



CONFERENCE

—NORTH AMERICA 2026—

State-of-the-Art KV Transfer for Disaggregated LLM Serving in vLLM

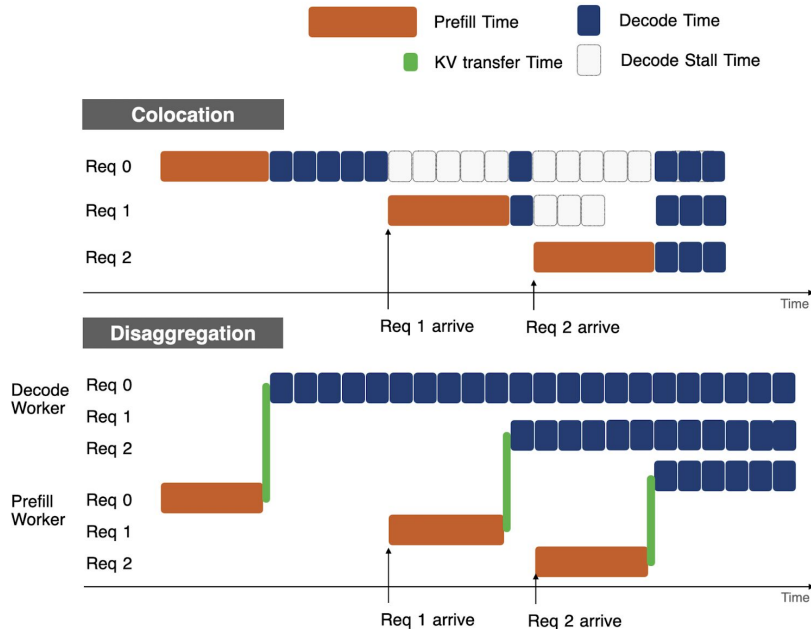
Nicolo' Lucchesi
Research Engineer
Mistral

Sunita Nadampalli
Principal Engineer
AWS

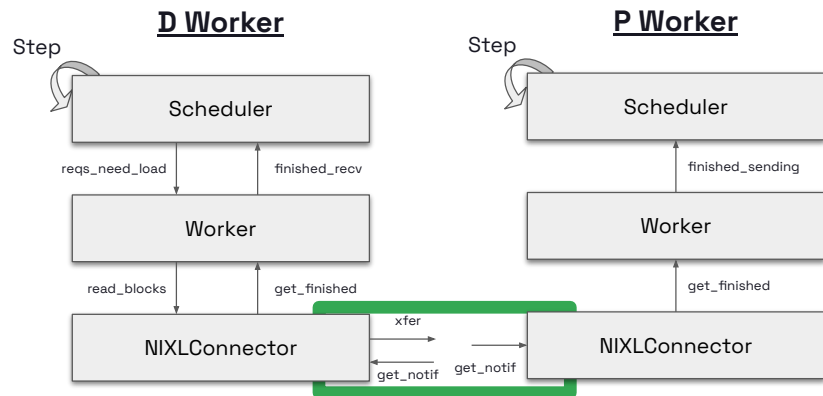
Zhanqiu Hu
Machine Learning Engineer
Red Hat

From Colocated to Disaggregated

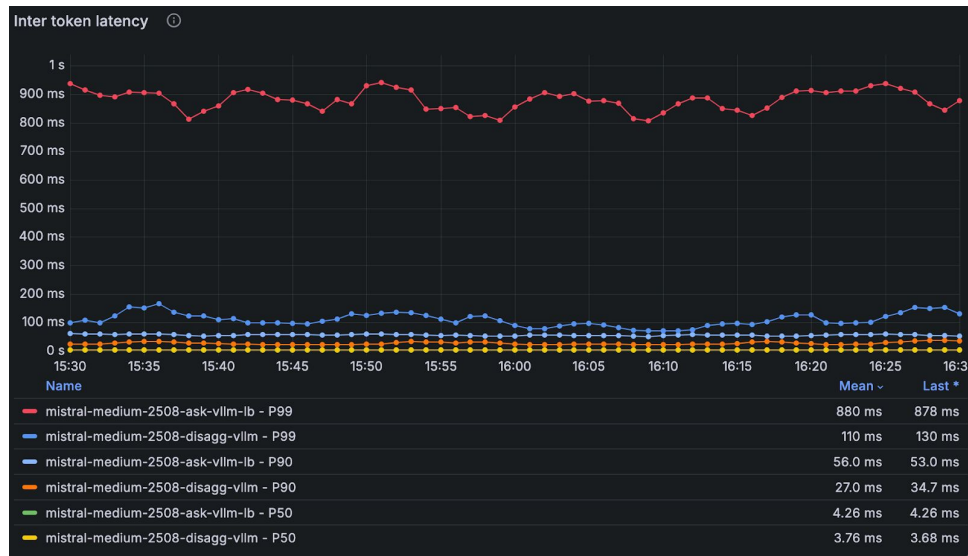
- Interference: bursty prefill workloads can delay decodes, spiking ITL
- Coupled Scaling: TTFT and ITL are on joint axis that we can't independently scale
 - Specialization: optimized environment (DeepEP HT/LL) and topologies (DPEP/TP)



