A Thermal-aware Workload Scheduler for High-performance LLM Inference in Cooling-regulated Datacenters

RUI LU, The Hong Kong Polytechnic University, Hong Kong
DAN WANG, The Hong Kong Polytechnic University, Hong Kong

The wide deployment of AI datacenters has led to an increasing demand for energy. Recently, there have been developments in the area of reference cooling temperatures in datacenter server rooms, with the aim of avoiding the setting of excessively low temperatures to save energy. For example, the recommendation in Singapore is [28°C–32°C] for level 4 datacenters, while the European Union code of conduct for datacenters suggests a temperature of 35°C. These standards were carefully tested to satisfy overall cooling requirements, yet we observe in this paper that there is a risk that, in certain scenarios, the heat that is dissipated may not match the heat generation of GPUs, especially for high-performance workloads such as LLM inference. Such a mismatch can lead to an increase in the temperature of the GPU and trigger its thermal-throttle mechanism. The GPU frequency will decrease in self-protection from the damage due to overheating and performance degradation. As such, the issue of cooling regulation poses challenges to high-performance computing in AI datacenters. In this paper, we study LLM inference serving in cooling-regulated datacenters. Specifically, a datacenter serves millions of LLM inference jobs. To maximize the throughput, a workload scheduler (e.g., Ray Serve) assigns the jobs to GPUs and determines the execution batch sizes on GPUs. We show that in cooling-regulated datacenters, existing schedulers can increase the probability of thermal throttling by 10 times, and the performance degradation can be as much as 34.2%. We develop a new thermal-aware workload scheduler, TAWS, which takes into consideration the GPU voltage and frequency. Our scheduler can maximize the throughput of LLM inference under a relatively high ambient temperature in datacenter server rooms. The evaluation results show that the new scheduler can lead to a maximum improvement of 40.94% of throughput under 41°C.

CCS Concepts: • **Social and professional topics** → **Sustainability**; • **Hardware** → **Power and energy**; • **Information systems** → *Data centers*; • **Computing methodologies** → *Artificial intelligence*.

Additional Key Words and Phrases: Thermal-aware Computing, LLM Inference Schedule, Cooling-regulated Datacenter, High-performance Computing

1 INTRODUCTION

Recently, there has been a rapid deployment of AI datacenters to support the growth of AI applications such as large language model (LLM) services [21, 40]. These AI datacenters consume huge amounts of energy [7, 13, 29, 33]. In AI datacenters, the cooling infrastructure consumes 35% of energy [44, 52]. Setting a low temperature in the server rooms will significantly increase energy consumption. Thus, we see that reference standards have been developed which try to support the performance of the datacenter while avoiding overcooling [18, 51]. As an example, a recent Singapore standard refined the ASHRAE standards [46]. It classified datacenter into four levels with reference temperature setpoints, e.g., inlet-air for level 4 is 28–32°C as compared to traditional settings of 25°C . It has been shown that an increase of 1°C can reduce the cooling energy by 8% [6].

Authors' addresses: Rui Lu, ruilu@polyu.edu.hk, The Hong Kong Polytechnic University, Hong Kong; Dan Wang, dan.wang@polyu.edu.hk, The Hong Kong Polytechnic University, Hong Kong.

However, with such cooling regulations, the heat dissipation capacity of the cooling infrastructure is reduced. Although the reference standards were carefully tested to satisfy the overall heat dissipation requirement for general scenarios, there is a risk that the heat removal may not match the heat generation of GPUs in certain scenarios, especially for high-intensity computing workloads such as LLM inference serving. Such a mismatch can lead to an increase in the GPU temperature. This can trigger the thermal-throttle mechanism of GPUs [24, 49], i.e., the frequency and voltage will automatically decrease to self-protect against the potential damage of overheating. This introduces challenges for high-performance AI computing in cooling-regulated datacenters.

In this paper, we study LLM inference serving in AI datacenters. Specifically, a datacenter serves millions of LLM inference jobs. To optimize system performance on throughput [30, 43], the GPU utilization [39], and service level objectives [28, 53], workload schedulers have been developed. For example, the Ray Serve [43] scheduler assigns LLM inference jobs to a fleet of GPUs and determines the batch size to execute the jobs on each GPU, with the intention of maximizing the throughput in terms of tokens per second.

All existing schedulers implicitly assume that the heat dissipation capacity is sufficient to manage the heat generated by GPUs [19, 27, 43, 56]. However, we show that in cooling-regulated datacenters, this assumption fails and the performance of LLM inference decreases. Thus, we develop a new thermal-aware scheduler, TAWS, for high-performance LLM inference in cooling-regulated datacenters.

We take thermal dynamics as an explicit optimization dimension. First, we construct analytical models for GPU heat generation, thermal-throttle behavior, and multistage cooling efficiency. Using these models, we pose an optimization problem that maximizes system throughput (tokens per second) by jointly selecting inference job allocation and per-GPU dynamic voltage–frequency settings, while accounting for ambient temperature and its impact on heat dissipation. To address the problem online, we introduce TAWS, a reinforcement-learning scheduler that adapts decisions in real time.

In the evaluation, we simulate GPUs under an air-cooling system and a water-cooling system for LLM inference serving and establish a controlled cooling-regulated environment with distinct ambient-temperature setpoints. We show that in some cases the heat removal of the cooling infrastructure may not match the heat generation of GPUs in cases, and that the throughput of the LLM inference can decrease by as much as 34.2%. We implement TAWS on a mixed cluster of RTX 3090 and RTX 4090 GPUs and integrate it with the vLLM runtime. Our experiments at various ambient temperatures show that TAWS eliminates throttling events and restores up to 32.62% of lost throughput, while reducing cooling energy by 17.46% and 18.61% relative to the conventional overcooled baseline.

