# CEDAR: Carbon Efficient Dynamic Allocation and Routing for Agentic LLM Inference

Amit More
University of York
York, United Kingdom
amit.more@york.ac.uk

Tarique Anwar
RMIT University
Melbourne, Australia
tarique.anwar@rmit.edu.au

Poonam Yadav
University of York
York, United Kingdom
poonam.yadav@york.ac.uk

## Abstract

LLM inference now dominates operational AI compute, yet production serving stacks typically optimise for performance alone, leaving cost and carbon unmanaged. We present CEDAR, a queue level multi objective control framework for agentic LLM inference that jointly optimises tail latency, cloud cost, and marginal carbon emissions. CEDAR observes backlog depth, waiting time percentiles, and service level objective (SLO) slack to route mixed criticality requests across heterogeneous, geo-distributed fleets. In trace-driven evaluation, CEDAR reduces cost by up to 26% and carbon by up to 27% relative to a *Performance-Only* baseline, while maintaining competitive p95 latency (0.88 s) and low SLO violation (4.3%). These results indicate queue level control as a practical path to sustainable agentic inference without unacceptable QoS degradation.

## CCS Concepts

• **Computer systems organization** → **Availability**; **Reliability**; • **Computing methodologies** → **Multi-agent planning**; **Multi-agent systems**; **Intelligent agents**; • **Information systems** → **Data centers**.

## Keywords

LLM inference, carbon aware scheduling, multi-objective optimization, agentic AI, queue management, sustainable computing, energy-efficient systems, workload scheduling

**ACM Reference Format:**
Amit More, Tarique Anwar, and Poonam Yadav. 2026. CEDAR: Carbon Efficient Dynamic Allocation and Routing for Agentic LLM Inference. In *Workshop on Systems and Methods for Sustainable Large-Scale AI (GreenSys '26), April 27–30, 2026, Edinburgh, Scotland UK*. ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3802973.3804457

## 1 Introduction

The AI inference market is projected to grow from $106B in 2025 to $255B by 2030, driven by widespread deployment of LLM-powered services across industries [11]. This growth is no longer confined to chatbots: LLMs now power *agentic coding assistants* (GitHub Copilot, Cursor, Devin) that autonomously generate, refactor, and debug production code; *enterprise automation* pipelines that process contracts, invoices, and customer support tickets at scale; *clinical decision support* tools that summarise patient records and flag drug interactions in real time; and *scientific research* accelerators that parse literature and generate hypotheses across biology, materials science, and drug discovery. Each of these use cases generates continuous, heterogeneous inference demand ranging from latency-critical interactive requests to deferrable batch jobs and runs 24 hours a day across global infrastructure. This growth comes at significant environmental and economic cost, as shown in Figure 1, global data centres consumed 415 TWh in 2024, growing at 12% per year over the last five years and are projected to reach 945 TWh by 2030, representing nearly 3% of global electricity consumption [11]. In the US alone, AI specific servers will grow from 53-76 TWh (2024) to 165-326 TWh by 2028 [19], with a single LLM query consuming 10× more electricity than a traditional web search [23].

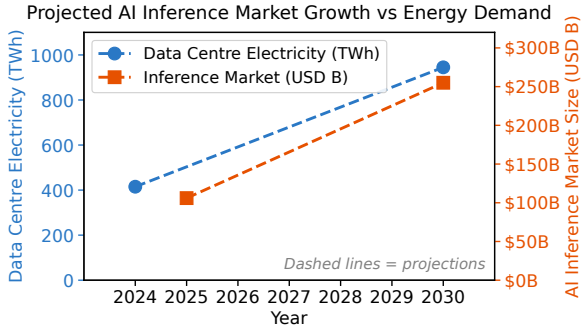


**Figure 1: Projected AI inference market growth (right axis) versus global data centre electricity demand (left axis). Energy figures from IEA [11] as reported by MIT Technology Review [14]: 415 TWh (2024) rising to 945 TWh by 2030. Inference market projections from $106B (2025) to $255B (2030) [13]. Dashed lines indicate projections between sourced anchor points.**

As agentic workloads scale from thousands to millions of daily requests [7], the aggregate carbon footprint of inference infrastructure becomes a first-class operational concern, yet current serving systems treat energy and carbon as afterthoughts, optimising exclusively for throughput and latency. The emergence of agentic AI systems amplifies these challenges. A single agentic coding task generates 10+ sequential LLM calls, creating sustained queue backlogs and 37% KV cache recomputation overhead [7]. These workloads exhibit mixed-criticality characteristics. For example, interactive requests (e.g., autocomplete) require sub-second response times, while background tasks such as code analysis, batch test generation, and offline document processing can tolerate delays of seconds to


*GreenSys '26, Edinburgh, Scotland Uk*

ACM ISBN 979-8-4007-2174-8/26/04
https://doi.org/10.1145/3802973.3804457

minutes, creating scheduling opportunities that current systems fail to exploit.

