# GPU-to-Grid
## Voltage Regulation via GPU Utilization Control



Zhirui Liang


Jae-Won Chung


Mosharaf Chowdhury


Jiasi Chen


Vladimir Dvorkin

University of Michigan, Ann Arbor

`https://gpu2grid.io`

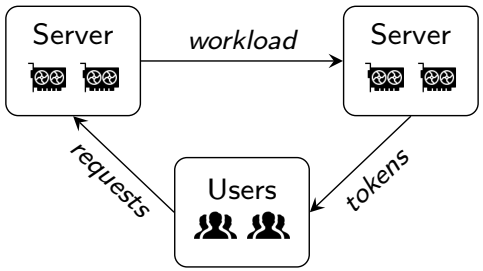


PowerUp Conference 2026
University of Colorado Boulder


MIDAS

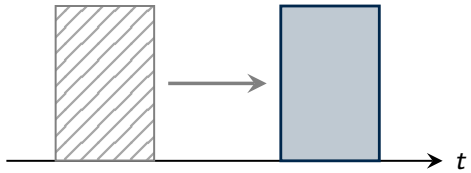
Schmidt Sciences

# Datacenter Power Flexibility




**Spatial shifting**

workload relocated across sites
(geographic load balancing)

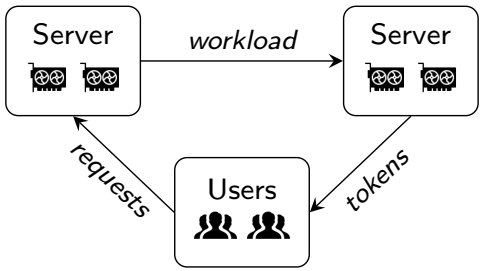



**Temporal shifting**

workload deferred in time
(interruptible training)

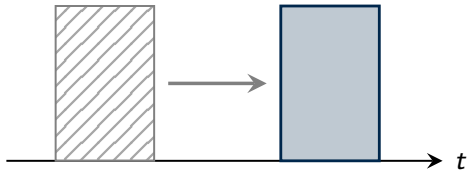
- Datacenter flexibility is conventionally realized by shifting workloads in space or in time
- Both displace workloads. Do not apply to workloads that can be neither migrated nor deferred

# Datacenter Power Flexibility



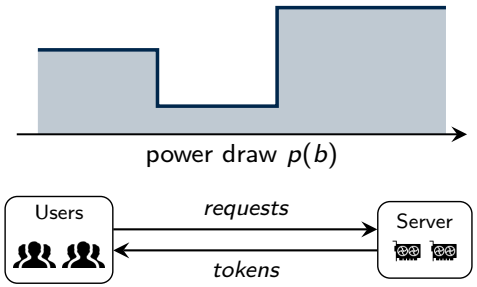

**Spatial shifting**

workload relocated across sites
(geographic load balancing)




**Temporal shifting**

workload deferred in time
(interruptible training)



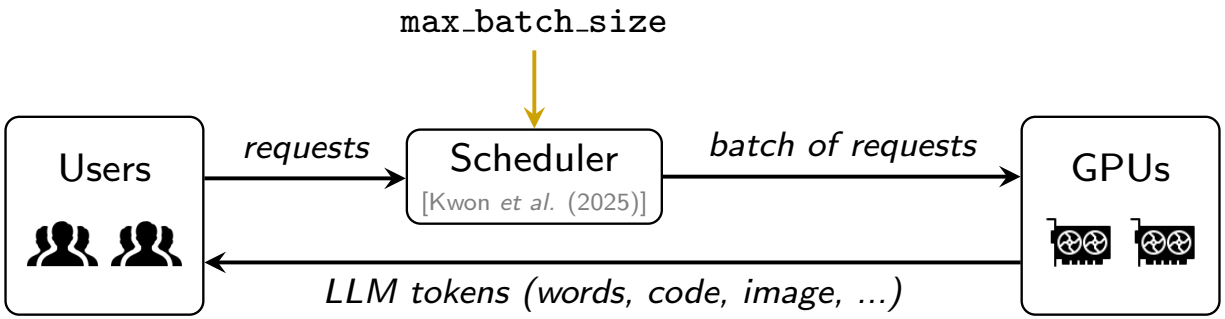

**Power modulation**

workload served at a controlled rate
(server-level knobs)

- Datacenter flexibility is conventionally realized by shifting workloads in space or in time
- Both displace workloads. Do not apply to workloads that can be neither migrated nor deferred

- Our work exposes real-time flexibility from **power modulation** of LLM inference
  - Realized by server-level control knobs: batch size, clock frequency, power caps
  - The same requests are served, trading response time against power draw
  - Power is increased or decreased in response to grid signals, not just curtailed

# LLM Inference as an Optimization Problem




The optimization problem of LLM inference datacenter:

- **Objective:** maximize revenue-per-GPU-hour → maximize throughput (tokens/sec)
- **Constraints:** inter-token latency (ITL), time-to-first-token (TTFT), ...

**Batch size** – number of user requests processed concurrently on the same GPU server

# Performance Trade-Offs in LLM Inference

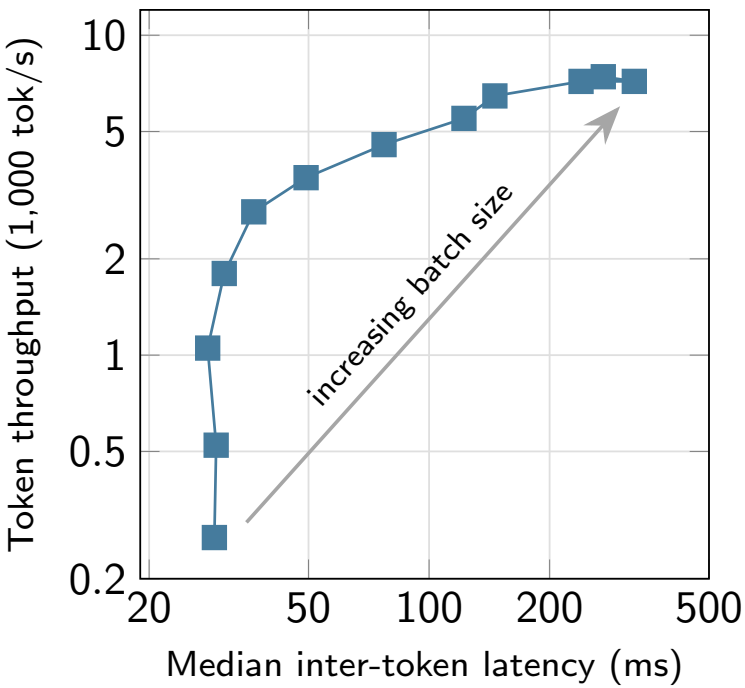



- NVIDIA DGX B200
- 8 Blackwell-type GPUs, each peaking at 1kW
- Open-weight Llama v3.1, 405B parameters
- `lm-arena-chat`: LMArena Human Preference dataset

