This article has been accepted for publication in IEEE Transactions on Services Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TSC.2024.3354202

Scaling Power Management in Cloud Data Centers: A Multi-Level Continuous-Time MDP Approach

Behzad Chitsaz, Ahmad Khonsari, Masoumeh Moradian, Aresh Dadlani, *Senior Member, IEEE,* and Mohammad Sadegh Talebi

***Abstract*—Power management in multi-server data centers especially at scale is a vital issue of increasing importance in cloud computing paradigm. Existing studies mostly consider thresholds on the number of idle servers to switch the servers on or off and suffer from scalability issues. As a natural approach in view of the Markovian assumption, we present a multi-level continuous-time Markov decision process (CTMDP) model based on state aggregation of multi-server data centers with setup times that interestingly overcomes the inherent intractability of traditional MDP approaches due to their colossal state-action space. The beauty of the presented model is that, while it keeps loyalty to the Markovian behavior, it approximates the calculation of the transition probabilities in a way that keeps the accuracy of the results at a desirable level. Moreover, near-optimal performance is attained at the expense of the increased state-space dimensionality by tuning the number of levels in the multi-level approach. The simulation results were promising and confirm that in many scenarios of interest, the proposed approach attains noticeable improvements, namely a near 50% reduction in the size of CTMDP while yielding better rewards as compared to existing fixed threshold-based policies and aggregation methods.**

***Index Terms*—Cloud data centers, power management, Markov decision process, setup time, state aggregation.**

I. Introduction

The inconceivable global surge in computing power demand from video streaming, cryptocurrencies, power-hungry artificial intelligence applications, and numerous cloud-connected devices has changed the operational landscape of data centers. Serving as indispensable powerhouses of the modern digital era, a large portion of the expenditure is dedicated to cooling data center servers and equipment [1]. Projected statistics on the enormous power consumption of data centers reveal a harsh reality in spite of committing to more efficient technologies [2]. In general, a server is said to be *on* when busy serving jobs. In absence of job requests, a server either remains *idle* or is turned *off*. While idle servers are ubiquitous in data centers, each amount for about 50 to 60 percent of the energy of its fully utilized state [3]. Energy-aware cloud data centers minimize the power waste of idle servers by either switching them to a low-power *standby* state or the inactive *off* state. In practice, however, turning the server back to the active state (physical or virtual machine (VM)) incurs extra power consumption and transition delay, known as *setup* or *spin-up* time, which hinders immediate service to incoming job requests. Over-provisioning servers with jobs adds to the energy costs while under-provisioning may result in delayed service delivery time, thus violating the service-level agreement (SLA). In order to shorten the service delay while saving energy, it is therefore necessary to determine the optimal number of idle and setup servers in cloud data centers under different loads.

B. Chitsaz and A. Khonsari are with the School of Electrical and Computer Engineering, College of Engineering, University of Tehran, Iran. A. Khonsari is also with the School of Computer Science, Institute for Research in Fundamental Sciences, Tehran, Iran (e-mail: {b.chitsaz, a_khonsari}@ut.ac.ir).

M. Moradian is with the School of Computer Engineering, K. N. Toosi University of Technology, Tehran, Iran, and also with the School of Computer Science, Institute for Research in Fundamental Sciences, Tehran, Iran (e-mail: mmoradian@kntu.ac.ir).

A. Dadlani is with the Department of Computing Science, University of Alberta, Edmonton, Canada, and also with the Department of Electrical and Computer Engineering, Nazarbayev University, Astana, Kazakhstan (e-mail: aresh@ualberta.ca).

M. S. Talebi is with the Department of Computer Science, University of Copenhagen, Copenhagen, Denmark (e-mail: m.shahi@di.ku.dk).

A. State-of-the-Art and Prior Work

The attention drawn towards the theoretical assessment of energy management in multi-server systems has grown significantly over recent years. In [4], a game-theoretic approach is proposed for the workload management in geographically distributed data centers considering the data transfer costs and queueing delay. In [5], a dynamic VM consolidation algorithm (EQ-DVMCA) is proposed which strikes the balance between the energy consumption and quality of service (QoS) in cloud data centers and provides efficient consolidation of virtual resources. In [6], a hysteresis queuing model is presented to minimize power costs in cloud systems without explicitly considering server setup delay. The mean power consumption in systems with exponential setup time is studied in [7] wherein various operational policies such as the ON/IDLE policy that turns no server off, the ON/OFF policy that turns off all servers immediately after becoming idle and has no limit on the number of setup servers, the ON/OFF/STAG policy that allows at most one server to be in setup at any point of time, and finally, the ON/OFF/k STAG policy that permits at most k servers to be in setup are investigated. In [8], the authors analyzed the k-staggered policy that permits some servers to remain idle after setup using a three-dimensional continuous-time Markov chain (CTMC). The power consumption and waiting time distributions for the ON/OFF policy have been studied in [9]. The work in [10] focused on the system queue length distribution and considered added policies that turn off servers with a finite delay, and also permit servers to go into the sleep mode which, compared to the off mode, induces lower setup time and power usage. In [11], the authors defined two priority queues for different levels of delay-sensitivity and in the case of peak loads, the jobs with lower priority were deferred to promote QoS. However, all these policies employ static or fixed-threshold approaches to manage the energy consumption of servers.

By partitioning the homogeneous physical machines into three pools (*hot*, *warm*, and *cold*) with different power levels, the scal-

This article has been accepted for publication in IEEE Transactions on Services Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TSC.2024.3354202

able model introduced in [12] used interacting Markov chains and fixed-point iteration to derive the mean waiting time and power consumption. In [13] and [14], the authors introduced heterogeneity in the CPU cores requested by each VM, where the numbers of cores conform to uniform and general distributions, respectively. General spin-up time distribution was investigated in [15] and the mean performance measures and energy consumption of the switching policies were modeled using the $M/G/1$ queuing discipline. Though insightful, none of the above efforts explicitly prioritize delay over power consumption.

The Markov decision process (MDP) framework [16] offers a powerful mathematical approach to deriving optimal power-switching strategies in multi-server systems; see, e.g., [8], [17]–[20]. In [17], a MDP model is used to minimize energy costs and rejected jobs in an Infrastructure-as-a-Service (IaaS) cloud system. A near-optimal solution is proposed in [18] for power switching of two dynamic servers to minimize power consumption, delay, and wear-and-tear costs using MDP and look-ahead approach. The MDP approach is also used in [8] to find the optimal policy for cost-performance trade-off in virtualized data centers, where VMs are modeled by process sharing queues. Moreover, the size of the proposed MDP is reduced using fixed thresholds to categorize the load of servers into three light, moderate, and high levels. In [19], an optimal job routing policy is proposed based on the Whittle index in a system of parallel servers, where each server follows an ON/OFF policy and is equipped with an infinite buffer. The authors in [20] conducted a detailed study on the performance of different multi-server power management policies. Denoting the job arrival and service rates as λ and μ, respectively, they showed that keeping $\lambda/\mu + \sqrt{\lambda/\mu}$ servers always on results in a near-optimal solution.

Despite their potential and competence in modeling resource management in multi-server systems, the MDP approaches suffer from curse of dimensionality in the case of hyperscale cloud data centers. More precisely, the associated state-action spaces could grow very large even for a moderate number of servers and queue sizes, thus making standard solution methods (e.g., value iteration [16]) for solving MDPs intractable. To overcome the curse of dimensionality, *state aggregation* and *state abstraction* methods have been widely studied in the literature (e.g., [21]–[24], [24]–[26]), which aim to derive smaller MDPs via merging states that are similar in terms of, e.g., model parameters (transition probabilities and rewards), value functions, etc. So far, a plethora of state aggregation and abstraction methods have been presented and analyzed in the literature in both discounted (e.g., [24], [25]) and undiscounted (e.g., [26]) settings. However, these methods often consider generic MDPs and are thus oblivious to the structure of MDPs arising in data canter problems. To our best knowledge, there exists no reported work on state aggregation for optimal energy management in large-scale data centers that incorporates the intrinsic structure of the underlying system.

Finally, it is worth mentioning that some studies investigate resource management in data centers using reinforcement learning (RL) approaches [27]–[31]. These approaches were used to deal with uncertainty in system parameters or to combat the curse of dimensionality due to large state-action spaces. For instance, the DeepEE optimization framework in [31] utilizes deep RL for jointly optimizing energy consumption in task scheduling and cooling control within a data center. In [28], a deep RL-based allocation algorithm in the presence of long-lasting and compute-intensive jobs is derived. Methods to minimize energy costs associated with performing tasks with deadlines on data center servers have been investigated in [29], which involves scheduling the tasks during periods of low energy costs. Lastly, the authors of [30] introduce an online resource management framework, termed energy budgeting, specifically designed for virtualized data centers. While use of RL approaches allows one to deal with unknown system parameters, it should be stressed that derived resource management policies in these works often fail to admit performance guarantees, and could be far from optimal in terms of rewards.

B. Main Contributions

In this paper, a multi-server system is considered where the servers experience setup delays and the jobs arrive according to a Poisson process. The Poisson process of arrivals, as supported by [32], not only enables effective system modeling through CTMDP but also serves as a reliable approximation for job arrivals in data centers. In the proposed model, a power manager is responsible for turning the servers on or off to manage the power usage of the entire system or the number of waiting jobs. We aim to find an approximately optimal power-switching policy in such a system that not only minimizes the weighted sum of power consumption of the servers and average delay of the jobs, but also characterizes the trade-off between the dimensionality of the system state space and closeness to the optimal performance. To this end, our main contributions are:

- We introduce a *continuous-time MDP* (CTMDP) formulation of the system, called the *basic CTMDP*, which facilitates the derivation of an optimal power-switching policy in our system model, and can be of independent interest as an accurate model for resource allocation in cloud data centers.
- The basic CTMDP suffers from the curse of dimensionality. By employing state aggregation, we propose an efficient approximate CTMDP, referred to as *multi-level CTMDP*, to reduce the size of state-action space of the basic CTMDP. However, unlike the classical state aggregation approaches, we leverage the intrinsic structure in the basic CTMDP to perform aggregation more efficiently. Though increasing the number of levels in the multi-level CTMDP yields higher dimensionality, the performance of the resulting policy becomes closer to the optimal policy derived from the basic CTMDP model.
- The proposed multi-level CTMDP is benchmarked against the uniform state aggregation method and fixed-threshold policies in [20] under different settings with precedence of delay over power. We show the better performance of our model in terms of the achieved expected average rewards. Our simulation results confirm that in many scenarios of interest, the proposed approach attains a near 50% reduction in the size of CTMDP while yielding better rewards as compared to conventional methods.

