PROMPTS:

PeRformance Optimization via Multi-Agent Planning for LLM Training and Serving

Yuran Ding[1,2] Ruobing Han[3] Xiaofan Zhang[3] Xinwei Chen[2]

[1] University of Maryland, College Park [2] Google [3] Google DeepMind

LLM Workloads and Sharding

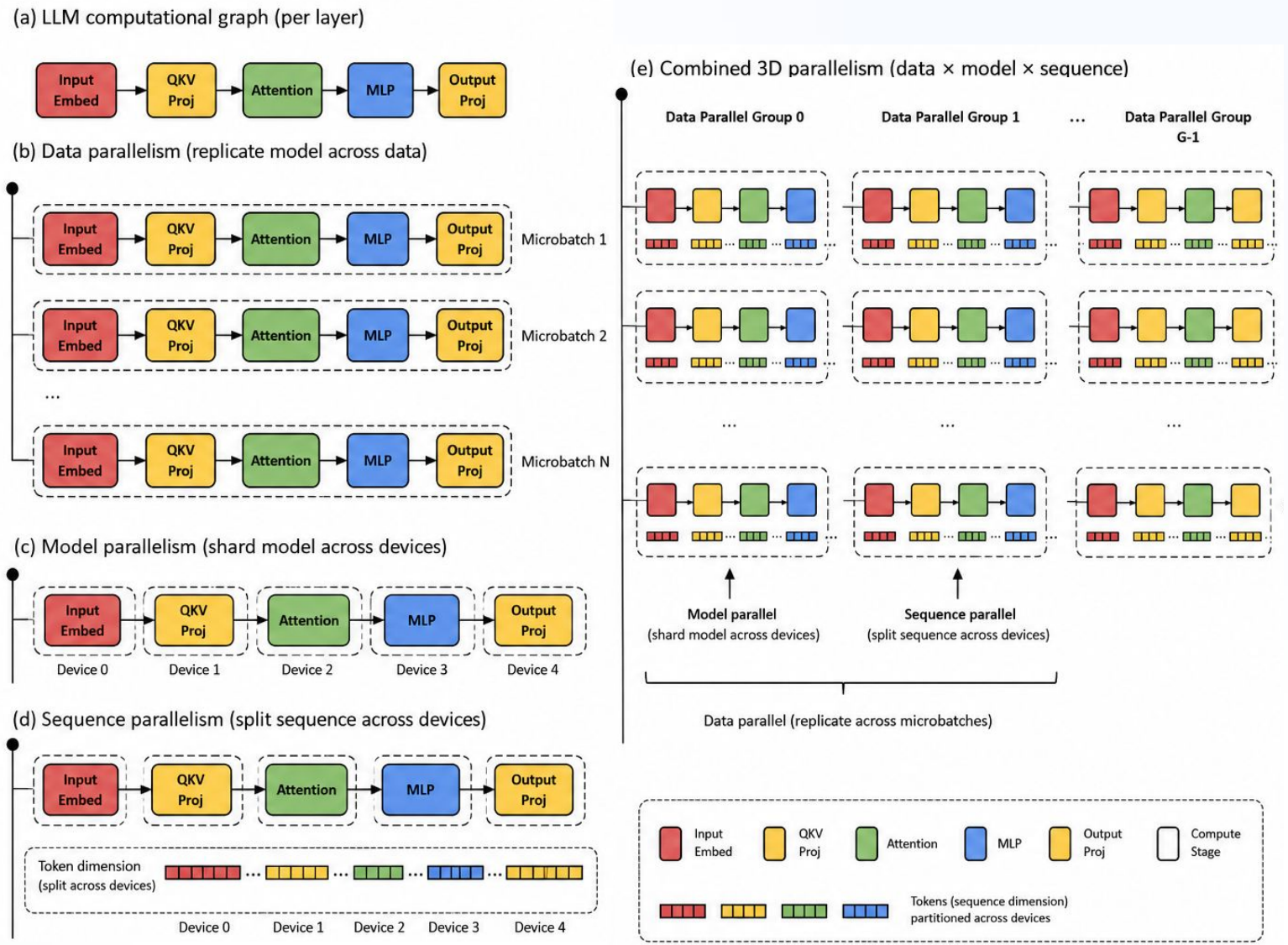
What is Sharding?

