The Mismatch Tax

A Preprint

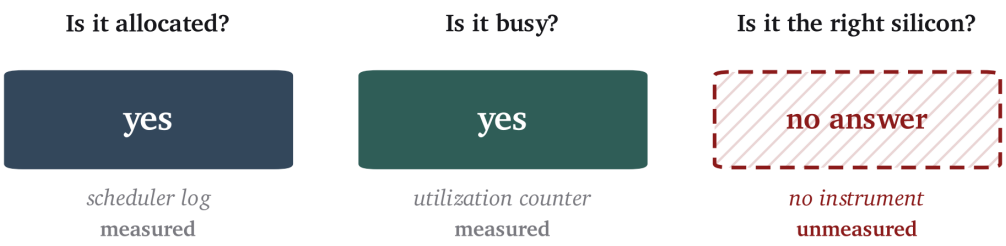
Sidney Scott
sidney@reckonresearch.com
reckonresearch.com | github.com/reckonresearch/mismatch-tax

August 2026

Abstract

Two years ago the Berkeley AI Research group identified a trend that was already happening: state-of-the-art results were coming out of compound systems, not monolithic models, and the orchestration layer above the models was becoming the place where competitive advantage lay. That thesis has held. This essay argues that it has an unnamed and unmeasured counterpart in the substrate. The compound compute substrate is defined as the execution surface of a single workload assigned to two or more architecturally distinct classes of silicon. There is a two-layered public accounting for AI compute waste: scheduling and chip utilization. Both have instruments. Allocation is logged. Peak FLOPs are published. There is no instrument to a third layer. This essay introduces the term mismatch tax to characterize the loss incurred when allocated, busy silicon performs operations outside its design point. The term names a mechanism, not a rate. The mechanism is recursive: each component added to a compound AI pipeline comes with its own optimal substrate so the substrate-layer loss compounds because the application-layer thesis is successful. A decade of trying to build the missing orchestration layer, from the Apache TVM lineage, through OctoML, Neural Magic, Run:ai and CentML, ended when the incumbents whose silicon those systems were designed to route around absorbed them. The essay is a cross between a position paper and a historical review. It concludes with four dated, falsifiable predictions, one of which requires no cooperation from any party holding a commercial position.

Keywords heterogeneous compute · compound AI systems · accelerator orchestration · compute governance · position paper



Schematic. Two of the three questions have an instrument; the third does not.

Code and data. An exploratory analysis of decode efficiency on the Reckon measured corpus is at github.com/reckonresearch/mismatch-tax. It was not pre-registered: its hypotheses file was committed with its intervals blank, four minutes before the first measured results were committed. The first result falsified one of the corpus's own microbench ceilings; see Defect 17 in the Reckon register. Section 7.4 reports the result.

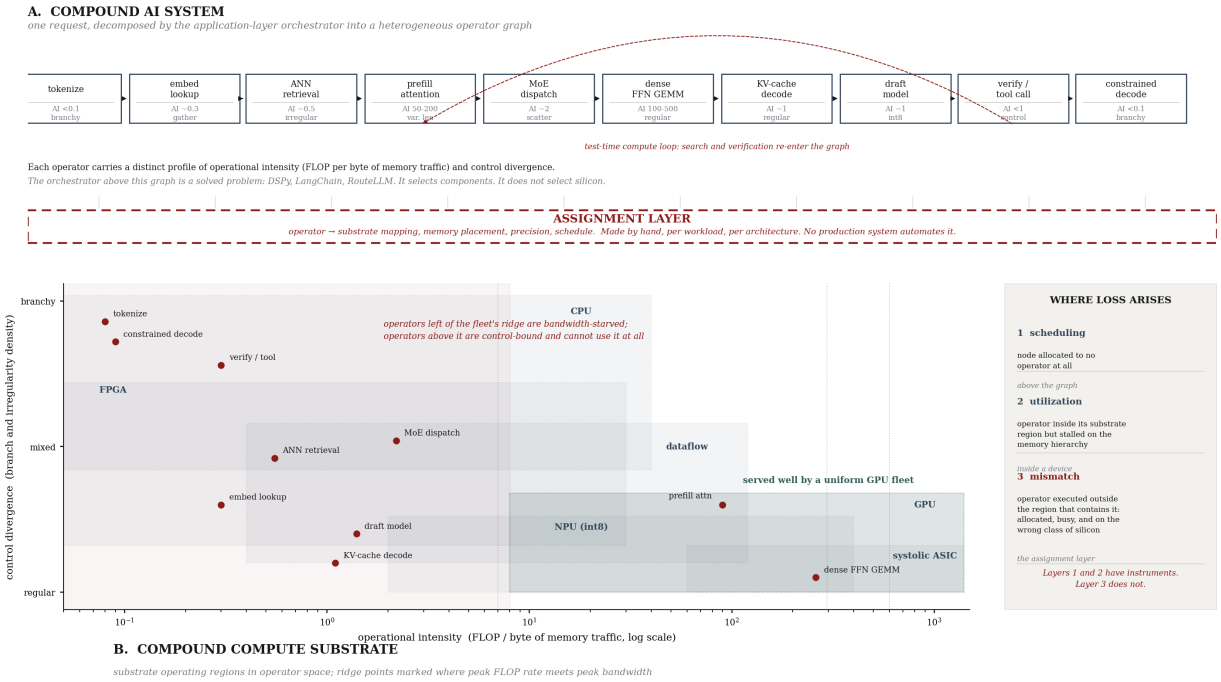


Figure 1: Schematic. The two strata in one view. (A) One request, decomposed by the application-layer orchestrator into an operator graph. Each operator carries a distinct profile of operational intensity and control divergence. That orchestrator is a solved problem, and it selects components rather than silicon. (Assignment layer) The mapping from operator to substrate, with memory placement, precision, and schedule. Made by hand, per workload and per architecture. (B) Substrate operating regions in operator space. A roofline has one axis and cannot carry this argument, because control divergence is what strands operators on throughput silicon irrespective of arithmetic intensity. A uniform GPU fleet serves the shaded band and leaves most of this pipeline outside it. The mismatch tax of Section 3 is an operator executed outside the region containing it. Intensity values are illustrative; the geometry is the claim, not the coordinates.

1 Introduction

History suggests that what gets measured shapes what gets fixed. In computing this is not merely an observation about management culture. It is an observation about instruments. A quantity with a cheap instrument attracts optimization effort, tooling, benchmarks, and eventually a research literature [Hooker, 2020]. A quantity without one attracts nothing, and its absence from the conversation is mistaken for evidence that it is small.

In February 2024, Zaharia and colleagues at Berkeley named a shift that was already happening: state-of-the-art AI results were no longer coming from monolithic models [Zaharia et al., 2024] but from compound systems with multiple interacting components. Retrieval, tool use, agentic loops, ensembles, routers, test-time compute scaling, speculative decoding. The thesis aged well. Every

frontier deployment of consequence since has been a compound system, and the orchestration layer above the models became where the value went.

The thesis of compound systems has a counterpart one stratum down, and this counterpart is invisible for instrumental rather than substantive reasons. The following is in part a position paper and in part a historical review. It introduces the concept of mismatch tax to describe the cost paid when allocated and busy silicon operates outside of its design point. The term describes a mechanism, not a rate. Its magnitude is not publicly measured, and Section 3 argues that this absence is structural, rather than incidental.

Two terms are introduced and they do different jobs. The first names the cost and is defined in Section 3. The second names the object, and is printed here because the rest of the argument depends on it.

A compound compute substrate is the execution surface of a single workload whose operations are assigned across two or more architecturally distinct classes of silicon, where that assignment is a design variable, not an artifact of what happened to be available (Figure 1). Every clause carries weight. Single workload instead of fleet: the parallel to the compound systems thesis is exact only at the level of a single job. A compound AI system is one task that spans multiple model calls, retrievers and tools, not a datacenter that just happens to have multiple models. The substrate counterpart is one inference or one control loop across several types of chip, with coordination on the order of microseconds.

Architecturally distinct classes need a functional test, not a taxonomy, or the term gets lost in vendor marketing categories. Two substrates are said to be different if an operation that maps well to one runs at a small fraction of achievable efficiency on the other. That separates a CPU from a systolic-array ASIC, keeps two generations of the same GPU family together, and allows the boundary to shift as workloads change, instead of fixing it to a product line.

The clause that prevents the definition from being circular is that assignment is a design variable and it is the one that is most easily got wrong. The definition does not require an orchestrator. Today, a person can make the assignment, at the cost described in Section 2.2. And mostly it is. There are already compound compute substrates in production. What does not exist is a system that makes the assignment automatically, which is a claim about tooling rather than about deployments. The taxonomy is derived from the definition. A fleet of one silicon class is not compound, but homogeneous. A mixed fleet where each job is completely in one class is heterogeneous procurement but not compound. Only a workload that spans classes is compound, and the mismatch tax defined in Section 3 is the cost of running a workload that wants a compound substrate on a uniform one instead.

This paper has a companion, and it is worth stating the division between them before either argument begins. The Chip Is Not The Product discusses compilation targets: why an architecture fails to get to market and what the software layer above silicon has to do with that failure. This paper is about deployment targets, about what it costs to have allocated busy silicon when operating outside its design point.

Same physics, seen at different times in the life of a workload. The two papers cite each other, and present a common counting rule (see Section 6.1). The argument is in five parts. Section 2 asks why the two strata developed separate orchestration cultures, and identifies the homogeneous fleet era as the substrate analog of the general purpose computing era. Section 3 defines the mismatch tax, decomposes public compute waste into three layers, and shows why the third has no instrument. Section 4 is about the decade spent trying to create the missing layer and what happened to those efforts. Section 6 claims that the tax is a compound tax, not a static tax, and that the engine of the

compound is the success of the application-layer thesis itself. Section 7 considers what would have to change, including for compute governance, which currently rests on an accounting assumption the heterogeneous-substrate literature has been undermining for a decade.

2 Separate Strata

> *Hardware is only economically viable if the lifetime of the use case lasts more than three years.*
> Jeff Dean, 2020

The application stratum and the substrate stratum came to orchestration on different schedules and for different reasons, and the gap between those schedules is most of what this essay is about (Figure 2).

At the application level, orchestration immediately became a research topic because the feedback loop was cheap. The execution of a routing policy on a set of model variants costs one benchmark run. Anyone with an API key can measure whether a cascade router matches frontier quality at lower cost. Within two years of the compound systems thesis there was a tooling ecosystem, a literature, and a set of headline numbers to argue about. At the substrate level the same loop costs a fleet. To evaluate a routing policy over chip families, the workload needs to be run on those chip families, at production scale, with power instrumented. The feedback signal is wall-clock and joules on physical hardware, not task accuracy on a held out set. That difference in measurement cost, more than any difference in the difficulty of the underlying optimization, is why one stratum has a research culture and the other has vendor documentation.

2.1 The Homogeneous Fleet Era

Hooker describes a general purpose computing era [Hooker, 2020; Moore, 1965] in which predictable increases in compute made hardware design risk-averse, because the benefit of moving to specialized silicon could be eclipsed by the next generation of general hardware. The substrate stratum had its own version of this, compressed into roughly a decade rather than four.

