Using GSPNs for Performance Evaluation of Networks with Repeated Calls and Different Vacation Policies

Nawel Gharbi
Computer Science Department
University of Sciences and Technology, USTHB
Algiers, Algeria
Email: ngharbi@wissal.dz

***Abstract*—This paper deals with retrial systems where servers are subject to random vacations. So far, these systems were analyzed only by queueing theory and almost works assuming that the service station consists of one server and the customers source is infinite. In this paper, we give a detailed performance analysis of finite-source multiserver networks with repeated calls of blocked customers and multiple or single vacations of servers or all station, using Generalized Stochastic Petri nets. We show how this high level stochastic model allows us to cope with the complexity of such networks involving the simultaneous presence of retrials and vacations, and how stationary performance indices can be expressed as a function of Petri net elements.**

Keywords-Repeated calls; Finite-source; Vacation policies; Generalized Stochastic Petri nets; Modeling and Performance measures.

I. Introduction

Models with repeated calls describe operation of many computer networks and telecommunication systems, e.g., call centers, cellular mobile networks [1][2][3][4] and wireless sensor networks [5]. Systems with repeated attempts are characterized by the following feature: When an arriving customer finds all servers (resources) busy or unavailable, is not put in a queue, but joins a virtual pool of blocked customers called orbit, and will repeat the request to try again to reach the servers after a random delay. Significant references reveal the non-negligible impact of repeated calls, which arise due to a blocking in a system with limited capacity resources or due to impatience of users. There has been a rapid growth in the literature on the queueing systems with repeated attempts (also called retrial queues). For a recent summary of the fundamental methods, results and applications on this topic, the reader is referred to [6][7] and [8].

In this paper, we consider multiserver retrial systems in which each server sometimes takes a vacation, i.e., becomes unavailable to the primary and repeated calls for a random period of time. These vacation periods are usually introduced in order to exploit the idle time of the servers for other secondary jobs as: servicing customers of another system, inspection tasks and preventive maintenance actions which are mainly doing to prevent the risk of failure, to preserve the sanity of the system, to provide a high reliability and to improve the quality of service. Similarly, the servers breakdowns which may occur randomly, and the repair periods, may be regarded as servers vacations.

A wide class of policies for governing the vacation mechanism, have been discussed in the literature, namely the multiple vacation policy and the single vacation policy. Other studies have considered synchronous vacations of some servers or all the station servers (station vacation). On the other hand, multi-server vacation models were mainly studied in the past decade. Zhang et al. [9][10] studied the multi-server models with either single vacation or multiple vacations. Later, Lin et al. [11] analyzed the multi-server model with working vacations. Ke et al. [12] studied the optimal threshold policies in a finite buffer multi-server vacation model with unreliable servers. Recently, Ke et al. [13] consider a multi-server queueing system with multi-threshold vacation policy and servers breakdowns. Excellent surveys on the vacation models have been reported by Doshi [14], Takagi [15], Tian et al. [16], and recently, by Ke et al. [17].

The main reason for the growing interests in multiple-server vacation models is because they can realistically represent some service/manufacturing systems and computer/telecommunication networks. However, all these works on multi-server vacation queueing models, assume that the customers source is infinite and do not take into account the repeated calls of blocked customers.

In retrial systems with vacations, customers who arrive while all servers are busy or on vacation, have to join the orbit to repeat their call after a random period. Thus, there is a natural interest in the study of this class of models, which has been used in concrete applications as digital cellular mobile networks [18], local area networks with nonpersistent CSMA/CD protocols [19], with star topology [20] and so on. However, almost works combining retrial and vacation phenomenon, assume that the service station consists of one single server and the customers source is infinite [20][21][22]. On the other hand, in all the works cited above, the retrial systems with vacations are analyzed only by the queueing theory.

In this paper, we propose the applicability of Generalized Stochastic Petri nets formalism (GSPNs) for modeling and performance evaluation of networks with repeated calls of blocked customers and servers vacations. To this end, we consider different vacation policies, namely the single and multiple vacations of servers or all the service station.

