

Gossamer: Distributed QoS-Aware Request Routing for Stateful Large Language Model (LLM) Inference at the Edge

Qifan Deng and Rajkumar Buyya

Quantum Cloud Computing and Distributed Systems (qCLOUDS) Laboratory
School of Computing and Information Systems
The University of Melbourne, Australia
{qifdeng,rbuyya}@unimelb.edu.au

Abstract. Large language models (LLMs) are expected to support latency-critical cyber-physical applications, whose stringent deadlines cluster-based serving cannot meet. Deploying LLM inference at the edge requires decentralized scheduling to meet such deadlines under heterogeneous, stateful resources. Centralized schedulers are unsuitable: control-plane round-trips already exceed sub-20 ms hazard-critical SLO budgets for autonomous driving. We propose **Gossamer**, a distributed scheduler where each edge node routes requests using type-weighted scoring over gossiped partial state, and **GLARE** (Gossip-driven Local Adaptive Routing Engine), a metaheuristic that uses gossip staleness to control per-node weight adaptation. We formulate the routing problem as a constrained optimization problem (DQRR), prove its NP-hardness via reduction from 3-PARTITION, and characterize three design properties. Evaluated via discrete-event simulation driven by empirical V2AIX vehicle trajectories with spatial deployment mapping and projected GPU hardware, Gossamer achieves H-type SLO attainment of 100% at 50–100% offered load, 94.3% at 200%, and 85.7% at 300% (a 29.1 pp gap at 300% over the next-best baseline). The full Gossamer system provides a 29.2 pp gain over Greedy (static per-type weights); GLARE contributes up to +3.2 pp over static-weight routing at 500% load, with gains modest at moderate overload. Decision latency measured at 153–209 μ s P99 (AMD EPYC 9474F) consumes under 1.1% of the H-type SLO budget.

Keywords: Edge computing · LLM inference · Distributed scheduling · QoS-aware routing · Gossip protocols · KV-cache · Mixture-of-experts

1 Introduction

Large language models (LLMs) now drive conversational assistants and code generation, and are entering cyber-physical systems such as autonomous driving, where they promise richer scene understanding and intent prediction [8,23]. Many of these applications are latency-critical: a driving assistance feature must reason about a pedestrian within the sense-plan-act cycle, not within a datacenter round trip. Edge deployment can meet such deadlines, but today’s serving

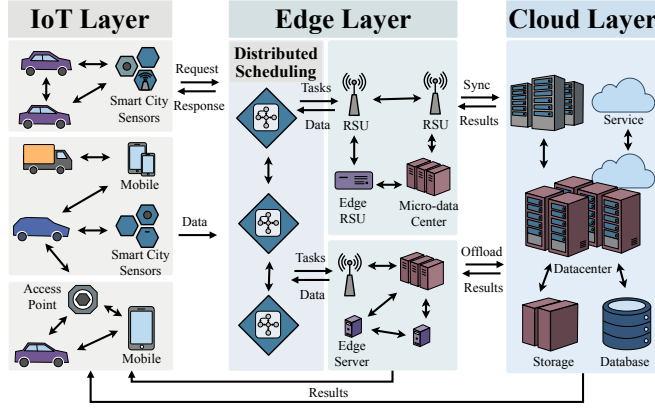


Fig. 1: Three-tier edge-cloud inference scenario.

systems are built for datacenter clusters [17] with centralized placement and batching. Across a metropolitan roadside unit (RSU) mesh, control-plane round trips already approach or exceed 20 ms before computation begins, and edge nodes hold a fraction of cluster memory, so routing must be local and partially informed. The binding target in our setting is a 20 ms hazard-critical SLO. At 130 km/h, each millisecond translates to 3.6 cm of vehicle travel, bounding the sense-plan-act cycle [11] to 0.72 m. Deploying LLM inference at the edge can satisfy this deadline, but only if the request-routing decision itself consumes a small fraction of the SLO budget. Figure 1 illustrates the three-tier infrastructure we target: vehicles at different positions along the motorway connect to their nearest RSUs wirelessly, creating spatially heterogeneous load patterns across the edge mesh. We propose **Gossamer**, a distributed scheduler using type-weighted scoring over gossiped partial state, and **GLARE**, a staleness-driven metaheuristic (Section 4.3).

We evaluate Gossamer on a projected near-future edge infrastructure overlaid on the empirical V2AIX dataset [16], which contains authentic ETSI ITS V2X messages [5] recorded on the A44 motorway (Düsseldorf–Aachen–Cologne region).¹ The deployment comprises 22 RSUs spaced at 1.0–1.8 km intervals connected via fiber and wireless V2X [5], with a cloud fallback near Aachen (hardware specifications and link parameters are detailed in Section 3). Each RSU co-locates an edge compute node housing GPU, memory, and LLM-serving infrastructure; we use *RSU* for the V2X gateway and *edge node* for the co-located compute entity. Using empirical vehicle trajectories, RSU positions, and V2X message patterns eliminates reliance on synthetic mobility models.

Vehicles (IoT layer) issue LLM inference requests wirelessly to roadside units (edge layer), and requests exceeding edge capacity escalate to the cloud fallback (Figure 1). Four distinct LLM-powered applications share these edge resources, each characterized by a specific QoS profile:

¹ Dataset and documentation: <https://v2aix.ika.rwth-aachen.de>.

- *Hazard-critical (H)*: Emergency braking and pedestrian intent prediction. P99 SLO $\tau_H=20$ ms (sense-plan-act pipeline [11]), 7B model tier, 10–50 output tokens.
- *Quality-critical (Q)*: Multi-turn reasoning queries prioritizing accuracy. SLO 500 ms, 13B–70B model tier (INT4 or tensor-parallel), 500–2000 output tokens.
- *Throughput-sensitive (T)*: Municipal transit optimization with batch-friendly, high-volume processing. SLO 2 s, 3–7B model tier, 100–500 output tokens.
- *Latency-tolerant (L)*: Casual infotainment queries exploiting idle capacity and KV-cache reuse. SLO 5 s, 3B model tier.

These four classes compete for finite edge resources; meeting the tightest deadline (H-type, 20 ms) while maintaining fairness across classes is the core scheduling challenge.

Among these four classes, the hazard-critical SLO of 20 ms is the binding constraint: scheduling overhead must be a small fraction of this budget. Centralized architectures fail this constraint for three reasons. First, control-plane latency dominates: with 22 RSUs at 1.0–1.8 km spacing, a centralized scheduler sited at the cloud fallback incurs 20–80 ms round-trip time (fiber propagation at $5 \mu\text{s}/\text{km}$ plus queueing and processing), already exceeding the H-type SLO before inference begins. Second, centralized controllers cannot track distributed per-node KV-cache prefixes and dynamic MoE expert placements at the granularity required for 20 ms decisions. Third, they represent a single point of failure during network partitions [7] and incur $O(N)$ per-request state-collection overhead. These limitations motivate *distributed*, *partially-informed* schedulers that make local decisions using gossiped partial state.