**The Problem.** Current LLM serving systems (TensorRT-LLM [17], SGLang [26], vLLM [12]) optimise a single objective (latency or throughput) using simple routing strategies (round robin, least connections). No production system jointly considers monetary cost, carbon intensity, and request priority in routing decisions. This single-objective approach leaves substantial performance and efficiency gains unexploited. Empirical evidence indicates that head-of-line blocking inflates average latency by up to 5.3× [18], prefill-decode interference causes generation stalls that degrade overall system responsiveness [15]. In parallel, infrastructure fragmentation results in underutilisation, with approximately 28% throughput [8], and systems rarely exploit real time carbon intensity variation across regions [5, 24]. Furthermore, naive per request carbon metrics conflate sunk carbon (baseline infrastructure) with marginal carbon (schedulable emissions), leading schedulers to optimise the wrong target [3]. The above problem and associated challenges motivate the central research question:

*Can queue-level, multi-objective control jointly minimise latency, cloud cost, and carbon emissions for heterogeneous agentic LLM inference workloads, without unacceptable degradation of any single objective?*

This question recognises that the optimisation target is inherently multi dimensional and that the queue not the individual request nor the individual instance is the natural control granularity at which latency, cost, and carbon interact and can be co-managed.

To answer this question, we make the following contributions:

- We introduce CEDAR, a queue level multi objective control framework for agentic LLM inference that integrates latency, cost, and marginal carbon into a unified routing and scaling policy.
- We formulate fleet routing as a constrained Markov Decision Process (CMDP) using queue-level Service level Objective slack and real time carbon signals.
- We show via trace driven evaluation that CEDAR reduces cost by up to 26% and carbon by up to 27% relative to a *Performance-Only* baseline while maintaining competitive tail latency.

The remainder of this paper is organised as follows. Section 2 reviews existing LLM serving systems and identifies key gaps in multi-objective and carbon-aware scheduling. Section 3 presents the proposed CEDAR framework, detailing its architecture, design components, and control formulation. Section 4 evaluates the proposed approach through trace-driven simulation and compares it against established baselines using latency, cost, and carbon metrics. Finally, section 5 concludes the paper and outlines directions for future work.

## 2 Existing Gaps in LLM Serving Systems

Despite rapid advances in LLM serving, existing systems optimise along isolated dimensions and at limited control granularities. No prior work jointly integrates multi objective optimisation, queue level abstraction, heterogeneous fleet routing, and real time carbon awareness within a unified serving framework.

### 2.1 Single-Objective Scheduling

Recent LLM schedulers primarily target latency and throughput. Latency Tail Reduction (LTR) scheduling [6, 18] mitigates head of line blocking, while Niyama [8] improves mixed criticality isolation. Astraea [16] formalises the NP-hardness of optimal LLM scheduling. However, these systems optimise exclusively for performance and do not incorporate cost or carbon objectives into their control decisions.

Energy oriented systems such as DynamoLLM [21] reduce cluster level energy consumption, and BrownoutServe [10] improves SLO resilience under bursty Mixture of Experts (MoE) workloads. Yet these approaches operate within single clusters and lack cross-region optimisation or carbon aware routing capabilities.

### 2.2 Infrastructure Level Carbon Optimisation

Carbon aware systems such as CarbonScaler [3] and GreenScale [25] focus on infrastructure level workload shifting. Prior work shows that naive temporal or spatial shifting may increase total emissions without careful accounting [2]. The Sunk Carbon Fallacy [3] emphasises the need for marginal rather than average carbon metrics, while Fair $CO_2$ [20] studies carbon attribution mechanisms. SLIT [4] proposes a multi objective framework for geo distributed LLM scheduling that co optimises TTFT, carbon emissions, water usage, and energy costs using a meta-heuristic approach. However, these efforts do not integrate carbon signals directly into LLM serving schedulers at the queue level. Although real time carbon intensity data is available via WattTime [24] and Electricity Maps [5], to our knowledge no LLM serving framework incorporates these signals into queue level routing or multi objective scheduling decisions. As a consequence, four persistent inefficiencies remain: (a) head-of-line blocking inflates latency by up to 5.3× [18], (b) prefill-decode interference causes generation stalls [15], (c) siloed infrastructure underutilised approximately 28% throughput [8, 22], (d) regional carbon intensity variation remains unexploited despite significant inter-region differentials [5, 24].

These deficiencies share a common root cause, i.e, the absence of multi objective, fleet wide, queue aware control that simultaneously reasons about latency, cost, and marginal carbon.

### 2.3 Agentic Workload Amplification

Agentic coding workloads [7], characterised by multi step session structures, mixed criticality tiers, bursty inter arrival patterns, and highly variable token lengths, amplify these inefficiencies. Existing schedulers treat requests largely independently and fail to exploit session level SLO slack or deferability across workflow stages. The proposed method, CEDAR, addresses these gaps by elevating control to the queue level, integrating marginal carbon signals into routing decisions, and enabling heterogeneous cross-region scheduling under a unified multi objective policy guided by a validated heuristic and a Soft Actor Critic (SAC) [1, 9] based DRL controller.

