Queueing Syst
DOI 10.1007/s11134-014-9409-7

Exact analysis of the M/M/k/setup class of Markov chains via recursive renewal reward

Anshul Gandhi · Sherwin Doroudi · Mor Harchol-Balter · Alan Scheller-Wolf

Received: 13 May 2013 / Revised: 20 March 2014

Abstract The M/M/k/setup model, where there is a penalty for turning servers on, is common in data centers, call centers, and manufacturing systems. Setup costs take the form of a time delay, and sometimes there is additionally a power penalty, as in the case of data centers. While the M/M/1/setup was exactly analyzed in 1964, no exact analysis exists to date for the M/M/k/setup with $k > 1$. In this paper, we provide the first exact, closed-form analysis for the M/M/k/setup and some of its important variants including systems in which idle servers delay for a period of time before turning off or can be put to sleep. Our analysis is made possible by a new way of combining renewal reward theory and recursive techniques to solve Markov chains with a repeating structure. Our renewal-based approach uses ideas from renewal reward theory and busy period analysis to obtain closed-form expressions for metrics of interest such as the transform of time in system and the transform of power consumed by the system. The simplicity, intuitiveness, and versatility of our renewal-based approach makes it useful for analyzing Markov chains far beyond the M/M/k/setup. In general, our renewal-based approach should be used to reduce the analysis of any 2-dimensional Markov chain which is infinite in at most one dimension and repeating to the problem of solving a system of polynomial equations. In the case where all transitions in

A. Gandhi (✉) · M. Harchol-Balter
School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, USA
e-mail: anshulg@cs.cmu.edu

M. Harchol-Balter
e-mail: harchol@cs.cmu.edu

S. Doroudi · A. Scheller-Wolf
Tepper School of Business, Carnegie Mellon University, Pittsburgh, PA, USA
e-mail: sdoroudi@andrew.cmu.edu

A. Scheller-Wolf
e-mail: awolf@andrew.cmu.edu

Published online: 21 April 2014

the repeating portion of the Markov chain are skip-free and all up/down arrows are unidirectional, the resulting system of equations will yield a closed-form solution.

Keywords Queueing theory · Performance · Resource allocation · Renewal reward · Repeating chains

Mathematics Subject Classification 60K25 · 68M20 · 60J25

1 Introduction

Setup times (a.k.a. exceptional first service) are a fundamental component of computer systems and manufacturing systems, and therefore, they have always played an important role in queueing theoretic analysis. In manufacturing systems, it is very common for a job that finds a server idle to wait for the server to "warm up" before service is initiated. In retail and hospitals, the arrival of customers may necessitate bringing in an additional human server, which requires a setup time for the server to arrive. In computer systems, setup times are once again at the forefront of research, as they are the key issue in dynamic capacity provisioning for data centers.

In data centers, it is desirable to turn idle servers off, or reallocate the servers, to save power. This is because idle servers burn power at 60–70 % of the peak rate, and so leaving servers on when idle is wasteful [4]. Unfortunately, most companies are hesitant to turn off idle servers because the setup time needed to restart these servers is very costly; the typical setup times for servers is 200 s, while a job's service requirement is typically less than 1 s [6,16]. Not only is the setup time prohibitive, but power is also burned at peak rate during the entire setup period, although the server is still not functional. Thus, it is not at all obviously true that turning off idle servers is advantageous.

Many ideas have been proposed to minimize the number of times that servers in a data center must undergo setup. One major line of research involves load prediction techniques [5,12,16,21]. In the case where load is unpredictable, research has turned to looking at policies such as delayedoff, which delays turning off an idle server for some fixed amount of time, in anticipation of a new arrival [9,11,14]. Another line of research involves reducing setup times by developing low-power sleep modes [11,19].

Surprisingly, despite all the importance associated with the setup times, very little is known about their analysis. The M/G/1 with setup times was analyzed in 1964 by Welch [26]. The analysis of an M/M/k system with setup times, which we refer to as M/M/k/setup, however, has remained elusive, owing largely to the complexity of the underlying Markov chain. (Fig. 1 shows an M/M/k/setup with exponentially distributed setup times.) In 2010, various analytic approximations for the M/M/k/setup were proposed in [10]. These approximations work well provided that either load is low or the setup time is low. The M/M/∞/setup was also analyzed in [10] and found to exhibit product form. Other than the above, no progress has been made on the M/M/k/setup. Even less is known about the M/M/k/setup/delayedoff, where idle servers delay for a finite amount of time before turning off, or the M/M/k/setup/sleep, where idle servers can either be turned off (high setup time, zero power) or put to sleep (lower setup time, low power). Section 3 describes these models in

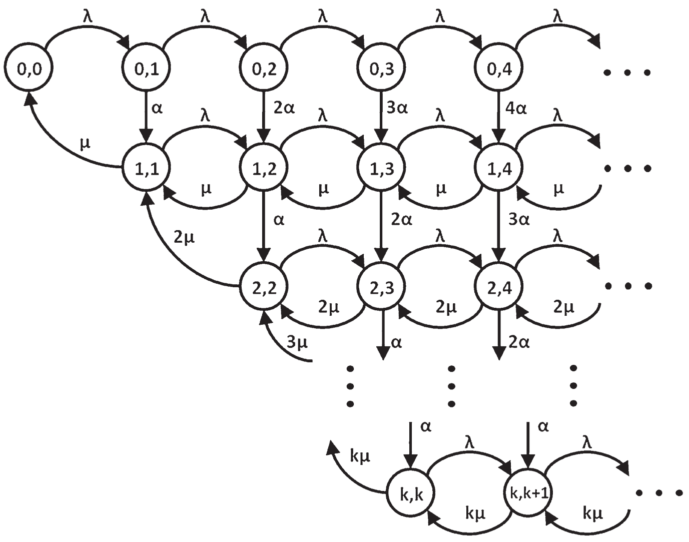

Fig. 1 M/M/k/setup Markov chain. Each state is denoted by the pair (i, j), where i is the number of on servers, and j is the number of jobs in the system. The number of servers in setup is $\min\{j - i, k - i\}$

greater detail. Section 2 describes related prior research, including existing methods for solving general Markov chains with a repeating structure.

This paper is the first to derive an exact, closed-form solution for the M/M/k/setup, the M/M/k/setup/delayedoff, and the M/M/k/setup/sleep. We obtain the Laplace transform of response time, the z-transform of power consumption, and other important metrics for all of the above models.

Our solution is made possible by a new way of combining renewal reward theory and recursive techniques for solving Markov chains with a repeating structure. Our renewal-based approach uses ideas from renewal reward theory to obtain the metrics of interest, while utilizing certain recursion lemmas about the chain. Given that we leverage recursions and the repeating structure of the Markov chain, our approach can be considered similar in spirit to matrix-analytic methods. However, unlike matrix-analytic methods [17], our approach does not require finding the "rate" matrix. Another strength of our approach is that it is very intuitive and is simple enough to be taught in an elementary stochastic processes course. Thus, our technique provides a new and powerful framework for solving Markov chains that are typically analyzed via matrix-analytic methods.

In general, our renewal-based approach should be able to reduce the analysis of any 2-dimensional Markov chain which is finite in one dimension, say the vertical dimension, and infinite (with repeating structure) in the other (horizontal dimension) to the problem of solving a system of polynomial equations. Further, if in the repeating portion all horizontal transitions are skip-free and all vertical transitions are unidirectional, the resulting system of equations will be at most quadratic, yielding a closed-form solution (see Sect. 10 and Fig. 6 for more details). We thus anticipate that our renewal-based approach will prove useful to other researchers in analyzing many new problems.

2 Prior work

The few papers that have looked at the M/M/k/setup are discussed in Sect. 1. For the M/M/k/setup/delayedoff, only iterative matrix-analytic approaches have been used [9]. No analysis exists for M/M/k/setup/sleep. We now discuss papers that have considered repeating Markov chains and have proposed techniques for solving these. We then comment on how these techniques might or might not apply to the M/M/k/setup.

2.1 Matrix-analytic-based approaches

Matrix-analytic methods are a common approach for analyzing Markov chains with repeating structure. Such approaches are typically numerical, generally involving iteration to find the rate matrix, R. These approaches do not, in general, lead to closed forms or to any intuition, but are very useful for evaluating chains under different parameters.

There are cases where it is known that the R matrix can be stated explicitly [17]. This typically involves using a combinatorial interpretation for the R matrix. As described in [17], the class of chains for which the combinatorial view is tractable is narrow. However, in [25], the authors show that the combinatorial interpretation extends to a broader class of chains. Their class does not include the M/M/k/setup, however, which is more complicated because the transition (setup) rates are not independent of the number of jobs in system. Much research has been done on improving matrix-analytic methods to make the iteration faster. An example is [24], which develops a fast iterative procedure for finding the rate matrix for a broader class of chains than that in [25]. The authors in [24] also provide an explicit solution for the rate matrix in terms of infinite sums. Their algorithm provides a solution technique that applies to a class of chains called tree-like QBDs that includes the M/M/k/setup, although the authors do not make reference to the M/M/k/setup or any variants thereof.

2.2 Generating function-based approaches

Generating functions have also been applied to solve chains with a repeating structure. Like matrix-analytic methods these are not intuitive: Generating function approaches involve guessing the form of the solution and then solving for the coefficients of the guess, often leading to long computations. In theory, they can be used to solve very general chains (see for example [1]). We initially tried applying a generating function approach to the M/M/2/setup and found it to be incredibly complex and without intuition. This led us to seek a simpler and more intuitive approach.

2.3 M/M/k with vacations

Many papers have been published regarding the M/M/k system with vacations; see, for example, [18,23,27,28]. While the Markov chain for the M/M/k with vacations looks similar to the M/M/k/setup, the dynamics of the two systems are very different.

A server takes a vacation as soon as it is idle, and there are no jobs in the queue. By contrast, a setup time is initiated by jobs arriving to the queue. In almost all of the papers involving vacations, the vacation model is severely restricted, allowing only a fixed group of servers to go on vacation at once. This is very different from our system in which any number of servers may be in setup at any time. The model in [18] comes closest to our model, although the authors of that study use generating functions and assume that *all* idle servers are on vacation, rather than one server being in setup for each job in queue, which makes the transitions in their chain to be independent of the number of jobs.

2.4 Restricted models of M/M/k with setup

There have been a few papers [2,3,10] that consider a very restricted version of the M/M/k/setup, wherein at most one server can be in setup at a time. There has also been prior work [20] that considers an M/M/k system wherein a fixed subset of servers can be turned on and off based on load. The underlying Markov chains for all of these restricted systems are analytically tractable and lead to very simple closed-form expressions, since the rate at which servers turn on is always fixed. Our M/M/k/setup system is more general, allowing any number of servers to be in setup. This makes our problem much more challenging.

2.5 How our work differs from all of the above

To the best of our knowledge, we are the first to derive exact closed-form results for the M/M/k/setup problem, with $k > 1$. Our solution was made possible by a new way of combining renewal reward theory and recursive techniques. Our approach results in exact solutions, does not require any iteration, and does not involve infinite sums. Importantly, our approach is highly intuitive and very easy to apply. Using our renewal-based approach, we go much further than the M/M/k setup, deriving exact closed-form results for important variants such as the M/M/k/setup/delayed-off and the M/M/k with multiple types of setups, neither of which has been solved analytically.

3 Model

In our model jobs arrive according to a Poisson process with rate λ and are served at rate $\mu = \frac{1}{E[S]}$, where S denotes the job size and is exponentially distributed. For stability, we assume that $k \cdot \mu > \lambda$, where k is the number of servers in the system.

3.1 M/M/k/setup

In the M/M/k/setup system, each of the k servers is in one of three states: off, on (being used to serve a job), or setup. When a server is on or in setup, it consumes peak power of P_{peak} watts. When a server is off, it consumes zero power. Thus, when servers are not in use, they are immediately turned off to save power. Every arriving job that comes into the system picks an off server, if one exists, and puts it into setup mode; the job then joins the queue. We use I to denote the setup times, with $E[I] = \frac{1}{\alpha}$. Unless stated otherwise, we assume that setup times are exponentially distributed. When a job completes service at a server, say server s_1, and there are no remaining jobs left in the queue, then server s_1 is immediately turned off. However, if the queue is not empty, then server s_1 is not turned off, and the job at the head of the queue is directed to server s_1. Note that if the job at the head of the queue was already waiting on another server, say server s_2, in setup mode, the job at the head of the queue is still directed to server s_1. At this point, if there is a job in the queue that did not setup an off server on arrival (because there were no off servers), then server s_2 continues to be in setup for this job. If no such job exists in the queue, then server s_2 is turned off.

The Markov chain for the M/M/k/setup system is shown in Fig. 1. Each state is denoted by the pair (i, j), where i is the number of on servers, and j is the number of jobs in the system. Thus, the number of servers in setup is $\min\{j - i, k - i\}$. Note that the Markov chain is infinite in one dimension.

