



Agentic Inference Performance on Vera Rubin

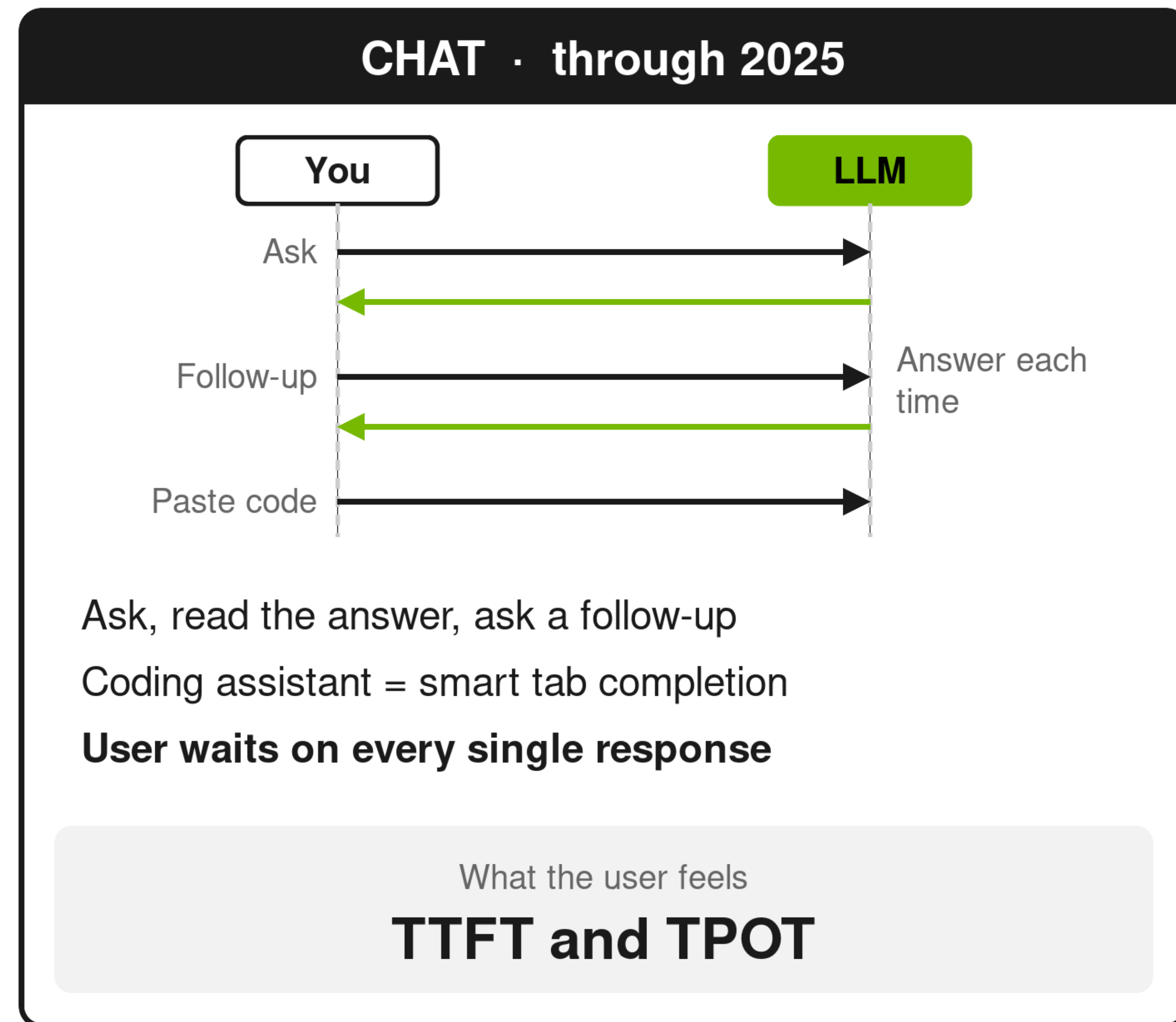
Alex Settle, NVIDIA

PyTorch Conference | 2026

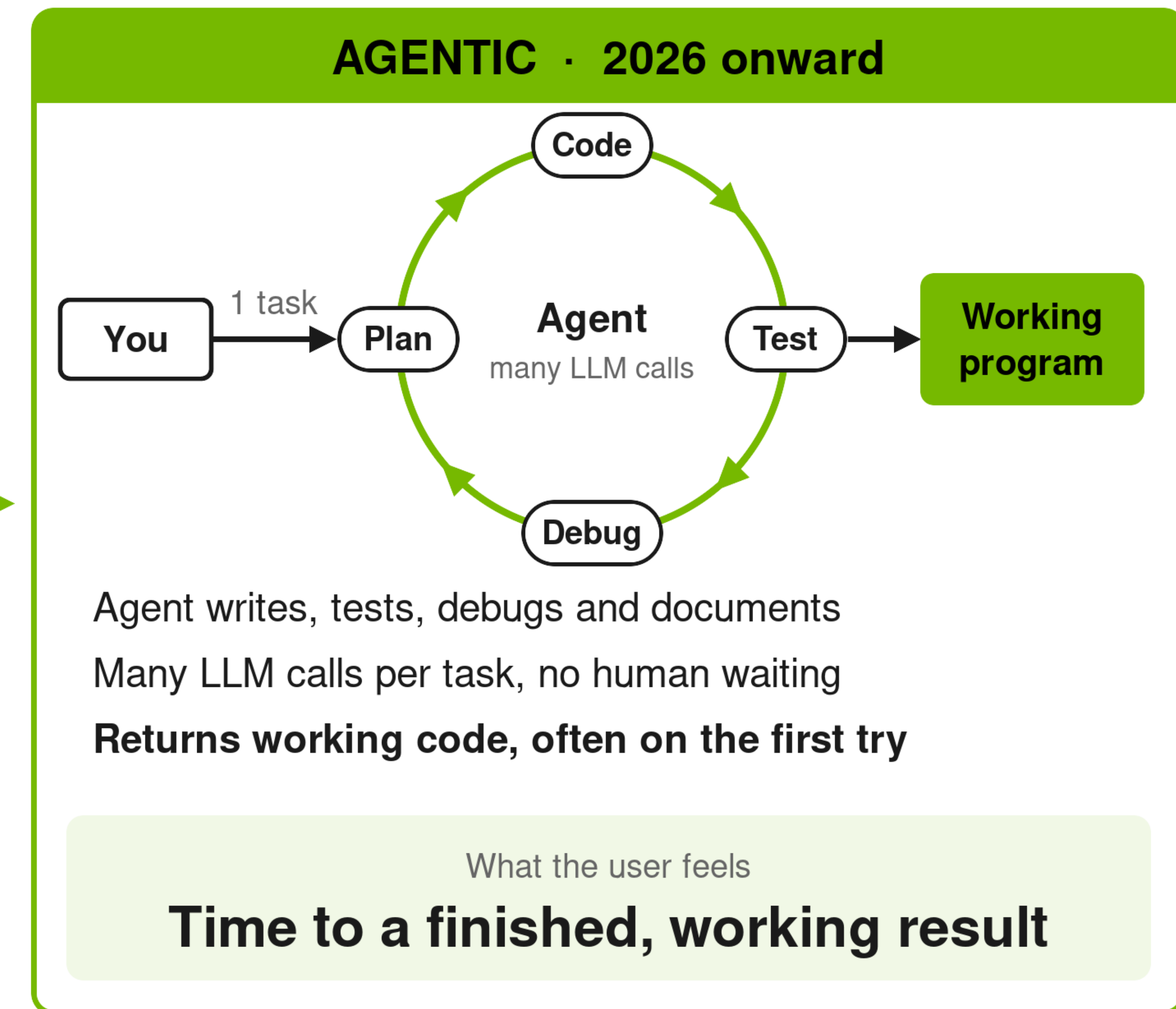


