Performance Modeling of Finite-Source Retrial Queueing Systems with Collisions and Non-reliable Server Using MOSEL

Tamás Bérczes[1(✉)], János Sztrik[1], Ádám Tóth[1], and Anatoly Nazarov[2]

[1] Faculty of Informatics, University of Debrecen, Debrecen, Hungary
berczes.tamas@inf.unideb.hu
[2] Tomsk State University, Tomsk, Russia

Abstract. In this paper we investigate a single-server retrial queueing system with collision of the customer and an unreliable server. If a customer finds the server idle, he enters into service immediately. The service times are independent exponentially distributed random variables. During the service time the source cannot generate a new primary call. Otherwise, if the server is busy, an arriving (primary or repeated) customer involves into collision with customer under service and they both moves into the orbit. The retrial time of requests are exponentially distributed. We assume that the server is unreliable and could be break down. When the server is interrupted, the call being served just before server interruption goes to the orbit. Our interest is to give the main steady-state performance measures of the system computed by the help of the MOSEL tool. Several Figures illustrate the effect of input parameters on the mean response time.

Keywords: Closed queuing system · Finite-source queuing system · Retrial queue · Collision · Unreliable server

1 Introduction

The performance analysis of computing and communicating systems has always been an important subject of computer science. The goal of this analysis is to make predictions about the quantitative behavior of a system under varying conditions, e.g., the expected response time of a server under varying numbers of service requests, the average utilization of a communication channel under varying numbers of communication requests, and so on.

Retrial queueing systems (RQS) are characterized by the feature that arriving customers finding all the servers busy upon arrival are obliged to leave the service area and repeat their requests for service after some random time [1–3]. This feature plays an important role in modeling many problems in telephone switching systems, telecommunication networks, computer networks, call centers, etc. The main difference between retrial queues and classic queues is that

V.M. Vishnevskiy et al. (Eds.): DCCN 2017, CCIS 700, pp. 248–258, 2017.
DOI: 10.1007/978-3-319-66836-9_21

the classic queueing theory does not take the actually existed retrial customers into account. It assumes these retrial customers are either lost due to congestion or delayed in the waiting line (if any).

Since in practice some components of the systems are subject to random breakdowns it is of basic importance to study reliability of retrial queues with server breakdowns and repairs. Finite-source retrial queues with unreliable server have been investigated in several recent papers, for example, [4–10].

Many times when different data is transmitted and there are only a limited number of available free channels may cause a conflict. This may in many cases result in collisions that lead to data loss. Recent results on retrial queues with collisions can be found in, for example [11–15].

The aim of the present paper is to investigate a single-server retrial queueing system with collision of the customer and an unreliable server.

Because of the fact, that the state space of the describing Markov chain is very large, it is rather difficult to calculate the system measures in the traditional way of writing down and solving the underlying steady-state equations. To simplify this procedure we used the software tool MOSEL (Modeling, Specification and Evaluation Language), see [16], to formulate the model and to obtain the performance measures. The organization of the paper is as follows. Section 2 contains the corresponding queueing model with components. In Sect. 3, we present some numerical examples.

2 System Model

Let us consider (Fig. 1) a closed retrial queuing system of type M/M/1//N with collision of the customers and an unreliable server. The number of sources is N and each of them can generate a primary request with rate λ/N. If a customer finds the server idle, he enters into service immediately. The service times are independent exponentially distributed random variables with parameter μ. During the service time the source cannot generate a new primary call. Otherwise,

