XWind: A Cross-site Router for Large Language Model Inference Serving at Renewable Energy Farms

Tella Rajashekhar Reddy, Atharva Deshmukh, Liangcheng Yu, Chaojie Zhang, Mike Shepperd, Rohan Gandhi,
Anjaly Parayil, Srinivasan Iyengar, Ajay Manchepalli, Debopam Bhattacherjee
Microsoft

Abstract

AI power demand is growing at an unprecedented rate while power grids are often ailing and struggle to keep up. Grid expansion comes with high capital expenditure and long-distance transmission losses, yet there is *abundant* renewable energy at the source, just not matched to demand.

This paper proposes a complementary AI infrastructure deployment model, *AI Greenferencing*, that brings modular AI compute to renewable energy sources, focusing on wind, allowing AI footprint expansion, generating local behind-the-meter demand for renewable sites, and helping ease the growing strain on power utilities. Our feasibility analysis shows that 890+ GW of wind capacity lies within 50 ms network round trip time of Azure data centers, and that site-wise right-sizing combined with spatial complementarity of wind energy keeps aggregate fleet utilization on par with traditional deployments.

To serve inference requests under variable wind power, we build XWind, a lightweight, reactive, and workload-agnostic AI inference router that uses only real-time signals: inference latency, KV-cache utilization, and queue depth, to dynamically configure sites and distribute requests. Evaluated on a real 64-GPU A100 testbed emulating three wind-powered sites with Azure production traces, XWind reduces P99 end-to-end latency by up to 52% over the strongest contender (also our idea) and by up to 98% over baselines such as power-capping and GPU idling, with consistent gains across workload types, load levels, and GPU generations.

1 Introduction

AI adoption is accelerating across the industry, governments, and individuals [20, 35, 67], and so is the energy bill. The IEA estimates [46] that global data center electricity consumption reached 415 TWh in 2024, roughly 1.5% of global demand, and projects it to more than double to 945 TWh by 2030, on par with Japan's electricity use. A key driver is the increasing power density of AI hardware [39, 57, 58], with rack scale demands shooting up well above 100 kW, thus significantly inflating data center scale demands. AI inferencing, which accounts for 90% of AI compute today [21, 59], is the dominant and fastest-growing segment of this demand.

Seeing this surge, hyperscalers have announced partnerships [23, 26, 45, 62] with energy providers to secure power. Unfortunately, there isn't a silver bullet that addresses the growing demand and ailing delivery. First, expanding grid infrastructure: new transmission lines, distribution systems, or energy storage is capital-intensive [29, 32, 72], often faces regulatory and logistic delays [48], and is especially difficult when renewable sources are located far from consumption hubs [44, 73]. A recent Berkeley Lab report [48] highlights that by 2024, pending grid approvals for new power generation exceeded twice the installed US capacity, with a median wait time of 4.5–5 years. Second, even approved projects often face curtailment due to grid congestion, leaving clean, already generated power underutilized [17, 30, 77]. Third, long-distance transmission and distribution (T&D) losses significantly inflate the cost of electricity [32, 72]. For example, the EIA reports [75] US industrial rates of 9.3 ¢/kWh, while wind farms sell at 2.3-4.5 ¢/kWh [49, 56] at the source. Finally, much of the grid infrastructure is aging [48, 52] and operators may be resorting to short-term measures [28, 64] that compromise long-term sustainability.

Several strategies have emerged to address these concerns. On-site fossil generation, such as gas turbines and fuel cells [9, 10, 34], offers speed but at the cost of carbon emissions. Nuclear energy promises clean baseload, with hyperscalers signing multi-gigawatt deals [2, 6, 38], but most deployments face regulatory hurdles and construction timelines stretching beyond 2030 [3] while the demand is *real* and *now*. Space-based computing [5, 65] leverages perpetual solar energy above clouds, but remains an early-stage, costly demonstration with significant deployment challenges.

This work is motivated by a more immediate lever: consuming renewable energy where it is generated *today*. Renewable energy farms produce power that could benefit from local on-site demand to cover grid uncertainties (interconnection queues [48] and curtailment [17, 55, 74]). Co-located compute startups [4, 7, 8] have begun deploying at such sites for crypto-mining and content streaming, but LLM inferencing, the dominant and fastest-growing AI workload, presents a far larger and yet unexplored opportunity.

AI Greenferencing. We propose AI Greenferencing that co-locates modular AI compute deployments at existing or upcoming (also, otherwise queued) renewable energy sites (to start with, wind farms), as seen in Fig. 1 (green boxes on the right), and helps run a significant share of AI inferencing workload sustainably at a lower cost of energy (no T&D loss or CAPEX). This deployment strategy aligns with the growing need for 'community-first' AI infrastructure [33] that benefits host communities. AI Greenferencing is broadly

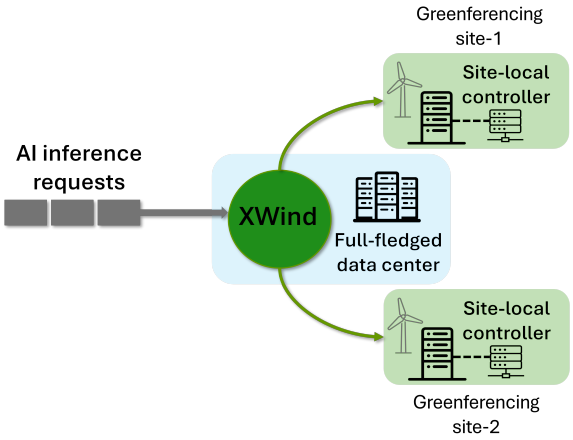


Figure 1. *AI Greenferencing with* XWind.

a win-for-all: (1) users gain access to sustainable AI services; (2) AI providers unlock additional compute capacity, user reach, and revenue; (3) wind farms can monetize output locally; and (4) power grids benefit from reduced load and a breather for expansion. Note that Greenferencing is designed not to obviate but to co-exist with traditional data centers as a complementary deployment model. Although probably intuitive in hindsight, realizing AI Greenferencing requires navigating several practical concerns that we discuss below.

Enough wind power to make a dent? We find that there is already a significant wind capacity (only 100+ MW large deployments considered, operating and under-construction) of 890+ GW globally, as per the Global Energy Monitor data [36] within 50 ms fiber (circuitous routes, slower speed-of-light than in air) round-trip time (RTT) of Azure DCs. 73% of this capacity is within 20 ms RTT. While our deployment right-sizing strategy is to deploy compute with peak demand at a low percentile (not the peak) of a site's generation, more than 10 million NVIDIA H100 equivalents could be deployed *today* in wind farms with Greenferencing.

Power variations a problem? We find that wind power, although variable, is highly predictable on 15-min timescales (autocorrelation > 0.99 [31, 81]), thus assisting the scheduler in making informed decisions while routing online. Furthermore, geographically dispersed wind sites exhibit spatial complementarity [13, 69] in generation. Cross-country combinations reduce the coefficient of variation by up to 36% [81] allowing to route around transient power drops.

XWind for AI Greenferencing. When co-locating GPU farms with wind sites, a key challenge is to serve AI inferencing requests under variable power and workload arrival. Existing energy-efficient schedulers [68] assume stable grid power and drop up to 50% of requests under power dips in our cross-site experiments, even where none of the drops is necessary with intelligent re-routing. To realize the Greenferencing vision, we design XWind (Fig. 1), a lightweight, reactive, and profiling-free power-variability-aware cross-site AI inference router. XWind works in tandem with site-local controllers (XW-Slc) that leverage online telemetry (KV-cache utilization, queue depth, and inferencing latency) to reconfigure sites during rare windows of power crunch. These controllers share useful reconfiguration and telemetry signals with XWind that could then intelligently re-route around constrained sites.

This paper makes the following contributions:

- **We propose AI Greenferencing**, a regionally geo-distributed deployment model that co-locates AI compute at wind farms behind-the-meter. Our opportunity analysis shows this is not a blip: 10+ million H100 GPU equivalents could be deployed today within a few tens of milliseconds of Azure data centers (§2). We are also working closely with a large renewable energy company that sees significant value in this strategy.
- **We design and build** XWind **cross-site router** for Greenferencing. XWind (§4) is lightweight, power-variability-aware, and uses only real-time telemetry signals, while efficiently routing AI inference requests acorss multiple variable-power sites.
- **We evaluate** XWind **on a** 64**-GPU** A100 **testbed** emulating three wind-powered Greenferencing sites, the first hardware demonstration of inference serving under variable renewable power. XWind reduces P99 end-to-end (E2E) latency by 22–52% over the strongest contender (also our idea) and by up to 98% over baselines (§5).

2 Feasibility Analyses

AI compute is increasingly power-dense [22, 39, 59], with the US data center power demand growing at 10–15% CAGR[50], approaching a significant fraction of residential consumption. This risks overwhelming aging grid infrastructure and forcing short-term decisions[28, 64] that undermine long-term sustainability. Wind farms offer a massive, yet underutilized alternative. As of February 2026, the global wind capacity pipeline stands at 3 TW [36]. However, much of this capacity remains stranded in interconnection queues[48], and even operating farms face curtailment due to grid congestion [17].

This section quantifies the Greenferencing opportunity that helps AI consume this green energy at its source: §2.1 assesses reachable capacity and economics; §2.2 addresses power and workload fluctuations; and §2.3 shows how to minimize lost compute cycles.

