sustainability

MDPI

Article

A Deployment-Aware Framework for Carbon- and Water-Efficient LLM Serving

Julian Hoxha [1,*], Marsela Thanasi-Boçe [2] and Tarek Khalifa [1]

[1] College of Engineering and Technology, American University of the Middle East, Egaila 54200, Kuwait; tarek.khalifa@aum.edu.kw
[2] College of Business Administration, American University of the Middle East, Egaila 54200, Kuwait; marsela.thanasi@aum.edu.kw
* Correspondence: julian.hoxha@aum.edu.kw

Abstract

Inference now dominates the lifecycle footprint of large language models, yet published estimates often use inconsistent boundaries and optimize carbon while ignoring water. We present a provider-agnostic framework that unifies scope-transparent measurement with time-resolved, SLO-aware orchestration and jointly optimizes carbon and consumptive water. Measurement reports daily medians at a comprehensive serving boundary that includes accelerators, host CPU/DRAM, provisioned idle, and PUE uplift, and provides accelerator-only whiskers for reconciliation. Optimization uses a mixed-integer linear program solved over five-minute windows; it selects region, batch size, and phase-aware hardware for prefill and decode while enforcing $p95$ TTFT and TPOT as well as capacity constraints. Applied to four representative models, a single SLO-aware policy reduces comprehensive-boundary medians by 57 to 59 percent for energy, 59 to 60 percent for water, and 78 to 80 percent for location-based CO_2, with SLOs met in every window. For a day with 500 million queries on GPT-4o, totals fall from 0.344 to 0.145 GWh, 1.196 to 0.490 ML, and 121 to 25 t CO_2 (location-based). The framework offers a deployable template for carbon- and water-aware LLM serving with auditable and scope-transparent reporting.

Keywords: LLM inference; carbon-aware routing; water-aware routing; service-level objectives (SLO); mixed-integer linear programming (MILP); power usage effectiveness (PUE); Water Usage Effectiveness (WUE)

check for updates

Academic Editor: Anna Mazzi

Received: 10 October 2025
Revised: 15 November 2025
Accepted: 18 November 2025
Published: 22 November 2025

Citation: Hoxha, J.; Thanasi-Boçe, M.; Khalifa, T. A Deployment-Aware Framework for Carbon- and Water-Efficient LLM Serving. *Sustainability* **2025**, *17*, 10473. https://doi.org/10.3390/su172310473

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

The deployment of large language models at scale has reshaped the sustainability discussion for artificial intelligence [1,2]. Early work focused almost exclusively on the training phase because one-off runs for frontier models were shown to consume on the order of thousands of megawatt-hours, emit hundreds of tons of CO_2 equivalent, and use large volumes of cooling water for a single training job [3–5]. These numbers rightly drew attention and motivated research on efficient architectures and carbon-aware training [6,7]. However, the landscape has since changed. Public adoption has turned inference into a continuous and interactive service that runs every minute of the day. The cumulative footprint of serving billions of queries now dominates the lifecycle of many deployments and can exceed training by a wide margin [5,8]. Several analyses estimate that inference can account for up to 90% of the lifetime energy and that the annual operating energy of the large-scale service can be tens of times higher than the energy used to train the

model [5,9,10]. A single short prompt appears small when measured in watt-hours, yet at hundreds of millions of prompts per day, the aggregate demand reaches utility-scale electricity and meaningful volumes of water [5]. It follows that sustainability efforts that target training alone are no longer sufficient.

Accurate accounting for inference is challenging, and the literature has only recently converged on methods that match production practice [11]. Early estimates combined accelerator nameplate power, theoretical floating-point operations per second (FLOPs), and assumed token lengths, producing per-prompt energies that differed by an order of magnitude because each study chose different utilization factors and boundaries [5,12,13]. Empirical instrumentation is now emerging through industry studies [1]. A key result from this line of work is that narrow measurement (accelerator-only) misses material contributors such as host central processing unit (CPU) and dynamic random access memory (DRAM), provisioned idle, and facility overhead captured by power usage effectiveness. Google's recent production study reports medians at a comprehensive serving boundary and shows that accelerator-only accounting can underestimate per-prompt energy by a factor of more than two for the same workload [1]. This boundary choice explains much of the spread in earlier per-prompt claims and makes clear that scope-transparent reporting is a prerequisite for credible comparison and for effective optimization.

Water has been even less visible in model reporting despite its central role in data center operations [3,5,14]. Cooling systems withdraw and consume water on site, and electricity generation for computing carries a significant upstream water intensity [15]. Model cards and environmental summaries often report scope 2 carbon while omitting water entirely [16,17]. The omission matters because the water intensity of electricity and the carbon intensity of the grid vary by region and time, and they are only weakly correlated. A policy that pursues the lowest carbon intensity without regard to water can inadvertently increase total water consumption, for example, by shifting the load to nuclear-heavy or hydro-dominated regions where the liters per kilowatt-hour are higher, while the converse can also occur [3,14]. Recent studies document that carbon and water can move in opposite directions, which means sustainability must be treated as a bi-objective problem rather than a single number to be minimized [14].

Any practical solution must also respect the realities of an interactive service [18]. LLM serving is governed by strict service-level objectives (SLOs) for responsiveness. The time to first token (TTFT) and time per output token (TPOT) directly shape perceived latency and throughput [17]. Production systems cannot delay user requests to wait for a greener grid, nor can they indiscriminately route traffic to distant regions if that violates latency budgets. The optimization space is therefore bounded by performance constraints and by the capacity of the serving fleet. A sustainability framework that ignores these constraints does not translate into production.

Our focus is on interactive LLM serving rather than generic datacenter load shifting. LLM inference has a two-phase pipeline (prefill then decode), user-visible quality of service expressed as a 95th percentile ($p95$) TTFT/TPOT latency targets, throughput that depends jointly on tokens and batch size with distinct phase bottlenecks, and a skewed request-length mix. These properties determine both the comprehensive serving boundary we measure (accelerators, host CPU/DRAM, provisioned idle, and power usage effectiveness) and the optimization levers modeled in Section 3 (SLO-gated routing, batch right-sizing, token-length directives, and phase-aware placement), distinguishing this setting from general-purpose datacenter scheduling that typically optimizes long-horizon averages without per-window $p95$ guarantees.

This paper proposes a deployment-aware framework that unifies measurement and control. The approach is provider-agnostic, scope-transparent by construction, and explic-

itly bi-objective in terms of carbon and water. Measurement follows production practice and reports daily medians at a comprehensive serving boundary that includes active accelerators, host CPU and DRAM, provisioned idle, with an uplift for power usage effectiveness (PUE), the ratio of total facility energy to IT equipment energy that captures data-center overheads and energy efficiency [1]. Consumptive water is computed as the sum of on-site and source impacts: $W = \text{PUE} \cdot \text{WUE}_{\text{site}} + \text{EWIF}$ [3]. Water Usage Effectiveness (WUE_{site}) captures direct, on-site water use and is scaled by PUE to reflect facility-level consumption. EWIF (Electricity–Water Intensity Factor) accounts for off-site water used in electricity generation, which varies by grid mix and region. Carbon is location-based by default with a market-based sensitivity when portfolio factors are disclosed. Optimization casts serving orchestration as a mixed-integer linear program solved over 288 five-minute windows per day. For each prompt profile, the solver chooses the region, batch size, and a phase-aware hardware assignment that separates prefill and decode. We impose 95th-percentile ($p95$) latency constraints on both TTFT and TPOT, ensuring tail-latency SLOs rather than just averages. Capacity constraints derive from measured tables of tokens per second. Token-length directives, server-side instructions such as "be brief" or "$\leq N$ words" that deliberately shorten the model's response are modeled as controllable reductions in decoding work.

This work combines four elements that, to our knowledge, have not been brought together for LLM serving: (i) A dual-objective, SLO-constrained MILP that co-optimizes carbon and consumptive water under realistic tail-latency and capacity gates—the objective is Equation (17), and feasibility is enforced by conservation (18), capacity with replicas (19) (linked to activation and bounds in (4) and (6)), and the $p95$ latency guard (20); (ii) a scope-reconciliation layer that translates accelerator-only measurements to a comprehensive serving boundary via the IT and facility lifts (13) and (14), reproducing the production narrow/comprehensive median ratio ($\approx$0.417) for comparable workloads [1]; (iii) phase-split assignments, prefill and decode, as first-class decisions with per-phase coefficients (10) and (11) and per-assignment energy (12); and (iv) integration of token-length directives as controllable variables (the multiplier α_d in (9)), so semantic concision can be co-optimized with routing, batching, and placement while respecting $p95$ SLOs. We analyze routing-only and joint routing + batch + token frontiers to make the carbon–water geometry explicit, which tells operators when improvements move both metrics together and when trade-offs must be managed. We also gate embodied impacts once per day per hardware class via (21), so life-cycle terms are optimized alongside operational metrics. Empirically, across four models, the resulting SLO-aware policy reduces comprehensive-boundary medians by roughly 57–59% (energy), 59–60% (consumptive water), and 78–80% (location-based CO_2) while meeting all per-window $p95$ SLOs (see Section 4.1).

The remainder of the paper is organized as follows. Section 2 surveys the literature and motivates comprehensive boundaries, water-aware metrics, and SLO-respecting orchestration. Section 3 details the methodology, including the functional unit, scope definitions, impact accounting, time resolution, decision variables, constraints, and parameterization. Section 4 presents the results, reporting comprehensive-boundary medians and daily totals, scope reconciliation between accelerator-only and comprehensive views, and carbon–water movement at a fixed quality of service with routing, batch, token, and phase-split analyses. Section 5 discusses implications, deployment guidance, and limitations and outlines directions for future work. Section 6 concludes the paper.

2. Optimizing the Environmental Footprint of LLM Inference: A Literature Review

2.1. From Training to Inference: Why the Burden Has Shifted

Early work on AI sustainability emphasized training, where single jobs for frontier models consumed thousands of MWh and emitted hundreds of tons of CO_2e while using significant cooling water [3–7]. As generative systems moved into daily use, the continuous nature of serving billions of prompts has become the dominant lifecycle driver for many deployments [5,8–10,19–21]. Per-request impacts that seem small in isolation compound at scale: a single ChatGPT-style prompt has been estimated at several grams of CO_2e, far above a web search [22], and daily volumes in the hundreds of millions translate to utility-scale electricity and meaningful water withdrawals [5]. These observations motivate methods that measure inference accurately and optimize it under interactive quality constraints.

2.2. Full Stack per Prompt Accounting: Energy, Carbon, Water, and Embodied Impacts

Production studies recommend a comprehensive serving boundary that includes the active accelerator, host CPU and DRAM, provisioned idle, and facility overhead via PUE [1,23,24]. Under this boundary, "accelerator only" views can undercount by more than a factor of two. The Google study reports a narrow-to-comprehensive median ratio near 0.417 for similar workloads [1]. Carbon per prompt should be reported as both location-based (grid average) and, when portfolios are disclosed, market-based [25], and it should include amortized embodied impacts from hardware manufacturing where possible [26–28]. Water deserves first-class treatment: consumptive site cooling scaled by PUE plus off-site electricity–water intensity (EWIF) yields a scope-consistent measure of liters per kWh that varies widely by region and generation mix [3,29–33]. Because carbon intensity (CIF) and EWIF are only weakly correlated, any realistic framework should measure both and co-optimize them [3,14].

2.3. Measurement Boundaries: Why Scope Transparency Matters

Inconsistent boundaries explain much of the order-of-magnitude spread in earlier per-prompt estimates. Studies that measured only chip power (sometimes at unrealistic utilization) omitted host, idle, and cooling overheads and thus overstated efficiency [5,34,35]. Converging practice now recommends a comprehensive inference boundary under operator control, with explicit exclusions outside that boundary (e.g., end user devices or wide-area network transit) and with scope-transparent reporting so results are comparable across systems [1,24]. The $\approx$0.417 narrow-to-comprehensive ratio provides a defensible translation when only accelerator-level numbers are available and site factors and mixes are held constant [1]. Our methods section adopts this convention and reports daily medians to handle skewed mixes.

2.4. Real-Time Orchestration: Carbon- and Water-Aware Routing Under SLOs

With consistent measurement in place, the next step is optimization. Carbon-aware request routing has been shown to cut serving emissions substantially without violating latency when traffic is steered to cleaner grids in space and time [10,36–41]. Projections suggest even larger reductions as grids decarbonize and capacity headroom grows [10]. Emerging frameworks treat scheduling as a multi-objective problem, co-optimizing carbon, water, and cost while enforcing latency SLOs and site capacity constraints; practical solutions range from MILPs to learning-based controllers [9,42–45]. Our Σ-Scale loop (MILPs) follows this line and enforces $p95$ TTFT and TPOT together with per-batch throughput limits.

2.5. Phase-Aware Hardware Scheduling (Prefill vs. Decode)

Transformer serving has two phases with different bottlenecks. Prefill is parallel and computation-heavy. Decode is sequential and often memory-bound [46,47]. Phase-aware scheduling assigns prefill and decode to different hardware or configurations to improve both responsiveness and efficiency [1,48–50]. Speculative decoding, KV-cache reuse, and dynamic token pruning further reduce decode work [51–55]. Our formulation exposes prefill and decode decisions and allows second-life hardware to serve decode when SLOs permit [22,56].

2.6. Semantic-Level Interventions

Beyond system levers, generation can be made more efficient with semantic directives that reduce unnecessary output length or avoid high-compute behaviors while preserving usefulness. SPROUT is a concise-generation approach that reduces generated tokens to reduce energy with minimal quality loss. We use the same idea as a controllable directive in our optimization [57]. Related work on eco-adaptive services explores small, user-acceptable adjustments during high-carbon periods to achieve large operational reductions [21,58]. These methods complement, rather than replace, hardware and orchestration optimizations.

2.7. Lifecycle and Circular Economy Strategies

Holistic sustainability includes upstream manufacturing and end of life. Hardware production for AI accelerators is energy- and material-intensive; embodied emissions can dominate in low-carbon operational settings [59,60]. Scenario analyses warn of rapid growth in AI-related e-waste and point to device life extension, refurbishment, parts harvesting, and improved recycling as the most effective mitigations [61]. Second-life deployment and modular upgrades align with extended producer responsibility and reduce both embodied and disposal impacts [1,22,61].

2.8. Toward a Unified, Deployment-Aware Framework

Inference now dominates the energy, carbon, and water impacts of modern LLM services [5,8–10,19–21]. Recent work recommends scope-transparent, full-stack per-prompt accounting so results are comparable and operationally useful [1,23–25]. Studies also show that optimization should treat carbon and water together while preserving interactive SLOs [3,9,10,14,36–39]. Lifecycle analyses motivate attention to embodied impacts and end of life [22,59–61].

Prior schedulers mainly shift the load by region or hour to reduce carbon and often rely on averages or deferral [9,10]. We instead co-optimize location-based CO_2 and consumptive water using a site+source definition that sums facility cooling and grid water intensity. Equation (3) captures this as $\mathrm{PUE} \times \mathrm{WUE}_{\mathrm{site}}$ plus EWIF. Carbon intensity and electricity–water intensity are weakly correlated [3,14], so optimizing a single metric can worsen the other. Our results show concurrent reductions at fixed quality of service. Feasibility is enforced for every five-minute window using explicit $p95$ TTFT, $p95$ TPOT, and capacity constraints in one MILP. These appear in Equations (18)–(20) and Algorithm 1. Reporting is scope-transparent at the comprehensive serving boundary and includes a narrow to comprehensive translationthat follows production guidance [1]. We expose phase-aware and semantic levers that reduce watt-hours per prompt while maintaining interactive quality [5]. Together, these elements provide a deployment-aware path for operators to measure, compare, and materially lower the environmental footprint of LLM inference.

Many controllers target cost or carbon proxies and do not model water or interactive tail latency [9]. We treat water as a first-class objective with the same site + source measure as above and we enforce $p95$ SLOs and capacity directly through the constraints in Equations (18)–(20). We summarize outcomes as daily medians at the comprehensive boundary and reconcile scopes for comparison. LLM inference systems improve throughput and tails with batching and with prefill and decode splitting [18,46,47]. Most do not report environmental effects jointly. Our solver unifies routing, batching, token-length guidance, and phase-aware placement as decision variables under dual objectives and the same $p95$ SLO guard. Appendix A Figures A1–A8 provide ablations with uncertainty. Scope transparency follows current production practice [1].

We compute daily medians over 288 five-minute windows. We report 95% confidence intervals and interquartile ranges and we show by-region distributions as violins in Appendix A Figures A1–A8. Cross-model results are consistent at the comprehensive boundary. The next section formalizes the site+source water model, the conservation, capacity, and $p95$ latency constraints.

3. Methodology

Our goal is to convert heterogeneous measurements of LLM inference into a portable and auditable workflow. The workflow performs three things. First, it reports scope-aligned, commensurable per-prompt medians under a comprehensive serving boundary that matches production practice. When needed for comparison, it also reports a narrower accelerator-only boundary. Second, it enforces real SLOs using explicit capacity and latency constraints derived from public tokens-per-second and latency quantiles. Third, it co-optimizes energy, LB greenhouse-gas emissions, consumptive water, and amortized embodied impacts with a mixed-integer linear program (MILP).

We adopt the comprehensive boundary used in a first-party production study for Gemini Apps. That study measures a median text prompt at 0.24 Wh (comprehensive) and an accelerator-only median of ≈0.10 Wh when active accelerator, host CPU/DRAM, provisioned idle, and facility overhead (PUE) are aggregated [1]. The ratio implies an uplift of ≈2.4× due to boundary choice alone (comprehensive vs. accelerator-only). We follow that full-stack paradigm to keep results comparable and to avoid the systematic undercounting highlighted by the production study [1].

3.1. Functional Unit and System Boundaries

The functional unit is one served prompt. We compute energy per prompt (Wh) at two nested boundaries so results remain comparable to both "narrow" and "comprehensive" studies [1]. The accelerator-only boundary counts only the active accelerator energy attributed to the prompt. The comprehensive serving boundary also includes host CPU/DRAM, provisioned idle capacity, and facility overhead via PUE. We use the comprehensive boundary as a default because it aligns with production telemetry and yields scope-consistent, methodologically comparable per-prompt medians across models and regions [1].

Let E^{active} denote accelerator-only energy per prompt (GPU/TPU compute in prefill and decode). It excludes host CPU/DRAM, provisioned-but-idle energy, and facility overhead captured by PUE. Let E^{IT} be the IT-side energy per prompt (accelerators + host + idle). Let E^{host} and E^{idle} be the non-accelerator components. We define $\kappa_{\text{host+idle}}$ as the ratio of total IT-side energy to the accelerator's active energy

$$\kappa_{\text{host+idle}} = \frac{E^{\text{IT}}}{E^{\text{active}}} = \frac{E^{\text{active}} + E^{\text{host}} + E^{\text{idle}}}{E^{\text{active}}}. \quad (1)$$

To match the production medians ($E^{\text{fac}} \approx 0.24$ Wh and $E^{\text{active}} \approx 0.10$ Wh [1]), we set $E^{\text{IT}} = E^{\text{fac}}/\text{PUE} \approx 0.24/1.09 \approx 0.220$ Wh, which gives $\kappa_{\text{host+idle}} \approx 0.220/0.10 \approx 2.20$. When host/idle shares are undisclosed, we use this aggregated $\kappa_{\text{host+idle}}$ rather than separate point estimates.

