Ann Oper Res (2009) 170: 95–112
DOI 10.1007/s10479-008-0433-z

Analysis of cyclic queueing networks with parallelism and vacation

Vittoria de Nitto Personè

Published online: 24 September 2008

Abstract The aim of this paper is to improve the machine interference model with vacation to deal with more recent problems of the communication area. To this scope the model is extended to include parallelism in the vacation station. The underlying Markov process is analyzed and a state arrangement is found that yields an efficient matrix-analytic technique that substantially lowers down the time- and space-complexity of standard methods. A numerical example of the method effectiveness is presented, and an example of resource allocation is introduced that finds applications in the QoS management of wireless networks.

Keywords Queueing models · Vacation server · Parallel processing · Matrix-analytic solutions · Bandwidth allocation · QoS management

Introduction

Queueing models with server vacations have received widespread attention in literature because of their extensive applications in the industrial engineering area (Takagi 1991; Gupta 1997; Chao and Zhao 1998; Zhang and Tian 2003; Gupta and Sikdar 2004) and the computer and communication areas (Takagi 1991; Frigui and Alfa 1998; Fiems et al. 2001; Lee and Choi 2001; Servi and Finn 2002; Xu and Alfa 2002; Mehmet-Ali et al. 2003; Sevcik 2005; de Nitto Personè and Iazeolla 2006). Recent applications are found in many fields, from distributed databases (Xu and Alfa 2002), to internet protocol access networks (Servi and Finn

The author is thankful to the anonymous referees for the improvements their comments have earned to the quality of the presentation and to the completeness of the paper. The author is thankful to Giuseppe Iazeolla, whose careful reading of the original draft of this paper led to significant improvements in its overall quality. This work was partially supported by funds from the FIRB project "Performance Evaluation of Complex Systems: Techniques Methodologies and Tools" and by the University of Roma TorVergata project on High Performance ICT Platforms.

V. de Nitto Personè (✉)
Department of Computer Science Systems and Industrial Engineering, University of Roma TorVergata, Via del Politecnico 1, 00133 Roma, Italy
e-mail: denitto@info.uniroma2.it

2002), to wireless ATM multiplexing systems (Lee and Choi 2001; Mehmet-Ali et al. 2003), to manufacturing systems, packet switching and polling systems (Frigui and Alfa 1998; Fiems et al. 2001), and wireless networks (de Nitto Personè and Iazeolla 2006).

The vacation server models can be divided into two broad classes: open models, i.e. models with arrivals from external sources (Takagi 1991; Chao and Zhao 1998; Frigui and Alfa 1998; Fiems et al. 2001; Lee and Choi 2001; Servi and Finn 2002; Xu and Alfa 2002; Mehmet-Ali et al. 2003; Zhang and Tian 2003; Gupta and Sikdar 2004), and closed models, i.e. with a fixed number of circulating customers (Gupta 1997).

This paper considers that the vacation model is a closed model with two stations. The closed two-station model has traditionally found applications in the Machine Interference Problem (MIP) of the industrial engineering area (Gupta 1997). However, to deal with systems where the vacation station carries on computer-intensive tasks, the MIP traditional model needs to evolve in at least two directions. Such systems can be found in the modern context of the communication area, more specifically in the QoS management of wireless networks. In such a context, two features are to be met by the vacation model to be of effective use: (1) the model should support the event that the vacation station has parallel processing capabilities and, (2) the model evaluation method should be a low time and low space complexity method.

The importance of such two features is discussed at length in Sect. 3.

The first contribution of this paper, therefore, is the model capability to deal with parallelism. Indeed, in the existing literature the vacation station is a sequential processing station (i.e. without parallelism). We instead assume that such a station consists of a series of parallel servers.[1]

For such a model, however, the existing literature does not provide efficient evaluation algorithms. The second contribution of this paper, therefore, is an efficient technique for a very low time- and space-consuming evaluation of the system performance. This is obtained by finding a state arrangement in the underlying Markov process that gives the generator matrix a repetitive block tridiagonal structure to yield an evaluation time-complexity and space-complexity substantially lower than in standard methods.

The paper is organized as follows. Section 1 describes the proposed model. In particular, Sects. 1.1 and 1.2 present the queueing network model and the underlying Markov process respectively. Section 1.3 proves that the process generator matrix can be arranged in such a way to obtain a repetitive block tridiagonal structure and thus a lower time and space complexity than standard methods. Section 2 introduces an efficient matrix-analytic technique with low time and space requirements, and shows numerical results that prove the technique effectiveness. Section 3 gives an application example to resource allocation in wireless networks and motivates the parallel processing requirements of the vacation station. Section 4 concludes the paper.

1 The model and the process structure

In this section the queueing model is presented and a proper system state definition is introduced that yields an ordered partition of the state space, which gives the process generator matrix the nice repetitive block tridiagonal structure for the method effectiveness.

[1] In the vacation modeling area, two cases can be considered: the "station" vacation and the "server" vacation case. This paper deals with the station vacation case, in other words it is assumed that all parallel servers take vacation simultaneously.

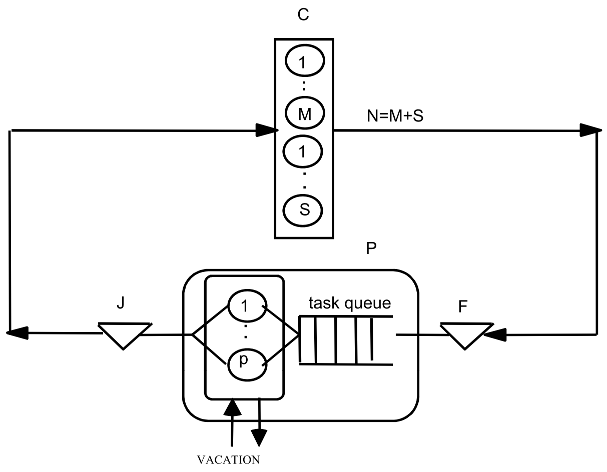

Fig. 1 The considered model

1.1 The queueing network model

The considered queueing model is depicted in Fig. 1 and consists of two centers, denoted C and P. Center C is an exponential IS center, where IS is used in literature to mean Infinite Server, in other words that there exists a sufficient number of servers (N) to accommodate all N existing jobs (and so no job waits in line in front of C). Node P is a FIFO exponential parallel-processing center with p processors (or servers). We assume an identical exponential service rate μ for each server in P. Extension of center C and P exponential services to the general (coxian) service case is immediate. Centers C and P are connected through a pair of fork and join centers. There exists a fixed number N of circulating jobs.

Upon passage through the fork node F each *job* splits itself into p sibling *tasks*, which are assigned to available processors, and the remaining tasks are enqueued. Upon completion of service, tasks direct themselves to the join node J, where all sibling tasks of a given job wait to be completed before the job is routed back to center C.

The service times of tasks in the processors of P have random durations. Therefore tasks belonging to different jobs can be processed simultaneously in center P.

It is assumed that center P takes a vacation of exponential duration (extension to the general coxian case is also immediate) every time it becomes empty (task queue + processors). It is assumed that there are three vacation parameters: β, γ and ε, to accommodate multiple, single or hybrid vacation cases (Gupta 1997). Parameter γ is the rate by which center P returns from vacation and finds no tasks in its queue (i.e. center C has routed no jobs to P during its vacation). In this case it is assumed that center P takes another vacation with rate ε. In the case that center C has instead routed jobs to P during its vacation, it is assumed that P returns from vacation with rate β.

