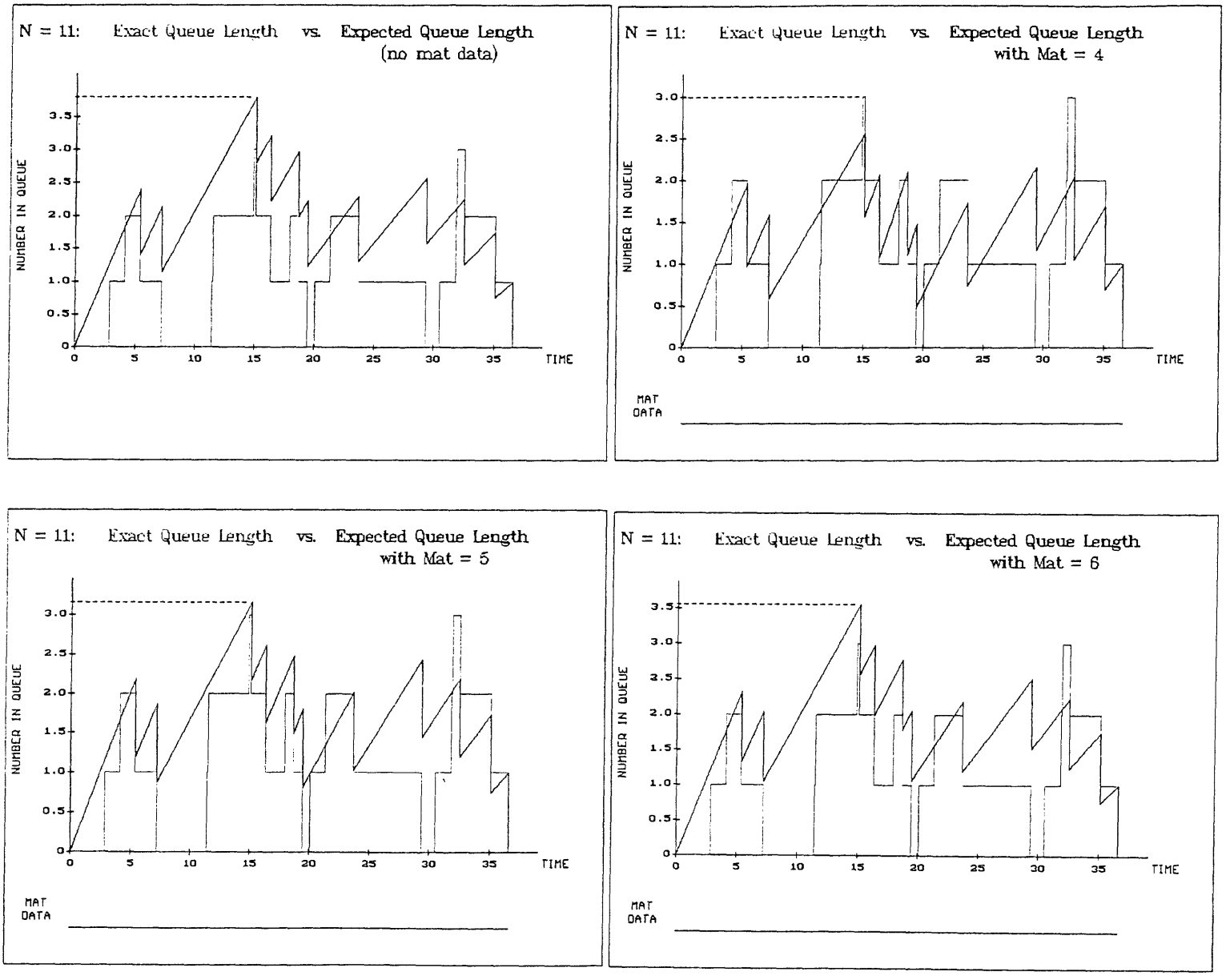
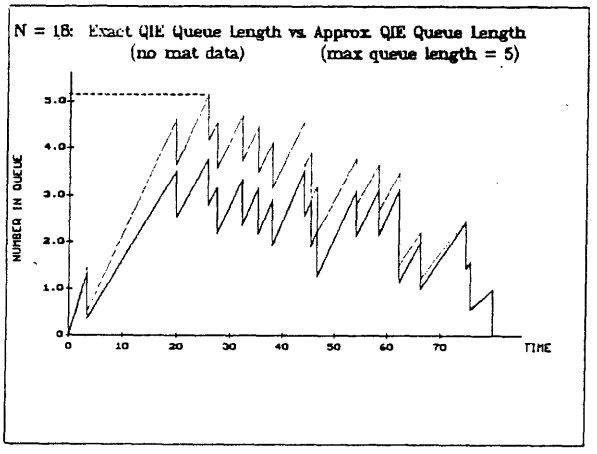
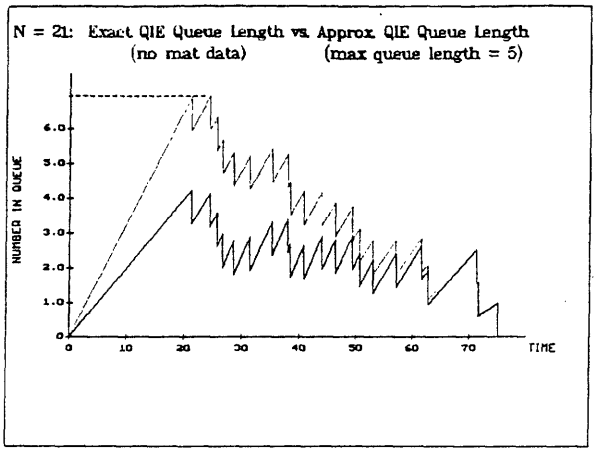