2.1 Wind Capacity and Economic Viability

The search for wind farms within a reasonable network distance from the data centers is crucial. We need to slightly digress here and first understand the AI inference latency components [68] with SLOs (service level objectives): Time To First Token (TTFT, few 100 ms to seconds) and Time Between Tokens (TBT, lower). E2E latency encompasses request

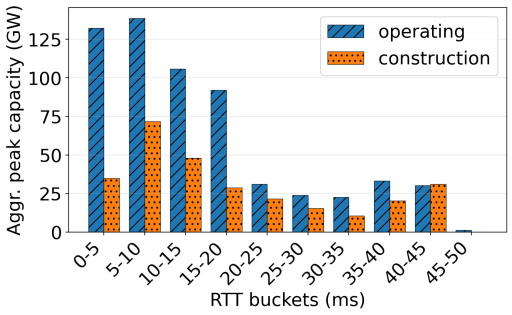


Figure 2. *Massive wind capacity (for large* 100+ *MW farms) lies within* 20 *ms fiber RTT of* Azure *data centers.*

queue time, TTFT, and TBT. Network latency primarily affects TTFT, not TBT (tokens are streamed). Using the Global Energy Monitor dataset [36]. we find that 890+ GW of operating and under-construction wind capacity (100+ MW farms only) lies within 50 ms fiber RTT of Azure DCs, with 73% within 20 ms (Fig. 2) thus not significantly affecting TTFT (and hence the E2E latency). Similar analysis with another hyperscaler's data centers reveals comparable proximity. Being able to tap into even 2% of this generation (often curtailed/queued) unlocks additional capacity larger than today's largest data centers.

Makes economic sense? CAPEX is comparable: infrastructure costs have a similar breakeven as modular data centers [18] that can be mass-produced at scale [24, 27, 43], and GPU costs are identical regardless of location. On the OPEX side, higher maintenance costs from the distributed setup [82] are easily offset by 2-4× lower power costs at source: the EIA [75] reports US industrial rates of 9.3¢/kWh, whereas wind PPAs are 2.3-4.5 ¢/kWh [49, 56]. Whether utilization at these sites can match traditional deployments is addressed in §2.3.

2.2 Operational Feasibility

AI Greenferencing must tame uncertainties at both ends: wind power generation and inference workload arrival.

Wind is predictable. Wind power, albeit variable, is predictable: a characteristic the AI Greenferencing software can leverage when routing workload across wind RE sites. Across 4 combinations of wind regions (Wallonia, Flanders) and power grids (Elia, Dso) at 15min granularity (Jan–Jul'24, ELIA [31]), the mean autocorrelation at a lag of 1 is 0.991. Across 235 wind farms in the EMHIRES dataset [81], the mean (median) autocorrelation at a lag of 1 is 0.99 over 1 year (2018–19) of hourly generation data. These scores confirm strong predictability at different temporal granularities with time series or ML-based models that can consume additional features such as historical data, seasonality, local weather, and turbine specifications. The industry uses standard predictors like TFT [51] (Google) and DeepMC [47] (Microsoft)

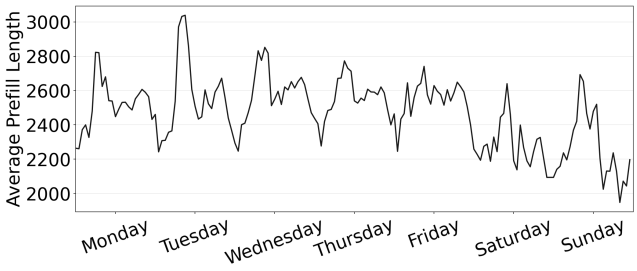


Figure 3. Azure *coding trace: average prefill length varies significantly over a week.*

for wind power prediction with very high accuracy. The broad availability of such predictors [1] helps us treat wind power generation as an oracle (variable yet predictable with high accuracy) in Greenferencing systems design.

Spatial complementarity smooths variability. Wind generation across geographically dispersed sites exhibits complementarity, when one site has low wind, others still tend to have high wind intensity. In the EMHIRES dataset, combinations of 4 cross-country sites (e.g., Iceland, Norway, Switzerland, UK) reduce the coefficient of variation (CoV) of aggregate generation by 36% compared to a single site [81].

Workload is harder to characterize than power. Wind power varies continuously but predictably. In contrast, AI inference workloads exhibit high variability in prefill lengths, decode lengths, and their ratios across workload types, shifting over time (Fig.3), with arrival patterns disrupted by flash crowds [16, 63]. Proactively predicting workload characteristics, as optimization-based routers require, demands offline profiling per workload type and accurate output-length predictors, both hard to maintain in production. A reactive approach is feasible: memory pressure, latency, and queue buildup are directly observable through runtime telemetry regardless of workload composition. This calls for a design that is *proactive on power* and *reactive on workload.*

2.3 Compute Feasibility

Another aspect of deploying GPU compute at these variable power wind sites is the potential loss of some compute cycles. Multiple guardrails mitigate this risk: *right-sizing* deploys compute at a conservative percentile (e.g., 20th) of each site's peak generation; *batteries* bridge transient dips before the router acts; *additional modalities* like solar complement wind; sites can *opportunistically tap the grid* during sustained shortfalls; and *cross-site complementarity* helps redistribute inference load to sites with available power, exploiting the spatial diversity discussed in §2.2.

How often does aggregate power fall short? To quantify this, we use the EMHIRES dataset [81] with 1 year of hourly wind generation across European NUTS2 regions. For each

[1] Our in-house framework (orthogonal work) yields a 20% relative improvement in prediction error over state-of-the-art predictors.

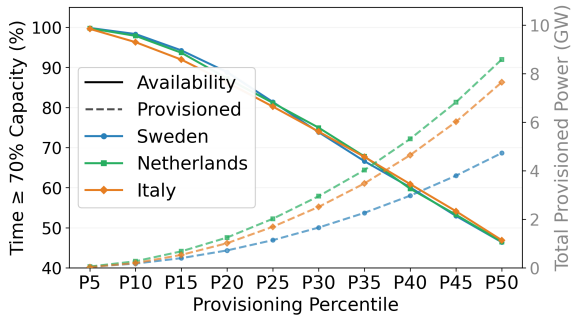


Figure 4. *Availability vs. provisioning tradeoff for* 3 Azure *DCs at* 20 *ms RTT.*

site within 20ms fiber RTT of a Azure DC, we compute the x^{th} percentile of its generation time series, cap output at that percentile, and sum capped power across all sites. Fig. 4 shows the resulting tradeoff for 3 geographically diverse Azure DCs in the EU spanning 15 degrees of latitude: Sweden (60°N), Netherlands (52°N), and Italy (45°N).

At the 20th percentile, the fleet stays above 70% of provisioned power 87-89% of the time; rare dips are absorbed by batteries, grid access, and the cross-site router. Lower percentiles (P10) yield higher availability (> 97%) with less capacity; higher ones (P30) unlock more compute at reduced guarantees. Since guardrails bridge infrequent shortfalls, average GPU utilization matches the 70-90% reported in traditional data centers [25, 53]. At P20 Greenferencing could already enroll today 10+ million H100 equivalents.

Having established economic viability and power availability, the remaining challenge is serving AI inference under residual power variability. This motivates a cross-site inference request router that uses power predictions and spatial complementarity and is reactive to workload variability.

3 AI Inference Under Constrained Power

Unlike traditional data centers, Greenferencing operates under variable power availability and must carefully navigate the tradeoff between constrained power and AI inference performance. This section characterizes the action space and its workload impact, informing the router design.

3.1 Greenferencing Site-local Knobs

Large language model inference is autoregressive: an input query is absorbed during the *prefill* phase, after which the model generates one token at a time autoregressively in the *decode* phase. The KV-cache stores key-value tensors for previously processed tokens, enabling reuse during generation. The KV-cache occupancy increases with the number of in-flight tokens and sequence lengths.

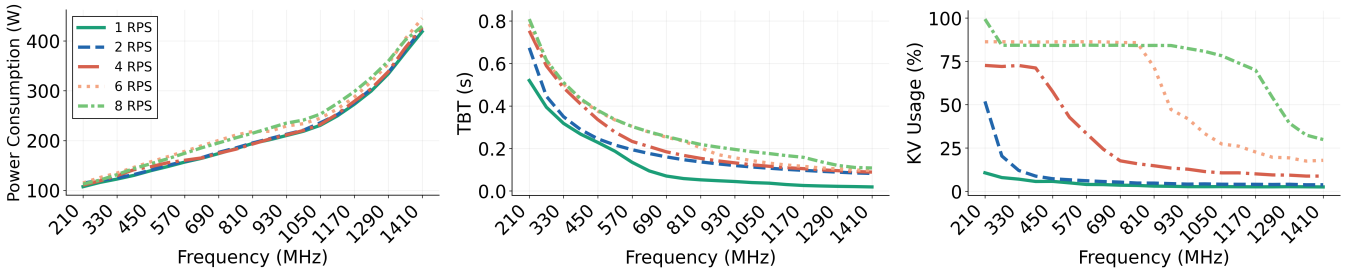
An inference system under a power budget can adjust two primary knobs at each site[2]: (1)*Active node count*: nodes can be idled (~30% static power tax, instant readiness) or shut down (saves power but minutes of boot latency). Fewer active nodes increases load on remaining instances, queues requests, inflates E2E latency, and shrinks the aggregate KV-cache pool, raising memory pressure risk. (2)*GPU frequency*: adjustable in milliseconds via `nvidia-smi`. Lower frequencies reduce throughput, increase TBT, and raise KV-cache occupancy, which, as shown in §3.2, can trigger sharp latency degradation beyond a frequency-dependent tipping point. A third knob, tensor parallelism (TP) degree, takes seconds to minutes to reconfigure; we fix TP at deployment to avoid re-sharding overhead. In a multi-site Greenferencing setting with variable renewable power, the challenge is to dynamically select the right <node count, frequency> configuration at each site while distributing requests across sites to meet latency SLOs, an online resource allocation problem. There is still a fourth knob, power-capping, which can be set using `-pl` in `nvidia-smi`.