The paper is organized as follows: First, we describe the systems under study. In Section 3, we present the GSPN models describing multiserver retrial systems with station and server vacations mechanisms and under multiple and single vacation policy. Performance indices are given in Section 4. Next, several numerical examples are presented with some comments and discussions. Finally, we give a conclusion.

II. Description of retrial networks with different vacation policies

In the analysis of retrial systems with vacations, it is usually assumed that the customers source is infinite. However, in many practical situations, it is very important to take into account the fact that the rate of generation of new primary calls decreases as the number of customers in the system increases. This can be done with the help of finite-source retrial models where each customer generates its own flow of primary demands.

In this paper, we consider retrial systems with finite source (population), that is, we assume that a finite number K of potential customers generate the so called quasi-random input of primary calls with rate λ. Each customer can be in three states: generating a primary call (free), sending repeated calls (in orbit) or under service by one of the servers.

If a customer is free at time t, it can generate a primary request for service in any interval $(t, t+dt)$ with probability $(K-n)\lambda dt + o(dt)$ as $dt \to 0$, where n is the number of customers in the system. Each customer requires to be served by one and only one server.

The service station consists of c $(c \geq 1)$ homogeneous and parallel servers. Each server can be idle, busy or on vacation. If one of the servers is idle at the moment of the arrival of a call, then the service starts. The requests are assigned to the free servers randomly and without any priority order. The service times are independent, identic and exponentially distributed with rate μ. After service, the customer becomes free, so it can generate a new primary call, and the server becomes idle.

We consider the two vacation mechanisms: *server vacation* and *station vacation*. For the first one, which is encountered even more often in practice, each server is an independent working unit, and it can take its own vacation independently of other servers states. In the model with station vacation mechanism, ALL the servers take vacations simultaneously. That is, whenever the system is empty, all the station leaves the system for a vacation, and returns when the vacation is completed. So, station vacation is group vacation for all servers. This occurs in practice, for example, when a system consists of several interconnected machines that are inseparable, or when all the machines are run by a single operator. In such situations, the whole station has to be treated as a single entity for vacation. Hence, if the system (or the operator who runs the system) is used for a secondary task when it becomes empty (or available), all the servers (the operator) will then be utilized to perform a secondary task. During this amount of time, the servers are unavailable to serve any primary or repeated call and this is equivalent to taking a station vacation.

The *exhaustive service discipline* is considered here. That is, each free server (or all station) can take a vacation only if the system is empty at either a service completion or at the end of a vacation, and only at these epochs. On the other hand, upon completing a vacation, the server returns to the idle state and starts to serve customers, if any, till the system becomes empty. Otherwise, if the server (or the station) at the moment of returning from vacation, finds the system empty, it takes one of the two actions:

- Under the *multiple vacation policy*, the server (station) shall leave immediately for another vacation and continues in this manner until he finds at least one customer (not being served) in the system upon returning from a vacation.
- Under the *single vacation policy*, the server (station) should wait until serving one call at least before commencing another vacation.

The vacation times of all servers (or station) are assumed to be independent and exponentially distributed with rate θ.

At the moment of the arrival of a call, if all the servers are busy or on vacation, the customer joins the orbit to repeat his demand after an exponential time with parameter ν.

As usual, we assume that the interarrival periods, service times, vacation times and retrial times are mutually independent.

III. GSPN models of multiserver retrial systems with vacations

In this section, we present our approach for modeling finite-source multiserver retrial systems with station and server vacations, under multiple and single vacation policies using the generalized stochastic Petri nets model.

A GSPN is a directed graph that consists of places (drawn as circles), timed transitions (drawn as rectangles) which describe the execution of time consuming activities and immediate transitions (drawn as thin bars) that model actions whose duration is negligible, with respect to the time scale of the problem. This class of transitions has priority over timed transitions and fire in zero time once they are enabled.

