Power-Saving Design for Server Farms with Response Time Percentile Guarantees

Shengquan Wang and Jun Liu
University of Michigan-Dearborn, USA
{shqwang,juli}@umd.umich.edu

Waqaas Munawar and Jian-Jia Chen
Karlsruhe Institute of Technology, Germany
{munawar,j.chen}@kit.edu

Xue Liu
McGill University, Canada
xueliu@cs.mcgill.ca

Abstract—The expense of power cost in server farms has driven the recent power-aware development in both industry and academia. At the same time, a Service Level Agreement (SLA) of service performance between a customer and a service provider is demanded to meet the customer satisfaction. This paper investigates the queuing-theoretical power-saving design strategy for server farms under a given SLA, which in particular is measured in a certain level percentile of the job response time. We consider server farms with servers that are equipped with the capabilities of Dynamic Voltage Scaling (DVFS) and Dynamic Power Management (DPM). We adopt an M/G/1/PS server model, where the job service time distribution is assumed heavy-tailed, and as discovered and validated by previous research. We propose an optimal design strategy called PowerTail to minimize the power consumption under the given SLA. Our data confirms that the proposed PowerTail strategy significantly outperforms the intuitive load-balancing strategy.

I. INTRODUCTION

Power has become an essential problem for computing systems. The advanced hardware technology has improved the performance per dollar spent on hardware, but the performance per watt remains roughly flat over time [1]. It is reported that the electricity cost of server clusters will exceed the hardware cost and become a very significant portion of the total cost of ownership for the maintenance of servers. Moreover, given the large number of servers in use today, the worldwide expenditure on enterprise power and cooling of these servers is estimated to surpass $30 billion [2].

From the customer side, Quality of Service (QoS) provided by server providers is one of the key factors to customer’s satisfaction. A QoS guarantee is normally contracted with a Service Level Agreement (SLA) between a customer and a service provider. For performance-oriented SLAs, a common approach is to use mean value or plus variance of the response time. If the goal is set for some companies like Amazon.com to satisfy as many customers as possible rather than just the majority, a SLA with mean (even with variance) guarantee however does not yield to good user experiences. Amazon [3] proposes to express and measure SLAs at the 99.9th percentile of the distribution. The choice for 99.9% over an even higher percentile has been made based on a cost-benefit analysis which demonstrated a significant increase in cost to improve performance that much. Experiences with Amazon’s production systems have shown that this approach provides a better overall experience compared to those systems that meet SLAs defined based on the mean. This motivates this research to study SLA with certain level percentile guarantees. To meet the performance constraint, the system has to guarantee that some percentile of arriving jobs are processed under a certain response time threshold.

To reduce the power consumption while maintaining a given SLA, power-aware and energy-efficient designs have been extensively explored in the literature. Operating with lower power consumption instead of the maximum power consumption indeed helps, but it has been shown that the electricity cost still remains significant [4]. On one hand, we would like to keep performance boosting to ensure the customer satisfactory. On the other hand, the power cost is expected to be as less as possible to reduce the total maintenance cost. There is a trade-off between the power consumption and the performance. Dynamic Voltage/Frequency Scaling (DVFS) has been widely adopted to study this trade-off. Specifically, for web servers, DVFS was applied to reduce the energy consumption with the minimal performance impact in [5], [6]. In [7], a queueing theoretic model was used to predict the optimal power allocation in a variety of scenarios with DVFS. An optimal speed scaling was investigated in [8] to balance the mean energy consumption and mean response time under processor sharing scheduling.

Unfortunately, as shown in [9]–[11], depending on the architecture, the static power consumption when a server is idle could be still significant, up to 60% of the peak power consumption. Furthermore, if the power dissipation of power delivery and cooling sub-systems is also counted, the idle power consumption could increase by up to 50~100% [12]. Even though a server might sometimes be fully utilized, average server utilization is only 20~30% in typical data centers [5], [9], [13]. Therefore, one can also apply Dynamic Power Management (DPM) to turn the server to a sleep mode for reducing the energy consumption. There are two different strategies in applying DPM: transient or long term. With the transient strategy, a server will be put into sleep for a short period of time during idle time, such as the PowerNap in [13] and the PowerSleep in our previous work [14]. An effective DPM scheme should consider when to wake the dormant server up and when to turn the server to the sleep mode. In [14], DVFS and DPM were considered jointly to decide which frequency to execute requests at and when to put the server into sleep mode. With the long-term strategy, the server will be
deactivated into a deep sleep for a long period of time, such as the server consolidation approach [15], where multiple servers will run on one machine through virtual machine technology.

In most of the work in the literature, the SLAs are expressed with either the mean value of the response time in server systems [13], [14] or the hard deadline guarantee in real-time systems [16], [17]. Modeling servers as hard real-time systems can guarantee the worst-case response time, but it is usually too pessimistic. To meet the percentile guarantee, the approaches in the literature to meet the mean response time constraint cannot be applied. Closely related to this paper, Bertini et al. [18] use deadline missing probability as the performance metrics in web server clusters and propose an online feedback control scheme to control the QoS. However, the study in [18] does not provide any statistical guarantee even if the workload distribution is known. Different from [18], in this paper, we investigate queueing-theoretical power-saving design strategies for server farms under a given SLA that is measured in a certain level percentile of the job response time. This is more appealing as we discussed above.

We make the following contributions in this paper:

- To the best of our knowledge, there is no previous work on this topic to minimize the expected power consumption while providing the percentile guarantee. We propose PowerTail design strategy, which applies DPM to activate/deactivate servers and DVFS to choose the execution speed for activated servers. We adopt an M/G/1/PS server model together with heavy-tailed job service time distribution. With this design, we are able to minimize the overall power consumption of a server farm under the given SLA.

