Cooling as You Wish: Component-Level Cooling for Heterogeneous Edge Datacenters

Fangming Liu, *Senior Member, IEEE*, Qiangyu Pei*, Shutong Chen, Yongjie Yuan, Qixia Zhang, Xinhui Zhu, Ziyang Jia, Fei Xu, Dong Zhang, Bingheng Yan

Abstract—As computing shifts toward the edge, edge datacenters are becoming essential for supporting diverse real-time applications. Unlike traditional cloud datacenters, edge datacenters face unique cooling challenges due to their requirements for *proximity to end users*, *high density*, and *hardware heterogeneity*. While warm water cooling is a promising technique for this infrastructure, current one-size-fits-all cooling strategies significantly compromise efficiency due to severe inter- and intra-component hotspots. In this work, we present CoolEdge^{+}, a cost-effective component-level water cooling system for enhancing the cooling efficiency of edge datacenters. Specifically, CoolEdge^{+} dynamically adjusts the inlet water temperature for each component through a carefully designed water circulation architecture to mitigate inter-component hotspots. To address intra-component hotspots, it employs vapor chamber–based cold plates that rapidly dissipate heat without manual intervention or additional energy consumption. We further design a fine-grained cooling control framework that leverages a well-managed power capping approach to decide on customized inlet water temperatures and hardware power limits. Based on a hardware prototype and a real-world trace from Alibaba PAI, evaluation results show that CoolEdge^{+} reduces cooling energy consumption by up to 27.19% compared to existing coarse-grained systems, while maintaining performance guarantees. Compared to the state-of-the-art CoolEdge, CoolEdge^{+} saves 35.24% more cooling costs with comparable energy consumption and no latency violations.

Index Terms—edge datacenter energy, warm water cooling, heterogeneity, hotspot mitigation, vapor chamber

1 Introduction

Edge datacenters are emerging as a crucial component of edge computing infrastructure. To provide real-time services to end-users, the edge datacenters are being widely deployed in urban areas for low network transmission overhead. It is predicted that the global edge computing market will expand from around 14 billion dollars in 2024 to around $182 billion dollars by 2032, representing a compound annual growth rate of 38.2% [1]. This significant growth emphasizes the importance of edge datacenters in meeting the needs of emerging edge services. Although the power rating of a single edge datacenter is generally low, e.g., in the range of tens to hundreds of kilowatts, which is typically three orders of magnitude smaller than a cloud datacenter [2], their increasing number will inevitably bring a heavy energy burden. It is estimated that by 2028, the energy demand of edge datacenters is comparable to the total electricity consumption of global datacenters in 2020 [3], [4]. With the rapid development of technologies like artificial intelligence (AI), the Internet of Things, and 5G, more and more data processing and analysis tasks will be completed at the edge, further driving the growth in the energy demand of edge datacenters.

Several leading cloud service providers have explored modular and distributed datacenter architectures to support edge computing. For example, Azure has designed modular edge datacenters for complex environments, such as emergency rescue, military missions, and mineral exploration, to meet the demands of low-latency, high-intensity, and secure computing at the edge [5]. Tencent Cloud has opened its first edge datacenter to provide real-time services like video processing, cloud gaming, and smart healthcare [6]. While many traditional cloud workloads like Web services can be easily processed by a central processing unit (CPU), those emerging computational edge workloads like real-time analytics rely heavily on accelerators for computation acceleration, such as graphics processing units (GPUs). Therefore, to support diverse performance-critical edge applications, edge servers typically need to be equipped with enough heterogeneous hardware components, which also results in a high power provisioning to edge servers [7].

Despite the small power capacity of an edge datacenter, its power density is generally much higher than that of a cloud datacenter due to space restrictions and dense deployment of heterogeneous hardware components. In

- *This work was supported in part by the National Key Research & Development (R&D) Plan under Grant 2022YFB4501703, in part by National Science Foundation of China under Grant 62232012, in part by the Major Key Project of PCL under Grant PCL2024A06 and PCL2022A05, and in part by the Shenzhen Science and Technology Program under Grant RCJC20231211085918010. (Corresponding author: Qiangyu Pei)*

- *Fangming Liu is with Huazhong University of Science and Technology, and Peng Cheng Laboratory, China. E-mail: fangminghk@gmail.com*
- *Qiangyu Pei, Shutong Chen, Yongjie Yuan, Qixia Zhang, and Xinhui Zhu are with the National Engineering Research Center for Big Data Technology and System, the Services Computing Technology and System Lab, Cluster and Grid Computing Lab in the School of Computer Science and Technology, Huazhong University of Science and Technology, 1037 Luoyu Road, Wuhan 430074, China. E-mail: {peiqiangyu, zhangqixia427, xhzhu}@hust.edu.cn, shutongcs@gmail.com, jayayuan@outlook.com*
- *Ziyang Jia is with the School of Computer Science and Technology, Huazhong University of Science and Technology, 1037 Luoyu Road, Wuhan 430074, China. E-mail: zjia016@ucr.edu*
- *Fei Xu is with the School of Computer Science and Technology, East China Normal University, China. E-mail: fxu@cs.ecnu.edu.cn*
- *Dong Zhang and Bingheng Yan are with Jinan Inspur Data Co.,Ltd. E-mail: {zhangdong, yanbh}@inspur.com*

Manuscript received xxxx xx, 2025; revised xxxx xx, 2025.

Table 1: Thermal Specifications of Some IT Hardware Components

Hardware type	Intel Xeon E5-2680 v4 CPU	Intel Xeon 6980P CPU	Nvidia GeForce RTX 2080 Ti GPU	Nvidia H100 80GB PCIe GPU	DRAM	Samsung 983 DCT SSD
MOT (°C)	86	80	89	87	85	70
TDP (W)	120	500	250	350	Typically ≤ 10	Read: 8.7, Write: 10.6

particular, the rack density in edge datacenters can reach as high as 35 kW/rack, whereas in cloud datacenters, it typically ranges from 6 to 12 kW/rack [3], [8]. Inspur has introduced the NE5260M5 edge server, featuring a chassis depth that is only 65% of the standard defined by the Open Compute Project [9]. This compact design not only saves space and offers greater deployment flexibility but also increases power density due to the use of short racks and compact aisle layouts.

Based on the above industrial examples, we summarize three unique requirements of edge datacenters as follows: *proximity to end-users*, *heterogeneity*, and *high density*. While these characteristics enable edge datacenters to better support edge computing, they also render previously widely adopted cooling techniques inefficient or even impractical. Specifically, the free cooling technique requires access to low ambient temperatures and natural cooling sources such as cold outdoor air [10], which often conflicts with the requirement for edge deployments to be close to end-users. Additionally, high density and heterogeneity further exacerbate the challenges of effective cooling. As power density increases significantly, the air cooling technique struggles to satisfy the cooling demand because of its low thermal capacity [11] and the difficulty in managing airflow in compact rack and aisle configurations. This is particularly problematic when dealing with thermal imbalances across heterogeneous hardware components. According to a report by Schneider Electric, air cooling becomes inefficient for rack densities exceeding 20 kW/rack [11]. In contrast, water cooling offers a promising alternative for edge datacenters, thanks to water's significantly higher density (775×), specific heat capacity (4.18×), and thermal conductivity (23.4×) compared to air [11], making it well-suited for efficient heat removal in high-density and heterogeneous edge scenarios.

As compared with conventional cold water cooling [12], recent literature [13] advocates the use of warm water cooling (e.g., 40°C∼50°C) to reduce cooling energy waste by avoiding the over-cooling of servers operating at low utilization. However, existing *coarse-grained* warm water cooling approaches can be highly inefficient for edge datacenters due to the severe hotspot issue at multiple levels. On the one hand, imbalanced hardware utilization as well as the different thermal specifications of heterogeneous components leads to inter-component thermal imbalance. To cool down a small subset of hotspot components, the global cooling water must be lowered to an unnecessarily low temperature. This over-provisioning results in inefficiency, as non-hotspot components also receive excessively cold water. On the other hand, thermal imbalance also occurs within individual components due to the uneven utilization and varying thermal characteristics of subunits, further increasing cooling costs and impacting hardware reliability [14].

In summary, conventional one-size-fits-all water cooling systems lead to significant cooling waste for dealing with local hotspots. To tackle this issue, we propose CoolEdge$^+$, a cost-effective and practical component-level water cooling system designed to improve cooling efficiency in high-density and heterogeneous edge datacenters. Specifically, our contributions are as follows:

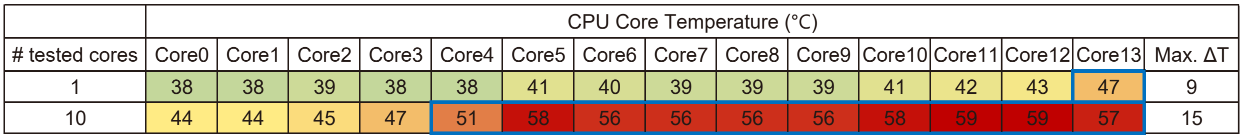
- We propose a cost-effective component-level water cooling architecture CoolEdge$^+$, featuring two key designs. First, through fine-grained cooling control enabled by a dual-circulation water system, CoolEdge$^+$ efficiently mitigates inter-component hotspots. Second, with our newly developed vapor chamber-based cold plates, intra-component hotspots can be effectively dissipated without manual intervention or additional energy consumption.
- We design a fine-grained cooling control mechanism to implement the customized cooling control. By incorporating a well-managed power capping approach, CoolEdge$^+$ can achieve similar cooling efficiency improvements as our preliminary work, CoolEdge, but at significantly lower cooling costs, and enables a flexible trade-off between cooling energy efficiency and hardware performance while ensuring hardware safety.
- We build a hardware prototype to validate the practicability of CoolEdge$^+$, and conduct datacenter-level simulations to evaluate its effectiveness in addressing multi-level hotspots and reducing cooling costs. The evaluation results reveal that compared with the existing coarse-grained water cooling architecture, CoolEdge$^+$ reduces cooling energy consumption by up to 27.19%. A cost saving analysis further estimates that CoolEdge$^+$ can save up to $3,598,400 yearly in a city, a 35.24% improvement over the state-of-the-art CoolEdge.

