UNIVERSITY OF ILLINOIS URBANA-CHAMPAIGN

MLSYS 2026 · BELLEVUE, WA

SuperInfer

SLO-Aware Rotary Scheduling and Memory Management for LLM Inference on Superchips

Jiahuan Yu SPEAKER
jiahuan2@illinois.edu

Mingtao Hu
mingtao4@illinois.edu

Zichao Lin
zichaol3@illinois.edu

Minjia Zhang
minjiaz@illinois.edu

SSAIL *(Supercomputing System AI Lab)* · ***University of Illinois Urbana–Champaign***

The Ninth Annual Conference on Machine Learning and Systems (**MLSys26**) · May 19, 2026

Research Questions

UNIVERSITY OF ILLINOIS URBANA-CHAMPAIGN

1. How do existing LLM inference engines execute on a Superchip?
2. How does this architecture shift the paradigm of LLM serving?
3. How do we materialize this potential into real performance improvements?

The PCIe Bottleneck

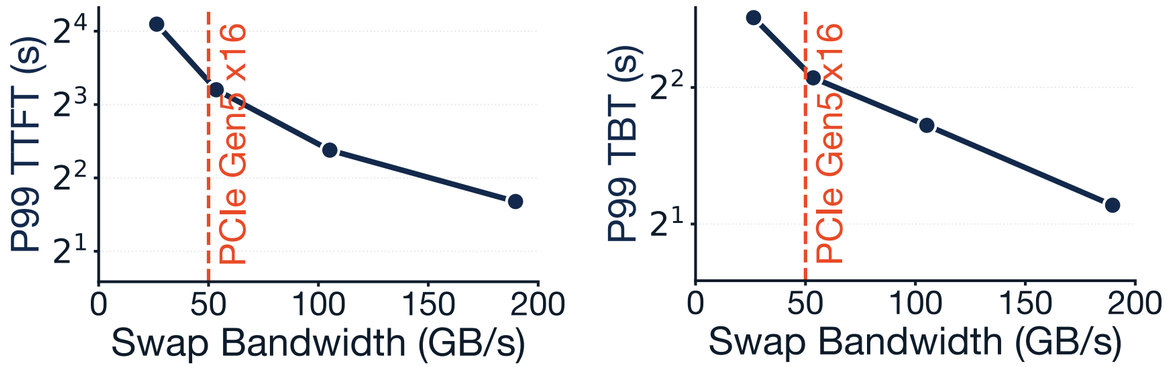
UNIVERSITY OF ILLINOIS URBANA-CHAMPAIGN

Memory capacity wall of LLM serving

- The KV cache grows linearly with request length, putting pressure on GPU HBM.
- When HBM is exhausted, new requests are blocked behind older ones → head-of-line (HOL) blocking, violating latency SLOs (Service Level Objectives).

PCIe bottleneck

- Natural idea: offloading the KV cache to CPU memory.
- But PCIe bandwidth becomes the bottleneck:
  - PCIe Gen5 x16 is only **~64 GB/s per direction** (H2D or D2H).