The rest of this paper is structured as follows. Section II outlines the system model and assumptions. Section III details the basic CTMDP as the optimal solution, followed by the proposed

This article has been accepted for publication in IEEE Transactions on Services Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TSC.2024.3354202

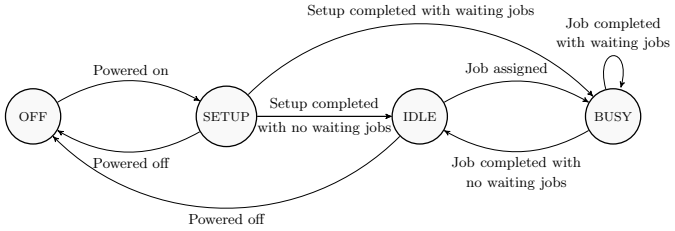

Fig. 1. Power state transition diagram of a data center server.

state-aggregated multi-level CTMDP in Section IV. The transition rate function and reward function of the multi-level CTMDP are derived in Section V. Numerical results are discussed in Section VI. Finally, Section VII concludes the paper.

II. System Model

Consider a data center comprising C servers that serve jobs arriving at the system and a power manager that switches the servers on and off independently. When on, a server is in one of the three states: BUSY, IDLE, or SETUP. Likewise, a server is in the OFF state if powered off by the manager. The Poisson process has been shown to be an acceptable approximation for job arrivals in data centers [32]. Thus, we assume job arrivals follow a Poisson process with rate λ. Furthermore, the service time of each job is exponentially distributed with rate μ. A server can process a job immediately only if it is in the IDLE state. Jobs that fail to receive service instantly upon arrival wait in a finite queue of capacity Q until served in a first-in-first-out (FIFO) manner. The state transition diagram of a server is shown in Fig. 1.

When powered on, the server enters the SETUP state and stays there for some amount of time called the *spin-up* time, which follows an exponential distribution with rate $\gamma > 0$. Upon completion of the setup process or the service process of a SETUP or BUSY server, respectively, the server transitions to the BUSY state if there is a head-of-line (HoL) job in the queue or enters the IDLE state, if otherwise. Finally, an IDLE server either becomes BUSY if a job is assigned to it or is switched off by the power manager. Note that the power manager can only turn off servers that are either in IDLE or SETUP states. Once entering the queue, a newly arrived job is served immediately only if no other job exists in the queue and an IDLE server is available. In other words, the manager assigns the HoL job to an IDLE server as soon as it is made available. This happens whenever a SETUP server finishes its setup process or a BUSY server finishes its service time.

Although having more IDLE and SETUP servers results in higher power consumption, it also leads to fewer waiting jobs and more immediate services. In this regard, the average number of IDLE and SETUP servers can serve as an indicator of the *power penalty* of the system, while the average number of waiting jobs can be interpreted as the *performance penalty*. Our aim is to determine a power-switching policy, governed by the power manager, that minimizes a weighted sum of the power and performance penalties.

III. Basic CTMDP Formulation

The model presented in Section II constitutes a dynamical system, where its state evolves as a Markov process, due to the memoryless property of the assumed arrival, setup, and service processes. Moreover, the decisions are made at continuous time instants when the system state changes. Hence, we may characterize the decision process as a CTMDP [16], [33]. An infinite-horizon CTMDP M under the average-reward criterion is a tuple $M = (\mathcal{S}, \mathcal{A}, q, r)$, where $\mathcal{S}$ denotes the (finite) state space, and where $\mathcal{A} = (\mathcal{A}_s)_{s\in\mathcal{S}}$ denotes the (finite) action space with $\mathcal{A}_s$ defining the set of actions available at state s. Further, q denotes the transition rate function such that $q(s'|s,a)$ is the transition rate to state $s' \in \mathcal{S}$ when executing action $a \in \mathcal{A}_s$ in state $s \in \mathcal{S}$. Finally, r denotes the reward function such that $r(s,a)$ is the reward obtained when selecting action $a \in \mathcal{A}_s$ in state $s \in \mathcal{S}$. The various components of the CTMDP M corresponding to the model in Section II will be specified below.

The State Space $\mathcal{S}$. We define the system state as $s \triangleq (b,i)$, where $b \in \{0,\dots,C\}$ denotes the number of BUSY servers, whereas $i \in \{-Q, -Q+1, \dots, C\}$ represents the number of IDLE servers if $i \geq 0$, or the *negated* number of waiting jobs if $i < 0$. Hence, $|i|$ indicates the number of IDLE servers or waiting jobs, and its sign determines whether we have IDLE server(s) or waiting job(s). Hereafter, we call b the *B-component* of s with 'B' signifying 'BUSY', whereas we refer to i as the *I/W-component*, where 'I' and 'W' stands for 'IDLE' and 'waiting', respectively.

The Action Space $\mathcal{A}$. At any state, the manager decides on the number of servers put in the SETUP and IDLE modes at the instants of job arrival, job completion, and setup completion. Concretely, at $s=(b,i) \in \mathcal{S}$, taking action $a \in \mathcal{A}_s$ corresponds to having a SETUP servers in s if $a \geq 0$, or to turning off $|a|$ IDLE servers if $a < 0$. In other words, when $a \geq 0$, the manager sets exactly a servers in the SETUP mode. To this end, if the number of current SETUP servers at the decision time exceeds a, then the extra servers are powered off. Otherwise, some servers are turned on to have a SETUP servers. We distinguish between two cases depending on the sign of i. When $i \geq 0$, there are i IDLE servers, and hence at most i servers can be turned off so that $a \geq -i$. Moreover, a maximum of $C-b-i$ servers could be in the SETUP mode. Thus, $\mathcal{A}_s = \{-i, \dots, C-b-i\}$. On the other hand, when $i < 0$, there is no IDLE server (thus, $a \geq 0$) and the manager can only manage the number of SETUP servers, which can be at most $C-b$. Hence, $\mathcal{A}_s = \{0,\dots,C-b\}$. In general, $\mathcal{A}_s = \{-i^+, \dots, C-b-i^+\}$, where $i^+ \triangleq \max(i,0)$. As shown in [20, Theorem 3], the optimal policy always turns on servers following a bulk setup policy; when it decides to turn on some servers, i.e., to put them in the SETUP mode, it turns on all OFF servers. Hence, we have $\mathcal{A}_s = \{-i^+, \dots, 0\} \cup \{C-b-i^+\}$.

The Transition Function q. Consider $s = (b,i) \in \mathcal{S}$, $s' = (b',i') \in \mathcal{S}$, and $a \in \mathcal{A}_s$. To determine $q(s'|s,a)$, we consider two cases based on the sign of a. For $a < 0$, an IDLE server must exist, implying $i > 0$. Since $|a|$ represents the number of IDLE servers to be turned off, we have $i + a \geq 0$. Hence,

$$q(s'|s,a) = \begin{cases} \lambda; & b' = b+1,\ i' = i+a-1,\ i+a > 0, \\ \lambda; & b' = b,\ i' = -1,\ i+a = 0, \\ b\mu; & b' = b-1,\ i' = i+a+1,\ b > 0. \end{cases} \tag{1}$$

To verify (1), note that upon arrival of a new job, the number of remaining IDLE servers decreases by one after taking action a (if $i+a>0$) and hence, the number of BUSY servers increases by one. If $i+a=0$, then $i'=-1$ implying that the arriving job waits

This article has been accepted for publication in IEEE Transactions on Services Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TSC.2024.3354202

in the queue. Finally, completion of a job with rate $b\mu$ increases and decreases the number of IDLE and BUSY servers by one, respectively. When $a \geq 0$, where a total of a servers will be in the SETUP mode, we have:

$$q(s'|s,a) = \begin{cases} \lambda; & b'=b+1,\ i'=i-1,\ i>0, \\ \lambda; & b'=b,\ i'=i-1,\ -Q<i\leq 0, \\ \lambda; & b'=b,\ i'=i,\ i=-Q, \\ b\mu; & b'=b-1,\ i'=i+1,\ i\geq 0, \\ b\mu; & b'=b,\ i'=i+1,\ i<0, \\ a\gamma; & b'=b,\ i'=i+1,\ i\geq 0, \\ a\gamma; & b'=b+1,\ i'=i+1,\ i<0. \end{cases} \quad (2)$$

The first case in (2) is verified by noting that when $i \geq 0$, then upon arrival of a new job, the number of IDLE and BUSY servers decreases and increases by one, respectively. However, in the second case where $i \leq 0$, the number of BUSY servers remains unchanged, but the number of waiting jobs increases by one. Specifically, we have $i' = i - 1$ under the condition $-Q < i$ (i.e., the queue is not full). Otherwise, if $i = -Q$, then the number of waiting jobs remains unchanged and the newly arriving job is dropped. In the fourth and fifth cases of (2), a BUSY server transitions to an IDLE state at rate $b\mu$, while the last two cases signify that a SETUP server becomes IDLE at a rate of $a\gamma$. In either case, the new IDLE server remains IDLE if $i \geq 0$, and transitions back to BUSY to serve a waiting job if otherwise. Note that the packet arrival at state $b = C$ is included in the second and third cases. As such, when $b = C$, we have $i \leq 0$ and the arrived packet awaits in the queue if it is not full.

The Reward Function r**.** Let $\Psi(s,a)$ denote sum of output transition rates at state $s = (b,i)$ under action $a \in \mathcal{A}_s$: $\Psi(s,a) = a^+\gamma + b\mu + \lambda$. The quantity $1/\Psi(s,a)$ indicates the average time spent at state s under action a. Observe that $|i^-|$ (with $x^- \triangleq \min(x,0)$ for any x) is linked to the performance penalty, whereas the power penalty may be defined using a weighted sum of the number of IDLE servers, i^+, and the number of SETUP servers, a^+. In order to define a reward function in line with the objective discussed in Section II, we define:

$$r(s,a) = \frac{-1}{\Psi(s,a)}\left(c_{\text{perf}}\,|i^-| + c_{\text{power}} i^+ + c'_{\text{power}} a^+\right), \quad (3)$$

where $c_{\text{perf}}, c_{\text{power}}$, and c'_{power} are application-specific positive numbers. In fact, $r(s,a)$ in (3) is a weighted sum of performance and power penalties scaled by the average dwell time at (s,a). Evidently, the ratio $c_{\text{power}}/c'_{\text{power}}$ controls the relative emphasis put on the power consumption of IDLE and SETUP modes. It is worth remarking that, firstly, we do not model the cost of job dropouts in the reward function since, due to the precedence of performance penalty over power penalty, the optimal policy never results in considerable job dropout probabilities, especially in the presence of a finite but sufficiently large queue size. Secondly, the analytical approach considered here is suited for moderate and light traffics, where the jobs do not occupy all servers almost surely and thus, the implementation of power management policies is justified.

Undertaking the approach in Chapter 11 of [16], the CTMDP M can be transformed into a discrete-time average-reward MDP and can be solved using standard methods such as policy iteration, value iteration, or linear programming. The following lemma presents the time complexity of solving the associated MDP to M.

