Available online at www.sciencedirect.com

ScienceDirect

Electronic Notes in Theoretical Computer Science 310 (2015) 135–155

Electronic Notes in Theoretical Computer Science

www.elsevier.com/locate/entcs

Obtaining Optimal Thresholds for Processors with Speed-Scaling

Ronny J. Polansky[1],
Samyukta Sethuraman[1] and Natarajan Gautam[1]

Department of Industrial and Systems Engineering
Texas A&M University
College Station, Texas, USA

Abstract

In this research we consider a processor that can operate at multiple speeds and suggest a strategy for optimal speed-scaling. While higher speeds improve latency, they also draw a lot of power. Thus we adopt a threshold-based policy that uses higher speeds under higher workload conditions, and vice versa. However, it is unclear how to select "optimal" thresholds. For that we use a stochastic fluid-flow model with varying processing speeds based on fluid level.
First, given a set of thresholds, we develop an approach based on spectral expansion by modeling the evolution of the fluid queue as a semi-Markov process (SMP) and analyzing its performance. While there are techniques based on matrix-analytic methods and forward-backward decomposition, we show that they are not nearly as fast as the spectral-expansion SMP-based approach. Using the performance measures obtained from the SMP model, we suggest an algorithm for selecting the thresholds so that power consumption is minimized, while satisfying a quality-of-service constraint. We illustrate our results using a numerical example.

Keywords: server speed-scaling, data center, power management, quality of service, fluid model, spectral expansion

1 Introduction

The well-studied area of analysis of queues with state-dependent service rates has seen a renewed interest in the last decade or so. The major difference between traditional and contemporary queueing models is that in the former case the service rates were control-knobs while the latter uses workload processing speeds as control-knobs. In the latter case, each arriving entity brings a random amount of work which is an attribute of the entity, whereas the speed of processing (a deterministic, but variable quantity) of an entity's work is an attribute of the server. Of course, if the workload brought by each arriving entity is exponentially distributed, then selecting

[1] Email: ronny.polansky@gmail.com,{samyukta,gautam}@tamu.edu

http://dx.doi.org/10.1016/j.entcs.2014.12.016

service rates or processing speeds are mathematically equivalent. However, if the distribution is not exponential or if the entities are fluids, there is a need for new analysis.

Analysis of queues with variable workload processing speeds can be classified broadly into two lines of research. In one case, the workload processing speed is variable but not in a controllable fashion, while in the other case the service speed is controllable. On one hand, examples of the former case include Huang and Lee [9], and Mahabhashyam and Gautam [14] where the workload is exponentially distributed but since the workload processing speeds vary as an uncontrollable Markov modulated process, the resulting service time is not exponential. Such situations are especially applicable in modeling wireless channels and available CPU capacities in shared systems. On the other hand, there is a wide variety of application domains where the workload processing speed is controllable. For example in manufacturing systems one could control the cutting speed in lathes, conveyor belt speeds in tunnel freezers, and variable speed drives in motors. Likewise in computer systems, especially servers in data centers, one could use dynamic voltage/frequency scaling (DVFS) to process jobs at different speeds. This research study falls into the latter category.

In all the aforementioned examples, controlling the service speeds is fueled by the need to manage energy consumption. Typically, higher speeds imply higher power consumption but faster service completion. Thus there is a trade off between energy and quality of service. We wish to explore this trade off in this research and strike a balance. We begin by considering a policy that is widely accepted in the literature and use it to determine workload processing speeds based on buffer contents. In particular, we consider a monotonic and threshold-type policy: if there are $N+1$ service speeds $c_1, c_2, \ldots, c_{N+1}$ such that $0 < c_1 < c_2 < \ldots < c_{N+1}$, then there are N thresholds $x_1, x_2, \ldots, x_N$, where $0 = x_0 \leq x_1 \leq x_2 \leq \ldots \leq x_N < x_{N+1} = \infty$ such that if the amount of fluid in the system is between x_{i-1} and x_i for some $i \in [1, N+1]$, then speed c_i is used.

The goal of our study is to determine the optimal thresholds $x_1^*, x_2^*, \ldots, x_N^*$ so that the long-run average energy consumption per unit time (i.e. time-averaged power) is minimized subject to satisfying a quality of service metric that we will subsequently describe. Although there are several articles that describe structures of optimal policies, our study belongs to the minority that delve into methods to obtain the resulting numerical values of the optimal parameters (viz. thresholds). Our models and numerical examples are motivated by applications in data center servers (using technology such as DVFS) where it is crucial to reduce energy consumption (see McKinsey Report [16]). Our focus is on DVFS with the understanding that other techniques such as HVAC control, air-flow control, virtualization, cluster-sizing and chip cooling have all been implemented. Our goal is to further reduce energy consumption through DVFS which can be done in a distributed manner with local information. Since the load experienced by data center servers is stochastic and bursty in nature, we analyze data center server systems as fluid queues with varying processing speeds corresponding to a discrete set of DVFS levels.

We consider a stochastic fluid flow setting where fluid arrives into a buffer at piece-wise constant rates modulated by a continuous time Markov chain (CTMC). Such a CTMC environment process that drives the fluid entering rates is also called Markov modulated fluid process. These models are frequently referred to as stochastic fluid-flow models or first-order fluid queues where the flow rate is piece-wise constant for random times (pioneering work by Anick et al [3]). Fluid is removed from the buffer at one of $N+1$ different rates $c_1, \ldots, c_{N+1}$ based on thresholds $x_1, \ldots, x_N$ as described earlier. For a given set of thresholds we evaluate the long-run average costs per unit time and other performance metrics based on steady-state distribution of buffer contents. Using that we determine optimal thresholds $x_1^*, \ldots, x_N^*$. There are two techniques to obtain the buffer-content distribution: (1) spectral expansion, and (2) matrix analytic methods.

There are a significant number of articles that use spectral expansion (some of the earlier ones include Anick et al [3]), where the researchers have assumed a constant output capacity. Only a few articles consider varying output capacities, but most of them are to accommodate multi-class traffic. For example, Narayanan and Kulkarni [17] analyze a multi-class fluid model that uses a static-priority service policy and Agarwal et al [1] consider a threshold-based policy where the processing rate is shared between two classes of fluid based on the amount of content waiting to be served. However, they consider only two possible service capacities and the inputs are on-off CTMCs. Further, there are a few articles in multi-class fluid models that work around varying output capacities by considering compensating sources (see Elwalid and Mitra [7] and Kulkarni and Gautam [13]). Finally, Kankaya et al [10] use spectral expansion for fluid models with varying output capacity which is identical to the setting we consider here. Their solution method involves reduction to the Schur form for stability when the input and output rates are very close. However, calculation of optimal thresholds requires solving the fluid models repeatedly for different set of thresholds. Therefore, computational efficiency is of utmost importance and we will present arguments in favor of our method in that respect.

The second technique, namely matrix analytic methods, has received significant attention. Soohan and Ramaswami [2] consider matrix-analytic methods for transient analysis, steady state analysis and first passage times of both finite and infinite sized buffers. Da Silva Soares and Latouche [5] explore the relationship between matrix-analytic methods and that of discrete state quasi-birth-and-death process and present a derivation for the stationary distribution of a fluid queue with a finite buffer. Additionally, similar to [6], Da Silva Soares and Latouche [6] study a numerically stable counting-process based matrix-analytic method to arrive at the stationary distribution of the buffer contents for a fluid model with varying output capacity which is again identical to the setting we consider here. Although matrix analytic methods are appealing in terms of simplicity and compact notations, it is unclear how it compares against spectral expansion methods, computationally. We seek to explore this as it is critical to obtain a fast algorithm for performance analysis so that one could expeditiously search through the space to obtain the optimal

thresholds $x_1^*, \ldots, x_N^*$.

In Section 2 we formally define the problem and formulate an abstract model. In Section 3 we describe our model and approach for buffer content analysis given $x_1, \ldots, x_N$. We devote Section 4 for discussion of the steady-state performance measures. In Section 5 we describe an algorithm for obtaining the optimal thresholds. In Section 6 we describe our approach through a numerical example. Finally, in Section 7 we present concluding remarks followed by some directions for future work.

