

VITAMIN-E: Determinism-Preserving GPU Spatial Sharing for LLM Systems

Zhenyuan Yang^{*,1,2,3}

Wenxin Zheng^{*,4}

Mingyu Li^{1,2,3}

¹ Key Laboratory of System Software (Chinese Academy of Sciences)

² Institute of Software Chinese Academy of Sciences

³ University of Chinese Academy of Sciences

⁴ Shanghai Jiao Tong University

Abstract

GPU spatial sharing can improve utilization as LLM workloads fluctuate across launches, but resource-driven kernel reshaping can change floating-point reduction order and output bits. We present VITAMIN-E, a runtime that adapts physical GPU allocation without rewriting the submitted computation. Its *parallel-structure invariant* preserves kernel binaries, launch structures, and application-required dependencies. Immutable operation descriptors and logical completion tokens enable late binding of ready work to compatible, pre-created Green Contexts. For fixed-structure deterministic workloads in an unchanged environment, VITAMIN-E preserves bitwise-identical outputs across eligible allocations. Across 1,000 operator trials and 100 fixed-state first-token replays, VITAMIN-E preserves bitwise equality; reshaping changes the selected token in five replays. Separately, VITAMIN-E achieves 1.38–1.63 $\times$ the geometric-mean aggregate normalized training throughput of five baselines and 8.1–58.4% lower geometric-mean inference p99 latency in mixed workloads. These results demonstrate that GPU spatial elasticity can improve performance without sacrificing bitwise determinism under the parallel-structure invariant.

Keywords

GPU spatial sharing, LLM systems, determinism, late binding

1 Introduction

LLM training and serving expose varying GPU demand: compute alternates with communication and transfer, while inference alternates between prefill, decode, and idle periods [4, 16, 22]. Spatial sharing can exploit complementary demand if allocations follow these changes across launches [30]. MPS, MIG, and Green Contexts preserve submitted kernels but bind resources at the process, instance, or context level, limiting adaptation to changing per-launch demand [6, 7, 18]. Recent systems pursue finer-grained, kernel-level scheduling to adapt resources dynamically [5, 15]. However, approaches

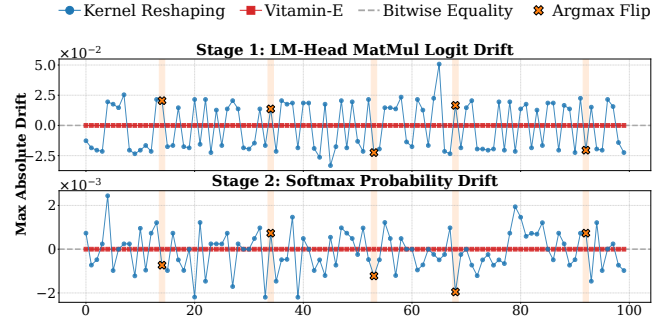


Figure 1: One prompt, one fixed state, 100 replays of the first generated token. Reshaping changes the selected token in five replays; VITAMIN-E preserves bitwise-identical logits and probabilities.

that reshape block sizes, grid dimensions, or work decomposition can change floating-point reduction order [5, 15, 35]. Consequently, numerical outputs may depend on transient resource availability and unrelated co-tenant activity.

For LLMs, small numerical differences can become visible at the output. When leading logits are close, a perturbation can change the greedy argmax even at temperature zero [12]. Figure 1 demonstrates this effect at one fixed decoding state. Such sensitivity complicates debugging, failure reproduction, regression testing, evaluation reproducibility, and rollout-training consistency [17, 23]. Batch formation and sequence length can also change the computation; here we isolate resource-driven kernel reshaping during GPU spatial sharing, holding the execution otherwise fixed.

Our key insight is that GPU execution decouples logical parallel structure from physical resource allocation. The same kernel grid can execute on different numbers of SMs by varying block placement and wave count, without changing its work decomposition. We formalize this separation as the *parallel-structure invariant*: physical allocation may vary, while the kernel binary, submitted parallel structure, and application-required happens-before order remain unchanged. For workloads that are already fixed-structure deterministic in an unchanged environment, preserving this invariant guarantees bitwise-identical outputs across eligible

* Both authors contributed equally to this work.

allocations. This establishes a foundation for fine-grained GPU elasticity without sacrificing determinism.

We realize this contract in VITAMIN-E, a runtime that late-binds ready operations to compatible Green Contexts. The challenge is to preserve stream and event semantics across changing contexts, safely reuse overlapping allocations, and minimize coordination overhead. Immutable operation descriptors and logical dependency tokens decouple application execution from physical resources, while a pre-created hierarchical context pool enables efficient and safe rebinding. This separation allows scheduling policies to dynamically assign available SM capacity to ready operations, adapting to workload demand without modifying kernel structure or required dependencies. By reducing resource waiting and reclaiming idle capacity, VITAMIN-E improves GPU utilization and colocated workload performance.

2 Parallel-Structure Invariance

Contract and invariant. For launch k , let B_k denote its kernel binary, C_k its submitted parallel structure (launch geometry and logical work decomposition), and P_k its physical allocation. Let X include inputs, argument values, and initial device-visible state, and let H denote the application-required happens-before order. A workload is *fixed-structure deterministic* if executions with identical X , $\{B_k\}$, $\{C_k\}$, and H produce bitwise-identical outputs in a fixed hardware/software environment. Our *parallel-structure invariant* requires $B'_k = B_k$, $C'_k = C_k$, and $H' = H$, while allowing physical allocation P'_k to vary with X and the environment unchanged. Such adaptation may change block placement, wave count, execution overlap, and completion time without modifying the submitted computation. For fixed-structure deterministic workloads, preserving this invariant guarantees bitwise equality across eligible physical allocations.

Why reshaping matters. Resource-driven reshaping instead replaces C_k with $\tilde{C}_k = g(C_k, R_k)$, where R_k denotes transient resource state. For example, μ Share changes block sizes, while LithOS decomposes kernels into execution atoms [5, 15]. Floating-point addition is non-associative: for some a, b, c , $(a \oplus b) \oplus c \neq a \oplus (b \oplus c)$ [10]. Repartitioning a reduction or Split-K GEMM can therefore alter the accumulation tree and output bits. Not every transformation changes the result, but resource-dependent reshaping can make numerical outputs depend on transient resource availability and co-tenant activity. Figure 2 illustrates this distinction.

