Truncating Multi-Dimensional Markov Chains with Accuracy Guarantee

Gagan Somashekar
Department of computer science
Stony Brook University
gsomashekar@cs.stonybrook.edu

Mohammad Delasay
College of Business
Stony Brook University
mohammad.delasay@stonybrook.edu

Anshul Gandhi
Department of computer science
Stony Brook University
anshul@cs.stonybrook.edu

***Abstract*—The ability to obtain the steady-state probability distribution of a Markov chain is invaluable for modern service providers who aim to satisfy arbitrary tail performance requirements. However, it is often challenging and even intractable to obtain the steady-state distribution for several classes of Markov chains, such as multi-dimensional and infinite state-space Markov chains with state-dependent transitions. Two examples include the M/M/1 with Discriminatory Processor Sharing (DPS) and the preemptive M/M/c with multiple priority classes and customer abandonment. This paper proposes a Lyapunov function-based state-space truncation technique for such Markov chains. Our technique leverages the available moments, or bounds on moments, of the state variables of the Markov chain to obtain tight truncation bounds while satisfying arbitrary probability mass guarantees for the truncated chain. We demonstrate the efficacy of our technique for the multi-dimensional DPS and M/M/c priority queue with abandonment and highlight the significant reduction in state space (as much as 72%) afforded by our approach compared to the state-of-the-art.**

***Index Terms*—Markov chains, state-space truncation, discriminatory processor sharing, priority queues, tail measures**

I. Introduction

Continuous-Time Markov chains (CTMCs) are widely used to model and analyze networked systems, such as processor-sharing (PS) and priority queue systems. Evaluating these systems' performance often requires the steady-state probability distribution of an underlying CTMC model. For example, to obtain *tail* measures (e.g., the tail queue length), which are the performance metric of choice for modern service operators such as those at Google [1] and Amazon [2], it is often necessary to first obtain the steady-state probability distribution by solving the balance equations governing the state transitions of a CTMC model.

Obtaining the exact steady-state probability distribution is not always practical or even possible. For many applications, the state space of the CTMC is infinite and multi-dimensional; exact analysis of such models is challenging. For CTMCs with specific structures, efficient numerical techniques exist for obtaining the exact steady-state distribution. For example, Matrix Analytic Methods are known to be efficient for solving CTMCs with a repeating pattern of transitions between adjacent states, including quasi-birth-and-death processes (QBDs, which are infinite state space multi-dimensional CTMCs in which states are organized into levels and transitions are skip-free between the levels) [3].

For chains with more general transitions, obtaining the exact steady-state probability distribution can be more challenging. For example, finding the distribution of the number of jobs in the system in the Discriminatory Processor Sharing (DPS) model (first introduced by Kleinrock in 1967 [4] and one of the models that we analyze in this paper) is still an open challenge. Even for CTMCs with a finite but large state space, computing the steady-state probability distribution can be computationally prohibitive [5]. We instead resort to obtaining accurate approximations of the steady-state probability distribution for such chains.

An approach to approximate the steady-state probability distribution of multi-dimensional infinite CTMCs with general state transitions is to truncate their state space in one or more dimensions and then solve for the steady-state probability distribution of the *truncated* CTMC using existing analytical or numerical methods. Truncation algorithms [6] (e.g., algorithms based on *Lyapunov* functions; see Section II) have been proposed in the literature to carefully find truncation bounds such that the steady-state probability distribution of the truncated CTMC closely approximates that of the *original* infinite CTMC. As acknowledged by prior work [6], an issue with such truncation techniques is that they lead to *loose truncation bounds*, which results in unnecessarily large truncated CTMCs and, consequently, *expensive computational effort* to analyze them.

In this paper, we use the moments (or bounds on moments) of the state variables of a CTMC to derive tighter truncation bounds while ensuring that these bounds satisfy the desired probability mass guarantees. By leveraging the moments and using concepts from probability theory (specifically the Paley-Zygmund inequality [7]), we scale the *drift* of the Lyapunov function (the expected rate of change in its value) more efficiently to achieve much tighter truncation bounds without increasing the computational complexity, compared to the existing Lyapunov-function truncation techniques. Our bounds are, in theory, at least as tight as those obtained by the state-of-the-art Lyapunov function-based procedure proposed by Dayar et al. [6]. In practice, our bounds are significantly tighter, resulting in as much as $7\times$ computational time efficiency.

We demonstrate the effectiveness of our proposed truncation technique by applying it to the M/M/1-DPS system, which is a multi-class extension of the classic processor sharing system where the server capacity can be unequally shared, via user-specified weights, among different job or customer classes. The M/M/1-DPS system is known to be a "*class of models notoriously hard to analyze in an exact manner*" [8]. We analyze the K-class DPS system (for $K=2,3,4$) using our truncation technique, leveraging the known moments of the queue-length distribution of the DPS system [9]. Across different parameter settings, our technique achieves, on average, 34% (and up to 72%) tighter truncation bounds than those obtained when applying the Dayar et al.'s procedure for the same desired accuracy. We also apply our truncation technique to the M/M/c+M model (an M/M/c queue with exponential abandonment) with multiple priority classes and preemptive service policy

as another example where our technique yields tighter truncation bounds. Our technique can reduce the size of the truncated state space by as much as 42% for the same accuracy level.

We numerically validate the accuracy of our truncation technique, where possible. For example, the M/M/1-DPS with equal weights for customer classes reduces to the well-studied M/M/1-PS system. Likewise, the marginal distribution of the higher priority jobs under the M/M/c+M queue with preemptive priority reduces to the distribution of jobs in the standard M/M/c. Across all validations, the maximum difference in per-state probability between our truncated CTMC and the original CTMC is about $3\times10^{-6}\%$ for the DPS system and about $9\times10^{-4}\%$ for the M/M/c+M queue with preemptive priority.

Finally, we use our truncation technique to conduct several performance analyses that are otherwise intractable, such as determining when the DPS system outperforms the PS system (and vice-versa) in terms of the tail of the number of jobs in the system or comparing the tail performance of the M/M/1-FCFS system with that of the M/M/1-DPS system. Such performance analyses are crucial for designing customer-facing web applications that meet strict tail performance targets [1, 2].

This work improves and refines the theory we proposed in an earlier, preliminary work [10] and demonstrates its broader application to processor sharing and priority systems through more in-depth analytical and numerical analysis

II. Background and Prior Work

A. State space reduction

When a CTMC with an infinite state space cannot be solved exactly, a natural alternative is to find an approximate solution by reducing the size of the state space. Prior research in this area has primarily focused on two key techniques: (i) state space *aggregation* and (ii) state space *truncation*.

The idea of aggregation is to replace a subset of the state space of a CTMC with a single state. However, a common limitation in aggregation approaches (e.g., see the works by Muntz et al. [5] and Mahévas and Rubino [11]) is the assumption that the state space can be decomposed by the user into two disjoint sets, with one set containing the states frequently visited by the system in steady state. Further, aggregation approaches either help find the performance measures or provide bounds on the performance measures (e.g., see Buchholz [12]) for Markov chains with infinite state space but do not find bounds on the steady-state distribution. Thus, guarantees on the probability mass after aggregation cannot be immediately obtained.