From Colocated to Disaggregated



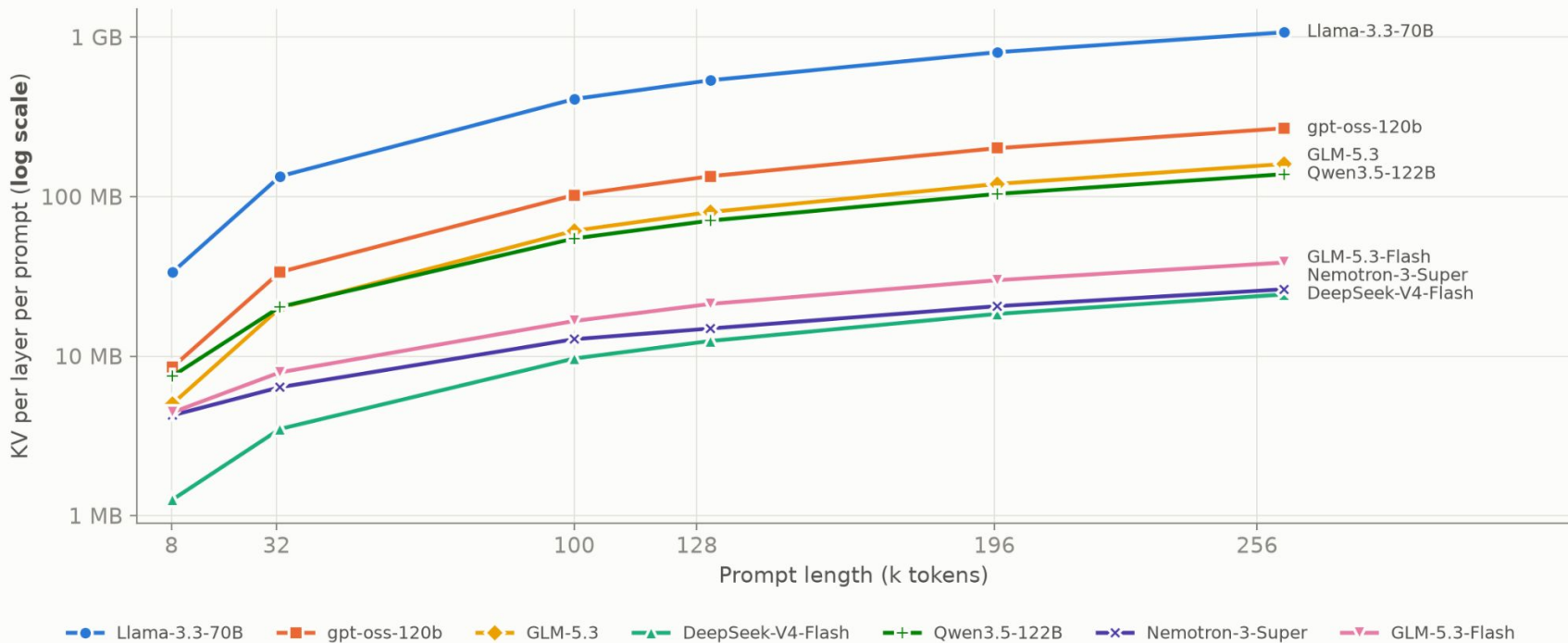
Async relative to normal engine execution.
RDMA based (does not contend for SM resources)



Longer Context, Smaller State

KV transfer per layer vs prompt length

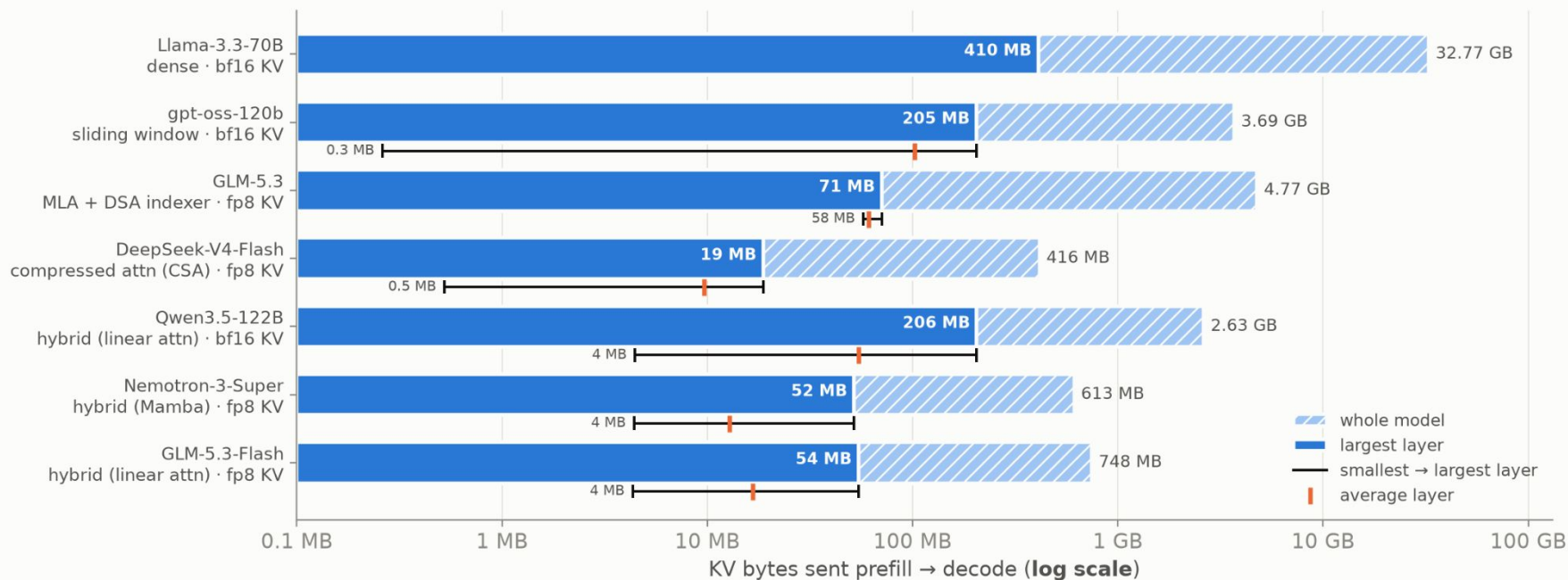
1P1D, NixlConnector, GB300 · per layer that holds KV or state



