Analysis of a QBD process that depends on background QBD processes

Takayuki Osogami

September, 2004
CMU-CS-04-163

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Abstract

We define a class of Markov chains that are called recursive foreground-background quasi-birth-and-death (RFBQBD) processes, and describe approximate (nearly exact) analyses of an RFBQBD process. An RFBQBD process consists of a foreground QBD process whose transitions depend on the level of a background QBD process, where the transitions of the background QBD process may depend on the level of another background QBD process, and this dependency may be repeated recursively. We also evaluate the running time and accuracy of the analyses numerically by applying them to analyze the performance of a particular task assignment policy in a multiserver system.

Keywords: Markov chains, multi-dimensional state space, recursive foreground-background QBD process, recursive dimensionality reduction, approximation, analysis, task assignment, cycle stealing, matrix analytic methods.

1 Introduction

A stochastic process (specifically, a Markov chain) is often difficult to analyze when the process is defined on a multi-dimensionally infinite state space. Such multi-dimensionally infinite state spaces appear, for example, in the performance analysis of various scheduling policies in multiserver systems with multiple classes of jobs, where the behavior of certain classes of jobs has inherent dependencies on the state (e.g. number) of other classes of jobs. (See [2] for examples of such multiserver systems and existing analytic solution methods for Markov chains on multi-dimensionally infinite state spaces.) Unfortunately, each existing analytic solution method is limited in the class of Markov chains to which it can be applied, either due to an essential restriction or due to computational complexity. Therefore, it is important to broaden the class of Markov chains that can be analyzed by developing a new analytical solution method and to identify which Markov chains can be analyzed by the new method.

A sequence of recent work [14, 15, 31, 32, 33, 34, 40] has developed an analytic solution method, which we refer to as recursive dimensionality reduction (RDR)[1], and applied RDR to the performance analysis of various multiserver systems with certain dependencies between classes of jobs. However, since individual models of computer systems are analyzed rather informally in [14, 15, 31, 32, 33, 34, 40], it is so far unclear *which* Markov chains can be analyzed via RDR and *how*. For example, [33] considers an M/PH/k queue with two priority classes, where high priority jobs have preemptive priority over low priority jobs. Since the behavior of low priority jobs depends on the number of high priority jobs in the system, the performance analysis of low priority jobs involves a two dimensionally infinite (2D-infinite) state space, where each dimension corresponds to the number of each class of jobs in the system. Utilizing the structure that the behavior of high priority jobs can be analyzed independently of low priority jobs, [33] approximates the 2D-infinite Markov chain by a 1D-infinite Markov chain (without truncating the state space), which can be analyzed more efficiently. At the expense of increased computational complexity, this approximation can be made as accurate as desired. When there are $m > 2$ priority classes, the performance analysis of the lowest priority classes involves an m dimensionally infinite (mD-infinite) state space. In [34, 40], the mD-infinite Markov chain is approximated by a 1D-infinite Markov chain, using the approach in [33] recursively.

The first contribution of this paper is the *formalization* and *generalization* of RDR. We first define a class of Markov chains called recursive foreground-background quasi-birth-and-death (RFBQBD) processes. We then derive an analytical (approximate) expression for the stationary probabilities in an RFBQBD process such that the expression can be evaluated efficiently. The analysis of an RFBQBD process constitutes the formalization and generalization of RDR, since the class of RFBQBD processes includes all the Markov chains analyzed in [14, 15, 31, 32, 33, 34, 40]. A "basic" RFBQBD process consists of a foreground quasi-birth-and-death (QBD) process and a background QBD process, where transitions (both structure and rates) of the foreground QBD process depend on the level of the background QBD process (there exists a certain level, d, such that the transitions of the foreground process stay the same while the background process is in levels $\geq d$). More generally, the transitions in the background process may depend on another background process, and this may be repeated recursively.

The formalization of RDR reveals an issue in the computational complexity of RDR. When RDR is applied to an RFBQBD process, the growth rate of the running time can be double

[1] A part of the idea in RDR is also used for example in [36, 37].

exponential in the number of background processes (i.e. the running time can be $\Theta(2^{2^m})$, where m is the number of background processes). In [34, 40], this is not an issue due to a special structure of the priority M/PH/k queue, where the running time is polynomial in m for any fixed number of servers, k.

The second contribution of this paper is approximations of RDR to reduce its computational complexity. We propose two new approximations of RDR: RDR with partial independence assumption (RDR-PI) and RDR with complete independence assumption (RDR-CI). We show that the running time of RDR-PI and RDR-CI is, in the worst case, exponential in the number of background processes (i.e. $O(2^m)$), and that there is a large subclass of RFBQBD processes for which RDR-CI runs in polynomial time (while RDR takes $\Theta(2^{2^m})$ time).

The third contribution of this paper is a *numerical* study of the running time and accuracy of RDR, RDR-PI, and RDR-CI. We apply RDR, RDR-PI, and RDR-CI to the analysis of a particular task assignment policy in a multiserver system. In this example, we will see that RDR becomes computationally prohibitive with more than three background processes, while the running time of RDR-CI is less than a minute for up to 20 background processes. We will also see that RDR-CI has accuracy comparative to that of RDR and RDR-PI.

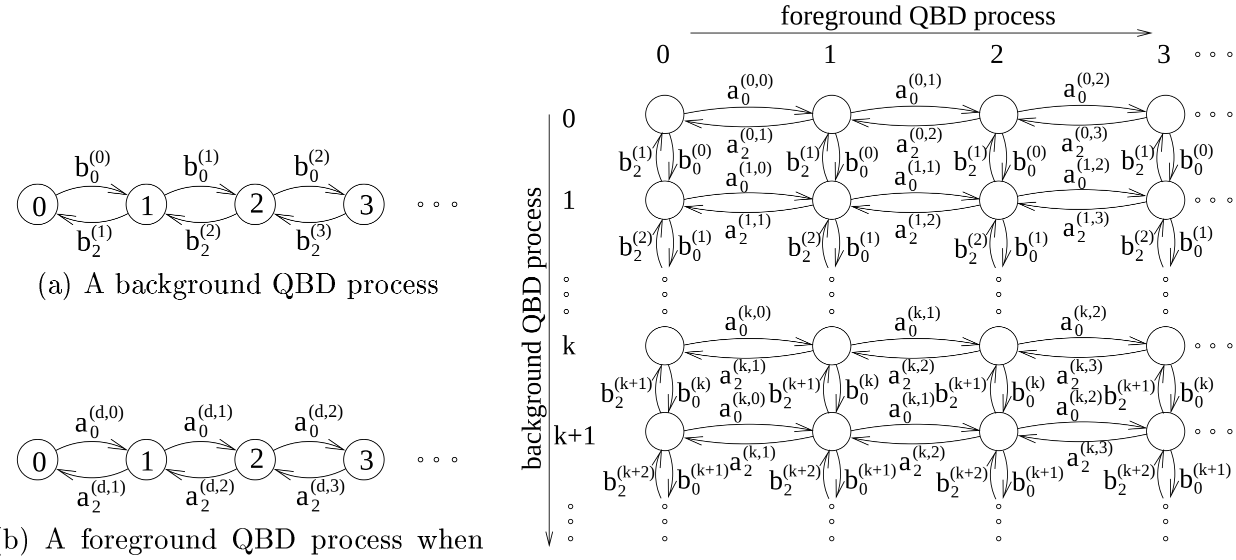
In Section 2, we start by defining the RFBQBD process. We also provide interesting examples of computer system models whose behavior can be captured by RFBQBD processes. In Section 3, we briefly summarize related work on analytical solution methods for Markov chains on multi-dimensionally infinite state spaces. In Section 4, we analyze the RFBQBD process. The analysis of an RFBQBD process constitutes the formalization and generalization of RDR. We also introduce two new approximations in RDR, namely RDR-PI and RDR-CI. In Section 5, we discuss extensions to the RFBQBD process and the analysis of such extended RFBQBD processes. In Section 6, we apply RDR, RDR-PI, and RDR-CI to the analysis of a particular task assignment policy, and numerically evaluate their running time and accuracy.

2 Recursive foreground-background QBD (RFBQBD) process

In this section, we define the recursive foreground-background QBD (RFBQBD) process, and provide examples of RFBQBD processes. Throughout the paper, we denote a matrix by a bold face letter such as $\mathbf{X}$ and its (s,t) element by $(\mathbf{X})_{s,t}$. Also, we use $\vec{x}$ to denote a vector and $(\vec{x})_i$ to denote its i-th element.

An RFBQBD process with parameters $((\mathbf{Q_0^{(0)}}, ..., \mathbf{Q_0^{(k_1)}}), (\mathbf{Q_1^{(0)}}, ..., \mathbf{Q_1^{(k_2)}}), ..., (\mathbf{Q_{m-1}^{(0)}}, ..., \mathbf{Q_{m-1}^{(k_m)}}), \mathbf{Q_m})$ consists of a foreground QBD process and m background QBD processes. Let $\mathbf{Q_0}$ denote the generator matrix for the foreground QBD process, and let $\mathbf{Q_i}$ denote the generator matrix for the i-th background QBD process for $1 \leq i \leq m$. In an RFBQBD process, $\mathbf{Q_i}$ depends on the level of the $(i+1)$-th background QBD for $0 \leq i \leq m-1$. Hence, $\mathbf{Q_i}$ can be expressed as a function of the level, d, of the $(i+1)$-th background QBD process for $0 \leq i \leq m-1$, and it has the following expression:

$$\mathbf{Q_i^{(d)}} = \begin{pmatrix} \mathbf{A_{i,1}^{(d,0)}} & \mathbf{A_{i,0}^{(d,0)}} & & & \\ \mathbf{A_{i,2}^{(d,1)}} & \mathbf{A_{i,1}^{(d,1)}} & \mathbf{A_{i,0}^{(d,1)}} & & \\ & \mathbf{A_{i,2}^{(d,2)}} & \mathbf{A_{i,1}^{(d,2)}} & \mathbf{A_{i,0}^{(d,2)}} & \\ & & \ddots & \ddots & \ddots \end{pmatrix}$$



(a) A background QBD process

(b) A foreground QBD process when the background process is in level d

(c) An RFBQBD process

Figure 1: *An RFBQBD process consisting of a foreground birth-and-death process and a background birth-and-death process.*

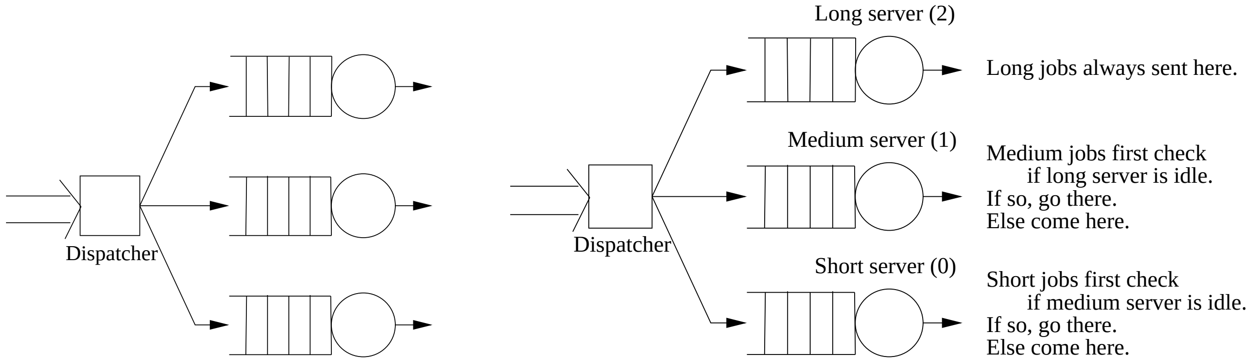
for $0 \le i \le m-1$. In an RFBQBD process, $\mathbf{Q_i^{(d)}}$, for $0 \le i \le m-1$, is assumed to have the following characteristics:

- $\mathbf{Q_i^{(d)}}$ stays the same while the $(i+1)$-th background process is in levels $\ge k_{i+1}$; i.e., $\mathbf{A_{i,j}^{(d,h)}} = \mathbf{A_{i,j}^{(k_{i+1},h)}}$ if $d > k_{i+1}$ for all j and h.
- The state space is independent of the $(i+1)$-th background process; i.e., the size of matrix $\mathbf{A_{i,j}^{(d,h)}}$ is independent of d for all j and h.