Finally, facility energy per prompt is $E^{\text{fac}} = \text{PUE} \times E^{\text{IT}}$. With PUE $\approx$ 1.09, this implies $E^{\text{fac}} \approx$ 0.24 Wh. The facility overhead is $E^{\text{overhead}} = (\text{PUE} - 1) \times E^{\text{IT}} \approx 0.02$ Wh.

3.2. Impact Accounting

We follow the location-based (LB) approach for operational emissions, as recommended for data-center Scope-2 reporting [1]. Facility energy per prompt E^{fac} (kWh prompt^{-1}) is multiplied by the regional LB grid carbon intensity CIF_r^{LB} (kg CO_2 kWh^{-1}) to yield:

$$g\text{CO}_2^{\text{LB}} = E^{\text{fac}} \times \text{CIF}_r^{\text{LB}} \times 10^3. \tag{2}$$

LB attributes emissions to the electricity drawn at the point of consumption. This supports statistical comparability across operators and regions and avoids year-to-year swings driven by portfolio accounting. As a sensitivity, we also report the provider's prior-year market-based factor EF_r^{MB} when disclosed [25]. We reserve $g\ CO_2e$ for MB factors and embodied or lifecycle terms.

For water, we follow the scope-consistent "site + source" rule advocated in the AI water-footprint literature [3]. The consumptive site water (cooling, at the facility) is proportional to facility energy via the site Water Usage Effectiveness $\text{WUE}_r^{\text{site}}$ (L kWh^{-1}), while source water accounts for the electricity-generation water intensity of the regional grid via the Electricity–Water Intensity Factor $\text{EWIF}_r^{\text{source}}$ (L kWh^{-1}), separating site water (cooling; WUE Category 2) from source water associated with electricity generation (EWIF), as recommended in the water-footprint literature [3]. Using $E^{\text{IT}}[kWh/prompt]$ for the electrical work of the IT load, the per-prompt consumptive water is

$$mL_r = [E^{\text{IT}} \times (\text{PUE}_r \times \text{WUE}_r^{\text{site}}) + E^{\text{IT}} \times \text{EWIF}_r^{\text{source}}] \times 10^3. \tag{3}$$

This form makes it explicit that site water scales with the facility energy (hence the PUE multiplier), while source water scales with the electricity used to power the IT equipment and is governed by the grid mix. Analysis emphasizes both the need to include scope 2 water and the empirical weak correlation between carbon and water intensities [3], motivating dual-metric reporting and optimization. The macro background on facility-level PUE and WUE levels across the U.S. fleet is taken from the 2024 U.S. Data Center Energy Usage Report, which shows hyperscale-weighted averages below ∼1.4 PUE and site WUE around 0.32–0.40 L kWh^{-1} in 2023 (with a likely rise as liquid cooling penetrates), giving realistic bounds for sensitivity tests [62].

We summarize per-prompt footprints (Wh, mL, gCO_2e) as daily medians by profile and season, mirroring the production study's choice of medians as the right statistic for skewed mixes; that study reports comprehensive-boundary medians near 0.24 Wh, 0.03 gCO_2e, and 0.26 mL for the median text prompt and documents a ∼2.4× uplift from accelerator-only to comprehensive accounting. Our framework reproduces this uplift when we apply $\kappa_{\text{host+idle}}$ and PUE to chip-only numbers, providing scope-transparent comparability [1].

We include embodied impacts and e-waste on a serving-time basis. For accelerator class h with embodied carbon C_h^{emb} (kg CO_2e board^{-1}), board mass M_h^{board} (g), and assumed service lifetime L_h (days), we compute daily rates C_h^{emb}/L_h (kg CO_2e day^{-1}) and M_h^{board}/L_h (g day^{-1}). When a class is activated (binary variable in the solver), we allocate its daily embodied rates to the prompts it serves that day, adding an amortized per-prompt

embodied term that brings the hardware lifecycle into scope. This aligns with the lifecycle scope of the production study [1,22].

3.3. Time Resolution, Traffic Mix, and SLOs

We partition each day into $|T| = 288$ decision windows of five minutes with budget a budget of $\Delta t = 300\,\text{s}$. The five-minute cadence balances temporal fidelity with tractable optimization: it is fine enough to reflect diurnal variation in arrivals and region-level grid factors while remaining small enough to keep the MILP size manageable. It also aligns with the resolution of the $p95$ throughput and latency tables we use in the capacity and SLO constraints. Coarser windows (for example, 15 min) yield similar daily medians but reduce control authority, whereas finer windows (for example, 1 min) materially increase problem size with limited additional benefit; a full sensitivity analysis is left to future work [5,25]. Demand is represented by three prompt profiles $p \in \{short, medium, long\}$ with token pairs $(T_p^{\text{in}}, T_p^{\text{out}}) = (100, 300)$, $(1000, 1000)$, and $(10{,}000, 15{,}000)$. Tokens are the subword units processed by the model. We denote per-profile input/output counts by $(T_p^{\text{in}}, T_p^{\text{out}})$ and apply a directive multiplier α_d to shorten the decode when enabled.

Unless noted, daily arrivals are $N_{\text{day}} = 500\,\text{M}$ prompts with a 70/25/5% mix. We use 70/25/5 as a production-style skew that mirrors telemetry-based reporting in which daily medians are computed per profile and then combined, which keeps summaries comparable across models and regions. The mixture is not a modeling assumption: the framework always reports per-profile medians, and operators may substitute any traffic mix (for example, uniform 1/3–1/3–1/3 or a long-heavy mix) without changing the method. Window-level arrivals $Q_{t,p}$ are provided or constructed from N_{day}, the mix π_p, and diurnal weights $f(t)$ as $Q_{t,p}$=round$\big(N_{\text{day}} \pi_p f(t) / \sum_{t'} f(t')\big)$.

The router chooses non-negative integer assignments $x_{t,r,h,b,\varphi,p,d}$ by region r, hardware class h, batch size b, phase $\varphi \in \{prefill, decode\}$, and directive $d \in \{default, brief\}$. Prefill and decode are two legs of the same request. Conservation holds per window and profile. Capacity in a five-minute window is enforced with $\text{TPS}_{h,b}(q)$ at $q = 95$ and the budget Δt.

Latency SLOs are profile-specific $p95$ targets. We use the published per-hardware and per-batch quantiles for TTFT and TPOT together with the associated TPS rows. The formulation enforces the $p95$ tail-latency gate and uses the same $p95$ throughputs in the capacity constraint, so feasibility reflects serving variability. Unless stated otherwise, reported summaries are daily medians by profile and a 70/25/5 mix of those medians, consistent with production practice [1].

3.4. Decision Variables, Objective, and Constraints

3.4.1. Setup: Decisions, Replicas, and Parameters

The decision variables are the assignment of prompts $x_{t,r,h,b,\varphi,p,d}$ in each window (non-negative integers), binary activation variables $u_{t,r,h,b} \in \{0, 1\}$ for hardware classes at a site, the chosen batch size b from a small discrete set per hardware, binary indicators $z_{r,h} \in \{0, 1\}$ that switch on embodied amortization when a hardware class is used, and $n_{t,r,h,b} \in \mathbb{Z}_{\geq 0}$, which is the number of replicas provisioned at (r, h, b) in window t. A replica is one schedulable serving lane, for example, a pod or worker bound to the measured hardware and batch, and each replica contributes a fixed amount of safe $p95$ throughput in the window, so increasing n increases capacity linearly while all choices remain gated by the profile-specific $p95$ latency budgets $L_p^\star$.

We include a per-site replica cap $U_{r,h,b} \in \mathbb{Z}_{\geq 0}$, which is the maximum number of serving replicas that can be provisioned at site r, with hardware h, and with batch b in a window; a "big-M" upper bound $M_{h,b,\varphi,p,d}$ in prompts per window is used to link activation

to assignments and an SLO-gating constant $M'_{h,b,p,d}$ in seconds that is used only in the optional indicator form of the $p95$ latency constraint. The activation–assignment linkage is

$$x_{t,r,h,b,\varphi,p,d} \;\le\; M_{h,b,\varphi,p,d}\, u_{t,r,h,b}, \tag{4}$$

so that $u_{t,r,h,b} = 0$ implies $x_{t,r,h,b,\varphi,p,d} = 0$. The optional indicator SLO form is

$$\mathrm{TTFT}_{h,b}(95) + \alpha_d\, T_p^{\mathrm{out}}\, \mathrm{TPOT}_{h,b}(95) \;\le\; L_p^{\star} \;+\; M'_{h,b,p,d}\,\big(1 - y_{t,r,h,b,p,d}\big), \tag{5}$$

which turns off the SLO when the pair is not used ($y_{t,r,h,b,p,d} = 0$) and enforces it when used ($y_{t,r,h,b,p,d} = 1$). Replica counts are bounded by

$$0 \;\le\; n_{t,r,h,b} \;\le\; U_{r,h,b}\, u_{t,r,h,b}, \tag{6}$$

and scale the right-hand side of the capacity constraint (Equation (19)) together with Δt and $\mathrm{TPS}_{95}(h,b)$.

$U_{r,h,b}$ has units of lanes per window and acts as a hard capacity limit arising from inventory, quotas, or operator policy. $M_{h,b,\varphi,p,d}$ has units of prompts per window and should be chosen tightly; a stable choice that respects arrivals is

$$M_{h,b,\varphi,p,d} \;=\; \max_{t}\, Q_{t,p}, \tag{7}$$

since no window can assign more prompts of profile p than arrive. $M'_{h,b,p,d}$ has units of seconds and is set large enough that (5) is trivially satisfied when $y = 0$; a principled tight choice is

$$M'_{h,b,p,d} \;\ge\; \max_{(h,b)} \Big[\mathrm{TTFT}_{h,b}(95) + \alpha_d\, T_p^{\mathrm{out}}\, \mathrm{TPOT}_{h,b}(95) - L_p^{\star}\Big]_{+}, \tag{8}$$

the worst-case positive slack across allowed (h,b). If the solver supports indicator constraints of the form $y_{t,r,h,b,p,d} = 1$ imply the SLO inequality, those can be used instead and M' can be omitted.

3.4.2. Token Accounting (Profiles, Directives, Per-Phase Tokens)

Each prompt of profile p consists of T_p^{in} input tokens in the prefill phase and T_p^{out} output tokens in the decode phase. Decode tokens are scaled by the directive factor $\alpha_d \in \{1, 0.80, 0.75, 0.70\}$ depending on whether the output is shortened. We set α_d on these values to bracket light, medium, and strong concision settings commonly used in practice, corresponding to approximately 20–30% reductions in output tokens. These values are consistent with concise-generation directives and eco-adaptive quality settings reported in prior work, and they can be replaced by empirically measured multipliers on an operator's traffic [5,21,22]. The number of tokens served by a routed block is therefore given by

$$\tau_{r,h,b,\varphi,p,d} = \begin{cases} T_p^{\mathrm{in}}, & \varphi = \mathit{prefill}, \\ \alpha_d\, T_p^{\mathrm{out}}, & \varphi = \mathit{decode}. \end{cases} \tag{9}$$

Throughput is modeled as $\mathrm{TPS}_{h,b}(q)$, the number of tokens per second at quantile q for hardware class h under batch size b. Latency SLOs are profile-specific constants $L_p^{\star}$, for example, 0.9 s for "short" prompts. We use per-batch quantiles $\mathrm{TPS}_{h,b}(q)$ and $\mathrm{TTFT}_{h,b}(q)$ at $q = 95$.

3.4.3. Per-Assignment Coefficients

Decode energy per output token:

$$e^{\mathrm{Wh/token}}(h, decode, b; q) = \bar{P}^{\mathrm{acc}}_{h,b}(decode)\,\frac{1}{\mathrm{TPS}_{h,b}(q)} \times \frac{1}{3600}. \tag{10}$$

where $\bar{P}^{\mathrm{acc}}_{h,b}$ is the accelerator-only average power (W). Before the first token appears, the system spends $\mathrm{TTFT}_{h,b}(q)$ seconds on the prefill. Spreading that energy over the T^{in}_{p} input tokens for profile p makes the prefill term linear and comparable. The prefill energy per token is defined as

$$e^{\mathrm{Wh/token}}(h, prefill, b; q) = \bar{P}^{\mathrm{acc}}_{h,b}(prefill)\,\frac{\mathrm{TTFT}_{h,b}(q)}{T^{\mathrm{in}}_{p}} \times \frac{1}{3600}. \tag{11}$$

Accelerator-only energy per assignment in each 5-minute window:

$$E^{\mathrm{acc}}_{t,r,h,b,\varphi,p,d} = x_{t,r,h,b,\varphi,p,d} \times e^{\mathrm{Wh/token}}(h, \varphi, b; q) \times \tau_{r,h,b,\varphi,p,d}. \tag{12}$$

We then lift the IT energy with the production-observed host + idle scaler $\kappa_{\mathrm{host+idle}}$ (cf. Equation (1)) and the facility (comprehensive) energy via the site's PUE:

$$E^{\mathrm{IT}}_{t,r,h,b,\varphi,p,d} = \kappa_{\mathrm{host+idle}} \times E^{\mathrm{acc}}_{t,r,h,b,\varphi,p,d}, \tag{13}$$

$$E^{\mathrm{fac}}_{t,r,h,b,\varphi,p,d} = \mathrm{PUE}_r \times E^{\mathrm{IT}}_{t,r,h,b,\varphi,p,d}. \tag{14}$$

Consumptive water is obtained by applying the site-level cooling term and source-site electricity water to the IT energy. Operational emissions use the facility energy times either the provided market-based factor or the location-based grid factor, with LB as the default. The water and carbon per assignment are defined as

$$\mathrm{mL}_{t,r,h,b,\varphi,p,d} = E^{\mathrm{IT}}_{t,r,h,b,\varphi,p,d} \times \left(\mathrm{PUE}_r \times \mathrm{WUE}^{\mathrm{site}}_r\right) + E^{\mathrm{IT}}_{t,r,h,b,\varphi,p,d} \times \mathrm{EWIF}^{\mathrm{source}}_r. \tag{15}$$

$$g\mathrm{CO}^{\mathrm{LB}}_{2,t,r,h,b,\varphi,p,d} = E^{\mathrm{fac}}_{t,r,h,b,\varphi,p,d} \times \mathrm{CIF}_r \times 10^3. \tag{16}$$

Equations (10)–(16) define the per-assignment terms that the objective will sum and later aggregate/normalize.

3.4.4. Objective (What the Σ-Scale Solver Minimizes)

We first aggregate the daily medians per profile, use a mix of 70/25/5, and define the objective as a weighted-sum scalarization with non-negative policy weights:

$$\min_{x,u,z,n} \sum_{t,r,h,b,\varphi,p,d} \left(\alpha\, E^{\mathrm{IT}}_{t,r,h,b,\varphi,p,d} + \beta\, mL_{t,r,h,b,\varphi,p,d} + \lambda\, \mathrm{gCO2}^{\mathrm{LB}}_{t,r,h,b,\varphi,p,d}\right) + \delta\, \mathrm{CO2}^{\mathrm{emb}}_{\mathrm{day}} + \varepsilon\, W^{\mathrm{ewaste}}_{\mathrm{day}}, \tag{17}$$

where the minimization is over the decision variables $x_{t,r,h,b,\varphi,p,d}$, $u_{t,r,h,b}$, $z_{r,h}$. The weights $\alpha, \beta, \lambda, \delta, \varepsilon \geq 0$ are chosen by the operator/policy. Equation (17) mirrors multi-impact scoring used in recent infrastructure-aware benchmarking (energy, water, carbon) while adding embodied and e-waste terms for life-cycle completeness [5]. The minimization of (17) is subject to the following constraints.

3.4.5. Feasibility Constraints

Prefill and decode are two legs of the same request. Counts must match and meet demand. The first requirement is a linear conservation constraint, which ensures that the

number of prefilled requests equals the decoded requests and matches the total incoming demand:

$$\sum_{r,h,b,d} x_{t,r,h,b,prefill,p,d} = \sum_{r,h,b,d} x_{t,r,h,b,decode,p,d} = Q_{t,p}. \tag{18}$$

Since we cannot push more tokens through a node than it can handle at the selected batch size, we impose a second capacity constraint. This limits the total load based on the benchmarked throughput quantiles (per hardware/batch):

$$\sum_{p,d} \left(\frac{x_{t,r,h,b,prefill,p,d}\, T_p^{\text{in}}}{\text{TPS}_{h,b}(q)} + \frac{x_{t,r,h,b,decode,p,d}\, \alpha_d\, T_p^{\text{out}}}{\text{TPS}_{h,b}(q)} \right) \le \Delta t\, n_{t,r,h,b} \qquad \forall\, t, r, h, b, \tag{19}$$

where $\Delta t = 300$ s is the five-minute budget and $q = 95$ unless noted. We enforce $0 \le n_{t,r,h,b} \le U_{r,h,b}\, u_{t,r,h,b}$ and $z_{r,h} \ge u_{t,r,h,b}$ for all (t, r, h, b), and gate assignments by $x_{t,r,h,b,\phi,p,d} \le M_{h,b,\phi,p,d}\, u_{t,r,h,b}$ for all (t, r, h, b, ϕ, p, d).

The third requirement is an SLO gating constraint that controls tail latency. This bounds the 95th percentile response time for each profile to its specified limit $L_p^\star$:

$$\text{TTFT}_{h,b}(95) + \alpha_d\, T_p^{\text{out}} \times \text{TPOT}_{h,b}(95) \le L_p^\star, \qquad \text{TPOT}_{h,b}(95) = \frac{1}{\text{TPS}_{h,b}(95)}. \tag{20}$$

Equation (18) ensures mandatory service levels, (19) caps the safe volume servable in 5 min with n replicas, and (20) enforces tail latency limits. All three define the feasible set for the objective.

3.4.6. Embodiment and Daily Coupling (Once/Day per Hardware Class)

The embodied CO_2 and e-waste are amortized as daily charges that activate once per hardware class used:

$$\text{CO}_{2,\text{day}}^{\text{emb}} = \sum_{r,h} z_{r,h}\, \frac{C_h^{\text{emb}}}{L_h}, \qquad W_{\text{day}}^{\text{ewaste}} = \sum_{r,h} z_{r,h}\, \frac{M_h^{\text{board}}}{L_h}. \tag{21}$$

Equation (21) treats hardware life-cycle impacts as a once-per-day charge that is activated exactly when a hardware class is used (via $z_{r,h}$), then amortized uniformly across that day's served load. This "use-gated daily amortization" mirrors sustainability reporting practice for server fleets and avoids per-request double-counting while keeping the decision to activate additional classes visible to the optimizer [1,16,62].