## 3 Proposed Solution: CEDAR

CEDAR is a fleet level serving framework that jointly optimises tail latency, cloud cost, and marginal carbon for mixed criticality *agentic* LLM workloads by elevating control to the *queue abstraction.*

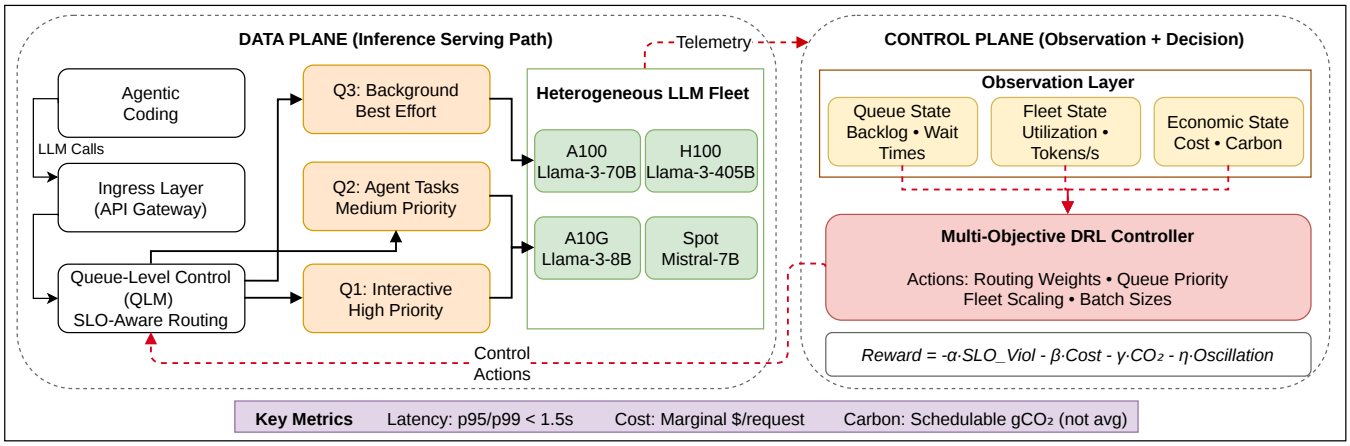


**Figure 2: CEDAR architecture and control flow. Agentic pipelines generate mixed criticality LLM invocations. The ingress layer classifies requests into virtual queues. The controller observes queue level state, fleet telemetry, and carbon/pricing signals to decide routing and scaling actions over a heterogeneous, geo distributed fleet.**

Figure 2 illustrates the full flow: (1) an *Ingress Layer* classifies each incoming agent step by criticality and SLO, because mixed criticality is the core enabling structure for trading slack for sustainability; (2) *Global Virtual Queues* maintain per class backlog and SLO slack signals, because queue level state best reveals impending tail-latency risk and available deferability; (3) a *Multi-Objective Controller* periodically observes queue/fleet/carbon state and selects routing and scaling actions, because latency, cost, and carbon interact across the fleet and must be co managed; (4) a *Heterogeneous Fleet* executes requests across regions and model sizes, because energy and $/token vary strongly across models and regions, enabling savings through model aware and region aware routing; and (5) a *Carbon Integration Layer* supplies real time (or replayed) carbon intensity signals, because emissions differ materially across regions and time and must enter the control objective. Algorithm 1 formalises this control loop. The remainder of this section unpacks each component in the same order as Figure 2, then defines the controller formulation and constraints.

## 3.1 Workload and System Assumptions

**Agentic workload model.** CEDAR targets agentic coding pipelines composed of multi step sessions (10+ LLM calls) spanning three criticality tiers: *completion* (HIGH, 500 ms SLO), *refactoring* (MEDIUM, 2 s SLO), and *analysis/test generation* (LOW, 5 s SLO). Approximately 40% of requests belong to LOW priority stages, creating temporal slack exploitable for cost and carbon optimisation.

**Carbon heterogeneity.** Grid carbon intensity varies across regions and time. CEDAR consumes live carbon signals ($gCO_2$/kWh) from WattTime [24] and Electricity Maps [5], routing deferable stages toward lower carbon regions when SLO slack permits.

**Queue-level observability.** Queue-level metrics (backlog, p95 wait, SLO slack) provide earlier overload signals than per instance utilisation and reduce routing oscillation relative to naive policies, motivating queue level control.

