The CommonCompute Network:

A Distributed Thermodynamic & Algorithmic Efficiency Framework for Decentralized Intelligence

CommonCompute Research
https://commoncompute.org
n@commoncompute.org

Abstract

The concentration of artificial-intelligence inference within a small number of hyperscale data centres creates systemic risks: single points of failure, energy hotspots, vendor lock-in, and an asymptotic cost structure that favours capital incumbents over the broader research community. We present **CommonCompute**, a decentralised inference network formalised as a directed graph of heterogeneous micro-data-centre nodes. The framework rests on four interlocking theoretical results: (i) an *Algorithmic Deflation Theorem* showing that the FLOPS required for a fixed accuracy target decay exponentially under continued algorithmic and quantisation improvements; (ii) a *Specialisation-Aware Routing* protocol that minimises a Global Compute Loss by matching task embeddings to node capability tensors; (iii) a *Federated Intelligence Manifold* that aggregates community-driven gradient updates with reputation-weighted convergence guarantees; and (iv) an *Entropy-Maximised Thermal Distribution* constraint that prevents energy concentration by maximising the Shannon entropy of the network's power profile. These four components are unified in a constrained multi-objective optimisation whose Pareto front is shown to dominate the centralised baseline in the throughput–energy–intelligence space. We provide convergence analysis, complexity bounds, and a comparative cost model, and outline a roadmap for empirical validation on the CommonCompute testnet.

1 Introduction

The global AI inference market is projected to consume hundreds of gigawatts of electrical power by the end of the decade, with the overwhelming majority routed through fewer than a dozen hyperscale operators [1, 2]. This concentration creates three compounding risks:

1. **Thermodynamic risk.** Spatially concentrated workloads produce thermal hotspots whose cooling overhead grows super-linearly, eroding net energy efficiency [3].
2. **Economic risk.** Centralised providers capture monopoly rents; the cost to query a frontier model is set by the provider, not by the marginal cost of compute [4].
3. **Resilience risk.** A single outage can cascade across millions of dependent services, and vendor lock-in constrains algorithmic diversity.

Simultaneously, algorithmic efficiency is improving at a remarkable pace. Recent studies estimate that the compute required to reach a fixed performance level on language-modelling benchmarks halves roughly every 8–12 months [5, 6], and quantisation to INT4/INT8 can reduce memory and FLOPS by 4–8× with negligible accuracy loss [7, 8]. This *algorithmic deflation* implies that the hardware cost of useful intelligence is falling faster than most infrastructure models assume.

Thesis. We argue that the optimal topology for AI inference is not a small number of large clusters but a *distributed mesh of specialised, thermodynamically balanced micro-data-centres* whose collective intelligence improves continuously via federated community feedback.

Contributions.
1. We formalise the network as a directed graph and define four theorems that govern its behaviour (§4).
2. We unify these theorems into a single constrained multi-objective optimisation (§5).
3. We provide convergence and complexity analysis (§6).
4. We present a comparative cost model against the centralised baseline (§7).

2 Related Work

Federated Learning. McMahan et al. [9] introduced FedAvg, which aggregates locally trained model updates on a central server without sharing raw data. Subsequent work has extended FL to non-IID settings [10], asynchronous protocols [11], and fully decentralised topologies without a central coordinator [12]. CommonCompute builds on the decentralised FL paradigm but introduces reputation-weighted aggregation driven by verified community usage.

Decentralised Inference. Petals [13] demonstrated that large language models can be served collaboratively across volunteer GPUs by pipelining transformer blocks. BOINC [14] pioneered volunteer distributed computing for scientific workloads. Unlike these systems, CommonCompute introduces a formal routing optimisation that matches tasks to *specialised* nodes rather than treating nodes as fungible.

Mixture-of-Experts & Routing. Shazeer et al. [15] introduced sparsely-gated MoE layers where a learned router selects a subset of expert sub-networks. Our specialisation-aware routing (§4.2) generalises this idea from intra-model expert selection to *inter-node* task assignment, using hardware-scaled cosine similarity as the gating signal.

Thermodynamic-Aware Scheduling. Energy-proportional computing [16] and thermal-aware job placement [17] have been studied in data-centre contexts. We formalise the thermal constraint as a Shannon-entropy maximisation over the network's energy distribution, which to our knowledge has not been proposed in the decentralised inference literature.