From the ImageNet moment through the early years of the frontier model era, the practical substrate choice for just about every organization building with AI was which NVIDIA part to rent (Figure 3). That was not irrational. The dominant operation, dense matrix multiplication, had the best silicon available from a single vendor, and the only mature software stack. The next generation reliably arrived before a heterogeneous strategy could pay back its integration cost. In such circumstances, substrate orchestration was a solution in search of a problem, and the market treated it as such.

That era ended for two reasons, neither of which had anything to do with changes in anyone's preferences. The first is the diversification of the workload, discussed in Section 5. The second is the emergence of substrate diversity on the supply side: custom accelerator programs at every hyperscaler and a cohort of dataflow and NPU architectures entering production. Operators did not change their minds; the physical landscape no longer supported the homogeneous fleet as the default.

2.2 Why there is no substrate orchestration literature

Hooker observes that hardware and machine learning research evolved in isolation partly because of cost: producing a next generation chip runs tens of millions [Feldman, 2019] of dollars and two to three years, which produces a closely guarded research culture, while machine learning publication volume

grew exponentially over the same period. The asymmetry at the orchestration layer is narrower but has the same shape, and it is worth itemizing because it explains an absence that would otherwise look like a judgment.

Access. Evaluating a substrate routing policy requires the substrates [Asanovic et al., 2006]. A graduate student can reproduce a model-layer routing result with an API key and a benchmark suite. The substrate-layer equivalent requires concurrent access to several architecturally distinct fleets, at sufficient scale that the measurement is not dominated by warm-up effects, with power instrumented at the rack. The organizations that can assemble that are few, and most of them are the vendors whose products would be compared.

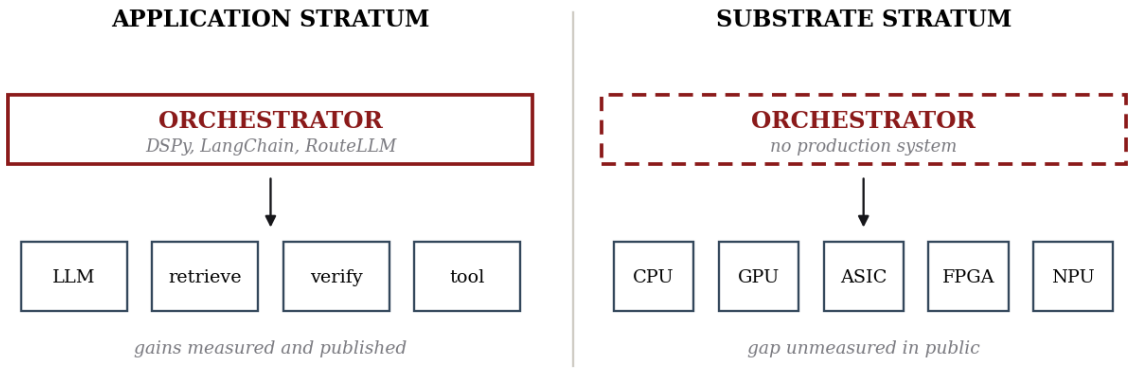


Figure 2: Schematic. The compound systems thesis and its substrate counterpart. At the application stratum, specialized components are composed by an orchestrator, and the gains from that composition are measured and published. At the substrate stratum, specialized substrates require composition by an orchestrator that no vendor-neutral party currently operates in production. The structural argument is isomorphic across the two strata. The evidentiary situation is not.

Terms. Access to pre-release and current-generation silicon frequently carries benchmarking restrictions. Comparative performance claims are a commercial matter for chip vendors in a way that comparative accuracy claims are not for model providers, and the contractual architecture reflects that. The effect is not that cross-vendor comparisons are forbidden. It is that publishing one is a decision with commercial consequences, taken by someone who has those consequences to weigh.

Benchmarks. The model layer inherited a benchmark culture, so a routing result has an obvious denominator. The substrate layer has vendor-run benchmark suites and a per-accelerator throughput literature [Mattson et al., 2020], but no standard mixed-workload basket executed at matched power [Reddi et al., 2020] across families. Without an agreed basket there is no comparison, and without comparisons there is nothing for a literature to accumulate around. None of these is a statement about difficulty. A field with cheap feedback loops develops a research culture and a field with expensive ones develops vendor documentation, and the substrate layer got the second.

The residue of the homogeneous fleet era is an accounting habit. Compute is still discussed as a scalar quantity, purchased in units of a single vendor's silicon, and its waste is still described with instruments built for a fleet in which every chip was the same kind of chip. Those instruments are the subject of the next section.



Figure 3: Photograph. A compute hall of the homogeneous fleet era. Every rack in view holds the same accelerator from the same vendor, cabled to the same fabric, and the uniformity is the point: a fleet of one part is tractable to schedule, to tune, and to reason about, which is why the industry built fleets this way for as long as one part remained adequate. The argument of this essay is not that such fleets were a mistake. It is that the workload distribution has moved and the uniformity that once simplified the problem now prices a share of the work at the wrong substrate. Compare the heterogeneous machine in Figure 7, built two decades earlier and retired in part because its heterogeneity cost more in programmer labor than it returned in performance. Photograph by the author.

3 The Mismatch Tax

> *It is hard to quantify the cost of being concerned.*
>
> Sara Hooker, 2020

Three measurements anchor the discussion, and each needs its scope stated, because all three are routinely quoted past what they support. GPT-4 training is widely described as having run at roughly a third of peak utilization across approximately 25,000 A100s. The figure originates in secondary industry analysis from 2023 and has never been confirmed by the laboratory concerned, so it belongs here as a widely held estimate rather than a measurement, and nothing below depends on it.

Meta's published Llama 3 work reports BF16 model FLOPs utilization of 38 to 43 percent on 16,384 H100s. This is a disclosed measurement on a frontier training run at scale, and it is the most reliable of the three.

On NERSC's Perlmutter [Li et al., 2023], maximum GPU memory utilization for a materials science code sat at roughly 10 percent at the tenth percentile and 21.5 percent at the seventy-fifth, with only the top few percent of jobs exceeding 35 percent. Two caveats belong with that figure, because it is often quoted without them. It is an open-science HPC workload rather than a frontier AI one, and the same measurement found GPU utilization itself high, above 80 percent at the tenth percentile, which

cuts against a naive reading of underuse. What it does show is the shape this essay is about: silicon whose arithmetic units are saturated while a resource it was provisioned for goes almost untouched, which is what a workload running on the wrong part looks like from the outside.

The most specific public account of AI compute waste comes from Anjney Midha, founder of AMP PBC and previously the general partner responsible for allocating a16z's GPU cluster, speaking on TBPN on 23 July 2026. His decomposition has two layers. At the fleet layer, a laboratory taking a long-term lease loses 30 to 40 percent of its FLOPs to scheduling alone, with nodes sitting unallocated. At the chip layer, model FLOPs utilization runs around 15 percent [Midha, 2026], because the chip waits on storage, memory, networking, or a handoff to another chip. Taken together he puts realized utilization at roughly 15 percent of every dollar leased, and characterizes the remaining 85 percent as a national security problem on the grounds that it is what keeps the United States off the frontier.

Two notes on that account, both of which strengthen it. Midha sells into the problem he describes, and that is worth stating rather than hiding: an interested party putting a specific number on the record under his own name is a stronger claim than an unattributed consensus, because the people it describes can dispute it. And his 15 percent does not contradict the 38 to 43 percent above, because the denominators differ: published model FLOPs utilization measures work inside an active run on allocated nodes, while his measures realized FLOPs per dollar of leased capacity, which folds allocation and utilization together. His two layers do not compose arithmetically to his headline figure, which is why they are reported here as two layers rather than as a product. The stack loses compute at both layers and the layers are counted separately.

Both layers are measured, and it is worth being precise about why. Scheduling loss is visible because allocation is logged: a scheduler knows which nodes were idle and for how long. Utilization loss is visible because vendors publish peak FLOPs and laboratories report realized ones, which makes model FLOPs utilization a ratio of two disclosed quantities [Meta AI, 2024]. In both cases the instrument predates the interest in the measurement.

There is a third layer, and it has no such ratio available. The scheduling loss is capacity that was never used. If a chip is waiting for its memory hierarchy, it is a loss of utilization. The third quantity is different in nature; the chip is allocated, it is busy, it is running at its own design point, and the design point is wrong for the operation in front of it. A dense matrix engine executing control logic at a few percent of CPU efficiency is fully occupied by every available instrument while producing almost no useful work. This essay calls that loss the mismatch tax (Figure 4).

One point of definition must be made clear, because the term is easy to widen until it means nothing at all. The mismatch tax is the loss from doing an operation on a substrate which is not well-matched to the operation's design point. That is not the loss from running an operation badly on a substrate that does fit. A bad batch size, a bad partitioning, or a bad process count on a single class of silicon produces real and often large recoverable headroom and that headroom is worth measuring but it is configuration loss rather than mismatch. The distinguishing test is whether the counterfactual requires a different type of chip. If the better configuration runs on the same part the quantity belongs to layer two. If the loss needs a different architecture for its recovery, it is in layer three. Each time the term below appears it is the second sense.

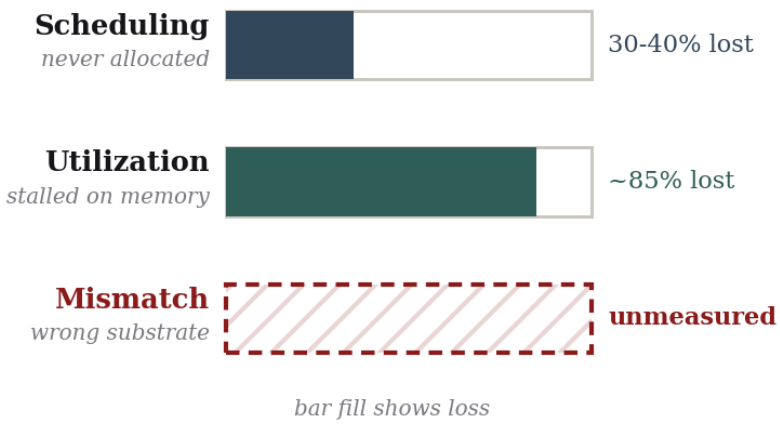


Figure 4: Measured, measured, unmeasured. Three layers of compute loss, two of which have instruments. Scheduling loss is visible because allocation is logged. Utilization loss is visible because peak and realized FLOPs are both disclosed. Mismatch has no analogous ratio, because isolating it requires executing one workload mix across architecturally distinct fleets at matched power. The unfilled third bar is the argument of this essay.

3.1 The term names a mechanism, not a rate

The word tax raises a question about a percentage, and the honest answer is that there is no defensible percentage. This is a feature of the definition, not an omission in the research, and the distinction is important enough to state directly. Consider what a measurement would require. One workload mix is run on architecturally different fleets, at matched power, or matched cost, and compared. Only the hyperscalers, who run heterogeneous fleets internally, and the chip vendors, who can put them together, can make the comparison. Neither has any incentive to publish a number saying their own silicon is the wrong silicon for a significant portion of the work. The measurement is not difficult. It is unincentivized, which is a harder problem.