# Performance Trade-Offs in LLM Inference

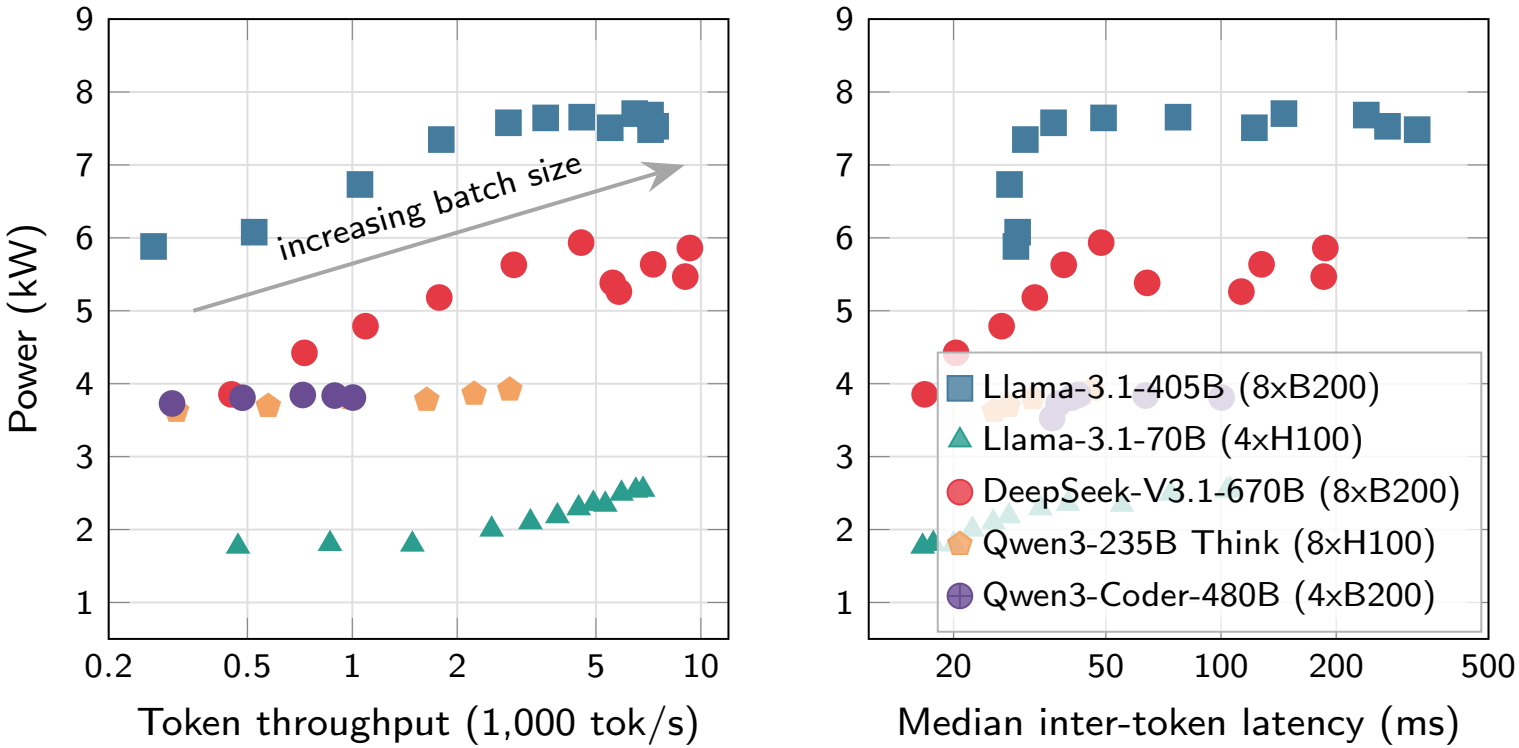


- NVIDIA DGX B200
- 8 Blackwell-type GPUs, each peaking at 1kW
- Open-weight Llama v3.1, 405B parameters
- `lm-arena-chat`: LMArena Human Preference dataset



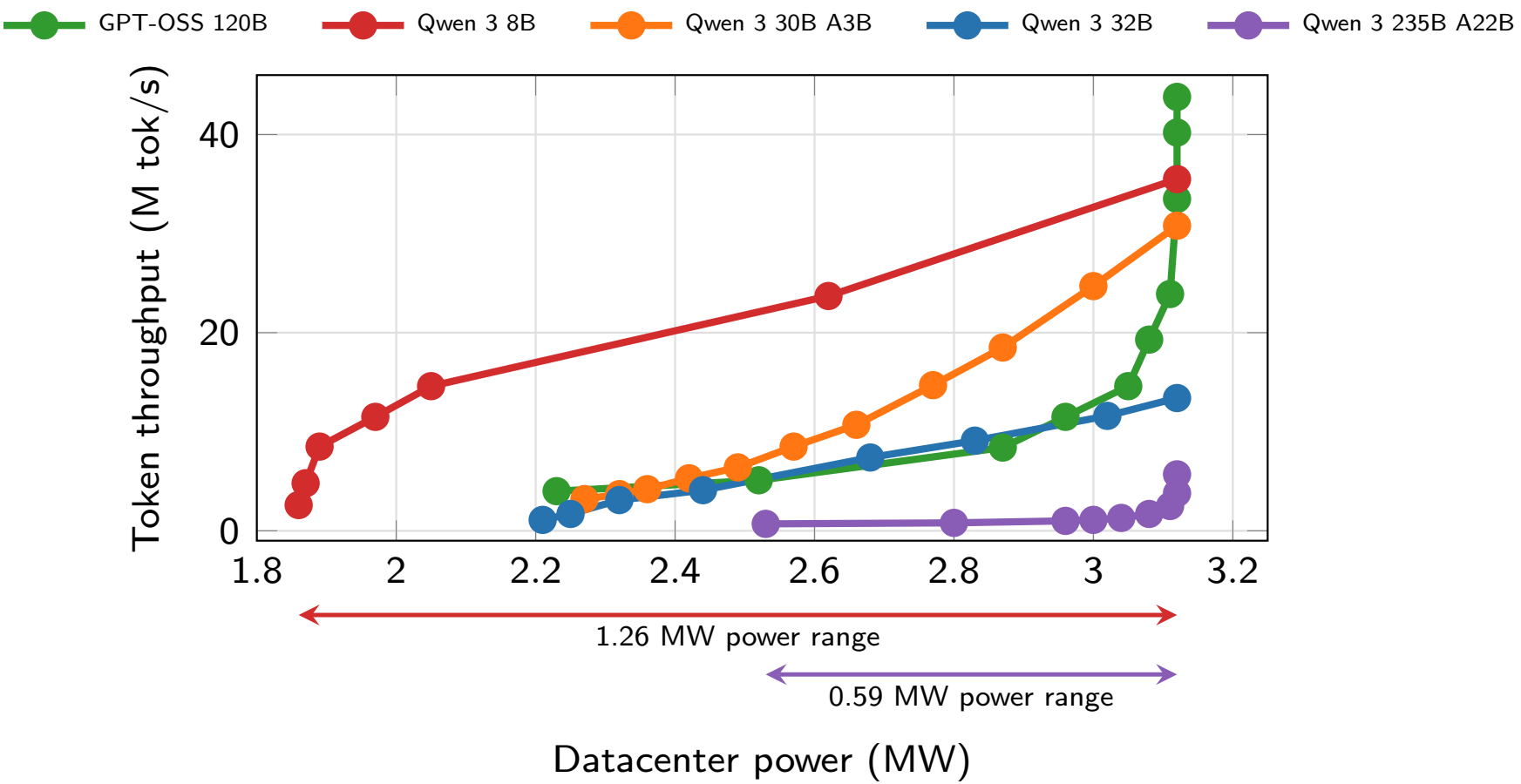

- Token throughput vs latency trade-off enabled by the batch size $\in [8, 1024]$
- More tokens served $\rightarrow$ slower model response
- Inter-token latency (ITL) standards:
  - Voice/real-time agent: $<$ 50 ms/token
  - Conversational chats: $\approx$ 50-150 ms/token
  - Offline (background) agents: unconstrained