To simplify the analysis of an RFBQBD process (description of RDR) in Section 4, we choose the above simple and limited definition of the RFBQBD process. However, as we will see later in this section, the class of RFBQBD process as defined above includes many processes that can capture the behavior of interesting computer systems. In Section 5, we discuss possible extensions to the above definition of the RFBQBD process. An analysis of such an extended RFBQBD process follows immediately from the analysis in Section 4.

Example 0: Foreground-background birth-and-death process The RFBQBD process having a single background QBD process can be modeled as a Markov chain on a 2D-infinite state space. Consider a simpler case where the foreground and background processes are birth-and-death processes. The background process does not depend on other processes, and hence has a single generator matrix, $\mathbf{Q_1}$ (see Figure 1(a)):

$$\mathbf{Q_1} = \begin{pmatrix} -b_0^{(0)} & b_0^{(0)} & & & \\ b_2^{(1)} & -(b_2^{(1)}+b_0^{(1)}) & b_0^{(1)} & & \\ & b_2^{(2)} & -(b_2^{(2)}+b_0^{(2)}) & b_0^{(2)} & \\ & & \ddots & \ddots & \ddots \end{pmatrix}.$$



(a) server farm with dispatcher (b) size-based assignment with cycle stealing

Figure 2: *(a) a server farm with a dispatcher and (b) size-based assignment with cycle stealing when there are three servers.*

On the other hand, the transitions of the foreground process depend on the level of the background process, and hence its generator matrix, $\mathbf{Q_0^{(d)}}$, is a function of the level of the background process, d, (see Figure 1(b)):

$$\mathbf{Q_0^{(d)}} = \begin{pmatrix} -a_0^{(0,d)} & a_0^{(0,d)} & & & \\ a_2^{(1,d)} & -(a_2^{(1,d)} + a_0^{(1,d)}) & a_0^{(1,d)} & & \\ & a_2^{(2,d)} & -(a_2^{(2,d)} + a_0^{(2,d)}) & a_0^{(2,d)} & \\ & & \ddots & \ddots & \ddots \end{pmatrix}.$$

The RFBQBD process assumes that there exists a level, k, of the background process such that $\mathbf{Q_0^{(d)}} = \mathbf{Q_0^{(k)}}$ for all $d > k$. Figure 1(c) shows the RFBQBD process consisting of the foreground and background processes in Figures 1(a)-(b).

Below, we provide illustrative examples of computer system models whose behavior is captured by RFBQBD processes.

Example 1: Size-based task assignment with cycle stealing [31] We consider a task assignment policy in a server farm with a dispatcher (see Figure 2(a)), where jobs at each server are served in first-come-first-serve order. When job sizes have high variability, it has been shown that a size-based assignment policy provides lower mean response time than common assignment policies [13]. The size-based assignment policy in [13] can be improved by size-based assignment with cycle stealing (`SBCS`) [15] (see Figure 2(b)). Under `SBCS`, an arriving long job is always dispatched to the long job server. An arriving medium job first checks to see if the long job server is idle. If so, the medium job is dispatched to the long job server; otherwise, it is dispatched to the medium job server. That is, medium jobs can steal idle cycles of the long job server. Likewise, an arriving short job first checks to see if the medium job server is idle. If so, the short job is dispatched to the medium job server; otherwise, it is dispatched to the short job server. After being dispatched, a job is never reassigned.

More formally, consider m homogeneous servers and m classes of jobs. Class i jobs arrive at a dispatcher according to a Poisson process with rate λ_i and have an exponential service time distribution with rate μ_i ($\mu_i > \mu_j$ if $i < j$) for $0 \le i < m$. Under SBCS, class $m-1$ jobs (largest jobs) are always dispatched to server $m-1$. For $i < m-1$, a class i job first checks to see if the $(i+1)$-th server is idle. If so, the class i job is dispatched to the $(i+1)$-th server; otherwise, it is dispatched to the i-th server. Observe that the arrival process (of class i jobs) at server i depends on whether server $i+1$ is idle or not.

The number of jobs in the system under SBCS can be modeled as an RFBQBD process. Here, the foreground process, B_0, is the number of jobs at server 0 (server for the smallest jobs), and the first background process, B_1, is the number of jobs (both class 0 and class 1) at server 1, which determines whether an arrival of a class 0 job is dispatched to server 0 or server 1 and hence the behavior of B_0. Likewise, the i-th background process, B_i, is the number of jobs (both class $i-1$ and class i) at server i, which determines whether an arrival of a class $i-1$ job is dispatched to server $i-1$ or server i and hence the behavior of B_{i-1}. We will analyze the performance under SBCS via RDR, RDR-PI, and RDR-CI in Section 6.

Example 2: Priority M/M/k queue [9, 25, 29, 40] Consider an M/M/k queue with m priority classes, where class i jobs have preemptive priority over class j jobs for all $1 \le i < j \le m$ (i.e. class 1 has the highest priority). The behavior of class 1 jobs is not affected by the jobs of other classes. However, the behavior of class 2 jobs depends on the number of class 1 jobs in the system. Specifically, when there are n_1 jobs of class 1, $\max\{k-n_1, 0\}$ servers are available for class 2 jobs. Likewise, the behavior of class i jobs depends on the total number of class 1 to class $i-1$ jobs for $i \ge 2$ (when there are n jobs of class 1 to class $i-1$, $\max\{k-n, 0\}$ servers are available for class i jobs).

When $m = 2$, the number of jobs in a priority M/M/k queue can be modeled as an RFBQBD process where the foreground process, A, captures the number of class 2 jobs (low priority jobs) and the background process, B, captures the number of class 1 jobs (high priority jobs). Here, the level d of process B corresponds to the state with d high priority jobs. Note that the number of servers available for low priority jobs (and hence the behavior of process A) is determined by the number of high priority jobs (equivalently, the level of process B).

When $m > 2$, the number of jobs in a priority M/M/k queue can be modeled as an RFBQBD process having $m-1$ background QBD processes with a trivial extension to the above definition of the RFBQBD process. The point of the extension is that the transitions in the i-th background QBD process depend on the level of a QBD process, $\widehat{B}_{i-1}$, that is equivalent to the $(i-1)$-th background QBD process, B_{i-1}. Here, the only difference between $\widehat{B}_{i-1}$ and B_{i-1} is how levels are defined in these two processes. Specifically, the level of B_{i-1} corresponds to the number of class $(m-i+1)$ jobs, while the level of $\widehat{B}_{i-1}$ corresponds to the *total* number of class 1 to class $(m-i+1)$ jobs. We will discuss this extension of the RFBQBD process in Section 5.

Examples 1 and 2 illustrate two types of recursive dependency that RFBQBD processes can have. In these examples, each of the foreground and background processes is quite simple. The next example illustrates more complicated foreground and background processes.

Example 3: Threshold-based policy for reducing switching costs in cycle stealing [31] We consider two processors, each serving its own M/M/1 queue, where one of the processors (the donor) can help the other processor (the beneficiary) while the donor's queue is empty (see Figure 3). Typically, there is a switching time, K_{sw}, required for the donor to start working on

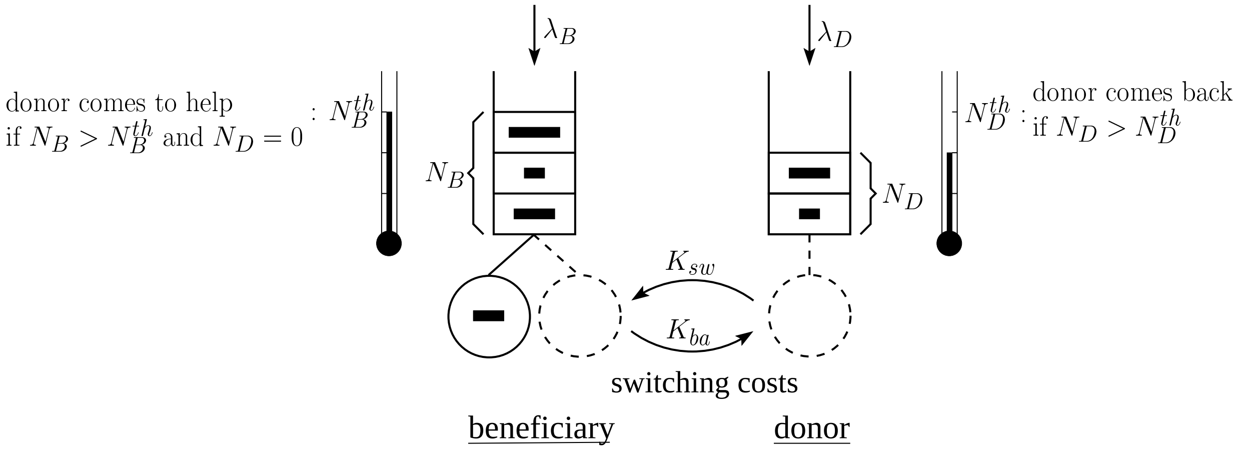


Figure 3: *A threshold-based policy for reducing switching costs in cycle stealing.*

the beneficiary's jobs, as well as a switching back time, K_{ba}. We assume that K_{sw} and K_{ba} have exponential distributions. Due to non-zero switching times, the donor's switching may pay only when the beneficiary's queue length, N_B, is sufficiently long, and the donor's switching back may pay only when the donor's queue length, N_D, is sufficiently long.

Thus, we consider the following threshold-based policy. If $N_B > N_B^{th}$ and $N_D = 0$, the donor starts switching to the beneficiary's queue. After K_{sw} time, the donor is available to work on the beneficiary's jobs. When N_D reaches N_D^{th}, the donor starts switching back to its own queue. After K_{ba} time, the donor resumes working on its own jobs.

The number of jobs in the above system can be modeled as an RFBQBD process. Here, the background process captures the number of the donor's jobs and the state of the donor (whether it is at its own queue, it is switching to the beneficiary's queue, it is at the beneficiary's queue, or it is switching back to its own queue), and the foreground process captures the number of the beneficiary's jobs. Note that the number of the donor's jobs (i.e. the level of the background QBD process) is sufficient to determine whether the donor can work on the beneficiary's jobs (and hence the behavior of the foreground QBD process).

Note that in all the above examples arrival processes can be extended to Markovian arrival processes (MAP) and job size distributions and switching time distributions can be extended to phase type (PH) distributions.

3 Related work

Existing analytic solution methods for Markov chains on multi-dimensionally infinite state spaces can be classified into two approaches: the direct approach and the approach via generating functions. The direct approach solves the equilibrium equations directly (without transforming them). This includes product form methods, compensation methods, power series methods, and matrix analytic methods. On the other hand, the approach via generating functions solves the functional equation for the generating function of the stationary probabilities. This includes uniformization methods and boundary value methods. Below, we briefly summarize how the above methods can or cannot be applied to RFBQBD processes. See [2] for more extensive review on the analytic solution methods for Markov chains on multi-dimensionally infinite state spaces.