DePIN Networks. Decentralised Physical Infrastructure Networks (DePIN) incentivise community-operated hardware through token rewards [18]. CommonCompute contributes a rigorous mathematical framework that complements the economic incentive layer with formal efficiency and convergence guarantees.

3 System Model & Topology

Definition 3.1 (CommonCompute Network)**.** The network is a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where each vertex $n_i \in \mathcal{V}$ represents a micro-data-centre node and each edge $(n_i, n_j) \in \mathcal{E}$ represents a communication link with bandwidth b_{ij} and latency ℓ_{ij}.

The instantaneous state of the network at time t is described by the tuple

$$\Omega(t) = \big(\mathcal{M}(t),\ \mathcal{R}(t),\ \mathcal{C}(t)\big), \tag{1}$$

where:

- $\mathcal{M}(t)$: the set of active model deployments (model identifiers, versions, quantisation levels);
- $\mathcal{R}(t) = \{(c_i, s_i, b_i)\}_{i=1}^{|\mathcal{V}|}$: the resource vector per node (compute capacity c_i in FLOPS, storage s_i, bandwidth b_i);
- $\mathcal{C}(t) = \{(T_i, P_i^{\max}, \rho_i)\}_{i=1}^{|\mathcal{V}|}$: the environmental constraint vector per node (thermal limit T_i, power budget $P_i^{\max}$, current thermal density ρ_i).

Definition 3.2 (Task)**.** An incoming inference task τ_j is characterised by a tuple $(\mathbf{v}_j, \alpha_j, d_j)$, where $\mathbf{v}_j \in \mathbb{R}^k$ is a learned embedding of the task type, α_j is the minimum acceptable accuracy, and d_j is the latency deadline.

Definition 3.3 (Specialisation Tensor)**.** Each node n_i maintains a specialisation tensor $\mathbf{S}_i \in \mathbb{R}^k$ encoding the task categories for which it has been fine-tuned, and a scalar hardware affinity factor $h_i \in (0, 1]$ reflecting accelerator suitability (e.g., INT4 throughput relative to peak).

4 Core Theorems

4.1 The Algorithmic Deflation Theorem

We first establish that the compute cost required to achieve a fixed benchmark accuracy is a *decreasing* function of time under sustained algorithmic and quantisation progress.

Assumption 4.1 (Algorithmic Improvement Rate)**.** There exists a monotonically non-decreasing function $\lambda : \mathbb{R}_{\geq 0} \to \mathbb{R}_{>0}$, termed the *algorithmic improvement rate*, such that the effective FLOPS saved per unit time by algorithmic advances (architecture improvements, training recipes, distillation, sparsity) is captured by the instantaneous rate $\lambda(t)$. Empirically, $\lambda(t)$ has been estimated at values corresponding to a doubling of efficiency every 8–12 months [5, 6].

Assumption 4.2 (Quantisation Factor)**.** For a model μ quantised from full precision (FP32) to a reduced format (e.g., INT8, INT4), the quantisation factor $Q(\mu) \in (0, 1]$ denotes the ratio of quantised FLOPS to full-precision FLOPS required for the same output quality. State-of-the-art methods achieve $Q(\mu) \approx 0.125$–0.25 with $< 1\%$ accuracy degradation [7, 8].

Definition 4.3 (Compute Cost Function)**.** Let C_0 be the baseline FLOPS required to achieve accuracy α at time $t = 0$. The *time-discounted compute cost* is

$$C_{\text{req}}(t) = C_0 \cdot e^{-\Lambda(t)} \cdot Q(\mu), \quad \Lambda(t) \triangleq \int_0^t \lambda(s)\, ds. \tag{2}$$