It is assumed that the center C is load dependent. In other words, its service rate is characterized by two parameters η and α, with the following rule: at most M jobs ($M < N$) can be processed in C at rate η, while the remaining ones are processed at rate α. As a consequence, the following load dependent service rate $g(k)$ can be defined for center C with k jobs:

$$g(k) = \begin{cases} k\eta & \text{if } 1 \le k \le M \\ (M\eta + (k-M)\alpha) & \text{if } M < k \le N \end{cases} \tag{1}$$

Table 1 View of the center parameters

	Service rate	
Center C	High level	Low level
	η	α
Center P	Service rate of each processor	
	μ	
	Center vacation rates	
	$\beta, \gamma, \varepsilon$	

In the industrial engineering area, such a service rate is used to represent failure rates for operating and spare machines in the Machine Interference Problem (MIP) (Gupta 1997), where center C consists of a pool of N homogeneous machines subject to random failures and operating under the supervision of a repair center P. The N homogeneous machines are distinguished into M operating machines and S spare machines ($N = M + S$), so that when an operating machine breaks down a spare machine is immediately substituted for it.

This paper introduces an extension of the classical MIP model to the case that center P consists of p parallel servers. When an operating or spare machine breaks down, a repair job arises for center P. Such a job is split into p identical tasks, and when all repair tasks of a given job have been served, the job is complete and the repaired machine comes back to center C.

In the computer communications area, the model in Fig. 1 can be used to represent the activities of a "base station and switching center" of a wireless network, as better seen in Sect. 3, where the problem of allocating the bandwidth among various service levels in an optimal way with respect to a negotiated QoS is considered. In this framework, function $g(k)$ is used to dynamically change the station service level by adopting an upgrade/degradation mechanism according to a negotiated QoS. Because of this, up to M jobs will be accommodated at the high service level (rate η), while the remaining $k - M$ at the lower level (rate α). The vacation period is used to model additional services that the base station and the switching center perform to manage bandwidth partition and assist customers, as better seen in Sect. 3.

Table 1 synthesizes the parameters of the two centers in the cyclic queueing model.

1.2 The Markov process definition

To define the Markovian model underlying the Fig. 1 network, a concept of system state σ is introduced that gives information on the status (vacation or not) of P, on the number of tasks in P and in J, and the number of jobs in C.

Let i be a job in center P, split by the fork node F into p sibling tasks. The *job state* is expressed by the pair:

$$s_i = (j, r)$$

with j the number of its tasks in the join node J, and r the number in the processors of P, the remaining $(p - j - r)$ ones being in the task queue. The *system state* is defined by:

$$\sigma = [f, k, v]$$

where k $(0 \le k \le N)$ is the number of jobs in C,

$$f = \{s_1, s_2, \dots, s_z\}, \quad 0 \le z \le p$$

is the set of job states s_i for jobs with at least one task in the processors of P, and

$$v = \begin{cases} 0 & \text{if center } P \text{ is on vacation} \\ 1 & \text{if center } P \text{ is available} \end{cases}$$

When center P is in vacation, the inclusion of set f in σ is meaningless and so we simply write $\sigma = [k, v]$ with $v = 0$, in other words we write $\sigma = [k, 0]$ to mean that P is in vacation and C holds k jobs.

Set f is called the system P-firing front and the set F_t defined by

$$F_t = \{f_1, f_2, \dots, f_{|F_t|}\}$$

denotes the set of all possible P-firing fronts when there are t jobs in P, each split into its p tasks. Note that f_i describes the state of the jobs with at least one task in the processors of P, and such jobs can be at most p in number. Therefore it is easy to be convinced that $\forall t, t' \geq p$, there results $F_t = F_{t'}$, and by denoting by F the cardinality of F_t one can write $|F_t| = F$. In other words F depends on p, and is independent of k and N. Note that the condition $t \geq p$ takes place only with $N > p$. So, as shown in Sect. 1.3, it is for such values of N that the generator matrix takes the block repetitive structure that yields the efficient model evaluation method.

Example 1 Let us consider the $N = 5$ and $p = 3$ case. It is easy to be convinced that: $F_1 = \{f_1 = \{(0,3)\},\ f_2 = \{(1,2)\},\ f_3 = \{(2,1)\}\}$, $F_2 = \{f_1 = \{(0,3)\},\ f_2 = \{(1,2),(0,1)\},$ $f_3 = \{(1,2),(1,1)\},\ f_4 = \{(2,1),(0,2)\},\ f_5 = \{(2,1),(1,2)\},\ f_6 = \{(2,1),(2,1)\}\}$, and, for $t = 3,4,5$, $F_t = \{f_1 = \{(0,3)\},\ f_2 = \{(1,2),(0,1)\},\ f_3 = \{(1,2),(1,1)\},\ f_4 = \{(2,1),(0,2)\},\ f_5 = \{(2,1),(1,2)\},\ f_6 = \{(2,1),(2,1),(0,1)\},\ f_7 = \{(2,1),(2,1),(1,1)\},$ $f_8 = \{(2,1),(2,1),(2,1)\}\}$.

In this case, $|F_t| = F = 8$, $\forall t \geq p$. The firing front $f_1 = \{(0,3)\}$ represents a job with all its three tasks in execution in P. This job-state is feasible for any value of $t = 1, 2, \dots, 5$. If $t = 1$ only one job is in P, so the task queue is empty and the remaining $N - 1$ jobs are all in C. On the contrary, if $t = 5$ all the jobs are in P, so the tasks of the remaining $N - 1$ jobs are all enqueued in the task queue. Note that the firing front $f_4 = \{(2,1),(0,2)\}$ in F_2 means that the two jobs in P are both in execution. The former, with job state $(2,1)$, has two tasks completed in J and the third in execution in P. The latter, with job-state $(0,2)$, has two tasks in execution in P and the third in the task queue. So tasks of different jobs can be processed simultaneously in center P.

Let E denote the system state space, in other words, $E = \{\sigma_1, \sigma_2, \dots, \sigma_{|E|}\}$. It is easy to be convinced that $|E| = O(NF)$ as shown in next section. Also, let $\pi(\sigma_i)$ denote the equilibrium probability of state σ_i and $\boldsymbol{\pi} = [\pi(\sigma_1), \dots, \pi(\sigma_{|E|})]$ the steady state probability vector. The system solution can then be expressed as:

$$\begin{gathered} \boldsymbol{\pi}\mathbf{S} = \mathbf{0} \\ \sum \pi(\sigma_i) = 1 \end{gathered} \tag{2}$$

where $\mathbf{S}$ is the generator matrix of order (NF) of the Markovian process.

The solution time-complexity of system (2) by conventional methods is of the order $O((NF)^3)$. It is however known that for block tridiagonal matrices with repetitive structure, such a complexity can be reduced to order $O(NF^3)$ (Le Boudec 1989; Grassman and Heyman 1990; Ye and Li 1994; de Nitto Personè and Grassi 1996). In the next section, an appropriate arrangement of matrix $\mathbf{S}$ is given in order to obtain the repetitive block tridiagonality, thus reducing the order $O((NF)^3)$ problem to an order of $O(NF^3)$. Besides that, in Sect. 2 an explicit low time, low space matrix-analytic solution is introduced for any practical application.

1.3 The arrangement of the Markov process

In this section, by exploiting the definition of system state σ, an ordered partition of the state space E is obtained to give matrix $\mathbf{S}$ the necessary block tridiagonal and repetitive structure.

To this purpose, let us partition the state space E into $N+1$ subspaces, according to the following:

$$E = \{E(0), E(1), \ldots, E(N-p-1), E(N-p), \ldots, E(N)\} \tag{3}$$

where $E(k)$ denotes the subspace that includes all system states with k jobs in center C, that is:

$$E(k) = \{\sigma = [k, 0]\} \cup \{\sigma = [f_i, k, 1] \mid \forall f_i \in F_{N-k}\}$$

In other words, $E(k)$ includes all states in which $N-k$ jobs are split into tasks and these tasks are partitioned among the p processors and the task queue according to all possible firing fronts f_i. Subspace $E(k)$ also includes state $[k, 0]$ in which P is on vacation and C holds k jobs. It is easy to be convinced that $|E| = O(NF)$, as the reader can also see from the example below.