2 Background and Motivation

In this section, we first investigate the hotspot issue both across and within hardware components. Then, we discuss the necessity of designing a new cooling architecture tailored to the unique requirements of edge datacenters.

2.1 The Hotspot Issue in Edge Datacenters

In conventional coarse-grained water cooling systems, different components share the same inlet water temperature and flow rate despite having distinct cooling demands. To ensure adequate cooling for high-utilization components operating at elevated temperatures, the inlet water temperature is typically set at a low level (e.g., 7°C∼10°C [13]),which results in a lot of cooling energy waste. While some cloud providers have proposed raising the water temperature set-point (i.e.,



	CPU Core Temperature (°C)														
# tested cores	Core0	Core1	Core2	Core3	Core4	Core5	Core6	Core7	Core8	Core9	Core10	Core11	Core12	Core13	Max. ΔT
1	38	38	39	38	38	41	40	39	39	39	41	42	43	47	9
10	44	44	45	47	51	58	56	56	56	56	58	59	59	57	15

Figure 1: Temperature variation among CPU cores when stressing different numbers of cores.

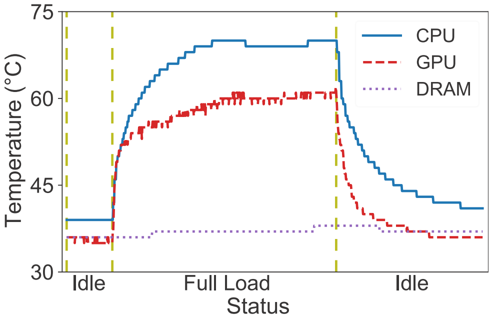


Figure 2: Temperature variation of heterogeneous hardware.

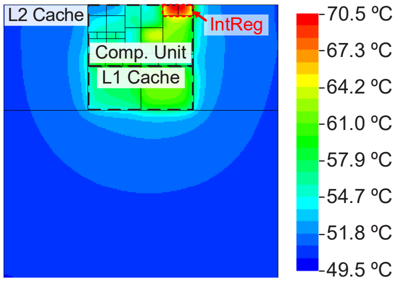


Figure 3: Temperature distribution within a CPU core.

adopting warm water cooling) to reduce cooling costs, existing coarse-grained warm water cooling architectures suffer from a severe hotspot issue, where high-utilization components are prone to overheating, leading to performance degradation and potentially affecting hardware reliability [15]. What is worse, compared to cloud datacenters, the hotspot issue is even more pronounced in edge datacenters due to the requirements of high density and heterogeneity, along with the skewed hardware utilization patterns of edge workloads [7] and the non-ideal ambient conditions at the edge. In the following, we analyze two kinds of thermal imbalance in edge datacenters: inter-component and intra-component thermal imbalance.

The inter-component hotspot issue: Previous literature [13], [16] has demonstrated that the hotspot issue exists among homogeneous hardware components, such as CPUs, GPUs, or dynamic random-access memories (DRAMs) of the same type. More recent research [17], [18] has further highlighted significant component-level hotspots when running popular AI workloads. For over 3,000 GPUs operating under high load and similar inlet temperatures within the same datacenter, the temperature disparity can reach nearly 30°C [17]. When serving different models, running the YOLOv8x model exclusively increases GPU temperature by only 2°C, whereas running the large Diffusion model can raise GPU temperature by up to 10°C [18], indicating a substantial inter-component thermal imbalance in multi-model inference serving scenarios.

For heterogeneous hardware components, the thermal imbalance becomes even more pronounced due to their differing thermal specifications and dynamic characteristics. Table 1 shows the Maximum Operating Temperature (MOT) and Thermal Design Power (TDP) specifications for various hardware types. As observed, there are considerable differences in both MOT and TDP across different hardware types, especially between compute hardware and memory or storage hardware. Additionally, we evaluate the dynamic thermal characteristics of heterogeneous components under varying load conditions. As illustrated in Figure 2[1], these components show distinct operating temperatures and temperature variation rates even in the same status. Usually, the operating temperatures of compute and memory hardware are above and below 40°C, respectively. Moreover, when transitioning between different load conditions, compute hardware exhibits significantly faster temperature variations compared to memory hardware and stabilizes at a new equilibrium temperature more quickly.

The intra-component hotspot issue: Considering the hardware type and workload characteristics, different internal units inside a component may operate at different utilization and power levels, leading to hotspot formation at the chip level. We investigate this hotspot issue in three scenarios: among CPU cores, within a CPU core, and within a GPU.

(1) Hotspots among CPU cores: A CPU typically consists of multiple processing units, i.e., cores. To analyze temperature variations among CPU cores, we conduct an experiment by applying stress to different numbers of cores, as illustrated in Figure 1[1]. The region enclosed by the blue box highlights the stressed cores. In particular, when ten cores are under stress, the maximum temperature difference reaches as high as 15°C.

(2) Hotspots within a CPU core: A micro CPU core contains several subunits, ranging from low-powered cache units to high-powered computing units. Using the HotSpot simulator [19], we analyze the temperature distribution within a CPU core while running integer workloads, as presented in Figure 3. As we can see, there exist several hotspots, especially in the integer register unit marked as IntReg. In particular, the temperature difference between computing units and cache units can be over 20°C.

(3) Hotspots within a GPU: A GPU consists of multiple functional units, including computing units, memory units, etc. According to the measurement result of an AMD GPU for a stress test, the internal hotspots can exceed 100°C and the maximum temperature difference is over 30°C. A recent study [17] further shows that when running large language models (LLMs), the GPU memory generally operates at a

[1] Details of the hardware components are presented in Section 5.1.

lower temperature than the GPU cores, though in some cases, it can be around 10°C higher.

2.2 Motivation for a New Cooling Architecture for Edge Datacenters

Some software-based solutions can be implemented to mitigate hotspots in a cloud datacenter, including power throttling [15], [20], [21], thermal-aware workload placement [16], [22], [23], [24], and workload deferral [10], [23]. However, applying software-based solutions alone could show poor performance, as the space for sacrificing performance for eliminating hotspots is marginal when serving edge workloads with strict requirements on performance like processing latency. For example, avoiding hotspots by lowering hardware frequency largely will degrade hardware performance and is likely to cause latency violations. Also, for many mission-critical edge applications, such as smart traffic management, there could be no deferrable workloads, and therefore, hotspots can emerge constantly. As a result, it is essential to combine the cooling architecture design to solve the hotspot issue for general cases at the edge.

Jiang et al. [13] propose a thermoelectric cooler-based (TEC-based) solution to address the hotspot issue in homogeneous cloud datacenters with only CPUs. Specifically, it uses warm water for uniform CPU cooling, while equipping each CPU with a TEC to provide additional localized cooling for hotspots. However, this approach faces significant challenges in meeting the unique requirements of edge datacenters, i.e., high density and heterogeneity. First, it necessitates substantial modifications to server internals. In addition to installing a TEC for each CPU, the solution requires attaching a copper plate twice the size of the CPU and an extra cold plate to maintain thermal conductivity when the TEC is disabled, which is somewhat impractical for already space-constrained edge servers. Second, this approach does not readily support heterogeneous hardware due to the same installation constraints and the limited cooling capacity of TECs, which are insufficient to handle high-powered components such as GPUs, whose TDP can reach over 1 kW.

In our preliminary work [25], we propose a fine-grained warm water cooling architecture called CoolEdge, which utilizes proportional valves to provide customized cooling water for each component. The water temperature can be dynamically adjusted within the range defined by the inlet temperatures of the chilled and hot water. While CoolEdge achieves significant cooling energy savings, we identify that the use of proportional valves not only incurs relatively high capital costs but also introduces increased complexity in cooling management (which will be discussed in Section 3.2). For some excessively underutilized datacenters, the expected energy savings may not sufficiently offset the capital expenditures associated with these valves. Motivated by this practical limitation, this study aims to develop CoolEdge$^+$, a highly efficient yet cost-effective cooling solution with low cooling complexity and improved cooling reliability for edge datacenters. The key enhancements of CoolEdge$^+$ include: (1) Adopting on/off valves for component-level cooling control with a customized power capping mechanism: Instead of flexible but costly proportional valves, CoolEdge$^+$ leverages simpler and more economical on/off valves. Although this approach restricts the inlet water temperature to a discrete set of values, it still effectively mitigates over-cooling by employing a service level objective-aware (SLO-aware) power capping mechanism. (2) Integrating a customized vapor chamber-based cold plate: Rather than relying on off-the-shelf vapor chambers, which are less efficient, CoolEdge$^+$ incorporates a fully integrated, custom-designed vapor chamber-based cold plate, which improves heat dissipation efficiency and enables a more uniform temperature distribution within hardware components.

To evaluate the effectiveness of these enhancements, we also perform extensive experiments. Specifically, we evaluate CoolEdge$^+$ using a new production trace and compare its performance against CoolEdge and an additional baseline from prior work [15]; we evaluate and analyze the vapor chamber-based cold plate from three perspectives under various cooling conditions. Finally, we provide practical recommendations for datacenter operators, guiding them in choosing between the two solutions of CoolEdge and CoolEdge$^+$ based on their specific financial constraints and operational requirements.

3 System Design

In this section, we formally propose CoolEdge$^+$, a component-level warm water cooling system specifically designed for edge datacenters. We begin with an overview of the system architecture and then detail its key design components.

3.1 System Overview

As shown in Figure 4, every server is equipped with multiple heterogeneous hardware components, such as CPU and GPU. There are two key cooling loops in CoolEdge$^+$, including Inner-and-Outer Loop and Mini Loop that deal with inter-component and intra-component hotspots, respectively. The control system is responsible for implementing fine-grained cooling control through the Inner-and-Outer Loop.

