Decision Locality: A Substrate for AI Inference and Continual Learning

Jerry Felix
Brain-CA Technologies
Sarasota, FL, USA
jfelix@brain-ca.com

Abstract

The dominant AI infrastructure paradigm, built around floating-point matrix engines and dense data movement, is increasingly mismatched to the workloads now driving real deployments: streaming inference, on-device adaptation, and continual learning under tight power and latency budgets. We argue that this mismatch is structural rather than a question of compiler maturity or memory bandwidth. We propose *Decision Locality* as an alternative organizing principle for AI infrastructure: each decision, the state required to produce it, and the learning update that refines future decisions should be co-located in the smallest local hardware unit that can express them. We instantiate this principle with three concrete artifacts: a hardware-native binary learning primitive whose inference behavior is consistent with Bayesian belief updating using only local state and bit operations; a learner-ensemble fabric that composes these primitives into a streaming classifier with no back-propagation; and an FPGA realization that synthesizes the core unit to 48 LUTs and 8 registers at 96 MHz on a commercial accelerator card. We discuss the infrastructure-level questions this substrate raises, from scheduling and programming-model questions to per-bit energy accounting, and argue that it is best positioned not as a replacement for dense matrix engines but as a complementary substrate for the long tail of inference and continual-learning workloads that increasingly drive total infrastructure cost.

CCS Concepts: • **Hardware** → **Reconfigurable logic and FPGAs**; • **Computer systems organization** → **Other architectures**; • **Computing methodologies** → *Machine learning*.

Keywords: AI infrastructure, decision locality, Brain-CA Estimator, Cincinnati Algorithm, bit-serial learning, continual learning, edge AI, von Neumann bottleneck, FPGA

ACM Reference Format:
Jerry Felix. 2026. Decision Locality: A Substrate for AI Inference and Continual Learning. In *Proceedings of The 3rd Workshop on Hot Topics in System Infrastructure (HotInfra'26), co-located with ISCA'26 (HotInfra '26).* ACM, New York, NY, USA, 4 pages.

The 3rd Workshop on Hot Topics in System Infrastructure (HotInfra'26), co-located with ISCA'26, June 28, 2026, Raleigh, NC, USA
2026

1 Introduction

For many deployed AI services, the operational cost of serving inference and supporting on-device adaptation already rivals the cost of training [12, 13]. Workloads where this bites hardest, such as streaming sensor classification, always-on perception, and continual personalization on edge devices, share a common shape: each decision is small, frequent, and latency-constrained. The substrate, however, has barely changed. Production systems still rely on dense floating-point matrix engines [9], hierarchical memory systems engineered around tensor reuse, and ahead-of-time-compiled execution graphs.

The workloads now driving deployment are mismatched to that substrate in a structural rather than a quantitative way. Continuous on-device adaptation does not amortize across batches. Streaming classification has limited use for cached weight reuse. The decisions an inference engine produces are often a few bits, while the machinery that produces them moves megabytes. Quantization, sparsity, and compiler advances have narrowed the gap, but they have not changed the fundamental shape of the substrate.

This paper proposes *Decision Locality* as an alternative organizing principle for the inference and continual-learning layers of the AI stack, and reports an architecture and hardware realization that takes the principle to its limit: each learning unit is a small register whose on-cycle update is a bit comparison and a conditional masked XOR.

2 The Decision Locality Principle

Decision Locality is the proposition that an AI infrastructure designed for inference and continual learning should co-locate three things: the decision itself, the state needed to produce that decision, and the learning update that refines future decisions.

In current substrates these three are spread apart. The decision lives at the output of a tensor accelerator. The state lives in a parameter cache or off chip. The learning update, when it happens at all, requires a round trip through a training framework, often on different hardware.

Decision Locality argues that this separation is a design choice, not a necessity. If the smallest unit of decision is a small bit-comparison cell, and the smallest unit of state is that cell's local register, then a substrate that co-locates the cell, its register, and its update logic can produce decisions,

update state, and refine behavior in one physical location, in a constant number of clock cycles.

The principle is most useful where it is most consequential: at the inference and adaptation layers, where decisions are small, frequent, latency-sensitive, and energy-constrained. We do not propose a replacement for the dense floating-point engines used to train foundation models. We propose a different substrate for the workloads that consume those models.

