Finite-Source $M/M/S$ Retrial Queue with Search for Balking and Impatient Customers from the Orbit

Patrick Wüchner[a,*], János Sztrik[b], Hermann de Meer[a]

[a] *Faculty of Informatics and Mathematics, University of Passau, Innstraße 43, 94032 Passau, Germany*
[b] *Faculty of Informatics, University of Debrecen, Egyetem tér 1, Po.Box 12, 4010 Debrecen, Hungary*

Abstract

The present paper deals with a generalization of the homogeneous multi-server finite-source retrial queue with search for customers in the orbit. The novelty of the investigation is the introduction of balking and impatience for requests who arrive at the service facility with a limited capacity and FIFO queue. Arriving customers may balk, i.e., they either join the queue or go to the orbit. Moreover, the requests are impatient and abandon the buffer after a random time and enter the orbit, too. In case of an empty buffer, each server searches for a customer in the orbit after finishing service. All random variables involved in the model construction are supposed to be exponentially distributed and independent of each other. The primary aim of this analysis is to show the effect of balking, impatience, and buffer size on the steady-state performance measures. Concentrating on the mean response time, several numerical examples are investigated by the help of the MOSEL-2 tool used for creating the model and calculating the stationary characteristics.

Key words: Performance modeling, Finite-source retrial queues, Orbital search, Balking customers, Impatient customers, MOSEL-2

* *Corresponding author.* Tel.: +49 851 509 3054; fax: +49 851 509 3052
Email addresses: `patrick.wuechner@uni-passau.de` (Patrick Wüchner), `jsztrik@inf.unideb.hu` (János Sztrik), `hermann.demeer@uni-passau.de` (Hermann de Meer)

1. Introduction

In many real-life situations, it is unrealistic to assume that a customer enters a queue if all service providers are busy at arrival time. Consider, for example, a telephone call, where in general there is no possibility to queue if the line is busy. Another example is a hairdresser's shop, where a finite number of chairs are provided to waiting customers. All chairs might be occupied at arrival instant, and thus, an arriving customer is not able to enter the queue. In both cases, the customer may not leave the system forever, but might retry to be served at another time.

Such situations are commonly modeled using the *retrial queue* formalism introduced in [12]. Basic retrial queues can be described as follows. Customers arrive at the system according to a Poisson process with state-independent arrival rate. If an arriving customer finds all (identical) servers busy, it does not leave the system but joins the so-called orbit of infinite size instead. All orbiting customers will retry to be served after an exponentially distributed retrial time. They will re-join the orbit, if they find all servers busy. Customers in service will leave the system after an exponentially distributed service time.

These basic retrial queues were generalized in various directions by a large number of publications. Surveys and bibliographies of these works are given in [3, 4, 5, 6, 13, 14, 18, 29].

Basic retrial queues and their generalizations have been applied to model telephone traffic in [12], Ethernet systems in [1], active queue management of Internet routers in [7], self-organizing P2P systems in [28], inventory systems in [24], and mobile communications in [2, 19, 21, 22, 25]. In [3, 6, 11, 29], further application examples are given.

In this article, we investigate finite-source retrial queues with multiple homogeneous servers that conduct orbital search and discuss the influence of customer balking and impatience as well as the effect of the service area's finite capacity on the mean response time. Here, the service area encompasses the waiting area (queue) by definition.

Finite-source retrial queues are motivated by the fact that in many applications the number of potential customers is far less than infinite. In these cases, the arrival process is non-Poisson but depends on the number of customers already in the system (see [6, page 32]).

Customer *balking* refers to (discouraged) customers that are entering the orbit although there is some space available in the service area. Joining the orbit might be more attractive than joining a long queue, e.g., in wireless sensor networks, where queueing sensors may need to be online and waste valuable energy, whereas orbiting sensors may retreat to a power-saving stand-by mode. The concept of balking was first introduced in [16] for classical $M/M/1$–FCFS queues. In [23], balking is considered for infinite-source retrial-queues. There, however, the balking is deterministic and depending on a fixed threshold value of the queue size. In this paper, we introduce a new method that describes a balking probability depending on the current number of customers located in the queue and an *encouraging parameter*.

Impatience (also known as *reneging*, see [17]) lets queueing customers leave the queue and join the orbit instead. For example, customers that joined the queue might be able to observe the service area while waiting. This allows them to estimate the mean service time and queue length and hence, also their remaining waiting time. Based on this estimation, they may decide to leave the queue and retry to get service later, hoping for a shorter queue on return. Impatience in retrial queues is already considered in [12].

In some scenarios, idle servers are able to inform orbiting customers of their status. This allows servers to fetch customers directly from the orbit with some probability if there are no other customers waiting in the queue at service completion instant. This behavior is called *orbital search* and introduced in [20]. We treated finite-source retrial queues without balking and impatience in [27].

Infinite-source multi-server retrial queues with balking and impatient customers have been treated recently in [23]. We are not aware of any publication discussing the effects of service area capacity, balking, impatience, *and* orbital search in a finite-source context.

The performance measures of our model are obtained by numerical analysis carried out using the MOSEL-2 performance evaluation tool. The first version of MOSEL (see [8]) was designed and developed in the late nineties at the research group for performance evaluation and process control, lead by Dr. Gunter Bolch, at the University of Erlangen, Germany. Due to a major revision in 2003 (see [9]), MOSEL was renamed to MOSEL-2, which is now maintained at the Chair of Computer Networks and Communications, University of Passau, Germany[1]. For a short introduction to MOSEL-2, we refer the interested reader to [26]. A discussion of the performance and scalability