Scope and limitations. The guarantee preserves existing determinism; it does not make arbitrary GPU programs deterministic. Data races, order-sensitive atomics, clock-dependent code, and nondeterministic library routines can violate the premise [17, 21, 33]. Changes in architecture,

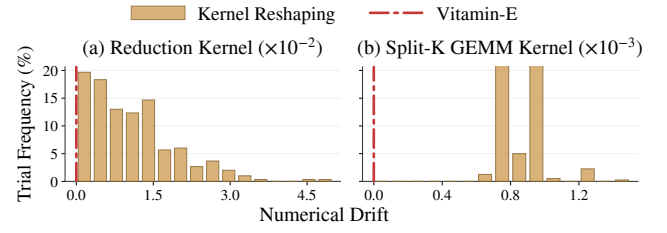


Figure 2: Output drift over 1,000 matched-input trials. Reshaping changes reduction and Split-K GEMM outputs; allocation-only rebinding preserves every output bit.

compiler, algorithm, or precision are also outside the contract [3, 31]. The invariant governs changes in physical allocation, not application-defined computation; reproducibility across application configurations may require complementary algorithms or serving-layer controls [1, 32]. Finally, allocation changes must remain compatible with the original launch, including any special residency requirements. Appendix A formalizes the contract.

3 The VITAMIN-E Runtime

Immutable logical execution. A virtual context (vCtx) maintains application-visible streams, events, synchronization state, and immutable operation descriptors independently of physical allocation. Each kernel descriptor captures entry point, launch-time argument values, grid and block dimensions, dynamic shared-memory size, and launch attributes. Arguments are copied by value, including pointer values; the application remains responsible for the lifetime and synchronization of referenced memory. Memory copies and memsets are represented similarly to preserve their dependencies. Scheduling policies may select where operations execute but cannot modify their descriptors or dependencies.

Dependencies across physical contexts. Each operation produces a completion token and captures its predecessors. A logical stream forms a token chain regardless of which backend stream executes each operation. An event record creates a generation tied to the recording stream's completion point; a wait captures the visible generation, preventing re-recording from retargeting waits. Default-stream dependencies are represented explicitly rather than inherited from backend streams. Only operations with completed predecessors enter the ready queue (Figure 3). This prevents rebinding from violating dependencies or occupying scarce allocations with work that cannot yet execute. Appendix B details event generations and default-stream semantics.

Allocation reuse and preparation. A physical context (pCtx) combines a Green Context, nonblocking backend

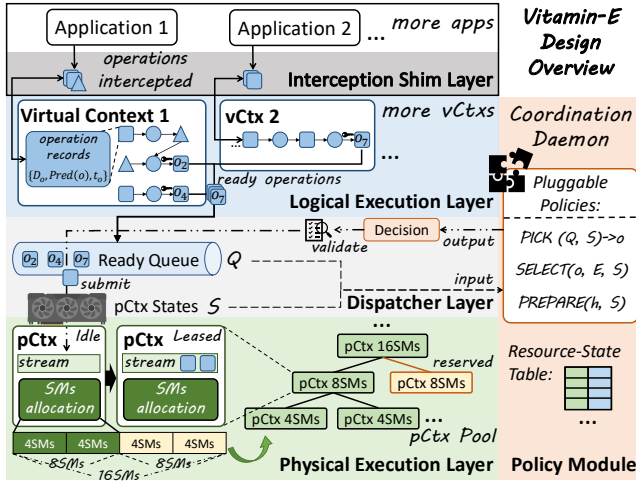


Figure 3: Immutable operations and logical dependencies are separated from replaceable physical contexts. The dispatcher validates each proposed binding before submission.

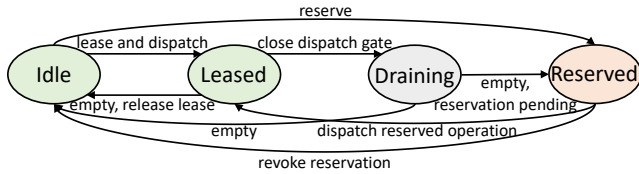


Figure 4: Physical-context transitions. Reuse of overlapping allocations waits for outstanding work to drain.

streams, and work-queue state [18]. To avoid context creation at dispatch time, VITAMIN-E maintains a bounded hierarchy of pre-created allocations. Leaves represent minimum allocation units, while internal nodes cover unions of their descendants; complementary contexts expose remaining SMs without enumerating arbitrary subsets. Overlapping allocations cannot hold leases simultaneously. A pCtx transitions among idle, leased, draining, and reserved states (Figure 4). Reclamation closes admission to conflicting allocations and waits for outstanding kernels to drain, without preemption, before transferring leases. Reservation protects recurring latency-critical work, while lookahead begins draining conflicting contexts before a known predecessor completes. The hierarchy tracks conflicting leases without rebuilding contexts at each scheduling decision. Appendix C illustrates reservation and lookahead.

Compatibility and enforcement. A policy selects a ready operation and a candidate allocation. Eligibility requires that the backend can execute the unchanged descriptor with valid code and memory access, supported launch attributes, and sufficient resources for its launch constraints. Immediately before submission, the dispatcher rechecks eligibility, readiness, the admission gate, and lease non-overlap. A rejected

proposal leaves the operation unchanged for another decision. Specialized CUDA APIs require explicit modeling of context-local state and synchronization (Appendix D).

Prototype. The prototype comprises approximately 6K lines of C++ and 1K lines of CUDA. An LD_PRELOAD library intercepts launches, memory operations, stream/event operations, and synchronization. A daemon manages scheduling decisions and context leases, while clients retain CUDA handles and submit operations through local Green Contexts. Clients and the daemon exchange compact records through pre-allocated shared-memory rings; client threads submit work and poll completion events in batches. CUDA Graph replay is disabled: supporting it requires preserving captured node descriptors and dependencies during rebinding.

4 Evaluation

Experimental setup. End-to-end colocation experiments use H20 GPUs; determinism, policy, and overhead experiments use an H200. The decoding test uses SGLang with Llama 3.1 and deterministic execution enabled [11, 20, 34]. CUDA Graph replay is disabled in all compared decoding and trace-driven serving configurations. We evaluate bitwise equality in operator tests and fixed-state decoding replays, and assess performance separately in colocated-training and mixed-workload experiments. Colocated-training baselines include temporal sharing, MPS, MIG, Orion, and Salus [25, 30]. Mixed-workload comparisons additionally include our reproduction of LithOS’s published atomization logic, which may omit optimizations from the original system [5]. For mixed workloads, we report both foreground inference latency and background training throughput to distinguish sharing gains from co-tenant starvation.