Lemma 1. *The (per-step) time complexity of solving the MDP associated to M using value iteration is $O(C^2(Q+C^2)(Q+C))$.*

Proof. The time complexity of an iteration in the value iteration algorithm is $O(|\mathcal{S}|\sum_{s\in\mathcal{S}}|\mathcal{A}_s|)$ [16]. Introduce $\mathcal{S}_+ := \{(b,i) \in \mathcal{S}, i > 0\}$ and $\mathcal{S}_- := \{(b,i) \in \mathcal{S}, i \leq 0\}$. Then, $\mathcal{S}_+$ and $\mathcal{S}_-$ define a partition of $\mathcal{S}$ so that $|\mathcal{S}| = |\mathcal{S}_+| + |\mathcal{S}_-|$. Furthermore, $\sum_{s\in\mathcal{S}}|\mathcal{A}_s| = \sum_{s\in\mathcal{S}_+}|\mathcal{A}_s| + \sum_{s\in\mathcal{S}_-}|\mathcal{A}_s|$. We consider two cases:

- *When $s = (b,i) \in \mathcal{S}_-$:* Recalling that $i \in \{-Q,\dots,0\}$ and $b \in \{0,\dots,C\}$, we have $|\mathcal{S}_-| = (Q+1)(C+1)$. Furthermore, $\mathcal{A}_s = \{0, a = C-b-i\}$ so that $\sum_{s\in\mathcal{S}_-}|\mathcal{A}_s| = 2(Q+1)(C+1)$.
- *When $s = (b,i) \in \mathcal{S}_+$:* Recalling that $0 \leq b \leq C-i$, we have $|\mathcal{S}_+| = \sum_{i=1}^{C}(C-i+1) = C(C+1)/2$. Also, $\mathcal{A}_s = \{-i,\dots,0\} \cup \{C-b\}$, and thus, $|\mathcal{A}_s| = i+2$. Consequently, $\sum_{s\in\mathcal{S}_+}|\mathcal{A}_s| = \sum_{i=1}^{C}(C+1-i)(i+2) = C(C+1)(C/6+4/3)$.

Putting them together, we obtain $|\mathcal{S}|\sum_{s\in\mathcal{S}}|\mathcal{A}_s| = (C+1)^2(Q+C/2+1)(C^2/6+4C/3+2Q+2) = O(C^2(Q+C^2)(Q+C))$, thus concluding the proof. □

IV. APPROXIMATION VIA STATE AGGREGATION: MULTI-LEVEL CTMDP

We focus on large-scale data centers, where C represents a significant number. For instance, in December 2014, Amazon Web Services operated approximately $C = 1.4$ million servers across 28 availability zones. According to Lemma 1, computing an optimal policy in the exact model M for such data centers incurs a cost of $O(C^5)$, even for moderate values of Q, which is unfeasibly large. We remedy this issue by introducing a more manageable approximation of M with a reduced state-action space, known as the multi-level CTMDP.

A. Assumptions and Approximation

We first need to introduce the queuing approximation for our system captured by the following assumptions:

Assumption 1. *The number of servers, C, is large enough to serve all arriving jobs.*

Assumption 1 is justified since the power-switching policy is used for systems with low to moderate traffic. In other words, each job finds at least one OFF, IDLE, or SETUP server upon its arrival, and the probability of all servers being BUSY is infinitesimal. Otherwise, the power manager would need to keep the servers permanently powered on to accommodate as many jobs as possible, in which case implementing the power-switching policy would be meaningless.

Assumption 2. *The optimal policy always turns on at least $|i|$ servers, i.e., $a \geq |i|$ for states with waiting jobs ($i < 0$).*

This assumption bears resemblance to the k-staggered policy discussed in [20]. Thus, for any state $s = (b,i)$ with $i < 0$, the action space $\mathcal{A}_s = \{a | -i < a < C-b\}$, ensuring that for any $s =$

This article has been accepted for publication in IEEE Transactions on Services Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TSC.2024.3354202

$(b,i) \in \mathcal{S}, \mathcal{A}_s = \{a | -i \leq a \leq C-b-i^+\}$. For now, let us assume that the setup delay is zero. In such a case, given that turning the servers on and off does not induce any costs, the optimal power-switching policy turns the servers off upon being IDLE since they can become immediately IDLE again when needed. Then, considering Assumptions 1 and 2, the proposed queueing system can be assumed to have infinite number of servers, and due to exponential inter-arrival and service times, it can be modeled as an $M/M/\infty$ queue.

In an $M/M/\infty$ queue, the number of BUSY servers in the steady state follows the Poisson distribution with rate $\rho = \lambda/\mu$. On the other hand, in realistic scenarios, with non-zero setup delays, some jobs which do not find an IDLE server upon their arrival incur a delay before joining a server, which depends on the system state and the taken action. However, according to Assumption 2, the waiting time never exceeds a setup delay (in a stochastic sense). In this regard, in the real system, we stick to the approximation that the number of BUSY servers in the steady state, denoted by b, follows the Poisson distribution with rate $\rho = \lambda/\mu$. This assumption helps us to analytically derive the transition probabilities of the proposed multi-level CTMDP, for which our numerical results show that the resulting optimal policy outperforms previously studied counterparts. Besides, such an approximation becomes more accurate when the implemented policy results in smaller waiting time for the jobs by having more SETUP and IDLE servers. Finally, since ρ is large enough in our problem (e.g., $\rho > 10$), the Poisson distribution behaves similarly to a normal distribution with mean ρ and standard deviation $\sqrt{\rho}$. The derivation of the multi-level CTMDP in the following subsections relies on the symmetry property of the normal distribution.

B. The Multi-level CTMDP: Construction

We now formally introduce the multi-level CTMDP. Given the CTMDP $M = (\mathcal{S}, \mathcal{A}, q, r)$ introduced earlier, we denote its corresponding multi-level CTMDP by $M^{\text{ml}} = (\mathcal{S}^{\text{ml}}, \mathcal{A}^{\text{ml}}, q^{\text{ml}}, r^{\text{ml}})$, where '$\text{ml}$' signifies *'multi-level'*. The multi-level CTMDP M^{ml} is created by aggregating states from M, where each state in M^{ml} represents a combination of multiple states from M. A detailed construction of state and action in M^{ml} follows.

The State Space $\mathcal{S}^{\text{ml}}$. We consider aggregation by partitioning the set of feasible B- and I/W-components via suitably defined intervals. Therefore, it is natural to define the state in M^{ml} as $S = (B, I)$, where B and I are related to *the aggregated* B-components and I/W-components, respectively. More specifically, B (resp. I) refers to the *index* of the interval to which the B-component (resp. I/W-component) of its aggregated states belongs. We consider L levels for each of the B and I/W components, for some suitably chosen natural number L. To define B, we partition the set of available servers $\{0, \ldots, C\}$ into L disjoint subsets as follows:

$$\{0, 1, \ldots, U_1 - 1\}, \{U_1, \ldots, U_2 - 1\}, \ldots, \{U_{L-1}, \ldots, C\},$$

where $U_1 < U_2 < \ldots < U_{L-1} < C$ are real numbers that will be determined momentarily. The B-component of any state $s = (b,i) \in \mathcal{S}$ belongs to one of the aforementioned sets. Let us index the sets with $B \in \{0, 1, \ldots, L-1\}$. Then, B corresponds to the states whose B-components belong to $\{U_B, \ldots, U_{B+1} - 1\}$. In other words, B corresponds to states in $\big\{s = (b,i) : b \in \{U_B, \ldots, U_{B+1} - 1\}\big\}$. For simplicity, we consider sets of equal size. That is to say, $K_{\text{B}} \triangleq U_{B+1} - U_B$ does not depend on B. Hence, to determine $U_1, \ldots, U_{L-1}$ (and thus, B), it suffices to determine K_{B}. We shall refer to K_{B} as BUSY *level size*. We use the $(1-\epsilon)$-confidence interval[1] for some small enough ϵ (e.g., $\epsilon = 0.01$) to set K_{B}. More precisely, recalling that the B-component in our model denotes the number of BUSY servers, it follows a Normal distribution as explained in Section IV-A. Hence, the above confidence interval is derived as $\big[F^{-1}\big(\frac{\epsilon}{2}; \rho\big), F^{-1}\big(1 - \frac{\epsilon}{2}; \rho\big)\big]$, where $F^{-1}(\cdot; \rho)$ is the inverse of $F(\cdot; \rho)$, and $F(\cdot; \rho)$ denotes the CDF of the Poisson distribution with rate $\rho = \lambda/\mu$. Now, the BUSY level size K_{B} is defined as:

$$K_{\text{B}} = -\left\lceil \Big(F^{-1}\big(\tfrac{\epsilon}{2}; \rho\big) - F^{-1}\big(1 - \tfrac{\epsilon}{2}; \rho\big) \Big) / L \right\rceil , \qquad (4)$$

where $\lceil \cdot \rceil$ is the ceiling function ensuring an integer level size. Further, we define $\underline{\beta} = \big\lfloor \rho - \frac{1}{2} K_{\text{B}} L \big\rfloor^+$ and $\overline{\beta} = \big\lceil \rho + \frac{1}{2} K_{\text{B}} L \big\rceil$, where $\lfloor \cdot \rfloor$ denotes the floor function. Hence, the endpoints $U_1, \ldots, U_{L-1}$ are obtained as[2] $U_B = BK_{\text{B}} + \underline{\beta}$, for $B = 1, \ldots, L-1$. The construction above implies that all values of $b < \underline{\beta}$ are allocated to the level $B = 0$, whereas those with $b > \overline{\beta}$ are allocated to the level $B = L-1$.

To define the I/W-component I, we undertake a similar approach to partition $\{-Q, \ldots, C\}$. Specifically, we partition the positive part (i.e., $\{0, \ldots, C\}$) into L subsets, each of size $K_{\text{I}} = C/L$.[3] Then, we partition the negative part (i.e., $\{-Q, \ldots, -1\}$) into levels of size K_{I}. Thus, the negative part of I/W-component has $\lceil Q/K_{\text{I}} \rceil$ levels so that $I \in \{-\lceil Q/K_{\text{I}} \rceil, \ldots, L-1\}$, where the level I aggregates states with $i \in \{IK_{\text{I}}+1, \ldots, (I+1)K_{\text{I}}\}$.