State space truncation: A more popular approach for state space reduction is truncation, whereby the state space of a Markov chain is truncated along one or more dimensions. However, ad-hoc truncation can result in inaccurate approximations. There has been prior work on truncation techniques that provide some upper bound on the loss of accuracy due to truncation. Bright and Taylor [13] propose a numerical method to solve LDQBDs, which involves iteratively finding a sufficiently large truncation level. However, the iterative procedure is computationally intensive and more concernedly, the authors explicitly state that the proposed method is not guaranteed to provide accurate results.

Lyapunov analysis has often been used in prior works to find bounds on the moments and tail probabilities for Markov chains. Bertsimas et al. demonstrate how lower and upper bounds on the moments and tail probabilities of a discrete-time Markov chain can be obtained provided that a suitable Lyapunov function can be found [14]. One of the conditions on the Lyapunov function is a finite "jump size," i.e., the requirement that the maximum change in the value of the drift function is bounded. Maguluri and Srikant build on prior works [14, 16] to find an upper bound on tail probabilities of CTMCs [15]. However, their approach requires the drift of the Lyapunov function to have a finite lower *and* upper bound, thus restricting the approach's applicability given the difficulty of finding Lyapunov functions even without these additional constraints [6]. By contrast, our technique does not impose any requirements on the Lyapunov function. In fact, the drift is unbounded from below for the Lyapunov functions we employ throughout this paper.

B. Lyapunov function-based truncation

Dayar et al. developed a Lyapunov function-based truncation method that provides probability mass guarantees for LDQBDs [6]. The central idea is to identify a subset of states towards which the CTMC drifts and then truncate the infinite state space to ensure that this subset is part of the truncated CTMC. Our truncation technique builds on this work, and so we provide an overview of Dayar et al..

Let $\{N(t),t\geq0\}$ be an ergodic k-dimensional CTMC with state space S and generic state $n=(n_1,n_2,...,n_k)$. Let N_i denote the random variable corresponding to n_i, with $N(t)=(N_1(t),...,N_k(t))$. Let $\pi(n)=\pi(n_1,n_2,...,n_k)$ denote the steady-state probability of being in state n, and let Q denote the infinitesimal generator matrix. Without loss of generality, assume that the CTMC is infinite in the first m-dimensions and finite in the remaining $(k-m)$ dimensions.

The stability of a Markov chain can be established if a ***Lyapunov function*** that maps the state space to positive real numbers is found such that its ***drift*** (the expected rate of change in the value of the Lyapunov function in a state) is negative outside a finite subset of the state space and is bounded in this finite subset; such a finite subset is referred to as the ***attractor set***, C. Formally, if $N(t)$ is ergodic, there exists a Lyapunov function $g:S\rightarrow R_{\geq0}$ and a set $C\subset S$ such that the following conditions hold for some $\gamma>0$ [6]:

(i) $d(n)\leq-\gamma, \forall n\in\overline{C}$, where $\overline{C}=S\setminus C$,
(ii) $d(n)<\infty, \forall n\in C$, and
(iii) $\{n\in S\,|\,g(n)\leq r\}$ is finite, $\forall r<\infty$,

where $d(n)$ denotes the value of the drift function in state n:

$$d(n)=(\mathrm{d}/\mathrm{d}t)\ E[g(N(t))\,|\,N(t)=n], \tag{1}$$

where $E[X]$ denotes the expectation of X. Dayar et al. use the above conditions to derive an upper bound on the probability mass (sum of steady-state probabilities of all states) in $\overline{C}$. The authors define a function $g^*(n)=g(n)/(c+\gamma)$, where $c=\sup_{n\in S}d(n)$ (note that c is finite from condition (ii)) and γ is as defined in condition (i). Figure 1 illustrates the above concepts pictorially, where the x-axis and y-axis correspond to the state variable and the value of the drift function, respectively. The figure also shows the parameter γ that partitions the state space into the attractor set and its complement. All the states whose value of the drift function is above the line corresponding to $-\gamma$ can be grouped into an attractor set.

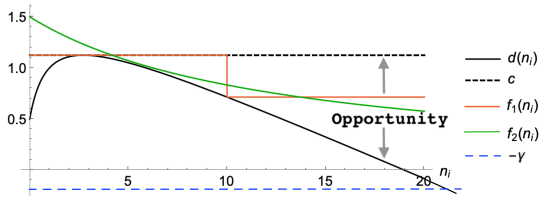


Fig. 1: Illustration of the drift $d(n_i)$, supremum c, parameter γ, and our state-dependent drift bounds $f_1(n_i)$ and $f_2(n_i)$.

Using conditions (i)–(iii), the authors derive an upper bound on the probability mass outside the attractor set C, thereby providing a lower bound on the probability mass inside C. Conditions (i) and (ii) and the fact that $c = \sup_{n\in S} d(n)$ yield:

$$d^*(n) = \frac{d(n)}{c+\gamma} \leq \frac{c}{c+\gamma}, \quad \forall n \in C; \text{ and } d^*(n) \leq -\frac{\gamma}{c+\gamma}, \quad \forall n \in \overline{C}$$
$$\implies d^*(n) \leq \frac{c}{c+\gamma} - I_{\overline{C}}\,, \tag{2}$$

where $I_{\overline{C}} = 1$ if $n \in \overline{C}$ and 0 otherwise. If d, g, and π are the vectors of drift function values, Lyapunov function values, and steady-state probabilities for all states, respectively, by the definition of drift in Eq. (1) and the fact that $\pi Q = 0$ [17], we have:

$$d^T = Qg^T \implies \pi d^T = \pi Q g^T = 0 \implies \pi d^{*T} = \pi Q g^{*T} = 0. \tag{3}$$

Using Eqs. (2) and (3), a bound on the probability mass in $\overline{C}$ is obtained as follows:

$$0 = \sum_{n\in S} d^*(n)\cdot\pi(n) \leq \sum_{n\in S} \pi(n)\cdot\frac{c}{c+\gamma} - \sum_{n\in\overline{C}} \pi(n)$$
$$\implies \sum_{n\in\overline{C}} \pi(n) \leq \sum_{n\in S} \pi(n)\cdot\frac{c}{c+\gamma} = \frac{c}{c+\gamma}. \tag{4}$$

This guarantees that the probability mass in C is at least $1 - c/(c+\gamma)$. Hence, the value of γ obtained by solving $c/(c+\gamma) = \epsilon$, where $0 < \epsilon < 1$, guarantees that a truncated CTMC containing C has at least $(1-\epsilon)$ fraction of the probability mass and thus loses at most ϵ fraction of the probability mass after truncation. Once γ is found, the set C can be found as follows:

$$C = \{n \in S \,|\, d(n) > -\gamma\}. \tag{5}$$

Omitting the states outside the attractor set C truncates the CTMC from "below" and from "above."

