

# PACMAN: Power Usage-Aware Capping and Management for All-Flash Storage Servers

Bodon Jeong<sup>1</sup>, Hongsu Byun<sup>1</sup>, Kyungkeun Lee<sup>2</sup>, Bumjun Kim<sup>2</sup>, Jeong-Uk Kang<sup>2</sup>, and Sungyong Park<sup>1,†</sup>

<sup>1</sup>Sogang University, Seoul, Republic of Korea, <sup>2</sup>Samsung Electronics Co.

{bd91jeong, byhs, parksy}@sogang.ac.kr, {kavin.lee, bj20.kim, ju.kang}@samsung.com

**Abstract**—Recently, power management has become increasingly important in storage-centric infrastructures, where SSD arrays are emerging as major power consumers. In such environments, static storage power capping (SSPC) has been used to reduce the peak power consumption of SSD arrays by fixing the power states of individual devices, aiming to stay within constrained power budgets. However, SSPC does not account for workload I/O characteristics or internal SSD behaviors such as garbage collection. This often leads to over-capping, which degrades performance, or underutilization of available power. To address these limitations, we propose PACMAN, a power usage-aware capping and management framework designed for the storage layer in all-flash storage servers. PACMAN dynamically reallocates SSD power caps in real time through predictive adjustments based on recent per-device power usage. Without relying on performance counters, PACMAN efficiently adapts to workload variability and optimizes performance and energy efficiency within a given power budget. PACMAN has been extensively evaluated using both the FIO benchmark and a real-world application, RocksDB. Compared with SSPC, PACMAN improved throughput by up to 41% and energy efficiency by up to 24%.

**Index Terms**—Solid-State Drive, Power Capping, Power Management, Storage Server, Energy Efficiency.

## I. INTRODUCTION

The surging demand for AI infrastructure is driving unprecedented growth in data center investments, with global capital expenditures (CapEx) projected to exceed \$1.1 trillion by 2029, up from \$430 billion in 2024 [1]. Alongside this trend, AI-optimized servers are expected to consume 500 TWh annually by 2027, 2.6× higher than in 2023, placing critical pressure on existing power infrastructure. As a result, 40% of AI data centers are predicted to face operational constraints due to power limitations [2].

To cope with this, modern hyperscalers aggressively adopt *power oversubscription*, provisioning more servers than the guaranteed power budget allows. While effective for maximizing resource utilization, oversubscription demands precise and adaptive *power capping* mechanisms to avoid outages and tripped circuit breakers caused by power overdraw [3]. To maximize the effectiveness of such strategies in real-world deployments, leading hyperscalers are actively researching and deploying diverse power management techniques across device, server, and facility levels [3]–[7].

Most prior research on power capping has focused on CPUs and memory, which have traditionally dominated power

consumption in commodity servers. These efforts typically leverage techniques such as dynamic voltage and frequency scaling (DVFS) and Intel RAPL [8]. More recently, power capping solutions for GPUs have gained attention due to the rise of AI accelerators like the NVIDIA H100 with a TDP of up to 700W [9], [10]. In parallel, all-flash storage servers are undergoing a shift, with high-density SSD arrays becoming a major power consumer in storage-centric infrastructures. A single server populated with 16 to 32 SSDs can easily consume 400W to 800W of peak disk power, often exceeding CPU power draw. However, little research has focused on power capping in such SSD arrays.

The NVMe specification defines support for per-device power capping, and commercial SSDs provide multiple power states to control maximum power. One common method to apply power limits in a storage system is static storage power capping (SSPC), which sets fixed power states across all SSDs [11]. However, this approach overlooks workload heterogeneity and temporal variations in device activity, such as differences in read/write intensity or active versus idle states. As a result, SSPC often leads to underutilization of the power budget or unnecessary over-capping of active devices, limiting overall system efficiency. To fully utilize the available power budget while maximizing performance, a dynamic and device-aware power management strategy is required.

In this paper, we propose PACMAN, a power usage-aware capping and management framework for all-flash storage systems. To the best of our knowledge, this is the first work to improve both performance and energy efficiency under a constrained storage power budget. PACMAN dynamically adjusts the power cap of each SSD based on predicted power demand, using a lightweight prediction method that analyzes recent power consumption history. Designing such a prediction method presents several challenges. We identify two key difficulties in estimating power from performance metrics such as throughput. First, energy efficiency (i.e., performance per watt) varies significantly across different I/O patterns, and accurately identifying these patterns at runtime is a challenge. Second, internal SSD operations introduce unobservable behaviors that decouple performance from actual power usage, making performance-based prediction inherently unreliable. To address these limitations, PACMAN leverages direct power measurements to enable accurate, workload-agnostic power prediction and robust dynamic control.

PACMAN operates as a feedback control loop that (1)

† Corresponding author.

periodically measures device-level power usage, (2) predicts future demand, and (3) reallocates power caps accordingly. It is designed to be lightweight enough to run on both the host CPU and resource-constrained baseboard management controller (BMC) [12].

This paper makes the following contributions:

- Identifies that SSPC and inaccurate device-level power prediction can degrade both system performance and energy efficiency.
- Develops PACMAN, a storage power capping framework that maximizes performance and energy efficiency under a storage power budget.
- Designs the framework to be lightweight, enabling deployment on either the host CPU or the BMC.
- Evaluates the proposed system on a real storage platform across multiple application scenarios, demonstrating up to 41% higher performance and 24% better energy efficiency compared with SSPC under the same power budget.

## II. BACKGROUND AND PRELIMINARY STUDY

This section identifies SSDs as a major source of power consumption in storage servers and explores their power management features and the effects of leveraging them.

### A. Key Components of Power in the Storage Server

Power management has become increasingly important in storage-centric infrastructures due to the rising demands of AI-driven infrastructures and their workloads [13]. These infrastructures are typically composed of all-flash storage servers, which integrate dozens of SSDs to deliver high throughput and low latency. These servers eliminate spinning disks in favor of flash memory, enabling more consistent and responsive performance for data-intensive applications. In such systems, SSD arrays have emerged as the primary consumers of power, necessitating SSD-centric power management techniques.