Longer Context, Smaller State

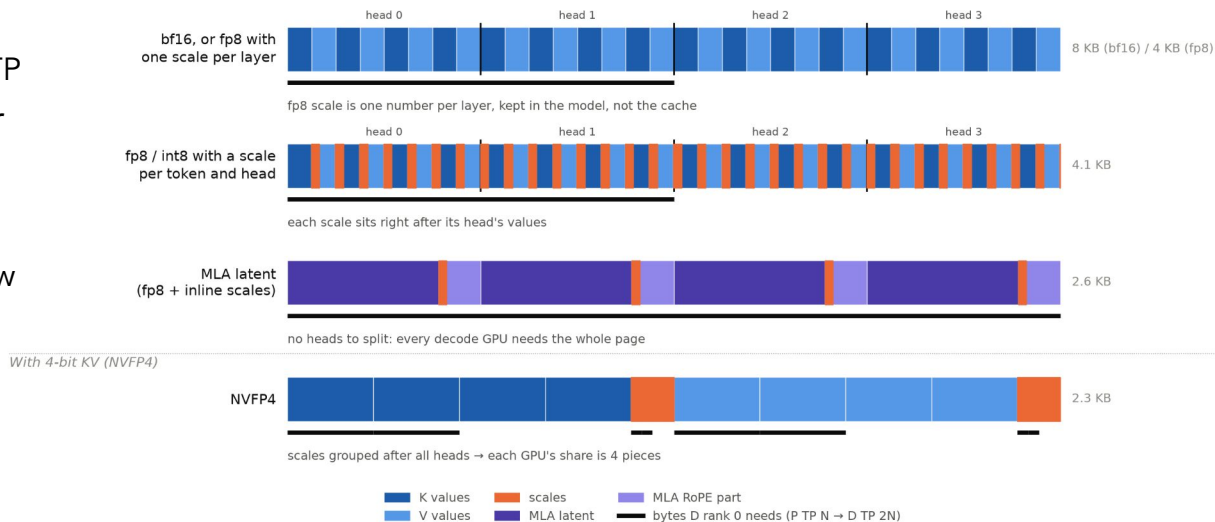
KV cache transfer size, 100k-token prompt

1P1D, NixlConnector, GB300 · solid = largest layer, whisker = smallest-largest layer



PD Challenges: More KV Page layouts

- HNC layout
 - Optimal head-splitting P TP<>D TP
- Quantization formats with finer granularity per head+token
 - Scales live within the page
 - Some scale placements don't allow heterogeneous/asymm TP
- BLHNC/*cross-layer* layout



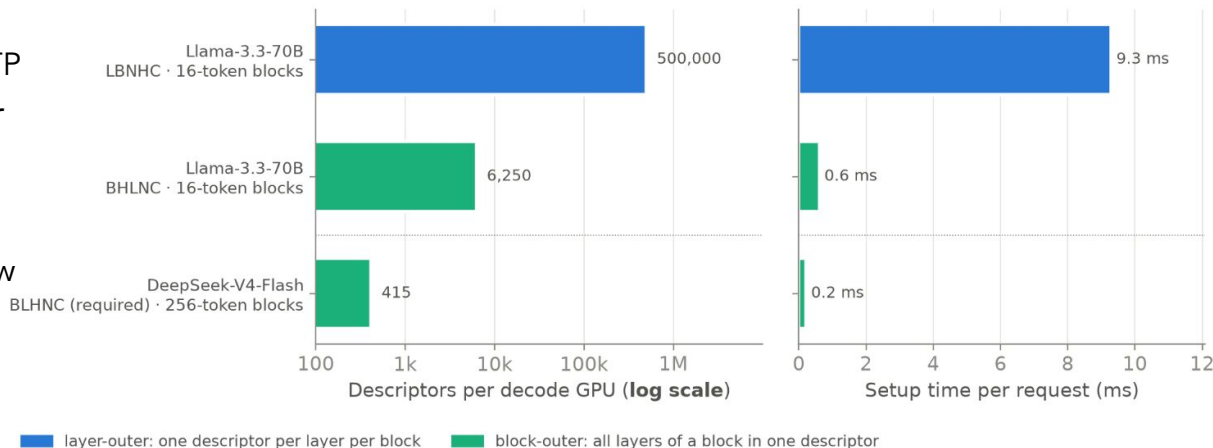
PD Challenges: More KV Page layouts

- HNC layout
 - Optimal head-splitting P TP<>D TP

- Quantization formats with finer granularity per head+token
 - Scales live within the page
 - Some scale placements don't allow heterogeneous/asymm TP

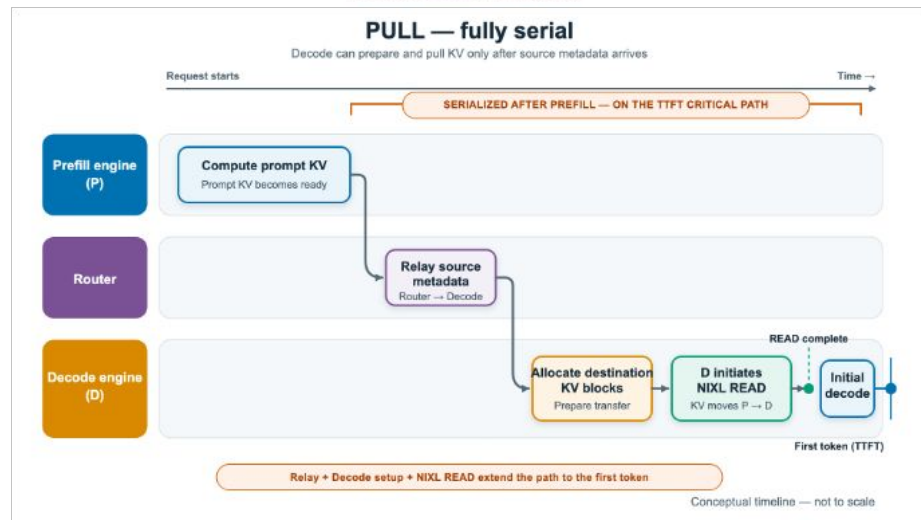
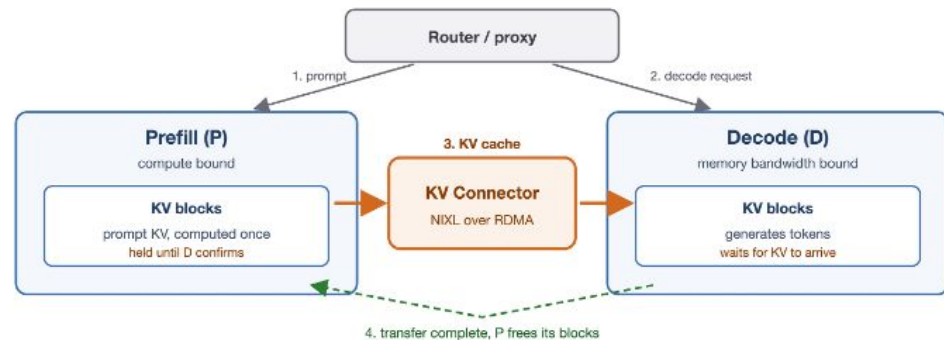
- BLHNC/*cross-layer* layout