3.5. Framework and Algorithmic Handoff

Given the per-assignment coefficients in Section 3.4 (energy lifts and impacts, Equations (12)–(21)) and the feasibility constraints (conservation (18), capacity with replicas (19) and bounds (6), and the $p95$ latency gate (20)), the MILP is defined by minimizing the scalarized objective (17) over one day at five-minute resolution. Aggregating over t and taking daily medians yields the reported statistics, with LB factors as the default and MB factors as a sensitivity [1,3,25,62]. Algorithm 1 solves this problem and returns the optimal routing and scale-out decisions $(x^\star, n^\star, u^\star, z^\star)$. Algorithm 2 aggregates $x^\star$ into per-window, per-profile totals $(E_{t,p}^{IT},\ w_{t,p},\ c_{t,p}^{LB})$, normalizes by arrivals $Q_{t,p}$ to obtain per-prompt series, and summarizes each day by profile medians and the 70/25/5 mixed median. Unless noted, Pareto fronts and lever ablations are reported at fixed $p95$ latency SLOs $L_p^\star$.

Algorithm 1 Σ-Scale (daily-coupled MILP; $p95$ SLOs, replicas, daily binaries).

Inputs: Sets: regions R, hardware classes H, batches B, phases $\Phi = \{\text{prefill}, \text{decode}\}$, profiles P, directives $D = \{\text{default}, \text{brief}\}$, windows $T = \{1, \ldots, 288\}$; window budget $\Delta t = 300$ s.

Inputs: Arrivals $Q_{t,p}$ per window and profile, either provided or constructed from N_{day}, mix π_p, and diurnal weights $f(t)$: $Q_{t,p} = \text{round}\big(N_{\text{day}} \pi_p f(t) / \sum_{t'} f(t')\big)$.

Inputs: $p95$ quantile tables: $\text{TPS}_{h,b}(95)$, $\text{TTFT}_{h,b}(95)$, $\text{TPOT}_{h,b}(95) = 1/\text{TPS}_{h,b}(95)$.

Inputs: Site rows per r: PUE_r, $\text{WUE}_{\text{site},r}$, $\text{EWIF}_r^{\text{source}}$, CIF_r^{LB} (optionally EF_r^{MB}).

Inputs: Lift and SLOs: $\kappa_{\text{host+idle}} \geq 1$, $L_p^\star$; weights $\alpha, \beta, \lambda, \delta, \varepsilon \geq 0$.

Inputs: Embodiment: C_h^{emb} ($\text{kg CO}_2\text{e}$/board), M_h^{board} (kg), lifetime L_h (days).

Inputs: Accelerator-only per-token models $e_{\text{Wh/token}}(h, \varphi, b)$; tokens T_p^{in}, $T_{p,\text{default}}^{\text{out}}$; directive multipliers $\alpha_d \in (0, 1]$; replica caps $U_{r,h,b} \in \mathbb{Z}_{\geq 0}$.

Outputs: Decision variables:

$x_{t,r,h,b,\varphi,p,d} \in \mathbb{Z}_{\geq 0}$ (assignments), $n_{t,r,h,b} \in \mathbb{Z}_{\geq 0}$ (replicas), $u_{t,r,h,b} \in \{0, 1\}$ (activation), $z_{r,h} \in \{0, 1\}$ (daily activation).

Optional: $y_{t,r,h,b,p,d} \in \{0, 1\}$ (usage, for gated SLO).

Per-prompt coefficients for each (r, h, b, φ, p, d) in Wh/prompt:

$$T_{p,d}^{\varphi} \leftarrow \big(T_p^{\text{in}} \text{ if } \varphi = \text{prefill};\ \alpha_d T_{p,\text{default}}^{\text{out}} \text{ if } \varphi = \text{decode}\big)$$

$$E_{h,b,\varphi,p,d}^{\text{acc}} \leftarrow e_{\text{Wh/token}}(h, \varphi, b)\, T_{p,d}^{\varphi}$$

$$E_{h,b,\varphi,p,d}^{\text{IT}} \leftarrow \kappa_{\text{host+idle}}\, E_{h,b,\varphi,p,d}^{\text{acc}}$$

$$E_{r,h,b,\varphi,p,d}^{\text{fac}} \leftarrow \text{PUE}_r\, E_{h,b,\varphi,p,d}^{\text{IT}}$$

$$\text{mL}_{r,h,b,\varphi,p,d} \leftarrow E_{h,b,\varphi,p,d}^{\text{IT}} \big(\text{PUE}_r\, \text{WUE}_{\text{site},r} + \text{EWIF}_r^{\text{source}}\big)$$

$$g\text{CO}_{2,r,h,b,\varphi,p,d}^{\text{LB}} \leftarrow E_{r,h,b,\varphi,p,d}^{\text{fac}}\, \text{CIF}_r^{\text{LB}}$$

Minimize (global daily sum):

$$\alpha \sum_{t,r,h,b,\varphi,p,d} x_{t,r,h,b,\varphi,p,d}\, E_{h,b,\varphi,p,d}^{\text{IT}} + \beta \sum_{t,r,h,b,\varphi,p,d} x_{t,r,h,b,\varphi,p,d}\, \text{mL}_{r,h,b,\varphi,p,d} + \lambda \sum_{t,r,h,b,\varphi,p,d} x_{t,r,h,b,\varphi,p,d}\, g\text{CO}_{2,r,h,b,\varphi,p,d}^{\text{LB}} + \sum_{r,h} z_{r,h} \left(\delta \frac{C_h^{\text{emb}}}{L_h} + \varepsilon \frac{M_h^{\text{board}}}{L_h} \right)$$

Subject to

(i) Conservation per (t, p) and phase link:

$$\sum_{r,h,b,d} x_{t,r,h,b,\text{prefill},p,d} = \sum_{r,h,b,d} x_{t,r,h,b,\text{decode},p,d} = Q_{t,p} \quad \forall t, p.$$

(ii) Capacity with replicas (RHS scaled by n):

$$\sum_{p,d} \left(\frac{x_{t,r,h,b,\text{prefill},p,d}\, T_p^{\text{in}}}{\text{TPS}_{h,b}(95)} + \frac{x_{t,r,h,b,\text{decode},p,d}\, \alpha_d T_{p,\text{default}}^{\text{out}}}{\text{TPS}_{h,b}(95)} \right) \leq \Delta t\, n_{t,r,h,b} \quad \forall t, r, h, b.$$

(iii) Replica bounds and activation:

$$0 \leq n_{t,r,h,b} \leq U_{r,h,b}\, u_{t,r,h,b} \quad \forall t, r, h, b.$$

(iv) Daily coupling (embodiment once/day):

$$z_{r,h} \geq u_{t,r,h,b} \quad \forall t, r, h, b.$$

(v) Link assignments to activation (tight big-M):

$$x_{t,r,h,b,\varphi,p,d} \leq M_{h,b,p,d,\varphi}\, u_{t,r,h,b} \quad \forall t, r, h, b, \varphi, p, d.$$

(vi) Latency SLO at p95 (choose one form):

Ungated: $\text{TTFT}_{h,b}(95) + \alpha_d T_{p,\text{default}}^{\text{out}} \text{TPOT}_{h,b}(95) \leq L_p^\star \ \forall h, b, p, d.$

Optional gated: $\text{TTFT}_{h,b}(95) + \alpha_d T_{p,\text{default}}^{\text{out}} \text{TPOT}_{h,b}(95) \leq L_p^\star + M'_{h,b,p,d}\big(1 - y_{t,r,h,b,p,d}\big)$, with

$$x_{t,r,h,b,\varphi,p,d} \leq \widehat{M}_{h,b,p,d,\varphi}\, y_{t,r,h,b,p,d} \text{ and } y_{t,r,h,b,p,d} \leq u_{t,r,h,b}.$$

Solve the MILP once over all $t \in T$; return $x^\star, n^\star, u^\star, z^\star$.

Algorithm 2 Aggregation and scope-transparent reporting (post-solve).

Inputs: Optimal decisions $x^\star, n^\star, u^\star, z^\star$; coefficients $E^{\mathrm{IT}}, E^{\mathrm{fac}}, \mathrm{mL}, g\mathrm{CO}_2^{\mathrm{LB}}$; arrivals $Q_{t,p}$; profile mix $(0.70, 0.25, 0.05)$.

Outputs: Daily *per-prompt* medians by profile, then mixed medians; comprehensive default, accelerator-only whiskers; LB default with MB sensitivity.

1: **for all** (t, p) **do**
2: $E^{\mathrm{IT}}_{t,p} \leftarrow \sum_{r,h,b,\varphi,d} x^\star_{t,r,h,b,\varphi,p,d}\, E^{\mathrm{IT}}_{h,b,\varphi,p,d}$
3: $\mathrm{mL}_{t,p} \leftarrow \sum_{r,h,b,\varphi,d} x^\star_{t,r,h,b,\varphi,p,d}\, \mathrm{mL}_{r,h,b,\varphi,p,d}$
4: $g\mathrm{CO}^{\mathrm{LB}}_{2,t,p} \leftarrow \sum_{r,h,b,\varphi,d} x^\star_{t,r,h,b,\varphi,p,d}\, g\mathrm{CO}^{\mathrm{LB}}_{2,r,h,b,\varphi,p,d}$
5: **Normalize to per-prompt**: $E^{\mathrm{IT}}_{t,p} = E^{\mathrm{IT}}_{t,p}/Q_{t,p}$, $w_{t,p} = \mathrm{Water}_{t,p}/Q_{t,p}$, $c^{\mathrm{LB}}_{t,p} = g\mathrm{CO}^{\mathrm{LB}}_{2,t,p}/Q_{t,p}$.
6: **end for**
7: For each profile p, take the median over $t = 1{:}288$ of $(E^{\mathrm{IT}}_{t,p}, w_{t,p}, c^{\mathrm{LB}}_{t,p})$; then mix by $0.70/0.25/0.05$ to get per-model medians.
8: Embodied and e-waste (once/day): $\mathrm{CO2}_{\mathrm{emb,day}} = \sum_{r,h} z^\star_{r,h}\, C^{\mathrm{emb}}_h / L_h$, $\mathrm{Waste}_{\mathrm{day}} = \sum_{r,h} z^\star_{r,h}\, M^{\mathrm{board}}_h / L_h$.
9: **Scopes.** Report comprehensive (default) via $E^{\mathrm{fac}} = \mathrm{PUE}_r\, E^{\mathrm{IT}}$; add accelerator-only whiskers using E^{acc} (or the observed narrow→comprehensive ratio when shares are undisclosed).
10: **Emissions.** Default LB via $\mathrm{CIF}^{\mathrm{LB}}_r$; add MB sensitivity by substituting EF^{MB}_r.

Figure 1 provides a visual overview of this pipeline, from inputs to final reporting.

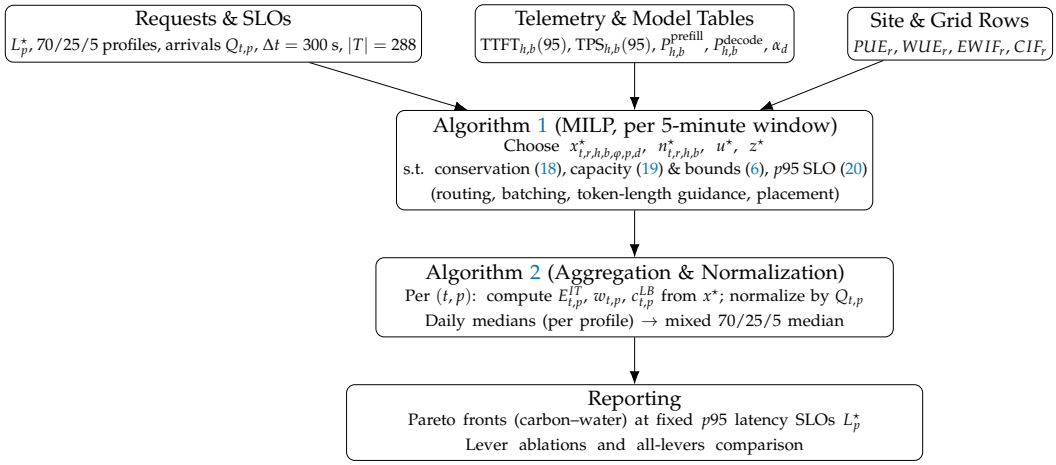


Figure 1. Framework overview. Inputs at five-minute resolution feed Algorithm 1 (per-window MILP). Algorithm 2 aggregates to per-window/per-profile impacts, normalizes per prompt, and forms daily medians and the mixed 70/25/5 value. Results are reported at fixed $p95$ latency SLOs $L^\star_p$.

3.6. Parameterization from Public Sources

We use the comprehensive serving boundary by default. Active accelerators, host CPU/DRAM, and provisioned idle are lifted to facility energy with the site PUE in Equations (13) and (14). The host + idle lift is anchored in the Gemini Apps production decomposition [1], and the PUE lift follows fleet reporting practice [62]. When only accelerator-only values are disclosed, we translate to the comprehensive boundary using the same lifts. If shares are undisclosed, we apply the production-observed narrow→comprehensive median ratio $E^{\mathrm{acc}}/E^{\mathrm{fac}} \approx 0.417$ for comparable workloads [1]. We report daily medians to match production practice and to keep results comparable across models and regions [1].

Per-prompt energy and the throughput and latency quantiles by model and profile come from the cross-model API benchmark [5]. We ingest Wh per prompt together with $p95$ TTFT, TPS, and TPOT rows. These quantiles parameterize the capacity and tail-latency

gates in Equations (19) and (20). As an example, the GPT-4o table lists a short-profile energy near 0.421 Wh and includes site multipliers. We keep the benchmark's provider and region mapping [5].

Prefill and decode coefficients follow the device-level accelerator-only models in Equations (10) and (11), using the average accelerator power $\bar{P}_{\mathrm{acc},h,b}(\varphi)$ by hardware and batch. We lift the IT energy with $\kappa_{\mathrm{host+idle}}$ in Equation (13) and the facility energy with the site PUE in Equation (14), consistent with the comprehensive boundary in [1].

Consumptive water follows the scope-consistent site+source rule in Equation (15). The site term scales with facility energy via $\mathrm{WUE}_{\mathrm{site},r}$, and the source term scales with IT energy via $\mathrm{EWIF}_{\mathrm{source},r}$. Priors and bounds for site WUE follow LBNL and AI water-footprint guidance [3,62]. Electricity–water intensity is applied as a location-based factor when hourly mixes are unavailable [3]. Operational emissions are location-based by default in Equation (16) using CIF with facility energy. We also report a market-based sensitivity by substituting the provider's prior-year portfolio factor when disclosed [1,25]. Units are reported in native form.

For the phase-aware variant, we use accelerator-only per-token energy curves that separate compute-bound prefill from memory-bound decode across relevant batches. This supports assignments that lower Wh per prompt while respecting $p95$ SLOs [22]. Embodied carbon per board, board mass, and lifetimes are taken from serving and lifecycle assessments [16,59]. Daily amortization follows the use-gated formulation in Section 3.4.6.

Reference ranges follow [62]. Hyperscale PUE is typically 1.15 to 1.35. Recent site WUE values are about 0.32 to 0.40 L kWh^{-1} with likely increases as liquid cooling penetrates. Scenario sweeps use ± 0.10 around a regional PUE baseline and $\pm 25\%$ around site WUE. When we construct hydro-like or nuclear-like sites for Pareto illustrations, CIF and electricity–water intensity remain within documented ranges and are flagged as replaceable medians when provider rows are disclosed [3,62].

All numerical inputs are packaged as machine-readable tables, so figures and results regenerate one-for-one when providers update disclosures. We include per-model Wh per prompt, $p95$ TTFT/TPS/TPOT by batch, and site rows with PUE, site WUE, electricity–water intensity, and location-based carbon intensity, as well as embodiment parameters [1,3,5,62]. When deployment caps are known, we set $U_{r,h,b}$ to the maximum provisionable replicas for class h at region r and batch b. Otherwise $U_{r,h,b}$ is a large operator-chosen bound.

Because grid carbon intensity and electricity–water intensity can be weakly correlated, improving one can worsen the other [3]. This motivates the bi-objective formulation and the routing-only and joint routing + batch + token analyses that make the trade-off geometry explicit under $p95$ SLOs in Sections 4.

Numerical parameters reflect a high-end GPU datacenter (A100-class accelerators) with host CPU/DRAM and typical PUE/WUE ranges; all energy terms are reported at the comprehensive serving boundary. The framework is hardware-agnostic: swapping to H100/TPU or mixed fleets only changes per-token power and the $p95$ throughput/latency tables used in the capacity and SLO constraints [1,5,62].

4. Results

We simulate a single 24-hour production day split into 288 five-minute decision windows and a fixed demand of 500 million prompts with a 70/25/5 short/medium/long mix. Short, medium, and long profiles use token pairs $(100, 300)$, $(1000, 1000)$, and $(10{,}000, 15{,}000)$, respectively. In every window the Σ Scale optimizer chooses, for each profile, the region, the batch size, and the phase-specific hardware (prefill vs. decode) to minimize an equal-weighted sum of energy, consumptive water, and location-based CO_2,

subject to explicit $p95$ SLOs and the capacity and conservation constraints. Capacity is enforced with per-hardware, per-batch tokens-per-second quantiles, and latency feasibility uses $p95$ time to first token and time per output token; the short prompt target is $\leq$0.9 s. Window outputs are aggregated to daily medians per profile and then mixed by 70/25/5 to obtain per-model medians. By default we report the comprehensive serving boundary—active accelerator + host CPU/DRAM + provisioned idle—lifted to facility energy by the site's PUE. Consumptive water is computed with the scope-consistent site + source rule, $E^{\mathrm{IT}}(\mathrm{PUE} \times \mathrm{WUE}_{\mathrm{site}} + \mathrm{EWIF})$; CO_2 is location-based using CIF, with a market-based sensitivity when prior-year portfolio factors are available. For reconciliation with chip-only studies, we also compute an accelerator-only subset and translate between scopes using the empirically observed narrow/comprehensive median ratio of 0.417 (approximately 2.4$\times$ uplift; e.g., 0.10 Wh accelerator-only vs. 0.24 Wh comprehensive at the median) [1]. Regional multipliers (PUE, $\mathrm{WUE}_{\mathrm{site}}$, EWIF, CIF) and the throughput/latency quantiles that bind the SLOs come from the cross-model public benchmark; a phase-aware variant uses device-level per-token energy curves for prefill/decode and includes embodied-carbon amortization when a hardware class is activated during the day [5]. Two policies are evaluated under identical demand and site multipliers. The *baseline* fixes batch = 8, uses the home region only, applies no token directive, and does not split phases. The Σ Scale *optimized* policy right-sizes the batch over the day, applies a concise token directive when SLO-safe, routes across sites in response to CIF and EWIF, assigns prefill and decode to different hardware when that lowers Wh/prompt, and amortizes embodied impacts; the contrast between these policies isolates the value of orchestration at a fixed model and boundary.