Sharding is how we distribute massive LLMs across physical hardware to overcome memory and compute limits.

Data Parallelism: Replicating the model to process different slices of the data batch simultaneously.

Model Parallelism: Slicing the actual neural network weights across multiple chips to prevent out-of-memory errors.

Sequence Parallelism: Dividing the input context window across devices to manage memory pressure.



The High-Dimensional Maze of Sharding

Why is it difficult?

Scaling LLMs across thousands of chips is notoriously hard due to combinatorial complexity and non-linear trade-offs.

Hardware Linkage: Strategies for TPU v5p do not generalize to v6e/tpu7x.

Opaque Bottlenecks: Diagnostic traces require deep expertise to interpret (HBM vs. Compute).

Traditional Black-box Search

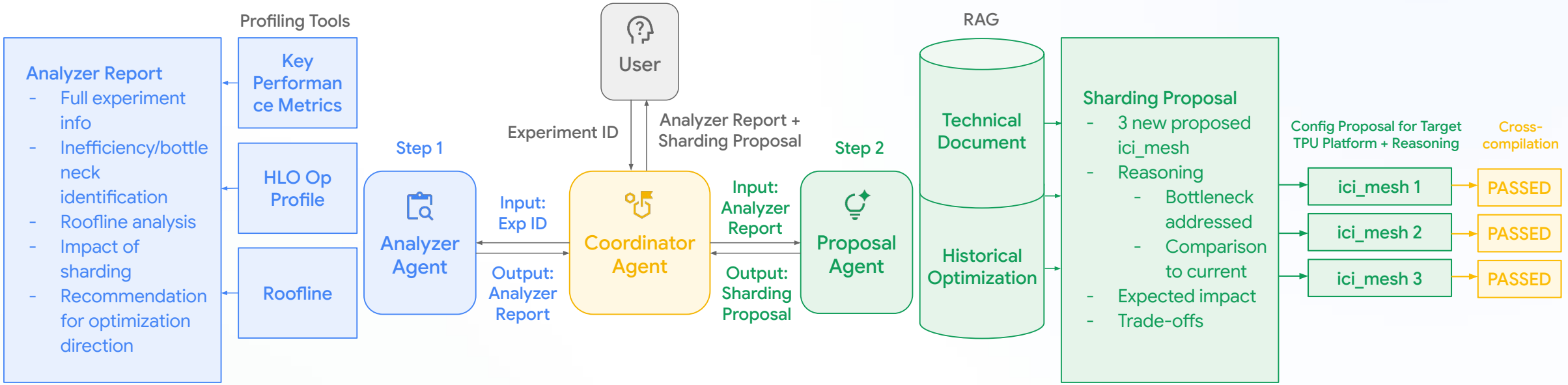
- **Knowledge-Blind:** Operates without deep system understanding.
- **Resource Intensive:** Hundreds of expensive trials.
- **Brittle:** Static maps fail on workload changes.

*We need a methodological shift to **Subspace Identification**.*

The Primary Contribution

A methodological shift that reframes performance optimization from unguided search to expert-reasoning-guided subspace identification.

