

Architectures and physical limits of fast and sustainable artificial intelligence

Baldo Alberto Luigi Dalporto

`baldo.dalporto@intec.edu.do`

Instituto Tecnológico de Santo Domingo INTEC

Sabine Mary

Instituto Tecnológico de Santo Domingo INTEC

Santiago Gallur²

Instituto Tecnológico de Santo Domingo INTEC

Research Article

Keywords: AI accelerators, memory bandwidth, interconnect latency, semiconductor physics, heterogeneous computing, sustainable data centers

Posted Date: October 8th, 2026

DOI: <https://doi.org/10.21203/rs.3.rs-11166087/v1>

License:   This work is licensed under a Creative Commons Attribution 4.0 International License.

[Read Full License](#)

Additional Declarations: No competing interests reported.

Architectures and physical limits of fast and sustainable artificial intelligence

Baldo Alberto Luigi Dalporto

25 September 2026

Instituto Tecnológico de Santo Domingo (INTEC), Santo Domingo, Dominican Republic

Baldo Alberto Luigi Dalporto^{1*}, Sabine Mary², Santiago Gallur²

¹ Department of Engineering, Instituto Tecnológico de Santo Domingo (INTEC), Santo Domingo, Dominican Republic

² Area of Social Sciences and Humanities, Instituto Tecnológico de Santo Domingo (INTEC), Santo Domingo, Dominican Republic

* **Correspondence:** Baldo Alberto Luigi Dalporto, baldo.dalporto@intec.edu.do

ORCID: Baldo Alberto Luigi Dalporto, 0009-0008-8719-1562; Sabine Mary, 0009-0004-3488-9511; Santiago Gallur, 0000-0001-6287-7340

Article type: Critical integrative review and analytical tutorial

Abstract

Fast artificial intelligence depends on the coordinated behavior of semiconductor devices, arithmetic units, memory hierarchies, communication fabrics, compilers and data center infrastructure. Peak arithmetic throughput alone cannot predict either response time or environmental performance. This critical integrative review connects these layers through a workload-dependent analysis of computation, data movement, communication, queueing and heat removal. It distinguishes training, prompt processing and autoregressive decoding; compares CPUs, GPUs, domain-specific accelerators, FPGAs and emerging computing paradigms; and examines semiconductor scaling, packaging, memory capacity, bus signaling and distributed execution. The analytical contribution is a decision framework that combines established performance bounds with explicit feasibility constraints and lifecycle accounting. Derived crossover conditions identify when increasing arithmetic throughput cannot improve latency, when batching encounters a memory limit, and when hardware replacement cannot yet recover its embodied carbon. A worked example uses published H100 SXM specifications exclusively as idealized hardware ceilings, not as measured application performance. Normalized sensitivity analyses expose the consequences of bandwidth, utilization and traffic reduction without manufacturing benchmark data. The synthesis indicates that sustainable acceleration requires minimizing useful work and transferred bytes while maintaining task quality, tail-latency limits and feasible power density. This is a literature-based study with transparent analytical derivations; no new device, prototype, experimental benchmark or field measurement is reported.

Keywords: AI accelerators; memory bandwidth; interconnect latency; semiconductor physics; heterogeneous computing; sustainable data centers.

Resumen

La velocidad de la inteligencia artificial depende de la coordinación entre los dispositivos semiconductores, las unidades de cálculo, las memorias, las interconexiones, el software y la infraestructura del centro de datos. Esta revisión crítica integra los límites físicos y arquitectónicos que condicionan el entrenamiento y la inferencia. Se examinan CPU, GPU, aceleradores especializados, FPGA y tecnologías emergentes, junto con la

transferencia de datos, el retardo, la refrigeración y la sostenibilidad. Se desarrolla un marco analítico de decisión que relaciona restricciones de memoria, cómputo, comunicación, calidad y energía. Los ejemplos numéricos derivados de especificaciones públicas se identifican como límites ideales y no como resultados experimentales. La contribución consiste en una síntesis técnica y en condiciones de cruce verificables matemáticamente, sin presentar prototipos ni mediciones propias.

1 Introduction

An AI service can execute a matrix multiplication rapidly and still deliver a slow response. The data may be waiting in host memory, the next layer may depend on a remote reduction, or the request may be queued behind a long sequence. A processor can also have excellent operations per joule while its deployment wastes energy through low utilization or unnecessary work. These observations motivate a system-level question: which resource actually limits useful AI computation, and what changes when that resource is improved?

Established surveys describe efficient neural-network processing and the significance of data reuse [1]. Roofline analysis relates computation to memory traffic [2], while energy-oriented architecture emphasizes the cost of moving operands [3]. More recent inference surveys organize model, data and system optimizations [29]. This article connects those perspectives to transistor physics, package-level interfaces, network serialization, thermal constraints and lifecycle emissions. It focuses on physically implemented classical AI systems, especially deep learning. It does not attempt an exhaustive catalog of every processor or algorithm marketed as AI.

Three questions organize the analysis. First, how do workload shape and dependency structure determine whether the limiting resource is arithmetic, memory capacity, bandwidth or communication? Second, which physical and software mechanisms make a nominal hardware improvement translate into lower completion time? Third, under what conditions does acceleration also reduce energy and environmental burden? These questions are relevant to hyperscale installations and to universities that must select modest infrastructure without owning semiconductor laboratories.

The contribution is an integrative decision framework rather than a new performance law or accelerator. It combines familiar bounds with capacity, quality and thermal feasibility checks, derives crossover conditions, and identifies evidence that would be required before claiming improvement. The mathematical results are conditional deductions from stated assumptions. They are not a substitute for hardware characterization, but neither do they require the construction of a prototype to be valid within those assumptions.

2 Review method and scope of evidence

This is a targeted critical integrative review, not a systematic review or meta-analysis. Source identification and verification were performed on 25 September 2026 through public scholarly search, publisher pages, author-deposited manuscripts and official technical documentation. Queries combined terms such as “efficient deep neural network processing,” “Roofline,” “FlashAttention,” “PagedAttention,” “distributed training,” “PCIe PAM4 FEC,” “CXL memory pooling,” “chiplet interconnect,” “in-memory computing,” and “data center energy.” Foundational sources were retained when they establish the physical or analytical basis of a current problem. Recent official sources were used for specification status, not to infer adoption in installed systems.

Sources were selected for a directly identifiable title and author or issuing organization, a persistent identifier or official URL, and a specific relationship to a technical claim. Peer-reviewed papers support scientific mechanisms and published implementations. Public preprint versions are identified as such where they provide accessible records. Consortium and manufacturer documents establish interface features and nominal product ceilings; their marketing comparisons are not treated as independent experiments. No proprietary Scopus or Web of Science search export, screening count, inter-rater agreement or fabricated PRISMA diagram is claimed.

Four evidence classes are kept distinct throughout the article: established analytical relationships; results reported by cited authors; published specification ceilings; and deductions made here. No performance figures from heterogeneous studies are pooled, because precision, task quality, workload, batching, software and

power boundaries differ. A paper’s existence and bibliographic identity do not establish universal applicability of its results. Where a source is available only as an abstract or technical landing page, the discussion is limited to its documented contribution rather than inaccessible implementation details.

The review has a deliberate selection bias toward sources that explain mechanisms and permit traceable reasoning. It cannot support claims of exhaustive coverage, bibliometric prevalence or identification of the globally fastest accelerator. Its scientific value rests on the consistency and usefulness of the cross-layer synthesis. Table 1 positions that synthesis against representative bodies of work without claiming that any earlier literature ignores a topic completely.

Evidence base	Main contribution used here	Extension made in this article
Sze et al. [1]	Neural processing architectures and data reuse	Connect reuse to service latency and facility constraints
Williams et al. [2]	Compute and bandwidth ceilings	Add capacity gates and exposed communication
Dao et al. [7] and Kwon et al. [8]	Attention traffic and KV memory management	Separate traffic reduction from capacity recovery
Rajbhandari et al. [10] and Narayanan et al. [11]	Distributed memory and execution strategies	Examine the communication crossover
Zhou et al. [29]	Taxonomy of efficient LLM inference	Connect software choices to physical interfaces and energy
Gupta et al. [26] and Masanet et al. [27]	Lifecycle and data center energy accounting	Derive replacement and utilization conditions

Table 1. Positioning of the review. This is a conceptual comparison, not a quantitative quality ranking.

3 Workloads determine the required architecture

3.1 AI is not a single arithmetic workload

Dense neural layers, convolutions, embedding lookups, graph traversal and autoregressive generation expose different resources. A convolution can reuse an input across multiple filters; an embedding lookup can perform little arithmetic for every retrieved byte; a graph algorithm may issue irregular accesses. CPU execution can remain appropriate for compact models, control-intensive inference or preprocessing. A high-throughput matrix engine benefits a workload only when the workload supplies supported operations with enough parallelism and local reuse [1].