3.2 M/M/k/setup/delayedoff

The M/M/k/setup/delayedoff system is the same as the M/M/k/setup system, except that idle servers are not immediately turned off. Specifically, when a job completes service at a server, say server s_1, and there are no remaining jobs in the queue, s_1 remains waiting in the idle state for an exponentially distributed amount of time with mean $t_{\text{wait}} = \frac{1}{\beta}$. If a new job arrives while server s_1 is waiting, the job is immediately directed to s_1, which is already on. However, if no jobs arrive during server s_1's waiting period, then server s_1 is turned off. Intuitively, a higher t_{wait} results in lower response time, since servers are on longer, but may also increase power usage, since idle servers consume significant power.

The Markov chain for the M/M/k/setup/delayedoff system is shown in Fig. 2. The chain is the same as that for M/M/k/setup, except for the new gray shaded states which represent states with idle servers. As before, each state is denoted by the pair (i, j), where i is the number of on or idle servers, and j is the number of jobs in the system. For the M/M/k/setup/delayedoff system, each server can be in one of four states: off, on (busy), idle, or setup. If $i < j$, then the number of servers in setup is $\min\{j - i, k - i\}$, and there are no idle servers. If $i > j$ (gray shaded states), the number of idle servers is $(i - j)$, and there are no servers in setup. If $i = j$, no servers are idle or in setup.

3.3 M/M/k/setup/sleep

The M/M/k/setup/sleep is motivated by servers with sleep modes [11,19], which allow an idle server to either be turned off or put to sleep. When a server is turned off, it consumes zero power. However, turning on an off server requires an exponentially distributed setup time, with rate α. By contrast, when a server is sleeping, it consumes some non-zero power, P_{sleep} watts, which is usually much smaller than the idle power, P_{idle} watts [11,19]. When a sleeping server is turned on, it requires an exponentially

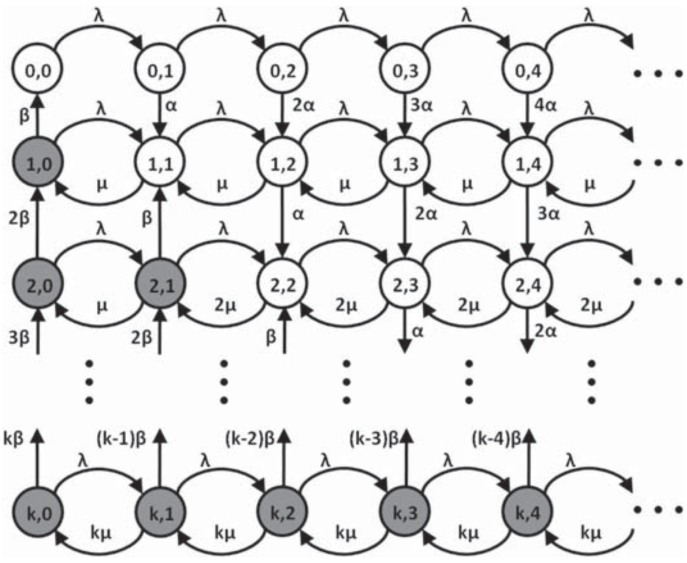

Fig. 2 M/M/k/setup/delayedoff Markov chain. Each state is denoted by the pair (i, j), where i is the number of on or idle servers, and j is the number of jobs in the system. If $i < j$, then the number of servers in setup is $\min\{j - i, k - i\}$, and there are no idle servers. If $i > j$ (*gray shaded states*), the number of idle servers is $(i - j)$, and there are no servers in setup. If $i = j$, no servers are idle or in setup

distributed setup time, with rate $\omega > \alpha$. Thus, there is a tradeoff between turning off an idle server versus putting it to sleep.

One simple idea that leverages sleep states is to designate some subset of the k servers, say the first s servers, to "sleep" when idle, whereas the remaining $(k - s)$ servers are turned off when idle. An interesting question is what is a good value of s. To answer this question we introduce the M/M/k/setup/sleep model, which is the same as the M/M/k/setup, except that $s \leq k$ servers have a fast setup rate of ω and $(k - s)$ servers have a slow setup rate of α (see Fig. 3). As we will see later, the tradeoff between mean response time and mean power is highly sensitive to the choice of s (see Fig. 7c, d).

For tractability, we make the following assumptions about the M/M/k/setup/sleep model: (i) In any group of servers in setup, we assume that the servers that have a fast setup rate (ω) complete setting up first. Thus, if we are in state (i, j) with $i < s$ (gray shaded states in Fig. 3), the first $(s - i)$ servers in setup will have a fast setup rate. Note that the i servers already on in state (i, j), with $i \leq s$, are those that had a fast setup rate. Thus, when we have $i \leq s$ servers busy, and a server is no longer in use, we put the server to sleep (as opposed to turning it off). (ii) If we have $i > s$ servers busy, and a server is no longer in use, we turn the server off (as opposed to putting it to sleep). This assumption allows us to save a lot of power when load goes down since off servers consume zero power. The above two assumptions are primarily for tractability of the M/M/k/setup/sleep Markov chain. In practice, ω is significantly higher than α. In this regime, we simulated an M/M/k/setup/sleep system with and without the above two assumptions and found the results to be qualitatively unchanged.

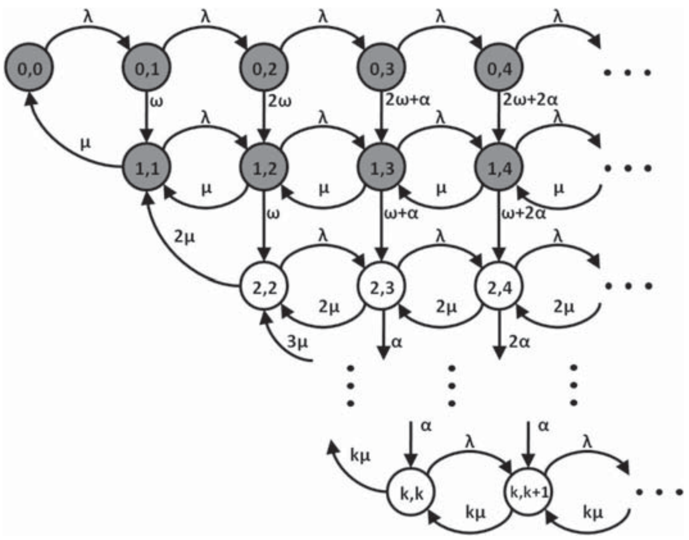

Fig. 3 M/M/k/setup/sleep Markov chain. Each state is denoted by the pair (i, j), where i is the number of on servers, and j is the number of jobs in the system. The number of servers in fast setup is s, which in this case is $s = 2$. The number of servers in setup is $\min\{j - i, k - i\}$. If $i < s$ (*gray shaded states*), the first $(s - i)$ servers setting up have a fast setup rate, ω, while the other servers in setup have a slow setup rate, α

4 Our renewal-based approach

In this section, we provide a high-level description of our renewal-based approach, which yields exact, closed-form solutions for a range of Markov chains, including the M/M/k/setup (see Sects. 5, 6 and 7), the M/M/k/setup/delayedoff (see Sect. 8) and the M/M/k/setup/sleep (see Sect. 9).

Our renewal-based approach works by deriving the *expected "reward" earned per unit time in a Markov chain*, where the reward could be any quantity of interest. In the context of our M/M/k/setup problem, the reward earned at time t, $R(t)$, could be the number of jobs in system at time t, the square of the number of jobs in system, the current power usage, the number of servers that are on, or any other reward that can be expressed as a function of the state of the Markov chain.

To analyze the average rate of earning reward, we designate a *renewal state*, say $(0, 0)$,[1] which we call the *home state*, and then consider a *renewal cycle* to be the process of moving from the home state back to the home state. By renewal-reward theory, the average rate of earning reward is the same as the mean reward earned over a renewal cycle, which we denote by $\mathcal{R}$, divided by the mean length of the renewal cycle, denoted by $\mathcal{T}$.

[1] In principle, any state can be chosen as the renewal state, but some states allow for an easier (or shorter) analysis.

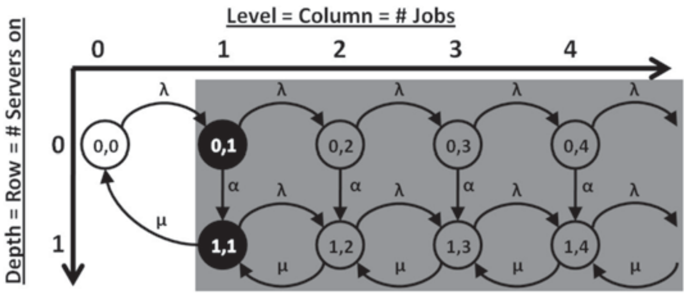

Fig. 4 M/M/1/setup Markov chain with the repeating portion highlighted in *gray* and the border states shaded *black*

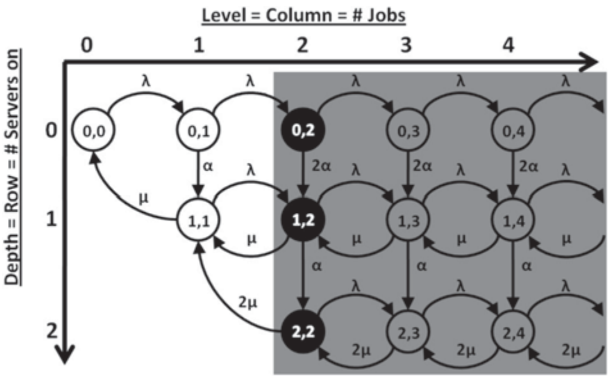

Fig. 5 M/M/2/setup Markov chain with the repeating portion highlighted in *gray* and the border states shaded *black*

$$\text{Average rate of earning} = \frac{\mathcal{R}}{\mathcal{T}} = \frac{E\left[\int_{\text{cycle}} R(t)\,\mathrm{d}t\right]}{E\left[\int_{\text{cycle}} 1\,\mathrm{d}t\right]}$$

For example, if the goal is to find the mean number of jobs, $E[N]$, for our chain, we simply define $R(t)$ to be the number of jobs at time t, which can be obtained from the state of the Markov chain at time t.

It turns out that the quantities $\mathcal{T}$ and $\mathcal{R}$ are very easy to compute! Consider a Markov chain, such as that in Fig. 4 or Fig. 5. The *repeating portion* of the chain is shown in gray. There are a finite number of *border states* which sit at the edge of the repeating chain and are colored black. We will see that computing $\mathcal{T}$ and $\mathcal{R}$ basically reduces to writing one equation for each border state[2]. For the case of $\mathcal{T}$, we will need the mean time to move one step left from each border state. For the case of $\mathcal{R}$, we will need the mean reward earned when moving one step left from each border state. Computing these border state quantities is made very easy via some neat recursion lemmas. We

[2] Several techniques in the literature such as matrix-analytic methods [17] and stochastic complementation [22] also deal with border states, although none of them involve renewal-reward theory.

demonstrate this process in the examples below. There are a few details which we will defer until after these examples. For instance, in general, it is necessary to also add equations for the non-repeating portion of the Markov chain. See Sects. 7 and 10 for more details on our renewal-based approach. Note that matrix analytic methods also sometimes make use of quantities such as the time to move one step left in a repeating chain [17].

5 M/M/1/setup

In this section, we illustrate our renewal-based approach by applying it to the simple M/M/1/setup system, whose Markov chain is shown in Fig. 4. Here, the state of the system is represented as (i, j), where $i \in \{0, 1\}$ is the number of servers on and $0 \le j < \infty$ is the number of jobs in the system. In general, i represents the *depth* (or row number) of the state, and j represents the *level* (or column number) of the state. We start by deriving $E[N]$, the mean number of jobs, and then move to more complex metrics. We choose the renewal state to be (0, 0) and we define the reward earned at time t, $R(t)$, to be $N(t)$, the number of jobs in the system at time t. As explained in Sect. 4, all we need is $\mathcal{T}$ and $\mathcal{R}$.