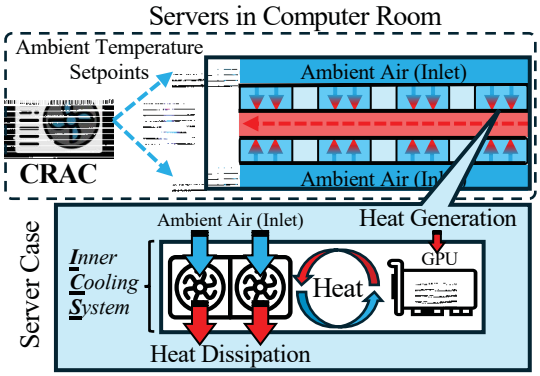
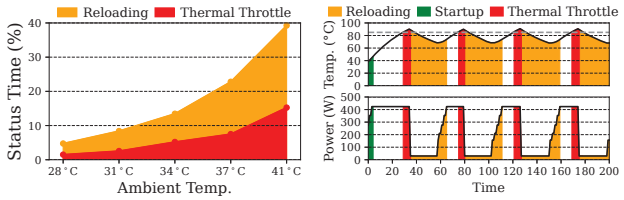


Fig. 1. The cooling systems for AI Datacenters.



(a) Thermal Throttle and Reloading Rates under Different Ambient Temperature.

(b) Runtime status in 41°C .

Fig. 2. Thermal Throttle and Data Reloading in Inference Serving

2 BACKGROUND AND RELATED WORK

The heat generation of the GPUs and the thermal-throttle mechanisms: When performing computing operations, the GPU's switching elements toggle at a high frequency, and the electrical power converts to heat. The magnitude of heat generation depends on the frequency and voltage of the core. Heat generation also depends on the manufacturing technology of GPUs [16, 47], e.g., the recent 5nm GAAFET tends to trap more heat internally than 7nm FinFETs, which can be profiled for different types of GPUs.

Heat accumulates, and if the heat that is generated exceeds the heat that is dissipated, the temperature of the GPUs will increase. High temperatures can damage the hardware. There is a *thermal limit*, the temperature at which GPUs will trigger self-protective measures to decrease their core voltage and operating frequency. This is called the *thermal throttle* mechanism [5]. When the voltage and frequency decrease, the heat that is generated will decrease. It will continue decreasing until the temperature is less than the thermal limit. The performance of the GPUs will also decrease. While thermal throttling is necessary to maintain hardware reliability, it limits GPU performance. Specifically, low voltage and frequency will slow the clock cycles of the computing units and degrade the computing performance. In some scenarios, e.g., if the temperature rises quickly, the clock cycle decreases fast, and this can lead to a CUDA kernel watchdog timeout. As a consequence, the CUDA driver would reset [37] and clear the VRAM data. For example, NVIDIA H100 has a shutdown temperature limit at 95°C, which will trigger the core rebooting and VRAM clearance [1]. In LLM inference, this leads to data reloading [34] and LLM inference operations stalling.

The heat dissipation of datacenter and the cooling regulations: The heat dissipation of a datacenter is shown in Fig. 1. There is a Computer Room Air Conditioner (CRAC) system to air-condition the server room, and there is an Inner Cooling System (ICS) to cool the servers [41]. Specifically, the external CRAC system will air-condition the air that will be sent to the server room. This inlet air sets the ambient temperature of the server room. The ICS system will use coolers (air or liquid) to cool the server's GPUs. The heat of the GPUs will be absorbed by the coolers and removed from the GPUs. The coolers will then circulate in the server room, and the heat will be dissipated in the ambient environment of the server room. Finally, heated air will be circulated out to the CRAC system of the server room.

In a datacenter, the dominant energy consumers are the CRAC system (30-40%) and the servers/GPUs (50-60%) [9, 12, 14, 15], whereas the energy consumption of the ICS system is less of a concern. Ambient temperature control is important for datacenters. A low ambient temperature can allow the coolers of the ICS to have a greater heat removal capacity; thus, reducing the probability of triggering the thermal throttle mechanism. As the performance of the GPUs is crucial, datacenters have an incentive to set unnecessarily low ambient temperatures. However, overcooling leads to a significant waste of energy. As a result, government and industry have developed guidelines for the ambient temperature settings of datacenters. For example, the ASHRAE guidelines advise that the inlet air should be 27°C [8]. The European Union Code of Conduct for Data Centers suggests a temperature of 35°C to avoid overcooling and to save energy [2, 3]. Recent research shows that increasing the ambient temperature to 41°C can maximize energy savings [55] without a serious impact on the lifespan of hardware.

Although the reference ambient temperatures were carefully studied and could satisfy the overall heat dissipation requirements, a higher temperature would lead to a higher probability of triggering the thermal throttle mechanisms. In this paper, we study LLM inference services in datacenters and show that the negative impact on the performance can be non-trivial.

Workload scheduling for LLM inference services Workload scheduling is an important research topic in datacenters. There are cluster-level [43, 48], single-GPU-level [30], real-time [10, 11, 43], day-ahead [4, 19, 27] schedulers with the objectives on system utilization [39], throughput [30, 43], service level objectives [7, 28, 53], energy [27, 42, 48], carbon [4, 10, 11, 19], and other objectives. As a first study, we look into LLM inference schedulers and study the scheduler that maximizes the throughput of tokens. We believe that our observation on the mismatch between heat dissipation that is dissipated and the heat that is generated in cooling-regulated datacenters can also lead to a review of other schedulers. We leave that to future works.

3 MOTIVATION

We perform an experiment to demonstrate how cooling-regulated datacenters can activate GPU thermal throttling. We emulate a cooling-regulated ambient environment. The key is to fix the temperature of the inlet air. This emulates the cooling regulation and also restricts the heat dissipation capacity of the ICS.

One challenge is that the temperature changes dynamically. Thus, we need to detect the change and then control it in real time. To detect temperature changes, we utilize a PT100 resistance temperature detector to continuously measure the temperature of the inlet air, and to control the temperature, we develop a closed-loop controller to precisely control the temperature to the target setpoints.