For a dense matrix product with dimensions $m \times k$ and $k \times n$, counting one multiplication and one addition as two operations gives

$$F_{\text{GEMM}} \simeq 2mkn. \quad (1)$$

This convention must be stated when comparing MAC/s, FLOP/s and TOPS. Integer TOPS cannot be equated to floating-point FLOP/s without qualifying numerical format and algorithm. A sparse peak presupposes a supported sparsity pattern; it is not the dense peak for an arbitrary matrix.

3.2 Training and the two phases of transformer inference

Training retains activations, propagates gradients and updates optimizer states. Its memory requirement can greatly exceed stored inference weights. Prompt processing, or prefill, evaluates multiple positions together and can produce large matrix products. Autoregressive decoding generates successive positions with a sequential dependency. Small-batch decoding often has low weight reuse and is therefore sensitive to memory traffic. These are tendencies, not universal classifications: long contexts, large batches and communication can change the limiting resource [6–11].

For a decoder with N_p active dense parameters and b simultaneous sequences, a useful approximation for the parameter-dominated forward arithmetic per decoding step is $2bN_p$. It excludes attention score and value aggregation, sampling, normalization and other operations. For full attention, the matrix products forming scores and combining values contribute approximately $4bLh_qs^2d_h$ operations during prefill if a full square attention matrix is computed. Causal kernels can avoid some upper-triangular work. Here L is layer count, h_q query-head count, s context length and d_h head dimension. The quadratic term cannot be neglected merely because parameter count is fixed.

The standard attention operation is

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right)V. \quad (2)$$

Its mathematical definition does not require materializing the entire score matrix in high-bandwidth memory. FlashAttention exploits tiling and a numerically stable online softmax to reduce transfers while computing exact attention up to floating-point effects [7]. Its benefit illustrates that an algorithmic schedule can alter data movement without changing the model’s functional objective.

3.3 Capacity is a feasibility condition

Let p_w and p_{kv} be the bytes used per weight and cached key/value element. The following are ideal storage expressions, excluding metadata and alignment:

$$M_w = N_p p_w, \quad M_{kv} = 2Lbsh_{kv}d_h p_{kv}. \quad (3)$$

The factor two represents keys and values. Grouped-query attention reduces h_{kv} relative to query-head count [34]. It is a model architectural choice, whereas paging changes allocation. PagedAttention reduces fragmentation and supports flexible sharing of cache blocks; it does not make a logically required tensor disappear [8].

A device-resident execution is feasible only if weights, cache, activations and workspaces fit:

$$M_w + M_{kv} + M_{\text{act}} + M_{\text{work}} \leq M_{\text{usable}}. \quad (4)$$

Usable memory is smaller than advertised capacity when runtime allocations and reserves are included. If (4) fails, the system must partition, compress, recompute or offload. Each response changes the performance model. Reporting an arithmetic ceiling for an infeasible mapping is not a meaningful inference estimate.

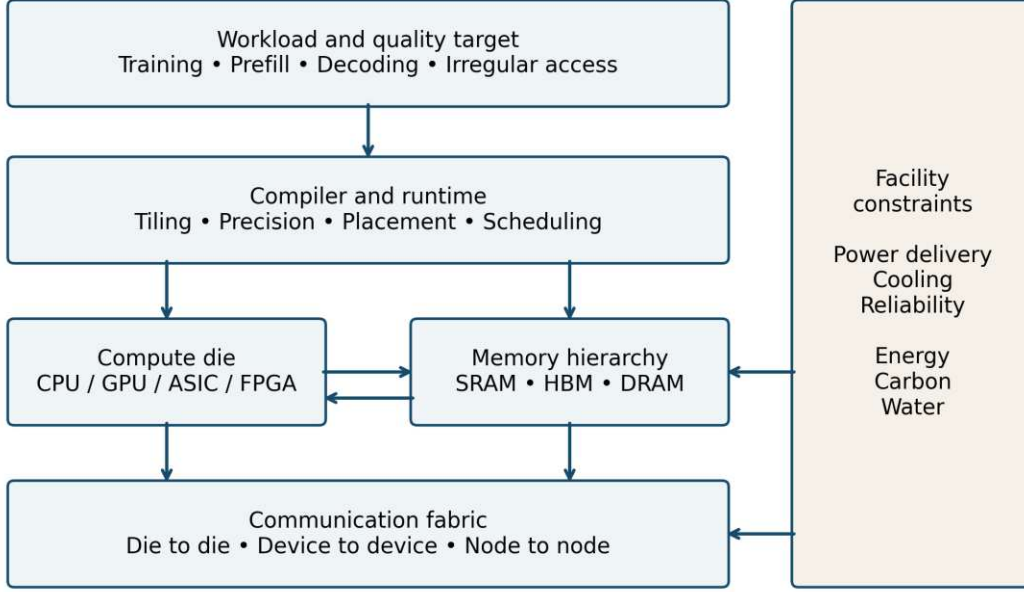


Figure 1. Original conceptual diagram. Workload and software select the computation and communication paths; electrical supply and cooling constrain all hardware layers. Arrows indicate dependencies, not measured traffic.

4 Semiconductor physics and chip implementation

4.1 Switching energy and the end of simple scaling

A CMOS gate changes state by charging and discharging capacitances. A first-order active-power relation is

$$P_{\text{dyn}} = \alpha C_{\text{sw}} V_{DD}^2 f, \quad P_{\text{leak}} = V_{DD} I_{\text{leak}}. \quad (5)$$

The activity factor α uses a convention consistent with the effective switched capacitance C_{sw} . Clock trees, short-circuit current, memory, I/O and voltage-regulator losses require additional accounting. Increasing frequency without reducing voltage raises dynamic power; raising voltage to preserve timing increases it more steeply. Lower frequency does not automatically minimize job energy, because static power persists for a longer execution.

Dennard scaling describes coordinated geometrical and voltage scaling under idealized constant-field conditions [4]. It is not a law guaranteeing that every newer manufacturing node has proportionally lower system energy. Leakage, short-channel electrostatics, parasitic resistance, interconnects and thermal density limit that extrapolation. FinFET and gate-all-around structures improve electrostatic control; nanosheet devices wrap the gate around multiple channels. TSMC’s published N2 research is an example of this device and interconnect co-design [5]. A commercial “nanometer” label should not be interpreted as the physical length of every transistor feature.

For a thermionic MOSFET, a useful subthreshold expression is

$$S = n \frac{k_B T}{q} \ln 10, \quad n \geq 1. \quad (6)$$

At 300 K, the ideal $n = 1$ value is approximately 59.5 mV per decade of drain current. This thermodynamic expression explains why reducing supply and threshold voltages while preserving both switching margin and low leakage is difficult. It is not a measured slope for a particular FinFET, nor a universal limit for every possible switching mechanism.

Landauer's minimum heat for erasing one unbiased bit in an ideal isothermal setting is

$$E_{\text{erase}} \geq k_B T \ln 2. \quad (7)$$

The value at 300 K is about 2.87×10^{-21} J [33]. This bound is not a FLOP energy specification: a floating-point operation entails many physical transitions, transport and control processes. Invoking a fundamental limit cannot establish that a practical accelerator is close to that limit.

4.2 Wires and power delivery matter as much as gates

In a simplified unbuffered on-chip wire of length ℓ , cross-sectional area A_c , resistivity ρ_e and capacitance per length c' , resistance and capacitance scale as

$$R = \frac{\rho_e \ell}{A_c}, \quad C = c' \ell, \quad t_{RC} \propto RC \propto \ell^2. \quad (8)$$

The proportionality coefficient depends on the distributed network, loading and chosen delay definition. Repeaters, hierarchical routing and pipeline registers can change the practical scaling at the cost of area, energy and latency. Equation (8) is therefore an explanatory wire model, not a prediction for a complete network-on-chip.

Voltage droop follows resistive and inductive mechanisms: $\Delta V \approx IR + L_p dI/dt$ for a lumped path. Here L_p is parasitic inductance, distinct from transformer layer count. Synchronized matrix activity creates large current transients. Decoupling capacitance, package inductance, voltage-regulator response and power-grid resistance consequently affect achievable clock frequency. Backside power delivery can redistribute routing resources, but its system benefit depends on implementation rather than the name of the process [5].

4.3 Packaging and manufacturing

Chiplets partition a system across dies; 2.5D integration places dies on an interposer or bridge, while 3D integration stacks active or memory layers. These arrangements can shorten critical links and increase wiring density. They also introduce bonding yield, test, thermal and repair challenges. High-bandwidth memory combines stacked DRAM with wide interfaces near the compute die. It increases data supply, but capacity, controller behavior and heat removal remain finite.