Formally, a GSPN [24] is an eight-tuple $(P, T, W^-, W^+, W^h, \pi, M_0, \theta)$ where :

- $P = \{P_1, P_2, ..., P_n\}$ is the set of places;

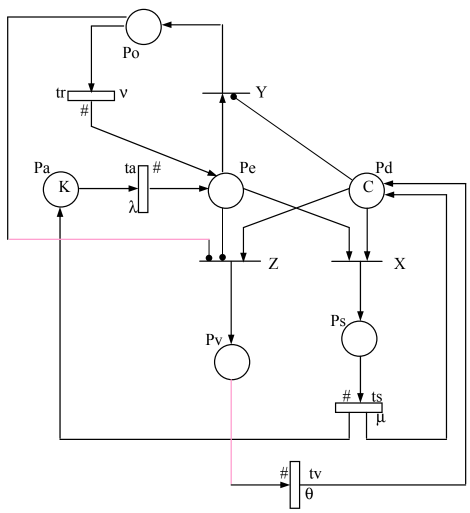

Figure 1. GSPN model of retrial systems with multiple vacations of servers

- $T = \{t_1, t_2, ..., t_m\}$ is the set of timed and immediate transitions;
- $W^-, W^+, W^h : P * T \to IN$ are the input, output and inhibitor functions respectively;
- $\pi : T \to IN$ is the priority function;
- $M_0 : P \to IN$ is the initial marking which describes the initial state of the system;
- $\theta : T \to IR^+$ is a function that associates rates of negative exponential distribution to timed transitions and weights to immediate transitions.

A. *Retrial systems with multiple vacations of servers*

This model is used for describing many practical problems where servers take individual vacations. This means, whenever a server completes servicing and there are no more requests in the system, it takes a vacation independently of other servers states. On the other hand, multiple vacations policy means that at the end of a vacation period, if the orbit is empty and there is no primary or repeated arrival, the server takes immediately another vacation. The process continues until the server upon returning finds any customer in the system.

Fig. 1 shows the GSPN model describing the above system.

- The place P_a contains the free customers;
- The place P_e contains the primary or repeated (returning) calls ready for service;
- The place P_d contains the free (available) servers;
- The place P_o represents the orbit;
- The place P_s contains customers in service (or busy servers);
- The place P_v contains the servers that are on vacation.

The initial marking of the net is:
$M_0 = \{M(P_a), M(P_e), M(P_d), M(P_o), M(P_s), M(P_v)\} = \{K, 0, c, 0, 0, 0\}$, which represents the fact that all customers are initially free, the c servers are available, no server is on vacation and the orbit is empty. Hence, at time $t = 0$, all servers take a vacation simultaneously. So, this initial state is vanishing and equivalent to the tangible state $(K, 0, 0, 0, 0, c)$.

- The firing of transition t_a indicates the arrival of a primary request generated by a free customer. It has an *infinite servers semantics*, which is represented by the symbol $\#$ placed next to transition. This means that the firing rate of t_a is marking dependent and equals $\lambda.ED(t_a, m)$ where $ED(t_a, m)$ is the enabling degree of the transition t_a in the marking m. Hence, all potential customers are able to generate requests for service.
- At the arrival of a primary or repeated request to the place P_e, if P_d contains at least one available server, the immediate transition X fires and one token is deposited in P_s, which represents the begin of the service. Otherwise, if all servers are busy or on vacation (ie. no token in P_d), the immediate transition Y fires and a token will be deposited in the place P_o. So, the customer joins the orbit.
- When the transition t_r fires, the customer in orbit tries again for service, so the system receives a repeated request.
- The firing of the immediate transition Z represents the event that an idle server is commencing a vacation since there is no call left to be served. This represents the exhaustive service discipline.
- The firing of transition t_v represents the end of the vacation time. Hence, the server is returned to the available state.
- When the timed transition t_s fires, the customer under service returns to the idle state and the server becomes ready to serve another customer.
- The service semantics of the timed transitions t_s and t_v are *infinite servers semantics*, because the c servers are parallel. So, several servers can be in service or on vacation at the same time. Similarly, the transition t_r is marking dependent because the customers in orbit are independent and can generate repeated calls simultaneously.