# Power-Performance Trade-Offs in LLM Inference




- Power-performance trade-offs enabled by the batch size control
- Fewer tokens served $\rightarrow$ faster model response AND smaller power draw
- Example: power draw for Llama v3.1 on 8xB200 GPU stack $\in [5.88, 7.70]$ kW

# Same Trade-Off at a Larger Scale



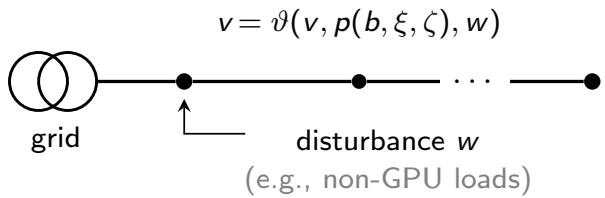

- Small datacenter, or one feeder branch of a larger facility's internal AC grid
- Same trade-offs at scale: five models on B200 GPUs, many replicas, common ≈3.1 MW peak
- Horizontal span = feasible power range = flexibility available to the grid

**Our work operationalizes these power-performance trade-offs for grid support**

# Control of the Integrated GPU-Grid Plant

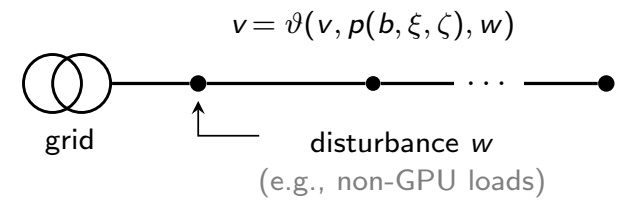
- Distribution feeder given by voltage model $\vartheta$

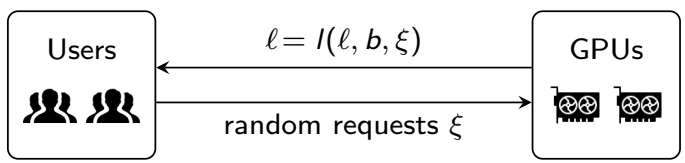
  (current voltage, injection, dist.) $\mapsto$ new voltage



# Control of the Integrated GPU-Grid Plant

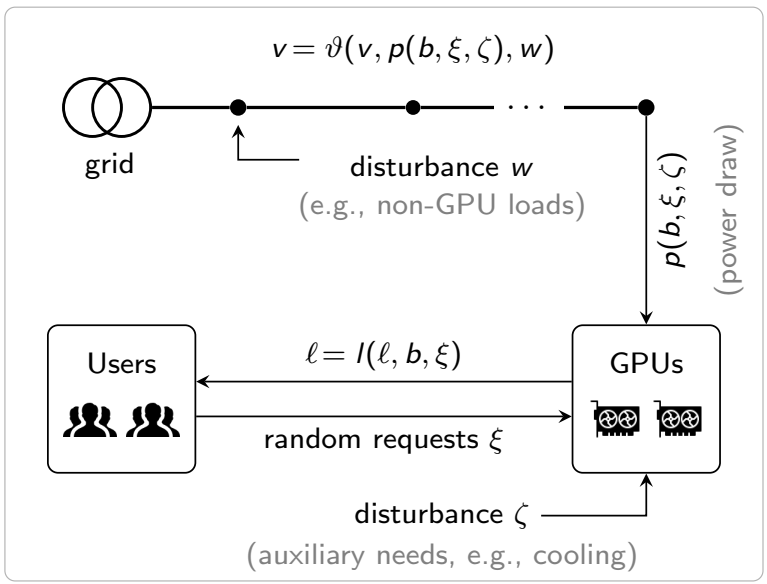
- Distribution feeder given by voltage model $\vartheta$
  (current voltage, injection, dist.) $\mapsto$ new voltage
- LLM inference given by latency model $l$
  (current latency, batch size, requests) $\mapsto$ new latency



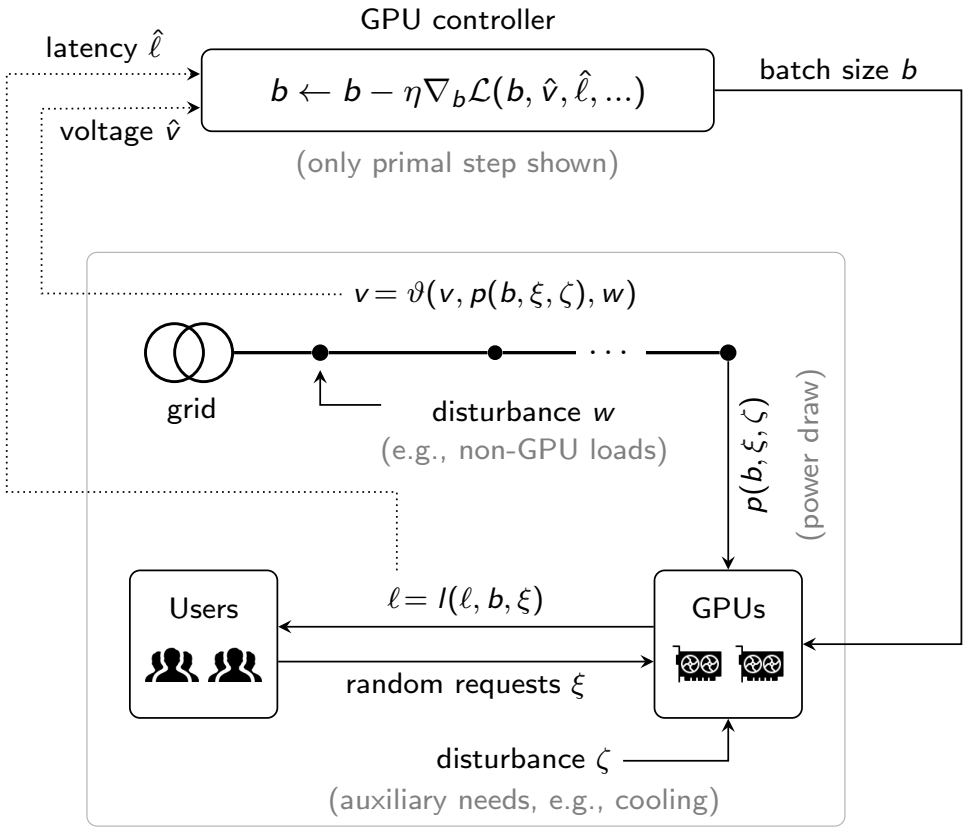
# Control of the Integrated GPU-Grid Plant

- Distribution feeder given by voltage model $\vartheta$
  (current voltage, injection, dist.) $\mapsto$ new voltage
- LLM inference given by latency model $l$
  (current latency, batch size, requests) $\mapsto$ new latency
- Power draw model $p$ couples grid and GPUs
  (batch size, requests, aux needs) $\mapsto$ power draw