The PROMPTS Framework



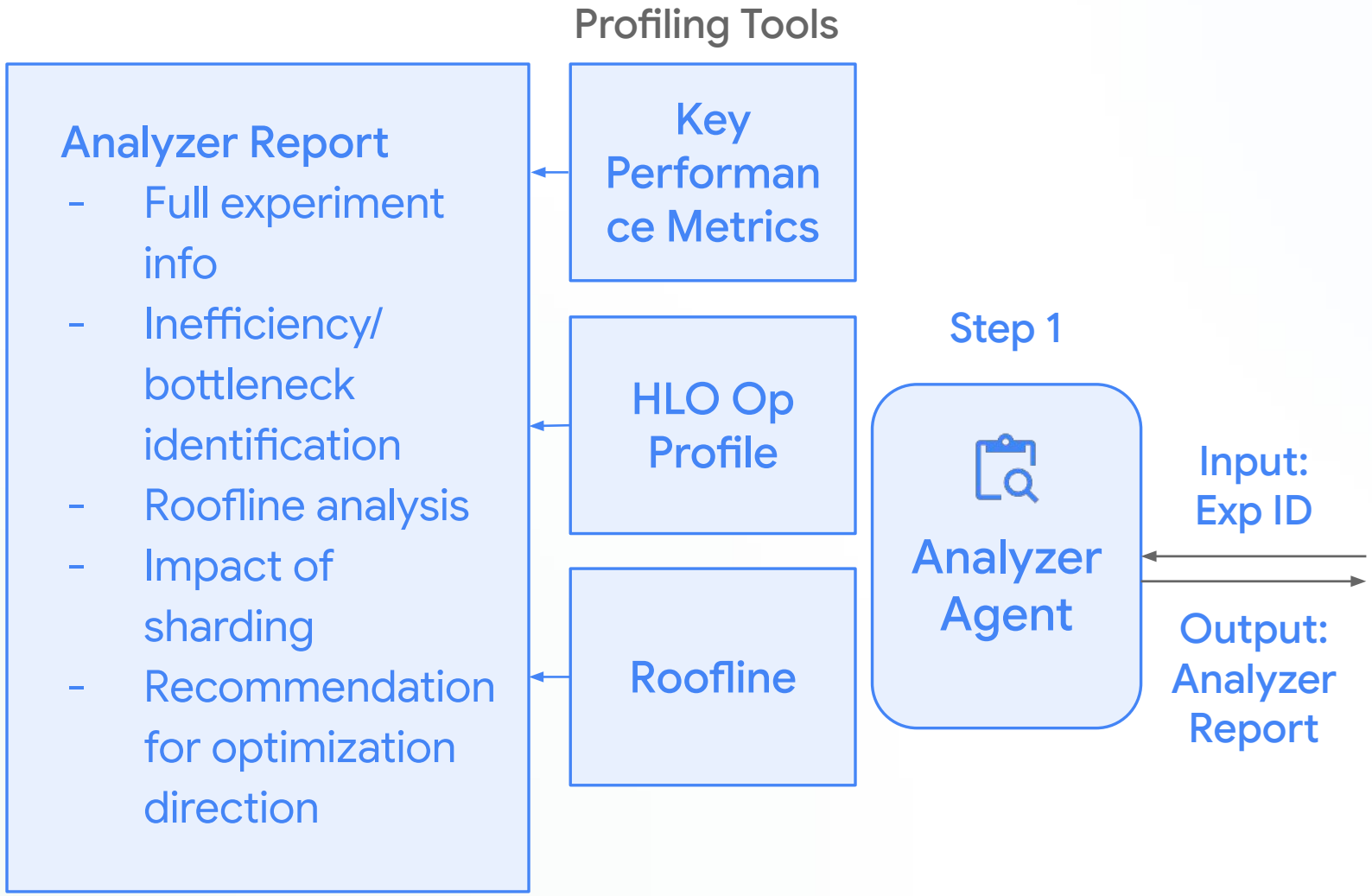
PROMPTS is a novel multi-agent framework that complements traditional search methods with expert-informed reasoning. It automates large-scale AI system optimization by embedding expert diagnostic reasoning, delivering performance improvements of up to 434%. The framework provides valid reasoning and accurate recommendations by considering LLM workload characteristics and backend hardware features. Sharding validity is enforced by the TPU compiler, serving as the authoritative correctness check.

Coordinator Agent: Orchestrates the workflow, manages agent interactions, and processes user requests.