We conduct our experiment on an air-cooled NVIDIA RTX 4090. We set the inlet air temperature to {28°C , 31°C , 34°C , 37°C , 41°C }. The inference workload is Llama 3-8B [32] quantized to four bits. Such workloads allow the GPU to operate at its peak capacity. We use various prompts from IBM's Text Generation Inference Server [20], and record throughput, power, and temperature. The experiment lasts 10 hours for each distinct setpoint.

We first analyze the possibilities of the triggered thermal throttle among different setpoints; see Fig. 2(a). Then we conduct an in-depth study to determine the impact of the throughput, see Fig. 2(b), and try to explain the reason. We have three main observations. First, the probability that the RTX 4090 enters thermal throttle mode increases steadily with the ambient set-point, climbing from 1.5% at 28°C to 15.3% at 41°C , about 10 times. Second, once throttling is activated, the GPU clock is reduced, and a protective reset removes the model weights and the KV cache, forcing them to be reloaded. Inference throughput, therefore, drops while GPU power falls, and a longer recovery interval is then required. We observe that the GPU requires taking a great proportion of time to recover in Fig. 2(a), which will lead to a 34.2% reduction in throughput. Third, the combined effect is a noticeable decrease in inference throughput, confirming that higher ambient temperatures degrade LLM performance through the thermal-throttle mechanism and trigger data reloading in GPUs, as shown in Fig. 2(b). The fundamental reason is that this overheating temperature triggers thermal throttling, which reduces the core and memory clocks. CUDA kernel runtimes breach the watchdog timeout, forcing a CUDA driver reset, wiping model weights and KV cache, compelling frameworks to reload everything, and stalling inference.

4 SYSTEM MODELS AND THE PROBLEM

4.1 System Modeling

4.1.1 Heat Generation Model of GPU. As the temperature of a GPU is dynamic during operation, we model the thermal temperature $TG_i(t)$ of the GPU $\mathbf{g}_i$ at time t as follows:

$$TG_i(t) = TG_i(t-\Delta t) + \frac{[\dot{QG}_i(t) - \dot{QR}_i(t)] \cdot \Delta t}{m_i \cdot c_i}, \tag{1}$$

where $T_i(t)$ depends on the last sample temperature $TG_i(t-\Delta t)$ with interval Δt, the heat generation rate $\dot{QG}_i(t)$, and the heat dissipation rate $\dot{QR}_i(t)$, i.e., the total heat dissipated from the substance, subject to the heat capacity of this substance, i.e., the increase in temperature of this substance given a certain amount of heat injection. Here, m_i is the mass and c_i is the heat capacity. Here, the heat generation rate $\dot{QG}_i(t)$ is approximated to the power consumption P_i of the GPU $\mathbf{g}_i$ according to the first law of thermodynamics [22]. According to the power model in [23], $P_i(t)$ can be modeled as a function of its core frequency f_t and voltage v_t as:

$$\dot{QG}_i(t) \sim P_i(t) = \alpha_i v_t + \beta_i C_i v_t^2 f_t + Pc_i \tag{2}$$

Here, C_i is the gate capacitance of the GPU. Pc_i is the constant power caused by peripheral components. α_i, β_i depend on the specific LLM layer structures required to be profiled, using modern deep learning frameworks, such as PyTorch Profiler and TensorFlow Profiler.

GPU under Thermal Throttle. When the GPU reaches its maximum allowable temperature, i.e., the thermal throttling point TT_i, its voltage and frequency are restricted to maintain thermal stability, ensuring that $|T_i(t) - T_i(t-\Delta t)| = 0$. We define the set of all feasible voltage and frequency pairs under thermal throttling as $\mathcal{T}_i(TT_i) = \{(v_i, f_i)\}$. By applying Dynamic Voltage and Frequency Scaling (DVFS) techniques [24], different operational settings can be selected from $\mathbb{T}_i$ to achieve specific objectives, such as minimizing power consumption or maximizing computational throughput.

4.1.2 Heat Dissipation Model of Cooling Systems. In the inner cooling system, we model two basic kinds in practice, including air cooling and water cooling. **i) Air Cooling:** The heat dissipation rate under air cooling can be formulated as:

$$\dot{QR}_A = \rho v_a A_{hs} \cdot c_p \cdot \Delta T, \tag{3}$$

where $\Delta T = |TG - T_{ENV}|$ is the temperature difference between the heat sink and the ambient air; v_a is the air flow rate; c_p is the heat capacity of air (~ 1005 J/kg°C); ρ is the density of air (~ 1.2 kg/m^3); and A_{hs} is the contact surface area of air flow and heat sink. **ii) Water Cooling:** The heat dissipation efficiency of water cooling depends on the dissipation of water pipes $\dot{QR}_{pipe}$ based on Fourier's law of heat conduction and Newton's law of cooling [49],

$$\dot{QR}_{pipe} = \xi v_w \Delta T (1 - exp(-\mu h / v_w)) \tag{4}$$

where $\Delta TC = |TW - T_{ENV}|$ is the temperature difference between the cooling water inside the pipe and the ambient air; v_w is the water flow rate; h is the convective heat transfer coefficient of the air; and $\{\xi, \mu\}$ related to the radius of the pipe and the density of water, etc.

4.1.3 Energy Model of LLM Inference. For an LLM inference job, we model the computing latency t of a LLM layer w that runs on the GPU $\mathbf{g}_i = \{v_i, f_i\}$ as follows:

$$t_w(\mathbf{g}_i) = \frac{\text{FLOP}_w}{\text{FLOPS}(f_i) \cdot eff_{i,w}}, \tag{5}$$

where the computation complexity of layer w (FLOP_w) is measured by floating point operations (FLOPs), $eff_{i,w}$ is the hardware efficiency required to profile, and the computing capability of GPU $\mathbf{g}_i$ under frequency f_i is $\text{FLOPS}(f_i)$, counted as floating point operations per second (FLOPS), which can be formulated as [17]:

$$\text{FLOPS}(f_i) \propto N^i_{core} \cdot f_i \cdot 2, \tag{6}$$

where N^i_{core} is the GPU core number. As the ICS energy is tiny, here we assume that it always runs at its maximum power all the time, but it is ignorable. The overall energy saved during the inference can be computed as:

$$\Delta E_{inf} = \Delta E_{GPU} + \Delta E_{CRAC}. \tag{7}$$

According to Eq. 2 to Eq. 5, the GPU energy can be obtained by:

$$E_{GPU} = \sum^{t} \sum_{\mathbf{g}_i \in G} \sum_{w \in W} t_w(\mathbf{g}_i) \cdot P_i(t) \tag{8}$$

Here we assume the ambient temperature in a non-carbon-efficient AI datacenter to be 25 °C , therefore, we define that the power saving per °C from 25 °C is 5%.