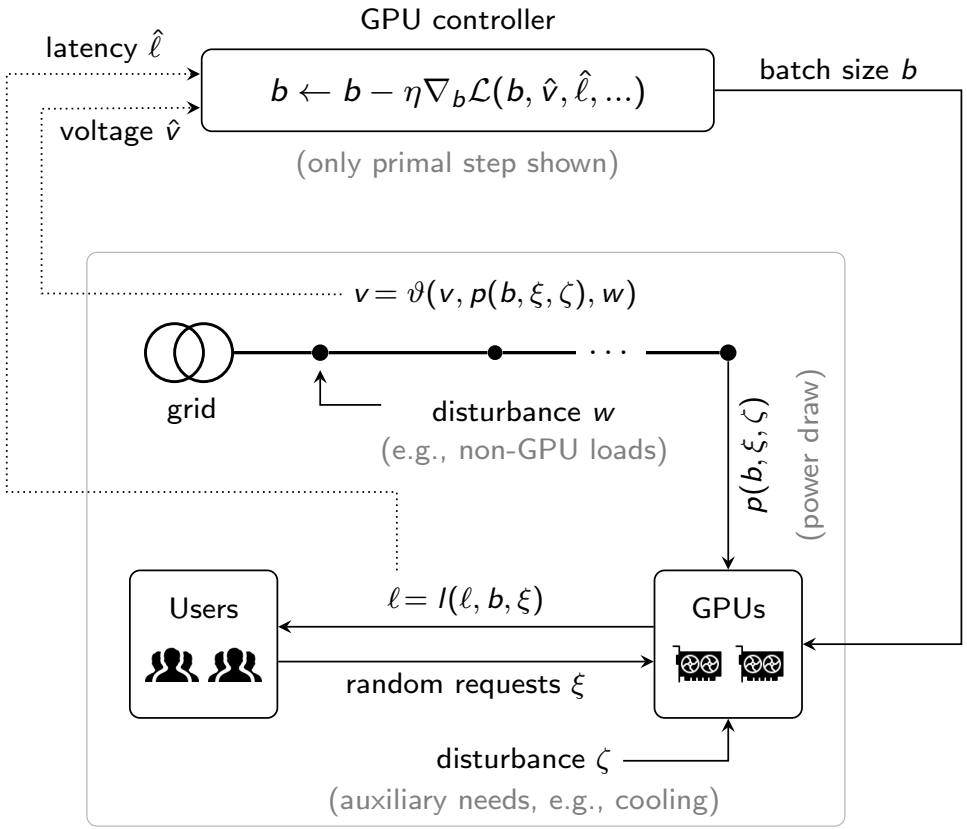
# Control of the Integrated GPU-Grid Plant

- Distribution feeder given by voltage model $\vartheta$
  (current voltage, injection, dist.) $\mapsto$ new voltage
- LLM inference given by latency model $l$
  (current latency, batch size, requests) $\mapsto$ new latency
- Power draw model $p$ couples grid and GPUs
  (batch size, requests, aux needs) $\mapsto$ power draw
- GPU (batch size) control using measurements
  **voltage** measurements $\hat{v}$ from the power grid
  **latency** measurements $\hat{\ell}$ from the GPU system



# Control of the Integrated GPU-Grid Plant

- Distribution feeder given by voltage model $\vartheta$
  (current voltage, injection, dist.) $\mapsto$ new voltage
- LLM inference given by latency model $l$
  (current latency, batch size, requests) $\mapsto$ new latency
- Power draw model $p$ couples grid and GPUs
  (batch size, requests, aux needs) $\mapsto$ power draw
- GPU (batch size) control using measurements
  **voltage** measurements $\hat{v}$ from the power grid
  **latency** measurements $\hat{\ell}$ from the GPU system




**The GPU controller closes the loop between grid and GPU operation**

# Offline Grid-Aware Optimization of LLM Inference

- *Objective:* maximize revenue-per-GPU-hour $\rightarrow$ maximize throughput (tokens/sec)
- *Constraints:* inter-token latency and grid voltage limits
- *Decision variable:* batch size

$$
\begin{aligned}
b_{k+1} \in \underset{b \in \mathcal{B}}{\operatorname{argmax}} \quad & r(b) && \textit{max token throughput} \\
\text{s.to} \quad & l(\ell_k, b, \xi_k) \leqslant \bar{\ell} && \textit{inter-token latency limit} \\
& \underline{v} \leqslant \vartheta(v_k, p(b, \xi_k, \zeta_k), w_k) \leqslant \overline{v} && \textit{grid voltage limits}
\end{aligned}
$$

# Offline Grid-Aware Optimization of LLM Inference

- *Objective:* maximize revenue-per-GPU-hour $\rightarrow$ maximize throughput (tokens/sec)
- *Constraints:* inter-token latency and grid voltage limits
- *Decision variable:* batch size

$$
\begin{aligned}
b_{k+1} \in \underset{b \in \mathcal{B}}{\operatorname{argmax}} \;\; & r(b) && \textit{max token throughput} \\
\text{s.to} \;\; & l(\ell_k, b, \xi_k) \leqslant \bar{\ell} && \textit{inter-token latency limit} \\
& \underline{v} \leqslant \vartheta(v_k, p(b, \xi_k, \zeta_k), w_k) \leqslant \overline{v} && \textit{grid voltage limits}
\end{aligned}
$$

**Why this cannot be solved as written:**

- Requires the feeder model $\vartheta$ (topology, impedances) – not available to datacenter operators
- Requests $\xi_k$ and disturbances $w_k, \zeta_k$ are unknown ahead of time and vary at (sub)second scale
- Open-loop: any error in the assumed latency $l$ or grid $\vartheta$ model is never corrected

# Online Feedback Optimization (OFO) Solution

Lagrangian, with duals $\underline{\lambda}, \overline{\lambda}$ on the voltage limits and $\mu$ on the latency limit:

$$\begin{aligned}\mathcal{L}(b, \underline{\lambda}, \overline{\lambda}, \mu) = & - r(b) + \mu^\top \big(l(\ell_k, b, \xi_k) - \overline{\ell}\big) \\ & + \overline{\lambda}^\top \big[\vartheta(v_k, p(b, \xi_k, \zeta_k), w_k) - \overline{v}\big] + \underline{\lambda}^\top \big[\underline{v} - \vartheta(v_k, p(b, \xi_k, \zeta_k), w_k)\big]\end{aligned}$$

# Online Feedback Optimization (OFO) Solution

Lagrangian, with duals $\underline{\lambda}, \overline{\lambda}$ on the voltage limits and $\mu$ on the latency limit:

$$\begin{aligned}\mathcal{L}(b, \underline{\lambda}, \overline{\lambda}, \mu) = & - r(b) + \mu^\top \big(l(\ell_k, b, \xi_k) - \overline{\ell}\big) \\ & + \overline{\lambda}^\top \big[\vartheta(v_k, p(b, \xi_k, \zeta_k), w_k) - \overline{v}\big] + \underline{\lambda}^\top \big[\underline{v} - \vartheta(v_k, p(b, \xi_k, \zeta_k), w_k)\big]\end{aligned}$$

The voltage and latency models cannot be evaluated but can be *measured*:

$$\underbrace{\vartheta(v_k, p(b, \xi_k, \zeta_k), w_k)}_{\text{needs } \vartheta,\ w_k,\ \zeta_k} \longrightarrow \hat{v}_k \qquad \underbrace{l(\ell_k, b, \xi_k)}_{\text{needs } \xi_k} \longrightarrow \hat{\ell}_k$$