Table I shows the peak power breakdown of each hardware component of the storage server [15]. In this 1U server, the 16 SSDs account for 400W of peak power consumption, representing the highest proportion among all components. Furthermore, in a 2U server, up to 32 SSDs can be installed, increasing the peak power of SSD arrays to 800W. The number of SSDs in storage servers (i.e., storage density) continues to increase, accounting for a growing share of total system power consumption. Prior study has already anticipated that disk power consumption would become a critical issue in data-intensive servers [16].

TABLE I: Peak power breakdown in a storage server.

Component	Peak Power	Count	Total
CPU (AMD EPYC 9334)	210 W	1	210 W
Memory	9 W	4	36 W
SSD (Samsung PM1743 [14])	25 W	16	400 W
NIC (ConnectX-6Dx)	26.6 W	2	53.2 W
FAN	39.6 W	4	158.4 W

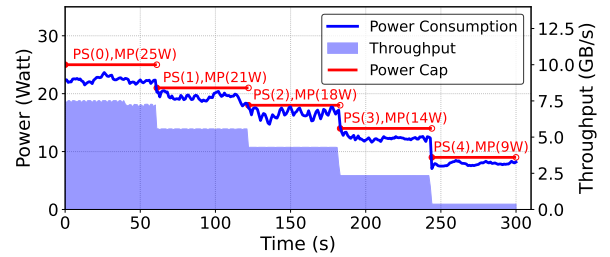


Fig. 1: Effect of NVMe power state transitions (PS0-PS4) on power consumption and throughput under a sequential write workload.

### B. NVMe Power Management Features

Server hardware components such as the CPU, memory, and GPU provide power capping/management features. Similarly, SSDs also support power management features defined in the specification. The main purpose of this feature is to adjust the maximum power (i.e., the power cap) of an SSD, which can be tuned by workload to achieve an optimal total cost of ownership (TCO) in data centers [11]. According to the NVMe specification, an SSD can support up to 32 power states (PS), each of which has characterized values such as maximum power (MP), entry latency (ENTLAT), and exit latency (EXLAT). By adjusting the PS, the MP of an SSD changes, which means power capping at the device level. Several vendors provide SSDs with power management capabilities, including the Samsung PM1743 (PS0-PS4), Solidigm D5-P5336 (PS0-PS2), and Micron 6550 ION (PS0-PS7), and each product provides multiple PS levels and the corresponding MP [17]. Users can check the power descriptors, including the number of PS and the details of each state, using the `nvme id-ctrl` command. They can also check or change the active PS of a device using the `nvme get-feature` or `nvme set-feature` commands (feature ID: 2).

### C. Effects of NVMe Power Management

We first evaluate the effectiveness of the NVMe power management features described above. We use the FIO benchmark [18], and the detailed experimental setup is described in Section VI-A. Figure 1 shows the power consumption and throughput changes measured by stepping PS from 0 to 4 while performing a sequential write workload on a single SSD. The experiment shows that applying power capping can save power budget, but at the expense of peak performance. For example, with the PS(3) setting, MP is reduced from 25W to 14W compared to the setting of PS(0), for a power savings of 11W, but the maximum throughput is reduced from about 7.3GB/s to 2.3GB/s, for a performance loss of about 5GB/s.

However, from a system perspective, given the statistical effects (i.e., not all servers reach full load simultaneously) and the time-variability of workloads (from idle to peak) [4], it is possible to minimize performance loss or, in some cases, save power budget without losing performance if power capping is applied in predicting SSD performance needs or power load at any given point in time.

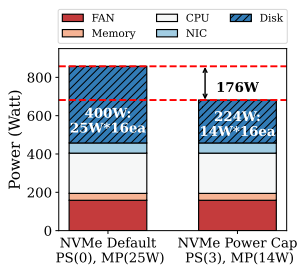


Fig. 2: Power savings with NVMe power capping.

capping: (i) the saved power budget can be reallocated to power-intensive resources such as AI-centric servers with high-end GPUs or CPUs. (ii) storage density can be increased by mounting more SSDs into additional slots or placing more storage servers per rack within the same power budget. For example, a power savings of 176W ( $\geq 12 \times 14W$ ) enables up to 12 additional SSDs with a power cap of 14W under PS3.

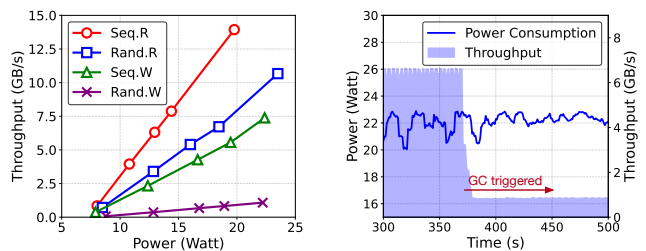
### III. RELATED WORK

**Modern Power Capping Studies.** Recent power capping research has mainly targeted compute resources such as CPUs, memory, and GPUs. Intel RAPL [8] enforces time-based limits on processor packages and DRAM, while systems like Facebook’s Dynamo [6] and SmoothOperator [7] manage datacenter-wide power through priority-based scheduling and power slack reduction. In HPC and AI, strategies such as coordinated GPU frequency and power control have been explored to smooth power spikes [10]. Model-based techniques have also been proposed to analyze power–performance trade-offs at server and cluster levels [19]–[21]. Although compute-side solutions have advanced rapidly, enabled by rich DVFS and telemetry support, the storage subsystem remains largely underexplored.

**Lack of Study on Power Capping in Storage Systems.** On the storage side, PARAID [22] investigated energy savings by gradually reducing disk rotation speed through a gear-shifting technique. However, its coarse-grained approach, involving motor spin-down and spin-up, led to latency and reliability issues. Recent work combined NVMe SSD power state transitions with I/O shaping to expand the power-performance range and explore trade-offs using workload-specific models [23], but relied on manual tuning at the device level. Similarly, theoretical studies analyzed the effects of power variation on SSD internals such as garbage collection and write amplification [24], yet remained limited to single-device models without addressing array-level dynamic capping. In summary, although localized and static methods—like HDD spin-down or single-SSD control—have been studied, no practical solution exists for real-time power capping across multiple NVMe devices with system-wide performance optimization.