3.2 Characterizing LLM Inference

Designing a robust control plane requires a precise understanding of the underlying hardware's power and performance dynamics. Toward this, we hosted the Llama3.1-8B model on two NVIDIA A100 (40 GB) GPUs with tensor parallelism (TP2) using vLLM v1 [11] serving engine (similar results for H100 GPUs also discussed below). To enable detailed performance analysis, we modified the vLLM code to log latency metrics outside the critical path. The input workloads were constructed using prefill (P) and decode (D) values from the Azure coding and conversation datasets [19] each with a distinct P/D ratio. Fig. 5 shows results for the conversation workload; coding exhibits consistent trends and is omitted for brevity. Request arrivals follow Poisson distributions, and each experiment ran for 30 minutes. In parallel, we collected fine-grained system metrics using DCGMI [1] with samples taken every 50 ms. We evaluated the system across a wide range of RPS values, scaling up until saturation, and under multiple GPU frequency settings. This exercise focuses on power consumption, inference latency, and KV-cache utilization, metrics that directly govern the feasibility of dynamic scaling. We make the following observations:

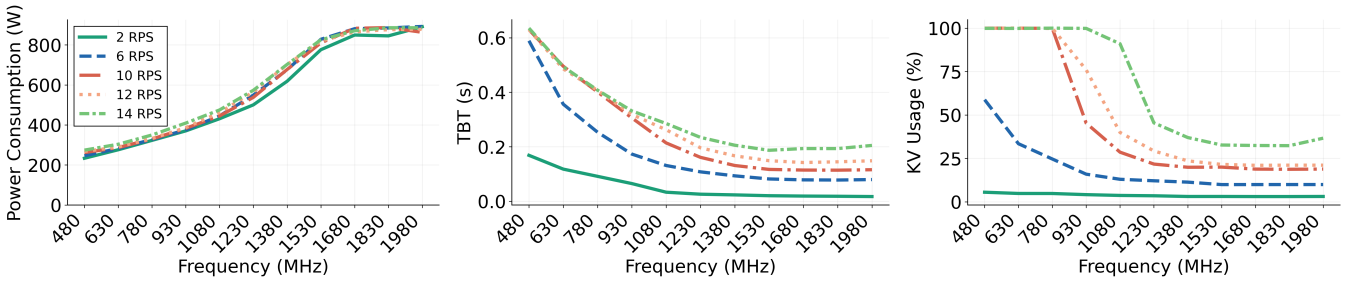
O1. Frequency versus peak power: Frequency downclocking is a deterministic knob to restrict power consumption. We stress-tested the system by saturating GPU utilization with high request loads; the power recorded via DCGMI represents peak power at each frequency. A key observation from Fig.5a is that the frequency-to-peak-power relationship is non-linear. The lookup table is largely consistent

[2]We assume homogeneous GPU clusters in each site. Heterogeneous clusters could be treated as multiple homogeneous clusters.



(a) *Power vs. Frequency (P99)* **(b)** *TBT vs. Frequency (P99)* **(c)** *KV Usage vs. Frequency (P99)*

Figure 5. *Profiling results of Llama* 3.1 8*B on A*100 40*GB for* Azure *conversation workload.*



(a) *Power consumption vs Frequency (P99)* **(b)** *TBT vs Frequency (P99)* **(c)** *KV Usage vs Frequency (P99)*

Figure 6. *Profiling results of Llama* 3.1 8*B on H*100 80 *GB for* Azure *conversation workload.*

across workloads at low frequencies but exhibits slight RPS-dependence at mid-range frequencies like 810 MHz. To remain conservative, we build the frequency-to-peak-power lookup table from the peak-load envelope, generated using `gpu_burn` [76] at high utilization.

Design implication: XW-Slcs use these lookup tables to pick GPU operating frequencies during periods of power constraint.

O2. Frequency versus inference latency: Latency scaling with frequency is hardware-dependent. Across 3 NVIDIA GPU generations (H100 80GB, A100 80GB, B200), inference latencies (TTFT, TBT, E2E) plateau beyond a frequency threshold due to hardware-enforced power-based throttling; on A100 40GB GPUs, no such plateau was observed. A common takeaway from Fig. 5b is that latency does not scale proportionally with frequency. This uncertainty across hardware and frequency-scaling dimensions makes latency an essential additional signal beyond frequency and capacity (active node count).

Design implication: XWind must use live inference latency signals alongside frequency and capacity signals from individual Greenferencing sites for routing.

O3. Frequency versus KV-cache usage: KV-cache usage is the fraction of reserved KV memory occupied by cached key/value tensors. Lowering frequency reduces throughput, so more tokens remain in flight, raising KV-cache occupancy. Since KV-cache size scales with sequence length, larger active contexts increase memory usage and inter-token latency as each generation step attends to all cached tokens. Fig.5c shows a steep increase in KV-usage below a frequency threshold, crucially just before the exponential TBT rise in Fig.5b. For instance, at 4RPS, KV-cache pressure intensifies below 690MHz, serving as a leading indicator of memory bandwidth saturation. Blindly reducing frequency can push the system past this tipping point, triggering massive KV-cache pressure that manifests as sharp TBT degradation. Because this point is workload-dependent, the controller must monitor this signal to tune the safe operating frequency dynamically. Offline profiling identified the optimal KV-cache threshold for the XW-Slc: empirically, 20% across all workloads for A100 40GB GPUs.

Profiling on H100s: We repeated the above profiling on NVIDIA H100 80 GB SXM GPUs and the observations O1-O3 generalize for them (Figs. 6a-6c). (O1) power scales non-linearly with frequency and remains largely workload-invariant; (O2) latency does not scale proportionally with frequency: on H100 TBT additionally exhibits a plateau beyond a hardware-enforced throttling threshold, unlike the smooth curve on A100; (O3) KV-cache utilization shows a sharp, workload-dependent inflection that precedes TBT degradation. The empirical KV threshold shifts from 20%

(A100 40 GB) to 35% (H100 80 GB), reflecting the larger KV-cache pool. These observations validate the XW-SLC's dual-signal design and threshold-based control transfer across GPU generations; only the threshold values require recalibration.

Design implication: XW-SLCs should avoid downclocking GPUs near this KV-cache threshold, resorting instead to idling some GPUs locally.

4 XWIND Design

We need a robust power-variability-aware request routing and site configuration framework to efficiently route inference requests across multiple Greenferencing sites with temporal variance in power generation while still being able to offer low latencies. We propose a lean, production-friendly router, XWIND, that works in tandem with XW-SLCs, which reconfigure individual sites and send live capacity and telemetry signals to XWIND router. Importantly, the XW-SLC operates *proactively with respect to power*: it receives a forecasted power budget derived from short-term wind power predictions, enabling it to reconfigure GPU capacity and frequency in anticipation of power changes. The reactive telemetry signals (KV-cache utilization, queue depth, and TBT latency) then serve as *corrective feedback*, refining the configuration within each forecast window.

4.1 Overview

Our high-level design goals for the Greenferencing system are twofold: (1) ensuring graceful operation under residual power variability at individual sites, while (2) keeping AI inference latencies low.

A naïve routing approach that works with static routing weights (§5.4) fails in this setting as power availability changes dynamically, leading to significant latency inflation. We rather rely on a site-local XW-SLC for local reconfigurations and a cross-site XWIND for routing under these uncertainties with high-level signals from each site. A hierarchical design is necessary for a lean yet robust design: while the XW-SLC deals with hyper-local signals such as KV-cache utilization and queue depth, the XWIND router has a global view of all Greenferencing sites in a region.

Greenferencing sites differ from traditional data centers as they rely on variable power supplies, which requires intelligent reconfiguration of the compute capacities to match the available power. Placing the reconfiguration logic within the XW-SLC allows the system to resolve local constraints autonomously. Such a design ensures that while the XW-SLC handles high-frequency hardware adjustments locally, the cross-site router focuses on global visibility and coarse-grained signals re-balancing inferencing traffic as necessary.

Why not an optimal/centralized approach? XWIND is deliberately suboptimal: global optimality would require a centralized approach, offline workload profiling, and accurate predictions of both request arrival rates and response output lengths, none of which are practical in real production deployments without significant overhead. A centralized approach also struggles to scale, as it can be easily overwhelmed by the fine-grained, site-level telemetry signals discussed in §4.2, making it ill-suited for production environments.

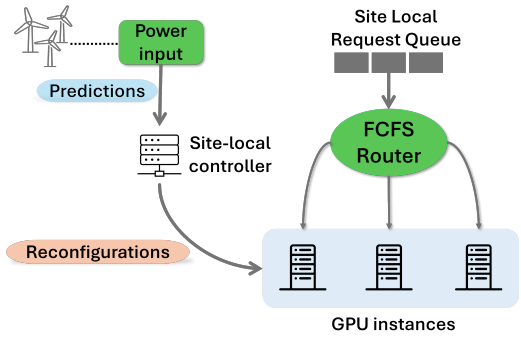


Figure 7. *A Greenferencing site of Fig. 1.*

Selective engagement. Greenferencing sites can accommodate batteries, additional renewable modalities, or opportunistic grid draw. Moreover, peak power needs at individual sites are much lower percentiles of the renewable sites' peak generation capacities, and these infrastructure-level measures significantly improve site availability. The software routing layer need only handle rare residual shortfalls after infrastructural leverages are exhausted. Our system remains passive when all sites meet peak demand; only when a XW-SLC anticipates a power crunch does it trigger local hardware reconfigurations and signal XWIND for cascading routing adjustments.

4.2 XWIND Site-local Controller (XW-SLC)

Greenferencing sites must dynamically scale compute capacity to remain within the instantaneous power budget, primarily by adjusting active node count or operating frequency. Conventional approaches tune only a single knob, which is inherently sub-optimal: frequency downclocking alone is inefficient under moderate loads, forcing the entire cluster to suppressed clock speeds when consolidating onto fewer high-frequency nodes would be more effective. Conversely, node shutdown alone creates quantization inefficiencies, stranding usable power when availability drops even marginally. Idling nodes avoids boot latency but incurs a high static power tax (~1,920 W idle per A100 DGX, roughly 30% of peak 6,500 W). Our XW-SLC design achieves the right balance by jointly exploring the state space of node counts (N) and frequencies (f), using real-time telemetry to filter configurations that satisfy current demand.

Decision window. The XW-Slc operates on a 3-minute decision cycle, yielding five discrete steps per 15-minute forecast window (with linear interpolation), allowing gradual transitions to the target power state. This interval balances two concerns: shorter cycles prevent XWind (running every 15 seconds) from converging on stable weights between reconfigurations, while longer cycles force conservative node idling at window start to accommodate anticipated power drops, stranding usable capacity.