4.1. Comprehensive Boundary Medians and Daily Totals

Table 1 reports per-prompt median-scaled total and associated daily totals for energy, water (consumptive, site + source), and location-based CO_2 for four representative models. The *Optimized* rows correspond to the Σ Scale loop with all levers active (batch right-sizing, semantic token control, geo-routing, phase-aware assignment, and second-life amortization). *Baseline* uses the same comprehensive boundary and medians, but with batch = 8, no token control, home region only, and no phase split. Across the four evaluated models, the Σ Scale policy delivers large and consistent savings at the comprehensive serving boundary. After aggregating window outputs to daily medians per profile and mixing by 70/25/5, the median energy per prompt falls by 57–59%, the median consumptive water (site + source) by 59–60%, and the median location-based CO_2 by 78–80%, all while meeting $p95$ capacity and $p95$ latency in every five-minute window. For GPT-4o, the comprehensive baseline is 0.6876 Wh/prompt, 2.3915 mL/prompt, and 0.2426 g CO_2/prompt; after optimization, the medians become 0.2898 Wh, 0.9803 mL, and 0.0507 g CO_2, respectively. At a volume of 500 M queries per day these medians translate to energy dropping 0.344 $\rightarrow$ 0.145 GWh, water 1.196 $\rightarrow$ 0.490 ML, and LB CO_2 121 $\rightarrow$ 25 t under the same SLA. The other models follow the same pattern, with absolute magnitudes scaling with baseline Wh/prompt (e.g., Claude 3.7 Sonnet shows the largest absolute reductions because its baselines are highest). These gains reflect full-stack serving and scope-consistent water accounting rather than chip-only power, and they arise from the combined effect of batch right-sizing (e.g., 8 $\rightarrow$ 16 off-peak), token-length directives, carbon- and water-aware geo-routing, phase-aware hardware assignment, and second-life amortization. See Figure 2 for a side-by-side visualization of the comprehensive-boundary medians (baseline vs. optimized) for energy, consumptive water, and LB CO_2, with lower "scope whiskers" showing accelerator-only values via the observed narrow/comprehensive $\approx$ 0.417 ratio.

Table 1. Comprehensive boundary medians (weighted by the 70/25/5 mix).

Metric	GPT-4o	GPT-4o Mini	Claude 3.7 Sonnet	LLaMA 3 70B †
Baseline Wh/prompt	0.6876	0.7545	1.55635	0.97145
Optimized Wh/prompt	0.289824	0.319896	0.664582	0.400321
Δ Energy %	−57.8	−57.6	−57.3	−58.8
Baseline mL/prompt	2.391473	2.624151	5.412985	3.378703
Optimized mL/prompt	0.980349	1.081547	2.245570	1.356734
Δ Water %	−59.0	−58.8	−58.5	−59.8
Baseline g CO_2/prompt (LB)	0.242585	0.266188	0.549080	0.342728
Optimized g CO_2/prompt (LB)	0.050739	0.055035	0.111840	0.074971
Δ CO_2 %	−79.1	−79.3	−79.6	−78.1
Baseline Energy (GWh/d)	0.3438	0.37725	0.778175	0.485725
Optimized Energy (GWh/d)	0.144912	0.159948	0.332291	0.200160
Baseline Water (ML/d)	1.196	1.312	2.706	1.689
Optimized Water (ML/d)	0.490	0.541	1.123	0.678
Baseline CO_2 (t/d, LB)	121.293	133.094	274.540	171.364
Optimized CO_2 (t/d, LB)	25.370	27.517	55.920	37.485

† Long-prompt Wh for LLaMA 3 70B is not reported in the public table; the line aggregates short+medium medians and is flagged accordingly. Units: 1 ML = 10^6 L = 10^9 mL. At 500 M prompts/day, ML/d = 0.5× mL/prompt; GWh/d = 0.5× Wh/prompt; t/d = 500× g/prompt.

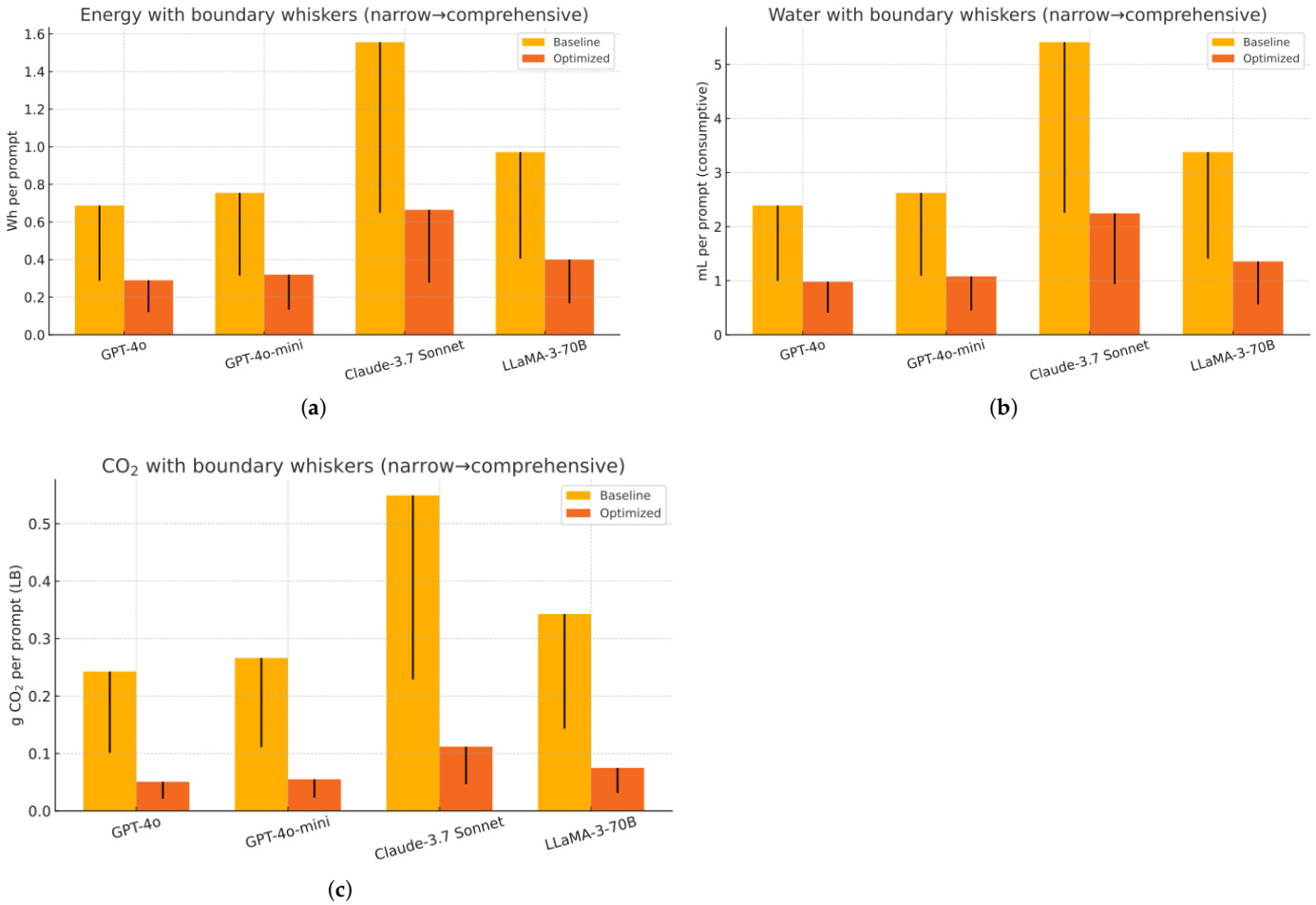


Figure 2. Comprehensive serving boundary medians per prompt (baseline vs. optimized). (**a**) Energy (Wh); (**b**) Consumptive water (mL, site + source); (**c**) Location-based CO_2 (g, LB). Lower whiskers (scope whiskers) indicate narrower boundaries: for energy and CO_2, they represent accelerator-only values derived via the observed ratio (≈0.417). For water, they indicate source-only usage ($E^{acc} \times$EWIF) excluding site water.

Absolute impacts scale with each model's baseline Wh/prompt and $p95$ quantiles (TTFT/TPOT, TPS), so heavier models deliver larger absolute savings even when percentage reductions are similar. Across GPT–4o, GPT–4o–mini, Claude–3.7 Sonnet, and LLaMA–3–70B, the optimized policy yields comparable percentage reductions in energy, water, and LB CO_2 (Table 1), while the absolute gains track each model's baseline. Architectural differences (e.g., dense vs. MoE, context length, batch sensitivity) enter only through the per-model, per-batch $p95$ tables used by the capacity (19) and $p95$-latency (20) gates; these reshape the feasible region but do not change the method. Consequently, the same orchestration levers, geo-routing, batch right-sizing, and concise directives α_d, apply uniformly across model families.

4.2. Scope Reconciliation: Accelerator-Only vs. Comprehensive

A persistent source of disagreement in the literature is the accounting boundary used for inference. To make our results directly comparable to chip-only studies, we compute both scopes for the same traffic, mix, and site multipliers and place them side by side. The empirical narrow/comprehensive median energy ratio in our runs is ≈0.417, reproducing the ≈2.4× uplift observed in production telemetry when host CPU/DRAM, provisioned idle, and facility overheads (PUE) are included [1]. This factor is not universal—it varies with architecture and fleet management—but when underlying shares are unavailable, it provides a defensible translation between scopes. Specifically, if a study reports only accelerator-level energy $E_{\text{acc-only}}$ (Wh prompt^{-1}), the comprehensive median may be approximated by $E_{\text{comprehensive}} \approx E_{\text{acc-only}}/0.417$, holding PUE, WUE_{site}, EWIF, and the prompt mix constant.

Table 2 provides scope translation to reconcile chip-only studies with comprehensive accounting. The numerical proximity between optimized comprehensive and baseline narrow in our run (∼0.42× of baseline comprehensive in both cases) is coincidental: the former reflects optimization levers and SLOs; the latter reflects boundary accounting shares measured in production. Together with Table 1 (policy effects at the comprehensive boundary), Table 2 shows a clear, citable translation layer so that chip-only results can be reconciled with full-stack accounting without ambiguity.

Table 2. Accelerator-only vs. comprehensive (medians, 70/25/5 mix).

Metric	GPT-4o	GPT-4o Mini	Claude 3.7 Sonnet	LLaMA 3 70B
Narrow Wh/prompt	0.2865	0.314375	0.648479	0.404771
Comprehensive Wh/prompt	0.6876	0.7545	1.55635	0.97145
Narrow/Comprehensive	0.417	0.417	0.417	0.417
Narrow mL/prompt	0.996447	1.093396	2.255411	1.407793
Comprehensive mL/prompt	2.391473	2.624151	5.412985	3.378703
Narrow g CO_2/prompt (LB)	0.101077	0.110911	0.228783	0.142803
Comprehensive g CO_2/prompt (LB)	0.242585	0.266188	0.549080	0.342728

4.3. Carbon–Water Movement at Fixed QoS

Because grid carbon intensity (CIF) and Electricity–Water Intensity (EWIF) are weakly correlated, carbon-optimal routing can be water-suboptimal. Our Σ Scale policy jointly optimizes both metrics. Figure 3 shows each model's movement from *baseline* to *optimized* in the $(\text{g CO}_2,\ \text{mL})$ plane. For each model, we plot the comprehensive-boundary median at the baseline configuration and the corresponding median under the Σ Scale policy; the two points are connected by an arrow. In every case, the arrow points down and to the left, showing that the optimized policy reduces both location-based CO_2 and consumptive water while honoring the $p95$ SLOs. This simultaneous reduction is non-trivial because

CIF and EWIF are only weakly correlated; a carbon-only router can increase water if it shifts load to a clean but water-intensive region. The Σ Scale objective avoids that pitfall by jointly reducing Wh per prompt—through batch right-sizing, concise token directives, and phase-aware placement—and by routing remaining Wh to regions and hours with favorable CIF and PUE $\times$ $\mathrm{WUE}_{\mathrm{site}}$ + EWIF.

The geometry is intuitive. Each site i defines a ray through the origin in the (g CO_2, mL) plane whose slope is W_i/CIF_i, with $W_i = \mathrm{PUE}_i \times \mathrm{WUE}_{\mathrm{site},i} + \mathrm{EWIF}_i$. Moving along a ray changes only energy (Wh) via batching and token control; switching rays changes the water-to-carbon ratio by altering site factors while holding Wh fixed. The optimizer exploits both degrees of freedom: it lowers Wh and, subject to the latency budget, selects rays with more favorable slopes. In a two-region sensitivity where the alternative site is hydro-like with both lower CIF and a lower combined water factor, routing alone becomes a Pareto improvement; the arrow's down-left direction then reflects a pure siting effect. In other configurations, the arrow still points down-left because energy reductions combine with selective routing to offset sites where low carbon coincides with higher water.

To keep the scope visible without obscuring directionality, Figure 3 overlays asymmetric boundary whiskers on both endpoints. The barbed end of each whisker marks the accelerator-only value obtained by multiplying the comprehensive coordinate by 0.417 on both axes; there is no upper whisker because our defined upper boundary is comprehensive. The tip-to-tail arrows and their whiskered counterparts are nearly parallel and scaled, illustrating that the direction and magnitude of the optimization gain are robust to scope choice: changing scope shifts absolute values by the familiar ≈2.4$\times$ but does not alter the qualitative improvement.

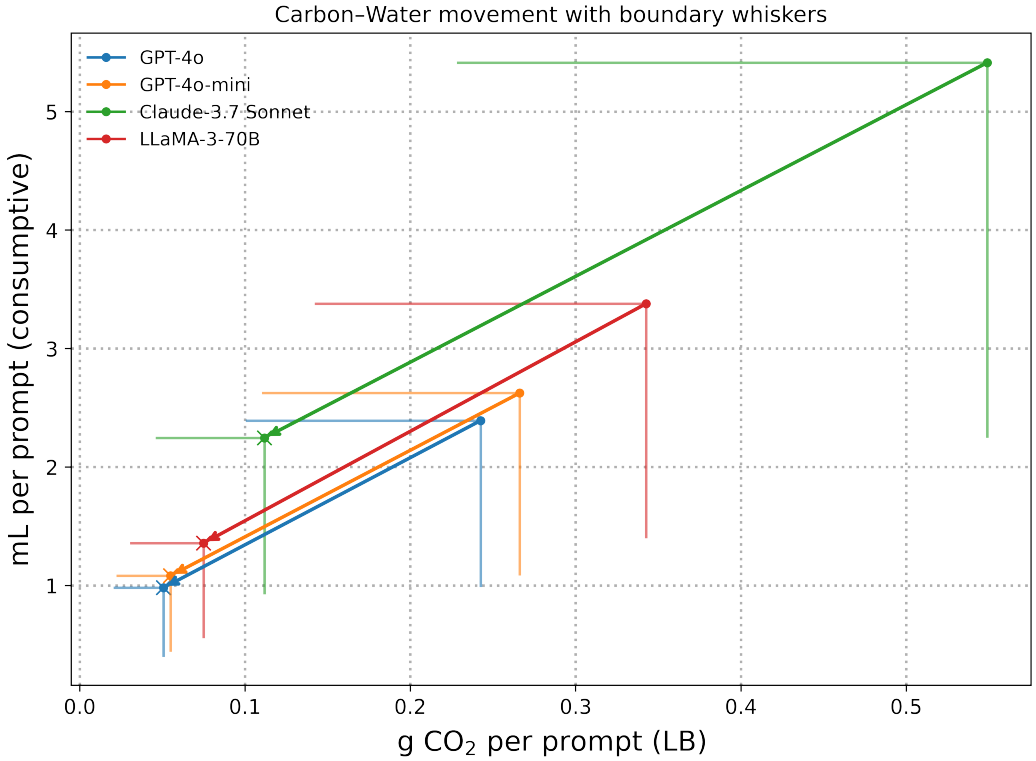


Figure 3. Carbon–water movement at fixed QoS. For each model, the arrow connects the comprehensive-boundary medians from baseline to optimized. Lower whiskers mark accelerator-only values: energy and LB CO_2 via the 0.417 energy ratio and water via source-only E^{acc} $\times$ EWIF.

This figure is diagnostic rather than a full carbon–water Pareto frontier. Constructing that frontier would require multiple MILP runs with different objective weights. At fixed

QoS, the comprehensive, SLO-aware policy achieves concurrent reductions in CO_2 and water for the evaluated workloads. In the next section, we quantify the additional reductions available if small latency relaxations or stronger token-length directives are permitted.

4.4. Routing-Only Carbon–Water Pareto Under SLOs

Carbon and water are not perfectly aligned—CIF and EWIF can diverge within and across regions—so a single weighted score can hide trade-offs. We therefore plot Pareto frontiers at fixed SLOs to expose the full set of non-dominated operating points, a standard practice in multi-objective environmental optimization [63,64]. The frontier reveals when both metrics improve together and where trade-offs arise, and it shows how joint routing + batch + token controls dominate routing-only. The Pareto view is weight-agnostic and complements the scalarized result by making the trade-off geometry explicit [65–68].

The previous subsection established that the Σ Scale policy moves every model down and to the left in the $(\mathrm{g\,CO_2},\ \mathrm{mL})$ plane: batching and token directives reduce watt-hours per prompt, while geo-routing chooses sites whose environmental multipliers further shrink carbon and water, all without violating the $p95$ latency SLO. We now isolate the routing lever to understand the geometry of those gains. To avoid conflating routing with operational changes, we hold each model's optimized comprehensive-boundary energy per prompt $E_m^\star$ (Wh prompt^{-1}) fixed and vary only the regional mix subject to the same SLO. This yields an interpretable map of what geo placement alone can deliver once the runtime system has already right-sized batch and applied concise generation.