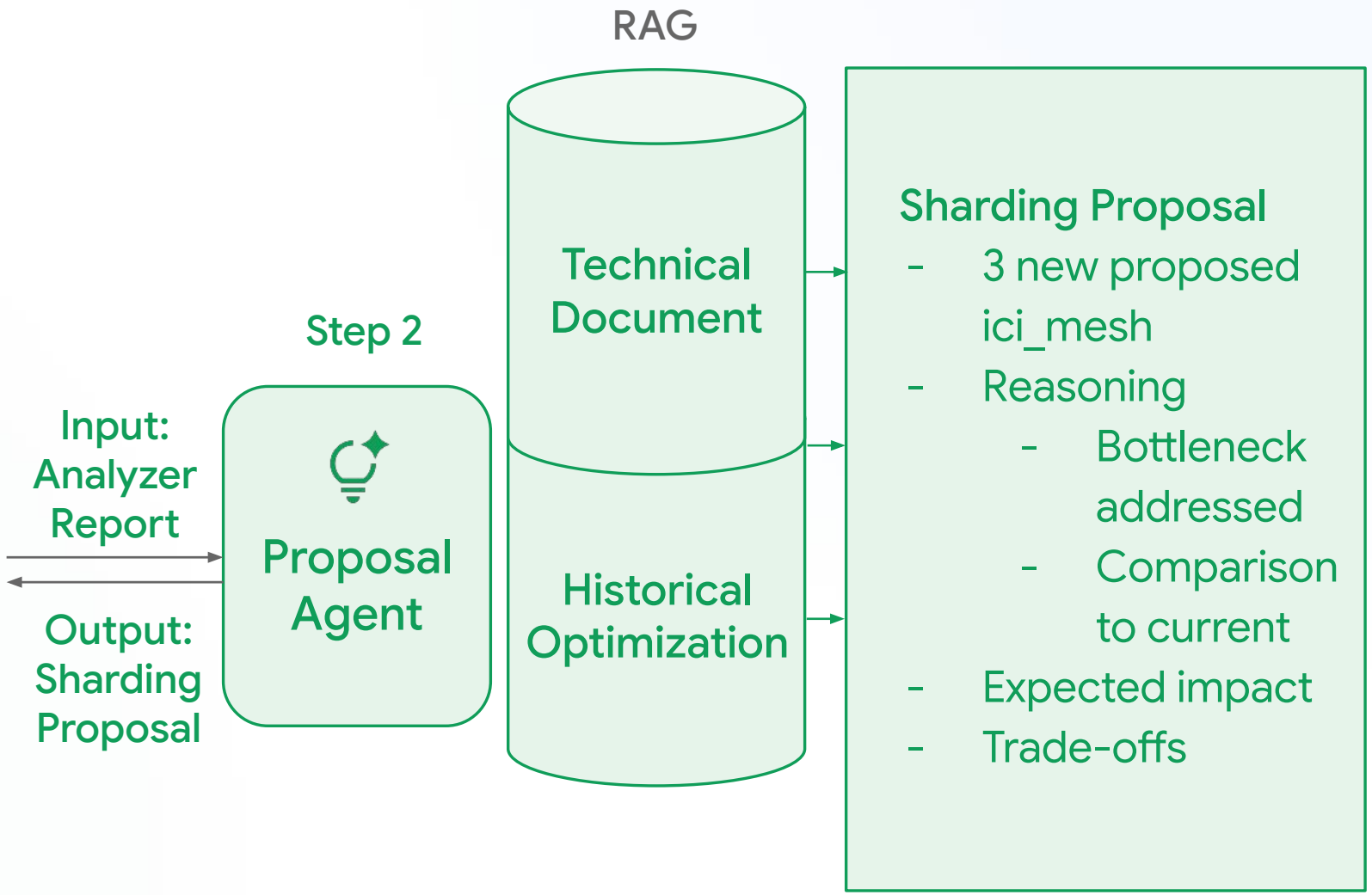
Analyzer Agent: Diagnoses performance bottlenecks by analyzing data from profiling tools and experiment logs.

Proposal Agent: Generates optimized sharding configurations with detailed reasoning by querying a knowledge base.