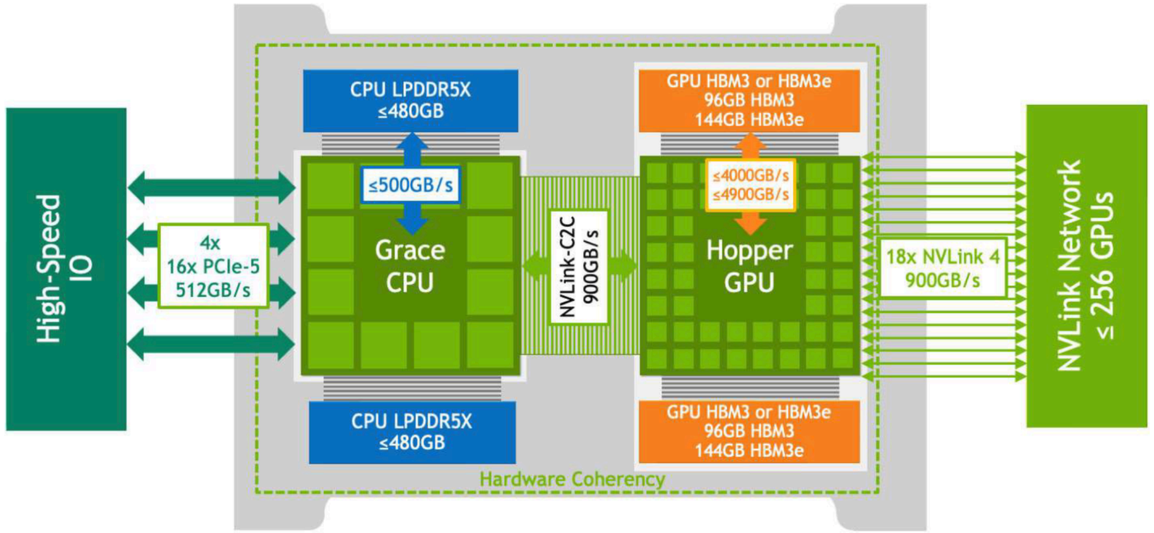
TTFT = Time to First Token · **TBT** = Time Between Tokens

Superchip Opportunity

Emerging CPU–GPU tightly-coupled Superchips (e.g., NVIDIA's GH200 and GB200) provide high-speed CPU–GPU interconnection.

- NVLink-C2C: **~900 GB/s**.
- PCIe Gen5 ×16: only **~128 GB/s**.

~7× bandwidth

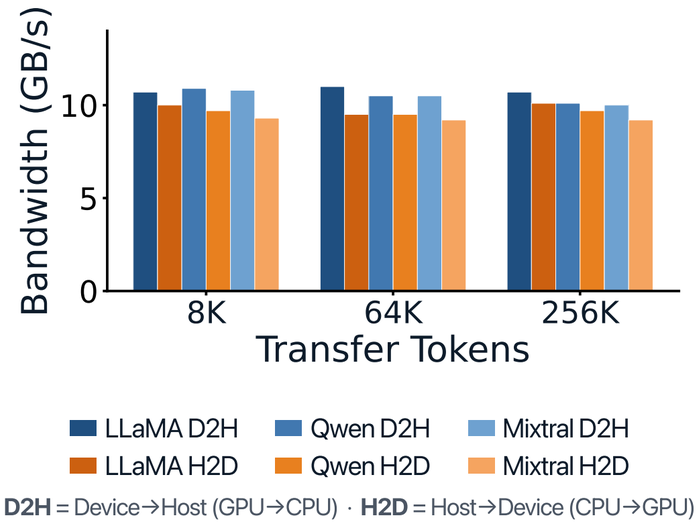


GH200 Grace Hopper Superchip architecture (source: NVIDIA datasheet)

The Underutilization of Superchips

Gap 1: cannot saturate NVLink-C2C

- Current LLM serving frameworks only utilize ~10 GB/s, **< 5%** of NVLink-C2C bandwidth.
- Root cause: PagedAttention scatters a request's KV cache into small non-contiguous blocks.

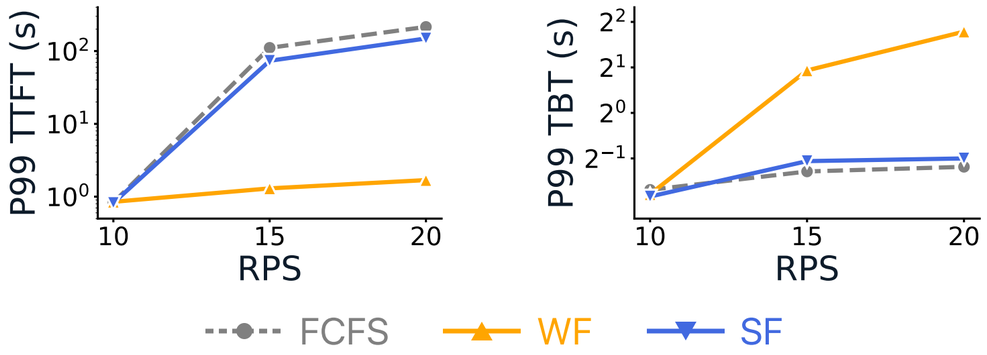


Gap 2: existing offloading is SLO-unaware

Common scheduling policies in production frameworks:

- **FCFS**: First-Come-First-Serve baseline.
- **WF** (Waiting-First): preempts running requests to favor new arrivals.
- **SF** (Swapped-First): resumes swapped requests before admitting new ones.