[1]Project homepage: `http://mosel2.net.fim.uni-passau.de/`

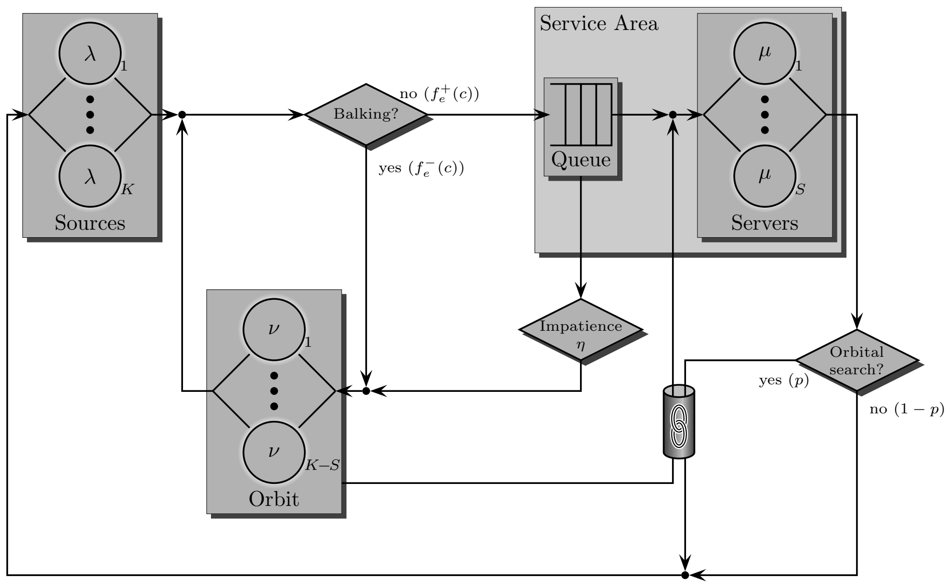

Figure 1: Queueing model of finite-source retrial queue with balking, impatience, and orbital search.

of MOSEL-2 in the context of finite-source retrial queues is provided in [27].

The remainder of this article is organized as follows. In Sec. 2, the investigated model is introduced and the notations used in this paper are defined. The underlying Markov chain and interesting performance measures are derived in Sec. 3. The numerical analysis is carried out using the MOSEL-2 tool in Sec. 4. In Sec. 5, various numerical results are presented and discussed in detail. Finally, in Sec. 6, a conclusion and directions for future work are given.

2. Model Description

The queueing model given in Fig. 1 illustrates the finite-source retrial queue with balking, impatience, and orbital search.

The behavior of the retrial queue can be described as follows. Each of the K sources is generating requests with rate λ as long as it is not waiting for a response to an active request located in the orbit or in the service area. A request arriving from the sources first checks whether there is an idle server. If there is an idle server, the request will enter this server immediately. If all S identical servers are busy, the request enters the queue of size Q with

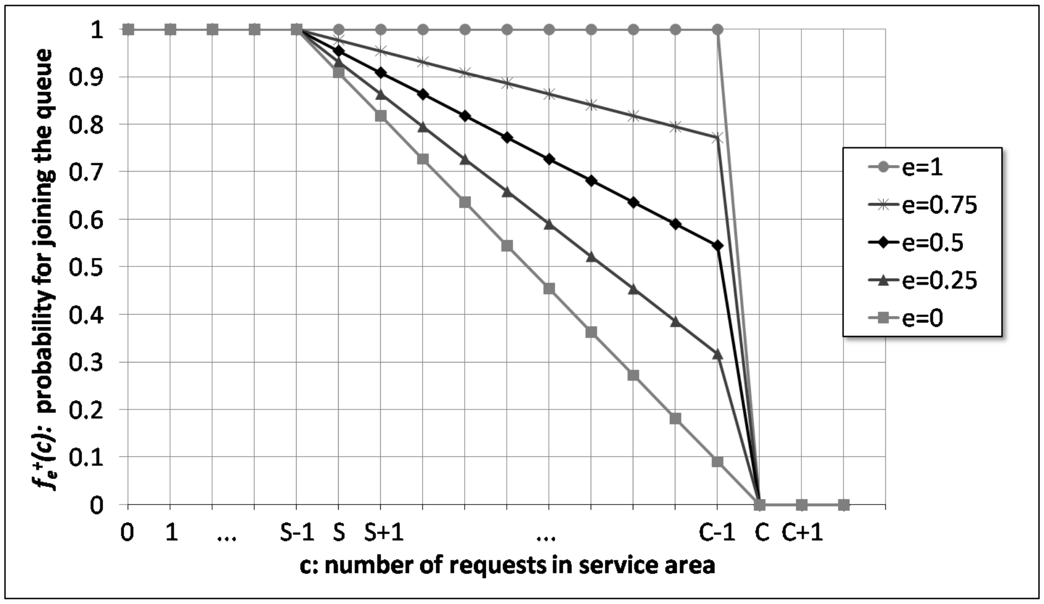

Figure 2: Probability for joining the queue $f_e^+(c)$ over number of requests c located in the service area for different values of the encouraging parameter e as given by Eq. (1).

probability $f_e^+(c)$ given by

$$f_e^+(c) = \begin{cases} 1 & : \quad c < S\,, \\ \frac{e-1}{Q+1}(q+1)+1 & : \quad S \leq c < C\,, \\ 0 & : \quad c = C\,, \end{cases} \tag{1}$$

where c is the current number of requests located in the service area, $q = \max(0, c - S)$ is the current number of requests waiting in the queue, $C = S + Q$ is the finite capacity of the service area, and $e, 0 \leq e \leq 1$, is the encouraging parameter. The outcome of Eq. (1) is illustrated in Fig. 2 for various values of e. With a probability of $f_e^-(c) = 1 - f_e^+(c)$, the request balks and enters the orbit instead. Since an arriving request will enter the orbit only if there are at least S request located in the servers, the maximum number of requests located in the orbit is equal to $K - S$.

Note that the linear balking probability $f_e^+(c)$ given by Eq. (1) is just an example that suggests itself and serves well in several application scenarios. However, the numerical evaluation using MOSEL-2 allows extending the model flexibly by more general (state-dependent) balking probabilities.

Requests located in the orbit retry to enter the service area with retrial

Table 1: Overview of model parameters.