Theorem 4.4 (Algorithmic Deflation)**.** *Under Assumptions 4.1 and 4.2, the compute cost satisfies*

$$\frac{dC_{\text{req}}}{dt} = -\lambda(t)\, C_{\text{req}}(t) < 0 \quad \forall\, t \geq 0, \tag{3}$$

while the achievable accuracy $\alpha^*(t)$ at fixed *compute budget C is non-decreasing:*

$$\frac{d\alpha^*}{dt} \geq 0. \tag{4}$$

Proof. Differentiating (2) with respect to t:

$$\frac{dC_{\text{req}}}{dt} = C_0\, Q(\mu)\, \frac{d}{dt}\big[e^{-\Lambda(t)}\big] = -\lambda(t)\, C_0\, Q(\mu)\, e^{-\Lambda(t)} = -\lambda(t)\, C_{\text{req}}(t).$$

Since $\lambda(t) > 0$ and $C_{\text{req}}(t) > 0$, the derivative is strictly negative. The accuracy claim follows by contrapositive: if the same compute budget C sufficed for accuracy α

at time t, it suffices for at least α at any $t' > t$ because $C_{\text{req}}(t') < C_{\text{req}}(t) \leq C$. Any surplus compute can be allocated to higher-accuracy decoding or ensembling, giving $\alpha^*(t') \geq \alpha^*(t)$. □

Remark 4.5. The exponential form in (2) is a modelling choice. If algorithmic progress is better described by a power law $C_{\text{req}}(t) \propto t^{-\beta}$, the deflation result $dC_{\text{req}}/dt < 0$ still holds for $\beta > 0$. The key qualitative insight—that required FLOPS decrease monotonically—is robust to the functional form chosen.

4.2 Specialisation-Aware Routing

We now formalise how the network routes each incoming task to the node best suited for it.

Definition 4.6 (Efficiency Coefficient)**.** For task τ_j with embedding $\mathbf{v}_j$ and node n_i with specialisation tensor $\mathbf{S}_i$ and hardware factor h_i, the *efficiency coefficient* is

$$\eta_{ij} = h_i \cdot \frac{\mathbf{v}_j \cdot \mathbf{S}_i}{\|\mathbf{v}_j\| \, \|\mathbf{S}_i\|} = h_i \cdot \cos\big(\mathbf{v}_j, \mathbf{S}_i\big). \tag{5}$$

Note $\eta_{ij} \in [-h_i, h_i]$. In practice we clip to $\eta_{ij} \in [0, h_i]$ since negative alignment indicates an unsuitable node.

The effective cost of executing τ_j on node n_i is the required FLOPS scaled inversely by the efficiency coefficient:

Definition 4.7 (Effective Node Cost)**.**

$$\mathcal{L}_{ij} = \frac{C_{\text{req}}(\tau_j)}{\eta_{ij} + \epsilon}, \tag{6}$$

where $\epsilon > 0$ is a small regularisation constant preventing division by zero, and $C_{\text{req}}(\tau_j)$ is the deflated compute cost from (2) evaluated for the model serving τ_j.

Definition 4.8 (Global Compute Loss)**.** The network minimises the aggregate cost over all tasks in a scheduling window $\mathcal{T}$:

$$\mathcal{L}_{\text{net}} = \sum_{\tau_j \in \mathcal{T}} \min_{n_i \in \mathcal{V}} \mathcal{L}_{ij} = \sum_{\tau_j \in \mathcal{T}} \min_{n_i \in \mathcal{V}} \frac{C_{\text{req}}(\tau_j)}{\eta_{ij} + \epsilon}. \tag{7}$$

Proposition 4.9 (Specialisation Advantage)**.** *Let n_s be a node with $\eta_{sj} = 1$ (perfect specialisation) and n_g be a generalist node with $\eta_{gj} = \eta_0 \in (0, 1)$. Then the cost ratio satisfies*

$$\frac{\mathcal{L}_{gj}}{\mathcal{L}_{sj}} = \frac{1 + \epsilon}{\eta_0 + \epsilon} \xrightarrow{\epsilon \to 0} \frac{1}{\eta_0}. \tag{8}$$

For a typical generalist $\eta_0 = 0.1$, the specialised node is $\approx 10\times$ more cost-efficient.

Proof. Immediate from the ratio of (6) evaluated at $\eta_{sj} = 1$ and $\eta_{gj} = \eta_0$. □

Routing as an Assignment Problem. In the batch setting (all tasks in $\mathcal{T}$ are known), the minimisation in (7) is an instance of the *minimum-cost bipartite matching* problem between tasks and nodes (with capacity constraints). This can be solved in $O(|\mathcal{T}|^2|\mathcal{V}|)$ time using the Hungarian algorithm [20] when $|\mathcal{T}| \leq |\mathcal{V}|$, or via successive shortest paths otherwise. In the online setting, a greedy policy that assigns each arriving task to its best available node achieves a 2-approximation to the offline optimum under standard assumptions [21].