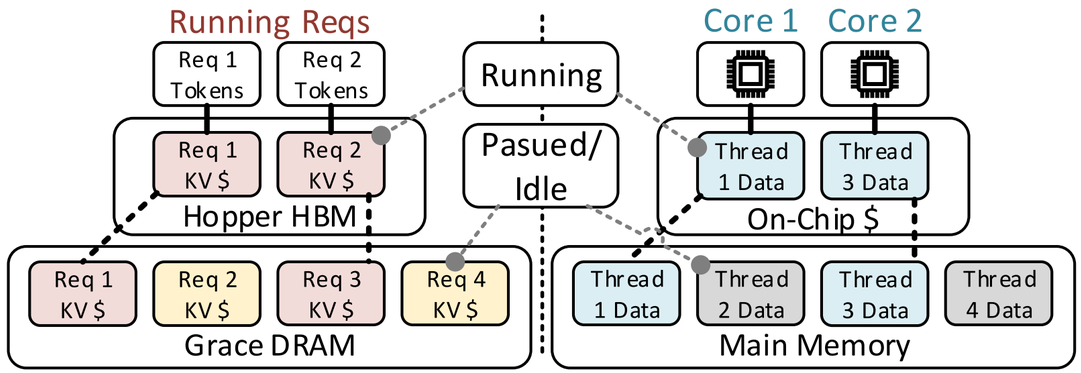
None achieves balanced TTFT and TBT.



Paradigm Shift: LLM Serving ↔ an OS

An LLM serving stack on a Superchip is analogous to an OS on a CPU:

- Request ↔ Thread
- Hopper HBM ↔ on-chip cache (fast, small)
- Grace DRAM ↔ main memory (slow, large)
- KV cache ↔ thread data



KEY TAKEAWAY

1. An OS hosts many threads without starvation via CPU time-slicing: threads are actively rotated and preempted under schedulers like CFS / EEVDF.
2. For LLM serving, instead of treating swapping as a *last resort to prevent OOM*, use it for proactive request rotation driven by SLO status.