5.1 Deriving $\mathcal{T}$ via $T^L_{0,1}$ and $T^L_{1,1}$

$\mathcal{T}$ is the mean time to get from our home state (0, 0) back to (0, 0). This can be viewed as $\frac{1}{\lambda}$, the mean time until we leave (0, 0) (which takes us to (0, 1)) plus the mean time to get home from (0, 1). We make the further observation that the mean time to get home from (0, 1) is equal to $T^L_{0,1}$ (using notation from Table 1), the mean time to move left one level from (0, 1) (since moving left can only put us in (0, 0)). We thus have

$$\mathcal{T} = \tfrac{1}{\lambda} + T^L_{0,1} \tag{1}$$

We now need an equation for $T^L_{0,1}$ for the border state (0, 1), which will require looking at the other border state, (1, 1), as well. Starting with border state (1, 1), it is clear that $T^L_{1,1}$ is simply the mean length of an M/M/1 busy period, B_1. Thus, we have

$$T^L_{1,1} = B_1 = \tfrac{1}{\mu-\lambda} \tag{2}$$

$T^L_{0,1}$ involves waiting in state (0, 1) for expected time $\frac{1}{\alpha+\lambda}$, before conditioning on where we transition to next. If we go to state (1, 1) we need an additional $T^L_{1,1}$. However if we go to state (0, 2) we need to add on the time to move one step left from (0, 2) (which by Fig. 4 takes us to (1, 1)) and then an additional $T^L_{1,1}$. That is,

$$T^L_{0,1} = \frac{1}{\lambda+\alpha} + \frac{\alpha}{\lambda+\alpha} \cdot T^L_{1,1} + \frac{\lambda}{\lambda+\alpha}\left(T^L_{0,2} + T^L_{1,1}\right) \tag{3}$$

It is now time to invoke one of our recursion lemmas, which holds for any M/M/k/setup chain:

Lemma 1 (Recursion lemma for mean time) *For the M/M/k/setup, the mean time to move one step left from state (i, j), $T^L_{i,j}$, is the same for all $j \geq k$.*

Lemma 1 follows from the fact that the repeating portion of the Markov chain is identical for all states in a given row. The full proof of Lemma 1 (along with the proofs of all other lemmas) is presented in Appendix 1.

Using Lemma 1, we replace $T^L_{0,2}$ in Eq. (3) with $T^L_{0,1}$ to get

$$T^L_{0,1} = \tfrac{1}{\lambda+\alpha} + \tfrac{\alpha}{\lambda+\alpha} \cdot T^L_{1,1} + \tfrac{\lambda}{\lambda+\alpha}\left(T^L_{0,1} + T^L_{1,1}\right) \tag{4}$$

Finally, noting that $T^L_{1,1} = B_1$ from Eq. (2), we have that

$$\begin{aligned} T^L_{0,1} &= \frac{1}{\lambda+\alpha} + \frac{\alpha}{\lambda+\alpha} \cdot B_1 + \frac{\lambda}{\lambda+\alpha}\left(T^L_{0,1} + B_1\right) \\ \implies T^L_{0,j} = T^L_{0,1} &= \frac{1 + (\lambda+\alpha)B_1}{\alpha} \end{aligned} \tag{5}$$

Substituting $T^L_{0,1}$ from above into Eq. (1) gives us $\mathcal{T}$:

$$\mathcal{T} = \tfrac{\mu(\lambda+\alpha)}{\lambda\alpha(\mu-\lambda)} \tag{6}$$

5.2 Deriving $\mathcal{R}$ via $R^L_{0,1}$ and $R^L_{1,1}$

$\mathcal{R}$ denotes the reward earned in moving from (0, 0) back to (0, 0). Observing that we earn 0 reward in state (0, 0) (because there are no jobs in the system in that state), and observing that from state (0, 0) we can only next move to (0, 1), we have (using notation from Table 1):

$$\mathcal{R} = R^L_{0,1} \tag{7}$$

It now remains to compute the reward earned in moving one step left from (0, 1), which will require looking at the other border state, (1, 1), as well.

To do this, we invoke another recursion lemma, which again holds for any M/M/k/setup system:

Lemma 2 (Recursion lemma for mean reward) *For the M/M/k/setup, the mean reward earned in moving one step left from state $(i, j+1)$, $R^L_{i,j+1}$, satisfies $R^L_{i,j+1} = R^L_{i,j} + T^L_{i,j}$ for all $j \geq k$, where the reward tracks the number of jobs in the system.*

Table 1 Variables used in our analysis of $E[N]$

Variable	Description
$\mathcal{T}$	Mean length of the renewal cycle
$\mathcal{R}$	Mean reward earned during a renewal cycle
$T^L_{i,j}$	Mean time until we first move one level left of (i, j), starting from (i, j)
$R^L_{i,j}$	Mean reward earned until we first move one level left of (i, j), starting from (i, j)
$p^L_{i\to d}$	Probability that after we first move one level left from state (i, j), we are at depth d
B_k	Mean length of an $M/M/k$ busy period

Applying Lemma 2 to the Markov chain shown in Fig. 4, we have

$$\begin{aligned} R^L_{1,1} &= \frac{1}{\lambda+\mu}\cdot 1 + \frac{\mu}{\lambda+\mu}\cdot 0 + \frac{\lambda}{\lambda+\mu}\left(R^L_{1,2}+R^L_{1,1}\right) \\ &= \frac{1}{\lambda+\mu} + \frac{\lambda}{\lambda+\mu}\left((R^L_{1,1}+T^L_{1,1})+R^L_{1,1}\right) \\ &= \frac{1}{\lambda+\mu} + \frac{\lambda}{\lambda+\mu}\left((R^L_{1,1}+B_1)+R^L_{1,1}\right) \quad \text{(from Eq. (2))} \qquad (8) \\ \implies R^L_{1,1} &= \frac{1+\lambda B_1}{\mu-\lambda} \qquad (9) \end{aligned}$$

Similarly, for border state (0, 1), we have

$$\begin{aligned} R^L_{0,1} &= \frac{1}{\lambda+\alpha}\cdot 1 + \frac{\alpha}{\lambda+\alpha}\cdot R^L_{1,1} + \frac{\lambda}{\lambda+\alpha}\left(R^L_{0,2}+R^L_{1,1}\right) \\ &= \frac{1}{\lambda+\alpha} + \frac{\alpha}{\lambda+\alpha}\cdot R^L_{1,1} \\ &\quad + \frac{\lambda}{\lambda+\alpha}\left((R^L_{0,1}+T^L_{0,1})+ R^L_{1,1}\right) \quad \text{(from Lemma 2)} \\ \implies R^L_{0,1} &= \frac{1+\lambda T^L_{0,1}+(\lambda+\alpha)R^L_{1,1}}{\alpha} \qquad (10) \end{aligned}$$

Substituting $R^L_{0,1}$ from above into Eq. (7) gives us $\mathcal{R}$:

$$\mathcal{R} = \frac{\mu(\lambda+\alpha)(\mu-\lambda+\alpha)}{\alpha^2(\mu-\lambda)^2} \qquad (11)$$

5.3 Deriving $E[N]$

Since $E[N] = \frac{\mathcal{R}}{\mathcal{T}}$, combining Eqs. (6) and (11), we get

$$E[N] = \frac{\mathcal{R}}{\mathcal{T}} = \frac{\lambda}{\alpha} + \frac{\lambda}{\mu-\lambda} \qquad (12)$$

Table 2 Variables used in our transform analyses

Variable	Description
$\dot{\mathcal{R}}$	Mean reward earned (for z-transform) during a renewal cycle
$\dot{\mathcal{E}}$	Mean reward earned (for transform of power) during a renewal cycle
$\dot{R}^L_{i,j}$	Mean reward earned (for z-transform) until we first move one level left of (i, j), starting from (i, j)
$\dot{E}^L_{i,j}$	Mean reward earned (for z-transform of power) until we first move one level left of (i, j), starting from (i, j)

The second term in the right-hand side of Eq. (12) can be identified [15] as the mean number of jobs in an M/M/1 system (without setup). Thus, Eq. (12) is consistent with the known decomposition property for the M/M/1/setup system [26].

5.4 Deriving $\widehat{N}(z)$ and $\widetilde{T}(s)$

Deriving the z-transform of the number of jobs, $\widehat{N}(z) = E[z^N]$, is just as easy as deriving $E[N]$. The only difference is that our reward function is now $R(t) = z^{N(t)}$, where $N(t)$ is again the number of jobs in the system at time t. Thus,

$$\widehat{N}(z) = E[z^N] = \frac{\dot{\mathcal{R}}}{\mathcal{T}},$$

where $\dot{\mathcal{R}} = E\left[\int_{\text{cycle}} z^{N(t)}\mathrm{d}t\right]$ and $\mathcal{T}$ is the same as before.

We will again invoke a recursion lemma which applies to any M/M/k/setup (using notation from Table 2):

Lemma 3 (Recursion lemma for transform of reward) *For the M/M/k/setup, $\dot{R}^L_{i,j+1} = z \cdot \dot{R}^L_{i,j}$, for all $j \geq k$, where $\dot{R}$ tracks the z-transform of the number of jobs in the system.*

Let us now express $\dot{\mathcal{R}}$ by conditioning on the first step from (0, 0):

$$\dot{\mathcal{R}} = \tfrac{1}{\lambda} + \dot{R}^L_{0,1} \tag{13}$$

We again need one equation per border state:

$$\dot{R}^L_{1,1} = \frac{1}{\lambda+\mu} \cdot z + \frac{\lambda}{\lambda+\mu}\left(z \cdot \dot{R}^L_{1,1} + \dot{R}^L_{1,1}\right)$$
$$\dot{R}^L_{0,1} = \frac{1}{\lambda+\alpha} \cdot z + \frac{\alpha}{\lambda+\alpha} \cdot \dot{R}^L_{1,1} + \frac{\lambda}{\lambda+\alpha}\left(z \cdot \dot{R}^L_{0,1} + \dot{R}^L_{1,1}\right)$$

Solving the above system and substituting $\dot{R}^L_{0,1}$ into Eq. (13) allows us to express $\dot{\mathcal{R}}$ in closed form. This gives us $\widehat{N}(z)$, after some algebra, as follows:

$$\widehat{N}(z) = E[z^N] = \tfrac{\dot{\mathcal{R}}}{\mathcal{T}} = \tfrac{\alpha(\mu-\lambda)}{(\mu-\lambda z)(\alpha+\lambda-\lambda z)} \tag{14}$$

To get the Laplace transform of response time, $\widetilde{T}(s)$, we use the distributional Little's Law [13] (since M/M/1/setup is a First-In-First-Out system):

$$\widetilde{T}(s) = \widehat{N}\left(1 - \tfrac{s}{\lambda}\right) = \tfrac{\alpha(\mu-\lambda)}{(s+\alpha)(\mu+s-\lambda)} \tag{15}$$

5.5 Deriving $\widehat{P}(z)$

We now derive $\widehat{P}(z)$, the z-transform of the power consumed for the M/M/1/setup. The server consumes zero power when it is off, but consumes peak power, P_{peak} watts, when it is on or in setup. This time, the reward is simply the transform of the energy consumed over the renewal cycle, $\dot{\mathcal{E}} = E\left[\int_{\text{cycle}} z^{P(t)}\mathrm{d}t\right]$, where $P(t)$ is the power consumed at time t. We begin with the recursive lemma for $\dot{E}^L_{i,j}$, just like we had Lemma 3 for $\dot{R}^L_{i,j}$.

Lemma 4 (Recursion lemma for transform of power) *For the M/M/k/setup,* $\dot{E}^L_{i,j+1} = \dot{E}^L_{i,j} = T^L_{i,j} \cdot z^{k\cdot P_{\text{peak}}}$*, for all* $j \geq k$.

Lemma 4 gives us $\dot{E}^L_{i,j}$ in closed form, in terms of $T^L_{i,j}$. Following the usual renewal-reward approach, we get

$$\widehat{P}(z) = E[z^P] = \frac{\dot{\mathcal{E}}}{\mathcal{T}} = \frac{\alpha(\mu-\lambda) + \lambda(\mu+\alpha)z^{P_{\text{peak}}}}{\mu(\lambda+\alpha)} \tag{16}$$

6 M/M/2/setup

The M/M/2/setup chain shown in Fig. 5 is analyzed similarly to the M/M/1/setup, except that there are now three border states, (0, 2), (1, 2), and (2, 2). The only complication is that when moving one level left from a given state, the resulting row is non-deterministic. For example, when moving left from (1, 3) in Fig. 5, we may end up in row 1 at (1, 2) or row 2 at (2, 2). We use $p^L_{i\to d}$ to denote the probability that once we move one level left from (i, j), we will be at depth d.[3] The following lemma proves that $p^L_{i\to d}$ is independent of j for all states (i, j) in the repeating portion.

Lemma 5 (Recursion lemma for probability) *For the M/M/k/setup, for each* $0 \leq d \leq k$*, and for each* $0 \leq i \leq k$*,* $p^L_{i\to d}$ *is the same for all* $j \geq k$.

Thus, it suffices to compute $p^L_{i\to d}$ for the border states. These probabilities are used in Sect. 6.2.

6.1 Deriving $p^L_{i\to d}$

Solving for the $p^L_{i\to d}$ is easiest "bottom-up" (starting from the greatest depth, i). For $i = 2$, we have $p^L_{2\to 2} = 1$ for all $j > 2$, since we stay at depth 2 after moving left.

[3] See Footnote 2.