Let $s = (s_b, s_h, s_n)$ denote the shares routed to a baseline U.S. thermal site b, a hydro-dominated site h, and a nuclear-like thermal site n, with $s_i \geq 0$ and $s_b + s_h + s_n = 1$. Each site i is characterized by a location-based grid carbon intensity CIF_i (kg $\mathrm{CO_2\,kWh^{-1}}$) and a consumptive water factor W_i, following a scope-consistent convention (on-site cooling scaled by PUE, plus source-side electricity–water intensity added directly). With $E_m^\star$ fixed, the per-prompt impacts for any routing mix s are linear:

$$\mathrm{g\,CO_2/prompt}(s) = E_m^\star \sum_{i\in\{b,h,n\}} s_i\,\mathrm{CIF}_i, \qquad \mathrm{mL/prompt}(s) = E_m^\star \sum_{i\in\{b,h,n\}} s_i\,W_i. \tag{22}$$

Equation (22) implies that the feasible set in the $(\mathrm{g\,CO_2},\ \mathrm{mL})$ plane is the convex hull of the three single-site vertices $\{\ E_m^\star \mathrm{CIF}_i,\ E_m^\star W_i\ \}$; the Pareto frontier is the lower-left convex envelope of those vertices. Each site also defines a ray through the origin with slope W_i/CIF_i: moving along a ray changes only the energy scale, whereas switching rays changes the water-to-carbon ratio at fixed energy.

Because the hydro site in our parameterization couples very low CIF_h with a high electricity–water intensity (we use 5.50 L $\mathrm{kWh^{-1}}$ to stress the conflict), while the nuclear-like site has a lower combined water factor W_n but a higher carbon intensity CIF_n than hydro, the non-dominated set collapses to the straight edge joining hydro and nuclear. The baseline site lies to the right of both in carbon and does not beat nuclear on water; hence, it is dominated. Along the efficient edge, a single parameter $\lambda \in [0,1]$ (the nuclear share) describes the trade-off curve:

$$\mathrm{mL/prompt}(\lambda) = E_m^\star\big[(1-\lambda)W_h + \lambda W_n\big], \tag{23}$$

$$\mathrm{g\,CO_2/prompt}(\lambda) = E_m^\star\big[(1-\lambda)\mathrm{CIF}_h + \lambda \mathrm{CIF}_n\big]. \tag{24}$$

The slope of this edge,

$$\frac{d\,(\mathrm{mL/prompt})}{d\,(\mathrm{g\,CO_2/prompt})} = \frac{W_n - W_h}{\mathrm{CIF}_n - \mathrm{CIF}_h}, \tag{25}$$

depends only on site properties and is therefore independent of $E_m^\star$. This scaling invariance explains why the four model panels share the same shape but span different numerical ranges: increasing $E_m^\star$ stretches both axes uniformly without changing which combinations are efficient.

Latency feasibility is enforced by a conservative $p95$ proxy:

$$\mathrm{TTFT}_{\mathrm{p95}}(s) = \sum_{i \in \{b,h,n\}} s_i \, \mathrm{TTFT}_{i,\mathrm{p95}} \;\le\; \mathrm{SLO}, \tag{26}$$

so the routing cloud shows exactly those mixes that respect the interactive QoS; infeasible mixes are shaded separately for transparency. With this setup in place, the four panels in Figure 4, for GPT-4o, GPT-4o-mini, Claude-3.7 Sonnet, and LLaMA-3-70B, respectively, the feasible routing simplex in $(\mathrm{g\,CO_2},\ \mathrm{mL})$ space, the hydro and nuclear vertices that determine the efficient edge via (23)–(25), and the dominated baseline vertex. Selecting any point on the red edge is equivalent to choosing λ to meet a carbon or water budget at unchanged quality of service.

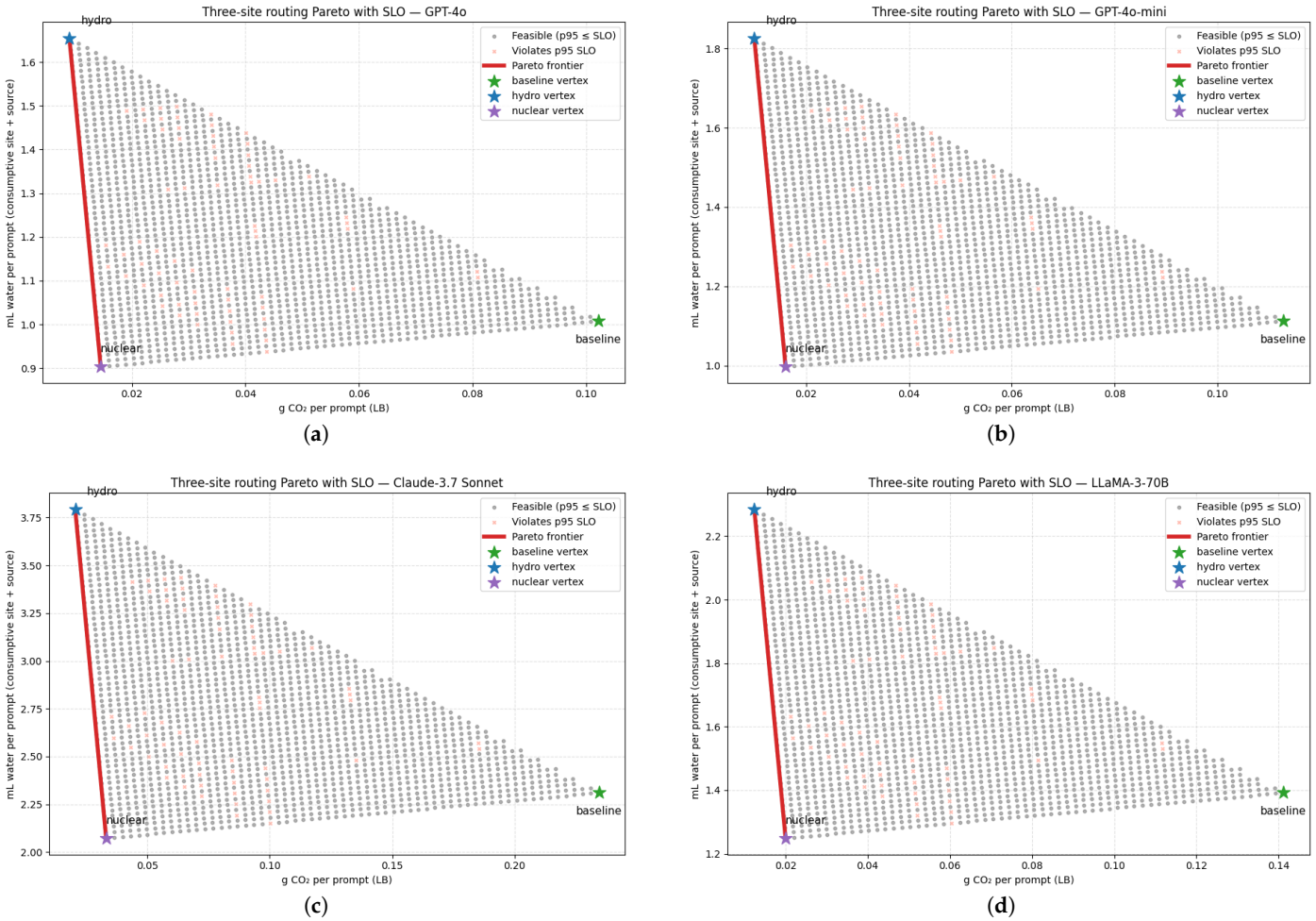


Figure 4. Three-site routing Pareto with SLO (one panel per model). (**a**) GPT-4o. (**b**) GPT-4o-mini. (**c**) Claude-3.7 Sonnet. (**d**) LLaMA-3-70B. Each dot is a routing mix $s = (s_b, s_h, s_n)$ across a baseline U.S. thermal site, a hydro-dominated site, and a nuclear-like site, evaluated at the model's optimized energy $E_m^\star$. Coordinates follow Equation (22) where $\mathrm{g\,CO_2} = E_m^\star \sum_i s_i \, \mathrm{CIF}_i$ and $\mathrm{mL} = E_m^\star \sum_i s_i \, W_i$. Gray points satisfy the $p95$ TTFT SLO in Equation (26), while salmon points violate it. The red polyline is the lower-left Pareto frontier. Under hydro (low CIF but high EWIF), the efficient set is the hydro↔ nuclear edge, and the baseline vertex is dominated due to higher carbon and no water advantage.

4.5. Joint Frontiers from Site + Batch + Token Sweeps Under the SLO

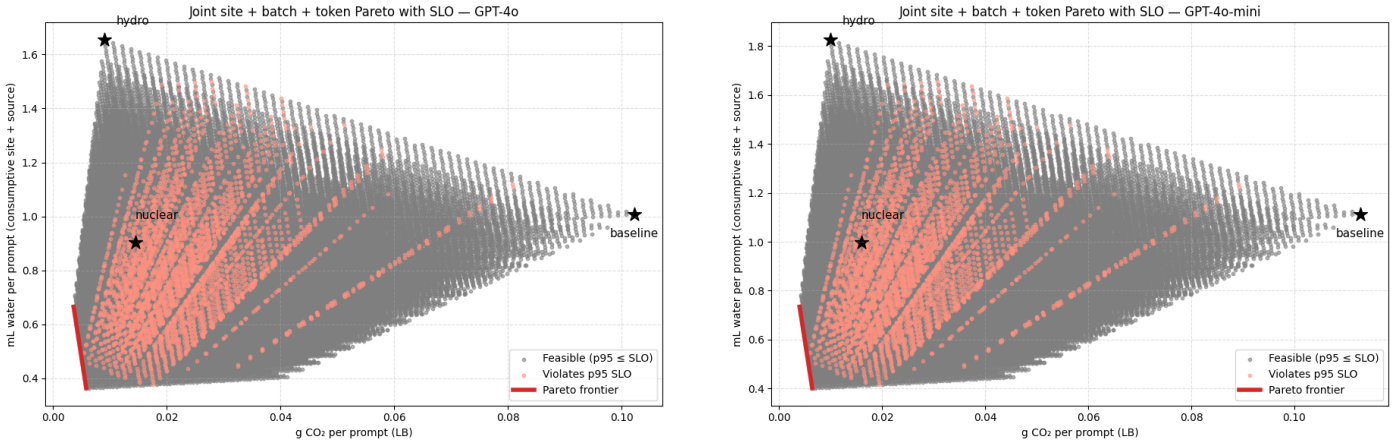
The routing-only view is conservative because it fixes $E_m^\star$. In practice, the serving stack can reduce energy by right-sizing the batch and by encouraging concise generations when quality allows. We extend the enumeration to include a batch multiplier b and a token-length multiplier t with $b, t \in [0, 1]$. In our grids $b \in [0.561, 1]$ for batch $8 \to 16$ medians and $t \in [0.7175, 1]$ for default→brief. For a routing mix s the effective energy is $E(b, t) = E_m^\star bt$, and the per-prompt impacts are determined by:

$$\text{g CO}_2/\text{prompt}(s, b, t) = E_m^\star bt \sum_{i \in \{b,h,n\}} s_i \, \text{CIF}_i, \tag{27}$$

$$\text{mL/prompt}(s, b, t) = E_m^\star bt \sum_{i \in \{b,h,n\}} s_i \, W_i. \tag{28}$$

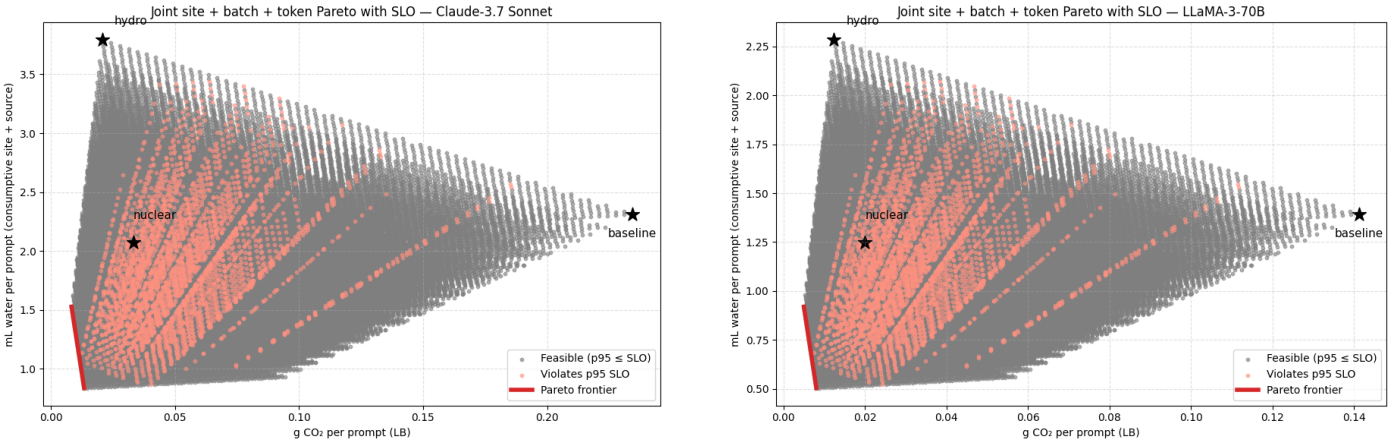
Both axes scale linearly with energy. Every point in the routing triangle moves down and left by the factor of bt. As b and t vary, the routing triangle becomes a wedge of scaled triangles, and the joint Pareto frontier is the lower left envelope of that wedge. Feasibility uses the conservative $p95$ proxy in (26), which mixes per-site $p95$ TTFT linearly by s. If providers supply measured $p95$ curves as functions of batch and region, they can be used in the same mechanism without changing the frontier construction.

Figure 5 sweeps s, b, and t for GPT-4o, GPT-4o-mini, Claude-3.7 Sonnet, and LLaMA-3-70B. Gray points satisfy the $p95$ SLO. Salmon points violate it. The red curve is the feasible lower left envelope. Black stars mark the three single-site anchors at $b = t = 1$. These stars lie inside the feasible cloud because they mix with the same s and smaller bt reduce both axes while remaining SLO-safe. Relative to the routing-only panels, the frontiers move down and left, which quantifies the additional reductions from batching and concise tokens. For GPT-4o and GPT-4o-mini, the wedge is dense, and the frontier tracks the extreme lower left. For Claude-3.7 Sonnet, the larger $E_m^\star$ tightens the SLO near high batch and trims the feasible set. Across all models, the efficient edge aligns with the hydro-to-nuclear trade-off because hydro is carbon-optimal and nuclear is water-optimal. In operational terms, routing chooses where energy is consumed through CIF and W. Batch and tokens set how much energy per prompt is consumed through $E_m^\star bt$.



(**a**) (**b**)

Figure 5. *Cont.*



(**c**) (**d**)

Figure 5. Joint site + batch + token Pareto with SLO (one panel per model). (**a**) GPT-4o. (**b**) GPT-4o-mini. (**c**) Claude-3.7 Sonnet. (**d**) LLaMA-3-70B. Each dot corresponds to a triple (s, b, t) of routing shares, batch multiplier, and token-length multiplier. Impacts follow Equations (27) and (28): g $CO_2 = E_m^\star bt\,\mathrm{CIF}_{\mathrm{mix}}$ and mL $= E_m^\star bt\,W_{\mathrm{mix}}$, where $\mathrm{CIF}_{\mathrm{mix}} = \sum_i s_i\,\mathrm{CIF}_i$ and $W_{\mathrm{mix}} = \sum_i s_i\,W_i$. Gray points satisfy the $p95$ TTFT SLO, and salmon points violate it. The red curve is the feasible lower-left Pareto frontier. Stars mark the single-site anchors at $b = t = 1$, which are dominated once batching and concise tokens are allowed. Relative to routing only, the feasible cloud becomes a wedge of scaled triangles, and the frontier moves down and left, illustrating how operational levers compound with siting.

4.6. Ablation: Where the Gains Come from (Fixed p95 SLOs)

We quantify the marginal contribution of each operational lever at fixed quality of service, using the same data and boundary choices. We evaluate feasible point clouds in the (gCO_2, mL) plane under the $p95$ latency constraint in (20), capacity in (19), and conservation in (18). We vary exactly one degree of freedom at a time: routing; routing + batch; routing + batch + token; routing + phase split. Appendix A (Figures A1–A8) reports per-model ablations for Claude-3.7 Sonnet (Figures A1 and A2), GPT-4o (Figures A3 and A4), GPT-4o-mini (Figures A5 and A6), and LLaMA-3-70B (Figures A7 and A8): mixed 70/25/5 daily medians with 95% CIs for LB CO_2 and consumptive water (bars), and by-region distributions (violins). Across all models, the ablation ladder from baseline to A1 (routing), A2 (routing + batch), A3 (routing + batch + token), A4 (routing + phase split), and *all* widens the feasible operating set under the $p95$ SLO and shifts medians down and left in the carbon–water plane, with most of the gain realized by A3 and a smaller but consistent increment from A4.

As in the main results, carbon is location-based by default and water is the consumptive site+source via (15). When an ablation holds energy fixed, we use the per-model optimized energy per prompt $E_m^\star$ from Section 4.1. Enumeration is deterministic with no RNG. Daily medians are computed by profile and then mixed 70/25/5 for short, medium, and long prompts.

A1 Routing-only. Holding $E_m^\star$ fixed, let $s = (s_b, s_h, s_n)$ denote the shares across the three representative sites with $s_i \geq 0$ and $\sum_i s_i = 1$. Impacts are affine in s by (22), so the feasible set is the routing simplex and the efficient set is its lower-left envelope. For very low CIF_h but higher EWIF_h for hydro, the efficient edge collapses to hydro↔nuclear, and the baseline vertex is dominated (see Figure 6). Latency feasibility overlays follow the conservative $p95$ mix proxy in (26).

A2 Routing + Batch. We add a batch multiplier $b \in [b_{\min}, 1]$ that scales energy nearly multiplicatively (while respecting (20)):

$$E_m^\star \mapsto b\, E_m^\star, \qquad (\text{g, mL}) \mapsto b\,(\text{g, mL}). \tag{29}$$

Each routing triangle is scaled down and left by b, which shifts the Pareto envelope toward the origin (see Figure 7).

A3 Routing + Batch + Token (wider wedge). We also sweep a token-length multiplier $t \in [t_{\min}, 1]$ (*default*→*brief*). With $E_m^\star \mapsto bt\, E_m^\star$, both axes contract by the same bt factor. The feasible cloud thickens and the efficient frontier extends down and left relative to A2 (see Figure 8).

A4 Routing + Phase split (prefill vs. decode). Using the phase-aware per-token energy models $e_{\text{Wh/token}}(h, \varphi, b)$ and the prefill and decode constructions in (10) and (11), we expose the energy-shaping effect of placing prefill and decode on different cohorts. For visualization, we sweep phase multipliers $(\eta_{\text{prefill}}, \eta_{\text{decode}}) \in [0.90, 1]^2$ with decode share $\rho \in [0, 1]$ and map $E_m^\star \mapsto E_m^{\star,\text{phase}} = E_m^\star\big((1-\rho)\eta_{\text{prefill}} + \rho\,\eta_{\text{decode}}\big)$ subject to the same SLO guard. The frontier shifts further toward the origin. The hydro↔nuclear edge remains slope-setting with slope given by (25) (see Figure 9).