Candidate generation. When anticipating a power change, the XW-Slc enumerates all feasible (N, f) tuples satisfying the new constraint, using the workload-invariant peak-power and frequency mappings from profiling (O1, §3.2). For each candidate node count (1 to n), it computes the per-node power budget and the corresponding maximum frequency. Since capping all GPUs to one frequency may strand power, the XW-Slc boosts a fraction of GPUs to the next frequency level to fully utilize the power budget.

Candidate filtering via telemetry. Selecting the optimal (N, f) tuple is challenging because the workload characteristics (input length, output length, and arrival rates) exhibit significant spatiotemporal volatility. A workload-agnostic strawman approach may select the tuple maximizing $N \times f$, since total FLOPs scale with this product. However, our experiments show that sub-maximal products can outperform the theoretical best depending on workload characteristics. For instance, under high RPS, using all nodes at a slightly lower frequency may beat using two-thirds of nodes at higher frequency, as the former provides more parallel entry points to mitigate queuing delays.

To tackle this challenge, the XW-Slc leverages temporal locality at the macro level. Since aggregate workload trends typically shift at coarser timescales, the XW-Slc uses system and latency metrics from the most recent time window as a proxy for near-future demand. It uses the following signals:

- **Queue depth:** The number of requests waiting to be scheduled. Sustained queue build-up indicates insufficient compute capacity and prevents the XW-Slc from reducing the number of active nodes.
- **KV-cache utilization:** The fraction of reserved KV-cache memory occupied by in-flight tokens. High KV utilization signals rising HBM bandwidth demand and an approaching memory-bound regime; when it crosses a threshold, the XW-Slc raises the operating frequency to avoid the latency knee point (O3, §3.2).
- **TBT latency:** The median token-by-token latency across GPUs in the previous window. TBT captures SLO violations and non-memory bottlenecks such as thermal throttling or compute-bound phases.

The reactive control policy. Using the telemetry signals described above, we design a reactive heuristic (Algorithm 1) that dynamically determines the minimum operating frequency and active node count to maintain stability.

Algorithm 1 Reactive Site-Level Controller (XW-Slc)

Input: Power budget P_t; telemetry $\phi_t = (\mathrm{KV}_t, Q_t, L_t)$
State: Active GPUs N_{curr}, frequency f_{curr}, floor F_{floor}
function ReactiveXW-Slc($P_t,\ \phi_t$)
  $\mathcal{T} \leftarrow \{(f, N) : \text{Power}(f, N) \le P_t\}$ ▷ Viable Configs
  $\pi \leftarrow (Q_t > Q_{\max})$ ▷ Congestion flag
  // Adjust frequency floor from telemetry
  if not π **then**
    if $\mathrm{KV}_t > \mathrm{KV}_{\max}$ **then** F_{floor} += $2\Delta_f$
    else if $L_t > L_{\max}$ **then** F_{floor} += Δ_f
    else if $L_t < L_{\max} \wedge \mathrm{KV}_t < \mathrm{KV}_{\max} \wedge Q_t < Q_{\max}/2$ **then**
      F_{floor} −= Δ_f
    end if
    $F_{\text{floor}} \leftarrow \text{clamp}(F_{\text{floor}},\ F_{\min},\ F_{\max})$
  end if
  // Select best (f, N) *maximizing* $f \cdot N$
  $\mathcal{S} \leftarrow \begin{cases} \{(f,N) \in \mathcal{T} : N \ge N_{\text{curr}}\} & \text{if } \pi \\ \{(f,N) \in \mathcal{T} : f \ge F_{\text{floor}}\} & \text{otherwise} \end{cases}$
  return $\arg\max_{(f,N) \in \mathcal{S} \cup \emptyset \mathcal{T}} f \cdot N$
end function

At a high level, it maintains two state variables, F_{floor} and N_{curr}, which are updated based on real-time signals. If a queue build-up is detected, N_{curr} is updated to match the current number of active nodes. If KV-cache utilization exceeds a hardware-dependent threshold, F_{floor} is increased by $2\Delta f$; similarly, if the TBT latency exceeds its threshold, F_{floor} is raised by Δf. The candidate selection process applies these constraints dynamically. In the event of a queue build-up, the XW-Slc discards all candidates with a node count lower than N_{curr} and selects the remaining tuple with the highest $N \times f$. In the absence of queuing, the XW-Slc filters out candidates operating below F_{floor} and selects the configuration that maximizes the capacity product for the next cycle.

Capacity priority over frequency.: When queues build up, we prioritize a higher N over f. Frequency scaling yields diminishing returns in memory-bound LLM inference, whereas adding a node scales both throughput and aggregate KV-cache capacity linearly. Reducing node count under high load causes queuing delays to inflate exponentially as new arrivals face reduced service capacity. Prioritizing N maintains a larger memory pool and sufficient concurrency to keep the queue stable.

Dual signals for F_{floor}.: We employ two orthogonal signals. KV-cache utilization is a *leading indicator* of memory bandwidth saturation (O3): the XW-Slc raises F_{floor} on KV-cache pressure, preemptively defending against memory bottlenecks before they surface as user-facing latency. TBT is a *lagging indicator* that captures non-memory bottlenecks, such as thermal throttling or compute-bound prefill phases, which KV-cache alone would miss.

Asymmetric correction steps.: The frequency correction applies differential gain: $2\Delta f$ for KV vs. Δf for TBT. KV-cache usage exhibits a sharp saturation cliff at lower frequencies, necessitating aggressive correction to immediately exit saturation and prevent out-of-memory errors. TBT follows a smooth degradation curve, permitting finer-grained adjustments that avoid overshooting.

Maximizing live capacity.: After filtering for stability, the XW-SLC selects the tuple that maximizes $N \times f$, ensuring the system fully utilizes the available power budget and eliminates stranded power.

4.3 XWIND Cross-Site Router

A XWIND cross-site router sits in a data center region and oversees multiple Greenferencing sites. Its main responsibility is to distribute the incoming inference requests across the associated sites. A strawman approach might use node count as a static weight for round-robin balancing, but at green energy sites, compute capacity fluctuates with the underlying power source, making fixed weights inadequate. XWIND therefore updates routing weights every second along two paths: (1) the *proactive* path reacts immediately to capacity or frequency changes signaled by the XW-SLCs and (2) the *reactive* path that corrects residual latency imbalance using observed TBT. Algorithm 2 summarizes the procedure; the rest of this subsection walks through its components.

Routing metric: live compute capacity. XWIND operates on live compute capacity. It periodically probes each XW-SLC for its current active configuration (N, f). The routing weight W_i for site i is: $W_i = N_i \times f_i$. This product serves as a robust proxy for the site's instantaneous token-processing capability, as FLOPs scale linearly with operating frequency [14].

The latency-corrective feedback loop. Relying solely on $N \times f$ can cause imbalances due to non-linear scaling (O2, §3.2). XWIND therefore implements a corrective feedback loop, polling XW-SLCs every second for active node count, frequency, and TBT. On capacity or frequency changes, weights immediately reset proportionally to the new $N \times f$. When no change occurs for 15 seconds, a latency-corrective loop computes an exponential moving average of each site's TBT, calculates the ratio to the global mean, clips it within a sensitivity bound δ, and asymmetrically penalizes only above-mean sites to prevent oscillatory migration. The weights are then renormalized. The XW-SLC signals XWIND 5 seconds before any change in node-count to prevent transient imbalances.

Breaking the cycle. The XW-SLC's dual-knob design introduces a potential oscillation cycle: KV-cache rises → frequency floor raised → node idled to stay in power budget → queue builds up → capacity-priority restores node count at lower frequency → KV-cache rises again. XWIND's cross-site load redistribution breaks this cycle: when the XW-SLC idles a node, XWIND immediately recalculates routing weights ($w_i = N_i \times f_i$), reducing traffic in proportion to diminished capacity. The arrival rate drops in lock-step with the node reduction, preventing the queue build-up that would trigger the next oscillation step. Additional safeguards include asymmetric hysteresis (frequency floor decreases require all three signals to be simultaneously benign) and a capacity-priority circuit breaker (once the queue exceeds $Q_{\max}$, the controller locks node count and only optimizes frequency).

Algorithm 2 XWIND Adaptive Weight Update (every 1 s)

Input: ℓ, c, f – latency, capacity, frequency vectors over sites $\mathcal{S}$
State: $w_s, \hat{L}_s$ – weight and EMA-smoothed latency per site
Params: α (EMA factor), δ (sensitivity bound), Δt=15 s

if $c \neq c_{\text{prev}}$ or $f \neq f_{\text{prev}}$ **then** ▷ Proactive path
  for all $s \in \mathcal{S}$ **do**
    $w_s \leftarrow \frac{c_s \cdot f_s}{\sum_s c_{s\_prev} \cdot f_{s\_prev}} \cdot \sum_s c_s$
  end for
else if Δt elapsed **then** ▷ Reactive path
  for all $s \in \mathcal{S}$ **do**
    $\hat{L}_s \leftarrow (1-\alpha)\hat{L}_s + \alpha\, \ell_s$
  end for
  $\bar{L} \leftarrow \text{mean}(\hat{L})$
  for all $s \in \mathcal{S}$ **do**
    $\rho_s \leftarrow \text{clip}(\hat{L}_s/\bar{L},\ 1-\delta,\ 1+\delta)$
    $w_s \leftarrow w_s/\rho_s$ **if** $\rho_s > 1$ ▷ Reduce only slower sites
  end for
  Normalize: $w_s \leftarrow w_s \cdot \sum c_s / \sum w_s\ \forall s$
end if

5 Evaluation

Our evaluation focuses on the following questions:

- **Q1** How does XWIND compare against baselines under cross-site power-constrained operations across workload types and load levels?
- **Q2** Why does the reactive XW-SLC outperform Max-FLOPS, and when does it choose to diverge?
- **Q3** Are both control signals (KV-cache and TBT) necessary for the XW-SLC decisions?
- **Q4** How much does each cross-site routing signal contribute to XWIND's end-to-end performance?