- Descriptor as unit of xfer
- +descriptors +cpu prep work



Role of KV Connector

- P/D disaggregated deployment **transfers KV from prefill to decode over RDMA** (Nixl Connector on InfiniBand or AWS EFA)
- The default mode for Nixl Connector is **pull-based**.
- **Pull mode serializes** transfer setup after prefill - metadata relay, block allocation, and NIXL READ setup can only start once prefill finishes

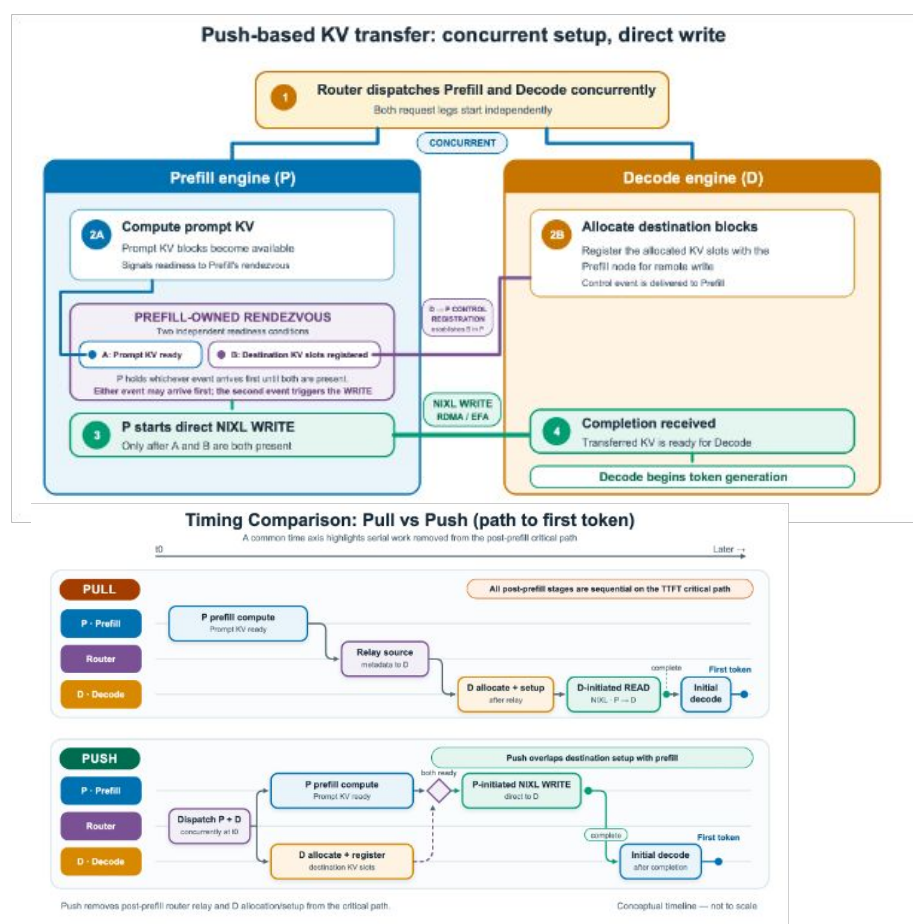


Push mode Nixl Connector

Push mode Nixl connector moves post-prefill transfer coordination and setup **off the critical path**.

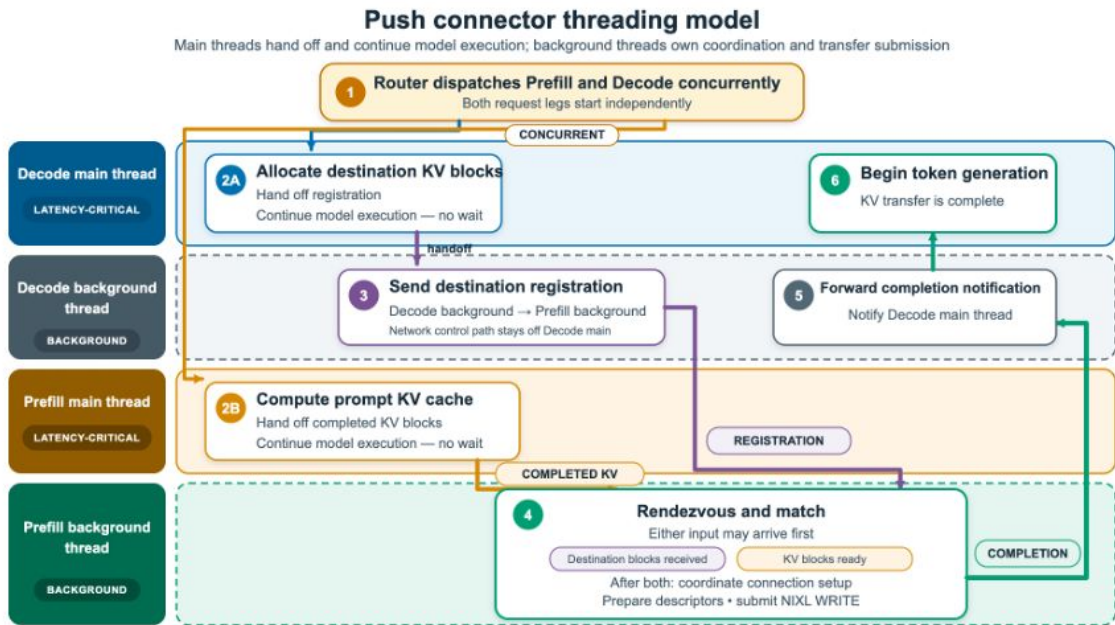
1. Overlaps producer and consumer work
2. Moves transfer setup off the decode instance

This can **improve application responsiveness by reducing TTFT and TPOT**, particularly at high request rates.