Product form methods express the stationary probability as a product of stationary probabilities for respective dimensions (see e.g. [5, 19, 38]). Although product form methods allow a simple analysis of Markov chains on high dimensions, the class of Markov chains that have product form solutions is limited. In particular, the RFBQBD process does not appear to have a product form solution in general.

Compensation methods express the stationary probability as a sum of (an infinite number of) product forms (see e.g. [1, 3, 4]). Compensation methods apply to a large class of Markov chains on a two dimensional grid of the first (positive) quadrant. However, an essential limitation of compensation methods is that transitions to the "north," "north-east," and "east" are prohibited. In particular, compensation methods do not apply to RFBQBD processes in general. It is possible to extend compensation methods to higher dimensions, but the limitation becomes more severe in higher dimensions [39].

Power series methods express the stationary probability as a power series of a certain parameter such as system load (see e.g. [7, 16, 21]). Power series methods can, in theory, be applied to *any* Markov chains. However, the application of power series methods is often limited to simple Markov chains on low dimensional state spaces due to its computational complexity.

Matrix analytic methods are algorithmic approaches for evaluating a broad class of Markov chains, including QBD processes, M/G/1 and G/M/1 type processes, and tree processes (see e.g. [22, 27]). Although the theory of matrix analytic methods has been developed for the broad class of Markov chains, matrix analytic methods are most efficient (in evaluating the solution) and simplest (in implementing the algorithm) when they are applied to QBD processes with a finite (small) number of phases. Therefore, most papers that evaluate Markov chains on 2D-infinite state spaces via matrix analytic methods first truncate the state space so that the resulting process becomes a QBD process with a finite number of phases (see e.g. [18, 17, 23, 24, 28, 35]). Tree processes are a class of Markov chains on multi-dimensionally infinite state spaces that allow an efficient evaluation via matrix analytic methods (see e.g. [6, 22]). However, the RFBQBD process does not appear to be modeled as a tree process. RDR [14, 15, 31, 32, 33, 34, 40] is an approach for approximating a QBD process having a multi-dimensionally infinite number of phases by a QBD process with a finite number of phases (without truncation), so that the approximate QBD process can be evaluated efficiently via matrix analytic methods.

Finally, approaches via generating functions solve the functional equation for the generating function of the stationary probabilities by uniformization (see e.g. [11, 20]) or by reducing the functional equation to a boundary value problem (see e.g. [10, 12]). Although approaches via generating functions apply to a broad class of Markov chains on a two dimensional state space, they do not appear to be applicable to the case of higher dimensions. For example, an analysis of an RFBQBD process having a single background QBD process can, in theory, be reduced to a boundary value problem if the foreground process repeats after a certain level (i.e. there exists k such that $\mathbf{A}_{\mathbf{i}}^{(\mathbf{d},\mathbf{e})} = \mathbf{A}_{\mathbf{i}}^{(\mathbf{d},\mathbf{k})}$ for all $e > k$ for $i = 0, 1, 2$ and $d \geq 0$). However, an RFBQBD process having $m > 1$ background processes does not appear to be solvable via a generating function. Also, approaches via generating functions often experience numerical instability.

4 Analysis of an RFBQBD process

In this section, we analyze the stationary probabilities in an RFBQBD process. In Section 4.1, we start with an analysis of a basic RFBQBD process that consists of a foreground QBD process and a single background QBD process. The analysis of this basic RFBQBD process constitutes the primary part of the formalization of RDR. In Section 4.2, we introduce two new approximations in RDR, namely RDR-PI and RDR-CI. In Section 4.3, we analyze a general RFBQBD process (having an arbitrary number of background QBD processes) by applying the analysis in Section 4.1 (possibly with an approximation in Section 4.2) recursively. Due to the recursive structure of the RFBQBD process, this recursive application of the analysis in Section 4.1 is straightforward, and this completes the formalization of RDR.

4.1 Analysis of RFBQBD process: Single background QBD process

In this section, we analyze the stationary probabilities in a foreground QBD process whose transitions depend on the level of a background QBD process. Note that the background process is simply a QBD process, and its stationary probabilities can be analyzed trivially. In Section 4.1.1, we introduce the notation. In Section 4.1.2, we describe a standard approach of modeling an RFBQBD process having a single background process as a QBD process with an *infinite* number of phases. Although we can derive an analytical expression for the stationary probabilities of the infinite-phase QBD process, there is a difficulty in numerically evaluating such an expression. In Section 4.1.3, we approximate this infinite-phase QBD process by a QBD process with a finite number of phases, so that the stationary probabilities in the approximate QBD process can be evaluated more efficiently. This approximation of the infinite number of phases by a finite number of phases is a key step in RDR.

4.1.1 Notation

Let $\mathbf{Q_B}$ be the generator matrix of the background QBD process:

$$\mathbf{Q_B} = \begin{pmatrix} \mathbf{B_1^{(0)}} & \mathbf{B_0^{(0)}} & & & \\ \mathbf{B_2^{(1)}} & \mathbf{B_1^{(1)}} & \mathbf{B_0^{(1)}} & & \\ & \mathbf{B_2^{(2)}} & \mathbf{B_1^{(2)}} & \mathbf{B_0^{(2)}} & \\ & & \ddots & \ddots & \ddots \end{pmatrix}.$$

Let $S_i^{(B)}$ be the size of matrix $\mathbf{B_1^{(i)}}$ for $i \geq 0$; i.e., $S_i^{(B)}$ is the number of phases in level i of the background QBD process.

Let $\mathbf{Q_A^{(d)}}$ be the generator matrix of the foreground QBD process when the background QBD process is in level d:

$$\mathbf{Q_A^{(d)}} = \begin{pmatrix} \mathbf{A_1^{(d,0)}} & \mathbf{A_0^{(d,0)}} & & & \\ \mathbf{A_2^{(d,1)}} & \mathbf{A_1^{(d,1)}} & \mathbf{A_0^{(d,1)}} & & \\ & \mathbf{A_2^{(d,2)}} & \mathbf{A_1^{(d,2)}} & \mathbf{A_0^{(d,2)}} & \\ & & \ddots & \ddots & \ddots \end{pmatrix}.$$

Recall that transitions in the foreground QBD process stay the same while the background QBD process is in levels $\geq k_B$, i.e.

$$\mathbf{Q_A^{(d)}} = \mathbf{Q_A^{(k_B)}} \quad \text{for all } d > k_B.$$

Also, recall that the state space of the foreground QBD process is fixed; i.e. the size of matrix $\mathbf{A_j^{(d,i)}}$ does not depend on d for all i and j. Let $S_i^{(A)}$ be the size of matrix $\mathbf{A_1^{(d,i)}}$; i.e., $S_i^{(A)}$ is the number of phases in level i of the foreground QBD process.

4.1.2 Modeling as a QBD process with infinite phases

In this section, we describe a standard approach of modeling an RFBQBD process having a single background process as a QBD process with an *infinite* number of phases. Figure 1(c) shows such an infinite-phase QBD process when the foreground and background processes are birth-and-death processes. Below, we consider a general case where the foreground and background processes are QBD processes. We will see that it is hard to evaluate the stationary probabilities in a QBD process with an infinite number of phases.

Let $\mathbf{Q}$ be the generator matrix of the RFBQBD process (i.e. a QBD process with an infinite number of phases):

$$\mathbf{Q} = \begin{pmatrix} \mathbf{A_1^{(0)}} & \mathbf{A_0^{(0)}} & & & \\ \mathbf{A_2^{(1)}} & \mathbf{A_1^{(1)}} & \mathbf{A_0^{(1)}} & & \\ & \mathbf{A_2^{(2)}} & \mathbf{A_1^{(2)}} & \mathbf{A_0^{(2)}} & \\ & & \ddots & \ddots & \ddots \end{pmatrix}.$$

Here, $\mathbf{A_j^{(i)}}$ is a matrix of an infinite number of rows and columns for any i and j. Let $\mathbf{I_n}$ denote an identity matrix of size n, and let $\otimes$ denote a Kronecker product. Then, we can express $\mathbf{A_j^{(i)}}$ as follows:

$$\mathbf{A_0^{(i)}} = \begin{pmatrix} \mathbf{I_{S_0^{(B)}}} \otimes \mathbf{A_0^{(0,i)}} & & & \\ & \mathbf{I_{S_1^{(B)}}} \otimes \mathbf{A_0^{(1,i)}} & & \\ & & \mathbf{I_{S_2^{(B)}}} \otimes \mathbf{A_0^{(2,i)}} & \\ & & & \ddots \end{pmatrix}$$

$$\mathbf{A_2^{(i)}} = \begin{pmatrix} \mathbf{I_{S_0^{(B)}}} \otimes \mathbf{A_2^{(0,i)}} & & & \\ & \mathbf{I_{S_1^{(B)}}} \otimes \mathbf{A_2^{(1,i)}} & & \\ & & \mathbf{I_{S_2^{(B)}}} \otimes \mathbf{A_2^{(2,i)}} & \\ & & & \ddots \end{pmatrix}$$

$$\mathbf{A_1^{(i)}} = \begin{pmatrix} \mathbf{I_{S_0^{(B)}}} \otimes \mathbf{A_1^{(0,i)}} & & & \\ & \mathbf{I_{S_1^{(B)}}} \otimes \mathbf{A_1^{(1,i)}} & & \\ & & \mathbf{I_{S_2^{(B)}}} \otimes \mathbf{A_1^{(2,i)}} & \\ & & & \ddots \end{pmatrix} + \mathbf{Q_B} \otimes \mathbf{I_{S_i^{(A)}}}$$

except that the diagonal elements of $\mathbf{A_1^{(i)}}$ are renormalized so that $\sum_{s=0}^{\infty}(\mathbf{Q})_{s,t} = 0$ for all t.

Using matrix analytic methods [22], the stationary probability of being in level n, $\overrightarrow{\pi_n}$, is then given recursively by

$$\overrightarrow{\pi_n} = \overrightarrow{\pi_{n-1}} \cdot \mathbf{R^{(n)}}, \tag{1}$$

where $\mathbf{R^{(n)}}$ is given recursively by:

$$\mathbf{A_0^{(n-1)}} + \mathbf{R^{(n)}} \cdot \mathbf{A_1^{(n)}} + \mathbf{R^{(n)}} \cdot \mathbf{R^{(n+1)}} \cdot \mathbf{A_2^{(n+1)}} = \mathbf{0}, \tag{2}$$

where $\mathbf{0}$ is a zero matrix of infinite size. Here, a row vector $\overrightarrow{\pi_0}$ is given by a positive solution of

$$\overrightarrow{\pi_0}\left(\mathbf{A_1^{(0)}} + \mathbf{R^{(1)}} \cdot \mathbf{A_2^{(1)}}\right) = \vec{0}, \tag{3}$$

normalized by

$$\overrightarrow{\pi_0} \sum_{n=0}^{\infty} \prod_{m=1}^{n} \mathbf{R^{(m)}} \cdot \vec{1} = 1, \tag{4}$$

where $\vec{0}$ and $\vec{1}$ are vectors with an infinite number of elements of 0 and 1, respectively. When a QBD process has an infinite number of levels, there is an issue of where to truncate [8], since $\mathbf{R^{(n)}}$ needs to be calculated from a certain large enough integer $n = N$ to $n = 1$ recursively via expression (2). However, when the QBD process repeats after level k (i.e., $\mathbf{A_j^{(n)}} = \mathbf{A_j^{(k)}}$ for all $n > k$ for $j = 0, 1, 2$), $\mathbf{R^{(n)}} = \mathbf{R}$ for all $n > k$, and $\mathbf{R}$ is given by the minimal solution to the following matrix quadratic equation:

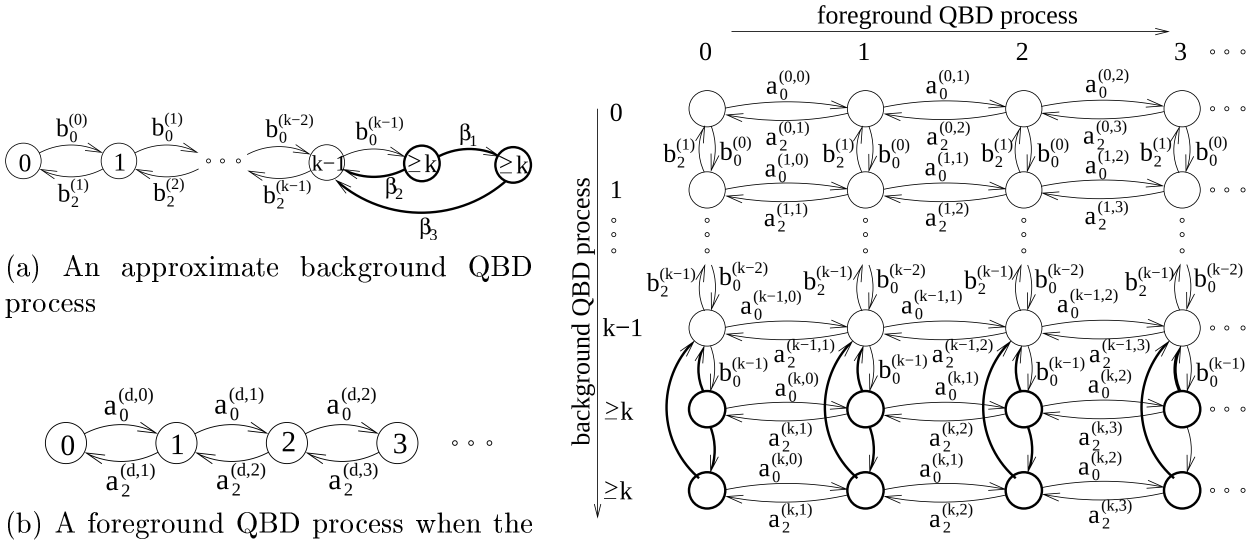
$$\mathbf{A_0^{(k)}} + \mathbf{R} \cdot \mathbf{A_1^{(k)}} + \mathbf{R}^2 \cdot \mathbf{A_2^{(k)}} = \mathbf{0}. \tag{5}$$

In any case, the expressions (1)-(5) are hard to evaluate, since the matrices $\mathbf{A_j^{(i)}}$ have infinite size.

4.1.3 Reducing infinite phases to finite phases

The infinite number of *phases* in the QBD process that models the RFBQBD process in Section 4.1.2 stems from the infinite number of *levels* in the background QBD process. However, recall that the transitions in the foreground QBD process are determined by whether the background QBD process is in level 0, level 1, ..., level $k-1$, or levels $\geq k$. The background QBD process determines the transitions among these levels (0, 1, ..., $k-1$, and $\geq k$), the distribution of the sojourn time in each level, and the dependencies among the transitions and the sojourn time distributions. The key idea is to approximate the background QBD process (process B) by a QBD process with a finite number of levels (process $\widetilde{B}$), so that process $\widetilde{B}$ captures the transitions, the sojourn time distributions, and the dependencies among the transitions and the sojourn time distributions in process B. Once process B is approximated by process $\widetilde{B}$, it is easy to establish a QBD process with a finite number of phases that models the RFBQBD process.

Figure 4 illustrates the idea of our analysis in the case where the foreground and background processes are birth-and-death processes (recall the RFBQBD process in Figure 1). Figure 4(a) shows process $\widetilde{B}$ that approximates the background process (process B) in Figure 1(a). Here, the sojourn time distribution in levels $\geq k$ of process B is approximated by a 2-phase PH distribution (with Coxian representation) in process $\widetilde{B}$. When process B is a QBD process (with more than one phases), we will see that a *collection* of PH distributions is needed to capture the dependency



(a) An approximate background QBD process

(b) A foreground QBD process when the background process is in level d

(c) An approximate RFBQBD process

Figure 4: *An analysis of the RFBQBD process in Figure 1. (a) The background process in Figure 1(a) is approximated by a QBD process with a finite number of levels. (c) The RFBQBD process in Figure 1(c) is approximated by a QBD process with a finite number of phases.*

in the sequence of sojourn times in levels $\geq k$. Figure 4(c) shows a QBD process with a finite number of phases that models the RFBQBD process when the background process is replaced by process $\widetilde{B}$ in Figure 4(a). Below, we formalize and generalize the above idea to a general RFBQBD process where foreground and background processes are QBD processes.

When the background process (process B) is a QBD process, we approximate process B by replacing levels $\geq k$ with a (finite) collection of PH distributions such that the approximate process $\widetilde{B}$ well captures the behavior of process B. Let $E_{s,t}$ be the event, in process B, that the first state that we visit in level $k_B - 1$ is in phase t given that we transitioned from phase s in level $k_B - 1$ to any state in level k_B. We require that process $\widetilde{B}$ has the following key properties:

- The probability of event $E_{s,t}$ in process $\widetilde{B}$ is the same as that in process B.
- The distribution of the sojourn time in levels $\geq k$ given event $E_{s,t}$ in process $\widetilde{B}$ well approximates that in process B (e.g. the first three moments of the two distributions agree).

Observe that if the second property was also exact, then the foreground process with background process B and the foreground process with background process $\widetilde{B}$ would be stochastically equivalent. In particular, B and $\widetilde{B}$ would have the same autocorrelation in the sequence of the sojourn time distributions between changing levels (0, 1, ..., $k-1$, and $\geq k$).

To construct process $\widetilde{B}$, we first analyze the probability of event $E_{s,t}$ and (the moments of) the distribution of the sojourn time in levels $\geq k$ given event $E_{s,t}$ (we denote this distribution as $D_{s,t}$). Let N be the number of phases in level $k_B - 1$ of process B (i.e. $N = S^{(B)}_{k_B-1}$). Then, there are N^2 events $E_{s,t}$.

The probability of event $E_{s,t}$ is relatively easy to analyze. Let $\mathbf{P}$ be a matrix of size $N \times N$ whose (s,t) element is the probability of event $E_{s,t}$. Matrix $\mathbf{P}$ is determined by $\mathbf{B_0^{(k_B-1)}}$ and $\mathbf{G^{(k_B)}}$, where $\mathbf{G^{(k_B)}}$ is the G matrix of process B in level k_B (i.e. $\mathbf{G^{(k_B)}} = (\mathbf{A_0^{(k_B-1)}})^{-1} \cdot \mathbf{R^{(k_B)}} \cdot \mathbf{A_2^{(k_B)}}$). Note that the (s,t) element of $\mathbf{G^{(k_B)}}$, $(\mathbf{G^{(k_B)}})_{s,t}$, is the probability that the first state that we visit in level $k_B - 1$ is in phase t, starting from phase s in level k_B. Also, let $\mathbf{F^{(k_B-1)}}$ be a matrix of size $N \times N$, whose (s,t) element is defined as

$$(\mathbf{F^{(k_B-1)}})_{s,t} = \frac{(\mathbf{B_0^{(k_B-1)}})_{s,t}}{\sum_{i=1}^{N}(\mathbf{B_0^{(k_B-1)}})_{s,i}},$$

where $\frac{0}{0}$ is defined as 0. Note that $(\mathbf{F^{(k_B-1)}})_{s,t}$ is the probability that process B transitions to phase t given that the transition is from phase s in level $k_B - 1$ to any state in level k_B. Matrix $\mathbf{P}$ is then given by

$$\mathbf{P} = \mathbf{F^{(k_B-1)}} \cdot \mathbf{G^{(k_B)}}.$$

Next, we analyze the moments of $D_{s,t}$ (the distribution of the sojourn time in levels $\geq k$ given event $E_{s,t}$). Let $\mathbf{M_h}$ be a matrix of size $N \times N$ whose (s,t) element is the h-th moment of $D_{s,t}$ for $h \geq 1$. Matrix $\mathbf{M_h}$ is related to $\mathbf{G_h^{(k_B)}}$, where $\mathbf{G_h^{(k_B)}}$ is a matrix of size $N \times N$ whose (s,t) element is $(\mathbf{H_h^{(k_B)}})_{s,t} \cdot (\mathbf{G^{(k_B)}})_{s,t}$, where $(\mathbf{H_h^{(k_B)}})_{s,t}$ is the h-th moment of the first passage time from phase s in level k_B to a state in level $k_B - 1$ given that the first state that we visit in level $k_B - 1$ is in phase t. We can calculate $\mathbf{G_h^{(k_B)}}$ by a trivial extension of Neuts' algorithm [26] for any h. Let

$$\begin{aligned} \mathbf{D} &= \mathbf{B_0^{(k_B-1)}} \cdot \mathbf{G^{(k_B)}} \\ \mathbf{E_h} &= \mathbf{B_0^{(k_B-1)}} \cdot \mathbf{G_h^{(k_B)}} \end{aligned}$$

for $h \geq 1$. The (s,t) element of $\mathbf{M_h}$ is then obtained by

$$(\mathbf{M_h})_{s,t} = \frac{(\mathbf{E_h})_{s,t}}{(\mathbf{D})_{s,t}}$$

for $s = 1, ..., N$, $t = 1, ..., N$, and $h \geq 1$.

We are now ready to construct process $\widetilde{B}$. We approximate $D_{s,t}$ by a phase type (PH) distribution, $(\overrightarrow{\tau_{s,t}}, \mathbf{T_{s,t}})$, as defined in [22][2]. For example, we can match the first three moments of $D_{s,t}$ by the approximate PH distribution:

$$(\overrightarrow{\tau_{s,t}}, \mathbf{T_{s,t}}) \quad = \quad \text{three\_moment\_matching}((\mathbf{M_1})_{s,t}, (\mathbf{M_2})_{s,t}, (\mathbf{M_3})_{s,t})$$

for all s and t. Here, three_moment_matching is a function, as defined in [30], that returns a PH distribution whose first three moments match the input three moments. Let

$$\overrightarrow{t_{s,t}} = -\mathbf{T_{s,t}} \cdot \vec{1},$$

[2]A PH distribution with parameter $(\vec{\tau}, \mathbf{T})$ is the distribution of the time until absorption into state 0 in a Markov chain on the states $\{0, 1, ..., n\}$ with initial probability vector $(\tau_0, \vec{\tau})$ and infinitesimal generator $\mathbf{Q} = \begin{pmatrix} 0 & \vec{0} \\ \vec{t} & \mathbf{T} \end{pmatrix}$, where $\mathbf{T} \cdot \vec{1} + \vec{t} = \vec{0}$ and $\tau_0 + \vec{\tau} \cdot \vec{1} = 1$.

where $\vec{1}$ is a column vector with all elements 1 and has size equal to the number of columns in $\mathbf{T_{s,t}}$. For simplicity of exposition, we assume that $\mathbf{T_{s,t}}$ is an $N_{PH} \times N_{PH}$ matrix for all s and t.