**Static Storage Power Capping (SSPC).** SSPC is the simplest and often the only approach used for storage power management, whereby all SSDs are configured to the same PS and corresponding MP within a given storage power budget. This

Figure 2 illustrates the effect of power capping on a system configured with 16 SSDs with the power breakdown detailed in Table I. By capping the drives, the peak power of the SSD array is reduced from 400W to 224W, resulting in 176W of power savings at the system level. This demonstrates two key infrastructure-level benefits of NVMe power



(a) Power-throughput relation by workload type (b) Decoupled throughput-power behavior under random write

Fig. 3: SSD power characteristics across workload patterns.

approach can operate optimally within the power budget only when all SSDs have the same load conditions, such as uniform data block sizes and consistent read/write access patterns.

### IV. MOTIVATION

In this section, we examine the limitations of SSPC in all-flash storage servers, motivating the need for dynamic and power-aware capping approaches.

#### A. Limitation of Static Storage Power Capping

**Limitation #1: Workload Variation Across Disks.** When multiple SSDs within a storage server are allocated to different applications, the load level and workload pattern of each SSD can vary over time, resulting in heterogeneous power consumption across SSDs. For example, in a cloud environment, the power usage of each physical SSD can differ due to variations in workloads per virtual machine.

**Limitation #2: Dynamic Active-to-Idle Disk Ratio.** The active-to-idle ratio across SSDs in a storage server can vary due to a variety of systemic factors. For example, in high availability or fault tolerance environments, hot spare SSDs are used, keeping a certain ratio of SSDs in idle state. Similarly, in network-shared storage environments, network bandwidth is often the performance bottleneck. In such cases, either all SSDs can be active within a limited bandwidth, or only a portion of SSDs remain active, while others stay idle and become active only when specific needs occur, such as volume expansion.

These two limitations highlight a key drawback of SSPC, as it cannot flexibly accommodate device-specific power demands in diverse and dynamic environments. Consequently, SSPC often leads to over-capping of active SSDs and underutilization of the power headroom within their current caps, degrading both system performance and energy efficiency. This motivates the need for a dynamic storage power capping approach that adaptively reallocates power caps based on per-device load.

#### B. Need for Power Usage-Based NVMe Power Capping

To effectively apply dynamic storage power capping, it is essential to estimate the power consumption of each SSD. This enables efficient power distribution within a given storage power budget by allocating higher power caps to SSDs with higher expected power demand, and lower caps to those with lower demand. However, estimating the power consumption

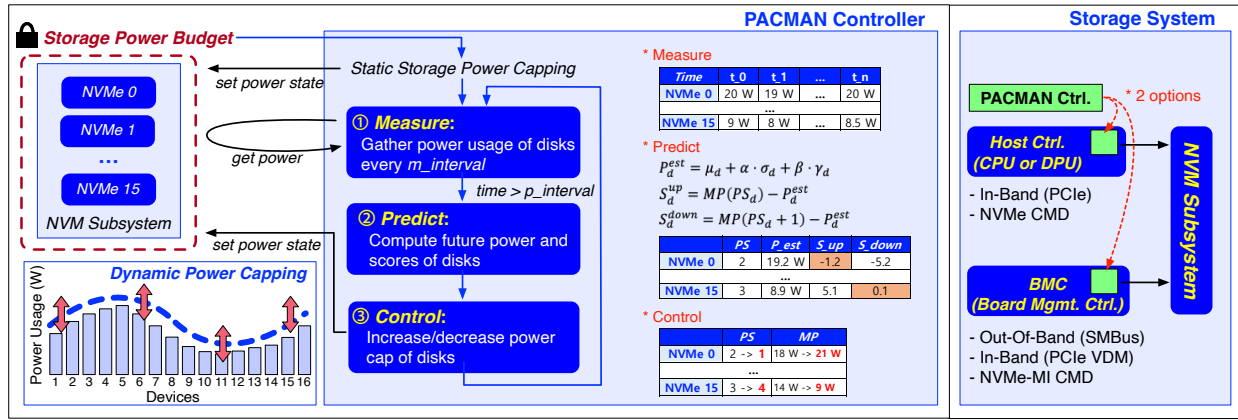


Fig. 4: A high-level architecture of PACMAN.

of an SSD based on performance metrics such as throughput is inherently challenging. Even at similar throughput levels, power usage can vary significantly due to differences in I/O type (read/write), access patterns (sequential/random), and internal SSD behaviors such as garbage collection (GC), read scrubbing, and thermal throttling.

Figure 3(a) shows the relationship between power consumption and throughput for the SSD under representative workload patterns. The detailed experimental setup is described in Section VI-A. Although performance and power show a linear relationship within each workload, throughput can differ by up to 2× or more across workloads at similar power levels. The slope, an energy-efficiency metric (i.e., performance per watt), also varies significantly across workloads. While power estimation would be feasible if the SSD’s workload patterns could be accurately identified, in practice, mixed I/O patterns make such identification challenging from outside the SSD.

Figure 3(b) shows a case where GC is triggered internally during sustained random writes. While power consumption remains around 20-24W, throughput drops sharply from around 6.5GB/s to 1GB/s. In such a case, lowering the power cap solely in response to the observed throughput drop can worsen performance degradation. This decoupling between throughput and power highlights the challenge of accurately estimating SSD power consumption based on performance metrics.

## V. PACMAN: DESIGN AND IMPLEMENTATION

In this section, we present PACMAN, a power usage-aware capping and management framework for dynamic storage power control in all-flash storage servers. It is designed to maximize both performance and energy efficiency within a given storage power budget.