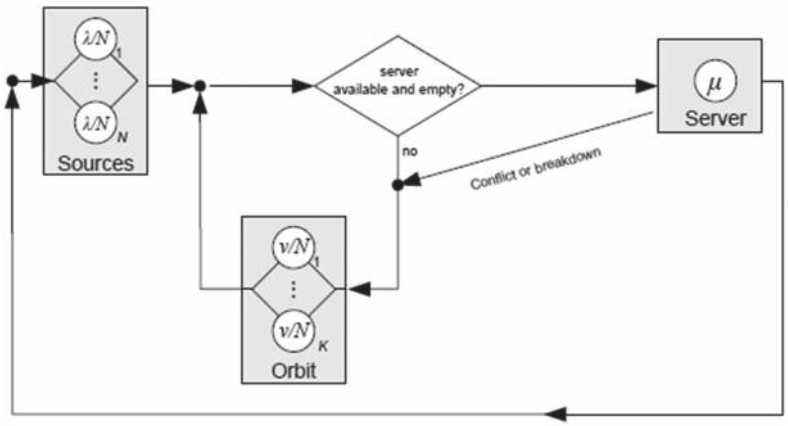

Fig. 1. System

if the server is busy, an arriving (primary or repeated) customer involves into collision with customer under service and they both moves into the orbit. The retrial time of requests are exponentially distribution with rate σ/N. We assume that the server is unreliable and could be break down. The lifetime is supposed to be exponentially distributed with failure rate γ_0 if the server is idle and with rate γ_1 if it is busy. When the server breaks down, it is immediately sent for repair and the recovery time is assumed to be exponentially distributed with rate γ_2. If the server fails in busy state, the interrupted request goes to the orbit.
The server can be in three states:

- *idle state:* If the server is available and it can start serving the arriving requests.
- *busy state:* If the server is available and busy.
- *failed state:* If the server is in failed state, it couldn't start serving any arriving requests until it wouldn't be repaired.

All random variables involved in the model construction are assumed to be independent of each other.

We introduce the following notations (see the summary of the model parameters in Table 1):

- $k(t)$ is the number of active source at time t,
- $o(t)$ is the number of jobs in the orbit at time t.
- $y(t) = 0$ if the server is up and $y(t) = 1$ if the server is failed at time t
- $c(t) = 0$ if the server is idle and $c(t) = 1$ if the server is busy at time t

It is ease to see that:

$$k(t) = \begin{cases} N - o(t), & y(t) = 1 \text{or} c(t) = 0 \\ N - o(t) - 1, & c(t) = 1 \end{cases}.$$

To maintain theoretical manageability, the distributions of inter-event times (i.e., request generation time, service time, retrial time, available state time, sleeping state time, failed state time) presented in the network are by assumption exponential and totally independent. The state of the network at a time t corresponds to a Continuous Time Markov Chain (CTMC) with 3 dimensions:

$$X(t) = (y(t); c(t); o(t)) \tag{1}$$

The steady-state distributions are denoted by

$$P(y, c, o) = \lim_{t \to \infty} P(y(t) = y, c(t) = c, o(t) = o) \tag{2}$$

Note, that the state space of this Continuous Time Markovian Chain is finite, so the steady-state probabilities surely exist. For computing the steady-state probabilities and the system characteristics, we use the MOSEL software tool in this paper. These computations are described in eg. [17,18].

Table 1. Overview of model parameters

Parameter	Maximum	Value at t
Number of sources	N (population size)	$k(t)$
Generation rate of active sources		λ/N
Total gen. rate	$\lambda_1 N$	$\lambda_1 k(t)$
Service rate		μ
Number of busy servers	1 (number of servers)	$c(t)$
Cust. in service area	N	$c(t) + o(t)$
Requests in Orbit	N (orbit size)	$o(t)$
Retrial rate		σ/N
Server's failure rate (idle case)		γ_0
Server's failure rate (busy case)		γ_1
Server's repair rate		γ_2

When we have calculated the distributions defined above, the most important steady-state system characteristics can be obtained in the following way:

– *Utilization of the server*

$$U_S = \sum_{o=0}^{N} P(0, 1, o) \tag{3}$$

– *Utilization of the repairman*

$$U_r = \sum_{c=0}^{1} \sum_{o=0}^{N} P(1, c, o) \tag{4}$$

– *Availability of the server*

$$A_S = \sum_{c=0}^{1} \sum_{o=0}^{N} P(0, c, o) = 1 - U_r \tag{5}$$

– *Average number of jobs in the orbit*

$$\overline{O} = E(o(t)) = \sum_{y=0}^{1} \sum_{c=0}^{1} \sum_{o=0}^{N} oP(y, c, o) \tag{6}$$