The Action Space $\mathcal{A}^{\text{ml}}$. We define action space in M^{ml} by aggregating actions in M. Let $\mathcal{A}^{\text{ml}}_S$ denote the set of actions available at $S = (B, I)$ in M^{ml}. Following a similar construction as states to define $\mathcal{A}^{\text{ml}}_S$ using a level size K_{I}, we define:

$$\mathcal{A}^{\text{ml}}_S = \big\{-I, \ldots, 0\big\} \cup \big\{ \big(C - U_B - I^+ K_{\text{I}}\big) / K_{\text{I}} \big\} ,$$

where $C - U_B - I^+ K_{\text{I}}$ is the number of available OFF servers, assuming U_B servers are BUSY and $I^+ K_{\text{I}}$ servers are IDLE. When $A > 0$, AK_{I} servers are powered on to be in the SETUP state; else, all SETUP and AK_{I} IDLE servers are turned off.

As mentioned earlier, the trade-off between applied precision and dimensionality can be adjusted by choosing different values for L. In particular, a larger L yields a larger (aggregated) state space while ensuring that the optimal policies in M and M^{ml} become closer in terms of expected average reward.

V. TRANSITION AND REWARD FUNCTIONS OF M

This section is devoted to deriving the transition rate function q^{ml} and reward function r^{ml} of the multi-level CTMDP M^{ml}. In view of the construction of M^{ml}, this task entails calculating the transition rates between various levels. In order to make the presentation more tractable, we begin with calculating a few key quantities that prove instrumental in formulating q^{ml} and r^{ml}.

[1] $(1-\epsilon)$-confidence interval is the interval to which the B-component belongs with probability greater than $1 - \epsilon$.

[2] Here, we use the symmetry of the Normal distribution of the B-component around ρ to define the new lower and upper endpoints of the confidence interval.

[3] For simplicity, we use the same number of partitions as for B-component.

This article has been accepted for publication in IEEE Transactions on Services Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TSC.2024.3354202

A. Preliminaries: Level Boundary Probabilities

The transition between levels occurs when the I/W-component or B-component (in M) are at the boundaries of their corresponding levels. Therefore, we need to derive the probability of being at the boundaries of BUSY and I/W levels.

We recall that $F(\cdot;\rho)$ denotes the CDF of the Poisson distribution with rate $\rho = \lambda/\mu$, which is in fact the distribution of the number of BUSY servers as a result of the assumption of Poisson arrivals; see Section IV-A. Further, let $f(\cdot;\rho)$ denote the probability mass function (pmf) of $F(\cdot;\rho)$. For brevity, we omit the dependence of F and f on ρ as it is fixed throughout.

Considering $s=(b,i)$, we observe that the probability that $b \in \{U_B,\dots,U_{B+1}-1\}$ (i.e., the number of BUSY servers belongs to the B-th level) is $F\big(U_{B+1}-1\big)-F\big(U_B-1\big)$. Define

$$\underline{p}(B) = \frac{f(U_B)}{F(U_{B+1}-1)-F(U_B-1)}, \tag{5}$$

$$\overline{p}(B) = \frac{f(U_{B+1}-1)}{F(U_{B+1}-1)-F(U_B-1)}. \tag{6}$$

It is evident that $\underline{p}(B)$ in (5) (resp. $\overline{p}(B)$ in (6)) is the probability that b coincides with the lower (resp. upper) boundary of the BUSY level B in M^{ml}. Furthermore, the average number of BUSY servers at level B, denoted by N_B, is:

$$N_B = \frac{1}{F\big(U_{B+1}-1\big)-F\big(U_B-1\big)} \sum_{x=U_B}^{U_{B+1}-1} x f(x)\,. \tag{7}$$

For the I/W-component, we are interested in computing the probability that M is at the lower (resp. upper) boundary of I/W level I conditioned on the event that M^{ml} is in state S and action A is chosen. These probabilities are denoted by $\underline{u}(S,A)$ and $\overline{u}(S,A)$, respectively, and derived in the following lemma.

Lemma 2. *Let* $\eta(S,A) = (\mu N_B + A^+ K_{\text{I}}\gamma)/\lambda$. *If* $\eta(S,A)\neq 1$,

$$\underline{u}(S,A) = \frac{1-\eta(S,A)}{1-\eta(S,A)^{K_{\text{I}}}}\,,\quad \overline{u}(S,A) = \eta(S,A)^{K_{\text{I}}-1}\underline{u}(S,A).$$

Furthermore, $\underline{u}(S,A) = \overline{u}(S,A) = K_{\text{I}}^{-1}$ *when* $\eta(S,A)=1$.

Proof. Let $S=(B,I)\in\mathcal{S}^{\text{ml}}$ and $A\in\mathcal{A}^{\text{ml}}$. The proof relies on constructing an approximate birth-death (BD) process associated to (S,A). Recall that by construction, for each $IK_{\text{I}} \le i \le (I+1)K_{\text{I}}-1$, we aggregate all states $s=(b,i)\in\mathcal{S}$ with $U_B \le b \le U_{B+1}-1$ into one state, called *meta-state* i. Fig. 2 and Fig. 3 show the transition probabilities of the states in the multi-level state $S=(B,I)$ when $I>0$ and $I\le 0$, respectively. In fact, we aggregate all states within the red dotted boxes in these two figures into one state to obtain K_{I} meta-states. Also, the arrival (resp. departure) rate of each meta-state is equal to the average of the arrival (resp. departure) rates of the corresponding aggregated states. When deriving these average rates, we ignore all transitions from other multi-level states into $S=(B,I)$ and vice versa. These transitions, in fact, belong to the boundary states $(b,i)\in\{U_B,\dots,U_{B+1}-1\}\times\{IK_{\text{I}},\dots,(I+1)K_{\text{I}}-1\}$. With this approximation, the average transition rate from each meta-state i to the next state $i+1$ becomes $\gamma A^+ K_{\text{I}} + N_B$, while that from each meta-state i to the previous state $i-1$ is equal to λ. Since these birth and death rates are the same in all meta-states, the resulting approximated Markov chain forms a BD process with the birth and death rates equal to $\gamma A^+ K_{\text{I}} + N_B$ and λ, respectively. Let $P(i|S,A)$ denote the probability that the I/W-component (in M) is i given that M^{ml} is in $S\in\mathcal{S}^{\text{ml}}$ and $A\in\mathcal{A}^{\text{ml}}_S$ is taken. Then, in the BD process, we have:

$$\lambda P(i+1|S,A) = \big(\mu N_B + \gamma A^+ K_{\text{I}}\big)\, P(i|S,A)\,.$$

so that $P(i+1|S,A) = \eta P(i|S,A)$, where for brevity we omit the dependence of η on (S,A).

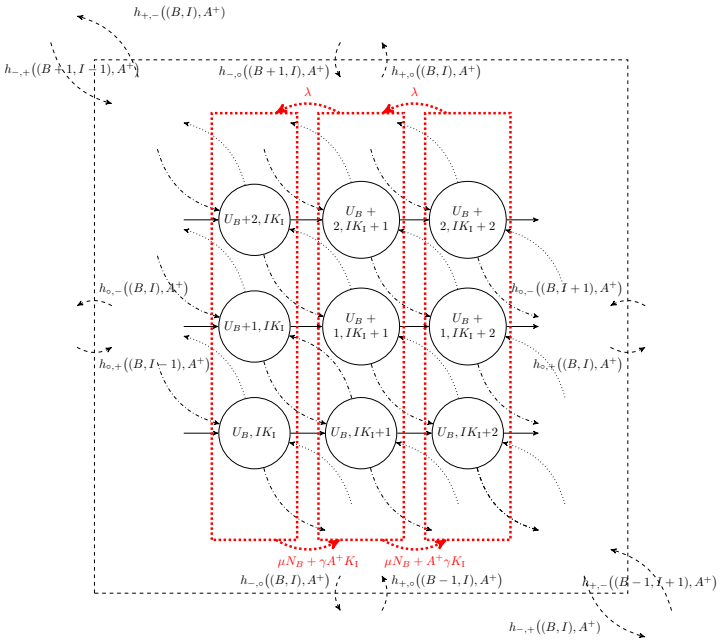

Fig. 2. Transition diagram of a multi-level state (B,I) when $I>0$ and $K_{\text{B}}=K_{\text{I}}=3$. Dotted transitions correspond to a job arrival with rate λ, solid transitions correspond to a setup completion with rate $\gamma A^+ K_{\text{I}}$, and dash-dotted transitions correspond to a service completion with rate $b\mu$, where b is the number of BUSY servers in the current state.

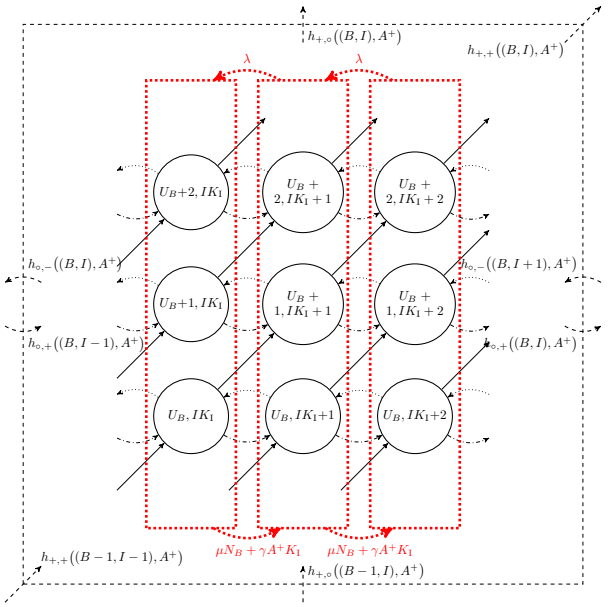

Fig. 3. Transition diagram of a multi-level state (B,I) when $I\le 0$ and $K_{\text{B}}=K_{\text{I}}=3$. Dotted transitions correspond to a job arrival with rate λ, solid transitions correspond to a setup completion with rate $\gamma A^+ K_{\text{I}}$, and dash-dotted transitions correspond to a service completion with rate $b\mu$, where b is the number of BUSY servers in the current state.

