Centrum voor Wiskunde en Informatica

REPORTRAPPORT

# Analysis of multiple-server polling systems by means of the power-series algorithm

R.D. van der Mei, S.C. Borst

Department of Operations Research, Statistics, and System Theory

**Report BS-R9410 March 1994**

CWI is the National Research Institute for Mathematics and Computer Science. CWI is part of the Stichting Mathematisch Centrum (SMC), the Dutch foundation for promotion of mathematics and computer science and their applications.
SMC is sponsored by the Netherlands Organization for Scientific Research (NWO). CWI is a member of ERCIM, the European Research Consortium for Informatics and Mathematics.


P.O. Box 94079, 1090 GB Amsterdam (NL)
Kruislaan 413, 1098 SJ Amsterdam (NL)
Telephone +31 20 592 9333
Telefax +31 20 592 4199

# Analysis of Multiple-Server Polling Systems by Means of the Power-Series Algorithm

R.D. van der Mei
*Tilburg University*
*P.O. Box 90153, 5000 LE Tilburg, The Netherlands*

S.C. Borst
*CWI*
*P.O. Box 94079, 1090 GB Amsterdam, The Netherlands*

**Abstract**

We consider a polling model with multiple servers in which each server visits the queues according to a general service order table. In general, such models are very hard to treat by means of analytical techniques. In this paper we show how the model can be analyzed with the aid of the so-called power-series algorithm (PSA), a tool for the numerical evaluation and optimization of the performance of a broad class of multi-queue models. Numerical experiments with the PSA are performed to investigate the tendency for the servers to cluster, to address the option of partitioning, and to make some comparisons with single-server systems carrying a comparable load.

*AMS Subject Classification (1991):* 60K25, 68M20.
*Keywords & Phrases:* Bernoulli service, polling system, multiple servers, waiting time, queue length, power-series algorithm.

## 1 Introduction

A multiple-server polling system is a multiple-queue system attended to by a number of servers, which visit the queues in some order. Like the single-server versions, multiple-server polling systems find many applications in computer systems, communication networks, and manufacturing environments. So far there are hardly any exact results known for these systems, apart from some mean value results for global performance measures like cycle times and intervisit times. In the model under consideration the order in which each of the servers visits the queues is prescribed by a fixed service order table (polling table). At each of the queues the servers operate according to the so-called Bernoulli service strategy. In this paper we show how the power-series algorithm (PSA) may be used to determine the joint distribution of the queue lengths and the positions of the servers in the system. From this joint distribution also relevant performance measures like the mean waiting times of customers and utilization factors of the individual servers may be obtained. In the remainder of the introduction we successively discuss some applications, present an overview of related literature, and describe the organization of the paper.

Report BS-R9410
ISSN 0924-0659
CWI
P.O. Box 94079, 1090 GB Amsterdam, The Netherlands

*Applications*
An example of a multiple-server polling system is a distributed system, consisting of a number of computers, interconnected by a communication medium, that cooperate as follows in sharing the total load of the system, cf. [27]. The jobs entering the 'front-end' systems (corresponding to the queues) are picked up in batches by the 'back-end' systems (corresponding to the servers) according to some cyclic schedule. As soon as a batch is served, the back-end system picks up the jobs from the next front-end system. When the communication delay is negligible, it may be preferable to use a more sophisticated strategy which keeps track of the evolution of the system, like "serve the longest queue". However, when the communication overhead is significant, it makes sense to use a cyclic service schedule which requires only very little processing information.
Examples of multiple-server polling systems also arise in communication networks, like the underlying communication medium in the above-mentioned distributed system. Consider e.g. a Local Area Network (LAN), consisting of a number of stations, interconnected by a transmission ring. There are various protocols known for the medium access control in a LAN with a ring architecture. One variant is the multiple slotted ring, i.e., the ring is subdivided into time slots of the size of a single message, circulating at constant speed. Occupying a slot corresponds to utilizing a server. Another medium access variant that may lead to multiple-server polling is the multiple token ring, i.e., there are multiple rings, each with a token circulating on it, representing the right of transmission on that particular ring. Holding a token corresponds to utilizing a server.
Other examples arise in manufacturing environments. In flexible manufacturing systems, e.g., one finds a number of machines that periodically change over from one type of operations to another. In some factories internal transport is provided by vehicles that follow a track along loading/unloading points, which conceptually comes very close to a slotted ring with destination release. A multiple-elevator facility is yet another example, although some features, like the acceleration effect when a floor is skipped, are not incorporated in the classical description of a polling model, cf. [19].

*Related literature*
Multiple-server polling systems have received remarkably little attention in the vast literature on polling systems. One of the first studies is Morris & Wang [27] in which the servers are assumed to be independent, i.e., to visit the queues independently of each other, each server according to some cyclic schedule. They obtain the mean cycle time of each server and the mean intervisit time to a queue, and derive approximate expressions for the mean sojourn time for both a gated-type and a limited-type service discipline. A very interesting phenomenon observed in [27] is the tendency for the servers to cluster if they follow identical routes, especially in heavy traffic. Numerical experiments indicate that the bunching of servers is likely to deteriorate the system performance. The bunching of servers is alleviated if they follow different routes. Therefore Morris & Wang advocate the use of 'dispersive' schedules to improve the system performance.
Browne & Weiss [17] is one of the few studies in which the servers are assumed to be coupled, i.e., to visit the queues together. They obtain index rules for the minimization of the mean length of individual cycles for both the exhaustive and the gated service discipline. Browne et al. [15] derive the mean waiting time for a completely symmetric two-queue system with

an infinite number of coupled servers and deterministic service times. Browne & Kella [16] obtain the busy period distribution for a two-queue system with an infinite number of coupled servers, exhaustive service, and deterministic service times at one queue and general service times at the other queue. Levy & Yechiali [24] and Kao & Narayanan [21] study the joint distribution of the queue length and the number of busy servers for a Markovian multiple-server queue, where the servers individually go on vacation when there are no waiting customers left. Mitrany & Avi-Itzhak [26] and Neuts & Lucantoni [28] analyze the joint distribution of the queue length and the number of busy servers for a Markovian multiple-server queue where servers break down at exponential intervals and then get repaired.

In references [7], [20], [22], [29], [31] mean response time approximations are developed to analyze the performance of LAN's with multiple-token rings. Mean response time approximations oriented to LAN's with a multiple slotted ring are contained in references [5], [7], [25], [31], [32]. Ajmone Marsan et al. [2], [3], [4] derive the mean cycle time and bounds for the mean waiting times in symmetric systems for the exhaustive, gated, and 1-limited service discipline. In [1] they illustrate how PETRI-net techniques may be used to study Markovian multiple-server polling systems.

The above-mentioned studies unanimously point out that multiple-server polling systems, combining the complexity of single-server polling systems and multiple-server systems, are extraordinarily hard to analyze. In fact none of the studies (except [15], [16] and the single-queue studies [21], [24], [26], [28]) presents any exact results, apart from some mean value results for global performance measures like cycle times and intervisit times.

Borst [12] explores the class of systems that allow an exact analysis in the case of *coupled* servers. The class in question includes most single-queue systems, two-queue two-server systems with exhaustive service and exponential service times, as well as infinite server systems with an arbitrary number of queues, exhaustive or gated service, and deterministic service times. In the case of *independent* servers the class of systems that allow an exact analysis is even smaller. In fact, to the best of the authors' knowledge, there is not a single non-trivial multiple-queue system with multiple independent servers for which any exact expressions for the waiting times are known. In the present paper we show however how a broad class of systems may still be analyzed numerically by means of the PSA. Benefiting from the insights provided by numerical experiments with the PSA, we present in a companion paper [13] waiting-time approximations for systems with multiple independent servers.

*Organization of the paper*

In Section 2 we present a detailed model description. In Section 3 we describe how the evolution of the system may be modeled as a continuous-time Markov process and we give the global balance equations for the model. In Section 4 the state probabilities are expressed as power series in the load of the system in light traffic. Then, we derive a complete computation scheme to determine the coefficients of these power series. The complexity of the PSA for the present model is discussed in Section 5. Finally, in Section 6 we give an extensive overview of the numerical results that we gathered. We investigate the tendency for the servers to cluster and the option of partitioning the system into a number of separate subsystems, each attended to by a single server. Further some comparisons are made with single-server systems carrying a comparable load.

## 2 Model description

Consider a model consisting of $n$ queues $Q_1, \ldots, Q_n$, each of infinite buffer capacity, attended to by $m$ servers $S_1, \ldots, S_m$. Customers arrive at $Q_i$ according to a Poisson process with rate $\lambda_i$ and are referred to as type-$i$ customers, $i = 1, \ldots, n$. The service times of type-$i$ customers are exponentially distributed with mean $\beta_i = 1/\mu_i$, $i = 1, \ldots, n$. Denote $\boldsymbol{\beta}^{(1)} = (\beta_1^{(1)}, \ldots, \beta_n^{(1)})$. Each server $S_j$ visits the queues periodically according to a fixed service order table $\boldsymbol{\pi}_j = (\pi_j(1), \ldots, \pi_j(L_j))$, where $L_j$ is the (finite) length of the service order table (also referred to as a polling table) of $S_j$; that is, the $l$-th queue visited by $S_j$ is $\pi_j((l-1) \mathrm{mod} L_j)+1)$, $l = 1, 2, \ldots$, $j = 1, \ldots, m$. Define $\Pi_j := \{\pi_j(1), \ldots, \pi_j(L_j)\}$ to be the index set of the queues visited by $S_j$, $j = 1, \ldots, m$. Note that the queues are not necessarily visited by each of the servers. The switch-over times needed by the servers to move from $Q_i$ to $Q_k$ are exponentially distributed with mean $1/\nu_{i,k}$, $i, k = 1, \ldots, n$. The servers are assumed to visit the queues independently of each other, under the restriction that at most $m_i$ servers may visit $Q_i$ simultaneously. An arrival of a server $S_j$ ($j = 1, \ldots, m$) at $Q_i$ will be called effective if $S_j$ finds less than $m_i$ other servers working at $Q_i$ and there are customers waiting at $Q_i$ (so that $S_j$ may start serving at $Q_i$). The service of a customer can not be interrupted. The number of customers that is served during one visit of a server $S_j$ ($j = 1, \ldots, m$) at $Q_i$ is determined by a so-called Bernoulli service discipline, which works as follows. If an arrival of $S_j$ at $Q_i$ is effective, then $S_j$ serves at least one customer at $Q_i$; otherwise, $S_j$ immediately starts to move to the next queue. Moreover, if after a service completion of $S_j$ at $Q_i$ there are still customers waiting at $Q_i$, then with probability $q_i$ ($0 \leq q_i \leq 1$) $S_j$ serves another customer at $Q_i$, $i = 1, \ldots, n$; otherwise, $S_j$ proceeds to the next queue according to its service order table. It should be noted that in the case $q_i = 1$ a server only departs from $Q_i$ (after an effective arrival at $Q_i$) when there are no waiting customers left at $Q_i$ (exhaustive service) and that in the case $q_i = 0$ a server serves at most one customer at $Q_i$ during a visit to $Q_i$ (1-limited service), $i = 1, \ldots, n$. At each queue the queueing disciplines may be general, but may not depend on the actual service times. All service times, switch-over times, and interarrival times are assumed to be mutually independent and independent of the state of the system.