Numerical preservation. Exclusive fixed-structure runs are bitwise identical, establishing the premise before adaptation. In 1,000 matched-input trials of a reduction and Split-K GEMM, reshaping repartitions work while VITAMIN-E varies only physical allocation under matched interference. Reshaping produces drift; VITAMIN-E preserves every output bit (Figure 2). For the token-level test, we fix one prompt and replay the state for its first generated token 100 times with temperature-zero greedy decoding. We compare LM-head logits, softmax probabilities, and the selected token. Reshaping changes the selected token in five replays; VITAMIN-E preserves bitwise-identical logits and probabilities with no flips (Figure 1). The fixed-state replays demonstrate token sensitivity to reshaping at this state.

Sharing performance. The eight training pairs span dense training, LoRA, QLoRA with CPU offload, and distributed Qwen workloads [8, 14, 26, 29]. VITAMIN-E achieves the highest aggregate normalized throughput in all pairs (Figure 5),

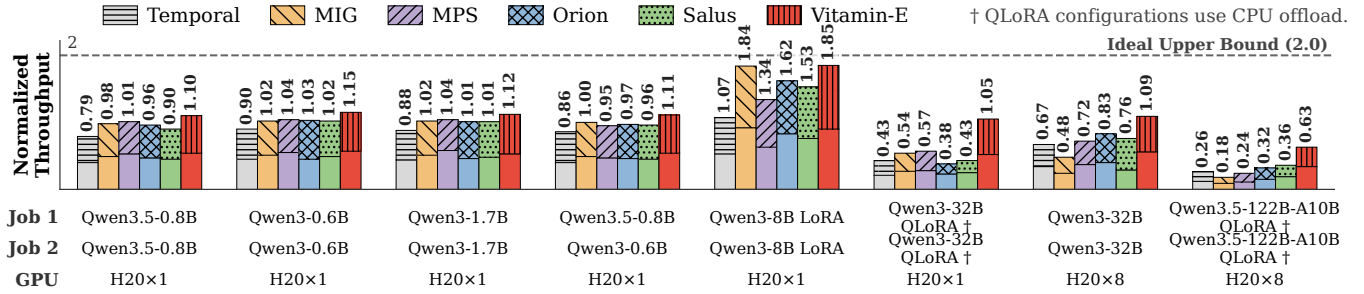


Figure 5: Aggregate normalized throughput for eight colocated-training pairs. Each job’s throughput is normalized to its exclusive execution; the two normalized throughputs are summed.

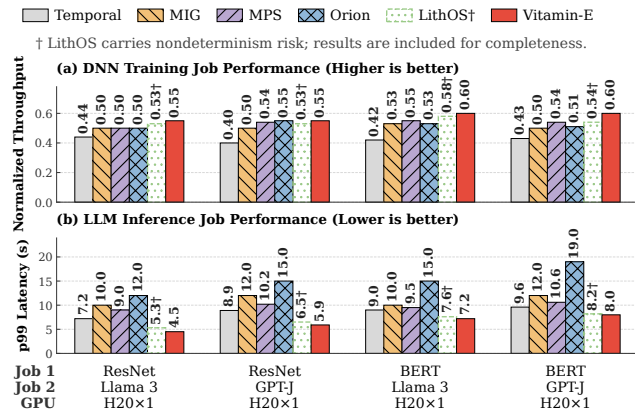


Figure 6: Mixed-workload performance: background-training throughput (top) and inference p99 latency (bottom). LithOS is our reproduction of its published atomization logic.

with geometric-mean gains of 1.38–1.63 $\times$ over the five baselines. Gains over the strongest baseline are 1.08–1.11 $\times$ for dense single-GPU pairs and reach 1.84 $\times$ when transfer or communication exposes more slack. This pattern is consistent with reclaiming idle capacity across launches; preserving structure does not itself make an individual kernel faster.

In four pairings of ResNet or BERT training with Llama 3 or GPT-J inference (Figure 6) [9, 13, 27], VITAMIN-E has the lowest inference p99 latency and highest or tied-highest background throughput. Geometric-mean results across workloads show 8.1–58.4% lower p99 latency and 1.05–1.36 $\times$ the background throughput of the compared systems. Reservation and selective reclamation are designed to reduce inference waiting while retaining background progress.

Policy flexibility. On Azure, LongBench, and BurstGPT traces [2, 24, 28], TPOT-FIRST reduces time-per-output-token (TPOT) SLO violations by 21.2%, 23.4%, and 46.1% relative to the throughput-oriented policy. The policies use offline profiles of recurring launches across the pool’s allocation

sizes. The throughput-oriented policy first favors concurrency, then assigns additional SMs where they yield greater normalized progress. TPOT-FIRST reserves decode capacity and prepares a wider allocation when a predicted deadline is at risk. Both policies preserve the same descriptors and dependencies. On BurstGPT, time-to-first-token (TTFT) violations rise from 4.33% to 6.37%: reserving decode capacity can delay prefills. This demonstrates mechanism–policy separation: scheduling policies can pursue different performance objectives without violating the parallel-structure invariant.

Runtime overhead. For BF16 GEMM with $M = 512$, $K = 8192$, and $N = 28672$, p95 end-to-end latency is 404 μ s natively, 2,069 μ s with on-demand context creation, 432 μ s with a flat pre-created pool, and 407 μ s with the hierarchical pool. This comparison isolates pooling choices within the Green Context backend. The 0.7% overhead is specific to this GEMM; shorter kernels may incur higher relative overhead. At the largest pool configuration, the hierarchy uses 1,163 MiB of GPU memory and 318 MiB of host RSS, compared with 2,533 MiB and 552 MiB for MPS-Pool, which provisions a separate MPS context for each supported resource share. Figure 10 details latency and pool memory costs.

5 Discussion and Conclusion

Our numerical evaluation covers operators and one fixed decoding state, while sharing performance is evaluated separately under the parallel-structure invariant. The benefits of late binding depend on exploitable idle capacity across launches, with greater opportunities in transfer- and communication-heavy workloads. Dynamic MoE expert routing requires targeted validation, while fused or persistent kernels offer fewer opportunities to adjust SM allocations between launches.