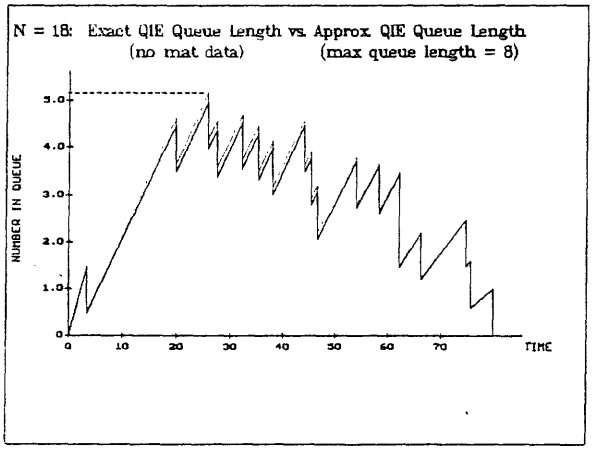
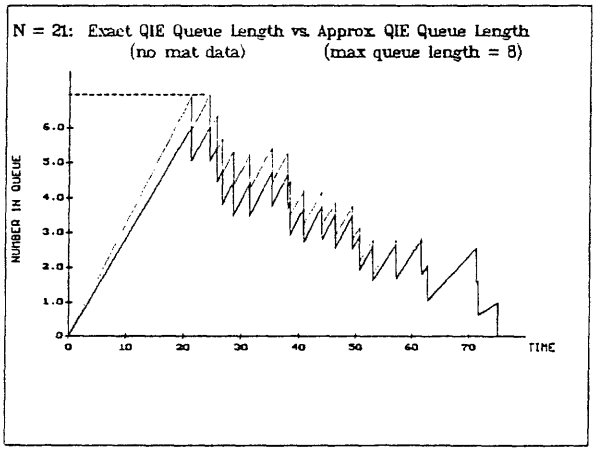