### A. Design of Power Usage-Aware Capping and Management

Figure 4 shows the high-level architecture of the PACMAN framework, where the controller dynamically manages the power caps of SSD arrays to operate within a given storage power budget. It initially applies SSPC, and then enters a feedback loop consisting of three steps: (1) Measure, (2) Predict, and (3) Control. The controller collects the real-time power usage of each SSD, estimates future power demand,

and computes scores. Based on these scores, it dynamically readjusts the power caps of individual SSDs, ensuring that the total power budget is not exceeded.

PACMAN can operate on two types of hardware controllers within storage servers. It manages SSD power either in-band (IB), through the host CPU or a DPU-based controller, or out-of-band (OOB), via a BMC. Given the limited hardware resources of the BMC, PACMAN is designed as a lightweight framework composed of simple, low-overhead algorithms for estimating SSD power consumption and managing power caps efficiently, thereby minimizing the computational overhead on the controller. The following sections provide a detailed description of each component of the PACMAN framework.

**Measure.** After SSPC, PACMAN periodically measures the power consumption of all SSDs at every  $m\_interval$ . As discussed in Section IV-B, accurately estimating power consumption based solely on SSD performance metrics is challenging. Therefore, PACMAN relies solely on historical power usage to estimate SSD future power, enabling a more straightforward analytical formulation and reducing both memory usage and computational overhead.

**Predict.** The power consumption of each SSD is periodically measured at every  $m\_interval$ , and when the cumulative measurement duration reaches  $p\_interval$ , the predict phase is triggered using the collected power history. In this phase, the future power demand of each SSD is estimated to compute scores that determine whether the power cap should be increased or decreased. Algorithm 1 describes the process of estimating the future power consumption of device  $d$ , denoted as  $P_d^{est}$ , and calculating two decision scores:  $S_d^{up}$  for evaluating the potential to increase the power cap, and  $S_d^{down}$  for evaluating the potential to decrease it.

First, the estimation is performed using recent power history data  $P_d$  collected over the past  $p\_interval$  (lines 1–5). The prediction formula is defined as  $P_d^{est} = \mu_d + \alpha \cdot \sigma_d + \beta \cdot \gamma_d$ , where  $\mu_d$  denotes the average power consumption of device  $d$ , and  $\sigma_d$  denotes the standard deviation, capturing the stability or elasticity of power usage.  $\gamma_d$  represents the slope of power consumption over time, capturing trends of increase or decrease, and is computed by applying an ordinary least

---

**Algorithm 1: PACMAN Predict Phase**

---

**Input:** Recent power history  $P_d = \{p_0, p_1, \dots, p_{N-1}\}$  for device  $d$ , current power state  $PS_d$ , power descriptor  $POWER\_DESC$ , hyperparameters  $\alpha, \beta$   
**Output:** Estimated power  $P_d^{est}$ , Scores  $S_d^{up}, S_d^{down}$

```
1  $\mu_d \leftarrow \text{mean}(P_d)$ ; // Average power consumption
2  $\sigma_d \leftarrow \text{std}(P_d)$ ; // Power variability
// Compute power slope  $\gamma_d$  using OLS regression
3  $t_i \leftarrow$  time index of each  $p_i$ ;
4  $\gamma_d \leftarrow \frac{N \cdot \sum t_i p_i - \sum t_i \cdot \sum p_i}{N \cdot \sum t_i^2 - (\sum t_i)^2}$ 
// Predict future power
5  $P_d^{est} \leftarrow \mu_d + \alpha \cdot \sigma_d + \beta \cdot \gamma_d$ 
// Compute score for upward power cap change
6 if  $PS_d - 1 \in POWER\_DESC$  then
7 |  $S_d^{up} \leftarrow MP(PS_d) - P_d^{est}$ 
8 else  $S_d^{up} \leftarrow \infty$ ;
// Compute score for downward power cap change
9 if  $PS_d + 1 \in POWER\_DESC$  then
10 |  $S_d^{down} \leftarrow MP(PS_d + 1) - P_d^{est}$ 
11 else  $S_d^{down} \leftarrow -\infty$ ;
12 return  $P_d^{est}, S_d^{up}, S_d^{down}$ 
```

---

squares (OLS) linear regression (line 4). The hyperparameters  $\alpha$  and  $\beta$  are weights that control the sensitivity of  $\sigma_d$  and  $\gamma_d$ , respectively. This power prediction method is further discussed in Section VI-D through comparative experiments against other prediction methods, including mean-based prediction and exponential moving average (EMA).

In the next step, two scores are computed for each device based on the estimated power consumption  $P_d^{est}$ . The score for evaluating the potential to increase the power cap is calculated as  $S_d^{up} = MP(PS_d) - P_d^{est}$  (lines 6-8). This score represents the estimated power headroom by measuring the gap between the current power cap and the estimated future power consumption. A smaller  $S_d^{up}$  indicates that device  $d$  is likely to utilize more power in the future if the cap is raised.

The score for evaluating the potential to decrease the power cap is calculated as  $S_d^{down} = MP(PS_d + 1) - P_d^{est}$  (lines 9-11). This value represents the difference between the power cap at one level lower than current setting and the estimated future power consumption. A positive score indicates that the reduced cap would still exceed the estimated power demand, suggesting that the cap can be safely lowered without impacting performance. Conversely, a negative score implies that the estimated demand would exceed the reduced cap, potentially leading to performance degradation.

**Control.** In Algorithm 2, PACMAN dynamically reallocates the power caps of devices based on the scores  $S_d^{up}$  and  $S_d^{down}$  calculated in the prediction phase. The goal is to lower the power caps of underutilized devices and redistribute the saved power to devices that may need more power in the near future. The control process consists of two main steps.