In all four panels, ∘ points satisfy $p95$ SLOs; × points violate it. To keep uncertainty visible without clutter, we show *vertex whiskers*: for water, PUE $\in [\text{PUE}_0 \pm 0.10]$ and $\text{WUE}_{\text{site}} \in [0.75, 1.25] \times \text{WUE}_{\text{site},0}$ (holding EWIF at the published median); for carbon, a ±15% CIF envelope when only LB is available, otherwise an LB↔MB swap whisker when provider MB factors are disclosed. These match the uncertainty bands and scope practice used elsewhere in the paper.

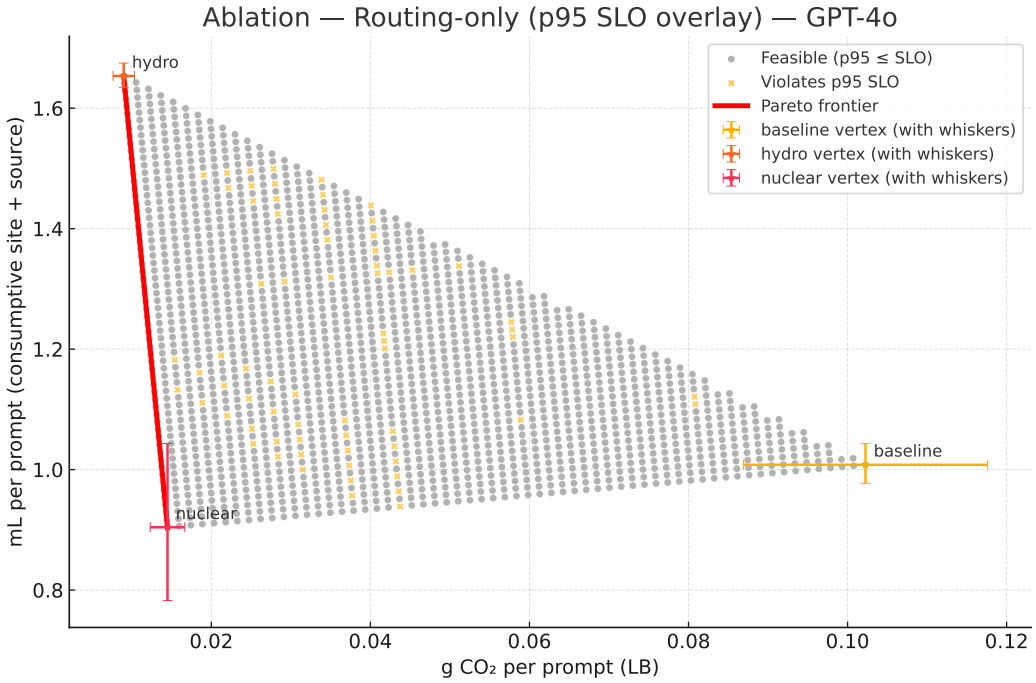


Figure 6. Ablation A1—Routing-only, SLO overlay (GPT-4o). Feasible routing simplex in the (g CO_2, mL) plane at fixed optimized energy $E_m^\star$, with site factors (hydro: very low CIF but higher EWIF). Stars mark the three single-site vertices; ∘: $p95$-feasible; ×: $p95$-violating (mix proxy (26)). The red polyline is the lower-left Pareto frontier; it coincides with the hydro↔nuclear edge. Sensitivity whiskers at vertices: PUE ± 0.10, WUE_{site} ± 25% (water), CIF ±15% (carbon) or LB↔MB when available.

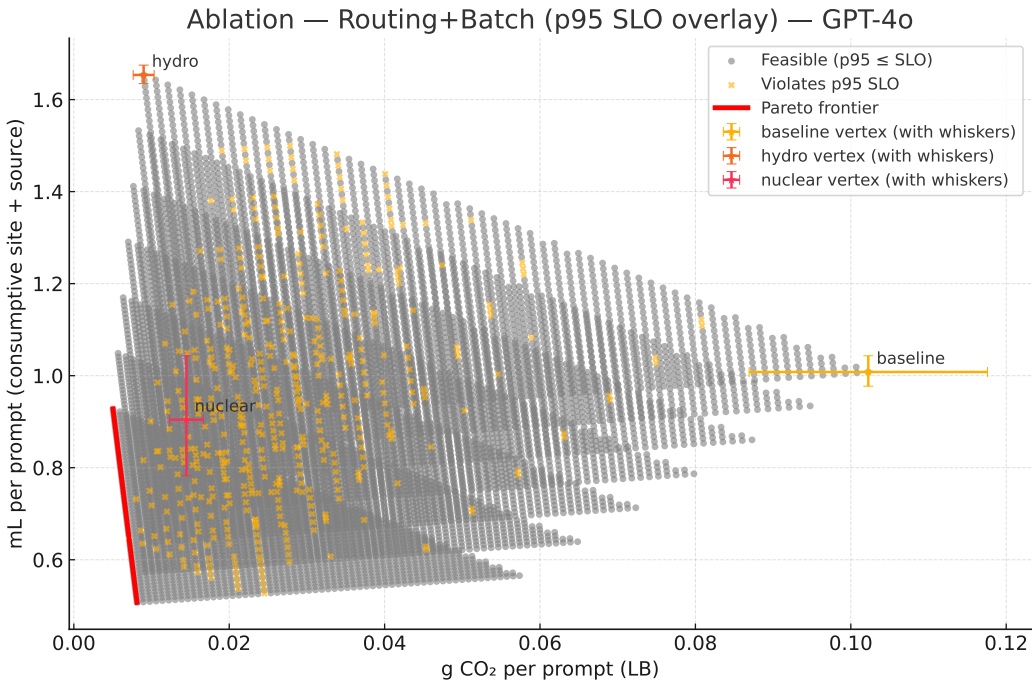


Figure 7. Ablation A2—Routing + Batch, SLO overlay (GPT-4o). Adds a batch multiplier $b \in [b_{\min}, 1]$ that scales energy and thus both axes by b while meeting $p95$ SLOs. The routing triangle thickens into a wedge of scaled triangles; the feasible red frontier shifts down-left relative to A1. Vertex whiskers are the same as in Figure 6.

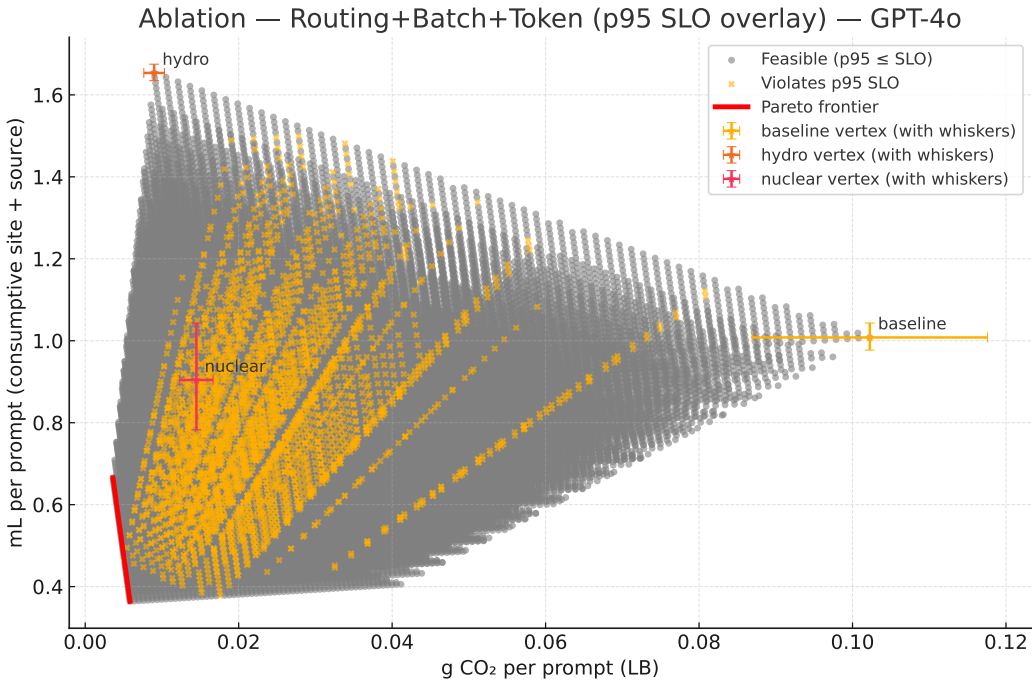


Figure 8. Ablation A3—Routing + Batch + Token, SLO overlay (GPT-4o). Adds a token-length multiplier $t \in [t_{\min}, 1]$ (default→brief), giving a uniform bt contraction on both axes. The feasible cloud expands and the lower-left frontier lengthens compared to A2; single-site stars at $b = t = 1$ are dominated once batching and concise tokens are allowed. Vertex whiskers as in Figure 6.

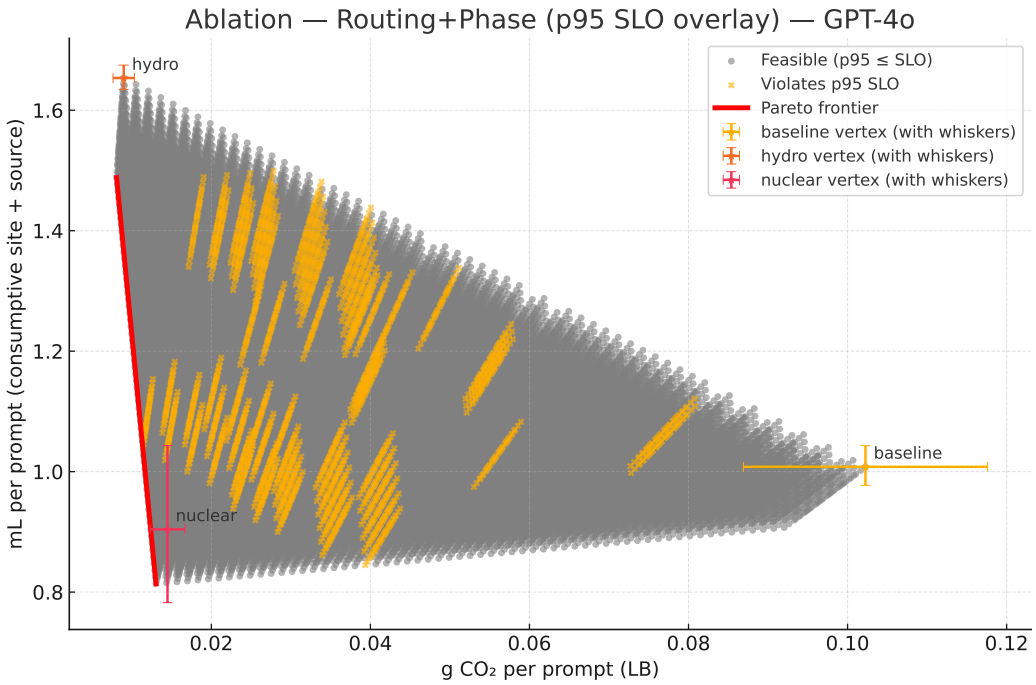


Figure 9. Ablation A4—Routing + Phase split, SLO overlay (GPT-4o). Prefill (compute-bound) and decode (memory-bound) are allowed to run on different cohorts using phase-aware $e_{\mathrm{Wh/token}}(h, \varphi, b)$; we sweep $(\eta_{\mathrm{prefill}}, \eta_{\mathrm{decode}}) \in [0.90, 1]^2$ with default decode share $\rho \approx 0.7$. The feasible cloud nudges further down-left relative to A1; the hydro↔nuclear edge still sets the trade-off slope (25). Whiskers and SLO overlay as before.

Across A1 to A4, the feasible set widens and the frontier moves toward the origin. Routing chooses the slope through the site mix. Batch, token, and phase split reduce watt-hours per prompt at fixed QoS, which shrinks both axes. This explains why the optimized medians in Table 1 land down and left of the baseline points without relaxing SLOs and why single-site anchors are dominated once the operational levers are enabled.

5. Discussion

This work introduces a provider-agnostic, time-resolved framework that couples scope-transparent measurement with a mixed-integer orchestration loop (Σ-Scale) to co-minimize the carbon and water footprints of LLM serving under production-grade SLOs. Reporting daily medians at a *comprehensive* boundary, active accelerators + host CPU/DRAM + provisioned idle with PUE, resolves a major source of disagreement in prior studies and aligns with operational telemetry. In this boundary, the empirical translation between accelerator-only and comprehensive scopes (narrow/comprehensive $\approx$ 0.417) enables direct, auditable comparisons across systems and papers; interventions that appear large at the chip boundary can attenuate—or reverse—once host, idle, and facility overheads are included [1].

Operationally, a *single* SLO-aware policy achieves large, consistent reductions without relaxing interactivity. Across four models, median per-prompt impacts fall by roughly 57–59% (energy), 59–60% (consumptive water; site + source), and 78–80% (location-based CO_2), with $p95$ TTFT/TPOT and capacity constraints met in every five-minute window. For a representative 500 M-query day on GPT-4o, totals drop from 0.344 $\rightarrow$ 0.145 GWh, 1.196 $\rightarrow$ 0.490 ML, and 121 $\rightarrow$ 25 t CO_2 (LB). These gains arise from complementary levers like batch right-sizing, concise token directives, carbon- and water-aware geo-routing (via CIF and EWIF), and phase-aware prefill/decode placement, rather than from any single mechanism.

Larger models exhibit higher per-prompt baselines and longer $p95$ quantiles, so absolute sustainability gains increase even when percentage reductions remain broadly stable. In our experiments, GPT–4o and GPT–4o–mini achieve comparable percentage reductions, yet at the same volumem GPT–4o delivers larger absolute daily savings. The MILP adapts via model-specific $p95$ tables—capacity (19) and SLO (20)—so the feasible region naturally tightens as models scale while SLO feasibility is preserved. Within those tighter budgets, the policy continues to exploit geo-routing, batch right-sizing, and concise directives α_d where allowed, which explains the similar percentages but larger absolute reductions for heavier models.

Model-side tables (Wh/prompt, $p95$ TTFT/TPOT, TPS by batch) can be profiled during development on the target hardware and then reused at deployment. Site and grid rows (PUE, $\mathrm{WUE}_{\mathrm{site}}$, $\mathrm{CIF}_{\mathrm{LB}}$, EWIF) admit day-ahead forecasts. The daily MILP in Equation (17), subject to (18)–(21), can be re-solved as forecasts update, enabling a rolling, model-predictive schedule at a five-minute cadence while preserving the $p95$ latency guard (20) and capacity bounds (19). Feasibility depends on the profiled $p95$ tables, so forecast errors shift objective values but do not compromise SLO satisfaction.

A central conceptual contribution is to treat *water* as a first-class objective, computed as site cooling scaled by PUE plus source-side electricity–water intensity, and to optimize it jointly with carbon [3]. Because CIF and EWIF are only weakly correlated, optimizing one can worsen the other. The framework's dual-objective design and the "ray" geometry in the (g CO_2, mL) plane make these trade-offs explicit: routing alone traces a Pareto edge determined by site factors, while adding batch and token controls translates the entire feasible cloud toward the origin, compounding siting with operational efficiency (Figures 2–5).

Choosing medians over means for skewed mixes matches production practice and provides a stable basis for policy comparison. Table 2 makes scope reconciliation explicit by showing the $\sim$2.4$\times$ uplift from accelerator-only to comprehensive across models; pairing comprehensive medians with scope "whiskers" avoids ad hoc conversion factors and helps readers reconcile results across boundary choices [1].

From a deployment standpoint, three near-term steps follow. (i) Integrate carbon- and water-aware geo-routing into global load balancers, enforcing $p95$ latency and using realistic tokens-per-second quantiles. (ii) Apply concise generation directives to curb unnecessary tokens, especially when combined with higher off-peak batching. (iii) Use phase-aware placement—fast, compute-efficient cohorts for prefill and memory-efficient or second-life cohorts for decode—to extend hardware life and bring embodied impacts into scope without sacrificing responsiveness [16,22].

Limitations point directly to future work. Hourly variability in PUE, site WUE, and electricity-mix water intensities suggests moving from annual to time-resolved site and grid factors to sharpen routing signals. Market-based carbon is treated here as a sensitivity; incorporating temporal matching and procurement constraints (e.g., REC/PPA portfolios) would enable joint LB/MB evaluations [25]. Water could be weighted by basin-level scarcity to reflect environmental equity [3,69]. Finally, richer user-experience models (percentile bands by profile/region) and live-traffic experiments would strengthen external validity; deeper lifecycle modeling (refurbishment yields, end-of-life pathways) would close the loop from serving policy to circularity outcomes [16].

6. Conclusions

This work delivers an operational template for reducing both the carbon and water footprints of LLM serving without compromising interactive quality. By adopting a comprehensive serving boundary, summarizing impacts with daily medians, and co-optimizing

carbon and water under explicit $p95$ latency and capacity constraints, the framework turns a fragmented literature into a deployable control policy with repeatable gains.

Across four models, the SLO-aware policy cuts comprehensive-boundary medians by about 58% in energy, about 59% in consumptive water, and about 79% in location-based CO_2. For a representative day with 500 M GPT-4o queries, totals fall from 0.344 to 0.145 GWh, from 1.196 to 0.490 ML, and from 121 to 25 t CO_2 (LB), with $p95$ SLOs satisfied in every five-minute window. These reductions stem from coordinated levers—carbon- and water-aware routing, batch right-sizing, concise token directives (α_d), and phase-aware assignment of prefill and decode—rather than from a single intervention.

The scope-reconciliation module reproduces the production-observed $\approx$0.417 ratio, which enables scope-consistent comparison between chip-only and full-stack accounting. Pareto views make the carbon–water geometry explicit and give operators a practical way to navigate trade-offs at fixed service levels. Taken together, these elements support industry adoption, transparent reporting, and continuous improvement as grids, cooling systems, and evolution of inference stacks.

To make the practical deployment steps explicit, we conclude with a concise checklist that operators can follow in production:

- Report daily medians at the comprehensive serving boundary (accelerators + host CPU/DRAM + provisioned idle, lifted by PUE). Provide accelerator-only whiskers for reconciliation so chip-only and full-stack studies remain comparable.
- Enable carbon- and water-aware geo-routing with explicit $p95$ latency gates. Right-size batches by five-minute windows. Apply concise directives (α_d) when SLO-safe to reduce decode work. Consider phase-aware placement (fast, compute-efficient cohorts for prefill, and memory-efficient or second-life cohorts for decode).
- Use site rows per region: PUE, site WUE, EWIF, and CIF. Default to location-based carbon for Scope-2 reporting. Treat market-based factors as a sensitivity when disclosed.
- Track per-profile (short/medium/long) medians for energy (Wh), consumptive water (mL, site + source), and CO_2 (g, LB by default). Mix these using your business-specific traffic shares to obtain service-level medians and daily totals, and use these to tune weights and capacity caps over time.