Even if it were released, there is a second reason why no single figure would be meaningful. The three layers do not compose additively. Scheduling loss and utilization loss are close to multiplicative. An unallocated node contributes no FLOPs, and a stalled node contributes none either. Mismatch is not a third multiplier on that product. It is re-weighting the value of the surviving FLOPs. The mismatch lives partly in the fraction the first two instruments count as realized, and partly in the fraction they count as lost, and no partition of the two recovers it. The quantity is real and the scalar is not well defined, which is the normal situation for a mechanism.

This is considered sufficient given the precedent of Hooker's hardware lottery. That essay quantified nothing and said why: the counterfactual of whether an idea would have succeeded on different hardware is hard to model. Instead, it gave a name to a mechanism, a collection of historical examples of the mechanism at work and an argument that the mechanism was intensifying. The word entered the vocabulary and the field began to look for cases. This essay attempts the same maneuver one stratum down.

3.2 The software lottery beneath the tax

Hooker's essay combines the hardware lottery with a software lottery: LISP and PROLOG were good for symbolic approaches and bad for connectionist ones, and the ergonomics of the available languages did some of the work of choosing which research directions flourished. The substrate stratum has its own version, discussed at length in 4.2. It is the proximate reason why the mismatch tax is expensive to avoid rather than simply present. Short form: the dominant programming model for accelerated compute is proprietary, the neutral alternative is lost on the ecosystem rather than on the spec, and the accumulated corpus of vendor-specific kernels now prices portability out of most engineering budgets. This has some specific significance. The mismatch tax is not just unmeasured, it is self-reinforcing: the lack of an instrument prevents anyone from pricing the portability decision, and the subsequent under-investment in portability increases the quantity the instrument would have measured. Hooker observed the same about her own subject when she said that it is hard to put a price on being concerned.

3.3 Two bounds available without proprietary data

Declining to state a rate is not the same as declining to bound the magnitude, and two bounds are available from published sources.

The first is the performance-per-watt spread documented by Jouppi and colleagues [Jouppi et al., 2017] for the TPU v1 on its design-point workloads. If specialized silicon delivers substantially more work per watt on the operations it was built for, then a workload mix containing those operations pays a proportional penalty when executed on a uniform fleet, and the size of that penalty is the integral of the published spreads across the mix. The bound requires no private data, only the vendor's own comparative claims taken seriously. Second is the composition of the workload. A representative production pipeline in 2026 includes dense matrix multiplication, sparse computation, mixture-of-experts routing, attention over variable sequence lengths, key-value cache management, retrieval over vector indices, embedding generation, tokenization, sampling, speculative decoding, draft model verification, encoders across several modalities, tool invocation, structured output parsing, and a substantial amount of CPU-bound control logic. The less dense matrix computation is a pipeline, the bigger the mismatched fraction is for a fleet optimized for dense matrix multiplication.

Neither constraint is small, and neither is closed by the usual remedies. Kernel optimization is about optimizing the implementation of an operation on a substrate, not the choice of substrate. Improved scheduling allocates nodes rather than choosing between kinds of node. Both address different bottlenecks, which is why a decade of progress on both has left the third layer unchanged.

4 Three Standardizations

> *Being too early is the same as being wrong.*
>
> quoted in Hooker, 2020

The strongest objection to everything above is that the missing layer is not missing but attempted. Several organizations built substantially the system this essay describes, and the historical record of what happened to them is the most informative evidence available about the mechanism.

4.1 The standardization that worked

Between 1974 and the mid-1980s the computer industry faced a structurally identical problem one layer up, and solved it. The solution is instructive less for the fact that it happened than for the specific mechanism by which it happened, which is not the mechanism usually cited.

The problem was proprietary incompatibility [Abbate, 1999]. IBM introduced Systems Network Architecture in 1974, Digital Equipment Corporation followed with DECnet, and Xerox with Xerox Network Systems. Each was competent within its own perimeter and none interoperated with the others. A customer's network topology was determined by a procurement decision rather than by engineering, which is the same complaint this essay makes about substrate selection in 2026.

Two neutral alternatives emerged, and this is the part usually collapsed. The first was Open Systems Interconnection, developed through the International Organization for Standardization and published as a reference model in 1984 [ISO 7498, 1984]. OSI was neutral by construction, better specified than its rival, and enjoyed formal backing from European telecommunications administrations and from the United States government, which mandated it through the Government OSI Profile [NIST, 1990] and did not relent until 1994. IBM and DEC were among its proponents while continuing to ship SNA and DECnet. The second was TCP/IP, developed under Defense Advanced Research Projects Agency [Hafner and Lyon, 1996] sponsorship following Cerf and Kahn's 1974 protocol proposal [Cerf and Kahn, 1974], adopted as a Department of Defense standard, and cut over on the ARPANET in January 1983 [Postel, 1981a; Postel, 1981b].

OSI lost. It is worth being exact about why, because the usual explanation, that committees are slow and the internet community was nimble, is true and insufficient. The decisive difference was patronage [Russell, 2013]. DARPA did not publish a specification and wait for the market. It funded implementations at BBN, MIT, Berkeley, UCLA and Washington, and when two competing implementations resulted it sent both to the Ballistics Research Laboratory for evaluation by a party with no stake in either [Leiner et al., 2009]. Berkeley then shipped a complete, working stack inside 4.2BSD [McKusick, 1999; Comer, 1988] in 1983, effectively free to any university that wanted it, at a moment when OSI implementations cost thousands of dollars. One historian of the period describes the cumulative effect of that patronage as leapfrogging market forces and pushing the protocols into operational use [Russell, 2014]. The lesson is not that neutral standards win. OSI was neutral and lost. The lesson is that a neutral standard wins when a patron with no position in the underlying products funds a reference implementation and gives it away, and loses when it is a specification produced by a consortium of parties who each have a position. Call this the patronage criterion.

4.2 The standardization that did not

Read against that criterion, the substrate stratum has already run the experiment and got the OSI outcome (Figure 5).

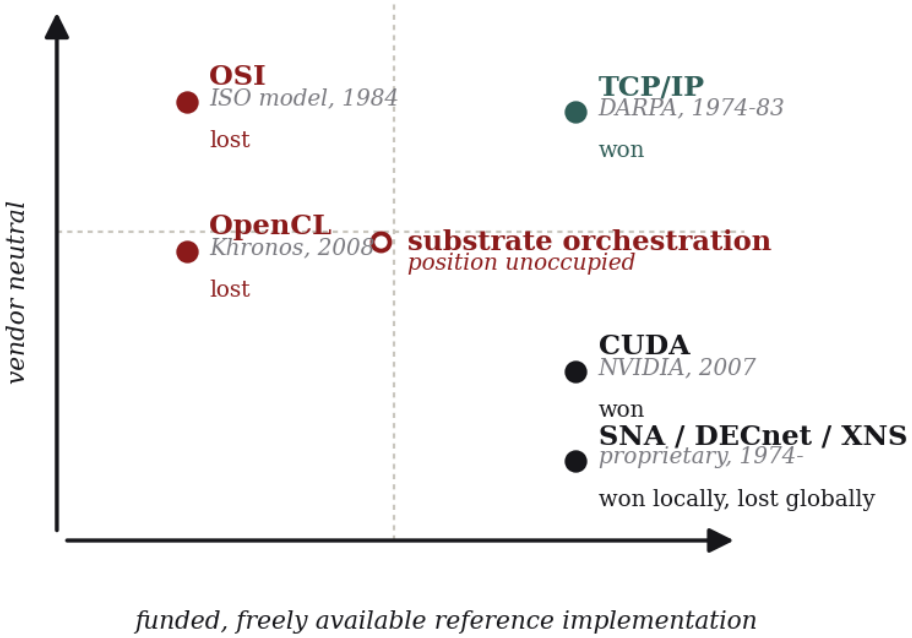


Figure 5: Historical record. Three standardization attempts, arranged by the variable that distinguished them. Neutrality did not decide the networking case: OSI was neutral, better specified, and government-mandated, and it lost. What separated the outcomes was whether a patron with no position in the underlying products funded a free reference implementation. On that criterion OpenCL sits with OSI, and the substrate orchestration layer is currently unpatronized.

OpenCL was initiated by Apple and ratified through the Khronos Group at the end of 2008 [Khronos Group, 2008; Stone et al., 2010], roughly a year behind CUDA [Nickolls et al., 2008], with adopter support from essentially every relevant silicon vendor. On the specification it was credible. On the criterion that actually decided the networking case it was structurally OSI rather than TCP/IP: a consortium document produced by members who each sold silicon, with no disinterested patron funding a reference implementation, and therefore no free, complete, pleasant-to-use stack to put against the incumbent's [Munshi, 2009]. Each vendor had every incentive to make its own implementation of the neutral standard adequate and its own proprietary stack excellent. That is what each did. The consequence compounds rather than fades. Every kernel written against a single vendor's stack [Lattner et al., 2021] raises the cost of executing that operation anywhere else, and the accumulated corpus of such kernels is now most of what makes substrate portability expensive. A workload does not run on the wrong chip because someone chose the wrong chip. It runs there because the code that implements it exists for that chip and nowhere else, and the labor to port it competes against the labor to ship.

4.3 The hardware precedents

The pattern predates the companies this essay is mostly concerned with, and the earlier cases matter because they establish that the outcome is a recurrence rather than a cohort effect.

IBM's Cell Broadband Engine was a heterogeneous processor [Kahle et al., 2005] in the strict sense: a general-purpose core paired with eight specialized elements, with the placement of work onto those elements left to the programmer (Figure 6). Roadrunner, built from 12,960 PowerXCell 8i processors paired with 6,480 dual-core AMD Opterons, sustained 1.026 petaflops on LINPACK [TOP500, 2008] on 25 May 2008 to become the first system past the petaflop barrier (Figure 7) [Gschwind et al., 2006]. It was decommissioned on 31 March 2013 while still ranked among the twenty-five fastest machines in the world.

Los Alamos was explicit about why. Alongside power efficiency, the laboratory cited the absence of a roadmap for the processor after its successor was cancelled in 2009, and the cost of adapting code to fit the Cell [LANL, 2013]. That second reason is the subject of this essay stated by an operator rather than an advocate: a machine that won on performance was retired in part because the labor of mapping work onto its heterogeneity exceeded what the performance was worth.

Intel's Larrabee followed a similar arc from the other direction. Announced in 2008 as a many-core x86 approach [Seiler et al., 2008] to throughput computing, it was withdrawn as a consumer graphics product in December 2009, the retail program was terminated in May 2010, and the architecture continued as Xeon Phi before that line was discontinued in turn. In each case the part was capable and the programming model was the binding constraint. The software case is the sharper one. OpenCL was a genuine attempt at the layer this essay describes, one stratum lower: a vendor-neutral programming model for heterogeneous compute, ratified with support from the whole industry, arriving barely a year behind the proprietary alternative. It did not fail technically. It lost on ecosystem, and the ecosystem accrued to the vendor selling the silicon, because that vendor had the strongest possible incentive to make its own stack pleasant and no incentive at all to make the neutral one pleasant.