Parameter	Maximum	Value at t	Mean ($t\to\infty$)
Number of active sources	K (population size)	$k(t)$	$\overline{K}$
Request generation rate		λ	
Total request gen. rate	λK	$\lambda k(t)$	$\widetilde{\lambda}=\lambda\overline{K}$
Encouraging parameter	1	e	
Requests in queue	$Q=C-S$ (queue size)	$q(t)$	$\overline{Q}$
Impatience rate		η	
Service rate		μ	
Busy servers	S (number of servers)	$s(t)$	$\overline{S}$
Server utilization	1		$U_S=\frac{\overline{S}}{S}$
Requests in service area	C (finite capacity)	$c(t)=s(t)+q(t)$	$\overline{C}=\overline{Q}+\overline{S}$
Orbiting requests	$O=K-S$ (orbit size)	$o(t)$	$\overline{O}$
Work in progress	K	$m(t)=c(t)+o(t)$	$\overline{M}=\overline{C}+\overline{O}$
Retrial rate		ν	
Orbital search probability	1	p	

rate ν. Requests waiting in the queue start being served as soon as an idle server gets assigned to them. However, requests leave the queue with rate η and join the orbit due to impatience.

Busy servers complete requests with service rate μ. A response returns to the requesting source after service. If the queue is empty and there are requests located in the orbit, at service completion instant, the server carries out orbital search, i.e., it instantly fetches a request directly from the orbit with a probability of p, where $0 \leq p \leq 1$.

Table 1 gives an overview on the model parameters.

To preserve mathematical tractability, all inter-event times (i.e., request generation time, impatience time, service time, and retrial time) involved in the model are assumed to be exponentially distributed and regulated by their rate parameter. Extending the model by including phase-type distributions is considered to be straight forward. We present a discussion of the solution method's scalability in Sec. 4.

3. Underlying Markov Chain and Performance Measures

The behavior of the finite-source retrial queue with balking, impatience, and orbital search as described in Section 2 can be represented by a bivariate continuous-time Markov chain (CTMC) with state variable $X(t) = (o(t),\, c(t))$, where variable $o(t), 0 \leq o(t) \leq O$, is the number of requests in

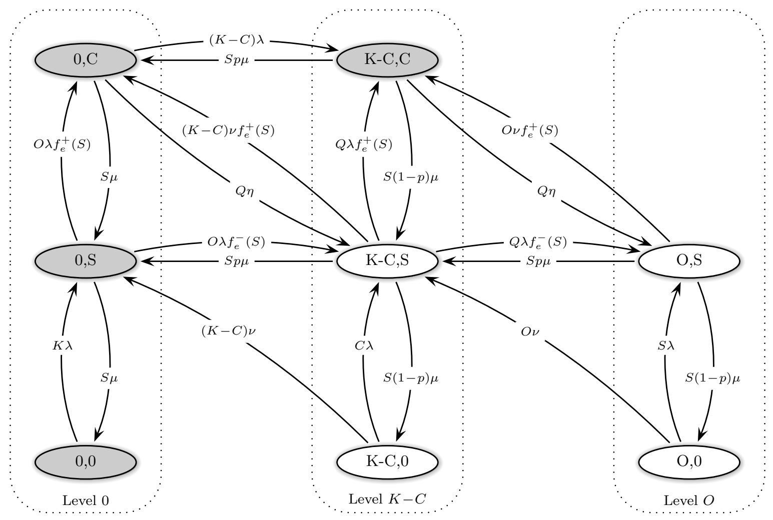

Figure 3: State transition diagram of finite-source retrial queue with balking, impatience, and orbital search for $K = 3, C = 2, O = 2$, and $S = 1$.

the orbit and variable $c(t), 0 \leq c(t) \leq C$, is the number of requests in the service area at time $t \geq 0$. Since the number of active sources is given by $k(t) = K - o(t) - c(t)$, and due to the memoryless property of the exponentially distributed inter-event times involved, $X(t)$ sufficiently describes the state of the system at any time t.

In Fig. 3, the state transition diagram of the CTMC is sketched for $K = 3, C = 2, O = 2$, and $S = 1$.

Note that for $p \approx 1$ or very large ν, the orbit tends to act like additional queue capacity and the retrial queue then behaves quite similar to the classical $M/M/S/K/K$–FCFS queue. Moreover, for $p = 1$, $\eta = 0$, and $f_e^-(S) = 0$ (i.e., $f_e^+(S) = 1$), Fig. 3 reduces to the gray-colored states which then even form the state transition diagram of the classical $M/M/S/K/K$–FCFS queue.

The CTMC illustrated in Fig. 3 is finite and irreducible for all reasonable (i.e., strictly positive) rates λ, μ, ν, and η. Hence, the CTMC is positive recurrent which implies ergodicity (see [10, page 70]). This also holds for higher (but finite) values of $K \geq C \geq S$ and $O = K - S$.

Figure 3, suggests a partitioning of the state space into levels reflecting $o(t)$, i.e., the number of requests in the orbit. Each level consists of a subset of states, called phases, indicating $c(t)$, i.e., the number of requests in the service area. Since transitions only take place between adjacent levels, the structured CTMC is skip-free and constitutes a finite quasi-birth-death process (QBD). This gives rise to efficient algorithms like the one presented in [15], which has potential to lead to closed-form equations of performance measures in special cases (e.g., $C = 2$). See also [1] for similar discussions.

In this paper, however, all performance measures are derived conveniently using the MOSEL-2 tool (see Sec. 4) which takes care of the generation of the CTMC and the numerical solution of the system of steady-state equations to obtain the steady-state probabilities.

Let us denote by $\pi_{o,c}$ the steady-state probability that there are $o, 0 \leq o \leq O$, requests located in the orbit, and $c, 0 \leq c \leq K$, customers located in the service area, i.e., $\pi_{o,c} = \lim_{t\to\infty} P(o(t) = o, c(t) = c)$.

In the following, interesting steady-state performance measures are presented assuming the steady-state probabilities $\pi_{o,c}$ are known.

