Outline

Introduction

Literature Review

Existing Methods

Research Gap

Problem Statement

Objectives

System Architecture

Results & Discussion

Conclusion & Future Scope

References

Thermal Aware Core-Level Scheduling For Kubernetes Workloads

Paper ID: wkp_ee104s1

Presented By: Amit Joshi

Authors: Viraj Nalbalwar, Pritesh Pagar, Ritesh Rokade, Shreeyash Patil, Amit Joshi

Conference: ISC High Performance 2026

Workshop: 2nd International Workshop on Energy Efficiency with Sustainable Performance

Friday, June 26, 2026

Outline

Introduction

Literature Review

Existing Methods

Research Gap

Problem Statement

Objectives

System Architecture

Results & Discussion

Conclusion & Future Scope

References

Data Sheet

- **Authors:** Viraj Nalbalwar, Pritesh Pagar, Ritesh Rokade, Shreeyash Patil, Amit Joshi
- **Affiliation:** Department of Computer Science & Engineering, COEP Technological University, Pune, India
- **Corresponding Author:** Viraj Nalbalwar
- **Email:** nalbalwarvs24.comp@coeptech.ac.in

Contents

Outline

Introduction

Literature Review

Existing Methods

Research Gap

Problem Statement

Objectives

System Architecture

Results & Discussion

Conclusion & Future Scope

References

1. Introduction
2. Literature Review
3. Existing Methods
4. Research Gap Analysis
5. Problem Statement
6. Objectives
7. System Architecture
8. Results & Discussion
9. Conclusion & Future Scope
10. References

Outline
Introduction
Literature Review
Existing Methods
Research Gap
Problem Statement
Objectives
System Architecture
Results & Discussion
Conclusion & Future Scope
References

Introduction

- Kubernetes is the de-facto standard for container orchestration, its default scheduler operates at node level only ignoring per-core CPU thermal state.
- Sustained CPU utilization causes localized thermal hotspots and triggers thermal throttling, degrading performance and hardware reliability.
- Thermally-aware schedulers exist for OS / Real-Time Operating System (RTOS) environments, but **none** target Kubernetes at CPU core granularity.
- This work implements a custom Kubernetes scheduler that selects the optimal CPU core per pod using real-time temperature and utilization metrics.

Literature Review

1. Thermal-Aware CPU Scheduling
   - Classical methods detect hotspots and redistribute CPU load.
   - Scheduling decisions can be improved using core temperature.
2. Per-Core Thermal & Utilization Modeling
   - Temperature depends on utilization, workload type, and cores.
   - Per-core sensors offer reasonably accurate thermal readings.
3. Core-Idling & Thermal Balancing Strategies
   - Core-idling is used to cool hotspots by temporarily reducing load.
   - Threads can be migrated to cooler cores to balance heat distribution.

Literature Survey contd...

4. Kubernetes Scheduler Architecture & Score Plugins
   - K8s scheduling pipeline uses plugin chain: PreFilter $\rightarrow$ Filter $\rightarrow$ Score $\rightarrow$ Reserve $\rightarrow$ Bind.
   - Score plugins enable custom ranking logic.
5. Kubernetes CPU Manager & cgroups
   - CPU Manager provides CPU assignment using cpuset cgroups.
   - It enforces core binding, does not choose which cores to allocate.
6. GPU Telemetry (NVIDIA Management Library - NVML)
   - NVML exposes GPU temperature, memory, and power metrics.
   - Useful for extending thermal-aware scheduling to GPUs.

Outline

Introduction

Literature Review

Existing Methods

Research Gap

Problem Statement

Objectives

System Architecture

Results & Discussion

Conclusion & Future Scope

References

Existing Methods

1. OS-level Thermal Schedulers
   - Approaches like Real-Time Thermal-Aware Scheduling (RT-TAS) and Multiprocessor System-on-Chip (MPSoC) schedulers use DVFS, migrations, and core-idling.
2. Kubernetes CPU Manager
   - Supports Guaranteed Quality of Service (QoS) with static core pinning via cpusets.
3. Telemetry-Aware Schedulers (Industry)
   - Some schedulers use platform metrics like utilization or power.
   - None use per-core temperature for scheduling decisions.
4. GPU Scheduling Frameworks
   - NVML exposes device-level GPU metrics.
   - Still not suitable for per-core CPU thermal scheduling.