– *Average number of jobs in the server*

$$\overline{C} = E(c(t)) = \sum_{o=0}^{N-1} oP(0, 1, o) \tag{7}$$

– *Average number of jobs in the network*

$$\overline{M} = \overline{O} + \overline{C} = \sum_{y=0}^{1} \sum_{c=0}^{1} \sum_{o=0}^{N} oP(y,c,o) + \sum_{o=0}^{N-1} oP(0,1,o) \tag{8}$$

– *Average number of active sources*

$$\overline{\Lambda} = N - M = N - \sum_{y=0}^{1} \sum_{c=0}^{1} \sum_{o=0}^{N} oP(y,c,o) - \sum_{o=0}^{N-1} oP(0,1,o) \tag{9}$$

– *Average generation rate of sources:*

$$\overline{\lambda} = \lambda / N \overline{\Lambda_1} \tag{10}$$

– *Average waiting time in orbit:*

$$ET_o = \frac{\overline{O}}{\overline{\lambda}} \tag{11}$$

– *Average waiting time in orbit:*

$$ET = \frac{\overline{M}}{\overline{\lambda}} \tag{12}$$

3 Numerical Results

The Table 2 shows the input parameters of the investigated Figures.

Figure 2 shows the steady-state distribution of the three investigated cases. In this figure we can see also the effect of the breakdown of the Server. We can see that the mean number of customers increases as the breakdown intensity are getting larger. From the shape of the curves it is clearly visible that the steady-state distribution of the cases are normally distributed.

Figures 3 and 4 shows the mean response time as a function of the customer generation rate. As we see the mean response time will be greater as we increase the generation rate, but after after λ/N is greater than 1.5 the mean response time starts to decrease.

On Fig. 5 the utilization of the server is displayed as a function of λ. As we see the Utilisation will be greater as we increase the λ/N generation rate. We can understand these property if we take into account that a higher generation rate result more requests in the System.

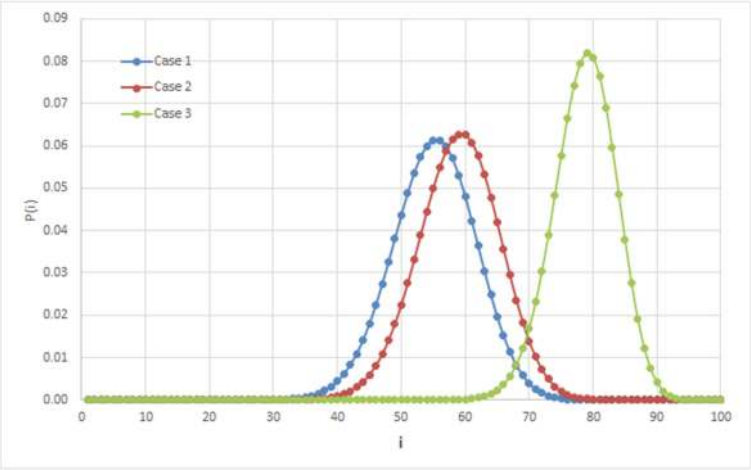

Fig. 2. Steady-state distributions

Table 2. Numerical values of model parameters

Case	N	λ/N	γ_0	γ_1	γ_2	σ/N
Fig. 2 case 1	100	0.01	0.01	0.01	1	0.1
Fig. 2 case 2	100	0.01	0.1	0.1	1	0.1
Fig. 2 case 3	100	0.01	1	1	1	0.1
Fig. 3	100	0.03 − 8.1	0.01	0.01	1	0.1
Fig. 4	100	0.03 − 8.1	0.1	0.1	1	0.1
Fig. 5	100	0.03 − 8.1	0.1	0.1	1	0.1
Fig. 6	100	3	0.01 − 1.01	γ_0	1	0.1
Fig. 7	100	3	0.01 − 1.01	γ_0	1	0.1
Fig. 8	100	3	0.2	0.2	γ_2	0.1
Fig. 9	100	3	0.2	0.2	γ_0	0.1

On Fig. 6 one can see the effect of the Server's failure rate for the Utilisation. As we see the Utilisation will be smaller as we increase the failure rate.