# Online Feedback Optimization (OFO) Solution

Lagrangian, with duals $\underline{\lambda}, \overline{\lambda}$ on the voltage limits and $\mu$ on the latency limit:

$$\begin{aligned}\mathcal{L}(b, \underline{\lambda}, \overline{\lambda}, \mu) = & -r(b) + \mu^\top \big(l(\ell_k, b, \xi_k) - \bar{\ell}\big) \\ & + \overline{\lambda}^\top \big[\vartheta(v_k, p(b, \xi_k, \zeta_k), w_k) - \overline{v}\big] + \underline{\lambda}^\top \big[\underline{v} - \vartheta(v_k, p(b, \xi_k, \zeta_k), w_k)\big]\end{aligned}$$

The voltage and latency models cannot be evaluated but can be *measured*:

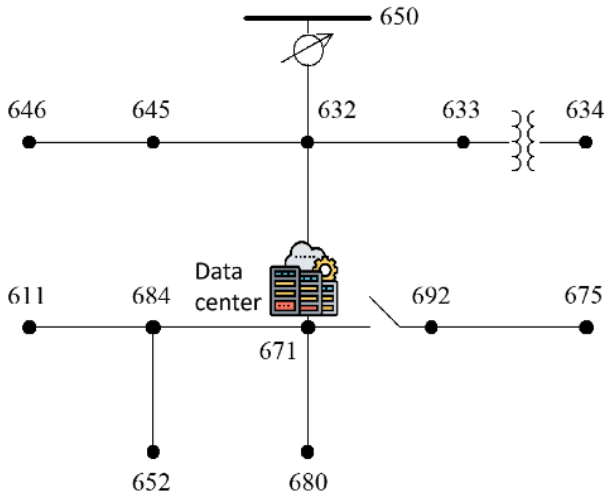
$$\underbrace{\vartheta(v_k, p(b, \xi_k, \zeta_k), w_k)}_{\text{needs } \vartheta,\ w_k,\ \zeta_k} \longrightarrow \hat{v}_k \qquad \underbrace{l(\ell_k, b, \xi_k)}_{\text{needs } \xi_k} \longrightarrow \hat{\ell}_k$$

Online feedback optimization (OFO) controller:

1: **at each control interval** $k$
2: measure $\hat{v}_k$ (grid), $\hat{\ell}_k$ (GPUs)
3: $\overline{\lambda}_{k+1} \leftarrow \big[\overline{\lambda}_k + \rho_v(\hat{v}_k - \overline{v})\big]_+$
4: $\underline{\lambda}_{k+1} \leftarrow \big[\underline{\lambda}_k + \rho_v(\underline{v} - \hat{v}_k)\big]_+$
5: $\mu_{k+1} \leftarrow \big[\mu_k + \rho_l(\hat{\ell}_k - \bar{\ell})\big]_+$
6: $b_{k+1} \leftarrow \Pi_{\mathcal{B}}\big(b_k - \rho_b \nabla_b \mathcal{L}(b_k, \underline{\lambda}_{k+1}, \overline{\lambda}_{k+1}, \mu_{k+1})\big)$
7: apply $b_{k+1}$ to the scheduler

- No feeder model (only $\partial\vartheta/\partial p$)
- Corrects model mismatch for $l$ and $\vartheta$
- One gradient step (no solver)
- Tracks the optimal KKT point
- Runs on (sub)second timescales

# Application to Volt/Watt Regulation in Distribution Systems



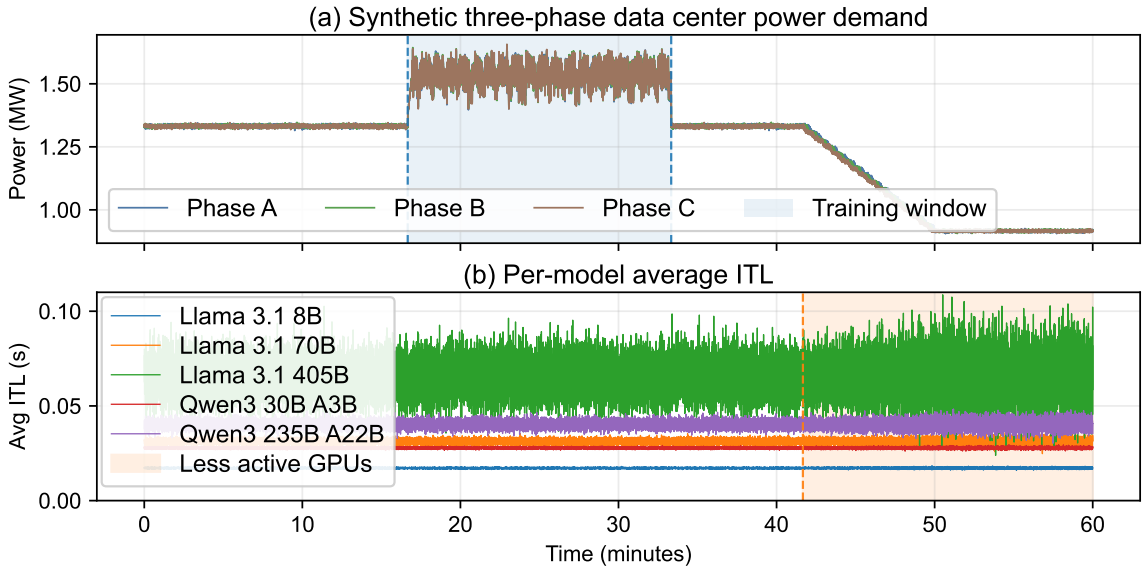

IEEE 13-Bus $3\phi$ Feeder

Parameters of LLM Inference Deployment

| Model | Replica count | GPUs per replica | $\bar{\ell}$ (ms) |
|---|---|---|---|
| Llama 3.1 8B | 720 | 1 | 80 |
| Llama 3.1 70B | 180 | 4 | 100 |
| Llama 3.1 405B | 90 | 8 | 120 |
| Qwen3-30B A3B | 480 | 2 | 60 |
| Qwen3 235B A22B | 210 | 8 | 140 |

Realistic deployment with $\approx$ 900 servers (peaking at 5MW)

# Building Realistic Datacenter Power Profiles



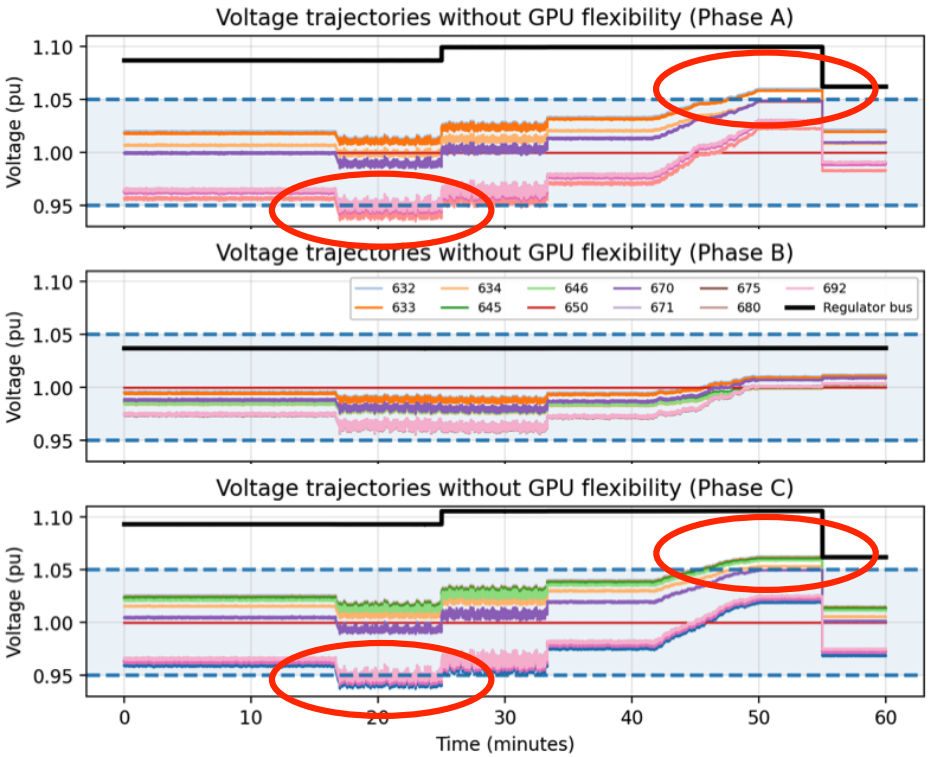

Fixed batch size 128 (no-control benchmark)