From Chat to Agents

Agents changed what users wait for, and what we should measure



2026

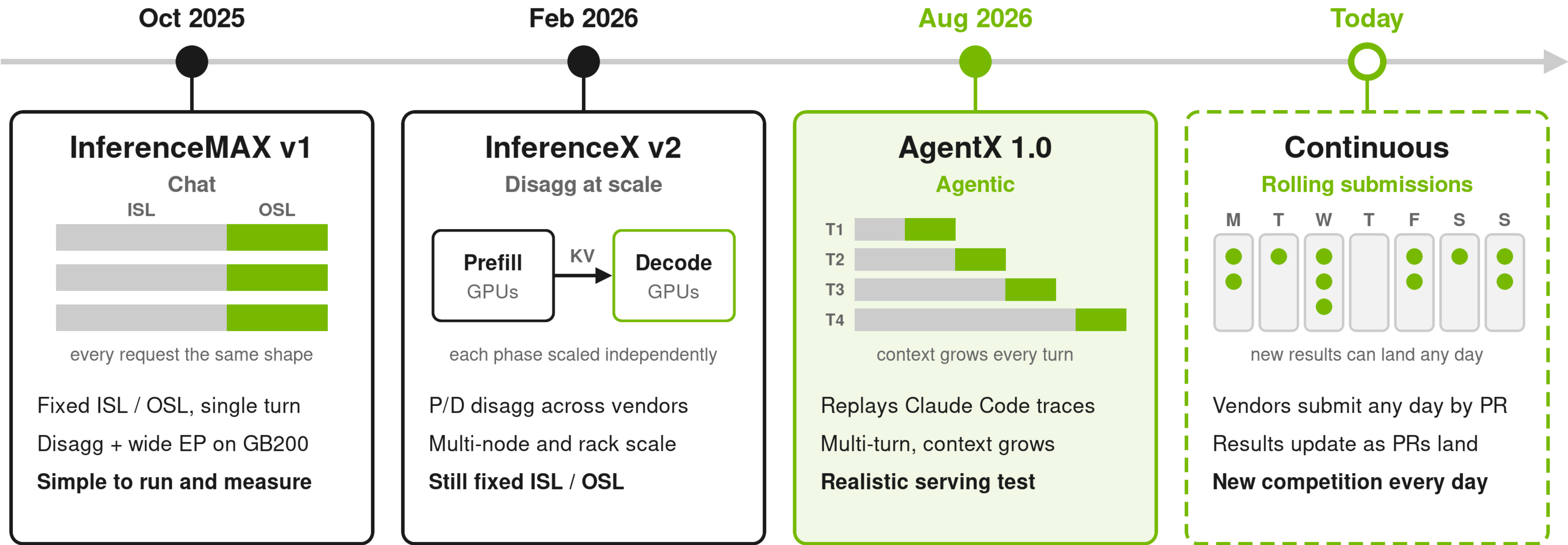


40x larger context length: 8K fixed ISL → 300K average session peak (up to 1M)

Source: analysis of semianalysisai/cc-traces-weka-062126 (public on Hugging Face), 98,827 model calls; 8K ISL = InferenceMAX fixed 8K/1K case