On Fig. 7 the effect of Server's breakdown with mean response time is displayed. The Mean Response Time is increasing steadily by increasing the failure of the server. On the one hand, this is due to the fact that if the server is breaks down in the busy state, the request is placed in the orbit. On the other hand, the defective server can not accept new requests. Therefore, the increase in the intensity of failure results in an increase in response times.

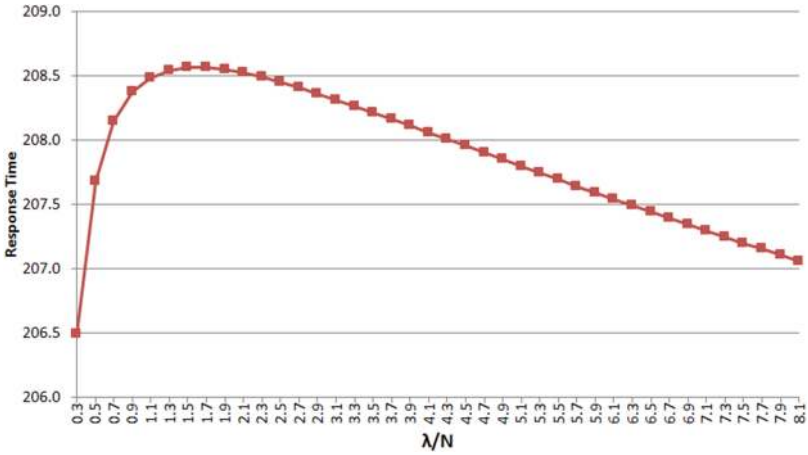

Fig. 3. vs λ/N, $\gamma_0 = \gamma_1 = 0.01$

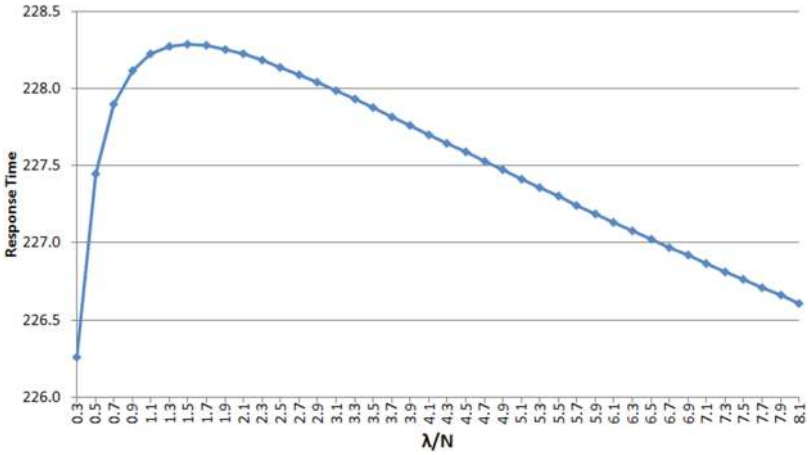

Fig. 4. vs λ/N, $\gamma_0 = \gamma_1 = 0.1$

On Fig. 9 we can see the effect of the Server's repair rate for the Response Time. As we see the Response Time will be smaller as we increase the repair rate. This is because a failed server can not receive or process requests. Thus, the higher repair intensity results in shorter response times.

On Fig. 8 we investigate the effect of the Server's repair rate on Utilisation of the server. As we see the Utilisation will be higher as we increase the repair rate.