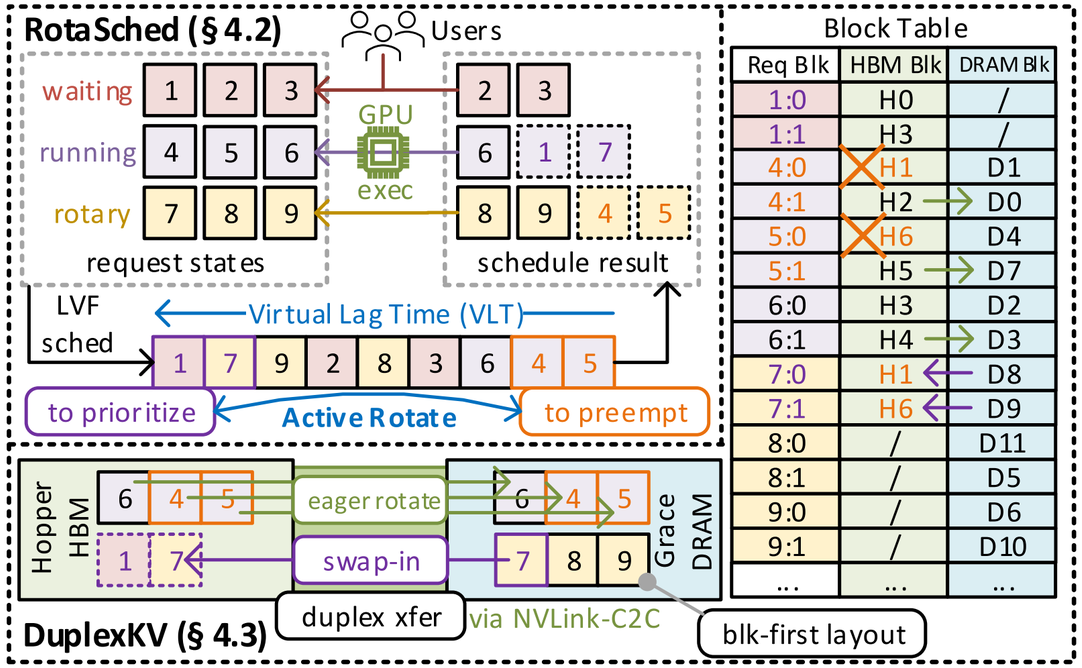
SuperInfer Architecture

RotaSched

OS-inspired proactive request rotation guided by Virtual Lag Time (VLT), moves requests between HBM and DRAM with SLO awareness.

DuplexKV

Saturates NVLink-C2C bandwidth by transferring KV cache via bidirectional concurrency and eliminating small-segment transfers.

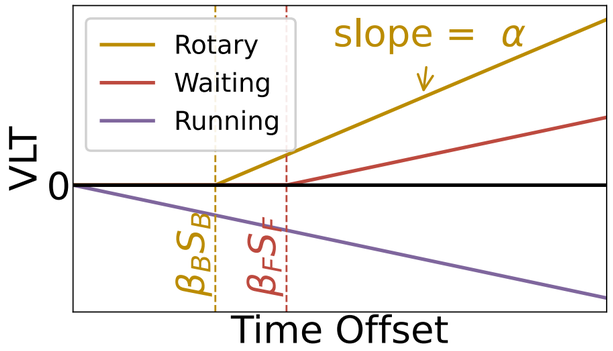


RotaSched: Proactive SLO-Aware Request Rotation

VLT (Virtual Lag Time)

One **urgency score** per request, re-computed every iteration. Higher VLT ↔ closer to SLO violation.

- **Running** (on HBM): negative. The more negative, the higher priority *to be preempted*.
- **Waiting / Rotary** (KV on CPU memory): positive. The larger the value, the closer to SLO violation, and the higher priority to *(re)admit*.

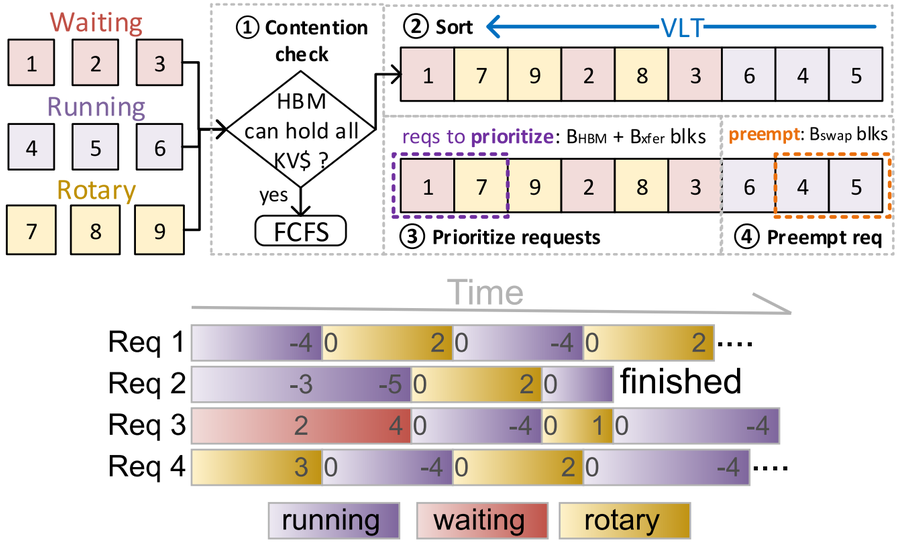


S_F, S_B: TTFT / TBT SLO targets. β_F, β_B (0–1): tolerance fraction. VLT stays at 0 while still within the safety margin, then rises. α: slope of rotary; larger α favors rotary (TBT) over waiting (TTFT).

LVF (Largest-VLT-First)

Algorithm core: in each iteration, recompute every request's VLT and sort. The number of swappable KV cache blocks B_{xfer} controls how many requests are preempted / (re)admitted.