2 Problem Description

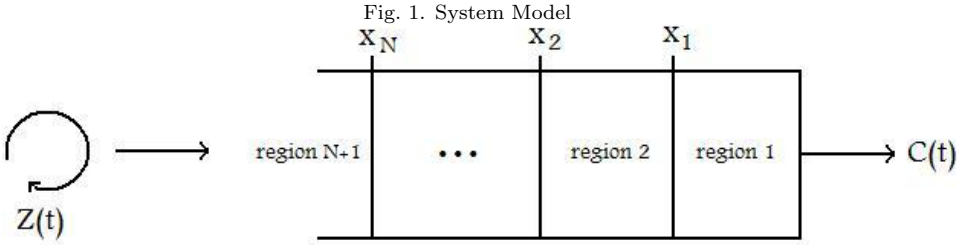
We consider a single infinite-sized buffer. Let $\{Z(t) : t \geq 0\}$ be the environment process that governs inputs to the buffer from a fictitious "source" such that $\{Z(t) : t \geq 0\}$ is a CTMC with finite state space $\mathcal{S}$ and infinitesimal generator matrix $\mathbf{Q} = [q_{uv}]_{u,v \in \mathcal{S}}$. We denote $r(Z(t))$ to be the rate at which fluid flows from the source into the buffer when the environment process is in state $Z(t)$ at time t. Let $C(t)$ be the drainage capacity of the buffer at time t. That means, if there is non-zero fluid in the buffer at time t, it gets removed from the buffer at rate $C(t)$. However, if the buffer is empty with $r(Z(t)) < C(t)$, then fluid is removed at rate $r(Z(t))$. The buffer can be emptied at one of $N+1$ different rates $c_1, c_2, \ldots, c_{N+1}$. Note that these correspond to the $N+1$ different speeds that servers in data centers can be run at using DVFS technology. We adopt a threshold policy with respect to the buffer contents to determine the rate at which fluid is drained from the buffer. There are N thresholds which we denote by $x_1, x_2, \ldots, x_N$ where $0 = x_0 \leq x_1 \leq x_2 \leq \ldots \leq x_N < x_{N+1} = \infty$. The N thresholds essentially partition the buffer into $N+1$ "regions". Throughout this article, we let $x_0 = 0$ and $x_{N+1} = \infty$, by convention. We define $X(t)$ to be the amount of fluid in the buffer at time t. Then $C(t)$ is indeed a piece-wise constant function of $X(t)$ and we define $C(t)$ for some $0 < c_1 < c_2 < \cdots < c_{N+1}$ by the following threshold policy:

$$C(t) = \begin{cases} c_1 & \text{if } x_0 < X(t) < x_1, \\ c_2 & \text{if } x_1 < X(t) < x_2, \\ \vdots & \vdots \\ c_N & \text{if } x_{N-1} < X(t) < x_N, \\ c_{N+1} & \text{if } X(t) > x_N. \end{cases}$$

A graphical representation of this system can be seen in Figure 1. By examining Figure 1, we can see that a threshold lies between each region. For example, threshold 1 lies between regions 1 and 2 within the buffer.

Remark 2.1 The setting is identical to that in Kankaya and Akar [10] and da Silva Soares and Latouche [6]. However, our approach is based on spectral expansion and semi-Markov process modeling while [10] uses spectral expansion and Schur form

Fig. 1. System Model



reduction and [6] uses matrix analytic methods. We first explain our approach in the next few sections and subsequently in Section 5 compare the approaches briefly.

Towards the end of this section we will make a case for the need for an extremely fast approach to compute the metrics when we use them in an optimization context. Thus, although it may be a matter of seconds for a single run, when one needs to run multiple cases, the other methods could take a possibly large amount of time. Before delving into further details, we present another remark.

Remark 2.2 Two main extensions to our setting have been considered by Malhotra et al [15] for $N = 1$ and O'Reilly [18] for any N: (i) a state-dependent arrival process, i.e. piece-wise constant arrival rates and environment processes are both state dependent; (ii) a hysteresis policy is used for switching between output capacities, i.e. when threshold x_{iu} is crossed from above, speed goes from c_i to c_{i-1} whereas while crossing x_{il} from below, speed goes from c_{i-1} to c_i (while we consider $x_{il} = x_{iu} = x_i$). However, it should be noted that extending our results to cases (i) and (ii) is not difficult. We do not consider them here because they are not applicable in our motivating application of data center servers. The arrival process is completely exogenous and independent in this case. Furthermore, since there is no cost for performing DVFS, the optimal policy is typically threshold-type (if there is a switching cost, then one would use a hysteretic policy).

In summary we have an infinite-sized buffer with piece-wise constant fluid input modulated by a CTMC and a piece-wise constant fluid output capacity that is state-dependent. Next we state some assumptions. For ease of exposition, we assume that $r(u)$ and c_i for every $u \in \mathcal{S}$ and $i = 1, 2, \ldots, N+1$ are chosen such that $r(u) - c_i \neq 0$. Although relaxing this assumption still maintains analytical tractability, it is notationally cumbersome. An important aspect that needs to be addressed in our system is what if there exist u and i such that $r(u) > c_i$ but $r(u) < c_{i+1}$? This would result in the buffer content process "bouncing back-and-forth" between region i and region $i+1$. To understand this, suppose that $\{X(t), t \geq 0\}$ is in region i so that $r(u) > c_i$. While $\{Z(t), t \geq 0\}$ remains in state u the buffer content process crosses threshold x_i from below and enters region $i+1$. At this point, if $\{Z(t), t \geq 0\}$ still remains in state u, then the buffer content process will instantaneously cross threshold x_i from above since $r(u) < c_{i+1}$. Once again, the buffer content process will instantaneously cross threshold x_i from below, repeating this process as long as the source modulating process remains in state u. In effect we would get $C(t) = r(Z(t))$ whenever $X(t) = x_i$, $Z(t) = u$, $r(u) > c_i$ and

$r(u) < c_{i+1}$. This is because the output capacities c_i and c_{i+1} would alternate for appropriate infinitesimal times so that the effective output capacity becomes $r(u)$. Thus $X(t)$ would have a mass at x_i.

Having described the scenario and assumptions we next explain the output we desire and subsequently the goal of this study. Given the definition of $X(t)$, the process $\{X(t), t \geq 0\}$ needs to be characterized and is typically called buffer content analysis. We assume the system is stable, i.e. $E[r(Z(\infty))] < c_{N+1}$ (see Kulkarni and Rolski [8] and da Silva Soares and Latouche [6]). For such a buffer, we seek to obtain the following steady-state measures (i.e. our output metrics):

$$p_i = \lim_{t\to\infty} P\{x_{i-1} < X(t) < x_i\} \text{ for all } i = 1, 2, \ldots, N+1,$$
$$\theta_i = \lim_{t\to\infty} P\{X(t) = x_i\} \text{ for all } i = 1, 2, \ldots, N, \text{ and}$$
$$\mathcal{O} = \lim_{t\to\infty} P\{X(t) > x\} \text{ for any } x > x_N.$$

Notice that p_i is the long-run fraction of time that the buffer content process $\{X(t), t \geq 0\}$ spends in region i for $i = 1, \ldots, N$. Note that since the last threshold is x_N, we let $p_{N+1} = \lim_{t\to\infty} P\{X(t) > x_N\}$. Similarly, θ_i is the long-run fraction of the time that the buffer content process spends on threshold i for $i = 1, \ldots, N$. When the buffer content process is in region i, the output capacity of the buffer is c_i. If the buffer content has mass at threshold i, the output capacity is assumed to be $r(u)$ where u is the state of the environment process $\{Z(t), t \geq 0\}$. Also notice that $\mathcal{O}$ is the steady-state probability of having more than x amount of fluid in the buffer. In this research, we are interested in $\mathcal{O}$ for some sufficiently large x. It is also important to realize that p_i, θ_i and $\mathcal{O}$ are functions of $x_1, x_2, \ldots, x_N$ which we are interested in determining in an optimal fashion (in Section 5).