Outline

Introduction

Literature Review

Existing Methods

Research Gap

Problem Statement

Objectives

System Architecture

Results & Discussion

Conclusion & Future Scope

References

Research Gap

Current Limitations in Kubernetes Scheduling:

- Most existing thermal-aware studies target embedded systems, RTOS, or OS-level CPU/GPU scheduling.
- No prior work applies thermal-aware techniques to Kubernetes.
- No system assigns pods to specific CPU cores based on real-time core thermal metrics.
- Kubernetes CPU Manager only enforces core binding, it cannot choose optimal cores.
- No thermal-aware per-core Score Plugin exists in the K8s scheduling framework.

Outline
Introduction
Literature Review
Existing Methods
Research Gap
Problem Statement
Objectives
System Architecture
Results & Discussion
Conclusion & Future Scope
References

Problem Statement

Core Problem: Existing Kubernetes scheduling operates only at the node level and ignores per-core thermal behavior. This work aims to develop a thermal-aware mechanism that selects cooler, lightly loaded cores for pod assignment.

Importance of the Problem:

- Reduces thermal throttling in CPU-heavy workloads.
- Achieves more predictable performance across nodes.
- Optimizes utilization of heterogeneous or thermally variant cores.
- Opens pathway for GPU-aware and energy-aware schedulers.

Outline

Introduction

Literature Review

Existing Methods

Research Gap

Problem Statement

Objectives

System Architecture

Results & Discussion

Conclusion & Future Scope

References

Objectives

Primary Objectives:

1. Design a real-time per-core thermal monitoring agent deployed as a Kubernetes DaemonSet.
2. Develop a custom scheduler that scores and selects the coolest, least-utilized CPU cores.
3. Enforce hard CPU core isolation on scheduled pods using Linux cgroup constraints.
4. Validate the system against the default Kubernetes scheduler under sustained CPU stress workloads.
5. Quantify gains in average temperature, peak heat, thermal throttling, and fan speed.

Outline

Introduction

Literature Review

Existing Methods

Research Gap

Problem Statement

Objectives

System Architecture

Results & Discussion

Conclusion & Future Scope

References

System Architecture

4-Layer Design: Workload → Scheduling & Control → Monitoring → Execution & Hardware

1. Node-Agent DaemonSet (Privileged Go Service)
   - Reads per-core temp from `/sys/class/hwmon` (coretemp), utilization from `/proc/stat`.
   - Simultaneous Multithreading (SMT) aware logical-to-physical core mapping, Exponential Moving Average (EMA) smoothing to suppress transient spikes.
   - Exposes real-time metrics via `GET /v1/readings/latest`.
2. Custom Thermal-Aware Scheduler (Go Deployment)
   - Polls every 5s for pods
   - Scores each core: $S = 0.65\,(1-\hat{T}) + 0.35\,(1-\hat{U})$ selects highest-scoring core.
   - Binds selected pod to node via standard Kubernetes API.

Outline

Introduction

Literature Review

Existing Methods

Research Gap

Problem Statement

Objectives

System Architecture

Results & Discussion

Conclusion & Future Scope

References

System Architecture contd...

3. CPU Affinity Enforcement (via Node-Agent)

- Writes selected logical CPU ID to pod's `cpuset.cpus` cgroup parameter.
- OS-level hard constraint, Linux kernel guarantees execution only on pinned core.
- Retry loop: 200 ms interval, 30 s timeout, zero Kubernetes core modifications.

