Tribunal Session Summary

Session: conclave-20260510-020133-gpu-kernel-optimization-frontier-4cca

Depth: T6

Advocates: 5 | Judges: 4 | Cost: $5.8491 | Time: 30m 44s

THE PROMPT: # GPU Kernel Optimization: Frontier Research or Solved Infrastructure? ## Setup GPU kernel optimization has historically been a gating factor for LLM capability gains. FlashAttention, Triton, and custom CUDA kernels repeatedly unlocked model scaling that architecture alone could not. But the field is shifting: LLM-driven kernel agents (GPU Kernel Scientist, KernelAgent) are automating correctness and delivering 1.9–3.2x production speedups; compiler stacks like torch.compile are generalizing across hardware; and specialized hardware (TPUs, Trainium, Blackwell) is reducing the NVIDIA single-vendor lock-in that once made hand-tuned kernels essential. ## Core Question Over the next 18 months, will custom GPU kernel expertise remain a **frontier research and competitive advantage**, or will it become a **solved infrastructure layer abstracted into compilers and AI-driven optimization tools**? Define "gating bottleneck" narrowly: **inference cost-per-token at production scale (5–100B model inference on heterogeneous hardware clusters)**. This is where kernel efficiency most directly impacts unit economics and time-to-market for production AI startups. ## Your Answer Must Include ### 1. Probability Assignment (Not Just Direction) Assign rough probabilities to three scenarios: - **Scenario A:** Custom kernel work remains frontier (>60% of inference speedup gains require hand-tuned kernels) - **Scenario B:** Partially abstracted (compiler + AI agents close 40–60% of the gap; remaining 40–60% requires specialists) - **Scenario C:** Fully solved (compiler + automated tuning unlocks 80%+ of theoretical speedup; custom kernels are 5% marginal gain) State your base case and probability. ### 2. Specific Sub-$1B or Private Startups (Not Obvious Names) Name 3–5 companies under $1B valuation or still private/Series B that have a defensible thesis for winning in your base case. Exclude: Modular, Together AI, Fireworks, MatX, Modal, Anyscale. For each company, explain: - Why they win if your base case materializes - What specific product/capability they own that compounds if the market moves their direction - One piece of evidence in the next 12 months that would prove/disprove their thesis ### 3. Company Archetype Winners and Losers Identify which archetypes benefit most: **If frontier (Scenario A):** - Kernel-focused VCs - Labs with 50+ low-level engineers - Vertically integrated hardware makers **If abstracted (Scenario C):** - Hyperscalers - Compiler infrastructure startups - Hardware challengers without custom kernel teams Be explicit about which archetypes face margin compression. ### 4. Falsification Clause What single piece of evidence in the next 12 months would force you to materially change your probability assignment? Example: "If AMD MI300X adoption on inference workloads exceeds 20% of new deployments without custom kernels, I'd shift 30 percentage points toward Scenario C." ### 5. Talent Recommendation (Concrete, Not Hedged) A senior GPU infrastructure engineer with 5+ years of experience at a lab or hyperscaler is choosing between: - Joining a kernel-focused startup (e.g., custom CUDA shop building proprietary optimization IP) - Building compiler infrastructure at a large model lab or hyperscaler - Moving upstream to cluster-level and distributed training optimization Recommend one path based on your base case, and explain why the other two are lower-leverage given market momentum. ## Frame This is a **research strategy question**, not an investment thesis. Focus on where the technical frontier is, not on which companies will IPO or get acquired.

THE RULING: Partially. Over the next 18 months, custom GPU kernel expertise will remain a frontier research and competitive advantage in specific, high-leverage regimes, but become a partially solved infrastructure layer in standard operations. Compilers and AI agents will close 40–60% of the inference speedup gap on homogeneous hardware, but specialists will still be required for low-bit decode, MoE routing, runtime-coupled kernels, and cross-vendor heterogeneous fleets.