VITAMIN-E improves throughput and tail latency through fine-grained GPU spatial sharing while preserving the submitted computation. By decoupling logical execution from physical allocation, VITAMIN-E enables flexible scheduling

under the parallel-structure invariant. These results demonstrate that GPU spatial elasticity need not sacrifice bitwise determinism for fixed-structure deterministic workloads.

Acknowledgments

This work was partially funded by the National Natural Science Foundation of China (No. 62502510), and the CAS Talent Introduction Project, Institute of Software, Chinese Academy of Sciences (No. E5YD230101).

References

- [1] Willow Ahrens, James Demmel, and Hong Diep Nguyen. 2020. Algorithms for Efficient Reproducible Floating Point Summation. *ACM Trans. Math. Softw.* 46, 3, Article 22 (July 2020), 49 pages. doi:10.1145/3389360
- [2] Yushi Bai, Xin Lv, Jiajie Zhang, Hongchang Lyu, Jiankai Tang, Zhidian Huang, Zhengxiao Du, Xiao Liu, Aohan Zeng, Lei Hou, Yuxiao Dong, Jie Tang, and Juanzi Li. 2024. LongBench: A Bilingual, Multitask Benchmark for Long Context Understanding. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Lun-Wei Ku, Andre Martins, and Vivek Srikumar (Eds.). Association for Computational Linguistics, Bangkok, Thailand, 3119–3137. doi:10.18653/v1/2024.acl-long.172
- [3] Sylvie Boldo, Jacques-Henri Jourdan, Xavier Leroy, and Guillaume Melquiond. 2015. Verified Compilation of Floating-Point Computations. *Journal of Automated Reasoning* 54 (02 2015), 135–163. doi:10.1007/s10817-014-9317-x
- [4] Guoyu Chen, Srinivasan Subramaniyan, and Xiaorui Wang. 2024. Latency-Guaranteed Co-Location of Inference and Training for Reducing Data Center Expenses. In *2024 IEEE 44th International Conference on Distributed Computing Systems (ICDCS)*. 473–484. doi:10.1109/ICDCS60910.2024.00051
- [5] Patrick H. Coppock, Brian Zhang, Eliot H. Solomon, Vasilis Kypriotis, Leon Yang, Bikash Sharma, Dan Schatzberg, Todd C. Mowry, and Dimitrios Skarlatos. 2025. LithOS: An Operating System for Efficient Machine Learning on GPUs. In *Proceedings of the ACM SIGOPS 31st Symposium on Operating Systems Principles* (Lotte Hotel World, Seoul, Republic of Korea) (*SOSP '25*). Association for Computing Machinery, New York, NY, USA, 1–17. doi:10.1145/3731569.3764818
- [6] NVIDIA Corporation. 2025. *CUDA Multi-Process Service (MPS) Overview*. <https://docs.nvidia.com/deploy/mps/>
- [7] NVIDIA Corporation. 2025. *NVIDIA Multi-Instance GPU (MIG) User Guide*. <https://docs.nvidia.com/datacenter/tesla/mig-user-guide/>
- [8] Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. QLoRA: efficient finetuning of quantized LLMs. In *Proceedings of the 37th International Conference on Neural Information Processing Systems* (New Orleans, LA, USA) (*NIPS '23*). Curran Associates Inc., Red Hook, NY, USA, Article 441, 28 pages.
- [9] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, Jill Burstein, Christy Doran, and Thamar Solorio (Eds.). Association for Computational Linguistics, Minneapolis, Minnesota, 4171–4186. doi:10.18653/v1/N19-1423
- [10] David Goldberg. 1991. What every computer scientist should know about floating-point arithmetic. *ACM Comput. Surv.* 23, 1 (March 1991), 5–48. doi:10.1145/103162.103163
- [11] Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurelien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Roziere, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, Danny Wyatt, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Francisco Guzmán, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Govind Thattai, Graeme Nail, Gregoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel Kloumann, Ishan Misra, Ivan Evtimov, Jack Zhang, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jency Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Karthik Prasad, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, Khalid El-Arini, Krithika Iyer, Kshitiz Malik, Kuenley Chiu, Kunal Bhalla, Kushal Lakhotia, Lauren Rantala-Yearry, Laurens van der Maaten, Lawrence Chen, Liang Tan, Liz Jenkins, Louis Martin, Lovish Madaan, Lubo Malo, Lukas Blecher, Lukas Landzaat, Luke de Oliveira, Madeline Muzzi, Mahesh Pasupuleti, Mannat Singh, Manohar Paluri, Marcin Kardas, Maria Tsimpoukelli, Mathew Oldham, Mathieu Rita, Maya Pavlova, Melanie Kambadur, Mike Lewis, Min Si, Mitesh Kumar Singh, Mona Hassan, Naman Goyal, Narjes Torabi, Nikolay Bashlykov, Nikolay Bogoychev, Niladri Chatterji, Ning Zhang, Olivier Duchenne, Onur Çelebi, Patrick Alrassy, Pengchuan Zhang, Pengwei Li, Petar Vasic, Peter Weng, Prajjwal Bhargava, Pratik Dubal, Praveen Krishnan, Punit Singh Koura, Puxin Xu, Qing He, Qingxiao Dong, Ragavan Srinivasan, Raj Ganapathy, Ramon Calderer, Ricardo Silveira Cabral, Robert Stojnic, Roberta Raileanu, Rohan Maheswari, Rohit Girdhar, Rohit Patel, Romain Sauvestre, Ronnie Polidoro, Roshan Sumbaly, Ross Taylor, Ruan Silva, Rui Hou, Rui Wang, Saghar Hosseini, Sahana Chennabasappa, Sanjay Singh, Sean Bell, Seohyun Sonja Kim, Sergey Edunov, Shaoaliang Nie, Sharan Narang, Sharath R-parthy, Sheng Shen, Shengye Wan, Shruti Bhosale, Shun Zhang, Simon Vandenhende, Soumya Batra, Spencer Whitman, Sten Sootla, Stephane Collet, Suchin Gururangan, Sydney Borodinsky, Tamar Herman, Tara Fowler, Tarek Sheasha, Thomas Georgiou, Thomas Scialom, Tobias Speckbacher, Todor Mihaylov, Tong Xiao, Ujjwal Karn, Vedanuj Goswami, Vibhor Gupta, Vignesh Ramanathan, Viktor Kerkez, Vincent Gouget, Virginie Do, Vish Voleti, Vitor Albiero, Vladan Petrovic, Weiwei Chu, Wenhan Xiong, Wenyan Fu, Whitney Meers, Xavier Martinet, Xiaodong Wang, Xiaofang Wang, Xiaoqing Ellen Tan, Xide Xia, Xinfeng Xie, Xuchao Jia, Xuewei Wang, Yaelle Goldschlag, Yashesh Gaur, Yasmine Babaei, Yi Wen, Yiwen Song, Yuchen Zhang, Yue Li, Yuning Mao, Zacharie Delpierre Coudert, Zheng Yan, Zhengxing Chen, Zoe Papakipos, Aaditya Singh, Aayushi Srivastava, Abha Jain, Adam Kelsey, Adam Shajnfeld, Adithya Gangidi, Adolfo Victoria, Ahuva Goldstand, Ajay Menon, Ajay Sharma, Alex Boesenberg, Alexei Baevski, Allie Feinstein, Amanda Kallet, Amit Sangani, Amos Teo, Anam Yunus, Andrei Lupu, Andres Alvarado, Andrew Senior, Andrew Gu, Andrew Gu, Andrew Poulton, Andrew Ryan, Ankit Ramchandani, Annie Dong, Annie Franco, Anuj Goyal, Aparajita Saraf, Arkabandhu Chowdhury, Ashley Gabriel, Ashwin Bharambe, Assaf Eisenman, Azadeh Yazdan,