- **(Re)admit** requests with the largest VLT.
- **Preempt** requests with the smallest VLT.



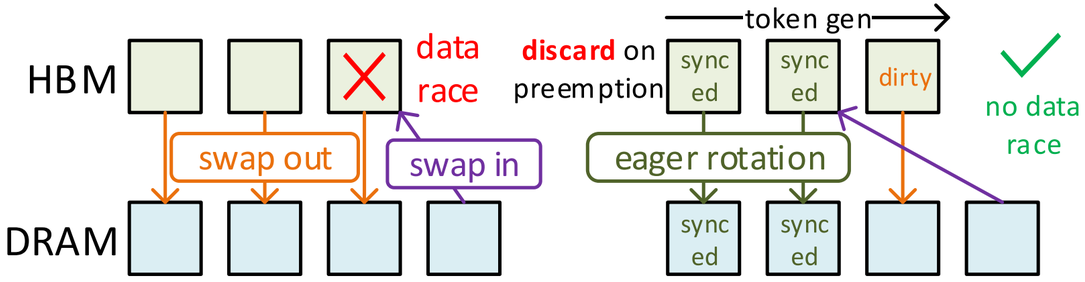
DuplexKV: Saturate NVLink-C2C

Eager Block Rotation

Naive KV offloading (e.g., vLLM): swap-out and swap-in target overlapping HBM blocks, so must serialize to avoid data races → **half-duplex**.

Ours is full-duplex:

- KV cache is **append-only**: once a block is full, it won't change until the request finishes.
- **Eager rotation**: sync each block to DRAM as soon as it is filled, even without preemption.
- At preempt time, synced blocks can be discarded directly → swap-in doesn't wait for swap-out.

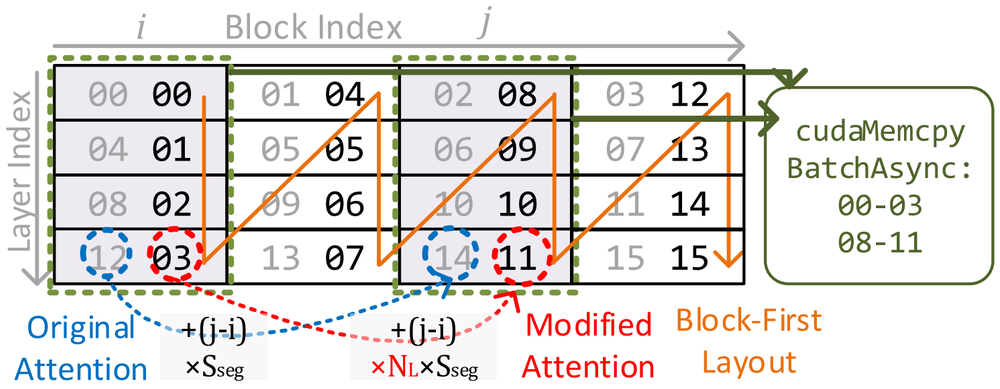


Block-First KV Layout

PagedAttention is layer-first: per-layer per-block contiguous memory is tiny (e.g., 64 KB) → one segment = one `cudaMemcpyAsync` → bandwidth under-utilization.

Ours:

- **Block-first layout**: all layers of a block are placed contiguously (64 KB → 4 MB for block_size=32, Qwen2.5-32B).
- `cudaMemcpyBatchAsync` to eliminate per-segment launch overhead.