Once we have expressions for p_i for all $i = 1, 2, \ldots, N+1$ and θ_i for all $i = 1, 2, \ldots, N$, as well as $\mathcal{O}$, our goal is to obtain optimal thresholds $x_1^*, \ldots, x_N^*$. For that we define the vector $\bar{\mathbf{x}} = [x_1 \ldots x_N]$ and explicitly write down our performance metrics as $p_i(\bar{\mathbf{x}})$ and $\theta_i(\bar{\mathbf{x}})$. Also note that $\mathcal{O}$ is a function of x and $\bar{\mathbf{x}}$, hence we say $\mathcal{O}(x, \bar{\mathbf{x}})$. For the purpose of clarity, we restate that x is a scalar denoting an element of the sample space of $X(t)$ while $\bar{\mathbf{x}}$ denotes a vector of thresholds. For the objective function we let $\mathscr{C}(\bar{\mathbf{x}})$ be the long-run average cost per unit time under threshold vector $\bar{\mathbf{x}}$. Assuming that we can write down $\mathscr{C}(\bar{\mathbf{x}})$ as a function of $c_1, \ldots, c_{N+1}$, $p_i(\bar{\mathbf{x}})$ and $\theta_i(\bar{\mathbf{x}})$, we state the following optimization problem for some given B and ϵ:

$$\begin{aligned} \text{Minimize} \quad & \mathscr{C}(\bar{\mathbf{x}}) \\ \text{subject to:} \quad & \mathcal{O}(B, \bar{\mathbf{x}}) \leq \epsilon \\ & x_0 = 0 \\ & x_{i-1} \leq x_i, \text{ for all } i = 1, 2, \ldots, N \\ & x_N \leq B \end{aligned}$$

which upon solving would give us the optimal thresholds $x_1^*, \ldots, x_N^*$.

Note that it is not possible to write down closed form algebraic expressions for $\mathscr{C}(\bar{\mathbf{x}})$ and $\mathcal{O}(B, \bar{\mathbf{x}})$ in terms of $x_1, \ldots, x_N$. Hence our algorithm to solve the optimization problem which we present in Section 5 would essentially search through the space of values that $\bar{\mathbf{x}}$ can take and compute $\mathscr{C}(\bar{\mathbf{x}})$ and $\mathcal{O}(B, \bar{\mathbf{x}})$ for every candidate $\bar{\mathbf{x}}$ value. Thus we need an approach that can compute $\mathscr{C}(\bar{\mathbf{x}})$ and $\mathcal{O}(B, \bar{\mathbf{x}})$ expeditiously. We will show in Section 5 that our method is faster than the approaches that use Schur form reduction or matrix analytic methods. However, we first explain our approach using spectral expansion. Given a set of thresholds $x_1, \ldots, x_N$, next we present an analysis of fluid queues to determine p_i, θ_i and $\mathcal{O}$ for all i (since $\bar{\mathbf{x}}$ is fixed, we remove the parenthesis for p_i, θ_i and $\mathcal{O}$).

3 Semi-Markov Process Model

In this section we focus on characterizing the $\{X(t), t \geq 0\}$ process. In particular, we use a semi-Markov process (SMP). We define the Markov regeneration epoch S_n to be the nth time the buffer content process $\{X(t), t \geq 0\}$ enters or leaves a *region* or the nth time at which the environment process $\{Z(t), t \geq 0\}$ changes from one state to another at a *threshold*. This means that a Markov regeneration epoch can occur when one of two possible events occur. The first event is when the buffer content process enters or leaves a region. The second event is when the buffer content process remains on a threshold and the environment process changes state. We now define Y_n that captures necessary information about the system at each Markov regeneration epoch S_n (assuming $S_0 = 0$) and encapsulates this information in two dimensions. We let

$$Y_n = (i, u) \text{ if } X(S_n) = x_i \text{ and } Z(S_n) = u \text{ for } i = 1, 2, \ldots, N \text{ and } u \in \mathcal{S}.$$

The first component of Y_n tells us which threshold the buffer content process $\{X(t), t \geq 0\}$ is crossing at the nth regeneration epoch. The second component of Y_n tells us the state of the source modulating process $\{Z(t), t \geq 0\}$ at the time of the nth regeneration epoch. Given this information along with values for c_i and $r(u)$, we can deduce which "direction" the threshold is being crossed from; above or below or whether it stays at a threshold. By this definition, we are able to take care of both types of epochs, i.e. entering or leaving a region or being stuck on a threshold.

Lemma 3.1 *The sequence* $\{(Y_n, S_n) : n \geq 0\}$ *is a Markov renewal sequence.*

Proof. The sequence of epochs $\{S_n, n \geq 0\}$ is nondecreasing in n and the conditional probability $P\{Y_{n+1} = (j, v), S_{n+1} - S_n \leq t | Y_n = (i, u), S_n, Y_{n-1}, S_{n-1}, \ldots, Y_0, S_0\} = P\{Y_1 = (j, v), S_1 \leq t | Y_0 = (i, u)\}$ because of the Markovian nature of both $X(t)$ and $Z(t)$. In particular, once we know the buffer contents at a given time and the environment, to predict the future we do not need anything from the past. Thus $\{(Y_n, S_n) : n \geq 0\}$ is a Markov renewal sequence. □

Lemma 3.2 *Let $N(t) = \sup\{n \geq 0 : S_n \leq t\}$. Thus, $N(t)$ is the number of times up to time t the buffer content process $\{X(t), t \geq 0\}$ enters or leaves a region or the environment process $\{Z(t), t \geq 0\}$ changes state at a threshold . Let $W(t) = Y_{N(t)}$ for $t \geq 0$. The stochastic process $\{W(t), t \geq 0\}$ is a semi-Markov process.*

Proof. By definition (see Definition 9.5. on pp. 510 in Kulkarni [11]), $\{W(t), t \geq 0\}$ is a semi-Markov process (SMP). □

The next step is to obtain the kernel of the Markov renewal sequence $\{(Y_n, S_n), n \geq 0\}$.

3.1 SMP Kernel

Since $\{W(t), t \geq 0\}$ is an SMP, we can obtain p_i and θ_i by using the kernel of the Markov renewal sequence $\{(Y_n, S_n), n \geq 0\}$. We denote the conditional probability $P\{Y_1 = (j, v), S_1 \leq t | Y_0 = (i, u)\}$ by $G_{(i,u),(j,v)}(t)$, so the kernel of the Markov renewal sequence is given by $\mathbf{G}(t) = [G_{(i,u),(j,v)}(t)]$.

Given values for c_i and $r(u)$, if $Y_n = (i, u)$, $r(u) - c_i > 0$ ($r(u) - c_i < 0$) and $r(u) - c_{i+1} > 0$ ($r(u) - c_{i+1} < 0$), then $\{X(t), t \geq 0\}$ is crossing the threshold x_i from below (above), while the environment process is in state u, corresponding to a positive (negative) drift. When $Y_n = (i, u)$, $r(u) - c_i > 0$ and $r(u) - c_{i+1} < 0$, the buffer content process $\{X(t), t \geq 0\}$ will "bounce back-and-forth" across threshold x_i.

Suppose, for example, that $X(S_n) = x_i$, $Z(S_n) = u$, and $Y_n = (i, u)$, where $r(u) - c_i < 0$ and $r(u) - c_{i+1} < 0$. Now for some $v \neq u$, if $r(v) - c_i < 0$ and $r(v) - c_{i+1} < 0$, then it is impossible to transition from $Y_n = (i, u)$ to $Y_{n+1} = (i, v)$ without first making a transition to some $Y_{n+1} = (i, v')$, such that $r(v') - c_i > 0$ for unique u, v, v'. Essentially, this means that if $X(t)$ crosses a threshold from above, then $X(t)$ must cross the same threshold from below before it can cross the threshold again from above. Similarly, it is impossible to transition from $Y_n = (i, u)$, where $r(u) - c_i > 0$, to some (i, v) with $r(v) - c_i > 0$ without first making a transition to some $Y_{n+1}(i, v')$, such that $r(v') - c_i < 0$ for unique u, v, v'. This means that if $X(t)$ crosses a threshold from below, then $X(t)$ must cross the same threshold from above before it can cross the threshold again from below. Additionally, if $X(t)$ crosses threshold i from below (above), then it is impossible to cross threshold $i + 2$ ($i - 2$) from below (above) without first crossing threshold $i + 1$ ($i - 1$) from below (above). Therefore, for the cases just mentioned, the corresponding elements of the kernel will be equal to 0.