Our contributions are:

1. **Decentralized LLM routing framework (DQRR)**: We formalize the distributed QoS-aware request routing problem as a constrained optimization with eight constraints, prove NP-hardness via reduction from 3-PARTITION, and characterize three design properties: bounded staleness, conservative SLO estimation, and regret under stale state (Section 3).
2. **GLARE metaheuristic**: A freshness-driven weight adaptation algorithm where gossip staleness controls the exploration-exploitation trade-off (Section 4.3). Each edge node maintains a weight matrix that interpolates toward high-performing peers in the fresh state and perturbs in the stale state, enabling per-node adaptation to spatial load heterogeneity.
3. **Multi-dimensional scoring for LLM routing**: A scoring framework combining six type-weighted functions (latency, queue, KV-cache affinity, batch coalescing, MoE expert awareness, model tier selection), four of them LLM-specific routing dimensions, into per-request scores computed from gossiped partial state, with freshness-aware staleness discounting across all dimensions (Section 4.2).
4. **Evaluation on empirical vehicle trajectories with spatial RSU mapping**: We evaluate Gossamer on the V2AIX dataset (A44 motorway, 2,388 ITS stations) using a discrete-event simulator with vehicles spatially mapped

Table 1: Feature comparison ($\checkmark$ = supported, $\circ$ = partial, $-$ = not addressed).

Approach	Edge Deployed	Decentral. Routing	KV-Cache Aware	MoE Aware	Fault- Tolerant	Gossip- Based	Staleness- Driven	<20 ms Decision
QAFHE [2]	$\checkmark$	$-$	$-$	$-$	$-$	$-$	$-$	$-$
HEART [19]	$\checkmark$	$-$	$-$	$-$	$-$	$-$	$-$	$-$
vLLM [17]	$-$	$-$	$\checkmark$	$-$	$-$	$-$	$-$	$-$
Flocky [7]	$\checkmark$	$\checkmark$	$-$	$-$	$\checkmark$	$\checkmark$	$-$	$-$
DEMon [10]	$\checkmark$	$\checkmark$	$-$	$-$	$\checkmark$	$\checkmark$	$-$	$-$
SLICE [24]	$\checkmark$	$-$	$\circ$	$-$	$-$	$-$	$-$	$-$
Zhang [23]	$\checkmark$	$-$	$\circ$	$-$	$-$	$-$	$-$	$-$
Habibi [8]	$\checkmark$	$\circ$	$-$	$-$	$\circ$	$-$	$-$	$-$
Jiang [13]	$\checkmark$	$-$	$\circ$	$-$	$-$	$-$	$-$	$-$
MoE2 [14]	$\checkmark$	$-$	$-$	$\checkmark$	$\circ$	$-$	$-$	$-$
SlimCaching [3]	$\checkmark$	$-$	$-$	$\checkmark$	$-$	$-$	$-$	$-$
Xie-MoE [21]	$\checkmark$	$\circ$	$-$	$\checkmark$	$-$	$-$	$-$	$-$
Xu [22]	$\checkmark$	$-$	$-$	$-$	$-$	$-$	$-$	$-$
Sun [20]	$\checkmark$	$-$	$\checkmark$	$-$	$-$	$-$	$-$	$-$
Zhou-DA [25]	$\checkmark$	$-$	$-$	$-$	$-$	$-$	$-$	$-$
GOSSAMER	$\checkmark$	$\checkmark$	$\checkmark$	$\checkmark$	$\checkmark$	$\checkmark$	$\checkmark$	$\checkmark$

to nearest RSUs, comparing against five decentralized baselines (Least-Loaded, Power-of-Two [18], Random, Greedy, and Gossamer-Static) in Section 5. All performance claims are simulation projections and are not hardware-validated.

The rest of the paper is organized as follows. Section 2 positions Gossamer against related work. Section 3 describes the three-tier system model, the gossip protocol, and the request taxonomy, and formalizes the DQRR problem, including its NP-hardness proof. Section 4 presents the Gossamer design: the scoring framework, GLARE, and the routing pipeline. Section 5 evaluates the seven hypotheses against decentralized baselines, and Section 7 discusses limitations and concludes.

2 Related Work

GOSSAMER is the only approach scoring $\checkmark$ in all eight dimensions (Table 1); no competitor combines decentralized routing, gossip-based coordination, staleness-driven adaptation, and sub-20 ms decisions.

Kubernetes-Native Edge Scheduling. QAFHE [2] and HEART [19] provide QoS-aware scheduling for edge-to-cloud deployments but rely on centralized control planes whose round-trip delay (20–80 ms) already exceeds the 20 ms H-type SLO. GOSSAMER routes locally using gossiped partial state, eliminating this round trip.

KV-Cache-Aware LLM Serving. vLLM [17] uses prefix caching and PagedAttention in datacenter clusters; Sun [20] caches KV states at RSUs in VEC networks. Neither addresses decentralized routing under partial state. Flocky [7] and DEMon [10] use gossip for failure detection and membership, not application-layer routing. GOSSAMER extends SWIM-style gossip with state exchange, Bloom-filter prefix awareness, and freshness-driven routing.

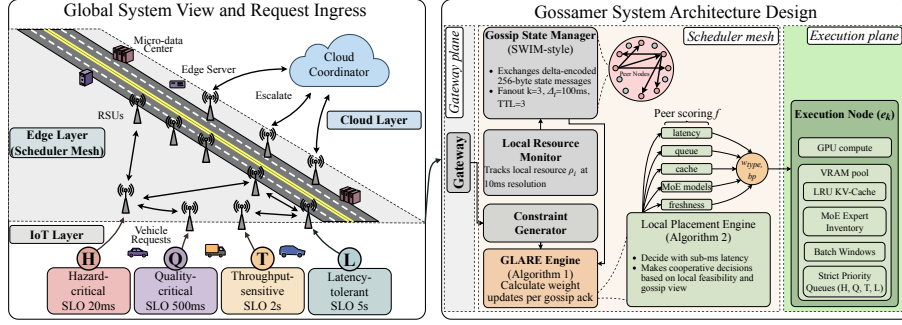


Fig. 2: Gossamer system architecture.

Edge LLM Inference Optimization. SLICE [24] uses SLO-driven scheduling for edge devices ($35\times$ higher SLO attainment than SLO-APMS). Zhang [23] formulates NP-hard batch scheduling with $\frac{1}{2}$ -approximation. Habibi [8] uses auction-based task allocation with adaptive deadline scheduling. Jiang [13] optimizes pipeline parallelism and batching (42.1% latency reduction). These optimize *within* a single edge node; GOSSAMER addresses cross-node routing under partial state, a complementary problem.

MoE-Aware Edge Inference. MoE2 [14] proposes expert selection with two-level gating. SlimCaching [3] designs expert caching with a $(1-1/e)$ approximation. Xie-MoE [21] deploys experts across vehicles via V2V parallelism. These focus on expert *placement*, not per-request routing. GOSSAMER integrates MoE inventory as one of six scoring dimensions.

Collaborative LLM Deployment. Xu [22] partitions LLMs across MEC servers with a PCC protocol (55.6% lower latency than tensor parallelism). Zhou-DA [25] optimizes LLM-SLM delay-accuracy trade-off. These address model partitioning and task offloading, not online routing across edge nodes.