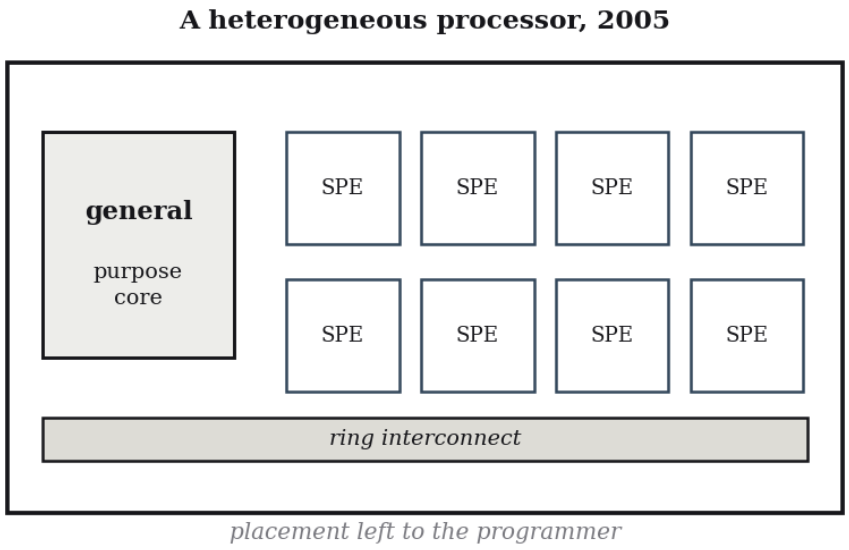


Figure 6: Schematic. A heterogeneous processor of the mid-2000s. A general-purpose core paired with eight specialized elements on a ring interconnect, with placement left to the programmer. Two decades later the placement is still the programmer's problem; only the granularity has changed, from elements on a die to families in a fleet.



Figure 7: Photograph. Roadrunner at Los Alamos National Laboratory, the first system to sustain a petaflop, in 2008. Its compute came from a heterogeneous pairing of general-purpose processors with Cell Broadband Engine elements, and the mapping of work onto those elements was the programmer's responsibility. The machine was decommissioned in 2013 while still among the fastest in the world. Photograph credit: Los Alamos National Laboratory.

Read together with the acquisitions below, the sequence suggests that vendor-neutral substrate layers do not fail for a single reason. The early ones were out-competed on developer experience by a party with more to gain from winning. The later ones were purchased. What the two have in common is the identity of the beneficiary.

4.4 The absorbed decade

OctoML was founded in 2019 as a spinout of the Apache TVM project [Chen et al., 2018] at the University of Washington. Its subject was precisely substrate portability: a compiler and serving stack that routed models across CPUs, GPUs, and accelerators including AWS Inferentia, with the explicit pitch that a customer should not have to choose a vendor in order to choose a model. NVIDIA acquired the company, by then renamed OctoAI [NVIDIA, 2024], in September 2024 for a reported 165 million dollars against a 2021 valuation near 900 million. Hosted services, accounts, and SDK access terminated on 31 October 2024, five weeks after the announcement, with no successor product.

Neural Magic, founded in 2018, pursued the adjacent problem of running inference efficiently on commodity silicon, and became a principal contributor to vLLM [Kwon et al., 2023], whose backend support spans AMD GPUs, AWS Neuron, Google TPUs, Intel Gaudi, NVIDIA GPUs, and x86 CPUs. Red Hat announced its acquisition in November 2024 [Red Hat, 2024]. NVIDIA had acquired Run:ai, an orchestration company, earlier that year, and subsequently acquired CentML.

The vendor-neutral substrate layer has been constructed several times. On each occasion an incumbent bought it (Figure 8).

4.5 Two readings of the record

The first reading is that the category failed on the demand side. The organizations that had real hardware diversity were large tech companies that already had internal compiler and kernel teams, and the long tail was built on NVIDIA GPUs via cloud providers with little need for portability. On this reading the market for cross-hardware tooling proved narrower than the funding implied. It casts the acquisitions as rational exits from a thin market, and this essay as a description of a problem that few organizations have.

The second reading is that the record sustains an incentive argument, not refutes it. A vendor-neutral routing layer is worth more to a dominant vendor as a removed option than to the market as a product. The observed transaction structure is consistent with that valuation.

Acquisitions in the low hundreds of millions, well below peak private marks, and termination within five weeks in at least one case, with no successor offering. That is not the profile of a company that is being acquired for its revenue.

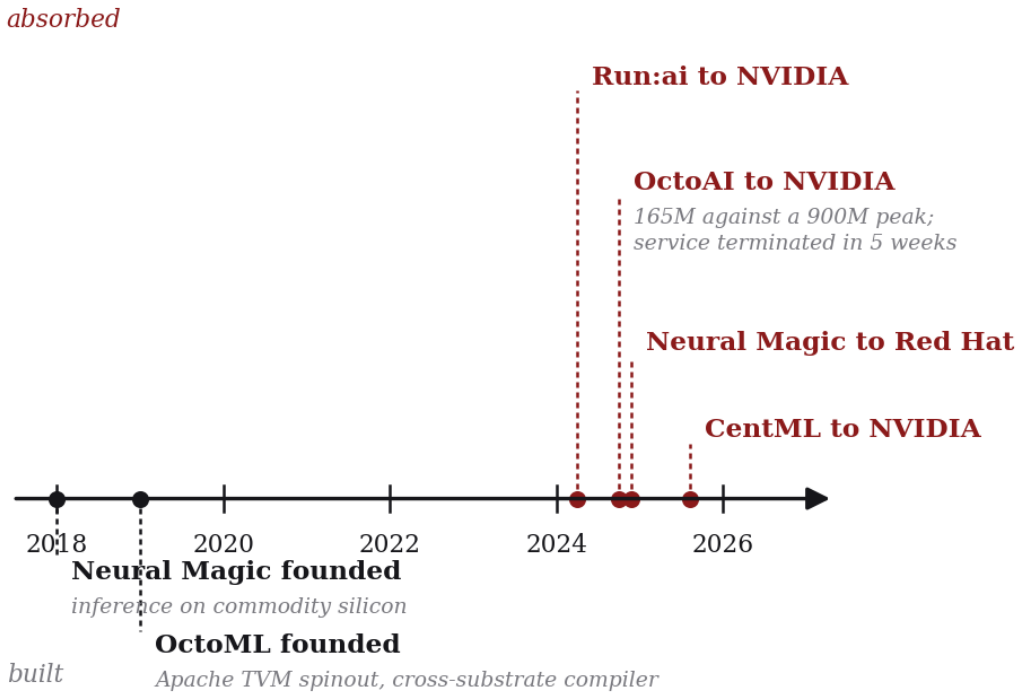


Figure 8: Historical record. The absorbed decade. Two independent efforts to build vendor-neutral substrate orchestration, founded 2018 and 2019, and the sequence of acquisitions between 2024 and 2025 that removed them and their successors from the market. Reported figures for the OctoAI transaction are approximately 165 million dollars against a 2021 valuation near 900 million, with hosted services terminated on 31 October 2024.

A division within the record is lost in the first pass that sharpens either reading. The acquisitions were not all in a single transaction. CentML optimized a single vendor's silicon, filling the gap between models and GPUs to get more from parts that vendor already sold. For that vendor, the feature is a capability, and buying it is standard product strategy. The compiler of OctoAI had run across vendors, a capability the acquirer can only leverage by continuing to make competitors' silicon run well. The observed results were the difference that the single-vendor optimizer was assimilated and merged and

the cross-vendor compiler was assimilated and eliminated in five weeks.

That pattern is more informative than the summary. It suggests that the cross-vendor layer is not just unprofitable in a thin market, but structurally unownable by anyone who sells silicon, because the asset loses its value in the hands of the acquirer the moment that the acquirer acts like itself. Inside a vendor, a neutral layer is a contradiction that resolves by shutdown.

The two readings are distinguishable. The distinguishing variable is diversity of substrate on demand side. The thin-market reading is correct for the period it describes, because in the homogeneous fleet era the practical substrate choice for virtually every buyer was which portion to rent from one vendor. It is wrong to the extent that Section 6 is right, because the population that needs cross-substrate routing is the population that runs heterogeneous workloads, and that population grows for structural reasons. The acquisitions are a hint of the size of the 2024 market. They are not evidence of the size of the one in 2028. The predictions in section 7 are formulated so that this disagreement gets resolved, not perpetuated.

4.6 The lottery and the tax

The relationship to Hooker's argument is worth stating explicitly, because the two mechanisms are adjacent and distinct. The hardware lottery concerns which research ideas succeed: an algorithm that maps poorly to available silicon becomes an academic curiosity regardless of merit [Hooker, 2020; Barham and Isard, 2019]. The mismatch tax concerns which deployed operations run efficiently: an operation that maps poorly to the silicon it was assigned wastes power and capital regardless of how well it was implemented.

They are the same physics observed at two different points in a workload's life. The lottery operates before an idea is deployed and selects against it. The tax operates after deployment and charges for it. Hooker's essay anticipated the second effect without naming it, in the observation that software kernels are often overly optimized for a specific type of hardware, producing large discrepancies in efficiency when used with different hardware. This essay takes that observation as its subject.

5 Adjacent Lineages

Four research lineages are converging on the problem this essay describes, and none of them individually amounts to the missing layer. Setting them out serves two purposes: it establishes that the optimization is tractable rather than speculative, and it makes plain which capability is absent.

5.1 Heterogeneous scheduling

The lineage runs from LATE at OSDI 2008 through FlexFlow, POP, Pathways and Alpa [Barham et al., 2022; Zheng et al., 2022], and it establishes that learned scheduling over heterogeneous resources beats hand-tuning [Zaharia et al., 2008; Narayanan et al., 2021] at every scale tested. POP reached allocations within 1.5 percent of optimal at a hundredfold runtime speedup. FlexFlow reached up to 3.3 times improvement [Jia et al., 2019] over hand-tuned baselines through learned search. The work has concentrated on homogeneous accelerator fleets or specific dual-substrate combinations; the substrate problem extends the lineage to the full multi-substrate distribution rather than departing from it.

5.2 Inference economics and learned routing

Post-2022 work demonstrates that learned orchestration at the inference layer cuts cost substantially at matched quality. FrugalGPT matches GPT-4 performance [Chen et al., 2023; Ong et al., 2024] at up to 98 percent cost reduction through a cascade router. RouteLLM reports 85 percent cost reduction at 95 percent of GPT-4 quality. vLLM with PagedAttention reports up to 24 times throughput improvement over HuggingFace Transformers at equivalent latency, and 2 to 4 times over the prior serving state of the art; the headline figure is against the weaker baseline, which matters when the number is quoted. FlexGen and SkyPilot [Yu et al., 2023] extend the same logic [Sheng et al., 2023] to heterogeneous memory tiers and to cross-cloud placement. Every result here routes among models or among memory tiers. None routes among chip families.

