“Please come back later”: Benefiting from deferrals in service systems

Anmol Kagrecha
EE, IIT Bombay
akagrecha@ee.iitb.ac.in

Jayakrishnan Nair
EE, IIT Bombay
jayakrishnan.nair@ee.iitb.ac.in

Abstract—The performance evaluation of loss service systems, where customers who cannot be served upon arrival get dropped, has a long history going back to the classical Erlang B model. In this paper, we consider the performance benefits arising from the possibility of deferring customers who cannot be served upon arrival. Specifically, we consider an Erlang B type loss system where the system operator can, subject to certain constraints, ask a customer arriving when all servers are busy, to come back at a specified time in the future. If the system is still fully loaded when the deferred customer returns, she gets dropped for good. For such a system, we ask: How should the system operator determine the ‘rearrival’ times of the deferred customers based on the state of the system (which includes those customers already deferred and yet to arrive)? How does one quantify the performance benefit of such a deferral policy?

Our contributions are as follows. We propose a simple state-dependent policy for determining the rearrival times of deferred customers. For this policy, we characterize the long run fraction of customers dropped. We also analyse a relaxation where the deferral times are bounded in expectation. Via extensive numerical evaluations, we demonstrate the superiority of the proposed state-dependent policies over naive state-independent deferral policies.

Index Terms—Queueing theory, loss system, deferral, blocking probability

I. INTRODUCTION

Many service systems have the property that denial of service upon the arrival of a request results in the request getting dropped. The classical example comes from telephony, where an incoming call request must either be connected, or dropped. Another contemporary example is an electric car charging facility, which cannot serve an incoming customer when all charging stations are occupied. Similarly, certain online services decline fresh requests when congested.

Such service systems, by their very nature, are unable to queue requests that arrive when service capacity is fully utilized. It is therefore natural to ask: What if these systems were instead able to defer some requests, i.e., have these requests ‘come back’ at a pre-specified time in the future? If so, what is optimal strategy for deferring requests? To what extent does the throughput of the system improve from such a deferral policy? The goal of this paper is to shed light on these questions.

Formally, we consider an Erlang B (M/M/K/K) service system, where the system operator can, subject to certain constraints, defer jobs that arrive when all servers are occupied. The constraints include an upper bound on the deferral time (which is natural given QoS considerations), and/or a constraint on the number of jobs that can be deferred at any time. In this setting, our goal is to design simple deferral policies that are effective, while also being analytically tractable. The main difficulty in the performance evaluation of deferral policies is that the state description must incorporate information about (future) arrival instants of deferred jobs.

Our main contribution is the design and analysis of a deferral policy, which spaces arrivals of deferred jobs uniformly in the future. Even though this policy induces a complicated (uncountable) Markovian state space, we are able to analytically characterize the blocking probability via a novel combination of steady state and transient analysis of certain Markov processes. Via numerical experiments, we show that the proposed policy outperforms naive state-independent deferral policies, as well as the Erlang B system (which does not allow deferrals).

A relaxation on the deferral time constraint is considered next, where the upper bound on the deferral time is imposed in expectation. For this relaxed model, we propose a deferral policy, which admits a considerably more explicit performance characterization. Via numerical experiments, we show that the performance of this policy is in fact a good approximation to the policy designed for the ‘hard’ deferral constraint.

The remainder of the paper is as follows. We provide a brief survey of related literature next. In Section II, we describe the primary system model and state some preliminary results. The ‘hard’ case where deferral times are bounded deterministically is considered in Section III and the relaxed model is considered in Section IV.

Related literature: This work is motivated by the recent work [3], which implemented a deferral system for a web-server that is prone to congestion. To the best of our knowledge, there is no prior work that provides an analytical treatment of queueing systems that allow deferral.

This work is peripheral to the considerable literature on retrial queues (we refer the reader to a recent survey paper [6] and the book [11]). Retrial queues are systems where blocked customers ‘try again later’. The main contrast between retrial queues and deferral queues is that in the former setting, the retrial model is taken to be exogenous and indicative of customer behavior, while in the latter setting, deferral is a control decision on part of the system operator.

Another related class of loss systems that have been studied...
are abandonment based queues (references include [2] and [7]). In these systems, customers who are denied service upon arrival wait for a certain (potentially random) amount of time. If service is not offered before this patience time runs out, the customer leaves. In contrast, in deferral queues, a deferred customer does not actually wait, but rather leaves the system to come back at a pre-specified time. So this deferred customer can only be served if service capacity is available at the precise moment of re-entry. In this sense, abandonment based systems provide lower bounds on the performance achievable in deferral systems.