**Modeling realism:**

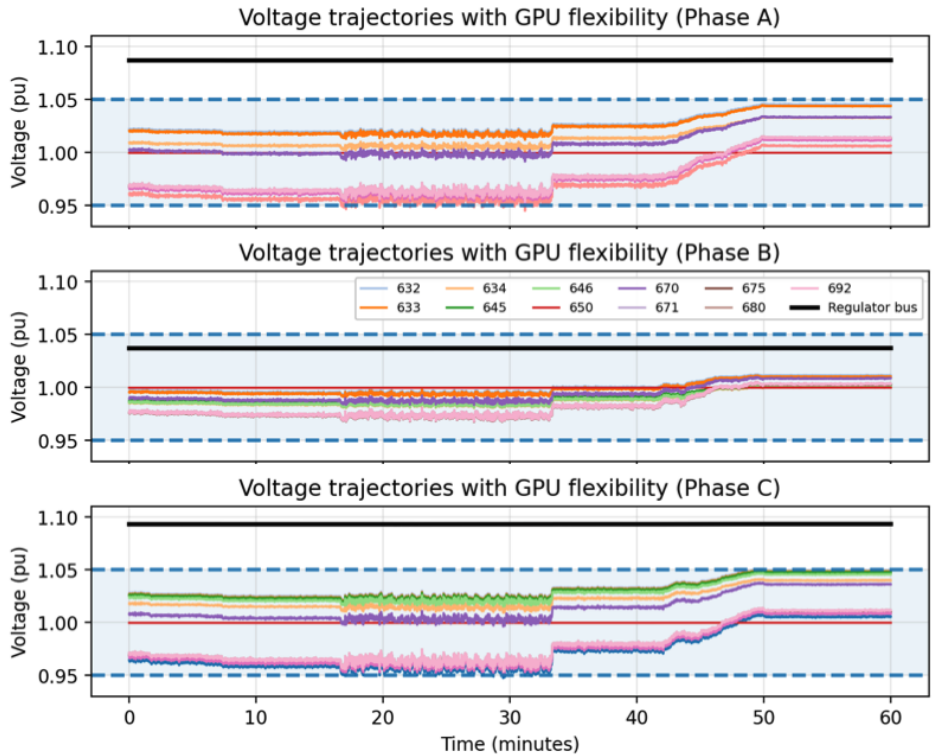
- Superimpose *replica-level* traces with randomized start times
- Not a single trace scaled up → avoids artificial transients
- ITL sampled from fitted **two-component lognormal mixtures** (heavy-tailed)
- ≈5 MW, 900 servers, 3 phases, 0.1 s resolution, 60 min horizon

Reproducible pipeline released in **OpenG2G** (see references)

# Results - Grid Operation




**w/o GPU control**

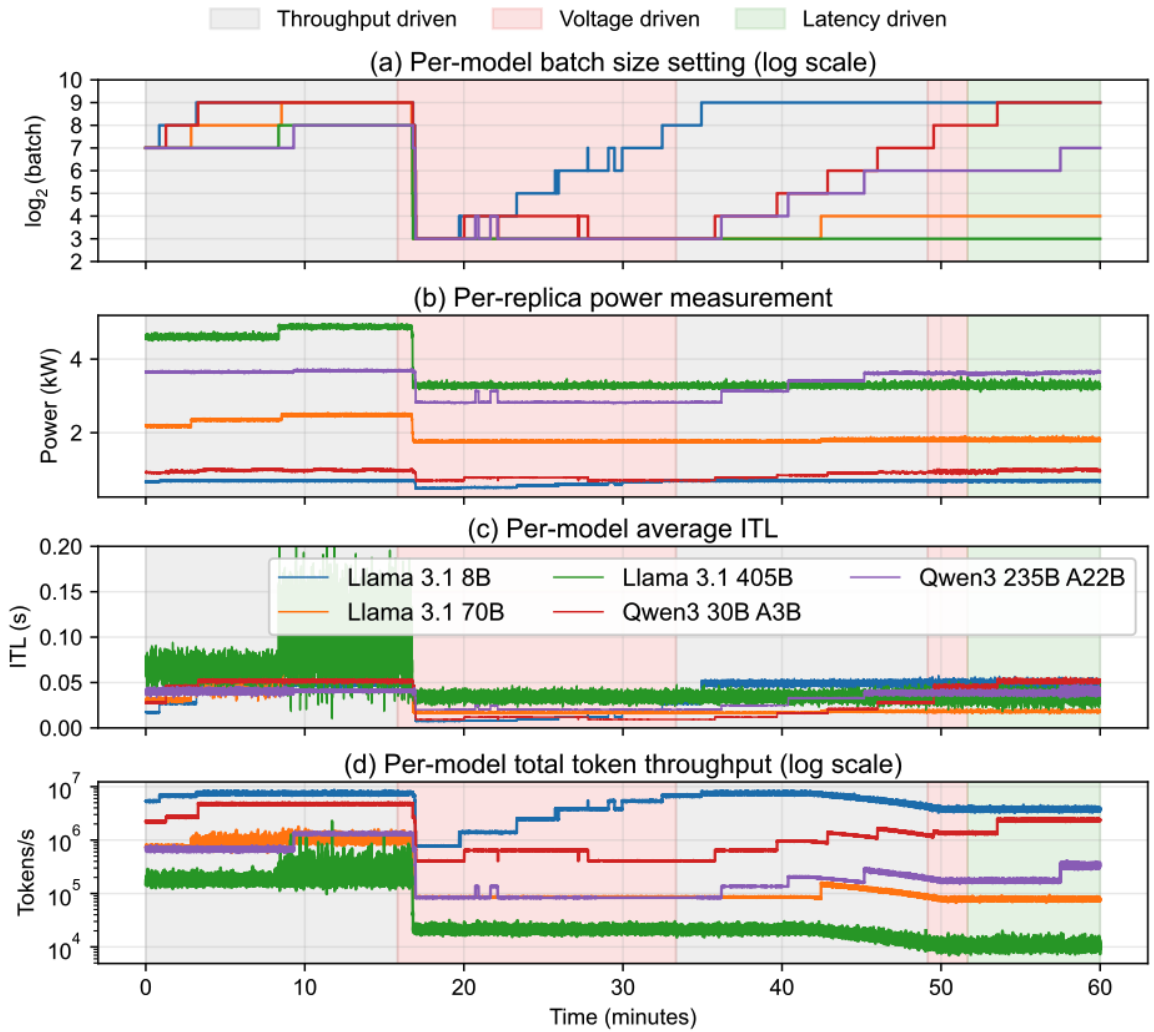



**w/ GPU control**

- LLM inference is sufficiently flexible to maintain voltages within $\pm 5\%$ limits
- With grid-aware GPU control, the tap changer is never used by the system operator