3 System Model and Problem Formulation

3.1 Three-Tier Architecture and Resource Model

The Gossamer architecture distributes request routing and execution across three tiers: gateway, scheduler mesh, and execution. Figure 2 illustrates this hierarchical network structure.

Gateway plane: The gateway manages rapid request ingress and localized dispatch operations. Incoming requests arrive at the nearest RSU via wireless V2X [5] protocols, where the gateway immediately parses each request’s QoS profile q_j and dispatches it to the local scheduler replica. Measured processing overhead is sub-200 μs (Section 5).

Scheduler mesh: The scheduler mesh is a SWIM-style gossip network. Each of the 22 RSUs (the V2AIX deployment count; simulation experiments in Section 5 use all $N=22$ RSUs) hosts an independent, active scheduler replica s_i that exchanges operational state at $\Delta t = 20$ ms intervals with fanout of $k = 3$ and

a TTL of $h = 3$. Each replica maintains a *partial, stale view* $V_i(t)$ containing at most $v_{\max} = \lceil 2.2 \log_2 N \rceil \approx 10$ nodes, evicting entries older than $\tau_{\max} = 500$ ms to prevent drift. The cloud coordinator near Aachen seeds gossip and provides a fallback.

Execution plane: The execution plane provides the GPU compute and memory for LLM serving. Edge nodes are equipped with NVIDIA L40S (48 GB) or Orin AGX (64 GB) GPUs, connected via 10 Gbps fiber backhaul; the cloud fallback provides $8 \times A100$ -80 GB. V2X wireless access [5] (≤ 10 ms latency), inter-RSU backhaul uses fiber (≤ 1 ms), and the cloud uplink uses fiber WAN (≤ 5 ms). Each edge node provisions dedicated local resources covering GPU VRAM, a localized KV-cache store (managed via per-prefix LRU with 128-bit Bloom filter summaries), dynamic MoE expert inventory mapping (via bitmask per GPU/host/neighbor), active batch windows, and strict priority-class request queues.

Resource Model.

Each node’s resource state at time t is a vector:

$$\rho_i(t) = \langle \text{VRAM}_i(t), Q_i(t), K_i(t), E_i(t), B_i(t), U_i(t) \rangle \quad (1)$$

Within this formulation, $\text{VRAM}_i(t)$ is the free GPU VRAM, $Q_i(t)$ indicates the active queue depth per priority class, and $K_i(t)$ denotes the KV-cache store compressed via Bloom filter summaries. $E_i(t)$ tracks the MoE expert inventory bitmask, $B_i(t)$ is the current batch window state, and $U_i(t)$ captures the aggregate compute utilization.

3.2 Gossip Protocol Properties and Staleness Model

We disseminate state by gossip rather than a central coordinator, whose control-plane round-trips already exceed the 20 ms budget, or a distributed hash table, whose multi-hop lookups add per-decision latency; a SWIM-style [4] protocol spreads updates in $O(\log N)$ rounds with no coordinator and within network bandwidth limits. Replicas exchange delta-encoded 256-byte state messages at $\Delta t = 20$ ms intervals with fanout $k = 3$ and TTL $h = 3$. This yields an expected propagation time of $\mathbb{E}[T_{\text{prop}}] \leq \Delta t \cdot \lceil \log_k N \rceil \approx 60$ ms and a coverage probability of $P(\text{cover}) = 1 - (1 - k/(N - 1))^{k^h} \approx 0.98$ [12]. Total bandwidth is analytically estimated as $22 \times 3 \times 256$ B per 20 ms round ≈ 845 KB/s ($< 0.08\%$ of a 10 Gbps link) under idealized steady-state gossip with 256-byte delta-encoded messages; empirical measurement on physical hardware is deferred to future work (real implementations may exhibit 2–5 $\times$ higher overhead due to framing, retransmission, and metadata). Because hazard-critical requests (SLO τ_H) cannot tolerate the full gossip propagation delay, routing decisions must account for staleness, as detailed in Section 3.

Staleness Model.

Each scheduler s_i maintains a partial view $V_i(t)$ where peer e_k ’s state has staleness $\sigma_{ik} = t - t_g$ (time since last update). Staleness is bounded by epidemic dissemination: $\sigma \leq \Delta t \cdot \lceil \log_k N \rceil \approx 60$ ms (Observation 1, Section 3). Freshness is exponentially weighted as $f_{\text{fr}}(e_k) = e^{-\sigma_{ik}/\sigma_0}$ with decay constant $\sigma_0 = 100$ ms.

Table 2: Request type taxonomy with QoS constraints and scheduling parameters.

Type	SLO (P99)	Model	Max batch delay d_j	Preempt
Hazard (H)	τ_H	7B	0 ms [†]	Yes
Quality (Q)	500 ms	13B–70B	≤ 50 ms	No
Throughput (T)	2 s	3–7B	≤ 200 ms	No
Latency-tol. (L)	5 s	3B	≤ 500 ms	No

[†]H-type requests bypass batching entirely ($d_j=0$); all other types may be delayed up to their listed maximum to form efficient batches (see constraint C4, Section 3).

3.3 Request Type Taxonomy

We categorize incoming requests using a QoS profile taxonomy that specifies latency bounds, model requirements, and preemption privileges. Each request r_j carries a QoS profile structured as:

$$q_j = (\mathbf{type}, L_j, m_j, c_j, k_j, s_j)$$

where $\mathbf{type} \in \{H, Q, T, L\}$ defines the core request class, L_j imposes the specific latency SLO, and m_j determines the required model tier. c_j defines the context length, k_j specifies the output length, and s_j provides an optional session ID. Table 2 summarizes the strict parameters mapping these four primary request types.

Building on this system model, we formalize the routing problem as a constrained optimization with index sets, decision variables, and eight constraints that maximize weighted SLO attainment. The state of the art closes parts of this problem but no approach closes all: centralized control planes (QAFHE [2], HEART [19]) violate the 20 ms budget before computation begins; single-node optimizers (SLICE [24], Zhang [23], Habibi [8], Jiang [13]) do not route across nodes; and gossip-based systems (Flocky [7], DEMon [10]) detect failures but do not route application traffic under partial state (Table 1). DQRR therefore combines per-node decisions from gossiped partial state, QoS-class-aware constraints, and staleness-aware scoring, with the H-type 20 ms P99 deadline as the binding constraint.

3.4 Optimization Formulation

Index Sets and Decision Variables. The formulation uses sets for compute nodes E , requests R , and schedulers S (one per node). Let $E = \{e_1, \dots, e_n\}$ denote the compute nodes, $R = \{r_1, \dots, r_m\}$ denote the specific inference requests, and $S = \{s_1, \dots, s_n\}$ denote the independent schedulers mapped one-to-one per node. Within this space, each scheduler s_i actively maintains a constrained subset view $V_i \subset \mathcal{V}$ where $|V_i| \leq v_{\max}$.