We define the traffic intensity at $Q_i$, $\rho_i$, and the total traffic intensity, $\rho$, by

$$\rho_i := \lambda_i \beta_i, \; i = 1, \ldots, n, \; \rho := \sum_{i=1}^{n} \rho_i. \tag{2.1}$$

In the PSA $\rho$ will be used as a variable. Therefore we define

$$a_i := \lambda_i / \rho, \; i = 1, \ldots, n. \tag{2.2}$$

Finally some words on the stability conditions. Denote by $s_j$ the mean total switch-over time incurred by $S_j$ in a cycle. Denote by $k_{ij}$ the total number of visits paid to $Q_i$ by $S_j$ in a cycle. For $m = 1$, the single-server case, necessary and sufficient conditions are $\rho + \lambda_i s_1 (1 - q_i)/k_{i1} < 1$, $i = 1, \ldots, n$, cf. Fricker & Jaïbi [18] for a rigorous proof. For $m > 1$, the multiple-server case, the stability conditions are not generally known. Evidently, necessary conditions are that $\rho_i < m_i$, $i = 1, \ldots, n$, and that for each set $I \subseteq \{1, \ldots, n\}$ the indices $i \in I$ occur in the polling table of at least $\sum_{i \in I} \rho_i$ servers (in particular for $I = \{1, \ldots, n\}$ implying $\rho < m$). We conjecture that these conditions are in fact also sufficient for $q_i = 1$, $i = 1, \ldots, n$, i.e., for the exhaustive service discipline (as well as for other service disciplines that do not impose any (probabilistic) restriction on the maximum number of customers served during a visit). For $q_i < 1$, when the mean maximum number of customers served during a visit to $Q_i$ is $1/(1-q_i)$, it is considerably harder to find the stability conditions. When $m_i = m$, $s_j = s$, $k_{ij} = 1$, $i = 1, \ldots, n$, $j = 1, \ldots, m$, simple balancing arguments suggest that necessary and sufficient conditions are $\rho + \lambda_i s(1-q_i) < m$, $i = 1, \ldots, n$, but in other cases with $q_i < 1$ and $m_i < m$, $s_j \neq s$, or $k_{ij} \neq 1$, the problem of establishing the stability conditions appears to be completely open. Although they are not generally known, throughout the paper the stability conditions are simply assumed to hold.

## 3 The balance equations

In this section we describe how the evolution of the system under consideration may be modeled as a continuous-time Markov process. Subsequently, we derive the balance equations for the Markov process in question. In the next section we show how the PSA may be used to solve these balance equations.

We first introduce some notation. Let $N_i(t)$ be the number of customers at $Q_i$ (including customers in service) at time $t$, $i = 1, \ldots, n$. Denote $\mathbf{N}(t) = (N_1(t), \ldots, N_n(t))$. Evidently, the joint queue length process $\{\mathbf{N}(t), t \geq 0\}$ itself is not a Markov process, as the transitions also depend on the status of the servers. So, to extend the joint queue length process to a Markov process, we need to introduce some supplementary variables describing the status of the servers. Let $H_j(t)$ be the actual entry in the polling table of $S_j$ at time $t$, $j = 1, \ldots, m$. Let $Z_j(t)$ indicate whether $S_j$ is switching ($Z_j(t) = 0$) or serving ($Z_j(t) = 1$) at time $t$, $j = 1, \ldots, m$. So, if $(H_j(t), Z_j(t)) = (l, 0)$ then $S_j$ is switching to queue $\pi_j(l)$ at time $t$; if $(H_j(t), Z_j(t)) = (l, 1)$ then $S_j$ is serving at queue $\pi_j(l)$ at time $t$. Denote $\mathbf{H}(t) = (H_1(t), \ldots, H_m(t))$, $\mathbf{Z}(t) = (Z_1(t), \ldots, Z_m(t))$. Define the supplementary space by $\Theta = \Theta_1 \times \Theta_2$, where

$$\Theta_1 = \{\mathbf{h} = (h_1, \ldots, h_m) : h_j \in \{1, \ldots, L_j\}, j = 1, \ldots, m\}, \tag{3.1}$$

$$\Theta_2 = \{\mathbf{z} = (z_1, \ldots, z_m) : z_j \in \{0, 1\}, j = 1, \ldots, m\}, \tag{3.2}$$

with $L_j$ the length of the polling table of $S_j$, $j = 1, \ldots, m$. Then it is easily verified that the stochastic process $\{(\mathbf{N}(t), \mathbf{H}(t), \mathbf{Z}(t)), t \geq 0\}$ is a continuous-time Markov process with state space $\mathbb{N}^n \times \Theta$. We now derive the balance equations for the process $\{(\mathbf{N}(t), \mathbf{H}(t), \mathbf{Z}(t)), t \geq 0\}$. Denote by $(\mathbf{N}, \mathbf{H}, \mathbf{Z})$ stochastic variables with as joint distribution the joint stationary distribution of $(\mathbf{N}(t), \mathbf{H}(t), \mathbf{Z}(t))$. For each state $(\mathbf{n}, \mathbf{h}, \mathbf{z}) \in \mathbb{N}^n \times \Theta$ denote the number of servers working at $Q_i$ by

$$x_i(\mathbf{h}, \mathbf{z}) := \mathrm{card}(\{j : (\pi_j(h_j), z_j) = (i, 1)\}), \tag{3.3}$$

where $\mathrm{card}(A)$ stands for cardinality of the set $A$. The state probabilities are defined as follows: for $(\mathbf{n}, \mathbf{h}, \mathbf{z}) \in \mathbb{N}^n \times \Theta$,

$$p(\mathbf{n}, \mathbf{h}, \mathbf{z}) = \Pr\{(\mathbf{N}, \mathbf{H}, \mathbf{Z}) = (\mathbf{n}, \mathbf{h}, \mathbf{z})\}. \tag{3.4}$$

Because the number of servers which may be working at $Q_i$ simultaneously is bounded by $m_i$ and by $n_i$, we have: for $(\mathbf{n}, \mathbf{h}, \mathbf{z}) \in \mathbb{N}^n \times \Theta$,

$$p(\mathbf{n}, \mathbf{h}, \mathbf{z}) = 0 \text{ if there is an } i \ (i = 1, \ldots, n) \text{ such that } x_i(\mathbf{h}, \mathbf{z}) > \min\{n_i, m_i\}. \tag{3.5}$$

The global balance equations for the present model read as follows: for $(\mathbf{n},\mathbf{h},\mathbf{z}) \in \mathbb{N}^n \times \Theta$, with $x_i(\mathbf{h},\mathbf{z}) \leq \min\{n_i, m_i\}$, $i = 1,\ldots,n$,

$$\begin{aligned}
&\left[\sum_{i=1}^{n}\lambda_i + \sum_{j=1}^{m}\nu_{\pi_j(h_j-1),\pi_j(h_j)}\mathrm{I}_{\{z_j=0\}} + \sum_{j=1}^{m}\mu_{\pi_j(h_j)}\mathrm{I}_{\{z_j=1\}}\right] p(\mathbf{n},\mathbf{h},\mathbf{z}) = \qquad (3.6)\\
&\sum_{i=1}^{n}\lambda_i p(\mathbf{n}-\mathbf{e}_i,\mathbf{h},\mathbf{z})\mathrm{I}_{\{n_i>0\}}\\
+&\sum_{j=1}^{m}\mu_{\pi_j(h_j)}p(\mathbf{n}+\mathbf{e}_{\pi_j(h_j)},\mathbf{h},\mathbf{z})q_{\pi_j(h_j)}\mathrm{I}_{\{z_j=1\}}\\
+&\sum_{j=1}^{m}\nu_{\pi_j(h_j-1),\pi_j(h_j)}p(\mathbf{n},\mathbf{h},\mathbf{z}-\mathbf{e}_j)\mathrm{I}_{\{z_j=1\}}\\
+&\sum_{j=1}^{m}\mu_{\pi_j(h_j-1)}p(\mathbf{n}+\mathbf{e}_{\pi_j(h_j-1)},\mathbf{h}-\mathbf{e}_j,\mathbf{z}+\mathbf{e}_j)\\
&\qquad\left[1 - q_{\pi_j(h_j-1)}\mathrm{I}_{\{x_{\pi_j(h_j-1)}(\mathbf{h},\mathbf{z})<n_{\pi_j(h_j-1)}\}}\right]\mathrm{I}_{\{z_j=0\}}\\
+&\sum_{j=1}^{m}\nu_{\pi_j(h_j-2),\pi_j(h_j-1)}p(\mathbf{n},\mathbf{h}-\mathbf{e}_j,\mathbf{z})\mathrm{I}_{\{x_{\pi_j(h_j-1)}(\mathbf{h},\mathbf{z})=\min\{n_{\pi_j(h_j-1)},m_{\pi_j(h_j-1)}\}\}}\mathrm{I}_{\{z_j=0\}},
\end{aligned}$$

where $\mathrm{I}_{\{E\}}$ stands for the indicator function of the event $E$ and where $\mathbf{e}_j$ is the $j$-th unit vector, $j = 1,\ldots,\max\{n,m\}$. The first term at the right-hand side of (3.6) indicates an arrival at $Q_i$; the second term indicates that $S_j$ starts to serve another customer at queue $\pi_j(h_j)$ after a service completion at that queue; the third term corresponds to an effective arrival of $S_j$ at queue $\pi_j(h_j)$ and a subsequent service initiation at that queue; the fourth term indicates that $S_j$ proceeds to the next queue after having completed a service at queue $\pi_j(h_j-1)$; the fifth term indicates an arrival of $S_j$ at queue number $\pi_j(h_j-1)$ which is not effective, either because there are no customers waiting at that queue or because the maximal allowable number of servers is already working at that queue.
In addition, the law of total probability implies

$$\sum_{(\mathbf{n},\mathbf{h},\mathbf{z})\in\mathbb{N}^n\times\Theta} p(\mathbf{n},\mathbf{h},\mathbf{z}) = 1. \qquad (3.7)$$

For $n > 1$ the balance equations can not be solved analytically, not even in the single-server case ($m = 1$).