II. MODEL AND PRELIMINARIES

We consider a loss service system consisting of $K$ identical parallel servers of unit speed. Jobs (or customers) arrive according to a Poisson process of rate $\lambda$, and service times are exponentially distributed with rate $\mu$. In the absence of deferrals, this system is the classical Erlang B (M/M/K/K) queue.

On this baseline Erlang B model, we allow the system operator (or the scheduling policy) to defer jobs that arrive when all servers are busy, to come back to the system at a pre-specified time in the future. If a free server is available when this deferred job comes back, it gets admitted into service. If not, the job gets dropped. In other words, we only allow a job to be deferred once. We refer to the time instant at which a deferred job comes back to the system as its rearival time, and we refer to the interval between the rearival time and the original arrival time as the deferral time.

Of course, it is natural to impose certain constraints on the deferral policy. Firstly, QoS considerations dictate that deferral time cannot be too large. We impose this constraint in two ways: In Section III, we consider the case where there is a deterministic upper bound $\hat{T} > 0$ on deferral times. In Section IV, we consider a relaxed model where the deferral times are bounded above in expectation by $\hat{T}$. A second constraint we impose is an upper bound $\hat{D}$ on the number of deferred jobs at any time. Finally, we impose work conservation, i.e., when a job arrives (either for the first time, or post defer), it must be served so long as there is at least one free server available.

Our metric for evaluating the performance of a deferral policy is the long run fraction of jobs blocked, a.k.a., the blocking probability. Note that jobs can be blocked in two ways: One, a job arrives when all servers are busy, and there are already $\hat{D}$ previous arrivals that have been scheduled for rearival in the future. Second, a deferred job on rearival finds that all the servers are (still) busy.

A naive deferral policy would be to assign a large deferral time (say $\hat{T}$), subject to limit $\hat{D}$ on number of deferred jobs. Another naive policy would be to assign a deferral time sampled uniformly at random in $[0, \hat{T}]$, again subject to the deferral limit. Such policies are state-independent, in that the deferral time of a job is assigned independently of the deferral times assigned to previous jobs. In this paper, we propose low overhead state-dependent deferral policies, that schedule rearivals of deferred jobs cognizant of the already scheduled rearivals.

Finally, we note that work conservation ensures that the blocking probability of any deferral policy is at most the blocking probability in the Erlang B system. In other words, the Erlang B formula provides an upper bound on the blocking probability of any deferral policy.

Throughout the paper, for $n \in \mathbb{N}$, the set $\{0, 1, 2, \ldots, n\}$ is denoted by $[n]$.

III. BOUNDED DEFERRAL TIMES

In this section, we consider the case where the deferral times are bounded by $\hat{T} > 0$, i.e., the system cannot defer an incoming job by more than $\hat{T}$ time units. For this model, we propose a simple state-dependent deferral policy, which we refer to as the Deterministically Spaced Rearival Times (DSRT) policy, which spreads the rearival instants of deferred jobs ‘uniformly’, so as to maximize the chances of admitting them. We provide a characterization of the blocking probability under this policy, via a novel combination of the steady state analysis of a Markov chain that captures the system evolution in the absence of deferrals, and the transient analysis of another Markov chain that captures the system evolution in the presence of deferrals. Finally, we present a case study that demonstrates the superiority of our state-dependent deferral policy over naive state-independent policies.

A. DSRT Policy

The proposed deferral policy is described as follows. It is parameterized by $x \in (0, \hat{T}]$, which specifies the spacing between arrivals of deferred jobs into the system. If there are no jobs currently deferred, and an arriving job finds all servers busy, then the policy defers that job by $x$ time units. On the other hand, if there are deferred jobs, and an arriving job finds all servers busy, then the policy schedules its rearival time to be $x$ time units after the rearival time of the last deferred job, subject to a maximum of $D = \min(\hat{D}, \lfloor \hat{T}/x \rfloor)$ deferred jobs at any time. This policy is defined formally as Algorithm 1.

Algorithm 1 DSRT policy

procedure DSRT
    if num_deferrals = 0 then
        rearival_time ← arrival_time + $x$
    else if num_deferrals < $D := \min(\hat{D}, \lfloor \hat{T}/x \rfloor)$ then
        rearival_time ← prev_rearival_time + $x$
    else
        Customer blocked
    end if
end procedure

Note that the proposed DSRT policy attempts to spread the rearival instants $x$ units of time apart, subject to the deferral constraints. Intuitively, if the spacing $x$ is too small, then

1While we take $\hat{T}$ to be an exogenous model parameter, its value may in practice be determined based on QoS considerations and customer patience.
the deferred jobs are likely to find all servers occupied upon rearrival. On the other hand, if $x$ is too large, the policy is inefficient, since fewer jobs may be deferred at any time (given the constraint $T$ on the deferral times), and fewer opportunities are created to admit deferred jobs. Thus, the policy must operate at an intermediate ‘soft spot’; we address this issue in our numerical experiments.