The decision variables are: binary assignment $x_{i,j}^{(t)} \in \{0, 1\}$; model tier $m_{i,j}^{(t)} \in \mathcal{M} = \{3B, 7B, 13B, 70B\}$; batching delay $d_j^{(t)} \in \mathbb{N}$ (ms); preemption flag $p_j^{(t)} \in$

$\{0, 1\}$; and the gossip view $V_i(t) \subset \mathcal{V}$ as an observation variable received from the protocol.

Optimization Objective.

The system maximizes weighted SLO attainment subject to eight resource and deadline constraints; response quality, resource efficiency, and throughput are deferred to future work.

SLO Attainment (evaluated): $\max J = \sum_j w_j \mathbb{I}(\text{SLO}_j \text{ met})$, with $w_H=100$, $w_Q=10$, $w_T=1$, $w_L=0.5$.

The forwarding decision f_j is a derived output: $f_j = 1$ iff $x_{i,j} = 1$ for $i \neq \text{local_edge}(r_j)$.

Constraints C1–C8.

The optimization is subject to eight constraints. *C1-Hazard*: Hazard-critical requests require $T_{\text{total}} \leq 20$ ms and $d_j=0$, with preemption allowed. *C2-VRAM*: Memory utilization: $\sum_j x_{i,j} \text{VRAM}(m) + \text{models} + \text{KV} \leq \text{VRAM}_i$. *C3-KV-cache*: Cache allocations bounded by $|K_i| \leq \text{KV}_{\text{max}}$, with LRU eviction and Bloom FPR < 1%. *C4-Batch*: Batching delays: $d_j \in \{0, 50, 200, 500\}$ ms for H/Q/T/L types with type-homogeneous batching. *C5-MoE*: Expert loading: $\sum_j \sum_e \text{capacity}(e) \leq \text{expert_capacity}_i$. *C6-Forwarding*: Maximum one forwarding hop ($\text{TTL} \in \{0, 1\}$). *C7-Deadline*: $T_{\text{arrival}} + T_{\text{exec}} \leq \text{SLO}$ for each request. *C8-Freshness VRAM*:

$$\text{VRAM}_{\text{eff}} = \kappa \cdot \text{VRAM}_{\text{reported}}(t_g) - \eta \cdot (t - t_g) \cdot \lambda_{\text{expected}} \quad (2)$$

where $\kappa = 0.9$ discounts the reported VRAM for staleness (recent reports are more reliable), $\eta = 1.5$ is a conservatism multiplier, and $\lambda_{\text{expected}}$ is the expected VRAM consumption rate (bytes/ms). Both κ and η are design parameters calibrated on our deployment trace via grid search; Experiment 2 validates that the chosen $\eta=1.5$ yields conservative SLO estimates (Figure 4(b)). We estimate $\lambda_{\text{expected}}$ via an exponentially weighted moving average (EWMA) over the last $w = 50$ requests observed at each scheduler: $\lambda_{\text{expected}} = \text{EWMA}(\Delta \text{VRAM} / \delta, \beta=0.3)$, where ΔVRAM is the VRAM allocated per completed request and δ is the inter-completion interval. The parameters $\eta \geq 1$ and $\kappa \leq 1$ produce a conservative admission threshold.

3.5 NP-Hardness

The DQRR problem is NP-hard via reduction from 3-PARTITION [6], which is strongly NP-complete. *Proof sketch*: Given a 3-PARTITION instance with $3m$ elements of sizes $a_1, \dots, a_{3m}$ and target bin capacity B , construct a DQRR instance with m edge nodes each having VRAM capacity B , and $3m$ request batches each consuming VRAM a_i on its assigned edge node. The VRAM capacity constraint (C2) maps directly to bin capacity; the SLO constraint (C1) enforces that every request is assigned. A solution to DQRR with all SLOs satisfied exists iff a valid 3-partition exists. The reduction is polynomial in m . This hardness holds even with a single request type and uniform weights, establishing that the general DQRR problem is NP-hard in the strong sense. We therefore pursue heuristic solutions.

Three properties characterize Gossamer’s operation under the design choices in Sections 3–4: bounded staleness (Observation 1), conservative SLO estimation (Observation 2), and regret decomposition under stale state (Observation 3). These are analytical consequences of the gossip protocol and conservatism parameters, not independent theorems; Section 5 provides empirical confirmation. **Observation 1 (Bounded Staleness).** For gossip with interval Δt and fanout $k \geq 2$, expected staleness is bounded by epidemic dissemination [12]:

$$\mathbb{E}[\text{staleness}] \leq \Delta t \cdot \lceil \log_k N \rceil$$

This follows from the push-pull epidemic model [15]: informed nodes contact k random peers per round, reaching all N nodes in $\lceil \log_k N \rceil$ rounds.

Observation 2 (Conservative SLO Estimation). Constraint C8 uses conservatism multiplier $\eta \geq 1$ to compute effective VRAM as a lower bound on true capacity. Under bounded, independent inter-arrival intervals, Hoeffding’s inequality [9] bounds the empirical arrival rate estimation error: $|\hat{\lambda} - \lambda| \leq (b - a) \sqrt{\ln(2/\alpha)/(2n)}$ with probability $1 - \alpha$. When $\eta \geq 1$, the capacity estimate is pessimistic; the staleness penalty measurement (Experiment 2, Figure 4(a)) validates this empirically.

Observation 3 (Regret Under Stale State). Routing regret against a zero-staleness oracle decomposes into state-change probability during staleness and incorrect threshold-crossing probability. GLARE mitigates this via staleness-proportional exploration: heuristic per-type weights (Table 3) are suboptimal under spatial load heterogeneity (Greedy: 65.1% H-SLO at 200% vs. GLARE’s 94.3%). GLARE learns better weights from peer performance, providing +2.0 pp at 200% load and +3.2 pp at 500% load over the static baseline (Table 5). Empirical regret measurement is deferred to future work.

4 Gossamer Design

Gossamer combines decentralized gossip-based scheduling with multi-dimensional LLM scoring and freshness-aware staleness discounting.

4.1 Architecture

Each Gossamer scheduler replica (Figure 2) operates autonomously with four subsystems:

Local Resource Monitor: Tracks local hardware state $\rho_i(t)$ at 10 ms resolution.

Gossip State Manager ($\Delta t = 20$ ms): Runs a SWIM-style state exchange protocol with fanout $k = 3$ and delta-encoded 256-byte messages [7].

Local Placement Engine: Processes incoming requests at measured 153–209 μs P99 latency across 14–100 nodes (Python implementation, profiled): parses the QoS profile, checks local feasibility, and invokes cooperative routing against the gossip-derived partial view if infeasible.

Request-Type Constraint Generator: Translates the QoS profile q_j into a mathematical constraint set C_j .

Table 3: Type-dependent scoring weights for the six scoring functions.

Type	Lat. Queue Cache MoE Model Fresh.					
Hazard	0.25	0.25	0.05	0.05	0.25	0.15
Quality	0.10	0.10	0.20	0.10	0.40	0.10
Throughput	0.10	0.10	0.10	0.30	0.10	0.30
Lat.-tol.	0.05	0.05	0.05	0.10	0.05	0.70

Each request follows the pipeline of Algorithm 2, discounting peer scores by staleness and transfer cost.