**Remark 3.1** For $m = 1$ the standard analysis is oriented to the stationary distribution of the joint queue length process embedded at polling epochs rather than the continuous-time Markov process. For a wide class of service disciplines the joint queue length distribution at a polling epoch at $Q_i$ may then be related to the joint queue length distribution at the previous polling epoch at $Q_i$. Subsequently an iterative procedure yields the stationary joint queue length distribution at polling epochs. The marginal queue length distribution at $Q_i$ at an arbitrary epoch may then be recovered from the queue length distribution at $Q_i$ at a polling epoch by using results for vacation systems. As a major benefit, the approach allows generally distributed service and switch-over times. In the multiple-server case ($m > 1$) however, the approach is not likely to succeed, as even at polling epochs there are always $m-1$ other ongoing service or switch-over times. Moreover, it is not clear how the joint queue length distribution at one embedded epoch could be related to the joint queue length distribution at another embedded epoch. Neither is it clear how the marginal queue length distribution at $Q_i$ at an arbitrary epoch could be recovered from the queue length distribution at $Q_i$ at a polling epoch. So for $m > 1$ the analysis is restricted to the continuous-time Markov process. □

In the next section we show how for any number of queues and any number of servers the PSA may, in principle, be used to solve the set of global balance equations (3.6), (3.7) numerically.

## 4 Computation scheme

The basic idea of the PSA is to express the state probabilities as power series in the load in light traffic and to derive a computation scheme to calculate the coefficients of these power series. For the single-server case ($m = 1$) a complete computation scheme to calculate the coefficients has been derived in [10]. In this section, we extend this computation scheme to the multiple-server case. We emphasize that this derivation is rather technical. The reader who is not interested in the details of the derivation is advised to proceed directly to the computation scheme at the end of this section.

The approach relies on the following property: for $(\mathbf{n}, \mathbf{h}, \mathbf{z}) \in \mathbb{N}^n \times \Theta$,

$$p(\mathbf{n}, \mathbf{h}, \mathbf{z}) = O(\rho^{n_1+\ldots+n_n}), \quad \rho \downarrow 0. \tag{4.1}$$

This property can be shown to be valid under the condition that for each reachable $\mathbf{n}$, $\mathbf{n} \neq \mathbf{0}$, there is at least one positive departure rate, and that for each reachable state $(\mathbf{n}, \mathbf{h}, \mathbf{z}) \in \mathbb{N}^n \times \Theta$, $\mathbf{n} \neq \mathbf{0}$, the probability that a departure occurs before any arrival takes place, after the process has entered this state, is positive; cf. [11] for a more detailed discussion about this condition. It is readily verified that this condition is satisfied in the present model. Based on this property, we introduce the following power-series expansions of the state probabilities: for $(\mathbf{n}, \mathbf{h}, \mathbf{z}) \in \mathbb{N}^n \times \Theta$, $x_i(h, z) \leq \min\{n_i, m_i\}$, $i = 1, \ldots, n$,

$$p(\mathbf{n}, \mathbf{h}, \mathbf{z}) = \rho^{n_1+\ldots+n_n} \sum_{k=0}^{\infty} \rho^k b(k; \mathbf{n}, \mathbf{h}, \mathbf{z}). \tag{4.2}$$

We now show how a computation scheme may be derived to calculate the coefficients $b(k; \mathbf{n}, \mathbf{h}, \mathbf{z})$ of the power series. Substituting the power-series expansions (4.2) into the balance equations (3.6), using $a_i = \lambda_i/\rho$, and equating corresponding powers of $\rho$ yields the following linear relations between the coefficients of the power series: for $(k; \mathbf{n}, \mathbf{h}, \mathbf{z}) \in \mathbb{N}^{1+n} \times \Theta$ with $x_i(\mathbf{h}, \mathbf{z}) \leq \min\{n_i, m_i\}$, $i = 1, \ldots, n$,

$$\left[\sum_{j=1}^{m} \nu_{\pi_j(h_j-1),\pi_j(h_j)} \mathrm{I}_{\{z_j=0\}} + \sum_{j=1}^{m} \mu_{\pi_j(h_j)} \mathrm{I}_{\{z_j=1\}}\right] b(k; \mathbf{n}, \mathbf{h}, \mathbf{z}) = \tag{4.3}$$

$$\sum_{i=1}^{n} a_i \left[b(k; \mathbf{n} - \mathbf{e}_i, \mathbf{h}, \mathbf{z}) \mathrm{I}_{\{n_i>0\}} - b(k-1; \mathbf{n}, \mathbf{h}, \mathbf{z}) \mathrm{I}_{\{k>0\}}\right]$$

$$+ \sum_{j=1}^{m} \mu_{\pi_j(h_j)} b(k-1; \mathbf{n}+\mathbf{e}_{\pi_j(h_j)}, \mathbf{h}, \mathbf{z}) q_{\pi_j(h_j)} \mathrm{I}_{\{z_j=1\}} \mathrm{I}_{\{k>0\}}$$

$$+ \sum_{j=1}^{m} \nu_{\pi_j(h_j-1),\pi_j(h_j)} b(k; \mathbf{n}, \mathbf{h}, \mathbf{z}-\mathbf{e}_j) \mathrm{I}_{\{z_j=1\}}$$

$$+ \sum_{j=1}^{m} \mu_{\pi_j(h_j-1)} b(k-1; \mathbf{n}+\mathbf{e}_{\pi_j(h_j-1)}, \mathbf{h}-\mathbf{e}_j, \mathbf{z}+\mathbf{e}_j)$$

$$\times \left[1 - q_{\pi_j(h_j-1)} \mathrm{I}_{\{x_{\pi_j(h_j-1)}(\mathbf{h},\mathbf{z}) < n_{\pi_j(h_j-1)}\}}\right] \mathrm{I}_{\{z_j=0\}} \mathrm{I}_{\{k>0\}}$$

$$+ \sum_{j=1}^{m} \nu_{\pi_j(h_j-2),\pi_j(h_j-1)} b(k; \mathbf{n}, \mathbf{h}-\mathbf{e}_j, \mathbf{z}) \mathrm{I}_{\{x_{\pi_j(h_j-1)}(\mathbf{h},\mathbf{z}) = \min\{n_{\pi_j(h_j-1)}, m_{\pi_j(h_j-1)}\}\}} \mathrm{I}_{\{z_j=0\}}.$$

By rearranging the terms at the right-hand side, the set of equations (4.3) can be rewritten as follows: for $(k; \mathbf{n}, \mathbf{h}, \mathbf{z}) \in \mathbb{N}^{1+n} \times \Theta$, with $x_i(\mathbf{h}, \mathbf{z}) \leq \min\{n_i, m_i\}$, $i = 1, \ldots, n$,

$$\left[\sum_{j=1}^{m} \nu_{\pi_j(h_j-1),\pi_j(h_j)} \mathrm{I}_{\{z_j=0\}} + \sum_{j=1}^{m} \mu_{\pi_j(h_j)} \mathrm{I}_{\{z_j=1\}}\right] b(k; \mathbf{n}, \mathbf{h}, \mathbf{z}) = \tag{4.4}$$

$$\sum_{j=1}^{m} \nu_{\pi_j(h_j-2),\pi_j(h_j-1)} b(k; \mathbf{n}, \mathbf{h}-\mathbf{e}_j, \mathbf{z}) \mathrm{I}_{\{x_{\pi_j(h_j-1)}(\mathbf{h},\mathbf{z}) = \min\{n_{\pi_j(h_j-1)}, m_{\pi_j(h_j-1)}\}\}} \mathrm{I}_{\{z_j=0\}}$$

$$+ \; y(k; \mathbf{n}, \mathbf{h}, \mathbf{z}),$$

where

$$y(k; \mathbf{n}, \mathbf{h}, \mathbf{z}) := \sum_{i=1}^{n} a_i \left[b(k; \mathbf{n}-\mathbf{e}_i, \mathbf{h}, \mathbf{z}) \mathrm{I}_{\{n_i>0\}} - b(k-1; \mathbf{n}, \mathbf{h}, \mathbf{z}) \mathrm{I}_{\{k>0\}}\right] \tag{4.5}$$

$$+ \sum_{j=1}^{m} \mu_{\pi_j(h_j)} b(k-1; \mathbf{n}+\mathbf{e}_{\pi_j(h_j)}, \mathbf{h}, \mathbf{z}) q_{\pi_j(h_j)} \mathrm{I}_{\{z_j=1\}} \mathrm{I}_{\{k>0\}}$$

$$+ \sum_{j=1}^{m} \nu_{\pi_j(h_j-1),\pi_j(h_j)} b(k; \mathbf{n}, \mathbf{h}, \mathbf{z}-\mathbf{e}_j) \mathrm{I}_{\{z_j=1\}}$$

$$+ \sum_{j=1}^{m} \mu_{\pi_j(h_j-1)} b(k-1; \mathbf{n}+\mathbf{e}_{\pi_j(h_j-1)}, \mathbf{h}-\mathbf{e}_j, \mathbf{z}+\mathbf{e}_j)$$

$$\times \left[1 - q_{\pi_j(h_j-1)} \mathrm{I}_{\{x_{\pi_j(h_j-1)}(\mathbf{h},\mathbf{z}) < n_{\pi_j(h_j-1)}\}}\right] \mathrm{I}_{\{z_j=0\}} \mathrm{I}_{\{k>0\}}.$$

We will now show how the relations (4.4), (4.5) can be used to compute the coefficients $b(k; \mathbf{n}, \mathbf{h}, \mathbf{z})$ mainly recursively. To this end, we first define the following partial ordering of the vectors $(k; \mathbf{n}, \mathbf{h}, \mathbf{z})$ (cf. [10]): for $(k; \mathbf{n}, \mathbf{h}, \mathbf{z}), (\hat{k}; \hat{\mathbf{n}}, \hat{\mathbf{h}}, \hat{\mathbf{z}}) \in \mathbb{N}^{1+n} \times \Theta$,

$$(k; \mathbf{n}, \mathbf{h}, \mathbf{z}) \prec (\hat{k}; \hat{\mathbf{n}}, \hat{\mathbf{h}}, \hat{\mathbf{z}}) \text{ if} \tag{4.6}$$

$$k + n_1 + \ldots + n_n < \hat{k} + \hat{n}_1 + \ldots + \hat{n}_n$$

or if $k + n_1 + \ldots + n_n = \hat{k} + \hat{n}_1 + \ldots + \hat{n}_n$ and $k < \hat{k}$.