B. *Retrial systems with multiple vacations of the station*

In this model, as soon as the system is empty, all the servers become idle, and consequently the station takes a vacation. As one may expect, this situation appears to be more complicated that the previous one. In fact, it is more simple, because all servers take a vacation simultaneously and return to the system at the same time also. Hence, the GSPN modeling this system with multiple vacations of the station, is the same model as the one given in Fig. 1, in which the multiplicity of the arc connecting the place P_d to

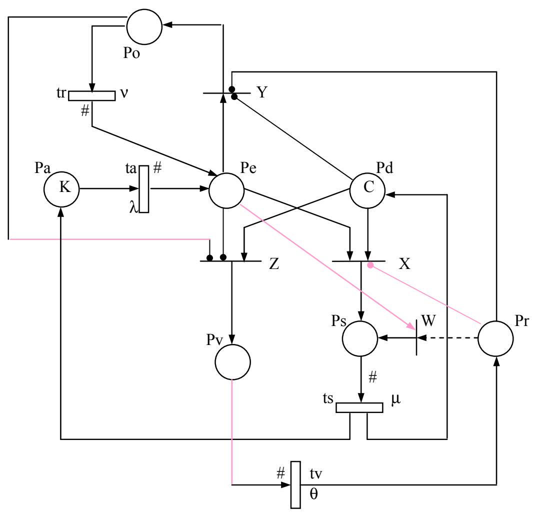

Figure 2. GSPN model of retrial systems with single vacations of servers

transition Z and transition t_v to place P_d equals c (rather than 1), because the c servers of the station take a vacation together. So, if the place P_d contains c idle servers, the orbit (P_o) is empty and there is no arrival to the place P_e, the immediate transition Z fires, which represents the begin of the station vacation time. At the end of this period (after a mean delay equals $1/\theta$), c tokens corresponding to the c servers of the station will be deposited in P_d.

C. Retrial systems with single vacations of servers

This model corresponds to systems where each server is an independent working unit. The single vacation policy means that at the end of a vacation period, even if the system is empty, the server is obliged to wait until serving one call at least, before commencing another vacation.

Fig. 2 shows the GSPN model describing the above system.

In the previous models with multiple vacations, the place P_d contains all the free servers. Hence, at the end of a service or vacation period, the server returns to the idle state represented by the place P_d. However, in the model with single vacations given in Fig. 2, at a service completion, the server joins the place P_d which contains the servers having served at least one call since the last vacation period. So, they can serve other calls if any (firing of transition X). Otherwise, they can take a vacation after the firing of the immediate transition Z. However, at the end of a vacation period, the server joins the place P_r which represents the servers having just finished a vacation. Hence, the servers of P_r are obliged to serve at least one call after the firing of the immediate transition W to join the place P_d, where they can commence another single vacation.

Initially, all customers are free, the orbit is empty and the c servers are available to serve the calls or to take a vacation.

At the arrival of a primary or repeated request to the place P_e, several alternatives are possible:

- If the place P_r of servers just returning from vacation, contains at least one server, the immediate transition W fires and the service of the arriving call starts.
- If the place P_r is empty and the place P_d contains at least one free server, the immediate transition X fires and the service period starts.
- If the two places P_d and P_r are empty which represents the fact that all the servers are busy or on vacation, the immediate transition Y fires and a token will be deposited in the place P_o. So, the customer joins the orbit.

D. Retrial systems with single vacations of the station

The GSPN modeling systems with single vacations of the station is the same as the model given in Fig. 2, in which the multiplicity of the arc connecting the place P_d to transition Z and transition t_v to place P_r equals c (rather than 1), because the c servers of the station take a vacation together. At the end of this period, c tokens corresponding to the c servers of the station will be deposited in P_r. Hence, the station can't take another vacation until each server serves at least one call.

IV. PERFORMANCE MEASURES

The aim of this section is to derive the formulas of the most important stationary performance indices. As, all the proposed models are bounded and the initial marking is a home state, the underlying continuous time Markov chains are ergodic for the different vacation policies. Hence, the steady-state probability distribution vector π exists and can be obtained as the solution of the linear system of equations $\pi.Q = 0$ with the normalization condition $\sum_i \pi_i = 1$, where π_i denotes the steady-state probability that the process is in state M_i and Q is the transition rates matrix. Having the probabilities vector π, several stationary performance indices of small cell wireless networks with different vacation policies can be derived as follows. In these formulas, $M_i(p)$ denotes the number of tokens in place p in marking M_i, A the set of reachable tangible markings, and $A(t)$ is the set of tangible markings reachable by transition t and $E(t)$ is the set of markings where the transition t is enabled.