4.2 Scoring Functions

Gossamer uses six type-weighted scoring functions to evaluate peer candidates across heterogeneous QoS requirements. Weights reflect each type’s QoS priorities. As defined in Table 3, the aggregate score for candidate peer e_k on request r_j is:

$$\text{score}(e_k, r_j) = \frac{\sum_{f \in \mathcal{F}} w_{\text{type}, f} \cdot f(e_k, r_j)}{\sum_f w_{\text{type}, f}} \cdot (1 - \text{bp}) \quad (3)$$

where $\mathcal{F} = \{\text{latency, queue, cache, moe, model, freshness}\}$ and $\text{bp} \in [0, 1]$ discounts nodes whose previously reported state overestimated available resources. Experiment 3 (Section 5) confirms that overlapping scoring signals provide redundancy: removing any single dimension changes overall SLO by at most ± 0.4 pp.

Each scoring function normalizes to $[0, 1]$: $f_{\text{lat}} = 2/(1 + \exp(L_{\text{est}}/50))$, $f_{\text{queue}} = 1/(1 + Q_k/10)$, $f_{\text{cache}} = \mathbb{I}(\text{Bloom match})$, $f_{\text{moe}} = |E_{\text{GPU}} \cap E_{\text{act}}|/|E_{\text{act}}|$, $f_{\text{model}} = \mathbb{I}(m_k \geq m_j) \cdot (1 + m_k)/(1 + m_j)$, $f_{\text{fr}} = e^{-\text{staleness}/\sigma_0}$.

4.3 GLARE: Gossip-Driven Local Adaptive Routing Engine

GLARE is a metaheuristic that uses gossip staleness as a control signal for per-edge-node weight adaptation. Vehicles at different positions along the motorway communicate with different RSUs, creating spatial load heterogeneity: an RSU near an urban interchange experiences different request patterns than one on a rural segment. GLARE exploits this heterogeneity by adapting each edge node’s weight matrix based on locally observed peer performance.

Design. Each edge node adapts its weight matrix $\mathbf{w}_i \in \mathbb{R}^{6 \times 4}$ at every gossip tick by staleness-driven simulated annealing (Algorithm 1): it exploits the best-scoring peer when fresh and explores via masked Gaussian perturbation when stale, where the variance-adaptive mask $\mathbf{M}$ suppresses near-zero-variance dimensions. GLARE thereby improves H-type SLO over static routing under spatial load heterogeneity (Section 5.2).

Algorithm 1 GLARE: Staleness-Driven Per-Tick Weight Adaptation

```

1: INIT:  $\mathbf{w}_i \leftarrow 0.5$  ( $6 \times 4$ );  $\tau_{\text{decay}}=1.0\text{s}$ ,  $\theta_{\text{stale}}=5$ ,  $\sigma=0.10$ 
2: PER GOSSIP TICK ( $\Delta t=20\text{ms}$ , staleness  $s$ ):
3:    $T \leftarrow \sigma(1 + s/\tau_{\text{decay}})$ ;  $\alpha \leftarrow \min(1, \Delta t/\tau_{\text{decay}})$ 
4: EXPLOIT (FRESH): if  $f_{\text{fr}}(s) > 0.1$ :  $\mathbf{w}_i \leftarrow (1 - \alpha)\mathbf{w}_i + \alpha\mathbf{w}_{\text{best}}$ 
5: EXPLORE (STALE): if  $s \geq \theta_{\text{stale}}$ :  $\mathbf{w}_i \leftarrow \mathbf{w}_i + \mathbf{M} \odot \mathcal{N}(0, \sigma s/\Delta t)$ 
6: ACCEPT: lower-scoring peer w.p.  $\exp(-\Delta/T)$  (Metropolis; H-type excl.)

```

4.4 Freshness-Aware SLO Probability Estimation

The SLO attainment probability for a candidate assignment is estimated by combining the predicted execution time distribution with a staleness penalty. We model total service time as log-normally distributed, consistent with profiled LLM inference latency distributions [17]:

$$P(\text{SLO}_j \text{ met}) = \Phi\left(\frac{L_j - \mu_{\text{total}}}{\sigma_{\text{total}}}\right) \quad (4)$$

where the total mean resolves to $\mu_{\text{total}} = T_{\text{exec}} + \frac{\lambda \cdot \text{staleness}}{2} \cdot \bar{T}_{\text{service}}$ and the aggregate variance resolves to $\sigma_{\text{total}} = \sigma_{\text{base}} + \beta \cdot \text{staleness}$.

All experiments use static forwarding thresholds (H: 0.70, Q: 0.50, T: 0.40, L: 0.30); adaptive threshold tuning is deferred to future work.

Algorithm 2 Per-Request Processing (Routing) Pipeline

```

1: INPUT: request  $r$  of type  $t \in \{\text{H}, \text{Q}, \text{T}, \text{L}\}$ , SLO  $L_t$ , gossip view  $V_i(t)$ 
2: LOCAL FAST-PATH: compute  $P(\text{SLO}_t \mid \rho_i)$  via Eq. 4
3:   if feasible then serve locally and return
4: SCORE PEERS: for each candidate  $e_k$  in top- $k$  of  $V_i(t)$ :
5:    $\text{score}(e_k, r) \leftarrow \frac{\sum_{f \in \mathcal{F}} w_{t,f} f(e_k, r)}{\sum_f w_{t,f}} \cdot (1 - \text{bp})$  (Table 3)
6: FORWARD:  $e^* \leftarrow \arg \max_k \text{score}(e_k, r)$ 
7:   if  $\text{score}(e^*, r) \geq \tau_t$  then forward to  $e^*$  else escalate to cloud

```

LLM Routing Dimensions. The scoring functions of Section 4.2 embed four LLM-specific mechanics absent from generic load balancers: *KV-cache affinity* (f_{cache}) routes to peers with matching cached prefixes, *batch coalescing* (f_{queue}) to peers with compatible in-flight requests, *MoE awareness* (f_{moe}) to peers with the required experts resident, and *model-tier selection* (f_{model}) to peers hosting an adequate model tier.

5 Performance Evaluation

We evaluate seven hypotheses: *H1*: Gossamer leads all baselines across 50–500% load. *H2*: KV-cache routing accuracy above 60% at $\Delta t=20$ ms. *H3*: Conservatism multiplier η trades SLO attainment against false-forward rate; $\eta=1.5$ balances the two. *H4*: Scoring dimensions provide overlapping, redundant signals:

removing any single dimension changes SLO by at most ± 0.4 pp. *H5*: Decision latency scales sub-linearly ($O(\log_k N)$). *H6*: Graceful fault tolerance: $\geq 99.8\%$ SLO through 11 of 22 failures. *H7*: GLARE improves H-type SLO over static-weight routing under spatial load heterogeneity.