- **Mean number of busy servers $\overline{S}$:** The mean number of busy servers is given by
$$\overline{S} = \sum_{o=0}^{O} \left\{ \sum_{c=1}^{S} c\pi_{o,c} + \sum_{c=S+1}^{C} S\pi_{o,c} \right\} . \tag{2}$$
- **Server utilization U_S:** The utilization of the servers can then be calculated via
$$U_S = \frac{\overline{S}}{S} . \tag{3}$$
- **Mean queue length $\overline{Q}$:** The mean number of requests located in the queue is given by
$$\overline{Q} = \sum_{o=0}^{O} \sum_{c=S+1}^{C} (c - S)\pi_{o,c} . \tag{4}$$
- **Mean orbit size $\overline{O}$:** The mean number of requests located in the orbit is given by
$$\overline{O} = \sum_{o=0}^{O} \sum_{c=0}^{C} o\pi_{o,c} . \tag{5}$$
- **Mean number of active requests $\overline{M}$:** The mean number of requests located in the service area or in orbit is given by
$$\overline{M} = \overline{S} + \overline{Q} + \overline{O} . \tag{6}$$

- **Mean number of active sources $\overline{K}$:** The mean number of request-generating sources is then given by

$$\overline{K} = K - \overline{M}\,. \tag{7}$$

- **Mean system throughput $\widetilde{\lambda}$:** The mean throughput of the finite-source retrial queue with orbital search can be obtained from

$$\widetilde{\lambda} = \overline{K}\lambda\,. \tag{8}$$

- **Mean orbit time $\overline{T}_O$:** The mean overall time spent by each request in the orbit before service can be calculated by

$$\overline{T}_O = \frac{\overline{O}}{\widetilde{\lambda}}\,. \tag{9}$$

  Equation (9) can be motivated as follows. Let us denote the orbit's throughput by λ_O. Following Little's Law (see [10, page 245]), the mean time spent at the orbit (per visit) by every request arriving at the orbit is $\overline{T}^v_O = \frac{\overline{O}}{\lambda_O}$. The mean number of visits to the orbit per request before being served is $e_O = \frac{\lambda_O}{\widetilde{\lambda}}$ (see [10, page 324]). Hence, the overall mean time spent in the orbit before being served is equal to $\overline{T}_O = e_O\overline{T}^v_O = \frac{\lambda_O}{\widetilde{\lambda}}\frac{\overline{O}}{\lambda_O} = \frac{\overline{O}}{\widetilde{\lambda}}$.

- **Mean number of retrials $\overline{R}$:** The mean number of retrials before service can then be calculated via

$$\overline{R} = \nu\overline{T}_O\,. \tag{10}$$

- **Mean response time $\overline{T}$:** The mean time spend by each request in the orbit and the service area can also be calculated by applying Little's Law, i.e.,

$$\overline{T} = \frac{\overline{M}}{\widetilde{\lambda}}\,. \tag{11}$$

Listing 1: MOSEL-2 model of finite-source retrial queue with balking, impatience, and orbital search.

```
/*** CONSTANTS AND PARAMETERS *****************************************************/
// request generation rate (x-axis):
PARAMETER  lambda := 0.001, 0.05 .. 1 STEP 0.05, 1.1 .. 2 STEP 0.1;

CONST       K       := 100;                                    // population size
CONST       S       := 3;                                      // number of servers
CONST       C       := 10;                                     // capacity
CONST       Q       := C-S;                                    // queue size
CONST       O       := K-S;                                    // orbit size
CONST       e       := 0.5;                                    // encouraging
CONST       eta     := 1;                                      // impatience rate
CONST       nu      := 0.4;                                    // retrial rate
CONST       mu      := 10;                                     // service rate
CONST       p       := 0.5;                                    // search probability

/*** NODES ************************************************************************/
NODE Sources[K]   := K;                                        // the sources
NODE Request[1]   := 0;                                        // arriving request
NODE Buffer[C]    := 0;                                        // the servers + queue
NODE Orbit[O]     := 0;                                        // the orbit
NODE Finished[1]  := 0;                                        // response

/*** RULES ************************************************************************/
FROM Sources      TO   Request RATE Sources*lambda;               // primary requests
IF (Buffer < S)  FROM Request TO Buffer;                          // server available
IF (Buffer == C) FROM Request TO Orbit;                           // buffer full

// balking customers:
IF (Buffer >= S AND Buffer < C) FROM Request TO Buffer
   WEIGHT (e-1)/(Q+1)*(Buffer-S+1)+1;
IF (Buffer >= S AND Buffer < C) FROM Request TO Orbit
   WEIGHT (1-e)/(Q+1)*(Buffer-S+1);

IF (Buffer > S)  FROM Buffer   TO Orbit RATE (Buffer-S)*eta; // impatient customers
FROM Orbit        TO   Request  RATE Orbit*nu;                // retrials
IF (Buffer <= S) FROM Buffer   TO Finished RATE Buffer*mu;    // service
IF (Buffer > S)  FROM Buffer   TO Finished RATE S*mu;         // service
IF (Buffer >= S) FROM Finished TO Sources;                    // without orb. search
IF (Buffer < S)  FROM Finished TO Sources WEIGHT 1-p;         // without orb. search
IF (Buffer < S)  FROM Finished, Orbit TO Sources, Buffer WEIGHT p;  // with orb. s.

/*** RESULTS **********************************************************************/
PRINT mM        := MEAN(Orbit)+MEAN(Buffer);               // mean # active req.
PRINT mK        := K-mM;                                   // mean # active sources
PRINT ml        := mK*lambda;                              // mean throughput
PRINT mT        := mM/ml;                                  // mean response time

PICTURE "mean_response_time" PARAMETER lambda CURVE mT;
```

4. Numerical Analysis Using MOSEL-2

Finite-source retrial queues with balking, impatience, and orbital search can be evaluated numerically quite easily by using the MOSEL-2 performance evaluation tool. The corresponding MOSEL-2 model is shown in Listing 1.

As a reference value, we can state that the solvers applied by MOSEL-2 are able to evaluate models that comprise up to approximately $9 \cdot 10^5$ tangible states. This is shown in [27] also in the context of finite-source retrial queues. The exact value, however, depends on the model details and on the hardware used.

In Tab. 2, we exemplarily give a selection of model parameter settings we expect to be evaluable using MOSEL-2 since the number of states taken by the Markov chain described in Sec. 3 is smaller than $9 \cdot 10^5$. However, as it is shown in [27], it takes about 20 minutes to evaluate a single parameter setting comprising $9 \cdot 10^5$ states using MOSEL-2 on a PC with Linux operating system, 2 GHz CPU, and 2 GB of RAM.