Next, we extend the partial ordering $\prec$ to the vectors of supplementary values $(\mathbf{h},\mathbf{z})$: for $(k;\mathbf{n},\mathbf{h},\mathbf{z})$, $(k;\mathbf{n},\hat{\mathbf{h}},\hat{\mathbf{z}}) \in \mathbb{N}^{1+n}\times\Theta$,

$$(k;\mathbf{n},\mathbf{h},\mathbf{z}) \prec (k;\mathbf{n},\hat{\mathbf{h}},\hat{\mathbf{z}}) \text{ if } \forall j=1,\dots,m: z_j \le \hat{z}_j \text{ and } \exists j^*: z_{j^*} < \hat{z}_{j^*}. \tag{4.7}$$

It is readily verified that all coefficients in the right-hand side of (4.5) are of lower order w.r.t. $\prec$ than $(k;\mathbf{n},\mathbf{h},\mathbf{z})$, and hence may be considered to be known in (4.4). So, for *given* $k$, $\mathbf{n}$ and $\mathbf{z}$, it remains to define an ordering of the vectors $(k;\mathbf{n},\mathbf{h},\mathbf{z})$, $\mathbf{h}\in\Theta_1$. For given $\mathbf{z}\in\Theta_2$, we partition the index set $\{1,\dots,m\}$ into the following two subsets (corresponding to the collections of switching and serving servers, respectively):

$$C^{(0)}(\mathbf{z}) := \{j : z_j = 0\}, \quad C^{(1)}(\mathbf{z}) := \{j : z_j = 1\}. \tag{4.8}$$

Now, in order to derive an ordering for the vectors $(k;\mathbf{n},\mathbf{h},\mathbf{z})$, $\mathbf{h}\in\Theta_1$, for given values of $k$, $\mathbf{n}$ and $\mathbf{z}$, it should be noted that the right-hand side of (4.4) motivates to partition the index set $C^{(0)}(\mathbf{z})$ (for fixed $\mathbf{n}$, $\mathbf{h}$ and $\mathbf{z}$) into the following two subsets:

$$C_A^{(0)}(\mathbf{n},\mathbf{h},\mathbf{z}) := \{j \in C^{(0)}(\mathbf{z}) : x_i(\mathbf{h},\mathbf{z}) = \min\{n_i, m_i\} \text{ for all } i \in \Pi_j\}, \tag{4.9}$$

i.e., the set of switching servers that cannot start serving at a queue as long as neither arrivals nor service completions occur; and

$$\begin{aligned} C_B^{(0)}(\mathbf{n},\mathbf{h},\mathbf{z}) &:= C^{(0)}(\mathbf{z}) \setminus C_A^{(0)}(\mathbf{n},\mathbf{h},\mathbf{z}) \\ &:= \{j \in C^{(0)}(\mathbf{z}) : \exists i^* \in \Pi_j \text{ for which } x_{i^*}(\mathbf{h},\mathbf{z}) < \min\{n_{i^*}, m_{i^*}\}\}, \end{aligned} \tag{4.10}$$

i.e., the set of switching servers that *can* start serving at a queue (namely $Q_{i^*}$), even before either an arrival, or a service completion, or a switch-over completion of another server occurs. We now distinguish, for *given* $k$, $\mathbf{n}$, $\mathbf{z}$ *and* $h_j$ $(j\in C^{(1)}(\mathbf{z}))$, between two cases: (i) $C_A^{(0)}(\mathbf{n},\mathbf{h},\mathbf{z}) = \emptyset$, and (ii) $C_A^{(0)}(\mathbf{n},\mathbf{h},\mathbf{z}) \neq \emptyset$.

*Case (i).* $C_A^{(0)}(\mathbf{n},\mathbf{h},\mathbf{z}) = \emptyset$ (for given $k$, $\mathbf{n}$, $\mathbf{z}$ and $h_j$ $(j \in C^{(1)}(\mathbf{z}))$).
We show how a recursive computation scheme for the coefficients $b(k;\mathbf{n},\hat{\mathbf{h}},\mathbf{z})$, $\hat{\mathbf{h}}\in\Theta_1$, with $\hat{h}_j = h_j$ $(j\in C^{(1)}(\mathbf{z}))$, can be accomplished in this case. From $C_A^{(0)}(\mathbf{n},\mathbf{h},\mathbf{z}) = \emptyset$ it follows that there exists an $\mathbf{h}^*\in\Theta_1$, with $h_j^* = h_j$ $(j\in C^{(1)}(\mathbf{z}))$, such that for any $j\in C^{(0)}(\mathbf{z})$ there exists an $i^*\in\Pi_j$ (namely $i^* = \pi_j(h_j^*-1)$) for which $x_{i^*}(\mathbf{h}^*,\mathbf{z}) = x_{i^*}(\mathbf{h},\mathbf{z}) < \min\{n_{i^*}, m_{i^*}\}$. Then the first term at the right-hand side of (4.4) vanishes, so that $b(k;\mathbf{n},\mathbf{h}^*,\mathbf{z})$ is expressed only in terms of lower order w.r.t. $\prec$, cf. (4.6)-(4.7). The coefficient $b(k;\mathbf{n},\mathbf{h}^*,\mathbf{z})$ will be used as starting point for a recursive computation of the coefficients $b(k;\mathbf{n},\hat{\mathbf{h}},\mathbf{z})$, $\hat{\mathbf{h}}\in\Theta_1$, with $\hat{h}_j = h_j = h_j^*$ $(j\in C^{(1)}(\mathbf{z}))$. To this end, we define the following ordering of the vectors $(k;\mathbf{n},\mathbf{h},\mathbf{z})$: for $(k;\mathbf{n},\mathbf{h}',\mathbf{z}), (k;\mathbf{n},\mathbf{h}'',\mathbf{z}) \in \mathbb{N}^{1+n}\times\Theta$ (with $h_j' = h_j'' = h_j^* = h_j$, $j\in C^{(1)}(\mathbf{z})$),

$$\begin{aligned} &(k;\mathbf{n},\mathbf{h}',\mathbf{z}) \prec (k;\mathbf{n},\mathbf{h}'',\mathbf{z}) \text{ if} \\ &\forall j \in C_B^{(0)}(\mathbf{n},\mathbf{h},\mathbf{z}) : (h_j' - h_j^*) \bmod L_j \le (h_j'' - h_j^*) \bmod L_j, \\ &\text{and } \exists j^* \in C_B^{(0)}(\mathbf{n},\mathbf{h},\mathbf{z}) : (h_{j^*}' - h_{j^*}^*) \bmod L_{j^*} < (h_{j^*}'' - h_{j^*}^*) \bmod L_{j^*}. \end{aligned} \tag{4.11}$$

As an illustration of the ordering defined in (4.11), consider the following parameters: $m = 4$, $L_1 = 3$, $L_2 = 2$, $L_3 = 3$, $L_4 = 2$ and $\mathbf{z} = (1,0,0,1)$. Then we have $C^{(0)}(\mathbf{z}) = \{2,3\}$, $C^{(1)}(\mathbf{z}) = \{1,4\}$. If $h_1 = 2$, $h_4 = 1$, and $\mathbf{h}^* = (2,2,2,1)$, then the vectors $\mathbf{h} \in \Theta_1$, with given $h_1 = h_1^*$, $h_4 = h_4^*$, are ranked in increasing order as first $(2,2,2,1)$, then $(2,2,3,1)$, $(2,1,2,1)$, then $(2,2,1,1)$, $(2,1,3,1)$, and finally $(2,1,1,1)$.

*Case (ii).* $C_A^{(0)}(\mathbf{n},\mathbf{h},\mathbf{z}) \neq \emptyset$ (for given $k$, $\mathbf{n}$, $\mathbf{z}$ and $h_j$ $(j \in C^{(1)}(\mathbf{z}))$).
In this case the first term at the right-hand side of (4.4) does not vanish, so that the coefficients $b(k;\mathbf{n},\hat{\mathbf{h}},\mathbf{z})$, $\hat{\mathbf{h}} \in \Theta_1$, with $\hat{h}_j = h_j$ $(j \in C^{(1)}(\mathbf{z}))$, can not be calculated recursively from (4.4) and (4.5). By definition, for each $j \in C_B^{(0)}(\mathbf{n},\mathbf{h},\mathbf{z})$ there exist an $i^* \in \Pi_j$, and an $h_j^* \in \{1,\ldots,L_j\}$ with $i^* = \pi_j(h_j^* - 1)$, for which $x_{i^*}(\mathbf{h},\mathbf{z}) < \min\{n_{i^*}, m_{i^*}\}$. Hence, the coefficients $b(k;\mathbf{n},\hat{\mathbf{h}},\mathbf{z})$, $\hat{\mathbf{h}} \in \Theta_1$, with $\hat{h}_j = h_j$ $(j \in C^{(1)}(\mathbf{z}))$ *and* $\hat{h}_j = h_j^*$ $(j \in C_B^{(0)}(\mathbf{n},\mathbf{h},\mathbf{z}))$, can be computed by solving the set of $\prod_{j \in C_A^{(0)}(\mathbf{n},\mathbf{h},\mathbf{z})} L_j$ linear equations induced by (4.4).
Then, the $\prod_{j \in C_B^{(0)}(\mathbf{n},\mathbf{h},\mathbf{z})} L_j$ coefficients $b(k;\mathbf{n},\hat{\mathbf{h}},\mathbf{z})$, $\hat{\mathbf{h}} \in \Theta_1$, with $\hat{h}_j = h_j$ $(j \in C^{(1)}(\mathbf{z}))$, can be computed by solving the set of equations (4.4) for the coefficients $b(k;\mathbf{n},\hat{\mathbf{h}},\mathbf{z})$, $\hat{\mathbf{h}} \in \Theta_1$, with $\hat{h}_j = h_j$ $(j \in C^{(1)}(\mathbf{z}))$, $\hat{h}_j = \tilde{h}_j$ $(j \in C_B^{(0)}(\mathbf{n},\mathbf{h},\mathbf{z}))$ in increasing order of the values of the combinations $\tilde{h}_j$ $(j \in C_B^{(0)}(\mathbf{n},\mathbf{h},\mathbf{z}))$ w.r.t. the partial ordering defined in (4.11), starting with $\tilde{h}_j = h_j^*$ $(j \in C_B^{(0)}(\mathbf{n},\mathbf{h},\mathbf{z}))$. We refer to Remark 4.2 for a more intuitive characterization of whether or not (4.4) is recursively solvable.
The same conditions which guarantee that (4.2) holds also guarantee that these sets possess a unique solution (cf. [10]), except for the case $\mathbf{n} = \mathbf{0}$. So the only states that need further attention are the states with $\mathbf{n} = \mathbf{0}$ (and hence, $\mathbf{z} = \mathbf{0}$, cf. (3.5)). In this case the set of equations (4.4) reads: for $(k;\mathbf{h}) \in \mathbb{N} \times \Theta_1$,

$$\sum_{j=1}^{m} \nu_{\pi_j(h_j-1),\pi_j(h_j)} b(k;\mathbf{0},\mathbf{h},\mathbf{0}) = \sum_{j=1}^{m} \nu_{\pi_j(h_j-2),\pi_j(h_j-1)} b(k;\mathbf{0},\mathbf{h}-\mathbf{e}_j,\mathbf{0}). \tag{4.12}$$