A simple Poisson defect model gives the illustrative yield relation $Y = \exp(-D_0 A_d)$, where D_0 is random defect density and A_d die area. It helps explain the incentive to partition large dies. Actual manufacturing yield also depends on defect clustering, systematic defects, redundancy, binning and assembly. No foundry yield or cost is inferred from this expression. A chiplet design may improve die yield yet lose part of that advantage in packaging cost and interface overhead.

UCIe standardizes die-to-die connectivity within a package [18]. It is not a replacement for a data center network, and standard compatibility alone does not establish coherent software behavior across arbitrary vendors. Published UCIe 3.0 information includes 48 and 64 GT/s modes. The existence of a specification is distinct from broad product availability.

5 Computing architectures and their operating domains

5.1 CPUs and vector processors

A CPU combines branch prediction, out-of-order execution, caches, vector instructions and general-purpose control. Its advantages are flexibility and relatively strong handling of irregular work. Host CPUs also handle scheduling, data preparation, storage and network services. A GPU can remain idle if those stages cannot feed

it. Memory locality across NUMA domains matters: a buffer allocated near one socket may require additional inter-socket traffic when accessed by another.

Vector or matrix instructions accelerate supported kernels, but instruction-set compatibility is not performance equivalence. x86, Arm and RISC-V identify architectural families with different implementations, caches, memory controllers and software maturity. An instruction set does not determine a processor’s sustained bandwidth or energy per inference. For a compact model and strict single-request latency, avoiding accelerator transfer and launch overhead can be more important than maximizing arithmetic throughput.

5.2 GPUs and specialized matrix engines

GPUs obtain throughput by executing many threads and overlapping memory stalls. SIMD/SIMT execution rewards regular access and penalizes divergent control. Matrix engines reuse operands in local structures and perform supported tile operations efficiently. Occupancy is only an intermediate metric: too many resident threads can be unhelpful when a kernel is already bandwidth-limited, while excessive register use can spill data to slower memory.

NVIDIA’s H100 documents describe specialized tensor computation and a high-bandwidth memory hierarchy [13,35]. These features are examples, not a claim of market leadership. Kernel shape, numerical format, tile alignment and fusion determine how much of the published peak is attainable. Combining small operations can eliminate intermediate writes, but too much fusion can increase register pressure and reduce parallel scheduling opportunities.

5.3 Domain-specific accelerators and systolic arrays

An application-specific accelerator can dedicate more area to useful datapaths and local storage by reducing general-purpose overhead. In a systolic array, data moves rhythmically between processing elements. For an ideal square $a \times a$ array performing a matrix tile with reduction length k , a simple schedule has an approximate fill and drain overhead of $2a - 2$ cycles. Its ideal temporal utilization is then

$$U_{\text{array}} \approx \frac{k}{k + 2a - 2}. \quad (9)$$

This expression assumes a compatible schedule, populated array and adequate operand supply. It excludes partial tiles, memory stalls and sparsity imbalance. It explains why increasing array dimensions can be ineffective for short reductions or poorly matched shapes. TPU v4 provides a published example of domain-specific processing integrated with a reconfigurable optical interconnect and support for embedding workloads [12]. Its reported comparisons remain conditional on the authors’ workloads and deployment environment.

5.4 FPGAs and emerging alternatives

FPGAs configure spatial pipelines, precision and dataflow with greater flexibility than a fixed ASIC. They are useful when a stable streaming task benefits from custom I/O and predictable processing. Their limitations include place-and-route effort, timing closure, resource fragmentation and memory supply. A lower clock frequency does not imply inferior performance for every task, because pipeline parallelism and elimination of unnecessary transfers can compensate.

Analog resistive crossbars exploit conductance and current summation to perform matrix-vector operations [21,22]. For ideal voltages v_i and programmed conductances G_{ij} , $i_j = \sum_i G_{ij} v_i$. Practical error includes device variation, drift, finite conductance range, line resistance and noise. ADCs, DACs and digital correction can dominate the energy budget. A crossbar-only TOPS/W figure is therefore insufficient to establish system superiority.

Photonic computation can perform optical interference or wavelength-parallel operations. Feldmann et al. demonstrated a photonic tensor-core approach [23]. That evidence does not remove the need to account for laser power, modulation, detection, conversion and electronic memory. Neuromorphic processors instead exploit event-driven state and spike communication; Loihi 2 is explicitly a research platform [24]. Neither approach can be assumed to accelerate an arbitrary dense transformer without a suitable mapping. Quantum computing is outside the performance model here: no general replacement of contemporary classical AI infrastructure is asserted.

Architecture	Suitable computational structure	Principal limitation to examine
CPU	Control, small batches, irregular kernels	Vector utilization and memory locality
GPU	Parallel dense arithmetic and mature kernels	HBM traffic, kernel shape and launch overhead
Domain-specific ASIC	Repeated supported operators	Shape flexibility and software support
FPGA	Custom streaming and specialized precision	External memory and implementation effort
Analog in-memory	Local matrix-vector products	Conversion, drift and numerical error
Photonic	Optically mapped linear operations	Conversion, laser and memory overhead
Neuromorphic	Sparse event-driven computation	Model mapping and task comparability

Table 2. Architectural suitability is conditional on workload and quality requirements. The table does not rank commercial products.

6 Memory bandwidth and the movement of operands

6.1 A hierarchy rather than a single memory

Registers, on-chip SRAM, caches, HBM, host DRAM and storage differ in capacity, bandwidth, latency and access granularity. They should not be reduced to a single “memory speed.” An operation can have sufficient capacity yet insufficient bandwidth, or high aggregate bandwidth but poor response time for dependent accesses. Layout, bank conflicts, transaction size and cache reuse determine the number of physically transferred bytes.

For an ideal GEMM that reads each input once and writes the output once, a minimal traffic estimate is $D = p(mk + kn + mn)$ bytes for element size p . Re-reading tiles, fetching old output values or spilling intermediates increases this traffic. The arithmetic intensity relative to a specified memory boundary is

$$I_j = \frac{F}{D_j} \quad [\text{operations/byte}]. \quad (10)$$

The subscript matters: HBM intensity differs from SRAM intensity. Increasing on-chip reuse can reduce D_{HBM} while increasing local accesses. This trade is favorable only when the added local cost is smaller than the avoided external cost [1–3].

6.2 Hierarchical performance ceilings

For sustained compute capacity C_{eff} and effective bandwidths B_j , an idealized resource lower bound on execution time is

$$T_{\text{res}} \geq \max\left(\frac{F}{C_{\text{eff}}}, \frac{D_1}{B_1}, \dots, \frac{D_J}{B_J}\right). \quad (11)$$

The corresponding single-boundary Roofline expression is $R \leq \min(C_{\text{eff}}, IB)$ [2]. Maximum overlap is implicit in the lower bound; it is not guaranteed by simultaneous availability of resources. If compute and transfers are serialized, their times add. Dependencies, issue limits and synchronization make actual completion time larger still.

An effective bandwidth is not a universal constant. It can be written $B_{\text{eff}} = \eta_B B_{\text{peak}}$ only as a workload-specific simplification. The efficiency factor absorbs transaction and access effects without explaining them. It must not be silently chosen to make a model agree with a desired conclusion. The same caution applies to a compute efficiency η_C .

6.3 Precision and sparsity

Quantization reduces representation size and can increase supported arithmetic throughput. It also introduces scale factors, packing, conversion and numerical sensitivity. LLM.int8() uses mixed treatment of outlier dimensions instead of assuming all activation values tolerate the same quantizer [9]. A lower precision is acceptable only if it meets the specified task-quality requirement. Matching parameter count is not enough to establish a fair quality comparison.

Unstructured sparsity saves arithmetic only when the hardware and software skip zeros efficiently. Index storage, irregular loads and imbalance can consume the theoretical benefit. Structured sparsity is easier to accelerate but constrains pruning or training. Compression should consequently be evaluated through the useful bytes and operations actually removed from the critical path, with all metadata included.

7 Buses interconnects and delay

7.1 Distinguishing bit rate from useful payload

Within a node, links carry transactions among CPUs, accelerators, NICs, memory and storage. Across nodes, switches route packets or configure circuits. A generic physical link with n_ℓ lanes, symbol rate f_s and m_b encoded bits per symbol has the useful throughput model

$$B_{\text{payload}} = \frac{n_\ell f_s m_b}{8} \eta_{\text{coding}} \eta_{\text{protocol}}. \quad (12)$$

This expression is valid only if the symbol-rate definition is used consistently. A consortium's GT/s figure may already express its standardized effective transfer-rate convention; multiplying such a figure again by PAM4's two bits per symbol can double-count the gain. Full-duplex aggregate bandwidth must also be divided appropriately when a workload transfers data in only one direction.