4.3 The Federated Intelligence Manifold

CommonCompute treats the network's collective intelligence as a continuously evolving parameter vector $\boldsymbol{\theta}_{\text{global}}(t) \in \mathbb{R}^d$, updated by aggregating community-derived gradient signals.

Definition 4.10 (Reputation Weight)**.** Node n_i accumulates a reputation weight $w_i(t) \in [0, 1]$ that is a monotonically non-decreasing function of the volume of *distinct, verified* feedback $f_i(t)$ contributed by the community through that node:

$$w_i(t) = \frac{f_i(t)}{\sum_{k=1}^{|\mathcal{V}|} f_k(t)}. \tag{9}$$

Definition 4.11 (Federated Update Rule)**.** Given local gradient updates $\nabla_{\boldsymbol{\theta}_i}$ computed at each participating node n_i using its local data shard, the global model is updated as:

$$\boldsymbol{\theta}_{\text{global}}^{(t+1)} = \boldsymbol{\theta}_{\text{global}}^{(t)} + \gamma \sum_{i=1}^{N} w_i(t) \cdot \nabla_{\boldsymbol{\theta}_i}, \tag{10}$$

where $\gamma > 0$ is the global learning rate and $N = |\mathcal{V}|$ is the number of participating nodes.

Remark 4.12. When $w_i = n_i / \sum n_k$ (proportional to local data size), this reduces to the standard FedAvg aggregation [9]. Our formulation generalises FedAvg by weighting according to verified community engagement rather than raw data volume, aligning incentives toward high-quality feedback.

Assumption 4.13 (Smoothness and Bounded Variance)**.** The global loss function $F(\boldsymbol{\theta})$ is L-smooth, and the stochastic local gradients satisfy $\mathbb{E}[\nabla_{\boldsymbol{\theta}_i}] = \nabla F_i(\boldsymbol{\theta})$ with bounded variance $\mathbb{E}[\|\nabla_{\boldsymbol{\theta}_i} - \nabla F_i(\boldsymbol{\theta})\|^2] \leq \sigma^2$. Furthermore, the inter-node gradient divergence is bounded: $\|\nabla F_i(\boldsymbol{\theta}) - \nabla F(\boldsymbol{\theta})\|^2 \leq \delta^2$ for all i.

Theorem 4.14 (Convergence of Reputation-Weighted Aggregation)**.** *Under Assumption 4.13, with learning rate $\gamma = O(1/\sqrt{T})$, the iterates (10) satisfy*

$$\frac{1}{T} \sum_{t=0}^{T-1} \mathbb{E}\big[\|\nabla F(\boldsymbol{\theta}^{(t)})\|^2\big] \leq O\!\left(\frac{L\,(F_0 - F^*)}{\sqrt{T}} + \frac{\sigma^2}{\sqrt{T}} + \delta^2\,\Phi_w \right), \tag{11}$$

where $F_0 = F(\boldsymbol{\theta}^{(0)})$, F^* *is the global minimum, and*

$$\Phi_w = \sum_{i=1}^{N} w_i^2 \tag{12}$$

is the reputation concentration index.

Proof sketch. The proof follows the standard analysis of weighted SGD with heterogeneous data [10, 23]. The key modification is that the aggregation weights w_i introduce a bias term proportional to $\Phi_w \cdot \delta^2$. When reputations are uniform ($w_i = 1/N$), $\Phi_w = 1/N$ and we recover the standard $O(\delta^2/N)$ heterogeneity term. When reputation is concentrated on a single node ($w_1 = 1$), $\Phi_w = 1$, and the bound degrades to the single-node case. Thus the protocol benefits from a diverse, well-distributed reputation landscape. The complete proof proceeds by unrolling the recursion and applying the co-coercivity of L-smooth functions; see Appendix (forthcoming). □

The Positive Feedback Loop. The reputation mechanism creates a virtuous cycle:

$$\text{Usage}\uparrow \Rightarrow\ w_i\uparrow \Rightarrow\ \text{Model Quality}\uparrow \Rightarrow\ \text{Usage}\uparrow. \tag{13}$$

To prevent this from degenerating into monopolistic capture by a single high-reputation node, we impose a reputation cap $w_i \leq w_{\max} < 1$ and redistribute excess weight uniformly, ensuring Φ_w remains bounded away from 1.