One may verify by summing the equations (4.12) over $\mathbf{h}$, $\mathbf{h} \in \Theta_1$, that this set of equations is dependent. However, the law of total probability (3.7) yields (together with (4.2)) the following additional equation: for $k = 0,1,\ldots$,

$$\sum_{(\mathbf{h},\mathbf{z}) \in \Theta} b(k;\mathbf{0},\mathbf{h},\mathbf{z}) = Y(k), \tag{4.13}$$

where $Y(0) := 1$ and for $k = 1,2,\ldots$,

$$Y(k) := -\sum_{0 < n_1 + \ldots + n_n \leq k} \sum_{(\mathbf{h},\mathbf{z}) \in \Theta} b(k - n_1 - \ldots - n_n;\mathbf{n},\mathbf{h},\mathbf{z}). \tag{4.14}$$

Note that the right-hand side of (4.14) contains only coefficients of lower order w.r.t. $\prec$ than $(k;\mathbf{0},\mathbf{h},\mathbf{z})$, cf. (4.6). Now, all but one of the equations in (4.12) together with either (4.13) or (4.14) uniquely determine the coefficients $b(k;\mathbf{0},\mathbf{h},\mathbf{0})$, $\mathbf{h} \in \Theta_1$, for $k = 0,1,\ldots$, provided the Markov process $\{(\mathbf{N}(t),\mathbf{H}(t),\mathbf{Z}(t)), t \geq 0\}$, conditioned on $\mathbf{N}(t) = 0$ and $\mathbf{Z}(t) = 0$, is irreducible. This condition is satisfied when in the case $\lambda_i = 0$ $(i = 1, \ldots, n)$ each state $(\mathbf{0}, \mathbf{h}, \mathbf{0})$, $\mathbf{h} \in \Theta_1$, can be reached from any other state with $(\mathbf{0}, \hat{\mathbf{h}}, \mathbf{0})$, $\hat{\mathbf{h}} \in \Theta_1$, i.e., the (conditioned) Markov process is irreducible. For the present model this condition is satisfied, because (conditioned on $\mathbf{N}(t) \equiv \mathbf{0}$) the servers keep on switching along the queues according to their respective service order tables in a periodic way, independently of each other.

In practice, the number of coefficients that can be computed is restricted by limitations on the available amounts of storage capacity and computation time. Here, we assume that the coefficients of the power series are computed up to a given power $M_{\max}$ of $\rho$.
Summarizing, the coefficients for $(k; \mathbf{n}, \mathbf{h}, \mathbf{z}) \in \mathbb{N}^{1+n} \times \Theta$ (with $x_i(\mathbf{h}, \mathbf{z}) \leq \min\{n_i, m_i\}$, $i = 1, \ldots, n$) can be computed according to the following computation scheme:

*step 1*: $M := 0$;
*step 2*: for all $(k; \mathbf{n}) \in \mathbb{N}^{1+n}$ with $\mathbf{n} \neq \mathbf{0}$ and with $k + n_1 + \ldots + n_n = M$ do
for all $\mathbf{z} \in \Theta_2$ (in increasing order of $\mathbf{z}$ w.r.t. $\prec$ (cf. (4.6), (4.7))) do
for all possible combinations of $h_j$ $(j \in C^{(1)}(\mathbf{z}))$ do
if $C_A^{(0)}(\mathbf{n}, \mathbf{h}, \mathbf{z}) = \emptyset$ then
determine $\mathbf{h}^*$ as follows:
for $j \in C^{(0)}(\mathbf{z})$, determine $i^* \in \Pi_j$ for which $x_{i^*}(\mathbf{h}, \mathbf{z}) < \min\{n_{i^*}, m_{i^*}\}$,
and determine $h_j^*$ such that $i^* = \pi_j(h_j^* - 1)$;
for $j \in C^{(1)}(\mathbf{z})$, let $h_j^* = h_j$;
determine the coefficients $b(k; \mathbf{n}, \hat{\mathbf{h}}, \mathbf{z})$, $\hat{\mathbf{h}} \in \Theta_1$, with $\hat{h}_j = h_j$ $(j \in C^{(1)}(\mathbf{z}))$,
*recursively* in increasing order of $\prec$, cf. (4.11);
if $C_A^{(0)}(\mathbf{n}, \mathbf{h}, \mathbf{z}) \neq \emptyset$ then
determine $\mathbf{h}^*$ $(j \in C_B^{(0)}(\mathbf{n}, \mathbf{h}, \mathbf{z}))$ as follows:
determine $i^* \in \Pi_j$ for which $x_{i^*}(\mathbf{h}, \mathbf{z}) < \min\{n_{i^*}, m_{i^*}\}$,
and determine $h_j^*$ such that $i^* = \pi_j(h_j^* - 1)$;
compute the coefficients $b(k; \mathbf{n}, \hat{\mathbf{h}}, \mathbf{z})$, $\hat{\mathbf{h}} \in \Theta_1$, with $\hat{h}_j = h_j$ $(j \in C^{(1)}(\mathbf{z}))$,
by successively solving the set of linear equations (4.4) for $\hat{h}_j = \tilde{h}_j$
$(j \in C_B^{(0)}(\mathbf{n}, \mathbf{h}, \mathbf{z}))$ in increasing order of the combinations $\tilde{h}_j$
$(j \in C_B^{(0)}(\mathbf{n}, \mathbf{h}, \mathbf{z}))$ w.r.t. (4.11) (starting with $\tilde{h}_j = h_j^*$ $(j \in C_B^{(0)}(\mathbf{n}, \mathbf{h}, \mathbf{z}))$);
*step 3*: determine $b(M; \mathbf{0}, \mathbf{h}, \mathbf{0})$, $\mathbf{h} \in \Theta_1$, by solving the set of equations (4.12) together with either (4.13) or (4.14);
*step 4*: $M := M + 1$;
*step 5*: if $M < M_{\max}$ then return to step 2; otherwise STOP.

This computation sche.ne allows for the calculation of the state probabilities up to, in principle, any level of accuracy. Once the coefficients of the power series have been obtained (up to some power of $\rho$), the expectation of an arbitrary function $g(\mathbf{N}, \mathbf{H}, \mathbf{Z})$ of the state probabilities can be expressed in terms of these coefficients as

$$\mathrm{E}(g(\mathbf{N}, \mathbf{H}, \mathbf{Z})) = \sum_{k=0}^{\infty} \sum_{n_1 + \ldots + n_n \leq k} \sum_{(\mathbf{h}, \mathbf{z}) \in \Theta} g(\mathbf{n}, \mathbf{h}, \mathbf{z}) b(k - n_1 - \ldots - n_n; \mathbf{n}, \mathbf{h}, \mathbf{z}). \tag{4.15}$$

**Remark 4.1** The assumption of exponentially distributed service and switch-over times mainly served the ease of the presentation. In fact, the approach presented in this section can be generalized in a straightforward manner to systems with Coxian distributed service times and switch-over times, cf. [10]. A Coxian distributed random variable consists of a stochastic number of not necessarily identical exponential phases. The class of Coxian distributions lies dense in the class of distribution functions of non-negative random variables (cf. [6]). In the case of Coxian distributions for the service times and switch-over times the supplementary space should be extended with variables describing the actual phases of the service times or the switch-over times. Similarly, Markov Arrival Processes (MAP's) could be incorporated into the model. It should be noted that Coxian distributions are preferred to more general phase-type distributions, because Coxian distributions allow for more efficient computations, cf. [10], [11]. The approach presented in this section can also readily be extended to multiple-server systems in which some of the switch-over times are negligible (i.e., $\nu_{i,k} = \infty$). In that case, some slight modifications of the balance equations and the computation scheme would have to be made. In the case that all switch-over times are negligible, the presence of a unique zero state simplifies the computation scheme; cf. [8] for single-server polling systems with zero switch-over times.

□

**Remark 4.2** In order to characterize whether or not the set of equations (4.4) is recursively solvable, let us reconsider the set of equations (4.4) for *given* values of $k$, $\mathbf{n}$, $\mathbf{z}$ and $h_j$ ($j \in C^{(1)}(\mathbf{z})$). That is, we consider the following information on the current state of the system to be known: (i) the joint queue length, (ii) whether each of the servers is serving or switching, and (iii) the queues (entries) at which the serving servers are working. Then, given this configuration, $C_A^{(0)}(\mathbf{n}, \mathbf{h}, \mathbf{z})$ is the set of indices corresponding to *those* switching servers $j$ that have to skip *each* queue ($i = \pi_j(h_j - 1)$) that they visit, either because the maximal allowable number of servers is already working at that queue (i.e. $x_i(\mathbf{h}, \mathbf{z}) = m_i$) or because there are no waiting customers at that queue (i.e. $x_i(\mathbf{h}, \mathbf{z}) = n_i$). So, as long as neither arrivals nor service completions occur, the server $S_j$ ($j \in C_A^{(0)}(\mathbf{n}, \mathbf{h}, \mathbf{z})$) will keep on moving along the queues without serving any customer. Thus, for *given* $k$, $\mathbf{n}$, $\mathbf{z}$ and $h_j$ ($j \in C^{(1)}(\mathbf{z})$), the set of equations (4.4) is not recursively solvable if and only if one or more servers will keep on moving along the queues (according to their respective polling tables) as long as no arrivals nor service completions occur.

□

**Remark 4.3** In the case that all polling tables are surjective mappings, i.e., each queue is visited by each of the servers, then (for given values of $k$, $\mathbf{n}$, $\mathbf{z}$ and $h_j$ ($j \in C^{(1)}(\mathbf{z})$)) either $C_A^{(0)}(\mathbf{n}, \mathbf{h}, \mathbf{z}) = \emptyset$ (and $C_B^{(0)}(\mathbf{n}, \mathbf{h}, \mathbf{z}) = C^{(0)}(\mathbf{z})$) or $C_B^{(0)}(\mathbf{n}, \mathbf{h}, \mathbf{z}) = \emptyset$ (and $C_A^{(0)}(\mathbf{n}, \mathbf{h}, \mathbf{z}) = C^{(0)}(\mathbf{z})$). To see this, suppose $C_A^{(0)}(\mathbf{n}, \mathbf{h}, \mathbf{z}) = \emptyset$, then there exists an $i \in \{1, \ldots, n\}$ for which $x_i(\mathbf{h}, \mathbf{z}) < \min\{n_i, m_i\}$. Because each server visits each of the queues, for each $j \in C^{(0)}(\mathbf{z})$ there exists an $\tilde{h}_j$ for which $\pi_j(\tilde{h}_j - 1) = i$, so that $j \in C_B^{(0)}(\mathbf{n}, \mathbf{h}, \mathbf{z})$. As a result, as long as neither arrivals nor service completions occur, either all switching servers or none of them will keep on moving along the queues without serving any customers.