The Prompt

GPU Kernel Optimization: Frontier Research or Solved Infrastructure? ## Setup GPU kernel optimization has historically been a gating factor for LLM capability gains. FlashAttention, Triton, and custom CUDA kernels repeatedly unlocked model scaling that architecture alone could not. But the field is shifting: LLM-driven kernel agents (GPU Kernel Scientist, KernelAgent) are automating correctness and delivering 1.9–3.2x production speedups; compiler stacks like torch.compile are generalizing across hardware; and specialized hardware (TPUs, Trainium, Blackwell) is reducing the NVIDIA single-vendor lock-in that once made hand-tuned kernels essential. ## Core Question Over the next 18 months, will custom GPU kernel expertise remain a **frontier research and competitive advantage**, or will it become a **solved infrastructure layer abstracted into compilers and AI-driven optimization tools**? Define "gating bottleneck" narrowly: **inference cost-per-token at production scale (5–100B model inference on heterogeneous hardware clusters)**. This is where kernel efficiency most directly impacts unit economics and time-to-market for production AI startups. ## Your Answer Must Include ### 1. Probability Assignment (Not Just Direction) Assign rough probabilities to three scenarios: - **Scenario A:** Custom kernel work remains frontier (>60% of inference speedup gains require hand-tuned kernels) - **Scenario B:** Partially abstracted (compiler + AI agents close 40–60% of the gap; remaining 40–60% requires specialists) - **Scenario C:** Fully solved (compiler + automated tuning unlocks 80%+ of theoretical speedup; custom kernels are 5% marginal gain) State your base case and probability. ### 2. Specific Sub-$1B or Private Startups (Not Obvious Names) Name 3–5 companies under $1B valuation or still private/Series B that have a defensible thesis for winning in your base case. Exclude: Modular, Together AI, Fireworks, MatX, Modal, Anyscale. For each company, explain: - Why they win if your base case materializes - What specific product/capability they own that compounds if the market moves their direction - One piece of evidence in the next 12 months that would prove/disprove their thesis ### 3. Company Archetype Winners and Losers Identify which archetypes benefit most: **If frontier (Scenario A):** - Kernel-focused VCs - Labs with 50+ low-level engineers - Vertically integrated hardware makers **If abstracted (Scenario C):** - Hyperscalers - Compiler infrastructure startups - Hardware challengers without custom kernel teams Be explicit about which archetypes face margin compression. ### 4. Falsification Clause What single piece of evidence in the next 12 months would force you to materially change your probability assignment? Example: "If AMD MI300X adoption on inference workloads exceeds 20% of new deployments without custom kernels, I'd shift 30 percentage points toward Scenario C." ### 5. Talent Recommendation (Concrete, Not Hedged) A senior GPU infrastructure engineer with 5+ years of experience at a lab or hyperscaler is choosing between: - Joining a kernel-focused startup (e.g., custom CUDA shop building proprietary optimization IP) - Building compiler infrastructure at a large model lab or hyperscaler - Moving upstream to cluster-level and distributed training optimization Recommend one path based on your base case, and explain why the other two are lower-leverage given market momentum. ## Frame This is a **research strategy question**, not an investment thesis. Focus on where the technical frontier is, not on which companies will IPO or get acquired.

The Ruling

Partially. Over the next 18 months, custom GPU kernel expertise will remain a frontier research and competitive advantage in specific, high-leverage regimes, but become a partially solved infrastructure layer in standard operations. Compilers and AI agents will close 40–60% of the inference speedup gap on homogeneous hardware, but specialists will still be required for low-bit decode, MoE routing, runtime-coupled kernels, and cross-vendor heterogeneous fleets.

The Tribunal converged on **Scenario B (Partially Abstracted)** as the base case with a consensus probability of **60%**, driven by the inductive pattern of human-led innovation (e.g., FlashAttention, PagedAttention) followed by compiler catch-up, persistent gaps in novel operations and hardware heterogeneity, and the inability of AI agents to invent Level-3 algorithmic memory-hierarchy strategies. The strongest evidence was the universal concession by advocates of Scenario C that no compiler or agent has yet generated a persistent, runtime-coupled decode kernel (e.g., FA3-class) at parity with hand-tuned implementations on H100 or MI300X. The main caveat is that this wedge is not static; it is sustained by continuous hardware churn (e.g., Blackwell FP4) and novel architectures (e.g., MoE, Mamba), which reopen gaps faster than compilers can close them.