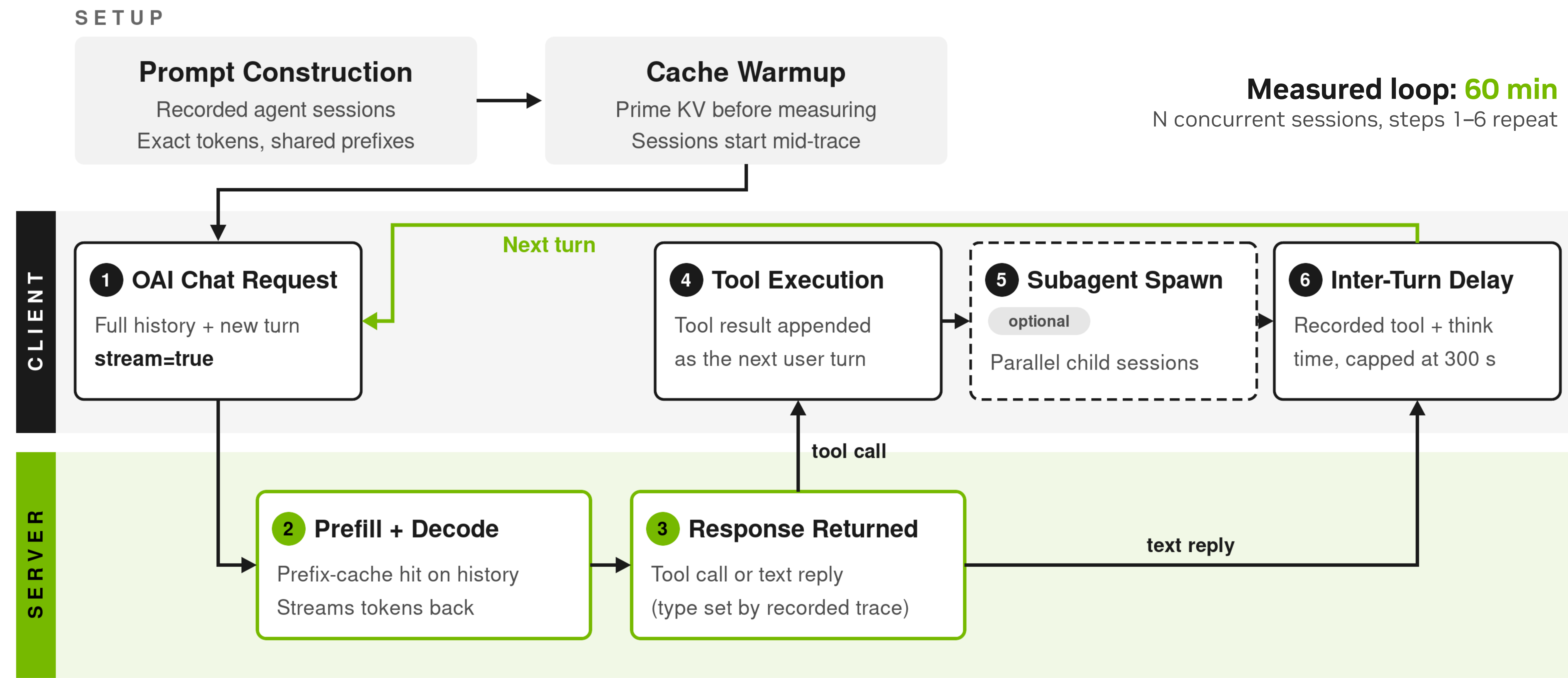
The Evolution of InferenceX

SemiAnalysis's open benchmark, from InferenceMAX to AgentX, now runs continuously



How AgentX Replays an Agent Session

Each session loops through request, response, tools and idle time for 60 minutes



Agentic Workloads Stress the Entire System

Realistic workloads end hyper-tuning and push load beyond the GPU: memory, CPU and interconnect

The tuning target changed

Fixed ISL / OSL

A few predictable request shapes
Kernels and configs hyper-tuned
Risk: over-fit to the benchmark



AgentX

Shapes change every turn and session
No single case to over-engineer
Optimize for the real use case