4.2 The Problem

Given an LLM inference task: an LLM with layers $w \in W$ and available AI datacenter hardware including GPUs $\{\mathbf{g}_1, \mathbf{g}_2, \dots\}$, where each GPU $\mathbf{g}_i = \{v_i, f_i, t_o\}$, Thermal Throttling profiling $\{TT_i\}$, Inner Cooling Systems specifications and increasing ambient temperature ΔT_{ENV} in a carbon-efficient value, determine: the GPU job scheduler $S = \{v_i, f_i, b_i\}$ at time t where b_i is the batch size, to maximize throughput TP (number of tokens generated per second) for such a task. The scheduler should limit itself to: 1) The GPU can only execute in an overheating state (thermal throttle) for a period t_o. 2) The extra energy consumed by the GPUs compared to operating at a carbon-efficient ambient temperature should never exceed the savings from the CRAC. For example, 1°C ambient temperature increasing could at most consume 4% extra energy on computing if the proportion of energy of the cooling system to GPUs is 35% to 55% [44].

5 THE THERMAL-AWARE WORKLOAD SCHEDULER

Overview: Raising the given ambient temperature by ΔT_{ENV} can significantly reduce the CRAC's energy consumption. However, this temperature increase also weakens the heat dissipation efficiency of the ICS, making it more difficult to dissipate heat from the GPU. As a result, residual heat accumulates, causing the GPU temperature to rise. Once the temperature reaches the thermal throttling threshold, the GPU activates its thermal protection mechanism, reducing the operating frequency, core voltage, and power consumption. Although this prevents overheating, it also degrades the computing performance of the GPU and increases the latency of LLM inference tasks. Consequently, the energy consumed per inference may increase, offset by some of the gains from CRAC energy savings. Therefore, we develop a Thermal-Aware Workload Scheduler (TAWS) based on reinforcement learning.

MDP Formulation. TAWS is first formulated as a finite-horizon Markov decision process (MDP) [54], $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, R, H \rangle$.

- **State $s_k \in \mathcal{S}$.** At every control step k the agent observes

$$s_k = \left[\Delta T_{\text{ENV}},\ \{TG_{i,k}, f_{i,k}, v_{i,k}, \tau_{i,k}, b_{i,k}\}_{i=1}^{N},\ E_k^{\text{extra}},\ E_k^{\text{save}}\right],$$

where $\tau_{i,k}$ is the estimated time-to-throttle for GPU i (Sec. 5).

- **Action $a_k \in \mathcal{A}$.** The scheduler chooses (i) a subset of GPUs to execute the next minibatch, (ii) a frequency–voltage pair (f_i, v_i) for each selected GPU, and (iii) the fraction of the batch assigned to every chosen GPU.
- **Transition $\mathcal{P}$.** State evolution follows a calibrated thermal model as shown in Eq. 5.
- **Horizon H.** An episode ends at a fixed number of steps.

Reward and Safety Costs. Throughout TP, energy saving balance and hardware safety are merged into a scalar reward

$$r_k = a\,\text{TP}_k - b\ \max\!\left(0, E_k^{\text{extra}} - E_k^{\text{save}}\right) - d\ \max\!\left(0,\ TG_{i,k} - T^{\max}\right),$$

where TP_k denotes tokens-per-second during step k. Two auxiliary cost signals enforce *hard* constraints:

$$C_k^{(1)} = E_k^{\text{extra}} - E_k^{\text{save}}, \qquad C_k^{(2)} = \max_i\!\left(TG_{i,k} - T^{\max}\right).$$

Time-to-Throttle Forecasting. According to the modeling in Eq. 1 to Eq. 5, we can estimate the time $\tau_{i,k}$ required for a GPU $\mathbf{g}_i$ to reach its thermal throttling temperature TT at time t under a given inference workload as follows.

$$\tau_{i,k} = \frac{TT - TG_{i,k}}{|\dot{Q}G(k) - \dot{Q}R(k)|} \cdot m_i c_i$$

Safe Policy Optimization. We employ *Constrained Policy Optimization* (CPO) to maximize the expected discounted return while guaranteeing energy and thermal limits.

$$\max_{\pi_\theta}\ \mathbb{E}_{\pi_\theta}\Big[\sum_{k=1}^{H} d^k\, r_k\Big]\ \text{s.t.}\ \mathbb{E}_{\pi_\theta}\Big[\sum_{k=1}^{H} C_k^{(1)}\Big]\ \le\ 0, \mathbb{E}_{\pi_\theta}\Big[\sum_{k=1}^{H} C_k^{(2)}\Big]\ \le\ 0.$$

At every update, CPO solves a quadratic program that projects the vanilla policy gradient step onto the feasible set defined by the linearized constraints, delivering first-order safety guarantees.

