DRAVA: An Event-Driven Runtime for Scientific Pipelines with Agentic Configuration Optimization

Ahmedur Rahman Shovon, Romain Pereira, Xingfu Wu, Brice Videau, Swann Perarnau
Argonne National Laboratory, Lemont, Illinois, USA
Emails: {ashovon, rpereira, xingfu.wu, bvideau, swann}@anl.gov

I. INTRODUCTION

Scientific facilities in X-ray science (XRS) increasingly rely on always-on instruments that generate data faster than it can be transported, reduced, and analyzed [1]. The Advanced Photon Source (APS) Upgrade at Argonne National Laboratory increases X-ray beam brightness by up to $500\times$, pushing data-transfer requirements to tens of Gbit/s and reconstruction workloads toward petascale compute [2], with future instruments expected to drive data rates orders of magnitude higher [3]. These workflows must coordinate ingestion, preprocessing, inference, analysis, and feedback while experiments run; intermediate results can steer adaptive sampling [4], imposing strict end-to-end latency and throughput requirements.

Custom FPGA/ASIC-based pipelines provide low latency [5] but are hard to reconfigure as detectors and algorithms change [2]; software coordination over general-purpose resources is a more flexible alternative through composable stages [6]. Yet current software remains fragmented: frameworks such as PvaPy support EPICS pvAccess streaming [7], but pipeline-wide control over scheduling, batching, placement, transport, and observability is often external or manually tuned [8], [9], making static heuristics brittle under changing data rates and resources.

We present DRAVA, an event-driven runtime for scientific streaming pipelines that offers a lightweight real-time serverless programming model. Developers express applications as reusable event-driven stages connected by data streams, while DRAVA manages batching, scheduling, data movement, resource placement, and observability, separating scientific logic from system concerns.

II. IMPLEMENTATION

A DRAVA workflow is a directed acyclic graph in which nodes represent *stages* and edges represent *transport routes*. Each stage declares an ingress route, a stateless callback that performs domain-specific computation, an egress route, and runtime parameters that control batching, scheduling, and thread resources. The application defines only the scientific transformation; the runtime parameters are supplied by external YAML configuration, so the same callback can be evaluated under different execution policies without modification.

The execution engine is built on XKRT [10], a tasking runtime for heterogeneous architectures. DRAVA forks a team of threads, where Thread 0 pulls messages from the transport layer and the remaining threads form a work-stealing compute pool. A modular transport layer supports NATS JetStream [11], a durable publish-subscribe model, and a low-latency Unix domain socket, enabling edge-to-HPC data movement. Microbatching is a first-class runtime policy: pending messages are flushed on a batch threshold, an end-of-stream event, or a receive timeout, amortizing per-callback overhead while bounding tail latency.

DRAVA has low-overhead observability with lock-free counters that record per-stage throughput, latency, byte counts, and callback timing, exposing how runtime decisions affect performance and energy. This creates a direct path from measurement to adaptation: an external optimizer consumes emitted metrics and proposes new configurations. We use an agentic autotuning workflow built on ytopt [12], [13], which applies surrogate-based Bayesian optimization [14] to search the joint configuration space (inference batch size, per-stage thread counts, callback batch thresholds, publisher rate) under a bounded budget, converging toward high-performing configurations without exhaustive search.

III. EVALUATION

We evaluate DRAVA on a single JLSE node [15] (dual-socket AMD EPYC 7532, NVIDIA A100 40 GB) using two representative XRS inference workloads: PtychoNN [16] for ptychographic phase retrieval and TomoGAN [17], [18] for low-dose tomographic denoising, with pretrained models and synthetic frame replay [19].