3 A Hardware-Native Learning Primitive

We instantiate Decision Locality with the *Brain-CA Estimator* [4]: a hyperparameter-free binary learning device whose state is a single positive integer s held in a b-bit register, representing its current probability estimate. With a sentinel bit and $m = \lfloor \log_2 s \rfloor$ payload bits $b_1 \dots b_m$, the state encodes the probability $p(s) = (2(s - 2^m) + 1)/2^{m+1}$. On each binary observation $x \in \{0, 1\}$, the unit draws m random bits and scans them MSB-to-LSB against the payload; at the first position j where the random bit differs from b_j, the prediction is b_j. The unit then holds if the prediction agrees with x, nudges $s \to s \pm 1$ if it disagrees, or extends $s \to 2s + x$ if no mismatch occurs and the register is not yet full. We refer to this update procedure as the Cincinnati Algorithm.

Once the register fills, fall-through becomes a no-op and the dynamics form a finite birth-death chain on $N = 2^{b-1}$ tracking states. By detailed balance between adjacent states, the stationary distribution under i.i.d. Bernoulli(θ) input is exactly Bin($N-1, \theta$) [8], so the unit's expected estimate behaves like a Bayesian-smoothed frequency estimate that interpolates between Jeffreys-style and Bayes-Laplace-style shrinkage, with no tuned decay parameter. This is what we mean by inference behavior consistent with Bayesian belief updating.

Composability is provided by a learner ensemble [7]: Estimators organized as a streaming tree take wider binary inputs and produce classification decisions. The inter-unit protocol is bit-serial, with short fixed-width binary queries and responses whose widths are chosen at design time and held constant across the fabric.

The Brain-CA Learning Fabric is not back-propagated [5, 6]. Each Estimator updates its own state from its own observations; the tree structure grows or prunes nodes through structural search against ensemble classification accuracy, with no gradient-based credit assignment or continuous-valued parameter updates propagated across units. Every local Estimator update (one step of the Cincinnati Algorithm) is a constant-time operation on constant local state; ensemble-level growth, pruning, and aggregation are separate infrastructure concerns.

4 Hardware Realization

The core Brain-CA Estimator synthesizes on a commercial FPGA (Xilinx Alveo U200) to 48 lookup tables and 8 registers,

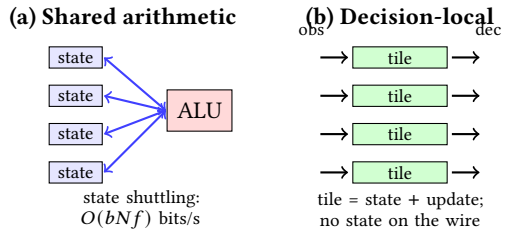

Figure 1. Decision Locality eliminates per-update state shuttling. In a fabric of N learners with b-bit state at update rate f, the shared-arithmetic path moves $O(bNf)$ bits of state across the on-die interconnect per second; the decision-local path keeps learner state at the update site, with only observation and decision streams plus local random and control bits on the wires.

operating at 96 MHz [8]. The integer-only datapath admits a fully bit-serial implementation: at each cycle, one bit of observation enters, the state register advances, and at most one bit of decision is produced. The footprint is small enough that thousands of Estimators fit on a single mid-range FPGA, and the bit-serial inter-unit protocol requires routing only a handful of wires per pair of communicating units.

The relevant comparison against fixed-point EWMA baselines is architectural, not merely numeric. On same-flow synthesis the Estimator uses 48 LUTs, between a shift-only EWMA path at 13–15 LUTs (which restricts α to powers of two and still requires an adder per update) and a multiplier-based EWMA at 55–71 LUTs; on switching-Bernoulli tracking all three reach comparable post-switch probability-estimate MSE around 0.002–0.003. The Estimator's 96 MHz clock also trails shift-only EWMA's 235 MHz on the same flow; the substrate case rests on fabric-scale properties, not per-unit speed. The differentiator is that EWMA in either variant requires update arithmetic and a tuned decay parameter, while the Estimator requires neither an on-cycle adder nor a tuned decay parameter: no ALU, just a bit comparison and a conditional masked XOR.

The energy and bandwidth profile that motivates the work is best characterized at the per-decision level: an Estimator emitting an inference bit performs a fixed, small number of bit-level operations on a fixed, small register file. The local energy of one Estimator's decision is independent of ensemble size, in structural contrast to tensor accelerators where decision energy is amortized across a batch; system-level energy additionally depends on routing, aggregation, and query traffic.