Beau James, Ben Maurer, Benjamin Leonhardi, Bernie Huang, Beth Loyd, Beto De Paola, Bhargavi Paranjape, Bing Liu, Bo Wu, Boyu Ni, Braden Hancock, Bram Wasti, Brandon Spence, Brani Stojkovic, Brian Gamido, Britt Montalvo, Carl Parker, Carly Burton, Catalina Mejia, Ce Liu, Changhan Wang, Changkyu Kim, Chao Zhou, Chester Hu, Ching-Hsiang Chu, Chris Cai, Chris Tindal, Christoph Feichtenhofer, Cynthia Gao, Damon Civin, Dana Beaty, Daniel Kreymer, Daniel Li, David Adkins, David Xu, Davide Testuggine, Delia David, Devi Parikh, Diana Liskovich, Didem Foss, Dingkan Wang, Duc Le, Dustin Holland, Edward Dowling, Eissa Jamil, Elaine Montgomery, Eleonora Presani, Emily Hahn, Emily Wood, Eric-Tuan Le, Erik Brinkman, Esteban Arcaute, Evan Dunbar, Evan Smothers, Fei Sun, Felix Kreuk, Feng Tian, Filippas Kokkinos, Firat Ozgenel, Francesco Caggioni, Frank Kanayet, Frank Seide, Gabriela Medina Florez, Gabriella Schwarz, Gada Badeer, Georgia Swee, Gil Halpern, Grant Herman, Grigory Sizov, Guangyi, Zhang, Guna Lakshminarayanan, Hakan Inan, Hamid Shojanazeri, Han Zou, Hannah Wang, Hanwen Zha, Haroun Habeeb, Harrison Rudolph, Helen Suk, Henry Aspegren, Hunter Goldman, Hongyuan Zhan, Ibrahim Damlaj, Igor Molybog, Igor Tufanov, Ilias Leontiadis, Irina-Elena Veliche, Itai Gat, Jake Weissman, James Geboski, James Kohli, Janice Lam, Japhet Asher, Jean-Baptiste Gaya, Jeff Marcus, Jeff Tang, Jennifer Chan, Jenny Zhen, Jeremy Reizenstein, Jeremy Teboul, Jessica Zhong, Jian Jin, Jingyi Yang, Joe Cummings, Jon Carvill, Jon Shepard, Jonathan McPhie, Jonathan Torres, Josh Ginsburg, Junjie Wang, Kai Wu, Kam Hou U, Karan Saxena, Kartikay Khandelwal, Katayoun Zand, Kathy Matosich, Kaushik Veeraraghavan, Kelly Michelen, Keqian Li, Kiran Jagadeesh, Kun Huang, Kunal Chawla, Kyle Huang, Lailin Chen, Lakshya Garg, Lavender A, Leandro Silva, Lee Bell, Lei Zhang, Liangpeng Guo, Licheng Yu, Liron Moshkovich, Luca Wehrstedt, Madian Khabza, Manav Avalani, Manish Bhatt, Martynas Mankus, Matan Hasson, Matthew Lennie, Matthias Reso, Maxim Groshev, Maxim Naumov, Maya Lathi, Meghan Keneally, Miao Liu, Michael L. Seltzer, Michal Valko, Michelle Restrepo, Mihir Patel, Mik Vyatskov, Mikayel Samvelyan, Mike Clark, Mike Macey, Mike Wang, Miquel Jubert Hermoso, Mo Metanat, Mohammad Rastegari, Munish Bansal, Nandhini Santhanam, Natascha Parks, Natasha White, Navyata Bawa, Nayan Singhal, Nick Egebo, Nicolas Usunier, Nikhil Mehta, Nikolay Pavlovich Laptev, Ning Dong, Norman Cheng, Oleg Chernoguz, Olivia Hart, Omkar Salpekar, Ozlem Kalinli, Parkin Kent, Parth Parekh, Paul Saab, Pavan Balaji, Pedro Rittner, Philip Bontrager, Pierre Roux, Piotr Dollar, Polina Zvyagina, Prashant Ratanchandani, Pritish Yuvraj, Qian Liang, Rachad Alao, Rachel Rodriguez, Rafi Ayub, Raghotham Murthy, Raghu Nayani, Rahul Mitra, Rangaprabhu Parthasarathy, Raymond Li, Rebekkah Hogan, Robin Battey, Rocky Wang, Russ Howes, Ruty Rinott, Sachin Mehta, Sachin Siby, Sai Jayesh Bondu, Samyak Datta, Sara Chugh, Sara Hunt, Sargun Dhillon, Sasha Sidorov, Satadru Pan, Saurabh Mahajan, Saurabh Verma, Seiji Yamamoto, Sharadh Ramaswamy, Shaun Lindsay, Shaun Lindsay, Sheng Feng, Shenghao Lin, Shengxin Cindy Zha, Shishir Patil, Shiva Shankar, Shuqiang Zhang, Shuqiang Zhang, Sinong Wang, Sneha Agarwal, Soji Sajuyigbe, Soumith Chintala, Stephanie Max, Stephen Chen, Steve Kehoe, Steve Satterfield, Sudarshan Govindaprasad, Sumit Gupta, Summer Deng, Sungmin Cho, Sunny Virk, Suraj Subramanian, Sy Choudhury, Sydney Goldman, Tal Remez, Tamar Glaser, Tamara Best, Thilo Koehler, Thomas Robinson, Tianhe Li, Tianjun Zhang, Tim Matthews, Timothy Chou, Tzook Shaked, Varun Vontimitta, Victoria Ajayi, Victoria Montanez, Vijai Mohan, Vinay Satish Kumar, Vishal Mangla, Vlad Ionescu, Vlad Poenaru, Vlad Tiberiu Mihailescu, Vladimir Ivanov, Wei Li, Wenchen Wang, Wenwen Jiang, Wes Bouaziz, Will Constable, Xiaocheng Tang, Xiaoqian Wu, Xiaolan Wang, Xilun Wu, Xinbo Gao, Yaniv Kleinman, Yanjun Chen, Ye Hu, Ye Jia, Ye Qi, Yenda Li, Yilin Zhang, Ying Zhang, Yossi Adi, Youngjin Nam, Yu, Wang, Yu