Experiments

UNIVERSITY OF ILLINOIS URBANA-CHAMPAIGN

UP TO **+74.7%**

TTFT SLO attainment

CPU DRAM offload mitigates HOL blocking

≈ par

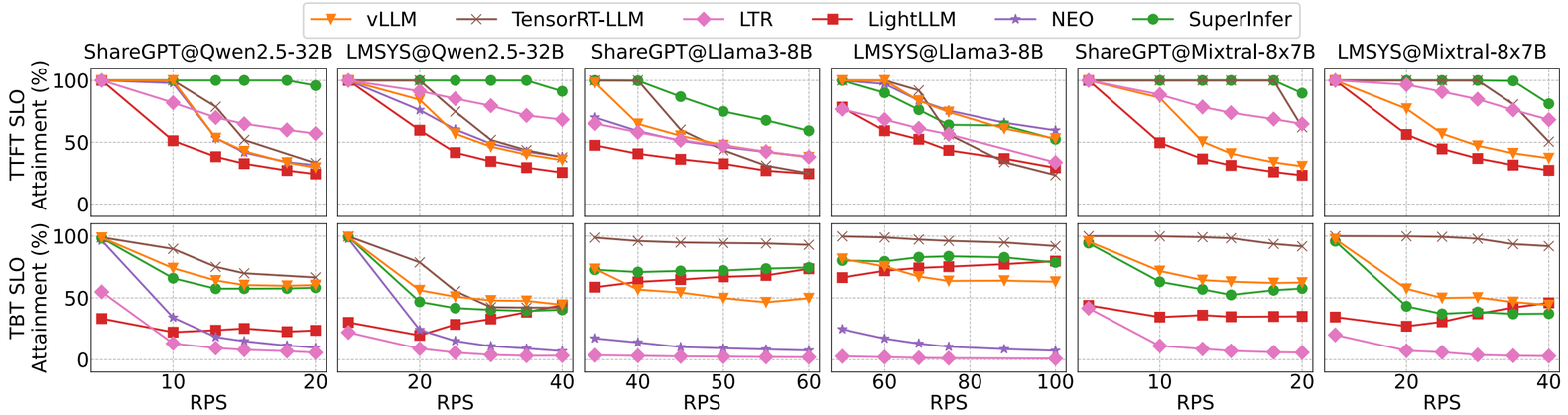
TBT SLO attainment

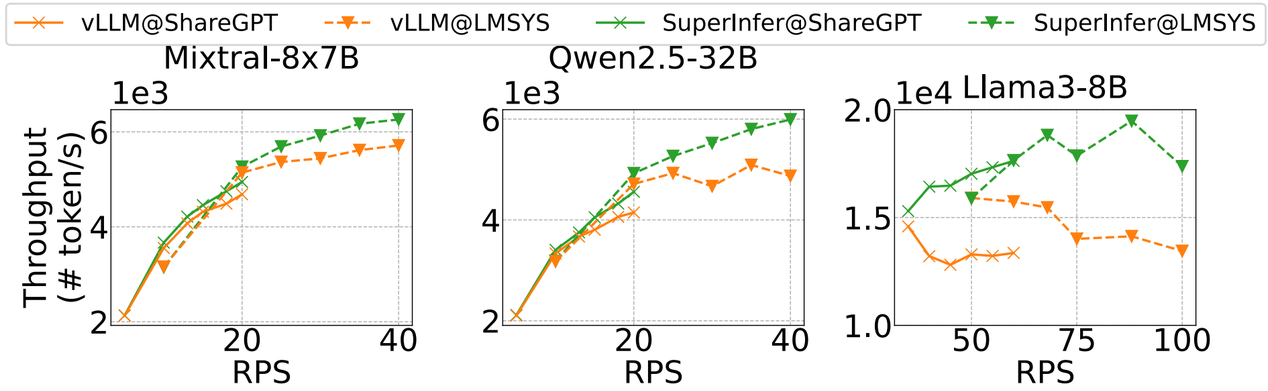
no regression on per-token latency

UP TO **+29.2%**

throughput

proactive rotation frees HBM → more chunked-batching





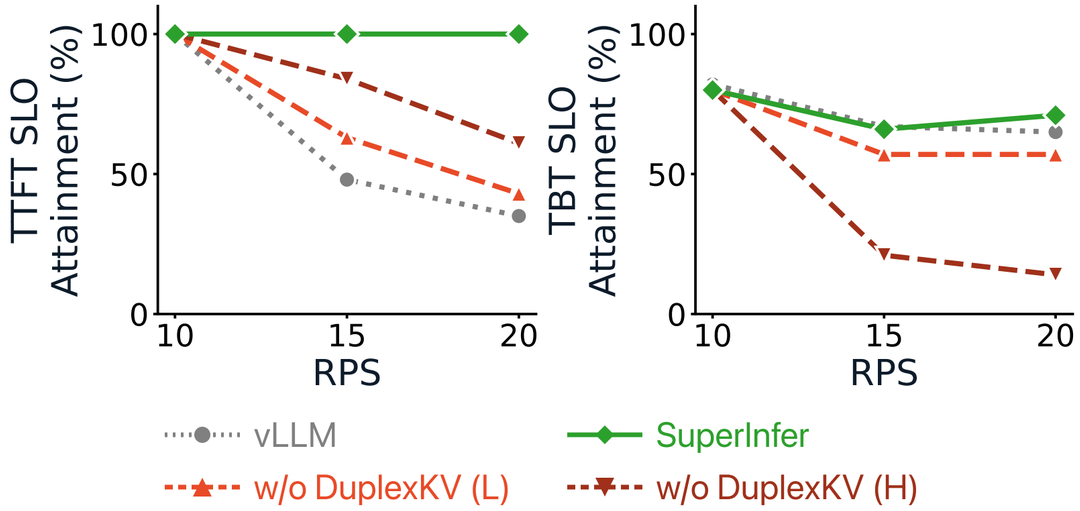
Setup: NVIDIA GH200 (144 GB HBM3e + 480 GB LPDDR5X) · **Models:** Llama3-8B, Qwen2.5-32B, Mixtral 8×7B · **Workloads:** ShareGPT, LMSYS-Chat-1M (Poisson) · **SLOs:** TTFT ≤ 5 s, TBT ≤ 100 ms · **Baselines:** vLLM, TensorRT-LLM, LightLLM, LTR, NEO

Ablation Studies

UNIVERSITY OF ILLINOIS URBANA-CHAMPAIGN

Contribution of each component

- **w/o DuplexKV (L)**, low swap budget (small B_{xfer}): slight TTFT improvement, bottlenecked by PCIe.
- **w/o DuplexKV (H)**, high swap budget (large B_{xfer}, over NVLink-C2C): large TTFT improvement, but significant TBT degradation due to C2C bandwidth underutilization.
- **Full SuperInfer**: best TTFT improvement with no TBT sacrifice.



NVLink-C2C bandwidth utilization

Transferring 16 GB of Qwen2.5-32B KV cache (8 GB per direction). **B** = bidirectional, **U** = uni-directional.

Method	D2H (GB/s)	H2D (GB/s)	E2E (ms)
Naive (vLLM)	10.75 (U)	9.86 (U)	1556.15
DuplexKV	**180.99 (B)**	**179.37 (B)**	**46.80**