5.3 Compound AI compilation

DSPy is the canonical model-layer example: a framework that compiles a compound system specification into an optimized execution graph [Khattab et al., 2023] through metric-driven search, prompt optimization, and selective weight updates. The substrate-layer analogue would compile a workload specification into an optimized routing graph through metric-driven search, learned cost models, and dynamic re-routing. The compiler infrastructure to support it partly exists, in TVM, MLIR, Halide and the polyhedral tradition [Ragan-Kelley et al., 2013]. What does not exist is a system that carries the compilation across vendor boundaries with a cost model calibrated on measured behavior rather than on vendor datasheets.

5.4 Substrate-aware compute accounting

Treated at length in Section 7.3. The short form is that the orchestrator's telemetry distinguishes a pretraining run on a homogeneous cluster from a mixed-precision agentic inference workload spread across four substrate families at the same nominal FLOP count, and that current thresholds cannot.

5.5 The absent fifth lineage

The four above establish that the optimization is tractable, that learned routing pays, that the compilation framework exists one stratum up, and that the governance case is live. What is missing is the existence proof: a published demonstration of concurrent orchestration spanning all four classical chip families, or spanning architecturally distinct vendor stacks, at production scale. As of 18 August 2026 no such demonstration appears in the published record. That absence is consistent with Section 4. The parties able to produce such a demonstration are the ones with a reason not to publish it, and the parties that tried to build the layer independently were acquired. A missing existence proof is what the incentive argument predicts, which is weaker evidence than a demonstration would be and stronger evidence than nothing.

6 The Persistence of the Mismatch Tax

The mismatch tax would be a static inefficiency, and a manageable one, if the workload distribution were stable. It is not, and the mechanism destabilizing it is the compound systems thesis itself. This is the part of the argument that establishes urgency rather than mere existence, and it is the part most readers miss on first encounter.

Every component added to a compound AI pipeline brings its own optimal silicon. Vector retrieval is memory-bandwidth-bound and poorly served by dense matrix engines. A symbolic verifier is control-heavy and well served by a CPU. A speculative decoding draft model wants a different precision [Pope et al., 2023] and memory footprint than the verifier model it works against. A tool-calling agent loop interleaves control flow with inference in a pattern no single substrate executes well. Test-time compute scaling adds search and verification loops [Snell et al., 2024] whose substrate preferences differ from those of the model inference beneath them.

6.1 The substrate landscape

The size of the set the orchestrator must address is worth stating with a rule attached, because the number is a function of the rule and quoting it without one is how these figures become folklore.

The companion paper derives the count from an optimization surface test [Scott, 2026a] rather than a taxonomy, and this paper adopts the same test. Two designs are the same design only if code tuned for one runs well on the other. A distinct design is therefore one that requires its own compiler backend and its own hand tuned kernels. The test is chosen because it is the property that matters to the party doing the work, and because taxonomies drawn from vendor product lines answer a marketing question rather than an engineering one.

On that test roughly 250 distinct designs are active today, on a range of 230 to 265, of which more than 180 are new within the last three years and about 130 are shipping in volume. Around 190 companies are designing AI silicon. The derivation, including the sources for each figure and the boundary cases the test decides awkwardly, is in the companion paper [Scott, 2026a] and is not repeated here.

A stricter cut is also useful, and it is a subset of that population rather than a competing total.

Restricting to microarchitecture families with a publicly announced deployment serving production traffic, and treating generations within a family as one entry, gives a figure in the high forties. That smaller number describes what an orchestrator must support to be useful in production this year. The 250 describes what it must eventually address and the 130 shipping in volume is the population that will produce the compiler and kernel work if nothing changes. Every number is an answer to a different question, none of which is the count.

The important feature of this landscape is its shape, not its size. The thermodynamic niches do not overlap meaningfully. A GPU is strong on dense throughput and weak on control flow. A systolic-array ASIC is strong on matrix arithmetic per watt and weak outside its design point. An NPU is efficient on quantized inference and poor on large unstructured contexts. An FPGA offers reconfigurability and deterministic latency at the cost of throughput and programming effort. A CPU handles dispatch and control and loses badly on parallel arithmetic. Flexibility and efficiency are traded off along a frontier that has no interior dominant point, which is why the set does not collapse and why the match probability in the model below declines rather than stabilizes (Figure 9).

6.2 The recursion

The structure can be stated compactly. Consider a compound system at time t, which is made up of k(t) components. Each component has an optimal substrate among the set of available substrate types. Let p be the probability that a component's optimal substrate is the one a homogeneous fleet provides, and let m be the mean per-component penalty when it is not. The aggregate mismatch tax is then

approximately k(1-p)m, normalized to the fraction of compute used by mismatched components. All three terms are moving in the same direction. The number of components grows because retrieval-augmented generation introduced retrieval, tool use introduced external calls, agentic systems introduced control loops, test-time compute introduced search and verification, and every generation of compound system is more composite than the last. Match probability falls. As the set of possible substrates increases, the probability that the optimal substrate for any given component is the one provided by the fleet decreases monotonically.

The per-component penalty increases as specialized silicon continues to widen its advantage over general silicon on design-point workloads, increasing the cost of running a specialized operation somewhere else.

The product compounds rather than accumulates (Figure 10). Under illustrative annual rates the aggregate tax reaches several times its baseline within five years. Those rates are not measurements and the figure should be read as an exercise in what the structure implies, not as a forecast of a quantity.

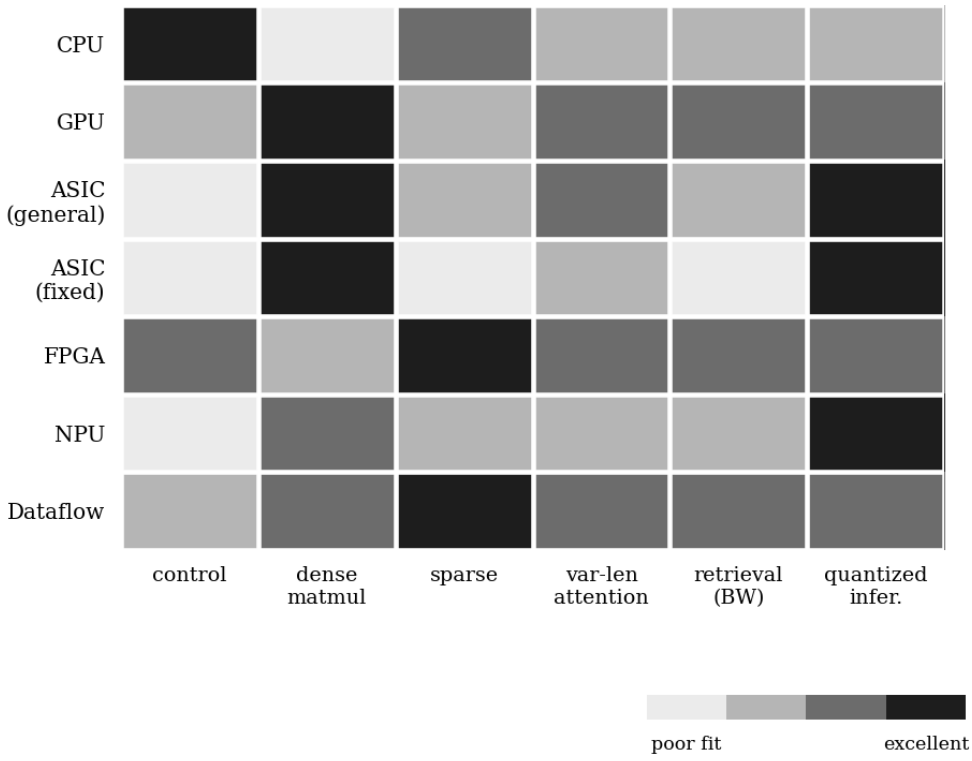


Figure 9: Modeled. Substrate fit across representative frontier workload classes. Shading is ordinal and illustrative rather than measured. The property that matters is structural: no row dominates. Every substrate family owns a column and loses badly on at least one other, which is why the set does not collapse and why the match probability in the model below declines as the set grows.

What survives without the rates is the direction of each term, and each direction is independently falsifiable. The mechanism fails if component counts reverse, which would require frontier pipelines to consolidate back toward single model calls. It fails if match probability reverses, which would require the substrate landscape to consolidate rather than diversify, the outcome Hennessy and Patterson

argued the end of Dennard scaling forecloses [Hennessy and Patterson, 2019; Esmaeilzadeh et al., 2011; Dennard et al., 1974]. It fails if the per-component penalty reverses, which would require specialized silicon to stop pulling away from general silicon on its design-point workloads. A reader who believes any of the three is occurring should discard this section and the argument that rests on it.

6.3 Hardwired silicon, and what it does and does not multiply

A development on the hardware side arrives at the same conclusion from the opposite direction, and the implication is easy to overstate.

Taalas etches a model's weights directly into a mask ROM fabric [AMD, 2026] rather than loading them from high bandwidth memory at runtime. Only two of the chip's hundred or more layers change between one model and the next, which the vendor states puts tape-out at roughly two months rather than the conventional cycle. The figures in this paragraph are vendor stated and have not been independently observed. Its first part, HC1 [Taalas, 2026], is a 6nm die of about 815 square millimeters carrying roughly 53 billion transistors, currently running a single model. AMD signed a definitive agreement to acquire the company on 6 August 2026, and expects the deal to close in the fourth quarter of that year. OpenAI and Broadcom announced a strategic collaboration in October 2025 to develop and deploy ten gigawatts of OpenAI-designed accelerators [OpenAI and Broadcom, 2025]. NVIDIA entered a definitive agreement to acquire Hugging Face, which hosts more than two million open models, with an event date of 2 September 2026, for approximately 11.9 billion dollars plus a retention program of up to a further billion [NVIDIA, 2026].

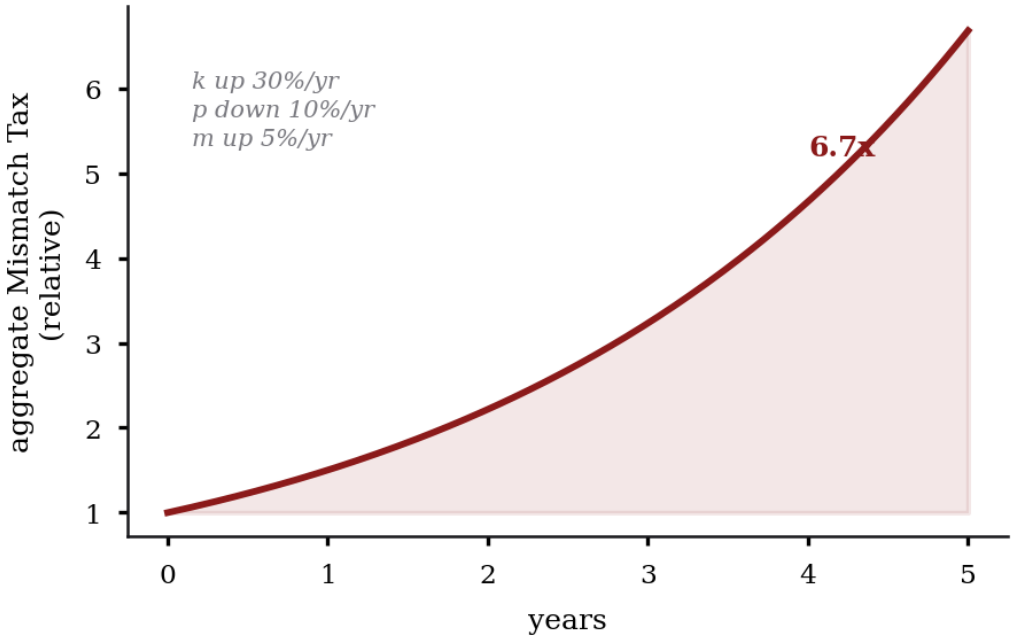


Figure 10: Modeled. Multiplicative compounding under stylized assumptions. The rates are illustrative rather than measured. What survives without them is the sign of each term, and each sign is independently checkable.