## 3.2 Ingress Layer: classification and admission

The ingress layer converts raw agentic steps into scheduling relevant metadata (priority tier, SLO deadline, token estimate, and task type). Agentic workloads are inherently mixed criticality: interactive steps (e.g., completion) require strict deadlines, while background steps (e.g., analysis, test generation) are deferrable. Without explicit classification, the controller cannot safely trade slack for cost and carbon savings without harming user facing QoS. Each request is tagged with (tier $\in$ {`HIGH`, `MEDIUM`, `LOW`}, deadline, estimated tokens, task type), then forwarded to the appropriate virtual queue.

## 3.3 Global Virtual Queues: queue level abstraction and slack tracking

Virtual queues maintain per tier backlog, waiting time percentiles, and SLO slack. Queueing dynamics dominate tail latency; per instance utilisation is a lagging indicator and per request scheduling is too myopic under bursty token-length variance. Queue level signals (backlog, p95 wait, slack to deadline) provide earlier warning of overload and expose safe windows to route to cheaper or greener options. Within each queue, requests are prioritised by deadline aware ordering (EDF within tier) to protect tail latency for critical tiers.

## 3.4 Carbon Integration Layer: decision relevant carbon signals

The carbon layer supplies region specific carbon intensity $c_r(t)$ ($gCO_2$/kWh) and optionally forecasts. Emissions are not uniform across regions and time; routing and deferral decisions are only meaningful if the controller can observe and learn from these differences. Following the marginal carbon principle [3], CEDAR treats carbon signals as decision relevant rather than attributing all baseline emissions to each request.

**Integration.** Carbon signals enter the controller state and reward

and are refreshed at a slower cadence than per request placement (every few minutes via WattTime [24] and Electricity Maps [5]).

### 3.5 Heterogeneous Fleet: regions, models, and energy asymmetry

The fleet consists of heterogeneous model deployments (different sizes and capabilities) across multiple regions. Model size drives large differences in energy and cost per generated token; region and instance choice also changes carbon intensity and price. This heterogeneity enables multi-objective optimisation: the controller can route *deferrable* or *less critical* steps to cheaper or greener resources while reserving high performance capacity for strict deadline tiers. Each region and model reports tokens/sec, GPU utilisation, and memory pressure to the controller.

### 3.6 Multi Objective Controller: CMDP formulation and actions

The controller selects routing weights and bounded scaling actions at fixed intervals (every 10 s). Independent heuristics (e.g., *least-loaded*, carbon-only) overfit a single objective and can oscillate under bursty mixed criticality workloads. A single controller that jointly reasons over queue state, fleet state, and carbon and pricing signals is required to maintain QoS while reducing cost and emissions.

*CMDP formulation.* We model control as a constrained Markov Decision Process (CMDP). The state $s_t$ includes: (i) queue state (backlog, waiting-time percentiles, SLO slack by tier), (ii) fleet state (tokens/sec, GPU utilisation, memory pressure by region/model), and (iii) external state (pricing and carbon intensity $c_r(t)$). Actions $a_t$ include routing weights from each queue to each region/model and bounded scaling targets. A primary QoS constraint enforces per-tier violation budgets:

$$\begin{aligned} \mathbb{E}[\text{SLOViol}_{\text{HIGH}}] &\le \epsilon_{\text{HIGH}}, \\ \mathbb{E}[\text{SLOViol}_{\text{MED}}] &\le \epsilon_{\text{MED}}, \\ \mathbb{E}[\text{SLOViol}_{\text{LOW}}] &\le \epsilon_{\text{LOW}}. \end{aligned} \tag{1}$$

*Reward.* The controller maximises a weighted reward that penalises violations, cost, marginal emissions, and oscillation:

$$R = -\alpha \cdot \text{Violations} - \beta \cdot \text{Cost} - \gamma \cdot \text{CO}_2^{\text{marg}} - \eta \cdot \text{Oscillation}. \tag{2}$$

Here $\text{CO}_2^{\text{marg}}$ denotes decision relevant marginal operational emissions computed from request energy and regional carbon intensity signals [3]. Tier dependent penalties implement mixed-criticality: HIGH requests receive the strongest violation penalty.

*Safety and stability.* To avoid harmful oscillations, actions are rate limited and scaling is bounded; the controller falls back to a *least-loaded* policy if constraints would be violated.

### 3.7 End to end control loop

Algorithm 1 formalises the end to end loop: carbon signals are refreshed on a slower cadence, while per request placement uses the latest queue and fleet state and the current policy (heuristic initially, SAC [1, 9] once training stabilises).