For $i = 1$ and $i = 0$, we follow the same approach of conditioning on the first step and using recursion lemmas:

$$p^L_{1\to 1} = \frac{\mu}{\lambda+\mu+\alpha} + \frac{\lambda}{\lambda+\mu+\alpha}\left(p^L_{1\to 1}\right)^2 \tag{17}$$

$$p^L_{0\to 1} = \frac{2\alpha}{\lambda+2\alpha}\left(p^L_{1\to 1}\right) + \frac{\lambda}{\lambda+2\alpha}\left(p^L_{0\to 1}\right)\left(p^L_{1\to 1}\right) \tag{18}$$

Equations (17) and (18) can now be solved in closed form since they are of degree at most 2. Note that $p^L_{1\to 2} = 1 - p^L_{1\to 1}$ and $p^L_{0\to 2} = 1 - p^L_{0\to 1}$.

6.2 Deriving $\widehat{\mathbf{N}}(\mathbf{z})$ via $\dot{R}^L_{0,2}$ $\dot{R}^L_{1,2}$, and $\dot{R}^L_{2,2}$

To derive $\widehat{N}(z) = E[z^N]$, we again find $\dot{\mathcal{R}}$ and $\mathcal{T}$, where $\dot{\mathcal{R}} = E\left[\int_{\text{cycle}} z^{N(t)}\mathrm{d}t\right]$, and $\mathcal{T} = E\left[\int_{\text{cycle}} 1\mathrm{d}t\right] = \dot{\mathcal{R}}\Big|_{z=1}$. Using $(1, 1)$ as our renewal state and the same arguments as in Sect. 5.4, we have

$$\dot{\mathcal{R}} = \frac{z}{\lambda+\mu} + \frac{\mu}{\lambda+\mu}\left(\frac{1}{\lambda} + \frac{z}{\lambda+\alpha} + \frac{\lambda}{\lambda+\alpha}\cdot\dot{R}^L_{0,2}\right) + \frac{\lambda}{\lambda+\mu}\cdot\dot{R}^L_{1,2} \tag{19}$$

It now remains to compute the reward equations for the border states: $\dot{R}^L_{0,2}$, $\dot{R}^L_{1,2}$, and $\dot{R}^L_{2,2}$.

$$\dot{R}^L_{2,2} = \frac{z^2}{\lambda+2\mu} + \frac{\lambda}{\lambda+2\mu}\left(z\cdot\dot{R}^L_{2,2} + \dot{R}^L_{2,2}\right) \tag{20}$$

$$\begin{aligned}\dot{R}^L_{1,2} = {} & \frac{z^2}{\lambda+\mu+\alpha} + \frac{\alpha}{\lambda+\mu+\alpha}\cdot\dot{R}^L_{2,2} \\ & + \frac{\lambda}{\lambda+\mu+\alpha}\left(z\cdot\dot{R}^L_{1,2} + \left(p^L_{1\to 1}\right)\dot{R}^L_{1,2} + \left(1 - p^L_{1\to 1}\right)\dot{R}^L_{2,2}\right)\end{aligned} \tag{21}$$

$$\begin{aligned}\dot{R}^L_{0,2} = {} & \frac{z^2}{\lambda+2\alpha} + \frac{2\alpha}{\lambda+2\alpha}\cdot\dot{R}^L_{1,2} \\ & + \frac{\lambda}{\lambda+2\alpha}\left(z\cdot\dot{R}^L_{0,2} + \left(p^L_{0\to 1}\right)\dot{R}^L_{1,2} + \left(1 - p^L_{0\to 1}\right)\dot{R}^L_{2,2}\right)\end{aligned} \tag{22}$$

Solving the above system of linear equations and substituting $\dot{R}^L_{0,2}$ and $\dot{R}^L_{1,2}$ into Eq. (19) allows us to solve for $\widehat{N}(z)$ in closed form as follows:

$$\begin{aligned}\widehat{N}(z) = E[z^N] = \frac{\dot{\mathcal{R}}}{\mathcal{T}} = \frac{\dot{\mathcal{R}}}{\dot{\mathcal{R}}\big|_{z=1}} \\ = \frac{\lambda(\lambda+\alpha)(z+\lambda\dot{R}^L_{1,2}) + \mu(\alpha+\lambda(1+z+\lambda\dot{R}^L_{0,2}))}{\lambda(\lambda+\alpha)(1+\lambda T^L_{1,j}) + \mu(\alpha+\lambda(2+\lambda T^L_{0,j}))}\end{aligned} \tag{23}$$

6.3 Deriving $\widetilde{T}(s)$

For the M/M/1/setup system, we were able to derive $\widetilde{T}(s)$ directly from $\widehat{N}(z)$ via the distributional Little's Law, since the M/M/1/setup is a FIFO system. Unfortunately, the M/M/2/setup system is not FIFO, since overtaking can occur. However, we can still apply the distributional Little's Law to the queue of the M/M/2/setup since the queue *is* FIFO. The analysis of $\widehat{N_Q}(z)$ is very similar to that of $\widehat{N}(z)$ and is thus omitted:

$$\widehat{N_Q}(z) = \frac{\lambda(\lambda+\alpha)(1+\lambda\dot{R}^L_{1,2}) + \mu(\alpha+\lambda(1+z+\lambda\dot{R}^L_{0,2}))}{\lambda(\lambda+\alpha)(1+\lambda T^L_{1,j}) + \mu(\alpha+\lambda(2+\lambda T^L_{0,j}))} \tag{24}$$

We now apply the distributional Little's Law to get $\widetilde{T_Q}(s)$ from $\widehat{N_Q}(z)$. Finally, since $T = T_Q + S$, where $S \sim \text{Exp}(\mu)$ is the job size distribution, we have

$$\begin{aligned}\widetilde{T}(s) &= \widetilde{T_Q}(s)\cdot\frac{\mu}{s+\mu} = \widehat{N_Q}\left(1-\frac{s}{\lambda}\right)\cdot\frac{\mu}{s+\mu}\\ &= \frac{\mu\left(\lambda(\lambda+\alpha)(1+\lambda\dot{R}^L_{1,2}) + \mu(\alpha - s + \lambda(2+\lambda\dot{R}^L_{0,2}))\right)}{(s+\mu)\left(\lambda(\lambda+\alpha)(1+\lambda T^L_{1,j}) + \mu(\alpha+\lambda(2+\lambda T^L_{0,j}))\right)}\end{aligned} \tag{25}$$

6.4 Deriving $\widehat{P}(z)$

The derivation of $\widehat{P}(z)$ is similar to that of $\widehat{N}(z)$ in Sect. 6.2, and is thus omitted.

$$\begin{aligned}\widehat{P}(z) &= \frac{\mu(\alpha+\lambda) + \lambda(\lambda+\mu+\alpha)z^{P_{\text{peak}}}}{\mu(\alpha+\lambda) + \lambda(\lambda+\mu+\alpha) + \lambda^2(\mu T^L_{0,j} + (\lambda+\alpha)T^L_{1,j})}\\ &\quad + \frac{\lambda^2(\mu T^L_{0,j} + (\lambda+\alpha)T^L_{1,j})z^{2P_{\text{peak}}}}{\mu(\alpha+\lambda) + \lambda(\lambda+\mu+\alpha) + \lambda^2(\mu T^L_{0,j} + (\lambda+\alpha)T^L_{1,j})}\end{aligned} \tag{26}$$

7 M/M/k/setup

The M/M/k/setup chain shown in Fig. 1 is analyzed similarly to M/M/2/setup. The border states for M/M/k/setup are (i, k), with $0 \le i \le k$. In the M/M/k/setup, the non-repeating portion consists of $O(k^2)$ states. For $k = 1$ and $k = 2$, we did not have to explicitly write reward equations for the non-repeating states; these were implicitly folded into other equations (see, for example, the term in parentheses in Eq. (19)). However, for arbitrarily large k, it is necessary to write reward equations for the states in the non-repeating portion. We use $R^H_{i,j}$ to denote the reward earned until we reach the home state, starting from state (i, j) in the non-repeating portion. The $R^H_{i,j}$ equations will be discussed in Sect. 7.3.

We illustrate our approach for M/M/k/setup by deriving $\widehat{N_Q}(z)$, from which we can obtain $\widetilde{T}(s)$. For a detailed demonstration of this technique for the case of $k = 3$, see [7]. One might think that analyzing the M/M/k/setup will require solving a kth degree equation. This turns out to be false. Analyzing the M/M/k/setup via our renewal-based approach only requires solving equations which are, at worst, quadratic.

We choose $(k-1, k-1)$ to be the renewal state. Using our approach, $\dot{\mathcal{R}}$ can be expressed as

$$\dot{\mathcal{R}} = \frac{1 + (k-1)\mu \dot{R}^H_{k-2,k-2} + \lambda \dot{R}^L_{k-1,k}}{\lambda + (k-1)\mu} \tag{27}$$

We now derive the necessary $p^L_{i\to d}$, $\dot{R}^L_{i,k}$, and $\dot{R}^H_{i,j}$ for computing $\dot{\mathcal{R}}$.

7.1 System of equations for $p^L_{i\to d}$

The system of equations for $p^L_{i\to d}$ is as follows:[4]

$$p^L_{i\to i} = \frac{\lambda (p^L_{i\to i})^2 + i\mu}{\lambda + i\mu + (k-i)\alpha}, \quad (i < k) \tag{28}$$

$$p^L_{i\to d} = \frac{\lambda \left(\sum_{\ell=i}^{d} \left\{(p^L_{i\to \ell})(p^L_{\ell\to d})\right\}\right) + (k-i)\alpha (p^L_{i+1\to d})}{\lambda + i\mu + (k-i)\alpha} \quad (i < d < k) \tag{29}$$

$$p^L_{i\to k} = 1 - \sum_{\ell=i}^{k-1} p^L_{i\to \ell}, \quad (i \le k) \tag{30}$$

The summation in Eq. (29) above denotes the possible intermediate depths ℓ through which we can move from initial depth i to final depth d. The above system of equations involves linear and quadratic equations (including products of two unlike variables), and can be solved *symbolically* to find $p^L_{i\to d}$ in closed form (see Appendix 2).

7.2 Deriving $\dot{R}^L_{i,k}$ for the repeating portion

The system of equations for $\dot{R}^L_{i,k}$ is as follows:

$$\dot{R}^L_{0,k} = \frac{z^k + \lambda \left(z\dot{R}^L_{0,k} + \sum_{\ell=1}^{k} \left\{(p^L_{0\to \ell})(\dot{R}^L_{\ell,k})\right\}\right) + k\alpha \dot{R}^L_{1,k}}{\lambda + k\alpha} \tag{31}$$

$$\dot{R}^L_{i,k} = \frac{z^{k-i} + \lambda \left(z\dot{R}^L_{i,k} + \sum_{\ell=i}^{k} \left\{(p^L_{i\to \ell})(\dot{R}^L_{\ell,k})\right\}\right)}{\lambda + i\mu + (k-i)\alpha} + \frac{(k-i)\alpha \dot{R}^L_{i+1,k}}{\lambda + i\mu + (k-i)\alpha}, \quad (0 < i < k) \tag{32}$$

[4] The definition given for $p^L_{i\to d}$ applies in all cases except when $j = k$ and $d \in \{k-1, k\}$. When $j = k$, we can never end in depth k when moving one step to the left; in this case, we interpret $p^L_{i\to k}$ (or $p^L_{i\to k-1}$) as the probability that we first moved one step left *by transitioning out of a state in depth* k (or $k-1$).

$$\dot{R}^{L}_{k,k} = \frac{1 + \lambda(z\dot{R}^{L}_{k,k} + \dot{R}^{L}_{k,k})}{\lambda + k\mu} \tag{33}$$

In the above, we have used the fact that $\dot{R}^{L}_{i,k+1} = z\dot{R}^{L}_{i,k}$ from Lemma 3. The above system of linear equations can be easily solved to find $\dot{R}^{L}_{i,k}$ in closed form (see Appendix 2).