1) Inner-and-Outer Loop. It involves two water circulations, i.e., Inner Loop and Outer Loop. Specifically, the Inner Loop refers to a hot water circulation that directly recycles the "used" water after it has absorbed heat from hardware components. In contrast, the Outer Loop is a cold water circulation that routes the heated water to a chiller, where it is cooled and refreshed. Differing from conventional water cooling systems, our design utilizes a valve to provide a customized inlet water temperature for each hardware component. This is achieved by selectively supplying hot water from the Inner Loop, cold water from the Outer Loop, or a mixture of both — resulting in appropriately tempered warm water.
2) Mini Loop. It refers to a small vapor-fluid circulation inside a two-phase vapor chamber, which is implemented on the cold plate to enhance thermal conductivity and mitigate local hotspots inside the component in an automated manner.
3) The control system. It comprises a global controller and multiple local controllers, i.e., one local controller

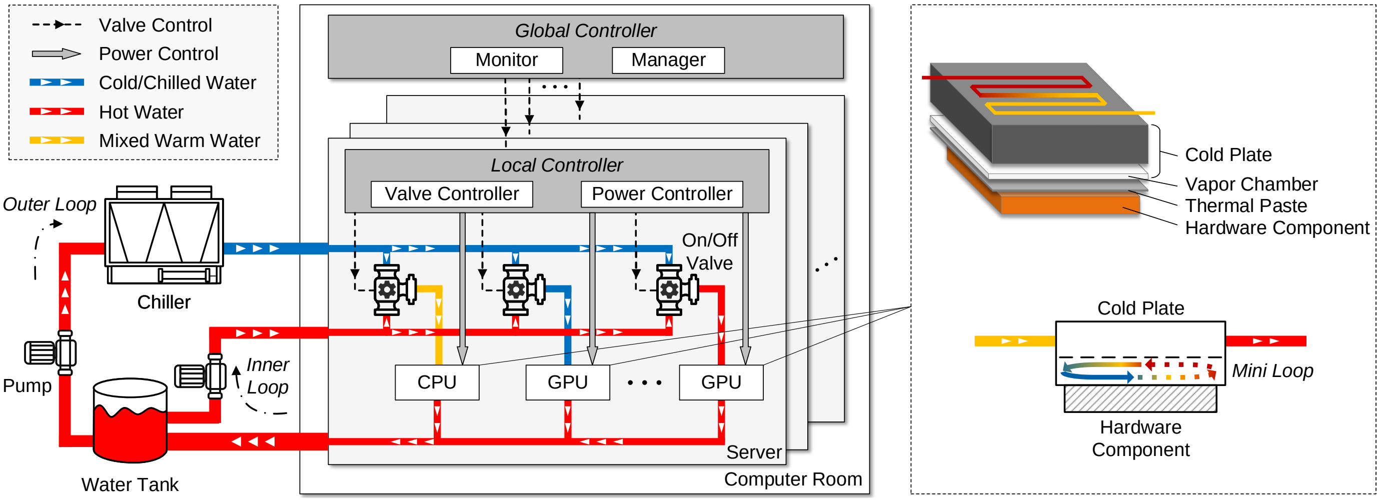


Figure 4: CoolEdge$^+$: Component-level water cooling system.

for each server. Specifically, based on the task information and the cooling control strategy (introduced in Section 4.3), the local controller determines the optimal cooling configuration, i.e., the inlet water temperature for each component. Subsequently, it sends control signals to each valve to set the water temperature accordingly.

3.2 Inner-and-Outer Loop: Inter-Component Hotspot Elimination with Customized Water

As illustrated in Section 2, both homogeneous and heterogeneous components have different cooling demands over time. To deal with hotspots among different components, we design two water circulations including the Inner Loop and the Outer Loop which achieve fine-grained component-level cooling control in an edge datacenter. As plotted in Figure 4, the Inner Loop gathers the “used” water from the outlet of each component to the water tank and pumps it to the inlet again. Since the hot water from the Inner Loop cannot cool down some high-utilization components, the Outer Loop pumps hot water from the water tank to the chiller and then sends the chilled water to the inlet. At the inlet of every component, there are valves that regulates the water temperature at a suitable value based on the component’s power demand. As compared to merely using the cold water directly, the choice between using cold water, hot water, or their mix helps reduce the required amount of chilled water and thus save cooling energy.

In our preliminary work [25], CoolEdge uses proportional valves to provide customized cooling water. Leveraging such valves, CoolEdge can provide any amounts of hot and cold water to regulate the inlet water temperature at any value within the range of the temperatures of the cold and hot water. However, those valves are somewhat costly (the purchase price is about $30 for each component [26]). Hence, we further design a cost-effective solution CoolEdge$^+$ here that replaces the proportional valves with economical on/off valves (about $14 for each component [27]) to save capital expenditures significantly. As on/off valves either allow unimpeded flow or stop flow completely, only three discrete water temperature values can be regulated, by allowing hot water only, cold water only, or the mix of both the hot and cold water that generates warm water. It is worth noting that directly using the on/off valves could reduce the cooling efficiency improvement largely since there are only three water temperature values available. To maintain high cooling efficiency, we devise a dynamic cooling control mechanism with a power capping approach which balances the cooling demand and computing performance by adjusting the maximum allowed hardware power, which will be detailed in Section 4.

3.3 Mini Loop: Intra-Component Hotspot Dispersion with Two-Phase Vapor Chambers

To mitigate intra-component hotspots, we incorporate a two-phase vapor chamber into the cold plate and implement a vapor–liquid mini loop within the chamber. As illustrated in Figure 4, the cold plate is attached directly to the hardware component to transfer heat into the circulating cooling water. A layer of thermal paste is applied between the hardware and the cold plate to eliminate air gaps and enhance thermal conductivity. It is worth noting that vapor chambers are typically used as standalone elements placed between a heat source and a cooling device to conduct heat directly. However, we observe that this conventional usage is inefficient when transferring heat from a hardware component to the cooling water in a full cold plate assembly, due to the extended thermal path and the additional layer of thermal paste. To address this, instead of attaching the vapor chamber to the bottom of the cold plate, we replace the cold plate’s baseplate entirely with the vapor chamber. This structural integration significantly enhances thermal conductivity.

In our preliminary work [25], CoolEdge employs off-the-shelf vapor chambers directly. In this study, we particularly focus on server-grade hardware and develop a customized, fully integrated vapor chamber-based cold plate with an internal fin structure to improve the heat conduction performance largely, as shown later in Figure 12. We provide a detailed description of its physical structure, working principle, and attractive characteristics in Appendix A of the Supplementary File. Further, in Section 5.4, we

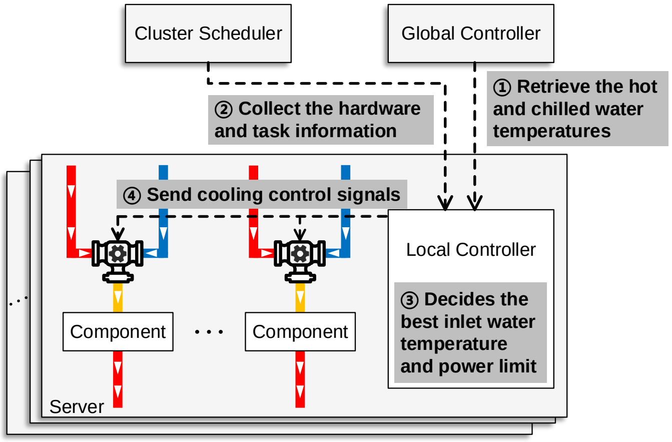

Cluster Scheduler
Global Controller
① Retrieve the hot and chilled water temperatures
② Collect the hardware and task information
④ Send cooling control signals
Local Controller
③ Decides the best inlet water temperature and power limit
Component
...
Component
Server

Figure 5: The controlling procedure of the control system.

conduct extensive experiments to demonstrate its superior performance.

3.4 Control System

In the control system, a global controller controls the overall thermal management, while each server is managed by an individual local controller. The global controller is responsible for periodically monitoring the temperatures of the hot water from the water tank and the chilled water from the chiller and distributing this information to each local controller via the *Monitor* module. Also, it adjusts the chilled water temperature set-point periodically based on the overall cooling demand through the *Manager* module. Each local controller, operating at the server level, determines the power limits for individual components and enforces these limits via the *Power Controller* module. It also regulates each valve accordingly through the *Valve Controller* module. In the event of an unexpected overheating incident, the *Power Controller* module is further responsible for continuously monitoring the temperatures of all components to ensure system reliability.

Figure 5 describes a typical controlling procedure. First, each local controller ① retrieves the hot and chilled water temperatures from the global controller. Next, upon a task scheduling decision made by the cluster scheduler (e.g., Kubernetes), the local controller ② collects the hardware and task information (e.g., hardware type, task type and latency constraint). Based on this collected information and the cooling control strategy (introduced in Section 4.3), the local controller ③ decides the best inlet water temperature and power limit for the component. Finally, the local controller ④ sends control signals to the corresponding valve installed on the server to implement the cooling control. At runtime, the local controller also monitors the real-time temperatures of all the CPUs and GPUs with the `lm_sensors` and `nvidia-smi` tools, respectively. As the cooling control decision is made by the local controller on each server, and every server operates its own local controller independently, even in large-scale deployments with hundreds of servers, the control loop latency remains stable. Such a distributed architecture demonstrates strong scalability and minimal centralized overhead.

To ensure hardware safety and system robustness in the event of valve or cooling loop failures, such as a valve becoming stuck in an open or closed position, or a circulation failure in the Inner or Outer Loop, CoolEdge$^+$ employs a *tiered mitigation strategy* based on the mismatch between actual cooling capacity and the real-time cooling demand of the executing workload. First, each local controller continuously monitors key thermal indicators including hardware temperatures and inlet water temperatures. If a valve fails but the resulting cooling capacity is still sufficient to maintain the component within its safe thermal operating range (i.e., within the T_{safe} threshold), the controller triggers a non-blocking warning to the datacenter operator via the global controller. The workload continues executing uninterrupted, and the faulty valve is scheduled for graceful maintenance or replacement. However, if the controller determines that the cooling capacity is insufficient, e.g., when the hardware temperature shows a rapid or sustained rise beyond the acceptable thermal margin, the system initiates a two-phase emergency response:

1) Immediate workload migration: The local controller promptly notifies the cluster scheduler to migrate the affected workload to an idle and better-cooled hardware component, thereby preventing thermal violations or unexpected performance degradation.
2) Fault isolation and escalation: The faulty valve or loop segment is logically marked as “unavailable” to avoid further task assignment, and an urgent alert is dispatched to the datacenter operator for inspection and repair.