5.1 Experimental Methodology

We emulate a multi-site Greenferencing deployment on a testbed of 64 NVIDIA A100 40 GB GPUs across three sites (Site-0, Site-1, Site-2) with a 2:1:1 allocation (32, 16, 16 GPUs). Each GPU pair hosts one Llama3.1 8B [54] instance served via vLLM [11] with TP2, yielding 16, 8, and 8 model instances respectively. We modified vLLM to asynchronously log request-level (E2E, TBT, TTFT, queuing delay) and system-level

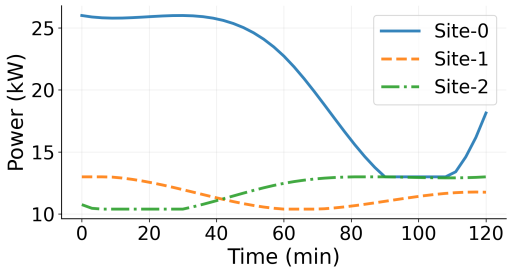


Figure 8. *Power availability across* 3 *sites scaled from real US wind farm data. Site-*0 *experiences a sustained* ~50% *power drop mid-trace. Note: y-axis does not start at* 0.

(queue length, KV-cache utilization) metrics. A lightweight Instance Telemetry process computes sliding-window averages (window=15 s, step=1 s) and exposes state to the XW-Slc, which polls every 15 s to construct site-wide aggregations for XWind. The XW-Slc also schedules (round-robin) requests across homogeneous local instances and signals XWind five seconds before any change in active node count.

Request arrival traces. We generate 120-minute synthetic traces using Poisson arrivals, sampling prefill and decode lengths from Azure production traces. We evaluate three workload types: *coding* (prefill-to-decode ratio 114, prefill-heavy), *conversation* (ratio 16, decode-heavy), and *mixed* (uniform random selection from both). Prompt tokens are generated using the Llama 3.1 8*B* tokenizer. Our custom load generator replays traces with an internal timer synchronized to the trace timeline, keeping arrival rates independent of system latency: critical for evaluating queue dynamics during power contention. We test for 150 RPS (requests per second; moderate) and 175 RPS (high load).

Power trace. We derive our power profiles from real-world wind power data, scaled to match the power footprint of our GPU testbed. We select three wind farm sites in the Central US that exhibit spatial complementarity in generation. The 2-hour power trace (Fig. 8) captures a realistic scenario where Site-0 (the largest site) experiences a sustained power drop of approximately 50% mid-trace, while one of the smaller sites dips by 20%. In our experiments, we deliberately do not use a backup data center to stress test the Greenferencing deployment. All requests are served by the three wind-powered sites; none are dropped or rejected.

Power forecasting. As mentioned in §2.2: here we assume oracular knowledge of power availability at 15 minute intervals and linearly interpolate between them, providing the XW-Slc with a continuous power budget for each 3-minute decision cycle. Enough redundancy in the power distribution and uninterruptible power supply units tackle any residual inaccuracies in prediction and also offers smoothened rather than abrupt changes in the input. Finally, empirically across GPU generations, workloads, and power profiles, a longer decision cycle wastes available power, while a shorter cycle results in convergence challenges at the XWind router.

GPU power modeling. GPU servers also incur non-GPU power overhead (CPUs, NICs, cooling). We amortize this by assigning a fixed 240 W (1,920 W idle state consumption split uniformly across 8 GPUs in a DGX) per active A100 in our power consumption models. This value could be trivially changed, as needed (we tested for H100s with a different assigned value), without affecting the utility of our solution.

Network latency. Greenferencing sites are co-located within a region; so network latency inflation is minimal and only affects the time when the user receives back the first token (not time between following tokens that are streamed). The marginal TTFT inflation is well within typical user-facing TTFT SLO bounds (few 100 ms to seconds), making it orthogonal to the power-driven latency dynamics that are the focus of our evaluation. Hence, we can safely ignore this latency component in our emulations.

5.2 Baselines and Metrics

We compare our reactive XW-Slc with four approaches, each paired with XWind for cross-site balance. (1) *Down-clock*: computes per-node power budgets and locks GPUs to the maximum feasible frequency via the workload-invariant lookup table (O1). (2) *Idle*: inactive nodes stay at lowest frequency with no requests (~30% peak power tax), enabling instant ramp-up; active instances run at max. 1,410 MHz. (3) *Power-Capping*: applies hardware power limits via `nvidia-smi -pl`; GPU frequency could vary with workload, so XWind uses observed aggregate frequency over a sliding window for routing weights. (4) *Max-FLOPS*: maximizes $N \times f$ within the power budget: an approach equivalent to reactive XW-Slc but without telemetry-driven filtering.

Metrics. Since no requests are dropped, the goal is graceful degradation under power contraction. We report: (1)*P*99 *E2E latency*, capturing the tail user experience, including queuing and inference time, and (2)*P*99 *queue time*, isolating the waiting time before inference. Since all approaches use identical hardware and model configurations, inference time at a given frequency is roughly constant across baselines. The differences in E2E latency are driven by queuing, making queue time P99 the diagnostic metric for routing and capacity decisions.

Thresholds. XW-Slc thresholds are calibrated from offline A100 40 GB profiling (§3.2). *KV-cache threshold* $\mathrm{KV}_{\max} = 20\%$, set at the sharp inflection in P99 KV utilization versus frequency (Fig.5c). *TBT threshold* $L_{\max} = 100$ ms per typical interactive SLOs [68]. Asymmetric corrections ($2\Delta f$ for KV vs. Δf for TBT) reflect the steeper KV cliff versus gradual TBT curve; $\Delta f = 60$ MHz (one A100 step). *Queue depth threshold* $Q_{\max} = 5$ per instance; exceeding this triggers capacity-priority mode (§4). Decision cycle $T_{\text{cycle}} = 180$ s balances queue drain time against 15-minute power forecast

granularity. XWind probes XW-Slcs every 1 s and rebalances routing weights every 15 s.

5.3 XW-Slc performance

Fig. 9 shows P99 E2E latencies and P99 queue times across all site-local baselines and XW-Slc, with XWind as the cross-site router, for three workload types at 150 and 175 RPS.

No single knob suffices. Single-knob baselines degrade catastrophically under power contraction. At 175 RPS, *Idle* reaches 370 s (likewise, 539 s) for coding (conversation). Essentially, the static power tax of idle nodes wastes desperately needed capacity. *Power-Capping* reaches 33.9 s (153 s): its non-deterministic frequency behavior causes unpredictable throughput collapses during power drops. Even *Downclock*, the most competitive single-knob approach, reaches 25.3 *s* (78.4 s) at 175 RPS: 1.8× (2.2×) worse than XW-Slc. The queue-time breakdown (Fig. 9, bottom) shows that these E2E gaps are driven primarily by queuing: *Idle* accumulates 368 s of queue time at 175 RPS, whereas XW-Slc keeps queuing to 5.5-27.4 s. These results validate the need for a dual-knob site-local strategy, as in XW-Slc (**Q**1), that simultaneously adjusts frequency and active node count.

XW-Slc **versus Max-FLOPS.** *Max-FLOPS*, while devoid of the telemetry-driven filtering in XW-Slc, also operates both knobs. While at 150 RPS, the performances are still comparable, at 175 RPS XW-Slc offers 22% (likewise, 52%) lower P99 E2E latencies for coding (conversation) traces. A deeper investigation into the decisions taken by the schemes at each step revealed that XW-Slc could more aggressively trade parallelism (active node count) for frequency than *Max-FLOPS*. This demonstrates that telemetry filtering (queue depth, KV-cache, TBT) is important for an efficient XW-Slc design in a variable-power setting (answers **Q**2).

XW-Slc**'s Dual-signal for frequency fine-tuning.** We validate the dual-signal design by running conversation at 175 RPS with three XW-Slc variants: full (KV+TBT+Q; Q being the queue depth), KV-only (KV+Q), and TBT-only (TBT+Q). Fig.10 shows tail E2E latency CDFs. Removing TBT has modest impact (P99.9 shifts from ∼72 s to ∼96 s), consistent with KV being the dominant signal. Removing KV causes severe degradation (P99.9 exceeds 120 s): without this leading indicator, the XW-Slc reacts only after TBT spikes, at which point KV saturation cascades into sustained queuing. The full system achieves the tightest tail: KV provides early warning against memory-driven blow-up, while TBT backstops non-memory bottlenecks (thermal throttling, prefill surges, hardware power-limit frequency drops) that degrade latency without manifesting in KV occupancy (answers **Q**3).

XW-Slc **deep-dive on the conversation trace.** At 175 RPS, XW-Slc diverges from *Max-FLOPS*. During power contraction at Site-0 (t = 5,400–7,200 s), KV utilization repeatedly exceeds 20%, driving the dynamic frequency floor from 600 to 840 MHz. This frequency-over-parallelism trade-off exploits a property of decode-heavy requests (P/D=16): per-token latency scales directly with GPU frequency, compounding over hundreds of tokens. *Max-FLOPS*'s 14 instances at 540 MHz offer more parallel slots, but each drains slowly, accumulating KV entries until memory bandwidth saturates. Fewer slots at higher frequency (840 vs. 540 MHz) drain requests faster, keeping KV utilization in check and preventing cascading TBT degradation. The XW-Slc discovers this trade-off at runtime purely from the KV signal, without workload labels.

Mixed workload. To confirm that XW-Slc's gains are substantial even for a mixed workload, we repeat the same experiment but the prefill and decode lengths are sampled uniformly at random from both coding and conversation distributions. At 175 RPS (Fig. 9, right), XW-Slc offers a 17% reduction in P99 E2E compared to *Max-FLOPS*. All other baselines inflate latency even more.

XW-Slc **robustness.** Each 2-hour experiment at 175 RPS generates 1.26 M requests (P99 over ∼12,600 samples); a percentile breakdown in Table1 confirms trends are consistent across P95/P99/P99.9. Forecast errors are mitigated by the 180 s decision cycle, which provides five corrective opportunities per forecast interval; combined with sub-second XW-Slc runtime, transient mismatches are absorbed within one cycle. Our power profile is deliberately challenging: a sustained 50% drop at the largest site is rare in practice, where batteries, complementary renewables, or grid draw would buffer variability. All of our high-level findings are robust across multiple complementary power profiles that we have tested with.