- `The mean number of customers in orbit` (n_o)`:`

$$n_o = \sum_{i:M_i \in RS} M_i(P_o).\pi_i \tag{1}$$

- `The mean number of busy servers` (n_s)`:`

$$n_s = \sum_{i:M_i \in RS} M_i(P_s).\pi_i \tag{2}$$

- `The mean number of customers in the system` (n)`:`

$$n = n_s + n_o \tag{3}$$

- The mean number of servers on vacation (n_v):

$$n_v = \sum_{i:M_i \in RS} M_i(P_v).\pi_i \tag{4}$$

- The mean number of idle servers (n_f):

$$n_f = c - (n_s + n_v) \tag{5}$$

$$= \begin{cases} \sum_{i:M_i \in RS} M_i(P_d).\pi_i, \\ \text{in multiple vacations,} \\ \\ \sum_{i:M_i \in RS} [M_i(P_d) + M_i(P_r)].\pi_i, \\ \text{in single vacations.} \end{cases} \tag{6}$$

- The mean rate of generation of primary calls ($\overline{\lambda}$):

$$\overline{\lambda} = \sum_{i:M_i \in E(t_a)} M_i(P_a).\lambda.\pi_i \tag{7}$$

- The mean rate of generation of repeated calls ($\overline{\nu}$):

$$\overline{\nu} = \sum_{i:M_i \in E(t_r)} M_i(P_o).\nu.\pi_i \tag{8}$$

- The mean rate of service ($\overline{\mu}$):

$$\overline{\mu} = \sum_{i:M_i \in E(t_s)} M_i(P_s).\mu.\pi_i \tag{9}$$

- The mean rate of vacation ($\overline{\tau}$):

$$\overline{\tau} = \sum_{i:M_i \in E(t_v)} M_i(P_v).\theta.\pi_i \tag{10}$$

- The blocking probability of a primary call (B_p):

$$B_p = \begin{cases} \frac{\sum_{i:M_i \in RS} \sum_{j=1}^{K} j.\lambda.Prob[M_i(P_a)=j \& M_i(P_d)=0]}{\overline{\lambda}}, \\ \text{in multiple vacations,} \\ \frac{\sum_{i:M_i \in RS} \sum_{j=1}^{K} j.\lambda.P_{(i,j)}}{\overline{\lambda}}, \\ \text{in single vacations.} \end{cases} \tag{11}$$

where:

$$P_{(i,j)} = Prob[M_i(P_a) = j \& M_i(P_d) = 0 \& M_i(P_r) = 0].$$

- The blocking probability of a repeated call (B_r):

$$B_r = \begin{cases} \frac{\sum_{i:M_i \in A} \sum_{j=1}^{K} j.\nu.Prob[M_i(P_o)=j \& M_i(P_d)=0]}{\overline{\nu}}, \\ \text{in multiple vacations,} \\ \frac{\sum_{i:M_i \in A} \sum_{j=1}^{K} j.\nu.P_{(i,j)}}{\overline{\nu}}, \\ \text{in single vacations.} \end{cases} \tag{12}$$

where:

$$P_{(i,j)} = Prob[M_i(P_o) = j \& M_i(P_d) = 0 \& M_i(P_r) = 0].$$

- The blocking probability (B):

$$B = B_p + B_r \tag{13}$$

- The admission probability (A):

$$A = 1 - B \tag{14}$$

- Utilization of s servers (U_s): ($1 \leq s \leq c$)

$$U_s = \sum_{i:M_i(P_s) \geq s} \pi_i \tag{15}$$

- Vacation of s servers (V_s): ($1 \leq s \leq c$)

$$V_s = \sum_{i:M_i(P_v) \geq s} \pi_i \tag{16}$$

- Availability of s servers (A_s): ($1 \leq s \leq c$)

$$A_s = 1 - \sum_{i:M_i(P_s)+M_i(P_v) \geq s} \pi_i \tag{17}$$

- The mean waiting time ($\overline{W}$):

$$\overline{W} = n_o / \overline{\lambda} \tag{18}$$

- The mean response time ($\overline{R}$):

$$\overline{R} = (n_o + n_s) / \overline{\lambda} \tag{19}$$

V. VALIDATION OF RESULTS

In this section, we consider some numerical results to validate the proposed models and also to show the influence of system parameters and vacation policies on the performance measures of multiserver retrial systems. The numerical results were established using the GreatSPN tool.