Target rationale. The target values above fall into four provenance classes. H1’s dominance claim is measured in Experiment 1. H2’s 60% floor is deliberately conservative: measured KV-cache routing accuracy is 73–74% across all gossip intervals (Experiment 2), and 60% is a self-imposed acceptance threshold, not a literature value. H3’s $\eta=1.5$ is selected at the knee of the measured SLO–false-forward trade-off (Experiment 2), trading 1.1 pp SLO for 2.5 pp lower false-forward rate relative to $\eta=0.5$. H4’s ± 0.4 pp ablation bound is measured in Experiment 3. H5 follows analytically from epidemic dissemination with fanout $k=3$ [12]. H6’s 11-of-22 failures is a self-imposed criterion matching half the fleet, with the $\geq 99.8\%$ bar keeping empirical SLO within the hazard-critical P99 floor. H7’s advantage over static routing is measured in the GLARE ablation. H2–H4 and H6 are thus engineering acceptance criteria established in the V2AIX workload (a multi-modal real-world dataset of ETSI ITS messages gathered in public road traffic in Germany).

5.1 Setup and Hypotheses

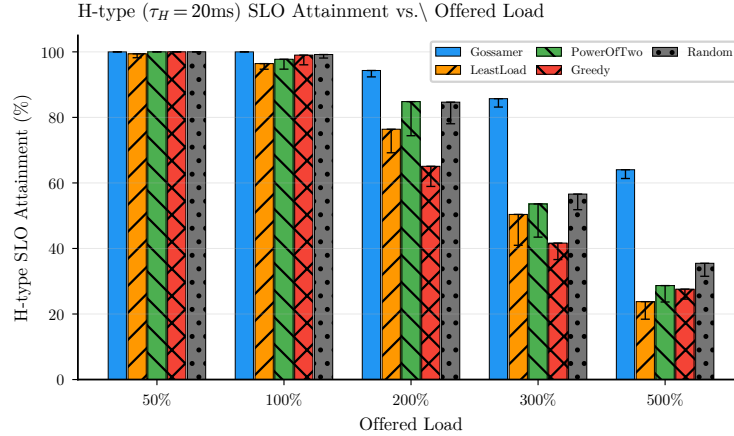


Fig. 3: H-type ($\tau_H=20$ ms) SLO attainment under spatial RSU mapping ($N=22$, 10 seeds; lower whiskers show -1 std.).

We use a discrete-event simulator driven by the V2AIX A44-Jackerath recording (1,035 CAMs, 158 s) [16], sampling inter-arrival gaps from the empirical distribution ($\mu=153$ ms) scaled by load factor. Requests are spatially mapped to RSUs along the A44 corridor: urban-proximal RSUs receive more traffic than rural segments, creating the heterogeneity GLARE exploits. LLM inference uses log-normal latency distributions fitted to profiled Llama-3B/8B on L40S [17];

network delays use 3GPP TR 38.901 urban-macro parameters [1]. **Baselines:** Gossamer-Static (no-GLARE), Random, Least-Loaded, Power-of-Two [18], and Greedy. **Statistics:** Mean over 10 independent runs ($N=22$). 95% bootstrap CIs (10,000 resamples, percentile, independent per method). Tables 4, 5, and 6 stem from separate experiment batches with independent RSU topologies; Gossamer’s H-type SLO at 200% load therefore varies by ± 1 pp across batches (94.1–95.5%). **Workload:** H:15%, Q:25%, T:35%, L:25%.

Table 4: Per-type SLO attainment (%) and mean latency (ms) at 200% load with spatial RSU mapping ($N=22$, 10 seeds). Bold marks best per column.

Scheduler	H (% / ms)	Q (% / ms)	T (% / ms)	L (% / ms)
Least-Loaded	76.4 / 15	89.3 / 201	88.1 / 813	85.9 / 2314
Power-of-Two [18]	84.8 / 12	95.2 / 137	95.1 / 436	92.5 / 1330
Random	84.7 / 12	96.5 / 131	94.5 / 527	91.8 / 1627
Greedy	65.1 / 85	65.0 / 4880	99.8 / 194	100.0 / 305
Gossamer (ours)	94.3 / 8	99.5 / 82	98.7 / 274	99.3 / 438

Experiment 1: SLO Attainment Under Load (H1)

Figure 3 shows H-type SLO attainment across 50%–500% load under spatial RSU mapping; Gossamer leads all baselines at every load level (100% at 50–100% load, 94.3% at 200%, 85.7% at 300%, 64.0% at 500%), a 29.1 pp gap over Random at 300% load, where the next-best baseline peaks. Table 4 reports per-type results at 200% load. Two patterns emerge. First, Gossamer dominates latency-sensitive types: H at 94.3%/8 ms vs. Greedy’s 65.1%; Q at 99.5% vs. Greedy’s 65.0%. Second, Greedy achieves 100% T/L-type SLO via aggressive cloud escalation, while Gossamer serves these types at the edge with 99+% SLO. Static weights reach 93.6% at 200% load and 84.0% at 300% (Section 5.2). Greedy uses fixed weights (Table 3) without GLARE, so the 29.2 pp H-type gap reflects the full system advantage over Greedy, of which GLARE contributes +2.0 pp (Table 5).

Experiment 2: KV-Cache and Conservatism (H2, H3).

KV-cache routing accuracy stays stable at 73–74% across gossip intervals of 10, 20, 50, 100, 200, and 500 ms. Bandwidth scales inversely (≈ 1.7 MB/s at 10 ms, ≈ 34 KB/s at 500 ms); $\Delta t=20$ ms balances accuracy with < 850 KB/s overhead. Figure 4(a) evaluates the staleness penalty: the measured Gossamer violation rate stays well below the Observation 1 theoretical bound across the full staleness range. Figure 4(b) sweeps $\eta \in \{0, 0.5, 1.0, 1.5, 2.0, 3.0\}$: $\eta=0.5$ is SLO-optimal (91.7%), while the chosen $\eta=1.5$ trades 1.1 pp SLO for 2.5 pp lower false-forward rate.

Experiment 3: Scoring Dimension Ablation (H4).

Figure 4(c) shows the result of removing each routing dimension at 200% load with 15 edge nodes. All single-dimension removals change overall SLO by at most ± 0.4 pp (full: 89.6%; no-kv: 89.2%; no-freshness: 89.2%; no-moe: 89.4%; no-model: 89.6%; no-batch: 89.5%), with no removal producing a statistically

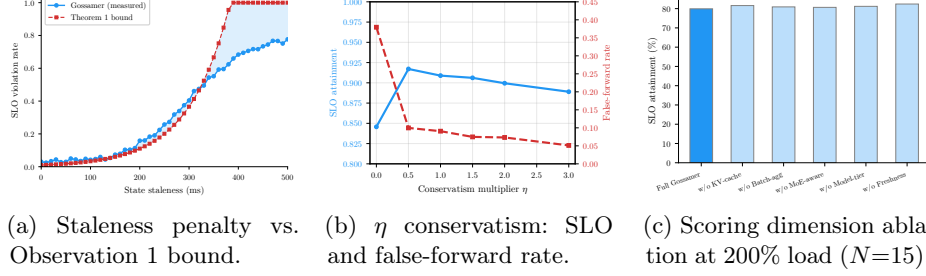


Fig. 4: Experiment 2-3: (a) staleness penalty, (b) η conservatism sweep, (c) dimension ablation.

Table 5: GLARE ablation: H-type SLO attainment under spatial load heterogeneity ($N=22$, 10 seeds; uniform 0.5 weights for direct comparison).