□

## 5 Complexity

The time and memory requirements of the PSA are known to increase exponentially with the number of queues, so that its use is restricted to systems with a fairly small number of queues, cf. [10]. In this section we will see that the required amounts of computation time and storage capacity also increase exponentially in the number of servers. More precisely, one may verify that the total number of terms that have to be evaluated in order to compute the coefficients of the power series up to the $M$-th power of $\rho$ is given by

$$\binom{M+n+1}{n+1} \times \prod_{j=1}^{m} L_j \times 2^m. \tag{5.1}$$

The first factor indicates the number of vectors $(k; \mathbf{n})$ for which $k + n_1 + \ldots + n_n \leq M$; the second and third factor together indicate the size of the supplementary space. In practice, however, only a limited number of performance measures have to be evaluated (e.g. mean waiting times) rather than all individual state probabilities. Then, the coefficients of the power-series expansions of the important performance measures can be aggregated during the execution of the PSA (cf. e.g. [10] (section 2)), and stored in (relatively small) arrays, while the coefficients of the state probabilities can be removed as soon as they are not needed anymore in further computations. This approach reduces the storage requirement for the calculation of $M$ terms of the power-series expansion to (cf. [8] (section 5)):

$$\binom{M+n}{n} \times \prod_{j=1}^{m} L_j \times 2^m. \tag{5.2}$$

As an illustration, consider a model in which all servers move along the queues in a strictly cyclic manner (i.e., $L_j = n$, $j = 1, \ldots, m$). Table 5.1 gives the maximal number of coefficients of the power series that can be computed according to (5.2) for given amounts of storage capacity and for various values of the number of servers and the number of queues.

| | memory $= 10^6$ coeff. | | | memory $= 10^7$ coeff. | | |
|---|---|---|---|---|---|---|
| | $n=2$ | $n=3$ | $n=4$ | $n=2$ | $n=3$ | $n=4$ |
| $m=1$ | 705 | 98 | 39 | 2234 | 213 | 71 |
| $m=2$ | 352 | 53 | 22 | 1116 | 116 | 41 |
| $m=3$ | 175 | 28 | 12 | 557 | 63 | 23 |
| $m=5$ | 42 | 7 | 2 | 138 | 17 | 6 |

**Table 5.1.** The maximal number of terms for given amounts of memory space.

Table 5.1 illustrates that the number of terms of the power series that can be computed for a given amount of storage capacity may decrease considerably when the numbers of servers and queues are increased. As a consequence, the use of the PSA is restricted to systems with a fairly small number of queues and with a rather small number of servers.

The reader is referred to [10], [11] for a detailed discussion on the practical implementation of the PSA. The ideas given there (on improving the rate of convergence of the power series and on efficient storage management) generally lead to strong improvements of the performance of the PSA.

In general, it is not easy to give rules of thumb for the number of terms of the power-series expansions of the state-probabilities that is needed to achieve an 'acceptable' degree of accuracy. This number generally depends on a number of factors such as the occupancy (load) of the system and on the 'degree of symmetry' of the system. If the system is fairly symmetrical then 10 terms may suffice to give rather accurate results for lightly loaded systems (say $\rho/m < 0.5$); if the system is heavily loaded (say $\rho/m = 0.9$) then 10 to 15 terms may still do well (applying extrapolation techniques, cf. [10]). If the system is rather asymmetrical the algorithm may converge rather slowly. If the system is lightly loaded then 10 or 15 terms may still do well, but if the system is more heavily loaded then typically 30 or 40 (or even more) terms may be needed to achieve accurate results.

## 6 Numerical results

In this section we give an overview of the numerical results that we gathered. Firstly, we investigate the tendency for the servers to cluster (especially in heavily-loaded systems in which the servers follow the same route) and the influence of the visit orders on the system performance. Numerical examples will show that the system performance can be improved considerably by using so-called dispersive schedules, i.e., schedules that somehow keep the servers apart. Secondly, we consider the option of partitioning a multiple-server system into separate systems. It is illustrated that this segmentation may be particularly beneficial if the queues are clustered and the clusters are somewhat isolated, i.e, if the switch-over times needed by the servers to move from one cluster to another are significant. Finally, we make some comparisons between multiple-server systems and single-server systems carrying a comparable load. These investigations suggest that, as far as the mean waiting times at the queues are concerned, multiple-server systems compare favourably with proportionally loaded single-server systems. Such comparisons are also relevant for developing approximations for multiple-server polling systems based on existing results for single-server polling systems.

*Coalescing of the servers; influence of the visit orders*

An interesting property of multi-server polling systems is the fact that the servers tend to coalesce, especially in heavily loaded systems in which the servers follow the same route. This phenomenon may be visualized as follows. A trailing server will tend to move fast, as it only encounters recently served queues, whereas a leading server will tend to be slowed down by queues that have not been served for a while, so that the servers tend to form bunches while constantly leapfrogging over one another. This explains why the visit orders and the system load play a role in this coalescing effect. To illustrate this, consider a model with the following parameters: $n = 4$; $m = 2$; $a_i = \lambda_i/\rho = 0.25$, $\mu_i = 1$, $m_i = 2$, $q_i = 1$ (i.e., exhaustive service), $i = 1, \ldots, n$; $\nu_{i,k} = 1/\alpha$ for $i, k = 1, \ldots, n$, with $\alpha$ to be specified later on; $\boldsymbol{\pi}_1 = \boldsymbol{\pi}_2 = (1, 2, 3, 4)$. We define the joint probability distribution of the server positions as follows:

$$\mathrm{P}(k_1, \ldots, k_m) := \Pr\{S_j \text{ is working at or switching to } Q_{k_j}, j = 1, \ldots, m\}, \tag{6.1}$$

for $k_j = 1, \ldots, n, j = 1, \ldots, m$. Table 6.1 gives the mean waiting times at the queues $\mathrm{EW}_{1-4}$ (which are equal for each queue) and the server-position distribution $P(k_1, \ldots, k_m)$ (cf. (6.1)) for $\alpha = 0.05$ and for varying values of the offered load to the system $\rho$.

| $\rho = 0.4$ | | $\rho = 1.6$ | | $\rho = 1.9$ | |
|---|---|---|---|---|---|
| $EW_{1-4}$ | P ($\times 10^{-2}$) | $EW_{1-4}$ | P ($\times 10^{-2}$) | $EW_{1-4}$ | P ($\times 10^{-2}$) |
| 0.14 | **6.3** 6.2 6.2 6.2 | 2.08 | **7.8** 5.9 5.4 5.9 | 10.68 | **14.6** 3.9 2.5 3.9 |
| | 6.2 **6.3** 6.2 6.2 | | 5.9 **7.8** 5.9 5.4 | | 3.9 **14.6** 3.9 2.5 |
| | 6.2 6.2 **6.3** 6.2 | | 5.4 5.9 **7.8** 5.9 | | 2.5 3.9 **14.6** 3.9 |
| | 6.2 6.2 6.2 **6.3** | | 5.9 5.4 5.9 **7.8** | | 3.9 2.4 3.9 **14.6** |

**Table 6.1.** The coalescing effect.

Table 6.1 illustrates that the tendency for the servers to cluster grows as the traffic intensity grows. Indeed, if the system is lightly loaded, the switch-over times tend to predominate the lengths of the visit periods and will thus constantly disperse the servers over the system. In heavily loaded systems on the other hand, the visit periods will dominate the switch-over times and drive the servers together.

In order to investigate the impact of the visit orders on the system performance, we have computed the mean waiting times at the various queues, $\mathbf{EW}_i$ (which are no longer equal for each queue when the visit orders of the servers are different), and the mean total amount of unfinished work in the system, $\mathbf{EV}$, for a model with the same parameters as before but with different visit orders; as before $\boldsymbol{\pi}_1 = (1, 2, 3, 4)$, but $\boldsymbol{\pi}_2$ is varied over all possible permutations of the index set $\{1, 2, 3, 4\}$. Note that because of the symmetry of the model there are only three non-equivalent cyclic service order combinations. Table 6.2 shows the results for various values of $\alpha$ and $\rho$; the case $\alpha = 0$ corresponds to a system with zero switch-over times, cf. also Remark 4.1. Using Little's law it is easily verified that $\mathbf{EW}_i$ and $\mathbf{EV}$ satisfy the relationship $\mathbf{EV} = \sum_{i=1}^{n} \rho_i \mathbf{EW}_i + \frac{1}{2} \sum_{i=1}^{n} \lambda_i \beta_i^{(2)}$, irrespective of the interarrival, service, and switch-over times. Here $\frac{1}{2} \sum_{i=1}^{n} \lambda_i \beta_i^{(2)} = \rho$, as the service times are exponentially distributed with mean 1.

| $\alpha$ | $\pi_2$ | $\rho = 1.6$ | | | | | $\rho = 1.8$ | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | $EW_1$ | $EW_2$ | $EW_3$ | $EW_4$ | EV | $EW_1$ | $EW_2$ | $EW_3$ | $EW_4$ | EV |
| 0.00 | (1, 2, 3, 4) | 1.78 | 1.78 | 1.78 | 1.78 | 4.44 | 4.26 | 4.26 | 4.26 | 4.26 | 9.47 |
| | (1, 2, 4, 3) | 1.78 | 1.85 | 1.74 | 1.74 | 4.44 | 4.42 | 4.67 | 3.98 | 3.98 | 9.47 |
| | (1, 4, 3, 2) | 1.78 | 1.78 | 1.78 | 1.78 | 4.44 | 4.26 | 4.26 | 4.26 | 4.26 | 9.47 |
| 0.05 | (1, 2, 3, 4) | 2.08 | 2.08 | 2.08 | 2.08 | 4.92 | 4.87 | 4.87 | 4.87 | 4.87 | 10.56 |
| | (1, 2, 4, 3) | 2.07 | 2.15 | 2.02 | 2.02 | 4.91 | 5.00 | 5.27 | 4.47 | 4.47 | 10.45 |
| | (1, 4, 3, 2) | 2.06 | 2.06 | 2.06 | 2.06 | 4.90 | 4.79 | 4.79 | 4.79 | 4.79 | 10.43 |
| 0.25 | (1, 2, 3, 4) | 3.29 | 3.29 | 3.29 | 3.29 | 6.87 | 7.36 | 7.36 | 7.36 | 7.36 | 15.05 |
| | (1, 2, 4, 3) | 3.24 | 3.40 | 3.09 | 3.09 | 6.73 | 7.52 | 7.87 | 6.39 | 6.39 | 14.48 |
| | (1, 4, 3, 2) | 3.16 | 3.16 | 3.16 | 3.16 | 6.67 | 6.86 | 6.86 | 6.86 | 6.86 | 14.15 |