- To show the effectiveness, we consider the load-balancing strategy as the baseline scheme. We evaluate both PowerTail and this baseline scheme on real HTTP trace. Our results show that PowerTail significantly outperforms the baseline scheme.

- PowerTail is easy to be implemented in server farms due to its simplicity.

The rest of this paper is organized as follows: Section II describes the system model. In Section III, we present a detailed analysis of power consumption and response time. The PowerTail design strategy is introduced in Section IV. The optimal parameters required in PowerTail are derived in Section V. Section VI presents performance evaluation. Finally, we conclude the paper in Section VII.

II. SYSTEM MODEL

We consider a server farm, which includes \( n \) homogeneous servers, as shown in Figure 1. The front end monitors the requests from the clients and analyzes the request arrival rate for the next control period. Then, the dispatcher at the front end will distribute the job requests in the front-end queue to the back-end queue at servers according to a load allocation policy. The arrival rate can change over time as we will see in Section VI. However, we could assume a Poisson arrival [7], [8], [19], [20] of job requests at the front-end queue of the system with an arrival rate \( \lambda_F \) during a control period.

To exploit the mechanism to activate and deactivate back-end servers dynamically, the control period is assumed to be sufficiently long to pay off the energy and timing overhead of the activation and deactivation of servers.

In this work, we need to determine the load allocation policy to minimize the overall mean power consumption under the given SLA. We also assume all jobs in each back-end queue are served with Process Sharing (PS) job scheduling algorithm, where PS approximates very well the Round-Robin job scheduling algorithm used in Linux.

Once a server finishes serving a job, it will send a feedback to the corresponding client.

We assume that the system is equipped with DVFS and DPM for power management. A subset of servers will be activated by the DPM. We denote \( n_a \) as the number of the activated servers. For an activated server, with the DVFS we can choose an execution speed (with a corresponding choice of the supply voltage) to serve jobs. We define \( r \) as the ratio of its execution speed to its maximum speed. The speed ratio \( r \) is bounded by a lower bound \( r_i \). Then we have \( r_i \leq r \leq 1 \).

When a server is activated, it is either (i) in the running mode executing jobs whenever its back-end queue is not empty, or (ii) in the idle mode at the lowest speed ratio \( r_f \) without executing any job whenever its back-end queue is empty.

In the literature of server performance, it has been shown by different research studies that M/G/1/PS server model that can model the modern web servers well [7], [8], [19], [20]. In this paper, we adopt an M/G/1/PS server model for an activated server. We assume that the requests to each activated server still follow a Poisson arrival with an arrival rate \( \lambda \), which is determined by the load allocation policy. We assume that jobs follow a generalized service time distribution with a given mean value \( E[S] \) when executing at the maximum speed. We denote \( \mu \) as the service rate of a single server. Then the mean job service time under the maximum speed is \( E[S] = \frac{1}{\mu} \). If the server runs at a speed ratio \( r \) in the running mode, we have \( E[S] = \frac{1}{r\mu} \). We denote \( \rho \) as the (absolute) server utilization with respect to the maximum speed, i.e.,

\[
\rho = \frac{\lambda}{\mu}.
\]

Then the relative server utilization with respect to the speed ratio \( r \) can be written as \( \lambda E[S] = \frac{\rho}{r} \).

\[1\]Our methodology can be applied to other job scheduling algorithms such as First-Come-First-Served (FCFS) as well.
In this work, we aim to choose a certain-level percentile of the response time as the SLA instead of the mean. Therefore we need more information on the job size in addition to the mean service time $E[S]$. It has been widely recognized that heavy-tailed distributions are suitable for modeling job sizes in information service networks [21]–[24]. In particular, in [24], it has been shown that web file sizes and HTTP request times follow heavy-tailed distributions. Therefore we adopt heavy-tailed distribution for the job service time. In particular, we choose the Pareto distribution.$^2$ By the definition of Pareto distribution [25], we assume that the job service time $S$ is lower-bounded by $\hat{x}$ when the server runs at the maximum speed, and the tail of $S$ is

$$
\mathbb{P}\{S \geq x\} = \left(\frac{x}{\hat{x}}\right)^{-\nu} \tag{2}
$$

as $x \geq \hat{x}$, where $\nu \in (1, 2)$. If the job service time is less than $\hat{x}$, the tail is 1. With the heavy-tailed distribution, we will more likely have a longer job service time than the light-tailed. From (2), we can obtain the mean value as $E[S] = \frac{1}{1-\nu} \hat{x}$. Therefore, we have the following equality: $\hat{x} = \frac{E[S]}{1-\nu}$. Hence $\mu$ is uniquely determined by $\hat{x}$. When the server runs at the speed ratio $r$, the tail of $S$ should be re-written as

$$
\mathbb{P}\{S \geq x\} = \left(\frac{rx}{\hat{x}}\right)^{-\nu} \tag{3}
$$

as $rx \geq \hat{x}$.

The power consumption in our study is the system-level power, including the power consumed by the processor and all other components within the server. When the server is deactivated by the DPM, its power consumption is negligible. For an activated server equipped with DVFS, its power consumption depends on the mode the server is in (running or idle), and also the execution speed in use. In this paper, we adopt the model in [7]. The activated server has the following power modes:

- **Idle power mode**: In the idle mode, the server consumes the static power $P_I$;
- **Running power mode**: In the running mode, the power consumption $P_{R}(r)$ by the server at a speed ratio $r$ is

$$
P_{R}(r) = \alpha[r - r_I]^{\gamma} + P_I, \tag{4}
$$

where $\gamma \geq 1$. The cubic rule is widely suggested in the literature for the processor power-to-speed relationship in the running mode, i.e., $\gamma = 3$. However, in server farms with DVFS or for some applications, the linear rule could be applied.