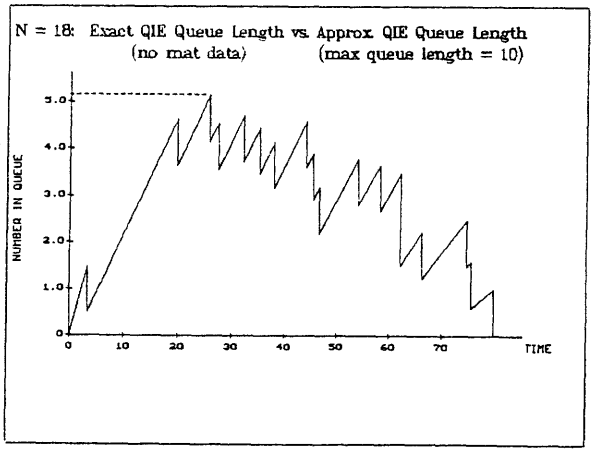
Figure 3: Exact Queue Length vs. Mean QIE (QIE^L) Queue Length for a Congestion Period with $N = 11$: Standard QIE and QIE^L with $L = 3, 4, 5$











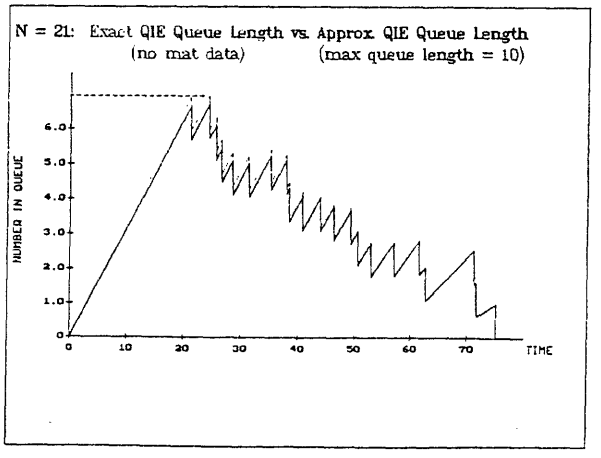


Figure 4: Mean Queue Length Estimate for Congestion Period of 18 Customers (Left) and 21 Customers (Right): Comparison between Exact QIE Calculation and QIE^L Calculation, with $L = 5, 8$, and 10

the values of $\overline{N}_Q(t)$ also increase, up to the maximum given by the standard QIE algorithm. Also note that for $L = 10$, the QIE^L algorithm is very close to the exact QIE algorithm yet requires considerably less running time. Comparisons of the values of the time-average queue length, the expected wait in queue, and the running time for these data are provided in Tables 2 and 3.

Algorithm Used	$\overline{N}_Q$	$\overline{W}_Q$ (minutes)	Run Time (seconds)
QIE	2.8649	12.6644	0.404
QIE^L, $L = 5$	2.1541	9.5221	0.084
QIE^L, $L = 8$	2.7703	12.2461	0.167
QIE^L, $L = 10$	2.8568	12.6282	0.235

Table 2: Comparison of QIE and QIE^L Algorithms (Used As an Approximation) for a Congestion Period with $N = 18$

Algorithm Used	$\overline{N}_Q$	$\overline{W}_Q$ (minutes)	Run Time (seconds)
QIE	3.3871	12.0615	0.694
QIE^L, $L = 5$	2.1681	7.7205	0.102
QIE^L, $L = 8$	2.9874	10.6383	0.208
QIE^L, $L = 10$	3.2773	11.6706	0.301

Table 3: Comparison of QIE and QIE^L Algorithms (Used As an Approximation) for a Congestion Period with $N = 21$

5.2 Comparison of the QIE and the QIE^M Algorithms

In Figures 5 and 6 we compare the QIE and the QIE^M algorithms. In those figures,