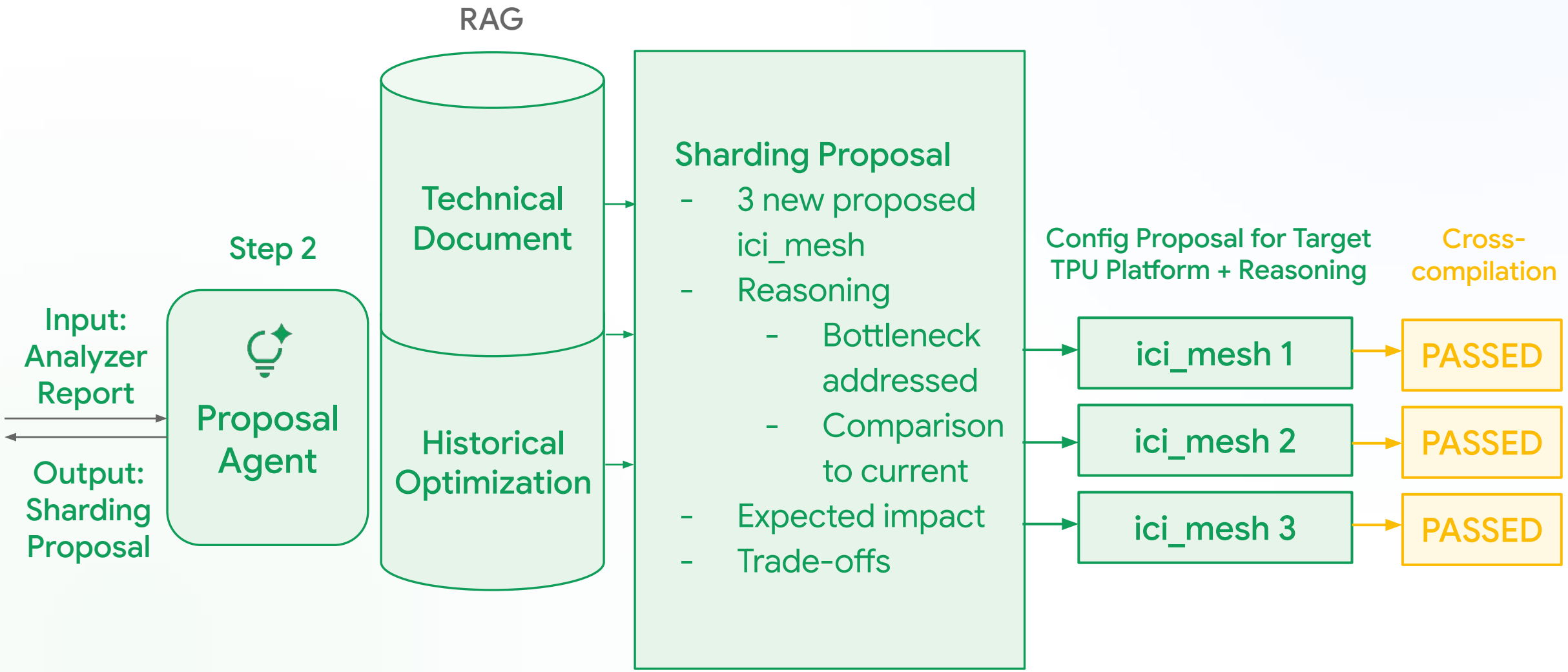
Analyzer Agent: Bottleneck Diagnosis from Raw Data