A design strategy should address how many servers should be activated by the DPM and what speed ratio should be chosen by the DVFS for each activated servers. Note that, we assume that the control period is sufficiently long to compensate the energy consumption of the activation and deactivation energy overhead of servers. The timing overhead to activate and deactivate a server is assumed to be hidden in the control period. That is, if the front end requires to change the system configuration at time $t$, the activation request of a server should be issued at $t - \delta$, in which $\delta$ is the upper bound of the timing overhead to active a server.

### III. Power Consumption and Response Time Analysis for an Activated Server

Recall that our objective is to minimize the power consumption under the given response time SLA. First we conduct the analysis of power consumption and response time for an activated server based on the system model in Section II.

#### A. Power Consumption

We consider an activated server that will switch in two power modes alternatively: running and idle. We define $\pi_R$ and $\pi_I$ as the probabilities that the server is in running and idle mode respectively, where $\pi_R + \pi_I = 1$, then its mean power consumption can be written as:

$$
E[P] = P_{R}(r)\pi_R + P_I\pi_I. \tag{5}
$$

According to (5), in order to calculate the mean power, we need to obtain the value of $\pi_R$. For an M/G/1/PS server with a speed ratio $r$ at the running state, an arrival rate $\lambda$, and a mean service time $E[S] = \frac{1}{\mu}$, by the traditional queueing theory [26] we have

$$
\pi_R = \lambda E[S] = \frac{\rho}{r}, \tag{6}
$$

where $\rho$ is the absolute server utilization defined in (1). Applying (4) and (6) into (5), we have the following theorem regarding the power consumption:

**Theorem 1**: In an M/G/1 server with an absolute server utilization $\rho$ and a speed ratio $r$ at the running state, its mean power consumption is

$$
E[P] = \alpha[r - r_I]^{\gamma} \frac{\rho}{r} + P_I. \tag{7}
$$

$E[P]$ defined in (7) is an increasing function of $r$ as $r_I \leq r \leq 1$ and of $\rho$ too.

#### B. Response Time Analysis

The response time (also called the sojourn time) is defined as the total time spent by a job in waiting in the queue and executing on the server. For an M/G/1/PS server with heavy-tailed service time distribution, we have the following result for the tail of the response time [27]:

**Lemma 1**: In an M/G/1/PS server running with the maximum speed at the running state where the job service time $S$ is heavy tailed, the tail of the response time $R$ can be asymptotically approximated as

$$
\mathbb{P}\{R \geq \tilde{R}\} \sim \mathbb{P}\{S \geq [1 - \rho]\tilde{R}\} \tag{8}
$$

as $\tilde{R} \to \infty$.

Lemma 1 shows that the response time is also heavy-tailed if the service time is heavy-tailed. With this lemma, given the tail of the service time, we are able to obtain the tail of the response time for a relatively large response time threshold $\tilde{R}$.
When the server runs at the speed ratio $r$ and the heavy-tailed distribution of the service time $S$ is defined as (3), then we apply (3) into Lemma 1, and obtain the tail of the response time as shown in the following theorem:

**Theorem 2:** In an M/G/1 server with a speed ratio $r$ at the running state and the heavy-tailed distribution of the service time $S$ is defined as (3), the tail of the response time can be asymptotically approximated as

$$P\{R \geq \hat{R}\} \sim P\{S \geq [1 - \frac{\rho}{r}]\hat{R}\} = \left[\frac{\hat{x}}{\hat{x}}\right]^{\nu}[r - \rho]^{-\nu}$$

as $\hat{R} \to \infty$.

In this work, we aim to minimize the overall mean power consumption of the server farm under a given response time SLA. The SLA is normally expressed and measured at a low bound of a certain level percentile (for example, 99.9-th percentile) of the distribution of the response time. We could also express the SLA in terms of the upper bound of its tail (for example, $1 - 0.999 = 0.001$ tail for 99.9th percentile). If we denote $\epsilon$ as the upper bound, the SLA can be expressed as

$$P\{R \geq \hat{R}\} \leq \epsilon,$$

where the value of $\hat{R}$ is usually relatively large in comparison to the mean value $E[S]$, and $\epsilon$ is relatively small. The SLA defined in (10) is contracted between the customer and the service provider. Applying (9) in (10), we have $[\frac{\hat{x}}{\hat{x}}]^{\nu}[r - \rho]^{-\nu} \leq \epsilon$. By denoting

$$\eta = \frac{x}{R}e^{-\frac{1}{\nu}},$$

the above equation can be re-written as

$$r \geq \rho + \eta.$$  (12)

(12) becomes the simplified version of the SLA constraint (10).

The analysis of power consumption and response time in this section establishes the foundations for the optimal design in the next section.

**IV. PowerTail: Optimal Design of Server Farm**

In this section, we will study the optimal design of a server farm and present our design strategy, called PowerTail. In order to achieve an optimal design, we introduce a new term, called *mean energy consumption per job*. It can be expressed mathematically as

$$Q(\lambda, r) = \frac{E[P]}{\lambda}.$$  (13)

In (13), $E[P]$ is the mean power consumption for a server, and $\lambda$ is the job arrival rate at the server. Therefore, $\frac{E[P]}{\lambda}$ is the mean energy consumed by a job.

Our strategy to achieve an optimal design is to minimize $Q$, the energy consumption per job, under a given SLA. The optimization problem can easily be formulated as follows:

$$\text{minimize } Q(\lambda, r)$$

subject to $P\{R \geq \hat{R}\} \leq \epsilon,$

$$\max\{r_1, \rho\} \leq r \leq 1.$$  (14c)

Inequality (14c) is based on the low bound of $r$ ($r_1 \leq r \leq 1$) and the stability condition of a server ($r > \rho$).