- Zhao, Yuchen Hao, Yundi Qian, Yunlu Li, Yuzi He, Zach Rait, Zachary DeVito, Zef Rosnbrick, Zhaoduo Wen, Zhenyu Yang, Zhiwei Zhao, and Zhiyu Ma. 2024. The Llama 3 Herd of Models. arXiv:2407.21783 [cs.AI] <https://arxiv.org/abs/2407.21783>
- [12] Horace He and Thinking Machines Lab. 2025. Defeating Nondeterminism in LLM Inference. *Thinking Machines Lab: Connectionism* (2025). doi:10.64434/tml.20250910 <https://thinkingmachines.ai/blog/defeating-nondeterminism-in-llm-inference/>.
- [13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep Residual Learning for Image Recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 770–778. doi:10.1109/CVPR.2016.90
- [14] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-Rank Adaptation of Large Language Models. In *International Conference on Learning Representations (ICLR)*.
- [15] Wenhao Huang, Zhaolin Duan, Laiping Zhao, Yuhao Zhang, Yanjie Wang, Yiming Li, Yihan Wang, Yichi Chen, Zhihang Tang, Kang Chen, Deze Zeng, Wenxin Li, and Keqiu Li. 2026. μ Share: Non-Intrusive Kernel Co-Locating on NVIDIA GPUs. In *Proceedings of the 2026 IEEE International Symposium on High-Performance Computer Architecture (HPCA)*. IEEE, 1–14. doi:10.1109/HPCA68181.2026.11408561
- [16] Myeongjae Jeon, Shivaram Venkataraman, Amar Phanishayee, Junjie Qian, Wencong Xiao, and Fan Yang. 2019. Analysis of large-scale multi-tenant GPU clusters for DNN training workloads. In *Proceedings of the 2019 USENIX Conference on Usenix Annual Technical Conference* (Renton, WA, USA) (*USENIX ATC '19*). USENIX Association, USA, 947–960.
- [17] Zhengqing Liu, Musa Unal, Matthew J. Parkinson, and Marios Kogias. 2025. DORADD: Deterministic Parallel Execution in the Era of Microsecond-Scale Computing. In *Proceedings of the 30th ACM SIGPLAN Annual Symposium on Principles and Practice of Parallel Programming (PPoPP '25)*. Association for Computing Machinery, New York, NY, USA, 282–296. doi:10.1145/3710848.3710872
- [18] NVIDIA. 2025. CUDA driver API – Green Contexts. https://docs.nvidia.com/cuda/cuda-driver-api/group__CUDA__GREEN__CONTEXTS.html Accessed: 2025-12-10.
- [19] NVIDIA Corporation. [n. d.]. CUDA Runtime API Reference: Event Management. https://docs.nvidia.com/cuda/cuda-runtime-api/cuda_runtime_api/group__CUDART__EVENT.html. Accessed: 2026-10-09.
- [20] SGLang Contributors. 2026. Deterministic Inference. SGLang Documentation. https://docs.sglang.io/docs/advanced_features/deterministic_inference Accessed: 2026-08-01.
- [21] Sanjif Shanmugavelu, Mathieu Taillefumier, Christopher Culver, Oscar Hernandez, Mark Coletti, and Ada Sedova. 2025. Impacts of floating-point non-associativity on reproducibility for HPC and deep learning applications. In *Proceedings of the SC '24 Workshops of the International Conference on High Performance Computing, Network, Storage, and Analysis* (Atlanta, GA, USA) (*SC-W '24*). IEEE Press, 170–179. doi:10.1109/SCW63240.2024.00028
- [22] Dharma Shukla, Muthian Sivathanu, Srinidhi Viswanatha, Bhargav Gulavani, Rimma Nehme, Amey Agrawal, Chen Chen, Nipun Kwatra, Ramachandran Ramjee, Pankaj Sharma, Atul Katiyar, Vipul Modi, Vaibhav Sharma, Abhishek Singh, Shreshth Singhal, Kaustubh Welankar, Lu Xun, Ravi Anupindi, Karthik Elangovan, Hasibur Rehman, Zhou Lin, Rahul Seetharaman, Cheng Xu, Eddie Ailijiang, Suresh Krishnappa, and Mark Russinovich. 2022. Singularity: Planet-Scale, Preemptive and Elastic Scheduling of AI Workloads. arXiv preprint. <https://www.microsoft.com/en-us/research/publication/singularity-planet-scale-preemptive-and-elastic-scheduling-of-ai-workloads/>
- [23] Yifan Song, Guoyin Wang, Sujian Li, and Bill Yuchen Lin. 2025. The Good, The Bad, and The Greedy: Evaluation of LLMs Should Not