6 IMPLEMENTATION AND EVALUATION

6.1 Implementation

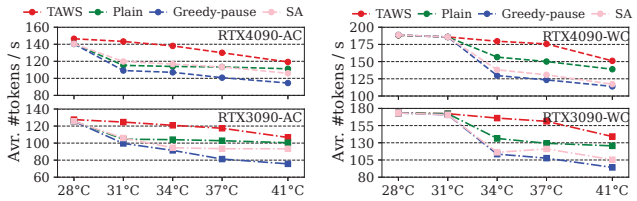
Implementation of Inference Task Workload Allocation. We establish the Thermal-Aware Workload Scheduler based on the Ray Serve [43] and vLLM [30] pod architecture, deploying a Python DVFS sidecar that interfaces with the NVIDIA Management Library [35][36] on each allocated GPU. During container initialization, the sidecar retrieves the GPU handle and sets graphics and memory clocks to 1650 MHz and 1313 MHz using `nvmlDeviceSetApplicationsClocks`, while capping board power at 500W through `nvmlDeviceSetPower ManagementLimit`. Runtime metrics such as `SM_ACTIVE`, `POWER_USAGE`, and temperature are streamed each second through a DCGM exporter [38] for status scraping. A rule engine samples utilization and chooses the next frequency and voltage setups. The NVIDIA persistence daemon preserves these settings across process restarts, and comprehensive NVML exception handling restores the last clocks if any command is rejected.

Implementation of TAWSA. Our RL workflow starts with offline calibration: one week of logs (GPU temperature, power, job arrivals) yields parameters C_i, h_i. A simulator reproduces these dynamics, adding bursts and up to 15°C ambient changes per episode. GPUs become nodes in a connected graph. Their temperature, frequency, voltage, time-to-throttle, and batch size pass through a two-layer GCN [26] whose output feeds an MLP actor–critic [31]. Training uses Proximal Policy Optimization [45] with a clipping coefficient of 0.1, an entropy bonus of 0.01, and a discount factor $\gamma = 0.995$. Each policy update covers 8000 simulation steps, divided into mini-batches of 512, and is optimized with Adam [25] at a learning rate of 3×10^{-4}. The policy converges within five hours on two RTX 4090.

6.2 Evaluation

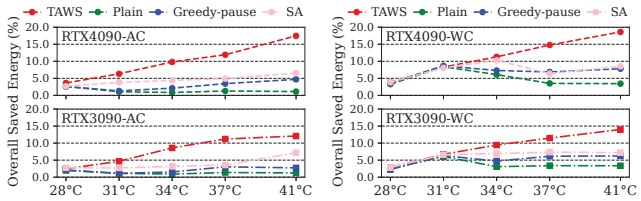
6.2.1 *Evaluation Setups.*

Testbeds. We evaluate the Thermal-aware Workload Scheduler on



(a) Air-cooling ICS (b) Water-cooling ICS

Fig. 3. Average throughput under distinct ambient conditions.



(a) Air-cooling ICS (b) Water-cooling ICS

Fig. 4. Overall energy saving under distinct ambient conditions.

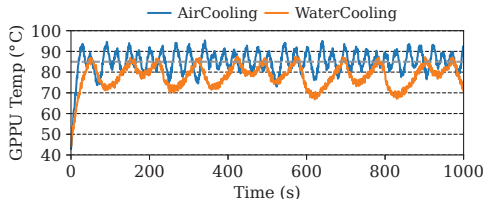


Fig. 5. Runtime temperature of TAWS on RTX 4090 featured with air-cooling and water-cooling in 41°C ambient during LLM inference.

a workstation that features an Intel Core i9-13900K CPU and 64GB of RAM. The system has two GPU models: RTX3090 and RTX4090, with standard air cooling denoted RTX3090-AC / RTX4090-AC and water cooling RTX3090-AC / RTX4090-WC.

LLM Inference Tasks. We implement a representative small-scale LLM: Llama3-8B [32], quantized into 4-bit. Larger models were intended to be excluded because the bottleneck onboard memory of our test GPUs would limit full utilization of their compute units and constrain the admissible batch size to at most one or two sequences. We adopt a diverse set of prompts from IBM's Text Generation Inference Server [20] to emulate production-level workloads and capture realistic usage patterns for LLM inference services.

Ambient Configuration. We configure the ambient environment in five levels, including {28°C , 31°C , 34°C , 37°C , 41°C }. To isolate the effects of CRAC, the workstation is installed in a 2m×2m environment served by a central air conditioning unit. The ambient temperature is continuously verified with a calibrated sensor to ensure that it remains within the prescribed limits. To emulate elevated ambient conditions with high precision, we deploy PT100 resistance temperature detectors that continuously monitor the inlet air and feed a closed-loop controller, which adjusts the ICS settings to maintain the desired temperature profile.

Baselines. We compared TAWS with three alternative strategies. i) ***Plain***. The scheduler uses the default workload allocation of vLLM with no temperature feedback. When the GPU hits its throttle limit, the firmware lowers frequency and voltage, yet operation continues near that critical temperature, exposing the hardware to potential long-term damage. We include this unsafe configuration solely as a worst-case reference; it is not used in production. ii) ***Greedy-Pause***. When GPU temperature hits the throttle limit, the scheduler pauses inference until the device cools to a preset resume temperature, then continues execution. iii) ***Simulated Annealing (SA)***. A simulated annealing controller proactively tunes frequency, voltage, and batch size as the throttle point approaches, trying to keep temperature below the limit while maintaining throughput.

Metrics. Performance is quantified as throughput, measured by generated tokens per second. We also record the proportion of the total energy consumption saved for each experimental configuration.

6.2.2 Evaluation Results. Fig. 3 shows the average inference throughput of all baselines on RTX 4090 and RTX 3090 across a range of ambient set-points. Under air cooling (Fig.,3(a)) on RTX 4090, throughput at 28°C remains stable because the GPUs can operate below their thermal throttle within the heat dissipation capability. However, once the inlet temperature rises to 31°C , every baseline experiences a noticeable decrease. Here, TAWS outperforms Plain, Greedy-pause, SA, 24.53%, 31.29%, 19.63%, respectively, with its advantage peaking at 41°C . Plain suffers because it continues running near the thermal limit, reducing effective compute frequency; Greedy-pause wastes inference time while waiting for the device to completely cool, although it sheds heat quickly. Compared with the RTX 4090, the RTX 3090 shows a smaller relative drop due to the denser 5 nm process of the RTX 4090. Even so, TAWS can outperform Plain, Greedy-pause, SA, 8.79%, 32.62%, 29.01%, respectively, in 41°C . In Fig. 3(b), a similar trend emerges, but throttling does not begin until the temperature exceeds 34°C . The higher heat capacity and conductivity of water delay the onset of thermal limits by carrying heat away from the chip more effectively than air. At 41°C , the TAWS on the RTX 4090 again leads Plain, Greedy-pause, SA, and the remaining baselines by the largest margin, 8.79%, 32.62%, 29.05%, respectively, confirming that our TAWS remains beneficial even when a superior inner cooling system is deployed.