In Table 1, some experimental results are collected when the servers vacation rate and the station vacation rate are very large. The results were validated by the Pascal program given in the book of Falin and Templeton [25] for the analysis of multiserver retrial queues without vacations. From this table, we can see that the corresponding performance measures are very close to the case without vacation and to each other with server or station vacation policy with very high vacation rate.

Define the parameter $\rho = N\lambda/\mu$, which is the largest offered load in the system. Table 2 shows the variation of the mean response time with ρ, for the single and multiple vacation policies, when the service station consists of one server and the retrial rate is very high. From this table, we can see that the numerical results are very close to those obtained by Trivedi [23] for single server queueing systems with vacations and without retrials, since the retrial rate is very large.

Table I
VALIDATION OF RESULTS IN MULTISERVER RETRIAL CASE WITHOUT VACATIONS

	Model without vacation [25]	Model with servers vacation	Model with station vacation
Number of servers	4	4	4
Size of source	20	20	20
Primary call generation rate	0.1	0.1	0.1
Service rate	1	1	1
Retrial rate	1.2	1.2	1.2
Vacation rate	-	1e+25	1e+25
Mean number of busy servers	1.800 748	1.800 768	1.800 758
Mean number of customers of repeated calls	0.191 771	0.191 788	0.191 786
Mean rate of generation of primary calls	1.800 748	1.800 744	1.800 746
Mean waiting time	1.106 495	1.106 518	1.106 510

Table II
MEAN RESPONSE TIME WITH $N = 50$, $\mu = 1$, $\theta = 0.5$, $c = 1$

ρ	Models without retrials [23]		Models with $\nu = 1e + 25$	
	Multiple vacations	Single vacations	Multiple vacations	Single vacations
0.1	3.107	1.494	3.106 810	1.493 581
0.3	3.391	2.370	3.390 962	2.370 404
0.5	3.834	3.172	3.833 990	3.172 221
0.7	4.592	4.152	4.592 591	4.152 760
0.9	6.000	5.718	6.000 657	5.719 090

VI. CONCLUSION

In this paper, we proposed a technique that allows modeling and analyzing finite-source multiserver retrial systems with different vacation policies using GSPNs. The novelty of the investigation is essentially the combination of multiplicity of servers with the simultaneous presence of repeated calls and vacations, which make the system rather complicated.

The flexibility of GSPNs modeling approach allowed us a simple construction of detailed and compact models for these systems. Moreover, it made it possible to verify many qualitative properties of interest by inspection of the reachability graph. From a performance point of view, the proposed approach offers a rich means of expressing interesting performance indices as a function of the Petri net elements.

Finally, many retrial and vacation systems problems and their solutions can be simplified using the stochastic Petri nets modeling approach with all the methods and tools developed within this framework.

REFERENCES

[1] J. R. Artalejo and M. J. Lopez-Herrero, *Cellular mobile networks with repeated calls operating in random environment*, Computers & Operations Research 37, no. 7, 2010, pp. 1158-1166.

[2] T. V. Do, *A new computational algorithm for retrial queues to cellular mobile systems with guard channels*, Computers & Indus. Engin. 59, 2010, pp. 865-872.

[3] N. Gharbi, L. Mokdad, and J. Ben-Othman, *Performance and Reliability Analysis of Small Cell Networks with Retrials and Different Breakdowns Disciplines: A Computational Approach*, The 5th IEEE International Workshop on Performance Evaluation of Communications in Distributed Systems and Web based Service Architectures (PEDISWESA'2013), Split, Croatia, July 7-10, 2013.