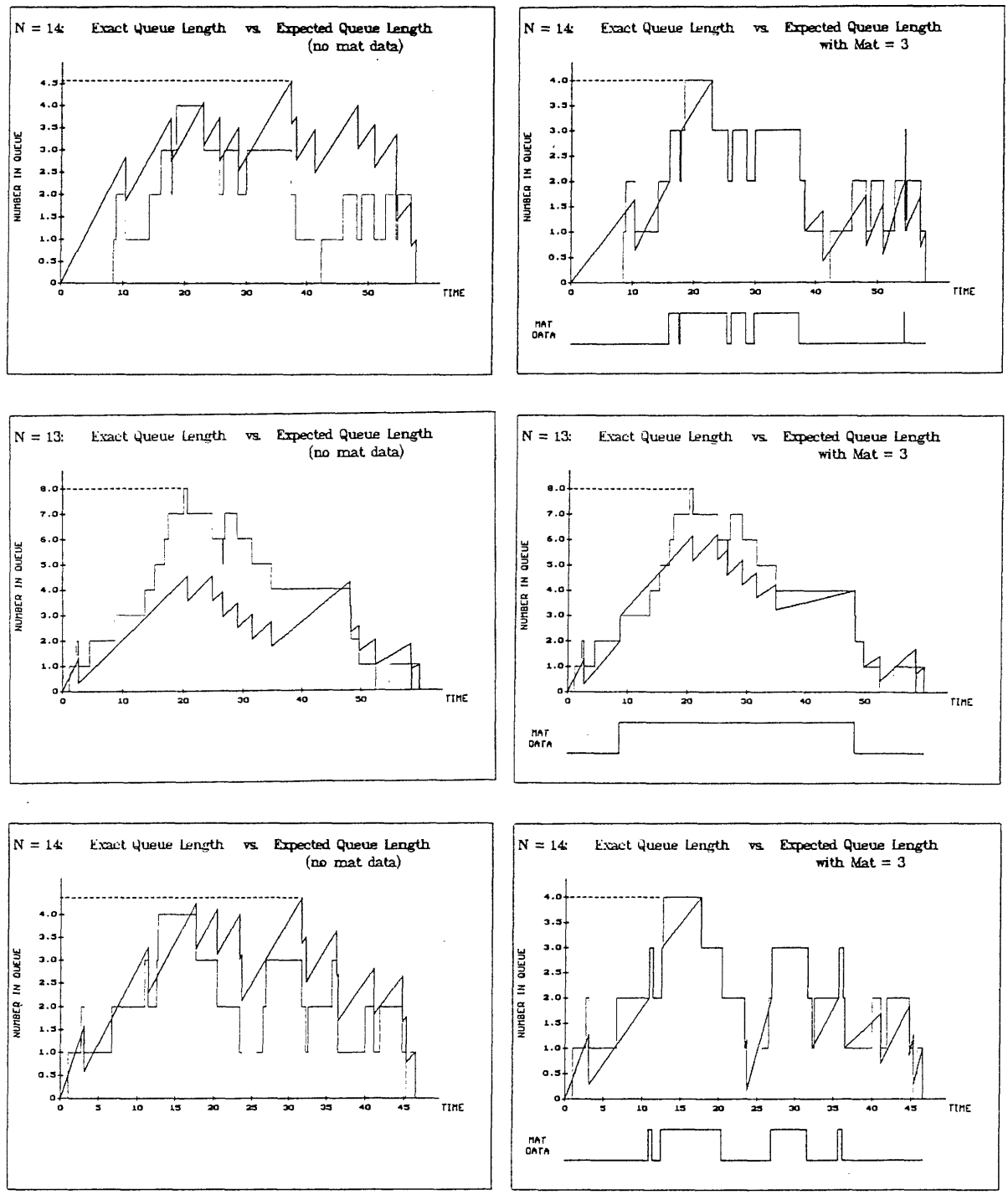


Figure 5: Mean Queue Length Estimate for Congestion Periods of 14, 13, and 14: Comparison between Standard QIE Calculation (Left) and QIE$^{\mathcal{M}}$ Calculation, with $M = 3$ (Right)