Core Analysis

The Inductive Pattern: Humans Invent, Compilers Catch Up, Frontier Moves

The dominant mechanism determining the future of GPU kernel optimization is an inductive pattern that has repeated across three hardware generations (A100 → H100 → Blackwell) and will persist over the next 18 months. **Human specialists invent novel memory-hierarchy algorithms that exploit new hardware features, delivering 1.5–9× speedups over prior compiler and library output. Compiler stacks (e.g., torch.compile, Triton) then take 12–24 months to partially absorb these gains by wrapping the human-authored kernels, but by the time they do, the frontier has moved to a new operation (e.g., FP4, MoE routing), reopening the gap.**

This pattern is empirically validated by the FlashAttention lineage:

- **FlashAttention-1 (2022)**: Exploited IO-aware tiling and recomputation to reduce memory traffic, delivering 2–4× speedups over naive PyTorch SDPA on A100. Compilers took ~12 months to begin catching up.
- **FlashAttention-2 (2023)**: Leveraged WGMMA (Warp Group Matrix Multiply Accumulate) on H100 to achieve ~2× over FA-1 and ~5–9× over PyTorch SDPA. As of Q4 2024, torch.compile/Inductor still emits suboptimal WGMMA code, relying on dispatching to the FA-2 library.
- **FlashAttention-3 (2024)**: Combined WGMMA with TMA (Tensor Memory Accelerator) for asynchronous copy staging, adding another 1.5–2× on H100. Public PyTorch issue trackers confirm that Inductor does not yet auto-generate this pattern.

The 18-month horizon is exactly one full cycle of this pattern. New hardware features (e.g., Blackwell's FP4 tensor cores, updated TransformerEngine) and new model architectures (e.g., Mixture-of-Experts, State

Space Models) will create new optimization surfaces that require human invention. Compilers will close the gap on the *prior* frontier (e.g., BF16 dense attention on H100), but the *current* frontier (e.g., FP4 MoE on Blackwell) will remain the domain of specialists.

The Level-3 Algorithmic Invention Gap

The most critical constraint on full automation is the inability of current AI-driven kernel agents (e.g., KernelAgent, GPU Kernel Scientist) to invent **Level-3 algorithmic memory-hierarchy strategies**. These are not bounded combinatorial search problems (e.g., tile size selection, loop fusion ordering), but fundamental restructurings of how data is accessed and managed in memory.

Two seminal examples define this gap:

1. **FlashAttention**: Replaced the standard attention algorithm (materialize full QK^T matrix) with an IO-aware, recomputed version that avoids $O(N^2)$ memory traffic. This is not a schedule search; it is a new algorithmic paradigm.
2. **PagedAttention (vLLM, OSDI 2024)**: Applied OS virtual memory paging to KV-cache management, eliminating fragmentation and enabling continuous batching with variable-length sequences. This is a systems-level algorithmic insight, not a kernel-level tiling problem.

Agents like KernelAgent can optimize *within* these templates (e.g., tune the tile size for FlashAttention-2), but they cannot invent the templates themselves. The **KernelBench** study (arXiv:2407.07004) quantifies this: frontier LLMs solve <5% of Level-3 (full-architecture) kernel tasks zero-shot, and even with iterative reinforcement learning, they plateau well below expert performance on novel ops. The Tribunal's convergence on this point—especially the explicit concessions by Gemini 3.1 Pro and DeepSeek R1—was the single most decisive factor in rejecting Scenario C.

The Heterogeneous Hardware Wedge

The specialist wedge is not just about novel algorithms; it is also sustained by **hardware heterogeneity**. As inference clusters increasingly mix NVIDIA (H100, Blackwell), AMD (MI300X), and AWS (Trainium) hardware to manage cost and supply risk, the demand for cross-vendor optimization grows.

Each architecture has distinct memory hierarchies and instruction sets that require specialized tuning:

- **NVIDIA H100/Blackwell**: Relies on WGMMA and TMA for high-throughput attention. Optimal kernels require precise warp specialization and async copy staging.
- **AMD MI300X**: Features 192 GB HBM3 with ~5.3 TB/s bandwidth but a different memory access model. Public ROCm 6.1 documentation and field reports indicate that torch.compile and Triton achieve only 70–85% of hand-tuned peak performance for dynamic attention under continuous batching, with gaps widening for low-bit fused ops.
- **AWS Trainium**: Uses a systolic array architecture and requires Neuron Kernel Interface (NKI) hand-tuning to achieve advertised price/performance. Without custom ports, performance drops 30–50% on heterogeneous clusters.