The first step is the power saving (lines 3-7). PACMAN identifies disks whose power caps can be safely lowered, those satisfying  $S_d^{down} > 0$ . For such devices, the PS value is

---

**Algorithm 2: PACMAN Control Phase**

---

**Input:** Set of devices  $D$ , power descriptor  $POWER\_DESC$ , each with current power state  $PS_d$ , scores  $S_d^{up}, S_d^{down}$ , residual power  $P^{save}$  from previous control phase  
**Output:**  $P^{save}$

```
1  $P^{save} \leftarrow P^{save}$ ; // Power saved for redistribution
2  $D_{up} \leftarrow \emptyset$ ; // Power cap - eligible devices
// Decrease the power cap of device
3 foreach device  $d \in D$  do
4 | if  $S_d^{down} > 0$  and  $PS_d + 1 \in POWER\_DESC$  then
5 | |  $P^{save} \leftarrow P^{save} + (MP(PS_d) - MP(PS_d + 1))$ ;
6 | |  $PS_d \leftarrow PS_d + 1$ ;
7 | | update power cap of  $d$  using nvme set-feature;
// Increase the power cap of device
8  $D_{up} \leftarrow$  devices where  $PS_d - 1 \in POWER\_DESC$ , sorted by ascending  $S_d^{up}$ ;
9 foreach device  $d \in D_{up}$  do
10 |  $\Delta \leftarrow MP(PS_d - 1) - MP(PS_d)$ ;
11 | if  $P^{save} \geq \Delta$  then
12 | |  $P^{save} \leftarrow P^{save} - \Delta$ ;
13 | |  $PS_d \leftarrow PS_d - 1$ ;
14 | | update power cap of  $d$  using nvme set-feature;
15 | else break;
16 return  $P^{save}$ 
```

---

increased (resulting in a lower cap), and the amount of reduced power (i.e.,  $MP(PS_d) - MP(PS_d + 1)$ ) is accumulated into a temporary variable  $P^{save}$ . The adjustment is applied using the `nvme set-feature` command.

The second step is the power redistribution (lines 8-15). Devices are sorted in ascending order of  $S_d^{up}$ , prioritizing those that are closer to exceeding their current cap. For each device in the sorted list, if the available  $P^{save}$  is sufficient, its PS is decreased (increasing the power cap), and the saved power is consumed accordingly. The loop terminates if the saved power is no longer enough for additional reallocations. Any residual power  $P^{save}$  is carried over to the next control interval. PACMAN then returns to the measurement phase and continues this cycle in a feedback-driven manner.

### B. Implementation

PACMAN is implemented on the host CPU running Linux and manages SSD power via NVMe commands over the PCIe for IB. For BMC-based implementations, since the OpenBMC [25] platform is also Linux-based, only the NVMe command interface and communication protocol are modified. The BMC-based PACMAN is implemented based on Intel-BMC's `nvme-mi` module [26], and uses NVMe Management Interface (NVMe-MI) [27] commands. These commands are transmitted via either MCTP over SMBus for OOB or MCTP over PCIe Vendor-Defined Messaging (VDM) for IB. In Section VI, we evaluate PACMAN on the host CPU environment.

## VI. EVALUATION

### A. Experimental Setup

This section describes the platform, workloads, metrics and compared systems for evaluating PACMAN.

**Platform.** We conducted our experiments in Samsung Memory Research Center (SMRC) [28]. We ran our experiments on an

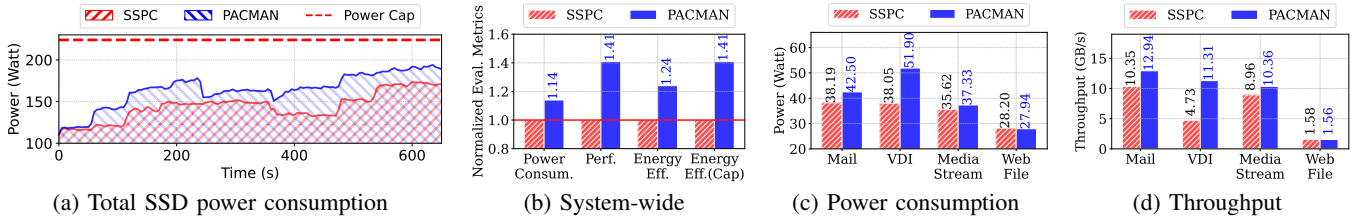


Fig. 5: Evaluation results of the FIO benchmark. Each subfigure shows (a) total SSD power consumption, (b) system-wide normalized metrics, and (c) power consumption and (d) throughput by workload. For all results, higher values are better.

all-flash storage server configured with a Supermicro ASG-1115S-NE316R system. The server is equipped with a single-socket AMD EPYC 9334 32-core processor and 16 Samsung PM1743 16TB NVMe SSDs. Each SSD supports multiple power states, including PS0 (25W), PS1 (21W), PS2 (18W), PS3 (14W), and PS4 (9W). The system runs on Linux 5.15.0.

**Parameter Configuration.** The total storage power cap is set to 224W ( $= 14W \times 16$ ), based on the total of 16 SSDs, each following the power cap defined for PS3 (14W), which is one level above the lowest PS4. In the PACMAN framework, we experimentally set  $m\_interval$  to 0.5 seconds and  $p\_interval$  to 5 seconds. A smaller  $m\_interval$  enables more frequent measurements and can improve the accuracy of power estimation, but if set too small, it may cause excessive polling and oversampling overhead. Likewise, a shorter  $p\_interval$  may lead to unnecessary PS adjustments, while a longer one may be too slow to react to workload changes. To determine the optimal hyperparameters  $\alpha$  and  $\beta$  in the power estimation formula used in the PACMAN prediction phase, we used Bayesian optimization with the Optuna framework [29]. For this experiment, we used  $\alpha = 0.48$  and  $\beta = 0.56$ , derived with the objective function set as the maximum storage power usage.

**Evaluation Metrics.** We used three evaluation metrics: power consumption, performance, and energy efficiency. Energy efficiency is defined as performance per watt, and we calculated it in two ways. One is based on actual power consumption, and the other is based on the allocated storage power cap. We use the latter method because it is common in colocation environments for service providers to charge based on contractually assigned power caps, rather than actual power usage [30].

**Compared Systems.** To evaluate the effectiveness of our proposed system, PACMAN, we performed a comparison experiment with the SSPC approach introduced in Section IV. SSPC applies the same fixed PS to the entire SSD array.

TABLE II: Detailed workload characteristics. On the left, FIO workload characteristics for micro benchmarks. On the right is the RocksDB configuration for application-level evaluation. SST count refers to the number of compaction trigger SSTs.