Note that Eq. (4) only provides an upper bound on the probability mass in $\overline{C}$; the actual probability mass in $\overline{C}$ could be much smaller than $c/(c+\gamma)$, as Dayar et al. acknowledge in their work [6]. Indeed, our experiments in Section IV show that the truncation bounds obtained via the above technique are loose. Our work aims to address this issue and provide tighter truncation bounds.

III. Our truncation technique

Figure 1 illustrates the high-level idea of our state space truncation technique for the Discriminatory Processor Sharing (DPS) system that we analyze later in Section IV. The solid black line is the drift as a function of the state variable, $d(n_i)$. Dayar et al.'s method obtains truncation bounds by bounding the drift function with the trivial upper bound of $c = \sup_{n\in S} d(n)$ (the dashed black line). The advantage of using the supremum is that the bound on $\sum_{n\in\overline{C}} \pi(n)$ in Eq. (4) can be easily obtained as $\sum_{n\in S} \pi(n)\cdot c/(c+\gamma) = c/(c+\gamma)$. However, there is a *gap* between the drift and the supremum, which tends to grow larger for higher values of n_i, as highlighted in Figure 1. The drift is a state-dependent function, but the supremum is a fixed function that does *not* adapt to changes in the state variate, thus making it a loose upper bound of the drift.

The key idea in our technique is to employ a *state-dependent bounding function* that mimics, to some extent, the changes in the drift function in response to the state variable to provide tighter upper bounds of the drift function; examples of such state-dependent bounding functions include a step function and a decaying function (e.g., $f_1(n_i)$ and $f_2(n_i)$ in Figure 1). However, when using a generic state-dependent bounding function, $f(n)$, in place of c in Eq. (4), the upper bound on the probability mass in $\overline{C}$ may not be easily obtained in closed-form, making it difficult to solve for the set C. We formalize this challenge below.

Generic bounding functions: Consider a generic state-dependent bounding function $f(n)$ that bounds the drift $d(n)$, i.e., $f(n) \geq \max(d(n), 0)$, $\forall n \in S$. Expanding Eq. (3) gives us:

$$0 = \sum_{n\in S} \pi(n)\cdot d(n) = \sum_{n\in C} \pi(n)\cdot d(n) + \sum_{n\in\overline{C}} \pi(n)\cdot d(n)$$
$$\implies 0 \leq \sum_{n\in C} \pi(n)\cdot d(n) - \gamma \sum_{n\in\overline{C}} \pi(n) \quad \because d(n) \leq -\gamma \quad \forall n \in \overline{C}$$
$$\implies \sum_{n\in\overline{C}} \pi(n) \leq \frac{\sum_{n\in C} \pi(n)\cdot f(n)}{\gamma} \quad \because f(n) \geq \max(d(n), 0). \tag{6}$$

The truncated CTMC consisting of the set C can now be obtained by setting the right-hand-side of Eq. (6) to ϵ, solving for γ, and then using Eq. (5). However, this requires knowing the exact value or an upper bound of the term $\sum_{n\in C} \pi(n)\cdot f(n)$. We show, in subsections III-A and III-B, examples of one-dimensional and two-dimensional generic bounding functions $f(n)$ that result in simple expressions for $\sum_{n\in C} \pi(n)\cdot f(n)$. In all the cases, the final expressions are in terms of the moments of the state variables.

Reliance on the CTMC moments: Though our technique relies on the knowledge of the first few moments of the original CTMC's state variables, it can also be applied to CTMCs with known lower bounds on the moments. In general, knowledge of moments may not be enough to obtain the steady-state probability distribution [18]. The DPS system is an example of a CTMC for which the moments of the number of jobs are computationally achievable [9]; however, obtaining its closed-form and exact steady-state distribution has proven to be extremely challenging [18]. The preemptive M/M/c priority queue with abandonment is another example with some known moments (e.g., for the highest priority jobs, since they constitute the classic M/M/c queue), but the exact steady-state probability distribution is not known, especially for $c > 2$ [19].

A. Truncation bounds based on the moments

We now employ a specific function, a simple *step function*, along one of the dimensions of a CTMC to bound its drift function. Consider a 1-D step function (depicted in Figure 1 as $f_1(n)$) that initially is equal to the supremum and then drops to a lower value, c_1, along one dimension of the state space, resulting in a tighter bounding

of the drift for larger values of the state variate. Note that the step function subsumes the supremum function used by Dayar et al. [6].

Consider a k-dimensional CTMC with state space S with a generic state denoted by $n=(n_1,n_2,...,n_k)$, and let the CTMC be infinite in m-dimensions and finite in the remaining $(k-m)$ dimensions. We denote with N_j the random variable corresponding to the j^{th} dimension of the state space, n_j. We improve the upper bound of the drift along an arbitrary infinite dimension, say dimension i, corresponding to N_i. We formally define the step function, which drops to $c_1 \le c$ for $R=\{n \mid n_i > \overline{n}\}$, where $\overline{n}$ is a parameter, as:

$$f_1(n)=\begin{cases} c & =\sup_{\forall n\in S} d(n), \quad \forall n\in \overline{R}, \\ c_1 & =\sup_{\forall n\in R} d(n), \quad \forall n\in R. \end{cases} \tag{7}$$

Substituting $f_1(n)$ in place of $f(n)$ in Eq. (6) and rearranging the terms gives us:

$$\begin{aligned} \gamma\sum_{n\in\overline{C}}\pi(n) &\le c\sum_{n\in C\cap\overline{R}}\pi(n)+c_1\sum_{n\in C\cap R}\pi(n) \\ &= c\sum_{n\in\overline{R}}\pi(n)+c_1\sum_{n\in R}\pi(n)-c\sum_{n\in\overline{C}\cap\overline{R}}\pi(n)-c_1\sum_{n\in\overline{C}\cap R}\pi(n) \\ &\le c\sum_{n\in\overline{R}}\pi(n)+c_1\sum_{n\in R}\pi(n)-c_1\sum_{n\in\overline{C}}\pi(n) \quad \because -c\le -c_1 \\ \implies \sum_{n\in\overline{C}}\pi(n) &\le \frac{c}{c_1+\gamma}-\frac{c-c_1}{c_1+\gamma}\sum_{n\in R}\pi(n). \end{aligned} \tag{8}$$

Recall from Section II-B that $\overline{C}$ is the set of states outside the attractor set. Thus, $\sum_{n\in\overline{C}}\pi(n)$ represents the probability mass outside the attractor set. To obtain the probability mass guarantee on $\sum_{n\in\overline{C}}\pi(n)$, we require a lower bound on the tail probability, $\sum_{n\in R}\pi(n)$. While any applicable lower bound can be employed, we leverage a generic lower bound.