272K

p90 input tokens per
request on AgentX
DeepSeek V4 Pro, SemiAnalysis

Where agents put the load

GPU memory

HBM

Weights: trillions of parameters
Hot KV cache for active turns

CPU memory

host

KV for idle sessions between turns
Reload on next turn, skip re-prefill

NVLink

GPU ↔ GPU

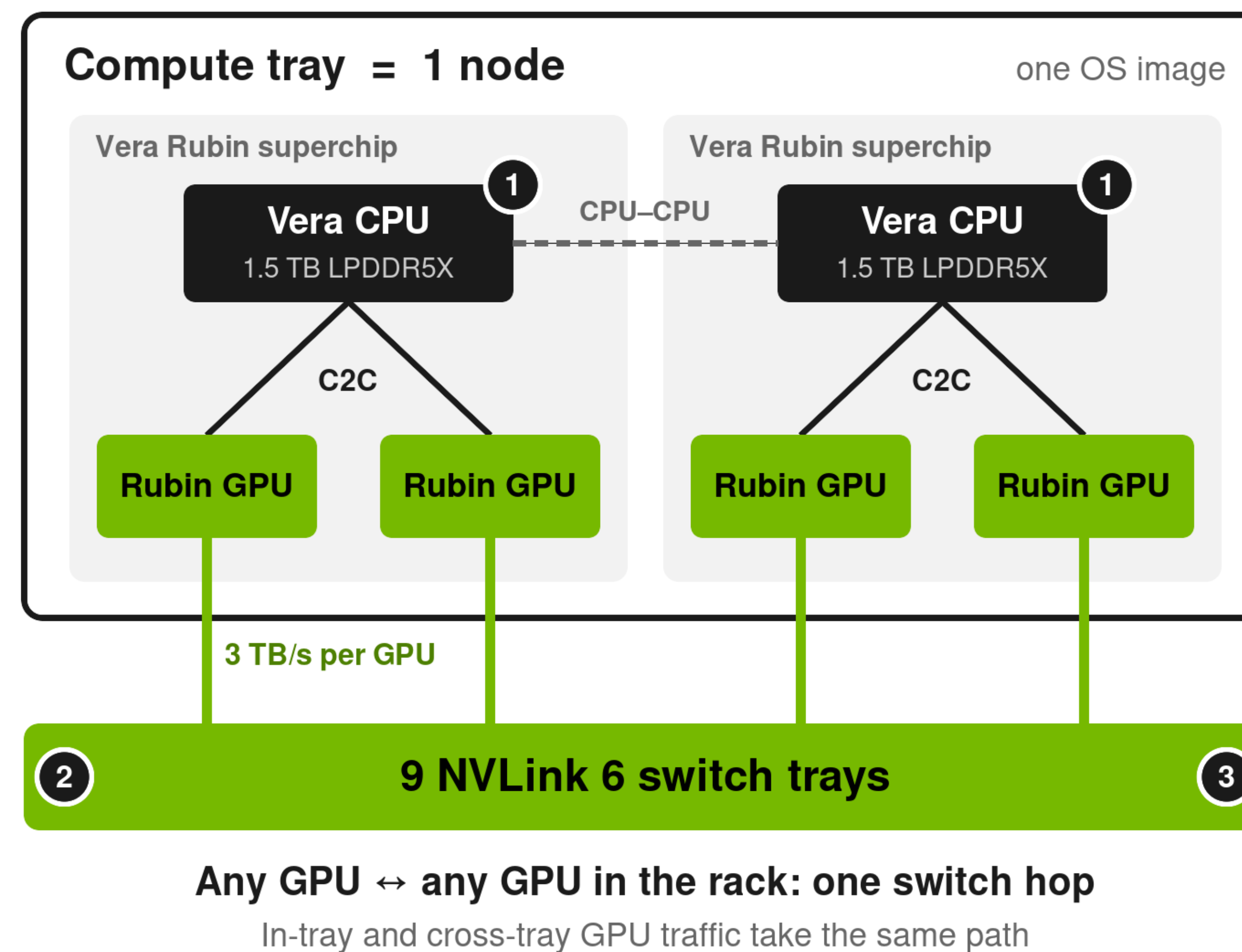
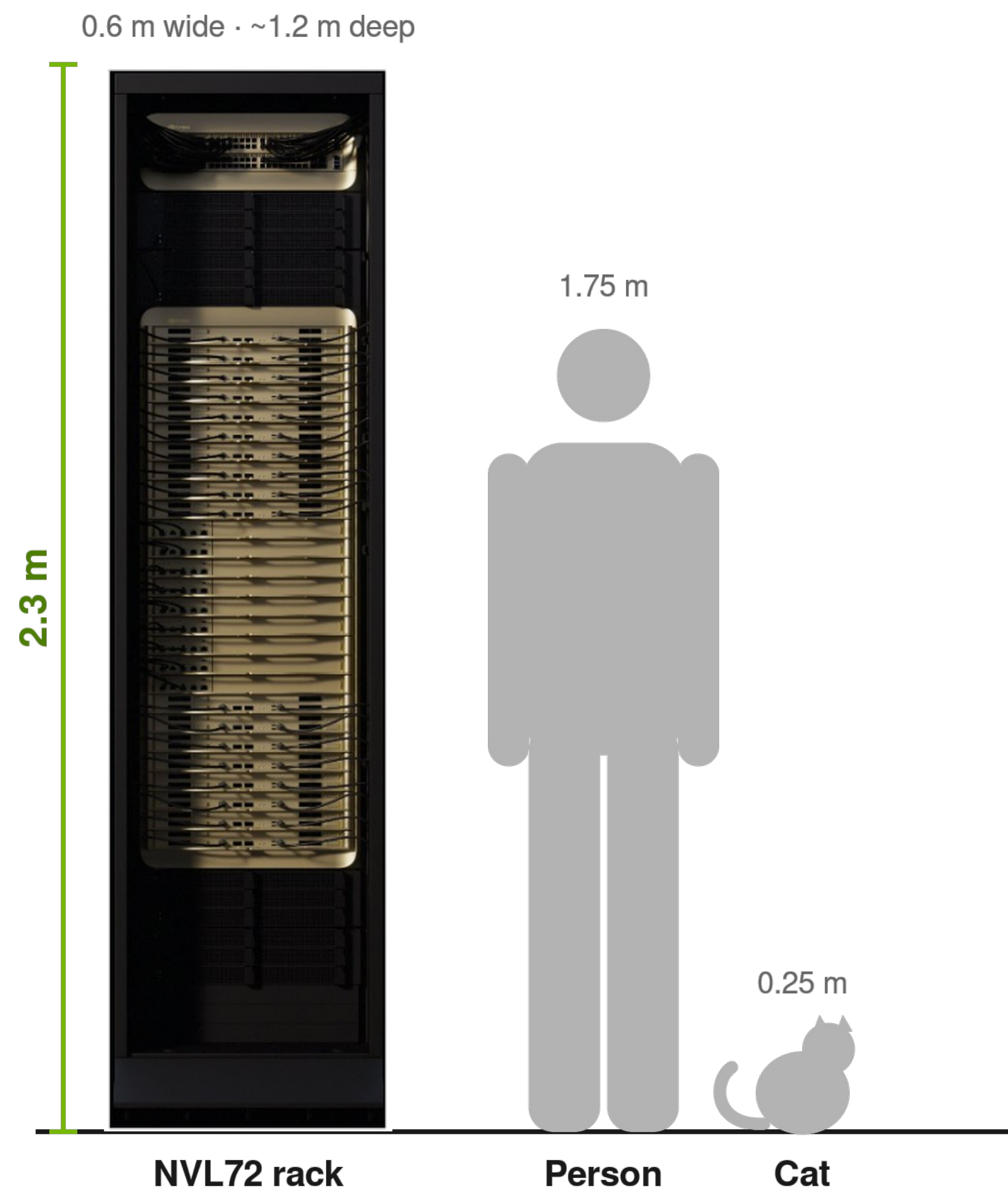
KV moves prefill → decode (disagg)
Expert all-to-all (wide EP)