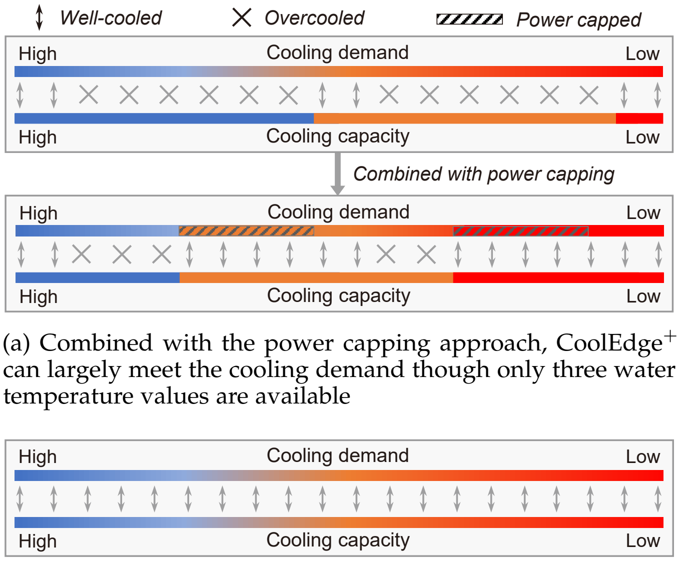
In more severe scenarios, such as pump failures in the Inner or Outer Loop, or chiller malfunction in the Outer Loop, the global controller can fall back to a conservative cold/hot-water-only configuration. In parallel, it applies SLO-aware power capping to all non-idle components to prevent thermal overload, albeit at the cost of possible performance degradation. Together, these fault-tolerant measures ensure hardware safety, system availability, and service continuity, even under partial cooling system degradation. This makes CoolEdge$^+$ a practical and resilient solution for edge deployments, where manual intervention may not always be timely or feasible.

4 Component-Level Cooling Control Framework

In this section, we first theoretically quantify the power saving achieved by the warm water cooling. Then, we introduce the cooling control mechanism of CoolEdge$^+$ and its main difference from that of CoolEdge. Finally, we present the details of the cooling control strategy.

4.1 The Natural Heat Dissipation under Warm Water Cooling

It is widely recognized that increasing the cooling water temperature has significant potential for reducing cooling energy consumption, as confirmed by numerous studies [13], [28]. When warm water is used, its temperature often exceeds that of the surrounding air, leading to a significant phenomenon of *natural heat dissipation* in pipes and water



(a) Combined with the power capping approach, CoolEdge$^+$ can largely meet the cooling demand though only three water temperature values are available

(b) CoolEdge always meets the cooling demand but incurs high cooling complexity

Figure 6: The cooling control mechanism of CoolEdge$^+$ vs. CoolEdge [25].

tanks. This passive heat loss can reduce the load on the chiller, resulting in lower cooling energy consumption compared with conventional cold water cooling. This section presents a quantitative analysis of warm water cooling, with a particular focus on the role of natural heat dissipation. According to the theoretical derivations[2], the efficiency of the natural heat dissipation depends on (1) ΔT: the temperature difference between the cooling water inside the pipe and the ambient air, (2) v: the water flow rate, (3) h: the convective heat transfer coefficient of the air, and (4) ξ and μ: parameters associated with the physical properties of the pipe and the water, respectively. Based on Fourier's law of heat conduction and Newton's law of cooling, the amount of heat dissipated through the pipe, denoted as P (in Watts), can be expressed as:

$$P = \xi v \Delta T \big(1 - \exp(-\mu h / v)\big). \tag{1}$$

Equation (1) analyzes the key factors affecting the amount of dissipated heat to the ambient, and thus the efficiency of water cooling. Our analysis reveals that, from the perspective of natural heat dissipation, increasing the water temperature contributes significantly to the heat dissipation and has a remarkable impact on improving cooling efficiency.

4.2 Component-Level Cooling Control Mechanism

To take full advantage of the natural heat dissipation phenomenon, it is beneficial to increase the water temperature while ensuring the hardware safety. However, in view of the unavoidable hotspots, it is crucial to customize the cooling water temperature for each hardware component based on its cooling demand, as discussed in Section 3.2. Rather than using costly proportional valves, CoolEdge$^+$ leverages simpler and more ecnomical on/off valves to implement component-level, fine-grained cooling control. Instead of customizing arbitrary cooling water temperatures as CoolEdge does, CoolEdge$^+$ can regulate three discrete water temperature values only, by allowing hot water only, cold water only, and the mix of both the hot and cold water in a fixed ratio. In the case that the number of allowed inlet water temperature values is restricted to three, to avoid potential efficiency drop because of over-cooling, we integrate an SLO-aware power capping approach into the design. By allowing limited performance degradation (e.g., 5%) through power capping, the cooling demand can be reduced slightly to match the cooling capacity provided by the cooling water under one of the three possible temperatures. Figure 6 summarizes the difference between CoolEdge$^+$ and CoolEdge in the cooling control mechanism. As shown in Figure 6a, leveraging the well-managed power capping approach, CoolEdge$^+$ can avoid over-cooling significantly and achieve comparable cooling efficiency as CoolEdge.

Algorithm 1 Fine-grained cooling control algorithm of CoolEdge$^+$

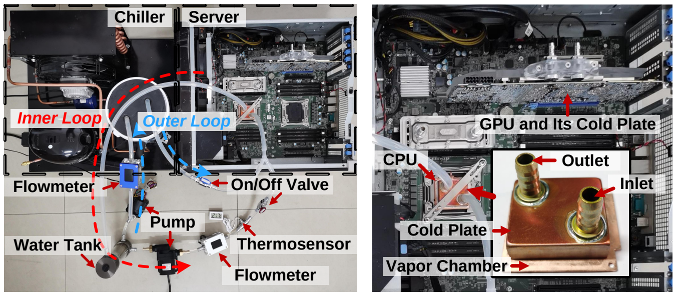
```
Initialize: the list $R$ recording all the running tasks, the temperature of the chiller water from the chiller $T_{cold}$, the temperature of the hot water directly from the water tank $T_{hot}$, the ratio of the flow rates of the hot water to the cold water $\alpha$, the power model $P = M_P(T_{water}, T_{safe}, j)$, and the latency model $L = M_L(P, i, j)$.
while a request $r$ of the $i$-th task type arrives, with the latency constraint of $L_{SLO}$ and demanding the $j$-th hardware type do
  Record $r$ in $R$;
  Update $T_{hot}$ according to the temperature reading;
  for $T_{water} = T_{hot}, \frac{\alpha T_{hot} + T_{cold}}{\alpha + 1}, T_{cold}$ do
    Estimate $P = M_P(T_{water}, T_{safe}, j)$;
    Estimate $L = M_L(P, i, j)$;
    if $L \leq L_{SLO}$ then
      break;
    end if
  end for
  Set the hardware power limit based on $P$, tune the valves and pumps based on $T_{water}$, and dispatch the request;
end while
for Every time period of length $C$ do
  Update $T_{hot}$ according to the temperature reading;
  for $r$ in $R$ do
    for $T_{water} = T_{hot}, \frac{\alpha T_{hot} + T_{cold}}{\alpha + 1}, T_{cold}$ do
      Estimate $P = M_P(T_{water}, T_{safe}, j)$;
      Estimate $L = M_L(P, i, j)$;
      if $L \leq L_{SLO}$ then
        break;
      end if
    end for
    Tune the hardware power limit based on $P$, and tune the valves and pumps based on all $T_{water}$;
  end for
end for
```

[2] The theoretical derivations are provided in Appendix B of the Supplementary File.

4.3 Fine-Grained Cooling Control Strategy

According to the above control mechanism, Algorithm 1 presents the control details of CoolEdge$^+$. In the offline phase, based on the thermal profiles collected from the hardware prototype shown later in Figure 7, we build a power model $P = M_P(j, T_{water}, T_{safe})$ of the j-th hardware type to estimate the maximum allowable power consumption P under its safe operating temperature T_{safe} while being cooled by water at temperature T_{water}. Specifically, for the hardware of the j-th type, we first let it run at each sampled power value P and measure its temperature $T_{hardware}$ under different cooling water temperatures T_{water}, including 30°C, 40°C, and 50°C. Then, we apply the linear regression method to describe the relationship among hardware power P, water temperature T_{water}, and hardware temperature $T_{hardware}$, and obtain the power model M_P of the j-th hardware type. Note that in this fine-grained cooling system with on/off valves only, all components will share the same water flow rate, so we do not consider the flow rate in the power model and set it at a fixed value. We define the ratio of the flow rates of the hot water to the cold water as α, a hyperparameter that influences the warm water temperature when mixing the hot and cold water. We also build a latency model $L = M_L(P, i, j)$ to obtain the processing latency of the i-th task type (e.g., machine learning inference) running on the j-th hardware type under the power limit of P. Specifically, on the hardware of the j-th type, we set the hardware power limit at each sampled value P and measure the processing latency L of each task type i. As the number of candidate P values is limited, we directly store the measured data and obtain the latency model M_L on the j-th hardware type.