Definition 1. Paley-Zygmund inequality [7]: *For a positive random variable X with finite variance and $0\le\theta\le 1$, $\Pr(X>\theta E[X]) \ge (1-\theta)^2 E[X]^2/E[X^2]$.*

Applying the Paley-Zygmund inequality for N_i and setting $\overline{n}=\theta E[N_i]$, we have, from Eq. (8):

$$\sum_{n\in\overline{C}}\pi(n) \le \frac{c}{c_1+\gamma}-\frac{c-c_1}{c_1+\gamma}(1-\theta)^2\frac{E[N_i]^2}{E[N_i^2]}. \tag{9}$$

The above upper bound on the probability mass in $\overline{C}$ (or the lower bound on the probability mass inside C) results in a provably tighter truncation bound compared to Dayar et al. when $\epsilon \le \frac{E[N_i]^2}{E[N_i^2]}$ (proof deferred to Appendix B of the technical report [20]):

Lemma 1. *The truncation obtained via our 1-D step bounding function (given in Eq. (7)) is at least as tight as that obtained via the supremum bounding function, as employed in Dayar et al., when $\epsilon \le \frac{E[N_i]^2}{E[N_i^2]}$, where i is an arbitrary infinite dimension of the CTMC.*

B. Tighter truncation bounds using joint moments of state variables

In general, since the drift function is defined on the state space of the CTMC, it can be multi-variate. To obtain tighter truncation bounds, we now consider multi-dimensional bounding functions. In such cases, the joint moments of the state variables may be needed to obtain the truncation bounds. For simplicity, we consider two-dimensional bounding functions; however, our proposed technique can be extended to higher-dimensional functions.

Figure 2a plots a two-dimensional step function that bounds the drift; here, we assume that the CTMC is k-dimensional, $k>1$, but we only show the bounding function for the two dimensions, say i and j, along which it takes a step. Mathematically, we define the 2-D step function as:

$$f_{12}(n)=\begin{cases} c & =\sup_{\forall n\in S} d(n), \quad \forall n\in \overline{R}, \\ c_1 & =\sup_{\forall n\in R} d(n), \quad \forall n\in R, \end{cases} \tag{10}$$

where $\overline{R}=\{n\in S \mid \max\{n_i,n_j\}\le\overline{n}\}$ and $R=S\setminus\overline{R}$. Thus, the bounding function takes the value c in the "square" region (if projected onto two dimensions) defined by $n_i\le\overline{n}$ and $n_j\le\overline{n}$, and takes the value c_1 outside this region. Substituting Eq. (10) in the generic upper bound on the probability mass in $\overline{C}$ (Eq. (6)) gives us a result similar to Eq. (8):

$$\sum_{n\in\overline{C}}\pi(n) \le \frac{c}{c_1+\gamma}-\frac{c-c_1}{c_1+\gamma}\sum_{n_i\in R}\pi(n). \tag{11}$$

To obtain a lower bound on the probability mass in R, we define the new set $\overline{T}=\{n\in S \mid n_i+n_j\le 2\cdot\overline{n}\}$ and $T=S\setminus\overline{T}$. Let $\pi(\mathcal{S})$ denote $\sum_{n\in\mathcal{S}}\pi(n)$. Since $T\subset R$, we have $\pi(R)\ge\pi(T)$. Figure 2b illustrates the different regions of the state space S. We now use the Paley-Zygmund inequality to find the lower bound on $\pi(T)$ by considering $Z=N_i+N_j$:

$$\begin{aligned} &P(Z>\theta E[Z]) \ge (1-\theta)^2\frac{E[Z]^2}{E[Z^2]}, \text{ with } 2\overline{n}=\theta E[Z] \\ &\implies \pi(R)\ge\pi(T)=P(Z>\theta E[Z])\ge(1-\theta)^2\frac{E[Z]^2}{E[Z^2]}. \end{aligned} \tag{12}$$

Finally, substituting Eq. (12) in Eq. (11) gives us our upper bound on the probability mass in $\overline{C}$ as:

$$\sum_{n\in\overline{C}}\pi(n) \le \frac{c}{c_1+\gamma}-\frac{c-c_1}{c_1+\gamma}(1-\theta)^2\frac{E[Z]^2}{E[Z^2]}. \tag{13}$$

The above bound is provably tighter than Dayar et al. when $\epsilon \le \frac{E[Z]^2}{E[Z^2]}$ (proof deferred to Appendix B of tech. report [20]):

Lemma 2. *The truncation obtained via the 2-D step bounding function (given in Eq. (10)) is at least as tight as that obtained via the supremum bounding function, as employed in Dayar et al., when $\epsilon \le \frac{E[Z]^2}{E[Z^2]}$ where $Z=N_i+N_j$ and N_i and N_j represent state variables corresponding to two arbitrary infinite dimensions of the CTMC.*

IV. Application to the DPS system

We now evaluate the efficacy of our truncation technique by applying the truncation bounds for the Discriminatory Processor Sharing (DPS) system. We first consider the DPS system with two customer classes for ease of exposition. We investigate the tightness of the drift bounding functions presented in Section III and compare the resulting bounds with those obtained from Dayar et al. [6].

DPS was first introduced by Kleinrock as a generalization to the PS system [4]. While all customer classes receive equal server capacity in the case of an egalitarian PS policy, the server capacity under DPS is processor shared based on a given weight

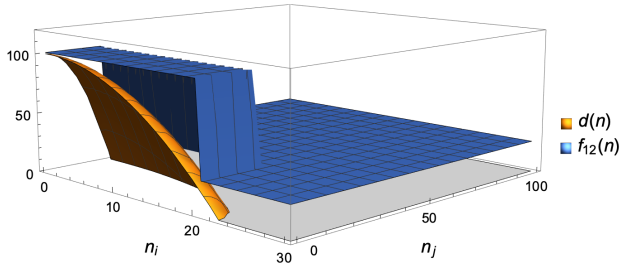


Fig. 2(a): Illustration of a 2-D drift bounding function.

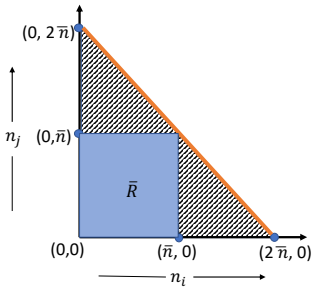


Fig. 2(b): State space subsets. The shaded square corresponds to the set $\overline{R} = \{n \in S \mid \max\{n_i, n_j\} \leq \overline{n}\}$ and the region bounded by the orange line and the axes corresponds to the set $\overline{T} = \{n \in S \mid n_i + n_j \leq 2 \cdot \overline{n}\}$.