Fig. 4 compares the total energy-saving ratio of all schedulers, with both GPUs and cooling electricity included. Raising the ambient set-point boosts savings for all baselines because the chiller load falls, but TAWS yields the greatest benefit at every temperature. By avoiding thermal throttling, it shortens the inference time, so the integral of power over time drops sharply. At the highest inlet temperature, TAWS reduces the total energy by 17.46% under air cooling and 18.61% under water cooling. The results also show the larger efficiency potential of water-cooled inner loops, indicating that future AI facilities can unlock even deeper savings by pairing advanced water cooling with thermal-aware scheduling. As such, in an AI datacenter with 10,000 A100 cards in West Virginia [50], the CO_2 saved per year is about 6,000 tons if the ambient is set to 41°C .

Fig. 5 shows that during inference, TAWS adaptively controls the frequency, voltage, and batch size of the RTX 4090 GPU under the air-cooling system and the water-cooling system in 41°C ambient temperature. We observe that water cooling has less average temperature stress than air cooling. Water cooling reduces the average GPU temperature by about 10°C and caps the peak at about 83°C . Air cooling, on the contrary, oscillates between 75°C and 94°C and triggers the throttle threshold more frequently. The thermal swing under water cooling remains within ±5°C , while air cooling sees ±9°C . Consequently, TAWS has higher throughput and saves more energy. These results confirm that stronger inner cooling provides extra heat dissipation capacities, allowing TAWS to hold higher frequencies to avoid thermal throttling, compared to air cooling.

7 CONCLUSION AND FUTURE WORK

In this paper, we studied cooling-regulated datacenters, where the cooling temperature of datacenter server rooms is regulated by standard references to avoid overcooling and save energy. We observed that such cooling-regulated datacenters introduce new challenges to high-performance AI computing because the heat dissipated by the cooling system may, in some cases, not match the heat generated by the GPUs. This can trigger GPU thermal throttling and degrade system performance. In this paper, we studied LLM inference services, and we developed a new thermal-aware scheduler for LLM inference services that optimizes the LLM inference throughput by taking GPU voltage and frequency into account.

As an early work, we evaluated our scheduler only on single-category GPU clusters. We plan a deployment on multi-GPU clusters, covering diverse datacenter GPUs. We will validate our system in the physical environment and develop CFD-based Fluent simulations of a realistic environment. We also believe that our observation of the mismatch between heat dissipation and heat generation in cooling-regulated datacenters could lead to a revisiting of other schedulers in datacenters. With advanced schedulers, we believe that datacenters can be more tolerant of higher cooling temperatures and can save significant amounts of energy.

ACKNOWLEDGMENTS

Dan Wang's work is supported in part by RGC GRF 15200321, 15201322, 15230624, ITC ITF-ITS/056/22MX, ITS/052/23MX, and PolyU 1-CDKK, G-SAC8.

REFERENCES

[1] 2022. *NVIDIA H100 PCIe GPU Product Brief.* Product Brief PB-11133-001_v02. NVIDIA Corporation, Santa Clara, CA. https://www.nvidia.com/content/dam/en-zz/Solutions/gtcs22/data-center/h100/PB-11133-001_v01.pdf Accessed: 2 July 2025.

[2] Mark Acton, Paolo Bertoldi, and John Booth. 2024. *2024 Best Practice Guidelines for the EU Code of Conduct on Data Centre Energy Efficiency.* JRC Technical Note JRC136986. European Commission, Joint Research Centre, Ispra, Italy. https://e3p.jrc.ec.europa.eu/sites/default/files/2024-04/JRC136986_2024_best_practice_guidelines.pdf European Code of Conduct for Energy Efficiency in Data Centres, 15th Edition.

[3] Mark Acton, John Booth, and Daniele Paci. 2025. *Best Practice Guidelines for the EU Code of Conduct on Data Centre Energy Efficiency.* JRC Technical Report EUR40267 EN EUR40267. Publications Office of the European Union, Joint Research Centre, European Commission. https://doi.org/10.2760/9449356

[4] Bilge Acun, Benjamin Lee, Fiodar Kazhamiaka, Kiwan Maeng, Udit Gupta, Manoj Chakkaravarthy, David Brooks, and Carole-Jean Wu. 2023. Carbon Explorer: A Holistic Framework for Designing Carbon-Aware Datacenters. In *Proc. of the 28th ACM International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS '23).* Vancouver, BC, Canada.

[5] Amirhossein Ahmadi, Hazem A Abdelhafez, Karthik Pattabiraman, and Matei Ripeanu. 2023. Edgeengine: A thermal-aware optimization framework for edge inference. In *Proc. of the Eighth ACM/IEEE Symposium on Edge Computing.*

[6] Odi Fawwaz Alrebei, Bushra Obeidat, Tamer Al-Radaideh, Laurent M Le Page, Sally Hewlett, Anwar H Al Assaf, and Abdulkarem I Amhamed. 2022. Quantifying CO2 emissions and energy production from power plants to run HVAC systems in ASHRAE-based buildings. *Energies* 15, 23 (2022), 8813.

[7] Thomas Anderson, Adam Belay, Mosharaf Chowdhury, Asaf Cidon, and Irene Zhang. 2023. Treehouse: A case for carbon-aware datacenter software. *ACM SIGENERGY Energy Informatics Review (HotCarbon'22)* 3, 3 (2023), 64–70.