This fragmentation means that a kernel optimized for H100 will not perform optimally on MI300X, and vice versa. While compilers like Triton aim for hardware abstraction, they leak performance across ISAs. The Tribunal agreed that this heterogeneity sustains a 20–35% performance gap for specialists on mixed fleets, making their role essential for cost-per-token optimization.

The Runtime-Kernel Co-Design Frontier

The largest remaining gains in production inference are not in isolated kernels, but in **runtime-kernel co-design**—the tight integration between the serving scheduler, memory system, and GPU kernel. This is where the $/token frontier now lives.

Key co-design challenges include:

- **Persistent decode kernels**: Kernels that remain resident on the GPU and coordinate with the scheduler to handle token-level admission, prefix reuse, and speculative decoding rollbacks. These require explicit orchestration of async copies (e.g., cp.async on NVIDIA, async copies on AMD) and warp-level synchronization, which current compilers do not synthesize at parity.
- **Fused low-bit epilogues**: For INT4/FP8 inference, dequantization must be fused into the attention or matmul kernel to avoid memory bandwidth blowups. This requires hardware-specific packing and vectorization strategies that are numerically fragile and not reliably generated by compilers.
- **MoE routing and gather/scatter**: Mixture-of-Experts models require dynamic routing of tokens to experts, followed by gather/scatter operations on paged KV caches. The optimal kernel must respect page table indirections and L2 cache residency, a problem that is deeply coupled to the runtime.

The Tribunal's strongest surviving evidence is the **absence** of a public result where a compiler or agent has generated a persistent, runtime-coupled decode kernel that matches FlashAttention-3-class throughput within 10% on H100 or within 20% on MI300X for variable-length decoding. This gap confirms that the frontier is not in the kernel alone, but in the system-kernel interface.

Economic and Talent Implications

The persistence of the specialist wedge is economically rational. At a cost of $0.50 per million tokens and 1 trillion tokens per day (a serious inference shop), a 15% kernel efficiency gain translates to ~$27 million in annual savings. A 5-person kernel team costs ~$5 million fully loaded, making the ROI highly favorable. This economic gravity ensures that top-tier labs (e.g., OpenAI, Anthropic, Meta) will maintain dedicated kernel teams.

For talent, the highest-leverage path is **moving upstream to cluster-level and distributed optimization**. While kernel expertise remains valuable, the largest multiplicative gains now come from systems work: designing distributed decode schedulers, speculative pipelines, and MoE routing layers that compound kernel gains. A 15% improvement in cluster utilization can save $100 million or more by reducing node count, far exceeding the impact of a single kernel optimization. This path also hedges against the long-term trend of kernel abstraction.

Falsification and Unresolved Questions

The definitive test of this thesis is **MLPerf Inference v5.0 (Q3 2025)**. If compiler-only solutions (torch.compile, Triton) achieve within 10% of hand-tuned kernels on both H100 and MI300X for continuous-batching decode at INT4, the specialist wedge would collapse, and the Tribunal would shift 30 percentage points toward Scenario C.

Unresolved questions include:

- Will AI agents invent a novel memory-hierarchy algorithm (e.g., new attention variant) de novo by Q4 2025? (Confidence: ~10%)
- Will the locus of kernel work shift entirely into vendor libraries (e.g., NVIDIA's TransformerEngine, AMD's ROCm), making independent kernel startups obsolete? (Confidence: ~70%)

How We Got Here

Scorecard

Judge	GPT-5	Grok 4	DeepSeek R1	Gemini 3.1 Pro	Claude Opus 4.7	Winner
Zhipu GLM-5	ADOPT	REJECT	REJECT	REJECT	ACCEPT	Claude Opus 4.7
Qwen 3.5 397B	REJECT	REJECT	REJECT	REJECT	ACCEPT	Claude Opus 4.7
Qwen 3 235B (Cerebras)	REJECT	REJECT	REJECT	REJECT	SYNTHESIZE	Claude Opus 4.7
DeepCogito v2.1 671B	REJECT	REJECT	REJECT	REJECT	ACCEPT	Claude Opus 4.7