In the online phase, for each incoming request of the i-th task type with the latency constraint of L_{SLO} and demanding the j-th hardware type (Line 2), the local controller first records its metadata (e.g., the task type i, the hardware type j, and the processing latency under no additional limit on the hardware power) and updates the temperature of the hot water in the water tank T_{hot} since it will change over time (Lines 3-4). Then, for each of the three water temperature values in descending order, the controller will estimate the maximum allowed hardware power P and the processing latency of the i-th task type running on the j-th hardware type under the power limit of P based on the power and latency models (Lines 5-7). Once the processing latency is within L_{SLO}, the controller will implement the cooling control and set the hardware power limit accordingly, and the request will be scheduled to that component (Lines 8-12). Finally, to avoid cooling failures when the hot water temperature T_{hot} rises too high as time goes by, every C time period the controllers will perform a global adjustment to all valves by repeating the above cooling steps (Lines 14-26). The above algorithm is very lightweight: for each hardware component, only three candidate cooling water temperatures need to be evaluated, resulting in a time complexity of $O(1)$ (i.e., Lines 5-11); during global adjustment, if there are n hardware components currently processing requests, the overall time complexity becomes $O(n)$ (i.e., Lines 16-25). Notably, when running on a core of the Intel Xeon E5-2697 v4 CPU, it takes only about $0.2\,\mathrm{ms}$ for CoolEdge$^+$ to make a cooling decision (i.e., Lines 5-11).



(a) The overall architecture (b) Inside the server

Figure 7: Hardware prototype for CoolEdge$^+$.

5 Evaluation

In this section, we first show the hardware prototype and present the evaluation setup. Then, we perform extensive simulations to evaluate the performance of CoolEdge$^+$ and estimate its cost savings as compared with other cooling systems including CoolEdge. Finally, we present our further experiments on the advanced vapor chamber-based cold plates.

5.1 Hardware Prototype

To verify the practicability of CoolEdge$^+$ and collect thermal profiles of hardware components, we develop a hardware prototype based on a Dell Precision Tower 7910 Workstation, as illustrated in Figure 7a. The cooling system consists of two water circulations: the Inner Loop and the Outer Loop. The Inner Loop includes a water tank, a pump, a flowmeter for monitoring the water flow rate, and a thermosensor for measuring the temperature of the inlet warm water from the water tank. The Outer Loop comprises a pump, a flowmeter, and a chiller for cooling the water. By controlling the inlet on/off valves, customized cooling water is ultimately delivered to each hardware component. Figure 7b shows the items in the server, including an Intel Xeon E5-2680 v4 CPU and an Nvidia GeForce RTX 2080 Ti GPU. For illustration clarity, we do not connect water pipes to all components and use the CPU as a representative example.

5.2 Evaluation Setup

Simulation methodology: To simulate an edge datacenter, we incorporate the essential physical infrastructure, including water pipe lengths as well as the shared use of the chiller and pumps. We also consider the fans to maintain the ambient temperature and improve the natural heat dissipation by increasing h. Considering that their energy consumption is much lower than the water cooling equipment, we compute the total energy consumption E_{total} by the summation of the energy consumption of the centralized chiller $E_{chiller}$ and the two pumps E_{pump}. Note that the calculation of $E_{chiller}$ takes into account the natural heat dissipation phenomenon, as described by Equation (1). The thermal profiles of the hardware components are collected using our hardware prototype.

Workload trace and workloads: We use the workload trace from Alibaba PAI [29] to evaluate CoolEdge$^+$. The Alibaba PAI trace contains high-level information of machine

Table 2: Parameter settings

Parameter	T_{safe}	h	Coefficient of Performance of the chiller	Ambient temperature
Value	70% of MOT	10 W/m^2°C	3.6	35°C

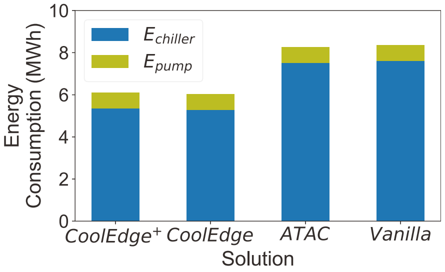


Figure 8: Total cooling energy consumption.

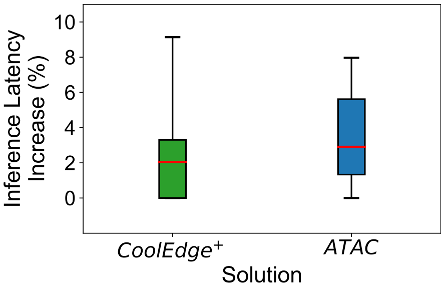


Figure 9: The inference latency increase.

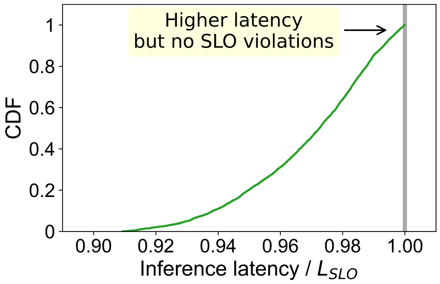


Figure 10: The CDF of SLO satisfaction when applying CoolEdge$^+$.

learning (ML) workloads during two months in a cluster with 6,500 GPUs, such as the task name, start time, and end time. We select the first seven days of the trace in the simulation. As the trace does not include the task type information, We divide the tasks into ten groups manually according to the remainder of its job name divided by 10, and assume that all the tasks in the same group are ML inference tasks on the same ML model, including ResNetV2-101, Inception, VGG16, EfficientNet-B3, EfficientNet-B5, EfficientNet-B7, YOLOv3, UNet, Pix2Pix, and XLNet. To build the power and latency models for each ML model type (i.e., task type) as mentioned in Section 4.3, we measure the average power consumption and inference latency of these models on one GPU in our hardware prototype under different power limits. The latency SLO is randomly set to 1.01∼1.10× the inference latency under no additional limit on the hardware power.

Baselines: Since air cooling has a hard time meeting the cooling demands of edge servers, we consider three water cooling baseline strategies as follows:

- Conventional coarse-grained water cooling system (*Vanilla*): For this baseline [12], we set the global water temperature and flow rate according to the highest cooling demand of all the hardware components.
- State-of-the-art fine-grained water cooling system (*CoolEdge*): As mentioned in Section 3.2, CoolEdge [25] leverages the proportional valves to mix a certain amount of hot and cold water and regulate the water temperature at the desired value as per each component's cooling demand.
- Coarse-grained water cooling system with SLO-aware power capping (*ATAC*): ATAC [15] proposes a dynamic power capping solution to reduce cooling energy consumption by turning down power usage of hotspot components in air-cooled datacenters. We apply this power capping solution to the water cooling system and control the number of tasks with a latency increase of more than 5% to be between 20% and 40% after capping the power. After that, we set the global cooling configuration according to the highest cooling demand of all the components.

Parameter settings: Table 2 lists all the parameter settings. As prolonged operation at near MOT may degrade performance and shorten hardware lifespan, we set the safe operating temperature as 70% of the MOT. Note that the ambient temperature is set to 35°C since it is stated that the Alibaba PAI trace was collected in summer days [29].

5.3 Evaluation Results

We analyze the simulation results of CoolEdge$^+$, CoolEdge, *ATAC*, and *Vanilla* from several aspects as follows.

Energy savings: As shown in Figure 8, CoolEdge$^+$ and CoolEdge reduce the cooling energy consumption by 27.19% and 28.05%, respectively, as compared with *Vanilla*, and *ATAC* lowers the cooling energy slightly by 1.30% than *Vanilla* at the expense of hardware performance. As we can see, although CoolEdge$^+$ can choose between only three inlet water temperature values for each hardware component, its energy consumption is very close to CoolEdge thanks to the well-managed power capping approach that helps improve the match between cooling demand and cooling supply significantly. However, by comparing *ATAC* and *Vanilla*, we can see that using the power capping approach alone shows little energy efficiency improvement when serving latency-critical, SLO-specified workloads.

Computing performance: Figure 9 plots the inference latency increase of all tasks as compared to the inference latency under no additional limit on the hardware power, and Figure 10 plots the CDF of the inference latency to its SLO constraint (L_{SLO}) when applying CoolEdge$^+$. Although CoolEdge$^+$ increases the inference latency by a ratio of 1∼1.09 and by 1.02 on average, the latency is still within the

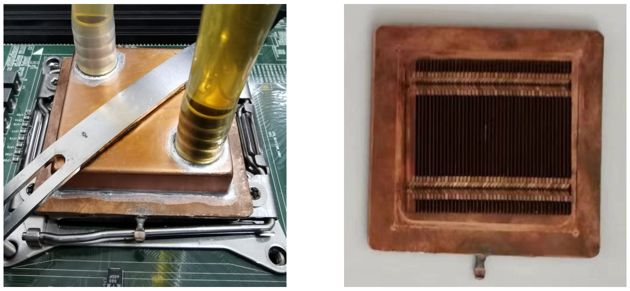
Table 3: Cost Saving Calculation (Unit: \$/(server×year))

Description	*ATAC*	*CoolEdge*		*CoolEdge*$^+$	
ExCapEx	0	Proportional valve	12.00	On/off valve	5.79
ChiSav	0.68	17.33		17.32	
EnerSav	0.53	11.30		10.96	
CoSav	1.21	16.63		22.49	

SLO constraint. By comparison, *ATAC* increases the latency by 1.03 on average based on the cooling setup presented in Section 5.2. It is worth noting that CoolEdge$^+$ provides the ability to balance the computing performance and cooling energy efficiency by setting different SLOs, as illustrated by the bar filled with diagonal stripes in Figure 6.

Cost savings: Here, we estimate the cost savings from CoolEdge$^+$, CoolEdge, and *ATAC* as compared with *Vanilla*. We consider extra capital expenditures (ExCapEx), capital expenditure savings of the chiller (ChiSav), and cooling energy savings (EnerSav) in the analysis. Specifically, ExCapEx mainly depends on the valves and can be calculated according to their purchase prices and lifespans [26], [27]; ChiSav can be calculated based on the demand on the cooling capacity of the chiller [30]; EnerSav can be calculated from Figure 8. Ultimately, Cost Savings (CoSav) can be calculated by ChiSav + EnerSav − ExCapEx. All the calculation results are summarized in Table 3. As we can see, CoolEdge$^+$ further improves the cost savings by as high as 35.24% than CoolEdge. For 2,000 small-scale edge datacenters (each equipped with 80 servers) in a city, the cost savings brought by CoolEdge$^+$ can reach \$3,598,400/year.