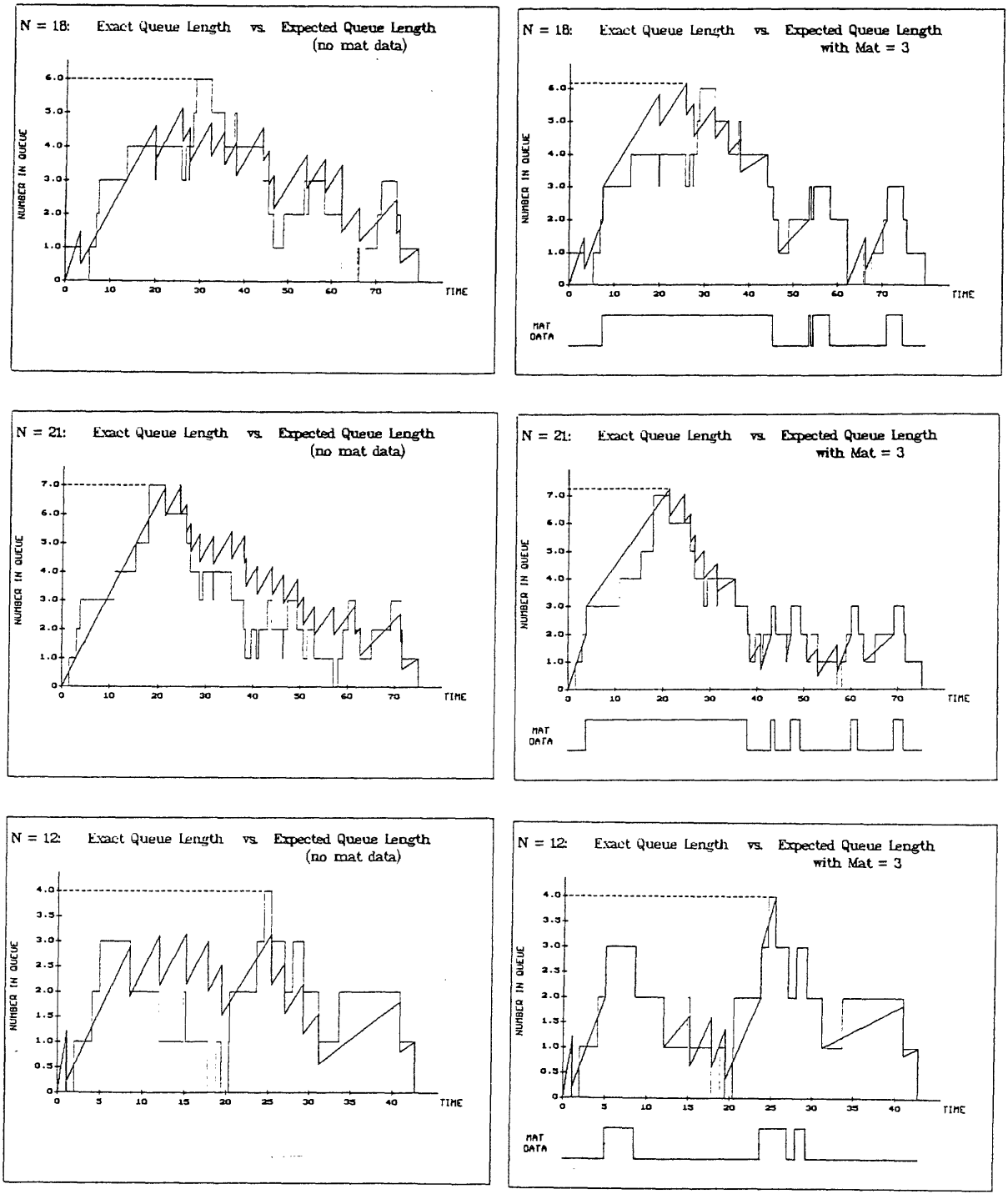


Figure 6: Mean Queue Length Estimate for Congestion Periods of 18, 21, and 12: Comparison between Standard QIE Calculation (Left) and QIE^M Calculation, with $M = 3$ (Right)

we present the six congestion periods with $N \geq 12$ in our simulation run, and compare the standard QIE performance (left set of graphs) with that of the QIE$^{\mathcal{M}}$ algorithm, with a mat added at position $M = 3$ (right set of graphs). The value 3 was arbitrarily chosen, although it was a value at which the QIE$^{\mathcal{M}}$ algorithm showed a marked improvement over the QIE algorithm, for all of the congestion periods considered, the extent of which is detailed in Table 4. Note the considerable improvement both in accuracy (particularly as evidenced by ϵ) and in running time.