- Ignore Non-Determinism. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*. Association for Computational Linguistics, Albuquerque, New Mexico, 4195–4206. doi:10.18653/v1/2025.naacl-long.211
- [24] Jovan Stojkovic, Chaojie Zhang, Íñigo Goiri, Josep Torrellas, and Esha Choukse. 2025. DynamoLLM: Designing LLM Inference Clusters for Performance and Energy Efficiency. In *2025 IEEE International Symposium on High Performance Computer Architecture (HPCA)*. 1348–1362. doi:10.1109/HPCA61900.2025.00102
- [25] Foteini Strati, Xianzhe Ma, and Ana Klimovic. 2024. Orion: Interference-aware, Fine-grained GPU Sharing for ML Applications. In *Proceedings of the Nineteenth European Conference on Computer Systems (Athens, Greece) (EuroSys '24)*. Association for Computing Machinery, New York, NY, USA, 1075–1092. doi:10.1145/3627703.3629578
- [26] Qwen Team. 2026. Qwen3.5: Accelerating Productivity with Native Multimodal Agents. <https://qwen.ai/blog?id=qwen3.5>
- [27] Ben Wang and Aran Komatsuzaki. 2021. GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model. <https://github.com/kingoflolz/mesh-transformer-jax>. EleutherAI.
- [28] Yuxin Wang, Yuhan Chen, Zeyu Li, Xueze Kang, Yuchu Fang, Yeju Zhou, Yang Zheng, Zhenheng Tang, Xin He, Rui Guo, Xin Wang, Qiang Wang, Amelie Chi Zhou, and Xiaowen Chu. 2025. BurstGPT: A Real-World Workload Dataset to Optimize LLM Serving Systems. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.2 (Toronto ON, Canada) (KDD '25)*. Association for Computing Machinery, New York, NY, USA, 5831–5841. doi:10.1145/3711896.3737413
- [29] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. 2025. Qwen3 Technical Report. arXiv:2505.09388 [cs.CL] <https://arxiv.org/abs/2505.09388>
- [30] Peifeng Yu and Mosharaf Chowdhury. 2020. Salus: Fine-Grained GPU Sharing Primitives for Deep Learning Applications. In *Proceedings of the 3rd MLSys Conference (MLSys)*. Austin, TX, USA.
- [31] Anwar Hossain Zahid, Ignacio Laguna, and Wei Le. 2025. Testing GPU Numerics: Finding Numerical Differences Between NVIDIA and AMD GPUs. In *Proceedings of the SC '24 Workshops of the International Conference on High Performance Computing, Network, Storage, and Analysis (Atlanta, GA, USA) (SC-W '24)*. IEEE Press, 547–557. doi:10.1109/SCW63240.2024.00077
- [32] Ziyang Zhang, Xinheng Ding, Jiayi Yuan, Rixin Liu, Huizi Mao, Jiarong Xing, and Zirui Liu. 2026. Deterministic Inference across Tensor Parallel Sizes That Eliminates Training–Inference Mismatch. In *Proceedings of the 43rd International Conference on Machine Learning (ICML)*. <https://icml.cc/virtual/2026/poster/66224>
- [33] Qi Zhao, Zhengyi Qiu, and Guoliang Jin. 2019. Semantics-aware scheduling policies for synchronization determinism. In *Proceedings of the 24th Symposium on Principles and Practice of Parallel Programming (PPoPP '19)*. Association for Computing Machinery, New York, NY, USA, 242–256. doi:10.1145/3293883.3295731
- [34] Lianmin Zheng, Liangsheng Yin, Zhiqiang Xie, Chuyue Sun, Jeff Huang, Cody Hao Yu, Shiyi Cao, Christos Kozyrakis, Ion Stoica, Joseph E. Gonzalez, Clark Barrett, and Ying Sheng. 2024. SGLang: efficient execution of structured language model programs. In *Proceedings of the 38th International Conference on Neural Information Processing Systems (Vancouver, BC, Canada) (NIPS '24)*. Curran Associates Inc., Red Hook, NY, USA, Article 2000, 27 pages.
- [35] Jianlong Zhong and Bingsheng He. 2014. Kernelet: High-Throughput GPU Kernel Executions with Dynamic Slicing and Scheduling. *IEEE Transactions on Parallel and Distributed Systems* 25, 6 (2014), 1522–1532. doi:10.1109/TPDS.2013.257

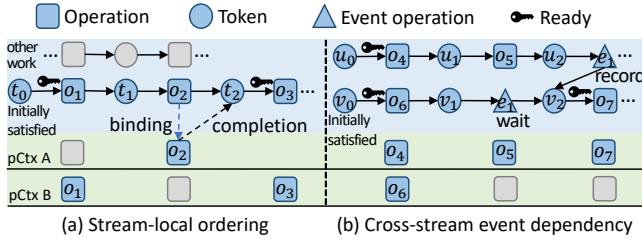


Figure 7: Logical dependency tracking across physical contexts: (a) stream-local ordering and (b) cross-stream event dependencies.

A Formal Execution Contract

Represent an execution by

$$E = (\mathcal{E}, X, \{B_k\}, \{C_k\}, H, P),$$

where $\mathcal{E}$ is the hardware/software environment, X the inputs and initial device-visible state, B_k the kernel binary, C_k the submitted parallel structure, H the application-required happens-before order, and $P = \{P_k\}$ the physical allocations. Let E' differ from E only in its admissible allocation sequence $P' = \{P'_k\}$, and let $F(E)$ denote the application-visible output. A workload is *fixed-structure deterministic* if $F(E) = F(E')$ bitwise for all such admissible allocation sequences. With X and $\mathcal{E}$ fixed, VITAMIN-E must preserve $\{B_k\}$, $\{C_k\}$, and H when selecting P' .

Immutable descriptors preserve kernel binaries, parallel structures, and captured argument values, while logical predecessor tokens preserve application-required dependencies. The application remains responsible for memory lifetimes and synchronization of referenced data. Physical scheduling may serialize operations incomparable in H , but this does not alter the required dependency relation. Thus, for fixed-structure deterministic workloads, admissible physical rebinding preserves bitwise-identical outputs.

B Dependency Semantics

For operations O , the virtual context represents application-required ordering as

$$H = (O, E_{\text{stream}} \cup E_{\text{event}} \cup E_{\text{default}}), o_i \prec_H o_j \iff o_i \rightsquigarrow_H o_j.$$

This graph is acyclic for a well-formed execution. Each operation snapshots its predecessor tokens at submission and becomes ready only after all are satisfied. Figure 7 illustrates how these tokens preserve stream-local ordering across physical contexts (a) and event-induced dependencies between streams (b).