Agents stress the entire system, not just the GPU

Vera Rubin: HBM4 · Vera CPU memory over NVLink-C2C · NVLink 6

Inside Vera Rubin NVL72

Why agentic workloads run well: CPU memory for KV offload, NVLink 6 for wide EP and large TP



Why AgentX runs well here

1 Idle sessions → CPU memory

KV cache offloads to Vera LPDDR5X
Coherent NVLink-C2C, no re-prefill

54 TB CPU memory per rack

2 Wide EP, large TP → NVLink 6

All 72 GPUs, one switch hop
3 TB/s per GPU, 1.7x GB300

216 TB/s per rack

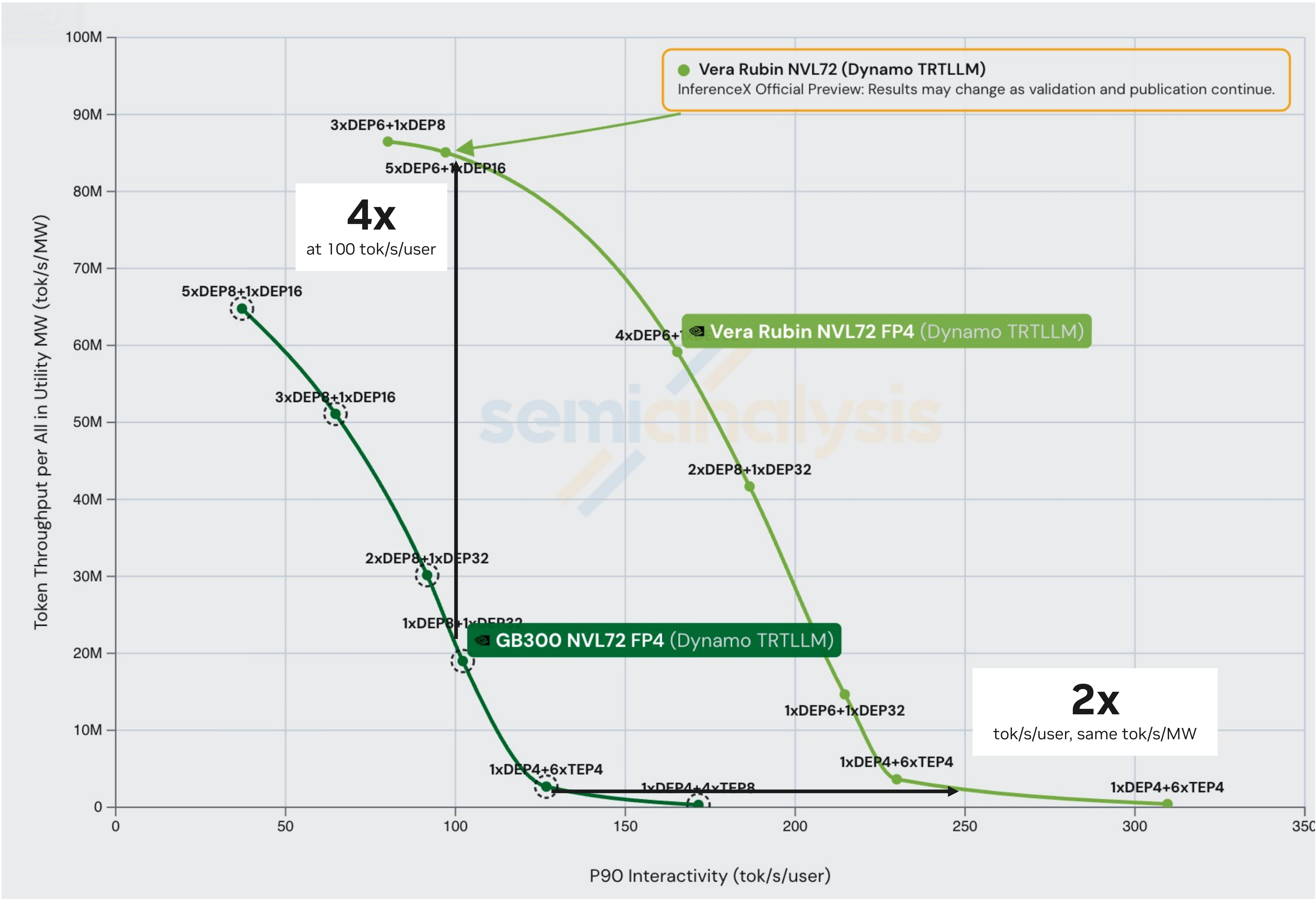
3 Low latency → more GPUs

Fast all-to-all lets big models
scale to more GPUs per instance

Trillion-parameter MoE, long context

Results: DeepSeek V4 Pro on TensorRT-LLM

Vera Rubin vs GB300 NVL72, both FP4 with Dynamo disaggregated serving



Source: SemiAnalysis InferenceX AgentX, updated Oct 14, 2026. Vera Rubin results are an InferenceX official preview. Multiples read from the Pareto frontiers, tok/s per MW at P90 interactivity

Built on Open Source, End to End

Every layer, from models to orchestration, is open and reproducible

Benchmark	InferenceX	AgentX traces	AIPerf	agentic traces and client
Launch and reproduce	srt-slurm	srt-slurm-recipes		declarative YAML recipes
Orchestration	NVIDIA Dynamo			NVIDIA submissions; other vendors bring their own
Inference engines	● vLLM	● SGLang	TensorRT-LLM	PyTorch-based serving engines
KV cache	● Mooncake (store)	NIXL (move)		KV across GPUs and hosts
Open-weight models	DeepSeek V4 Pro	DeepSeek V4.1 Flash	Kimi K3	
	MiniMax M3	GLM 5.2	Qwen3.5	Qwen3.8
Framework	● PyTorch			