Example 2 By assuming $N = 5$ and $p = 3$, as in Example 1, the state space E can be partitioned into six subspaces $E = \{E(0), E(1), \ldots, E(5)\}$ as follows:

$$\begin{aligned}
E(0) = \{&[0,0],\\
&[\{(0,3)\}, 0, 1],\\
&[\{(1,2),(0,1)\}, 0, 1],\\
&[\{(1,2),(1,1)\}, 0, 1],\\
&[\{(2,1),(0,2)\}, 0, 1],\\
&[\{(2,1),(1,2)\}, 0, 1],\\
&[\{(2,1),(2,1),(0,1)\}, 0, 1],\\
&[\{(2,1),(2,1),(1,1)\}, 0, 1],\\
&[\{(2,1),(2,1),(2,1)\}, 0, 1]\}
\end{aligned}
\qquad
\begin{aligned}
E(1) = \{&[1,0],\\
&[\{(0,3)\}, 1, 1],\\
&[\{(1,2),(0,1)\}, 1, 1],\\
&[\{(1,2),(1,1)\}, 1, 1],\\
&[\{(2,1),(0,2)\}, 1, 1],\\
&[\{(2,1),(1,2)\}, 1, 1],\\
&[\{(2,1),(2,1),(0,1)\}, 1, 1],\\
&[\{(2,1),(2,1),(1,1)\}, 1, 1],\\
&[\{(2,1),(2,1),(2,1)\}, 1, 1]\}
\end{aligned}$$

$$\begin{aligned}
E(2) = \{&[2,0],\\
&[\{(0,3)\}, 2, 1],\\
&[\{(1,2),(0,1)\}, 2, 1],\\
&[\{(1,2),(1,1)\}, 2, 1],\\
&[\{(2,1),(0,2)\}, 2, 1],\\
&[\{(2,1),(1,2)\}, 2, 1],\\
&[\{(2,1),(2,1),(0,1)\}, 2, 1],\\
&[\{(2,1),(2,1),(1,1)\}, 2, 1],\\
&[\{(2,1),(2,1),(2,1)\}, 2, 1]\}
\end{aligned}
\qquad
\begin{aligned}
E(3) = \{&[3,0],\\
&[\{(0,3)\}, 3, 1],\\
&[\{(1,2),(0,1)\}, 3, 1],\\
&[\{(1,2),(1,1)\}, 3, 1],\\
&[\{(2,1),(0,2)\}, 3, 1],\\
&[\{(2,1),(1,2)\}, 3, 1],\\
&[\{(2,1),(2,1)\}, 3, 1]\}
\end{aligned}$$

$$\begin{aligned}
E(4) = \{&[4,0],\\
&[\{(0,3)\}, 4, 1],\\
&[\{(1,2)\}, 4, 1],\\
&[\{(2,1)\}, 4, 1]\}
\end{aligned}
\qquad
\begin{aligned}
E(5) = \{&[5,0],\\
&[5,1]\}
\end{aligned}$$

$$
\mathbf{S} = \begin{bmatrix}
\mathbf{S}_{00} & \mathbf{S}_{01} & & & & & \\
\mathbf{S}_{10} & \mathbf{S}_{11} & \mathbf{S}_{12} & & & & \\
 & \cdot & \cdot & \cdot & & & \\
 & & \mathbf{S}_{k,k-1} & \mathbf{S}_{k,k} & \mathbf{S}_{k,k+1} & & \\
 & & & \mathbf{S}_{N-p-1,N-p-2} & \mathbf{S}_{N-p-1,N-p-1} & \mathbf{S}_{N-p-1,N-p} & \\
 & & & & \cdot & \cdot & \cdot \\
 & & & & \mathbf{S}_{N-1,N-2} & \mathbf{S}_{N-1,N-1} & \mathbf{S}_{N-1,N} \\
 & & & & & \mathbf{S}_{N,N-1} & \mathbf{S}_{N,N}
\end{bmatrix}
$$

Fig. 2 The block-tridiagonal structure of generator matrix $\mathbf{S}$

According to definitions in Sect. 1.2, the notation is such that e.g. system state $[\{(1,2),(1,1)\},0,1]$ in $E(0)$ means that:

- Center P is available ($v = 1$);
- There are $k = 0$ jobs in C, in other words all $N = 5$ jobs are in center P in a way that:
  - 2 of such jobs are in state $(1,2)$ and state $(1,1)$ respectively. Job $(1,2)$ has 1 task in J and the remaining 2 tasks each in one of the processors of P, while job $(1,1)$ has 1 task in J, 1 task in one of processors of P, and the third task is understood (being $p = 3$) to stay in the task queue;
  - all the remaining $5 - 2$ jobs are understood (being $k = 0$) to stay waiting in line in the task queue, each split into 3 tasks.

The example state space consists of $|E| = 40$ states, with $N = 5$ and $F = 8$. In other words, $|E| = O(NF)$, as expected.

The partition (3) of state space E into subspaces $E(k)$ (ordered according to increasing values of k), yields the generator matrix $\mathbf{S}$ of Fig. 2.

The reader can find in Appendix 1 an illustration of the semantics of the diagonal and side-diagonal blocks $\mathbf{S}_{k,k}$, $\mathbf{S}_{k,k-1}$ and $\mathbf{S}_{k,k+1}$.

Theorem 1 below proves that the structure of $\mathbf{S}$ is of block-tridiagonal and repetitive type.

Theorem 1 *The generator matrix* $\mathbf{S}$ *is block-tridiagonal with repetitive structure in rows* $0, 1, \ldots, N - p - 1$ *as follows* (*see Fig.* 3 *ahead*):

- *the upper-diagonal blocks* $\mathbf{S}_{k,k+1}$, *for* $0 \le k < N - p$, *are identical blocks denoted* $\mathbf{A}$;
- *the diagonal blocks* $\mathbf{S}_{k,k}$, *for* $1 \le k \le N - p$, *are repetitive blocks with* $\mathbf{S}_{k,k} = \mathbf{B} - g(k)\mathbf{I}$ *where* $\mathbf{B}$ *denotes block* $\mathbf{S}_{00}$, $\mathbf{I}$ *denotes the identity matrix and* $g(k)$ *is defined in* (1);
- *the lower-diagonal blocks* $\mathbf{S}_{k,k-1}$, *for* $1 \le k \le N - p$, *are repetitive blocks with* $\mathbf{S}_{k,k-1} = g(k)\mathbf{I}$;
- *blocks* $\mathbf{S}_{ij}$ *such that* $|i - j| > 1$ *are null blocks.*

Blocks $\mathbf{S}_{k,k+1}$, $\mathbf{S}_{k,k}$ *and* $\mathbf{S}_{k,k-1}$, *for* $0 \le k < N - p$, *and blocks* $\mathbf{S}_{N-p,N-p}$ *and* $\mathbf{S}_{N-p,N-p-1}$ *are square matrices of order* $(F + 1)$.

Proof As defined above, let k be the number of jobs in center C, let t be the number of jobs in center P and z the number of jobs with at least one task in the processors of P.

$$\mathbf{S}=\left[\begin{array}{cccccc} \mathbf{B} & \mathbf{A} & & & & \\ g(1)\mathbf{I} & \mathbf{B}-g(1)\mathbf{I} & \mathbf{A} & & & \\ & \cdot & \cdot & \cdot & & \\ & & g(k)\mathbf{I} & \mathbf{B}-g(k)\mathbf{I} & \mathbf{A} & \\ & & \cdot & \cdot & \cdot & \\ & & & g(N-p)\mathbf{I} & \mathbf{B}-g(N-p)\mathbf{I} & \mathbf{G} \\ & & & & \mathbf{H} & \mathbf{L} \end{array}\right]\begin{array}{c} 0 \\ 1 \\ \\ \\ \\ N-p \\ N-p+1,\dots,N \end{array}$$

Fig. 3 The repetitive structure of generator matrix **S**

The proof of tridiagonality comes from the consideration that the partition (3) of E is based on the number k of jobs in center C, and is ordered by increasing values of k. So, for any given state $\sigma \in E(k)$, the only possible transitions are from σ to states σ' belonging to $E(k-1)$, $E(k)$ or $E(k+1)$. Indeed, a state transition occurs for a job arrival to or departure from center C (that changes k in $k+1$ or in $k-1$ respectively), or for a service completion internal to center P (that does not change k), or finally for a P vacation move (that also does not change k).