PCIe 6.0 uses PAM4, flit-based transfer and forward error correction [14]. PAM4 carries more information per symbol but has smaller voltage separation between adjacent levels than two-level signaling at the same swing. Equalization, error correction and retry behavior must be considered alongside nominal rate. PCIe 7.0 was released in 2025 with a 128 GT/s specification rate [15]; this is a standard milestone, not evidence that every deployed accelerator supports it.

Interface example	Published feature or ceiling	Interpretation boundary
PCIe 5.0 x16	About 63.0 GB/s per direction after 128/130 coding	Payload lower after transaction overhead

Interface example	Published feature or ceiling	Interpretation boundary
PCIe 6.0 x16 [14]	64 GT/s; up to 256 GB/s aggregate bidirectional	About 128 GB/s per direction before remaining overhead
PCIe 7.0 x16 [15]	128 GT/s; up to 512 GB/s aggregate bidirectional	Specification availability is not deployment
CXL 3.0 [16]	Coherent expansion and fabrics over PCIe 6.0 technology	Added topology and access semantics matter
CXL 4.0 [17]	Announced specification rate of 128 GT/s	Not equivalent to local HBM access
UCIe 3.0 [18]	48 and 64 GT/s die-to-die modes	Package-scale interface, not rack networking
H100 SXM HBM [13]	3.35 TB/s device memory bandwidth	Nominal ceiling, not measured application traffic

Table 3. Traceable interface examples. Decimal GB and TB are used. Different rows describe different architectural boundaries and must not be treated as interchangeable bandwidth resources.

7.2 The latency decomposition

For a message of D bytes, a useful decomposition is

$$T_{\text{msg}} = T_{\text{setup}} + T_{\text{queue}} + T_{\text{ser}} + T_{\text{prop}} + T_{\text{switch}} + T_{\text{recv}}, \quad (13)$$

$$T_{\text{ser}} = D/B_{\text{payload}}, \quad T_{\text{prop}} = \ell/v. \quad (14)$$

The switch term must reflect cut-through or store-and-forward behavior. Serialization must not be charged as a full additional message at every hop if packet transmission is pipelined. Optical propagation in fiber is approximately a few nanoseconds per meter; transceiver and protocol latency are additional. Conversely, the resistance-capacitance model of an on-chip wire should not be extrapolated directly to a terminated high-speed transmission line.

Let α collect fixed latency and let B be sustained payload bandwidth under low congestion. Then $T = \alpha + D/B$. The crossover message size is $D^* = \alpha B$. Below it, reducing setup and synchronization is more effective than increasing bandwidth. Above it, serialization becomes dominant. Figure 2 shows this relationship in dimensionless form without assuming a latency for any vendor product.

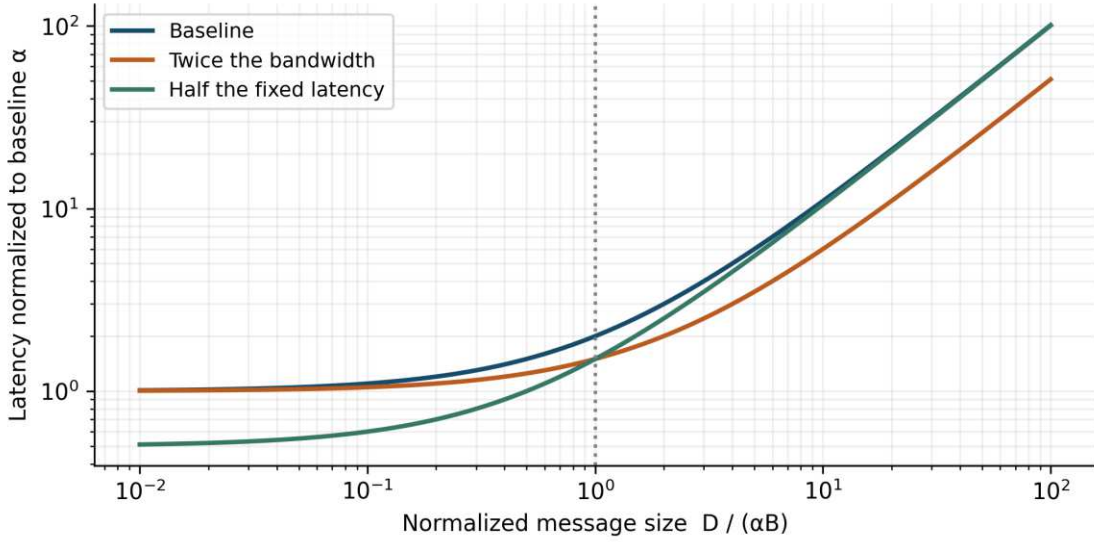


Figure 2. Original analytical plot of $T/\alpha = 1 + x$, with $x = D/(\alpha B)$. Doubling bandwidth gives $1 + x/2$; halving fixed latency gives $0.5 + x$. No network measurements are represented.

7.3 Coherence and remote memory

CXL adds memory and coherency capabilities to a PCIe-derived physical foundation [16,17]. Pooling can improve allocation flexibility or enable a model to fit, but a remote cache-line fetch is not equivalent to a local HBM access. Directory operations, switches, contention and access granularity can dominate. A large virtual address space also does not imply that all addresses have equal performance.

A necessary concurrency condition to approach bandwidth B when round-trip latency is T_r is approximately $D_{\text{outstanding}} \geq B T_r$. A dependent pointer chain cannot create unlimited independent transactions, so it may remain latency-bound on a high-bandwidth fabric. Prefetching helps only when future addresses can be predicted and unnecessary transfers do not overwhelm the system.

7.4 Distributed training and topology

Data parallelism replicates model computation across different examples and synchronizes updates. Tensor parallelism divides operations within layers; pipeline parallelism partitions layers; expert parallelism routes tokens to different experts. Each changes communication volume, message count and dependency depth [10,11]. An all-reduce result cannot be inferred from link speed alone. NCCL documents the collective semantics used to combine and redistribute tensors [19].

For an ideal ring all-reduce across n participants with G bytes per participant, per-step setup latency α and effective directional bandwidth B , a common model is

$$T_{\text{ring}} \approx 2(n-1)\alpha + \frac{2(n-1)G}{nB}. \quad (15)$$

The expression assumes a balanced ring and excludes congestion and protocol details. Tree algorithms, hierarchical collectives and in-network operations have different cost structures. A well-suited network topology must supply the collective's communication pattern, including its bisection demand. The sum of endpoint port speeds is not the bandwidth available across an oversubscribed cut.

Communication can overlap with independent computation, but not with computation that requires its result. With overlap fraction θ and separately accounted launch overhead, a simple exposure model is

$$T_{\text{step}} \approx T_{\text{local}} + (1 - \theta)T_{\text{comm}} + T_{\text{launch}}, \quad 0 \leq \theta \leq 1. \quad (16)$$

This is an accounting approximation rather than a fitted predictor. Changing partitioning can change every term. More devices reduce work per participant but can increase the relative cost of synchronization. ZeRO reduces replicated states through partitioning, while Megatron-LM explores combinations of parallelism; neither establishes that adding devices always improves energy or time [10,11].

7.5 RDMA fabrics and storage paths

InfiniBand and Ethernet-based RDMA deployments reduce some host-mediated data movement, but they do not remove physical propagation, switch queues or endpoint processing. RoCEv2 carries RDMA traffic across IP networks. Congestion notification, endpoint rate control and, in some configurations, priority flow control determine behavior under incast. Official documentation describes both lossless and lossy operating configurations [36]. It would therefore be inaccurate to claim that every RoCE deployment universally requires the same pause mechanism.

A leaf-spine fabric can provide multiple paths, yet effective bisection capacity depends on port allocation and oversubscription. Expert routing can create skewed all-to-all traffic that differs from the balanced ring assumed in (15). Direct accelerator-to-NIC transfer can avoid a host staging copy, but its feasibility depends on topology, memory registration and software support. Storage should also be included: dataset reads, checkpoint writes and recovery can compete for network bandwidth. A faster collective cannot remedy an input pipeline that fails to supply batches.

8 Software service scheduling and utilization

8.1 The compiler is part of the architecture

The path from model graph to executable kernels determines layouts, fusion, tiling, placement and communication. OpenXLA is an example of a compiler infrastructure that transforms machine-learning computations for supported backends [20]. CUDA, ROCm and other ecosystems provide different libraries, tools and deployment interfaces; a mathematical operation supported in a framework may still fall back to a slow implementation on a particular device.