Using $\mathbf{P}$ and $(\overrightarrow{\tau_{s,t}}, \mathbf{T_{s,t}})$ defined above, the generator matrix of process $\widetilde{B}$ is defined as

$$\mathbf{Q_{\widetilde{B}}} = \left(\begin{array}{cccc|cccc} \mathbf{B_1^{(0)}} & \mathbf{B_0^{(0)}} & & & & & & \\ \mathbf{B_2^{(1)}} & \ddots & \ddots & & & & & \\ & \ddots & \ddots & \mathbf{B_0^{(k_B-2)}} & & & & \\ & & \mathbf{B_2^{(k_B-1)}} & \mathbf{B_1^{(k_B-1)}} & (\vec{\lambda})_1 \cdot \boldsymbol{\tau_1} & (\vec{\lambda})_2 \cdot \boldsymbol{\tau_2} & \cdots & (\vec{\lambda})_N \cdot \boldsymbol{\tau_N} \\ \hline & & & \mathbf{t_1} & \mathbf{T_1} & & & \\ & & & \mathbf{t_2} & & \mathbf{T_2} & & \\ & & & \vdots & & & \ddots & \\ & & & \mathbf{t_N} & & & & \mathbf{T_N} \end{array}\right),$$

where

$$\begin{aligned} \boldsymbol{\tau_i} &= \begin{pmatrix} & \mathbf{O_{i-1,N}} & \\ (\mathbf{P})_{i,1} \cdot \overrightarrow{\tau_{i,1}} & \cdots & (\mathbf{P})_{i,N} \cdot \overrightarrow{\tau_{i,N}} \\ & \mathbf{O_{N-i,N}} & \end{pmatrix} \\ \mathbf{T_i} &= \begin{pmatrix} \mathbf{T_{i,1}} & & \\ & \ddots & \\ & & \mathbf{T_{i,N}} \end{pmatrix} \\ \mathbf{t_i} &= \begin{pmatrix} \overrightarrow{t_{i,1}} & & \\ & \ddots & \\ & & \overrightarrow{t_{i,N}} \end{pmatrix} \\ \vec{\lambda} &= \mathbf{B_0^{(k_B-1)}} \cdot \vec{1} \end{aligned}$$

for $i = 1, ..., N$, where $\mathbf{O_{s,t}}$ denotes a zero matrix of size $s \times t$, and $\vec{1}$ is a column vector with all elements 1 and has size equal to the number of columns in $\mathbf{B_0^{(k_B-1)}}$.

Once process $\widetilde{B}$ is defined, the QBD process with a finite number of phases that models the RFBQBD process can be easily defined. For any i and j, we can express $\mathbf{A_j^{(i)}}$ as follows:

$$\begin{aligned} \mathbf{A_0^{(i)}} &= \begin{pmatrix} \mathbf{I_{S_0^{(B)}}} \otimes \mathbf{A_0^{(0,i)}} & & & \\ & \ddots & & \\ & & \mathbf{I_{S_{k_B-1}^{(B)}}} \otimes \mathbf{A_0^{(k_B-1,i)}} & \\ & & & \mathbf{I_{N^2 N_{PH}}} \otimes \mathbf{A_0^{(k_B,i)}} \end{pmatrix} \\ \mathbf{A_2^{(i)}} &= \begin{pmatrix} \mathbf{I_{S_0^{(B)}}} \otimes \mathbf{A_2^{(0,i)}} & & & \\ & \ddots & & \\ & & \mathbf{I_{S_{k_B-1}^{(B)}}} \otimes \mathbf{A_2^{(k_B-1,i)}} & \\ & & & \mathbf{I_{N^2 N_{PH}}} \otimes \mathbf{A_2^{(k_B,i)}} \end{pmatrix} \end{aligned}$$

$$\mathbf{A_1^{(i)}} = \begin{pmatrix} \mathbf{I_{S_{H,0}}} \otimes \mathbf{A_1^{(0,i)}} & & & \\ & \ddots & & \\ & & \mathbf{I_{S^{(B)}_{k_B-1}}} \otimes \mathbf{A_1^{(k_B-1,i)}} & \\ & & & \mathbf{I_{N^2 N_{PH}}} \otimes \mathbf{A_1^{(k,i)}} \end{pmatrix} + \mathbf{Q_{\widetilde{B}}} \otimes \mathbf{I_{S_i^{(A)}}},$$

except that the diagonal elements of $\mathbf{A_1^{(i)}}$ are renormalized so that $\sum_{s=0}^{\infty}(\mathbf{Q})_{s,t} = 0$ for all t. Note that expressions (1)-(4) can now be evaluated more efficiently, as the size of matrices $\mathbf{A_j^{(i)}}$ is finite.

4.2 New approximations

The analysis in Section 4.1.3 can still be computationally prohibitive, in particular when it is used recursively as we do in Section 4.3. In this section, we introduce two new approximations in RDR, namely RDR-PI and RDR-CI, which can significantly reduce the computational complexity while keeping a reasonable accuracy. The key idea in our new approximations is that distributions $D_{s,t}$ (the distribution of the sojourn time in levels $\geq k$ given event $E_{s,t}$) can be aggregated without loosing too much information.

RDR-PI (RDR with partial independence assumption) ignores the dependency that the sojourn time in levels $\geq k_B$ has on how it *starts*. Specifically, we assume that $D_{s,t}$ is independent of s. Let $D'_t = D_{s,t}$ for all s for each t, and let $\widetilde{B}'$ denote the process that is the same as process $\widetilde{B}$ except that $D_{s,t}$ is replaced by D'_t for all s and t. Process $\widetilde{B}'$ has two important properties:

- The probability of event $E_{s,t}$ is the same as that in process $\widetilde{B}$.
- We choose D'_t such that the marginal distribution of the sojourn time in levels $\geq k_B$ well approximates that in process $\widetilde{B}$ (e.g. the first three moments of the two distributions agree).

Hence, although $\widetilde{B}'$ and $\widetilde{B}$ have different autocorrelation in the sequence of the sojourn times in levels $\geq k_B$, they have stochastically the same total sojourn time in levels $\geq k_B$ in the long run.

More formally, the generator matrix of process $\widetilde{B}'$, $Q_{\widetilde{B}'}$, is determined as follows. Let $(\overrightarrow{M'_h})_t$ be the h-th moment of D'_t for $h \geq 1$ and for $1 \leq t \leq N$. $(\overrightarrow{M'_h})_t$ is determined so that $\widetilde{B}$ and $\widetilde{B}'$ have the same marginal h-th moment of the sojourn time in levels $\geq k_B$:

$$(\overrightarrow{M'_h})_t = \frac{\sum_{s=1}^{N}(\overrightarrow{\pi_{k_B-1}})_s \cdot (\vec{\lambda})_s \cdot (\mathbf{P})_{s,t} \cdot (\mathbf{M_h})_{s,t}}{\sum_{s=1}^{N}(\overrightarrow{\pi_{k_B-1}})_s \cdot (\vec{\lambda})_s \cdot (\mathbf{P})_{s,t}},$$

where $\overrightarrow{\pi_{k_B-1}}$ denotes the stationary probability that process $\widetilde{B}$ is in level $k_B - 1$, which can be calculated via expressions (1)-(5). We approximate D'_t by a PH distribution, $(\overrightarrow{\tau'_t}, \mathbf{T'_t})$, for example, by matching the first three moments of D'_t by the approximate PH distribution:

$$(\overrightarrow{\tau'_t}, \mathbf{T'_t}) = \text{three\_moment\_matching}((\overrightarrow{M'_1})_t, (\overrightarrow{M'_2})_t, (\overrightarrow{M'_3})_t).$$

Let $\overrightarrow{t_t'} = -\mathbf{T_t'} \cdot \vec{1}$ as before. $\mathbf{Q'_{\widetilde{B}}}$ is then defined by

$$\mathbf{Q'_{\widetilde{B}}} = \left(\begin{array}{cccc|c} \mathbf{B_1^{(0)}} & \mathbf{B_0^{(0)}} & & & \\ \mathbf{B_2^{(1)}} & \ddots & \ddots & & \\ & \ddots & \ddots & \mathbf{B_0^{(k_B-2)}} & \\ & & \mathbf{B_2^{(k_B-1)}} & \mathbf{B_1^{(k_B-1)}} & \boldsymbol{\tau'} \\ \hline & & & \mathbf{t'} & \mathbf{T'} \end{array} \right),$$

where

$$\begin{aligned} \boldsymbol{\tau'} &= \begin{pmatrix} (\vec{\lambda})_1 & & \\ & \ddots & \\ & & (\vec{\lambda})_N \end{pmatrix} \begin{pmatrix} (\mathbf{P})_{1,1} \cdot \overrightarrow{\tau_1'} & \cdots & (\mathbf{P})_{1,N} \cdot \overrightarrow{\tau_N'} \\ \vdots & & \vdots \\ (\mathbf{P})_{N,1} \cdot \overrightarrow{\tau_1'} & \cdots & (\mathbf{P})_{N,N} \cdot \overrightarrow{\tau_N'} \end{pmatrix} \\ \mathbf{T'} &= \begin{pmatrix} \mathbf{T_1'} & & \\ & \ddots & \\ & & \mathbf{T_N'} \end{pmatrix} \\ \mathbf{t'} &= \begin{pmatrix} \overrightarrow{t_1'} & & \\ & \ddots & \\ & & \overrightarrow{t_N'} \end{pmatrix}. \end{aligned}$$

Observe that the number of PH distributions used to approximate the sojourn time distributions in levels $\geq k_B$ is reduced from N^2, in Section 4.1, to N. The next approximation, RDR-CI, uses only one PH distribution.

RDR-CI (RDR with complete independence assumption) ignores not only the dependency that the length of the sojourn time in levels $\geq k_B$ has on how it starts but also the dependency on how it *ends*. Specifically, we assume that $D_{s,t}$ is independent of s and t. Let $D'' = D_{s,t}$ for all s and t, and let $\widetilde{B}''$ denote the process that is the same as process $\widetilde{B}$ except that $D_{s,t}$ is replaced by D'' for all s and t. We choose D'' such that process $\widetilde{B}''$ has the above two important properties that process $\widetilde{B}'$ has. The difference between $\widetilde{B}'$ and $\widetilde{B}''$ lies in the dependencies in the sequence of the sojourn times in levels $\geq k_B$. Observe that the sequence of the sojourn times in levels $\geq k_B$ is i.i.d. in process $\widetilde{B}''$, while it has some dependencies in process $\widetilde{B}'$.

More formally, the generator matrix of process $\widetilde{B}''$, $Q_{\widetilde{B}''}$, is determined as follows. Let M_h'' be the h-th moment of D'' for $h \geq 1$. $(\overrightarrow{M_h''})$ is determined so that $\widetilde{B}$ and $\widetilde{B}''$ have the same marginal h-th moment of the sojourn time in levels $\geq k_B$:

$$M_h'' = \frac{\sum_{t=1}^{N} \sum_{s=1}^{N} (\overrightarrow{\pi_{k_B-1}})_s \cdot (\vec{\lambda})_s \cdot (\mathbf{P})_{s,t} \cdot (\mathbf{M_h'})_{s,t}}{\sum_{t=1}^{N} \sum_{s=1}^{N} (\overrightarrow{\pi_{k_B-1}})_s \cdot (\vec{\lambda})_s \cdot (\mathbf{P})_{s,t}}.$$

We approximate D'' by a PH distribution, $(\overrightarrow{\tau''}, \mathbf{T''})$, for example, by matching the first three moments of D'' by the approximate PH distribution:

$$(\overrightarrow{\tau''}, \mathbf{T''}) = \text{three\_moment\_matching}(M_1'', M_2'', M_3'').$$

Let $\overrightarrow{t''} = -\mathbf{T}'' \cdot \vec{1}$ as before. $\mathbf{Q}''_{\widetilde{\mathbf{B}}}$ is then defined by

$$\mathbf{Q}''_{\widetilde{\mathbf{B}}} = \left(\begin{array}{cccc|c} \mathbf{B_1^{(0)}} & \mathbf{B_0^{(0)}} & & & \\ \mathbf{B_2^{(1)}} & \ddots & \ddots & & \\ & \ddots & \ddots & \mathbf{B_0^{(k_B-2)}} & \\ & & \mathbf{B_2^{(k_B-1)}} & \mathbf{B_1^{(k_B-1)}} & \boldsymbol{\tau''} \\ \hline & & & \overrightarrow{t''} \cdot \overrightarrow{\xi} & \mathbf{T''} \end{array} \right),$$

where

$$\boldsymbol{\tau''} = \mathrm{transpose}(\vec{\lambda}) \cdot \overrightarrow{\tau''},$$

and $\overrightarrow{\xi}$ is a row vector with i-th element:

$$(\overrightarrow{\xi})_i = \frac{\sum_{s=1}^{N} (\overrightarrow{\pi_{k_B-1}})_s \cdot (\vec{\lambda})_s \cdot (\mathbf{P})_{s,i}}{\sum_{s=1}^{N} \sum_{t=1}^{N} (\overrightarrow{\pi_{k_B-1}})_s \cdot (\vec{\lambda})_s \cdot (\mathbf{P})_{s,t}}.$$

4.3 Recursive analysis of RFBQBD process: $m \geq 2$ background QBD processes

Finally, we consider a general RFBQBD process that consists of a foreground QBD process and $m \geq 2$ background QBD processes, where the transitions of the foreground QBD process depend on the level of the first background QBD process, and the transitions of the i-th background QBD process depend on the level of the $(i+1)$-th background QBD process for $i = 1, ..., m-1$. We analyze the stationary probabilities in the foreground and background processes by applying the analysis in Section 4.1 (possibly with an approximation in Section 4.2) recursively. Note that if we took the approach as in Section 4.1.2 to model the RFBQBD process having m background processes as a QBD process, the number of phases within a level in such a QBD process would now grow infinitely in m dimensions. RDR (RDR-PI, and RDR-CI) approximates the mD-infinite phase QBD process by a QBD process with a finite number of phases. In Sections 4.1-4.2, we have analyzed the case of $m = 1$.