Under the DSRT policy, the temporal evolution of the system can be captured as a Markov process over state space $[K] \times [D] \times [0, x]$. Here, the state $(k, d, \theta)$ indicates that $k$ servers are currently busy, there are $d$ jobs in deferral, and that the earliest rearrival of a deferred job will occur after time $\theta$. Note that the state space is uncountable, making an analysis of the stationary distribution cumbersome. We overcome this difficulty by directly characterizing $\Pi_{k,d}$ which is the long run fraction of time the state is of the form $(k, d, \theta)$ for some $\theta \in [0, x]$. This is done by separately analysing the temporal evolution of the system in the presence and absence of deferred jobs, and then combining these analyses in a novel manner.

We begin by considering the special case of a single server, and a limit of at most one deferral at any time ($K = D = 1$). This special case is instructive, not just because it illustrates our analysis approach, but also because it admits a closed form characterization of the blocking probability. We then consider the case of general $K$, $D$ in Section III-C; in this case, our characterization is less explicit, though amenable to an exact computation.

B. Single-Server, Single-Deferral System

The temporal evolution of the system for the case $K = D = 1$ is illustrated in Figure 1. The key observation that informs our analysis is the following. Considering only the times when the system has no deferrals (i.e., disregarding the deferral phases, when $d > 0$), the system configuration evolves as per the Markov chain depicted at the bottom of Figure 1.

On the other hand, over each deferral phase of length $x$ (when $d > 1$) the system configuration evolves as per the Markov chain depicted at the top of Figure 1.

Since the system configuration evolves as per the Markov chain depicted at the bottom of Figure 1 when there are no deferrals, the ratio between $\Pi_{0,0}$ and $\Pi_{1,0}$ is dictated by the flow balance equations for that Markov chain:

$$\lambda \Pi_{0,0} = \mu \Pi_{1,0}. \quad (1)$$

Turning now to the configurations where $d = 1$, we note that the relationship between $\Pi_{0,1}$ and $\Pi_{1,1}$ is dictated by the transient behavior of the Markov chain depicted on the top of Figure 1 over an interval of length $x$, starting in configuration $(1, 1)$. Focusing on this Markov chain over the interval $[0, x]$, the distribution of the chain at time $t$, captured by $p(t) = [p_{0,1}(t), p_{1,1}(t)]$, is given by:

$$p(t) = [0, 1]e^{Qt},$$

where

$$Q = \begin{bmatrix} -\lambda & \lambda \\ \mu & -\mu \end{bmatrix}.$$

It now follows that

$$\frac{\Pi_{0,1}}{\Pi_{1,1}} = \frac{\int_0^x p_{0,1}(t) dt}{\int_0^x p_{1,1}(t) dt} \quad (2)$$

Next, we need to relate the long run fraction of time spent in non-deferral configurations ($d = 0$) to those under deferral configurations ($d = 1$). To do this, we note, using the PASTA property, that the rate at which transitions occur from non-deferral configurations to deferral configurations equals $\lambda \Pi_{1,0}$. On the other hand, transitions from deferral configurations to non-deferral configurations occur at rate $\frac{\Pi_{0,1} + \Pi_{1,1}}{x}$. Equating the two, we get:

$$\lambda \Pi_{1,0} = \frac{\Pi_{0,1} + \Pi_{1,1}}{x}. \quad (3)$$

Lastly, we have the normalization condition

$$\Pi_{0,0} + \Pi_{1,0} + \Pi_{0,1} + \Pi_{1,1} = 1. \quad (4)$$

Solving the system of equations (1), (2), (3) gives yields a closed form characterization of the long run fraction of time spent in each configuration.

Finally, we are interested in characterizing the blocking probability $P_B$, which is the long run fraction of customers dropped. This can be done as follows:

$$P_B = \Pi_{1,1} + \frac{(\Pi_{0,1} + \Pi_{1,1})p_{1,1}(x)}{\lambda x} \quad (5)$$

The first term above captures the long run fraction of customers dropped on arrival (because the server was busy and there was already a deferred customer). The second term captures the long run fraction of customers dropped post deferral on rearrival. Indeed, the probability that any deferred customer gets dropped on rearrival equals $p_{1,1}(x)$. Thus, the rate of drops of deferred customers equals $\frac{\Pi_{0,1} + \Pi_{1,1}}{x} p_{1,1}(x)$.
To summarize, solving equations (1)–(4), and applying the solution to (5) yields the following result.

**Theorem 1:** For the case \( K = D = 1 \), under the DSRT policy, the blocking probability \( P_B \) is given by

\[
P_B = \frac{\rho}{1 + \rho} - \frac{\rho(1 - \exp\{-\rho(\lambda + \mu)x\})}{(1 + \rho)^2(1 + \rho(1 + \lambda x))},
\]

where \( \rho := \lambda/\mu \).