4.4 Entropy-Maximised Thermal Distribution

We formalise the goal of preventing energy hotspots as an entropy-maximisation problem.

Definition 4.15 (Energy Distribution)**.** Let $E_i(t)$ be the energy consumed by node n_i in the scheduling window. The normalised energy distribution is

$$P_i(t) = \frac{E_i(t)}{\sum_{k=1}^{N} E_k(t)}, \qquad \sum_{i=1}^{N} P_i = 1. \tag{14}$$

Definition 4.16 (Network Energy Entropy)**.** The Shannon entropy of the energy distribution is

$$H(\mathbf{E}) = -\sum_{i=1}^{N} P_i \log P_i. \tag{15}$$

This is maximised at $H^* = \log N$ when $P_i = 1/N$ (uniform distribution), corresponding to perfectly balanced energy consumption.

Definition 4.17 (Thermal Density Constraint)**.** The *thermal density* at node n_i is defined as

$$\rho_i(t) = \frac{E_i(t)}{V_i \cdot \Delta t}, \tag{16}$$

where V_i is the effective cooling volume. The system enforces

$$\rho_i(t) < \rho_{\text{critical}} \quad \forall\, i \in \mathcal{V}. \tag{17}$$

Proposition 4.18 (Entropy Penalty in Routing)**.** *Incorporating the entropy objective as a Lagrangian penalty into the routing cost, the modified per-assignment cost becomes*

$$\tilde{\mathcal{L}}_{ij} = \frac{C_{\text{req}}(\tau_j)}{\eta_{ij} + \epsilon} - \mu\, \frac{\partial H}{\partial E_i}\, C_{\text{req}}(\tau_j), \tag{18}$$

where $\mu > 0$ *is the entropy multiplier and*

$$\frac{\partial H}{\partial E_i} = -\frac{1}{E_{\text{total}}}\big(\log P_i + 1 - H(\mathbf{E})\big). \tag{19}$$

Routing to an already-hot node (P_i *large,* $\log P_i$ *close to 0) incurs a higher penalty, steering traffic toward cooler nodes.*

Proof. By the chain rule, $\partial H/\partial E_i = (\partial H/\partial P_i)(\partial P_i/\partial E_i)$. Since $\partial H/\partial P_i = -(\log P_i + 1)$ and $\partial P_i/\partial E_i = (E_{\text{total}} - E_i)/E_{\text{total}}^2 \approx 1/E_{\text{total}}$ for large N, the result follows. □

Remark 4.19 (Centralised vs. Distributed Entropy)**.** In a centralised data centre with N_c racks, the maximum entropy is $\log N_c$. A distributed mesh with $N \gg N_c$ nodes achieves $H^*_{\text{dist}} = \log N \gg \log N_c$, providing a structurally larger feasible entropy space and thus greater thermal headroom.

5 The Grand Optimisation Objective

The CommonCompute protocol unifies the four theorems into a single constrained multi-objective optimisation.

Definition 5.1 (Network Utility Function)**.**

$$\boxed{\max_{\pi}\ J(\pi) = \underbrace{\sum_{\tau_j \in \mathcal{T}} \frac{\text{Throughput}_j(\pi)}{\text{Perf}_j(\pi)}}_{\text{Efficiency term}} + \lambda_1 \underbrace{\frac{H(\mathbf{E}(\pi))}{\log N}}_{\text{Entropy term}} + \lambda_2 \underbrace{\sum_{i=1}^{N} w_i \cdot \Delta I_i(\pi}_{\text{Intelligence gai}}} \tag{20}$$

subject to:

$$\begin{aligned}
&\text{(Latency)} && \ell(\text{user}, n_{\pi(j)}) \leq \delta_{\max} && \forall\, \tau_j \in \mathcal{T}, && (21)\\
&\text{(Thermal)} && \rho_i(t) < \rho_{\text{critical}} && \forall\, n_i \in \mathcal{V}, && (22)\\
&\text{(Capacity)} && \sum_{\tau_j:\, \pi(j)=i} C_{\text{req}}(\tau_j) \leq c_i && \forall\, n_i \in \mathcal{V}, && (23)\\
&\text{(Rep. cap)} && w_i \leq w_{\max} && \forall\, n_i \in \mathcal{V}, && (24)
\end{aligned}$$

where $\pi : \mathcal{T} \to \mathcal{V}$ is the routing policy, Throughput_j is tokens per second for task j, Perf_j is a normalised performance metric, and ΔI_i is the intelligence gain (reduction in global loss) contributed by node i's federated update.

Interpretation of Trade-off Parameters. The scalars $\lambda_1, \lambda_2 \geq 0$ control the Pareto trade-off:

- $\lambda_1 \to \infty$: the system prioritises thermal balance, distributing load uniformly even at some throughput cost.

- $\lambda_2 \to \infty$: the system prioritises intelligence improvement, routing to nodes with the highest reputation and feedback volume.
- $\lambda_1 = \lambda_2 = 0$: pure throughput maximisation, equivalent to a centralised scheduler.

Proposition 5.2 (Pareto Dominance over Centralised Baseline)**.** *Let J_C be the utility achieved by a centralised system with N_c co-located nodes ($H(\mathbf{E}) \leq \log N_c$) and no federated intelligence gain ($\Delta I_i = 0$ for external nodes). For any $\lambda_1, \lambda_2 > 0$ and sufficiently large N, there exists a routing policy π^* such that $J(\pi^*) > J_C$.*

Proof sketch. The efficiency term is comparable by Theorem 4.4 (both systems benefit from algorithmic deflation). The entropy term satisfies $H(\mathbf{E}(\pi^*))/\log N \geq H(\mathbf{E}_C)/\log N_c$ when the distributed system's entropy-aware router achieves at least the same relative utilisation uniformity. The intelligence gain term is strictly positive for the distributed system whenever $N > N_c$ nodes contribute verified feedback, while it is zero for the closed centralised system. The sum therefore exceeds J_C for any $\lambda_2 > 0$ and sufficiently many contributing nodes. $\square$

6 Convergence & Complexity Analysis

6.1 Federated Convergence

Theorem 4.14 establishes a convergence rate of $O(1/\sqrt{T})$ for the reputation-weighted aggregation, matching the optimal rate for non-convex stochastic optimisation [19]. The key insight is that maintaining a low reputation concentration index Φ_w (enforced by the cap $w_{\max}$) keeps the heterogeneity penalty small.

Corollary 6.1. *With uniform reputation ($w_i = 1/N$) and T rounds, an ϵ-stationary point ($\mathbb{E}[\|\nabla F\|^2] \leq \epsilon$) is reached in*

$$T = O\left(\frac{L^2(F_0 - F^*) + \sigma^2}{\epsilon^2} + \frac{\delta^2}{N\epsilon}\right) \tag{25}$$

communication rounds.

6.2 Routing Complexity

Proposition 6.2 (Routing is NP-hard in the capacitated case)**.** *The capacitated version of (7) with constraints (21)–(23) is NP-hard by reduction from the Generalised Assignment Problem (GAP) [22].*

In practice, the online greedy algorithm (assign each task to its best feasible node) runs in $O(|\mathcal{T}| \cdot |\mathcal{V}|)$ time per scheduling window and achieves the following guarantee:

Proposition 6.3 (Greedy Approximation)**.** *The online greedy policy achieves a total cost within a factor of $(1 + \ln|\mathcal{V}|)$ of the optimal offline solution for the uncapacitated case, and a constant-factor approximation for the capacitated case under uniform capacities.*

6.3 Entropy Constraint Satisfaction

The entropy penalty in (18) acts as a soft constraint. For hard enforcement, the router rejects any assignment $\pi(j) = i$ if $\rho_i(t) + \Delta\rho_{ij} \geq \rho_{\text{critical}}$, where $\Delta\rho_{ij}$ is the projected thermal density increase. This reduces the feasible node set for each task and may increase $\mathcal{L}_{\text{net}}$, but guarantees thermal safety.