**Table 6.2.** Influence of the visit order.

Table 6.2 shows that the service orders may have a considerable impact on the individual mean waiting times. For single-server systems similar observations have been made by Blanc [9]. However, for single-server systems with exhaustive service it is well-known that $\mathbf{EV}$, and hence also $\sum_{i=1}^{n} \rho_i \mathbf{EW}_i$, is completely insensitive to the service order (as long as it is strictly cyclic), cf. Boxma & Groenendijk [14]. Table 6.2 suggests that in multiple-server systems $\mathbf{EV}$ is perhaps not extremely sensitive to the service orders but definitely not completely insensitive. In fact also the individual mean waiting times appear to be more sensitive to the service orders in multiple-server systems. Table 6.2 shows e.g. that in a multiple-server system even in case of a completely symmetric configuration the individual mean waiting times depend on the service order, unlike in a single-server system. The difference in sensitivity may be intuitively explained as follows. In case of exhaustive service the individual mean waiting times strongly depend on the mean residual intervisit time. In a single-server system the intervisit time is the time needed for the server *itself* to reach the queue again, i.e., the time involved in passing through the complete system once, which usually only marginally depends on the service order. In a multiple-server system the intervisit time is typically the time needed for *one* of the *other* servers to reach the queue again, which strongly varies with the degree of clustering as implied by the service order. Table 6.2 points out e.g. that $\boldsymbol{\pi}_2 = (1, 4, 3, 2)$ yields the best global performance (i.e., minimal $\mathbf{EV}$) in all considered cases. Indeed, $\boldsymbol{\pi}_2 = (1, 4, 3, 2)$ is likely to minimize the degree of clustering. The latter observation is in line with the observation of Morris & Wang [27] that the system performance can be improved when the coalescing effect is alleviated by using dispersive schedules. Note that in the case $\alpha = 0$, because of the symmetry of the model, $\mathbf{EV}$ does not depend on the service orders and has the same distribution as in a classical $M/M/m$ system with the same parameters.

*Partitioning versus non-partitioning*

When studying multiple-server polling systems it is interesting to consider the option of partioning the system into a number of subsystems, each of which is served by one or more specific servers. Such a partitioning (or segmentation) seems to be particularly beneficial if the queues are clustered, and the switch-over times to move between the clusters are relatively large. On the other hand, when the system is partitioned into subsystems, the servers operating in mutually isolated clusters are not able to 'help' each other. Hence, it will happen from time to time that one server is idle while another server is still busy, so that the processing power is only partially used. As a consequence, in systems with negligible switch-over times the amount of work in the system will always be smaller in the non-partitioned system. To illustrate the effect of partitioning, consider the following model: $n = 4$; $m = 2$; $a_i = \lambda_i/\rho = 0.25$, $\mu_i = 1$, $m_i = 2$, $i = 1, \ldots, n$; if $i, k \in \{1, 2\}$ or $i, k \in \{3, 4\}$ then $\nu_{i,k} = 1/0.05$, otherwise $\nu_{i,k} = 1/\alpha$, $i, k = 1, \ldots, n$. We compare the system performance between (i) the non-partitioned model, in which both servers visit each of the queues, with $\boldsymbol{\pi}_1 = \boldsymbol{\pi}_2 = (1, 2, 3, 4)$, and (ii) the partitioned model, in which $S_1$ serves $Q_1$ and $Q_2$ and $S_2$ serves $Q_3$ and $Q_4$, with $\boldsymbol{\pi}_1 = (1, 2)$ and $\boldsymbol{\pi}_2 = (3, 4)$. Table 6.3 gives the mean total amount of waiting work in the system (which is proportional to the mean waiting time of an arbitrary customer here) for various values of $\alpha$ and offered load $\rho$, for the cases $\mathbf{q} = (0, 0, 0, 0)$ (i.e., 1-limited service at each queue) and $\mathbf{q} = (1, 1, 1, 1)$ (i.e., exhaustive service).

| $\alpha$ | $\mathbf{q} = (0,0,0,0)$ | | | | $\mathbf{q} = (1,1,1,1)$ | | | |
|---|---|---|---|---|---|---|---|---|
| | $\rho = 0.4$ | $\rho = 0.8$ | $\rho = 1.6$ | $\rho = 1.8$ | $\rho = 0.4$ | $\rho = 0.8$ | $\rho = 1.6$ | $\rho = 1.8$ |
| 0.01 | 0.10 | 0.28 | 2.30 | 6.48 | 0.10 | 0.27 | 1.96 | 4.63 |
| 0.05 | 0.14 | 0.34 | 2.72 | 9.14 | 0.14 | 0.31 | 2.08 | 4.87 |
| 0.10 | 0.19 | 0.42 | 3.39 | $\infty$ | 0.18 | 0.37 | 2.23 | 5.17 |
| 0.25 | 0.36 | 0.67 | 7.31 | $\infty$ | 0.33 | 0.56 | 2.69 | 6.11 |
| 0.50 | 0.65 | 1.15 | $\infty$ | $\infty$ | 0.58 | 0.87 | 3.50 | 7.75 |
| 1.00 | 1.27 | 2.43 | $\infty$ | $\infty$ | 1.08 | 1.50 | 5.24 | 11.93 |
| partitioning | 0.35 | 0.82 | 5.47 | 17.73 | 0.33 | 0.76 | 4.18 | 9.30 |

**Table 6.3.** Effect of partitioning.

Table 6.3 confirms the conjecture that when the switch-over times are relatively small, partitioning will generally be disadvantageous. Moreover, it is illustrated that when the switch-over times become large, the loss of service capacity due to the switch-over times may tend to predominate the benefits from a non-partitioned system. Table 6.3 also suggests that the question as to whether or not partitioning is beneficial generally depends on the offered load to the system. In fact, when the offered load is increased, it may well occur (for limited-type service disciplines) that some queues become instable in the non-partitioned system, whereas in the partitioned system all queues remain stable.

*Comparisons with single-server systems carrying a comparable load.*
When investigating the performance of multiple-server polling systems, it is interesting to make comparisons with single-server systems with a comparable load. To this end, we first compare the performance of a multiple-server polling system (with $m$ servers) with a single-server polling system in which the server operates at $m$-fold processing rate. Then, we will compare the multiple-server system with a single-server system with $1/m$-fold arrival rates.

Consider a multiple-server system versus a single-server system in which the server operates at $m$-fold processing rate. In the single-server case all processing power is concentrated, so that the single-server system might roughly be seen as a multiple-server system with extreme coalescence of the servers. Hence, one would expect the waiting times at the queues to be smaller in the multiple-server case, because the processing power would be more homogeneously distributed over the queues, cf. the discussion above. On the other hand, although the waiting times at the queues are expected to be smaller in the multiple-server case, the sojourn time (i.e., waiting time plus service time) of a customer in the system might be smaller in the single-server case (especially in light traffic), because the service times are (stochastically) smaller. In fact, for zero switch-over times the amount of work and hence, in a symmetrical system, the sojourn time, is smaller in the single-server $m$-speed case than in the multiple-server case. As an illustration, we compare the system performance in both situations for the following model: $n = 4$; $a_i = \lambda_i/\rho = 0.25$, $\mu_i = \mu$, $m_i = 2$, $q_i = 1$, $i = 1, \dots, n$ (i.e., exhaustive service); $\nu_{i,k} = 1/\alpha$, $i, k = 1, \dots, n$; $\boldsymbol{\pi}_1 = \boldsymbol{\pi}_2 = (1,2,3,4)$. Table 6.4 shows the mean waiting time, $\mathbf{E}W$, and the mean sojourn time, $\mathbf{E}R$ $(= \mathbf{E}W + 1/\mu)$ of an arbitrary customer for the system with two servers both processing at normal speed ($\mu = 1$) and the system with a single server processing at double speed ($\mu = 2$).

| $\alpha$ | 2 normal-speed servers | | | | | | 1 double-speed server | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | $\rho = 0.4$ | | $\rho = 1.6$ | | $\rho = 1.8$ | | $\rho = 0.4$ | | $\rho = 1.6$ | | $\rho = 1.8$ | |
| | **EW** | **ER** | **EW** | **ER** | **EW** | **ER** | **EW** | **ER** | **EW** | **ER** | **EW** | **ER** |
| 0.01 | 0.06 | 1.06 | 1.84 | 2.84 | 4.38 | 5.38 | 0.15 | 0.65 | 2.09 | 2.59 | 4.66 | 5.16 |
| 0.10 | 0.23 | 1.23 | 2.37 | 3.37 | 5.45 | 6.45 | 0.41 | 0.91 | 2.85 | 3.35 | 6.10 | 6.60 |
| 0.25 | 0.50 | 1.50 | 3.29 | 4.29 | 7.37 | 8.37 | 0.84 | 1.34 | 4.13 | 4.63 | 8.50 | 9.00 |
| 0.50 | 0.95 | 1.95 | 4.96 | 5.96 | 10.95 | 11.95 | 1.56 | 2.06 | 6.25 | 6.75 | 12.50 | 13.00 |
| 1.00 | 1.84 | 2.84 | 8.73 | 9.73 | 18.72 | 19.72 | 3.00 | 3.50 | 10.50 | 11.00 | 20.50 | 21.00 |

**Table 6.4.** Two servers versus a single server with two-fold processing rate: mean waiting time and mean sojourn time of an arbitrary customer.

Table 6.4 points out that the mean waiting times are generally smaller in the multiple-server case, but that this is generally not true for the mean sojourn times. So in order to answer the question if the multiple-server system outperforms a single-server system operating at proportional speed, we have to deal with a trade-off between (i) the increase of the mean waiting time, and (ii) the decrease of the mean service time. The results show that for small switch-over times the decrease of the mean service times dominates the increase of the mean waiting times, whereas the reverse is true when the switch-over times are relatively large.