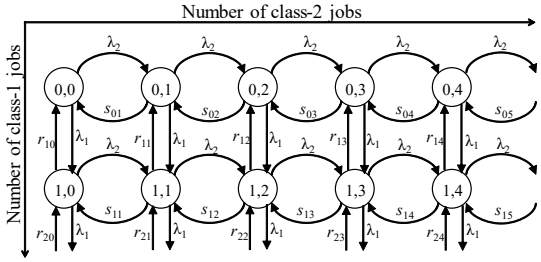


Fig. 3: M/M/1-DPS with two customer classes; for state (n_1,n_2), n_1 and n_2 are the number of class-1 and class-2 jobs, respectively.

vector $\alpha = (\alpha_1,\alpha_2,...,\alpha_k)$, where α_i is the weight associated with class-i customers. If there are N_i jobs of class-i, each class-j job gets a fraction $\alpha_j / \sum_{i=1}^{k} \alpha_i N_i$ of the server's capacity.

M/M/1-DPS system: Consider an M/M/1 system operating under the DPS policy with two customer classes, where the server's capacity is shared between two customer classes with the service priority expressed through weights α_1 and α_2. Arrivals for each customer class follow a Poisson distribution with mean $\lambda_i, i \in \{1,2\}$, and the service times for each class follow an Exponential distribution with mean $1/\mu_i, i \in \{1,2\}$. Figure 3 shows the CTMC for M/M/1-DPS in which a state is represented by the pair (n_1,n_2), where n_1 and n_2 are the number of jobs in the system for classes 1 and 2, respectively; note that the CTMC is infinite in both dimensions. The transition rates from state (n_1,n_2) to (n_1-1,n_2) and $(n_1,\ n_2\ -\ 1)$ are $r_{n_1,n_2} = n_1\alpha_1\mu_1/(n_1\alpha_1+n_2\alpha_2)$ and $s_{n_1,n_2} = n_2\alpha_2\mu_2/(n_1\alpha_1+n_2\alpha_2)$, respectively.

Significance of the M/M/1-DPS system and the challenges in solving its underlying CTMC: The M/M/1-DPS system, with the proper choice of weights, has been shown to outperform the classical M/M/1-PS system for more than one customer class [21]. This has sparked interest in the community in the last decade to investigate the performance of DPS under various traffic regimes. Despite the popularity of the DPS model, which was first introduced in the late 1960s [4], the exact steady-state probability distribution of its underlying CTMC continues to remain elusive. This is because of the complex and non-repeating structure of its multi-dimensional and infinite CTMC. Specifically, the per-class service rate transitions (r_{n_1,n_2} and s_{n_1,n_2} in Figure 3) depend on the current number of customers in each class. Exact analysis has been performed only for *finite* DPS queues [22].

A. Results for truncation bounds

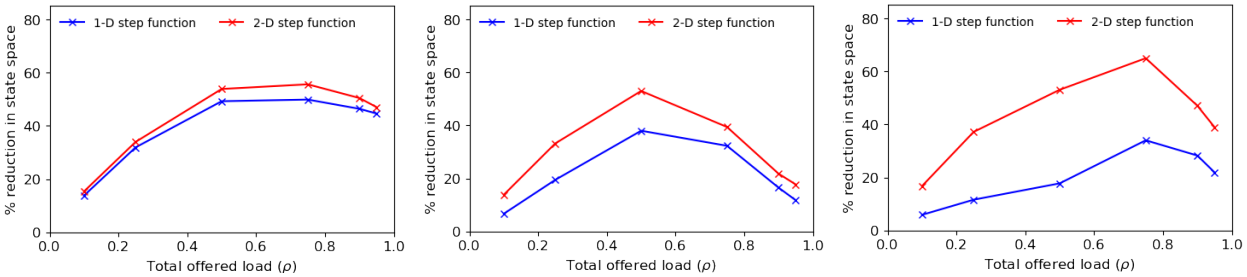
Fortunately, the exact moments of M/M/1-DPS queue-length distribution are known [9] in terms of the solution to a system of linear equations. This allows us to apply our moment-based truncation bounds via Eqs. (9) and (13). To apply our truncation bounds (and Dayar et al.'s bounds for comparison) to the M/M/1-DPS system, we employ the following feasible Lyapunov function, motivated by prior work in the stability literature [23]: $g(n_1,n_2) = (\alpha_1 n_1^2)/2\lambda_1 + (\alpha_2 n_2^2)/2\lambda_2$. The drift function for the M/M/1-DPS chain in the state (n_1,n_2), obtained by substituting $g(n_1,n_2)$ in Eq. (1), is as follows:

$$d(n_1,n_2) = \lambda_1(g(n_1+1,n_2) - g(n_1,n_2)) + \lambda_2(g(n_1,n_2+1) - g(n_1,n_2)) + s_1(g(n_1-1,n_2) - g(n_1,n_2)) + s_2(g(n_1,n_2-1) - g(n_1,n_2)). \quad (14)$$

For the *1-D step* drift bounding function, we start by setting an appropriate value for $\overline{n}$ in Eq. (7), which in turn is determined via θ since $\overline{n} = \theta E[N_i]$ where $i \in \{1,2\}$ is the dimension along which the drift bound is being improved. Noting that a smaller θ provides a tighter bound in Eq. (9), we set $\theta = 0.01$; this value of θ also satisfies the requirements of Lemmas 1 and 2. We then derive $\overline{n}$ by obtaining $E[N_i]$ via the known first moment of the i^{th} state variable of the M/M/1-DPS model [9]. We compute c and c_1 via Eq. (7). Using the known second moments [9], we compute the right-hand-side of Eq. (9); by setting this to ϵ (the tolerance for probability mass loss due to truncation), we solve for γ, which in turn gives us the attractor set C via Eq. (5). The chain is then truncated to include all states in C. We employ the step function over either dimension ($i = \{1,2\}$ in Eq. (7)) and use the tighter of the two. Finally, the CTMC is truncated along the two dimensions at $m_1 = \max_{(n_1,n_2)\in C} n_1$ and $m_2 = \max_{(n_1,n_2)\in C} n_2$. A step-by-step illustration is provided in Appendix A of the technical report [20].

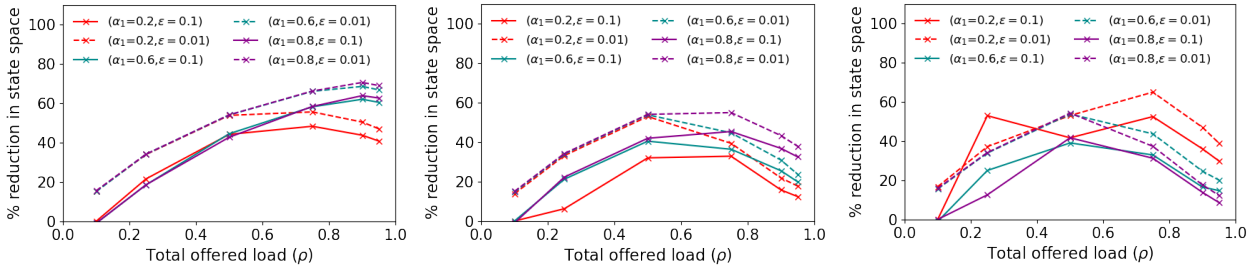
For the *2-D step* drift bounding function, we again set θ to a low value and set $\overline{n} = (\theta E[Z])/2$, where $Z = N_i + N_j$. The remaining steps are similar to the 1-D step bounding function discussed above.