**Algorithm 1:** CEDAR Control Loop

**Input:** Request stream $\mathcal{S}$; fleet $\mathcal{F}$ (geo distributed regions/models); active policy $\pi$ (heuristic → SAC [1, 9]); replay buffer $\mathcal{B}$.
**Output:** Routing decisions, cost/carbon metrics, SLO attainment.
Set active policy ← heuristic ▷ `SAC takes over once training stabilises`

Cache carbon intensity $\hat{c}_r \leftarrow 0$ for each region $r$
`// Control plane: refresh carbon signals periodically`
**while** *system is running* **do**
  **foreach** *region* $r$ **do**
    $\hat{c}_r \leftarrow$ fetch carbon intensity via WattTime / Electricity Maps
  **end**
  `// Data plane: process each incoming request`
  **foreach** *request* $x$ *arriving from* $\mathcal{S}$ **do**
    Ingress: assign tier, deadline, token estimate, task type
    Build observation $s$ from queue state, fleet telemetry, pricing, and $\hat{c}$
    Select (region, model) action $(r^*, m^*) \leftarrow \pi(s)$
    **if** *request is* `DEFERRABLE` ***and*** *queue at* $r^*$ *is full* **then**
      defer $x$ to a future control interval (up to its deadline) and continue
    **end**
    Enqueue $x$ into tiered virtual queue at $r^*$
    Execute on $m^*$ when scheduled; observe latency and outcome
    Compute cost and $\text{CO}_2^{\text{marg}}$ using $\hat{c}_{r^*}$
    Store transition in $\mathcal{B}$ for DRL training
  **end**
  `// SAC training: update periodically from replay buffer`
  **if** $|\mathcal{B}| \ge$ *minimum buffer size* **then**
    Sample mini batch; update critics and actor; soft update targets
  **end**
**end**

## 4 Evaluation and Results

*Experimental Setup.* We evaluate CEDAR via trace-driven discrete-event simulation, modelling queue dynamics, batching, and token-level generation latency using throughput profiles measured offline from vLLM [12] on NVIDIA A100 and H100 GPUs for Llama-3-7B and Llama-3-70B (batch sizes 1-128). Carbon intensity signals are sourced from WattTime [24] at 5-minute granularity across three AWS regions (us-east-1, us-west-2, eu-west-1), covering diurnal and weather-driven variation from January-March 2025.

The fleet spans 12 GPU instances across three regions (4× A100-80GB us-east-1, 4× H100-80GB us-west-2, 4× A100-40GB eu-west-1), running Llama-3-70B on the larger and Llama-3-7B on the smaller instances. Routing decisions are made every 100 ms; all baselines share the same fleet, model assignments, and carbon traces for controlled comparison.

**Simulation rationale.** We use simulation rather than live deployment due to the resource requirements of the heterogeneous fleet and the need for reproducible carbon intensity replay across baselines. Throughput profiles are grounded in offline vLLM measurements, providing realistic serving dynamics. Live deployment is the primary next step (Section 5).

*Workload.* We emulate agentic coding sessions as sequences of dependent LLM calls with mixed criticality tiers. Each experiment replays *10,000 requests* drawn from three classes in a fixed ratio: *30% HIGH* (interactive completion, 500 ms SLO), *30% MEDIUM* (refactoring, 2 s SLO), and *40% LOW* (analysis/test generation, 5 s SLO) — consistent with the 40% deferrable share assumed in Section 3.1. Requests arrive according to a Poisson process at a *mean rate of*

*50 req/s*, overlaid with synthetic bursts (3× peak rate, 10 s duration, every 5 min) to stress tail-latency behaviour. Token lengths follow a heavy-tailed log-normal distribution ($\mu$=6.5, $\sigma$=1.2, in log-tokens) to reflect real prompt/response variability [7]. Incoming requests are mapped to a region/model pair by the CEDAR controller policy; baselines apply their own routing rules (*round-robin*, *least-loaded*, or *performance-only* latency scoring) across the same 12-instance fleet. A synthetic workload was used to provide reproducible, controlled experiments across all baselines and to isolate the contribution of each criticality class; evaluation against publicly available LLM serving traces (e.g., Azure conversation datasets) is planned as future work.

*Baselines.* We compare against: (i) *Round Robin* routing, (ii) *Least Loaded* routing (routes to lowest utilisation), and (iii) *Performance-Only* routing (optimises latency without carbon/cost terms). All baselines share identical fleet and batching parameters unless explicitly varied.

*Metrics.* We report: (i) p95 end-to-end latency, (ii) SLO violation rate (fraction of requests exceeding their deadline), (iii) cost per request ($/req), (iv) marginal carbon emissions per request ($gCO_2$/req), and (v) *control oscillation*, defined as:

$$\text{Osc} = \frac{1}{T}\sum_{t=1}^{T}\frac{1}{|\mathcal{F}|}\sum_{f\in\mathcal{F}}\left|w_t^f - w_{t-1}^f\right|, \tag{3}$$

where $w_t^f \in [0,1]$ is the routing weight assigned to fleet endpoint $f$ at control interval $t$, $|\mathcal{F}|$ is the number of fleet endpoints, and $T$ is the total number of intervals. Lower oscillation indicates more stable routing decisions; high oscillation causes unnecessary load rebalancing and can degrade latency under bursty workloads.