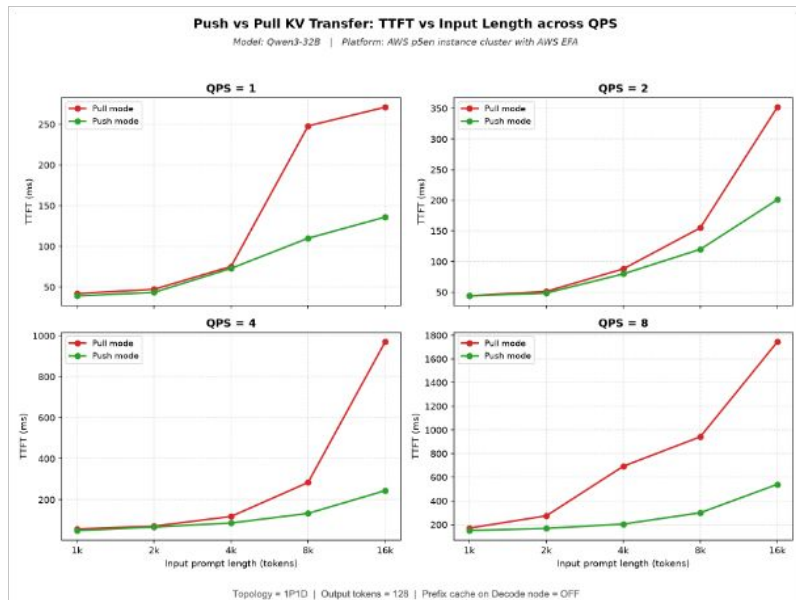


Push mode Nixl Connector Architecture

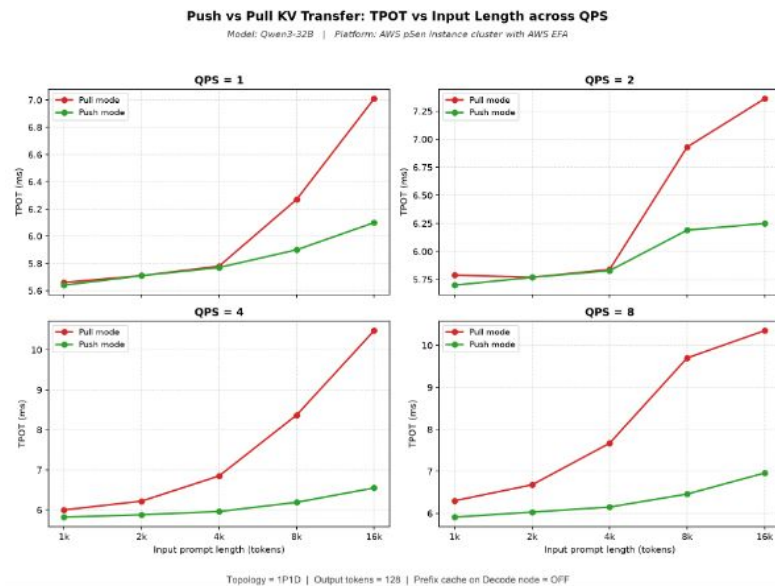
- **Registration overlaps prefill:** decode node allocates and registers destination blocks while prefill node computes the prompt KV.
- **Dedicated background thread per prefill worker:** main loop hands off the blocks and returns immediately to model execution.



3x lower TTFT and 1.5x lower TPOT



~3x lower TTFT under load - push mode overlaps the transfer setup



~1.5x lower TPOT under load - push mode keeps descriptor prep and submission off the decode worker

vLLM supports both Push and Pull mode Nixl Connectors

- **Use push mode connector when lower TTFT is the priority** and committing to a decode node early is acceptable.
- **Use pull mode connector for bidirectional KV transfer** or when decode selection must wait until after prefill for load-aware routing.

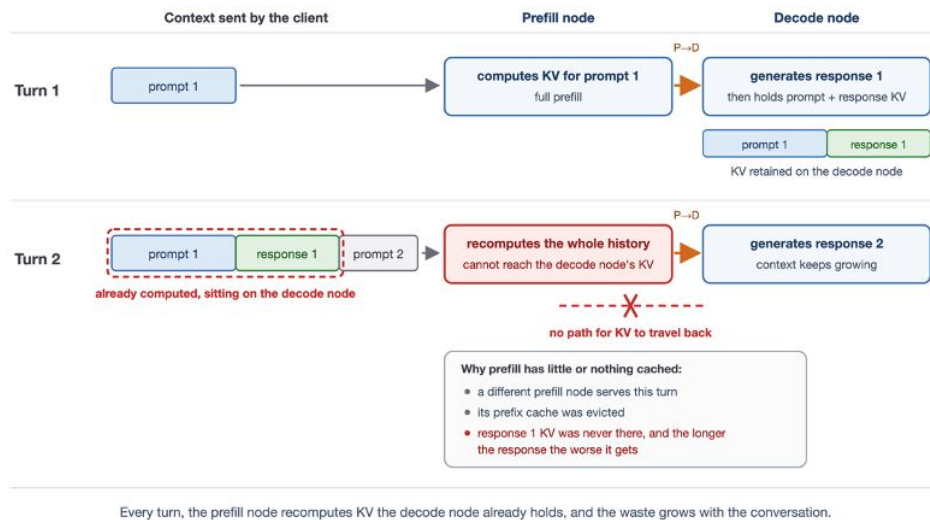
Resources:

- <https://aws.amazon.com/blogs/machine-learning/push-based-kv-transfer-to-improve-ttft-and-tpot-for-disaggregated-inference-in-vllm/>
- <https://vllm.ai/blog/2026-10-14-nixl-push-connector>
- https://docs.vllm.ai/en/latest/design/nixl_kv_push_connector/

Future work: background NIXL threading for pull mode , PR #45211

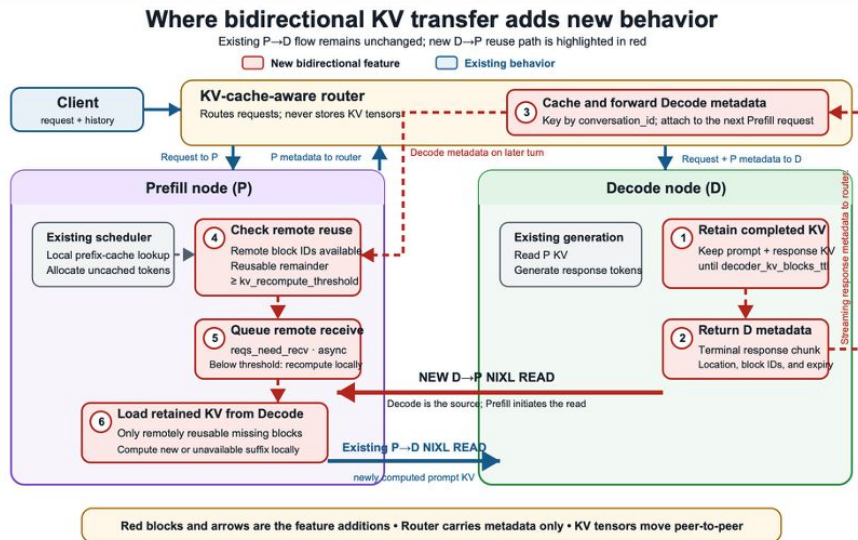
Multi-turn Agentic Workloads

- **Each turn resends the whole history:** system prompt, prior prompts, tool results, and model responses.
- **The decode node holds that KV, the prefill node can't reach it :** KV only flows $P \rightarrow D$, so prefill recomputes every response token.
- **Cost grows with the conversation.**