With regard to the repetitive structure proof, let us first consider that the condition $k \le N-p$ guarantees that at least p jobs are in P ($t \ge p$). Under this condition, the relationship $F_t = F_{t'}$, $|F_t| = F$, $\forall t, t' \ge p$ holds, as mentioned in Sect. 1.2. Then for any $k, k' \le N-p$ we have:

$$|E(k)| = |E(k')| = F + 1 \quad \text{and} \tag{4}$$

$$\forall [f, k, 1] \in E(k) : \exists [f', k', 1] \in E(k') \quad \text{such that } f = f' \tag{5}$$

From (4) there follows that blocks $\mathbf{S}_{k,k+1}$, $\mathbf{S}_{k,k}$ and $\mathbf{S}_{k,k-1}$, for $0 \le k < N-p$, and blocks $\mathbf{S}_{N-p,N-p}$ and $\mathbf{S}_{N-p,N-p-1}$ are square matrices of order $(F+1)$.

Moreover, we give the states $\sigma = [f_i, k, v]$ in $E(0)$ the same internal ordering as in $E(1)$ through $E(N-p)$ with respect to f_i, see Example 2. Under this assumption one obtains:

- Matrices $\mathbf{S}_{k,k+1}$ describe transitions from $E(k)$ to $E(k+1)$. Such transitions take place only if P is available (i.e. not in vacation) and, in particular, if the concluding task of a given job is completed and the job is routed back to center C. The transitions are from $[f, k, 1]$ to $[f', k+1, 1]$; the rate of such transitions depends only on f and f'; hence from (5) we have $\mathbf{S}_{k,k+1} = \mathbf{A}$ for $k < N-p$.
- For $1 \le k \le N-p$, matrices $\mathbf{S}_{k,k-1}$ describe transitions from $[k, 0]$ to $[k-1, 0]$ and from $[f, k, 1]$ to $[f, k-1, 1]$, since each job arriving in P finds all processors busy and hence f does not change. Hence, $\mathbf{S}_{k,k-1}$ is a diagonal matrix of order $(F+1)$, with the diagonal entries of value $k\eta$ if at most M jobs are in C ($1 \le k \le M$), and of value $M\eta + (k-M)\alpha$ if more than M jobs are in C ($M < k \le N-p$) according to (1). Hence we have $\mathbf{S}_{k,k-1} = g(k)\mathbf{I}_F$.
- Matrices $\mathbf{S}_{k,k}$ describe transitions from $E(k)$ to $E(k)$, that is moves of tasks from the processors to the J center. Such transitions are from $[f, k, 1]$ to $[f', k, 1]$; the rate of such transitions depends only on f and f', hence the non-diagonal entries are independent of k. Each diagonal entry is by definition the negative sum of all other entries in the same

row of matrix $\mathbf{S}$, that, by the above considerations, are all independent of k except for the entry $g(k)$ that comes from $\mathbf{S}_{k,k-1}$. Hence we have $\mathbf{S}_{k,k} = \mathbf{B} - g(k)\mathbf{I}_F$ for $1 \le k \le N - p$. □

Being of the type defined in Le Boudec (1989), Grassman and Heyman (1990), Ye and Li (1994), de Nitto Personè and Grassi (1996), the Fig. 2 repetitive block tridiagonal structure reduces the order $O((NF)^3)$ problem to an order $O(NF^3)$ one.

In Sect. 2 a matrix-analytic solution is introduced. To this purpose the Fig. 2 matrix is rewritten in Fig. 3 to better clarify the nature of the various blocks.

Example 3 For our example case with $N = 5$ and $p = 3$, blocks $\mathbf{A}$ and $\mathbf{B}$ are as follows:

$$\mathbf{A} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \mu & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \mu & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2\mu & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 2\mu & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 3\mu & 0 & 0 \end{pmatrix}$$

$$\mathbf{B} = \begin{pmatrix} -\beta & \beta & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -3\mu & 3\mu & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -3\mu & \mu & 2\mu & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -3\mu & 0 & 3\mu & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -3\mu & 2\mu & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -3\mu & 2\mu & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -3\mu & \mu & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -3\mu & \mu \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -3\mu \end{pmatrix}$$

As we stated in the proof of Theorem 1, all state transitions appearing in blocks $\mathbf{A}$ and $\mathbf{B}$ are independent of k, the number of jobs in C, and depend only on f and f'. As a consequence, the transition rates take the value $x\mu$, with $x = 1, \dots, p$, the number of processors that complete a task.

Finally, the element $b_{12} = \beta$ of $\mathbf{B}$, corresponds to the transition rate of center P returning from vacation when the task queue is not empty. For example from the state $[\{(0,3)\}, 2, 0]$ to the state $[\{(0,3)\}, 2, 1]$ in $E(2)$ of Example 2.

Note that the remaining vacation parameters γ and ε are the ones that regulate the transition rates of return from vacation (γ) and return to vacation (ε) when P is empty. Such parameters therefore appear in block $\mathbf{S}_{N,N}$ that is part of matrix $\mathbf{L}$ better defined below.

Besides blocks $\mathbf{B}$ and $\mathbf{A}$ that belong to the repetitive structure, blocks $\mathbf{G}$, $\mathbf{H}$ and $\mathbf{L}$ are to be considered, which are no part of the repetitive structure and relate to states in $E(N-p+1)$, $E(N-p+2), \dots, E(N)$. Let us group such states in subset E^*, defined as follows:

$$E^* = \bigcup_{k=N-p+1}^{N} E(k)$$

On this basis, the state space partition (3) can be rewritten as:

$$E = \{E(0), E(1), \ldots, E(N-p), E^*\} \tag{6}$$

By this notation one can say that:

- $\mathbf{L}$ is a matrix of order L, with $L = |E^*|$. Matrix $\mathbf{L}$ groups the blocks $\mathbf{S}_{N-p+1,N-p+1}$, $\mathbf{S}_{N-p+1,N-p+2}, \mathbf{S}_{N-p+2,N-p+1}, \ldots, \mathbf{S}_{N-1,N}, \mathbf{S}_{N,N}$ ($3p-2$ blocks in total), each of dimension $< (F+1)$, since less than p jobs can be in the processors[2] of P. Therefore the result is that $L < (3p-2)(F+1)$;
- $\mathbf{G}$ is a matrix of dimension $(F+1) \times L$;
- $\mathbf{H}$ is a matrix of dimension $L \times (F+1)$.

The knowledge of the dimension of matrices $\mathbf{G}$, $\mathbf{H}$ and $\mathbf{L}$ together with that of matrices $\mathbf{B}$ and $\mathbf{A}$ (of order $(F+1)$ as stated in Theorem 1) will be used in Sect. 2 to reason about the complexity of the matrix-analytic solution.

2 The matrix-analytic solution and its complexity

In this section an explicit matrix-analytic solution is proved to exist for the evaluation of vector $\boldsymbol{\pi}$. Its time-complexity is shown to be of order $O(NF^3)$. The space complexity is also evaluated and shown to be of order $O(pF^2)$.