This checklist concretizes the path from measurement to control: instrument once at the comprehensive boundary, parameterize from provider site rows, deploy SLO-aware policies in the global load balancer and serving layer, and report per-profile medians that reflect real traffic. In practice, these steps can be adopted incrementally (routing $\rightarrow$ routing + batch $\rightarrow$ routing + batch + tokens $\rightarrow$ phase-aware placement) while maintaining user-visible quality of service.

Author Contributions: Conceptualization, J.H. and M.T.-B.; methodology, J.H.; software, J.H.; validation, J.H.; formal analysis, J.H. and M.T.-B.; investigation, J.H.; resources, J.H.; data curation, J.H. and T.K.; writing—original draft preparation, J.H., M.T.-B. and T.K.; writing—review and editing, J.H., M.T.-B. and T.K.; visualization, J.H. and T.K.; supervision, J.H. and M.T.-B.; project administration, J.H. and M.T.-B. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data is contained within the article.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

AI	Artificial Intelligence.
API	Application Programming Interface (used when referring to cross-model API benchmarks).
CIF	Carbon Intensity of the Grid (typically kg CO_2 kWh^{-1}); used for location-based emissions.
CO_2	Carbon dioxide; operational emissions are reported in grams per prompt and in tons per day.
CO_2e	Carbon-dioxide equivalent (used when referring to greenhouse-gas accounting).
CPU	Central Processing Unit (host side of serving stack).
DRAM	Dynamic Random-Access Memory (host memory included in comprehensive boundary).
EF_{MB}	Market-Based portfolio emission factor (kg CO_2e kWh^{-1}) used as a sensitivity to LB.
EWIF	Electricity–Water Intensity Factor (L kWh^{-1}) capturing off-site, generation-mix water.
$EWIF_{source}$	"Source" component of water from electricity generation in the site + source accounting.
GHG	Greenhouse Gas.
GPU	Graphics Processing Unit (accelerator).
GWh	Gigawatt-hour (10^9 Wh).
IT	Information Technology load (accelerators + host CPU/DRAM + provisioned idle).
kWh	Kilowatt-hour (10^3 Wh).
KV cache	Key–Value cache (used in decode optimizations).
LB	Location-Based (grid-average, point-of-consumption reporting for emissions; the default in this work).
LBNL	Lawrence Berkeley National Laboratory (source for PUE/WUE context).
LLM	Large Language Model.
MB	Market-Based (portfolio accounting sensitivity for emissions).
MILP	Mixed-Integer Linear Program (optimization formulation).
mL	Milliliter (10^{-3} L).
ML	Megaliter (10^6 L); in results tables, ML day^{-1} is used for daily totals.
s	Second.
PUE	Power Usage Effectiveness (facility/IT energy ratio).
QoS	Quality of Service (used when discussing interactive service constraints).
SLO	Service Level Objective (latency/throughput targets enforced in the optimizer).
Σ-Scale	The time-resolved, SLO-aware bi-objective orchestration loop proposed in the paper.
TPOT	Time Per Output Token (latency metric for decode).
TPS	Tokens Per Second (throughput metric used in capacity constraints).
TPU	Tensor Processing Unit (accelerator).
TTFT	Time To First Token (latency metric for prefill).
Wh	Watt-hour (unit for per-prompt energy).
WUE	Water Usage Effectiveness (L kWh^{-1} at the facility; site cooling).
WUE_{site}	Site-level WUE used in the site + source water formulation.
$p95$	95th-percentile statistic (used for latency and throughput SLO enforcement).

Appendix A. Per-Model Ablation Under $p95$ SLOs (70/25/5 Mix, Comprehensive Serving Boundary)

We summarize the marginal contribution of the four serving levers defined in Section 4.6—A1: routing only, A2: routing + batch, A3: routing + batch + token, A4: routing + phase split— evaluated under the same $p95$ TTFT/TPOT latency guard and capacity constraints as the main results (Equations (18)–(21)). The bars show daily medians (70/25/5 short/medium/long mix) with 95% confidence intervals at the comprehensive serving boundary (accelerators, host CPU/DRAM, and provisioned idle, lifted by PUE). Water is the scope-consistent site+source quantity, that is, Equation (3) in the Methods. These boundaries and SLO choices follow the Methods and Results sections of the manuscript.

We compute uncertainty with a nonparametric bootstrap over the 288 five-minute windows in a day. For each model and run, we resample the window indices with replacement $B = 3000$ times. Within each draw, we compute per-profile daily medians, form the mixed 70/25/5 median, and take the 2.5th and 97.5th percentiles across draws as the

95% interval. The MILP and the enumerations are deterministic. The intervals quantify within-day temporal variability rather than solver randomness. In all bar charts, the error bars are these 95% bootstrap intervals.

From baseline → A1 → A2 → A3 → A4 → all (all levers active), medians decrease monotonically in LB CO_2 and consumptive water for every model. A3 (batch + concise tokens) accounts for the largest step, and A4 (phase split) adds a smaller yet non-dominated gain, matching Section 4.6. Violin plots show the run×region distributions that underlie the bars. The violins' thick mid-section is the IQR, and the hydro vs. nuclear "anchor" geometry reproduces Section 4.4's routing frontiers. All panels enforce $p95$ SLOs and capacity gates exactly as in the Methods section.

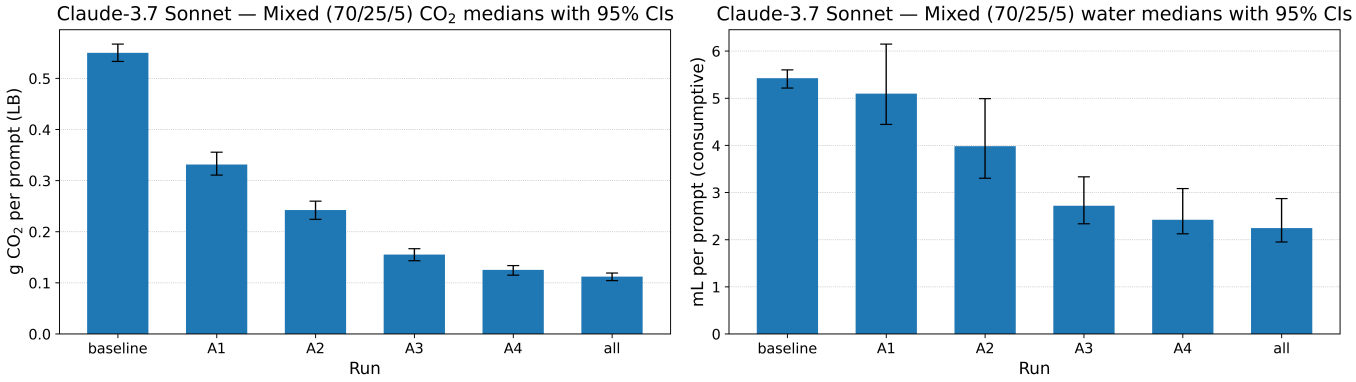


Figure A1. Claude-3.7 Sonnet—ablation bars (CO_2 and water). Left: CO_2 (g CO_2/prompt, location-based). Right: consumptive water (mL/prompt). Daily medians with 95% CIs at the comprehensive boundary with the $p95$ TTFT/TPOT SLO enforced. Sequence shows baseline, A1–A4, and all levers combined.

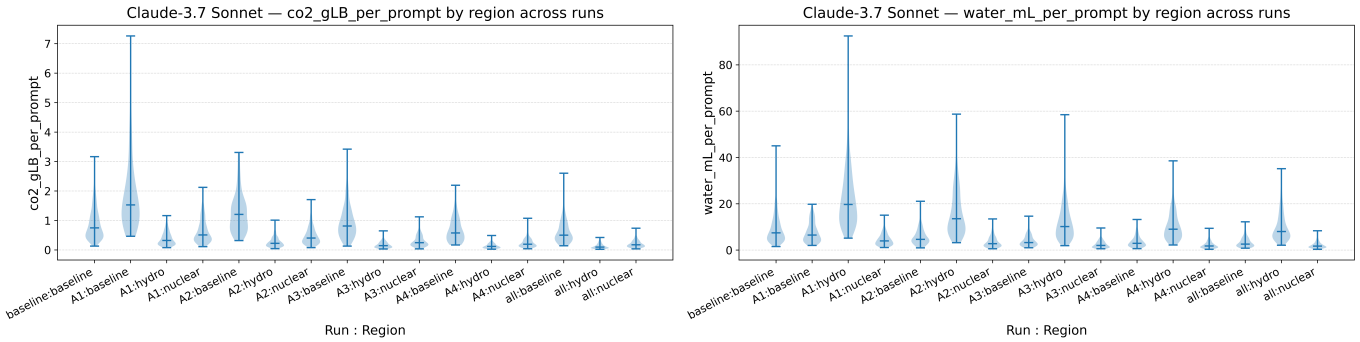


Figure A2. Claude-3.7 Sonnet—violins by region across runs. Distributions of CO_2 and water for each run and region. Hydro vertices tend to be CO_2-light and water-heavy, nuclear the opposite, matching the trade-off geometry in Sections 4.4–4.6.

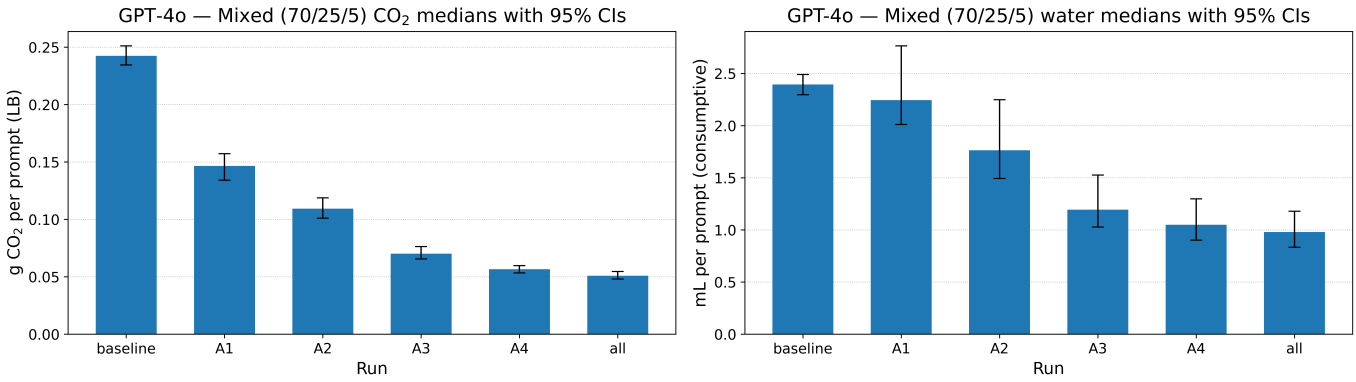


Figure A3. GPT-4o—ablation bars (CO_2 and water). Same conventions as Figure A1.

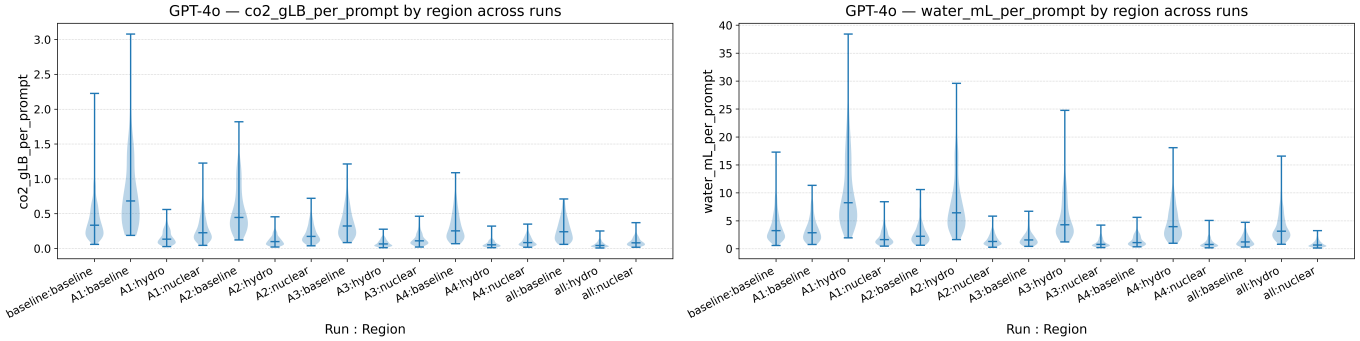


Figure A4. GPT-4o—violins by region across runs. Same conventions as Figure A2.

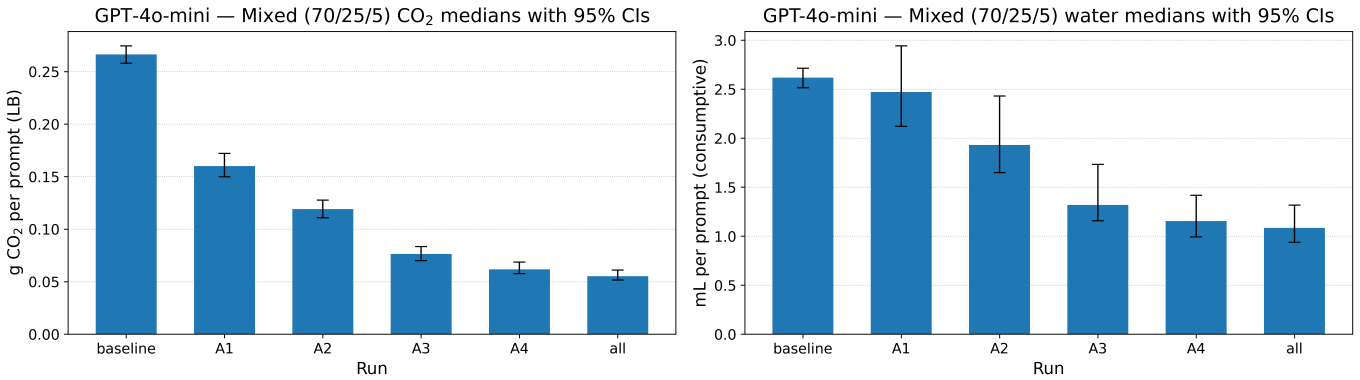


Figure A5. GPT-4o-mini—ablation bars (CO_2 and water). Same conventions as Figure A1.

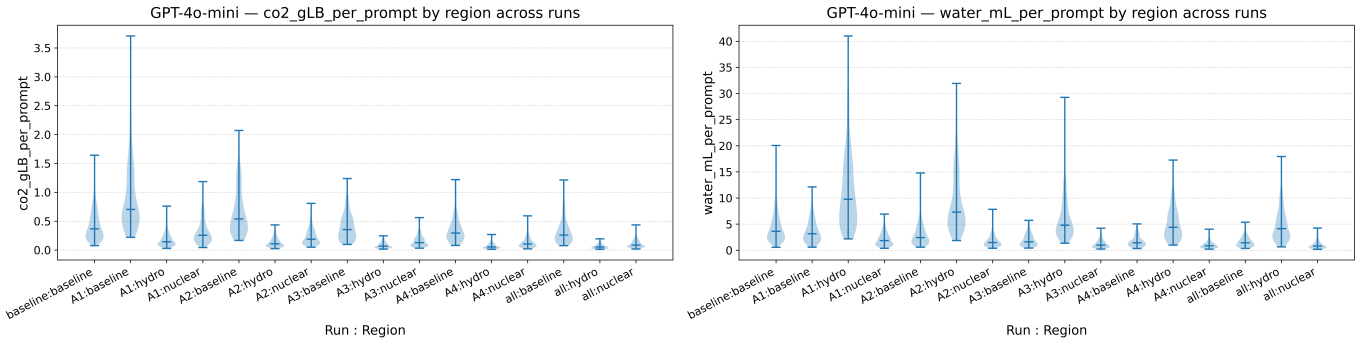


Figure A6. GPT-4o-mini—violins by region across runs. Same conventions as Figure A2.

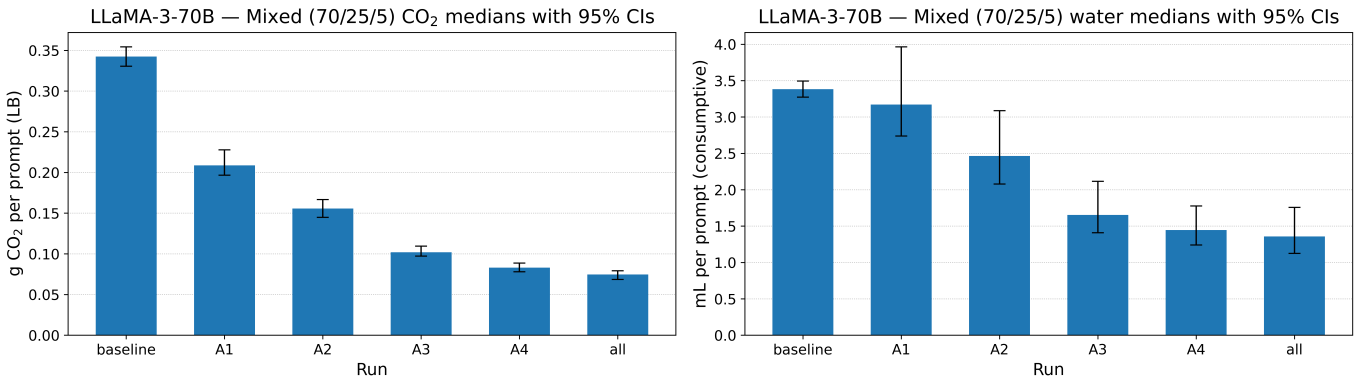


Figure A7. LLaMA-3-70B—ablation bars (CO_2 and water). Same conventions as Figure A1.

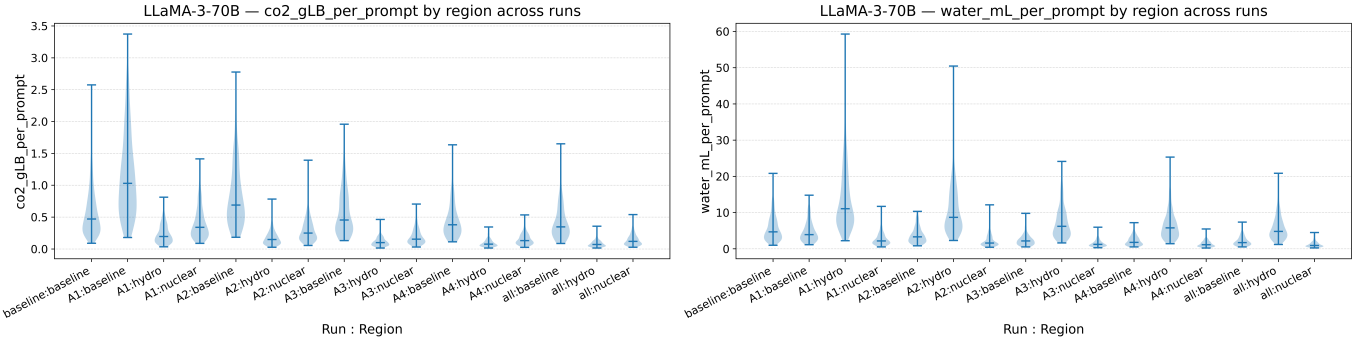


Figure A8. LLaMA-3-70B—violins by region across runs. Same conventions as Figure A2.