This article has been accepted for publication in IEEE Transactions on Services Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TSC.2024.3354202

We claim $\sum_{k=0}^{K_\mathrm{I}-1}\eta^k P(IK_\mathrm{I}|S,A)=1$. This can be verified by observing that the left-hand side is the probability that M^{ml} is at level I for the given state S and action A. Hence, for $S=(B,I)$, this happens with probability 1 and the claim follows. Using algebraic manipulations, it then implies that for $i=IK_\mathrm{I}$:

$$P(IK_\mathrm{I}|S,A)=\begin{cases}1/K_\mathrm{I}; & \eta=1,\\ \frac{1-\eta}{1-\eta^{K_\mathrm{I}}}; & \text{otherwise.}\end{cases}$$

Following a similar reasoning, we obtain:

$$P((I+1)K_\mathrm{I}-1|S,A) = \begin{cases}P(IK_\mathrm{I}|S,A)K_\mathrm{I}; & \eta=1,\\ P(IK_\mathrm{I}|S,A)\frac{1-\eta^{K_\mathrm{I}}}{1-\eta}; & \text{otherwise.}\end{cases}$$

Observing that $\underline{u}(S,A)=P(IK_\mathrm{I}|S,A)$ and $\overline{u}(S,A)=P((I+1)K_\mathrm{I}-1|S,A)$ concludes the proof. □

We conclude this subsection by deriving the expected number of either IDLE servers or waiting jobs under a given pair (S,A), which we denote by $\overline{I}(S,A)$. We have:

$$\begin{aligned}\overline{I}(S,A) &= \sum_{i=K_\mathrm{I}I}^{K_\mathrm{I}(I+1)-1} iP(i|S,A)\\ &= \frac{1-\eta(S,A)}{1-\eta(S,A)^{K_\mathrm{I}}}\sum_{i=K_\mathrm{I}I}^{K_\mathrm{I}(I+1)-1} i\eta(S,A)^{i-K_\mathrm{I}I}\\ &= \underline{u}(S,A)\sum_{i=K_\mathrm{I}I}^{K_\mathrm{I}(I+1)-1} i\eta(S,A)^{i-K_\mathrm{I}I}, \qquad (8)\end{aligned}$$

where the second line follows from the recursive property of P established in the proof of Lemma 2.

B. The Transition Function q^{ml}

We are now ready to fully characterize q^{ml} using the level boundary probabilities in the preceding subsection. This entails calculating the transition rates when the BUSY level, I/W level, or both change. In what follows, we derive the level transition probabilities resulting from the positive part of the action, i.e., A^+. These transitions, corresponding to the cases $I>0$ and $I\le 0$, are shown on the outer dotted boxes in Fig. 2 and Fig. 3, respectively. The value of A^- does not affect the BUSY level B and only changes the I/W level I to $I+A^-$ instantly with probability one. Hence, its effect is incorporated into the conditions. The transition function q^{ml} admits the following form:

$$q^{\mathrm{ml}}(S'|S,A)=\begin{cases}h_{+,\circ}(S,A^+); & B'=B+1, I'=I+A^-,\\ h_{-,\circ}(S,A^+); & B'=B-1, I'=I+A^-,\\ h_{\circ,+}(S,A^+); & B'=B, I'=I+1+A^-,\\ h_{\circ,-}(S,A^+); & B'=B, I'=I-1+A^-,\\ h_{+,+}(S,A^+); & B'=B+1, I'=I+1+A^-,\\ h_{+,-}(S,A^+); & B'=B+1, I'=I-1+A^-,\\ h_{-,+}(S,A^+); & B'=B-1, I'=I+1+A^-,\end{cases}$$

where $h_{i,j}$ with $i,j\in\{-,+,\circ\}$ are (output) rate functions that will be derived momentarily. Here, the first subscript i represents a change in B, and the second j captures a change in I. The symbol $\circ$ indicates no change during the transition, whereas $+$ and $-$ indicate increment and decrement, respectively. For example, $h_{+,\circ}(S,A)$ denotes the transition rate from state $S=(B,I)$ to state $S'=(B+1,I)$ under action A. The rest of this subsection is devoted to deriving the rate functions $h_{i,j}$. Here $|A^-|$ indicates the number of IDLE servers that will be turned off. Thus, as an immediate result of taking action A, the number of IDLE servers is reduced instantaneously by $|A^-|$, i.e., $I'=I+A^-$. Obviously, if $A\ge 0$, then I' will not change as a result of A^-. Other changes to I occur as a result of action, A^+, and in a probabilistic manner.

BUSY Level Transition Rates ($h_{+,\circ}$ and $h_{-,\circ}$). If a transition from state $S=(B,I)$ to state $S'=(B+1,I)$ under the action A^+ occurs, then the number of BUSY servers should be equal to the upper boundary of the current BUSY level B (i.e., $b=U_{B+1}-1$) and increases by one as well. This increment occurs when a job arrives with rate λ when $I\ge 0$ or a SETUP server becomes IDLE with rate γA^+K_I when $I<0$. In both cases, the I/W-component remains unchanged if that of M is different than the lower and upper boundaries of the current I/W-component, respectively. Hence,

$$h_{+,\circ}(S,A^+)=\begin{cases}\lambda\overline{p}(B)\,(1-\underline{u}(S,A))\,; & I\ge 0,\\ \gamma A^+K_\mathrm{I}\overline{p}(B)\,(1-\overline{u}(S,A))\,; & I<0.\end{cases} \qquad (9)$$

The transition rate from $S=(B,I)$ to $S'=(B-1,I)$ under A^+ is captured by $h_{-,\circ}(S,A^+)$. When $I\ge 0$, such a transition can arise only if the number of busy servers is equal to the lower boundary of the current BUSY level B (with probability $\underline{p}(B)$), the number of IDLE servers is not equal to the upper boundary of the current I/W level I (with probability $1-\overline{u}(S,A)$), and a BUSY server turns IDLE (with rate μU_B). We thus have:

$$h_{-,\circ}(S,A^+)=\begin{cases}\mu U_B\underline{p}(B)\,(1-\overline{u}(S,A))\,; & I\ge 0,\\ 0\,; & I<0.\end{cases} \qquad (10)$$

I/W Level Transition Rates ($h_{\circ,+}$ and $h_{\circ,-}$). Introduce:

$$N_B^- = \frac{1}{F\big(U_{B+1}-1\big)-F\big(U_B-1\big)}\sum_{x=U_B+1}^{U_{B+1}-1} xf(x). \qquad (11)$$

In fact, N_B^- captures the average number of BUSY servers in level B without including the lower boundary of the level. Transitions from $S=(B,I)$ to $S'=(B,I+1)$ under A^+ occurs at rate:

$$h_{\circ,+}(S,A^+)=\begin{cases}\overline{u}(S,A)\big(\gamma A^+K_\mathrm{I}+(1-\underline{p}(B))\mu N_B^-\big); & I\ge 0,\\ \overline{u}(S,A)\big(\mu N_B+(1-\overline{p}(B))\gamma A^+K_\mathrm{I}\big); & I<0.\end{cases}$$

To verify this, observe that when $I\ge 0$, I increases by one if i is equal to the upper boundary of current I/W level (with probability $\overline{u}(S,A)$) and increases by one as well (see Fig. 2). Also, i increases if either a SETUP server becomes IDLE with rate γA^+K_I or a BUSY server turns IDLE. In the latter, to remain in the same BUSY level, the number of busy levels should not be equal to the lower boundary of the current BUSY level. Thus, the rate of having an IDLE server in the latter case is equal to $(1-\underline{p}(B))\mu N_B^-$. In the second case of $I\le 0$, I increases if i is equal to the upper boundary of I/W level and increases by one (see Fig. 3). This occurs when either a BUSY server turns IDLE with rate μN_B or a SETUP server becomes IDLE with rate γA^+K_I, thus serving a waiting job. In the latter, a SETUP server becomes BUSY and thus, to have the same BUSY level, b should not be equal to the upper boundary of the current BUSY level.

This article has been accepted for publication in IEEE Transactions on Services Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TSC.2024.3354202

We now turn to deriving $h_{\circ,-}(S,A^+)$, which captures the transition rate from $S=(B,I)$ to $S'=(B,I-1)$ under A^+. Note that I decreases by one if a job arrives with rate λ and i is equal to the lower boundary of I/W level. Then, if $I<0$, B remains unchanged since the newly arrived job increases the number of waiting jobs. However, if $I\geq 0$, then the new job turns an IDLE server BUSY; thus, in order to remain in the same BUSY level, the number of BUSY servers should not be equal to the upper boundary of level B. We thus get:

$$h_{\circ,-}(S,A^+)=\begin{cases}(1-\overline{p}(B))\,\lambda\underline{u}(S,A); & I\geq 0,\\ \lambda\underline{u}(S,A); & I<0.\end{cases} \tag{12}$$

Joint BUSY and I/W Level Transition Rates ($h_{+,+}$, $h_{+,-}$ and $h_{-,+}$). Transition from $S=(B,I)$ to $S'=(B+1,I-1)$ under A^+ occurs with rate λ only when $I\geq 0$ and the values of BUSY and IDLE servers are equal to the upper and lower boundaries of the corresponding levels, respectively. It thus transpires with a rate:

$$h_{+,-}(S,A^+)=\begin{cases}\lambda\overline{p}(B)\underline{u}(S,A); & I\geq 0,\\ 0; & I<0.\end{cases} \tag{13}$$

On the other hand, $S=(B,I)$ transits to $S'=(B-1,I+1)$ under A^+ only when $I\geq 0$ and the number of BUSY and IDLE servers are equal to the lower and upper boundaries of the corresponding levels, respectively, which further yields:

$$h_{-,+}(S,A^+)=\begin{cases}U_B\mu\underline{p}(B)\overline{u}(S,A); & I\geq 0,\\ 0; & I<0.\end{cases} \tag{14}$$

Finally, transition from $S=(B,I)$ to $S'=(B+1,I+1)$ under A^+ happens at rate:

$$h_{+,+}(S,A^+)=\begin{cases}\gamma A^+K_{\mathrm{I}}\overline{p}(B)\overline{u}(S,A); & I<0,\\ 0; & I\geq 0.\end{cases} \tag{15}$$

C. The Reward Function r^{ml}

Let us define the rate function Ψ^{ml} associated to q^{ml} as:

$$\Psi^{\mathrm{ml}}=h_{+,\circ}+h_{-,\circ}+h_{\circ,+}+h_{\circ,-}+h_{-,+}+h_{+,-}+h_{+,+}. \tag{16}$$

It is evident that $1/\Psi^{\mathrm{ml}}(S,A)$ is the average time spent at state S under action A. Hence, the reward in state S under action A is:

$$r^{\mathrm{ml}}(S,A)$$
$$=\frac{-1}{\Psi^{\mathrm{ml}}(S,A)}\Big(c_{\mathrm{perf}}|\overline{I}(S,A)^-|+c_{\mathrm{power}}\overline{I}(S,A)^+ + c'_{\mathrm{power}}K_{\mathrm{I}}A^+\Big),$$

where $\overline{I}(S,A)$ is defined in (8).

D. Solving Multi-level CTMDP

Armed with the characterization of M^{ml}, we can derive a similar result to Lemma 1 for M^{ml}.