Key Design Principles:

- No modifications to Kubernetes core components, fully cluster-compatible.
- All control-plane interactions via official Kubernetes client-go library.
- Proactive thermal balancing rather than reactive throttling.

Outline

Introduction

Literature Review

Existing Methods

Research Gap

Problem Statement

Objectives

System Architecture

Results & Discussion

Conclusion & Future Scope

References

Architectural Model

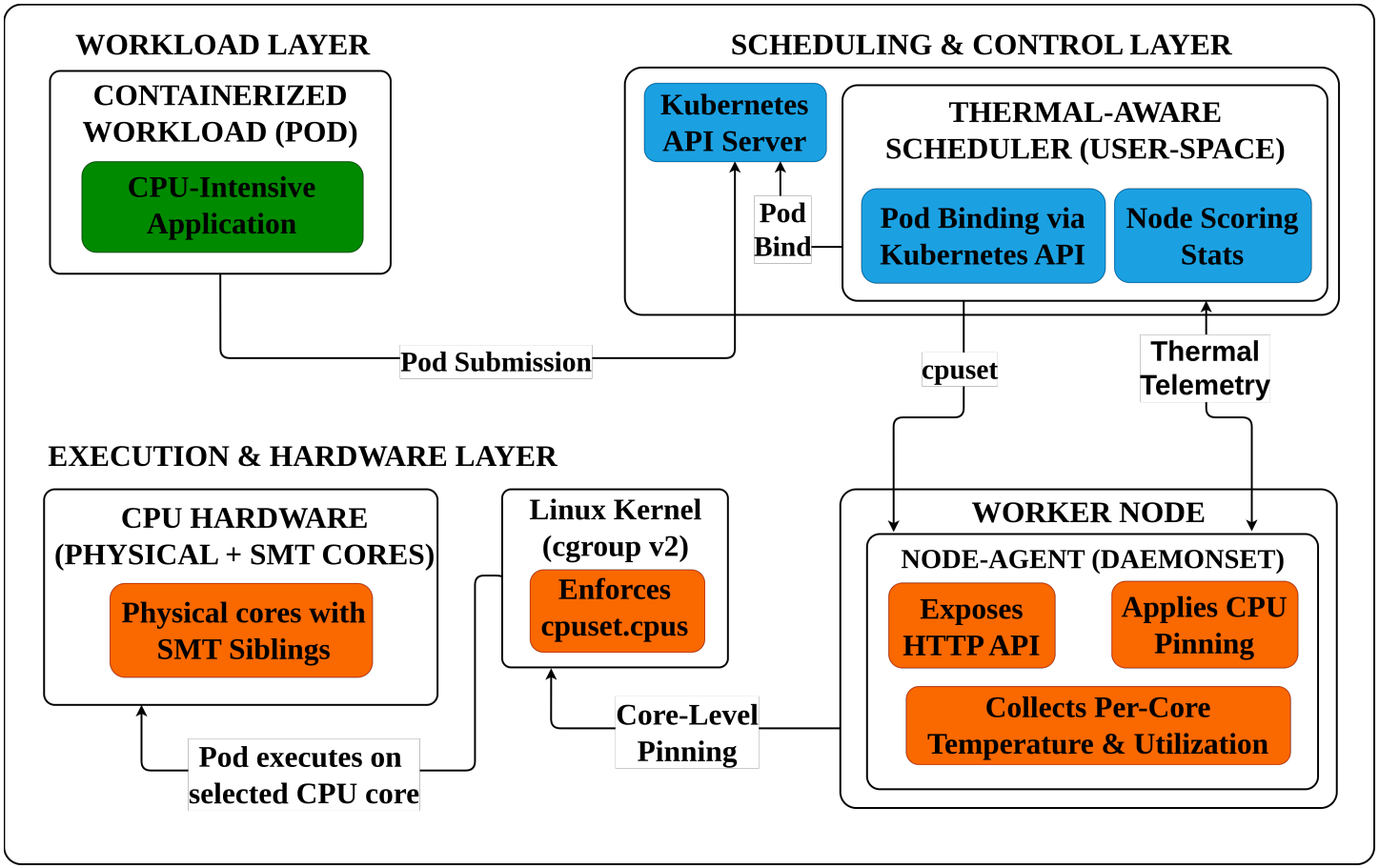

Figure 1: Thermal-Aware Core-Level Scheduling Pipeline

Outline

Introduction

Literature Review

Existing Methods

Research Gap

Problem Statement

Objectives

System Architecture

Results & Discussion

Conclusion & Future Scope

References

Results & Discussion

Experimental Setup:

- **Comparison:** Default Kubernetes Scheduler vs. Proposed Thermal-Aware Scheduler.
- **Platform:** Single-node cluster, Intel Core i7-13700HX, 16 physical / 24 logical CPUs, air-cooled.
- **Workload:** stress-ng benchmark, CPU-intensive pods.
- **Enforcement:** Thermal-aware pods pinned to logical CPU via Linux cpuset cgroups.
- **Scoring:** $S = 0.65\,(1 - \hat{T}) + 0.35\,(1 - \hat{U})$

Outline
Introduction
Literature Review
Existing Methods
Research Gap
Problem Statement
Objectives
System Architecture
Results & Discussion
Conclusion & Future Scope
References

Results & Discussion contd...

Scheduler Performance Comparison:

Table 1: Thermal Performance Comparison

Metric	Default	Thermal-Aware	Improvement
Avg. Temperature (°C)	80.0	63.2	21% ↓
Peak Temperature (°C)	95.0	66.0	30.5% ↓
Temp. Range (°C)	35.0	6.0	82.9% ↓
Fan Speed (RPM)	≈2900	≈2250	22.4% ↓
Thermal Throttling	Multiple (>90°C)	No	Eliminated
Core Pinning	Unconstrained	CPU 22 (0.726)	Optimized

Discussion:

- Temperature range: 35°C → 6°C near-uniform thermal distribution across cores.
- Thermal throttling completely eliminated, fan speed reduced by 22.4%.
- Stable 60–66°C operating band maintained under sustained stress workloads.

Outline

Introduction

Literature Review

Existing Methods

Research Gap

Problem Statement

Objectives

System Architecture

Results & Discussion

Conclusion & Future Scope

References

Conclusion & Future Scope

Conclusion:

- First thermal-aware Kubernetes scheduler at CPU *core* granularity, fully Kubernetes-native.
- End-to-end pipeline: Node-Agent $\rightarrow$ Thermal Scheduler $\rightarrow$ cpuset cgroup enforcement.
- **21%** avg. temp reduction (80°C $\rightarrow$ 63.2°C); **30.5%** peak reduction (95°C $\rightarrow$ 66°C).
- Throttling **completely eliminated**; temp range down 82.9%; fan speed reduced 22.4%.
- Zero Kubernetes core modifications — fully compatible with standard cluster deployments.

Outline

Introduction

Literature Review

Existing Methods

Research Gap

Problem Statement

Objectives

System Architecture

Results & Discussion

Conclusion & Future Scope

References

Conclusion & Future Scope contd...

Future Scope:

- Extend to **multi-node** clusters with cross-node thermal-aware pod placement.
- Integrate **RAPL** energy metrics for power-aware scoring alongside temperature.
- Dynamic weight tuning — adapt the 0.65/0.35 ratio based on workload intensity.
- **GPU thermal scheduling** via NVML for heterogeneous GPU-accelerated workloads.

References

[1] Carrión, Carmen. "Kubernetes scheduling: Taxonomy, ongoing issues and challenges." ACM Computing Surveys 55.7 (2022): 1-37.

[2] Wen, Shilin, et al. "K8sSim: A simulation tool for Kubernetes schedulers and its applications in scheduling algorithm optimization." Micromachines 14.3 (2023): 651.

[3] Kubernetes, Cloud Native Computing Foundation, https://kubernetes.io/.