Now we consider the case where the buffer content bounces back-and-forth across threshold x_i while the environment process remains in state u. For a given i, there are only three possible transitions that could occur. The first two possible transitions happen when the $\{Z(t), t \geq 0\}$ process changes to state v, the drift could be either positive or negative causing the buffer content to drift off of threshold x_i. The third possible transition is when the $\{Z(t), t \geq 0\}$ process changes to another zero drift state, v that will cause the buffer content to bounce back-and-forth across threshold x_i. Finally, $G_{(i,u),(i,u)}(t) = 0$ for all i and u.

Having listed the elements of $\mathbf{G}(t)$ that are equal to zero, now, the non-zero elements of the kernel need to be obtained, which we show next.

3.2 Modularization of the kernel

In this section, we demonstrate how to obtain the non-zero elements of the kernel. We take a modular approach to obtaining the elements of the kernel by considering each region and threshold of the buffer separately; and, after analyzing each region and threshold, we combine the results to obtain the kernel of the SMP.

3.2.1 Computing elements of the kernel for a threshold

Computing elements of the kernel for a given threshold is relatively straightforward. Here we consider the case where $Y_n = (i, u)$, such that $r(u) - c_i > 0$ and $r(u) - c_{i+1} < 0$, for all $i = 1, 2, \ldots, N$ and $u \in \mathcal{S}$. Recall that $G_{(i,u),(i,v)}(t) = P\{Y_1 = (i, v), S_1 \leq t | Y_0 = (i, u)\}$. This is the probability of making a transition to state (i, v) before time t given that the initial state is (i, u). This is equivalent to the probability that $\{Z(t), t \geq 0\}$ will be in state v when the Markov regeneration epoch S_1 occurs, given that $\{Z(t), t \geq 0\}$ is initially in state u. That is, $G_{(i,u),(i,v)}(t) = P\{Z(S_1) = v | Z(0) = u\}$. Since $\{Z(t), t \geq 0\}$ is a CTMC,

$$G_{(i,u),(i,v)}(t) = \frac{q_{uv}}{-q_{uu}} \left(1 - e^{\sum_{v \neq u} q_{uv} t}\right) \text{ if } r(u) - c_i > 0,\ r(u) - c_{i+1} < 0 \text{ and } v \neq u. \tag{1}$$

Thus, given $\mathbf{Q}$, we can obtain the elements of the kernel that correspond to being stuck on a threshold using Equation (1). Now we compute elements of the kernel for each region.

3.2.2 Computing elements of the kernel for a region

Now that we have shown how to compute the elements of the kernel that correspond to states where the buffer content process is stuck on a threshold, we show how to compute the remaining elements that correspond to the buffer content process drifting through a region. Although the dynamics of the buffer content process do not change according to which region the buffer content is in, there are some key differences between each type of region, which we now discuss. Essentially, there are three types of regions. The first type is that of region 1, which only has an upper threshold (hitting or remaining on the lower threshold in this region does not result in a Markov regeneration epoch). The second type of region is that of region $N + 1$, which only has a lower threshold. The third type of region is that of regions $2, \ldots, N$. Regions of this type have an upper *and* lower threshold. Ultimately, the type of region that we analyze will effect our boundary conditions, which will be discussed shortly. To obtain all remaining kernel elements, the steps that follow are carried out for each region individually, using a first passage time analysis. We leverage upon the results of Mahabhashyam et al [14] and similarly define a first passage time by $T = \inf_{t \in \mathbb{R}^+} \{t \geq 0 : X(t) = x_{i-1} \text{ or } X(t) = x_i\}$ where x_{i-1} and x_i are the lower and upper thresholds of the given region. Recall that we assume

$x_0 = 0$ and $x_{N+1} = \infty$. Now, we define $H_{uv}(x,t) = P\{T \leq t, Z(T) = v | X(0) = x, Z(0) = u\}$ for $x_i < x < x_{i+1}$ and $u, v \in \mathcal{S}$, and $\mathbf{H}(x,t) = [H_{uv}(x,t)]_{u,v\in\mathcal{S}}$. To compute the nonzero elements of the kernel for the given region, we need to obtain $\mathbf{H}(x,t)$. Actually, it turns out that we only need $\widetilde{\mathbf{H}}(x,w)$ which is the Laplace-Stieltjes Transform (LST) of $\mathbf{H}(x,t)$ with respect to t, and this will become apparent shortly. The dynamics (Mahabhashyam et al [14]) of $\mathbf{H}(x,t)$ are governed by the following partial differential equation

$$\frac{\partial \mathbf{H}(x,t)}{\partial t} - \mathbf{D}\frac{\partial \mathbf{H}(x,t)}{\partial x} = \mathbf{Q}\mathbf{H}(x,t) \tag{2}$$

where $\mathbf{D} = \mathbf{R} - c\mathbf{I}$. Here c is the output capacity for the given region while $\mathbf{I}$ is the identity matrix and $\mathbf{R} = diag\{r(1), r(2), \ldots, r(|\mathcal{S}|)\}$. Since each region has a different output capacity, states that have positive drift could be different from region to region and likewise for states with negative drift. The boundary conditions will change depending on which region we are analyzing.

Theorem 3.3 *The boundary conditions for region 1 are*

$$H_{uv}(x_1 - x_0, t) = 0 \text{ for } r(u) - c_1 < 0,\ r(v) - c_1 < 0, \tag{3}$$
$$H_{uu}(x_1 - x_0, t) = 1 \text{ for } r(u) - c_1 > 0, \tag{4}$$
$$H_{uv}(x_1 - x_0, t) = 0 \text{ for } r(u) - c_1 > 0,\ u \neq v, \tag{5}$$
$$\widetilde{H}_{uv}(0,w) = \sum_{v' \neq u} \widetilde{H}_{v'u}(0,w)\frac{q_{uv'}}{-q_{uu}}\left(\frac{-q_{uu}}{-q_{uu}+w}\right) \text{ for } r(u) - c_1 < 0,\ r(v) - c_1 > 0. \tag{6}$$

The boundary conditions for region i (for $i = 2, \ldots, N$) are

$$H_{uu}(0,t) = 1 \text{ for } r(u) - c_i < 0, \tag{7}$$
$$H_{uv}(0,t) = 0 \text{ for } r(u) - c_i < 0,\ u \neq v, \tag{8}$$
$$H_{uu}(x_i - x_{i-1}, t) = 1 \text{ for } r(u) - c_i > 0, \tag{9}$$
$$H_{uv}(x_i - x_{i-1}, t) = 0 \text{ for } r(u) - c_i > 0,\ u \neq v. \tag{10}$$

The boundary conditions for region $N+1$ are

$$H_{uu}(0,t) = 1 \text{ for } r(u) - c_{N+1} < 0, \tag{11}$$
$$H_{uv}(0,t) = 0 \text{ for } r(u) - c_{N+1} < 0,\ u \neq v. \tag{12}$$

Proof. Equation (3) is because the first-passage time cannot occur when the drift is negative (the only way a first-passage time will occur is when the buffer content process hits threshold x_1 from below, which cannot happen when the drift is negative). Equation (4) is because the first-passage time occurs instantaneously and the state of the source modulating process cannot change states instantaneously. Therefore, the probability that the first-passage time occurs before t and the source modulating process remains in the same state is one. Equation (5) is similar to the second boundary condition in that the first-passage time occurs instantaneously but the source cannot be in two different states when this occurs. Therefore, the probability that the first-passage time occurs before t and the source modulating process

is in two different states is zero. To derive Equation (6) which is in terms of the LST of $H_{uv}(0,t)$, where $r(u) - c_1 < 0$ and $r(v) - c_1 > 0$, notice that if the buffer content process reaches zero while the environment process is in state u, then the buffer content will remain at zero until the environment process changes to some state $v' \neq u$. Necessarily, state v' must be such that $r(v') < c_1$ so that the buffer content can actually reach zero. Thus, the first passage time will be equal to the amount of time spent in state u plus some remaining time from state v' until the first passage time starting in state v. So by conditioning on v' and then unconditioning, we get Equation (6).
Equation (7) is because if the buffer content process is initially at the lower threshold, the first-passage time will occur instantaneously since the drift is negative. Therefore, the probability that the first-passage time will occur before time t and the source modulating process remains in the same state is one. Equation (8) is similar to the first in that the first-passage time occurs instantaneously, but the source modulating process cannot be in two different states at the same time. Therefore, the probability that the first-passage time will occur before time t and the source modulating process is in two different states is zero. The same reasoning that was applied to Equations (4) and (5) also applies to Equations (9) and (10).
Equations (11) and (12) are derived in the same way as Equations (7) and (8). □