Lemma 3. *The (per-step) time complexity of solving the MDP associated to M^{ml} using value iteration is $O(L^2(Q/K_{\mathrm{I}}+L^2)(Q/K_{\mathrm{I}}+L))$.*

Proof. The proof follows a similar argument as in the proof of Lemma 1. Each iteration in value iteration costs $O(|\mathcal{S}|\sum_{S\in\mathcal{S}^{\mathrm{ml}}}|\mathcal{A}_S^{\mathrm{ml}}|)$. Introduce $\mathcal{S}_+^{\mathrm{ml}}:=\{(B,I)\in\mathcal{S}^{\mathrm{ml}},I>0\}$ and $\mathcal{S}_-^{\mathrm{ml}}:=\{(B,I)\in\mathcal{S}^{\mathrm{ml}},I\leq 0\}$ so that $\mathcal{S}_-^{\mathrm{ml}}\cup\mathcal{S}_+^{\mathrm{ml}}=\mathcal{S}^{\mathrm{ml}}$. Furthermore, $\sum_{S\in\mathcal{S}^{\mathrm{ml}}}|\mathcal{A}_S^{\mathrm{ml}}|=\sum_{S\in\mathcal{S}_+^{\mathrm{ml}}}|\mathcal{A}_S^{\mathrm{ml}}|+\sum_{S\in\mathcal{S}_-^{\mathrm{ml}}}|\mathcal{A}_S^{\mathrm{ml}}|$.

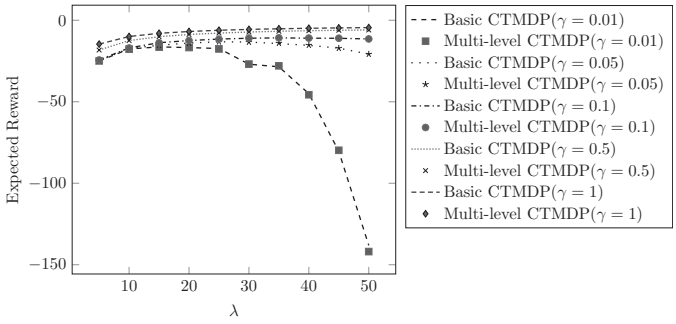

Fig. 4. Validation of the Multi-level CTMDP model M^{ml} with respect to the basic CTMDP model M for $L=C=100$.

- *When $S=(B,I)\in\mathcal{S}_-^{ml}$:* Recalling that $B\in\{0,\ldots,L-1\}$ and $-\lfloor Q/K_{\mathrm{I}}\rfloor\leq I\leq -1$, we have $|\mathcal{S}_-^{\mathrm{ml}}|=L\lfloor Q/K_{\mathrm{I}}\rfloor$. Furthermore, $\sum_{s\in\mathcal{S}_-^{\mathrm{ml}}}|\mathcal{A}_S^{\mathrm{ml}}|=2L\lfloor Q/K_{\mathrm{I}}\rfloor$ since there are two possible actions in S.
- *When $S=(B,I)\in\mathcal{S}_+^{ml}$:* Since $0\leq I\leq L-1$, B can take different values depending on the value of I. But we assume the worst case where B can take all values in $\{0,1,\cdots,L-1\}$ at any value of I. Then, we have $|\mathcal{S}_+^{\mathrm{ml}}|=L^2$. Also, is such states, we have $|\mathcal{A}_S^{\mathrm{ml}}|=I+2$. Consequently, $\sum_{S\in\mathcal{S}_+^{\mathrm{ml}}}|\mathcal{A}_S^{\mathrm{ml}}|=\sum_{I=0}^{L-1}L(I+2)=L^2(L/2+3/2)$.

Consequently, $|\mathcal{S}^{\mathrm{ml}}|\sum_{S\in\mathcal{S}^{\mathrm{ml}}}|\mathcal{A}_S^{\mathrm{ml}}|=L^2(L+\lfloor Q/K_{\mathrm{I}}\rfloor)(L^2/2+2\lfloor Q/K_{\mathrm{I}}\rfloor+3L/2)$, leading to the approximate complexity $O(L^2(L^2+Q/K_{\mathrm{I}})(Q/K_{\mathrm{I}}+L))$. This completes the proof. □

VI. SIMULATION RESULTS AND DISCUSSIONS

In this section, we assess the efficacy of our multi-level CTMDP using numerical experiments. Since our proposed approach assumes known and fixed parameters, it is considered an offline optimization method. As a result, we do not utilize real traces with time-varying rates for performance evaluation in this paper. This aligns with previous analytical studies on multi-server scenarios with setup time ([7]–[10], [12]–[14], [18], [20]), which evaluate their methods using events generated from distribution functions with known parameters. Moreover, existing traces, to the best of our knowledge, do not provide information about server setup times, which are fundamental components of the system model in this paper. To evaluate the performance, we compare the results with the *staggered threshold* and *bulk setup* policies, using parameters from [20], and the *uniform state-aggregation* method in [21]. The equivalent (discrete-time) MDPs are solved using linear programming methods for multi-chain MDPs [16] and the Gurobi Java plugin [34]. The equivalent discrete-time MDP is used considering the fact that the effect of the residual time in each state is considered in the reward function in (3) through the term $1/\Psi(s,a)$. Moreover, to derive the transition probabilities, all transition rates at state s under action a are normalized by the total transition rate at state s under action a. Henceforth, we assume $c_{\mathrm{power}}=1$ and $c'_{\mathrm{power}}=2$, indicating that each SETUP server consumes twice the power of an IDLE server. Also, in all experiments, we assume that the number of servers is $C=100$.

Fig. 4 compares the multi-level and basic CTMDPs assuming $Q=C=100$, where we set $L=C=100$ (thus, $K_{\mathrm{B}}=K_{\mathrm{I}}=1$). The optimal expected average reward is plotted versus the arrival rate λ, where $\mu=1$, $c_{\mathrm{perf}}=50$ are fixed. As the figure shows,

This article has been accepted for publication in IEEE Transactions on Services Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TSC.2024.3354202

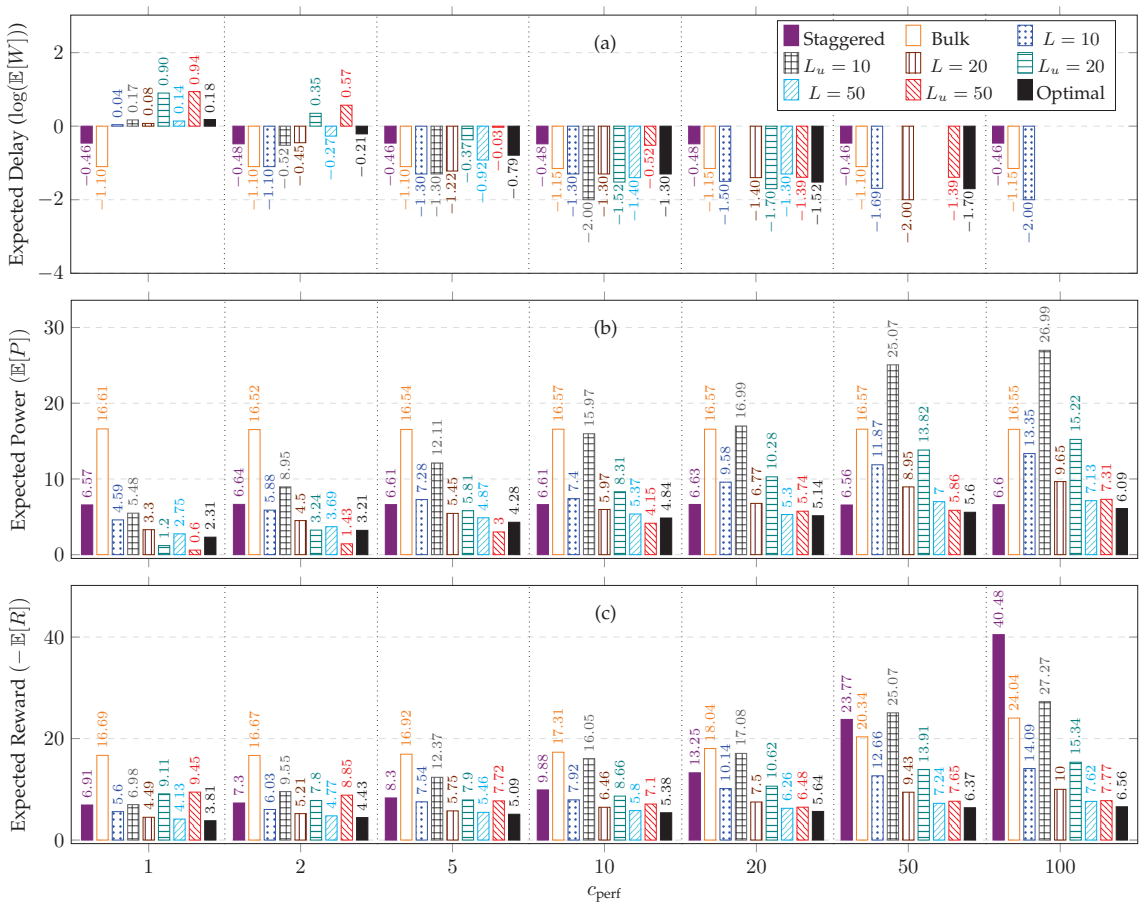

Fig. 5. Expected delay, power, and reward per time unit for varying c_{perf} values under different policies ($\gamma = 2, \lambda = 30$).

the multi-level and basic CTMDPs exhibit exactly the same performance for varying setup and arrival rates.

Two fixed-threshold methods, namely staggered threshold and bulk setup policies [20], have been reported in the literature for power management in multi-server systems with setup times. We denote them by π_{stag} and π_{bulk}, respectively. Both policies use a threshold parameter C_{s}, called '*static* ON' servers, which represents the number of servers that should always be powered on. Mathematically, they are defined as [20]:

$$\pi_{\text{bulk}}(b,i) = \begin{cases} C_{\text{s}} - b - i^{+}; & b + i^{+} \leq C_{\text{s}}, \\ (C_{\text{s}} - b)^{+} - i; & b + i > C_{\text{s}}, i > -k, \\ C - b; & b > C_{\text{s}}, i \leq -k. \end{cases} \tag{17}$$

Here, π_{stag} is the same as π_{bulk}, except that $\pi_{\text{stag}}(b,i) = |i|$ when $b + i > C_{\text{s}}$ and $i \geq 0$. In both policies, greater values of k means higher priority of power over delay. Here, we set the threshold $k = 1$ to get the highest priority of delay over power. We consider $C_{\text{s}} = \rho + \sqrt{\rho}$, which is shown in [20] to be the optimal value C_{s}.