We define the optimal solution $(\lambda_c, r_c)$ obtained from (14) as the critical arrival rate and the critical speed ratio, i.e.,

$$(\lambda_c, r_c) = \arg\min_{(\lambda,r)} \{Q(\lambda, r) \text{ in (14)}\}.$$  (15)

With the above critical arrival rate and critical speed ratio, we propose our design strategy PowerTail, which is described as follows.

**PowerTail:** If $\lambda_F$ is divisible by $\lambda_c$ obtained in (15) and $n_a \leq n$, where $n_a = \left\lfloor \frac{\lambda_c}{\lambda_F} \right\rfloor$, we activate $n_a$ servers and evenly distribute the load into these $n_a$ servers. Each activated server is allocated with the exact critical arrival rate $\lambda_c$ and runs at the running state with the exact critical speed ratio $r_c$ obtained in (15).

The following theorem shows that PowerTail is an optimal design:

**Theorem 3:** PowerTail minimizes the overall mean power consumption of the server farm under the given SLA defined in (10).

**Proof:** With PowerTail, the overall job requests at the front-end queue is evenly distributed to $n_a$ activated homogeneous servers. Each activated server is allocated with the critical job arrival rate $\lambda_c$ and configured with the critical speed ratio $r_c$. The rest $n - n_a$ servers are deactivated and consume negligible power. Then the overall mean power consumption of the server farm is

$$n_aE[P] = \frac{\lambda_F}{\lambda_c}E[P] = \frac{\lambda_F}{\lambda_c}E[P] = \lambda_FQ(\lambda_c, r_c).$$  (16)

Since $Q(\lambda_c, r_c)$ in (16) is minimized, then the overall mean power consumption $n_aE[P]$ is also minimized.

Theorem 3 shows that minimizing the overall mean power consumption is equivalent to minimizing the mean energy consumption per job.

In Theorem 3, notice that we have assumed that $\lambda_F$ is divisible by $\lambda_c$ and $n_a \leq n$ holds. If $\lambda_F$ is not divisible by $\lambda_c$ and $n_a < n$, where $n_a = \left\lfloor \frac{\lambda_c}{\lambda_F} \right\rfloor$, the PowerTail should be revised as follows: We still allocate the critical arrival rate $\lambda_c$ into $n_a$ homogeneous servers and apply the critical speed ratio $r_c$ to them, and allocate the residue arrival rate $\lambda_F - n_a\lambda_c$ into an additional server and apply the critical speed ratio $r_c$. In this case of PowerTail, although the additional server does not achieve the optimal power consumption, we still prefer PowerTail over an optimal design strategy because of the following three reasons:

1) The difference between the resulting overall mean power consumption with PowerTail and the optimal overall mean power consumption is negligible when $n_a$ is relatively large (tens to hundreds or above of activated servers);

2) If $\lambda_F > n\lambda_c$, it means that the servers are not enough to achieve the optimal configuration with PowerTail. In this case, load balancing with even job allocation to all servers is the optimal configuration, i.e., we choose $\frac{\lambda_c}{\lambda_F}$ as the arrival rate for each server. We can easily obtain an optimal speed ratio for each server.
2) The method used in PowerTail – minimizing mean energy consumption per job – is very elegant. Instead of solving the optimization for the whole server farm, we only focus on a single server;

3) PowerTail is much easier to implement than the optimal design strategy. As the arrival rate $\lambda_F$ at the front end changes, we only need to activate or deactivate more servers without changing the running speed ratio of activated servers as long as we have enough available servers. However, with the optimal design strategy, for any change of $\lambda_F$, we need to re-solve the optimization problem and obtain a new arrival rate and a new speed ratio for each activated server, and then re-configure these parameters in the system.

The key in PowerTail design is to obtain the critical arrival rate $\lambda_c$ and the critical speed ratio $r_c$ defined in (14), which is the focus of the next section.

V. OBTAINING CRITICAL UTILIZATION AND CRITICAL SPEED RATIO IN POWERTAIL

We observe that the mean energy consumption per job is focused on a single server and the utilization $\rho$ in a single server is easier to picture than the arrival rate $\lambda$ such as the boundaries of $\rho$. In the following, we replace $\lambda$ with $\rho\mu$ based on (1) and also $Q(\lambda, r)$ with $Q(\rho, r)$. We introduce the critical utilization $\rho_c$ to replace the critical arrival rate $\lambda_c$.

Based on the power consumption in (7) and the simplified SLA constraint (12), the optimization problem (14) can be re-written as

$$\begin{align*}
& \text{minimize } Q(\rho, r) = \frac{1}{\mu} \left( \frac{\alpha (r - r_t)}{r} \right)^\gamma + \frac{P_l}{\rho} \\
& \text{subject to } \max\{r_t, \rho + \eta\} \leq r \leq 1.
\end{align*}$$

(17a) (17b)

The detailed discussion of deriving optimal $Q(\rho, r)$ in the above optimization problem can be found in Appendix A.

A. Main Result

The following theorems summarize the main results of the critical utilization $\rho_c$ and its corresponding speed ratio $r_c$ for both $\gamma = 1$ and $\gamma > 1$:

**Theorem 4 (linear-power servers):** The critical utilization $\rho = \rho_c$ and the critical speed ratio $r_c$ for linear-power servers ($\gamma = 1$) are obtained as

$$\begin{align*}
(\rho_c, r_c) = \begin{cases} \\
(\rho_m, \max\{r_t, \eta\}), & \text{if } 0 < \xi < r_t \\
(\rho^*, \max\{r_t, \xi\}), & \text{if } r_t \leq \xi < 1 \\
(\rho_r, 1), & \text{if } \xi \geq 1 \lor \xi \leq 0
\end{cases}
\end{align*}$$

(18)

where $\rho_m$ and $\rho_r$ are defined in (30) and (27) respectively, $\xi$ and $\rho^*$ are defined respectively as

$$\xi = \frac{\eta}{1 - \frac{P_l}{\alpha r_t}}, \quad \text{and} \quad \rho^* = \xi - \eta.$$  

(19)

The proofs of Theorems 4 and 5 can be found in Appendices B and C respectively.