[4] Senjab, Khaldoun, et al. "A survey of Kubernetes scheduling algorithms." Journal of Cloud Computing 12.1 (2023): 87.

[5] Mondal, Subrota Kumar, Zhen Zheng, and Yuning Cheng. "On the optimization of Kubernetes toward the enhancement of cloud computing." Mathematics 12.16 (2024): 2476.

[6] Rao, Wei, and Hongjian Li. "Energy-aware Scheduling Algorithm for Microservices in Kubernetes Clouds." Journal of Grid Computing 23.1 (2025): 2.

[7] Piontek, Tobias, Kawsar Haghshenas, and Marco Aiello. "Carbon emission-aware job scheduling for Kubernetes deployments." Journal of supercomputing 80 (2023): 549-569.

[8] Dowling, Anthony, et al. "Regulating CPU temperature with thermal-aware scheduling using a reduced order learning thermal model." Future Generation Computer Systems 166 (2025): 107687.

[9] Agrawal, M., and D. A. Mehta. "Thermal Aware Process Scheduling for Multicore Processors". Transactions on Engineering and Computing Sciences, vol. 12, no. 06, Dec. 2024, pp. 39-53, doi:10.14738/tmlai.1206.18051.

[10] Ilager, Shashikant, Kotagiri Ramamohanarao, and Rajkumar Buyya. "Thermal prediction for efficient energy management of clouds using machine learning." IEEE Transactions on Parallel and Distributed Systems 32.5 (2020): 1044-1056.

[11] Ladge, Leena, and Y. Srinivasa Rao. "Analysis of Core Temperature Dynamics in Multi-Core Processors." Journal of Low Power Electronics and Applications 15.4 (2025): 68.

References contd...

[12] Li, Jie, et al. "Towards energy-efficient and thermal-aware data placement for storage clusters." IEEE Transactions on Sustainable Computing 9.4 (2024): 631-647.

[13] Chu, Kexin, et al. "eInfer: Unlocking Fine-Grained Tracing for Distributed LLM Inference with eBPF." Proceedings of the 3rd Workshop on EBPF and Kernel Extensions. 2025.

[14] Schöne, Robert, et al. "Energy efficiency features of the intel alder lake architecture." Proceedings of the 15th ACM/SPEC International Conference on Performance Engineering. 2024.

[15] NVIDIA Corporation. NVIDIA Management Library (NVML) Developer Guide. NVIDIA Developer, https://developer.nvidia.com/management-library-nvml.

[16] Lu, Rui, and Dan Wang. "A Thermal-aware Workload Scheduler for High-performance LLM Inference in Cooling-regulated Datacenters." ACM SIGENERGY Energy Informatics Review 5.2 (2025): 98-104.

[17] Stojkovic, Jovan, et al. "Tapas: Thermal-and power-aware scheduling for LLM inference in cloud platforms." Proceedings of the 30th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 2. 2025.

[18] Chaudhry, Muhammad Tayyab, et al. "Thermal-aware scheduling in green data centers." ACM Computing Surveys (CSUR) 47.3 (2015): 1-48.

[19] Mao, Li, et al. "A resource scheduling method for cloud data centers based on thermal management." Journal of Cloud Computing 12.1 (2023): 84.

[20] Kocot, Bartłomiej, Paweł Czarnul, and Jerzy Proficz. "Energy-aware scheduling for high-performance computing systems: A survey." Energies 16.2 (2023): 890.

[21] Senjab, Khaldoun, et al. "A survey of Kubernetes scheduling algorithms." Journal of Cloud Computing 12.1 (2023): 87.

[22] Rejiba, Zeineb, and Javad Chamanara. "Custom scheduling in kubernetes: A survey on common problems and solution approaches." ACM Computing Surveys 55.7 (2022): 1-37.

References contd...

[23] Monaco, Gabriele, Gautam Gala, and Gerhard Fohler. "Shared resource orchestration extensions for kubernetes to support real-time cloud containers." 2023 IEEE 26th international symposium on real-time distributed computing (ISORC). IEEE, 2023.