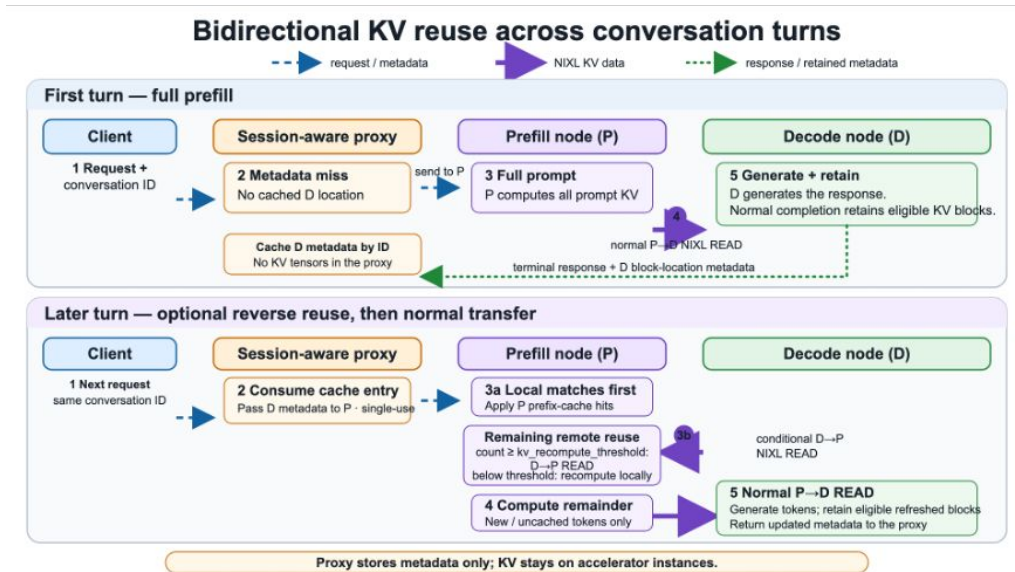
Bi-directional KV transfers to the rescue!

- **Bi-directional KV transfer:** the prefill node fetches the prior turn's KV from the decode node that still holds it.
- **Extended the NIXL KV connector so prefill nodes can consume remote KV blocks**, mirroring what decode nodes already do.
- **New D→P path:** red in the diagram



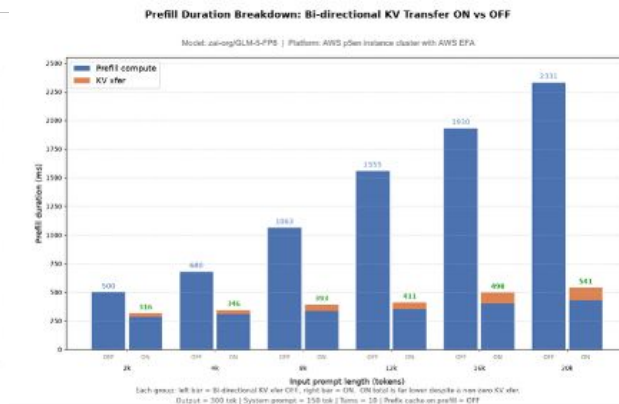
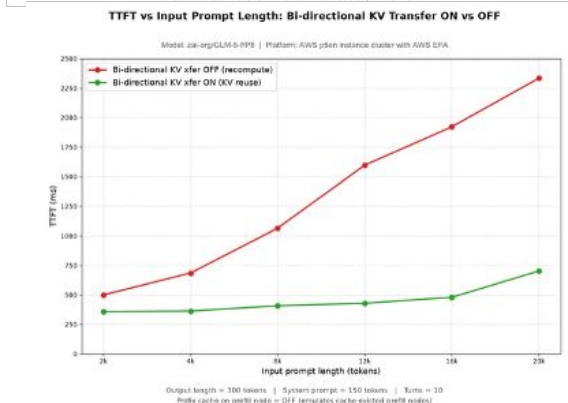
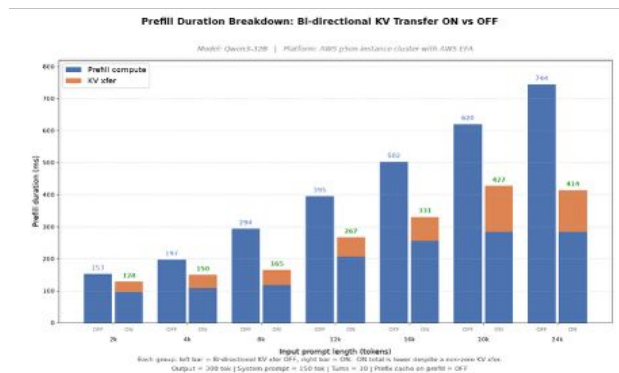
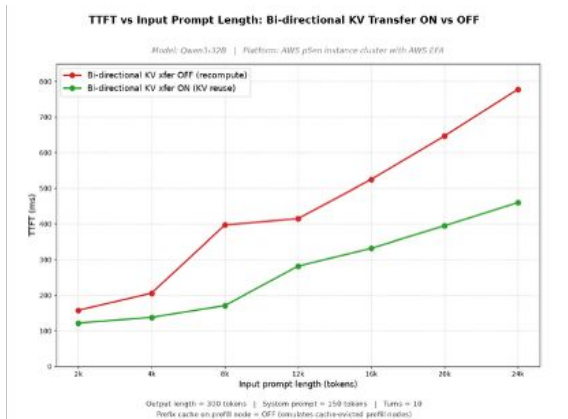
Session-aware proxy

- **Session-aware proxy**: stateful proxy mapping “conversation_id” to decode node’s “kv_transfer_params”
- **Turn 1** runs plain $P \rightarrow D$ and the router saves the metadata; **later turns** look it up so prefill reuses decode node's KV.