5.4 XWind Router Ablation Study

To isolate the contribution of each signal in XWind's cross-site routing (**Q**4), we evaluate four router variants, all paired

Table 1. P95/P99/P99.9 E2E latency (in seconds) for all site-local methods for coding and conversation workloads at 150 and 175 RPS. **Bold** highlights XW-Slc.

	150 RPS			175 RPS		
Site-local Logic	P95	P99	P99.9	P95	P99	P99.9
Coding Workload						
XW-Slc	**2.8**	**7.9**	**26.4**	**5.8**	**14.0**	**48.6**
Max-FLOPS	2.8	7.9	26.1	7.2	17.9	49.1
Downclock	2.9	8.5	27.6	12.0	25.3	57.8
Power-Capping	3.1	18.7	465.4	11.8	33.9	77.6
Idle-GPUs	44.8	71.5	89.3	330.6	369.9	386.3
Conversation Workload						
XW-Slc	**9.7**	**16.7**	**27.5**	**14.1**	**31.4**	**72.2**
Max-FLOPS	9.6	17.0	30.0	18.4	66.2	126.9
Downclock	10.4	19.5	38.4	27.4	78.4	137.0
Power-Capping	11.6	36.9	184.6	87.6	153.3	260.4
Idle-GPUs	156.8	217.6	276.0	474.1	539.4	597.4

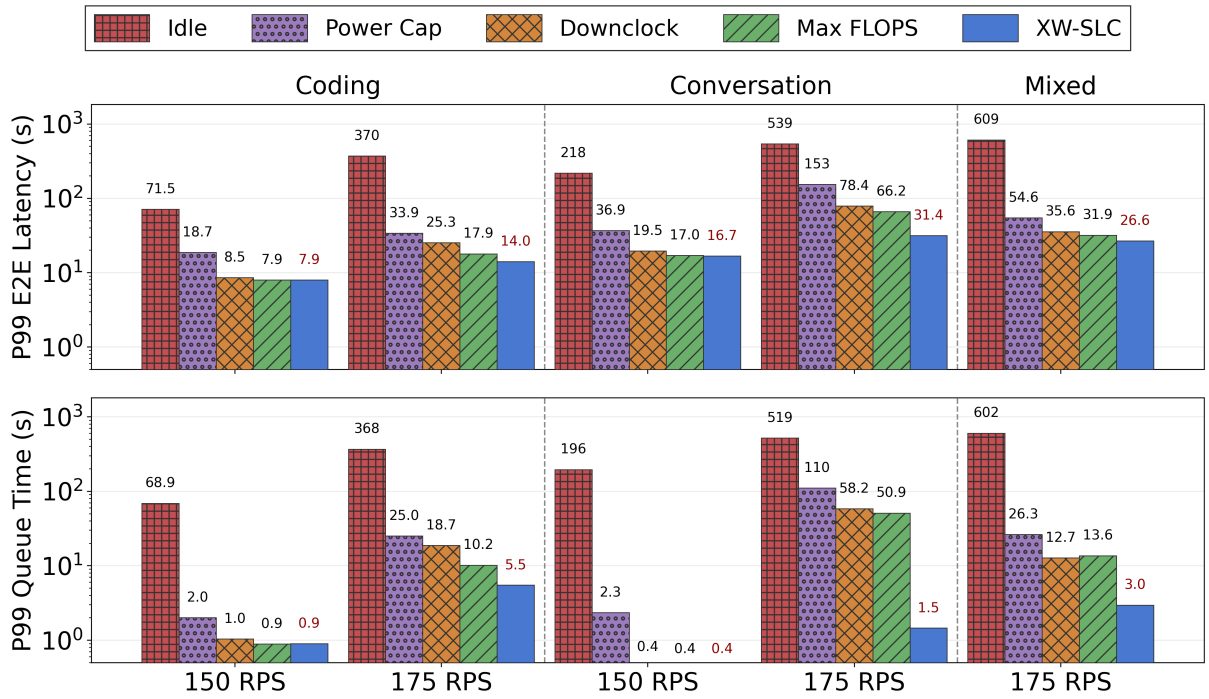


Figure 9. *P99 E2E latency (top) and P99 queue time (bottom) across site-local logic variants for coding and conversation workloads. Note: y-axis is log-scale.*

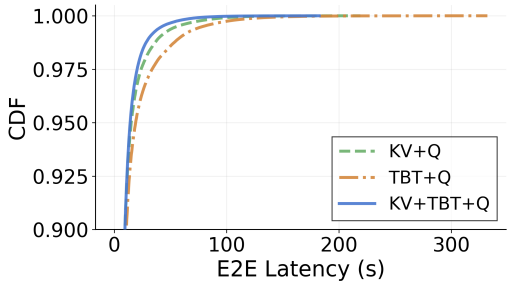


Figure 10. *CDF (tail) of E2E latency for conversation trace, 175 RPS, for different site-local logic.*

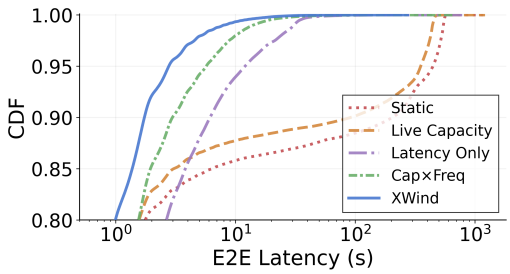


Figure 11. *CDF of E2E latency for different routing strategies using the same reactive* XW-Slc *(coding, 150 RPS).*

with the same reactive XW-Slc at each site, on the coding workload at 150 RPS (Fig. 11). The variants differ *only* in how the cross-site router computes routing weights.

Static routing is oblivious to power changes. The *Static* (2:1:1) router distributes traffic in proportion to site capacities, ignoring runtime signals entirely. Even when Site-0 loses 50% of its power mid-trace, the router continues sending statically decided request volume there, causing significant queue buildup: P99 E2E: 546 s, P99 queue-time: 540 s.

Live capacity alone is insufficient. *Live Capacity* routing sets weights proportional to each site's active GPU count, adapting to XW-Slc-driven reconfigurations. This yields only a 19% P99 E2E improvement (443 s) over *Static* routing: knowing *how many* GPUs are active does not capture *how congested* they are: a low-frequency site with a deep queue still attracts traffic proportional to its node count.

Latency feedback is the single largest lever. *Latency Only* routing weights sites inversely by EMA-smoothed TBT, with no capacity or frequency knowledge. This achieves 30.8 s P99 E2E, 18× lower than *Static*, because latency directly reflects congestion: rising TBT under power contraction diverts traffic before queues build up. However, latency is a lagging indicator; by the time TBT increases, requests are already queued, thus inflating E2E latency.

Capacity×frequency provides a leading signal. *Cap×Freq* router weighs each site by the product of active GPU count and operating frequency, capturing both quantity

and quality of available compute. This yields P99 E2E of 14 s (2.2× lower than *Latency Only*) because the router *anticipates* throughput changes at reconfiguration time rather than waiting for latency inflation to react.

XWind **combines leading and lagging signals.** The XWind router integrates Cap×Freq with EMA-smoothed per-site TBT latency, achieving 7.9 s, a further 1.8× improvement over *Cap×Freq* alone. Cap×Freq signal sets the coarse routing weights based on each site's announced compute capacity, while the latency feedback loop applies fine-grained corrections that account for transient congestion not captured by the capacity signal. The CDF (Fig. 11) reveals that *Cap×Freq* and XWind track closely below P90; the latency corrections primarily tighten the tail, where transient imbalances would otherwise cascade. Together, the two signals reduce P99 E2E by 69× over *Static* routing.

We ran similar experiments as above on a smaller H100 setup with 3 sites (4, 2, and 2 GPUs). The high-level takeaways are same. The results have been omitted here for brevity.

6 Related Work

Carbon-aware and sustainable computing. Previous works [12, 13, 41, 60, 66, 69] shift workloads temporally or geographically to reduce carbon footprint but still rely on grid power. AI Greenferencing instead deploys compute directly at the renewable source, eliminating transmission losses, interconnection queues, and renewable energy credit abstractions while focusing on the AI inference workload.

Energy-efficient LLM serving. DynamoLLM [68] minimizes AI inferencing energy by tuning parallelism and GPU frequencies, but assumes unconstrained power at a single site. VoltanaLLM [80] proposes frequency control and state-space routing for efficient LLM serving, also in a single-site grid-powered setting. Other GPU energy optimizations [15, 78, 79] target training or accelerator-level efficiency using analytical models and fine-grained DVFS, not inference under power variability. XWind instead dynamically routes the load between multiple sites based on fluctuating renewable power budgets.

Renewable-powered and modular data centers. Prior work [37, 40, 42, 61, 83] explored renewable-powered data centers using solar, batteries, and scheduling optimization. AI Greenferencing differs by: (1) targeting wind energy with its distinct variability; (2) right-sizing deployments at lower percentiles of peak generation to reduce power uncertainty; (3) exploiting power complementarity across dispersed sites; and (4) leveraging the stateless, request-level granularity of AI inferencing. Companies like Windcores [8], Soluna [4], and Westfalenwind [7] deploy compute in wind farms for cryptocurrency and streaming; AI Greenferencing rather focuses on an emerging high-value AI workload.

7 Discussions

Logistics. Greenferencing requires complex logistics: ROI/TCO analysis, footprint expansion planning, etc., all ongoing work with finance teams, renewable energy partners, and modular DC vendors. This paper rather focuses on the core technical contributions. Recent developments [70, 71] validate modular edge expansion; we target bringing these to renewable sites for AI inference.

Co-existence with traditional data centers. Greenferencing could complement rather than replace conventional data centers. Requests are preferentially routed to Greenferencing sites for lower energy costs; overflow is absorbed by traditional DCs as elastic peak-load capacity. The router's per-site volume-to-latency mapping dynamically adjusts this allocation each cycle. Given the spatial complementarity and other guardrails, spillage is rare and needs minimal provisioning.