B. Remarks

Note that the control period is assumed sufficiently long to compensate the energy consumption of the activation and deactivation energy overhead of servers. If the control period of the front end for estimating the arrival rate is long, to compensate the bursty request arrivals, which is unpredicted, we need to spare some idle servers to handle this issue. In PowerTail, the strategy is to spare one back-end server, which can be used to accommodate the unpredicted requests. The strategy can be easily extended to allow the designers to spare more servers. However, the more spare servers are used for accommodating bursty requests, the more additional energy the system consumes.

Even though we focus in our study by far on the PS job scheduling policy, the proposed methodology still work for other scheduling policies. We observe that in the analysis for PowerTail, the power consumption analysis is based on M/G/1 queueing model and is independent of the underlying scheduling policy. This will not hold for the response time analysis. We need to make some corresponding changes. For instance, for FCFS scheduling policy, it has been shown [28] that the tail of the response time is regularly varying of index $1 - \nu$ if the service time is regularly varying of index $-\nu$ as shown in (2). The tail probability of the response time in the server running at the maximum speed can be written as

$$\mathbb{P}\{R \geq \hat{R}\} \sim \frac{\rho}{r - \rho \nu - 1} \hat{R}^{1 - \nu}$$

as $\hat{R} \to \infty$. If the server runs at the speed ratio $r$, then we have

$$\mathbb{P}\{R \geq \hat{R}\} \sim \frac{\rho}{r - \rho \nu - 1} \hat{R}^{1 - \nu}.$$  

(23)

as $\hat{R} \to \infty$. 

---

\footnote{The power-law characteristics of the job service time does not change dramatically on servers. Otherwise, we need to reconfigure all these parameters.}

\footnote{If the denominator of $\xi$ in (19) is zero, we assume $\xi = +\infty$ and include it in the case of $\xi > 1$.}
In the evaluation, we will consider an intuitive baseline design strategy, called PowerEven. With PowerEven, the load is evenly allocated to all servers. Therefore, the arrival rate at each server is \( \frac{\lambda_F}{n} \). The absolute utilization for each server is

\[
\rho_{PB} = \frac{\lambda}{\mu} = \frac{\lambda_F}{n\mu}.
\]

We need to determine the speed ratio \( r \) to minimize the mean power consumption under the given SLA. The optimization problem can be formulated as

\[
\begin{align*}
\text{minimize } & \mathbb{E}[P] = \alpha[r - r_t] \frac{\rho_{PB}}{r} + P_I \\
\text{subject to } & \max\{r_t, \rho_{PB} + \eta\} \leq r \leq 1.
\end{align*}
\]

Since \( \mathbb{E}[P] \) is an increasing function of \( r \), the minimum power consumption will be achieved at \( r_{PB} = \rho_{PB} + \eta \), where \( \rho_{PB} \) is defined in (24).

VI. PERFORMANCE EVALUATION

In this section, we first verify the heavy-tailed phenomenon of the job response time for heavy-tailed job service time as shown in Lemma 8. Then we present performance evaluation of our proposed PowerTail design.

A. Verifying Heavy-Tailed Phenomenon

In order to emulate job service time, we download a real 24-hour HTTP trace of accesses on January 20th, 2011 from IRCache system [29]. The trace is with a minimum arrival rate 5.4 per second and a maximum rate 238.5 per second. We consider the first one-minute trace at the 8th hour with 10206 requests and the arrival rate 170.8 per second. Based on [30], the service time of a job request is proportional to the size of the requested file on web servers. We adopt the power-law distribution fitting tool in [31] to obtain the parameters \( \nu \) and \( \hat{x} \) of Pareto distribution for the service time. Figure 2 shows the power-law distribution fitting of the one-minute trace with \( \nu = 1.8154 \) and \( \hat{x} = 0.013 \). Then we obtain \( \mu = 34.5506 \).

The purpose of the following experiment is to verify the asymptotical approximation of the job response time in Theorem 2. In our verification, we sample every one of five jobs in the one-minute trace. We scale the job service time so that we have \( \mu = 34.5506 \) per second at the maximum speed on the server. We consider a Poisson arrival with average arrival rate 8 per second and it lasts for 4 minutes. We run it on an Intel i3 dual core processor based server with 6GB of RAM. The server is equipped with DVFS capability with option to switch between 16 frequencies between 1.6GHz and 3.1GHz. We disabled multicore and hyperthreading. We choose two different frequencies: 2.5GHz and 3.1GHz to examine the response time performance. The result is shown in Figure 3. Both cases confirm that the tail of the job response time can be asymptotically approximated as a function of the tail of the job service time as shown in Theorem 2.

B. Evaluating PowerTail

We use three types of servers to evaluate the performance of PowerTail: the one used in the previous experiment, and the other two from [7]. We measure the power consumption for the one used in the previous experiment. Based on the measured power, we obtain the modeled power as shown in Figure 4. The power profiles of all three types of servers in Table I, where Type-B is the one used in the previous experiment.

We compare PowerTail with a baseline design strategy PowerEven, which applies the load balancing allocation policy. We use the same job profile of the one-minute except that the job size is scaled in each type of server with \( \mu = 34.5506 \) per second at the maximum speed. For fair comparison, we need to decide the number of required servers for a peak demand: We consider a worst-case arrival rate 300.0 per second, and fix \( \hat{x} = 0.013 \) sec, \( \frac{\hat{R}}{\hat{F}} = 0.02 \), and \( \epsilon = 0.001 \), then the minimum number of required servers under PowerTail for all three types of servers is 86.\(^5\) We assume that both PowerTail and PowerEven use 86 servers.