The temptation is to read this as a large increase in the count from Section 6.1, and it is not. A family of hardwired parts sharing a base die and a toolchain is one architecture instantiated many times. It

does not multiply compiler backends, and on the optimization surface test adopted above, it contributes roughly one distinct design rather than one per model etched. A paper that counted every hardwired part as a new architecture would be inflating its own premise, and the test exists precisely to prevent that.

What it multiplies is the number of deployment targets, a different quantity. Each completed part runs one model and nothing more. It cannot be repurposed as the workload mix changes, cannot absorb an operator it was not etched for, and cannot be reconfigured at any price. A fleet assembled like this is not tuned, it is routed across.

This makes it the hardest instance of the placement problem described by this paper, and the purest possible instance of the mismatch tax. The three loss layers of Section 3 degenerate here. Scheduling loss and utilization loss remain ordinary. Mismatch stops being a matter of degree: silicon that is allocated and busy and structurally incapable of executing anything but the one operator it was made for either gets that operator or goes to waste entirely. The tax on a hardwired part is not a percent of its throughput. When routing is wrong it is the whole part.

One qualification, because the claim as stated is stronger than the evidence warrants. Total loss assumes no sharing of the part. If the demand for the etched model is high enough to queue requests against the etched model, then the part is saturated and the loss is zero. The failure mode is not a partial mismatch, but a step function between saturation and idleness. And this paper cannot answer an empirical question about whether a given fleet is near one end or the other. That question can be answered only by those who run the fleets.

In both cases the structural point is the same: the recovery lever is routing and procurement, not tuning, because tuning has been removed from the design.

The effect on the recursion needs to be carefully stated, because the model in Section 6.2 does not extend to hardwired parts without an adjustment. p is the probability that the optimal substrate for a component is that provided by a homogeneous fleet, which assumes a fleet whose substrate is a property of the fleet, not the workload. A part etched for one model inverts that: the substrate becomes a property of a single component, and every other component's match probability against it is zero by construction. Such parts do not lower p literally, they are outside the space over which p is defined.

The tweak is to view a hardwired fleet as a partition and not a pool. The paper describes the best case as being within the partition that serves its etched model, where match probability is one and the per component penalty is zero. The worst is between partitions: a component with nowhere to go is not mismatched by degree but excluded. The aggregate then depends on how well the workload mix is predicted at tape-out, months in advance, which is a forecasting problem, not a routing one.

That is a different argument from the recursion. This paper does not make it.

7 The Way Forward

> *In order to avoid future hardware lotteries, we need to make it easier to quantify the opportunity cost of settling for the hardware and software we have.*
>
> Sara Hooker, 2020

Hooker's prescription was to make the opportunity cost cheaper to quantify. The same prescription applies here, and the obstacles are different in an instructive way. Hers were technical: benchmarking

an algorithm against multiple hardware types was awkward and the tooling did not exist. The obstacle here is that the tooling would have to be operated by a party with a reason to publish the result.

7.1 The independence problem

The chip vendors have software stacks that are optimized to make the best use of the chips that those vendors sell. They would lose part of their share in the load distribution, and their non-participation would become structural rather than a posture open to revision, with the creation of a neutral orchestrator.

Every hyperscaler operates a captive silicon program, and an orchestrator operated by its owner has an incentive to direct demand to that silicon at the margin. A neutral orchestrator would direct work to captive silicon only where captive silicon is optimal, which is a smaller portion of the distribution than internal planning assumes. The customer cannot verify the routing either way, which is what makes the incentive load-bearing rather than merely present.

Frontier labs can build the layer and have better uses for engineering capacity. A lab that makes one absorbs the cost of an industry-wide problem and either keeps the solution to themselves, wasting the externality, or puts it out there, giving away the advantage.

The residual conclusion is more limited than such arguments typically produce, and the limitation is due to Section 4. The layer has to be built by a party whose sole business is the orchestrator and whose neutrality is a structural commitment, not a stated policy. The historical record proves such a party is also acquirable. Independence is a posture a balance sheet can end. Whether the layer is an independent one is an ownership question that no engineering argument settles.

7.2 What an instrument would consist of

Arguing that a quantity is unmeasured obliges the arguer to say what measuring it would involve. The specification is not complicated, which is part of the point.

A mixed-workload basket, published and fixed, with per-operation weights drawn from a disclosed production trace rather than from the preferences of whoever assembled it. Execution of that basket on a homogeneous fleet and on a heterogeneous fleet, with both fleets specified by part number and count. Matched aggregate power at the rack rather than matched device count, because device count is the wrong denominator when the devices differ. Reporting of realized useful work per joule for both configurations, with the basket weights published so that a reader can recompute the result under different weights. Nothing in that procedure is novel. Every element exists in the high-performance computing benchmarking tradition [Hoefler and Belli, 2015], and the per-accelerator throughput literature already does the hard engineering parts. What is missing is a party with the fleets, the incentive, and the standing to publish. The natural candidates are the national laboratories, which operate heterogeneous systems, instrument them carefully, publish utilization studies as a matter of course, and sell no silicon.

Section 4.1 suggests the stronger version of this prescription. The networking case was not decided by a specification or by a measurement but by a patron who funded a reference implementation and distributed it free. The substrate stratum has produced consortium specifications and has not produced a patron. The parties positioned to play that role are the ones that played it before: defense research agencies, national laboratories, and the sovereign compute programs now committing capital to silicon for which no neutral orchestration layer exists. A procurement condition requiring

cross-substrate portability, attached to programs of that size, would do more than any argument in this essay.

A second instrument is worth naming because it would be cheaper and nearly as informative. Rather than measuring the tax directly, publish the basket and invite vendors to report their own results against it. Vendors will report selectively, and the pattern of what goes unreported is itself evidence. A standard basket that no vendor will run on a mixed workload tells a reader something, and it costs almost nothing to construct.

7.3 Substrate-aware compute accounting

Hooker's 2024 critique of compute thresholds [Hooker, 2024; Heim and Koessler, 2024] as a governance instrument applies with additional force once the substrate is acknowledged to be heterogeneous. A FLOP threshold that is meaningful on a uniform fleet is not meaningful on a mixed one, because the FLOPs are not commensurable. An ASIC FLOP, an FPGA FLOP, a CPU FLOP, and a GPU FLOP carry different precisions, memory access patterns, energy costs [Strubell et al., 2019], and capability profiles. Aggregating them into a single budget for regulatory purposes treats a unit of compute as a unit of capability [Sastry et al., 2024], which is a category error.

The error is embedded in current law. Article 51(2) of the EU AI Act [EU, 2024] presumes systemic risk for general-purpose models trained above ten to the twenty-fifth FLOP, and the general-purpose obligations under Articles 51 to 56 have applied since 2 August 2025, left untouched by the Digital Omnibus on AI which deferred the high-risk timelines but not these. In the United States, Executive Order 14110 [US, 2023] set a ten to the twenty-sixth threshold for dual-use foundation model reporting and was rescinded on 20 January 2025. The threshold outlived the instrument that created it: California's Transparency in Frontier Artificial Intelligence Act [California, 2025] adopts the same figure. That a quantity survives the repeal of the order that introduced it is reasonable evidence that the underlying assumption is entrenched rather than incidental.

The connection to the rest of this essay is direct. An orchestrator necessarily maintains calibrated cost models per substrate, routes across substrates, and collects execution telemetry, which makes it the only layer in the stack positioned to produce substrate-aware compute accounting as a byproduct of ordinary operation. The instrument that would measure the mismatch tax and the instrument that would make compute thresholds coherent are the same instrument. Neither exists, for the same reason.

7.4 The first measurement

The three preceding subsections describe an instrument that does not exist. One component of it now does. An exploratory analysis of decode efficiency, published at github.com/reckonresearch/mismatch-tax under Apache-2.0, computes for each measured configuration the gap between a card's own bandwidth-implied decode roofline and what a production serving stack extracted from it. It was not pre-registered. A hypotheses file was committed before the measured results, but with its intervals blank, so no prediction was registered [Scott, 2026b]. The ceiling used in Table 1, vendor spec bandwidth, was also chosen after the first results, for the reason given below. The corpus is five accelerators from two vendors, the L40S, H100 PCIe, H100 SXM, A100 80GB and MI300X, serving Llama-3-8B at bf16 under vLLM, at thirty configurations with two repetitions each.

The quantity is one minus the ratio of the roofline step time to the measured time per output token. The roofline reads the weights once per step and adds key-value cache traffic at the midpoint context, and it is computed from each card's own ceiling rather than from a common reference, so the figures are not a ranking of silicon against silicon. This is not the mismatch tax of Section 3. It is the layer-two residual of Section 3.2 measured directly, on one operator class, and the distinction governs everything that follows.

The first result was a falsification of the apparatus rather than of the thesis. Against the corpus's own measured device-to-device copy bandwidth the L40S returned a negative tax on all thirty configurations, at a median of minus 0.10. A roofline cannot be beaten, so the ceiling was wrong, and it was wrong in a way the number's provenance concealed. A copy benchmark counts bytes read plus bytes written, and the interleaved read and write traffic of a copy pays bus turnaround and write-allocate costs that decode, which is a pure read stream, does not. On the L40S the copy reached 652 of 864 GB/s of spec where a read stream reaches more than ninety percent of it. The same defect appeared on the H100 PCIe, which returned a negative tax on nine of its thirty configurations. The copy figure is retained in the corpus as what it is, a copy benchmark, and is no longer used as a decode ceiling; the defect is filed as Defect 17 in the Reckon register [Reckon Research, 2026].

Recomputed against vendor spec-sheet bandwidth the anomaly closes. The L40S best configuration returns 0.043, which is to say that vLLM on an L40S reaches about ninety-six percent of spec read bandwidth at its best point. Table 1 gives the five cells.

Silicon	**Provenance**	**Median**	**Best**	**Worst**
L40S	self-reported	0.17	0.04	0.24
H100 PCIe	self-reported	0.24	0.20	0.34
A100 80GB	captured	0.27	0.24	0.77
MI300X	self-reported	0.46	0.35	0.86
H100 SXM	captured	0.55	0.38	0.70

Table 1: Measured. Decode efficiency gap against each card's vendor spec-sheet bandwidth, Llama-3-8B at bf16 under vLLM, thirty configurations with two repetitions per cell. Every configuration is memory-bound. Provenance is the corpus's own silicon-identity flag. Because the ceiling is a vendor figure, each value is an upper bound on serving-stack inefficiency measured against that card's own ceiling, and the column is not a comparison between vendors.