To solve the partial differential equations (PDE) in Equation (2), we take the LST across the PDE with respect to t and let $\mathbf{A}(w) = \mathbf{D}^{-1}(w\mathbf{I} - \mathbf{Q})$ to obtain the following ordinary differential equation (ODE):

$$\frac{d\widetilde{\mathbf{H}}(x,w)}{dx} = \mathbf{A}(w)\widetilde{\mathbf{H}}(x,w). \tag{13}$$

By using a spectral decomposition technique, the solution to Equation (13) can be obtained. For some $j \in \mathcal{S}$, a general solution to the ODE is

$$\widetilde{H}_{\cdot v} = \begin{pmatrix} \widetilde{H}_{1v} \\ \vdots \\ \widetilde{H}_{|\mathcal{S}|v} \end{pmatrix} = \sum_{u=1}^{|\mathcal{S}|} a_{uv}(w) e^{\lambda_u(w)x} \boldsymbol{\phi}_u(w)$$

where $a_{uv}(w)$ are unknown coefficients and $\lambda_u(w)$ and $\boldsymbol{\phi}_u(w)$ are, respectively, the eigenvalues and eigenvectors that are obtained by solving

$$\mathbf{D}\lambda_u(w)\boldsymbol{\phi}_u(w) = (w\mathbf{I} - \mathbf{Q})\boldsymbol{\phi}_u(w) \text{ for } u \in \mathcal{S}.$$

Remark 3.4 To obtain the unknown coefficients $a_{uv}(w)$ we use (i) the boundary conditions of the appropriate region through Theorem 3.3; and (ii) set $a_{uv}(w) = 0$ if $\lambda_u(w) > 0$ to ensure that $H_{uv}(x,t)$ will be a joint probability distribution as $x \to \infty$. This would result in the right number of equations to solve for the unknown coefficients $a_{uv}(w)$.

Solving the differential equation for each region actually yields the LST $\widetilde{\mathbf{H}}(x,w)$ but this is all that is needed for our analysis, which become apparent shortly. Hence we do not need to invert the LST at all. We now show how to use the results of this section to obtain the LST of the kernel $\widetilde{\mathbf{G}}(w)$.

3.2.3 Obtaining the kernel

In this section, we show how to combine the results from previous sections to obtain $\widetilde{\mathbf{G}}(w)$. Having solved Equation (13) for each region and using Equation (1), we now have all the elements of $\widetilde{\mathbf{G}}(w)$. Recall that

$$G_{(i,u),(j,v)}(t) = P\{Y_1 = (j,v), S_1 \leq t | Y_0 = (i,u)\} \text{ and}$$

$$H_{uv}(x,t) = P\{T \leq t, Z(T) = v | X(0) = x, Z(0) = u\}.$$

We let $\widetilde{\mathbf{H}}^i(x,w)$ be the matrix obtained from the first-passage time analysis of region i. With this in mind, we can determine the *nonzero* elements of $\widetilde{\mathbf{G}}(w)$ by

$$\widetilde{G}_{(i,u),(j,v)}(w) = \begin{cases} \widetilde{H}^{i+1}_{uv}(0,w) & \text{if } j = i+1,\ r(u) > c_{i+1} \text{ and } r(v) > c_{i+1}, \\ \widetilde{H}^{i}_{uv}(x_i - x_{i-1}, w) & \text{if } j = i-1,\ r(u) < c_i \text{ and } r(v) < c_i, \\ \widetilde{H}^{i+1}_{uv}(0,w) & \text{if } j = i,\ r(u) > c_{i+1} \text{ and } r(v) < c_{i+1}, \\ \widetilde{H}^{i}_{uv}(x_i - x_{i-1}, w) & \text{if } j = i,\ r(u) < c_i \text{ and } r(v) > c_i. \end{cases} \tag{14}$$

Note that $G_{(i,u),(j,v)}(t)$ is the conditional probability that a sojourn time will end at threshold x_j when the state of the source modulating process is in state v in a time less than t, given that the previous Markov regeneration epoch occurred at threshold x_i and the state of the source modulating process was initially state u. Notice that if $j = i+1$, then the buffer content process drifts from threshold x_i to threshold x_{i+1} while the state of the source modulating process is in state u initially, and in state j when the buffer content crosses threshold x_{i+1}. Therefore, $\widetilde{G}_{(i,u),(i+1,v)}(w)$ is equivalent to $\widetilde{H}^{i+1}_{uv}(0,w)$. In a similar fashion, the other cases follow. Having obtained the kernel, next we show how to compute the steady-state performance measures.

4 Obtaining the Steady-State Measures

Having obtained $\widetilde{\mathbf{G}}(w)$, we are in a position to compute the steady-state measures p_i and θ_i and subsequently $\mathcal{O}$. Recall that these measures are a function of $x_1, \ldots, x_N$. With that understanding, for ease of exposition we leave p_i, θ_i and $\mathcal{O}$ as they are and not write them as $p_i(\bar{\mathbf{x}})$, $\theta_i(\bar{\mathbf{x}})$ and $\mathcal{O}(x, \bar{\mathbf{x}})$ where $\bar{\mathbf{x}}$ is a vector containing the threshold values and for some arbitrary $x > x_{N+1}$.

4.1 Expressions for p_i and θ_i for all i

We first need to obtain the limiting probabilities for the states of the SMP $\{W(t), t \geq 0\}$ as well as their expected sojourn times. Let $\tau_{(i,u)}$ to be the expected sojourn time the SMP $\{W(t), t \geq 0\}$ spends in state (i,u) for $i = 1, 2, \ldots, N$ and $u \in \mathcal{S}$ and $\boldsymbol{\tau} = [\tau_{(1,1)} \ \tau_{(1,2)} \ \cdots \ \tau_{(N,|\mathcal{S}|)}]$. Thus,

$$\tau_{(i,u)} = - \sum_{(j,v) \in \{1,2,\ldots,N\} \times \mathcal{S}} \frac{d}{dw} \widetilde{G}_{(i,u),(j,v)}(w) \Big|_{w=0} \tag{15}$$

Notice that $\{Y_n, n \geq 0\}$ is a discrete-time Markov chain (DTMC) embedded in $\{W(t), t \geq 0\}$. Let $\boldsymbol{\pi} = [\pi_{(1,1)} \ \pi_{(1,2)} \ \cdots \ \pi_{(N,|\mathcal{S}|)}]$ be the steady-state transition probabilities of the embedded DTMC. Using the fact that $\boldsymbol{\pi} = \boldsymbol{\pi}\widetilde{\mathbf{G}}(0)$ (since $\boldsymbol{\pi} = \boldsymbol{\pi}\mathbf{G}(\infty)$) and $\sum_{i=1}^{N} \sum_{u \in \mathcal{S}} \pi_{(i,u)} = 1$, we can solve for $\boldsymbol{\pi}$. After computing the vectors $\boldsymbol{\pi}$ and $\boldsymbol{\tau}$, the steady-state measure p_i can be computed for $i = 1, 2, \ldots, N+1$ by

$$p_i = \frac{\sum_{(i,u) \in \mathcal{R}_i} \pi_{(i,u)} \tau_{(i,u)}}{\sum_{(i,u) \in \mathcal{T} \times \mathcal{S}} \pi_{(i,u)} \tau_{(i,u)}} \tag{16}$$

where $\mathcal{T}$ is an index set for the set of thresholds and $\mathcal{R}_i$ is an index set that determines an appropriate subset of states from the state space of $\{W(t), t \geq 0\}$. We let $\mathcal{R}_1 = \{(1,u) : r(u) - c_1 < 0, \ u \in \mathcal{S}\}$ which is the set of all states that drift through region 1. Similarly, we let $\mathcal{R}_{N+1} = \{(N+1,u) : r(u) - c_{N+1} > 0, \ u \in \mathcal{S}\}$ which is the set of all states that drift through region $N+1$. Finally, for $i = 2, \ldots, N$, we let $\mathcal{R}_i = \{(i-1,u) : r(u) - c_{i-1} > 0, \ u \in \mathcal{S}\} \cup \{(i,u) : r(u) - c_i < 0, \ u \in \mathcal{S}\}$. In this case, $\mathcal{R}_i$ is the set of all states that drift through region i for $i = 2, \ldots, N$. Similarly, the steady-state measure θ_i can be computed for $i = 1, 2, \ldots, N$ by

$$\theta_i = \frac{\sum_{(i,u) \in \mathcal{T}_i} \pi_{(i,u)} \tau_{(i,u)}}{\sum_{(i,u) \in \mathcal{T} \times \mathcal{S}} \pi_{(i,u)} \tau_{(i,u)}} \tag{17}$$

where $\mathcal{T}_i = \{(i,u) : r(u) - c_i > 0, \ r(u) - c_{i+1} < 0, \ u \in \mathcal{S}\}$ for $i = 1, 2, \ldots, N$. In other words, $\mathcal{T}_i$ is the set of states that get stuck on threshold i (with the understanding that $\theta_i = 0$ if $\mathcal{T}_i$ is the null set).