Portability should therefore be evaluated at two levels: semantic portability, meaning that the program computes the intended result, and performance portability, meaning that its mapping uses a new architecture effectively. Compilation time, graph specialization, dynamic shapes and warm-up affect deployment. A benchmark excluding those costs can be appropriate for a persistent service but misleading for short-lived jobs.

8.2 Batching and tail latency

Batching amortizes weight reads and launches across requests. It can also increase waiting time. Continuous batching admits and retires sequences at token boundaries, but variable sequence length, cache size and scheduler policy determine fairness. Prefill and decoding may contend for the same memory and arithmetic resources. Their disaggregation can reduce interference while introducing cache transfer and additional orchestration.

For a stable service, Little’s law gives

$$N_{\text{sys}} = \lambda \bar{T}. \quad (17)$$

Here N_{sys} is average requests in the system, λ arrival rate and $\bar{T}$ average residence time [32]. This identity does not predict tail latency. Under the much narrower M/M/1 assumptions, mean residence time is $1/(\mu - \lambda)$

for service rate $\mu > \lambda$. That expression illustrates rising delay near saturation, but real token services rarely have exponential independent service times. Measured arrival and service distributions are needed for credible p95 or p99 predictions.

Latency reporting should distinguish time to first token, inter-token delay, complete-response time and queueing. Throughput should state whether it counts input tokens, generated tokens or requests. A service-level objective should specify its percentile, workload and allowed failure rate. MLPerf provides a relevant precedent for coupling benchmark scenarios and quality constraints [28]. A high throughput obtained by relaxing quality or latency is a different operating point, not an unconditional improvement.

8.3 Reliability and useful completion

Error correction, checkpointing, retries and fault isolation consume resources but preserve useful work. A large training job that frequently restarts can waste more energy than a slightly slower, more reliable configuration. Security isolation and encryption can also change data paths. Consequently, the denominator of efficiency should be successfully completed work meeting a quality target, rather than raw arithmetic issued.

Amdahl’s expression $S(n) = 1/[s + (1 - s)/n]$ bounds ideal speedup for a fixed serial fraction s [31]. Its lesson applies beyond processor cores: a faster GPU cannot accelerate preprocessing, communication or user-visible queueing that lies outside the accelerated fraction. In distributed AI, s itself can change with scheduling and partitioning, so it should not be treated as a permanent property of the model.

9 A constrained analytical framework

9.1 Defining the feasible design space

Let x describe the architecture and deployment, including device count, precision, batching, partitioning and software mapping. A useful optimization is to minimize response time and lifecycle impact subject to constraints:

$$\min_x (T_{\text{task}}(x), E_{\text{task}}(x), C_{\text{life}}(x)), \quad (18)$$

$$Q(x) \geq Q_{\min}, \quad T_{p99}(x) \leq T_{\max}, \quad M(x) \leq M_{\text{usable}}(x), \quad (19)$$

$$P_{\text{rack}}(x) \leq P_{\text{supply}}, \quad T_j(x) \leq T_{j,\max}. \quad (20)$$

The quality functional Q must be chosen for the task; it cannot be inferred from numerical precision alone. No Pareto frontier is numerically solved here because that would require workload and deployment measurements. Equations (18–20) identify what a valid comparison must hold constant or constrain.

The practical sequence is to test capacity first, identify the local resource bound second, add exposed communication and serial dependencies third, and evaluate service and energy boundaries last. Each step can invalidate an apparently favorable hardware choice. The framework deliberately distinguishes a lower bound from an operational prediction.

9.2 Arithmetic and bandwidth crossover

For a low-batch dense decoder whose dominant traffic is reading weights once per step, the assumptions of Section 3 give

$$I_{\text{weights}} \approx \frac{2bN_p}{N_p p_w} = \frac{2b}{p_w}. \quad (21)$$

Equating IB with C yields the ideal batch crossover

$$b^* = \frac{p_w C}{2B}. \quad (22)$$

The cancellation of parameter count does not mean that model size is irrelevant. It still determines whether the model fits, and cache traffic, attention arithmetic and scheduling alter intensity. Equation (22) is a reduced-model diagnostic for the weight-dominated regime, not a universal optimal batch size.

Combining (3–4) gives a capacity-limited batch estimate when non-cache memory is held fixed:

$$b_{\max} = \text{floor}\left(\frac{M_{\text{usable}} - M_w - M_{\text{other}}}{2Lsh_{kv}d_h p_{kv}}\right). \quad (23)$$

If $b_{\max} < b^*$, the simplified mapping cannot reach its compute ceiling through batching alone. Reducing cache precision or key/value heads can relax the capacity condition; reducing weight precision changes both traffic and the hardware’s applicable compute ceiling. If other memory grows with batch, (23) must be replaced by the full inequality.

9.3 Worked example using published hardware ceilings

The H100 SXM product specification lists 80 GB memory, 3.35 TB/s memory bandwidth and 1,979 TFLOP/s BF16 tensor performance explicitly with sparsity [13]. A dense-equivalent ceiling of approximately 989.5 TFLOP/s follows from the associated factor-two sparse convention documented for the architecture [35]. The example uses the rounded dense value 989 TFLOP/s. It does not mix the sparse peak with a dense workload.

With those ceilings, the Roofline ridge is $989/3.35 \approx 295.2$ operations/byte. At $p_w = 2$ bytes per weight, (22) gives an ideal weight-only crossover near batch 295. For an analytical workload defined as $N_p = 7 \times 10^9$ dense parameters and batch one, weights occupy 14 GB, nominal weight-read time is $14/3350 \approx 4.18$ ms, and arithmetic time is $14 \times 10^9 / (989 \times 10^{12}) \approx 14.2$ microseconds. The corresponding weight-only ceiling is about 239 decoding steps/s. These calculations do not describe a benchmark of a named 7B model and exclude all non-weight work and traffic. They must not be advertised as observed tokens/s.

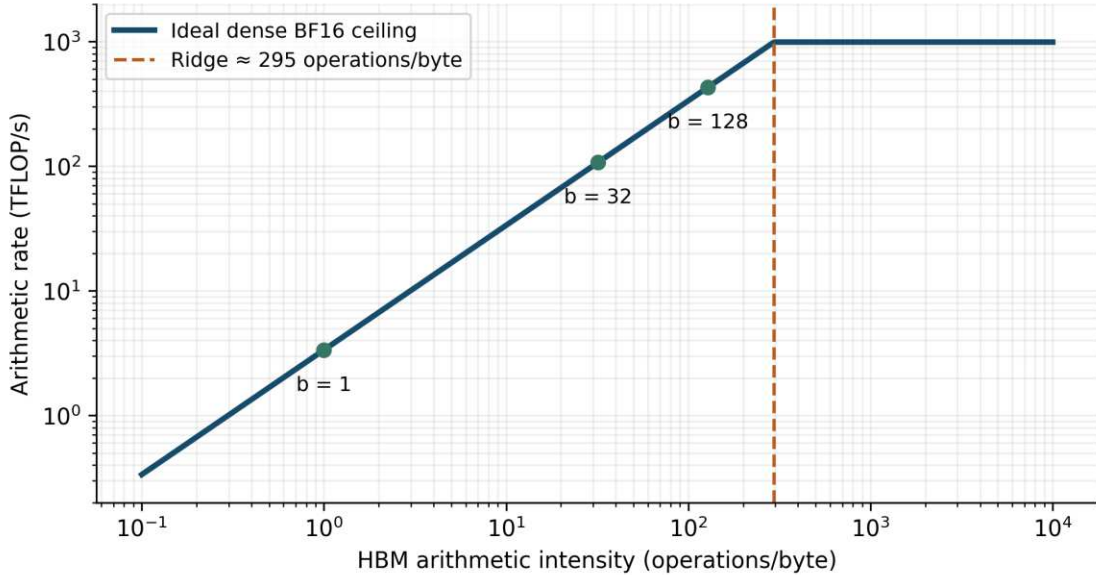


Figure 3. Original ideal Roofline calculation for the published H100 SXM bandwidth and a rounded dense BF16 tensor ceiling [13,35]. The batch mapping assumes two bytes per weight and weight-only traffic. Curve positions are derived ceilings, not measurements.

The sensitivity result is more general than the numerical example. If compute and bandwidth utilization factors are η_C and η_B , the ridge shifts by their ratio: $I^* = (\eta_C/\eta_B)(C_{\text{peak}}/B_{\text{peak}})$. Lower effective bandwidth moves the crossover to higher intensity. Lower compute efficiency moves it to lower intensity. Neither shift alone establishes improved performance; both ceilings may have fallen.