We also compare our CTMDP model with the *uniform state-aggregation* approach, which derives the reward and transition rate of a meta-state by averaging those of the corresponding aggregated states. Assuming L_u levels in this method, we have $K = K_{\text{B}} = K_{\text{I}} = C/L_u$. Hence, under this method, $A = x$ corresponds to $a = xK$ in M, and for $S, S' \in \mathcal{S}^{\text{ml}}$, we have:

$$q(S'|S,A) = \frac{1}{K^4} \sum_{w=0}^{K-1} \sum_{x=0}^{K-1} \sum_{y=0}^{K-1} \sum_{z=0}^{K-1} q\big((B'K+w, I'K+x)|(BK+y, IK+z), AK\big).$$

In what follows, terms $\mathbb{E}[W]$ and $\mathbb{E}[P]$ refer to the average delay and average power of the policies per time unit, respectively. However, in regard to the definition of the reward in (3), the term $\mathbb{E}[W]$ essentially denotes the average number of waiting jobs ($|i^{-}|$), and $\mathbb{E}[P]$ is the average weighted sum of the number of IDLE and SETUP servers ($c_{\text{power}} i^{+} + c'_{\text{power}} a^{+}$). Fig. 5 compares the policies discussed in terms of $\mathbb{E}[W]$, $\mathbb{E}[P]$, and expected reward ($\mathbb{E}[R]$) for values of c_{perf} taken from $\{1, 2, 5, 10, 20, 50, 100\}$ and $\lambda = 30$, $\mu = 1$, $\gamma = 2$. Note that these values are not estimated; rather, they are computed since model parameters are known. Since π_{bulk} turns on all OFF servers whenever the number of waiting jobs is greater than the given threshold k, it prioritizes the delay as compared to π_{stag} and thus, consumes more power and the jobs receive service with less delay (see Fig. 5a and Fig. 5b). Furthermore, for larger c_{perf} values ($c_{\text{perf}} \geq 50$), which indicate delay being prioritized over power, π_{bulk} results in higher reward since it prioritizes delay, while for smaller c_{perf} values, π_{stag} outperforms π_{bulk}. Since both π_{bulk} and π_{stag} are independent of c_{perf}, the power and delay in these methods do not change for different c_{perf} values. Moreover, it is evident in Fig. 5c that the absolute value reward of the multi-level CTMDP decreases with L since at larger values of L, we have a more accurate model. On the other hand, based on the dimensionality analysis

This article has been accepted for publication in IEEE Transactions on Services Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TSC.2024.3354202

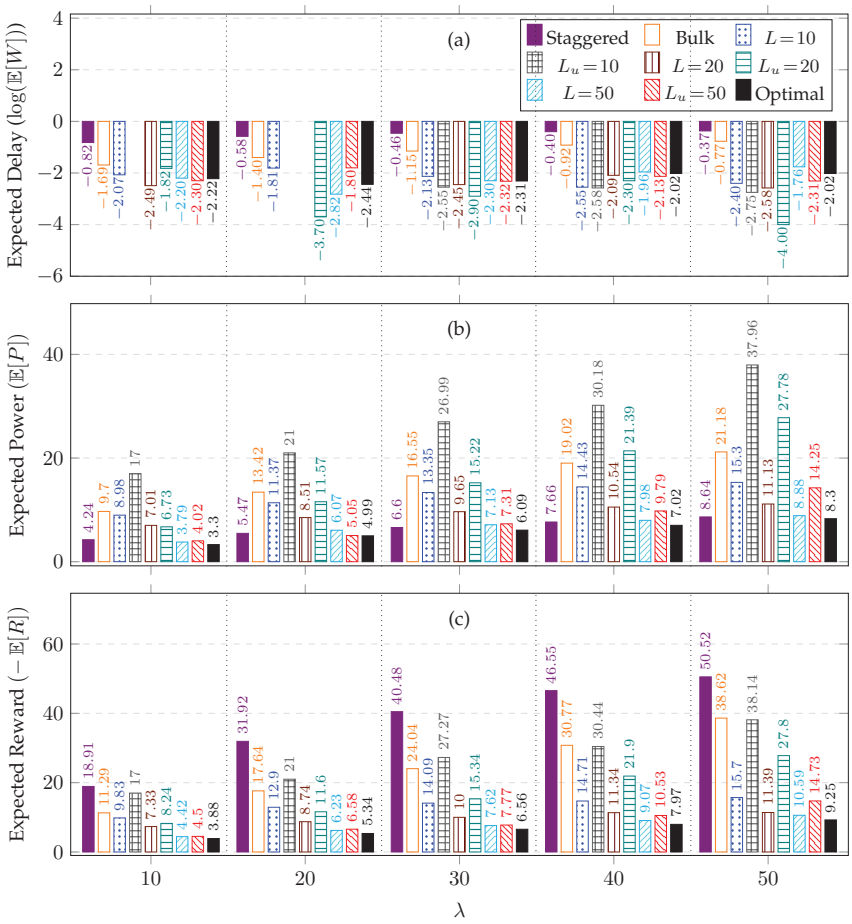

Fig. 6. Expected delay, power, and reward per time unit for varying λ values under different policies ($\gamma = 2, c_{\text{perf}} = 100$).

made in Section IV, for $L = 50$, $L = 20$ and $L = 10$, the size of multi-level CTMDP is 32, 3125, and 100,000 times smaller than the optimal CTMDP, respectively. Moreover, even for the smallest L value ($L=10$) in our experiment, the reward achieved from multi-level CTMDP is better than both π_{bulk} and π_{stag}. It should also be noted in Fig. 5c that the reward of the uniform state-aggregation method is at most equal to that of the multi-level CTMDP. This shows that our method offers a better approximation of the basic CTMDP than the uniform state-aggregation method.

For varying values of $\lambda \in \{10, 20, 30, 40, 50\}$, $\mathbb{E}[W]$, $\mathbb{E}[P]$, and $\mathbb{E}[R]$ of the different policies are compared in Fig. 6. The parameters $c_{\text{perf}} = 100$, $\mu = 1$, and $\gamma = 2$ are set to be fixed. As shown in Fig. 6a and Fig. 6b, the mean delays of both π_{bulk} and π_{stag} are much larger (about 100 times) than optimal and multi-level approaches. The power and delay of π_{bulk} and π_{stag} increase with λ in Fig. 6b and Fig. 6c, thus leading to lower rewards as depicted in Fig. 6d. Indeed, by increasing λ, the traffic density of the system increases, which results in longer average waiting time. On the other hand, more servers will be in SETUP state to deal with higher traffic which leads to more energy consumption too. However, such a monotonic increase in power and delay cannot be observed in CTMDP-based approaches since they minimize the weighted sum of power and performance penalties and thus, improving one component may affect the other component for different λ values. Nevertheless, it is apparent in Fig. 6d that the reward decreases with λ in CTMDP-based approaches.

Finally, Fig. 7 compares the mean delay, power, and reward calculated for different values of $\gamma \in \{0.1, 0.5, 1, 2, 5\}$, where $c_{\text{perf}} = 100$, $\mu = 1$, and $\lambda = 30$. In Fig. 7a and Fig. 7b, it can be observed that for bulk setup and staggered threshold policies, increasing γ results in the decrease of the delay, because the setup process finishes faster and the jobs experience less delay. The power also drops with γ as shown in Fig. 7c since decrease in the SETUP delay brings about reduction in the number of SETUP servers. Similar to Fig. 6, for CTMDP-based approaches, such a monotonic decrease cannot be observed for delay and power separately, but the increase in reward with respect to γ is clearly evident in Fig. 7d.

VII. CONCLUSION

We have presented a multi-level CTMDP as an approximate model for power management in large scale cloud data centers with setup time. The multi-level CTMDP is derived using a novel state aggregation technique that exploits the intrinsic structure of the model. It is fully characterized under mild assumptions and approximations and is shown to admit a significantly smaller state-action space than the exact model, which makes it a viable solution to remedy the curse of dimensionality in large-scale systems. Through numerical simulations, we demonstrated that the resulting power management policies are superior to existing fixed threshold methods. As future work, it would be

This article has been accepted for publication in IEEE Transactions on Services Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TSC.2024.3354202

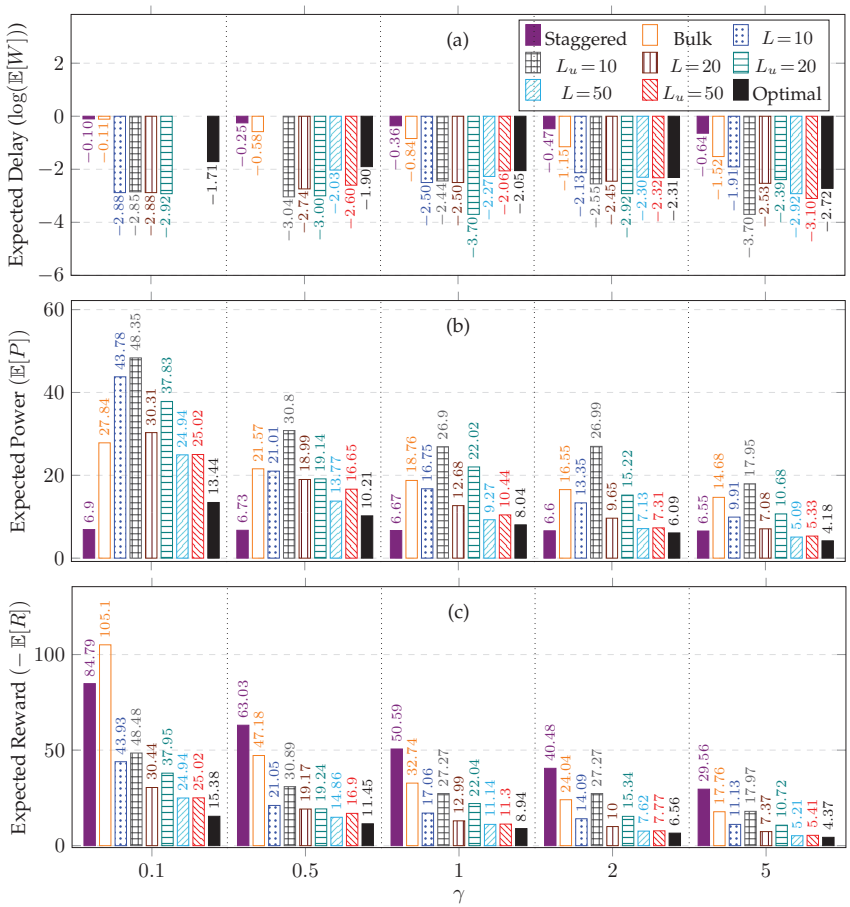

Fig. 7. Expected delay, power, and reward per time unit for varying γ values under different policies ($\lambda = 30, c_{\text{perf}} = 100$).

intriguing to extend this model by explicitly incorporating power consumption for physical machines and considering the server power switching cost, commonly referred to as the *wear-and-tear* cost. Another promising research direction is to investigate power management using derived models within an online reinforcement learning setting, such as the approaches presented in [35], [36], where system parameters are unknown.

ACKNOWLEDGMENT

This work was supported by the University of Tehran and the Institute for Research in Fundamental Sciences under grant number CS1399-2-02, and in part by the Social Policy Grant (SPG) funded by Nazarbayev University, Kazakhstan.