Variant	50%	100%	200%	300%	500%
Gossamer-no-GLARE (static)	100.0	100.0	93.6	84.0	62.1
Gossamer (GLARE)	100.0	100.0	95.5	84.5	65.3

significant difference at 10 seeds. Mean H-type latency follows the same pattern (full: 78.3 ms; no-freshness: 80.2 ms, the largest increase at +1.9 ms). The near-equivalence of all dimension subsets confirms H4: overlapping scoring signals provide redundancy, and no single dimension is essential for routing performance. The ablation demonstrates that removing any signal shifts rather than degrades routing quality because the remaining dimensions compensate.

Experiment 4: Scalability (H5).

In simulation, Gossamer achieves $\geq 99.5\%$ SLO at ≥ 12 edge nodes, maintained through $N=100$ (Figure 5(a)). Decision latency scales as $O(\log_k N)$ with P99 measured at 153–209 μs across all counts; centralized architectures are unsuitable because their control-plane RTT already exceeds the 20 ms H-type SLO budget (Section 1).

Experiment 5: Fault Tolerance (H6).

Figure 5(b) shows SLO degradation under node failures (0–22, crash-stop, no cloud). $\geq 99.8\%$ empirical SLO is maintained with 11 failures (half the fleet), $> 90\%$ with 18, and drops to 0% only when all 22 are offline. Recovery is bounded by gossip convergence (≤ 300 ms).

5.2 GLARE Ablation (H7)

To isolate GLARE’s contribution, we compare Gossamer against a static-weight variant (Gossamer-no-GLARE) where each edge node uses a fixed weight matrix $\mathbf{w}_i \in \mathbb{R}^{6 \times 4}$ with all entries set to 0.5 without adaptation. Table 5 reports H-type SLO attainment across load levels with spatial RSU mapping and 20 ms gossip.

The H-type SLO advantage and its bootstrap 95% CI (confidence interval) are: +2.0 pp at 200% load (95% CI: -0.8 to $+5.0$ pp), +0.5 pp at 300% (95%

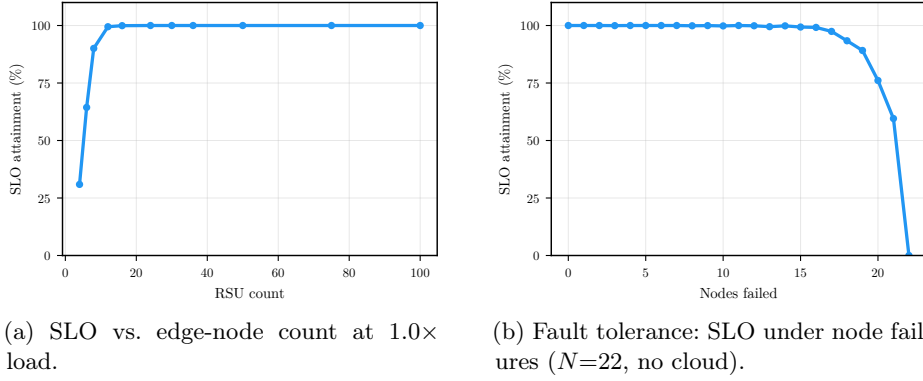


Fig. 5: Experiment 4-5: (a) scalability to 100 edge nodes, (b) adversarial fault tolerance.

CI: -2.2 to $+3.2$ pp), and $+3.2$ pp at 500% (95% CI: $+1.7$ to $+4.8$ pp). Only the 500% improvement is statistically significant; at moderate loads, the CIs cross zero. GLARE’s per-gossip-tick computational cost is dominated by the SA temperature update and weight interpolation (both $O(1)$ per dimension, yielding negligible overhead at $\Delta t=20$ ms). Section 5.3 compares GLARE against generic adaptive heuristics. Bootstrap CIs are tight: GLARE ± 0.5 – 1.3 pp, static ± 0.8 – 2.6 pp.

5.3 Adaptive Baselines (H7 extension)

We compare GLARE against three generic adaptive heuristics: ε -greedy ($\varepsilon = 0.20 \rightarrow 0.01$), UCB1 (Upper Confidence Bound, $c=0.5$), and EWMA ($\alpha=0.3$).² All use GLARE’s composite peer quality score but lack staleness awareness and gossip state. Table 6 reports H-type SLO at 200% load. GLARE achieves 94.1%, vs. 76.2% (ε -greedy), 78.2% (EWMA), and 80.1% (UCB). All three underperform Random (84.7%) because they always forward to peers rather than serving locally when possible. GLARE’s local fast-path, type-weighted scoring, and staleness-driven adaptation together provide a 14.0–17.9 pp advantage.

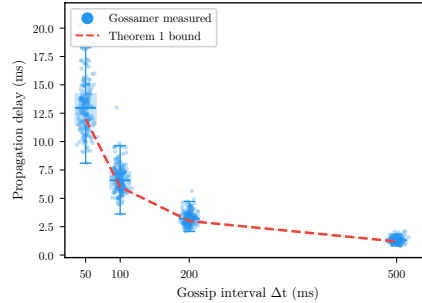
5.4 Supplementary Results and Synthesis

Across all experiments, Gossamer achieves the highest H-type SLO among decentralized baselines (H1–H6), with GLARE providing up to $+3.2$ pp over static-weight routing under spatial heterogeneity (H7). Analytic gossip coverage reaches 98% within 3 rounds for fanout $k=3$, consistent with the $\lceil \log_k N \rceil$ -round dissemination bound (Observation 1); measured propagation delay tracks this bound across gossip intervals (Figure 6).

² ε -greedy: random peer w.p. ε , else best-scoring. UCB1: exploration bonus $c\sqrt{\ln N/n_j}$. EWMA: smooth each peer’s score over time.

Table 6: GLARE vs. adaptive baselines at 200% load ($N=22$, 10 seeds).

Method	H-type SLO (%)
ϵ -greedy	76.2
EWMA	78.2
UCB1 (Upper Confidence Bound)	80.1
Greedy (static per-type weights)	65.1
Random	84.7
Gossamer (GLARE)	94.1

Fig. 6: Gossip propagation delay vs. gossip interval Δt (measured vs. staleness bound; $N = 23$; 22 RSUs + cloud coordinator).

6 Discussion and Conclusion

6.1 Simulation Fidelity, Limitations, and Future Work

Sensitivity analysis. We conduct one-dimensional parameter sweeps but not multi-parameter interactions. Results are conditioned on the V2AIX trace; transfer to other topologies is untested.

Simulation fidelity. The simulator is calibrated against profiled single-node measurements. Decision latency is profiled at 153–209 μs P99 (Python); other results use analytical projections.

Limitations and future work. (1) Physical testbed validation with GPU-equipped edge nodes remains future work. (2) Per-type scoring weights are hand-tuned and overlap; learned per-node weights may improve differentiation. (3) Scalability to thousands of edge nodes may require hierarchical gossip. (4) Gossamer assumes honest participants. Lightweight per-message authentication (e.g., HMAC-SHA256, 2.5 μs P99 measured on our development host, under 25 μs even with a $10\times$ embedded-hardware margin) would relax this to authenticated participants at negligible cost to the 20 ms budget; validating this on real RSU hardware, Byzantine-robust routing, key management, and trust establishment remain future work.