Every configuration in the corpus is memory-bound, which is the expected regime for an eight billion parameter model at bf16 and batch thirty-two and below. The ordering is the result worth arguing with: the tax rises with the card's bandwidth. The two HBM3 parts leave the most on the table and the GDDR6 part leaves the least. That is the opposite of what one would expect if the silicon were the binding constraint, and it is what Section 3.2 predicts. At this model size and batch range the serving stack, not the silicon, is what fails to reach the ceiling on the newest hardware, and the software lottery dominates the operating point. Section 3's argument is unaffected in structure and bounded in application: the mismatch tax turns on a counterfactual requiring a different class of silicon, and nothing measured here supplies that counterfactual.

The loss is not spread evenly across the configuration space. The worst rows belong to the MI300X at a prompt length of 7,681 and batch sizes of sixteen and thirty-two, where measured time per output token runs from 36 to 65 milliseconds against a roofline of 6 to 9 milliseconds, and where throughput

rises about eleven percent while the batch doubles. At that context length the step is dominated by attention over the key-value cache rather than by weight streaming, which is a different operator with a different design point. That is the shape Section 6.1 describes: as a pipeline acquires components, the operators that depart furthest from the fleet's design point are the ones that set the cost.

Five caveats travel with these numbers. Three were written into the hypotheses file before the results: self-reported silicon provenance, serving-stack version, and the roofline's no-overlap assumption. The H100 SXM's older vLLM is a specific case of the second, found afterward, and the spec-bandwidth bound followed from the falsified copy ceiling. Three of the five cells, the MI300X, L40S and H100 PCIe, carry self-reported rather than captured silicon provenance. The serving-stack version varies by cell, and an effect larger than twofold has been observed on identical hardware from stack version alone. The H100 SXM environment record lists an older vLLM than its capture date implies, which is the leading candidate explanation for its 0.55 median and is the next re-measurement in the corpus. Spec bandwidth is a vendor figure, so the spec-ceiling tax is an upper bound on serving-stack inefficiency and the true ceiling lies between the copy figure and the spec figure. The roofline assumes no overlap of compute and memory, which biases it optimistic and the tax high by a small and unquantified margin.

The measurement says nothing vendor against vendor. A card's figure is the distance between its own ceiling and what one serving stack extracted from it on one day.

8 Four Predictions

A position paper that makes no checkable claim is no different than advocacy. The following are dated and falsifiable. Probabilities are the author's own.

Prediction 1. Custom AI accelerator silicon will represent more than 25 percent of combined datacenter AI accelerator revenue (Broadcom AI semiconductor revenue + Marvell custom silicon revenue) / (Broadcom AI semiconductor revenue + Marvell custom silicon revenue + NVIDIA data center revenue + AMD data center GPU revenue), on trailing four-quarter figures as reported in each company's filings, by 31 December 2027. Probability 0.55. The basis: the ratio has been increasing on disclosed quarterly numbers, the threshold is near the extrapolation, not far above it, and the principal uncertainty is merchant GPU revenue growth in the denominator rather than custom silicon growth in the numerator. Falsified if the ratio is at or below 25 percent on the trailing four quarters reported as of that date. This is a revenue proxy for substrate diversification, not a unit or FLOP measure, and it reads high because custom silicon carries a different margin structure. It adjudicates the two readings of Section 4.5: the thin-market interpretation predicts the ratio stalls.

Prediction 2. By 31 December 2028, no major chip vendor or hyperscaler will have shipped a generally available orchestration product that by default routes production inference workloads to a competitor's silicon. Probability 0.80. The basis: Section 7.1 argues that the incentive is structural, not strategic, and the historical record shows no case of a vendor shipping such a default. Most of the residual probability is given to a hyperscaler doing it as a cloud differentiator rather than a chip vendor doing it at all. Falsified if NVIDIA, AMD, Google, Amazon Web Services or Microsoft reaches general availability on a product whose documented default behavior includes routing to a named competitor's accelerator.

Prediction 3, conditional. If a peer-reviewed or vendor-released benchmark, by 31 December 2028, reports a like-for-like comparison of a mixed frontier AI workload on a homogeneous fleet versus a

heterogeneous fleet at matched power or matched cost, the reported gap will exceed 25 percent. Probability 0.7, conditional on such a comparison appearing. The basis is the two bounds in Section 3.3: published performance-per-watt spreads on design-point workloads are large, and the non-matmul share of a contemporary pipeline is substantial. Falsified if every such comparison published by that date reports a gap at or below 25 percent.

In an earlier draft, the absence of any such publication was disconfirming. That was a mistake and it is more important to fix it than the prediction is. Section 3.1 notes that the parties able to conduct the comparison have no incentive to publish it. An argument cannot hold that silence is structurally guaranteed and then count silence against itself. Here non-publication is uninformative, so this prediction is explicitly conditional, and the next prediction takes on the burden that conditionality gives up.

Prediction 4. By 31 December 2028, at least one national laboratory, defense research agency, or sovereign compute program will either publish substrate-resolved utilization for a production heterogeneous system or attach a cross-substrate portability requirement to a procurement above one billion dollars. Probability 0.4. The basis is that there is clearly institutional capability and the precedent in Section 4.1 is exact, but there is nothing that currently requires any of these bodies to act and procurement language moves slowly. Falsified if neither occurs. This is a load-bearing test, as it does not require the cooperation of any commercial stakeholder. If the argument of this essay is correct, and nothing of this kind happens within three years, then the argument is correct and inert, which is a distinction worth being able to observe.

9 Limitations

There are three weaknesses in this argument worth stating plainly rather than leaving for a reader to find.

The central quantity remains unmeasured. An adjacent one is now measured at one operating point: Section 7.4 reports an exploratory measurement on one corpus, one model, one serving stack and five cards, which is not a measurement across model sizes and not a measurement of a fleet. What it establishes is the layer-two residual of Section 3.2 at a single operating point, not the mismatch tax as Section 3 defines it, which turns on a counterfactual across architecturally distinct silicon that no public corpus yet supplies. Section 3.1 argues that the fleet-scale version of the measurement is structurally unavailable, and nothing in Section 7.4 relieves that. The reader who suspects motivated reasoning is entitled to, and the two bounds in Section 3.3 are offered precisely because they can be checked without granting the framing. If in practice both bounds are small, the essay is wrong, measurement or no measurement.

The historical argument selects on the dependent variable. Section 4 examines the neutral layers that have been absorbed or out-competed, and one that survived, and infers a criterion from the contrast.

It does not survey neutral layers which succeeded quietly or failed for reasons unrelated to their neutrality and no such survey is available to draw on. The patronage criterion should be understood as a hypothesis inferred from four cases, not a consequence established in a population.

The recursion rests on the direction of three terms, none of which is measured here. Section 6.2 gives falsification conditions for each, which is the most that can be given without the underlying series. Should the reader think component counts are flattening out or the substrate landscape is

consolidating, he/she should discard Section 6 and the concomitant claim that the problem is urgent, not just real.

Correspondence disputing any of the above is welcome, and the predictions of Section 8 are made in such a way that the disagreement can be settled by observation rather than by argument.

10 Conclusion

The thesis of compound systems held at the application stratum. As predicted, composed systems achieve state-of-the-art results and the orchestration layer on top of the models has become the locus of competitive advantage.

The same structure is found one stratum down, with the same mathematics and for the same reasons, and it is not visible because the quantity it produces has no instrument. Scheduling loss is measured because allocation is logged. Utilization loss is measured because peak and realized FLOPs are both disclosed. The mismatch tax is unmeasured, because the fleet-scale comparison requires concurrent access to architecturally distinct fleets that only the incumbents hold, and none of them can profitably publish it. Section 7.4 measures only the adjacent same-chip residual, at one operating point. Ten years of independent attempts to build the layer that would produce the measurement culminated in acquisition by those same incumbents.

This essay proposes the term mismatch tax for the loss incurred when allocated, busy silicon performs operations outside its design point. The loss is compounding because the success of compound AI systems is what drives substrate fragmentation. The absence of a number should not be read as evidence of a small quantity. It should be read as evidence about who owns the instruments.

References

v1.2, November 2026: relabels the Section 7.4 analysis as exploratory, because its hypotheses file was committed with its intervals blank four minutes before the first measured results were committed, and its ceiling was chosen after those results; this corrects the v1.1 line below. Corrects the 7.4 statement that all five caveats were declared before the run (three were). Corrects Sections 9 and 10, which said the central quantity is measured; Section 7.4 measures an adjacent quantity, and the mismatch tax remains unmeasured.

v1.1, November 2026: adds Section 7.4 and Table 1, reporting the first pre-registered measurement; adds the Code and data block; updates Section 9 and Section 10 to reflect that the central quantity is now measured at one operating point and still unmeasured at fleet scale; sources the OpenAI-Broadcom and NVIDIA-Hugging Face items to primary documents; and removes five claims that could not be sourced to a primary document.

References

Abbate, J. Inventing the Internet. MIT Press, 1999. https://mitpress.mit.edu/9780262261333/inventing-the-internet/

AMD. Definitive agreement to acquire Taalas. Announcement, 6 August 2026. Expected to close Q4 2026. https://ir.amd.com/news-events/press-releases/detail/1296/amd-acquires-taalas-to-advance-compute-solutions-for-rapidly-growing-ai-inference-market

Asanovic, K. et al. The landscape of parallel computing research: a view from Berkeley. Technical Report UCB/EECS-2006-183, 2006. https://www2.eecs.berkeley.edu/Pubs/TechRpts/2006/EECS-2006-183.pdf

Barham, P. and Isard, M. Machine learning systems are stuck in a rut. In Proceedings of the Workshop on Hot Topics in Operating Systems (HotOS), 2019. https://doi.org/10.1145/3317550.3321441

Barham, P. et al. Pathways: asynchronous distributed dataflow for ML. In Proceedings of Machine Learning and Systems (MLSys), 2022. https://proceedings.mlsys.org/paper_files/paper/2022/hash/37385144cac01dff38247ab11c119e3c-Abstract.html

Cerf, V. G. and Kahn, R. E. A protocol for packet network intercommunication. IEEE Transactions on Communications, 22(5), May 1974. https://doi.org/10.1109/TCOM.1974.1092259

Chen, L., Zaharia, M., and Zou, J. FrugalGPT: how to use large language models while reducing cost and improving performance. 2023. https://bair.berkeley.edu/blog/2024/02/18/compound-ai-systems/

Chen, T. et al. TVM: an automated end-to-end optimizing compiler for deep learning. In Proceedings of OSDI, 2018. https://www.usenix.org/conference/osdi18/presentation/chen

Comer, D. E. Internetworking with TCP/IP: Principles, Protocols, and Architecture. Prentice Hall, 1988.