9.4 Why adding arithmetic can fail

Define $r = T_C/T_M$, where $T_C = F/C$ and $T_M = D/B$. If compute capacity increases by factor k without changing any other term, the ideal resource-bound speedup is

$$S_C(k) = \frac{\max(r, 1)}{\max(r/k, 1)}. \quad (24)$$

For $r \leq 1$, this speedup is exactly one. This is an algebraic consequence of the model, not a new empirical law. The useful design response is to reduce traffic or raise effective bandwidth, unless another neglected term dominates. Figure 4 compares this result with the analogous bandwidth upgrade.

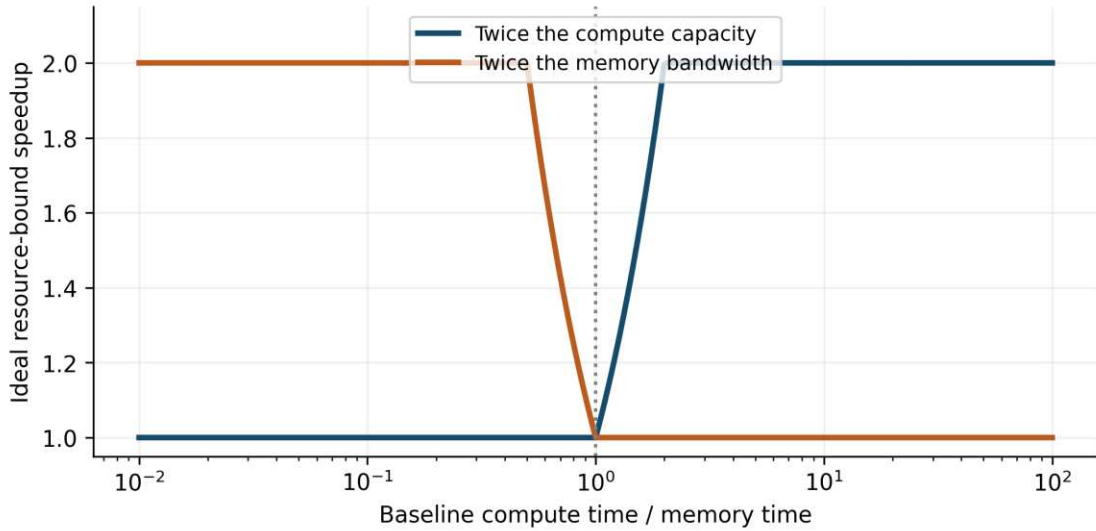


Figure 4. Original normalized sensitivity analysis of the maximum-of-resources model. A twofold improvement in a non-dominant resource can yield no time reduction. At the crossover, improving only one of two equally limiting resources leaves the other as the bound.

This result also explains why a technical review should resist a single universal hardware ranking. A device optimized for one workload phase can be poorly balanced for another. An improved compiler can move the workload between regimes by changing traffic, while an interface upgrade can be irrelevant if fixed synchronization latency dominates.

10 Energy sustainability and infrastructure

10.1 Energy accounting at the appropriate boundary

A physically meaningful first-order job-energy model separates arithmetic, memory, communication and baseline consumption:

$$E_{\text{IT}} \approx F e_{\text{op}} + \sum_j D_j e_j + E_{\text{control}} + P_0 T. \quad (25)$$

The energy coefficients refer to distinct operations or transfer boundaries and must not double-count shared circuitry. They depend on technology, voltage, precision and utilization. Historical energy-per-operation values are useful for architectural reasoning [3], but they should not be transplanted unchanged into a modern accelerator. Equation (25) is a decomposition, not a calibrated power simulator.

The facility boundary includes cooling and electrical overhead. Power usage effectiveness is the ratio of total facility energy to IT energy over a consistent interval [30]:

$$\text{PUE} = \frac{E_{\text{facility}}}{E_{\text{IT}}}, \quad E_{\text{facility}} = \text{PUE} E_{\text{IT}}. \quad (26)$$

Applying an annual site-average PUE to a short incremental workload is only an allocation approximation. PUE does not measure algorithmic efficiency, carbon intensity or useful output. A facility can have a favorable PUE while performing unnecessary computation. For a completed quality-constrained task, the relevant operational metric is $E_{\text{facility}}/N_{\text{useful}}$, with the denominator explicitly defined.

The IEA estimates data center electricity consumption at approximately 415 TWh in 2024 and projects approximately 945 TWh in 2030 in its 2025 Base Case [25]. These are all-data-center quantities, not measurements of AI alone, and the latter is a scenario projection rather than an observed future value. Their relevance is the scale of infrastructure planning; they cannot be used to calculate one model's energy without an allocation method.

10.2 Utilization and the cost of idle infrastructure

Assume a simplified power curve $P(u) = P_0 + u(P_1 - P_0)$ and useful throughput $R(u) = uR_1$, for utilization $0 < u \leq 1$. Then

$$\frac{E}{N_{\text{useful}}} = \frac{P_0}{uR_1} + \frac{P_1 - P_0}{R_1}. \quad (27)$$

The first term grows as utilization falls. This symbolic result supports consolidation, but not saturation: high utilization can violate tail-latency constraints. The linear assumptions are illustrative and must be replaced by measured curves for purchasing decisions. Figure 5 makes the idle-power penalty visible without assigning a real power curve to any server.

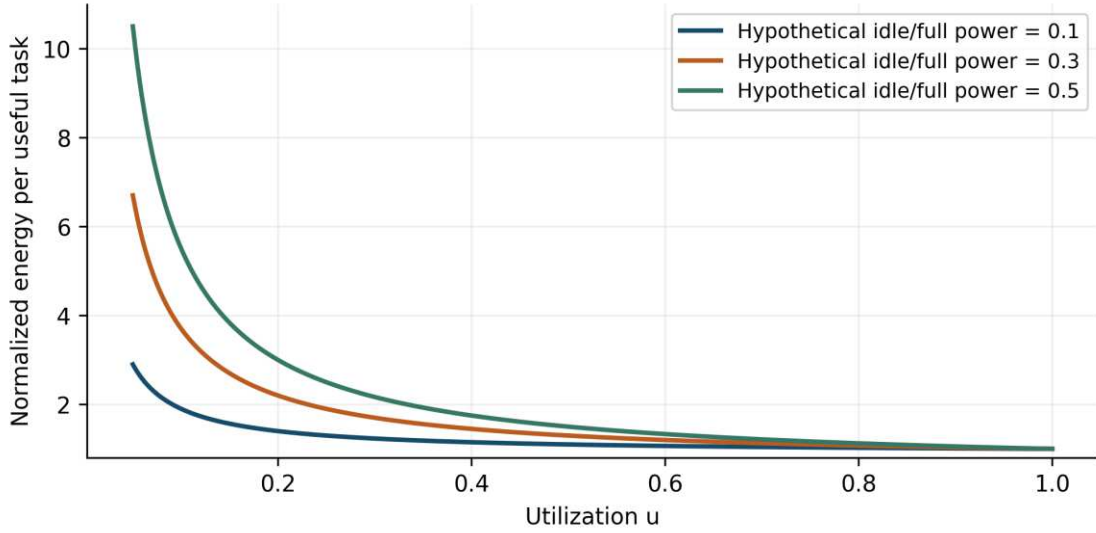


Figure 5. Original analytical curves from (27), normalized by full-load energy per useful task. The values $P_0/P_1 = 0.1, 0.3$ and 0.5 are explicitly hypothetical sensitivity parameters, not measured server characteristics. The plot assumes throughput proportional to utilization and excludes queueing constraints.

Efficiency improvements also face demand effects. If each task uses less energy but the number of tasks grows sufficiently, total energy can increase. This does not negate efficiency; it means that energy per task and aggregate environmental impact answer different questions. Both should be reported where possible.

10.3 Heat removal and electrical supply

A first-order steady thermal model is $T_j = T_c + PR_\theta$, with coolant or ambient reference temperature T_c and thermal resistance R_θ . Transient workloads require thermal capacitance and spatially resolved hotspots. Throttling can reduce sustained arithmetic throughput even when a kernel is otherwise compute-limited. More devices in a rack can also exceed the power distribution capacity before their aggregate arithmetic capability is useful.

For a coolant mass flow $\dot{m}$, heat capacity c_p and temperature rise ΔT , the heat carried is

$$\dot{Q} = \dot{m}c_p\Delta T. \quad (28)$$

This heat balance does not by itself select a cooling system. Pump energy, pressure drop, fluid compatibility, leak management, redundancy and final heat rejection remain necessary. Liquid cooling can address high heat flux, but neither direct-to-chip liquid loops nor immersion automatically guarantee lower total energy or water use. A dry heat-rejection design and an evaporative design can exchange water consumption for electricity under different climates.