7.3 Deriving $\dot{R}^{H}_{i,j}$ for the non-repeating portion

The system of equations for $\dot{R}^{H}_{i,j}$ is as follows:

$$\dot{R}^{H}_{0,j} = \frac{z^{j} + \lambda\dot{R}^{H}_{0,j+1} + j\alpha\dot{R}^{H}_{1,j}}{\lambda + j\alpha}, \quad (j < k-1) \tag{34}$$

$$\dot{R}^{H}_{i,j} = \frac{z^{j-i} + \lambda\dot{R}^{H}_{i,j+1} + i\mu\dot{R}^{H}_{i,j-1} + (j-i)\alpha\dot{R}^{H}_{i+1,j}}{\lambda + i\mu + (j-i)\alpha} \quad (0 < i < j < k-1) \tag{35}$$

$$\dot{R}^{H}_{i,i} = \frac{1 + \lambda\dot{R}^{H}_{i,i+1} + i\mu\dot{R}^{H}_{i-1,i-1}}{\lambda + i\mu}, \quad (0 < i < k-1) \tag{36}$$

$$\dot{R}^{H}_{i,k-1} = \frac{z^{k-1-i} + \lambda\left(\dot{R}^{L}_{i,k} + \sum_{\ell=i}^{k}\left\{(p^{L}_{i\to\ell})(\dot{R}^{H}_{\ell,k-1})\right\}\right)}{\lambda + i\mu + (k-1-i)\alpha} + \frac{i\mu\dot{R}^{H}_{i,k-2} + (k-1-i)\alpha\dot{R}^{H}_{i+1,k-1}}{\lambda + i\mu + (k-1-i)\alpha}, \quad (i < k-1) \tag{37}$$

$$\dot{R}^{H}_{k-1,k-1} = 0 \tag{38}$$

Equations (34), (35), and (36), are simply based on the rate transitions in the non-repeating portion of the Markov chain. Equation (37) describe the rewards earned when starting in states in the non-repeating portion of the chain that can transition to the repeating portion of the chain via the border states. When we have an arrival in one of these states, we transition to the repeating portion of the chain, and after earning some reward, return to the non-repeating portion of the chain. Finally, Eq. (38) guarantees that any transition to state $(k-1, k-1)$ will end the renewal cycle. The above system of linear equations can again be easily solved to find $\dot{R}^{H}_{i,j}$ in closed form (see Appendix 2).

After solving for $p^{L}_{i\to d}$, $\dot{R}^{L}_{i,k}$ and $\dot{R}^{H}_{i,j}$, we can derive $\dot{\mathcal{R}}$, and consequently $\widehat{N_Q}(z)$, via Eq. (27). $\widetilde{T}(s)$ can then be derived by using the fact $\widetilde{T}(s) = \widetilde{T_Q}(s) \cdot \frac{\mu}{s+\mu} = \widehat{N_Q}\left(1 - \frac{s}{\lambda}\right) \cdot \frac{\mu}{s+\mu}$.

We applied the above steps to obtain a closed-form expression for $\widehat{N_Q}(z)$ for the M/M/3/setup. We refer the reader to [7] for full details.

8 M/M/k/setup/delayedoff

The Markov chain for M/M/k/setup/delayedoff is shown in Fig. 2. Our renewal state this time will be $(k, k-1)$; thus, $\dot{\mathcal{R}}$, the reward earned when going from $(k, k-1)$ back to $(k, k-1)$ can be expressed as

$$\dot{\mathcal{R}} = \frac{1 + (k-1)\mu \dot{R}^H_{k,k-2} + \lambda \dot{R}^L_{k,k} + \beta \dot{R}^H_{k-1,k-1}}{\lambda + (k-1)\mu + \beta} \tag{39}$$

The analysis for M/M/k/setup/delayedoff via our renewal-based approach is very similar to that of M/M/k/setup in Sect. 7 above. In fact, since the repeating portion for the two chains is the same, the system of equations for $p^L_{i\to d}$ and $\dot{R}^L_{i,j}$ is identical, but the non-repeating portion for the two chains is different. We now set up the system of equations for solving $\dot{R}^H_{i,j}$.

8.1 Deriving $\dot{R}^H_{i,j}$ for the non-repeating portion

The system of equations for $\dot{R}^H_{i,j}$ is as follows:

$$\dot{R}^H_{0,j} = \frac{z^j + \lambda \dot{R}^H_{0,j+1} + j\alpha \dot{R}^H_{1,j}}{\lambda + j\alpha} \quad (j < k-1) \tag{40}$$

$$\dot{R}^H_{i,0} = \frac{1 + \lambda \dot{R}^H_{i,1} + i\beta \dot{R}^H_{i-1,0}}{\lambda + i\beta} \quad (0 < i \le k) \tag{41}$$

$$\dot{R}^H_{k,j} = \frac{1 + \lambda \dot{R}^H_{k,j+1} + j\mu \dot{R}^H_{k,j-1} + (k-j)\beta \dot{R}^H_{k-1,j}}{\lambda + j\mu + (k-j)\beta} \quad (1 \le j < k-1) \tag{42}$$

$$\dot{R}^H_{i,j} = \frac{z^{j-i} + \lambda \dot{R}^H_{i,j+1} + i\mu \dot{R}^H_{i,j-1} + (j-i)\alpha \dot{R}^H_{i+1,j}}{\lambda + i\mu + (j-i)\alpha} \quad (0 < i < j < k-1) \tag{43}$$

$$\dot{R}^H_{i,i} = \frac{1 + \lambda \dot{R}^H_{i,i+1} + i\mu \dot{R}^H_{i,i-1}}{\lambda + i\mu} \quad (0 < i < k-1) \tag{44}$$

$$\dot{R}^H_{i,j} = \frac{1 + \lambda \dot{R}^H_{i,j+1} + j\mu \dot{R}^H_{i,j-1} + (i-j)\beta \dot{R}^H_{i-1,j}}{\lambda + j\mu + (i-j)\alpha} \quad (0 < j < i < k) \tag{45}$$

$$\dot{R}^H_{i,k-1} = \frac{z^{k-1-i} + \lambda \left(\dot{R}^L_{i,k} + \sum_{\ell=i}^{k} \left\{ (p^L_{i\to\ell})(\dot{R}^H_{\ell,k-1}) \right\} \right)}{\lambda + i\mu + (k-1-i)\alpha} + \frac{i\mu \dot{R}^H_{i,k-2} + (k-1-i)\alpha \dot{R}^H_{i+1,k-1}}{\lambda + i\mu + (k-1-i)\alpha} \quad (i \le k-1) \tag{46}$$

$$\dot{R}^H_{k,k-1} = 0 \tag{47}$$

The above system of linear equations can again be solved to find $\dot{R}^H_{i,j}$ in closed form. This yields $\dot{\mathcal{R}}$, and consequently $\widehat{N_Q}(z)$, via Eq. (39).

9 M/M/k/setup/sleep

The Markov chain for M/M/k/setup/sleep is shown in Fig. 3. The analysis for M/M/k/setup/sleep via our renewal-based approach is again similar to that of M/M/k/setup in Sect. 7. The only difference is in the setup transition rate (downward transition arrows in the Markov chain): For the M/M/k/setup, the setup rate in state (i, j) is $\alpha \cdot \min\{j - i, k - i\}$. For the M/M/k/setup/sleep, the setup rate in state (i, j) is more complicated. When $i \geq s$, the setup rate is still $\alpha \cdot \min\{j - i, k - i\}$. However, if $i < s$, the setup rate is $\omega \cdot (j - i)$ if $j \leq s$ and $\omega \cdot (s - i) + \alpha \cdot \min\{j - s, k - s\}$ if $j > s$. This can be explained based on the M/M/k/setup/sleep model description in Sect. 3.3 and the Markov chain in Fig. 3. Based on the above setup rates, we can easily modify the M/M/k/setup sets of equations for $p^L_{i \to d}$, $\dot{R}^L_{i,k}$ and $\dot{R}^H_{i,j}$ from Sects. 7.1, 7.2 and 7.3 respectively, to represent the M/M/k/setup/sleep system of equations. The equation for $\dot{\mathcal{R}}$ will change accordingly.

10 Generalizing our renewal-based approach

Our renewal-based approach can be applied to a very broad class of Markov chains beyond the M/M/k/setup: Our approach can reduce the analysis of any 2-dimensional, irreducible Markov chain which is repeating and infinite in one dimension to the problem of solving a system of polynomial equations. Further, if in the repeating portion all horizontal transitions are skip-free, and all vertical transitions are unidirectional (as shown in Fig. 6), the resulting system of equations will be of degree at most two, yielding a closed-form solution. In this section, we explain the application of our renewal-based approach to general repeating Markov chains in this skip-free, unidirectional class, and also provide justification for the above claims regarding Fig. 6. We

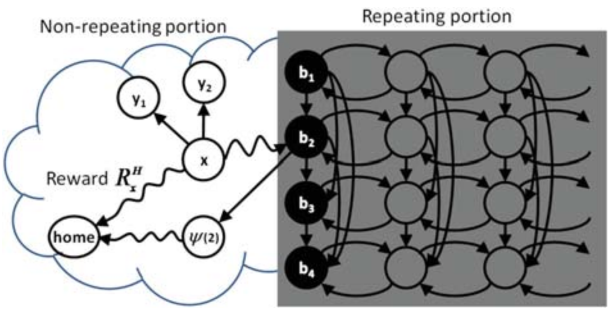

Fig. 6 Figure depicting the class of Markov chains that can be analyzed in *closed-form* via our renewal-based approach. In this class, the horizontal transitions are skip-free and the vertical transitions are unidirectional. The repeating portion is highlighted in *gray* and the border states, b_i, are shaded *black*. Note that y_i are the neighbors of x, and $\psi(2)$ is the exit state in the non-repeating portion accessible from the border state b_2

note that in [24], the authors advance a different solution methodology that applies to a similar, but not identical, class of Markov chains, which they call tree-like QBDs.

10.1 Formal description of Markov chains that are solvable in closed form

Formally, we consider Markov chains partitioned into a one-directionally infinite repeating portion, and a finite non-repeating portion; in principle this partition is not unique. We name the states in the repeating portion so that the set of these states is given by $\{(i, j) : i \in \{1, \ldots, m\}, j \in \{0, 1, 2, \ldots\}\}$, where i gives the "row" (depth) associated with a state and j gives the "column" (level) associated with a state. We will typically use x to denote an arbitrary state in the non-repeating portion.

For tractability, we restrict attention to reward functions taking on the form $R(t) = f(i) + aj$ when the state at time t is (i, j) in the repeating portion of the chain (where f is a real-valued function and a is a real constant), and $R(t) = g(x)$ (where g is a real-valued function) when the state at time t is x in the non-repeating portion of the chain. We also allow for the z-transforms associated with such reward functions; when computing the transform, the rate at which reward is earned in the state (i, j) is $\dot{R}(t) = z^{f(i)+aj}$, while the rate at which reward is earned in state x is $\dot{R}(t) = z^{g(x)}$. Since the case of transforms is more general, the remainder of the analysis in this section will consider the case of computing the z-transform.

The only states in the repeating portion that may be accessible to states in the non-repeating portion in one transition are the *border states* $b_i = (i, 0)$, which are depicted in Fig. 6. Moreover, the border states are the only states in the repeating portion that are permitted to have nonzero transition rates to states in the non-repeating portion of the chain.

The transitions out of states in the repeating portion are as follows:

- Horizontal forward transitions, moving one column to the right, from state (i, j) to state $(i, j + 1)$, with rate $\lambda_i \geq 0$.
- Horizontal backward transitions, moving one column to the left, either (a) from non-border state (i, j), i.e., $j \geq 1$, to state $(i, j - 1)$ with rate $\mu_i \geq 0$ or (b) from border state $b_i = (i, 0)$ to some *exit state* $\psi(i)$ in the non-repeating portion, with rate $\mu_i \geq 0$. We note that ψ may map multiple rows to the same exit state in the non-repeating portion.
- Vertical downward transitions, moving one or several rows lower, from state (i, j) to state (d, j) for each $d \in \{i + 1, \ldots, m\}$ with rate $\alpha_{i \to d} \geq 0$.

We note that the repeating structure of the chain is a consequence of the fact that the transition rates do not depend on the column, j, associated with the state (i, j).

We denote by M the set of *terminal rows*, representing rows in the repeating portion of the Markov chain from which no downward transitions occur (i.e., $M = \{i' : d \in \{i' + 1, \ldots, m\} \Rightarrow \alpha_{i' \to d} = 0\}$). Clearly, M is nonempty, as $m \in M$; we require $\mu_{i'} > \lambda_{i'} \geq 0$ for all $i' \in M$, in order to ensure that the Markov chain is irreducible. For the same reason, we require $\lambda_i > 0$ whenever $\alpha_{d \to i} = 0$ for all $d \in \{1, \ldots, i - 1\}$; in particular, we always require that $\lambda_1 > 0$.

For more general chains, we relax the requirements of skip-free forward horizontal transitions and unidirectional vertical transitions (we still require skip-free backward transitions, although even this restriction can be relaxed by modifying the procedure outlined in this section). That is, we allow the horizontal transitions in the forward direction to skip columns (i.e., transitions from state (i, j) to state $(i, j+\ell)$ for various positive values of ℓ) and the vertical transitions to be bidirectional (i.e., transitions from state (i, j) to state (i, d) even when $1 \le d < i \le m$). However, for these general chains, our renewal-based approach does not guarantee closed-form solutions.