Multi-turn benchmarking Performance

- **Qwen3-32B**: reusing decode node's KV keeps TTFT nearly flat as the conversation grows, **up to 1.6x better** than recomputing.
- **GLM-5-FP8**: the same reuse pays off more on the larger model, **up to 3.3x better TTFT** at long prompts.



Tune bi-directional KV transfer threshold and TTL

- **Tune the KV recompute threshold** to your model's compute time, KV size, and interconnect performance; the default is 64 tokens.
- The decode node's **block-retention TTL** defaults to 480 seconds; **tune it to the expected gap between turns**.

Resources:

- <https://aws.amazon.com/blogs/machine-learning/improve-ttft-for-multi-turn-agentic-workloads-with-bi-directional-kv-transfer-in-vllm/>
- <https://vllm.ai/blog/2026-10-12-bidirectional-kv-transfer-multiturn-agentic-workload>
- https://docs.vllm.ai/en/stable/features/nixl_connector_usage/#bidirectional-kv-transfer-multi-turn

Future work: *extend the heartbeat-renewed lease to the decode side*

Hybrid Models: Attention KV and Recurrent State

- Hybrid models combine attention KV with recurrent state.
 - Attention KV grows with context.
 - Each recurrent snapshot has a fixed size.
- Need to manage and transfer both cache types with different layouts and growth patterns.

Attention KV

K0	K1	K2	K3	K4
V0	V1	V2	V3	V4

Each K/V:

[tokens, KV heads, head dim]

Recurrent state (e.g., Mamba, GDN, KDA)

Convolution state

[conv dim, state length]

SSM state

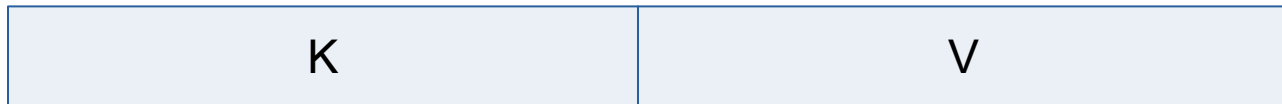
[SSM heads, head dim, state size]

*For a fixed model and TP. Shapes: one layer, one request, local TP rank. Conv in DS layout.

Hybrid Cache Layouts in vLLM

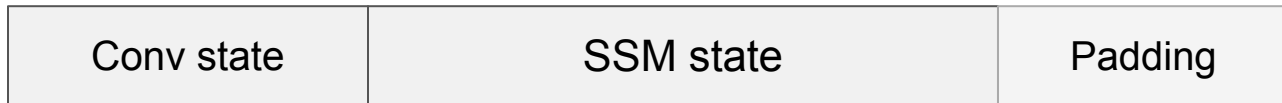
- The Hybrid Memory Allocator (HMA) pools memory across cache types.
- Shared pool, equal-sized allocation pages.
- HMA adjusts attention block sizes to fit recurrent state and reduce padding.

Attention page



Block size chosen to accommodate recurrent state

Mamba page

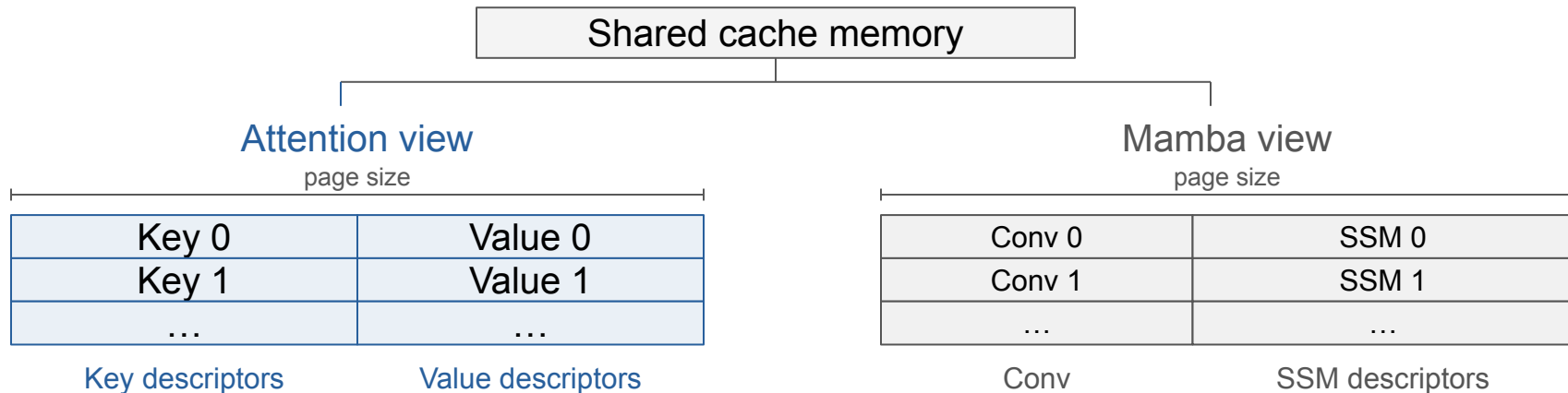


Complete state snapshot

Unused bytes

Dual Descriptor Views

- A view maps cache components in memory.
- Different layouts require different maps.
- Both views reference the same memory without copying state.

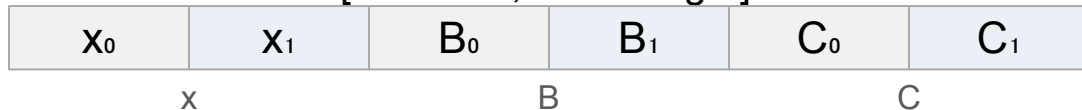


Heterogeneous TP Transfer

- Conv state shards channels within each x/B/C projection, with Mamba as an example.
- SSM state shards along SSM heads, preserving each head's state matrix.
- DS layout makes each convolution shard contiguous for direct transfer.