GPU phase-out. With GPU leaders like NVIDIA announcing a new generation almost every year, hyperscalers need to come up with concrete plans to phase-out the quickly aging GPUs and make room for the newer ones to run training workloads. AI Greenferencing could also help address this mid-life GPU crisis, if needed, by shipping them to wind sites and running a fraction of the AI workload at lower CAPEX (already offset heavily at the full-fledged data center).

Elephants versus mice. Asymmetric Greenferencing site sizes could cause routing imbalances under power variability. Our analysis (§2.1) confirms that Azure regions host diverse wind farms within low network latency, allowing sizeable deployments that absorb power uncertainty; in rare cases, requests can spill over to the full-fledged data center.

8 Conclusion

This paper introduces AI Greenferencing, a deployment model that co-locates modular AI inference compute at wind farms to bypass grid delivery bottlenecks, and XWind, a reactive cross-site router that serves LLM inference under variable renewable power without offline workload profiling. Evaluated on 64 NVIDIA A100 GPUs across 3 sites with Azure production traces, XWind reduces P99 E2E latency by 22–52% over the strongest dual-knob contender (MaxFLOPS) and by up to 98% over single-knob baselines.

References

[1] 2025. DCGMI. https://microsoft.github.io/VirtualClient/docs/workloads/dcgmi/.

[2] 2025. Microsoft & Constellation's Bid to Restart Three Mile Island. https://datacentremagazine.com/critical-environments/microsoft-constellation-restarting-a-nuclear-reactor. 20-year, $16B deal for 835 MW dedicated to Microsoft AI DCs. Accessed: 2026-04-02.

[3] 2025. Nuclear power for AI: inside the data center energy deals. https://introl.com/blog/nuclear-power-ai-data-centers-microsoft-google-amazon-2025. Amazon investing >$20B in nuclear-adjacent DC sites. Accessed: 2026-04-02.

[4] 2025. Soluna. https://www.solunacomputing.com/.

[5] 2025. Starcloud Launches Orbital AI Data Center With NVIDIA H100 GPU. https://www.datacenterfrontier.com/site-selection/article/55337494/starcloud-launches-orbital-ai-data-center-with-nvidia-h100-gpu. First orbital AI DC, Nov 2025, LLM inference in orbit. Accessed: 2026-04-02.

[6] 2025. US Approves $1B Loan to Restart Three Mile Island for Microsoft Data Centers. https://gizmodo.com/us-approves-1b-loan-to-restart-three-mile-island-as-microsoft-data-centers-drive-demand-2000688138. DOE $1B loan, targeting 2027 restart. Accessed: 2026-04-02.

[7] 2025. WestfalenWIND. https://www.westfalenwind.de/.

[8] 2025. windCORES. https://www.windcores.de/en/.

[9] 2026. Crusoe Announces New 900 MW AI Factory Campus in Abilene, Texas for Microsoft. https://www.cxodigitalpulse.com/crusoe-announces-new-900-mw-ai-factory-campus-in-abilene-texas-to-support-microsoft-ai-infrastructure/. Behind-the-meter natural gas, 1.2 GW campus. Accessed: 2026-04-02.

[10] 2026. Powering the Intelligence Age: Bloom Energy and Wyoming 1.8 GW AI Data Center Project. https://markets.chroniclejournal.com/chroniclejournal/article/marketminute-2026-1-8-powering-the-intelligence-age-bloom-energy-shares-surge-as-wyoming-approves-massive-18-gw-ai-data-center-project. 1 GW solid-oxide fuel cells for behind-the-meter DC power. Accessed: 2026-04-02.

[11] 2026. vLLM. https://docs.vllm.ai/en/latest/.

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

[13] Anup Agarwal, Jinghan Sun, Shadi Noghabi, Srinivasan Iyengar, Anirudh Badam, Ranveer Chandra, Srinivasan Seshan, and Shivkumar Kalyanaraman. 2021. Redesigning data centers for renewable energy. In *ACM HotNets*.

[14] Ghadah Ali, Mulya Side, Sridutt Bhalachandra, Nicholas J Wright, and Yong Chen. 2023. Performance-aware energy-efficient GPU frequency selection using DNN-based models. In *Proceedings of the 52nd International Conference on Parallel Processing*. 433–442. https://doi.org/10.1145/3605573.3605600

[15] Ghazanfar Ali, Mert Side, Sridutt Bhalachandra, Nicholas J. Wright, and Yong Chen. 2023. Performance-Aware Energy-Efficient GPU Frequency Selection using DNN-based Models. In *Proceedings of the 52nd International Conference on Parallel Processing (ICPP)*. 433–442. https://doi.org/10.1145/3605573.3605600

[16] Sam Altman. 2025. “it’s super fun seeing people love images in chatgpt. but our GPUs are melting”. https://x.com/sama/status/1905296867145154688. Accessed: 2025-06-15.

[17] Amperon. 2024. US Solar and Wind Curtailment Is Exploding. https://www.amperon.co/blog/us-solar-and-wind-curtailment-is-exploding. Estimated 20 TWh curtailed in US in 2024. Accessed: 2026-04-02.

[18] Azure. 2025. Azure Modular Data Center (MDC) Operator and User Documentation. https://learn.microsoft.com/en-us/azure-stack/mdc/.

[19] AzurePublicDataset. 2025. Azure LLM inference trace 2024. https://github.com/Azure/AzurePublicDataset/blob/master/AzureLLMInferenceDataset2024.md.

[20] Alexander Bick, Adam Blandin, and David J Deming. 2024. *The rapid adoption of generative AI*. Technical Report. National Bureau of Economic Research.

[21] David Chernicoff. 2024. How Data Centers Are Harnessing AI Workloads for Enhanced Cloud, LLM, and Inference Capabilities. *Data Center Frontier* (2024). https://rb.gy/31u0ni Accessed: 2025-06-15.

[22] CoreSite. 2024. AI and the Data Center: Driving Greater Power Density. https://www.coresite.com/blog/ai-and-the-data-center-driving-greater-power-density.

[23] Casey Crownhart. 2024. Why Microsoft made a deal to help restart Three Mile Island. https://www.technologyreview.com/2024/09/26/1104516/three-mile-island-microsoft/.

[24] Databank. 2024. Exploring Modular Data Centers: Benefits, Design, And Deployment. https://www.databank.com/resources/blogs/exploring-modular-data-centers.

[25] DCSMI. 2024. Data Center Workloads, Hyperscale Utilization Rates, and AI GPU Impact. https://www.dcsmi.com/blog/data-center-workloads-hyperscale-utilization-rates-and-ai-gpu-impact. GPU clusters 70–90% utilization. Accessed: 2026-04-04.

[26] Tim De Chant. 2024. Google kicks off $20B renewable energy building spree to power AI. https://techcrunch.com/2024/12/10/google-kicks-off-20b-renewable-energy-building-spree-to-power-ai/.

[27] Delta Power Solutions. 2024. Modular Data Centers: The Rise and the Advantages. https://www.deltapowersolutions.com/en-in/mcis/technical-article-modular-data-centers-the-rise-and-the-advantages.php.

[28] Diana DiGangi. 2023. Dominion Energy projects adding up to 9 GW of gas-fired capacity in Virginia to bolster reliability. https://tinyurl.com/4fbcdx24.

[29] Edison Electric Institute. 2025. Electric Companies to Invest Nearly $208B in 2025 to Strengthen Grid and Drive Economic Growth. https://www.eei.org/en/news/news/all/electric-companies-to-invest-nearly-208b-in-2025-to-strengthen-grid-and-drive-economic-growth. Record annual grid investment. Accessed: 2026-04-02.

[30] EIA. 2024. Why are Midwest grid operators turning away wind power? https://www.eia.gov/todayinenergy/detail.php?id=62406.

[31] Elia. 2025. Wind power generation. https://www.elia.be/en/grid-data/generation-data/wind-power-generation.

[32] Robert L Fares and Carey W King. 2017. Trends in transmission, distribution, and administration costs for US investor-owned electric utilities. *Energy Policy* (2017).

[33] Financial Times. 2026. Microsoft vows to ‘pay its way’ as it seeks to defuse data centre backlash. *Financial Times* (Jan. 2026). https://www.ft.com/content/3f392c9b-c07d-42f5-b000-0a7347ad1ec0 Accessed: 2026-03-09.

[34] GE Vernova. 2025. Going Big: To Support Data Center Growth and Rising Renewables, Crusoe Ordering Flexible Gas Turbines. https://www.gevernova.com/news/articles/going-big-support-data-center-growth-rising-renewables-crusoe-ordering-flexible-gas. 29 LM2500XPRESS turbines, 1 GW. Accessed: 2026-04-02.

[35] GitHub. 2025. GitHub Copilot. https://github.com/features/copilot.

[36] Global Energy Monitor. 2026. Global Wind Power Tracker. https://globalenergymonitor.org/projects/global-wind-power-tracker/.

[37] Íñigo Goiri, William Katsak, Kien Le, Thu D Nguyen, and Ricardo Bianchini. 2013. Parasol and greenswitch: Managing datacenters powered by renewable energy. *ACM SIGPLAN Notices* (2013).

[38] Google. 2024. New nuclear clean energy agreement with Kairos Power. https://blog.google/company-news/outreach-and-initiatives/sustainability/google-kairos-power-nuclear-energy-agreement/. First corporate SMR fleet deal, up to 500 MW by 2030–2035. Accessed: 2026-04-02.

[39] Diana Goovaerts and Matt Hamblen. 2024. Could GPU power levels break the data center ecosystem? https://www.fierce-network.com/cloud/could-gpu-power-levels-break-data-center-ecosystem.

[40] Sriram Govindan, Anand Sivasubramaniam, and Bhuvan Urgaonkar. 2011. Benefits and Limitations of Tapping into Stored Energy for Datacenters. In *Proceedings of the 38th Annual International Symposium on Computer Architecture (ISCA)*. ACM, 341–352.

[41] Walid A. Hanafy, Qianlin Liang, Noman Bashir, David Irwin, and Prashant Shenoy. 2023. CarbonScaler: Leveraging Cloud Workload Elasticity for Optimizing Carbon-Efficiency. *Proceedings of the ACM*

on Measurement and Analysis of Computing Systems (POMACS) 7, 3 (2023).