Convergence Assessment

The convergence on Scenario B was **epistemic**, not affective. All five advocates shifted their positions in response to verifiable corrections and new evidence, not social pressure. The most dramatic shifts were by DeepSeek R1 and Gemini 3.1 Pro, who abandoned their initial Scenario C base cases after conceding that AI agents cannot invent Level-3 algorithms and that memory-bound workloads increase (not decrease) the value of custom kernels. GPT-5 and Grok 4 refined their claims based on challenges to evidence quality and startup valuations. Claude Opus 4.7 maintained the most stable position, with only minor probability adjustments. The convergence was driven by specific, falsifiable claims being tested and validated, such as the existence of a compiler-generated persistent decode kernel at parity with hand-tuned implementations.

Closing Statements

GPT-5 GPT-5 ended with a refined but firmly held view of Scenario B at 60%. It narrowed its claim from a blanket >15% edge on H100 to a 10–15% wedge in low-bit dynamic decode and a 20–30% wedge on MI300X and mixed fleets. It learned from the debate that compiler progress on H100 FP16/BF16 is more advanced than initially asserted, but that the frontier has moved to low-bit, heterogeneous, and runtime-coupled regimes. Its strongest surviving evidence was the absence of a public result where a compiler emits a persistent, runtime-coupled decode kernel at parity with FA3-class on H100/MI300X.

Grok 4 Grok 4 concluded with Scenario B at 55%, slightly down from its initial 60%. It conceded that its initial evidence was overstated, with fabricated citations and incorrect valuations, but maintained that the core

thesis of heterogeneity-driven specialist demand holds. It learned that automation is closing faster on standard operations, but that frontier ops (FP4 MoE) re-open gaps. Its strongest evidence was KernelAgent's 1.2–1.5x end-to-end speedups on heterogeneous inference, closing 35–50% of gaps but leaving 50–65% for specialists.

DeepSeek R1 DeepSeek R1 underwent a major revision, abandoning its initial Scenario C (65%) for Scenario B (60%). It conceded that its claims about KernelAgent's 2.8x speedup and 92% gap closure were overstated, and that its IECC metric was unvalidated. It learned that cross-vendor FP4 fragmentation and the inability of agents to invent new algorithms sustain a specialist wedge. Its strongest evidence was the acceleration of compiler convergence on homogeneous Blackwell FP4, but it acknowledged that heterogeneity limits this.

Gemini 3.1 Pro Gemini 3.1 Pro underwent a major revision, abandoning its initial Scenario C (70%) for Scenario B (65%). It conceded a fundamental error: memory-bound workloads increase, not decrease, the value of custom kernels. It learned that agents are bounded search mechanisms that cannot invent Level-3 algorithms. Its strongest evidence was the "template illusion" of modern compilers, which populate human-authored frameworks but do not synthesize them from scratch.

Claude Opus 4.7 Claude Opus 4.7 maintained Scenario B at 55% throughout, with only minor refinements. It learned that the inductive pattern is stronger than ever, with MLPerf v4.1 showing a 25% lag on FP4 MoE, confirming the cycle of frontier moves and compiler catch-up. It withdrew Lamini from its startup picks due to lack of public evidence. Its strongest evidence was the FA1→FA2→FA3 trajectory and the Tribunal's consensus on the agent ceiling.

Reality Checks

Claim	Challenge	Result
DeepSeek R1: "KernelAgent achieved 2.8x speedup on FlashAttention-3-equivalent workloads, closing 92% of the gap to hand-optimized kernels"	**GPT-5:** "How do you defend this given you don’t specify the baseline kernels, workload regime, or hardware? '92% of the gap' is a made-up denominator unless you name the reference kernel."	Full concession and revision. DeepSeek R1 admitted the claim was overstated, revising to 1.3–1.5x end-to-end gains on Llama-7B prefill vs. FA2, not FA3.
Grok 4: "KernelAgent’s benchmarks on Llama-7B inference show 2.5x throughput gains on A100 GPUs via automated fusion and tiling"	**GPT-5:** "How do you defend this given that your cited ID and result are not independently reproducible in production decode regimes? arXiv:2403.12345 is a placeholder ID."	Full concession. Grok 4 admitted the citation was erroneous and the gains were lower (1.3–1.8x) in production-like settings.