**Datacenter proves itself as a grid resource on both counts!**

# Results - Datacenter Operation




**OFO modulates batch size for each model to maximize throughput while meeting inter-token latency and grid voltage constraints.**

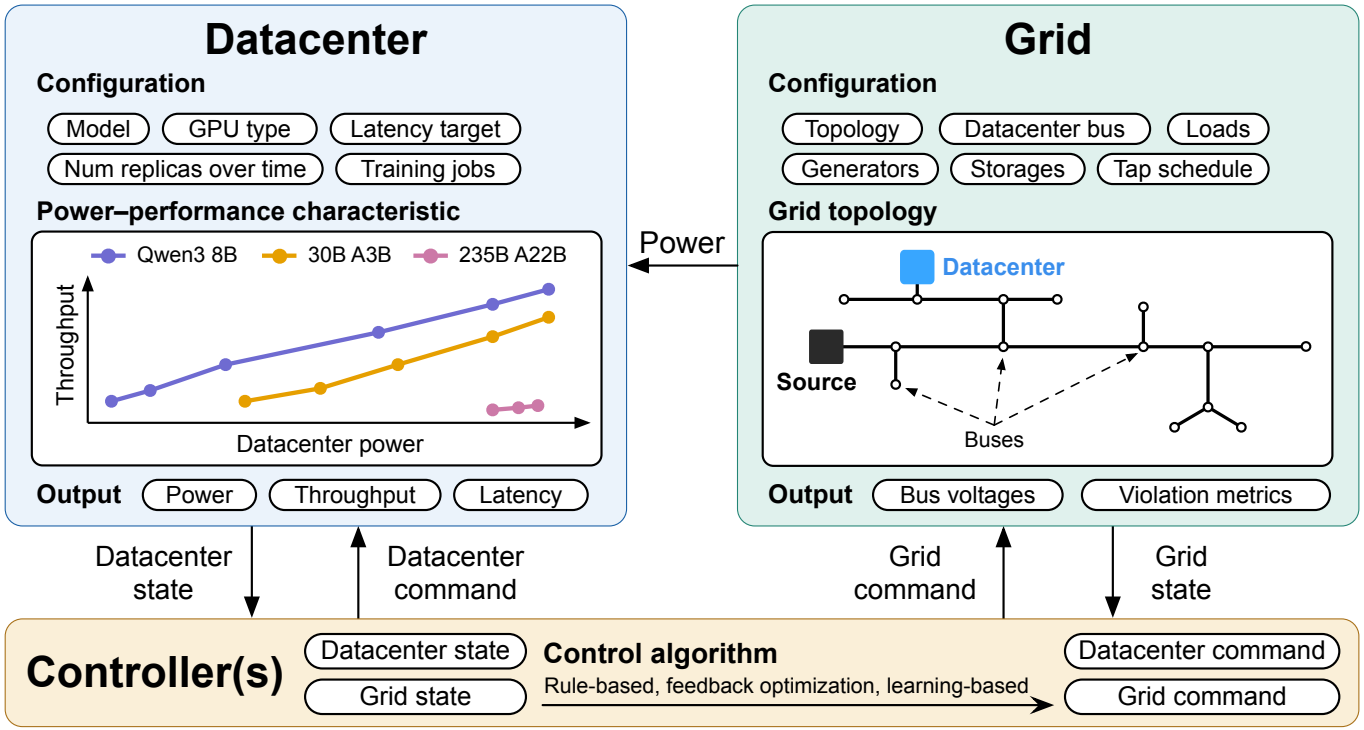
# Voltage Regulation Performance

| Case | Violation time (s) | Worst $V_{\min}$ (pu) | Worst $V_{\max}$ (pu) | Integral viol. (pu·s) |
|---|---|---|---|---|
| No control, no tap | 994 | 0.935 | 1.043 | 30.86 |
| Tap change only | 1,005 | 0.935 | 1.057 | 22.15 |
| **GPU control only** | **62** | **0.945** | **1.045** | **0.08** |

- Integral violation reduced by **more than two orders of magnitude**
- Achieved with **zero tap operations** over the horizon
- Tap-only is *worse* than no control: actuation delay + overcorrection
- Overvoltage is corrected by **increasing** GPU power (more tokens served, better grid support)

*Minimizing GPU power is not always favorable to the grid!*

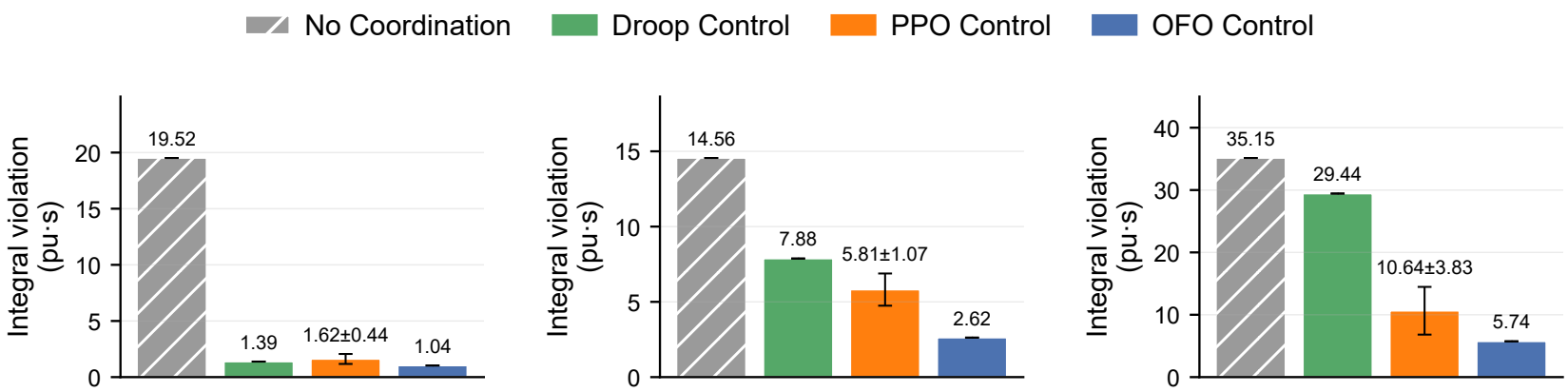
# Follow-up Work: OpenG2G Platform for Real-Time Coordination




- Datacenter backend driven by **real production-grade measurements**
- Grid backend wraps OpenDSS and other simulators
- Benchmarking different controllers on identical scenarios