Electrical infrastructure includes the utility connection, transformers, switchgear, UPS, distribution and local voltage conversion. Its efficiency depends on load and redundancy. Backup generation can alter operational emissions during outages. For a Dominican or Caribbean university installation, ambient conditions, reliable power, maintenance capacity and network availability should be evaluated explicitly. No local electricity tariff, carbon factor or outage frequency is assumed in this review.

10.4 Carbon water and replacement decisions

Lifecycle emissions include operational and embodied components [26,27]. With consistent units and accounting boundaries,

$$C_{\text{life}} = C_{\text{emb}} + \int_0^H P_{\text{facility}}(t) \gamma(t) dt + C_{\text{end}}. \quad (29)$$

Here $\gamma(t)$ is the selected electricity emission factor, H the assessment horizon and C_{end} end-of-life emissions or credits under an explicit allocation method. Average grid factors support inventory accounting; marginal factors address a different question about changes in electricity demand. They should not be mixed silently. Manufacturing emissions must be allocated across lifetime services rather than counted again in every task.

Suppose replacing existing equipment causes additional embodied emissions $\Delta C_{\text{emb}} > 0$ and saves ΔE kWh per useful task at unchanged quality. With constant carbon factor $\gamma > 0$, the simplified break-even task count is

$$N_{\text{break}} = \frac{\Delta C_{\text{emb}}}{\Delta E \gamma}. \quad (30)$$

If fewer tasks are expected over the remaining horizon, the operational carbon savings alone do not repay that increment. The expression excludes salvage, changed service demand and time-varying grid factors; those require the full lifecycle model. It is not a financial payback calculation.

Water accounting should distinguish direct cooling water, electricity-supply water and manufacturing water. Withdrawal and consumption are not interchangeable. A low-carbon operating location can still be water-stressed, so optimizing carbon alone is insufficient. The sustainable choice is consequently a constrained tradeoff among useful service, energy, emissions, water, reliability and equipment life.

11 Discussion and research implications

11.1 Cross-layer conclusions

The analysis produces three conditional conclusions. First, the architectural value of additional arithmetic is determined by intensity and dependencies, not by peak operations alone. Second, memory capacity and bandwidth are coupled but distinct: a model that fits through pooling can remain slow, and a bandwidth-rich device can be unusable if cache requirements exceed capacity. Third, energy and carbon are not interchangeable: reducing runtime or electricity does not automatically recover manufacturing emissions.

These conclusions change how designs should be compared. A proposed accelerator should state which bytes and operations it avoids, which phase it accelerates, and which overheads remain outside its boundary. An optical or analog core should include conversion and memory costs. A distributed method should include exposed communication and imbalance. A compiler improvement should preserve numerical quality and account for shape coverage. A cooling improvement should identify whether it reduces facility electricity, increases feasible heat density, or both.

The novelty claimed here is limited to the organization and coupling of these questions into a traceable decision sequence. Roofline, queueing identities, collective cost models and lifecycle break-even analysis are established concepts. The review does not rename them as new laws. Its potential value is that a single candidate improvement can be followed from transistor or dataflow mechanism to application feasibility and environmental boundary.

11.2 A reproducible comparison protocol without a new prototype

A literature-based architectural comparison can be strengthened without building hardware. For each source, an evidence matrix should record workload, numerical precision, sparsity, model quality, batch, context length, device count, topology, software version, measured boundary and whether the number is a

measurement or specification. A missing field should remain unknown. Public artifacts can support independent recalculation; unavailable proprietary measurements should not be reconstructed as facts.

For the worked example in this article, reproduction requires only the two published ceilings, the declared dense convention and equations (21–24). Appendix A supplies the numerical steps and plotting definitions. Those checks verify arithmetic and internal consistency; they do not validate actual H100 performance. The distinction is important for a review submitted without new experiments.

Future analytical work could replace the maximum-of-resources bound with a dependency graph whose nodes represent kernels and communication. A phase-specific model could then estimate critical paths while retaining explicit capacity constraints. Public traces could eventually calibrate it, but such calibration is future work, not a result reported here. A further useful direction is uncertainty propagation: effective bandwidth, useful lifetime and electricity factors should be treated as ranges when reliable point estimates are unavailable.

11.3 Limitations

The targeted search is not exhaustive and does not justify a “complete state of the art” claim. Several sources are vendor documents, and the numerical example uses their stated ceilings without independent measurement. The workload reduction neglects nonlinear interactions, detailed cache behavior, scheduling and thermal transients. The framework cannot predict p99 latency from mean throughput. Its environmental expressions require local and lifecycle inputs that are not provided here.

The scope also emphasizes neural computation and transformer-serving mechanisms. Symbolic AI, large graph workloads, retrieval systems and specialized real-time pipelines require additional workload models. Emerging analog, photonic and neuromorphic approaches are discussed as alternatives with different evidence maturity, not as demonstrated replacements for every GPU workload. These limitations bound the conclusions and prevent conceptual analysis from being mistaken for experimental validation.

12 Conclusions

Fast AI is a property of a coordinated computing system. Transistor switching, interconnect resistance, memory organization, numerical representation, kernel mapping, communication topology and service scheduling jointly determine whether arithmetic capability becomes useful throughput. The appropriate architecture is the one that satisfies capacity and quality requirements while reducing the dominant exposed cost.

The derived crossover conditions show why more arithmetic can have zero idealized speedup, why batching can be limited by cache capacity, and why a faster replacement can remain unfavorable in lifecycle carbon terms. Sustainable acceleration therefore requires phase-specific performance metrics, explicit environmental boundaries and attention to utilization and useful completion. A critical review can establish these relationships without claiming a new prototype; subsequent empirical studies would be needed to estimate their magnitude for any particular deployment.

Declarations

Study design and ethics. This manuscript reports a literature review and analytical derivations. It contains no recruitment, human or animal experimentation, personal dataset or prototype measurements. No institutional ethics decision is claimed.

Data availability. The source identifiers, assumptions and numerical derivations used in the illustrative analysis are included in the text, references and Appendix A. No experimental dataset was generated.

Use of generative AI. OpenAI ChatGPT was used to assist source discovery, manuscript drafting, analytical calculations and original figure preparation. This assistance does not constitute authorship. The named author must critically verify and approve the scientific content and references before submission.

Author contributions

Baldo Dalporto: Conceptualization, Formal analysis, Methodology, Visualization, Writing – original draft, Writing – review and editing, Project administration. Sabine Mary: Visualization, Writing – review and editing. Santiago Gallur: Supervision, Project administration, Writing – review and editing. All authors approved the submitted version and agree to be accountable for the work.

Funding

The authors declare that no financial support was received for the research, authorship, and/or publication of this article.