FIO Benchmark					RocksDB	
Workload	Block Size	R/W (%)	Rand./Seq. (%)	Job (#)	Workload	SST (#)
Mail Serv.	4KB	67/33	100/0	16	SST <sub>256</sub>	256
VDI Serv.	4, 16KB	20/80	80/20	1-5		
Media Serv.	64KB	98/2	0/100	5		
Web Serv.	4, 8, 16KB	95/5	75/25	1	SST <sub>1024</sub>	1024

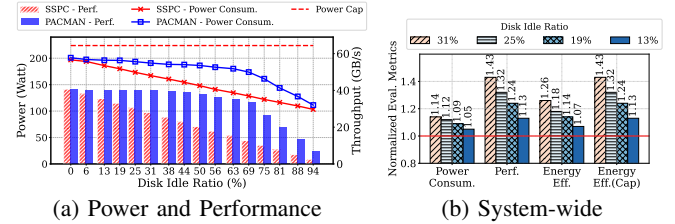


Fig. 6: Evaluation results varying SSD idle ratios. (a) shows power consumption and throughput, and (b) shows normalized system-wide evaluation metrics (ratio of PACMAN to SSPC).

### B. Micro Benchmark

**FIO Workloads.** We first evaluated PACMAN using the FIO benchmark tool [18]. For this evaluation, we simulated four enterprise server application workloads with varying traffic patterns, reflecting the scale and diversity of data processed in enterprise environments. The left side of Table II summarizes the I/O characteristics of each workload used in the FIO benchmark. To introduce more diverse traffic patterns, we dynamically varied the number of jobs for the VDI Server from 1 to 5 and incorporated periods of idleness into each workload. The entire SSD array was divided into four logical segments, with a different workload assigned to each segment.

**Results.** Figure 5(a) compares power consumption across the entire SSD array. PACMAN generally consumes more power than SSPC, averaging 160W compared to SSPC’s 140W. This is because PACMAN dynamically reallocates power caps based on per-device workload intensity, while SSPC applies a uniform fixed cap.

Figure 5(b) shows normalized system-wide metrics across the entire SSD array. Compared to SSPC, PACMAN improves performance by 41% with a 14% increase in power consumption, resulting in a 24% gain in energy efficiency based on actual power and a 41% gain based on the power cap.

Figure 5(c) and (d) present per-workload power consumption and throughput. For the Mail Server and VDI Server, PACMAN consumes 11% and 36% more power, respectively, while achieving 1.2 $\times$  and 2.4 $\times$  higher throughput. For Media Stream Server, a 5% power increase yields a 1.15 $\times$  throughput improvement. On the Web File Server, both approaches perform similarly due to its light workload, which leaves little room for power increase relative to the cap.

**Impact of SSD Idle Ratios.** In this experiment, we varied the active-to-idle ratio of the SSD array and ran VDI Server workloads on the active SSDs. Figure 6(a) compares power

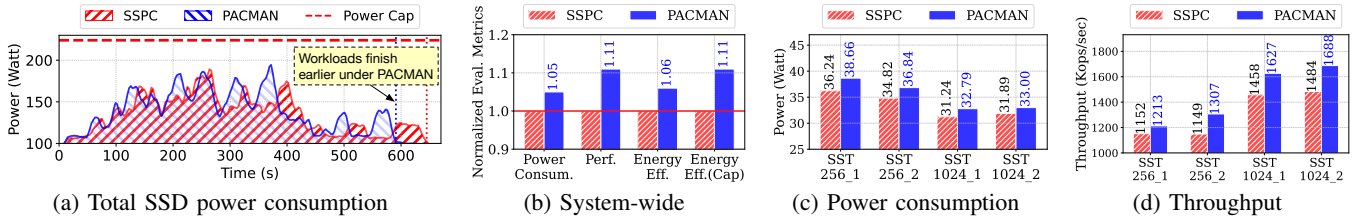


Fig. 7: Evaluation results of the RocksDB. Each subfigure shows (a) total SSD power consumption, (b) system-wide normalized metrics, and (c) power consumption and (d) throughput by workload. For all results, higher values are better.

consumption and throughput between SSPC and PACMAN across different idle ratios. When the idle ratio is 0%, both approaches perform similarly. As more SSDs become idle, SSPC degrades linearly, while PACMAN maintains stable throughput (38–40 GB/s) by reallocating more power to active SSDs. As a result, PACMAN consistently outperforms SSPC across all idle ratios.

Figure 6(b) shows normalized results relative to SSPC. At low idle ratios (13–31%), PACMAN uses 5–14% more power, but achieves 13–43% better performance, a 7–26% gain in energy efficiency based on actual power, and a 13–43% gain based on the power cap. These results demonstrate that PACMAN effectively reallocates unused power to maximize overall performance and energy efficiency.

### C. Application-Level Benchmark

**RocksDB Workloads.** We also evaluated the effectiveness of PACMAN in a real application using RocksDB [31], a widely used log-structured merge-tree based key-value store. In RocksDB, compaction [32] is triggered when the number of Sorted String Table (SST) files—immutable data stored on storage media—reaches a predefined threshold. Since compaction performs a merge-sort over data inserted in an append-only manner, it generates a significant amount of disk I/O.

To simulate diverse disk I/O patterns in RocksDB, we configured two types of workloads by adjusting the compaction trigger threshold (i.e., the number of SST files). Detailed workload characteristics are shown on the right side of Table II. For example, SST<sub>256</sub> refers to a workload in which compaction is triggered when the number of SST files reaches 256. Following the same setup as the FIO benchmark, we deployed four workloads—two instances each of SST<sub>256</sub> and SST<sub>1024</sub>—across the entire SSD array, divided into four logical segments. All workloads used 16-byte keys and 1KB values, and inserted 300GB of data.

**Results.** Figures 7(a)–(d) present the total power consumption, system-wide metrics, and per-workload power consumption and throughput, respectively. In a similar manner to Figure 5, PACMAN consistently outperforms SSPC across all metrics.