Finally, in Figure 7 we present a comparison of the performance of the QIE$^{\mathcal{M}}$ algorithm, as the value of M varies, for the congestion period of length $N = 18$. (To see how the standard QIE algorithm performs on this congestion period, see Figure 4 and Table 2.) As demonstrated by this figure, the values of M which provide the most accurate prediction of queue length are not necessarily predictable for a single congestion period. However, it is hypothesized that an optimal mat placement does exist in an ergodic sense for a queue with specified parameters. The queue statistics corresponding to this figure are provided in Table 5.

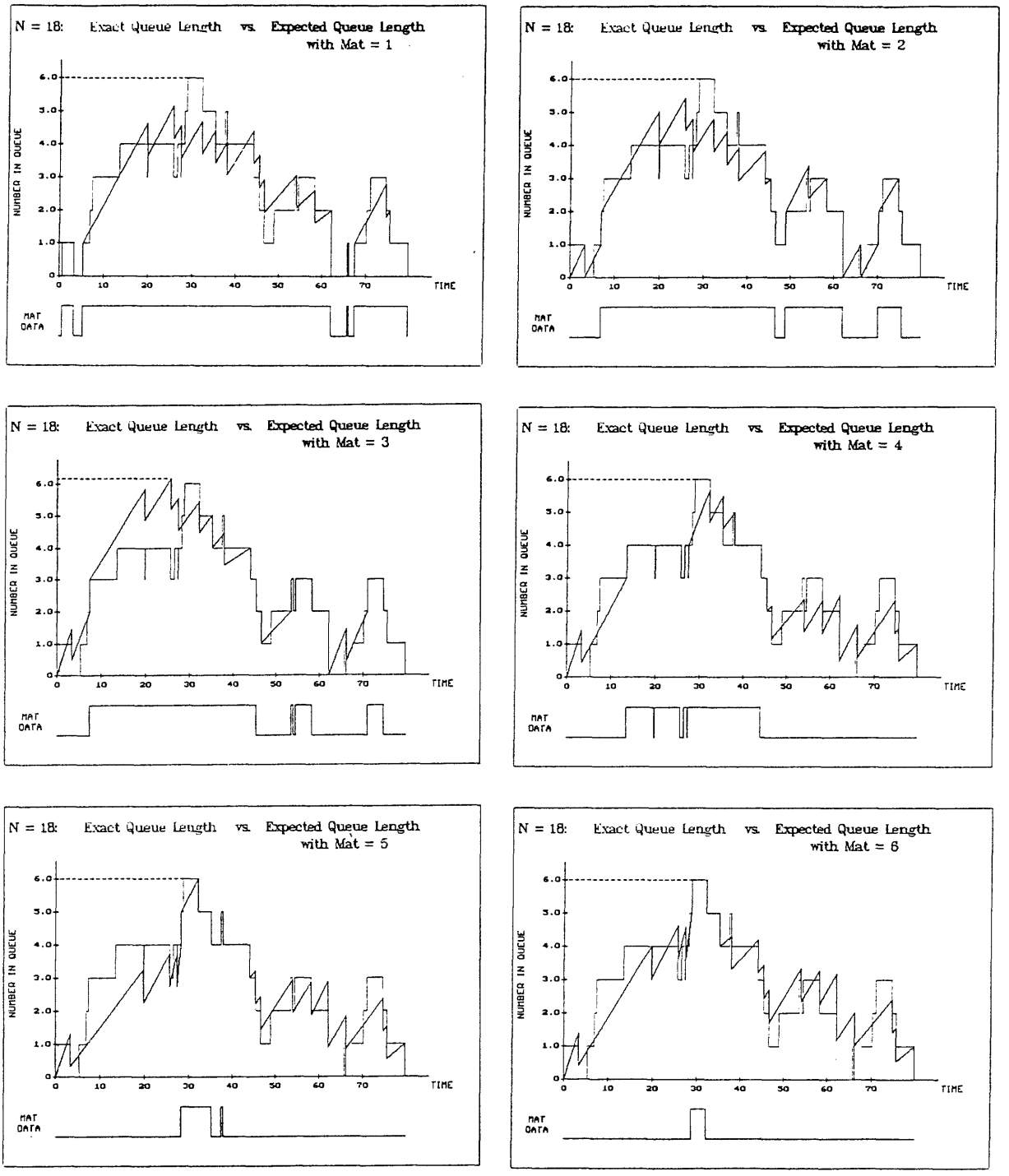


Figure 7: QIE$^{\mathcal{M}}$ Mean Queue Length Estimate for Congestion Period with $N = 18$ and Mat Placements at $M = 1, 2, 3, 4, 5$, and 6

Size of Cong. Period	Data Used	ϵ	$\overline{N}_Q$	$\overline{W}_Q$ (minutes)	Run Time (secs)
	Actual	—	1.8149	7.4835	—
$N = 14$	QIE	1.0962	2.8067	11.5727	0.172
	QIE$^{\mathcal{M}}$, $M = 3$	0.2909	1.8123	7.4727	0.027
	Actual	—	3.6868	16.8793	—
$N = 13$	QIE	1.4417	2.5231	11.5515	0.133
	QIE$^{\mathcal{M}}$, $M = 3$	0.6742	3.2784	15.0092	0.039
	Actual	—	1.9843	6.5792	—
$N = 14$	QIE	0.8015	2.6239	8.6997	0.173
	QIE$^{\mathcal{M}}$, $M = 3$	0.2935	1.8419	6.1070	0.027
	Actual	—	2.7095	11.9771	—
$N = 18$	QIE	0.7426	2.8649	12.6644	0.404
	QIE$^{\mathcal{M}}$, $M = 3$	0.5127	3.0174	13.3384	0.040
	Actual	—	2.8922	10.2991	—
$N = 21$	QIE	0.8013	3.3871	12.0615	0.694
	QIE$^{\mathcal{M}}$, $M = 3$	0.4056	3.1268	11.1346	0.026
	Actual	—	1.7390	6.1643	—
$N = 12$	QIE	0.7353	1.8193	6.4488	0.103
	QIE$^{\mathcal{M}}$, $M = 3$	0.2582	1.6969	6.0151	0.021