Note that \( \frac{\rho}{1 + \rho} \) is the blocking probability of an M/M/1/1 system. It is therefore clear from (6) that the blocking probability under the DSRT policy is a strict improvement over the scenario where deferral is not allowed. Moreover, the improvement diminishes to zero when \( x = 0 \) and as \( x \to \infty \). While the optimal value of \( x \) that minimizes \( P_B \) does not admit an explicit formula, it can be shown that the optimal value is bounded between \( \frac{1}{\lambda + \mu} \) and \( \sqrt{2} \).

**C. Multiple-Servers, Multiple-Deferrals System**

We now turn to the performance evaluation in the general setting, i.e., the number of servers \( K \) and the maximum number of deferrals \( D \) are arbitrary. Figure 2 illustrates the temporal evolution of \( \theta \) in this case. Note that each deferral phase (an interval over which \( \alpha > 0 \)) is no longer necessarily of duration \( x \) anymore, but is instead made up of a random number of deferral stages, each of duration \( x \). Moreover, at the end of each deferral phase, the configuration is of the form \((k, 0)\) where \( 0 < k \leq K \). Thus, disregarding the deferral phases, the system configuration evolves as per the Markov chain depicted in Figure 3. Here, \( \alpha_i \) denotes the probability that a deferral phase ends with \( i \) active servers. (The values of \( \alpha_i \) are in turn dictated by the behavior of the system during each deferral phase; we will characterize \( \alpha_i \) shortly.) Moreover, over each deferral stage of duration \( x \), the system configuration evolves as per the Markov chain shown in Figure 4.

In order to characterize \( \alpha_i \), and to arrive at the system of equations relating the long run fractions of time spent in the different configurations, we first analyse the evolution of the system configuration over a single deferral stage, and then consider the sequence of starting configurations across deferral stages within a deferral phase.

1) **Evolution of system configuration within a single deferral stage:** Deferral stages begin in a configuration of the form \((k, d)\), where \( 1 \leq k \leq K \), and \( 1 \leq d < D \). Subsequently for duration \( x \), the system configuration evolves as per the

3Specifically, the first deferral stage in any deferral phase begins in configuration \((K, 1)\).
Markov chain depicted in Figure 3. The transition rate matrix corresponding to this chain is given by

\[
Q = \begin{bmatrix}
Q_1 & \Lambda & 0 & 0 & \cdots & 0 \\
0 & Q_1 & \Lambda & 0 & \cdots & 0 \\
0 & 0 & \ddots & \ddots & \ddots & \vdots \\
0 & 0 & \cdots & 0 & Q_1 & \Lambda \\
0 & 0 & \cdots & 0 & 0 & Q_2 \\
\end{bmatrix},
\]

where \( \mathbf{0} \) is an all-zero square matrix of size \((K+1)\), \(Q_2\) is the transition rate matrix of an M/M/K/K queue, \(\Lambda\) is a diagonal matrix with entries \((0, \ldots, 0, \lambda)\), and \(Q_1 = Q_2 - \Lambda\). In writing this rate matrix, the configurations \((k, d)\) have been ordered by stacking them row-wise from Figure 3.

Considering the evolution of this chain over the interval \([0, x]\), the distribution at time \(t\) is denoted by the vector \(\mathbf{p}(t)\), where

\[
\mathbf{p}(t) = \mathbf{p}_{\text{stat}} e^{Q t}.
\]

Here, the initial probability vector \(\mathbf{p}_{\text{init}}\) would have an entry 1 according to the starting configuration of the stage under consideration, the other entries being zero.

2) Evolution of deferral stage starting configurations: As mentioned before, the starting configuration in a deferral stage is of the form \((k, d)\), where \(1 \leq k \leq K\), and \(1 \leq d < D\). We construct a discrete time Markov chain to capture the evolution of the starting configurations across successive deferral stages. To make the chain recurrent, we add a dummy state \(D\) which captures the end of the deferral phase. Note that the transition from the dummy state is to the configuration \((K, 1)\), since that is the starting configuration on the next deferral phase.

To describe the transition probability matrix corresponding to this discrete time Markov chain, we need the following notation. Within a deferral stage, let \(p_{k,d'}^c(t)\) denote the probability of being in configuration \((k', d')\) at time \(t \in [0, x]\), with starting configuration \((k, d)\). Note that these probabilities are defined in Section 3.2.3.

Now, let \(\hat{\mathbf{p}}(c'|c)\) denote the probability that the starting configuration in the next deferral stage is \(c'\) given that the starting configuration in the present deferral stage is \(c\), where \(c, c' \in \bar{S} := \{(k, d) \mid 1 \leq k \leq K, 1 \leq d < D\} \cup \{D\}\).