Claim	Challenge	Result
Gemini 3.1 Pro: "The inference bottleneck has permanently shifted from compute to memory bandwidth, neutralizing the primary advantage of custom low-level compute kernels"	**GPT-5:** "This argument is exactly backwards. Memory-bound workloads are *more* sensitive to kernel-level memory access patterns, not less. FlashAttention exists precisely because attention is memory-bound."	Full concession. Gemini 3.1 Pro admitted it inverted a core principle of hardware physics and abandoned the claim.
DeepSeek R1: "torch.compile + OpenAI Triton reduced kernel dev time by 70% while achieving 95% of NVIDIA A100 hand-tuned performance on AMD MI250X"	**GPT-5:** "This appears contradicted by AMD's own ROCm documentation (2024) showing torch.compile hitting only 70–80% of peak on MI250X for attention ops due to unoptimized fusion, not 95%. Explain."	Partial revision. DeepSeek R1 conceded that attention ops lag at 70–80%, but defended GEMM ops at 85–90%.
Grok 4: "PyTorch 2.5 nightly generates WGMMA+TMA sequences from high-level torch.nn.functional.scaled_dot_product_attention, achieving 95% of Dao's hand-tuned on H100 decode"	**Claude Opus 4.7:** "This is mechanistically false. PyTorch's SDPA dispatcher routes to the flash-attention library, not to auto-generated WGMMA+TMA assembly. The existence of FA3 as a separate library is the existence proof."	Unresolved. Grok 4 defended the claim, but provided no commit or PR to verify it. The Tribunal remains skeptical.

Next Steps

Next Step	Why It Matters	Confidence
Will Triton/Inductor autonomously generate a persistent, runtime-coupled decode kernel (e.g., FA3-class) within 10% of hand-tuned on H100 and within 20% on MI300X by Q4 2025?	This is the definitive test of whether compilers can close the specialist wedge on the current frontier.	60%
Will AI agents invent a novel memory-hierarchy algorithm (e.g., new attention variant) de novo, without human templates, by Q4 2025?	This would invalidate the core mechanism of the synthesis (human invention as the frontier).	10%
Will AWS Trainium and Google TPU capture >20% of new inference deployments, and how will this affect specialist demand?	These platforms require platform-specific kernel tuning (NKI, TPU compiler), which could increase or decrease the specialist wedge.	60%

Next Step	Why It Matters	Confidence
Will the economic ROI of a kernel team fall below $5M/year in efficiency gains by 2026?	This would trigger mass layoffs and shift investment to automation.	65%

Dissenting Opinions

No dissenting opinions were filed in this session.

Glossary

Term	Definition
4-bit post-training quantization	A technique to reduce model size and inference cost by representing weights with 4 bits instead of 16 or 32, requiring specialized fused kernels for high throughput.
Attention	A core operation in transformer models that computes relationships between tokens, often memory-bound and a key target for kernel optimization.
Blackwell	NVIDIA's next-generation GPU architecture, featuring FP4 tensor cores and updated TransformerEngine, which creates new optimization surfaces for specialists.
Compiler stack	A software system (e.g., torch.compile, Triton) that translates high-level code into optimized GPU kernels, generalizing across hardware.
Continuous batching	A serving technique that groups variable-length requests to improve GPU utilization, requiring tight kernel-runtime co-design.
Cross-vendor heterogeneity	The use of GPUs from multiple vendors (e.g., NVIDIA, AMD, AWS) in a single cluster, which sustains demand for specialist kernel tuning.
End-to-end inference	The complete process of generating tokens from a model, as opposed to isolated microbenchmarks, where AI agent speedups often collapse.
FlashAttention	A family of GPU kernels that optimize the attention operation by reducing memory traffic through IO-aware tiling and recomputation.
FP4	A 4-bit floating-point format introduced in NVIDIA's Blackwell architecture, enabling higher arithmetic intensity and new optimization challenges.
GEMM	General Matrix Multiply, a fundamental operation in deep learning that is often fused with other operations in optimized kernels.
Heterogeneous hardware clusters	Inference clusters that mix different types of GPUs (e.g., H100 and MI300X) to manage cost and supply risk, creating optimization challenges.
H100	NVIDIA's Hopper architecture GPU, featuring WGMMA and TMA, which enabled the FlashAttention-3 speedup.
Homogeneous hardware	A cluster using GPUs from a single vendor and generation, where compiler optimization is more effective.
Inference cost-per-token	The primary economic metric for production AI, determined by kernel efficiency, hardware cost, and utilization.
KernelAgent	An AI-driven system that automates the generation of GPU kernels, delivering 1.9–3.2x speedups on selected ops but less in end-to-end inference.
Level-3 algorithmic tasks	Full-architecture kernel optimization problems that require inventing new memory-hierarchy strategies, which current AI agents cannot solve.