Comparison between CoolEdge$^+$ and CoolEdge: Here, we provide practical guidelines for choosing between CoolEdge$^+$ and the preliminary work CoolEdge [25] for real-world deployment. According to the aforementioned results, we can see that CoolEdge$^+$ achieves comparable energy savings as CoolEdge while reducing the CapEx of valves by over half, thus increasing the cost savings by nearly one million dollars every year for a city. However, the main concern of CoolEdge$^+$ is the slightly degraded computing performance. Specifically, CoolEdge$^+$ is able to satisfy all SLO constraints on the condition that the computing performance is allowed to decline marginally. Otherwise, CoolEdge$^+$ may behave fairly worse than CoolEdge in avoiding over-cooling and improving cooling efficiency, as depicted in Figure 6. Therefore, CoolEdge$^+$ is not suitable for edge datacenters with extreme performance requirements (e.g., the processing latency should be reduced as much as possible), where the power capping approach needed by CoolEdge$^+$ could be no longer feasible. On the other hand, since CoolEdge$^+$ maintains similar cooling efficiency but cuts down cooling complexity and costs significantly, CoolEdge$^+$ could be a better choice as long as slight performance degradation is allowable (e.g., 2% on average), even if it is allowed just sometimes (e.g., when the datacenter is lightly loaded). In conclusion, CoolEdge$^+$ and CoolEdge differ in the cooling mechanism, and the selection between them is highly dependent on the requirements of the workloads supported by the edge datacenters.



(a) The appearance (b) The fin structure

Figure 11: Our newly developed cold plate.

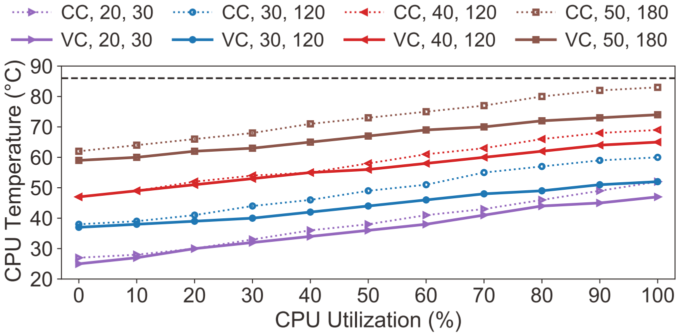


Figure 12: The CPU temperature under different cold plates, CPU utilization, water temperature, and flow rate.

Supporting AI clusters: CoolEdge$^+$ offers a practical and promising solution for AI clusters, particularly for AI inference workloads for the following reasons. (1) Component-level cooling tailored for heterogeneous hardware: AI clusters typically use a mix of CPUs, GPUs, and memory/storage components with diverse thermal characteristics. CoolEdge$^+$ provides both inter- and intra-component thermal control, effectively addressing the thermal imbalance problem among heterogeneous hardware components. (2) Discrete temperature control with SLO-aware power capping: AI inference workloads are typically latency-sensitive, requiring careful thermal control without violating performance constraints. CoolEdge$^+$ allows three discrete cooling water temperatures and introduces an SLO-aware power capping approach that dynamically adjusts hardware performance to maintain inference latency within acceptable bounds. This trade-off between thermal efficiency and performance preservation is crucial for AI clusters.

Performance impacts under heavy and bursty workloads: CoolEdge$^+$ is designed to enhance cooling efficiency through fine-grained, component-level control and SLO-aware power capping. While this approach could slightly affect hardware performance, it offers a favorable trade-off by reducing cooling complexity and energy consumption significantly. Here, we outline potential performance considerations under heavy and bursty workloads as well as future directions for improvement. First, by deliberately capping hardware power to align with limited cooling capacity, especially when only three discrete inlet water temperatures are available, the execution time of latency-sensitive tasks may slightly increase. Although almost all workloads remain

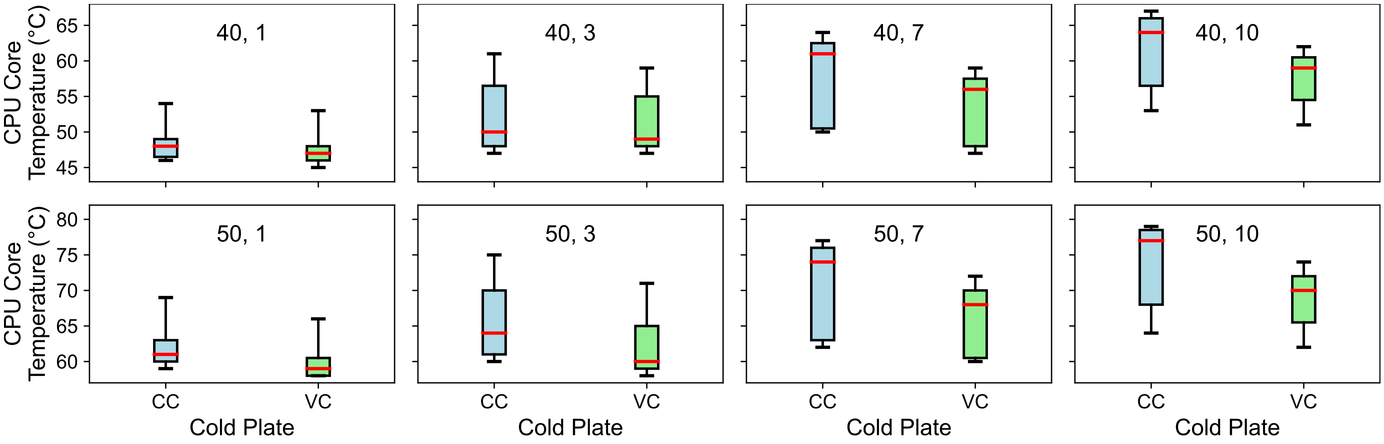
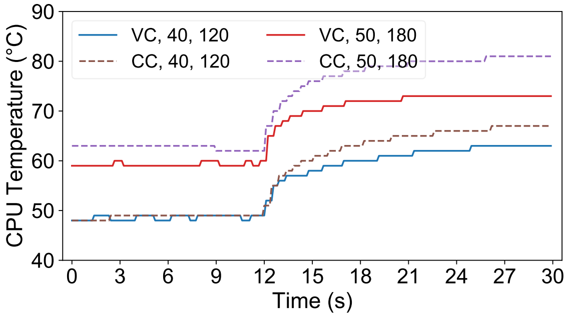
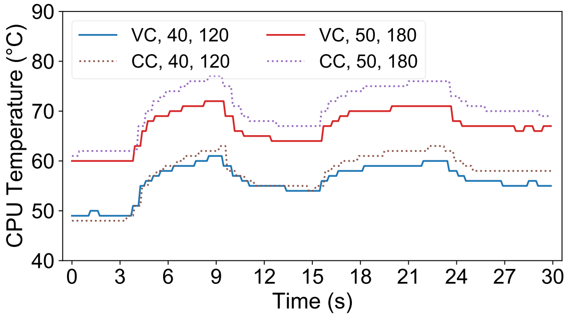


Figure 13: The CPU core temperature distribution under various cooling conditions and utilization patterns. The first number in the title of each subfigure denotes the water temperature (°C), and the second number denotes the number of tested cores. The tested cores are kept at 100% utilization, and the rest remain 0%.



(a) At time = 11.6 s, the CPU utilization grows from 0% to 100%



(b) The CPU utilization grows from 0% to 100% at time = 3.6 s, to 20% at time = 9.3 s, to 80% at time = 15.0 s, and finally, to 40% at time = 23.2 s

Figure 14: CPU temperature variation as its utilization varies.

within their SLO constraints, the cumulative effect may reduce the peak supported throughput of the system and its ability to absorb short-term workload surges, thus degrading overall responsiveness during traffic bursts. Second, under dynamic and unpredictable load conditions, hardware components may experience rapid and non-uniform increases in thermal output. Due to the discrete nature of on/off valve control and the absence of continuous inlet temperature modulation, CoolEdge$^+$ may not always react with sufficient granularity to match instantaneous cooling demand. In such cases, thermal safeguards (e.g., DVFS-based frequency throttling) may be involuntarily triggered to prevent overheating, leading to temporary but unanticipated performance degradation. To deal with these implications, future work could explore predictive and adaptive control strategies, such as ML-based workload forecasting or thermal trajectory modeling. These approaches could further enhance the system's ability to anticipate upcoming load spikes and proactively adjust power caps or cooling configurations, thereby improving reactivity and robustness under highly dynamic workload patterns.

5.4 Experiments on Advanced Vapor Chamber-Based Cold Plates

In this work, we develop a customized, fully integrated vapor chamber-based cold plate with an internal fin structure, as shown in Figure 11. We perform several experiments to compare the newly developed vapor chamber-based cold plate (VC) with the commonly used cold plate (CC), both of which include the internal fin structure. The results demonstrate the following three characteristics that are promising to edge datacenters.

Reducing the overall hardware temperature: Figure 12 shows the overall CPU temperature, where the horizontal line indicates MOT (i.e., 86°C), and *CC, 20, 30* refers to using the commonly used cold plate under the inlet water temperature of 20°C and flow rate of $30\,\mathrm{L/h}$. We can see that VC outperforms CC, especially when the CPU utilization and inlet water temperature get high. For example, when the CPU utilization and inlet water temperature are 100% and 50°C, respectively, the CPU temperature difference reaches 9°C. This characteristic helps narrow the temperature difference between hotspot components and others, especially for high-powered hardware components and in the scope of warm water cooling, saving the cooling energy for dispersing hotspots. As the TDP of modern server components continues to grow (e.g., $1.4\,\mathrm{kW}$ of the latest Nvidia B300 GPU), we think that VC could make a valuable and lasting contribution to datacenter cooling.