References

1. Elsworth, C.; Huang, K.; Patterson, D.; Schneider, I.; Sedivy, R.; Goodman, S.; Manyika, J. Measuring the environmental impact of delivering AI at Google Scale. *arXiv* **2025**, arXiv:2508.15734. [CrossRef]
2. Huang, Y. Advancing industrial sustainability research: A domain-specific large language model perspective. *Clean Technol. Environ. Policy* **2025**, *27*, 1899–1901. [CrossRef]
3. Li, S. Making AI less "thirsty": Uncovering and addressing the secret water footprint of AI models. *arXiv* **2023**, arXiv:2304.03271. [CrossRef]
4. Desislavov, R.; Martínez-Plumed, F.; Hernández-Orallo, J. Trends in AI inference energy consumption: Beyond the performance-vs-parameter laws of deep learning. *Sustain. Comput. Inform. Syst.* **2023**, *38*, 100857. [CrossRef]
5. Jegham, N.; Abdelatti, M.; Elmoubarki, L.; Hendawi, A. How hungry is AI? Benchmarking energy, water, and carbon footprint of LLM inference. *arXiv* **2025**, arXiv:2505.09598. [CrossRef]
6. Jagannadharao, A.; Beckage, N.; Nafus, D.; Chamberlin, S. Time shifting strategies for carbon-efficient long-running large language model training. *Innov. Syst. Softw. Eng.* **2025**, *21*, 517–531. [CrossRef]
7. Naveed, H.; Khan, A.U.; Qiu, S.; Saqib, M.; Anwar, S.; Usman, M.; Mian, A. A comprehensive overview of large language models. *ACM Trans. Intell. Syst. Technol.* **2025**, *16*, 1–72. [CrossRef]
8. Husom, E.J.; Goknil, A.; Shar, L.K.; Sen, S. The price of prompting: Profiling energy use in large language models inference. *arXiv* **2024**, arXiv:2407.16893.
9. Moore, H.; Qi, S.; Hogade, N.; Milojicic, D.; Bash, C.; Pasricha, S. Sustainable Carbon-Aware and Water-Efficient LLM Scheduling in Geo-Distributed Cloud Datacenters. *arXiv* **2025**, arXiv:2505.23554.
10. Chien, A.A.; Lin, L.; Nguyen, H.; Rao, V.; Sharma, T.; Wijayawardana, R. Reducing the Carbon Impact of Generative AI Inference (today and in 2035). In Proceedings of the 2nd Workshop on Sustainable Computer Systems (HotCarbon '23), Boston, MA, USA, 9 July 2023; pp. 1–7.
11. Argerich, M.F.; Patiño-Martínez, M. Measuring and improving the energy efficiency of large language models inference. *IEEE Access* **2024**, *12*, 80194–80207. [CrossRef]
12. De Vries, A. The growing energy footprint of artificial intelligence. *Joule* **2023**, *7*, 2191–2194. [CrossRef]
13. Luccioni, A.S.; Viguier, S.; Ligozat, A.L. Estimating the carbon footprint of BLOOM, a 176B parameter language model. *J. Mach. Learn. Res.* **2023**, *24*, 1–15.
14. Jiang, Y.; Roy, R.B.; Kanakagiri, R.; Tiwari, D. WaterWise: Co-optimizing Carbon-and Water-Footprint Toward Environmentally Sustainable Cloud Computing. In Proceedings of the PPoPP '25: 30th ACM SIGPLAN Annual Symposium on Principles and Practice of Parallel Programming, Las Vegas, NV, USA, 1–5 March 2025; pp. 297–311.
15. Islam, M.A.; Ren, S.; Quan, G.; Shakir, M.Z.; Vasilakos, A.V. Water-constrained geographic load balancing in data centers. *IEEE Trans. Cloud Comput.* **2015**, *5*, 208–220. [CrossRef]
16. Schneider, I.; Xu, H.; Benecke, S.; Patterson, D.; Huang, K.; Ranganathan, P.; Elsworth, C. Life-cycle emissions of AI hardware: A cradle-to-grave approach and generational trends. *arXiv* **2025**, arXiv:2502.01671.
17. Wu, Y.; Hua, I.; Ding, Y. Unveiling environmental impacts of large language model serving: A functional unit view. *arXiv* **2025**, arXiv:2502.11256. [CrossRef]
18. Cheng, K.; Wang, Z.; Hu, W.; Yang, T.; Li, J.; Zhang, S. SCOOT: SLO-Oriented Performance Tuning for LLM Inference Engines. In Proceedings of the Web Conference 2025, Sydney, NSW, Australia, 28 April–2 May 2025; pp. 829–839.
19. Wu, C.J.; Raghavendra, R.; Gupta, U.; Acun, B.; Ardalani, N.; Maeng, K.; Chang, G.; Behram, F.A.; Huang, J.; Bai, C.; et al. Sustainable AI: Environmental implications, challenges and opportunities. In Proceedings of the Machine Learning and Systems, Santa Clara, CA, USA, 29 August–1 September 2022; Volume 4, pp. 795–813.

20. Samsi, S.; Zhao, D.; McDonald, J.; Li, B.; Michaleas, A.; Jones, M.; Bergeron, W.; Kepner, J.; Tiwari, D.; Gadepally, V. From words to watts: Benchmarking the energy costs of large language model inference. In Proceedings of the 2023 IEEE High Performance Extreme Computing Conference (HPEC), Wakefield, MA, USA, 15–19 September 2025; pp. 1–9.
21. Wiesner, P.; Grinwald, D.; Weiß, P.; Wilhelm, P.; Khalili, R.; Kao, O. Carbon-Aware Quality Adaptation for Energy-Intensive Services. In Proceedings of the 16th ACM International Conference on Future and Sustainable Energy Systems, Rotterdam, The Netherlands, 17–20 June 2025; pp. 415–422.
22. Nguyen, S.; Zhou, B.; Ding, Y.; Liu, S. Towards sustainable large language model serving. *ACM Sigenergy Energy Inform. Rev.* **2024**, *4*, 134–140. [CrossRef]
23. Falk, S.; Ekchajzer, D.; Pirson, T.; Lees-Perasso, E.; Wattiez, A.; Biber-Freudenberger, L.; van Wynsberghe, A. More than Carbon: Cradle-to-Grave environmental impacts of GenAI training on the Nvidia A100 GPU. *arXiv* **2025**, arXiv:2509.00093.
24. Mistral AI. Our Contribution to a Global Environmental Standard for AI. 2025. Available online: https://mistral.ai/news/our-contribution-to-a-global-environmental-standard-for-ai (accessed on 17 November 2025).
25. Soares, I.V.; Yarime, M.; Klemun, M.M. Estimating GHG emissions from cloud computing: Sources of inaccuracy, opportunities and challenges in location-based and use-based approaches. *Clim. Policy* **2025**, *25*, 1335–1353. [CrossRef]
26. Anquetin, T.; Coqueret, G.; Tavin, B.; Welgryn, L. Scopes of carbon emissions and their impact on green portfolios. *Econ. Model.* **2022**, *115*, 105951. [CrossRef]
27. Różycki, R.; Solarska, D.A.; Waligóra, G. Energy-Aware Machine Learning Models—A Review of Recent Techniques and Perspectives. *Energies* **2025**, *18*, 2810. [CrossRef]
28. Fu, Z.; Chen, F.; Zhou, S.; Li, H.; Jiang, L. LLMCO2: Advancing accurate carbon footprint prediction for LLM inferences. *ACM Sigenergy Energy Inform. Rev.* **2025**, *5*, 63–68. [CrossRef]
29. Daraghmeh, H.M.; Wang, C.C. A review of current status of free cooling in datacenters. *Appl. Therm. Eng.* **2017**, *114*, 1224–1239. [CrossRef]
30. Ebrahimi, K.; Jones, G.F.; Fleischer, A.S. A review of data center cooling technology, operating conditions and the corresponding low-grade waste heat recovery opportunities. *Renew. Sustain. Energy Rev.* **2014**, *31*, 622–638. [CrossRef]
31. Mytton, D. Data centre water consumption. *NPJ Clean Water* **2021**, *4*, 8. [CrossRef]
32. Sharma, N.; Mahapatra, S.S. A preliminary analysis of increase in water use with carbon capture and storage for Indian coal-fired power plants. *Environ. Technol. Innov.* **2018**, *9*, 51–62. [CrossRef]
33. Chlela, S.; Selosse, S. Water use in a sustainable net zero energy system: What are the implications of employing bioenergy with carbon capture and storage? *Int. J. Sustain. Energy Plan. Manag.* **2024**, *40*, 146–162. [CrossRef]
34. Chung, J.W.; Liu, J.; Ma, J.J.; Wu, R.; Liu, J.; Kweon, O.J.; Xia, Y.; Wu, Z.; Chowdhury, M. The ML.ENERGY Benchmark: Toward Automated Inference Energy Measurement and Optimization. *arXiv* **2025**, arXiv:2505.06371 [CrossRef]
35. Luccioni, S.; Gamazaychikov, B. AI Energy Score Leaderboard. 2025. Available online: https://huggingface.co/spaces/AIEnergyScore/Leaderboard (accessed on 17 November 2025).
36. Sarkar, S.; Naug, A.; Luna, R.; Guillen, A.; Gundecha, V.; Ghorbanpour, S.; Babu, A.R. Carbon footprint reduction for sustainable data centers in real-time. In Proceedings of the AAAI Conference on Artificial Intelligence, Vancouver, BC, Canada, 26–27 February 2024; Volume 38, pp. 22322–22330.
37. Mondal, S.; Faruk, F.B.; Rajbongshi, D.; Efaz, M.M.K.; Islam, M.M. GEECO: Green data centers for energy optimization and carbon footprint reduction. *Sustainability* **2023**, *15*, 15249. [CrossRef]
38. Riepin, I.; Brown, T.; Zavala, V.M. Spatio-temporal load shifting for truly clean computing. *Adv. Appl. Energy* **2025**, *17*, 100202. [CrossRef]
39. Rahman, A.; Liu, X.; Kong, F. A survey on geographic load balancing based data center power management in the smart grid environment. *IEEE Commun. Surv. Tutor.* **2013**, *16*, 214–233. [CrossRef]
40. Cao, Z.; Zhou, X.; Hu, H.; Wang, Z.; Wen, Y. Toward a systematic survey for carbon neutral data centers. *IEEE Commun. Surv. Tutor.* **2022**, *24*, 895–936. [CrossRef]
41. Islam, M.A.; Mahmud, H.; Ren, S.; Wang, X. A carbon-aware incentive mechanism for greening colocation data centers. *IEEE Trans. Cloud Comput.* **2017**, *8*, 4–16. [CrossRef]
42. Wiesner, P.; Behnke, I.; Scheinert, D.; Gontarska, K.; Thamsen, L. Let's wait awhile: How temporal workload shifting can reduce carbon emissions in the cloud. In Proceedings of the 22nd International Middleware Conference, Virtual Event, 6–10 December 2021; pp. 260–272.
43. Silva, C.A.; Vilaça, R.; Pereira, A.; Bessa, R.J. A review on the decarbonization of high-performance computing centers. *Renew. Sustain. Energy Rev.* **2024**, *189*, 114019. [CrossRef]
44. Radovanović, A.; Koningstein, R.; Schneider, I.; Chen, B.; Duarte, A.; Roy, B.; Cirne, W. Carbon-aware computing for datacenters. *IEEE Trans. Power Syst.* **2022**, *38*, 1270–1280. [CrossRef]
45. Faiz, A.; Kaneda, S.; Wang, R.; Osi, R.; Sharma, P.; Chen, F.; Jiang, L. LLMCarbon: Modeling the end-to-end carbon footprint of large language models. *arXiv* **2023**, arXiv:2309.14393.

46. Patel, P.; Choukse, E.; Zhang, C.; Shah, A.; Goiri, Í.; Maleki, S.; Bianchini, R. Splitwise: Efficient Generative LLM Inference Using Phase Splitting. In Proceedings of the 2024 ACM/IEEE 51st Annual International Symposium on Computer Architecture (ISCA), Buenos Aires, Argentina, 29 June–3 July 2024; pp. 118–132.
47. Fan, H.; Lin, Y.C.; Prasanna, V. ELLIE: Energy-Efficient LLM Inference at the Edge Via Prefill-Decode Splitting. In Proceedings of the 2025 IEEE 36th International Conference on Application-specific Systems, Architectures and Processors (ASAP), Vancouver, BC, Canada,28–30 July 2025; pp. 139–146.
48. Zhu, K.; Gao, Y.; Zhao, Y.; Zhao, L.; Zuo, G.; Gu, Y.; Kasikci, B. NanoFlow: Towards Optimal Large Language Model Serving Throughput. In Proceedings of the 19th USENIX Symposium on Operating Systems Design and Implementation (OSDI 25), Boston, MA, USA, 7–9 July 2025; pp. 749–765.
49. Zhong, Y.; Liu, S.; Chen, J.; Hu, J.; Zhu, Y.; Liu, X.; Jin, H.; Zhang, H. DistServe: Disaggregating prefill and decoding for goodput-optimized large language model serving. In Proceedings of the 18th USENIX Symposium on Operating Systems Design and Implementation (OSDI 24), Santa Clara, CA, USA, 10–12 July 2024; pp. 193–210.
50. Feng, J.; Huang, Y.; Zhang, R.; Liang, S.; Yan, M.; Wu, J. WindServe: Efficient Phase-Disaggregated LLM Serving with Stream-based Dynamic Scheduling. In Proceedings of the 2nd Annual International Symposium on Computer Architecture, Tokyo, Japan, 21–25 June 2025; pp. 1283–1295.
51. Svirschevski, R.; May, A.; Chen, Z.; Chen, B.; Jia, Z.; Ryabinin, M. Specexec: Massively parallel speculative decoding for interactive LLM inference on consumer devices. *Adv. Neural Inf. Process. Syst.* **2024**, *37*, 16342–16368.
52. Liu, A.; Liu, J.; Pan, Z.; He, Y.; Haffari, G.; Zhuang, B. MiniCache: KV cache compression in depth dimension for large language models. *Adv. Neural Inf. Process. Syst.* **2024**, *37*, 139997–140031.
53. Wang, Y.; Chen, K.; Tan, H.; Guo, K. Tabi: An efficient multi-level inference system for large language models. In Proceedings of the Eighteenth European Conference on Computer Systems, Rome, Italy, 8–12 May 2023; pp. 233–248.
54. Ahmadpanah, S.H.; Sobhanloo, S.; Afsharfarnia, P. Dynamic token pruning for LLMs: Leveraging task-specific attention and adaptive thresholds. *Knowl. Inf. Syst.* **2025**, *67*, 7431–7450. [CrossRef]
55. Belhaouari, S.B.; Kraidia, I. Efficient self-attention with smart pruning for sustainable large language models. *Sci. Rep.* **2025**, *15*, 10171. [CrossRef]
56. Jiang, Y.; Roy, R.B.; Li, B.; Tiwari, D. Ecolife: Carbon-aware serverless function scheduling for sustainable computing. In Proceedings of the SC24: International Conference for High Performance Computing, Networking, Storage and Analysis, Atlanta, GA, USA, 17–22 November 2024; pp. 1–15.
57. Li, B.; Jiang, Y.; Gadepally, V.; Tiwari, D. SPROUT: Green generative AI with carbon-efficient LLM inference. In Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, Miami, FL, USA, 12–16 November 2024; pp. 21799–21813.
58. Kim, H.; Young, S.; Chen, X.; Gupta, U.; Hester, J. Slower is Greener: Acceptance of Eco-feedback Interventions on Carbon Heavy Internet Services. *ACM J. Comput. Sustain. Soc.* **2025**, *3*, 1–21. [CrossRef]
59. Jiang, P.; Sonne, C.; Li, W.; You, F.; You, S. Preventing the immense increase in the life-cycle energy and carbon footprints of LLM-powered intelligent chatbots. *Engineering* **2024**, *40*, 202–210. [CrossRef]
60. Morsy, M.; Znid, F.; Farraj, A. A critical review on improving and moving beyond the 2 nm horizon: Future directions and impacts in next-generation integrated circuit technologies. *Mater. Sci. Semicond. Process.* **2025**, *190*, 109376. [CrossRef]
61. Wang, P.; Zhang, L.Y.; Tzachor, A.; Chen, W.Q. E-waste challenges of generative artificial intelligence. *Nat. Comput. Sci.* **2024**, *4*, 818–823. [CrossRef] [PubMed]
62. Shehabi, A.; Smith, S.J.; Hubbard, A.; Newkirk, A.; Lei, N.; Siddik, M.A.B.; Holecek, B.; Koomey, J.G.; Masanet, E.; Sartor, D.A. *2024 United States Data Center Energy Usage Report (LBNL-2001637)*; Technical Report; Lawrence Berkeley National Laboratory: Berkeley, CA, USA, 2024.
63. Kasprzyk, J.R.; Nataraj, S.; Reed, P.M.; Lempert, R.J. Many objective robust decision making for complex environmental systems undergoing change. *Environ. Model. Softw.* **2013**, *42*, 55–71. [CrossRef]
64. Marler, R.T.; Arora, J.S. Survey of multi-objective optimization methods for engineering. *Struct. Multidiscip. Optim.* **2004**, *26*, 369–395. [CrossRef]
65. Mavrotas, G. Effective implementation of the ε-constraint method in multi-objective mathematical programming problems. *Appl. Math. Comput.* **2009**, *213*, 455–465. [CrossRef]
66. Tamiz, M.; Jones, D.F.; El-Darzi, E.S. Goal programming for decision making: An overview of the current state-of-the-art. *Eur. J. Oper. Res.* **1998**, *111*, 569–581. [CrossRef]
67. Pati, R.K.; Vrat, P.; Kumar, P. A goal programming model for the paper recycling system. *Omega* **2008**, *36*, 405–417. [CrossRef]

68. Eskandarpour, M.; Dejax, P.; Miemczyk, J.; Péton, O. Sustainable supply chain network design: An optimization-oriented review. *Omega* **2015**, *54*, 11–32. [CrossRef]
69. Li, P.; Yang, J.; Wierman, A.; Ren, S. Towards environmentally equitable AI via geographical load balancing. In Proceedings of the 15th ACM International Conference on Future and Sustainable Energy Systems, Singapore, 4–7 June 2024; pp. 291–307.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.