The analysis for the case of $m > 1$ follows immediately from the analysis for the case of $m = 1$. We argue, by induction, that all the m background processes and the foreground process (the 0-th background process) can be approximated by QBD processes with a finite number of phases via the approach in Section 4.1.3. Then, the stationary probabilities in the foreground and background processes can be obtained by analyzing the stationary probabilities in the approximate QBD processes. The m-th background process is a QBD process with a finite number of phases by our assumption, which proves the base case. Suppose that the i-th background process is approximated by a QBD process with a finite number of *phases*, B_i. The QBD process B_i typically has an infinite number of *levels*. However, by the analysis in Section 4.1.3 (possibly with an approximation in Section 4.2), B_i can be approximated by a QBD process with a finite number of *levels*, $\widetilde{B}_i$, such that B_i and $\widetilde{B}_i$ have stochastically similar effect on the $(i-1)$-th background process, B_{i-1}. Now, using $\widetilde{B}_i$, process B_{i-1} can be modeled as a QBD process with a finite number of *phases*. This completes our argument.

The running times of RDR, RDR-PI, and RDR-CI differ primarily due to the size of the submatrices, $A_i^{(j)}$, of the generator matrix for the QBD process (i.e. the number of phases in the QBD pro-

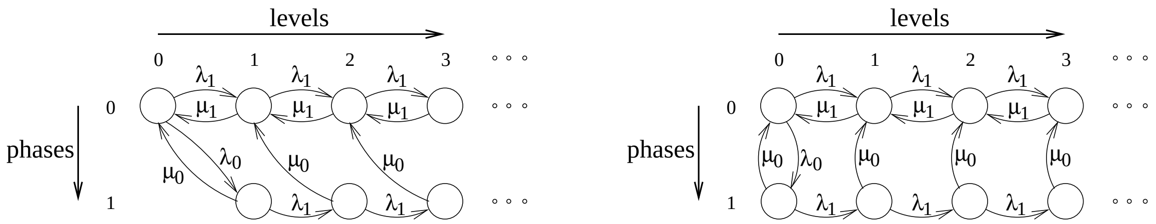


Figure 5: *Two equivalent QBD processes with different definitions of levels.*

cess) that approximates the RFBQBD process. For simplicity, assume that the transitions in the i-th background QBD process stay the same while the $(i+1)$-th background QBD process is in levels k for all i (i.e. $k = k_1 = \cdots = k_m$), and that the foreground and background QBD processes have the same number of phases, P_{QBD}. In RDR, the size of the submatrices, $S_{RDR(m)}$, grows double-exponentially with the number of background processes (i.e. $O(2^{2^m})$); specifically, $S_{RDR(m)}$ can be determined by the following recursive formula: $S_{RDR(m)} = kS_{RDR(m-1)} + (S_{RDR(m-1)})^2 N_{PH}$ for $m > 0$ and $S_{RDR(0)} = P_{QBD}$, where N_{PH} is the number of phases used in a PH distribution that approximates a (conditional) sojourn time in levels $\geq k$ (there are $(S_{RDR(m-1)})^2$ sojourn times). In RDR-PI, the size of the submatrices, $S_{PI(m)}$, grows exponentially with m: $S_{PI(m)} = (k + N_{PH})^m P_{QBD}$. In RDR-CI, the size of the submatrices, $S_{CI(m)}$, grows exponentially with m (but slower than $S_{PI(m)}$) when $k > 1$: $S_{CI(m)} = k^m \left(P_{QBD} + \frac{N}{k-1}\right) - \frac{N_{PH}}{k-1}$, and it grows linearly with m when $k = 1$: $S_{CI(m)} = mN_{PH} + P_{QBD}$.

5 Extensions

The definition of the RFBQBD process in Section 2 can be extended such that it can still be analyzed via RDR, RDR-PI, and RDR-CI with a slight modification. In this section, we discuss two examples of such extensions.

The first extension concerns the way multiple background processes depend on each other. In Section 2, the transitions of the i-th background QBD process, B_i, are assumed to depend on the level of the $(i+1)$-th background QBD process, B_{i+1}. Observe, however, that a QBD process can have different definitions of levels and phases. Figure 5 shows two equivalent QBD processes with different definitions of levels. Observe that the two QBD processes are exactly the same with respect to the state space and the transitions among the states and that the only difference lies in how levels are defined. Therefore, we can relax the assumption of the RFBQBD process such that the transitions of B_i depend on the level of a QBD process, $\widehat{B}_{i+1}$, that is equivalent to B_{i+1}. Here, $\widehat{B}_{i+1}$ and B_{i+1} differ only in how their levels and phases are defined. One might think that the above extension of the RFBQBD process is not essential since the levels in B_{i+1} could be defined in the same way as in $\widehat{B}_{i+1}$ from the beginning. However, there are cases where the above extension is in fact essential.

Recall the priority M/M/k queue in Example 2 in Section 2, where class i jobs have preemptive priority over class j jobs for all $1 \leq i < j \leq m$. The number of jobs in a priority M/M/k queue can be modeled as an extended RFBQBD process. We first try defining the i-th background QBD process, B_i, (and the foreground process, B_0) such that the level of B_i corresponds to the number of class $(m - i)$ jobs for $0 \leq i \leq m - 1$. When there are only two priority classes ($m = 2$), this definition has no problem, i.e., the transitions in B_0 is determined by the level of B_1 (see Example 2 in Section 2). However, when $m > 2$, the transitions in B_i cannot be determined by the level in B_{i+1} (i.e. the number of class $(m-i-1)$ jobs). A solution to this problem is to define a QBD process $\widehat{B}_{i+1}$ which is equivalent to B_{i+1} such that the level of $\widehat{B}_{i+1}$ corresponds to the *total* number of jobs of class 1 to class $(m-i-1)$. Now the level of $\widehat{B}_{i+1}$ determines the transitions in B_i. It is easy to see that there is such an equivalent QBD process $\widehat{B}_{i+1}$, since any event (arrival or job completion) in process B_{i+1} changes the total number of jobs (and hence the level of $\widehat{B}_{i+1}$) by 1. Note that there is also a problem in defining the level of B_i such that it corresponds to the total number of jobs of class 1 to class $(m-i)$. In this case, when the level of B_{i+1} changes, it also changes the level of B_i, which is not allowed in our definition of the RFBQBD process.

The second extension allows a background process to depend on a foreground process. Recall the basic RFBQBD process, process P, in Figure 1(c) and a QBD process with a finite number of phases, process $\widetilde{P}$, in Figure 4(c) that approximates process P. It is intuitively clear that in order to approximate process P by a QBD process with a finite number of phases such as process $\widetilde{P}$, the transitions in the top k rows in process P do not have to have any special structure as long as process P is QBD. A similar observation holds for general RFBQBD processes, and the definition of the RFBQBD process can be extended accordingly. An interesting subclass of such extended RFBQBD processes is the one where transitions in levels $< k$ of the background QBD process depend on the level of the foreground QBD process. (Note that levels $\geq k$ of the background QBD process must be independent of the level of the foreground process.) Further discussion on such a subclass of extended RFBQBD processes is left as a future work.

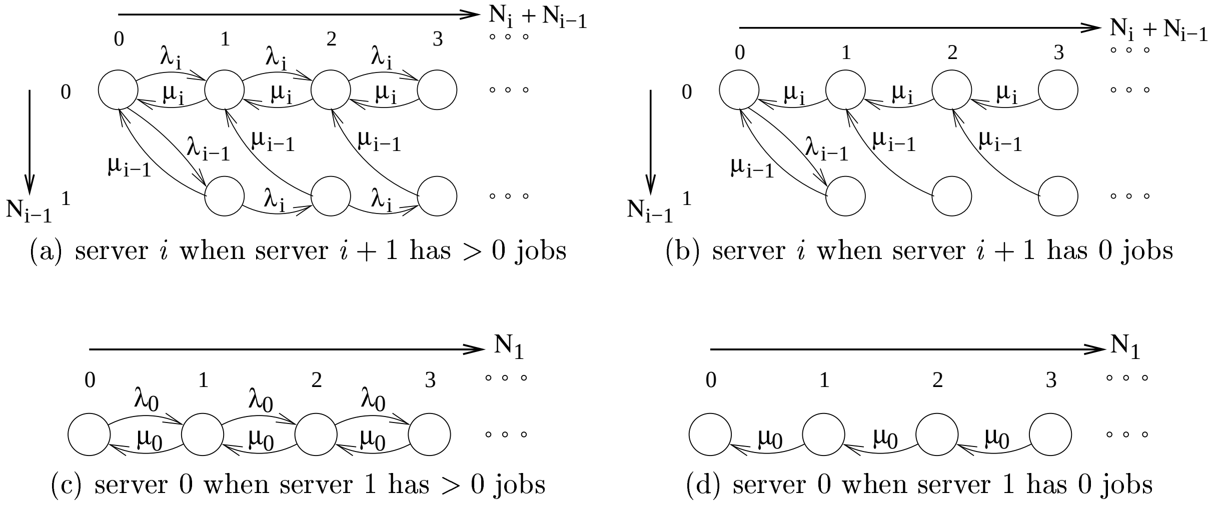
6 Case study: Task assignment with cycle stealing

In this section, we apply RDR, RDR-PI, and RDR-CI to the performance analysis of a task assignment policy, SBCS (recall Example 1 in Section 2). We consider m homogeneous servers and m classes of jobs. Class i jobs arrive at a dispatcher according to a Poisson process with rate λ_i, and their service time has an exponential distribution with rate μ_i for $0 \leq i \leq m-1$. Class $m-1$ jobs are always dispatched to server $m-1$. For $i < m-1$, a class i job first checks to see if the $(i+1)$-th server is idle. If so, the class i job is dispatched to the $(i+1)$-th server; otherwise, it is dispatched to the i-th server. We analyze the mean queue length at each server, and numerically evaluate the running time and accuracy of RDR, RDR-PI, and RDR-CI.