Evaluation results: To evaluate the truncation improvement over the Dayar et al., we numerically experiment with different parameter values spanning the total offered load in the range $[0.1,0.95]$; the total offered load is expressed as $\rho = \rho_1 + \rho_2$, where $\rho_i = \lambda_i/\mu_i$. We set $\mu_2 = 1$ and $\mu_1 = 1.2$ and vary λ_1 and λ_2. Figure 4 shows the reduction in the state space afforded by our drift bounding functions as a function of the total offered load for different class-1 load shares



(a) Case 1: $\rho_1 < \rho_2$ ($\rho_1/\rho = 0.1$). (b) Case 2: $\rho_1 = \rho_2$ ($\rho_1/\rho = 0.5$). (c) Case 3: $\rho_1 > \rho_2$ ($\rho_1/\rho = 0.8$).

Fig. 4: State space reduction by our bounding functions over Dayar et al. [6] for M/M/1-DPS under different total offered loads (ρ) and class-1 load shares (ρ_1/ρ); $\epsilon = 0.01$, $\alpha_1 = 0.2$, and $\alpha_2 = 1 - \alpha_1 = 0.8$.



(a) Case 1: $\rho_1 < \rho_2$ ($\rho_1/\rho = 0.1$). (b) Case 2: $\rho_1 = \rho_2$ ($\rho_1/\rho = 0.5$). (c) Case 3: $\rho_1 > \rho_2$ ($\rho_1/\rho = 0.8$).

Fig. 5: State space reduction by our 2-D step bounding function over Dayar et al. [6] for M/M/1-DPS under different DPS weights (α_1), total offered loads (ρ), and class-1 load shares (ρ_1/ρ).

when $\alpha_1 = 0.2$, $\alpha_2 = 1 - \alpha_1 = 0.8$, and $\epsilon = 0.01$. The improvement is typically greater for moderate total offered loads ($\rho \approx 0.5$).

In general, the 2-D step function provides more improvement over Dayar et al. compared to the 1-D step function, with as much as 65% reduction in state space. In other words, using our technique, the truncated DPS CTMC can be up to 65% smaller while providing the same probability mass accuracy guarantee ($\epsilon = 0.01$). For this peak reduction case, Dayar et al.'s method truncates the CTMC at $n_1 = 1358$ and $n_2 = 101$, whereas our 2-D step bounding function truncates at $n_1 = 801$ and $n_2 = 60$; the truncation bounds from "below" are $n_1 = 0$ and $n_2 = 0$ in both cases.

Across all experiments in Figure 4, the average improvements over Dayar et al. are around 39% and 27% for the 2-D and 1-D step functions, respectively. The corresponding average improvements for $0.5 \leq \rho \leq 0.95$ are 45% and 33%; since the truncated CTMC contains more states for higher loads, the absolute reduction in state space (the number of states) is much higher for this range.

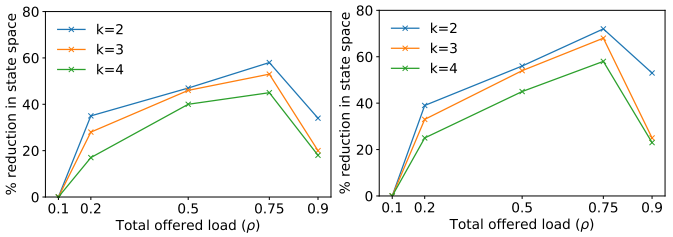
Further analysis: We consider the better-performing drift bounding function and the 2-D step function and experiment with different α_1 values. Figure 5 shows the state space reduction over [6] as a function of total offered load for different class-1 load shares and for different α_1 and ϵ values. As before, the improvement is higher for moderate offered loads. In general, the improvement increases as α_1 increases, except when the load share of class-1 is high, in which case the improvement tends to decrease as α_1 increases. The improvements are largely *insensitive* to the truncation error guarantee, ϵ; we also experimented with smaller ϵ values with similar insensitivity results.

Across all experiments in Figure 5, the average improvement is around 33%, 34%, and 35% for $\alpha_1 = 0.2$, $\alpha_1 = 0.6$, and $\alpha_1 = 0.8$, respectively. The corresponding improvements for $0.5 \leq \rho \leq 0.95$ are 41%, 42%, and 44%. The maximum improvement is 71%, with Dayar et al. truncating at $n_1 = 108$ and $n_2 = 3292$, while our 2-D step bounding function truncates at $n_1 = 65$ and $n_2 = 1610$ (here, the load share of class-2 was higher, so n_2 is truncated at a larger value than n_1).

By providing tighter truncation, our technique *significantly reduces the computational effort required* to solve the truncated CTMC. Averaged across all cases in Figure 5, solving the truncated CTMC (by obtaining the steady-state probabilities via solving the relevant balance equations) is $3\times$ faster when employing our bounds than those obtained by Dayar et al. [6].

Validation results: For validation, we compare the obtained moments from the truncated CTMC with the exact moments provided in Rege and Sengupta [9]. Using the 1-D step function and setting $\epsilon = 0.1$, the average difference between our results and the exact results for the first, second, and third moments of the number of jobs in the system (for either class) across various parameter settings is around 4×10^{-4}%, 10^{-3}%, and 5×10^{-3}%, respectively. We further validate our technique by comparing the steady-state distribution of the truncated CTMC for $\alpha_1 = \alpha_2 = 0.5$ with that of the classical processor sharing system (a DPS with $\alpha_1 = \alpha_2$). The maximum observed difference in per-state probability is only around 10^{-5}%.

Application to higher-dimensional DPS chains: Since the exact moments of the M/M/1-DPS queue-length distribution are known for any number of customer classes, our truncation technique can be readily applied to the M/M/1-DPS CTMC for an arbitrary number of customer classes (along the same lines as for the 2-class M/M/1-DPS CTMC analyzed above). Note that for a k-class M/M/1-DPS system, the CTMC will be k-dimensional and infinite in all k dimensions. We use the same form of Lyapunov function for the k-dimensional CTMCs as employed for the 2-dimensional CTMC.



(a) 1-D step function (b) 2-D step function

Fig. 6: State space reduction by our step bounding functions over Dayar et al. for M/M/1-DPS with k customer classes; $\epsilon = 0.1$.

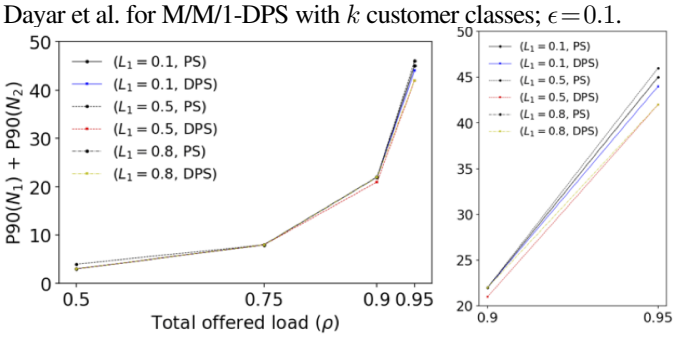


Fig. 7: Comparison of the sum of $P90$ for both classes of jobs in the system under the PS and DPS scheduling policies ($L_1 = \rho_1/\rho$); results in the $[0.9\ 0.95]$ range are zoomed in for illustration.