Per-unit gate count is not the right metric at fabric scale (Fig. 1). Arithmetic-based learning primitives like EWMA force the architect into a familiar tension with the von Neumann bottleneck: either pay a per-learner area tax for local arithmetic, or pay an interconnect tax to shuttle state between learners and shared arithmetic units. As an

order-of-magnitude bound, at N=10^6 learners with b=8 and f=1 GHz under the worst-case assumption that every learner updates every cycle, the shared-arithmetic path moves $2bNf \approx 1.6\times10^{16}$ bits per second, on the order of 2 PB/s of on-die state traffic before any application data is moved. The decision-local path pays neither cost: comparison and update happen at the storage cell, no state crosses a wire, and the only per-update I/O is the $\leq b-1$ random bits, supplied by a small grid of distributed LFSRs as local-mesh traffic. The same bandwidth profile simplifies chiplet partitioning: a decision-local fabric can replicate across package boundaries with zero cross-chiplet state traffic during local learning, in structural contrast to gradient-synchronized training where each step requires parameter or gradient exchange that scales with model size. The on-cycle datapath is itself a single comparator and a masked XOR; a priority encoder precomputes the carry boundary off cycle from the cached state and prediction bit, so no propagation length scales with the state width.

5 Related Substrates

Several substrate alternatives push against FP-dense compute: quantization [2] and sparsity reduce precision or skip operations within the matrix-multiply core; neuromorphic substrates [3] commit to event-driven analog or spiking accumulation; in-memory computation [14] challenges the von Neumann separation; stochastic computing [1] and hyperdimensional computing [10] replace floating-point with alternative representations. Each targets a specific cost axis without changing the substrate's organizing principle.

Decision Locality is closer in spirit to neuromorphic computing than to quantization, but commits to a different abstraction: discrete integer state evolving under deterministic update rules, with no spike-timing dependence and no analog accumulation. Bit-serial FPGA-mapped binary networks [15] share the integer-and-bit-operation discipline but assume an externally trained model; the substrate proposed here closes the loop with local, hyperparameter-free learning.

6 Open Infrastructure Questions

A substrate organized around Decision Locality raises questions that contemporary AI infrastructure does not naturally answer.

Scheduling. With a single bit-serial decision as the unit of work rather than a tensor operation, the scheduling problem changes shape. There is no natural batch dimension; cross-Estimator pipelining is the relevant performance dimension, and the scheduling unit is closer to a packet than a kernel launch.

Programming model. The tensor has been the lingua franca of AI frameworks for a decade, and Decision Locality has no tensor at any level. Candidate abstractions include bit-level dataflow graphs, behavioral specifications of input-output bit relations, and emergent-protocol approaches in which the inter-Estimator query-and-response language is itself learned.

Energy accounting. Power profiling tools assume a baseline of dense FP compute. A substrate whose smallest unit is a single-bit register and comparator requires energy models that measure per-bit-decision rather than per-FLOP. Workload-level energy then depends on how decisions compose, and the relevant tooling does not yet exist.

Heterogeneity. Decision-local fabrics are unlikely to displace dense matrix engines for training large foundation models. They are, rather, a candidate substrate for the long tail of inference and continual-learning workloads. The right infrastructure question is not "Decision Locality or tensor accelerator" but how a mixed fleet should be designed and how workloads should be routed across it.

Continual learning at scale. Continual learning on conventional dense substrates [11] is bottlenecked by some combination of gradient updates, replay, synchronization, and periodic retraining; in many cases the per-adaptation cost scales with model size or with the size of the adapted submodel. In a decision-local fabric the per-unit update cost is constant. System-level adaptation cost still depends on which units update, how labeled signal reaches them, and ensemble aggregation, but a dominant term, per-parameter update arithmetic, is eliminated. The empirical question this paper invites is whether that change is decisive on workloads where periodic batched retraining is currently the binding cost driver.

7 Discussion

The case for Decision Locality is strongest where the workload is inference-heavy, latency-sensitive, energy-constrained, and increasingly continuous. We do not claim Decision Locality solves the infrastructure problem, only that it is a coherent alternative organizing principle at the inference and adaptation layers whose open questions are worth early discussion.