Let us denote $\boldsymbol{\pi} = [\boldsymbol{\pi}_0, \boldsymbol{\pi}_1, \ldots, \boldsymbol{\pi}_{N-p}, \boldsymbol{\pi}^*]$, the steady state probability vector, rearranged according to partition (6). The following can be stated:

Theorem 2 *The explicit matrix-analytic form for the solution of system* (2) *can be written as*:

$$[\boldsymbol{\pi}_0, \boldsymbol{\pi}_1, \boldsymbol{\pi}^*] = [1, \mathbf{0}](\hat{\mathbf{R}})^{-1} \tag{7}$$

$$\begin{aligned}\boldsymbol{\pi}_k = {} & -g(2)^{-1}\boldsymbol{\pi}_0\mathbf{A}\mathbf{C}_{11}(k-2) + \boldsymbol{\pi}_1(\mathbf{C}_{21}(k-2) - g(2)^{-1}(\mathbf{B} - g(1)\mathbf{I}_F)\mathbf{C}_{11}(k-2)) \\ & (k = 2, \ldots, N-p)\end{aligned} \tag{8}$$

where

- $[1, \mathbf{0}]$ *denotes a row vector consisting of a* 1 *and a zero sub vector*;
- $\hat{\mathbf{R}}$ *is a square matrix of order* $(2(F+1)+L)$ *obtained by the boundary equations related to states in* $E(0)$, $E(N-p+1), \ldots, E(N)$ *and the normalization condition* (*see Appendix* 1);
- *Matrices* $\mathbf{C}_{11}(k-2)$ *and* $\mathbf{C}_{21}(k-2)$ *are order* $(F+1)$ *square sub matrices of matrix* $\mathbf{C}(k-2) = \begin{bmatrix} \mathbf{C}_{11}(k-2) & \mathbf{C}_{12}(k-2) \\ \mathbf{C}_{21}(k-2) & \mathbf{C}_{22}(k-2) \end{bmatrix}$, *defined as follows*:

$$\mathbf{C}(j) = \begin{cases} \mathbf{I} & \text{if } j = 0 \\ \mathbf{C}(j-1)\begin{bmatrix} -g(j+2)^{-1}(\mathbf{B} - g(j+1)\mathbf{I}_F) & \mathbf{I} \\ -g(j+2)^{-1}\mathbf{A} & 0 \end{bmatrix} & \text{if } j > 0 \end{cases}$$

in product iterative form with $g(.)$ *given by* (1).

[2]Note that if $k > N-p$, less than p jobs are in P. As a consequence, for a given $k' \le N-p$, for each state $[f', k', 1] \in E(k')$ such that $f' = \{s_1, s_2, \ldots, s_p\}$, no state $[f, k, 1] \in E(k)$ exists such that $f = f'$. Hence: $|E(k)| < |E(k')|$.

Proof See Appendix 2. □

On this basis the time and space complexity of solution (7), (8) can be evaluated.

1. Time complexity The matrix-analytic solution (7), (8) requires the computation of the boundary state probabilities $[\boldsymbol{\pi}_0, \boldsymbol{\pi}_1, \boldsymbol{\pi}^*]$ in (7) and of the remaining state probabilities $\boldsymbol{\pi}_k$ $(k = 2, \ldots, N - p)$ in (8).

Matrix $\hat{\mathbf{R}}$ is of order $2(F + 1) + L$, or $O(pF)$ since $L \leq (3p - 2)(F + 1)$. Hence, the time complexity to obtain $(\hat{\mathbf{R}})^{-1}$ is $O((pF)^3)$. On the other hand, the computation of the column vector (15) (see Appendix 2), necessary to derive $\hat{\mathbf{R}}$, has time complexity $O((N - p)F^3)$, therefore the asymptotic time complexity of (7) is $O(NF^3)$. On the other hand, the computation of the steady state probabilities $\boldsymbol{\pi}_k$ $(k = 2, \ldots, N - p)$ in (8) can be conveniently carried out by (11) (see Appendix 2) whose asymptotic time complexity is $O((N - p)F^2)$. In conclusion, the proposed solution method has asymptotic time-complexity $O(NF^3)$.

Table 2 shows the time-consumption versus N of the standard method (2) compared to this paper method (7) and (8). Times are seconds of Intel Pentium 4 (3.4 GHz, 2 GB RAM) and use of Matlab for the inversion of matrices $\mathbf{S}$ and $\hat{\mathbf{R}}$. The results show the efficiency of the method used in this paper. By comparing the $(p = 4, \mu = 125)$ columns of the standard and this paper method, it is easily seen that the time complexity of the paper method is very limitedly sensitive to the growth of the number N of users, while in the conventional method there is a strong sensitivity versus N. This is still more evident in the $p = 5$ case where the standard method is unable (na) to give results for $N = 200$, and in the $p = 6$ case for $N = 80$ through 200. An insight into the effectiveness of this paper method is easily obtained by considering that the solution matrix $\mathbf{S}$ of the conventional method grows from 576×576 for $N = 30$ to 3976×3976 for $N = 200$ in the $p = 4$ case, while the corresponding matrix $\hat{\mathbf{R}}$ of this paper method remains of dimension 76×76 for any N. A similar situation can be found in the other p cases, for example matrix $\mathbf{S}$ grows from 2253×2253 for $N = 30$ to 16023×16023 for $N = 200$ in the $p = 6$ case, while matrix $\hat{\mathbf{R}}$ remains 390×390 for any N.

In Table 2 the practical insensitivity (as expected) of the time complexity to the model parameters is also shown. The case of the processor parameter μ is shown, and it is easy to be convinced that a similar insensitivity effect holds for the remaining parameters $(\eta, \alpha, \beta, \gamma, \varepsilon)$. All parameters indeed only affect the entries of the solution matrix (see matrices $\mathbf{A}$ and $\mathbf{B}$ of Example 3) and do not affect its dimension. Values given to $\eta, \alpha, \beta, \gamma, \varepsilon$ are illustrated in Sect. 3.2.

Table 2 Time complexities of this paper and the standard method

N	Standard $O((NF)^3)$ method				This paper $O(NF^3)$ method			
	$p = 4$, $\mu = 125$	$p = 4$, $\mu = 1250$	$p = 5$, $\mu = 125$	$p = 6$, $\mu = 125$	$p = 4$, $\mu = 125$	$p = 4$, $\mu = 1250$	$p = 5$, $\mu = 125$	$p = 6$, $\mu = 125$
30	0.453	0.469	1.953	9.336	0.078	0.079	0.469	3.367
50	1.25	1.312	5.938	32.84	0.109	0.109	0.813	5.814
80	3.298	3.310	19.297	na	0.171	0.172	1.344	9.57
130	8.438	8.704	31.859	na	0.281	0.281	1.687	15.835
200	20.952	21.422	na	na	0.438	0.439	3.422	24.735

The low time complexity of this paper method combines with a similarly low space complexity as discussed next. Such a complexity is also shown to be independent of N.

2. Space complexity The state space cardinality is of order $O(NF)$, as already stated in relation to (3). On the other hand it is easy to be convinced that only number $O(pF) \ll O(NF)$ states are to be generated. In fact, subsets $E(0), \ldots, E(N-p)$ appearing in (6) are each of cardinality $(F+1)$ due to (4) and (5), and only $O(F)$ states are to be individually generated. Such a number combined with the $O(L)$ states in E^* gives $O(F+L)$ that is easily shown to be $O(pF)$ since $L \leq (3p-2)(F+1)$. In conclusion the number of states to be actually generated for the method application only depends on p and is independent of N.

The space complexity is however lower than $O(p^2F^2)$ since only matrices $\mathbf{A}$, $\mathbf{B}$, $\mathbf{G}$, $\mathbf{H}$, and $\mathbf{L}$ are to be explicitly stored, with a resulting space requirement of order $O(pF^2)$. Indeed, matrices $\mathbf{A}$ and $\mathbf{B}$ require $O(F^2)$ in space and matrices $\mathbf{G}$, $\mathbf{H}$, $\mathbf{L}$, consist of order $O(p)$ blocks each of $O(F)$ dimension. In conclusion the space requirement of the method is $O(pF^2)$ and is independent of N. This is an additional feature of interest to the efficient QoS management process, as seen next.