Table 2: State space sizes calculated using Eq. (13) for various values of the population size K, queue size $Q = C - S$, number of servers S, and number of phases P per server.

K	Q	S	P	Number of States
7000	60	60	1	838031
65000	5	7	1	844907
5000	10	10	2	877811
20000	4	5	2	899760
1000	9	10	3	869110
1000	5	5	5	876582
100	3	10	10	852852

To calculate the number of states given in Tab. 2, we proceed as follows. We assume that $K \geq C \geq S$.

In the case of an empty queue ($q(t) = 0$), the orbit may contain up to $(K - S)$ requests and hence, can take $(K - S + 1)$ states (including being empty). The servers may contain up to S requests. All requests not located in the orbit or in the servers will be located in the sources. The number of possible states for $q(t) = 0$ is then given by $(K - S + 1)(S + 1)$.

In the case of a non-empty queue ($q(t) > 0$), all servers have to be busy ($s(t) = S$). The orbit may then contain up to $(K - (S + q(t)))$ requests, resulting in $(K - (S + q(t)) + 1)$ possible orbit states. The number of requests waiting in the queue may vary from one to Q. Thus, if there is at least one request waiting in the queue, the system can take

$$\begin{aligned}\sum_{q(t)=1}^{Q} (K - S + 1 - q(t)) &= Q(K - S + 1) - \sum_{q(t)=1}^{Q} n \\ &= Q(K - S + 1) - \frac{Q(Q+1)}{2} \qquad (12)\end{aligned}$$

different states.

Hence, for arbitrary queue lengths ($0 \leq q(t) \leq Q$), the total number of possible states is

$$(K-S+1)(S+1)+Q(K-S+1)-\frac{Q(Q+1)}{2}\,.$$

We can generalize this discussion by allowing for phase-type service with P phases at each server. We then additionally have to keep track of the number of requests located in each phase. If there are $s(t)$ requests located in the servers, these requests are distributed throughout the phases which is possible in

$$\binom{P+s(t)-1}{P-1}=\binom{P+s(t)-1}{s(t)}=\frac{(P+s(t)-1)!}{s(t)!(P-1)!}$$

different ways (see also [10, page 346]).

In the case of an empty queue ($q(t)=0$), up to S servers might be busy and up to $(K-S)$ request might stay at the orbit, resulting in

$$(K-S+1)\left(1+\sum_{s(t)=1}^{S}\frac{(P+s(t)-1)!}{s(t)!(P-1)!}\right)$$

different states.

If there is at least one request waiting in the queue ($q(t)>0$), all servers are busy ($s(t)=S$), leading to

$$\frac{(P+S-1)!}{S!(P-1)!}$$

different server states and the number of states taken by the orbit is following Eq. (12).

Finally, the Markov chain can take

$$\begin{aligned}(K-S+1)\quad&\left(1+\sum_{s(t)=1}^{S}\frac{(P+s(t)-1)!}{s(t)!(P-1)!}\right)\\&+\left(Q(K-S+1)-\frac{Q(Q+1)}{2}\right)\frac{(P+S-1)!}{S!(P-1)!}\end{aligned}\qquad(13)$$

different states in the case of phase-type service.

Table 3: Numerical values of model parameters.

Parameter	Symbol	Value
Request generation rate	λ	0.4
Number of sources	K	100
Number of servers	S	3
Service area capacity	C	10
Queue size	$Q = C - S$	7
Orbit size	$O = K - S$	97
Encouraging parameter	e	0.5
Impatience rate	η	1
Retrial rate	ν	0.4
Service rate	μ	10
Orbital search probability	p	0.5

5. Discussion of Results

In this section, we discuss the influence of balking, impatience, and service area capacity on the mean response time of the finite-source multi-server retrial queue. If not stated otherwise, the model parameters are chosen according to Tab. 3. All results given in this section are obtained and discussed based on the given parameters and considered to hold for a wide range of similar parameter settings.

5.1. Effect of Increasing the Service Area Capacity

The influence of the request generation rate $\lambda, 0.001 \leq \lambda \leq 2$ (x-axis), and of the service area capacity $C \in \{3, 4, 10\}$ on the mean response time $\overline{T}$ (y-axis) is shown in Fig. 4.

For very small request generation rates ($\lambda < 0.1$), the mean response time $\overline{T}$ is close to the mean service time given by $\mu^{-1} = 0.1$. This is because there is a high probability that an arriving request will find at least one server idle which will be joined immediately. Hence, the time spent in the queue or in orbit is negligible and the queue size $Q = C - S$ does not have tangible effect on the mean response time. Further investigations that are not presented here also show that the effect of varying the encouraging parameter, impatience rate, or retrial rate only has negligible effect.

For high request generation rates ($\lambda > 1$), the system approaches saturation, i.e., the probability that all K requests are located within the retrial queue gets close to 1. Due to the large number of requests located in the service area, the servers are kept busy most of the time. If $Q = 0$ (i.e.,