P TP1: rank 0

Convolution state: [conv dim, state length]



SSM state



D TP2: rank 0



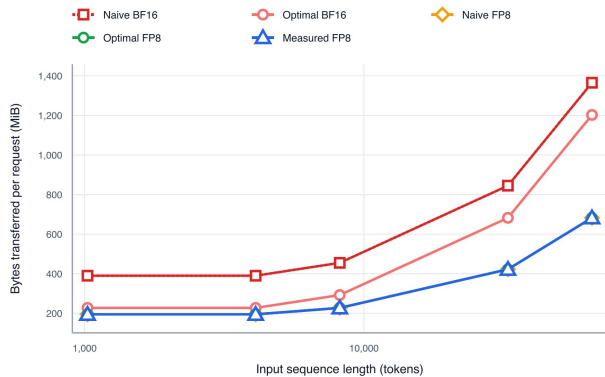
D TP2: rank 1



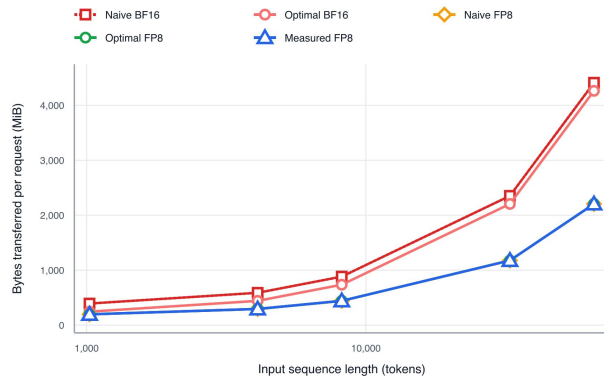
Hybrid State Transfer

- Direct mapping between prefill and decode instances, no extra staging buffers or reshuffling.
- Skip potential HMA padding.
- Transfer volume equals the exact useful recurrent state bytes.

Nemotron-3-Super-120B-A12B-FP8



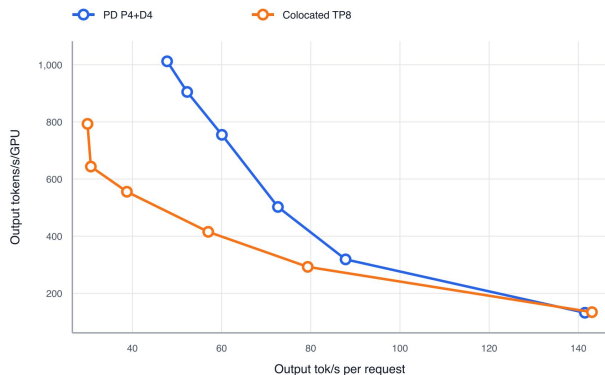
Qwen3.8-27B-FP8



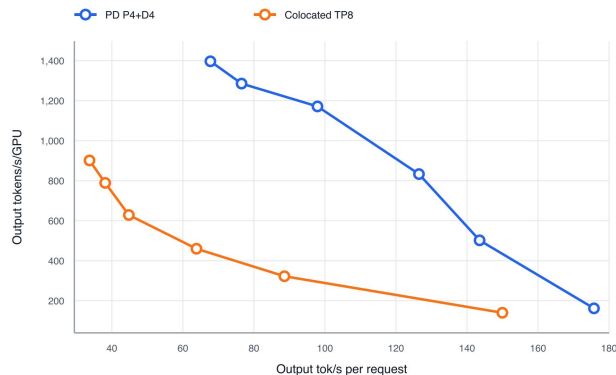
Disaggregated vs. Co-located Serving

- H200, 8 GPUs: Prefill TP4 + Decode TP4 vs. co-located TP8.
- At high concurrency, PD serves more output tokens per GPU while maintaining faster generation for each request.

Nemotron-3-Super-120B-A12B-FP8

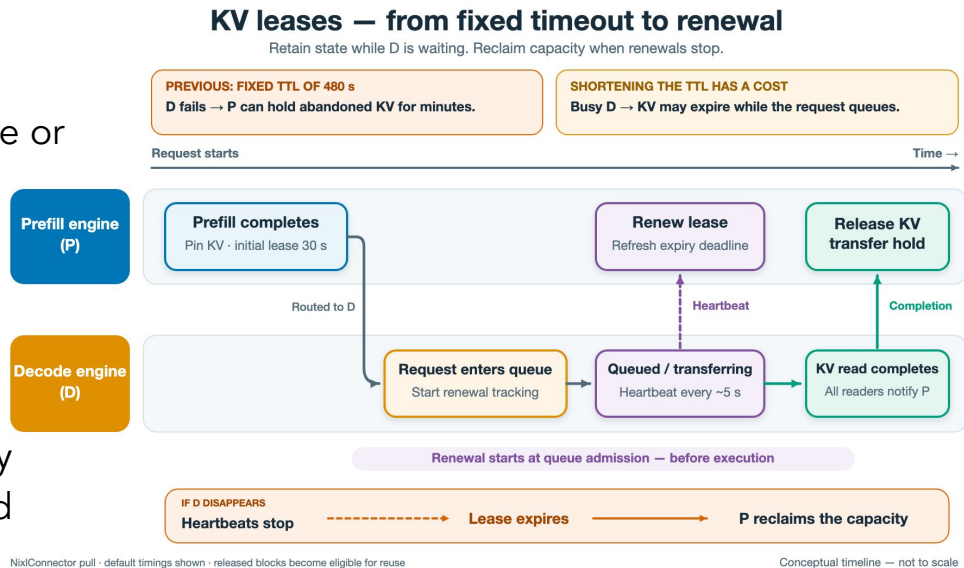


Qwen3.8-27B-FP8



PD Operational Challenges: Blocks Lifetime on P

- Fixed timeout (~480s default)
 - Traffic surges and scale-down may expire or leave blocks on P pinned
- KV Lease mechanism
 - Lease is initiated as request enters the queue on D
 - Refreshed periodically w/ heartbeats
 - D dies => blocks can be dropped swiftly
 - D overloaded=>P blocks are maintained



Recap

- Challenges in developing for PD: architectures & kv layouts
- PushMode and the importance of off hot-path processing
- Bidirectional KV Transfer for agentic traces
- Disaggregating Hybrid models in vLLM
- Managing lifetime of blocks on remote

Q&A