8 Conclusion

We have proposed Decision Locality as an organizing principle for inference and continual learning, instantiated it with a hardware-native binary primitive, and reported an FPGA realization at 48 LUTs and 96 MHz. The immediate question is whether decision-local fabrics can deliver measurable energy and latency advantages on streaming inference and continual-adaptation workloads that conventional batched tensor execution serves poorly.

References

[1] Armin Alaghi and John P. Hayes. 2013. Survey of Stochastic Computing. *ACM Transactions on Embedded Computing Systems* 12, 2s (2013), 1–19.

[2] Dan Alistarh, Demjan Grubic, Jerry Li, Ryota Tomioka, and Milan Vojnovic. 2017. QSGD: Communication-Efficient SGD via Gradient Quantization and Encoding. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., Red Hook, NY, USA, 1709–1720.

[3] Mike Davies, Narayan Srinivasa, Tsung-Han Lin, Gautham Chinya, Yongqiang Cao, Sri Harsha Choday, Georgios Dimou, Prasad Joshi, Nabil Imam, Shweta Jain, et al. 2018. Loihi: A Neuromorphic Manycore Processor with On-Chip Learning. *IEEE Micro* 38, 1 (2018), 82–99.

[4] Gerard E. Felix. 2023. Artificial Intelligence Based on Cellular Automata. U.S. Patent 11,847,386 B1. https://patents.google.com/patent/US11847386B1/

[5] Gerard E. Felix. 2024. Artificial Intelligence Based on Cellular Automata. U.S. Patent 12,050,846 B1. https://patents.google.com/patent/US12050846B1/

[6] Gerard E. Felix. 2026. Artificial Intelligence Based on Cellular Automata. U.S. Patent 12,572,717 B2. https://patents.google.com/patent/US12572717B2/

[7] Jerry Felix. 2026. Convergence-Based Learning Fabrics: From Elemental Devices to Communication-Minimal Distributed AI Architectures. Workshop on Cognitive Architectures (CogArch), co-located with ISCA 2026, Raleigh, NC, USA. To appear.

[8] Jerry Felix. 2026. A Hardware-Native Bit-Serial Learner with Exact Statistical Structure. Workshop on Machine Learning for Computer Architecture and Systems (MLArchSys), co-located with ISCA 2026, Raleigh, NC, USA. To appear.

[9] Norman P. Jouppi, Cliff Young, Nishant Patil, David Patterson, et al. 2017. In-Datacenter Performance Analysis of a Tensor Processing Unit. In *Proceedings of the 44th Annual International Symposium on Computer Architecture (ISCA)*. Association for Computing Machinery, New York, NY, USA, 1–12.

[10] Pentti Kanerva. 2009. Hyperdimensional Computing: An Introduction to Computing in Distributed Representation with High-Dimensional Random Vectors. *Cognitive Computation* 1, 2 (2009), 139–159.

[11] German I. Parisi, Ronald Kemker, Jose L. Part, Christopher Kanan, and Stefan Wermter. 2019. Continual Lifelong Learning with Neural Networks: A Review. *Neural Networks* 113 (2019), 54–71.

[12] David Patterson, Joseph Gonzalez, Quoc Le, Chen Liang, Lluis-Miquel Munguia, Daniel Rothchild, David So, Maud Texier, and Jeff Dean. 2021. Carbon Emissions and Large Neural Network Training. arXiv:2104.10350 [cs.LG]

[13] Vijay Janapa Reddi, Christine Cheng, David Kanter, Peter Mattson, Guenther Schmuelling, Carole-Jean Wu, Brian Anderson, Maximilien Breughe, Mark Charlebois, William Chou, et al. 2020. MLPerf Inference Benchmark. In *Proceedings of the 47th Annual International Symposium on Computer Architecture (ISCA)*. IEEE Press, Piscataway, NJ, USA, 446–459.

[14] Abu Sebastian, Manuel Le Gallo, Riduan Khaddam-Aljameh, and Evangelos Eleftheriou. 2020. Memory Devices and Applications for In-Memory Computing. *Nature Nanotechnology* 15 (2020), 529–544.

[15] Yaman Umuroglu, Nicholas J. Fraser, Giulio Gambardella, Michaela Blott, Philip Leong, Magnus Jahre, and Kees Vissers. 2017. FINN: A Framework for Fast, Scalable Binarized Neural Network Inference. In *Proceedings of the 2017 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays (FPGA)*. Association for Computing Machinery, New York, NY, USA, 65–74.