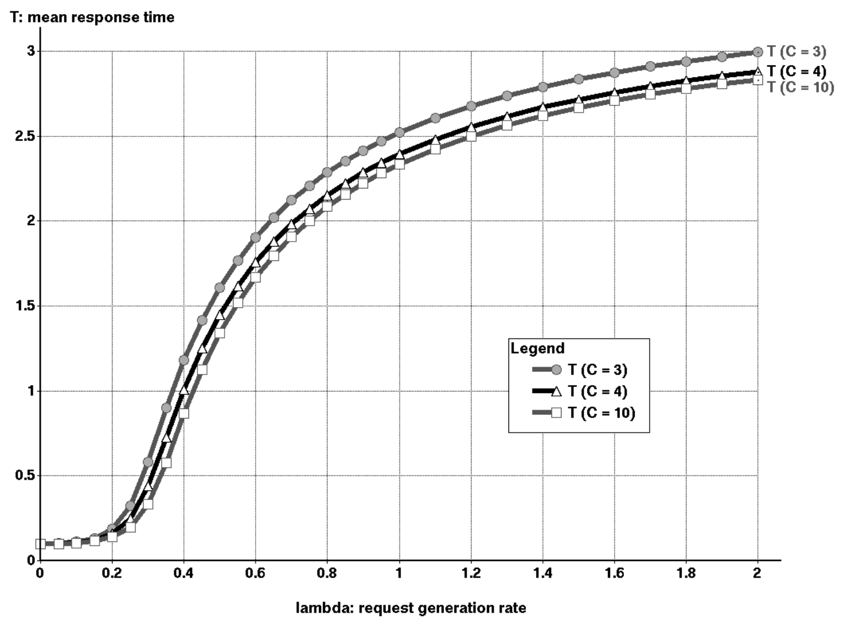

Figure 4: Mean response time $\overline{T}$ over call generation rate λ for different values of the service area capacity C.

$C = Q + S = 3$), a finishing server will not find any request waiting in a queue. However, with probability p it will directly get a job from the orbit, which contains requests with a high probability. Even if no orbital search is done by the server, it will not stay idle for a long time since there is a high number of retrying requests. If $Q = 1$ (i.e., $C = 4$), there is a high probability that a finishing server will find a request in the queue at service completion instant. Due to the large number of orbiting requests, the queue will be refilled quickly, even for moderate values of the impatience rate. However, increasing the queue size to greater values ($C \gg 4$) does not decrease the mean response time significantly, because it does not make a difference whether the requests are waiting in the queue or in the orbit. Hence, also for high values of λ changing the encouraging parameter, impatience rate, or retrial rate will not have a significant effect.

The main influence of the queue size Q on the mean response time $\overline{T}$ can be seen for moderate request generation rates ($0.3 < \lambda < 0.5$). In this case, there is enough load to utilize the queue, but there is only a limited number of orbiting requests to be searched by a server or quickly refilling the queue. The main influence of the retrial rate, encouraging parameter, and

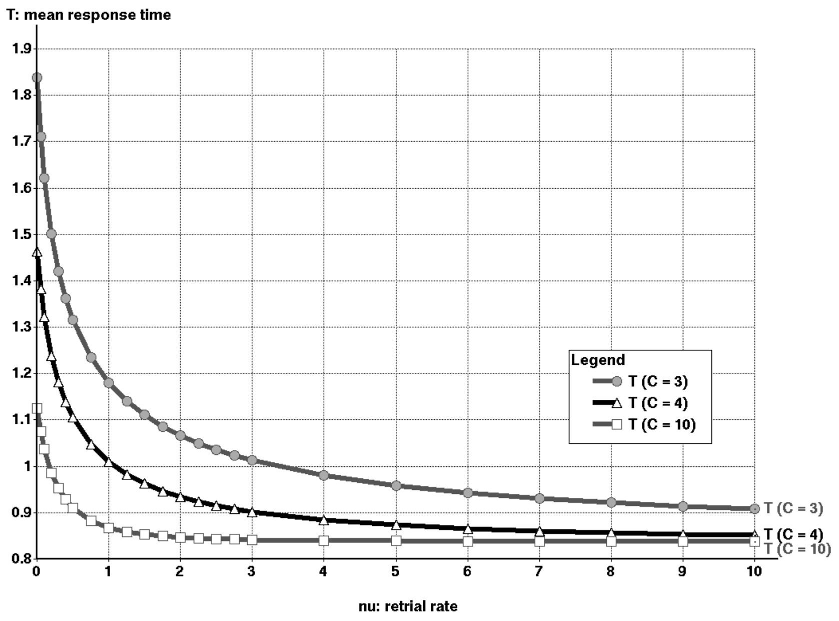

Figure 5: Mean response time $\overline{T}$ over retrial rate ν for different values of the service area capacity C.

impatience rate is expected in this parameter range. This is why the call generation rate is set to $\lambda = 0.4$ in Sections 5.2 to 5.4.

5.2. Effect of Retrial Rate

Figure 5 shows the influence of the retrial rate $\nu, 0.001 \leq \nu \leq 10$ (x-axis), and of the service area capacity $C \in \{3, 4, 10\}$ on the mean response time $\overline{T}$ (y-axis).

Comparing Fig. 5 to Fig. 4, it can be seen that the retrial rate ν has less effect on the mean response time $\overline{T}$ of the investigated retrial queue than the call generation rate λ. Moreover, Fig. 5 shows that the impact of the queue size Q on the mean response time $\overline{T}$ depends on the retrial rate ν.

For relatively high values of the retrial rate ($\nu > 4$), the orbit itself acts similar to a queue. Hence, the actual size of the service area's queue does not have a significant impact. It can be shown that for very high ν all curves approach a minimum response time of $\overline{T}_{\nu\to\infty} \approx 0.84$ for any $C \geq S$. Note that $\overline{T}_{\nu\to\infty}$ is close to the mean response time $\overline{T}_{\text{FCFS}} \approx 0.838$ of a finite-source multi-server $M/M/S/K/K$–FCFS queue which can be calculated algorithmically (finite birth-death process).

Thus, if K, λ, S, and μ are kept constant, the response time $\overline{T}_{\mathrm{FCFS}}$ of the classical finite-source multi-server $M/M/S/K/K$–FCFS queue provides a lower bound for the response time $\overline{T}$ of the finite-source multi-server retrial queue with the search for balking and impatient customers from the orbit, regardless of the balking behavior (configured via e), the size of the queue Q, the impatience rate η, or the orbital search probability p. On the other hand, the mean response time of a finite-source multi-server retrial queue without any queue ($C = S$) provides an upper bound for $\overline{T}$.

For small values of the retrial rate ($\nu < 2$), there is a notable difference in the mean response time $\overline{T}$ when considering different queue sizes. Therefore, in Sections 5.3 and 5.4, the retrial rate is set to $\nu = 0.4$. Note that in most application scenarios, it is not realistic to assume $\nu < \lambda$.

5.3. Effect of Balking

Remember that the term *balking* of arriving requests describes their probabilistic decision to join either the orbit or the service area. The probability $f_e^+(c)$ of joining the service area is given in Eq. (1).