We finally make a comparison of the performance of a single-server system with a multiple-server system (with $m$ servers) with $m$-fold arrival rates. In general multiple-server systems tend to outperform single-server systems with a proportional arrival stream, as the servers in a sense have the opportunity to cooperate. Stoyan [30] shows e.g. that in an ordinary $M/G/m$ system the mean waiting time is indeed always smaller than in an $M/G/1$ with proportional arrival rate. Although hard to prove, it is likely that in polling systems the situation is similar. As an illustration, consider the model with the following system parameters: $n = 4$; $a_i = \lambda_i/\rho = 0.25$, $\mu_i = 1$, $m_i = 2$, $q_i = 1$ (i.e., exhaustive service), $i = 1, \dots, n$; $\nu_{i,k} = 1/\alpha$, $i, k = 1, \dots, n$; $\boldsymbol{\pi}_1 = (1, 2, 3, 4)$. We have computed the mean waiting time for the single-server case and the two-server case (with $\boldsymbol{\pi}_2 = (1, 2, 3, 4)$) in which the arrival rate at each of the queues is doubled. Note that in the latter case the symmetry of the model implies that both servers carry the same load $\rho/2$. Table 6.5 shows the results for various values of $\alpha$ and offered load $\rho$.

| $\alpha$ | single server | | | | two servers | | | |
|---|---|---|---|---|---|---|---|---|
| | $\rho = 0.2$ | $\rho = 0.4$ | $\rho = 0.8$ | $\rho = 0.9$ | $\rho = 0.4$ | $\rho = 0.8$ | $\rho = 1.6$ | $\rho = 1.8$ |
| 0.00 | 0.25 | 0.67 | 4.00 | 9.00 | 0.04 | 0.19 | 1.78 | 4.26 |
| 0.05 | 0.39 | 0.84 | 4.43 | 9.80 | 0.14 | 0.31 | 2.08 | 4.87 |
| 0.25 | 0.97 | 1.54 | 6.13 | 13.00 | 0.50 | 0.77 | 3.29 | 7.37 |

**Table 6.5.** Single server versus two servers with doubled load: mean waiting time of an arbitrary customer.

Table 6.5 confirms the conjecture that multiple-server systems lead to a better system performance than single-server systems with proportional arrival rates.

## 7 Conclusions and topics for further research

We have considered a polling model with multiple servers in which each server visits the queues according to a given service order table. In general, such models are mathematically intractable. In this paper it is shown how the model can be implemented into the PSA, a device for the numerical evaluation (and optimization) of performance measures of the system. This implementation into the PSA has been used for various numerical experiments. We have observed some interesting phenomena, such as the tendency for the servers to bunch together (cf. also [27]), and the fact that this coalescence of the servers deteriorates the system performance. We have also observed that the service orders may have a considerable impact on the system performance, as opposed to single server systems. We have considered the option of partitioning the system into a number of subsystems. The latter has been shown to be particularly beneficial when the queues are somewhat clustered. Moreover, we have made comparisons of the performance of a multiple-server system with $m$ servers with the situation that (i) the system is attended by a single server with $m$-fold processing rate and (ii) the system is attended by a single server, while the arrival rates at the queues are divided by $m$. These comparisons have suggested that the mean waiting times (at the queues) are generally smaller in the multiple-server case.

Finally, we discuss a number of topics for further research. For the case in which each of the servers visits all queues in a strictly cyclic order (not necessarily identical for each server) and in which the servers incur the same switch-over times per cycle, we consider the question: 'Does each server carry the same load?'. In the papers that have appeared in the literature it is assumed that the servers indeed carry the same load. Based on simulation results, Morris & Wang [27] did not find any significant differences between the loads carried by each of the servers. We did not find significant differences with the aid of the PSA either. This interesting phenomenon might be explained by the following intuitive argument. Consider a system with two servers. Assume that the mean switch-over time incurred per cycle is $s$ for both servers. Then from simple balancing arguments it follows that the respective mean cycle times are given by $\mathrm{EC}_i = s/(1-\tau_i)$, where $\tau_i$ is the load carried by $S_i$, $i = 1, 2$. Suppose $\mathrm{EC}_1 < \mathrm{EC}_2$. Then, because of the fact that $S_1$ is moving around faster, $S_1$ will visit the queues more frequently and hence, will find more customers to be served. The latter would imply $\tau_1 > \tau_2$. Contradiction. We emphasize that this argument is only intuitive. Therefore, it would be interesting to investigate this intriguing question further.

Another interesting point is the following. For the single-server case with cyclic server routing, Levy et al. [24] proved that the amount of work in the system (at any time) is minimal when all queues are served exhaustively, i.e., when the server only leaves a queue when there are no waiting customers left at that queue. It is easy to construct examples showing that for multiple-server polling the latter is not the case in a sample-path sense. Nevertheless, this does not exclude that the monotonicity with respect to the 'exhaustiveness' of the service disciplines may hold for the steady-state amount of work in the system. However, it is not impossible that in the multiple-server case the steady-state system performance may be improved by serving non-exhaustively, because the bunching effect (discussed earlier) would be alleviated. It would be interesting to investigate this monotonicity further.

In the model description in Section 2 the number of servers working at $Q_i$ simultaneously may be restricted by a maximum $m_i$. Numerical experiments have suggested that decreasing the values of $m_i$ deteriorates the system performance, which seems to be intuitively clear.

However, in heavily-loaded systems with negligible switch-over times, tightening the restrictions on the number of servers simultaneously working at the queues may somewhat disperse the server-position distribution and possibly alleviate the coalescing effect, leading to a better system performance. It would be an interesting topic for further research to investigate if putting such an extra restriction on the behaviour of the servers may improve the system performance.

**Acknowledgement** The authors wish to thank O.J. Boxma and J.P.C. Blanc for interesting discussions and for reading earlier versions of this manuscript.

REFERENCES

[1] Ajmone Marsan, M., Donatelli, S., Neri, F. (1990). GSPN models of Markovian multiserver multiqueue systems. *Perf. Eval.* **11**, 227-240.

[2] Ajmone Marsan, M., Donatelli, S., Neri, F. (1991). Multiserver multiqueue systems with limited service and zero walk time. In: *Proc. INFOCOM '91*, 1178-1188.

[3] Ajmone Marsan, M., De Moraes, L.F., Donatelli, S., Neri, F. (1990). Analysis of symmetric nonexhaustive polling with multiple servers. In: *Proc. INFOCOM '90*, 284-295.

[4] Ajmone Marsan, M., De Moraes, L.F., Donatelli, S., Neri, F. (1992). Cycles and waiting times in symmetric exhaustive and gated multiserver multiqueue systems. In: *Proc. INFOCOM '92*, 2315-2324.

[5] Arem, B. van (1990). Queueing Models for Slotted Transmission Systems. *Ph.D. Thesis, Twente University.*

[6] Asmussen, S. (1987). *Applied Probability and Queues.* (Wiley, Chichester).

[7] Bhuyan, L.N., Ghosal, D., Yang, Q. (1989). Approximate analysis of single and multiple ring networks. *IEEE Trans. Comp.* **38**, 1027-1040.

[8] Blanc, J.P.C. (1990). A numerical approach to cyclic-service queueing models. *Queueing Systems* **6**, 173-188.

[9] Blanc, J.P.C. (1990). Cyclic polling systems: limited versus Bernoulli schedules. Report FEW 422, Tilburg University.

[10] Blanc, J.P.C. (1992). Performance evaluation of polling systems by means of the power-series algorithm. *Ann. Oper. Res.* **35**, 155-186.

[11] Blanc, J.P.C. (1993). Performance analysis and optimization with the power-series algorithm. In: *Performance Evaluation of Computer and Communication Systems*, eds. L. Donatiello and R. Nelson (Springer, Berlin), 53-80.

[12] Borst, S.C. (1994). Polling systems with multiple coupled servers. CWI Report BS-R9408.

[13] Borst, S.C., Van der Mei, R.D. (1994). Waiting-time approximations for multiple-server polling systems. CWI Report (in preparation).

[14] Boxma, O.J., Groenendijk, W.P. (1987). Pseudo-conservation laws in cyclic-service systems. *J. Appl. Prob.* **24**, 949-964.

[15] Browne, S., Coffman, E.G. jr., Gilbert, E.N., Wright, P.E.W. (1992). Gated, exhaustive, parallel service. *Prob. Eng. Inf. Sc.* **6**, 217-239.

[16] Browne, S., Kella, O. (1992). Parallel service with vacations. Technical Report Columbia University.

[17] Browne, S., Weiss, G. (1992). Dynamic priority rules when polling with multiple parallel servers. *Oper. Res. Lett.* **12**, 129-137.

[18] Fricker, C., Jaïbi, M.R. (1994). Monotonicity and stability of periodic polling models. *Queueing Systems* **15**, 211-238.

[19] Gamse, B., Newell, G.F. (1982). An analysis of elevator operation in moderate height buildings - II. Multiple elevators. *Transp. Res.* **B 16**, 321-335.

[20] Kamal, A.E., Hamacher, V.C. (1989). Approximate analysis of non-exhaustive multi-server polling systems with applications to local area networks. In: *Comp. Netw. ISDN Syst.* **17**, 15-27.

[21] Kao, E.P.C., Narayanan, K.S. (1991). Analyses of an $M/M/N$ queue with servers' vacations. *EJOR* **54**, 256-266.

[22] Karmarkar, V.V., Kuhl, J.G. (1989). An integrated approach to distributed demand assignment in multiple-bus local networks. *IEEE Trans. Comp.* **38**, 679-695.

[23] Levy, H., Sidi, M., Boxma, O.J. (1990). Dominance relations in polling systems. *Queueing Systems* **6**, 155-171.

[24] Levy, Y., Yechiali, U. (1976). An $M/M/s$ queue with servers' vacations. *INFOR* **14**, 153-163.

[25] Loucks, W.M., Hamacher, V.C., Preiss, B.R., Wong, L. (1985). Short-packet transfer performance in local area ring networks. *IEEE Trans. Comp.* **34**, 1006-1014.

[26] Mitrany, I.L., Avi-Itzhak, B. (1968). A many-server queue with server interruptions. *Oper. Res.* **16**, 628-638.

[27] Morris, R.J.T., Wang, Y.T. (1984). Some results for multi-queue systems with multiple cyclic servers. In: *Performance of Computer-Communication Systems*, eds. W. Bux and H. Rudin (North-Holland, Amsterdam), 245-258.

[28] Neuts, M.F., Lucantoni, D.M. (1979). A Markovian queue with $N$ servers subject to breakdowns and repairs. *Mgmt. Sc.* **25**, 849-861.

[29] Raith, T. (1985). Performance analysis of multibus interconnection networks in distributed systems. In: *Proc. ITC-11*, 662-668.

[30] Stoyan, D. (1974). Some bounds for many-server systems $GI/G/s$. *Math. Oper. Stat.* **5**, 117-129.

[31] Yang, Q., Ghosal, D., Bhuyan, L.N. (1986). Performance analysis of multiple token ring and multiple slotted ring networks. In: *Proc. 1986 Comp. Netw. Symp.*, 79-86.

[32] Zafirovic-Vukotic, M., Niemegeers, I.G., Valk, D.S. (1988). Performance modelling of slotted ring protocols in HSLAN's. *IEEE J. Sel. Areas Comm.* **6**, 1001-1024.