There are three parameters to evaluate the performance of PowerTail: the arrival rate \( \lambda_F \) at the front-end queue, the tail upper-bound \( \epsilon \), and the ratio of the minimum response time to the guaranteed maximum response time \( \frac{R}{F} \). Both \( \epsilon \) and \( \frac{R}{F} \) will determine the value of \( \eta \) defined in (11). In the following, we will fix two of these parameters and vary the other. Based on these settings, we compare PowerTail with PowerEven.

\(^5\)Recall that the job size is scaled in each type of server with \( \mu = 34.5506 \) per second at the maximum speed, therefore the total numbers of servers under the peak demand are the same.
In Figure 5, we observe that $R$ decreases. That means we have a tight response time for Type-A, Type-B, and Type-C servers significantly. For example, when $\lambda = 0.001$, we obtain the mean power consumption of the server with $\text{PowerTail}$ and $\text{PowerEven}$ under all three types of servers. The result is shown in Figure 5. The corresponding critical speed ratio and the critical utilization with $\text{PowerTail}$ are shown in Figure 6.

In Figure 5, we observe that $\mathbb{E}[P]$ increases with respect to $\frac{P}{R}$ for both $\text{PowerTail}$ and $\text{PowerEven}$. However, $\text{PowerTail}$ outperforms $\text{PowerEven}$ significantly. For example, when $\lambda = 0.006$, $\text{PowerTail}$ consumes only 10.1%, 8.9%, and 12.0% of power by $\text{PowerEven}$ for Type-A, Type-B, and Type-C servers respectively. In this case, $\text{PowerTail}$ only activates 7, 7, and 8 servers out of 86 servers respectively. As $\frac{P}{R}$ increases, for a fixed $\frac{\hat{x}}{R}, \hat{R}$ decreases. That means we have a tight response time requirement. In order to achieve this goal, we need to increase the processing capacity. For instance, we activate more servers, which results in more power consumption. Eventually, all servers will be activated, in that case, $\text{PowerTail}$ will be same as $\text{PowerEven}$.

For $\gamma = 1$, we have $\alpha r_1 \leq P_f$ and hence $\xi < 0$ by (19). Based on Theorem 4, $r_c = 1$ holds always, which is observed in Figure 6(a). For $\gamma = 1.2137$, we have $\xi < 0$ by (19). Therefore, based on Theorem 5, $r_c = 1$, which is also observed in Figure 6(b). For $\gamma = 3$, $\chi > 0$ as $\frac{\hat{x}}{R} < 0.0085$ and $\chi < 0$ as $\frac{\hat{x}}{R} > 0.0085$. Therefore, based on Theorem 5, $r_c = 1$ as $\frac{\hat{x}}{R} > 0.0085$, which is also observed in Figure 6(c). In Figure 6, we also observe that $\rho_c$ is a (piecewise) linear function of $\frac{\hat{x}}{R}$, which is based on Theorems 4 and 5 and (11).

1) Varying $\frac{\hat{x}}{R}$. In this case, we assume the arrival rate $\lambda_F = 170.8$ per second. We choose $\epsilon = 0.001$. Based on the feasible region of $\frac{\hat{x}}{R}$, given the value of $\epsilon$, $\frac{\hat{x}}{R}$ is upper-bounded by $\epsilon = 0.0223$ and it could not reach the upper bound. In our evaluation, we fix $\frac{\hat{x}}{R} = 0.013$ sec and vary $\frac{\hat{x}}{R}$ from 0.00001 to 0.02. We obtain the mean power consumption of the server farm with $\text{PowerTail}$ and $\text{PowerEven}$ under all three types of servers. The result is shown in Figure 5. The corresponding critical speed ratio and the critical utilization with $\text{PowerTail}$ are shown in Figure 6.

In this case, we assume the arrival rate $\lambda_F = 170.8$ per second. We choose $\epsilon = 0.001$. Based on the feasible region of $\frac{\hat{x}}{R}$, given the value of $\epsilon$, $\frac{\hat{x}}{R}$ is upper-bounded by $\epsilon = 0.0223$ and it could not reach the upper bound. In our evaluation, we fix $\frac{\hat{x}}{R} = 0.013$ sec and vary $\frac{\hat{x}}{R}$ from 0.00001 to 0.02. We obtain the mean power consumption of the server farm with $\text{PowerTail}$ and $\text{PowerEven}$ under all three types of servers. The result is shown in Figure 5. The corresponding critical speed ratio and the critical utilization with $\text{PowerTail}$ are shown in Figure 6.

In Figure 5, we observe that $\mathbb{E}[P]$ increases with respect to $\frac{P}{R}$ for both $\text{PowerTail}$ and $\text{PowerEven}$. However, $\text{PowerTail}$ outperforms $\text{PowerEven}$ significantly. For example, when $\lambda = 0.006$, $\text{PowerTail}$ consumes only 10.1%, 8.9%, and 12.0% of power by $\text{PowerEven}$ for Type-A, Type-B, and Type-C servers respectively. In this case, $\text{PowerTail}$ only activates 7, 7, and 8 servers out of 86 servers respectively. As $\frac{P}{R}$ increases, for a fixed $\frac{\hat{x}}{R}, \hat{R}$ decreases. That means we have a tight response time requirement. In order to achieve this goal, we need to increase the processing capacity. For instance, we activate more servers, which results in more power consumption. Eventually, all servers will be activated, in that case, $\text{PowerTail}$ will be same as $\text{PowerEven}$.