The transition probabilities \(\hat{\mathbf{p}}(\cdot|\cdot)\) are given by

\[
\hat{\mathbf{p}}((k', d')(k, d)) = p_{k',1,d'+1}^d(x)
\]

for \(1 \leq k' < K, 1 \leq d' < D\),

\[
\hat{\mathbf{p}}((K, d')(k, d)) = p_{K,1,d'+1}^d(x) + p_{K,d'+1}^d(x)
\]

for \(1 \leq d' < D\),

\[
\hat{\mathbf{p}}(D|k, d) = \sum_{l=0}^{K} p_{l,1}^d(x),
\]

\[
\hat{\mathbf{p}}((K, 1)|D) = 1.
\]

Given these transition probabilities corresponding to this finite irreducible discrete time Markov chain, one can obtain the stationary distribution. Let \(\hat{\mathbf{p}}_{\text{stat}}\) denote the stationary probability associated with configuration \(c \in \bar{S}\).

3) The equations relating \(\Pi_{k,d}\): We begin by relating the long run fractions of time spent in the non-deferral configurations. This is in turn based on the Markov chain depicted in Figure 3. The probabilities \(\alpha_l\) are characterized as follows.

Lemma 1: For \(1 \leq l < K\),

\[
\alpha_l = \frac{\sum_{j=1}^{K-l} p_{\text{start}}^{j,1} p_{l-1,j}^{j,1}(x)}{\sum_{j=1}^{K-l} p_{\text{start}}^{j,1} \sum_{l'=0}^{j} p_{l',1}^{j,1}(x)}.
\]

Proof: The characterization of \(\alpha_l\) follows directly from the discrete-time Markov chain discussed in Section 3.2.3. Indeed, \(\alpha_l\) is simply the probability that a deferral phase ends with a transition to the configuration \((l, 0)\). Consider a (discrete-time) renewal process where renewal instance correspond to visits to the dummy state (i.e., the end of a deferral phase) in the above DTMC. Each renewal cycle produces a reward of 1 if it ends with a transition to the configuration \((l, 0)\), and zero otherwise. An application of the renewal reward theorem yields

\[
\sum_{j=1}^{K-l} p_{\text{start}}^{j,1} p_{l-1,j}^{j,1}(x) = \alpha_l \Pi_{D}^{\text{start}}.
\]

The statement of the lemma now follows, once we note that the balance equations for the DTMC under consideration imply

\[
\Pi_{D}^{\text{start}} = \sum_{j=1}^{K-l} p_{\text{start}}^{j,1} \sum_{l'=0}^{j} p_{l',1}^{j,1}(x).
\]

Given this result, we have:

\[
\lambda \Pi_{0,0} = \mu \Pi_{1,0},
\]

\[
(l + 1) \mu \Pi_{l+1,0} + \lambda \alpha_l \Pi_{K,0},
\]

for \(1 \leq l < K\).

Next, we focus on the ratio of long run fraction of time spent in deferral configurations. This is given by

\[
\frac{\Pi_{k_1,d_1}}{\Pi_{k_2,d_2}} = \frac{\sum_{k=1}^{K} \sum_{d=1}^{D-l} p_{k,d}^{k_1,d_1}(t) dt}{\sum_{k=1}^{K} \sum_{d=1}^{D-l} p_{k,d}^{k_2,d_2}(t) dt},
\]

where \(0 \leq k_1, k_2 \leq K\) and \(1 \leq d_1, d_2 \leq D\).

Finally, it remains to relate the time fractions corresponding to deferral configurations to those corresponding to non-deferral configurations. This is done as follows:

\[
\lambda \Pi_{K,0} = \left( \sum_{k=1}^{K} \sum_{d=1}^{D} \Pi_{k,d} \right) \left( \sum_{j=1}^{K-l} p_{\text{start}}^{j,1} \sum_{l'=0}^{j} p_{l',1}^{j,1}(x) \right) \frac{x}{1 - \hat{\mathbf{p}}_{\text{stat}}^{D}}.
\]

This is justified as follows. \(\lambda \Pi_{K,0}\) gives the rate of transition from non-deferral configurations to deferral configurations. On the other hand, the rate of rearivals equals \(\sum_{j=1}^{K-l} p_{\text{start}}^{j,1} \sum_{l'=0}^{j} p_{l',1}^{j,1}(x) \frac{x}{1 - \hat{\mathbf{p}}_{\text{stat}}^{D}}\) gives the probability of exiting the
positive deferral configurations at the end of any deferral stage of duration \( x \). This follows by combining these arguments.

Solving the system of linear equations \((7) - (9)\) along with the normalization condition yields \( \Pi_{k,d} \) for \((k, d) \in [K] \times [D] \).