A synthetic dataflow benchmark isolates the runtime message path and establishes a ceiling of about 31K frames/s once callback batches are large; adding an empty CUDA launch raises callback time but not end-to-end throughput. Against a direct PvaPy baseline [7] on the same PtychoNN inference stream, DRAVA remains loss-free at publisher rates where PvaPy drops frames and reaches $2.36\times$ higher throughput on unpaced input. Studying manually selected two-stage configurations shows the fastest individual stage does not predict the best pipeline: a GPU-only heuristic yields a configuration roughly $2\times$ slower end-to-end than the best balanced one. Using DRAVA's observability, we attribute the batch-size trade-off to two runtime mechanisms—callback dispatch overhead at small batches and stage-boundary output transport at large batches—and obtain up to a $2.44\times$ end-to-end throughput improvement from batching-aware tuning. The agentic search [12] explores only 0.52% of a 12,288-point configuration grid (64 evaluations) yet finds a configu-

ration $2.24\times$ faster than the best manually selected one. On TomoGAN, integrating the sampled GPU power trace shows throughput and energy do not scale identically with batch size: the most energy-efficient point is 3.37 J/frame, the highest-throughput point reaches 38 fps, and runtime overhead stays below 5%. Reporting joules per frame alongside throughput exposes explicit operating points for facilities that balance turnaround time with accelerator energy use under fixed power budgets [20], [21].

IV. Conclusion

This poster presents DRAVA, an event-driven runtime that sustains multi-stage scientific streaming pipelines while separating application logic from runtime policy through external configuration. Lock-free observability enables per-stage and end-to-end analysis with negligible overhead, and agentic Bayesian optimization efficiently navigates the combinatorial configuration space, reaching $2.24\times$ speedup after exploring 0.52% of the space and $2.36\times$ higher throughput than a state-of-the-art baseline. DRAVA complements FaaS-style scientific computing frameworks [22] with a runtime layer that measures throughput, latency, and energy. Future work will extend DRAVA to distributed multi-node streaming and energy-aware online adaptation, and evaluate it on emerging data-centric architectures such as UpDown [23] for irregular, communication-intensive workflows at PB/s-scale data rates.

References

[1] A. V. Babu, T. Zhou, S. Kandel, T. Bicer, Z. Liu, W. Judge, D. J. Ching, Y. Jiang, S. Veseli, S. Henke *et al.*, "Deep learning at the edge enables real-time streaming ptychographic imaging," *Nature Communications*, vol. 14, no. 1, p. 7059, 2023.

[2] Argonne National Laboratory, "Advanced photon source strategic plan 2025–2029," https://www.aps.anl.gov/The-Advanced-Photon-Source-Strategic-Plan, 2025, accessed: 2026-05-01.

[3] H. R. Stueber, A. Y. Pan, T. Chattopadhyay, S. W. Allen, M. W. Bautz, K. Donlon, C. E. Grant, S. Herrmann, B. LaMarr, A. Malonis *et al.*, "Fast, low noise ccd systems for future strategic x-ray missions," in *UV, X-Ray, and Gamma-Ray Space Instrumentation for Astronomy XXIV*, vol. 13625. SPIE, 2025, pp. 501–516.

[4] J.-W. Buurlage, F. Marone, D. M. Pelt, W. J. Palenstijn, M. Stampanoni, K. J. Batenburg, and C. M. Schlepütz, "Real-time reconstruction and visualisation towards dynamic feedback control during time-resolved tomography experiments at tomcat," *Scientific reports*, vol. 9, no. 1, p. 18379, 2019.

[5] D. Windisch, O. Knodel, G. Juckeland, U. Hampel, and A. Bieberle, "Fpga-based real-time data acquisition for ultrafast x-ray computed tomography," *IEEE Transactions on Nuclear Science*, vol. 68, no. 12, pp. 2779–2786, 2021.

[6] D. Gürsoy, F. De Carlo, X. Xiao, and C. Jacobsen, "Tomopy: a framework for the analysis of synchrotron tomographic data," *Synchrotron Radiation*, vol. 21, no. 5, pp. 1188–1193, 2014.

[7] S. Veseli, J. Hammonds, S. Henke, H. Parraga, B. Frosik, and N. Schwarz, "Pvapy streaming framework for real-time data processing," *Synchrotron Radiation*, vol. 32, no. 3, 2025.

[8] Y. Kim, S. Park, S. Perarnau, and A. Raj, "Charon: An end-to-end infrastructure for connecting ai@edge to hpc," in *2025 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*, 2025, pp. 1112–1119.