[42] Md E Haque, IŽigo Goiri, Ricardo Bianchini, and Thu D Nguyen. 2015. Greenpar: Scheduling parallel high performance applications in green datacenters. In *ACM ICS*.

[43] Astrid Hennevogl-Kaulhausen and Ulrike Ostler. 2024. Modular data centers: Faster, more flexible and more energy-efficient in the data centers. https://www.deltapowersolutions.com/en-in/mcis/technical-article-modular-data-centers-faster-more-flexible-and-more-energy-efficient-in-the-data-centers.php.

[44] Javier C. Hernandez. 2017. It Can Power a Small Nation. But This Wind Farm in China Is Mostly Idle. https://www.nytimes.com/2017/01/15/world/asia/china-gansu-wind-farm.html.

[45] Bobby Hollis. 2024. Accelerating the addition of carbon-free energy: An update on progress. https://www.microsoft.com/en-us/microsoft-cloud/blog/2024/09/20/accelerating-the-addition-of-carbon-free-energy-an-update-on-progress/.

[46] International Energy Agency. 2025. Energy and AI: Energy Demand from AI. https://www.iea.org/reports/energy-and-ai/energy-demand-from-ai. Global DC demand 415 TWh (2024), projected 945 TWh by 2030. Accessed: 2026-04-02.

[47] Peeyush Kumar, Ranveer Chandra, Chetan Bansal, Shivkumar Kalyanaraman, Tanuja Ganu, and Michael Grant. 2021. Micro-climate prediction-multi scale encoder-decoder based deep learning framework. In *ACM SIGKDD*.

[48] Lawrence Berkeley National Laboratory. 2025. Queued Up: 2025 Edition — Characteristics of Power Plants Seeking Transmission Interconnection. https://emp.lbl.gov/queues. 2,300 GW in queue (end 2024), 13–19% completion rate, 4.5–5 yr median wait. Accessed: 2026-04-02.

[49] Lazard. 2025. Levelized Cost of Energy+ (LCOE+). https://www.lazard.com/media/eijnqja3/lazards-lcoeplus-june-2025.pdf. June 2025 edition. Accessed: 2026-04-02.

[50] Vivian Lee. 2024. U.S. Data Center Power Outlook: Balancing competing power consumption needs. https://www.linkedin.com/pulse/us-data-center-power-outlook-balancing-competing-consumption-lee-iz4pe/.

[51] Bryan Lim, Nicolas Loeff, Sercan Arik, and Tomas Pfister. 2021. Temporal Fusion Transformers for Interpretable Multi-horizon Time Series Forecasting.

[52] McKinsey & Company. 2024. The role of power in unlocking the European AI revolution. https://tinyurl.com/bdf952sr.

[53] McKinsey & Company. 2025. The next big shifts in AI workloads and hyperscaler strategies. https://www.mckinsey.com/industries/technology-media-and-telecommunications/our-insights/the-next-big-shifts-in-ai-workloads-and-hyperscaler-strategies. Accessed: 2026-04-04.

[54] Meta. 2024. Llama 3.1. https://ai.meta.com/blog/meta-llama-3-1/.

[55] Modo Energy. 2024. The Curtailment Crisis: Saving Wind and Solar Investments in ERCOT. https://modoenergy.com/research/en/ercot-curtailment-crisis-solar-wind-data-battery-colocated-trends-maps-texas. Over 8 TWh curtailed in ERCOT in 2024. Accessed: 2026-04-02.

[56] National Renewable Energy Laboratory. 2025. Cost of Wind Energy Review: 2024 Edition. https://docs.nrel.gov/docs/fy25osti/91775.pdf. Onshore wind LCOE 2.6–5.4 cents/kWh, PPA 2.3–4.5 cents/kWh. Accessed: 2026-04-02.

[57] NVIDIA. 2023. NVIDIA DGX SuperPOD Data Center Design. https://docs.nvidia.com/nvidia-dgx-superpod-data-center-design-dgx-h100.pdf.

[58] NVIDIA. 2025. NVIDIA Data Center GPUs. https://www.nvidia.com/en-in/data-center/data-center-gpus/.

[59] Dylan Patel, Daniel Nishball, and Jeremie Eliahou Ontiveros. 2024. AI Datacenter Energy Dilemma – Race for AI Datacenter Space. https://semianalysis.com/2024/03/13/ai-datacenter-energy-dilemma-race/#datacenter-math.

[60] Ana Radovanovic, Ross Koningstein, Ian Schneider, Bokan Chen, Alexandre Duarte, Binz Roy, Diyue Xiao, Maya Haridasan, Patrick Hung, Nick Care, Saurav Talukdar, Eric Mullen, Kendal Smith, MariEllen Cottman, and Walfredo Cirne. 2021. Carbon-Aware Computing for Datacenters. arXiv:2106.11750 [cs.DC]

[61] Chuangang Ren, Di Wang, Bhuvan Urgaonkar, and Anand Sivasubramaniam. 2012. Carbon-Aware Energy Capacity Planning for Datacenters. In *Proceedings of the 2012 IEEE 20th International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS)*. IEEE, 391–400. https://doi.org/10.1109/MASCOTS.2012.51

[62] Reuters. 2025. Amazon CEO sets out AI investment mission in annual shareholder letter. https://rb.gy/emuya0 Accessed: 2025-06-16.

[63] Reuters. 2025. Ghibli effect: ChatGPT usage hits record after rollout of viral feature. https://rb.gy/vx2m0n Accessed: 2025-06-16.

[64] Martin Rosenberg. 2024. Evergy Struggles to Cut Carbon as Energy Demand Soars. https://tinyurl.com/mtw7rmj6.

[65] James Sanders. 2025. Tech Billionaires Race to Build AI Data Centers in Space. https://www.techrepublic.com/article/news-ai-data-centers-space-race/. SpaceX Starlink V3, Google Suncatcher. Accessed: 2026-04-02.

[66] Abel Souza, Noman Bashir, Jorge Murillo, Walid Hanafy, Qianlin Liang, David Irwin, and Prashant Shenoy. 2023. Ecovisor: A Virtual Energy System for Carbon-Efficient Applications. In *Proceedings of the 28th ACM International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*. ACM.

[67] Stanford Institute for Human-Centered AI. 2025. Artificial Intelligence Index Report 2025. *arXiv preprint arXiv:2504.07139* (2025).

[68] Jovan Stojkovic, Chaojie Zhang, Íñigo Goiri, Josep Torrellas, and Esha Choukse. 2025. Dynamollm: Designing llm inference clusters for performance and energy efficiency. In *IEEE HPCA*.

[69] Jinghan Sun, Zibo Gong, Anup Agarwal, Shadi Noghabi, Ranveer Chandra, Marc Snir, and Jian Huang. 2024. Exploring the Efficiency of Renewable Energy-based Modular Data Centers at Scale. In *ACM SoCC*.

[70] Dan Swinhoe. 2026. Nvidia, Prologis, EPRI, InfraPartners target prefab data centers at substation sites. https://www.datacenterdynamics.com/en/news/nvidia-prologis-epri%2Dinfrapartners-target-prefab-data-centers-at-substation-sites/.

[71] TechInAsia. 2026. China turns to underwater data centers to fuel AI boom. https://www.techinasia.com/news/china-turns-underwater-data-centers-ai.

[72] Thunder Said Energy. 2024. US electric utilities: transmission and distribution costs? https://thundersaidenergy.com/downloads/us-electric-utilities-transmission-and-distribution-costs/.

[73] US Dept. of Energy. 2022. Advantages and Challenges of Wind Energy. https://www.energy.gov/eere/wind/advantages-and-challenges-wind-energy.

[74] U.S. Energy Information Administration. 2024. Solar and wind power curtailments are increasing in California. https://www.eia.gov/todayinenergy/detail.php?id=65364. California curtailment up 29% YoY to 3,400 GWh. Accessed: 2026-04-02.

[75] U.S. Energy Information Administration. 2026. Electric Power Monthly — Average Retail Price of Electricity. https://www.eia.gov/electricity/monthly/epm_table_grapher.php?t=epmt_5_3. Industrial: 9.3 cents/kWh, Commercial: 13.6 cents/kWh (Jan 2026). Accessed: 2026-04-02.

[76] Ville-Pekka Vainio. 2024. gpu-burn: Multi-GPU CUDA stress test. https://github.com/wilicc/gpu-burn. Accessed: 2026-02-03.

[77] Wikipedia. 2025. Wind power in the United Kingdom: Constraint payments. https://en.wikipedia.org/wiki/Wind_power_in_the_United_Kingdom?#Constraint_payments.

[78] Weihang Xian, Phuong Nguyen, and Lieven Eeckhout. 2025. Using Analytical Performance/Power Model and Fine-Grained DVFS to Enhance AI Accelerator Energy Efficiency. In *Proceedings of the 30th ACM International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*. https://doi.org/10.1145/3669940.3707231

[79] Jie You, Jae-Won Chung, and Mosharaf Chowdhury. 2023. Zeus: Understanding and Optimizing GPU Energy Consumption of DNN Training. In *20th USENIX Symposium on Networked Systems Design and Implementation (NSDI)*. USENIX Association.

[80] Jiahuan Yu, Aryan Taneja, Junfeng Lin, and Minjia Zhang. 2025. VoltanaLLM: Feedback-Driven Frequency Control and State-Space Routing for Energy-Efficient LLM Serving. *arXiv preprint arXiv:2509.04827* (2025).

[81] Zenodo. 2021. EMHIRES dataset: wind and solar power generation. https://zenodo.org/records/8340501.

[82] Mary Zhang. 2023. Data Center Maintenance: A Comprehensive Guide. https://dgtlinfra.com/data-center-maintenance/.

[83] Yanwei Zhang, Yefu Wang, and Xiaorui Wang. 2011. GreenWare: Greening Cloud-Scale Data Centers to Maximize the Use of Renewable Energy. In *Middleware 2011 (Lecture Notes in Computer Science, Vol. 7049)*. Springer, 143–164. https://doi.org/10.1007/978-3-642-25821-3_8