4) Blocking probability characterization: The blocking probability can be characterized in a similar manner as before.

Lemma 2: Under the DSRT policy,

\[
P_B = \Pi_{K,D} + \Pi_{K,0} \sum_{k=1}^{K} \sum_{d=1}^{D-1} p_{k,d}^{\text{start}} \sum_{j=1}^{D} p_{k,j}^{\text{end}}(x) \sum_{l=0}^{K} p_{l,1}^{\text{start}} \sum_{j=0}^{D} p_{l,j}^{\text{end}}(x). \tag{10}
\]

Proof: Customers are blocked in two ways. First, a customer arrives when the system configuration is \((K, D)\). Using the PASTA property, the long run fraction of arrivals blocked this way equals \( \Pi_{K,D} \), which accounts for the first term in \((10)\).

Second, deferred customers can get blocked when all servers are busy upon rearrival. Let \( DP(t) \) denote the number of deferral phases completed until time \( t \), and \( B_i \) denote the number of deferred customers blocked in deferral phase \( i \). Let \( A(t) \) denote the number of arrivals until time \( t \). The long run fraction of arrivals blocked upon rearrival equals

\[
\lim_{t \to \infty} \frac{\sum_{i=1}^{DP(t)} B_i}{A(t)} = \lim_{t \to \infty} \frac{t}{A(t)} \frac{DP(t)}{\sum_{i=1}^{DP(t)} B_i} = \frac{1}{\lambda} \left( \Pi_{K,0} \lambda B_1 \right).
\]

Finally, a renewal reward argument analogous to that in the proof of Lemma \((1)\) yields

\[
\mathbb{E}[B_1] = \sum_{k=1}^{K} \sum_{d=1}^{D-1} p_{k,d}^{\text{start}} \sum_{j=1}^{D} p_{k,j}^{\text{end}}(x) \sum_{l=0}^{K} p_{l,1}^{\text{start}} \sum_{j=0}^{D} p_{l,j}^{\text{end}}(x) - \sum_{k=1}^{K} \sum_{d=1}^{D-1} p_{k,d}^{\text{start}} \sum_{j=1}^{D} p_{k,j}^{\text{end}}(x) \sum_{l=0}^{K} p_{l,1}^{\text{start}} \sum_{j=0}^{D} p_{l,j}^{\text{end}}(x).
\]

D. Case Study

In this section, we numerically compare the performance of the DSRT policy with state-independent policies and the \(M/M/K/K\) system in the Halfin-Whitt regime (a.k.a. quality and efficiency driven regime; see \([4]\)). We consider two simple state independent policies: the first policy assigns a deferral time equal to \( T \) and the second policy assigns a deferral time uniformly sampled in the interval \([0, T]\). The Halfin-Whitt regime corresponds to a sequence of queueing systems parameterized by the number of servers \( K \). The arrival rate in the system with \( K \) servers equals \( K + \beta \sqrt{K} \), where \( \beta \) is a constant. In our experiments, the service rate for each server is set to be 1, the patience time \( \hat{T} \) is set to be 10, and \( \beta \) is set to be 0.1.

We compare the policies for the cases when \( \hat{D} \) equals 1 and 2. We emphasize that \( \hat{D} \) acts as a parameter for the state-independent policies while the deferral limit for DSRT

\[D = \min(\hat{D}, \lfloor T/x \rfloor)\], where \( x \leq T \). For DSRT, the blocking probability is evaluated numerically using the procedure developed in previous sections and is optimized by tuning the parameter \( x \). For the state independent policies, we run MCMC simulations where the blocking probability is calculated for 100,000 customers. The blocking probability for \(M/M/K/K\) system is calculated numerically using the Erlang-B formula. The blocking probabilities of the policies for the cases where \( \hat{D} \) is equal to 1 and 2 are given in Figure 5a and Figure 5b respectively. We note that DSRT outperforms the state-independent policies, which produce a blocking probability close to the Erlang-B formula.

![Fig. 5: Performance Evaluation](image)

(a) Maximum Deferrals: 1  
(b) Maximum Deferrals: 2

Fig. 5: Performance Evaluation

![Fig. 6: Optimal \( x \) for DSRT](image)

Fig. 6: Optimal \( x \) for DSRT

IV. DEFERRAL TIMES BOUNDED IN EXPECTATION

In this section, we consider a relaxation of the deferral constraint considered before, in that the upper bound on deferral times is imposed in expectation (rather than on each instance, as in Section III). Specifically, we require that the expected deferral time should not exceed \( \hat{T} \). This relaxation may be interpreted as an approximation of the ‘hard’ deferral constraint considered before.