6.1 Modeling as an RFBQBD process

In this section, we model the number of jobs in the system under SBCS as an RFBQBD process. The RFBQBD process can then be analyzed via RDR, RDR-PI, and RDR-CI to obtain the stationary probabilities for the number of jobs (per class) at each server; the mean queue length follows immediately from the stationary probabilities.

Figure 6 shows the Markov chains for the number of jobs at each server. There are two Markov chains for server i, depending on whether there are any jobs at server $i+1$, for $0 \leq i \leq m-2$. There is only one Markov chain for server $m-1$, however, since the behavior (arrival and service) at server $m-1$ is independent of the other servers. Specifically, Figure 6(a) shows the Markov chain for the number of jobs at server i when server $i+1$ has > 0 jobs, for $1 \leq i \leq m-1$ ("server m" is defined to have > 0 jobs always). N_j denotes the number of class j jobs at the server. Since



(a) server i when server $i+1$ has >0 jobs (b) server i when server $i+1$ has 0 jobs

(c) server 0 when server 1 has >0 jobs (d) server 0 when server 1 has 0 jobs

Figure 6: *Markov chains for the number of jobs at each server. The top row shows the Markov chains for server $1 \le i \le m-1$, (a) when server $i+1$ has >0 jobs ("server m" is defined to have >0 jobs always), and (b) when server $i+1$ has 0 jobs. The bottom row shows the Markov chains for server 0, (c) when server 1 has >0 jobs, and (d) when server 1 has 0 jobs.*

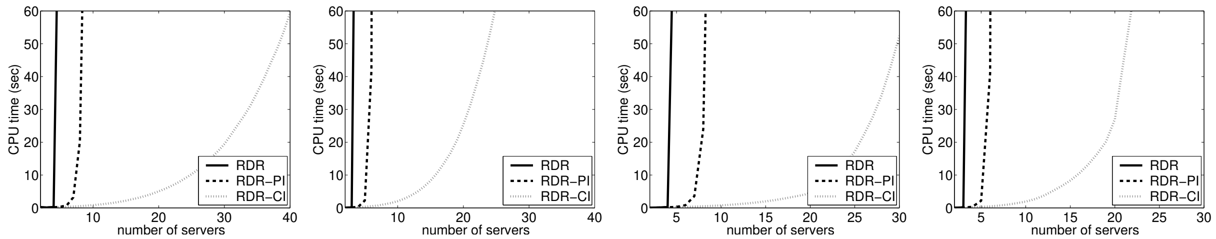
server $i+1$ has >0 jobs, an arrival of a class i job is dispatched to server i. Figure 6(b) shows the Markov chain for the number of jobs at server i when server $i+1$ has 0 jobs, for $1 \le i \le m-2$. Since server $i+1$ has 0 jobs, an arrival of a class i job is dispatched to server $i+1$; hence there is no transition due to class i job arrival. Figures 6(c)-(d) show the Markov chains for the number of jobs at server 0 when server 1 has >0 jobs and when server 1 has 0 jobs, respectively. Since "class -1" does not exist, there are no transitions corresponding to λ_{i-1} and μ_{i-1} in Figures 6(c)-(d).

Now, it is easy to model the number of jobs in the system under `SBCS` as an RFBQBD process. Here, the Markov chain at server i, B_i, corresponds to the i-th background process for $1 \le i \le m-1$, and the Markov chain at server 0, B_0, corresponds to the foreground process (the 0-th background process). Recall that there is only one Markov chain for server $m-1$, B_{m-1}. Since B_{m-1} is a QBD process, it can be seen as the $(m-1)$-th background process. There are two forms of transitions in B_{m-2}, depending on the number of jobs at server $m-1$, i.e. depending on the level of B_{m-1}. Thus, B_{m-2} can be seen as the $(m-2)$-th background process. Likewise, B_i, can be seen as the i-th background process, since its transitions are determined by the level of B_{i+1} for $0 \le i \le m-2$.

6.2 Numerical evaluation

We now evaluate the running time and accuracy of RDR, RDR-PI, and RDR-CI, when they are applied to the analysis of `SBCS`. In all the plots, we assume that the load made up of each class is fixed at 0.8 (i.e. $\frac{\lambda_i}{\mu_i} = 0.8$), and μ_i is chosen such that class 0 jobs are the shortest and class $m-1$ jobs are the longest ("stealing idle cycles of a server for longer jobs"; specifically, $\mu_i = 2^{-i}$), or μ_i is chosen such that class 0 jobs are the longest and class $m-1$ jobs are the shortest ("stealing idle cycles of a server for shorter jobs"; specifically, $\mu_i = 2^i$). For the analysis of a QBD process, which we need in the analysis via RDR, RDR-PI, and RDR-CI, we use algorithms in Sections 8.4

Stealing idle cycles of server for longer jobs | Stealing idle cycles of server for shorter jobs



(a) 1-phase approx. (b) 2-phase approx. (c) 1-phase approx. (d) 2-phase approx.

Figure 7: *The running time of RDR, RDR-PI, and RDR-CI.*

and 12.2 of [22], where the error bound, ϵ, is set at 10^{-6}.

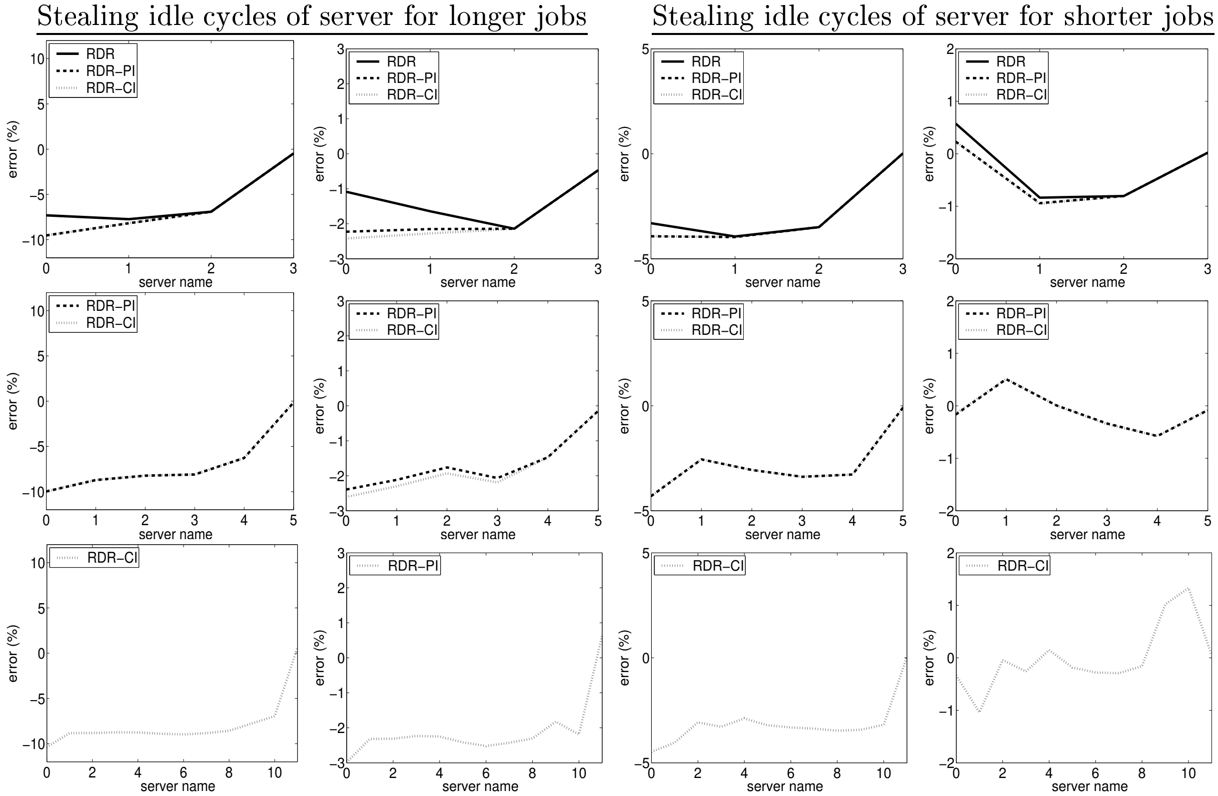
Figure 7 shows the running time (CPU time) of RDR, RDR-PI, and RDR-CI as a function of the number of servers. The running time is measured on a 1 GHz Pentium III with 512 MB RAM, using Matlab 6 running on Linux. In columns (a) and (c), the length of a (conditional) busy period (sojourn time in levels ≥ 1) is approximated by an exponential distribution ($N_{PH} = 1$), while in columns (b) and (d) it is approximated by a 2-phase PH distribution ($N_{PH} = 2$). In all the cases, the evaluation of RDR becomes prohibitive when $m \geq 5$. The running time of RDR-PI also quickly grows: the evaluation of RDR-PI becomes prohibitive when $m \geq 9$ in all the cases. The running time of RDR-CI grows far more slowly than RDR and RDR-PI. We are able to evaluate RDR-CI for up to 21-40 servers in less than a minute, depending on the input instance and the number of phases used in an approximate PH distribution. The running time of RDR, RDR-PI, and RDR-CI can be compared to the size of the submatrices (i.e. the number of phases) of the QBD process that approximates the RFBQBD process. In RDR, the size of the submatrices, $S_{RDR(m)}$, grows double exponentially; specifically, $S_{RDR(m)}$ can be determined by the following recursive formula: $S_{RDR(m)} = S_{RDR(m-1)} + (S_{RDR(m-1)})^2 N_{PH}$ and $S_{RDR(1)} = 1$. In RDR-PI, the size of the submatrices, $S_{PI(m)}$, grows exponentially: $S_{PI(m)} = (1 + N_{PH})^{m-1}$. In RDR-CI, the size of the submatrices, $S_{PI(m)}$, grows linearly: $S_{CI(m)} = 1 + (m-1)N_{PH}$.

Finally, we evaluate the accuracy of RDR, RDR-PI, and RDR-CI. We compare the mean queue length at each server predicted by RDR, RDR-PI, and RDR-CI against that predicted by simulation. Simulation is kept running until the simulation error becomes less than 1% with probability 0.95. The error of analysis is defined as a relative difference against simulated value:

$$\text{error (\%)} = 100 \times \frac{\text{(value by analysis)} - \text{(value by simulation)}}{\text{(value by simulation)}}.$$

Figure 8 shows the error of RDR, RDR-PI, and RDR-CI. The length of a (conditional) busy period is approximated by an exponential distribution ($N_{PH} = 1$) in columns (a) and (c) and by a 2-phase PH distribution ($N_{PH} = 2$) in columns (b) and (d). In the top row, the number of servers is $m = 4$, and all of RDR, RDR-PI, and RDR-CI are evaluated. In the middle row, the number of servers is $m = 6$, and here only RDR-PI and RDR-CI are evaluated, since the running time of RDR is too long with $m = 6$. In the bottom row, the number of servers is $m = 12$, and here only RDR-CI is evaluated[3]. The x-axis shows the "server name," where server name i denotes the

[3]Although RDR-CI can be evaluated with $m > 12$ (see Figure 7), it is very hard to have simulation converge



(a) 1-phase approx. (b) 2-phase approx. (c) 1-phase approx. (d) 2-phase approx.

Figure 8: *The accuracy of RDR, RDR-PI, and RDR-CI.*

i-th server, and hence corresponds to the i-th background process (the 0-th background process is the foreground process). Note that the scale of the y-axis is different for each column.