[24] Furnadzhiev, Radoslav. "An Experimental Evaluation of Latency-Aware Scheduling for Distributed Kubernetes Clusters." Engineering Proceedings 100.1 (2025): 25.

[25] Marchese, Angelo, and Orazio Tomarchio. "Enhancing the Kubernetes Platform with a Load-Aware Orchestration Strategy." SN Computer Science 6.3 (2025): 1-15.

[26] Sofia, Rute C., et al. "A framework for cognitive, decentralized container orchestration." IEEE Access 12 (2024): 79978-80008.

[27] Ye, Zhisheng, et al. "Deep learning workload scheduling in gpu datacenters: A survey." ACM Computing Surveys 56.6 (2024): 1-38.

[28] Sheshadri, K. R., and J. Lakshmi. "Qos aware faas for heterogeneous edge-cloud continuum." 2022 IEEE 15th International Conference on Cloud Computing (CLOUD). IEEE, 2022.

[29] Raith, Philipp, et al. "Opportunistic Energy-Aware Scheduling for Container Orchestration Platforms Using Graph Neural Networks." 2024 IEEE 24th International Symposium on Cluster, Cloud and Internet Computing (CCGrid). IEEE, 2024.

[30] Liu, Peini, and Jordi Guitart. "Dynamic in-node group-aware scheduling for multi-tenant machine learning services on Kubernetes." 2025 IEEE 18th International Conference on Cloud Computing (CLOUD). IEEE, 2025.

[31] Yang, Jialin, et al. "A Survey on Task Scheduling in Carbon-Aware Container Orchestration." arXiv preprint arXiv:2508.05949 (2025).

[32] Rao, Wei, and Hongjian Li. "Energy-aware Scheduling Algorithm for Microservices in Kubernetes Clouds." Journal of Grid Computing 23.1 (2025): 2.

Outline

Introduction

Literature Review

Existing Methods

Research Gap

Problem Statement

Objectives

System Architecture

Results & Discussion

Conclusion & Future Scope

References

References contd...

[33] Jain, Rutwik, et al. "PAL: A Variability-Aware Policy for Scheduling ML Workloads in GPU Clusters." SC24: International Conference for High Performance Computing, Networking, Storage and Analysis. IEEE, 2024.

[34] Cheng, Wenliang, et al. "CSFRL: A Reinforcement Learning Technology Enabled Computing Power Scheduling Framework Based on Kubernetes." 2023 IEEE 34th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC). IEEE, 2023.

[35] Muddada, Satya Teja. "Carbon-Aware Cloud Architecture: Dynamic Multi-Cloud Scheduling for Sustainable Computing." Journal of Computer Science and Technology Studies 7.10 (2025): 511-520.

[36] Chen, Rui, et al. "Power and thermal-aware virtual machine scheduling optimization in cloud data center." Future Generation Computer Systems 145 (2023): 578-589.

[37] Akgün, Gökhan, and Diana Göhringer. "Balancing Power and Performance With Task Dependencies in Multi-Core Systems." IEEE Access (2025).

[38] Shor, Joseph. "Compact thermal sensors for dense CPU thermal monitoring and regulation: A review." IEEE Sensors Journal 21.11 (2020): 12774-12788.

[39] Marchese, Angelo, and Orazio Tomarchio. "Telemetry-driven microservices orchestration in cloud-edge environments." 2024 IEEE 17th International Conference on Cloud Computing (CLOUD). IEEE, 2024.

[40] Pourmohseni, Behnaz, et al. "Task migration policy for thermal-aware dynamic performance optimization in many-core systems." IEEE Access 10 (2022): 33787-33802.

Outline

Introduction

Literature Review

Existing Methods

Research Gap

Problem Statement

Objectives

System Architecture

Results & Discussion

Conclusion & Future Scope

References

Questions/ Discussion