7 Comparative Analysis

We contrast CommonCompute against a centralised hyperscaler model across five dimensions.

1. Compute Model. The hyperscaler relies on brute-force scaling: $C \propto |\text{Params}|^k$ with $k > 1$. CommonCompute leverages algorithmic deflation (§4.1): the effective cost is $C_{\text{req}}(t) = C_0 e^{-\Lambda(t)} Q(\mu)$, which decreases exponentially in the cumulative algorithmic progress $\Lambda(t)$.

2. Topology. The hyperscaler operates a low-entropy topology with energy concentrated in $\sim 10^1$–10^2 locations. CommonCompute's distributed mesh targets $\sim 10^3$–10^5 nodes with entropy $H(\mathbf{E}) \to \log N$, yielding structurally superior thermal headroom.

3. Optimisation Target. Hyperscalers maximise *utilisation per server*. CommonCompute maximises *specialised matching* (η_{ij}) and *network-wide entropy*, producing lower aggregate energy per inference.

4. Scaling Strategy. Hyperscalers add GPUs to existing clusters (vertical scaling with diminishing thermal returns). CommonCompute adds specialised nodes (horizontal scaling with increasing entropy).

5. Intelligence Improvement. Hyperscalers improve models via internal R&D cycles. CommonCompute supplements this with continuous federated updates from community usage (the intelligence gain term $\sum w_i \Delta I_i$ in (20)).

Proposition 7.1 (Asymptotic Cost Advantage)**.** *Let $E_C(t)$ be the per-inference energy cost of the centralised system and $E_D(t)$ the per-inference energy cost of the CommonCompute network, both serving the same accuracy target. Under the algorithmic deflation model, with specialisation advantage ratio η_0^{-1} and thermal overhead factor $\Theta(N_c) > 1$ for the centralised system (due to cooling costs scaling super-linearly in thermal density), we have:*

$$\frac{E_D(t)}{E_C(t)} = \frac{\eta_0}{\Theta(N_c)} \xrightarrow[N_c \text{ large}]{} 0. \tag{26}$$

That is, the distributed system's cost advantage grows as centralised clusters become thermally constrained.

8 Discussion

Limitations. Several assumptions merit scrutiny:

1. The exponential form of algorithmic deflation (2) is empirically supported for 2012–2025 but may saturate as low-hanging algorithmic improvements are exhausted.
2. The specialisation tensor $\mathbf{S}_i$ is assumed to be known and static; in practice it must be learned and updated as models evolve.
3. The reputation weight w_i relies on verified feedback, which itself requires a trust infrastructure (e.g., signed attestation, TEEs) that is non-trivial to deploy.
4. The Shannon entropy objective treats all nodes as equally desirable targets; geographic latency constraints may make some distributions infeasible even if entropy-optimal.

Open Problems.

1. *Empirical validation.* The theoretical framework must be validated via simulation and live deployment on the CommonCompute testnet. A Python-based discrete-event simulator modelling network state $\Omega(t)$, routing decisions, and federated updates is planned.
2. *Incentive compatibility.* Integrating the token-economic layer (CCT, Compute Credits) with the mathematical framework to prove that rational nodes are incentivised to report truthful specialisation tensors and provide genuine feedback.
3. *Privacy guarantees.* Augmenting the federated aggregation with differential privacy [24] or secure aggregation to prevent gradient inversion attacks while maintaining convergence rates.
4. *Dynamic $\lambda(t)$ estimation.* Developing an online estimator for the algorithmic improvement rate that adapts as the field evolves.
5. *Hardware co-design.* Formally connecting the software routing layer to the open-hardware inference accelerator being developed under the CommonCompute hardware philosophy [25].

9 Conclusion

We have presented the CommonCompute framework: a mathematically grounded architecture for decentralised AI inference that exploits algorithmic deflation, specialisation-aware routing, federated intelligence aggregation, and entropy-maximised thermal distribution. The four core theorems are unified by a constrained multi-objective optimisation whose Pareto front is shown to dominate centralised alternatives in the throughput–energy–intelligence space.