For $\gamma = 1$, we have $\alpha r_1 \leq P_f$ and hence $\xi < 0$ by (19). Based on Theorem 4, $r_c = 1$ holds always, which is observed in Figure 6(a). For $\gamma = 1.2137$, we have $\xi < 0$ by (19). Therefore, based on Theorem 5, $r_c = 1$, which is also observed in Figure 6(b). For $\gamma = 3$, $\chi > 0$ as $\frac{\hat{x}}{R} < 0.0085$ and $\chi < 0$ as $\frac{\hat{x}}{R} > 0.0085$. Therefore, based on Theorem 5, $r_c = 1$ as $\frac{\hat{x}}{R} > 0.0085$, which is also observed in Figure 6(c). In Figure 6, we also observe that $\rho_c$ is a (piecewise) linear function of $\frac{\hat{x}}{R}$, which is based on Theorems 4 and 5 and (11).

2) Varying $\epsilon$. In this case, we still assume the arrival rate $\lambda_F = 170.8$ per second. We choose $\frac{\hat{x}}{R} = 0.001$. Based on the feasible region of $\frac{\hat{x}}{R}$, given the value of $\epsilon$, $\frac{\hat{x}}{R}$ is lower-bounded by $\frac{\hat{x}}{R} = 3.6e^{-6}$ and it could not reach the lower bound. In our evaluation, we vary $\epsilon$ from $4e^{-6}$ to 0.1. We obtain the mean power consumption of the server farm with $\text{PowerTail}$ and $\text{PowerEven}$ under all three types of servers. The result is shown in Figure 7.

Based on Theorems 4 and 5, $\text{PowerTail}$ design depends on the value of $\eta$. By (11), $\eta$ is an increasing function of $\frac{\hat{x}}{R}$, but a decreasing function of $\epsilon$. Therefore, we observe that in Figure 7 $\mathbb{E}[P]$ decreases with respect to $\epsilon$ for both $\text{PowerTail}$ and $\text{PowerEven}$. $\text{PowerTail}$ still outperforms $\text{PowerEven}$ significantly. For example, when $\epsilon = 0.001$, $\text{PowerTail}$ consumes only 9.8%, 7.8%, and 10.2% of power by $\text{PowerEven}$ for Type-A, Type-B, and Type-C servers respectively. In this case, $\text{PowerTail}$ only activates 6, 6, and 7 servers out of 86 servers respectively. Due to the symmetricity of this case with the previous one, the reason behind it can be drawn easily. We
also skip the result of the critical speed ratio and the critical utilization.

3) Varying $\lambda$: In this case, we set $\frac{\lambda}{R} = 0.01$ and $\epsilon = 0.001$. We vary $\lambda$ from 0 per second to 300 per second. We obtain the mean power consumption of the server farm with PowerTail and PowerEven under all three types of servers. The result is shown in Figure 8.

For the given $\frac{\lambda}{R}$ and $\epsilon$, by (11) we have $\eta = 0.0449$. Since $\xi = -0.0401$ for $\gamma = 1$, $\xi = -32.4906$ for $\gamma = 1.2137$, and $\chi = 228.6752$ for $\gamma = 3$, based on Theorems 4 and 5, the critical speed ratio and the critical utilization under PowerTail are $r_c = 1$ and $\rho_c = 0.9551$ for Type-A and Type-B servers and $r_c = 0.8243$ and $\rho_c = 0.7794$ for Type-C servers respectively.

The overall mean power consumption is a almost-linear function of $\lambda$ under PowerTail. The steps in the curve is due to the server with non-optimal configuration. PowerTail aim to activate more servers for a higher arrival. For PowerEven, since all servers are always activated, each server will consume the power in the idle mode, which is relatively large.

VII. CONCLUSION

In this paper, we studied power consumption optimization in server farm under response time SLA. The SLA in study was measured at a certain level percentile or tail of the job response time, which is more appealing than the commonly-used mean value. We adopted an M/G/1/PS server model, where the job service time distribution is assumed heavy-tailed. We proposed a simple and elegant design strategy PowerTail. It is based on our observation that minimizing the overall mean power consumption under a given SLA is equivalent to minimizing the mean energy consumption per job. We investigated how to find the optimal critical utilization and speed ratio setting for PowerTail design. Our evaluation results have showed that our PowerTail strategy significantly outperforms the intuitive load-balancing strategy PowerEven.

REFERENCES

We observe that where \( \rho \) of \( r \) as a function of \( Q \) in (29). In order to remove this max operator, we divide the \( Q \) at the critical speed \( r \)

\[ r \leq \frac{c}{l} \] \( \rho \)

Based on (17b), the feasible value of \( \rho \) should satisfy:

\[ 0 \leq \rho \leq \rho_r, \]

where \( \rho_r \) is the upper bound of the feasible \( \rho \) and defined as

\[ \rho_r \leq 1 - \eta. \]

In order to obtain the optimal \( Q = Q(\rho, r) \), we first consider \( Q \) as a function of \( r \). We define its first partial derivative:

\[ \frac{\partial Q}{\partial r} = \frac{\alpha}{\mu} \frac{[r - r_l]^{\gamma - 1} [\gamma - 1] r + r_l}{r^2}. \]

We observe that \( \frac{\partial Q}{\partial r} > 0 \) holds for any feasible \( r \). Therefore \( Q \) is an increasing function of \( r \). The minimal \( Q \) will be achieved at the critical speed \( r_c \), which is the lower bound of the feasible \( r \). Based on (17b), we have

\[ r_c = \max\{r_l, \rho + \eta\}. \]

We apply (29) into (17a) and then \( Q \) becomes a function of \( \rho \) only. Next, we need to find out an optimal \( \rho \) to minimize \( Q \), which is defined as the critical utilization \( \rho_c \). Notice that \( r_c \) that was just applied in (17a) has a max operator as shown in (29). In order to remove this max operator, we divide the feasible region of \( \rho \) in (26) into the following sub-intervals:

\[ [0, \rho_m] \text{ and } [\rho_m, \rho_r], \]

where

\[ \rho_m \triangleq \max\{0, r_l - \eta\}. \]

We have \( r_l \geq \rho + \eta \) in \([0, \rho_m] \) and \( r_l \leq \rho + \eta \) in \([\rho_m, \rho_r]\).