Table 4: Comparison of QIE and QIE$^{\mathcal{M}}$($M = 3$) Algorithms for Congestion Periods with $N = 14$, 13, 14, 18, 21, and 12

Mat Placement	ϵ	$\overline{N}_Q$ (actual = 2.7095)	$\overline{W}_Q$ (minutes) (actual = 11.9771)
$M = 1$	0.4868	2.5744	11.3802
$M = 2$	0.4536	2.6432	11.6843
$M = 3$	0.5127	3.0174	13.3384
$M = 4$	0.4034	2.5925	11.4599
$M = 5$	0.6000	2.4720	10.9272
$M = 6$	0.5528	2.7509	12.1601

Table 5: Queue Statistics for QIE$^{\mathcal{M}}$ Algorithm for a Congestion Period with $N = 18$ and $M = 1, 2, 3, 4, 5$, and 6

6 Practical Implications and Future Research

As discussed in Section 1, the QIE has many potential practical applications. Its performance could be dramatically improved by the addition of sensor mats, electric eyes, or some other determinant of $M-1 \rightarrow M$ and $M \rightarrow M-1$ queue transitions. One problem that immediately presents itself (particularly in the case of human queues) is how to ensure that person number M in queue is indeed the person who first depresses the mat; or, to turn the question around, how badly would the QIE$^{\mathcal{M}}$ estimates be thrown off if, say, the $(M-1)$-th person in queue were to stand on the mat, rather than the M-th (as assumed by the algorithm)? This is an issue of sensitivity analysis: if the algorithm is highly sensitive to which person is standing on the mat, then it may be preferable to run the standard QIE algorithm (i.e., no information may be better than bad information). If, on the other hand, the sensitivity is not that high, then it may still be possible to use the QIE$^{\mathcal{M}}$ algorithm, although its performance will be degraded.