REFERENCES

[1] S. G. Umamaheswaran, S. A. Mamun, A. Ganguly, M. Kwon, and A. Kwasinski. Reducing power consumption of datacenter networks with 60GHz wireless server-to-server links. In *Proc. IEEE Global Communications Conference (GLOBECOM)*, pages 1–7, 2017.

[2] M. Dayarathna, Y. Wen, and R. Fan. Data center energy consumption modeling: A survey. *IEEE Commun. Surv. Tutor.*, 18(1):732–794, 2016.

[3] C. Gu, Z. Li, H. Huang, and X. Jia. Energy efficient scheduling of servers with multi-sleep modes for cloud data center. *IEEE Trans. Cloud Comput.*, 8(3):833–846, Jul.-Sep. 2020.

[4] N. Hogade, S. Pasricha, and H. J. Siegel. Energy and network aware workload management for geographically distributed data centers. *IEEE Trans. Sustain. Comput.*, 7(2):400–413, Apr. 2022.

[5] W. Li, Q. Fan, W. Cui, F. Dang, X. Zhang, and C. Dai. Dynamic virtual machine consolidation algorithm based on balancing energy consumption and quality of service. *IEEE Access*, 10:80958–80975, 2022.

[6] T. Tournaire, H. Castel-Taleb, E. Hyon, and T. Hoche. Generating optimal thresholds in a hysteresis queue: Application to a cloud model. In *Proc. International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS)*, pages 283–294, 2019.

[7] A. Gandhi, M. Harchol-Balter, and I. Adan. Server farms with setup costs. *Perform. Evaluation*, 67(11):1123–1138, Nov. 2010.

[8] T. Phung-Duc and K. Kawanishi. Delay performance of data-center queue with setup policy and abandonment. *Ann. Oper. Res.*, 293(1):269–293, Oct. 2020.

[9] A. Gandhi, S. Doroudi, M. Harchol-Balter, and A. Scheller-Wolf. Exact analysis of the M/M/k/setup class of Markov chains via recursive renewal reward. *ACM SIGMETRICS Perform. Evaluation Rev.*, 41(1):153–166, Jun. 2013.

[10] T. Phung-Duc. Exact solutions for M/M/c/Setup queues. *Telecommun. Syst.*, 64(2):309–324, Feb. 2017.

[11] C. Hu, Y. Deng, G. Min, P. Huang, and X. Qin. QoS promotion in energy-efficient datacenters through peak load scheduling. *IEEE Trans. on Cloud Comput.*, 9(2):777–792, Apr. 2021.

[12] F. Longo, R. Ghosh, V. K. Naik, and K. S. Trivedi. A scalable availability model for Infrastructure-as-a-Service cloud. In *Proc. IEEE/IFIP International Conference on Dependable Systems Networks (DSN)*, pages 335–346, 2011.

[13] B. Wang, X. Chang, and J. Liu. Modeling heterogeneous virtual machines on IaaS data centers. *IEEE Commun. Lett.*, 19(4):537–540, Apr. 2015.

[14] X. Chang, B. Wang, J. K. Muppala, and J. Liu. Modeling active virtual machines on IaaS clouds using an M/G/m/m+K queue. *IEEE Trans. Serv. Comput.*, 9(3):408–420, May-Jun. 2016.

[15] M. E. Gebrehiwot, S. Aalto, and P. Lassila. Optimal energy-aware control policies for FIFO servers. *Perform. Evaluation*, 103:41–59, Sep. 2016.

[16] M. L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Wiley, Apr. 1994.

[17] Z. Yang, M.-H. Chen, Z. Niu, and D. Huang. An optimal hysteretic con-

trol policy for energy saving in cloud computing. In *Proc. IEEE Global Telecommunications Conference (GLOBECOM)*, pages 1–5, 2011.
[18] E. Hyytiä, D. Down, P. Lassila, and S. Aalto. *Dynamic Control of Running Servers*, pages 127–141. Springer International Publishing, 2018.
[19] S. Aalto and P. Lassila. Near-optimal dispatching policy for energy-aware server clusters. *Perform. Evaluation*, 135:102034, Nov. 2019.
[20] V. J. Maccio and D. G. Down. Structural properties and exact analysis of energy-aware multiserver queueing systems with setup times. *Perform. Evaluation*, 121–122:48–66, May 2018.
[21] Z. Ren and B. H. Krogh. State aggregation in Markov decision processes. In *Proc. IEEE Conference on Decision and Control (CDC)*, pages 3819–3824, 2002.
[22] L. Li, T. J. Walsh, and M. L. Littman. Towards a unified theory of state abstraction for MDPs. In *Proc. International Symposium on Artificial Intelligence and Mathematics (AI&Math)*, pages 1–10, 2006.
[23] M. Hutter. Extreme state aggregation beyond Markov decision processes. *Theor. Comput. Sci.*, 650:73–91, Oct. 2016.
[24] D. Abel, D. E. Hershkowitz, and M. L. Littman. Near optimal behavior via approximate state abstraction. In *Proc. International Conference on Machine Learning (ICML)*, pages 2915–2923, 2016.
[25] J. Taylor, D. Precup, and P. Panagaden. Bounding performance loss in approximate MDP homomorphisms. In *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, volume 21, 2008.
[26] N. Saldi, T. Linder, and S. Yuksel. Finite state approximations of Markov decision processes with general state and action spaces. In *Proc. IEEE American Control Conference (ACC)*, pages 3589–3594, 2015.
[27] Y. Ran, H. Hu, X. Zhou, and Y. Wen. DeepEE: Joint optimization of job scheduling and cooling control for data center energy efficiency using deep reinforcement learning. In *Proc. IEEE International Conference on Distributed Computing Systems (ICDCS)*, pages 645–655, 2019.
[28] D. Yi, X. Zhou, Y. Wen, and R. Tan. Toward efficient compute-intensive job allocation for green data centers: A deep reinforcement learning approach. In *Proc. IEEE International Conference on Distributed Computing Systems (ICDCS)*, pages 634–644, 2019.
[29] W. Zhang, Y. Wen, L. L. Lai, F. Liu, and R. Fan. Electricity cost minimization for interruptible workload in datacenter servers. *IEEE Trans. Serv. Comput.*, 13(6):1059–1071, Nov.-Dec. 2017.
[30] M. A. Islam, S. Ren, A. H. Mahmud, and G. Quan. Online energy budgeting for cost minimization in virtualized data center. *IEEE Trans. Serv. Comput.*, 9(3):421–432, May 2016.
[31] Y. Ran, H. Hu, Y. Wen, and X. Zhou. Optimizing energy efficiency for data center via parameterized deep reinforcement learning. *IEEE Trans. Serv. Comput.*, 16(2):1310–1323, Mar. 2023.
[32] S. Di, D. Kondo, and F. Cappello. Characterizing and modeling cloud applications/jobs on a Google data center. *J. Supercomput.*, 69(1):139–160, Apr. 2014.
[33] X. Guo and O. Hernández-Lerma. *Continuous-Time Markov Decision Processes: Theory and Applications*. Springer Berlin Heidelberg, 2009.
[34] LLC Gurobi Optimization. Gurobi optimizer reference manual, 2021.
[35] T. Jaksch, R. Ortner, and P. Auer. Near-optimal regret bounds for reinforcement learning. *J. Mach. Learn. Res.*, 11(51):1563–1600, 2010.
[36] H. Bourel, O. Maillard, and M. S. Talebi. Tightening exploration in upper confidence reinforcement learning. In *Proc. International Conference on Machine Learning (ICML)*, pages 1056–1066, 2020.

Behzad Chitsaz received the B.Sc. degree in information technology and the M.Sc. degree in computer engineering from the Department of Computer Engineering and Information Technology, Amirkabir University of Technology, Tehran, Iran, in 2010 and 2013, respectively, and the Ph.D. degree in computer engineering from the Electrical and Computer Engineering Department, University of Tehran, Tehran, Iran, in 2020. His research interests include performance evaluation and modeling, high performance distributed systems, and cloud computing.

Ahmad Khonsari received the B.Sc. degree in electrical and computer engineering from Shahid Beheshti University, Iran, in 1991, the M.Sc. degree in computer engineering from the Iran University of Science and Technology (IUST), Iran, in 1996, and the Ph.D. degree in computer science from the University of Glasgow, UK, in 2003. He is currently an Associate Professor in the Department of Electrical and Computer Engineering, University of Tehran, Iran, and a Researcher at the School of Computer Science, Institute for Research in Fundamental Sciences (IPM), Iran. His research interests include simulation and data analysis, performance modeling/evaluation, wired/wireless networks, cloud and distributed systems, quantum information processing, and high-performance computer architectures.

Masoumeh Moradian received the B.S., M.S., and Ph.D. degrees in electrical engineering from the Sharif University of Technology, Tehran, Iran, in 2007, 2010, and 2016, respectively. She was a Visiting Scholar with the Chinese University of Hong Kong in 2015. She is currently an Assistant Professor with the School of Computer Engineering, K. N. Toosi University of Technology, Tehran, Iran, and a Researcher with the School of Computer Science, Institute for Research in Fundamental Sciences (IPM), Tehran, Iran. Her current research interests include status-updating networks, content caching networks, energy harvesting communication networks, and network stochastic optimization.

Aresh Dadlani (Senior Member, IEEE) received the B.Sc. and M.Sc. degrees in electrical and computer engineering from the University of Tehran, Tehran, Iran, in 2007 and 2010, respectively, and the Ph.D. degree from the School of Information and Communications, Gwangju Institute of Science and Technology (GIST), Gwangju, South Korea, in 2015. From August 2015 to July 2017, he was a Postdoctoral Researcher with the Center for Integrated Access Systems at GIST. He held the position of Assistant Professor at the School of Engineering and Digital Sciences, Nazarbayev University (NU), Kazakhstan, from September 2017 to February 2022. Currently, he is a Researcher at the Department of Computing Science, University of Alberta, Edmonton, Canada. His research interests include modeling and analysis of complex system dynamics, network science, and applications of optimization techniques and modern queuing theory in wireless communication networks.

Mohammad Sadegh Talebi received the B.Sc. degree in electrical engineering from the Iran University of Science and Technology, Tehran, Iran, in 2004, the M.Sc. degree in electrical engineering from the Sharif University of Technology, Tehran, in 2006, and the Ph.D. degree in electrical engineering from the KTH Royal Institute of Technology, Stockholm, Sweden, in 2017. From June 2018 to January 2020, he was a Postdoctoral Researcher with the SequeL (currently, Scool) Team, Inria Lille–Nord Europe, Lille, France. Since February 2020, he has been a Tenure-Track Assistant Professor with the Department of Computer Science, University of Copenhagen, Copenhagen, Denmark. His primary research interests include theoretical reinforcement learning, and adaptive control under uncertainty.