In the sub-interval \([0, \rho_m]\), by (29) we obtain the critical speed ratio as \( r_c = r_l \). Applying it in (17a), we obtain the minimum \( Q \) as

\[ Q = \frac{P_l}{\mu \rho}. \]

In order to obtain the minimal \( Q \) in (32), we study its first partial derivative, which can be obtained as

\[ \frac{\partial Q}{\partial \rho} = \frac{H(\rho)}{[\rho + \eta]^2}, \]

where

\[ H(\rho) = \alpha [\rho - \rho_m]^{\gamma - 1} [\gamma - 1] [\rho + \eta] + r_l - P_l [1 + \frac{\eta}{\rho}]^2. \]

In order to find the minimum \( Q \), we need to study the properties of \( H(\rho) \): If \( H(\rho) \geq 0 \), then \( \frac{\partial Q}{\partial \rho} \geq 0 \) and hence \( Q \) defined in (32) is increasing; If \( H(\rho) \leq 0 \), then \( Q \) is decreasing; If there exists a \( \rho^* \) such that \( H(\rho^*) = 0 \), then \( Q \) will have a critical value at \( \rho = \rho^* \). Based on these cases, we can obtain the critical utilization \( \rho_c \).

**APPENDIX B**

**PROOF OF THEOREM 4 WITH \( \gamma = 1 \)**

We set \( \gamma = 1 \) in (34), then we have

\[ H(\rho) = \alpha r_l - P_l [1 + \frac{\eta}{\rho}]^2. \]
Consider $\xi$ defined in (19). If $\xi \leq 0$, then $\alpha r_1 < P_1$ and hence $H(\rho) < 0$ holds always. In this case, by (33) $Q$ is always decreasing. Then the critical utilization will fall at the right boundary, i.e., $\rho_c = \rho_r$, as demonstrated in first plot of Figure 9(a).

If $\xi > 0$, then $\alpha r_1 > P_1$. $H(\rho) \geq 0$ means $\rho \geq \rho^*$ and $H(\rho) \leq 0$ means $\rho \leq \rho^*$, where $\rho^*$ is defined as

$$\rho^* = \frac{\eta}{(\frac{\partial Q}{\partial \rho})^2 + 1} = \xi - \eta. \quad (36)$$

We consider the following cases:

- If $0 < \xi < r_1$, then $\rho^* < \rho_m$ and hence $H(\rho) \geq 0$ holds for $\rho \in [\rho_m, \rho_r]$. In this case, $Q$ is increasing in this sub-interval. The minimum $Q$ is achieved at the left boundary, i.e., $\rho_c = \rho_m$, as demonstrated in second plot of Figure 9(a).

- If $r_1 \leq \xi < 1$, then $\rho_m \leq \rho^* \leq \rho_r$. $H(\rho) \leq 0$ holds for $\rho \in [\rho_m, \rho^*]$ and $H(\rho) \geq 0$ holds for $\rho \in [\rho^*, \rho_r]$. Hence $Q$ is decreasing first and then increasing in this sub-interval. The minimum $Q$ is achieved inside the sub-interval, i.e., $\rho_c = \rho^*$, as demonstrated in third plot of Figure 9(a), where $\rho^*$ is defined in (36).

- If $\xi \geq 1$, then $\rho^* > \rho_r$ and hence $H(\rho) \leq 0$ holds for $\rho \in [\rho_m, \rho_r]$. In this case, $Q$ is always decreasing in this sub-interval. The minimum $Q$ is achieved at the right boundary, i.e., $\rho_c = \rho_r$, as demonstrated in first plot of Figure 9(a).

With the above obtained critical utilization $\rho_c$, based on (29), we can also obtain the critical speed ratio $r_c$. We can summarize the above observations in Theorem 5.

**APPENDIX C**

**PROOF OF THEOREM 5 WITH $\gamma > 1$**

We consider the left boundary of the sub-interval $[\rho_m, \rho_r]$, i.e., $\rho = \rho_m$. In this case, we have

$$H(\rho_m) = -P_1 \left[\frac{T_1}{\rho_m}\right]^2 < 0. \quad (37)$$

From (34), we also observe that $H(\rho)$ is strictly increasing in the sub-interval $[\rho_m, \rho_r]$. We consider the right boundary of the sub-interval $[\rho_m, \rho_r]$, i.e., $\rho = \rho_r$.

Consider $\chi$ defined in (21). If $\chi \leq 0$, then $H(\rho) \leq 0$ holds through the sub-interval $[\rho_m, \rho_r]$. By (33) we know that $\frac{\partial Q}{\partial \rho} \leq 0$ also holds through $[\rho_m, \rho_r]$. Hence $Q$ is always decreasing in this sub-interval. The minimum $Q$ is achieved at $\rho_c = \rho_r$, as demonstrated in first plot of Figure 9(b).

If $\chi > 0$, then we the following observations: $H(\rho_m) < 0$, $H(\rho_r) > 0$, and $H(\rho)$ is increasing in the sub-interval $[\rho_m, \rho_r]$. Therefore there exists one and only one $\rho = \rho^*$ such that $H(\rho^*) = 0$ holds. Then $H(\rho) \leq 0$ holds for $\rho \in [\rho_m, \rho^*]$ and $H(\rho) \geq 0$ holds for $\rho \in [\rho^*, \rho_r]$. Hence, by (33) $Q$ is decreasing first and then increasing in this sub-interval as demonstrated in second plot of Figure 9(b). The minimum $Q$ is achieved inside the sub-interval, i.e., $\rho_c = \rho^*$, which is defined in (22).

With the above obtained critical utilization $\rho_c$, based on (29), we can also obtain the critical speed ratio $r_c$. We can summarize the above observations in Theorem 5.