[4] N. Gharbi and L. Mokdad, *Performance Evaluation of Telecommunication Systems with Repeated Attempts and Two Servers Classes*, International Symposium on Performance Evaluation of Computer and Telecommunication Systems (SPECTS'2013), Toronto, Canada, July 7-10, 2013.

[5] P. Wuechner, J. Sztrik, and H. De Meer, *Modeling Wireless Sensor Networks Using Finite-Source Retrial Queues with Unreliable Orbit*, Proc. of the Workshop on Perf. Eval. of Computer and Communication Systems (PERFORM'2010), vol. 6821 of LNCS Publisher: Springer-Verlag, 2011.

[6] G. I. Falin and J. G. C. Templeton, *Retrial Queues*, Chapman and Hall, London, 1997.

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

[8] J. R. Artalejo, *Accessible bibliography on retrial queues: Progress in 2000-2009*, Mathematical and Computer Modelling 51, 2010, pp. 1071-1081.

[9] Z. G. Zhang and N. Tian, *Analysis of queueing systems with synchronous single vacation for some servers*, Queueing Systems: Theory and Applications, vol. 45, no. 2, 2003, pp. 161-175.

[10] Z. G. Zhang and N. Tian, *Analysis of queueing systems with synchronous vacations of partial servers*, Performance Evaluation, vol. 52, no. 4, 2003, pp. 269-282.

[11] C. H. Lin and J. C. Ke, *Multi-server system with single working vacation*, Applied mathematical Modelling, vol. 33, 2009, pp. 2967-2977.

[12] J. C. Ke, C. H. Lin, J. Y. Yang, and Z. G. Zhang, "Optimal (d,c) vacation policy for a finite buffer M/M/c queue with unreliable servers and repairs", Applied Mathematical Modelling, vol. 33, 2009, pp. 3949-3962.

[13] J. C. Ke, C. H. Wu, and Z. G. Zhang, "A Note on a Multi-server Queue with Vacations of Multiple Groups of Servers", Quality Technology and Quantitative Management, vol. 10, no. 4, 2013, pp. 513-525.

[14] B. T. Doshi, "Queueing systems with vacations: a survey", Queueing Systems, vol. 1, no. 1, 1986, pp. 29-66.

[15] H. Takagi, Queueing Analysis: A Foundation of Performance Evaluation, Volume I, Vacation and Priority Systems, Part I, North-Holland, Amsterdam, 1991.

[16] N. Tian and Z. G. Zhang, Vacation Queueing Models: Theory and Applications, Springer, NewYork, 2006.

[17] J. C. Ke, C. H. Wu, and Z. G. Zhang, "Recent Developments in Vacation Queueing Models: A Short Survey", International Journal of Operations Research, vol. 7, no. 4, 2010, pp. 3-8.

[18] S. J. Kwon, "Performance analysis of CDPD Sleep mode for power conservation in mobile end systems", IEICE Trans. Commun. E84, 2001.

[19] H. Li and T. Yang, "A single server retrial queue with server vacations and a finite number of input sources", European Journal of Operational Research 85, 1995, pp. 149-160.

[20] G. K. Janssens, "The quasi-random input queueing system with repeated attempts as a model for collision-avoidance star local area network", IEEE Transactions on Communications 45, 1997, pp. 360-364.

[21] G. Choudhury, "A two phase batch arrival retrial queueing system with Bernoulli vacation schedule", Applied Mathematics and Computation 188, 2007, pp. 1455-1466.

[22] Z. Wenhui, "Analysis of a single-server retrial queue with FCFS orbit and Bernoulli vacation", Applied Mathematics and Computation 161, 2005, pp. 353-364.

[23] K. S. Trivedi and O. C. Ibe, "Stochastic Petri net analysis of finite-population vacation queueing systems", Queueing Systems 8, 1991, pp. 111-128.

[24] M. A. Marsan, G. Balbo, G. Conte, S. Donatelli, and G. Franceschinis, Modelling with Generalized Stochastic Petri Nets, John Wiley and Sons, New York, 1995.

[25] G. I. Falin and J. G. C. Templeton, Retrial Queues, Chapman and Hall, London, 1997.

 ISBN: 978-1-61208-347-6