In the above setting, we propose and analyse a deferral policy, referred to as the Exponentially Spaced Rearrival Times (ESRT) policy. As the name suggests, this policy introduces an exponentially distributed spacing between rearrival instants. Specifically, the ESRT policy allots exponentially separated deferral times, i.e., the difference between the rearrival times
of deferred arrivals is exponential with rate $\alpha$, where $1/\alpha \leq \bar{T}$, the deferral limit being $D = \min(D, \bar{T}/\alpha)$; this policy is described formally in Algorithm 2. Note that $\text{Exp}(\alpha)$ refers to an exponential random variable with rate $\alpha$ that is independent of the workload, as well as other inter-rearival intervals.

**Algorithm 2** ESRT Policy

```plaintext
procedure
    if num_deferrals = 0 then
        rearrival_time = arrival_time + Exp($\alpha$)
    else if 0 < num_deferrals < D then
        rearrival_time = prev_rearival_time + Exp($\alpha$)
    else
        customer blocked
end if
end procedure
```

Given the memorylessness of the interarrival times, service times, and the time until next rearrival, the evolution of the system under the ESRT policy is described by a continuous time Markov chain over the state space $[K] \times [D]$, where state $(k, d)$ indicates that $k$ servers are currently active, and there are $d$ jobs in deferral. The transition rate diagram for this Markov chain is depicted in Figure 7. Note that memorylessness of the time until the next rearrival results in a much simpler system description (compared to the DSRT policy analysed in the previous section), and yields a much more explicit performance evaluation. In the following, we first compute the stationary distribution corresponding to the ESRT Markov chain, and then use that to characterize the blocking probability.

![Figure 7: Dynamics of ESRT](image)

A. Obtaining the stationary distribution

Matrix geometric methods can be used to obtain the stationary distribution corresponding to the Markov chain shown in Figure 7. Let $\Pi_{k,d}$ represent the steady state probability of state $(k,d)$.

We begin by writing the flow balance equations for states with $D$ deferrals

$$(\lambda + \alpha)\Pi_{0,D} = \mu\Pi_{1,1},$$

$$(\lambda + j\mu + \alpha)\Pi_{j,D} = (j + 1)\mu\Pi_{j+1,D} + \lambda\Pi_{j-1,D},$$

for $(0 < j < K)$.

Letting $\bar{\Pi}_D := [\Pi_{0,D}, \ldots, \Pi_{K,D}]$, the above system of $K$ equations can be used to characterize $\bar{\Pi}_D$ up to a multiplicative constant. Specifically, we write

$$\bar{\Pi}_D = c\bar{V}_D,$$

where $\bar{V}_D$ is a row vector whose last entry is 1.

Now, consider the collection of states corresponding to $d$ deferrals, where $d \in \{1, \ldots, D-1\}$ deferrals. The flow balance equations for these states are as follows

$$(\lambda + \alpha)\Pi_{0,d} = \mu\Pi_{1,d},$$

$$(\lambda + j\mu + \alpha)\Pi_{j,d} = (j + 1)\mu\Pi_{j+1,d} + \lambda\Pi_{j-1,d} + \alpha\Pi_{j-1,d+1},$$

for $(0 < j < K)$,

$$\lambda\Pi_{K,d} = \alpha\sum_{i=0}^{K} \Pi_{i,d+1}.$$
that the long run fraction of jobs dropped on rearrival equals the following explicit form:

$$\alpha$$

be blocked either on arrival into state in terms of the stationary distribution. As before, a job may with the system state, we now express the blocking probability $B$. The blocking probability $R$ where $c, \beta, \hat{\alpha}$ are obtained by applying the law of total probability:

$$\lambda \Pi_{K,0} = \alpha \sum_{i=0}^{K} \Pi_{i,1}.$$  

The equations above can be written as $\tilde{\Pi}_0 A(0) = \tilde{\Pi}_1 B(\alpha)$, implying

$$\tilde{\Pi}_0 = \tilde{\Pi}_1 R_1(\alpha)$$  

(13)

where $R_1(\alpha) = B(\alpha) A(0)^{-1}$.

Using (11)–(13), the stationary distribution is completely determined up to the multiplicative constant $c$, which can be obtained by applying the law of total probability:

$$c \nu D \left[R_1 R D^{-1} + (I - R)^{-1}(I - R D)\right] \mathbb{1} = 1$$

B. The blocking probability

Having characterized the stationary distribution associated with the system state, we now express the blocking probability in terms of the stationary distribution. As before, a job may be blocked either on arrival into state $(K, D)$, or on rearrival if all servers are occupied. Since $\alpha \sum_{d=1}^{D} \Pi_{K,d}$ is the rate of dropped rearrivals, an elementary renewal argument shows that the long run fraction of jobs dropped on rearrival equals $\alpha \sum_{d=1}^{D} \Pi_{K,d}$. Thus,

$$P_B = \Pi_{K,D} + \frac{\alpha}{\lambda} \sum_{d=1}^{D} \Pi_{K,d}.$$  

For the special case $K = 1$, the blocking probability takes the following explicit form:

$$P_B(1, D) = \frac{\lambda}{\lambda + \mu} \times \frac{(ab)^{D+1} + (b - 1)(ab)^{D} - b}{ab - 1 + a \frac{\alpha}{\lambda + \mu} [(ab)^{D} - 1]}$$

where $a = \frac{\lambda}{\alpha}$ and $b = \frac{\lambda + \alpha}{\lambda + \alpha + \mu}$.