*Implementation.* Algorithm 1 defines the control loop and the interaction between ingress classification, tiered virtual queues, controller actions, and fleet execution. Carbon intensity is refreshed on a slower cadence than request arrivals; routing and queuing use the latest cached values. The DRL policy (SAC) is trained from replay buffer transitions; when training is not yet stable, CEDAR operates under a validated heuristic policy to ensure safe behaviour.

*Runtime overhead.* Each CEDAR scheduling cycle building the CMDP state vector, SAC actor inference, and routing table dispatch completes well within the 100 ms control interval. All steps are $O(|\mathcal{F}|)$ or a single forward pass, incurring negligible overhead. Carbon refresh runs on a separate 5-minute cadence. Formal profiling on live infrastructure is left to future work.

*Overall performance.* Table 1 summarises results across all baselines described in Section 4. Relative to *Performance-Only*, CEDAR reduces cost by 26% ($0.00256→$0.00189/req) and marginal carbon by 27% (93.2→67.8 $gCO_2$/req), while maintaining competitive p95 latency (0.88 s vs 0.76 s, +16%). Compared to *Round Robin*, CEDAR lowers SLO violations (22.3% →4.3%) and reduces routing oscillation by 63% (0.51→0.19). These results demonstrate that queue level multi objective control achieves sustainability gains without unacceptable QoS degradation.

*Cost carbon trade-off and latency preservation.* Figure 3 shows the cost carbon frontier: CEDAR occupies the Pareto-dominant region, achieving the lowest cost and marginal emissions simultaneously. *Performance-Only* achieves slightly lower latency but incurs 35% higher cost and 37% higher emissions, indicating that performance centric routing fails to exploit deferrable workload slack. Figure 4 shows latency percentiles (p50–p99.9): while *Performance-Only* achieves the lowest p95, CEDAR maintains competitive tail behaviour and significantly outperforms *Round Robin* and *Least Loaded* at higher percentiles. This confirms that carbon aware optimisation does not introduce uncontrolled tail amplification.

**Table 1: CEDAR vs baselines**

| System | p95(s) | SLO% | $/req | $gCO_2$/req | Osc |
|---|---|---|---|---|---|
| CEDAR | 0.88 | 4.3 | **0.00189** | **67.8** | **0.19** |
| Round Robin | 1.58 | 22.3 | 0.00245 | 95.7 | 0.51 |
| Least Loaded | 1.21 | 12.8 | 0.00215 | 79.3 | 0.42 |
| Performance-Only | **0.76** | **3.2** | 0.00256 | 93.2 | 0.29 |

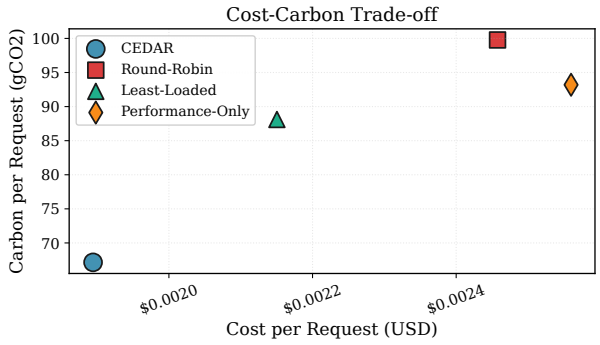


**Figure 3: Cost–carbon Pareto frontier. CEDAR achieves lowest cost and emissions simultaneously, while *Performance-Only* incurs 35% higher cost and 37% higher emissions.**

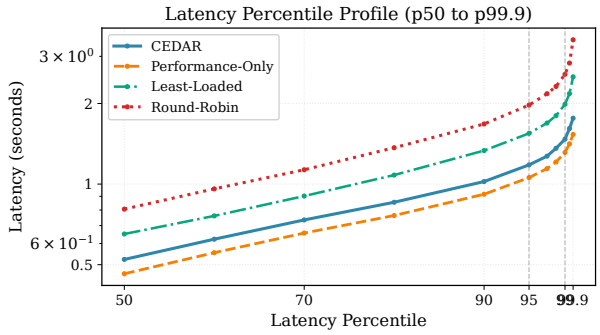


**Figure 4: Latency percentiles (p50–p99.9). CEDAR maintains competitive tail latency while optimizing sustainability.**

*Component analysis.* Table 2 evaluates the contribution of each mechanism. Removing carbon signals increases emissions (67.8→83.0 $gCO_2$/req). Removing queue level control degrades both latency and cost (p95 0.88→1.05 s; $0.00189→$0.00215/req). Disabling model

aware routing increases cost ($0.00189→$0.00206/req). These results indicate that carbon aware routing, queue-level slack tracking, and model right sizing each contribute to the overall multi objective gains.