[8] ASHRAE Technical Committee 9.9. 2021. *Thermal Guidelines for Data Processing Environments* (fifth edition, revised and expanded ed.). American Society of Heating, Refrigerating and Air-Conditioning Engineers (ASHRAE), Peachtree Corners, GA. https://www.ashrae.org/file%20library/technical%20resources/bookstore/supplemental%20files/therm-gdlns-5th-r-e-refcard.pdf

[9] Alyssa Bersine. 2025. Reducing Data Center Peak Cooling Demand and Energy Costs With Underground Thermal Energy Storage. https://www.nrel.gov/news/detail/program/2025/reducing-data-center-peak-cooling-demand-and-energy-costs-with-underground-thermal-energy-storage. National Renewable Energy Laboratory news feature.

[10] Roozbeh Bostandoost, Walid A Hanafy, Adam Lechowicz, Noman Bashir, et al. 2024. Data-driven Algorithm Selection for Carbon-Aware Scheduling. *ACM SIGENERGY Energy Informatics Review (HotCarbon'24)* 4, 5 (2024), 148–153.

[11] Roozbeh Bostandoost, Adam Lechowicz, et al. 2024. LACS: Learning-Augmented Algorithms for Carbon-Aware Resource Scaling with Uncertain Demand. In *Proc. of 15th ACM International Conference on Future and Sustainable Energy Systems.*

[12] Curtis Breville. 2025. *Your CRAC Problem is Affecting Us All.* ByteBridge Blog. https://www.bytebt.com/crac-problem-is-affecting-us/

[13] Andrew A Chien, Liuzixuan Lin, Hai Nguyen, Varsha Rao, Tristan Sharma, et al. 2023. Reducing the Carbon Impact of Generative AI Inference (today and in 2035). In *Proc. of the 2nd workshop on sustainable computer systems (HotCarbon'23).*

[14] Boyd Corporation. 2023. *Energy Consumption in Data Centers: Air versus Liquid Cooling.* https://www.boydcorp.com/blog/energy-consumption-in-data-centers-air-versus-liquid-cooling.html

[15] Klaus Dafinger. 2024. *Meeting data center cooling demands in the AI era.* Data Center Dynamics. https://www.datacenterdynamics.com/en/opinions/meeting-data-center-cooling-demands-in-the-ai-era/

[16] Semiconductor Engineering. 2020. Quantum Effects At 7//5nm. https://semiengineering.com/quantum-effects-at-7-5nm/ Accessed: 2025-05-19.

[17] Ahmad Faiz, Sotaro Kaneda, Ruhan Wang, Rita Osi, Prateek Sharma, Fan Chen, and Lei Jiang. 2024. LLMCARBON: MODELING THE END-TO-END CARBON FOOTPRINT OF LARGE LANGUAGE MODELS. In *Proc. of International Conference on Learning Representations (ICLR'24).*

[18] Wedan Emmanuel Gnibga, Andrew A. Chien, Anne Blavette, and Anne-Cécile Orgerie. 2024. FlexCoolDC: Datacenter Cooling Flexibility for Harmonizing Water, Energy, Carbon, and Cost Trade-offs. In *Proc. 15th ACM Int. Conf. on Future Energy Systems (e-Energy '24).* Singapore.

[19] Walid A. Hanafy, Qianlin Liang, Noman Bashir, Abel Souza, David Irwin, and Prashant Shenoy. 2024. Going Green for Less Green: Optimizing the Cost of Reducing Cloud Carbon Emissions. In *Proc. of the 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS '24).* La Jolla, CA, USA.

[20] IBM Research AI. 2025. Text Generation Inference Server (TGIS). https://github.com/IBM/text-generation-inference.

[21] Ziheng Jiang, Haibin Lin, Yinmin Zhong, Qi Huang, Yangrui Chen, Zhi Zhang, Yanghua Peng, Xiang Li, et al. 2024. {MegaScale}: Scaling large language model training to more than 10,000 {GPUs}. In *21st USENIX Symposium on Networked Systems Design and Implementation (NSDI 24).*

[22] Baris Burak Kanbur, Chenlong Wu, Simiao Fan, et al. 2020. Two-phase liquid-immersion data center cooling system: Experimental performance and thermoeconomic analysis. *International Journal of Refrigeration* 118 (2020), 290–301.

[23] Vijay Kandiah, Scott Peverelle, Mahmoud Khairy, Junrui Pan, Amogh Manjunath, Timothy G Rogers, Tor M Aamodt, and Nikos Hardavellas. 2021. AccelWattch: A power modeling framework for modern GPUs. In *Proc. of the IEEE/ACM International Symposium on Microarchitecture (MICRO'21).* Virtual Event.

[24] Seyeon Kim, Kyungmin Bin, et al. 2021. zTT: Learning-Based DVFS with *Zero Thermal Throttling* for Mobile Devices. In *Proc. of the 19th ACM International Conference on Mobile Systems, Applications, and Services (MobiSys '21).*

[25] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[26] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *International Conference on Learning Representations (ICLR).* arXiv:1609.02907.

[27] kube-green Maintainers. 2025. kube-green: A Kubernetes Operator to Reduce CO_2 Footprint of Your Clusters. https://github.com/kube-green/kube-green.

[28] Munkyu Lee, Sihoon Seong, Minki Kang, Jihyuk Lee, Gap-Joo Na, In-Geol Chun, Dimitrios Nikolopoulos, and Cheol-Ho Hong. 2024. ParvaGPU: Efficient Spatial GPU Sharing for Large-Scale DNN Inference in Cloud Environments. In *Proc. Int. Conf. for High Performance Computing, Networking, torage and Analysis (SC '24)*.

[29] Amy Li, Sihang Liu, and Yi Ding. 2024. Uncertainty-aware decarbonization for datacenters. *ACM SIGENERGY Energy Informatics Review (HotCarbon'24)* 4, 5 (2024), 141–147.