C. Case Study

Given that the ESRT policy is more tractable than the DSRT policy, our first goal is to check if the performance of the former is a good approximation to the performance of the latter. The numerical analysis is done for the case when $\hat{D} = 2$, and $\hat{T} = 10$. i.e., the system is allowed to defer one or two customers. We then consider a sequence of queueing systems under the Halfin-Whitt scaling regime, with $\beta = 0.1$. In Figure 8a, we compare the blocking probability under ESRT using the optimal computed value $\alpha^*(K)$ of the policy parameter, with the blocking probability under DSRT using $x = 1/\alpha^*(K)$.

Note that the two curves are nearly indistinguishable, suggesting that the approximation is sound (at least for small values of $\hat{D}$). In Figure 8b, we plot $1/\alpha^*(K)$ v/s $K$. Since $1/\alpha^*(K)$ decreases with $K$, it follows that it is optimal to decrease the spacing between rearrivals as the size of the system grows under the Halfin-Whitt regime. This is to be expected, since the arrival rate and the service rate of the system grow nearly proportionately under this scaling regime.

Next we study the impact of the maximum number of deferrals $\hat{D}$. Under the Halfin-Whitt scaling regime as before, in Figure 9a, we plot the optimal (with respect to the deferral parameter $\alpha$) blocking probability when $\hat{D} = 10$ and $\hat{D} = 1000$, along with the Erlang B formula (which corresponds to $\hat{D} = 0$.) For this case, we set $\hat{T} = 1$. Note that increasing the limit on the number of deferrals does improve the performance significantly. Moreover, it is well-known that the blocking probability of the Erlang B (M/M/K/K) system decays as $\Omega(1/\sqrt{K})$ under the Halfin-Whitt scaling (see [5]). The numerical results as shown in Figure 9a suggest that the decay of blocking probability of ESRT is also $\Omega(1/\sqrt{K})$ (note the nearly linear scaling of blocking probability with $K$ on a log-log scale), at least when $\hat{D}$ is small. Having a large value of $\hat{D}$ could possibly allow a $o(1/\sqrt{K})$ decay in the blocking probability. Formalizing this scaling behavior is an interesting avenue for future work.

When $\hat{D}$ is 1000, we also plot the inverse of optimal deferral parameter $\alpha$ against the number of servers in Figure 9b. Notice the nearly linear decay of $1/\alpha$ in the log-log plot, indicating a regularly varying scaling with the number of servers in the Halfin-Whitt regime. Characterizing the optimal deferral parameters for both ESRT and DSRT would also be illuminating.

V. CONCLUDING REMARKS

Our work motivates several directions for future work including the following. One, characterizing the optimal parameter values and optimal blocking probabilities in a large system limit for DSRT/ESRT. Second, designing deferral policies operating under constraints on the number of deferrals and patience time that can beat the $1/\sqrt{K}$ blocking probability scaling under the Halfin-Whitt regime. Finding fundamental lower bounds on the blocking probability of such deferral policies would also be interesting.
REFERENCES

[1] J. R. Artalejo and A. Gómez-Corral. Retrial queueing systems. Mathematical and Computer Modelling, 30(3-4):xiii–xv, 1999.
[2] F. Baccelli and G. Hebuterne. On queues with impatient customers. 1981.
[3] B. Doshi, C. Kumar, P. Piyush, and M. Vutukuru. Webq: A virtual queue for improving user experience during web server overload. In 2015 IEEE 23rd International Symposium on Quality of Service (IWQoS), pages 135–140. IEEE, 2015.
[4] S. Halfin and W. Whitt. Heavy-traffic limits for queues with many exponential servers. Operations research, 29(3):567–588, 1981.
[5] D. L. Jagerman. Some properties of the erlang loss function. Bell System Technical Journal, 53(3):525–551, 1974.
[6] J. Kim and B. Kim. A survey of retrial queueing systems. Annals of Operations Research, 247(1):3–36, 2016.
[7] S. Zeltyn and A. Mandelbaum. Call centers with impatient customers: many-server asymptotics of the m/m/n+ g queue. Queueing Systems, 51(3-4):361–402, 2005.