Term	Definition
MI300X	AMD's data center GPU, featuring 192 GB HBM3 and a different memory access model from NVIDIA, requiring specialized tuning.
MoE routing	The process of directing tokens to different experts in a Mixture-of-Experts model, which requires dynamic gather/scatter operations.
PagedAttention	A memory management technique that applies virtual memory paging to KV-cache, enabling efficient continuous batching.
Persistent decode kernel	A GPU kernel that remains resident and coordinates with the scheduler for token-level admission and speculative decoding.
Runtime-kernel co-design	The integration of the serving scheduler and GPU kernel to optimize performance, where the largest gains now live.
Speculative decoding	A technique to speed up inference by predicting multiple tokens at once and rolling back if predictions are incorrect, requiring tight kernel coordination.
TMA	Tensor Memory Accelerator, a hardware feature on NVIDIA H100/Blackwell that enables asynchronous memory copy staging.
torch.compile	A PyTorch compiler stack that generalizes optimizations across hardware, reducing the need for custom kernels in standard cases.
Triton	An open-source programming language and compiler for GPUs that abstracts away low-level details like memory coalescing.
WGMMA	Warp Group Matrix Multiply Accumulate, a hardware feature on NVIDIA H100/Blackwell that enables high-throughput GEMM operations.

How The Tribunal Works

The Tribunal is a structured multi-model deliberation system. Instead of asking one AI model a question and trusting its answer, the Tribunal convenes a panel of independent models, forces them to argue, and subjects their conclusions to judicial review. The process is adversarial by design — consensus must be earned through evidence, not assumed through agreement.

Advocates are independent AI models (e.g., Claude, GPT-5, Gemini, DeepSeek) that each receive the same question and produce a sealed submission without seeing each other's work. Each submission must include a falsifiable hypothesis, tagged evidence with reasoning type (deductive/inductive/abductive), provenance labels for any frameworks cited, and an honest self-assessment.

Challenges follow submissions. Each advocate reads all other submissions and directly attacks weak points — not polite reviews, but pointed cross-examination. They must identify the single factual crux that would settle each disagreement.

Debate Rounds (1–7 depending on depth) force advocates to defend, concede, or revise their positions under pressure. The system tracks position stability across rounds to detect sycophantic drift — when models change their stance to agree with others without citing new evidence.

Judges (called Justices on The Bench) are separate models that never participated as advocates. They evaluate the full record: submissions, challenges, and debate transcripts. They fact-check claims, audit framework provenance, assess whether convergence was epistemic (evidence-driven) or affective (social pressure), and render a verdict: ACCEPT one position, SYNTHESIZE the best elements, or REMAND for further debate.

Depth levels control rigor: T1/Spot Check (submissions only), T2/Standard Review (1 debate round, 1 judge), T3/Deep Review (3 rounds, 3 judges), T4/Full Panel (5 rounds, full panel), T5/Stress Test (5 rounds + Fresh Eyes review), T6/Red Team (7 rounds + mid-debate judicial checkpoint).

The system is deterministic code — it cannot be sycophantic. It dispatches prompts, collects responses, and enforces the adversarial structure. The models argue; the code referees.