As shown in Figure 7(a), PACMAN generally consumes more power than SSPC across the entire SSD array, with a total power consumption approximately 5% higher. However, due to improved workload throughput, PACMAN completes execution about 54 seconds earlier than SSPC. Figures 7(b)–(d) show that PACMAN improves both power consumption and throughput across all workloads. Notably, PACMAN achieves

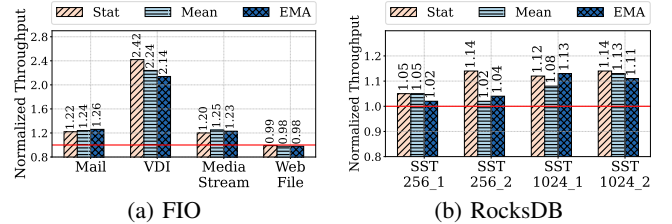


Fig. 8: Normalized throughput comparison of different PACMAN power prediction methods against SSPC. "Stat" uses the proposed statistical formula, "Mean" uses average power and "EMA" uses exponential moving average.

an 11% improvement in system-wide energy efficiency based on the power cap. These results indicate that PACMAN can effectively improve both performance and energy efficiency not only in micro benchmarks such as FIO, but also in real-world applications with intensive disk I/O, such as RocksDB.

### D. Analysis of the Power Prediction Model

In the prediction phase of PACMAN, SSD power consumption is estimated using the statistical formula  $P_d^{\text{est}} = \mu_d + \alpha \cdot \sigma_d + \beta \cdot \gamma_d$  which combines the mean, standard deviation, and slope of recent power measurements. In this section, we compare the prediction method used in PACMAN with alternative methods and discuss its prediction accuracy and strategic advantages.

**Accuracy of Prediction Methods.** Figure 8 presents a comparison of different power prediction methods evaluated using FIO and RocksDB workloads. In the legend, Stat refers to the proposed method used in PACMAN, Mean uses only the average power over the  $p\_interval$  (excluding  $\sigma_d$  and  $\gamma_d$  from the Stat formula), and EMA applies exponential moving average for power estimation. All throughput values are normalized against SSPC.

Overall, all three prediction methods outperform SSPC in both FIO and RocksDB benchmarks. Notably, the Stat method achieves the highest throughput, outperforming other approaches by up to 13% for the VDI Server in FIO and 12% for the SST<sub>256\_2</sub> workload in RocksDB. These results suggest that while simple methods like Mean and EMA can improve performance and energy efficiency, incorporating workload variability and trend, as demonstrated by the Stat method, leads to better effectiveness.

### Prediction Accuracy and Overestimation Strategy.

Figure 9 compares the actual power consumption of a single disk with the power estimation results of PACMAN.

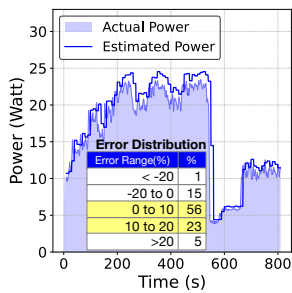


Fig. 9: Power Estimation Accuracy for a Single SSD.

to SSD performance degradation. Therefore, to ensure stable operation without performance loss, PACMAN adopts a conservative strategy that deliberately allows a certain level of overestimation in prediction to prevent over-capping and prioritize system reliability.

## VII. CONCLUSION

In this paper, we introduced PACMAN, a lightweight and practical power capping and management framework for all-flash storage servers. Unlike SSPC approaches that apply uniform power limits across all SSDs, PACMAN dynamically adjusts SSD power caps based on per-device power usage to maximize performance under a given power budget. By leveraging a low-overhead statistical prediction method, PACMAN adapts to workload variability without relying on performance counters, making it suitable for both host CPU and BMC environments. Through experiments using a diverse set of micro and application-level benchmarks, we demonstrated that PACMAN improved throughput by up to 41% and energy efficiency by up to 24% compared with the SSPC approach.

## ACKNOWLEDGMENT

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korean government (MSIT) (RS-2024-00453929) (RS-2024-00416666). The authors would like to thank SMRC (Samsung Memory Research Center) for providing the infrastructure for this work.