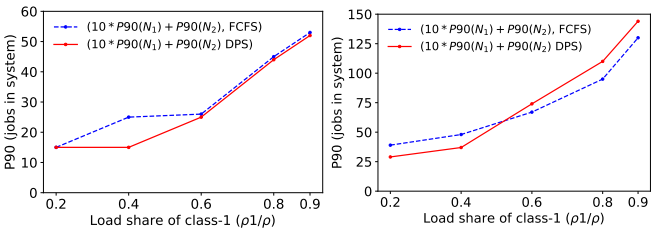
Figure 6 shows the reduction in state space afforded by our truncation technique over Dayar et al. [6] as a function of the total offered load for the M/M/1-DPS with $k = 2,3,4$ customer classes. We set $\epsilon = 0.1$ and consider equally-distributed load shares with $\alpha_i = 1/k$ for all k customer classes. We observe that the *reduction in state space increases with* k, with the 2-D step function generally providing better improvements.

Across all experiments shown in Figure 6, the average (and maximum) state space by our technique is 44% (72%), 36% (68%), and 30% (58%) for $k = 4$, $k = 3$, and $k = 2$, respectively, using the 2-D step bounding function. The corresponding improvements for the 1-D step bounding function are 35% (58%), 30% (53%), and 24% (45%). In terms of computational effort, across all experiments in Figure 6, solving the truncated CTMC is as much as 7× faster when employing our bounds.

B. Applications of the truncated DPS CTMC

For modern web services, such as Amazon [2] and Google [1], *tail performance measures*, e.g., tail latency or tail queue length, are critical to providing acceptable performance to customers. To analyze the M/M/1-DPS performance, we consider the 90$^{\text{th}}$ percentile of the number of jobs in the system, denoted as $P90$. Other tail measures can be similarly computed; for example, tail response time can be computed by truncating the CTMC and leveraging existing results for finite DPS queues [22]. We employ our 2-D step drift bounding function to find the truncation bounds by setting $\epsilon = 0.01$. We then solve the balance equations for the truncated CTMC and obtain its steady-state probability distribution; we use the resulting probability distribution to compute the $P90$ values.

1) Use case 1: DPS versus PS, for tail metrics: Prior work has shown that DPS can outperform, in terms of the mean queue length, the classical M/M/1-PS system with more than one customer class



(a) $\rho = 0.7$ (b) $\rho = 0.9$

Fig. 8: Performance of M/M/1-DPS and non-preemptive M/M/1-FCFS with priority for different load conditions.

when a larger weight is assigned to the class with a smaller mean service time [21]. We now investigate whether this result still holds for tail measures. For both PS and DPS, we consider (the same) two customer classes with the offered load for class i being $\rho_i = \lambda_i/\mu_i$, and the total offered load $\rho = \rho_1 + \rho_2$.

Figure 7 shows the summation of $P90$ values of the number of customers in the system for both classes, $P90(N_1) + P90(N_2)$, as a function of ρ for the M/M/1-PS (black line with circles) and different M/M/1-DPS systems with varying ρ_1 values; we set the parameters for the figure such that the mean service time of class-1 customers is lesser than that of class-2 customers ($1/\mu_1 < 1/\mu_2$) and set $\alpha_1 = 0.9$ to give preferential treatment to the class-1. We separately zoom in and plot the $[0.9\ 0.95]$ x-axis range results for illustration. Note that the $P90$ values for M/M/1-PS for different values of ρ_1/ρ are quite similar and appear as a single line.

We find that the *DPS system outperforms the PS system* for almost all parameter configurations shown in the figure, with more pronounced (and visible) improvements, ranging from 2%–9%, at higher offered loads (see zoomed-in plot on the right of Figure 7). The average improvement over all cases shown in Figure 7 is about 4%. For the DPS cases in Figure 7, $\alpha_1 = 0.1$, and thus $\alpha_2 = 0.9 > \alpha_1$. By providing higher priority, or weights, for the class with smaller mean service time (with smaller jobs), DPS can achieve better performance by as much as 9%. We also experimented with $\alpha_1 > \alpha_2$, and found that, in this case, PS outperforms DPS. We analyze the impact of class weights on the performance of M/M/1-DPS in detail as a separate use case in our technical report [20].

2) Use case 2: When is DPS better than FCFS with priority?: This use case focuses on the long-standing debate between PS and FCFS policies [24]. We investigate the performance of M/M/1-DPS when compared with that of an M/M/1-FCFS with priority; as before, we consider two customer (priority) classes. For SLO, we consider the weighted sum of the tail of the number of jobs in system: $10 \times P90(N_1) + P90(N_2)$. For the M/M/1-FCFS with priority, we employ existing analytical results to obtain the $P90$ values [25].

We start by considering the non-preemptive version of the M/M/1-FCFS with priority. Figure 8 shows our results for the weighted metric as a function of ρ_1/ρ. To prioritize class-1 jobs, we set $\alpha_1 = 0.95$ for DPS; results are qualitatively similar, but not as pronounced, under other $\alpha_1 > 0.5$ values. We set $\mu_1 = 0.6$ and $\mu_2 = 1$ and experiment with total offered loads $\rho = 0.7$ and $\rho = 0.9$. For each case, we find values of λ_1 and λ_2 such that the total load is ρ and the fractional load of customer 1 is as shown on the x-axis.

When $\rho = 0.7$, both policies perform similarly. However, when $\rho = 0.9$, we observe an interesting behavior. *When the load*

share of customer class-1 is low, DPS performs better, whereas when the load share of class-1 is high, FCFS performs better. This is because for the low load share of class-1, the load is higher for class-2 jobs, and due to the non-preemptive FCFS policy we consider, class-2 jobs can "hold up" incoming class-1 jobs, resulting in a high penalty under our metric that gives a higher weight to $P90(N_1)$. For the high load share of class-1, FCFS outperforms DPS since FCFS provides strict priority, as opposed to the α_1-weighted DPS policy, which still provides a weight of $1-\alpha_1 = 0.05$ for class-2 jobs. We observed a similar trend for other values of μ_1 and μ_2 as well.

We also compared the performance of M/M/1-DPS with that of preemptive M/M/1-FCFS. However, in this case, across different parameter settings, the preemptive FCFS always outperforms M/M/1-DPS, with respect to the tail of the number of jobs in the system.