`https://gpu2grid.io` · `github.com/gpu2grid/openg2g`

# OpenG2G: Controller Benchmarking Results

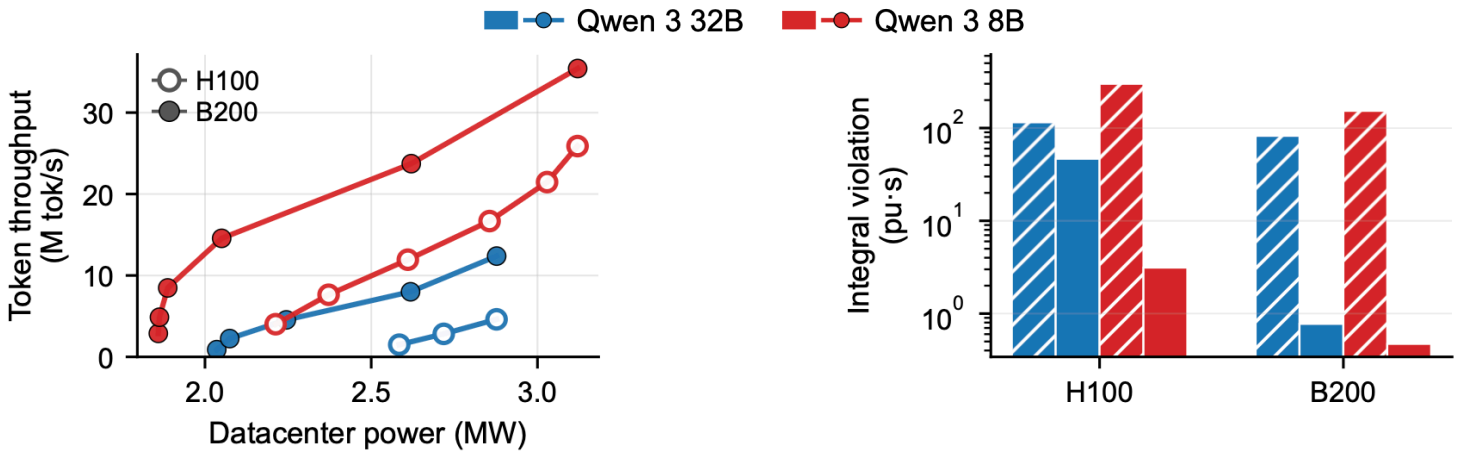



(a) IEEE 13-bus (b) IEEE 34-bus (c) IEEE 123-bus

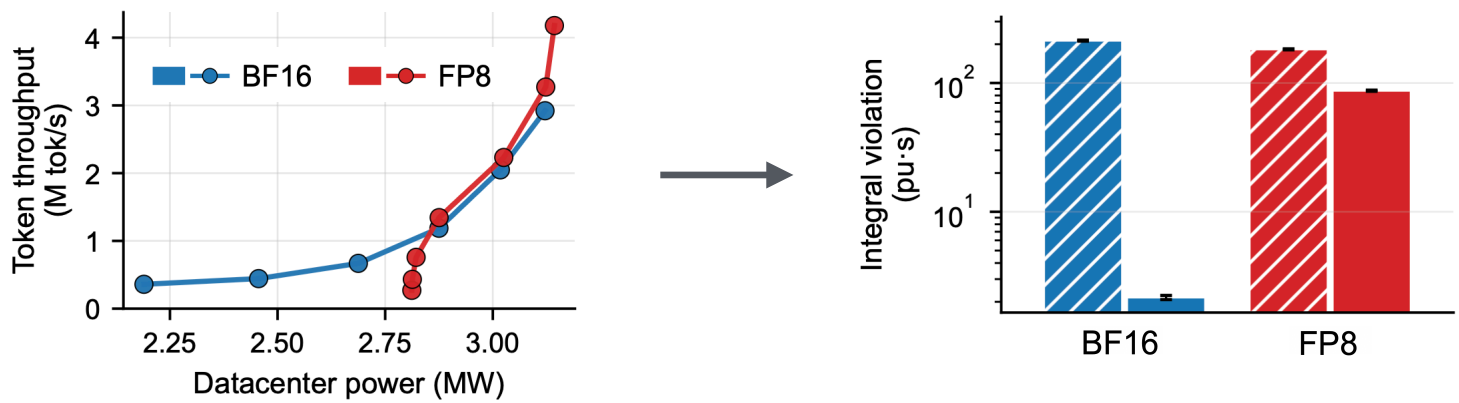
- Three controllers: Droop, Proximal Policy Optimization (PPO), and OFO
- Greater voltage regulation gains as network size grows
- More applications (e.g., in transmission networks) to come

# OpenG2G: Datacenter Flexibility Assessment for Grid Planning

- Power flexibility range depending on GPU stacks




- Power flexibility range depending on LLM weight precision



# Current work: Bilevel Control with Authority Separation

Bilevel optimization problem for hierarchical OFO control:

$$
\begin{aligned}
&\underset{\underline{v}^{\mathrm{r}} \leqslant v^{\mathrm{r}} \leqslant \overline{v}^{\mathrm{r}}}{\text{minimize}} && f_{\mathrm{v}}(\vartheta(v_k, p(b^\star(v^{\mathrm{r}}), \xi_k, \zeta_k), w_k)) && \textit{min voltage violations} \\
&\text{subject to} && b^\star(v^{\mathrm{r}}) \in \underset{\underline{b} \leqslant b \leqslant \overline{b}}{\text{argmax}} \; r(b) && \textit{max token throughput} \\
&&& \text{subject to} \;\; l(\ell_k, b, \xi_k) \leqslant \overline{\ell}, && \textit{satisfy latency constraints} \\
&&&\qquad\qquad |\vartheta^{\mathrm{r}}(v_k, p(b, \cdot), \cdot) - v^{\mathrm{r}}| \leqslant \varepsilon. && \textit{stay close to the voltage ref}
\end{aligned}
$$

- Upper level (grid): sets voltage reference $v^{\mathrm{r}}$ for DC
- Lower level (DC): tracks $v^{\mathrm{r}}$ by modulating GPU power
- Upper-level grid dynamics parametrizes lower-level DC dynamics
- Authority separation by design; two-timescale operation
- Minimum info exchange (local voltage model/meas, gradients)

# Conclusions

**A third axis of datacenter flexibility:**

- **Power modulation** while serving inference demand
- Batch size trades throughput and latency for power draw
- Batch size OFO controller acts on measurements and limited grid info

**Key results:**

- Reducing GPU power resolves undervoltage; **increasing** it resolves overvoltage
- Integral voltage violation cut by two orders of magnitude, zero tap operations
- Datacenters can independently resolve the voltage issues they induce

**Next:**

- Bilevel coordination respecting grid/datacenter authority boundaries
- Hardware-in-the-loop validation with live GPU feedback

## The GPU-to-Grid Project


1. Z. Liang, J.-W. Chung, M. Chowdhury, J. Chen, V. Dvorkin
   **GPU-to-Grid: Voltage Regulation via GPU Utilization Control**
   Proceedings of the PowerUp Conference, 2026
2. J.-W. Chung, Z. Liang, Y. Mao, M. Chowdhury, J. Chen, V. Dvorkin
   **OpenG2G: A Simulation Platform for AI Datacenter–Grid Runtime Coordination** (preprint)
3. Y. Mao, J. Mathieu, V. Dvorkin
   **Online Feedback Optimization of Energy Storage to Smooth Data Center Grid Impacts**
   North American Power Symposium, 2026
4. Z. Liang, J.-W. Chung, M. Chowdhury, J. Chen, V. Dvorkin
   **Grid-Aware Control of LLM Inference via Online Feedback Optimization** (in progress)
5. **GPU-to-Grid Project webpage** `https://gpu2grid.io`


**Hiring a postdoc!** Schmidt AI in Science Fellowship (U-M), deadline 23 Nov 2026. Happy to discuss during the conference.