## REFERENCES

- [1] Dell'Oro Group, "Data Center Capex to Surpass \$1 Trillion by 2029." <https://www.delloro.com/news/data-center-capex-to-surpass-1-trillion-by-2029/>, 2025. Accessed: 2025-06-01.
- [2] Gartner, "Gartner Predicts Power Shortages Will Restrict 40% of AI Data Centers By 2027." <https://www.gartner.com/en/newsroom/press-releases/2024-11-12-gartner-predicts-power-shortages-will-restrict-40-percent-of-ai-data-centers-by-2027>, 2024. Accessed: 2025-06-01.
- [3] A. G. Kumbhare, R. Azimi, I. Manousakis, A. Bonde, F. Frujeri, N. Mahalingam, P. A. Misra, S. A. Javadi, B. Schroeder, M. Fontoura, et al., "Prediction-based power oversubscription in cloud platforms," in *2021 USENIX Annual Technical Conference (USENIX ATC 21)*, pp. 473–487, 2021.
- [4] X. Fan, W.-D. Weber, and L. A. Barroso, "Power provisioning for a warehouse-sized computer," *ACM SIGARCH computer architecture news*, vol. 35, no. 2, pp. 13–23, 2007.
- [5] V. Sakalkar, V. Kontorinis, D. Landhuis, S. Li, D. De Ronde, T. Blooming, A. Ramesh, J. Kennedy, C. Malone, J. Clidaras, et al., "Data center power oversubscription with a medium voltage power plane and priority-aware capping," in *Proceedings of the Twenty-Fifth International Conference on Architectural Support for Programming Languages and Operating Systems*, pp. 497–511, 2020.
- [6] Q. Wu, Q. Deng, L. Ganesh, C.-H. Hsu, Y. Jin, S. Kumar, B. Li, J. Meza, and Y. J. Song, "Dynamo: Facebook's data center-wide power management system," *ACM SIGARCH Computer Architecture News*, vol. 44, no. 3, pp. 469–480, 2016.
- [7] C.-H. Hsu, Q. Deng, J. Mars, and L. Tang, "Smoothoperator: Reducing power fragmentation and improving power utilization in large-scale datacenters," in *Proceedings of the Twenty-Third International Conference on Architectural Support for Programming Languages and Operating Systems*, pp. 535–548, 2018.
- [8] H. David, E. Gorbatov, U. R. Hanebutte, R. Khanna, and C. Le, "Rapl: Memory power estimation and capping," in *Proceedings of the 16th ACM/IEEE international symposium on Low power electronics and design*, pp. 189–194, 2010.
- [9] P. Patel, E. Choukse, C. Zhang, Í. Goiri, B. Warriar, N. Mahalingam, and R. Bianchini, "Characterizing power management opportunities for llms in the cloud," in *Proceedings of the 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 3*, pp. 207–222, 2024.
- [10] T. Patki, Z. Frye, H. Bhatia, F. Di Natale, J. Glosli, H. Ingolfsson, and B. Rountree, "Comparing gpu power and frequency capping: A case study with the mummy workflow," in *2019 IEEE/ACM Workflows in Support of Large-Scale Science (WORKS)*, pp. 31–39, IEEE, 2019.
- [11] NVM Express, "Technology Power Features." <https://nvmexpress.org/resource/technology-power-features/>, 2022. Accessed: 2025-06-01.
- [12] J. Frazelle, "Opening up the baseboard management controller: If the cpu is the brain of the board, the bmc is the brain stem.," *Queue*, vol. 17, no. 5, pp. 5–12, 2019.
- [13] Brian Beeler, "Boosting Data Center Efficiency with Solidigm SSDs and Liquid-Cooled Servers." <https://www.storagereview.com/review/boosting-data-center-efficiency-with-solidigm-ssds-and-liquid-cooled-servers>, 2024. Accessed: 2025-06-01.
- [14] Samsung Electronics, "Samsung SSD PM1743." [https://download.semiconductor.samsung.com/resources/white-paper/PM1743\\_White\\_Paper\\_240510.pdf](https://download.semiconductor.samsung.com/resources/white-paper/PM1743_White_Paper_240510.pdf), 2024. Accessed: 2025-06-01.
- [15] SUPERMICRO, "Storage A+ Server ASG-1115S-NE316R." <http://supermicro.com/en/products/system/datasheet/asg-1115s-ne316r>, 2025. Accessed: 2025-06-01.
- [16] E. V. Carrera, E. Pinheiro, and R. Bianchini, "Conserving disk energy in network servers," in *Proceedings of the 17th annual international conference on Supercomputing*, pp. 86–97, 2003.
- [17] Brian Beeler, "The Micron 6550 ION SSD: Gen5 Performance, Energy Efficiency, and High Capacity in One Drive." <https://www.storagereview.com/review/the-micron-6550-ion-ssd-gen5-performance-energy-efficiency-and-high-capacity-in-one-drive>, 2024. Accessed: 2025-06-01.
- [18] J. Axboe, "FIO - Flexible I/O Tester." <https://github.com/axboe/fio>, 2024. Accessed: 2025-06-01.
- [19] J. v. Kistowski, H. Block, J. Beckett, K.-D. Lange, J. A. Arnold, and S. Kounev, "Analysis of the influences on server power consumption and energy efficiency for cpu-intensive workloads," in *Proceedings of the 6th ACM/SPEC International Conference on Performance Engineering*, pp. 223–234, 2015.
- [20] L. Yu, Z. Zhou, S. Wallace, M. E. Papka, and Z. Lan, "Quantitative modeling of power performance tradeoffs on extreme scale systems," *Journal of Parallel and Distributed Computing*, vol. 84, pp. 1–14, 2015.
- [21] K. Molka and G. Casale, "Energy-efficient resource allocation and provisioning for in-memory database clusters," in *2017 IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*, pp. 19–27, IEEE, 2017.
- [22] C. Weddle, M. Oldham, J. Qian, A.-I. A. Wang, P. Reiher, and G. Kuenning, "Paraid: A gear-shifting power-aware raid," *ACM Transactions on Storage (TOS)*, vol. 3, no. 3, pp. 13–es, 2007.
- [23] D. Xie, T. Stavrinou, K. Zhu, S. Peter, B. Kasikci, and T. Anderson, "Can storage devices be power adaptive?," in *Proceedings of the 16th ACM Workshop on Hot Topics in Storage and File Systems*, pp. 47–54, 2024.
- [24] B. Van Houdt, "On the power of asymmetry and memory in flash-based ssd garbage collection," *Performance Evaluation*, vol. 97, pp. 1–15, 2016.
- [25] "OpenBMC." <https://www.openbmc.org/>.
- [26] "Intel BMC NVMe-MI." <https://github.com/Intel-BMC/nvme-mi>.
- [27] NVM Express, "NVM Express Management Interface Specification Revision 2.0." <https://nvmexpress.org/wp-content/uploads/NVM-Express-Management-Interface-Specification-Revision-2.0-2024.08.05-Ratified.pdf>, 2024. Accessed: 2025-06-01.
- [28] Samsung Electronics, "Samsung Memory Research Center." <https://smrc.biz.samsung.com/>, 2025. Accessed: 2025-06-01.
- [29] T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama, "Optuna: A next-generation hyperparameter optimization framework," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 2623–2631, ACM, 2019.
- [30] Jeff Bettencourt, "Colocation Pricing Guide: Power, Space and Key Questions to Ask Your Provider." <https://www.horizoniq.com/blog/colocation-pricing-guide/>, 2019. Accessed: 2025-06-01.
- [31] Facebook, "RocksDB: A Persistent Key-Value Store for Fast Storage Environments." <https://github.com/facebook/rocksdb>, 2013. Accessed: 2025-06-01.
- [32] Facebook, "RocksDB Compaction." <https://github.com/facebook/rocksdb/wiki/Compaction>, 2023. Accessed: 2025-06-01.