Overall, we can observe in Figure 8 that ignoring the variability (and the third moment)[4] of the sojourn time in levels ≥ 1 (see columns (a) and (c)) can lead to as high an error as 10%, while the error due to ignoring the dependency in the sequence of the sojourn times in levels ≥ 1 is bounded by 3% (see RDR-PI and RDR-CI in columns (b) and (d)). RDR tends to result in smaller error than RDR-PI and RDR-CI, but this is not entirely clear from Figure 8, since RDR can be evaluated only up to four servers. Also, from the top two rows, we can observe that RDR-PI and RDR-CI have similar error. Finally, we see that the error is greater in columns (a) and (b), where $\mu_i = 2^{-i}$, than in columns (c) and (d), where $\mu_i = 2^i$. We conjecture that this is primarily due to the fact that the busy period of server $i+1$, which is the sojourn time in levels ≥ 1 in the $(i+1)$-th background process, is relatively larger as compared to the service time and interarrival time at server i in columns (a) and (b), and hence the error in approximating/ignoring the distribution and dependency of the busy period has larger effect in predicting the mean number at server i.

with $m > 12$. Note that the fraction of arrivals of jobs with the largest size is very small: on average, there is only one arrival of a job with the largest size while there are 2^{m-1} arrivals of jobs with the smallest size. When $m = 12$, more than 1,000,000,000 events are needed for simulation to converge.

[4]In our instances, two phases are sufficient, in most cases, to match the first three moments by the approximate PH distribution.

7 Conclusion

In this paper, we have defined a class of Markov chains called recursive foreground-background QBD (RFBQBD) processes, and analyzed the stationary probabilities in an RFBQBD process. The analysis of an RFBQBD process constitutes a formalization and generalization of an analysis approach called recursive dimensionality reduction (RDR), which is developed and applied in a sequence of papers [14, 15, 31, 32, 33, 34, 40]. The formalization of RDR reveals that the computational complexity of RDR can grow double-exponentially in the number of recursions (the number of background processes). In fact, in the analysis of a task assignment policy (SBCS), RDR becomes computationally prohibitive for a system with more than four servers (i.e. more than three background processes).

Although RDR is not suitable for an RFBQBD process having a large number of background QBD processes (unless it has a special structure as in a priority M/M/k queue [34, 40]), our experience shows that RDR is computationally efficient and accurate when it is applied to an RFBQBD process having a small number of background QBD processes. For example, we have applied RDR to study the effectiveness of new task assignment policies [14, 15], to study the optimal threshold values in threshold-based policies [31, 32], and to study the effect of job size variability and priority on the performance of multiserver systems [31, 32, 33, 40]. Note that even with only a single background QBD process an exact analysis of the RFBQBD process is in general computationally difficult.

We have also proposed two new approximations in RDR, namely RDR-PI and RDR-CI, to reduce the computational complexity of RDR. These approximations ignore the dependency in the sequence of "lengths of busy periods," while keeping their marginal distribution. In the analysis of SBCS, the running time of RDR-CI is less than a minute for up to 21-40 servers (i.e. up to 20-39 background processes), depending on the settings. Furthermore, when the "length of a busy period" is approximated by a 2-phase PH distribution, the error of RDR-PI and RDR-CI is within a few percent, which is only slightly worse than that of RDR, for all the instances that we evaluate. The speed and accuracy of RDR-PI and RDR-CI allows us to analyze RFBQBD processes that cannot be analyzed via RDR due to computational complexity. Since there is a tradeoff between the speed and accuracy in RDR, RDR-PI, and RDR-CI, we can choose the degree of approximation (RDR, RDR-PI, or RDR-CI) depending on the size/type of the input.

Future work includes further approximations and evaluation of the error in the RDR-based analyses, including RDR, RDR-PI, and RDR-CI. First, to analyze a "larger" RFBQBD process (e.g. having more background QBD processes), one might want to introduce further approximations in RDR to reduce its computational complexity. In RDR, RDR-PI, and RDR-CI, only a portion with an infinite number of levels of a background QBD process is approximated by PH distributions with a small number of phases. A similar approach can be applied to a portion with a finite but large number of levels so that the background QBD process has a smaller number of levels. Second, since the RDR-based analyses involve approximations, it is important to study the error in such approximations. Although we evaluate the accuracy of RDR-based analyses numerically, no theoretical guarantee is proved. The formalization of RDR in this paper may be useful in establishing such a guarantee mathematically.

RDR, RDR-PI, and RDR-CI as described in this paper have been largely implemented and tested. The latest implementation is available at an online code repository[5].

[5] http://www.cs.cmu.edu/~osogami/code/

References

[1] I. Adan. *A compensation approach for queueing problems.* PhD thesis, Eindhoven University of Technology, 1991.

[2] I. Adan, O. Boxma, and J. Resing. Queueing models with multiple waiting lines. *Queueing Systems: Theory and Applications*, 37:65–98, 2001.

[3] I. Adan and J. Wessels. Analysis of the symmetric shortest queue problem. *Stochastic Models*, 6:691–713, 1990.

[4] I. Adan, J. Wessels, and W. Zijm. A compensation approach for two-dimensional Markov processes. *Advances in Applied Probabilities*, 25:783–817, 1993.

[5] F. Baskett, K. Chandy, R. Muntz, and F. Palacios-Gomez. Open, closed and mixed networks of queues with different classes of customers. *Journal of the Association for Computing Machinery*, 22:248–260, 1975.

[6] D. Bini, G. Latouche, and B. Meini. Solving nonlinear matrix equations arising in tree-like stochastic processes. *Linear Algebra and Applications*, 366:39–64, 2002.

[7] J. Blanc. Performance analysis and optimization with the power-series algorithm. In *Performance Evaluation of Computer and Communication Systems (Lecture Notes in Computer Science, Vol. 729)*, pages 53–80, 1993.

[8] L. Bright and P. Taylor. Calculating the equilibrium distribution in level dependent quasi-birth-and-death processes. *Stochastic Models*, 11:497–514, 1995.

[9] J. Buzen and A. Bondi. The response times of priority classes under preemptive resume in M/M/m queues. *Operations Research*, 31:456–465, 1983.

[10] J. Cohen. Boundary value problems in queueing theory. *Queueing Systems: Theory and Applications*, 3:97–128, 1988.

[11] J. Cohen. Two-dimensional nearest-neighbor queueing models, a review and an example. In F. Baccelli, A. Jean-Marie, and I. Mitrani, editors, *Quantitative Methods in Parallel Systems*, pages 141–152. Springer-Verlag, 1995.

[12] J. Cohen and O. Boxma. *Boundary Value Problems in Queueing System Analysis.* North-Holland Publ. Cy., 1983.

[13] M. Harchol-Balter, M. Crovella, and C. Murta. On choosing a task assignment policy for a distributed server system. *IEEE Journal of Parallel and Distributed Computing*, 59:204–228, 1999.

[14] M. Harchol-Balter, C. Li, T. Osogami, A. Scheller-Wolf, and M. Squillante. Task assignment with cycle stealing under central queue. In *Proceedings of the 23rd International Conference on Distributed Computing Systems (ICDCS)*, pages 628–637, May 2003.

[15] M. Harchol-Balter, C. Li, T. Osogami, A. Scheller-Wolf, and M. Squillante. Task assignment with cycle stealing under immediate dispatch. In *Proceedings of the Fifteenth ACM Symposium on Parallelism in Algorithms and Architectures (SPAA)*, pages 274–285, June 2003.

[16] G. Hooghiemstra, M. Keane, and S. van de Ree. Power series for stationary distributions of coupled processor models. *SIAM Journal on Applied Mathematics*, 48:1159–1166, 1988.

[17] E. Kao and K. Narayanan. Computing steady-state probabilities of a nonpreemptive priority multiserver queue. *Journal on Computing*, 2:211–218, 1990.

[18] E. Kao and K. Narayanan. Modeling a multiprocessor system with preemptive priorities. *Management Science*, 2:185–97, 1991.

[19] F. Kelly. *Reversibility and stochastic networks.* Wiley, 1979.

[20] J. Kingman. Two similar queues in parallel. *Annals of Mathematical Statistics*, 32:1314–1323, 1961.

[21] G. Koole. On the power series algorithm. In O. Boxma and G. Koole, editors, *Performance Evaluation of Computer and Communication Systems — Solution Methods, CWI Tract 105*, pages 139–155, 1998.

[22] G. Latouche and V. Ramaswami. *Introduction to Matrix Analytic Methods in Stochastic Modeling.* ASA-SIAM, Philadelphia, 1999.

[23] H. Leemans. *The Two-Class Two-Server Queue with Nonpreemptive Heterogeneous Priority Structures.* PhD thesis, K.U.Leuven, 1998.

[24] D. Miller. Steady-state algorithmic analysis of M/M/c two-priority queues with heterogeneous servers. In R. L. Disney and T. J. Ott, editors, *Applied probability - Computer science, The Interface, volume II*, pages 207–222. Birkhauser, 1992.

[25] I. Mitrani and P. King. Multiprocessor systems with preemptive priorities. *Performance Evaluation*, 1:118–125, 1981.

[26] M. Neuts. Moment formulas for the Markov renewal branching process. *Advances in Applied Probabilities*, 8:690–711, 1978.

[27] M. Neuts. *Matrix-Geometric Solutions in Stochastic Models: An Algorithmic Approach.* The Johns Hopkins University Press, 1981.

[28] B. Ngo and H. Lee. Analysis of a pre-emptive priority M/M/c model with two types of customers and restriction. *Electronics Letters*, 26:1190–1192, 1990.

[29] T. Nishida. Approximate analysis for heterogeneous multiprocessor systems with priority jobs. *Performance Evaluation*, 15:77–88, 1992.

[30] T. Osogami and M. Harchol-Balter. A closed-form solution for mapping general distributions to minimal PH distributions. In *Proceedings of the Performance TOOLS (Lecture Notes in Computer Science, Vol. 2794)*, pages 200–217, September 2003.

[31] T. Osogami, M. Harchol-Balter, and A. Scheller-Wolf. Analysis of cycle stealing with switching cost. In *Proceedings of the ACM SIGMETRICS*, pages 184–195, June 2003.

[32] T. Osogami, M. Harchol-Balter, and A. Scheller-Wolf. Analysis of cycle stealing with switching times and thresholds. *Performance Evaluation*, 2004 (accepted for publication).

[33] T. Osogami, A. Wierman, M. Harchol-Balter, and A. Scheller-Wolf. How many servers are best in a dual-priority FCFS system? Technical Report CMU-CS-03-213, School of Computer Science, Carnegie Mellon University, 2004.

[34] T. Osogami, A. Wierman, M. Harchol-Balter, and A. Scheller-Wolf. A recursive analysis technique for multi-dimensionally infinite Markov chains. In *Proceedings of the sixth workshop on mathematical performance modeling and analysis (MAMA)*, 2004 (to appear in *Performance Evaluation Review*).

[35] B. Rao and M. Posner. Parallel exponential queues with dependent service rates. *Computers and Operations Research*, 13(6):681–692, 1986.

[36] J. Sethuraman and M. Squillante. Analysis of parallel-server queues under spacesharing and timesharing disciplines. In *Matrix-Analytic Methods — Theory and Applications: Proceedings of the Fourth International Conference on Matrix-Analytic Methods in Stochastic Models*, pages 357–380. Imperial College, July 2002.

[37] M. Squillante, F. Wang, and M. Papaefthymiou. Stochastic analysis of gang scheduling in parallel and distributed systems. *Performance Evaluation*, 27/28:273–296, 1996.

[38] N. van Dijk. *Queueing networks and product forms: A systems approach.* Wiley, 1993.

[39] G. van Houtum. *New approaches for multi-dimensional queueing systems.* PhD thesis, Eindhoven University of Technology, 1995.

[40] A. Wierman, T. Osogami, M. Harchol-Balter, and A. Scheller-Wolf. Analyzing the effect of prioritized background tasks in multiserver systems. Technical Report CMU-CS-03-213, School of Computer Science, Carnegie Mellon University, 2004.