10.2 Analysis of Markov chains that are solvable in closed form

To begin our analysis, after specifying the nonrepeating portion and the repeating portion, we fix a renewal point, or home state, say state 0, within the nonrepeating portion. Our goal is to determine the mean reward, $\dot{\mathcal{R}} = \dot{R}_0^H$, earned in traveling from the home state back to the home state again. After computing this quantity, we can determine the z-transform of the metric being tracked by the reward function, $\widehat{R}(z)$, by normalizing $\dot{\mathcal{R}}$, in accordance with renewal theory:

$$\widehat{R}(z) = \frac{\dot{\mathcal{R}}}{\mathcal{T}} = \frac{\dot{\mathcal{R}}}{\dot{\mathcal{R}}\ |_{z=1}} \tag{48}$$

Since $\dot{R}_0^H$ is determined by rewards earned in other states of the Markov chain, we are interested in the quantities $\dot{R}_y^H$, the mean rewards earned in traveling from each state y to the home state. For each state x in the nonrepeating portion, let Y_x be the set of states accessible from x within a single transition, and let the the transition rate from x to $y \in Y_x$ be $\nu_{x \to y}$. We observe that $\dot{R}_x^H$ is a sum of the mean reward during our residence in x, which is earned at rate $z^{g(x)}$, and a weighted linear combination of the rewards $\dot{R}_y^H$, with $y \in Y_x$, (see Fig. 6). Therefore, for each x in the nonrepeating portion, we have

$$\dot{R}_x^H = \frac{z^{g(x)} + \sum_{y \in Y_x} \{\nu_{x \to y} \dot{R}_y^H\}}{\sum_{y \in Y_x} \{\nu_{x \to y}\}} \tag{49}$$

We call the set of these equations (**Ia**), and note that they are linear in the $\dot{R}_x^H$.

Since the chain is irreducible, at least one state in the nonrepeating portion of the chain transitions directly to a state in the repeating portion of the chain. That is, at least one state in the nonrepeating portion must transition to some border state, b_i (see Fig. 6). Hence, equation set (**Ia**) forms an underdetermined system of linear equations. In order to solve for the $\dot{R}_x^H$ (and in particular, $\dot{\mathcal{R}} = \dot{R}_0^H$), we would like to solve for the mean rewards, $\dot{R}_{b_i}^H$, earned in traveling from each border state b_i to the home state, as at least one of these variables occurs in the system (**Ia**).

The mean reward, $\dot{R}_{b_i}^H$, consists of two parts: (a) the mean reward earned from the time we enter b_i until we leave the repeating portion, which we can think of as the mean reward, $\dot{R}_i^L$, earned until we move "left" of the column $j = 0$ in the repeating portion into the nonrepeating portion; and (b) the mean reward earned from when we first exit the repeating portion until returning home, which is a probabilistic mixture of some $\dot{R}_{\psi(i)}^H$ for exit states $\psi(i)$.

In order to formulate equations for $\dot{R}^H_{b_i}$, we introduce the notation $p^L_{i\to d}$. Given that we are currently in state (i, j), $p^L_{i\to d}$ gives the probability that we first move "left" of the column j from row d. In particular, given that we are currently in state $b_i = (i, 0)$, $p^L_{i\to d}$ gives the probability that we first leave the repeating portion from row d. Using this definition, we formulate equations for $\dot{R}^H_{b_i}$:

$$\dot{R}^H_{b_i} = \dot{R}^L_i + \sum_{d=i}^{m}\{(p^L_{i\to d})\dot{R}^H_{\psi(d)}\} \quad (1 \le i \le m) \tag{50}$$

$$\dot{R}^H_{b_{i'}} = \dot{R}^L_{i'} + \dot{R}^H_{\psi(i')} \quad (i' \in M) \tag{51}$$

We call the set of these equations (**Ib**), and again note the linearity of these equations in the variables $\dot{R}^H_x$ and $\dot{R}^H_{b_i}$. We note that the set of equations (51) are simplifications of Equations (50) in the particular cases where i is a terminal row.

Let (**I**) be the (finite) system of linear equations formed by taking the union of Equation sets (**Ia**) and (**Ib**). Provided that for each $i \in \{1, \ldots, m\}$, $\dot{R}^L_i$ is known in closed form and for each i, d pair satisfying $1 \le i \le d \le m$, $p^L_{i\to d}$ is known in closed form, the system of linear equations (**I**) can be solved symbolically in closed form. Hence, it remains to determine the $\dot{R}^L_i$ and the $p^L_{i\to d}$. The fact that the chain has a repeating structure allows us to express the $\dot{R}^L_i$ using a "recursion lemma" (similar to Lemmas 2 and 3). In particular, we observe that the mean reward to move one step to the left starting from (i, j) is $z^{aj}\dot{R}^L_i$. Recalling that R^L_i is the mean reward earned in moving one step left starting from state $b_i = (i, 0)$, we have

$$\dot{R}^L_i = \frac{z^{f(i)} + \lambda_i\left(z^a\dot{R}^L_i + \sum_{d=1}^{m}\{(p^L_{i\to d})\dot{R}^L_d\}\right) + \sum_{d=i+1}^{m}\{(\alpha_{i\to d})\dot{R}^L_d\}}{\lambda_i + \mu_i + \sum_{d=i+1}^{m}\{\alpha_{i\to d}\}} \quad (1 \le i \le m) \tag{52}$$

$$\dot{R}^L_{i'} = \frac{z^{f(i')} + \lambda_{i'}(z^a\dot{R}^L_{i'} + \dot{R}^L_{i'})}{\lambda_{i'} + \mu_{i'}} \quad (i' \in M) \tag{53}$$

The equations given by (53) are simplifications of particular cases of the equations given by (52). As each equation given by (53) is an equation in only one variable, $\dot{R}^L_{i'}$, these can easily be solved, yielding

$$\dot{R}^L_{i'} = \frac{z^{f(i')}}{\mu_{i'} - \lambda_{i'}z^a} \quad (i' \in M) \tag{54}$$

Equations (52) form a (finite) system of linear equations in the variables $R^L_{i'}$. We call this linear system (**II**).

We note that the linear system (**II**) does not depend on (**I**), and so (**II**) can be solved independently of (**I**), but not vice-versa. However, as with the system of equations (**I**), solving (**II**) requires the determination of the $p^L_{i\to d}$ in closed form. We proceed to formulate yet another system of equations, (**III**), for the $p^L_{i\to d}$, again making use of a recursion lemma (similar to Lemma 5) exploiting the repeating structure of the Markov chain:

$$p^L_{i\to i} = \frac{\lambda_i (p^L_{i\to i})^2 + \mu_i}{\lambda_i + \mu_i + \sum_{\ell=i+1}^{m}\{\alpha_{i\to\ell}\}} \quad (1 \le i \le m) \tag{55}$$

$$p^L_{i\to d} = \frac{\lambda_i \left(\sum_{\ell=i}^{m}\{(p^L_{i\to\ell})(p^L_{\ell\to d})\}\right) + \sum_{\ell=i+1}^{m}\{(\alpha_{i\to\ell})(p^L_{\ell\to d})\}}{\lambda_i + \mu_i + \sum_{\ell=i+1}^{m}\{\alpha_{i\to\ell}\}} \quad (1 \le i < d \le m) \tag{56}$$

$$p^L_{i'\to i'} = 1 \quad (\forall i' \in M) \tag{57}$$

Unlike the system of equations (**I**) and (**II**), the system of equations (**III**) is nonlinear, as when moving to the left, starting in row d and ending in row i, we may transition through various intermediate rows. Thus, $p^L_{i\to d}$ may involve several other probability terms. However, the equations in this system will be of degree at most two. This is because skip-free horizontal transitions guarantee that the probability $p^L_{i\to d}$ can be expressed as a linear sum of products of only two intermediate terms of the form $(p^L_{i\to\ell}) \cdot (p^L_{\ell\to d})$, where ℓ represents the intermediate rows that we can transition to when going from i to d (as in Sect. 7.1). Further, the unidirectional vertical transitions guarantee that $i \le \ell \le d$, which ensures that the intermediate probability terms do not lead to higher-order dependencies between one other. Moreover, solving (**III**) does not depend on the solution to (**I**) or (**II**). Thus, the probabilities, $p^L_{i\to d}$ can be solved for in closed-form by solving quadratic equations (including products of two unlike terms) in a particular "bottom-up" order as explained in Appendix 2.

Once we have solved (**III**) symbolically, we substitute the symbolic values of the $p^L_{i\to d}$ into the linear system (**II**), and solve that system symbolically for the $\dot{R}^L_i$. We next substitute the symbolic values of the $\dot{R}^L_i$ together with the symbolic values of the $p^L_{i\to d}$ into linear system (**I**), which we can finally solve for the $\dot{R}^H_x$, and in particular $\dot{\mathcal{R}} = \dot{R}^H_0$. The final step is the straight-forward computation of the z-transform $\hat{R}(z)$ from Equation (48), which completes the analysis.

For more general chains (where horizontal transitions can skip columns and vertical transitions can be bidirectional), we can set up the system of equations similarly to Eqs. (48)–(57). The equations for these more general chains may include additional terms for the new horizontal transitions in the equations for $\dot{R}^L_i$ and $p^L_{i\to d}$. Moreover, the quantities $p^L_{i\to d}$ now exist even for cases where $0 \le d < i \le m$. These additional complications can result in systems of degree greater than 2 without closed-form solutions.

11 Applications

In this section, we use our analytic results to evaluate the performance of M/M/k, M/M/k/setup, M/M/k/setup/delayedoff and M/M/k/setup/sleep. In particular, we will be interested in the mean response time, $E[T]$, and the mean power consumption, $E[P]$, under these policies. Throughout, we assume a load of $\rho = \frac{\lambda}{k\mu} = 0.3$ (or 30 % load), setup times of $\frac{1}{\alpha} = 100$ s (when the server is off) and $\frac{1}{\omega} = 25$ s (when the server is sleeping), and power consumption values of $P_{\text{peak}} = 200$ W, $P_{\text{idle}} = 140$ W, and $P_{\text{sleep}} = 14$ W. These parameter values are based on empirical measurements from prior work [4,11]. We consider job sizes with mean $E[S] = 1$ s (typical web workloads [6]), $E[S] = 10$ s (database queries or secured transactions), and $E[S] = 100$ s (file download or upload), and system sizes ranging from $k = 5$ to $k = 100$ servers.

The M/M/k policy keeps k servers always on. Servers that are not busy serving jobs are left idle. The M/M/k/setup policy (see Sect. 3.1) immediately turns off idle servers to save power. However, restarting an off server requires a setup time of $\frac{1}{\alpha} = 100$ s. The M/M/k/setup/delayedoff policy (see Sect. 3.2) is the same as the M/M/k/setup policy, except that idle servers wait for an exponentially distributed amount of time with mean $t_{\text{wait}} = \frac{1}{\beta}$ before turning off. The performance of this policy depends on the choice of the t_{wait} parameter. Finally, the M/M/k/setup/sleep policy (see Sect. 3.3) is the same as the M/M/k/setup policy, except that s of the k servers go to sleep as opposed to turning off, when idle. A sleeping server has a small setup time of $\frac{1}{\omega} = 25$ s. The performance of this policy depends on the choice of the s parameter. Before comparing the above four policies, we first discuss how we choose the parameter value of t_{wait} for M/M/k/setup/delayedoff and s for M/M/k/setup/sleep.

11.1 Choosing optimal parameter values

The tradeoff between $E[P]$ and $E[T]$ for M/M/k/setup/delayedoff is shown in Fig. 7a, b. Each plotted point represents an $(E[T], E[P])$ pair associated with a specific value of t_{wait}. Intuitively, as t_{wait} increases, $E[T]$ decreases since we avoid setup times. Moreover, before some threshold t_{wait}, $E[P]$ decreases as t_{wait} increases, because we avoid consuming power at peak rate by repeatedly putting servers in setup. However, beyond this threshold t_{wait}, $E[P]$ starts increasing on account of idle servers. Thus, as t_{wait} increases, we get the plots in Fig. 7a, b, from right to left. We choose the t_{wait} value that optimizes (i.e., maximizes) the popular *Performance-Per-Watt* metric [8,11], given by $PPW = (E[T] \cdot E[P])^{-1}$. These optimal values are shown in Fig. 7a, b. We find that the optimal t_{wait} value decreases with an increase in $E[S]$.

Figure 7c, d illustrate the tradeoff between $E[P]$ and $E[T]$ under M/M/k/setup/sleep for different values of s. Intuitively, as s increases, $E[T]$ decreases since we benefit from faster setup times afforded by sleeping servers. As s increases, $E[P]$ first decreases since we avoid the severe power penalty of longer setup times. However, beyond a certain s, $E[P]$ increases on account of the sleeping servers. Thus, as s increases, we get the plots in Fig. 7c, d, from right to left. Note that $E[P]$ monotonically decreases for the case of $E[S] = 1$s in Fig. 7c. This is because $\frac{1}{\alpha} \gg E[S]$, and thus, the decrease in power consumption by avoiding power penalties of longer setup times outweighs the increase in power consumption because of P_{sleep}. We choose the s value that optimizes the PPW metric, as indicated in Fig. 7c, d. We find that the optimal s value decreases with an increase in $E[S]$.