Smoothing the temperature distribution spatially: Figure 13 plots the core temperature distribution under various cooling conditions and utilization patterns. Across these eight settings, VC reduces the median and the maximum core temperature by 1°C∼7°C and 1°C∼5°C, respectively, as compared with CC. Also, the standard deviation drops from 2.38°C∼5.99°C to 2.08°C∼4.73°C after using VC. This characteristic brings two benefits. (1) Hardware safety: VC helps reduce the probability of local overheating inside a component automatically, especially when the component is partially utilized. This improves hardware safety and lifespans since there cannot be thermosensors everywhere inside the component to monitor local temperatures. (2) Cooling energy usage: VC helps reduce the maximum core temperature that usually determines the cooling demand. The cooling energy for dealing with hotspots can be reduced, especially in the existing coarse-grained cooling system.

Smoothing the temperature variation temporally: Figure 14 plots the CPU temperature variation as the utilization varies. As we can see, the CPU temperature varies more smoothly when using VC rather than CC. For example, as shown in Figure 14a, when the water temperature is 50°C, it takes 1.0 s and 2.9 s for the CPU temperature to reach 70°C with CC and VC, respectively. This characteristic helps slow down the instantaneous hardware temperature rise in the face of the cooling lag and thus improves hardware safety, especially for high-powered hardware components running edge workloads with high utilization variation [7].

6 Related Work

Power and thermal management in datacenters. With the rapid growth of cloud and edge computing, the power consumption of datacenters is rising sharply, especially under the pressure of AI workloads [31]. Extensive research has been conducted on power and thermal management in datacenters, such as workload scheduling [17], [18], [23], [32], [33], [34], hotspot mitigation [13], [15], [35], and oversubscription [10], [36], [37]. Stojkovic et al. [17] systematically analyze GPU power and temperature behavior in datacenters running LLM inference tasks. Based on the findings, they propose a thermal- and power-aware inference workload scheduling framework that reduces the peak GPU temperature and power consumption through three strategies: virtual machine placement, request routing, and instance configuration. Patel et al. [37] systematically analyze the power consumption characteristics of LLMs during training and inference. They propose a GPU power management framework that reduces peak power demands through frequency locking while meeting quality-of-service requirements. This approach enables power over-subscription in datacenters and improves server deployment density. Differing from them, this work focuses on improving cooling efficiency without requiring the redistribution of workloads or impacting workload performance, through component-level cooling control designed specifically for high-density, heterogeneous edge datacenters.

Increasing coolant temperature. Recent literalture suggests to increase the coolant temperature to reduce cooling energy consumption while maintaining hardware safety and reliability [13], [28], [38], [39], [40]. El-Sayed et al. [38] point out that increasing the datacenter's temperature set-point by 1°C can lead to 2%∼5% energy savings. They conduct an in-depth analysis of the impact of temperature on hard reliability, as well as the changes in server performance and power consumption. They further propose practical temperature management guidelines aimed at achieving energy efficiency while ensuring system reliability and performance. Higher coolant temperatures also expand the feasibility of utilizing the free cooling technique, thereby further reducing cooling energy. To mitigate the potential reliability risks caused by unstable outdoor temperatures, Goiri et al. [23] propose a prediction-based method that manages datacenter temperature, relative humidity, and temperature fluctuations by dynamically adjusting workload distribution, server states, and cooling modes. To balance cooling energy savings and hardware thermal requirements under the water cooling technique, Jiang et al. [13] propose a fine-grained warm water cooling architecture. The method supplies warm water for global cooling, while addressing local hotspots using additional cooling capacity provided by TECs.

Exploiting renewable energy in edge datacenters. Compared to centralized cloud datacenters, small-scale and geo-distributed edge datacenters offer greater flexibility in harnessing local renewable energy sources, such as wind and solar power [41], [42], [43], [44]. To maximize the use of renewable energy for edge services, Gu et al. [42] propose a deep reinforcement learning–based service management strategy that dynamically schedules tasks and decides energy provisioning among edge servers. In contrast to energy management at the datacenter level, Souza et al. [45] propose an application-oriented energy management mechanism that allows upper-layer applications to directly access information such as grid carbon intensity and local renewable energy availability, and to adapt their energy usage based on computational demands at a fine granularity, thereby improving carbon efficiency. While prior efforts manage workloads at the datacenter level to better utilize renewable energy, our proposed component-level cooling architecture not only reduces the overall datacenter energy demand, but also enables fine-grained adjustment of coolant temperature for each component, which further helps align datacenter energy demand with the amount of renewable energy available to the edge datacenter.

7 Concluding Remarks

In this study, we propose CoolEdge^{+}, a cost-effective component-level water cooling system tailored for edge datacenters. To mitigate inter-component hotspots, CoolEdge^{+} combines fine-grained cooling control and hardware power management that significantly reduce cooling energy consumption and capital expenditures with only an average latency increase of 2%, while consistently satisfying the latency SLO. To mitigate intra-component hotspots, CoolEdge^{+} integrates our well-developed vapor chamber-based cold plates that provide superior heat conduction capabilities. Based on a hardware prototype, the simulation results indicate that CoolEdge^{+} delivers cooling efficiency improvements comparable to the original CoolEdge, while achieving up to 35.24% additional cost savings, making the solution highly practical, scalable, and well-suited for widespread edge deployment. For a city with 2,000 small-scale edge datacenters, our estimates suggest that CoolEdge^{+} can yield $3,598,400 in annual cost savings.

References

[1] Fortune Business Insights, "Edge computing market size, share & industry analysis," 2025, https://www.fortunebusinessinsights.com/edge-computing-market-103760.

[2] LF Edge, "State of the edge 2020," 2020, https://www.lfedge.org/wp-content/uploads/2020/04/SOTE2020.pdf.

[3] ——, "State of the edge 2021," 2021, https://www.lfedge.org/2021/03/12/state-of-the-edge-2021-report/.

[4] E. Çam, Z. Hungerford, N. Schoch, F. P. Miranda, and C. D. Y. de León, "Electricity 2024, analysis and forecast to 2026," 2024, https://www.iea.org/reports/electricity-2024.

[5] B. Karagounis, "Introducing the microsoft azure modular datacenter," 2020, https://azure.microsoft.com/en-us/blog/introducing-the-microsoft-azure-modular-datacenter/.

[6] Z. Yu, "Tencent cloud opens 5g edge computing center," 2020, https://www.eyeshenzhen.com/content/2020-10/21/content_23648410.htm.

[7] M. Xu, Z. Fu, X. Ma, L. Zhang, Y. Li, F. Qian, S. Wang, K. Li, J. Yang, and X. Liu, "From cloud to edge: a first look at public edge platforms," in *Proceedings of the 21st ACM Internet Measurement Conference*, 2021.

[8] Vapor IO, "Intelligent data centers built for automation and performance at the edge," 2025, https://www.vapor.io/technology/modular-data-centers/.

[9] Open Compute Project, "Open Rack/SpecsAndDesigns," 2021, https://www.opencompute.org/wiki/Open_Rack/SpecsAndDesigns.

[10] I. Manousakis, Í. Goiri, S. Sankar, T. D. Nguyen, and R. Bianchini, "CoolProvision: Underprovisioning datacenter cooling," in *Proceedings of the 6th ACM Symposium on Cloud Computing*, 2015.

[11] Schneider Electric, "Liquid cooling technologies for data centers and edge applications," 2019, https://download.schneider-electric.com/files?p_Doc_Ref=SPD_VAVR-AQKM3N_EN.

[12] R. B. Roy, T. Patel, R. Kettimuthu, W. Allcock, P. Rich, A. Scovel, and D. Tiwari, "Operating liquid-cooled large-scale systems: Long-term monitoring, reliability analysis, and efficiency measures," in *2021 IEEE International Symposium on High-Performance Computer Architecture (HPCA)*. IEEE, 2021, pp. 881–893.

[13] W. Jiang, Z. Jia, S. Feng, F. Liu, and H. Jin, "Fine-grained warm water cooling for improving datacenter economy," in *Proceedings of the 46th Annual International Symposium on Computer Architecture*, 2019.

[14] S. Lee, D. Pandiyan, J.-s. Seo, P. E. Phelan, and C.-J. Wu, "Thermoelectric-based sustainable self-cooling for fine-grained processor hot spots," in *2016 15th IEEE Intersociety Conference on Thermal and Thermomechanical Phenomena in Electronic Systems (ITherm)*. IEEE, 2016, pp. 847–856.

[15] S. Yeo, M. M. Hossain, J.-C. Huang, and H.-H. S. Lee, "ATAC: Ambient temperature-aware capping for power efficient datacenters," in *Proceedings of the 5th ACM Symposium on Cloud Computing*, 2014, pp. 1–14.

[16] S. Liu, B. Leung, A. Neckar, S. O. Memik, G. Memik, and N. Hardavellas, "Hardware/software techniques for DRAM thermal management," in *Proceedings of the 17th International Symposium on High Performance Computer Architecture*, 2011.

[17] J. Stojkovic, C. Zhang, Í. Goiri, E. Choukse, H. Qiu, R. Fonseca, J. Torrellas, and R. Bianchini, "TAPAS: Thermal- and power-aware scheduling for LLM inference in cloud platforms," in *Proceedings of the 30th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 2*, 2025, pp. 1266–1281.

[18] Q. Pei, L. Wang, D. Zhang, B. Yan, C. Yu, and F. Liu, "InferCool: Enhancing AI inference cooling through transparent, non-intrusive task reassignment," in *Proceedings of the 2024 ACM Symposium on Cloud Computing*, 2024, pp. 487–504.

[19] W. Huang, S. Ghosh, S. Velusamy, K. Sankaranarayanan, K. Skadron, and M. R. Stan, "HotSpot: A compact thermal modeling methodology for early-stage VLSI design," *IEEE Transactions on very large scale integration systems*, vol. 14, no. 5, pp. 501–513, 2006.

[20] L. Ramos and R. Bianchini, "C-Oracle: Predictive thermal management for data centers," in *Proceedings of the 14th International Symposium on High Performance Computer Architecture*, 2008.

[21] A. Iranfar, M. Kamal, A. Afzali-Kusha, M. Pedram, and D. Atienza, "TheSPoT: Thermal stress-aware power and temperature management for multiprocessor systems-on-chip," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 37, no. 8, pp. 1532–1545, 2018.