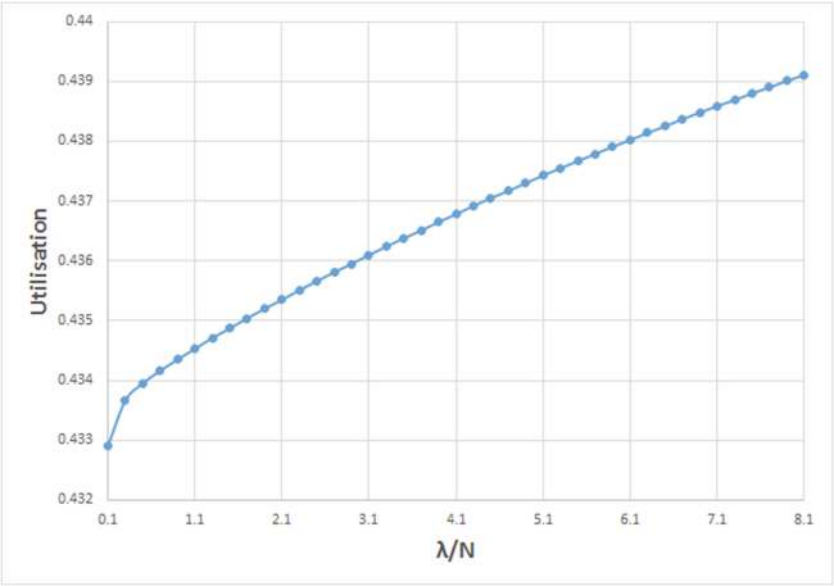

Fig. 5. Utilisation vs λ/N, $\gamma_0 = \gamma_1 = 0.1$, $\gamma_2 == 1$

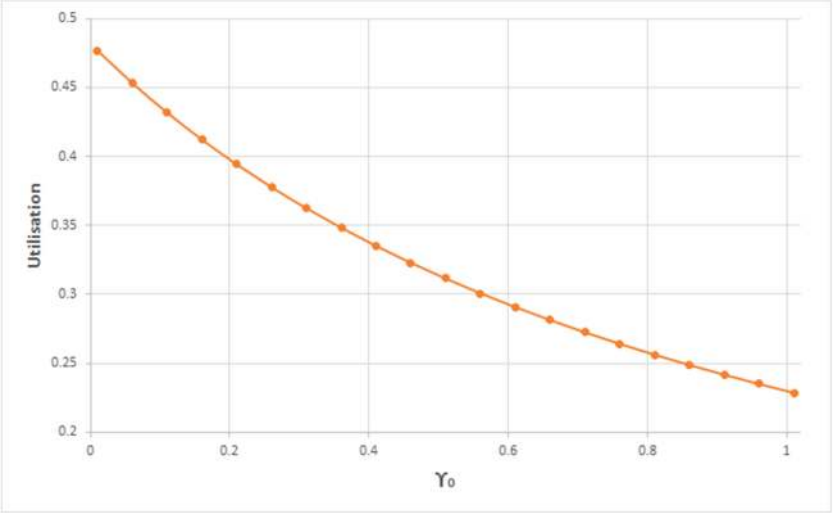

Fig. 6. Utilisation vs γ_0, $\gamma_0 = \gamma_1$, $\lambda/N = 3$

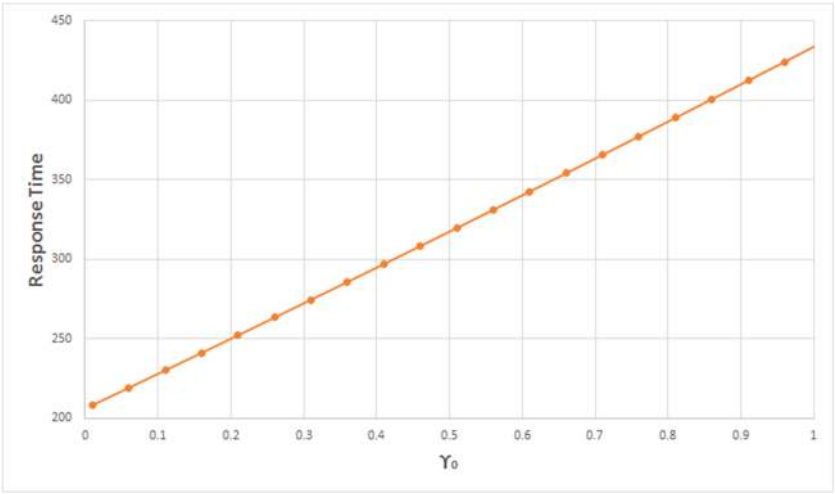

Fig. 7. Mean Response Time vs γ_0, $\gamma_0 = \gamma_1$, $\lambda/N = 3$

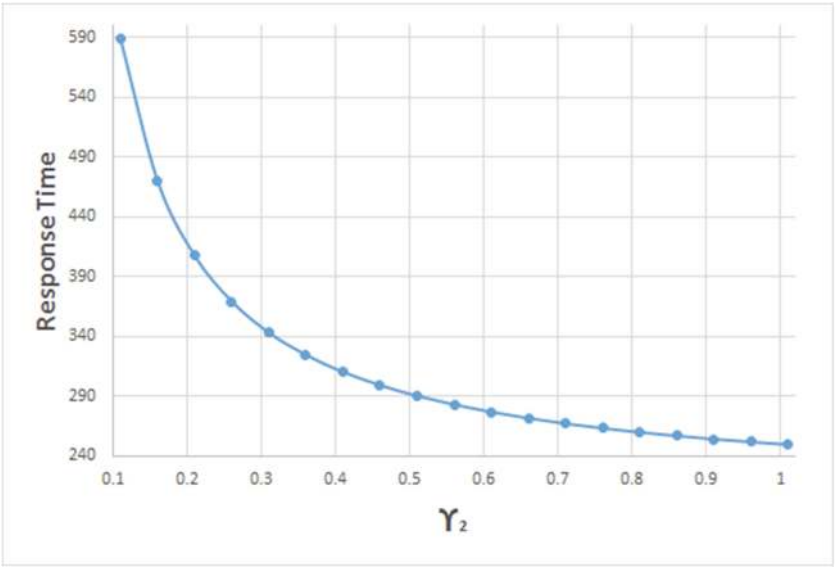

Fig. 8. Response Time vs γ_2, $\gamma_0 = \gamma_1 = 0.2$, $\lambda/N = 3$

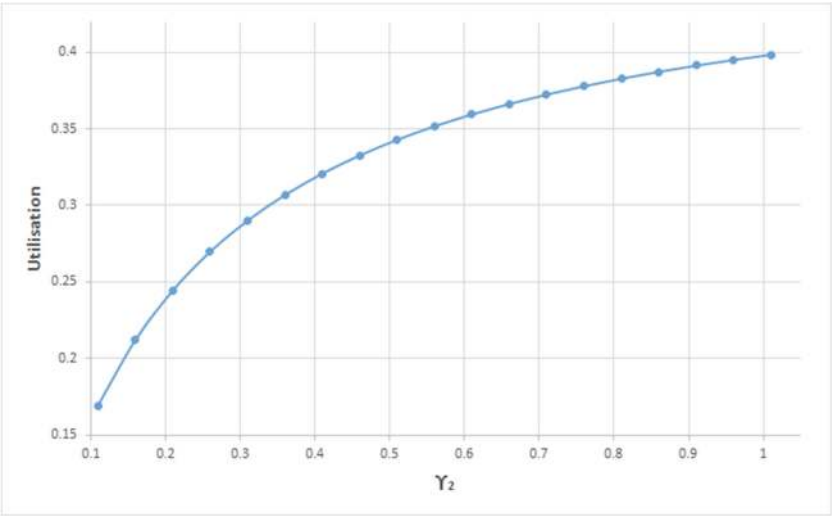

Fig. 9. Utilisation vs γ_2, $\gamma_0 = \gamma_1 = 0.2$, $\lambda/N = 3$

4 Conclusions

In this paper we investigated a single-server retrial queueing system with collision of the customer and an unreliable server. The MOSEL tool was used to formulate and solve the problem, and the main performance and reliability measures were derived and analyzed graphically. We have investigated also the impact of server's failure rate and the server's repair rate on the performance of the system. To our best knowledge, this is the first proposal for the use of single-server retrial queueing system with collision of the customer and an unreliable server.

Acknowledgments. The publication was financially supported by the Ministry of Education and Science of the Russian Federation (the Agreement number 02.a03.21.0008).

The present work of Ádám Tóth and János Sztik was supported by the Austro-Hungarian Cooperation Grant No 96öu8, 2017.