11.2 Comparison of all policies

Figure 8 shows our results for $E[T]$ and $E[P]$ as a function of k for the case of $E[S] = 1$ s. Comparing M/M/k (squares) and M/M/k/setup (circles), we see that

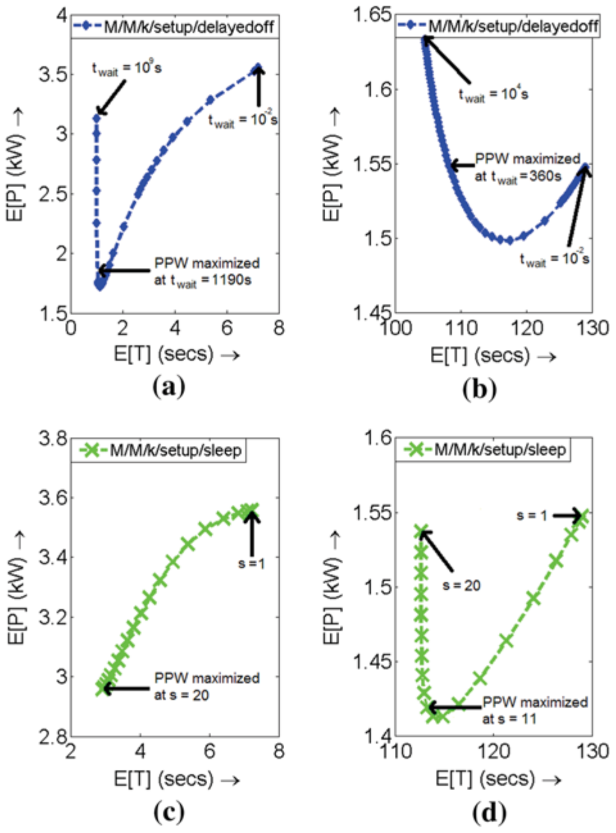

Fig. 7 $E[P]$ versus $E[T]$ for various values of t_{wait} and s. **a** $E[S] = 1$ s, **b** $E[S] = 100$ s, **c** $E[S] = 1$ s, **d** $E[S] = 100$ s

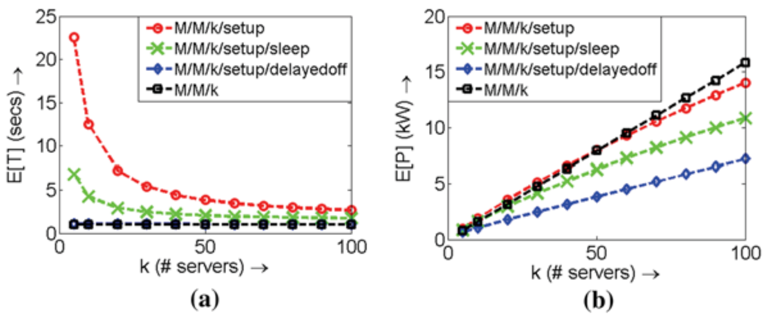

Fig. 8 Results when mean job size $E[S] = 1$. **a** $E[T]$ versus k, **b** $E[P]$ versus k

M/M/k/setup has a much higher $E[T]$, and only a slightly lower $E[P]$. In fact, when k is low, $E[P]$ for M/M/k/setup is *higher* than that of M/M/k. This is because of the power penalty involved in the setup cost. Thus, M/M/k/setup is not a good policy for small job sizes. The M/M/k/setup/sleep (crosses) has lower $E[T]$ *and* lower

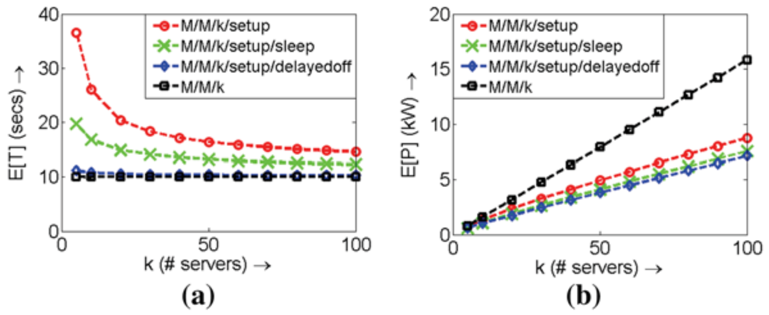

Fig. 9 Results when mean job size $E[S] = 10$. **a** $E[T]$ versus k, **b** $E[P]$ versus k

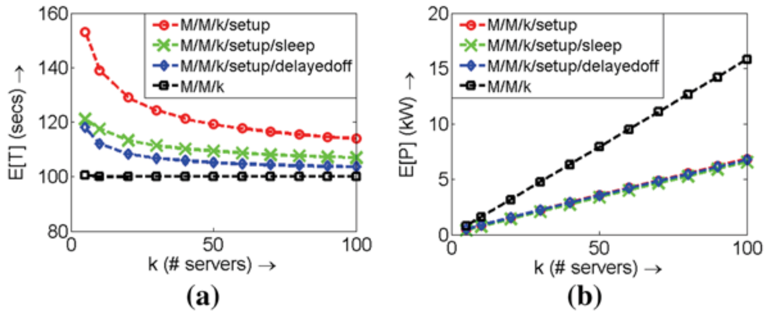

Fig. 10 Results when mean job size $E[S] = 100$. **a** $E[T]$ versus k, **b** $E[P]$ versus k

$E[P]$ than the M/M/k/setup. Thus, using sleep modes improves the M/M/k/setup policy. Finally, we see that M/M/k/setup/delayedoff (diamonds) has $E[T]$ virtually as low as that of M/M/k, and has the lowest power consumption among all other policies. Thus, M/M/k/setup/delayedoff is superior to all the other policies for small job sizes. The reason for lower $E[P]$ under M/M/k/setup/delayedoff is because of t_{wait} which avoids unnecessary setups (and the associated power penalties).

Figures 9 and 10 show our results for the case of $E[S] = 10$ s and $E[S] = 100$ s respectively.

The $E[T]$ results for these job sizes are qualitatively similar to the results for $E[S] = 1$ s. The percentage difference between the $E[T]$ under different policies goes down as $E[S]$ goes up. This is because the setup time is not changing as $E[S]$ goes up, and thus, the queueing delay caused by setup times is not as severe for large $E[S]$. Note that the $E[T]$ under M/M/k/setup/delayedoff actually goes up as $E[S]$ goes up. This is a side effect of the optimal t_{wait} setting which trades off lower $E[P]$ at the expense of a slightly higher $E[T]$ for bigger job sizes.

The $E[P]$ results for these job sizes indicate that $E[P]$ under M/M/k/setup and M/M/k/setup/sleep decreases with an increase in job size, and approaches the $E[P]$ of M/M/k/setup/delayedoff. This is because an increase in $E[S]$ necessitates an increase in the inter-arrival time, given fixed load, ρ. Thus, servers now spend more time in the off or sleep states, and consequently, consume less power. In fact, the M/M/k/set-

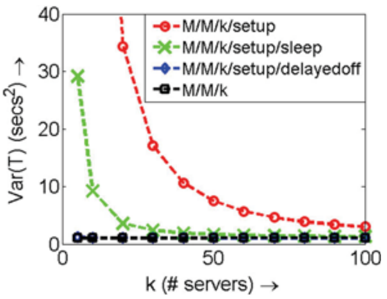

Fig. 11 $Var(T)$ versus k when mean job size $E[S] = 1$ s

up/sleep has *lower* $E[P]$ as compared to M/M/k/setup/delayedoff for the case of $E[S] = 100$ s. We take a closer look at these two policies in Sect. 11.3. Note that under M/M/k, $E[P] = k \cdot \rho \cdot P_{\text{peak}} + k \cdot (1 - \rho) \cdot P_{\text{idle}}$, which is linear in k and independent of $E[S]$.

The $E[T]$ results for these job sizes are qualitatively similar to the results for $E[S] = 1$ s. The percentage difference between the $E[T]$ under different policies goes down as $E[S]$ goes up. This is because the setup time is not changing as $E[S]$ goes up, and thus, the queueing delay caused by setup times is not as severe for large $E[S]$. Note that the $E[T]$ under M/M/k/setup/delayedoff actually goes up as $E[S]$ goes up. This is a side effect of the optimal t_{wait} setting which trades off lower $E[P]$ at the expense of a slightly higher $E[T]$ for bigger job sizes. As mentioned in Sect. 7, our renewal-based approach also provides closed-form solutions for higher moments of response time and power. Figure 11 shows our results for $Var(T)$, the variability in response time, for the case of $E[S] = 1$s. We see that $Var(T)$ follows the same trends as $E[T]$ in Fig. 8a. Note that $Var(T)$ is close to 1 for M/M/k and M/M/k/setup/delayedoff. Also, $Var(T)$ converges to 1 for all policies for high k. This is because $Var(T)$ converges to $Var(S)$ (no queueing delay) in these cases, and since S is exponentially distributed with mean $E[S] = 1$ s, we get $Var(T) \to Var(S) = 1\text{s}^2$.

All the results above assumed exponential setup times and exponential delay times. However, in real-world scenarios, these times would be deterministic. We use simulations to find $E[T]$ and $E[P]$ under deterministic setup times for all the above cases. We find that the relative ordering of the policies and the trends in $E[T]$ and $E[P]$ do not change significantly, despite the fact that all values become slightly higher due to the setup rates no longer being additive.

11.3 A closer look at M/M/k/setup/delayedoff versus M/M/k/setup/sleep

Figure 12 shows the tradeoff between $E[P]$ and $E[T]$ for M/M/k/setup/delayedoff and M/M/k/setup/sleep for $E[S] = 100$ s. These plots are identical to Fig. 7b, d. We see that no policy dominates the other. If we are more concerned about reducing $E[P]$, M/M/k/setup/sleep is the better choice. However, if we are more concerned about reducing $E[T]$, M/M/k/setup/delayedoff is the better choice. Interestingly, by taking a probabilistic mixture of the two policies, we can find additional policies that are superior to the M/M/k/setup/delayedoff and the M/M/k/setup/sleep. The probabilistic

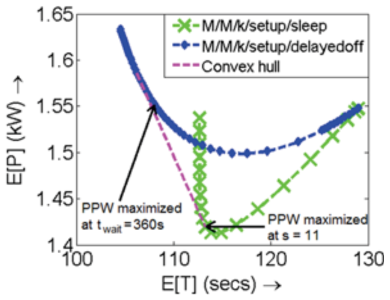

Fig. 12 $E[P]$ versus $E[T]$ for various values of t_{wait} and s for mean job size $E[S] = 100$ s

mixture can be obtained by taking the convex hull of the two policies, as shown by the dashed line in Fig. 12. This suggests the potential for a policy that combines M/M/k/setup/sleep with delayedoff.

12 Conclusion

In this paper we develop a new way of combining renewal reward theory and recursive techniques that allows us to solve the M/M/k/setup class of Markov chains. Our renewal-based approach is very intuitive, easy to apply, and can be used to analyze many important Markov chains that have a repeating structure. Our approach combines renewal reward theory with the development of recursion lemmas for the Markov chain to yield exact, closed form results for metrics of interest such as the transform of time in system and the transform of power consumed by the system. Our approach reduces the solution of the M/M/k/setup chains to solving k quadratic equations and a system of $O(k^2)$ linear equations. On an Intel Core i5-based processor machine we found our renewal-based approach to be almost 5–10 times faster than the iterative matrix-analytic-based methods, when using standard MATLAB implementations of both methods.

While we have only considered the M/M/k/setup, the M/M/k/setup/delayedoff, and the M/M/k/setup/sleep in this paper, we have also been able to use our renewal-based approach for the derivation of exact, closed-form solutions for other important Markov chains with a repeating structure such as (i) M/M/k/stag [10], wherein at most one server can be in setup, (ii) M/M/k/setup/threshold, wherein the servers are turned on and off based on some threshold for number of jobs in queue, (iii) M/M/k/disasters, wherein the system can empty abruptly due to disasters, and (iv) M/E_2/k, where the job size distribution is Erlang-2. We have also been able to apply our renewal-based approach to analyze other Markov chains such as (i) M_t/M/1, where the arrival process is Poisson with a time dependent parameter, (ii) M/H_2/k, where the job size distribution is a 2-phase hyperexponential, and (iii) M^x/M/k, where there is a Poisson batch arrival process. In the above three cases, our approach reduces the analysis to solving a system of polynomial equations with degree >2. In general, our renewal-based approach should be able to reduce the analysis of any 2-dimensional Markov chain (with an affine reward function), which is finite in one dimension and infinite (with repeating structure) in the other, to solving a system of polynomial equations.