4.2 Computing the Tail Probability (O)

In this section we demonstrate how to compute the tail probability $\mathcal{O} = lim_{t \to \infty} P\{X(t) > x\}$ for some $x > x_N$. For that, let $\mathbf{D}_{N+1}$ be the diagonal drift matrix in region $N+1$. Therefore, $\mathbf{D}_{N+1} = \mathbf{R} - c_{N+1}\mathbf{I}$. Now, let $\gamma_1, \ldots, \gamma_{|S|}$ be the eigenvalues of $(\mathbf{D}_{N+1})^{-1}\mathbf{Q}$ and $\boldsymbol{\psi}_1, \ldots, \boldsymbol{\psi}_{|S|}$ be the corresponding *left* eigenvectors. Let M be the set of states with positive drifts, i.e. $M = \{u \in \mathcal{S} : r(u) > c_{N+1}\}$. Define constants ℓ_u for all $u \in \mathcal{S}$ that satisfy the following conditions:

$$\begin{aligned} \ell_u &= 0 && \text{if } Re(\gamma_u) > 0 \\ \ell_u &= 1/(\boldsymbol{\psi}_u \overline{\mathbf{1}}) && \text{if } \gamma_u = 0 \\ \sum_{v \in M \cup \{0\}} \ell_v \psi_v(u) &= 0 && \text{if } u \in M \end{aligned}$$

where $\overline{\mathbf{1}}$ is a column vector of ones and ℓ_0 is equal to the ℓ_u for which $\gamma_u = 0$ (likewise $\boldsymbol{\psi}_0$ as well). Thereby, the following theorem gives an expression for the tail probability $\mathcal{O}$.

Theorem 4.1 *For any given $x > x_N$, the tail probability $\mathcal{O}$ is given by*

$$\mathcal{O} = \frac{1 - \sum_{u=1}^{|\mathcal{S}|} \sum_{v=1}^{|\mathcal{S}|} \ell_v e^{\gamma_v (x - x_N)} \psi_v(u)}{1 - \sum_{u=1}^{|\mathcal{S}|} \sum_{v=1}^{|\mathcal{S}|} \ell_v \psi_v(u)} p_{N+1}$$

.

Proof. By conditioning on the region that the buffer content process is in we get

$$\begin{aligned} \mathcal{O} &= \lim_{t \to \infty} P\{X(t) > x\} \\ &= \lim_{t \to \infty} P\{X(t) > x | X(t) < x_1\} p_1 + \sum_{i=2}^{N} \lim_{t \to \infty} P\{X(t) > x | x_{i-1} < X(t) < x_i\} p_i \\ &\quad + \lim_{t \to \infty} P\{X(t) > x | X(t) > x_N\} p_{N+1} \\ &= \lim_{t \to \infty} P\{X(t) > x | X(t) > x_N\} p_{N+1} \end{aligned}$$

where the last equality comes from the fact that the buffer content process cannot simultaneously be less than x_N and greater than x_N. Notice that $P\{X(t) > x | X(t) > x_N\}$ is stochastically identical to $P\{\xi(t) > x - x_N | \xi(t) > 0\}$ where $\{\xi(t), t \geq 0\}$ corresponds to the amount of fluid in an infinite-sized buffer with constant output capacity of c_{N+1}. Therefore, we get

$$\begin{aligned} \mathcal{O} &= \lim_{t \to \infty} P\{\xi(t) > x - x_N | \xi(t) > 0\} p_{N+1} \\ &= \lim_{t \to \infty} \frac{P\{\xi(t) > x - x_N, \xi(t) > 0\}}{P\{\xi(t) > 0\}} p_{N+1} \\ &= \lim_{t \to \infty} \frac{P\{\xi(t) > x - x_N\}}{P\{\xi(t) > 0\}} p_{N+1} \end{aligned} \tag{18}$$

Thus, we need to compute $\lim_{t \to \infty} P\{\xi(t) > y\}$ and to do so, we leverage upon the results of a standard single-buffer model where the output capacity of the buffer is equal to c_{N+1} and the source modulating process is a CTMC with state space $\mathcal{S}$. Using the results from Kulkarni [12] we get

$$\lim_{t \to \infty} P\{\xi(t) > y\} = 1 - \sum_{u=1}^{|\mathcal{S}|} \sum_{v=1}^{|\mathcal{S}|} \ell_v e^{\gamma_v y} \psi_v(u).$$

Now substituting $x - x_N$ and 0 for u in the numerator and denominator in Equation

(18), we get

$$\mathcal{O} = \frac{1 - \sum_{u=1}^{|\mathcal{S}|} \sum_{v=1}^{|\mathcal{S}|} \ell_v e^{\gamma_v (x - x_N)} \psi_v(u)}{1 - \sum_{u=1}^{|\mathcal{S}|} \sum_{v=1}^{|\mathcal{S}|} \ell_v \psi_v(u)} p_{N+1}$$

□

5 Computing Optimal Thresholds

Given a set of thresholds $x_1, \ldots, x_N$, in Section 3 we outlined how to obtain the probabilities p_i for all $i \in \{1, \ldots, N+1\}$, θ_i for all $i \in \{1, \ldots, N\}$ and $\mathcal{O}$. In this section we devote our attention to obtaining optimal thresholds $x_1^*, \ldots, x_N^*$ by solving the optimization problem described in Section 2. We now describe an algorithm to search through the space of x_1, …, x_N values by computing the objective function and constraint using the method in Section 3 to obtain the optimal one.

5.1 Search Algorithm

To obtain the optimal thresholds x_1^*, …, x_N^*, we describe an iterative algorithm as follows:

(i) *Generate* a candidate vector of thresholds $\overline{\mathbf{x}} = [x_1 \ \ldots \ x_N]$ such that $0 \leq x_1 \leq \ldots \leq x_N \leq B$.

(ii) Compute $p_i(\overline{\mathbf{x}})$ for all $i \in \{1, \ldots, N+1\}$ and $\theta_i(\overline{\mathbf{x}})$ for all $i \in \{1, \ldots, N\}$ using Equations (16) and (17) respectively.

(iii) Letting $x = B$ in Theorem 4.1, compute $\mathcal{O}(B, \overline{\mathbf{x}})$ and check if $\mathcal{O}(B, \overline{\mathbf{x}}) \leq \epsilon$. If TRUE go to step 4, else to step 1 for a new $\overline{\mathbf{x}}$ and report current $\overline{\mathbf{x}}$ as infeasible.

(iv) Calculate $\mathscr{C}(\overline{\mathbf{x}})$ using

$$\mathscr{C}(\overline{\mathbf{x}}) = \sum_{i=1}^{N+1} \beta_i^R p_i(\overline{\mathbf{x}}) + \sum_{i=1}^{N} \beta_i^T \theta_i(\overline{\mathbf{x}})$$

where β_i^R is the cost per unit time incurred when speed c_i is used and β_i^T is the cost per unit time incurred toggling between speeds c_i and c_{i+1}.

(v) If $\mathscr{C}(\overline{\mathbf{x}})$ is the lowest cost then set the optimal threshold $\overline{\mathbf{x}}^* = [x_1^* \ \ldots \ x_N^*]$ as $\overline{\mathbf{x}}$.

(vi) If *stopping criterion* has not been attained go to step 1, else stop.