Proposal Agent: Explainable Sharding Proposal via RAG



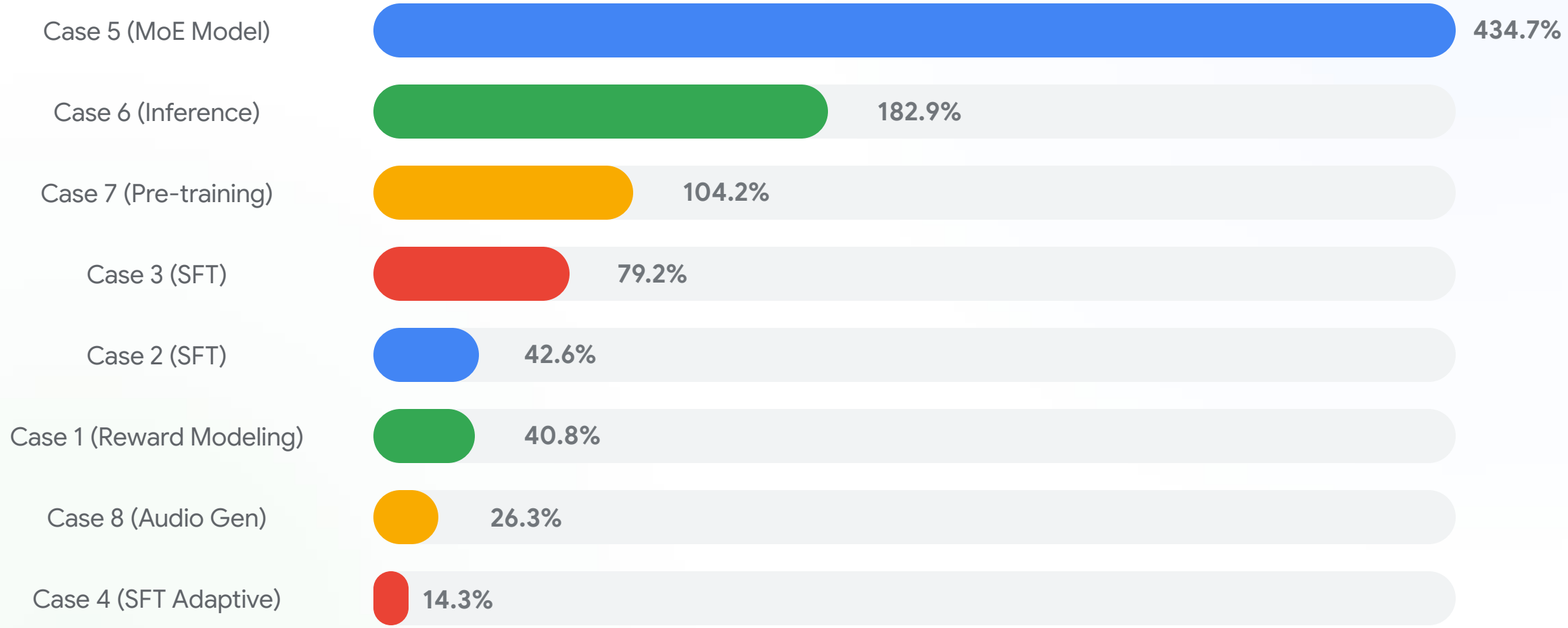
Validity Check: TPU compiler Cross-Compilation



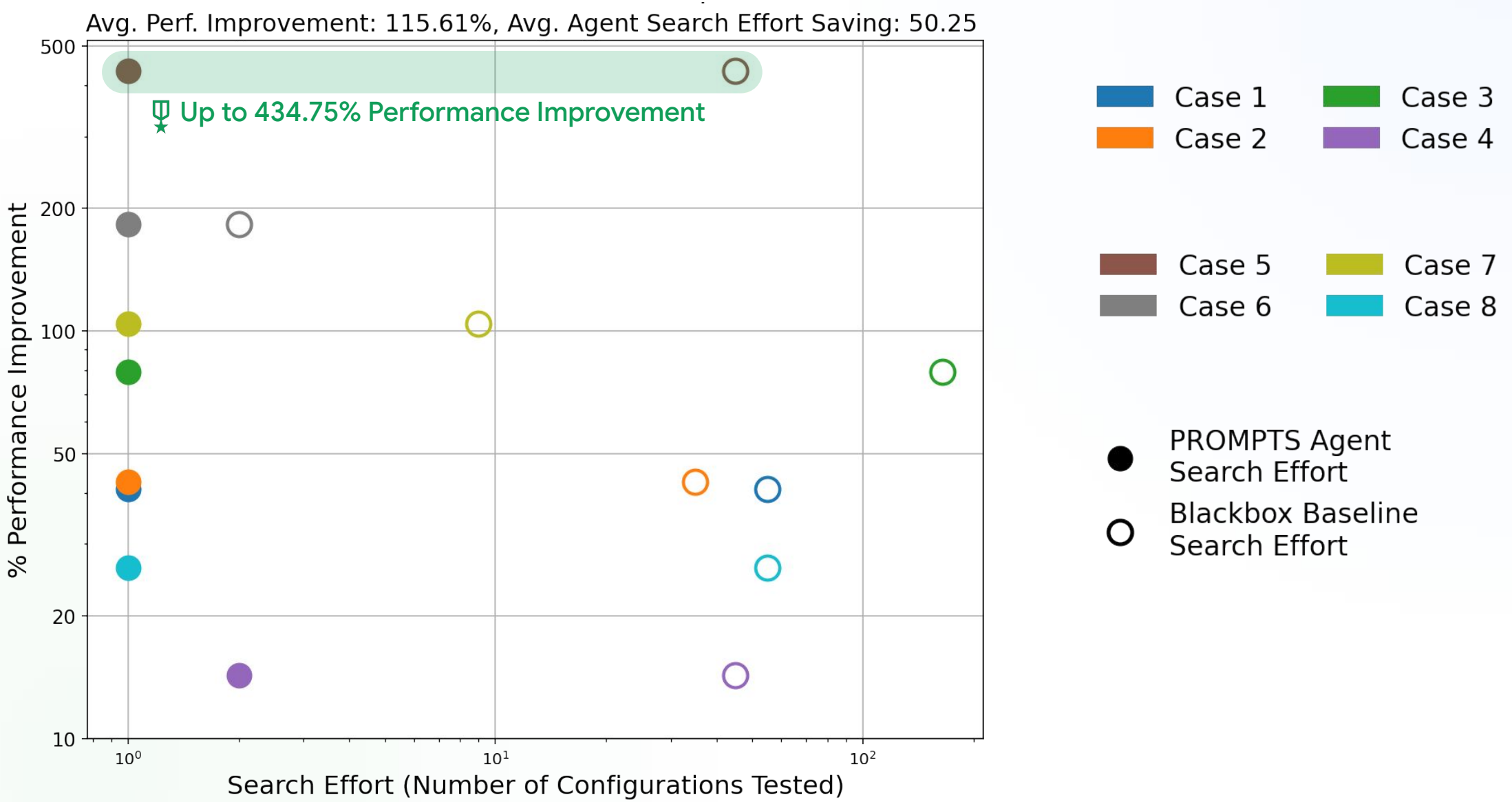
Extensibility Across 8 Production Workloads