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

References

- [1] Sze V, Chen YH, Yang TJ, Emer JS (2017) Efficient Processing of Deep Neural Networks: A Tutorial and Survey. *Proceedings of the IEEE* 105:2295–2329. <https://doi.org/10.1109/JPROC.2017.2761740>
- [2] Williams S, Waterman A, Patterson D (2009) Roofline: An Insightful Visual Performance Model for Multicore Architectures. *Communications of the ACM* 52(4):65–76. <https://doi.org/10.1145/1498765.1498785>
- [3] Horowitz M (2014) Computing’s energy problem (and what we can do about it). *IEEE International Solid-State Circuits Conference*, pp 10–14. <https://doi.org/10.1109/ISSCC.2014.6757323>
- [4] Dennard RH, Gaensslen FH, Yu HN, Rideout VL, Bassous E, LeBlanc AR (1974) Design of ion-implanted MOSFET’s with very small physical dimensions. *IEEE Journal of Solid-State Circuits* 9:256–268. <https://doi.org/10.1109/JSSC.1974.1050511>
- [5] TSMC (n.d.) Late News: 2nm Platform Technology featuring Energy-efficient Nanosheet Transistors and Interconnects co-optimized with 3DIC for AI, HPC and Mobile SoC Applications. Official research record. <https://research.tsmc.com/page/transistor-structure/59.html> Accessed 25 September 2026.
- [6] Vaswani A et al. (2017) Attention Is All You Need. *Advances in Neural Information Processing Systems* 30. Public manuscript: <https://arxiv.org/abs/1706.03762>
- [7] Dao T, Fu DY, Ermon S, Rudra A, Ré C (2022) FlashAttention: Fast and Memory-Efficient Exact Attention with IO-Awareness. *Advances in Neural Information Processing Systems* 35. Public manuscript: <https://arxiv.org/abs/2205.14135>
- [8] Kwon W et al. (2023) Efficient Memory Management for Large Language Model Serving with PagedAttention. *Proceedings of the 29th Symposium on Operating Systems Principles*. Public manuscript: <https://arxiv.org/abs/2309.06180>
- [9] Dettmers T, Lewis M, Belkada Y, Zettlemoyer L (2022) LLM.int8(): 8-bit Matrix Multiplication for Transformers at Scale. *Advances in Neural Information Processing Systems* 35. Public manuscript: <https://arxiv.org/abs/2208.07339>
- [10] Rajbhandari S, Rasley J, Ruwase O, He Y (2020) ZeRO: Memory Optimizations Toward Training Trillion Parameter Models. *SC20 International Conference for High Performance Computing, Networking, Storage and Analysis*. Public manuscript first posted 2019: <https://arxiv.org/abs/1910.02054>
- [11] Narayanan D et al. (2021) Efficient Large-Scale Language Model Training on GPU Clusters Using Megatron-LM. *International Conference for High Performance Computing, Networking, Storage and Analysis*. Public manuscript: <https://arxiv.org/abs/2104.04473>
- [12] Jouppi NP et al. (2023) TPU v4: An Optically Reconfigurable Supercomputer for Machine Learning with Hardware Support for Embeddings. *International Symposium on Computer Architecture*. Public manuscript: <https://arxiv.org/abs/2304.01433>
- [13] NVIDIA (n.d.) H100 Tensor Core GPU. Official product specifications, H100 SXM column. <https://www.nvidia.com/en-us/data-center/h100/> Accessed 25 September 2026.
- [14] PCI-SIG (n.d.) PCI Express 6.0 Specification. Official specification overview. <https://pcisig.com/pci-express-6.0-specification> Accessed 25 September 2026.
- [15] PCI-SIG (2025) PCI Express Base Specification Revision 7.0. Official specification listing, release 11 June 2025. <https://pcisig.com/specifications> Accessed 25 September 2026.
- [16] CXL Consortium (2022) Compute Express Link 3.0 White Paper. https://computeexpresslink.org/wp-content/uploads/2023/12/CXL_3.0_white-paper_FINAL-1.pdf
- [17] CXL Consortium (n.d.) Compute Express Link 4.0 specification overview. Official consortium website. <https://computeexpresslink.org/> Accessed 25 September 2026.

- [18] UCIE Consortium (2025) UCIE 3.0 Specification: Redefining Chiplet Interconnects. 3 September 2025. <https://www.uciexpress.org/post/ucie-3-0-specification-redefining-chiplet-interconnects>
- [19] NVIDIA (n.d.) Collective Operations. NCCL User Guide. <https://docs.nvidia.com/deeplearning/nccl/user-guide/docs/usage/collectives.html> Accessed 25 September 2026.
- [20] OpenXLA Project (n.d.) XLA architecture. Official compiler documentation. <https://openxla.org/xla/architecture> Accessed 25 September 2026.
- [21] Ielmini D, Wong HSP (2018) In-memory computing with resistive switching devices. *Nature Electronics* 1:333–343. <https://doi.org/10.1038/s41928-018-0092-2>
- [22] Sebastian A, Le Gallo M, Khaddam-Aljameh R, Eleftheriou E (2020) Memory devices and applications for in-memory computing. *Nature Nanotechnology* 15:529–544. <https://doi.org/10.1038/s41565-020-0655-z>
- [23] Feldmann J et al. (2021) Parallel convolutional processing using an integrated photonic tensor core. *Nature* 589:52–58. <https://doi.org/10.1038/s41586-020-03070-1>
- [24] Intel Labs (n.d.) Taking Neuromorphic Computing to the Next Level with Loihi 2. Technology brief. <https://www.intel.com/content/www/us/en/research/neuromorphic-computing-loihi-2-technology-brief.html> Accessed 25 September 2026.
- [25] International Energy Agency (2025) Energy and AI. IEA, Paris. <https://www.iea.org/reports/energy-and-ai>
- [26] Gupta U et al. (2021) Chasing Carbon: The Elusive Environmental Footprint of Computing. IEEE International Symposium on High-Performance Computer Architecture. Public manuscript first posted 2020: <https://arxiv.org/abs/2011.02839>
- [27] Masanet E, Shehabi A, Lei N, Smith S, Koomey J (2020) Recalibrating global data center energy-use estimates. *Science* 367:984–986. <https://doi.org/10.1126/science.aba3758>
- [28] Reddi VJ et al. (2020) MLPerf Inference Benchmark. ACM/IEEE International Symposium on Computer Architecture. <https://doi.org/10.1109/ISCA45697.2020.00045> Public manuscript: <https://arxiv.org/abs/1911.02549>
- [29] Zhou Z et al. (2024) A Survey on Efficient Inference for Large Language Models. arXiv preprint, version 3. <https://doi.org/10.48550/arXiv.2404.14294>
- [30] The Green Grid (n.d.) Power Usage Effectiveness (PUE). Official metric definition. <https://www.thegreengrid.org/node/372> Accessed 25 September 2026.
- [31] Amdahl GM (1967) Validity of the single processor approach to achieving large scale computing capabilities. AFIPS Spring Joint Computer Conference, pp 483–485. <https://doi.org/10.1145/1465482.1465560>
- [32] Little JDC (1961) A Proof for the Queuing Formula: $L = \lambda W$. *Operations Research* 9:383–387. <https://doi.org/10.1287/opre.9.3.383>
- [33] Landauer R (1961) Irreversibility and Heat Generation in the Computing Process. *IBM Journal of Research and Development* 5:183–191. <https://doi.org/10.1147/rd.53.0183>
- [34] Ainslie J et al. (2023) GQA: Training Generalized Multi-Query Transformer Models from Multi-Head Checkpoints. Proceedings of EMNLP 2023. Public manuscript: <https://arxiv.org/abs/2305.13245>
- [35] NVIDIA (2022) NVIDIA Hopper Architecture In-Depth. NVIDIA Technical Blog, 22 March 2022. <https://developer.nvidia.com/blog/nvidia-hopper-architecture-in-depth/>
- [36] NVIDIA (n.d.) RDMA over Converged Ethernet — RoCE. Cumulus Linux 5.7 documentation. <https://docs.nvidia.com/networking-ethernet-software/cumulus-linux-57/Layer-1-and-Switch-Ports/Quality-of-Service/RDMA-over-Converged-Ethernet-RoCE/> Accessed 25 September 2026.

Appendix A Reproduction of analytical figures

All five figures were generated specifically for this manuscript. Diagrams describe relationships, and plots evaluate the equations printed in the article. They contain no experimental data or digitized measurements from other publications.

The Roofline figure uses decimal units, $C = 989 \times 10^{12}$ operations/s and $B = 3.35 \times 10^{12}$ bytes/s. For intensity I from 0.1 to 10,000 operations/byte, the plotted ordinate is $\min(C, IB)/10^{12}$. The ridge is 295.2239 operations/byte. The illustrative 7-billion-parameter weight store contains 14×10^9 bytes. Dividing by B gives 0.00417910 s; dividing 14×10^9 operations by C gives 0.0000141557 s. These are independent resource times; adding neglected terms can only weaken the idealized performance claim within this fixed workload description.

The message-delay figure evaluates $1 + x$, $1 + x/2$ and $0.5 + x$ for x from 0.01 to 100. The sensitivity figure evaluates $\max(r, 1)/\max(r/2, 1)$ for a compute upgrade, and $\max(r, 1)/\max(r, 1/2)$ for a bandwidth upgrade, with r from 0.01 to 100. No measured latency or bandwidth is needed for these dimensionless identities.

The energy figure uses the normalized expression $\epsilon(u) = a/u + (1 - a)$, where $a = P_0/P_1$ takes the hypothetical values 0.1, 0.3 and 0.5, and utilization runs from 0.05 to 1. At full utilization, every curve equals one by definition. This normalization is a mathematical comparison of assumptions, not a claim that three actual servers share the same full-load efficiency.

Appendix B Evidence and interpretation checks

Claim or quantity	Evidence category	Permitted conclusion
H100 memory and sparse tensor specifications	Manufacturer documentation [13,35]	Nominal ceilings for the specified variant
Approximately 4.18 ms weight-read time	Calculation from stated workload and ceiling	Ideal resource time, not observed response time
295 operations per byte crossover	Algebra from the same ceilings	Ridge of the simplified Roofline model
FlashAttention and PagedAttention mechanisms	Published research [7,8]	Traffic scheduling and memory-allocation mechanisms
945 TWh in 2030	IEA Base Case projection [25]	Scenario for all data centers, not AI-only observation
Replacement carbon threshold	Conditional derivation in equation (30)	Break-even under fixed quality and accounting assumptions
Superiority of a proposed device or deployment	Not established by this study	Requires comparable external evidence or new evaluation

Table 4. Boundaries between published information, mathematical deduction and claims not supported by this review.