V. Application to the preemptive M/M/c+M queue

Multi-server priority queues with preemption have been widely employed to model differentiated service for multiple customer classes. For the case of customer abandonment (impatient customers), referred to as the M/M/c+M priority queue, the exact steady-state probability distribution for the low-priority customers is not known. However, truncating using the 1-D step bounding function (See Section III-A) only requires the moments of one of the customer classes, which are readily available for the high-priority class [26]. Employing the 1-D step function for the M/M/3+M model provides 21% (52%) average (peak) reduction in state space over Dayar et al. [6]. A detailed discussion of our truncation bounds and results is provided in Section V of the technical report [20]. We also applied our truncation technique to the preemptive M/M/c priority queue with two priority classes but *without* abandonment, for which the analysis is known to be cumbersome when $c > 2$ [19]. For the preemptive M/M/3 queue with two priority classes, our 1-D step bounding function provides, on average, a 23% reduction in state space compared to Dayar et al.'s truncation.

VI. Conclusion

This paper presents a Lyapunov function-based technique to obtain tight truncation bounds with probability mass guarantees for multi-dimensional and infinite state-space CTMCs. By leveraging the known moments of the CTMC, our technique significantly truncates the state space by an average of around 34% and by as much as 72% (compared to Dayar et al. [6]) in the case of M/M/1-DPS model; The improvement in the M/M/3+M model is around 21% on average and by as much as 52%. Importantly, we prove that the truncation guarantees a user-specified bound on loss in probability mass, thus allowing for arbitrary accuracy guarantees.

Acknowledgment: This work was partially supported by NSF grant CNS 1750109.

References

[1] J. Dean and L. A. Barroso, "The tail at scale," *Communications of the ACM*, vol. 56, no. 2, pp. 74–80, 2013.

[2] G. DeCandia, D. Hastorun, M. Jampani, G. Kakulapati, A. Lakshman, A. Pilchin, S. Sivasubramanian, P. Vosshall, and W. Vogels, "Dynamo: Amazon's highly available key-value store," *ACM SIGOPS operating systems review*, vol. 41, no. 6, pp. 205–220, 2007.

[3] G. Latouche and V. Ramaswami, *Introduction to Matrix Analytic Methods in Stochastic Modeling*, 1999.

[4] L. Kleinrock, "Time-shared systems: A theoretical treatment," *Journal of the ACM (JACM)*, vol. 14, no. 2, pp. 242–261, 1967.

[5] R. R. Muntz, E. De Souza E Silva, and A. Goyal, "Bounding availability of repairable computer systems," *ACM SIGMETRICS Performance Evaluation Review*, vol. 17, no. 1, pp. 29–38, 1989.

[6] T. Dayar, W. Sandmann, D. Spieler, and V. Wolf, "Infinite level-dependent QBD processes and matrix-analytic solutions for stochastic chemical kinetics," *Advances in Applied Probability*, vol. 43, no. 4, pp. 1005–1026, 2011.

[7] R. E. A. C. Paley and A. Zygmund, "On some series of functions," *Math. Proc. of the Camb. Philos. Soc.*, pp. 337–357, 1932.

[8] P. Vis, "Performance analysis of multi-class queueing models," Ph.D. dissertation, Vrije Universiteit, Sep. 2017.

[9] K. M. Rege and B. Sengupta, "Queue-length distribution for the discriminatory processor-sharing queue," *Operations Research*, vol. 44, no. 4, pp. 653–657, 1996.

[10] G. Somashekar, M. Delasay, and A. Gandhi, "Tighter Lyapunov truncation for multi-dimensional continuous time markov chains with known moments," *ACM SIGMETRICS Performance Evaluation Review*, vol. 47, no. 2, pp. 33–35, 2019.

[11] S. Mahévas and G. Rubino, "Bound computation of dependability and performance measures," *IEEE Transactions on Computers*, vol. 50, no. 5, pp. 399–413, 2001.

[12] P. Buchholz, "Bounding stationary results of Tandem networks with MAP input and PH service time distributions," *ACM SIGMETRICS Performance Evaluation Review*, vol. 34, no. 1, pp. 191–202, 2006.

[13] L. Bright and P. G. Taylor, "Calculating the equilibrium distribution in level dependent quasi-birth-and-death processes," *Stochastic Models*, vol. 11, no. 3, pp. 497–525, 1995.

[14] D. Bertsimas, D. Gamarnik, and J. N. Tsitsiklis, "Geometric bounds for stationary distributions of infinite Markov chains via Lyapunov functions," MIT, Tech. Rep. WP 4027-98, 1998.

[15] S. T. Maguluri and R. Srikant, "Heavy traffic queue length behavior in a switch under the MaxWeight algorithm," *Stochastic Systems*, vol. 6, no. 1, pp. 211–250, 2016.

[16] B. Hajek, "Hitting-time and occupation-time bounds implied by drift analysis with applications," *Advances in Applied probability*, vol. 14, no. 3, pp. 502–525, 1982.

[17] M. Harchol-Balter, *Performance Modeling and Design of Computer Systems: Queueing Theory in Action*. Cambridge Univ. Press, 2013.

[18] E. Altman, K. Avrachenkov, and U. Ayesta, "A survey on discriminatory processor sharing," *Queueing systems*, vol. 53, no. 1, pp. 53–63, 2006.

[19] J. Wang, O. Baron, and A. Scheller-Wolf, "M/M/c queue with two priority classes," *Operations Research*, vol. 63, no. 3, pp. 733–749, 2015.

[20] G. Somashekar, M. Delasay, and A. Gandhi, "Truncating Multi-Dimensional Markov Chains with Accuracy Guarantee: Applications to Discriminatory Processor Sharing and Priority Queues," https://tr.cs.stonybrook.edu/tr/sbcs-tr-2022-17, Stony Brook University, Tech. Rep. SBCS-TR-2022-17, 2022.

[21] B. Kim and J. Kim, "Comparison of DPS and PS systems according to DPS weights," *IEEE communications Letters*, vol. 10, no. 7, pp. 558–560, 2006.

[22] O. J. Boxma, N. Hegde, and R. Núñez-Queija, "Exact and approximate analysis of sojourn times in finite discriminatory processor sharing queues," *AEU-International Journal of Electronics and Communications*, vol. 60, no. 2, pp. 109–115, 2006.

[23] G. De Veciana, T.-J. Lee, and T. Konstantopoulos, "Stability and performance analysis of networks supporting elastic services," *IEEE/ACM Transactions on Networking*, vol. 9, no. 1, pp. 2–14, 2001.

[24] R. Hassin and M. Haviv, *To queue or not to queue: Equilibrium behaviour in queuing systems*. Kluwer Academic Publishers, 2002.

[25] D. R. Miller, "Computation of steady-state probabilities for M/M/1 priority queues," *Operations Research*, vol. 29, no. 5, pp. 945–958, 1981.

[26] A. Mandelbaum and S. Zeltyn, "Service engineering in action: The Palm/Erlang-A queue, with applications to call centers," in *Advances in services innovations*. Springer, 2007, pp. 17–45.