[9] R. Vescovi, R. Chard, N. D. Saint, B. Blaiszik, J. Pruyne, T. Bicer, A. Lavens, Z. Liu, M. E. Papka, S. Narayanan, N. Schwarz, K. Chard, and I. T. Foster, "Linking scientific instruments and computation: Patterns, technologies, and experiences," *Patterns*, vol. 3, no. 10, 2022.

[10] R. Pereira, S. Perarnau, P.-E. Polet, B. Videau, T. Applencourt, and T. Gautier, "XKRT: a Runtime System for Macro-Dataflow Programming on Multi-Devices Architectures," 2026.

[11] NATS.io, "Nats concepts: Jetstream," 2021, accessed: 2026-04-25. [Online]. Available: https://docs.nats.io/nats-concepts/jetstream

[12] X. Wu, P. Balaprakash, M. Kruse, J. Koo, B. Videau, P. Hovland, V. Taylor, B. Geltz, S. Jana, and M. Hall, "ytopt: Autotuning scientific applications for energy efficiency at large scales," *Concurrency and Computation: Practice and Experience*, vol. 37, no. 1, p. e8322, 2025.

[13] X. Wu, P. Paramasivam, and V. Taylor, "Autotuning apache tvm-based scientific applications using bayesian optimization," in *Proceedings of the SC'23 Workshops of the International Conference on High Performance Computing, Network, Storage, and Analysis*, 2023, pp. 29–35.

[14] P. Balaprakash, J. Dongarra, T. Gamblin, M. Hall, J. K. Hollingsworth, B. Norris, and R. Vuduc, "Autotuning in high-performance computing applications," *Proceedings of the IEEE*, vol. 106, no. 11, pp. 2068–2083, 2018.

[15] Argonne National Laboratory, "Joint Laboratory for System Evaluation (JLSE)," https://www.jlse.anl.gov, 2026, accessed: 2026.

[16] M. J. Cherukara, T. Zhou, Y. Nashed, P. Enfedaque, A. Hexemer, R. J. Harder, and M. V. Holt, "Ai-enabled high-resolution scanning coherent diffraction imaging," *Applied Physics Letters*, vol. 117, no. 4, 2020.

[17] Z. Liu, T. Bicer, R. Kettimuthu, D. Gursoy, F. De Carlo, and I. Foster, "Tomogan: low-dose synchrotron x-ray tomography with generative adversarial networks: discussion," *Journal of the Optical Society of America A*, vol. 37, no. 3, pp. 422–434, 2020.

[18] Z. Liu, T. Bicer, R. Kettimuthu, and I. Foster, "Deep learning accelerated light source experiments," in *2019 IEEE/ACM Third Workshop on Deep Learning on Supercomputers (DLS)*, 2019, pp. 20–28.

[19] M. J. Cherukara, "PtychoNN_data," https://huggingface.co/datasets/mcherukara/PtychoNN_data, 2024, accessed: 2026-04-27.

[20] E. Choukse, B. Warrier, S. Heath, L. Belmont, A. Zhao, H. A. Khan, B. Harry, M. Kappel, R. J. Hewett, K. Datta *et al.*, "Power stabilization for ai training datacenters," *arXiv preprint arXiv:2508.14318*, 2025.

[21] W. Olin-Ammentorp, X. Wu, and A. A. Chien, "Modeling performance of data collection systems for high-energy physics," *APL Machine Learning*, vol. 2, no. 4, 2024.

[22] R. Chard, Y. Babuji, Z. Li, T. Skluzacek, A. Woodard, B. Blaiszik, I. Foster, and K. Chard, "Funcx: A federated function serving fabric for science," in *Proceedings of the 29th International symposium on high-performance parallel and distributed computing*, 2020, pp. 65–76.

[23] A. A. Chien, C. Colley, J. Ding, A. Fell, D. F. Gleich, H. Hoffmann, M. Jeje, R. Khandelwal, Y. Li, J. M. Monsalve-Diaz, M. Nourian, A. Rajasukumar, J. Su, T. Su, Y. Wang, W. Wang, R. Xu, and T. Zhang, "Updown: A supercomputer co-designed for scalable graph processing," *IEEE Transactions on Parallel and Distributed Systems*, pp. 1–19, 2026.