[22] R. Ayoub, R. Nath, and T. Rosing, "JETC: Joint energy thermal and cooling management for memory and CPU subsystems in servers," in *Proceedings of the 18th International Symposium on High Performance Computer Architecture*, 2012.

[23] Í. Goiri, T. D. Nguyen, and R. Bianchini, "CoolAir: Temperature- and variation-aware management for freecooled datacenters," in *Proceedings of the 20th International Conference on Architectural Support for Programming Languages and Operating Systems*, 2015.

[24] D. Yi, X. Zhou, Y. Wen, and R. Tan, "Toward efficient compute-intensive job allocation for green data centers: A deep reinforcement learning approach," in *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 2019, pp. 634–644.

[25] Q. Pei, S. Chen, Q. Zhang, X. Zhu, F. Liu, Z. Jia, Y. Wang, and Y. Yuan, "CoolEdge: hotspot-relievable warm water cooling for energy-efficient edge datacenters," in *Proceedings of the 27th International Conference on Architectural Support for Programming Languages and Operating Systems*, 2022, pp. 814–829.

[26] Alibaba, "Price of the proportional solenoid valve," 2021, https://www.alibaba.com/product-detail/proportional-solenoid-valve_60730339977.html.

[27] Alibaba, "Price of the 2-way solenoid valve," 2023, https://www.alibaba.com/product-detail/2-2-Way-NC-2W160-15_1600473913429.html.

[28] X. Zhu, W. Jiang, F. Liu, Q. Zhang, L. Pan, Q. Chen, and Z. Jia, "Heat to power: Thermal energy harvesting and recycling for warm water-cooled datacenters," in *Proceedings of the 47th Annual International Symposium on Computer Architecture*, 2020.

[29] Q. Weng, W. Xiao, Y. Yu, W. Wang, C. Wang, J. He, Y. Li, L. Zhang, W. Lin, and Y. Ding, "MLaaS in the wild: Workload analysis and scheduling in large-scale heterogeneous GPU clusters," in *Proceedings of the 19th USENIX Symposium on Networked Systems Design and Implementation*, 2022.

[30] FPL, "Air-cooled chillers," 2025, https://infpl.fpl.com/business/pdf/air-cooled-chillers-primer.pdf.

[31] K. Kirkpatrick, "The carbon footprint of artificial intelligence," *Communications of the ACM*, vol. 66, no. 8, pp. 17–19, 2023.

[32] Y. Ran, H. Hu, X. Zhou, and Y. Wen, "DeepEE: Joint optimization of job scheduling and cooling control for data center energy efficiency using deep reinforcement learning," in *Proceedings of the 39th International Conference on Distributed Computing Systems*, 2019.

[33] F. Liu, Z. Zhou, H. Jin, B. Li, B. Li, and H. Jiang, "On arbitrating the power-performance tradeoff in saas clouds," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 10, pp. 2648–2658, 2013.

[34] J. Li, Y. Deng, Y. Zhou, Z. Zhang, G. Min, and X. Qin, "Towards thermal-aware workload distribution in cloud data centers based on failure models," *IEEE Transactions on Computers*, vol. 72, no. 2, pp. 586–599, 2022.

[35] O. Sarood, P. Miller, E. Totoni, and L. V. Kale, ""cool" load balancing for high performance computing data centers," *IEEE Transactions on Computers*, vol. 61, no. 12, pp. 1752–1764, 2012.

[36] L. Piga, I. Narayanan, A. Sundarrajan, M. Skach, Q. Deng, B. Maity, M. Chakkaravarthy, A. Huang, A. Dhanotia, and P. Malani, "Expanding datacenter capacity with DVFS boosting: A safe and scalable deployment experience," in *Proceedings of the 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 1*, 2024, pp. 150–165.

[37] P. Patel, E. Choukse, C. Zhang, Í. Goiri, B. Warrier, 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*, 2024, pp. 207–222.

[38] N. El-Sayed, I. A. Stefanovici, G. Amvrosiadis, A. A. Hwang, and B. Schroeder, "Temperature management in data centers: Why some (might) like it hot," in *Proceedings of the 12th ACM SIGMETRICS/PERFORMANCE joint international conference on Measurement and Modeling of Computer Systems*, 2012, pp. 163–174.

[39] M. P. David, M. Iyengar, P. Parida, R. Simons, M. Schultz, M. Gaynes, R. Schmidt, and T. Chainer, "Experimental characterization of an energy efficient chiller-less data center test facility with warm water cooled servers," in *Proceedings of the 28th Annual IEEE Semiconductor Thermal Measurement and Management Symposium*, 2012.

[40] Q. Pei, Y. Yuan, H. Hu, L. Wang, D. Zhang, B. Yan, C. Yu, and F. Liu, "Working smarter not harder: Hybrid cooling for deep learning in edge datacenters," *IEEE Transactions on Sustainable Computing*, 2025.

[41] W. Deng, F. Liu, H. Jin, C. Wu, and X. Liu, "Multigreen: Cost-minimizing multi-source datacenter power supply with online control," in *Proceedings of the fourth international conference on Future energy systems*, 2013, pp. 149–160.

[42] L. Gu, W. Zhang, Z. Wang, D. Zeng, and H. Jin, "Service management and energy scheduling toward low-carbon edge computing," *IEEE Transactions on Sustainable Computing*, vol. 8, no. 1, pp. 109–119, 2022.

[43] Í. Goiri, W. Katsak, K. Le, T. D. Nguyen, and R. Bianchini, "Parasol and greenswitch: Managing datacenters powered by renewable energy," *ACM SIGPLAN Notices*, vol. 48, no. 4, pp. 51–64, 2013.

[44] J. Sun, Z. Gong, A. Agarwal, S. Noghabi, R. Chandra, M. Snir, and J. Huang, "Exploring the efficiency of renewable energy-based modular data centers at scale," in *Proceedings of the 2024 ACM Symposium on Cloud Computing*, 2024, pp. 552–569.

[45] A. Souza, N. Bashir, J. Murillo, W. Hanafy, Q. Liang, D. Irwin, and P. Shenoy, "Ecovisor: A virtual energy system for carbon-efficient applications," in *Proceedings of the 28th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 2*, 2023, pp. 252–265.

Fangming Liu (S'08, M'11, SM'16) received the B.Eng. degree from the Tsinghua University, Beijing, and the Ph.D. degree from the Hong Kong University of Science and Technology, Hong Kong. He is currently a Full Professor with the Huazhong University of Science and Technology, Wuhan, China. His research interests include cloud computing and edge computing, datacenter and green computing, SDN/NFV/5G and applied ML/AI. He received the National Natural Science Fund (NSFC) for Excellent Young Scholars, and the National Program Special Support for Top-Notch Young Professionals. He is a recipient of the Best Paper Award of IEEE/ACM IWQoS 2019, ACM e-Energy 2018 and IEEE GLOBECOM 2011, the First Class Prize of Natural Science of Ministry of Education in China, as well as the Second Class Prize of National Natural Science Award in China.

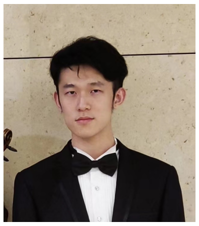
Qiangyu Pei received the B.S. degree in physics from Huazhong University of Science and Technology, China, in 2019, and the Ph.D. degree in computer architecture from Huazhong University of Science and Technology, in 2025. He is currently a research staff at the Central Software Institute, Distributed LAB of Huawei. His research interests include ML systems, edge computing, and sustainable computing.

Shutong Chen is an Assistant Professor at the School of Computer, Electronics and Information at Guangxi University. She received her Ph.D. in computer architecture from the School of Computer Science and Technology, Huazhong University of Science and Technology, and received her B.Sc. degree from the School of Mathematics, Hunan University. Her research interests include edge computing and green computing.

Yongjie Yuan received the B.Eng. degree from the School of Computer Science and Technology, Huazhong University of Science and Technology, China, in 2021, where he is currently pursuing the M.Eng. degree. His research interests include edge computing and serverless computing.

Qixia Zhang received his Ph.D. degree in 2021 and B.Eng. degree in 2016 from School of Computer Science and Technology, Huazhong University of Science and Technology, China. His research interests include network function virtualization, cloud computing and edge computing, datacenter and green computing, 5G network and network slicing. He is a recipient of the Best Paper Award of IEEE/ACM IWQoS 2019.

Xinhui Zhu received her M.S. degree in Computer Science and Technology from Huazhong University of Science and Technology, Wuhan, China, in 2021. She received her B.Eng. degree in Software Engineering from Hunan University, Changsha, China, in 2018. Her research interests include green computing and datacenter energy.



Ziyang Jia received his B.E. degree from Huazhong University of Science and Technology, Wuhan, China, in 2021. He is a Ph.D. student in University of California, Riverside now. His research interest includes computer architecture, high-performance computing, and GPU architecture.

Fei Xu received the B.S., M.E., and Ph.D. degrees in 2007, 2009, and 2014, respectively, all from the Huazhong University of Science and Technology (HUST), Wuhan, China. He received Outstanding Doctoral Dissertation Award in Hubei province, China, and ACM Wuhan & Hubei Computer Society Doctoral Dissertation Award in 2015. He is currently an associate professor with the School of Computer Science and Technology, East China Normal University, Shanghai, China. His research interests include cloud computing and datacenter, virtualization technology, and distributed systems.

Dong Zhang is a professorate senior engineer of Jinan Inspur Data Technology Co., Ltd. He has led the development of the world's highest computing and storage density rack server, the first China UNIX operating system. He has made creative contributions in areas such as converged architecture and high-end system software, earning one national award, and eleven provincial-level awards.

Bingheng Yan is a senior engineer of Jinan Inspur Data Technology Co., Ltd. He received his Ph.D. at Xi'an JiaoTong University in 2010, and broadly interested in the area of OS, virtualization, and cloud computing. He has led a wide range of virtualization research projects, and the development of Inspur's virtualization product InCloud Sphere, which break the global record of SpecVirt.