The framework provides a theoretical foundation for the CommonCompute network—a community-operated mesh of micro-data-centres with open-source hardware, federated model improvement, and token-economic incentives. Future work will validate these results empirically, integrate formal incentive-compatibility proofs, and extend the model to heterogeneous accelerator types and multi-modal workloads.

Acknowledgements. We thank the CommonCompute community contributors for discussions that shaped the system design, and the open-source AI research community whose work on efficient inference and federated learning underpins this framework.

References

[1] Epoch AI. Machine learning trends, 2025. https://epoch.ai/trends.

[2] Stanford HAI. The 2025 AI Index Report. Stanford Institute for Human-Centered AI, 2025.

[3] E. Masanet, A. Shehabi, N. Lei, S. Smith, and J. Koomey. Recalibrating global data center energy-use estimates. *Science*, 367(6481):984–986, 2020.

[4] Epoch AI. LLM inference price trends, 2025. https://epoch.ai/research/llm-inference-price-trends.

[5] A. Ho, T. Besiroglu, et al. Algorithmic progress in language models. *arXiv preprint arXiv:2403.05812*, 2024.

[6] D. Hernandez and T. B. Brown. Measuring the algorithmic efficiency of neural networks. *arXiv preprint arXiv:2005.04305*, 2020.

[7] T. Dettmers, M. Lewis, Y. Belkada, and L. Zettlemoyer. GPT3.int8(): 8-bit matrix multiplication for transformers at scale. In *NeurIPS*, 2022.

[8] E. Frantar, S. P. Ashkboos, T. Hoefler, and D. Alistarh. GPTQ: Accurate post-training quantization for generative pre-trained transformers. In *ICLR*, 2023.

[9] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas. Communication-efficient learning of deep networks from decentralized data. In *AISTATS*, 2017.

[10] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith. Federated optimization in heterogeneous networks. In *MLSys*, 2020.

[11] C. Xie, S. Koyejo, and I. Gupta. Asynchronous federated optimization. *arXiv preprint arXiv:1903.03934*, 2019.

[12] X. Lian, C. Zhang, H. Zhang, C.-J. Hsieh, W. Zhang, and J. Liu. Can decentralized algorithms outperform

centralized algorithms? A case study for decentralized parallel stochastic gradient descent. In *NeurIPS*, 2017.

[13] A. Borzunov, D. Baranchuk, T. Dettmers, et al. Petals: Collaborative inference and fine-tuning of large models. In *ACL System Demonstrations*, 2023.

[14] D. P. Anderson. BOINC: A system for public-resource computing and storage. In *GRID*, 2004.

[15] N. Shazeer, A. Mirhoseini, K. Mahdavi, et al. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *ICLR*, 2017.

[16] L. A. Barroso and U. Hölzle. The case for energy-proportional computing. *Computer*, 40(12):33–37, 2007.

[17] Q. Tang, S. K. S. Gupta, and G. Varsamopoulos. Energy-efficient thermal-aware task scheduling for homogeneous high-performance computing data centers. *IEEE Trans. Parallel Distrib. Syst.*, 19(5):672–681, 2008.

[18] Messari. The DePIN sector map, 2024. https://messari.io/report/the-depin-sector-map.

[19] S. Ghadimi and G. Lan. Stochastic first- and zeroth-order methods for nonconvex stochastic programming. *SIAM J. Optim.*, 23(4):2341–2368, 2013.

[20] H. W. Kuhn. The Hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2(1–2):83–97, 1955.

[21] R. M. Karp, U. V. Vazirani, and V. V. Vazirani. An optimal algorithm for on-line bipartite matching. In *STOC*, 1990.

[22] C. Chekuri and S. Khanna. A polynomial time approximation scheme for the multiple knapsack problem. *SIAM J. Comput.*, 35(3):713–728, 2005.

[23] S. P. Karimireddy, S. Kale, M. Mohri, S. Reddi, S. Stich, and A. T. Suresh. SCAFFOLD: Stochastic controlled averaging for federated learning. In *ICML*, 2020.

[24] M. Abadi, A. Chu, I. Goodfellow, et al. Deep learning with differential privacy. In *CCS*, 2016.

[25] CommonCompute. Hardware philosophy: Open inference silicon & systems, v0.1, 2025. https://github.com/common-compute.