In the above algorithm, we need to determine an approach to generate candidates for the vector $\overline{\mathbf{x}}$ as well as a criterion to stop. A natural candidate for that is a metaheuristic such as simulated annealing, genetic algorithm, tabu search, ant colony optimization, evolutionary computations, etc. However, since the elements of $\overline{\mathbf{x}}$ are not discrete and there is an order $0 \leq x_1 \leq \ldots \leq x_N \leq B$, some of the metaheuristics may need to be modified suitably. Since this is not the focus of the article, we have not delved into methods to solve the optimization problem. One may be able to take advantage of the characteristics of the function to be optimized.

In a few examples we found that with respect to one variable, say x_i, the objective function and constraint function are convex or concave with respect to x_i. Thus it may be possible to obtain more efficient gradient descent type algorithms as well. On the other end of the spectrum, especially for small problem instances (such as the one in the next section) it would be possible to do a complete (discretized) enumeration. For example by dividing the segment from 0 to B into B/Δ discrete values that x_i can take by mapping into tiny segments of length Δ, one can completely enumerate by considering all $(B/\Delta)^N$ alternative locations for the N thresholds.

It is worthwhile observing that as an alternative approach to the one outlined in Section 3, one could use the results in [6] and [10] based on matrix-analytic methods and Schur approximations respectively and obtain p_i, θ_i and $\mathcal{O}$. Since we need to compute the metrics for several candidate threshold values in the above algorithm, it is crucial to select the approach that is faster.

Remark 5.1 Based on our experiments in MATLAB version R2013B, we observed that for one iteration of the algorithm described above, our method requires on an average, half or less of the time required by the methods in [6] and [10]. Even though, the theoretical complexity of the three methods is the same, $(O(N^3|S|^3))$, the difference in performance of the three methods arises from the fact that our method computes the p_i values directly. The other two methods result in the probability density function which has to be integrated over the regions to obtain the p_i values.

Now we present a numerical example for computing the optimal thresholds for a given input process.

6 Numerical Example

Having described an algorithm to compute the steady-state measures p_i and θ_i as well as an algorithm to obtain optimal threshold, next we demonstrate an application to energy consumption in data centers. A commonly used relationship between the output capacity and the energy consumed by a server is that of a cubic relationship (see for example Chen et al [4]). Using that, we let the energy cost per unit time associated with operating the server under output capacity c_i be $k_0+kc_i^3$ where k and k_0 are fixed constants for all $i \in \{1,\ldots,N+1\}$. However, the output capacity of the server will be c_i only when the buffer content process is in region i. Therefore, $\beta_i^R = k_0 + kc_i^3$ is equal to the energy cost incurred per unit time while the buffer content process is in region i. As previously stated, when the buffer content process gets stuck on a threshold, the buffer contents spend an infinitesimal amount of time in each region that surrounds the threshold bouncing back-and-forth between the two regions. Thus, we denote the energy cost associated with operating on threshold i (for all $i \in \{1,\ldots,N\}$) by $\beta_i^T = \sum_{j\in\mathcal{S}} \alpha_{ij}\left(k_0 + kc_i^3\right) + \sum_{j\in\mathcal{S}}(1-\alpha_{ij})\left(k_0 + kc_{i+1}^3\right)$ where $0 < \alpha_{ij} < 1$. In order to compute the energy cost associated with operating on threshold i while being in state j, we first consider which threshold the buffer content process is stuck on and the state of the environment process to compute α_{ij},

then we compute β_i^T. The output capacity $C(t)$ will be equal to $r(Z(t))$ whenever, say $X(t) = x_i$, $Z(t) = j$ and $c_j < r(i) < c_{j+1}$. Therefore, we define α_{ij} to be the solution to $r(j) = \alpha c_i + (1-\alpha)c_{i+1}$. With this in mind, we can compute the long-run average cost per unit time for running the server which we denote by $\mathscr{C}$. If we know the output capacity at a given time then we know how much cost we are incurring per unit time. Say, for example, that the output capacity is c_1 and we are in region 1, then we know that we are incurring a cost of β_1^R per unit time. Similarly, if the buffer content process is stuck on, say, threshold x_1 then we are incurring a cost of β_1^T per unit time. Therefore, it can be shown that, by conditioning on the SMP $\{W(t), t \geq 0\}$ in steady-state and unconditioning, the long-run expected cost per unit time is given by

$$\mathscr{C} = \sum_{i=1}^{N+1} p_i \beta_i^R + \sum_{i=1}^{N} \theta_i \beta_i^T \tag{19}$$

Further, using Equation (19) we can select the optimal set of thresholds $\overline{\mathbf{x}}^*$ by minimizing the long-run energy cost for running the server subject to $\mathcal{O}(B, \overline{\mathbf{x}}^*) \leq \epsilon$ and $0 \leq x_1^* \leq \ldots \leq x_N^* \leq B$. Now that we have shown how to compute the steady-state measures p_i and θ_i as well as the long-run expected cost for running the server $\mathscr{C}$, we present a numerical example.

For our numerical example to demonstrate how the above procedure can be used to compute the steady-state measures p_i and θ_i, and obtain the optimal set of thresholds $\overline{\mathbf{x}}^*$, we consider $N = 3$, i.e. a buffer with 3 thresholds, x_1, x_2, and x_3, which creates four regions. We first consider the case where x_1, x_2 and x_3 are fixed (subsequently we will optimize it). In particular, we let $x_i = i$ for $i = \{1, 2, 3\}$. This means that the distance between each threshold is 1. We let the buffer output capacity (in MB/sec) for region i, $c_i = i$ for $i = \{1, 2, 3, 4\}$. In this example, the environment process that governs fluid input into the buffer $\{Z(t), t \geq 0\}$ has state-space $\mathcal{S} = \{1, 2, 3, 4, 5, 6\}$ and the infinitesimal generator matrix $\mathbf{Q}$ is given by:

$$\mathbf{Q} = \begin{pmatrix} -11 & 2.5 & 3 & 1.5 & 2.5 & 1.5 \\ 3 & -12 & 1 & 3.5 & 1 & 3.5 \\ 1 & 1.5 & -10 & 2.5 & 3 & 2 \\ 5 & 4 & 3 & -15 & 1 & 2 \\ 1 & 3 & 1 & 4 & -14 & 5 \\ 5 & 3 & 4 & 1.5 & 1.5 & -15 \end{pmatrix}.$$

The rate at which fluid flows (in MB/sec) into the buffer for each state is given by the vector $[r(1)\ r(2)\ r(3)\ r(4)\ r(5)\ r(6)] = [0.7\ 0.9\ 2.2\ 3.7\ 4.2\ 8.0]$. We solve Equation (13) for each region to determine the nonzero elements of the kernel. We use Equation (1) to obtain elements of the kernel that correspond to states where the buffer content process bounces back-and-forth across a threshold. To obtain the

LST of the kernel we use Equation (14). In this example, there are two states that bounce back-and-forth across thresholds, namely states $(2,3)$ and $(3,4)$. To see this, consider the state $(2,3)$, i.e. threshold 2 and environment state 3. Fluid flows into the buffer at rate 2.2 MB/sec when $Z(t) = 3$ and the output capacity for region 2 is 2 MB/sec. This results in a positive drift of 0.2 MB/sec, however, in region 3, the output capacity is 3 MB/sec. This results in a negative drift of 0.8 MB/sec. As a result of this, the buffer content process bounces back-and-forth across threshold x_2 when the state of the environment process is $Z(t) = 3$. A similar situation occurs at threshold x_3 when $Z(t) = 4$. The next step is to compute the expected sojourn times that the SMP $\{W(t), t \geq 0\}$ spends in each state. Using Equation (15), we get the sojourn times found in Table 1.

Table 1
Expected Sojourn Times

(i,j)	$\tau_{(i,j)}$	(i,j)	$\tau_{(i,j)}$	(i,j)	$\tau_{(i,j)}$
(1,1)	0.1371	(2,1)	0.3287	(3,1)	0.5944
(1,2)	0.1227	(2,2)	0.2885	(3,2)	0.5409
(1,3)	0.5856	(2,3)	0.1000	(3,3)	0.4380
(1,4)	0.5395	(2,4)	0.2450	(3,4)	0.0667
(1,5)	0.6400	(2,5)	0.4266	(3,5)	0.2284
(1,6)	0.5951	(2,6)	0.4710	(3,6)	0.3636

Recall that $\boldsymbol{\pi}$ is the vector of steady-state transition probabilities of the embedded DTMC. In this example $\boldsymbol{\pi} = [\pi_{(1,1)}\ \pi_{(1,2)} \cdots \pi_{(3,6)}]$ and using the fact that $\boldsymbol{\pi} = \boldsymbol{\pi}\widetilde{G}(0)$ and $\sum_{(i,j)} \pi_{(i,j)} = 1$ we can compute the steady-state transition probabilities which are shown in Table 2.