Event re-recording. Suppose stream s_1 records event e after operation a , and s_2 then issues a wait on e before operation b . The wait captures generation $e^{(1)}$, whose token represents completion through a . If s_1 subsequently records e after operation c , this creates $e^{(2)}$; it does not replace the predecessor of

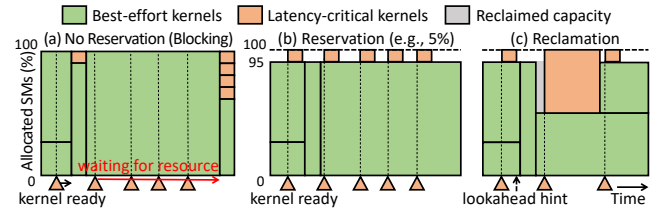


Figure 8: Reservation avoids resource waiting for recurring latency-critical work; lookahead starts reclaiming a wider allocation before anticipated work becomes ready.

the already issued wait. Thus b depends on a , without acquiring a new dependency on c merely because the event handle was reused [19]. Event synchronization observes the selected generation; stream synchronization waits on the captured stream tail, and device synchronization waits on the relevant stream tails. Queries inspect the logical completion state.

Default streams. The logical dependency graph must retain the application’s CUDA stream mode. For legacy default-stream behavior, a default-stream operation depends on earlier work in the relevant blocking streams; subsequent blocking-stream work depends on the default-stream tail. Nonblocking streams are exempt from these implicit edges. A per-thread default stream behaves as a regular thread-local stream, without legacy cross-stream ordering, except when interacting with an explicitly used legacy default stream. These are application-level edges: choosing nonblocking backend streams must neither remove required edges nor introduce legacy ordering where the application did not request it. For example, launches a in a blocking stream, b in the legacy default stream, and c back in the blocking stream require $a \prec_H b \prec_H c$, regardless of their physical contexts.

C Reservation and Lookahead

Reservation and lookahead address different sources of waiting (Figure 8). A standing reservation gives recurring latency-critical work an allocation without first draining admitted background kernels. Lookahead uses a known predecessor or an advisory to begin reclaiming a wider allocation before an operation becomes ready. If reclamation finishes late, dispatch selects another eligible allocation or waits for the requested capacity.

D Rebinding Scope

Eligibility requirements. Table 1 summarizes the representation and rebinding requirements of each operation category. Cooperative and residency-sensitive kernels require additional validation against the original launch’s execution constraints.

Table 1: Logical representation and rebinding requirements by operation category.

Category	Representation and requirement
Kernel launch	Immutable entry, arguments, geometry, shared memory, and attributes; candidate allocation must satisfy the original launch requirements.
Copy / memset	Immutable transfer or fill parameters; preserve memory validity, ordering, and API synchronization semantics.
Stream / event	Logical stream tails and event generations preserve application ordering across backend handles.
Sync / query	Observe logical completion state at the required stream, event, or device scope.
CUDA Graph replay	Disabled in the prototype; support requires preserving captured descriptors and dependencies.
Other CUDA APIs	Require explicit modeling of state, lifetimes, and synchronization before rebinding.

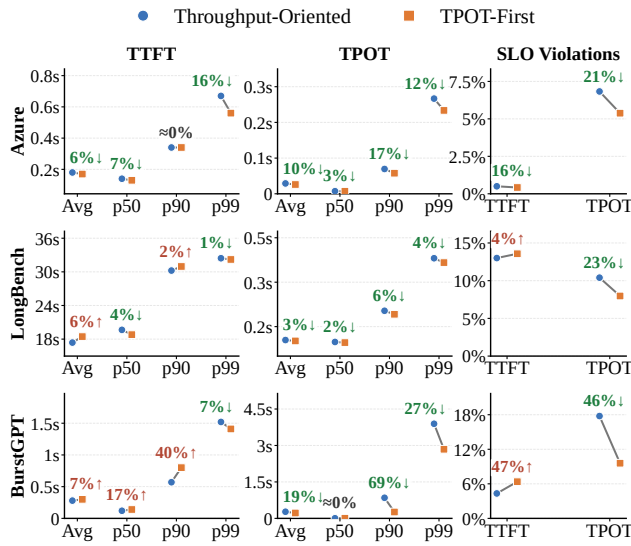
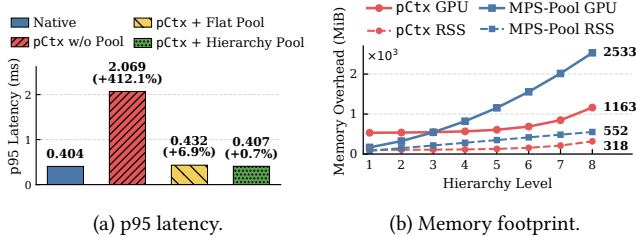


Figure 9: Policy comparison on Azure, LongBench, and BurstGPT. TPOT-First reduces TPOT SLO violations, with a workload-dependent TTFT tradeoff.

Figure 10: Runtime overhead on H200. (a) End-to-end latency for BF16 GEMM, $(M, K, N) = (512, 8192, 28672)$. (b) GPU memory and host RSS of the hierarchical pool and MPS-Pool.

API-specific handling. Rebinding APIs with context-local state or specialized synchronization requires explicit modeling of their state, lifetimes, and ordering semantics.

E Additional Evaluation

Numerical protocol. The operator tests use matched inputs and fixed-structure reference executions; reshaping changes grid partitioning or work decomposition, while VITAMIN-E changes only physical allocation under matched interference. For decoding, each trial replays the same first-generated-token state of one fixed prompt, comparing LM-head logits, softmax probabilities, and token identity.

Policy tradeoffs. Figure 9 details TTFT, TPOT, and SLO violations under the two scheduling policies on Azure, LongBench, and BurstGPT. The BurstGPT TTFT increase reflects contention for capacity between reserved decode work and prefills.

Context-management overhead. Figure 10(a) compares native execution, on-demand context creation, and two pre-created pool designs for the same BF16 GEMM on H200. The reported end-to-end latency includes interception, queue communication, policy execution, submission, and completion polling; these components are not timed independently. Figure 10(b) reports GPU and host memory consumption across pool configurations, including the MPS-Pool baseline.