Case	TPU System	Topology	Model Type	ML Task	Agent Task
1	v5p (3D)	8x8x8	Large Dense	Reward Modeling	Optimize ici_mesh
2	v6e (2D)	4x4	Small Dense	SFT (Post-train)	Optimize ici_mesh
3	v5e (2D)	16x16	Medium Dense	SFT (Post-train)	Optimize ici_mesh
4	v6e (2D)	16x16→8x16	Medium Dense	SFT (Post-train)	Adapt to new topology
5	v6e (2D)	16x16	MoE Model	Reward Modeling	Optimize ici_mesh
6	tpu7x (3D)	2 chips	Qwen 32B	Inference	Propose for new model
7	tpu7x (3D)	256 chips	DeepSeek MoE	Pre-training	Optimize ici_mesh
8	v5p (3D)	4x8x16	Audio Gen	Audio Generation	Optimize ici_mesh

Significant Performance Gains Across Workloads



Search Effort vs Performance Improvement



Conclusion and Key Impact

87.5%
Top Pick Production Match

434%
Max Throughput Increase

One
Shot Invocation Success

PROMPTS transforms performance engineering from unguided black-box search to explainable, expert-informed subspace identification.

Reasoning in Action

Compute-Bound (Case 1)

Diagnosis: 99.7% duty cycle with negligible communication overhead.

Proposal: Trade sequence parallelism for data parallelism (seq: 8⟶4, data: 8⟶16).

Logic: Better saturates TPU cores while reducing activation memory pressure.

HBM-Bound (Case 2)

Diagnosis: Collective operations stalled by memory bandwidth, not network.

Proposal: Introduce 4-way Model Parallelism to shard model parameters.

Logic: Directly alleviates HBM pressure caused by full model replication per chip.

Fail-Fast Verification & Compiler Loop

- **Hard Feasibility Constraints:** Memory capacity is a binary constraint. Performance tuning is secondary to "runnability."
- **Cross-Compilation:** Proposals are automatically sent to the TPU compiler. Invalid meshes or memory violations are rejected immediately.

Example (Case 4): A performance-optimized proposal caused an OOM. The system automatically pivoted to a "safe" configuration preserving high sequence parallelism.