Ideal (duplex)	*192.00 (B)*	*192.00 (B)*	*41.66*

Ceiling is DRAM, not C2C: Grace LPDDR5X sustains ~384 GB/s, split between read & write under duplex traffic → **192 GB/s per direction**.

~33×
faster KV transfer vs. naive

~94%
of ideal situation

Conclusion

UNIVERSITY OF ILLINOIS URBANA-CHAMPAIGN

We present SuperInfer, a high-performance, SLO-aware LLM serving system optimized for Superchips.

- Transforms passive preemption into proactive, fine-grained request scheduling, via an OS-inspired active rotary scheduler (RotaSched) and a co-designed rotation engine (DuplexKV), achieving high NVLink-C2C utilization.
- Across diverse models and workloads, SuperInfer substantially improves SLO attainment under high request rates while also improving throughput.
- Superchips unlock new opportunities for LLM serving, but realizing their full potential requires careful co-design of scheduling and memory movement in the software system.

UNIVERSITY OF ILLINOIS URBANA-CHAMPAIGN

THANK YOU

Q&A

SuperInfer: SLO-Aware Rotary Scheduling and Memory Management for LLM Inference on Superchips

Jiahuan Yu SPEAKER
jiahuan2@illinois.edu

Mingtao Hu
mingtao4@illinois.edu

Zichao Lin
zichaol3@illinois.edu

Minjia Zhang
minjiaz@illinois.edu

SSAIL *(Supercomputing System AI Lab)* · ***University of Illinois Urbana–Champaign***