Table 2
Transition Probabilities

(i,j)	$\pi_{(i,j)}$	(i,j)	$\pi_{(i,j)}$	(i,j)	$\pi_{(i,j)}$
(1,1)	0.0021	(2,1)	0.1289	(3,1)	0.0851
(1,2)	0.0014	(2,2)	0.1108	(3,2)	0.0710
(1,3)	0.0008	(2,3)	0.0540	(3,3)	0.0506
(1,4)	0.0010	(2,4)	0.0712	(3,4)	0.0320
(1,5)	0.0008	(2,5)	0.0711	(3,5)	0.0330
(1,6)	0.0010	(2,6)	0.1267	(3,6)	0.1585

At this point, we have everything needed to compute the steady-state measure p_i and θ_i. Using Equations (16) and (17) we compute values for p_i and θ_i which can be found in Table 3.

Table 3
Steady-State Measures

i	p_i	θ_i
1	0.0013	0
2	0.2075	0.0147
3	0.5937	0.0058
4	0.1771	-

From Table 3, we can see that the buffer content process spends approximately 60% of the time in region 3. Notice that $\theta_1 = 0$. This is because there is no state so that the buffer content process can bounce back-and-forth across threshold x_1. Also, there is no θ_4 since there are only three thresholds. After computing the steady-state measures p_i and θ_i, we can now use Equation (19) to compute the long-run average cost per unit time for running the server. For example, letting $k_0 = 10$ and $k = 1.5$, and using Equation (19) along with values from Table 3 results in a long-run expected cost of 53.63. However, if we always run the server at the highest speed, the long-run expected cost per unit time would be 106 since $k_0 + kc_4^3 = 10 + 1.5 \cdot 4^3$. Therefore, we have significantly reduced the long-run expected cost by approximately 51%.

So far in this section we have considered threshold values $x_1 = 1$, $x_2 = 2$ and $x_3 = 3$. Now we turn our attention to obtaining optimal thresholds x_1^*, x_2^* and x_3^*. For that we use $B = 10$ MB and $\epsilon = 0.0001$ to find the smallest average cost per unit time given in Equation (19) so that $\mathcal{O}(B, x_1, x_2, x_3)$ computed by letting $x = B$ in Theorem 4.1 is such that $\mathcal{O}(B, x_1^*, x_2^*, x_3^*) \leq \epsilon$ and $0 \leq x_1^* \leq \ldots \leq x_N^* \leq B$. We generate new instances of x_1, x_2 and x_3 described in the algorithm in Section 5 using complete enumeration with a step size of 0.5, and the stopping criteria is when all the points are enumerated. The algorithm resulted in an optimal solution of $x_1^* = 0.5$, $x_2^* = 1.0$ and $x_3^* = 7.5$ (all in MB).

7 Concluding Remarks

In this article, we present the computation of steady-state performance measures for an infinite-sized fluid buffer with multiple thresholds and state-dependent output capacity and consequently an algorithm to come up with the optimal set of thresholds. In our model, the N thresholds partition the buffer into $N + 1$ regions where each region has a given output capacity, while the environment process that controls the rate at which fluid flows into the buffer is a continuous-time Markov chain. When the environment process is in state u, fluid flows into the buffer at rate $r(u)$. We model the system as a semi-Markov process with the state of the system represented by the region in which the buffer fluid level is at any point of time and the state of the source modulating process. We show how to compute the steady-state performance measures p_i, which is the long-run fraction of time the

buffer content process spends in region i and θ_i, which is the long-run fraction of time the buffer content process spends on threshold x_i.

It is also possible to obtain p_i and θ_i using other methods (see [10] and [6]). However, we found that our method using semi-Markov process modeling and spectral methods is much faster. In particular, while obtaining optimal thresholds x_1^*, ..., x_N^*, we certainly need an extremely fast way to obtain the metrics. In this article we also obtained the probability of exceeding a buffer content level B in steady state. We used those to formulate and solve an optimization problem to minimize energy cost per unit time subject to quality of service (in terms of fluid level in the buffer) constraints. We suggested simple algorithms such as complete enumeration and meta-heuristics for this purpose. In the future we will consider other methods that leverage upon the structure and relationship between the threshold values and the costs. In addition, we seek to model an entire data center as a network of fluid buffers so that we can determine an optimal routing strategy for k classes of fluid customers in an attempt to further minimize energy consumption in data centers. Our work continues in this direction.

Acknowledgment

This material is based upon work partially supported by NSF under grant CMMI-0946935 and the AFOSR under Contract No. FA9550-13-1-0008.

References

[1] V. Aggarwal, N. Gautam, S. R. T. Kumara, and M. Greaves. Stochastic fluid flow models for determining optimal switching thresholds with an application to agent task scheduling. *Performance Evaluation*, 59(1):19–46, 2004.

[2] S. Ahn and V. Ramaswami. Efficient algorithms for transient analysis of stochastic fluid flow models. *Journal of Applied Probability*, 42(2):531–549, 2005.

[3] D. Anick, D. Mitra, and M. M. Sondhi. Stochastic theory of a data handling system with multiple sources. *Bell System Technical Journal*, 61:1871–1894, 1982.

[4] Y. Chen, A. Das, W. Qin, A. Sivasubramaniam, Q. Wang, and N. Gautam. Managing server energy and operational costs in hosting centers. *SIGMETRICS Perform. Eval. Rev.*, 33:303–314, June 2005.

[5] A. da Silva Soares and G. Latouche. Matrix-analytic methods for fluid queues with finite buffers. *Performance Evaluation*, 63:295–314, May 2006.

[6] A. da Silva Soares and G. Latouche. Fluid queues with level dependent evolution. *European Journal of Operational Research*, 196(3):1041–1048, August 2009.

[7] A. I. Elwalid and D. Mitra. Analysis, approximations and admission control of a multi-service multiplexing system with priorities. In *Proc. IEEE INFOCOM*, volume 2, pages 463–472, Apr 1995.

[8] N. Gautam, V. G. Kulkarni, Z. Palmowski, and T. Rolski. Bounds for fluid models driven by semi-markov inputs. *Probability in Engineering and Informational Sciences*, 13(4):429–475, 1999.

[9] L. Huang and T.T. Lee. Generalized pollaczek-khinchin formula for markov channels. *IEEE Transactions on Communications*, 61(8):3530–3540, August 2013.

[10] H. E. Kankaya and N. Akar. Solving multi-regime feedback fluid queues. *Stochastic Models*, 24(3):425–450, 2008.

[11] V. G. Kulkarni. *Modeling and analysis of stochastic systems*. CRC Press, 1996.

[12] V. G. Kulkarni. Fluid models for single buffer systems. *Frontiers in queueing: Models and applications in science and engineering*, 321:338, 1997.

[13] V. G. Kulkarni and N. Gautam. Admission control of multi-class traffic with service priorities in high speed networks. *Queueing systems: Theory and Applications*, 27(1-2):79–97, 1997.

[14] S. R. Mahabhashyam, N. Gautam, and S. R. T. Kumara. Resource-sharing queueing systems with fluid-flow traffic. *Operations Research*, 56:728–744, May 2008.

[15] R. Malhotra, M. Mandjes, W. Scheinhardt, and J. van den Berg. A feedback fluid queue with two congestion control thresholds. *Mathematical Methods of Operations Research*, 70:149–169, 2009.

[16] McKinsey & Company. *Revolutionizing Data Center Efficiency.* Uptime Institute Symposium, http://uptimeinstitute.org/content/view/168/57, 2008.

[17] A. Narayanan and V. G. Kulkarni. First passage times in fluid models with an application to two-priority fluid systems. In *IEEE International Computer Performance and Dependability Symposium*, 1996.

[18] M. OReilly. Multi-stage stochastic fluid models for congestion control. *European Journal of Operations Research (submitted)*, 2013.