[30] Yao Li, Diyi Qin, Junjie Li, Jiannan Wang, and Ion Stoica. 2023. vLLM: Easy and Fast Large Language Model Serving with PagedAttention. *arXiv:2309.02983* (2023).

[31] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. 2015. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971* (2015).

[32] Meta AI. 2024. Llama 3 8B. https://huggingface.co/meta-llama/Meta-Llama-3-8B.

[33] Sophia Nguyen, Beihao Zhou, Yi Ding, and Sihang Liu. 2024. Towards sustainable large language model serving. *ACM SIGENERGY Energy Informatics Review (HotCarbon'24)* 4, 5 (2024), 134–140.

[34] NVIDIA Corporation. 2024. NVIDIA GPU Debug Guidelines, v560 Section 4.1. https://docs.nvidia.com/deploy/gpu-debug-guidelines/.

[35] NVIDIA Corporation. 2024. *NVIDIA Management Library (NVML) API Reference Guide.* NVIDIA. https://docs.nvidia.com/deploy/nvml-api/

[36] NVIDIA Corporation. 2024. *nvidia-smi Command Line Utility: User Guide.* NVIDIA. https://developer.nvidia.com/nvidia-system-management-interface

[37] NVIDIA Corporation. 2024. XID Errors – Release r555 Documentation. https://docs.nvidia.com/deploy/xid-errors/.

[38] NVIDIA Corporation. 2025. *Data Center GPU Manager (DCGM) User Guide.* NVIDIA. https://docs.nvidia.com/datacenter/dcgm/latest/user-guide/index.html Version 4.2.3 — accessed 19 May 2025.

[39] NVIDIA Corporation. 2025. TensorRT-LLM. https://github.com/NVIDIA/TensorRT-LLM.

[40] Pratyush Patel, Esha Choukse, Chaojie Zhang, Íñigo Goiri, Brijesh Warrier, et al. 2024. Characterizing power management opportunities for llms in the cloud. In *Proc. of the 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS'24)*.

[41] Xiangyu Pei, Kai Wang, Yujie Zhang, Liang Zhang, and Jia Rao. 2022. CoolEdge: Hotspot-Relievable Warm-Water Cooling for Energy-Efficient Edge Datacenters. In *Proc. of the 27th ACM International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS'22)*.

[42] Feitong Qiao, Yiming Fang, and Asaf Cidon. 2024. Energy-Aware Process Scheduling in Linux. *ACM SIGENERGY Energy Informatics Review* 4, 5 (2024), 91–97.

[43] Ray Project Contributors. [n. d.]. *Ray Serve: Scalable and Programmable Serving — Ray 2.46.0.* Anyscale Inc. https://docs.ray.io/en/latest/serve/index.html Ray documentation.

[44] Isabelle Riu, Dieter Smiley, Stephen Bessasparis, and Kushal Patel. 2024. *Load Growth Is Here to Stay, but Are Data Centers?: Strategically Managing the Challenges and Opportunities of Load Growth.* White Paper. San Francisco, CA, USA. https://www.ethree.com/wp-content/uploads/2024/07/E3-White-Paper-2024-Load-Growth-Is-Here-to-Stay-but-Are-Data-Centers-2.pdf

[45] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal Policy Optimization Algorithms. *arXiv:1707.06347* (2017).

[46] Singapore Standards Council. 2023. Singapore Standard SS 697:2023: Deployment and Operation of Data Centre IT Equipment under Tropical Climate. Standard.

[47] Matej Špet'ko, Ondřej Vysockỳ, et al. 2021. Dgx-a100 face to face dgx-2—performance, power and thermal behavior evaluation. *Energies* 14, 2 (2021), 376.

[48] Jovan Stojkovic, Chaojie Zhang, Íñigo Goiri, et al. 2025. Dynamollm: Designing llm inference clusters for performance and energy efficiency. In *2025 IEEE International Symposium on High Performance Computer Architecture (HPCA'25)*.

[49] Jovan Stojkovic, Chaojie Zhang, Íñigo Goiri, Esha Choukse, Haoran Qiu, Rodrigo Fonseca, Josep Torrellas, and Ricardo Bianchini. 2025. Tapas: Thermal-and power-aware scheduling for LLM inference in cloud platforms. In *Proc. of the 30th ACM International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS'25)*.

[50] U.S. Energy Information Administration. 2024. Virginia Electricity Profile 2023. https://www.eia.gov/electricity/state/virginia/.

[51] Duc Van Le, Jing Zhou, Rongrong Wang, Rui Tan, and Fei Duan. 2024. Impacts of Increasing Temperature and Relative Humidity in Air-Cooled Tropical Data Centers. *IEEE Transactions on Sustainable Computing* (2024).

[52] Ruihang Wang, Zhiwei Cao, Xin Zhou, Yonggang Wen, and Rui Tan. 2024. Green Data Center Cooling Control via Physics-Guided Safe Reinforcement Learning. In *Proc. of 15th ACM International Conference on Future Energy Systems*. Singapore.

[53] Bingyang Wu, Yinmin Zhong, Zili Zhang, Shengyu Liu, Fangyue Liu, Yuanhang Sun, Gang Huang, Xuanzhe Liu, and Xin Jin. 2024. Fast Distributed Inference Serving for Large Language Models. In *Proc. 18th USENIX Symposium on Operating Systems Design and Implementation (OSDI '24)*. Santa Clara, CA, USA.

[54] Yaodan Xu, Jingzhou Sun, Sheng Zhou, and Zhisheng Niu. 2023. Smdp-based dynamic batching for efficient inference on gpu-based platforms. In *Proc. of IEEE International Conference on Communications (ICC'23)*.

[55] Yingbo Zhang, Hangxin Li, and Shengwei Wang. 2023. The global energy impact of raising the space temperature for high-temperature data centers. *Cell Reports Physical Science* 4, 10 (2023).

[56] Yang Zhou, Feng Liang, Ting-wu Chin, and Diana Marculescu. 2022. Play it cool: Dynamic shifting prevents thermal throttling. *arXiv:2206.10849* (2022).