3 Application to resource allocation

As introduced above, the performance model presented in this paper can be advantageously used to direct the QoS management process in wireless networks. The architecture of a wireless network is based on a set of so-called base stations and switching centers (BSSC) (Stallings 2005). Each BSSC takes care of the mobile terminals residing in its serving cell. The BSSC decides whether to accept or reject a connection request and, in the positive case, allocates bandwidth to the customer. Each BSSC is responsible for a given amount of bandwidth and allocates the bandwidth to user connection requests by segments, according to their QoS requirements.

3.1 Description of the reference system

Basing on the architecture of second generation (GSM) and third generation (UMTS) systems, each BSSC consists of the so-called Base Station (BS) and the Mobile Switching Center (MSC). It is easy to be convinced (de Nitto Personè and Iazeolla 2006) that the BS + MSC system can be conveniently modeled by the closed cyclic network in Fig. 1, where each connection request arriving to the BSSC from its serving cell, is taken care of and treated by centers C and then P, which model the BS and the MSC respectively. In more detail:

- Center C models the bandwidth allocation by segments, performed by the BS. There exist N segments, M of which give high level service, at rate η, and the remaining S give low level service, at rate α.
- Center P models the remaining activities of the BSSC, performed by the MSC unit consisting of *service to connection requests* (such as routing, link establishment, etc.) and *additional services* (such as bandwidth partition and computational services) requested by the users. The MSC activities relating to the service of connection requests (routing, link establishment, etc.) are modeled by the P center when available (i.e. not in vacation) and performed by the processors at a rate μ. The additional activities (bandwidth partition, services to the users, etc.) are modeled by the P center when in vacation (that takes place when P's queue for connection requests is empty). It is assumed that the vacation

rates are $\beta, \gamma, \varepsilon$, as described in Sect. 1.1. In other words it is assumed that the time center P spends in vacation to process the additional activities is on the average of value $\beta^{-1}, \gamma^{-1}, \varepsilon^{-1}$ (see Sect. 3.2 for example values assigned to such parameters).

All above computational tasks require the MSC be endowed with sufficiently high computing capabilities, also since (Stallings 2005) one MSC may coordinate the activities of more that one BS. Such high computing capabilities can be given to the MSC by a single special purpose high-performance processor, or by a set of off-the shelf commercial processors operating in parallel. Single special purpose high performance processors are too expensive or difficult to maintain, to be conveniently included in the stations of wireless communication systems. Such stations, indeed, are diffused in great number in various sites, generally disseminated on the territory, without specifically worrying about the climatic and the operating conditions required for the special purpose high-performance processors. A solution to this problem is to give the vacation server parallel processing capabilities. In this case a set of low-cost off-the-shelf commercial processors can be used to operate in parallel to carry on the computer-intensive tasks of the vacation station, with further obvious cost/performance and scalability advantages (Iazeolla and Marinuzzi 1989) with respect to a single high performance processor.

In addition to the adoption of low-cost processors in the MSC, the paper method also allows the adoption of low-cost MSC memory chips due to the above seen low space complexity of the evaluation method.

3.2 The efficient QoS management process

The decision of partitioning the bandwidth into $M + S$ segments, with M segments to the high users and S to the low ones, can be called bandwidth allocation policy (BAP). According to the operating conditions, the BSSC may dynamically decide to change the user service level by adopting upgrade/degradation mechanisms according to the negotiated QoS. In presence of an increase in the rate of connection requests, the BSSC may decide to accept a larger number $N' > N$ of users. A way to perform this is accommodating a smaller ($M' < M$) number of users to the high level and a larger ($S' > S$) number to the low level. To give the BSSC the intelligence of understanding how to change a given BAP into a BAP$'$, it is important to predict the effects of bandwidth redistribution on system performance. This prediction has to be done in a short time to on-line direct the QoS management process.

The low time consumptions of this paper method prove that the method can be directed to this scope. The Table 2 values are for system parameters: $\eta = 50\ \text{s}^{-1}, \alpha = 12.5\ \text{s}^{-1}$, $p = 4$, 5 or 6 parallel processors in P with $\mu = 125$ or 1250 task/s, and vacation parameters $\beta = \gamma = 2\ \text{s}^{-1}$ and $\varepsilon = 0.8\ \text{s}^{-1}$.

It is easy to be convinced that the values of the η and α parameters are for a BS (center C) that allocates an amount of 100 Mb/s bandwidth to requests from mobile terminals (de Nitto Personè and Iazeolla 2006) (for example, in the case of $N = 80$ users, 40 of them are served at high service level (segments of 2 Mb/s), while the 40 remaining ones have to adapt their application to a low level (segments of 0.5 Mb/s)).

The value of the μ parameter is for an MSC (center P) that takes an average processor time of 125^{-1} s (or else 1250^{-1} s) to give service to connection requests while available, and the values of the β, γ and ε parameters are for an MSC that takes a service time 3 to 4 orders of magnitudes larger than μ^{-1} to give additional services while in vacation.

The Table 2 times are for the on-line evaluation of the average system throughput defined as:

$$\lambda = \sum_{k=0}^{N} g(k)p(k)$$

where $g(k)$ is given by (1), or the station C throughput with k jobs in C, and $p(k)$ is the marginal steady state distribution of the population in station C.

Such a quantity can be rewritten by (1) as follows:

$$\lambda = \eta \sum_{k=0}^{M} kp(k) + M\eta \sum_{k=M+1}^{N} p(k) + \alpha \sum_{k=M+1}^{N} (k-M)p(k) = \lambda 1 + \lambda 2$$

where

$$\lambda 1 = \eta \sum_{k=0}^{M} kp(k) + M\eta \sum_{k=M+1}^{N} p(k)$$

$$\lambda 2 = \alpha \sum_{k=M+1}^{N} (k-M)p(k)$$

with $\lambda 1$ the high level throughput and $\lambda 2$ the low level one.

The computation of such performance indices requires the calculation (based on (7) and (8)) of the distribution $\{p(k), k = 0, \ldots, N\}$ for changing values of N according to different BAPs, from where the importance is evident of the low time-consumption feature of the paper method.

By use of the method, on-time predictions can be obtained of the effects of various BAPs on the system throughputs $\lambda 1, \lambda 2$ and λ to maintain the actual QoS as close as possible to the negotiated QoS, as seen in the example below.

Example 4 Assume the negotiated QoS is:

> *Accommodate at the high QoS level as many users as possible under the constraint the connection blocking probability be not superior to 0.2.*

In terms of system performance indices, such a QoS requirement becomes:

> *Maximize* $\lambda 1$ *under the constraint that the connection blocking probability be not superior to 0.2.*

The blocking probability constraint can be easily translated (de Nitto Personè and Iazeolla 2006) into a constraint on the total system throughput λ, e.g. $\lambda \geq 128$. By using the model, the BAP is soon identified that maximizes $\lambda 1$ subject to the constraint on λ, and it is found that $M = 37$ *highs*, $S = 50$ *lows* is the BAP to set into operation.

Now assume that an increase in the rate of connection requests takes place. From the model predictions one can derive the new BAP that satisfies the max $\lambda 1$ requirement compatible with the new constraint on λ (e.g. $\lambda \geq 144$). In this case, it is easily found that $M' = 31$ *highs*, $S' = 74$ *lows* is the new BAP to set into operation, that means that the BSSC has to adopt a degradation mechanism, by moving $37 - 31 = 6$ highs to the low level. The resulting released bandwidth will be used to accept additional $74 - 50 - 6 = 18$ users at this level, thus minimizing the connection blocking probability.