The QIE$^{\mathcal{M}}$ algorithm may be particularly useful in a setting in which the Poisson rate of arrivals varies slowly within a single congestion period, or in which there is some semi-state-dependent balking. In the first case, since the QIE$^{\mathcal{M}}$ algorithm breaks the congestion period up into several congestion period partitions, then even if the Poisson rate varies significantly over the entire congestion period, it may be relatively stable over the course of any congestion period partition. Similarly, say that the system experiences balking such that the probability of a customer not joining the queue for queue lengths less than M is p_1, and the probability of a customer not joining the queue for queue lengths greater than or equal to M is p_2. In this case, if the Poisson rate of arrivals is approximately constant with rate λ, then during any type 1, 3, or 4 congestion period partition, arrivals are Poisson at rate λp_1, and during any type 2 congestion period partition, arrivals are Poisson at rate λp_2. If there were more than two balking rates, then, theoretically, one could add mats to correspond to the points at which the balking rate changed. This may be a simple way to capture

the effects of balking on expected queue length.

Another issue that must be addressed in a practical setting is that of queue-length propagation delays. In a human queueing situation, when a customer enters service, it takes some time for the entire queue to move forward to take up the slack. If M were relatively large, then in the analysis of the type 2 congestion period partition, it might make sense to use the high-frequency information provided by people stepping off and back onto the mat, as the queue moves forward, in place of the actual service-initiation times, the t_i's. This is because these service initiations might take a significant amount of time to propagate back to the mat, and analysis based on them may not provide accurate queue-length estimates.

Several areas of research still need to be addressed. First, as was seen in the previous section, the placement of the mat for a given queueing situation is critical. The placement may be optimized in terms of accuracy: i.e., how close do the queue estimates come to the actual values of those statistics? It may also be optimized in terms of computational complexity: how do we minimize the amount of computation to be done in the calculation of the $\beta^{\mathcal{M}}$-matrix? It is hypothesized that, in terms of accuracy, there is a single optimal location for the mat, for a given queue. Clearly, with $M = 0$ or $M = \infty$, we get zero additional information. It remains to be shown that, for a given set of queue parameters, the "best" mat location is at a single value of M.

Finally, the issue of multiple mats must be addressed. That is, say we have partial queue-length information that includes all $M_1 - 1 \rightarrow M_1$ transitions (and back) *and* all $M_2 - 1 \rightarrow M_2$ transitions (and back). This should be a fairly simple extension of the existing QIE$^{\mathcal{M}}$ algorithm, since we are simply breaking down the congestion period further into more congestion period partitions, now with two different sets of queue constraints. The number of mats to be used could, of course, exceed two; and the optimal number of mats to use would require a cost-benefit analysis.

References

[Barl 72] Barlow, R. E., D. J. Bartholomew, J. M. Brenner, and H. D. Brunk, *Statistical Inference under Order Restriction,* John Wiley and Sons, New York, NY, 1972.

[Bert 91] Bertsimas, D. J. and L. D. Servi, "Deducing Queueing from Transactional Data: the Queue Inference Engine Revisited," Technical Report OR 212-90, Operations Research Center, Massachusetts Instititute of Technology, Cambridge, MA, 1991.

[Dale 91] Daley, D. J. and L.D. Servi, "Exploiting Markov Chains to Infer Queue-Length from Transactional Data," submitted to *Journal of Applied Probability,* 1991.

[Davi 81] David, H. A., *Order Statistics,* John Wiley and Sons, New York, NY, 1981.

[Lars 90] Larson, R. C., "The Queue Inference Engine: Deducing Queue Statistics from Transactional Data," *Management Science,* Vol. 36, No. 5, 1990, pp. 586–601.

[Lars 91] Larson, R. C., "The Queue Inference Engine: Addendum," to appear in *Management Science,* 1991.