In Fig. 6, the mean response time $\overline{T}$ of the retrial queue under study is shown for different values of the encouraging parameter e and the service area capacity C. According to Eq. (1), $\overline{T}$ is independent of e for $C = S$, since in this case, only two cases exist: 1) there is at least one idle server ($c < S$) and hence the arriving request will join the service area with probability $f_e^+(c) = 1$, and 2) all servers are busy ($c = S$) and hence an arriving request will enter the orbit with probability $f_e^-(c) = 1$. This independence of e is directly reflected in Fig. 6 where for $C = 3$, the graph of $\overline{T}$ takes a constant value.

It is interesting to note that for any queue sizes $Q > 1$, the impact of e on $\overline{T}$ is rather independent of Q. On the other hand, for larger queue sizes, additional buffer space does not have a significant effect on $\overline{T}$. Hence, to improve the system's performance, it is advisable to install a few buffers and to ensure that they are used with a high probability instead of installing a large number of buffers.

5.4. Effect of Impatience

The impact of the impatience rate $\eta, 0.001 \leq \eta \leq 40$ (x-axis), on the mean response time $\overline{T}$ (y-axis) is studied in Fig. 7.

Obviously, if there is no queue ($C = 3$), there is no effect of η on $\overline{T}$. This is clearly reflected in Fig. 7. It can also be seen that increasing the

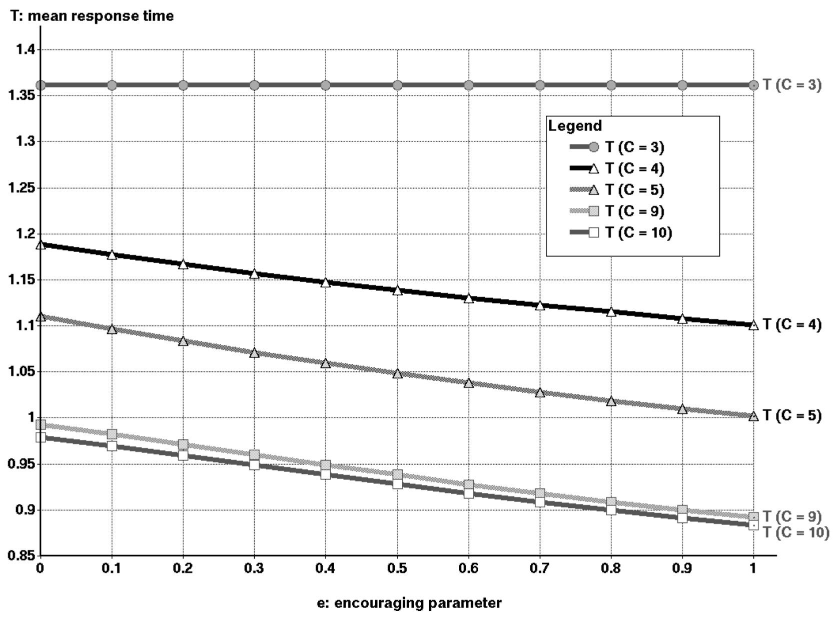

Figure 6: Mean response time $\overline{T}$ over encouraging parameter e for different values of the service area capacity C.

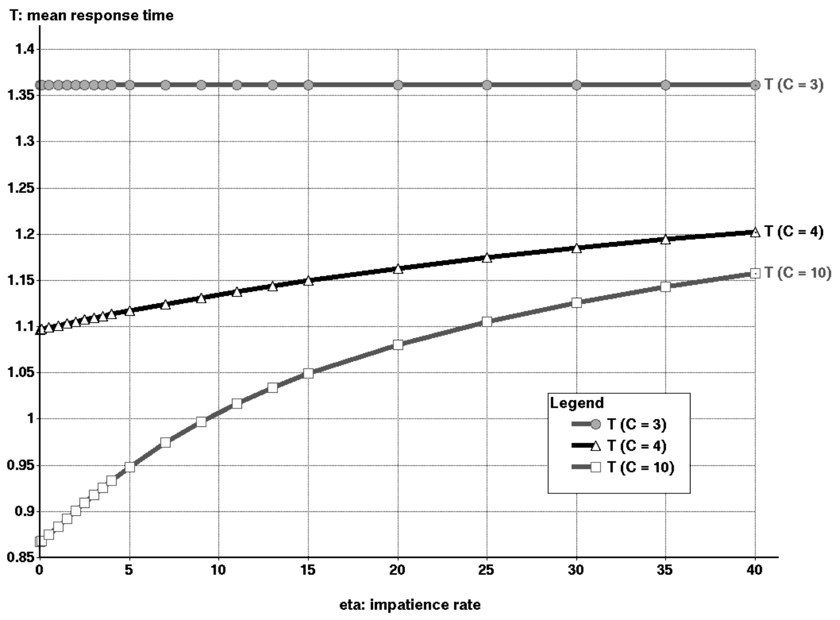

Figure 7: Mean response time $\overline{T}$ over impatience rate η for different values of the service area capacity C.

queue size reduces the mean response time more significantly if the customers impatience is low. By increasing $\eta \to \infty$, it can be shown that the mean response time approaches the one of the case $C = 3$ regardless of the queue length. This is because the available queue will then be used with a very small probability only.

6. Conclusion and Future Work

In this article, we present a novel model that describes the behavior of finite-source retrial queues with search for balking and impatient customers from the orbit. The model is flexible with respect to the configuration of the finite number of customers, their request generation rate, the number of (identical) servers, the service area capacity, the probability of orbital search, as well as the balking and impatience of the customers. A novel method of describing the balking probability as a function of the current queue size provides more modeling flexibility by introducing an encouraging parameter.

Numerical analysis of the model is carried out using the MOSEL-2 tool. The results are discussed in detail and show that in the given scenario it is advisable to provide at least a small queue to the customers. However, it is more important to minimize the balking and impatience of customers than to provide a large queue. Moreover, the steady-state performance measures are shown to be bounded by the performance measures of the classical finite-source multi-server FCFS queue (optimum) and by the classical finite-source retrial queue (worst case).