The work of Tamás Bérczes was supported in part by the project EFOP-3.6.2-16-2017-00015 supported by the European Union, co-financed by the European Social Fund.

References

1. Artalejo, J.R., Gómez-Corral, A.: Retrial queueing systems. A computational approach. Springer, Berlin (2008). p. xiii + 318. ISBN 978-3-540-78724-2/hbk;978-3-642-09748-5/pbk; 978-3-540-78725-9/ebook
2. Gómez-Corral, A., Phung-Duc, T.: Retrial queues and related models. Ann. Oper. Res. **247**(1), 1–2 (2016). http://dx.doi.org/10.1007/s10479-016-2305-2

3. Kim, J., Kim, B.: A survey of retrial queueing systems. Ann. Oper. Res. **247**(1), 3–36 (2016). http://dx.doi.org/10.1007/s10479-015-2038-7
4. Almási, B., Roszik, J., Sztrik, J.: Homogeneous finite-source retrial queues with server subject to breakdowns and repairs. Math. Comput. Model. **42**(5–6), 673–682 (2005)
5. Dragieva, V.I.: Number of retrials in a finite source retrial queue with unreliable server. Asia-Pac. J. Oper. Res. **31**(2), 23 (2014)
6. Gharbi, N., Dutheillet, C.: An algorithmic approach for analysis of finite-source retrial systems with unreliable servers. Comput. Math. Appl. **62**(6), 2535–2546 (2011)
7. Roszik, J.: Homogeneous finite-source retrial queues with server and sources subject to breakdowns and repairs. Ann. Univ. Sci. Bp. Sect. Comput. **23**, 213–227 (2004). Rolando Eötvös
8. Wang, J., Zhao, L., Zhang, F.: Performance analysis of the finite source retrial queue with server breakdowns and repairs. In: Proceedings of the 5th International Conference on Queueing Theory and Network Applications, pp. 169–176. ACM (2010)
9. Wang, J., Zhao, L., Zhang, F.: Analysis of the finite source retrial queues with server breakdowns and repairs. J. Ind. Manag. Optim. **7**(3), 655–676 (2011)
10. Zhang, F., Wang, J.: Performance analysis of the retrial queues with finite number of sources and service interruptions. J. Korean Stat. Soc. **42**(1), 117–131 (2013)
11. Ali, A.-A., Wei, S.: Modeling of coupled collision and congestion in finite source wireless access systems. In: Wireless Communications and Networking Conference (WCNC), pp. 1113–1118. IEEE (2015)
12. Balsamo, S., Rossi, G.-L.D., Marin, A.: Modelling retrial-upon-conflict systems with product-form stochastic petri nets. In: Dudin, A., De Turck, K. (eds.) ASMTA 2013. LNCS, vol. 7984, pp. 52–66. Springer, Heidelberg (2013). doi:10.1007/978-3-642-39408-9_5
13. Choi, B.D., Shin, Y.W., Ahn, W.C.: Retrial queues with collision arising from unslotted CSMA/CD protocol. Queueing Syst. **11**(4), 335–356 (1992)
14. Gómez-Corral, A.: On the applicability of the number of collisions in p-persistent CSMA/CD protocols. Comput. Oper. Res. **37**(7), 1199–1211 (2010)
15. Kim, J.-S.: Retrial queueing system with collision and impatience. Commun. Korean Math. Soc. **25**(4), 647–653 (2010)
16. Begain, K., Bolch, G., Herold, H.: Practical Performance Modeling, Application of the MOSEL Language. Kluwer Academic Publisher, Boston (2001)
17. Bolch, G., Greiner, S., de Meer, H., Trivedi, K.: Queueing Networks and Markov Chains, 2nd edn. Wiley, Amsterdam (2006). ISBN 0-471-56525-3
18. Wüchner, P., Sztrik, J., de Meer, H.: Modeling wireless sensor networks using finite-source retrial queues with unreliable orbit. In: Hummel, K.A., Hlavacs, H., Gansterer, W. (eds.) PERFORM 2010. LNCS, vol. 6821, pp. 73–86. Springer, Heidelberg (2011). doi:10.1007/978-3-642-25575-5_7