4 Conclusions

The conventional two-station model with vacation of the Machine Interference Problem has been improved to deal with parallelism in the vacation station and low time and low space requirements in the evaluation technique. This makes the model of effective use when the vacation station carries on computing intensive tasks and in which case an efficient evaluation technique is needed to direct in real time the system decision process. This is shown to be the case of the QoS management in wireless networks.

The underlying Markov process of the two-station closed network with parallelism and station vacation has been studied, and an order $O(NF^3)$ solution method has been introduced, with N the number of customers and F a quantity depending on the parallelism degree. Numerical experiments are included to compare the paper method with the standard $O((NF)^3)$ method. The reduced space complexity of the method is also illustrated. An example of resource allocation in communication systems with application to the QoS management of wireless systems has been presented.

Appendix 1

The semantics of the diagonal and side-diagonal blocks $\mathbf{S}_{k,k}$, $\mathbf{S}_{k,k-1}$ and $\mathbf{S}_{k,k+1}$ is here illustrated by considering a few example transitions between states.

Let us for example consider state $\sigma = [\{(2,1),(2,1),(0,1)\},1,1]$ of subspace $E(1)$ in Example 2. State σ says that 1 job is in center C and thus the remaining $5-1=4$ jobs are in P, each split into $p(=3)$ tasks.

Denote by g the job in center C and by a,b,c,d the ones in P. According to σ, jobs a and b are in state $(2,1)$ which means that each has 2 tasks in J and 1 task in the processors of P. Job c is in state $(0,1)$ which means it has no tasks in J, has 1 task in one processor of P, and its remaining two tasks are in the task queue. Job d has all its 3 tasks in the task queue.

Job a (and the same can be said of b) is in the situation that 2 of its tasks have already been served and stay in J, waiting for their sibling still in service in P. As soon as this service ends, all three tasks of job a are completed and job a can be routed back to center C. The released processor is then taken by one of the waiting tasks of job c.

As a result, a transition takes place from state $\sigma = [\{(2,1),(2,1),(0,1)\},1,1]$ to state $\sigma' = [\{(2,1),(0,2)\},2,1]$ of $E(2)$ with rate 2μ. Such a transition belongs to matrix $\mathbf{S}_{1,2}$ (of $\mathbf{S}_{k,k+1}$ type in Theorem 1).

Starting from the original state $\sigma = [\{(2,1),(2,1),(0,1)\},1,1]$ now instead assume that the service ends in P of the task belonging to job c. The task enters the J node and the released processor is taken by one of the waiting tasks of job c. In this case a transition takes place from state σ to state $\sigma'' = [\{(2,1),(2,1),(1,1)\},1,1]$ of $E(1)$ at rate μ. Such a transition belongs to matrix $\mathbf{S}_{1,1}$ (of $\mathbf{S}_{k,k}$ type in Theorem 1).

Finally, assume that, starting from the original state σ, the service ends of job g in center C. The job moves to center P, and the fork node F splits it into $p(=3)$ tasks, which are enqueued in the task queue. In this case a transition takes place from state σ to state $\sigma''' = [\{(2,1),(2,1),(0,1)\},0,1]$ of $E(0)$ at rate η. Such a transition belongs to matrix $\mathbf{S}_{1,0}$ (of $\mathbf{S}_{k,k-1}$ type in Theorem 1).

Appendix 2

Proof of Theorem 2 Equation (8) of the matrix-analytic form will be proved by induction, while (7) by construction.

First note that in the considered model, the generator matrix $\mathbf{S}$ is level dependent and as a consequence equations (8) and matrix $\mathbf{C}(j)$ are level dependent, too.

By definition, the sub matrices of $\mathbf{C}(j)$ can be written as:

$$\begin{aligned}
&\mathbf{C}_{12}(j) = \mathbf{C}_{11}(j-1), \qquad \mathbf{C}_{22}(j) = \mathbf{C}_{21}(j-1)\\
&\mathbf{C}_{11}(j) = -g(j+2)^{-1}\mathbf{C}_{11}(j-1)(\mathbf{B} - g(j+1)\mathbf{I}_F) - g(j+2)^{-1}\mathbf{C}_{12}(j-1)\mathbf{A}\\
&\mathbf{C}_{21}(j) = -g(j+2)^{-1}\mathbf{C}_{21}(j-1)(\mathbf{B} - g(j+1)\mathbf{I}_F) - g(j+2)^{-1}\mathbf{C}_{22}(j-1)\mathbf{A}
\end{aligned} \tag{9}$$

By using the rearranged vector $\boldsymbol{\pi}$ according to the state space partition (6) and the generator matrix $\mathbf{S}$ notation of Fig. 3, vector $\boldsymbol{\pi}_2$ can be easily derived from system (2) and written:

$$\boldsymbol{\pi}_2 = -g(2)^{-1}\boldsymbol{\pi}_0\mathbf{A} - g(2)^{-1}\boldsymbol{\pi}_1(\mathbf{B} - g(1)\mathbf{I}_F) \tag{10}$$

By noting that, from the definition of matrix $\mathbf{C}(0)$, one can write $\mathbf{C}_{11}(0) = \mathbf{I}$ and $\mathbf{C}_{21}(0) = \mathbf{0}$, one can easily see that (10) corresponds to (8) for $k = 2$.

Let the thesis be true up to $k-1$. The proof for k can be completed by writing from system (2)

$$\boldsymbol{\pi}_k = -g(k)^{-1}\boldsymbol{\pi}_{k-2}\mathbf{A} - g(k)^{-1}\boldsymbol{\pi}_{k-1}(\mathbf{B} - g(k-1)\mathbf{I}_F) \tag{11}$$

and by substituting the values of $\boldsymbol{\pi}_{k-2}$ and $\boldsymbol{\pi}_{k-1}$ obtained from (8) by the induction hypothesis.

Let us now prove the boundary (7) by construction. From system (2), the boundary equations are:

$$\begin{aligned}
\boldsymbol{\pi}_0\mathbf{B} + g(1)\boldsymbol{\pi}_1 &= \mathbf{0}\\
\boldsymbol{\pi}_{N-p-1}\mathbf{A} + \boldsymbol{\pi}_{N-p}(\mathbf{B} - g(N-p)\mathbf{I}_F) + \boldsymbol{\pi}^*\mathbf{H} &= \mathbf{0}\\
\boldsymbol{\pi}_{N-p}\mathbf{G} + \boldsymbol{\pi}^*\mathbf{L} &= \mathbf{0}
\end{aligned}$$

By substituting the values of $\boldsymbol{\pi}_{N-p-1}$ and $\boldsymbol{\pi}_{N-p}$ obtained by (8), the boundary equations become:

$$\begin{aligned}
&\boldsymbol{\pi}_0\mathbf{B} + g(1)\boldsymbol{\pi}_1 = \mathbf{0}\\
&-g(2)^{-1}\boldsymbol{\pi}_0\mathbf{A}(\mathbf{C}_{11}(N-p-3)\mathbf{A} + \mathbf{C}_{11}(N-p-2)(\mathbf{B} - g(N-p)\mathbf{I}_F))\\
&\quad + \boldsymbol{\pi}_1((\mathbf{C}_{21}(N-p-3) - g(2)^{-1}(\mathbf{B} - g(1)\mathbf{I}_F)\mathbf{C}_{11}(N-p-3))\mathbf{A}\\
&\quad + (\mathbf{C}_{21}(N-p-2) - g(2)^{-1}(\mathbf{B} - g(1)\mathbf{I}_F)\mathbf{C}_{11}(N-p-2))(\mathbf{B} - g(N-p)\mathbf{I}_F))\\
&\quad + \boldsymbol{\pi}^*\mathbf{H} = \mathbf{0}\\
&-g(2)^{-1}\boldsymbol{\pi}_0\mathbf{A}\mathbf{C}_{11}(N-p-2)\mathbf{G} + \boldsymbol{\pi}_1(\mathbf{C}_{21}(N-p-2)\\
&\quad - g(2)^{-1}(\mathbf{B} - g(1)\mathbf{I}_F)\mathbf{C}_{11}(N-p-2))\mathbf{G} + \boldsymbol{\pi}^*\mathbf{L} = \mathbf{0}
\end{aligned} \tag{12}$$