In future, we plan to generalize the model by considering customer loss, phase-type distributed request generation, impatience, service, and/or retrial times, as well as heterogeneous and unreliable servers. Further research directions include the application of the model to practical applications, the search for closed-form solutions of performance measures applicable to special cases of the model, the search for higher moments and bounds of performance measures, as well as the tool supported analysis of retrial queues comprising transitions with deterministic firing times.

7. Acknowledgments

This research is partially supported by the German-Hungarian Intergovernmental Scientific Cooperation (HAS-DFG, 436 UNG 113/197/0-1), by the Hungarian Scientific Research Fund (OTKA K60698/2006), by the AutoI project (STREP, FP7 Call 1, ICT-2007-1-216404), by the Networks of Excellence EuroFGI (FP6, IST 028022) and EuroNF (FP7, IST 216366).

References

[1] A. S. Alfa and K. S. Isotupa. An M/PH/k retrial queue with finite number of sources. *Computers and Operations Research*, 31:1455–1464, 2004.

[2] A. S. Alfa and W. Li. PCS networks with correlated arrival process and retrial phenomenon. *IEEE Transactions on Wireless Communications*, 1(4):630–637, October 2002.

[3] J. R. Artalejo. Retrial queues with a finite number of sources. *J. Korean Math. Soc.*, 35:503–525, 1998.

[4] J. R. Artalejo. Accessible bibliography on retrial queues. *Mathematical and Computer Modelling*, 30:1–6, 1999.

[5] J. R. Artalejo. A classified bibliography of research on retrial queues: Progress in 1990–1999. *TOP*, 7:187–211, 1999.

[6] J. R. Artalejo and A. Gómez-Corral. *Retrial Queueing Systems: A Computational Approach.* Springer Verlag, 2008.

[7] K. Avrachenkov and U. Yechiali. Retrial networks with finite buffers and their application to internet data traffic. *Probability in the Engineering and Informational Sciences*, 22:519–536, 2008.

[8] K. Begain, G. Bolch, and H. Herold. *Practical Performance Modeling – Application of the MOSEL Language.* Kluwer Academic Publishers, 2001.

[9] B. Beutel. Integration of the Petri net analysator TimeNET into the model analysis environment MOSEL. Diploma thesis, University of Erlangen-Nürnberg, April 2003.

[10] G. Bolch, S. Greiner, H. de Meer, and K. Trivedi. *Queueing Networks and Markov Chains.* John Wiley & Sons, New York, 2nd edition, 2006.

[11] B. D. Choi and Y. Chang. Single server retrial queues with priority calls. *Mathematical and Computer Modelling*, 30:7–32, 1999. Invited paper.

[12] J. W. Cohen. Basic problems of telephone traffic theory and the influence of repeated calls. *Philips Telecommun. Rev.*, 18(2):49–100, 1957.

[13] G. Falin. A survey of retrial queues. *Queueing Systems*, 7(2):127–167, 1990.

[14] G. Falin and J. Templeton. *Retrial Queues.* Chapman & Hall, 1997.

[15] D. Gaver, P. Jacobs, and G. Latouche. Finite birth-and-death models in randomly changing environments. *Adv. Appl. Prob.*, 16:715–731, 1984.

[16] F. A. Haight. Queueing with balking. *Biometrika*, 44(3–4):360–369, 1957.

[17] F. A. Haight. Queueing with reneging. *Metrika*, 2(1):186–197, December 1959.

[18] V. G. Kulkarni and H. M. Liang. *Frontiers in Queueing: Models and Applications in Science and Engineering*, chapter Retrial queues revisited, pages 19–34. CRC Press, 1997.

[19] M. A. Marsan, G. De Carolis, E. Leonardi, R. Lo Cigno, and M. Meo. Efficient estimation of call blocking probabilities in cellular mobile telephony networks with customer retrials. *IEEE Journal on Selected Areas in Communications*, 19(2):332–346, February 2001.

[20] M. F. Neuts and M. F. Ramalhoto. A service model in which the server is required to search for customers. *Journal of Applied Probability*, 21(1):157–166, March 1984.

[21] E. Onur, H. Delic, C. Ersoy, and M. Caglayan. Measurement-based replanning of GSM cell capacities considering retrials, redials and handoffs. In *IEEE International Conference on Communications, 2002 (ICC 2002)*, 2002.

[22] J. Roszik, C. Kim, and J. Sztrik. Retrial queues in the performance modeling of cellular mobile networks using MOSEL. *International Journal of Simulation: Systems, Science and Technology*, 6:38–47, 2005.

[23] Y. W. Shin and T. S. Choo. M/M/s queue with impatient customers and retrials. *Applied Mathematical Modelling*, In Press, Corrected Proof:–, 2008.

[24] B. Sivakumar. A perishable inventory system with retrial demands and a finite population. *Computational and Applied Mathematics*, 224(1):29–38, February 2008.

[25] P. Tran-Gia and M. Mandjes. Modeling of customer retrial phenomenon in cellular mobile networks. *IEEE Journal of Selected Areas in Communications*, 15:1406–1414, 1997.

[26] P. Wüchner, H. de Meer, J. Barner, and G. Bolch. A brief introduction to MOSEL-2. In R. German and A. Heindl, editors, *Proc. of MMB 2006 Conference*. GI/ITG/MMB, University of Erlangen, VDE Verlag, 2006.

[27] P. Wüchner, J. Sztrik, and H. de Meer. Homogeneous finite-source retrial queues with search of customers from the orbit. In *Proc. of 14th GI/ITG Conference on Measurement, Modelling and Evaluation of Computer and Communication Systems (MMB 2008)*, Dortmund, Germany, March 2008.

[28] P. Wüchner, J. Sztrik, and H. de Meer. The impact of retrials on the performance of self-organizing systems. *Praxis der Informationsverarbeitung und Kommunikation (PIK)*, 31(1):29–33, March 2008.

[29] T. Yang and J. G. C. Templeton. A survey on retrial queues. *Queueing Systems*, 2:201–233, 1987.