Zero-Knowledge Stress Test (Case 6)

Probing Diagnostic Depth

We deliberately withheld specific documentation for the Qwen 32B model and tpu7x platform to test raw reasoning.

> **Result:** Agent identified the symptom (copy.2700 HBM stall) correctly as a "junior partner."

- **Expert Gap:** Identified as tensor relayout due to shape mismatch.
- **Path to Expert:** Bridgeable via targeted RAG context updates.

Zero-Knowledge Stress Test (Case 6)

Probing Diagnostic Depth

We deliberately withheld specific documentation for the Qwen 32B model and tpu7x platform to test raw reasoning.

> **Result:** Agent suggested fusing or overlapping memory-bound operations to hide latency.

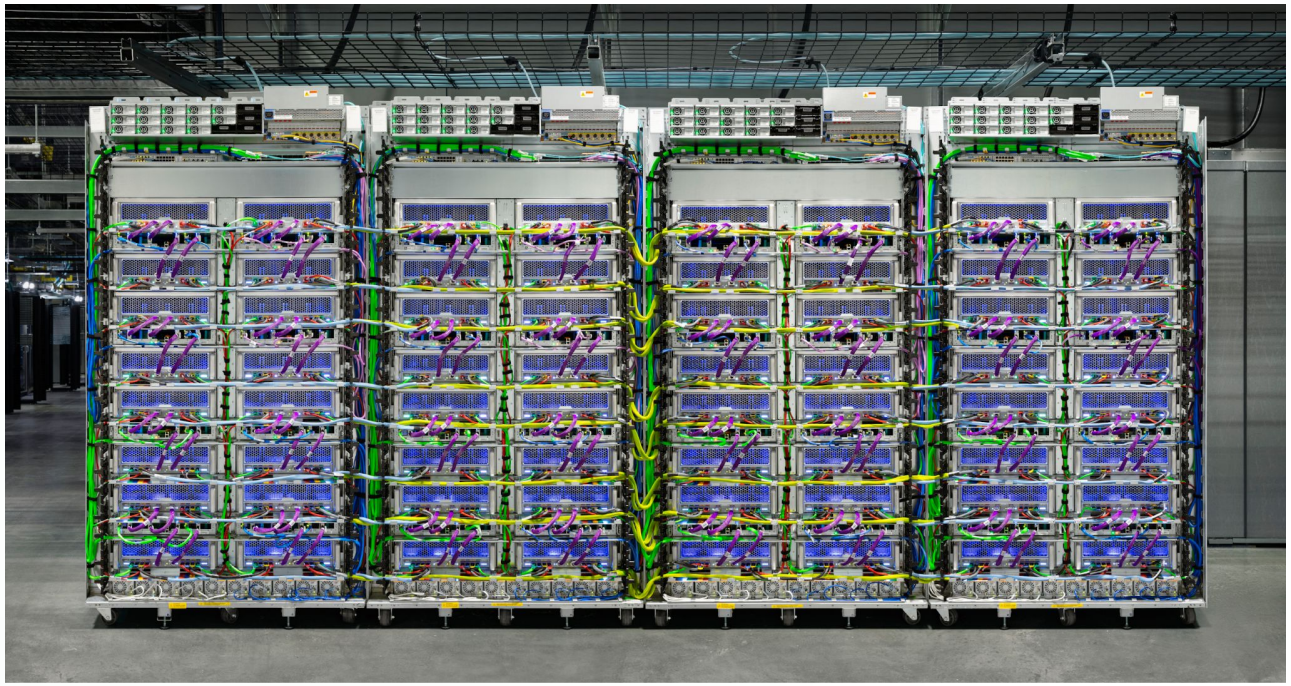
- **Expert Gap:** This is ineffective on tpu7x due to shared HBM bandwidth saturation. The actual fix should involve platform-specific techniques like sparse core offloading
- **Path to Expert:** Bridgeable via targeted RAG context updates.

Lessons Learned

Extensibility is a function of **Declarative Knowledge**, not algorithmic re-engineering.
To support new hardware, we simply provide new technical guides for the agent to read.

Summary

A methodological shift that reframes performance optimization from unguided search to expert-reasoning-guided subspace identification.



Questions?

dingyuran@google.com

MLSys Conference 2026