Dean, J. The deep learning revolution and its implications for computer architecture and chip design. In IEEE International Solid-State Circuits Conference (ISSCC), 2020. Source of the epigraph to Section 2, as quoted in Hooker, 2020. https://doi.org/10.1109/ISSCC19947.2020.9063049

Dennard, R. H., Gaensslen, F. H., Yu, H., Rideout, V. L., Bassous, E., and LeBlanc, A. R. Design of ion-implanted MOSFET's with very small physical dimensions. IEEE Journal of Solid-State Circuits, 9(5), October 1974. https://doi.org/10.1109/JSSC.1974.1050511

Esmaeilzadeh, H. et al. Dark silicon and the end of multicore scaling. In Proceedings of ISCA, 2011. https://doi.org/10.1145/2000064.2000108

European Parliament and Council [EU]. Regulation (EU) 2024/1689 laying down harmonised rules on artificial intelligence, Articles 51 to 56. Obligations for general-purpose AI models applicable from 2 August 2025; subsequently amended by the Digital Omnibus on AI, which deferred high-risk timelines without altering Articles 51 to 56. https://eur-lex.europa.eu/eli/reg/2024/1689/oj

Feldman, M. The era of general purpose computers is ending. 2019. https://www.nextplatform.com/2019/02/05/the-era-of-general-purpose-computers-is-ending/

Gschwind, M. et al. Synergistic processing in Cell's multicore architecture. IEEE Micro, 26(2), 2006. https://doi.org/10.1109/MM.2006.41

Hafner, K. and Lyon, M. Where Wizards Stay Up Late: The Origins of the Internet. Simon and Schuster, 1996.

Heim, L. and Koessler, L. Training compute thresholds: features and functions in AI regulation. 2024. https://arxiv.org/abs/2405.10799

Hennessy, J. L. and Patterson, D. A. A new golden age for computer architecture. Communications of the ACM, 62(2), February 2019. Turing Lecture delivered June 2018. https://doi.org/10.1145/3282307

Hoefler, T. and Belli, R. Scientific benchmarking of parallel computing systems. In Proceedings of Supercomputing, 2015. https://doi.org/10.1145/2807591.2807644

Hooker, S. On the limitations of compute thresholds as a governance strategy. 2024. https://arxiv.org/abs/2407.05694

Hooker, S. The hardware lottery. arXiv:2009.06489v2, 21 September 2020. https://arxiv.org/abs/2009.06489

International Organization for Standardization [ISO 7498]. ISO 7498: Information processing systems, Open Systems Interconnection, basic reference model. 1984. https://www.iso.org/standard/14252.html

Jia, Z., Zaharia, M., and Aiken, A. Beyond data and model parallelism for deep neural networks (FlexFlow). In Proceedings of MLSys, 2019. https://bair.berkeley.edu/blog/2024/02/18/compound-ai-systems/

Jouppi, N. P. et al. In-datacenter performance analysis of a tensor processing unit. In Proceedings of the International Symposium on Computer Architecture (ISCA), 2017. https://doi.org/10.1145/3079856.3080246

Kahle, J. A. et al. Introduction to the Cell multiprocessor. IBM Journal of Research and Development, 49(4/5), 2005. https://doi.org/10.1147/rd.494.0589

Khattab, O. et al. DSPy: compiling declarative language model calls into self-improving pipelines. 2023. https://arxiv.org/abs/2310.03714

Khronos Group. The OpenCL specification, version 1.0. December 2008. https://www.khronos.org/opencl/

Kwon, W. et al. Efficient memory management for large language model serving with PagedAttention (vLLM). In Proceedings of SOSP, 2023. Note: the frequently quoted 24x figure is measured against HuggingFace

Transformers; the comparison against prior serving systems is 2 to 4x. https://doi.org/10.1145/3600006.3613165

Lattner, C. et al. MLIR: scaling compiler infrastructure for domain specific computation. In Proceedings of CGO, 2021. https://doi.org/10.1109/CGO51591.2021.9370308

Leiner, B. M. et al. A brief history of the Internet. ACM SIGCOMM Computer Communication Review, 39(5), 2009. https://doi.org/10.1145/1629607.1629613

Li, J., Michelogiannakis, G., Cook, B., Cooray, D., and Chen, Y. Analyzing resource utilization in an HPC system: a case study of NERSC's Perlmutter. In International Conference on High Performance Computing (ISC), 2023. https://doi.org/10.1007/978-3-031-32041-5_16

Los Alamos National Laboratory [LANL]. End of the road for Roadrunner. Press release, 29 March 2013. https://content.govdelivery.com/accounts/USLANL/bulletins/738666

Mattson, P. et al. MLPerf training benchmark. In Proceedings of MLSys, 2020. https://proceedings.mlsys.org/paper_files/paper/2020/hash/411e39b117e885341f25efb8912945f7-Abstract.html

McKusick, M. K. Twenty years of Berkeley Unix: from AT&T-owned to freely redistributable. In Open Sources: Voices from the Open Source Revolution. O'Reilly, 1999. [4.2BSD, 1983, shipped a complete TCP/IP stack.] https://www.oreilly.com/openbook/opensources/book/kirkmck.html

Meta AI. The Llama 3 herd of models. 2024. Reports BF16 model FLOPs utilization of 38 to 43 percent on 16,384 H100s. https://arxiv.org/abs/2407.21783

Midha, A. Interview. TBPN, 23 July 2026. https://www.youtube.com/live/8Jqi1yCniHk?t=2159 (decomposition from 36:00)

Moore, G. Cramming more components onto integrated circuits. Electronics, 38(8), April 1965.

Munshi, A. The OpenCL specification. In IEEE Hot Chips Symposium, 2009. https://doi.org/10.1109/HOTCHIPS.2009.7478342

Narayanan, D. et al. Solving large-scale granular resource allocation problems efficiently with POP. In Proceedings of SOSP, 2021. https://doi.org/10.1145/3477132.3483588

Nickolls, J., Buck, I., Garland, M., and Skadron, K. Scalable parallel programming with CUDA. ACM Queue, 6(2), 2008. https://doi.org/10.1145/1365490.1365500

NVIDIA. Acquisition of OctoAI, September 2024; acquisition of Run:ai, 2024; acquisition of CentML, 2025.

NVIDIA Corporation. Current Report on Form 8-K, Item 8.01, event date 2 September 2026. United States Securities and Exchange Commission. https://www.sec.gov/Archives/edgar/data/1045810/000104581026000078/nvda-20260902.htm

Ong, I. et al. RouteLLM: learning to route LLMs with preference data. 2024. https://arxiv.org/abs/2406.18665

OpenAI and Broadcom. OpenAI and Broadcom announce strategic collaboration to deploy 10 gigawatts of OpenAI-designed AI accelerators. Press release, 13 October 2025. https://investors.broadcom.com/news-releases/news-release-details/openai-and-broadcom-announce-strategic-collaboration-deploy-10

Pope, R. et al. Efficiently scaling transformer inference. In Proceedings of MLSys, 2023. https://proceedings.mlsys.org/paper_files/paper/2023/hash/c4be71ab8d24cdfb45e3d06dbfca2780-Abstract-mlsys2023.html

Postel, J. Internet Protocol. RFC 791, September 1981a. https://www.rfc-editor.org/info/rfc791

Postel, J. Transmission Control Protocol. RFC 793, September 1981b. https://www.rfc-editor.org/info/rfc793

Ragan-Kelley, J. et al. Halide: a language and compiler for optimizing parallelism, locality, and recomputation in image processing pipelines. In Proceedings of PLDI, 2013. https://doi.org/10.1145/2491956.2462176

Reckon Research. Measured corpus and defect register. https://github.com/reckonresearch/reckon (accessed 9 November 2026).

Reddi, V. J. et al. MLPerf inference benchmark. In Proceedings of ISCA, 2020. https://doi.org/10.1109/ISCA45697.2020.00045

Red Hat. Red Hat to acquire Neural Magic. Press release, November 2024. https://www.redhat.com/en/about/press-releases/red-hat-acquire-neural-magic

Russell, A. L. Open Standards and the Digital Age: History, Ideology, and Networks. Cambridge University Press, 2014. https://doi.org/10.1017/CBO9781139856553

Russell, A. L. OSI: the internet that wasn't. IEEE Spectrum, 30 July 2013. https://spectrum.ieee.org/osi-the-internet-that-wasnt

Sastry, G. et al. Computing power and the governance of artificial intelligence. 2024. https://arxiv.org/abs/2402.08797

Scott, S. Mismatch tax: an exploratory analysis on the measured corpus. 2026b. Apache-2.0. https://github.com/reckonresearch/mismatch-tax (accessed 18 November 2026).

Scott, S. The Chip Is Not The Product. Preprint, 2026a. Companion paper; derives the optimization surface counting rule and the population figures used in Section 6.1. https://www.sidney.fyi/papers/the-chip-is-not-the-product.pdf

Seiler, L. et al. Larrabee: a many-core x86 architecture for visual computing. ACM Transactions on Graphics, 27(3), 2008. https://doi.org/10.1145/1360612.1360617

Sheng, Y. et al. FlexGen: high-throughput generative inference of large language models with a single GPU. 2023. https://arxiv.org/abs/2303.06865

Snell, C., Lee, J., Xu, K., and Kumar, A. Scaling LLM test-time compute optimally can be more effective than scaling model parameters. 2024. https://arxiv.org/abs/2408.03314

State of California [California]. Transparency in Frontier Artificial Intelligence Act. 2025. https://leginfo.legislature.ca.gov/faces/billTextClient.xhtml?bill_id=202520260SB53

Stone, J. E., Gohara, D., and Shi, G. OpenCL: a parallel programming standard for heterogeneous computing systems. Computing in Science and Engineering, 12(3), 2010. https://doi.org/10.1109/MCSE.2010.69

Strubell, E., Ganesh, A., and McCallum, A. Energy and policy considerations for deep learning in NLP. In Proceedings of ACL, 2019. https://doi.org/10.18653/v1/P19-1355

Taalas. HC1: hardwired model inference on a mask ROM fabric. Product documentation, 2026. 6nm, approximately 815 square millimeters, approximately 53 billion transistors. https://taalas.com/products/

TOP500. Roadrunner, Los Alamos National Laboratory. TOP500 list, June 2008. https://www.top500.org/lists/top500/2008/06/highlights/

United States National Institute of Standards and Technology [NIST]. Government Open Systems Interconnection Profile (GOSIP), FIPS 146. 1990. Superseded 1994.

United States [US]. Executive Order 14110 on the safe, secure, and trustworthy development and use of artificial intelligence. October 2023. Rescinded 20 January 2025. https://www.federalregister.gov/documents/2023/11/01/2023-24283/safe-secure-and-trustworthy-development-and-use-of-artificial-intelligence

Yu, L. et al. SkyPilot: an intercloud broker for sky computing. In Proceedings of NSDI, 2023. https://www.usenix.org/conference/nsdi23/presentation/yang-zongheng

Zaharia, M. et al. The shift from models to compound AI systems. Berkeley Artificial Intelligence Research Blog, February 2024. https://bair.berkeley.edu/blog/2024/02/18/compound-ai-systems/

Zaharia, M., Konwinski, A., Joseph, A. D., Katz, R., and Stoica, I. Improving MapReduce performance in heterogeneous environments (LATE). In Proceedings of OSDI, 2008. https://bair.berkeley.edu/blog/2024/02/18/compound-ai-systems/

Zheng, L. et al. Alpa: automating inter- and intra-operator parallelism for distributed deep learning. In Proceedings of OSDI, 2022. https://www.usenix.org/conference/osdi22/presentation/zheng-lianmin