In matrix form, system (12) can be written as:

$$[\boldsymbol{\pi}_0, \boldsymbol{\pi}_1, \boldsymbol{\pi}^*]\mathbf{R} = \mathbf{0} \tag{13}$$

where matrix $\mathbf{R}$ is the block coefficient matrix of system (12):

$$\mathbf{R} = \begin{bmatrix} \mathbf{B} & \mathbf{R}_{12} & \mathbf{R}_{13} \\ g(1)\mathbf{I}_F & \mathbf{R}_{22} & \mathbf{R}_{23} \\ \mathbf{0} & \mathbf{H} & \mathbf{L} \end{bmatrix}$$

with

$$\begin{aligned}
\mathbf{R}_{12} &= -g(2)^{-1}\mathbf{A}(\mathbf{C}_{11}(N-p-3)\mathbf{A} + \mathbf{C}_{11}(N-p-2)(\mathbf{B} - g(N-p)\mathbf{I}_F)) \\
\mathbf{R}_{13} &= -g(2)^{-1}\mathbf{A}\mathbf{C}_{11}(N-p-2)\mathbf{G} \\
\mathbf{R}_{22} &= ((\mathbf{C}_{21}(N-p-3) - g(2)^{-1}(\mathbf{B} - g(1)\mathbf{I}_F)\mathbf{C}_{11}(N-p-3))\mathbf{A} \\
&\quad + (\mathbf{C}_{21}(N-p-2) - g(2)^{-1}(\mathbf{B} - g(1)\mathbf{I}_F)\mathbf{C}_{11}(N-p-2))(\mathbf{B} - g(N-p)\mathbf{I}_F)) \\
\mathbf{R}_{23} &= (\mathbf{C}_{21}(N-p-2) - g(2)^{-1}(\mathbf{B} - g(1)\mathbf{I}_F)\mathbf{C}_{11}(N-p-2))\mathbf{G}
\end{aligned}$$

The normalization condition for system (2) is $\sum_{k=0}^{N-p} \boldsymbol{\pi}_k\mathbf{e} + \boldsymbol{\pi}^*\mathbf{e} = 1$, where $\mathbf{e}$ is the column unit vector. By using (8) the normalization condition becomes:

$$\begin{aligned}
&\boldsymbol{\pi}_0\left(\mathbf{I} - g(2)^{-1}\mathbf{A}\sum_{k=2}^{N-p}\mathbf{C}_{11}(k-2)\right)\mathbf{e} \\
&\quad + \boldsymbol{\pi}_1\left(\mathbf{I} + \sum_{k=2}^{N-p}(\mathbf{C}_{21}(k-2) - g(2)^{-1}(\mathbf{B} - g(1)\mathbf{I}_F)\mathbf{C}_{11}(k-2))\right)\mathbf{e} + \boldsymbol{\pi}^*\mathbf{e} = 1
\end{aligned} \tag{14}$$

Hence, we can replace one column in matrix $\mathbf{R}$ with the column that represents the normalization condition given by (14), and appropriately change the right-hand side of (13). For instance, by replacing the first column of $\mathbf{R}$, one obtains:

$$[\boldsymbol{\pi}_0, \boldsymbol{\pi}_1, \boldsymbol{\pi}^*]\hat{\mathbf{R}} = [1, \mathbf{0}]$$

where $[1, \mathbf{0}]$ is a row vector consisting of a 1 and a zero sub vector of length $2F + L + 1$, and $\hat{\mathbf{R}}$ is obtained from $\mathbf{R}$ by replacing the first column of the block column $[\mathbf{B}, g(1)\mathbf{I}_F, \mathbf{0}]$ with the column vector:

$$\begin{bmatrix} (\mathbf{I} - g(2)^{-1}\mathbf{A}\sum_{k=2}^{N-p}\mathbf{C}_{11}(k-2))\mathbf{e} \\ (\mathbf{I} + \sum_{k=2}^{N-p}(\mathbf{C}_{21}(k-2) - g(2)^{-1}(\mathbf{B} - g(1)\mathbf{I}_F)\mathbf{C}_{11}(k-2)))\mathbf{e} \\ \mathbf{e} \end{bmatrix} \tag{15}$$

That completes the proof. □

References

Chao, X., & Zhao, Y. Q. (1998). Analysis of multi-server queues with station and server vacations. *European Journal of Operational Research*, *110*, 392–406.

de Nitto Personè, V., & Grassi, V. (1996). Solution of finite QBD processes. *Journal of Applied Probability*, *33*(4), 1003–1010.

de Nitto Personè, V., & Iazeolla, G. (2006). QoS Management in Wireless Networks. In *SPECTS'06, international symposium on performance evaluation of computer and telecommunication systems*, Calgary, Canada.

Fiems, D., Steyaert, B., & Bruneel, H. (2001). Performance evaluation of CAI and RAI transmission modes in a GI-G-1 queue. *Computers & Operations Research*, *28*, 1299–1313.

Frigui, I., & Alfa, A. S. (1998). Analysis of a time-limited polling system. *Computer Communications*, *21*, 558–571.

Grassman, W. K., & Heyman, D. P. (1990). Equilibrium distribution of block-structured Markov chains with repeating rows. *Journal of Applied Probability*, *27*, 557–576.

Gupta, S. M. (1997). Machine interference problem with spares, server vacations and exhaustive service. *Performance Evaluation*, *29*, 195–211.

Gupta, U. C., & Sikdar, K. (2004). The finite-buffer $M/G/1$ queue with general bulk-service rule and single vacation. *Performance Evaluation*, *57*, 199–219.

Iazeolla, G., & Marinuzzi, F. (1989). A few remarks on cost/performance relationships in parallel computer architectures. In *ESM 89, European simulation multiconference*, Rome, Italy.

Le Boudec, J. Y. (1989). *A generalization of matrix geometric solutions for Markov models*. IBM Research Division Zurich, Res. Rep. RZ 1903 (#67353).

Lee, Y., & Choi, B. D. (2001). Queueing system with multiple delay and loss priorities for ATM networks. *Information Sciences*, *138*, 7–29.

Mehmet-Ali, M., Zhang, X., & Hayes, J. F. (2003). A performance analysis of a discrete-time queueing system with server interruption for modeling wireless ATM multiplexer. *Performance Evaluation*, *51*, 1–31.

Servi, L. D., & Finn, S. G. (2002). $M/M/1$ queues with working vacations ($M/M/1/WV$). *Performance Evaluation*, *50*, 41–52.

Sevcik, K. (2005). The origin of queueing network models. *Ubiquity*, *6*(3).

Stallings, W. (2005). *Wireless communications & networks*. Upper Saddle River, NJ: Prentice Hall.

Takagi, H. (1991). *Queueing analysis—a foundation of performance evaluation. Vacation and Priority Systems* (Part I, Vol. I). Amsterdam: Elsevier.

Xu, E., & Alfa, A. S. (2002). A vacation model for the non-satured Readers and Writers system with a threshold policy. *Performance Evaluation*, *50*, 233–244.

Ye, J., & Li, S. Q. (1994). Folding algorithm: a computational method for finite QBD processes with level-dependent transitions. *IEEE Transactions on Communications*, *42*(2/3/4), 625–639.

Zhang, Z. G., & Tian, N. (2003). Analysis on queueing systems with synchronous vacations of partial servers. *Performance Evaluation*, *52*, 269–282.