6.2 Conclusion

This paper proposed Gossamer, a decentralized, staleness-aware request-routing framework designed for stateful LLM inference in edge environments. Gossamer integrates type-weighted, multi-dimensional scoring over gossiped partial state with explicit staleness accounting, prioritizing the 20 ms hazard-critical SLO whose violation directly translates into meters of vehicle travel. The proposed framework holds hazard-critical SLOs where centralized schedulers cannot, by keeping each routing decision within 1.1% of the SLO budget. Furthermore, Gossamer incorporates GLARE, a staleness-driven weight-adaptation engine that balances exploration and exploitation per edge node, improving H-type SLO by up to +3.2 pp over static routing at extreme overload. Established in simulation over empirical V2AIX vehicle trajectories, these results suggest gossip-driven local adaptation is a viable coordination primitive for latency-critical edge inference beyond autonomous driving; extending the objective beyond SLO attainment to response quality and efficiency remains future work.

Acknowledgments: This work is partially supported by an ARC Discovery Project. We used DeepSeek-V4-Pro to correct LaTeX syntax and to debug the simulation code.

References

1. 3rd Generation Partnership Project: 3GPP TR 38.901: Study on channel model for frequencies from 0.5 to 100 GHz. Tech. rep., 3GPP (2017), release 14
2. Cámara-Miró, J., Costero, L., Igual, F.D.: Qafhe: A qos-aware framework for heterogeneous and dynamic edge-to-cloud kubernetes deployments. *International Journal of Networked and Distributed Computing* **14**(1), 11 (Feb 2026)
3. Chen, Q., Chen, X., Huang, K.: Slimcaching: Edge caching of mixture-of-experts for distributed inference. *IEEE Transactions on Mobile Computing* **25**(7), 10924–10938 (July 2026)
4. Das, A., Gupta, I., Motivala, A.: Swim: scalable weakly-consistent infection-style process group membership protocol. In: *Proceedings International Conference on Dependable Systems and Networks*. pp. 303–312 (2002)
5. European Telecommunications Standards Institute: ETSI EN 302 637-2: Intelligent transport systems (ITS); vehicular communications; basic set of applications; part 2: Specification of cooperative awareness basic service. Tech. rep., ETSI (2014)
6. Garey, M.R., Johnson, D.S.: “strong” np-completeness results: Motivation, examples, and implications. *J. ACM* **25**(3), 499–508 (Jul 1978)
7. Goethals, T., Sebrechts, M., Al-Naday, M., De Turck, F., Volckaert, B.: Flocky: Decentralized intent-based edge orchestration using open application model. *IEEE Transactions on Services Computing* **19**(1), 380–395 (2026)
8. Habibi, S., Ercetin, O.: Edge-llm inference with cost-aware layer allocation and adaptive scheduling. *IEEE Access* **13**, 131614–131637 (2025)
9. Hoeffding, W.: Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association* **58**(301), 13–30 (1963)

10. Ilager, S., Fahringer, J., Tundo, A., Brandic, I.: A decentralized and self-adaptive approach for monitoring volatile edge environments. *ACM Transactions on Autonomous and Adaptive Systems* **21**(2) (Apr 2026)
11. International Organization for Standardization: ISO 21448:2022: Road vehicles — safety of the intended functionality. Tech. rep., ISO (2022)
12. Jelasity, M., Voulgaris, S., Guerraoui, R., Kermarrec, A.M., van Steen, M.: Gossip-based peer sampling. *ACM Transactions on Computer Systems* **25**(3), 8–es (Aug 2007)
13. Jiang, J., Chen, Z., Shin, H., Nallanathan, A.: Edge inference for large language models with pipeline parallelism and batching. *IEEE Transactions on Communications* **74**, 8390–8406 (2026)
14. Jin, L., Zhang, Y., Li, Y., Wang, S., Yang, H.H., Wu, J., Zhang, M.: Moe2: Optimizing collaborative inference for edge large language models. *IEEE Transactions on Networking* **34**, 4637–4651 (2026)
15. Kermack, W.O., McKendrick, A.G.: A contribution to the mathematical theory of epidemics. *Proceedings of the Royal Society of London. Series A, Containing Papers of a Mathematical and Physical Character* **115**(772), 700–721 (08 1927)
16. Kueppers, G., Busch, J.P., Reiher, L., Eckstein, L.: V2aix: A multi-modal real-world dataset of etsi its v2x messages in public road traffic. In: 2024 IEEE 27th International Conference on Intelligent Transportation Systems (ITSC). pp. 392–398 (2024)
17. Kwon, W., Li, Z., Zhuang, S., Sheng, Y., Zheng, L., Yu, C.H., Gonzalez, J., Zhang, H., Stoica, I.: Efficient memory management for large language model serving with pagedattention. In: *Proceedings of the 29th Symposium on Operating Systems Principles*. p. 611–626. SOSP '23, Association for Computing Machinery, New York, NY, USA (2023)
18. Mitzenmacher, M.: The power of two choices in randomized load balancing. *IEEE Transactions on Parallel and Distributed Systems* **12**(10), 1094–1104 (2001)
19. Park, H., Kim, D., Kim, H., Lim, G., Yu, H.: Heart: Heterogeneous-aware traffic allocation in multi-replica deployments on kubernetes. In: 2025 IEEE 18th International Conference on Cloud Computing (CLOUD). pp. 409–419 (2025)
20. Sun, H., Tian, H., Zheng, J., Niyato, D.: Computational caching-empowered collaborative llm inference framework in vec networks. *IEEE Transactions on Vehicular Technology* pp. 1–6 (2025)
21. Xie, G., Xiong, Z., Xie, R., Deng, X., Guo, S., Guizani, M., Han, Z.: Mixture of experts-enabled parallel scheduling and processing for vehicular generative ai services. *IEEE Transactions on Cognitive Communications and Networking* **12**, 994–1011 (2026)
22. Xu, D., Duan, L., Zhu, J., Zhu, H.: Decentralized llm deployment in mobile edge computing networks. *IEEE Transactions on Vehicular Technology* pp. 1–16 (2026)
23. Zhang, X., Nie, J., Huang, Y., Xie, G., Xiong, Z., Liu, J., Niyato, D., Shen, X.: Beyond the cloud: Edge inference for generative large language models in wireless networks. *IEEE Transactions on Wireless Communications* **24**(1), 643–658 (Jan 2025)
24. Zhou, P., Lei, Y., Liu, L., Xu, X., Yu, H., Sun, G., Ergu, D., Dai, Y.: Slice: Slo-driven scheduling for llm inference on edge computing devices. *IEEE Transactions on Services Computing* pp. 1–14 (2026)
25. Zhou, X., Zheng, C., Liu, W., Hu, W., Gao, L., Xiao, K.: Delay-accuracy trade-off optimization for collaborative llm-slm inference in vehicular edge computing. *IEEE Internet of Things Journal* pp. 1–1 (2026)