While not shown in this paper, it is possible to derive an explicit rate matrix for the M/M/k/setup, which leads to closed-form expressions for the limiting probabilities. While our renewal-based approach does not utilize the rate matrix in any way, we believe that the set of Markov chains that can be solved in closed form via our approach should have an explicit rate matrix. If this hypothesis is true, our renewal-based approach can provide an alternative, intuitive technique for analyzing Markov chains that are typically amenable to matrix-analytic methods. A formal investigation into the connection between matrix analytic methods and our renewal-based approach is left as future work.

Appendix 1: Recursion lemmas

Lemma 1 (Recursion lemma for mean time) *For the M/M/k/setup, the mean time to move one step left from state* (i, j), $T^L_{i,j}$, *is the same for all* $j \geq k$.

Proof For any $j \geq k$, observe that when moving one step left from any state (i, j), we only visit states with level j or greater, until the final transition to level $j - 1$. Hence, $T^L_{i,j}$ depends only on the structure of the "subchain" of the M/M/k/setup consisting of levels $\{j, j + 1, \ldots\}$, including transition rates to level $j - 1$. Now consider the subchain for each $j \geq k$; these subchains are isomorphic, by the fact that the chain is repeating from level k onward. Hence, the time to move one step left is the same regardless of the initial level $j \geq k$. □

Lemma 2 (Recursion lemma for mean reward) *For the M/M/k/setup, the mean reward earned in moving one step left from state* $(i, j+1)$, $R^L_{i,j+1}$, *satisfies* $R^L_{i,j+1} = R^L_{i,j} + T^L_{i,j}$ *for all* $j \geq k$, *where the reward tracks the number of jobs in the system.*

Proof Consider the process of moving one step left from a given state (i, j) where $j \geq k$. At the same time, consider the same process where everything is shifted over one level to the right, so that the initial state is $(i, j + 1)$ At any point in time, the number of jobs seen by the second process is exactly one greater than that seen by the first process. Therefore, the total number of jobs accumulated (total reward) during the second process is $T^L_{i,j}$ greater than that of the first process, since the duration of both processes is $T^L_{i,j}$ by Lemma 1. □

Lemma 3 (Recursion lemma for transform of reward) *For the M/M/k/setup,* $\dot{R}^L_{i,j+1} = z \cdot \dot{R}^L_{i,j}$, *for all* $j \geq k$, *where* $\dot{R}$ *tracks the z-transform of the number of jobs in the system.*

Proof The proof is identical to that of Lemma 2, except that in any moment in time the second process (starting in level $(i, j + 1)$) earns z times as much reward as the first process (starting at (i, j)). □

Lemma 4 (Recursion lemma for transform of power) *For the M/M/k/setup,* $\dot{E}^L_{i,j+1} = \dot{E}^L_{i,j} = T^L_{i,j} \cdot z^{k \cdot P_{\text{peak}}}$, *for all* $j \geq k$.

Proof When $j \geq k$, all k servers are either on or in setup, putting power consumption at $k \cdot P_{\text{peak}}$. So the transform of power usage is $z^{k \cdot P_{\text{peak}}}$, yielding $\dot{E}^L_{i,j} = T^L_{i,j} \cdot z^{k \cdot P_{\text{peak}}}$. It then follows immediately from Lemma 1 that $\dot{E}^L_{i,j+1} = \dot{E}^L_{i,j}$. □

Lemma 5 (Recursion lemma for probability) *For the M/M/k/setup, for each* $0 \leq d \leq k$ *and for each* $0 \leq i \leq k$, $p^L_{i \to d}$ *is the same for all* $j \geq k$.

Proof Recall that $p^L_{i \to d}$ is the probability that, given that we start at depth i, we end at depth d when moving one step to the left, except when $j = k$ and $d \in \{k-1, k\}$; in these cases we interpret $p^L_{i \to k}$ (or $p^L_{i \to k-1}$) as the probabilities that we first moved one step left by *transitioning out of a state* in depth k (or $k-1$).

As with $T^L_{i,j}$, $p^L_{i \to d}$ depends only on the structure of the "subchain" consisting of levels $\{j, j+1, \ldots\}$, including transition rates to level $j-1$. Since for all $j \geq k$ the resulting subchains are isomorphic, $p^L_{i \to d}$ must be the same for all $j \geq k$. □

Appendix 2: Solution of the system of equations for M/M/k/setup

The steps below illustrate how to solve the system of equations for M/M/k/setup. All of the operations in the steps below can be performed *symbolically* to obtain closed-form results.

Solving for $p^L_{i \to d}$

The system of equations for $p^L_{i \to d}$ consists of equation sets (28), (29) and (30). Equation (28) are k quadratic equations, each in one variable: $p^L_{0 \to 0}, p^L_{1 \to 1}, \ldots, p^L_{k-2 \to k-2}, p^L_{k-1 \to k-1}$. Thus, we can solve each equation easily using the quadratic formula. It can be easily shown that among the two roots of each equation, the greater root exceeds 1, and is thus disregarded. The lesser root can be shown to lie in the interval [0, 1), making it the unique solution of interest to the quadratic equation. Note that $p^L_{0 \to 0} = 0$, as expected (we cannot move to the left when we have no servers on).

The set of equations (29) is a collection of $O(k^2)$ equations involving linear terms and products of two unlike variables. However, the structure of this system of equations reduces solving the system to solving a set of linear equations via back substitution. Consider solving this set of equations for the unknown values of $p^L_{i \to d}$ in this order:

$$p^L_{k-1 \to k-1}, p^L_{k-2 \to k-2}, p^L_{k-2 \to k-1}, \ldots, p^L_{0 \to 1}\ p^L_{0 \to 2}, \ldots, p^L_{0 \to k-1}$$

That is, solving from greatest $(k-1)$ to least (0) "original depth," but within each original depth, solving from least to greatest "target depth." Solving in this order, each equation we solve will only have one unknown, as all other variables will already have been solved for in an earlier step (including the $p^L_{i \to i}$ from Eq. (28)), so these variables can be viewed as coefficients and constant terms. Once we have solved Eq. (29), we can easily solve Eq. (30), yielding $p^L_{0 \to k}, p^L_{1 \to k}, \ldots, p^L_{k-1 \to k}$, by taking complements. It follows that all $p^L_{i \to d}$ can be solved in closed forms that are, at worst, linear combinations of radicals (i.e., square roots).

Solving for $\dot{R}^L_{i,k}$

The system of equations for $\dot{R}^L_{i,k}$ consists of Eqs. (31) and (32), and (33). This system is a collection of $(k+1)$ linear equations with $(k+1)$ unknowns. Although we could solve this system using standard linear algebraic techniques, the structure of this system suggests an even simpler approach using back substitution. Solving for each $\dot{R}^L_{i,k}$ only requires knowing the $\dot{R}^L_{i,\ell}$ such that $\ell \in \{i+1, \ldots, k\}$. Equation (33) readily gives us $\dot{R}^L_{k,k}$. Thus, we can now solve for $\dot{R}^L_{k-1,k}$, then $\dot{R}^L_{k-2,k}$, and so on. In this way, each $\dot{R}^L_{i,k}$ is found by solving a linear equation for one unknown variable.

Solving for $\dot{R}^H_{i,j}$

The system of equations for $\dot{R}^H_{i,j}$ consists of Eqs. (34), (35), (36) and (37), and (38). This system is a collection of $O(k^2)$ dependent linear equations with just as many unknowns. Unlike the earlier systems of equations, there is no apparent structure we can exploit, so the system can be solved via standard linear algebraic techniques such as (symbolic) matrix inversion.

References

1. Adan, I., Resing, J.: A class of Markov processes on a semi-infinite strip. Technical Report 99–03, Eindhoven University of Technology, Department of Mathematics and Computing Sciences, (1999)
2. Adan, I., van der Wal, J.: Combining make to order and make to stock. OR Spektrum **20**, 73–81 (1998)
3. Artalejo, J.R., Economou, A., Lopez-Herrero, M.J.: Analysis of a multiserver queue with setup times. Queueing Syst. Theory Appl. **51**(1–2), 53–76 (2005)
4. Barroso, L.A., Hölzle, U.: The case for energy-proportional computing. IEEE Comput. **40**(12), 33–37 (2007)
5. Castellanos, M., Casati, F., Shan, M.-C., Dayal, U.: iBOM: A platform for intelligent business operation management. Proceedings of the 21st International Conference on Data Engineering. ICDE '05, pp. 1084–1095. Tokyo, Japan (2005)
6. DeCandia, G., Hastorun, D., Jampani, M., Kakulapati, G., Lakshman, A., Pilchin, A., Sivasubramanian, S., Vosshall, P., Vogels, W.: Dynamo: Amazon's highly available key-value store. Proceedings of twenty-first ACM SIGOPS Symposium on Operating Systems Principles. SOSP '07, pp. 205–220. Stevenson, WA (2007)
7. Gandhi, A., Doroudi, S., Harchol-Balter, M., Scheller-Wolf, A.: Exact analysis of the M/M/k/setup class of Markov chains via recursive renewal reward. Technical Report CMU-CS-13-105, Carnegie Mellon University, (2013)
8. Gandhi, A., Gupta, V., Harchol-Balter, M., Kozuch, M.: Optimality analysis of energy-performance trade-off for server farm management. Perform. Eval. **67**, 1155–1171 (2010)
9. Gandhi, A., Harchol-Balter, M.: How data center size impacts the effectiveness of dynamic power management. 49th Annual Allerton Conference on Communication, Control, and Computing, (2011)
10. Gandhi, A., Harchol-Balter, M., Adan, I.: Server farms with setup costs. Perform. Eval. **67**, 1123–1138 (2010)
11. Gandhi, A., Harchol-Balter, M., Kozuch, M.: Are sleep states effective in data centers? 3rd IEEE International Green Computing Conference, (2012)
12. Horvath, T., Skadron, K.: Multi-mode energy management for multi-tier server clusters. Proceedings of the 17th International Conference on Parallel Architectures and Compilation Techniques. PACT '08, pp. 270–279. Canada, Toronto (2008)
13. Keilson, J., Servi, L.: A distributional form of little's law. Oper. Res. Lett. **7**(5), 223–227 (1988)

14. Kim, J., Rosing, T.S.: Power-aware resource management techniques for low-power embedded systems. In: Son, S.H., Lee, I., Leung, J.Y.-T. (eds.) Handbook of Real-Time and Embedded Systems. Taylor-Francis Group LLC, Boca Raton (2006)
15. Kleinrock, L.: Queueing Systems, Volume I: Theory. Wiley, New York (1975)
16. Krioukov, A., Mohan, P., Alspaugh, S., Keys, L., Culler, D., Katz, R.: NapSAC: design and implementation of a power-proportional web cluster. Proceedings of the First ACM SIGCOMM Workshop on Green Networking. Green Networking '10, pp. 15–22. New Delhi, India (2010)
17. Latouche, G., Ramaswami, V.: Introduction to Matrix Analytic Methods in Stochastic Modeling. ASA-SIAM, Philadelphia (1999)
18. Levy, Y., Yechiali, U.: An M/M/s queue with servers' vacations. INFOR **14**, 153–163 (1976)
19. Meisner, D., Gold, B.T., Wenisch, T.F.: PowerNap: eliminating server idle power. In Proceeding of the 14th international conference on Architectural Support for Programming Languages and Operating Systems, ASPLOS '09, pages 205–216, Washington, DC, (2009)
20. Mitrani, I.: Managing performance and power consumption in a server farm. Ann. Oper. Res. **202**(1), 121–134 (2013)
21. Qin, W., Wang, Q.: Modeling and control design for performance management of web servers via an IPV approach. IEEE Trans. Control Syst. Technol. **15**(2), 259–275 (2007)
22. Riska, A., Smirni, E.: M/G/1-type Markov Processes: A Tutorial. Performance Evaluation of Complex Systems: Techniques and Tools, pp. 36–63. Springer, New York (2002)
23. Tian, N., Li, Q.-L., Gao, J.: Conditional stochastic decompositions in the M/M/c queue with server vacations. Stoch. Models **15**(2), 367–377 (1999)
24. Van Houdt, B., van Leeuwaarden, J.: Triangular M/G/1-type and tree-like quasi-birth-death Markov chains. INFORMS J. Comput. **23**(1), 165–171 (2011)
25. Van Leeuwaarden, J., Winands, E.: Quasi-birth-and-death processes with an explicit rate matrix. Stoch. Models **22**(1), 77–98 (2006)
26. Welch, P.: On a generalized $M/G/1$ queueing process in which the first customer of each busy period receives exceptional service. Oper. Res. **12**, 736–752 (1964)
27. Xu, X., Tian, N.: The M/M/c queue with (e, d) setup time. J. Syst. Sci. Complex. **21**, 446–455 (2008)
28. Zhang, Z.G., Tian, N.: Analysis on queueing systems with synchronous vacations of partial servers. Perform. Eval. **52**(4), 269–282 (2003)