● PyTorch Foundation or PyTorch ecosystem project

Every layer is open source

Code, recipes, traces and results are all public

Dynamo: NVIDIA's open layer

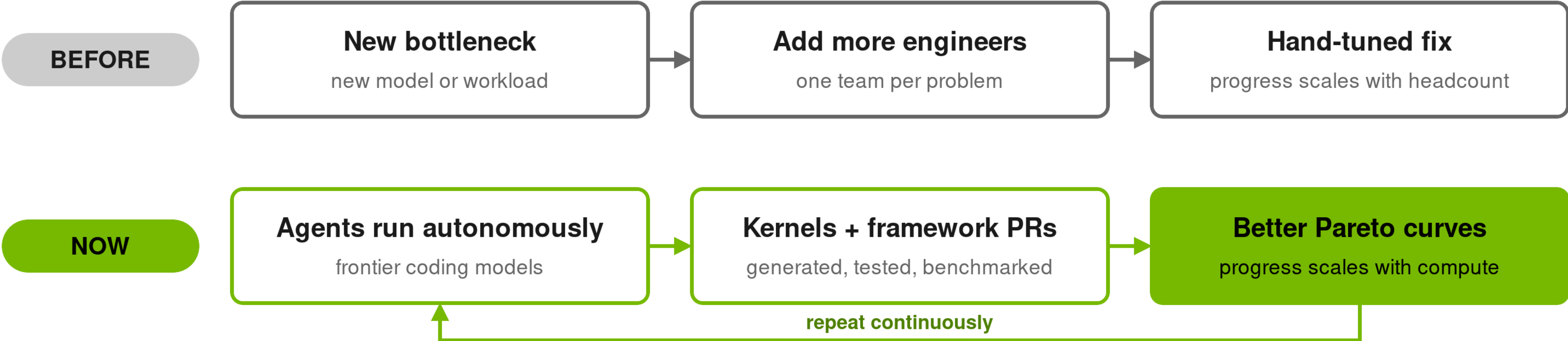
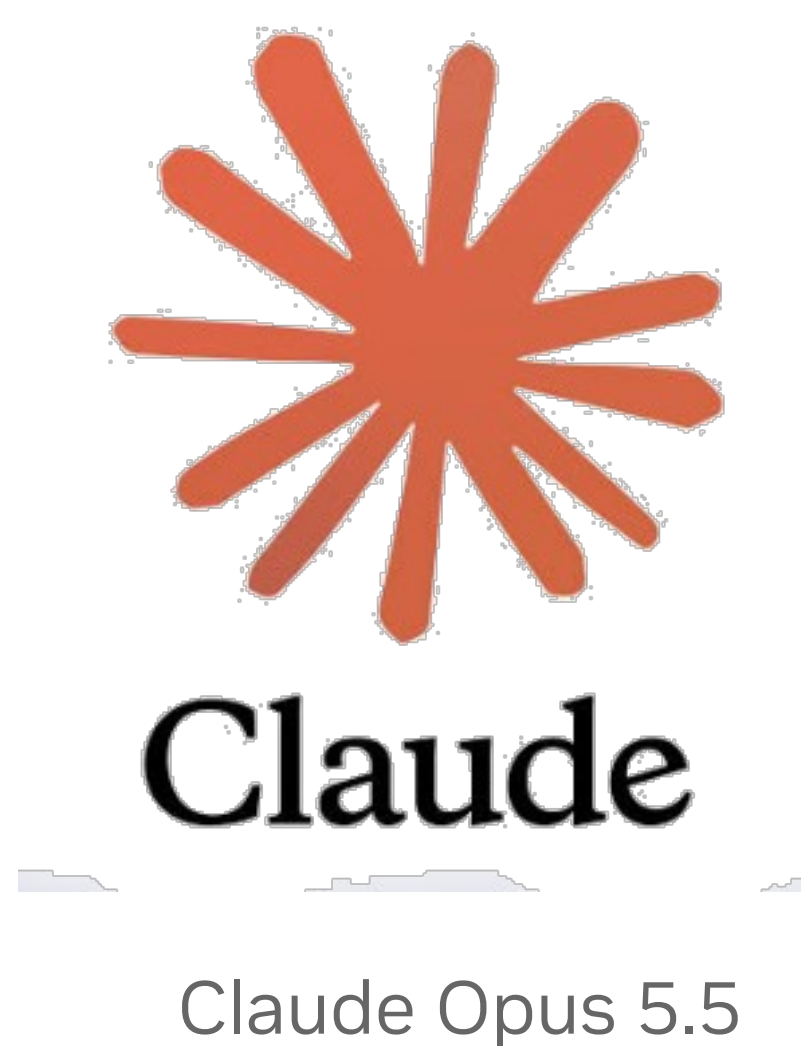
Disagg, KV routing, rack-scale scheduling over vLLM, SGLang and TRT-LLM

Reproducible by anyone

A public srt-slurm recipe per result
Goal: rerun any NVIDIA result in 1 command

What's Next: AI Optimizes the Inference Stack

Frontier models now write the kernels and framework changes that move the Pareto curve



GPT-6 Astra

New bottleneck: code review

Generated code is growing faster than review

Where should all this code live?

Should humans review every change?

Competition gets more dynamic

Everyone has access to the same models

Breakthroughs can land at any time

and are replicated quickly by everyone

END