**Table 2: CEDAR design component analysis**

| Variant | p95(s) | $/req | $gCO_2$/req |
|---|---|---|---|
| Full CEDAR | 0.88 | 0.00189 | 67.8 |
| No Carbon Signal | 0.86 | 0.00192 | 83.0 |
| No Queue-Level Control | 1.05 | 0.00215 | 74.6 |
| No Model-Aware Routing | 0.90 | 0.00206 | 69.1 |

## 5 Conclusion

CEDAR demonstrates that queue-level, carbon aware routing can jointly optimise latency, cost, and marginal emissions for agentic LLM inference, achieving 26% cost and 27% carbon reduction while preserving competitive tail latency. By elevating control to the queue abstraction, CEDAR exposes decision relevant slack invisible to per-request or instance level routing, enabling practical sustainability gains without unacceptable QoS degradation. Future work includes: (i) real deployment on a live vLLM cluster to measure end-to-end scheduling overhead and validate the simulation results under production traffic; (ii) evaluation against real LLM traces (e.g., Azure conversation datasets and publicly available LLM serving logs) to complement the synthetic workload; (iii) stabilising SAC training under non-stationary demand [1, 9]; (iv) evaluating long-horizon workloads (24 h–7 d) [7]; (v) extending spatial routing with temporal deferral [5, 24]; and (vi) exploring deployment at scale with spot capacity and multi-tenant fleets.

## Acknowledgements

This work is supported, in part, by EPSRC and DSIT under grants: EP/X040518/1, EP/Y037421/1, and EP/Y019229/1.

## References

[1] Joshua Achiam. [n. d.]. Spinning Up in Deep Reinforcement Learning. https://spinningup.openai.com.

[2] Bilge Acun, Benjamin C. Lee, Fiodar Kazhamiaka, Kiwan Maeng, Udit Gupta, Carole-Jean Wu, and David Brooks. 2024. On the Limitations of Carbon-Aware Temporal and Spatial Workload Shifting in the Cloud. In *Proceedings of the 19th European Conference on Computer Systems (EuroSys '24)*. Athens, Greece, 112–128. doi:10.1145/3627703.3629569

[3] Andrew A. Chien, Liuzixuan Lin, Hai Nguyen, Varsha Rao, Tristan Sharma, and Rajini Wijayawardana. 2024. The Sunk Carbon Fallacy: Rethinking Carbon Footprint Metrics for Effective Carbon-Aware Scheduling. In *Proceedings of the 2024 ACM Symposium on Cloud Computing (SoCC '24)*. Redmond, WA, USA, 103–119. doi:10.1145/3698038.3698517

[4] Aditya Deshpande, Young Geun Lee, and Vijay Janapa Reddi. 2025. Sustainable Carbon-Aware and Water-Efficient LLM Scheduling in Geo-Distributed Cloud Datacenters. In *Proceedings of the 2025 Great Lakes Symposium on VLSI (GLSVLSI '25)*. Phoenix, AZ, USA, 117–122. doi:10.1145/3716368.3735301

[5] Electricity Maps. 2025. Real-Time Carbon Intensity of Electricity Worldwide. https://www.electricitymaps.com.

[6] Yichao Fu, Siqi Zhu, Runlong Su, Aurick Qiao, Ion Stoica, and Hao Zhang. 2024. Efficient LLM Scheduling by Learning to Rank. In *Advances in Neural Information Processing Systems 37 (NeurIPS 2024)*, Lun-Wei Ku, André Martins, and Vivek Srikumar (Eds.). Curran Associates, Inc., 52301–52322. https://proceedings.neurips.cc/paper_files/paper/2024/file/6c8985579293e0209bdaa4f21bb1d237-Paper-Conference.pdf

[7] Gartner. 2025. Multi-Agent System Inquiries Surge 1,445% from Q1 2024 to Q2 2025. Industry Report.

[8] Kanishk Goel, Jayashree Mohan, Nipun Kwatra, Ravi Shreyas Anupindi, and Ramachandran Ramjee. 2025. Niyama: Breaking the Silos of LLM Inference Serving. *arXiv preprint arXiv:2503.22562* (2025). https://arxiv.org/abs/2503.22562

[9] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. 2018. Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor. In *Proceedings of the 35th International Conference on Machine Learning (ICML '18)*, Vol. 80. PMLR, 1861–1870. https://proceedings.mlr.press/v80/haarnoja18b.html

[10] Jianmin Hu, Minxian Xu, Kejiang Ye, and Chengzhong Xu. 2026. BrownoutServe: SLO-Aware Inference Serving under Bursty Workloads for MoE-Based LLMs. *IEEE Trans. Comput.* 75, 3 (2026), 712–726. doi:10.1109/TC.2026.3655019

[11] International Energy Agency. 2025. *Energy Demand from AI.* Technical Report. IEA, Paris, France. https://www.iea.org/reports/energy-and-ai/energy-demand-from-ai

[12] Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient Memory Management for Large Language Model Serving with PagedAttention. In *Proceedings of the 29th ACM Symposium on Operating Systems Principles (SOSP '23)*. Koblenz, Germany, 611–626. doi:10.1145/3600006.3613165

[13] MarketsandMarkets. 2025. AI Inference Market Size, Share & Growth Analysis, 2025–2030. Market Research Report. https://www.marketsandmarkets.com/Market-Reports/ai-inference-market-189921964.html

[14] MIT Technology Review. 2025. We Did the Math on AI's Energy Footprint. Here's the Story You Haven't Heard. https://www.technologyreview.com/2025/05/20/1116327/ai-energy-usage-climate-footprint-big-tech/.

[15] Preetam Patil Umamaheswari Devi Felix George Pratibha Moogi Moonmoon Mohanty, Gautham Bolar and Parimal Parag. 2025. Deferred Prefill for Throughput Maximization in LLM Inference. In *Proceedings of the 2025 European Conference on Machine Learning Systems (EuroMLSys '25)*. Amsterdam, Netherlands. https://euromlsys.eu/pdf/euromlsys25-39.pdf

[16] Hongqiu Ni, Yizhou Zhou, Zhuohan Li, Zi Ye, Ion Stoica, and Lianmin Zheng. 2024. Astraea: A State-Aware Scheduling Engine for LLM-Powered Agents. *arXiv preprint arXiv:2512.14142* (2024). https://arxiv.org/abs/2512.14142

[17] NVIDIA Corporation. 2024. *TensorRT-LLM: High-Performance Inference Library for Large Language Models.* https://github.com/NVIDIA/TensorRT-LLM

[18] Archit Patke, Dhemath Reddy, Saurabh Jha, Haoran Qiu, Christian Pinto, Shengkun Cui, Chandra Narayanaswami, Zbigniew Kalbarczyk, and Ravishankar Iyer. 2024. Queue Management for SLO-Oriented Large Language Model Serving. In *Proceedings of the 2024 ACM Symposium on Cloud Computing (SoCC '24)*. Redmond, WA, USA, 245–260. doi:10.1145/3698038.3698523

[19] Arman Shehabi, Sarah J. Smith, Eric Masanet, and Jonathan Koomey. 2024. *2024 United States Data Center Energy Usage Report.* Technical Report LBNL-2001552. Lawrence Berkeley National Laboratory, Berkeley, CA, USA. https://escholarship.org/uc/item/32d6m0d1

[20] Yiyang Shen, Li Yue, K.K. Ramakrishnan, and Adam Wierman. 2025. Fair-CO2: Fair Attribution for Cloud Carbon Emissions. In *Proceedings of the 52nd Annual International Symposium on Computer Architecture (ISCA '25)*. 789–802. doi:10.1109/ISCA59077.2025.00068

[21] Jovan Stojkovic, Chaojie Zhang, Íñigo Goiri, Josep Torrellas, and Esha Choukse. 2025. DynamoLLM: Designing LLM Inference Clusters for Performance and Energy Efficiency. In *Proceedings of the 2025 IEEE International Symposium on High-Performance Computer Architecture (HPCA '25)*. Las Vegas, NV, USA, 418–433. doi:10.1109/HPCA61900.2025.00102

[22] Ahmad Massud Tota Khel, Olufemi Isaac Olayiwola, Anas Abubakar Bisu, Hongjian Sun, and Poonam Yadav. 2025. *Towards Carbon-Neutrality for 6G Networks.* White Paper. Communications Hub for Empowering Distributed Cloud Computing Applications and Research (CHEDDAR). https://cheddarhub.org/wp-content/uploads/sites/168/White_Paper__Final_.pdf Version 1.0.

[23] United Nations Regional Information Centre for Western Europe. 2025. Artificial Intelligence: How Much Energy Does AI Use? https://unric.org/en/artificial-intelligence-how-much-energy-does-ai-use/.

[24] WattTime. 2025. WattTime API: Real-Time Grid Carbon Intensity Data. https://www.watttime.org/api-documentation.

[25] David Wong, Bodin Uddamvathanak, and Linh Thi Xuan Phan. 2023. GreenScale: CO2e-Aware Autoscaling for Carbon-Efficient Cloud Computing. *arXiv preprint arXiv:2304.00404* (2023). https://arxiv.org/abs/2304.00404

[26] Lianmin Zheng, Liangsheng Yin, Zhiqiang Xie, Jeff Huang, Chuyue Sun, Cody Hao Yu, Shiyi Cao, Christos Kozyrakis, Ion Stoica, Joseph E. Gonzalez, Clark Barrett, and Ying Sheng. 2024. SGLang: Efficient Execution of Structured Language Model Programs. In *Advances in Neural Information Processing Systems 37 (NeurIPS 2024)*, Lun-Wei Ku, André Martins, and Vivek Srikumar (Eds.). Curran Associates, Inc., 41203–41228. https://proceedings.neurips.cc/paper_files/paper/2024/file/724be4472168f31ba1c9ac630f15dec8-Paper-Conference.pdf