Towards Traffic Modelling of Multi-Agent Systems: The Role of Coordination Topology

Davide Lamagna
UPC, BarcelonaTech
Barcelona, Spain
davide.lamagna@estudiantat.upc.edu

Albert Cabellos
UPC, BarcelonaTech
Barcelona, Spain
acabello@ac.upc.edu

Alberto Rodriguez-Natal
Cisco
Barcelona, Spain
natal@cisco.com

Gábor Rétvári
Budapest University of Technology and Economics
Budapest, Hungary
retvari@tmit.bme.hu

Berta Serracanta
UPC, BarcelonaTech
Barcelona, Spain
berta.serracanta@upc.edu

Abstract

Multi-agent LLM systems are an emerging networked workload whose rapid deployment raises questions about the traffic patterns they generate. Compared to conventional applications, these systems generate requests internally: a single user task can induce a structured sequence of model calls whose timing is governed by coordination logic rather than by user arrival rate. It is not clear whether classical traffic models, designed for human-driven workloads, apply to this setting.

We present an empirical characterisation of LLM-call inter-arrival time distributions across sequential, star, and full-mesh agentic coordination topologies, using a multi-layer measurement framework over 500 repeated runs per topology. We find that topology fundamentally shapes the arrival process of requests to the LLM backend: fan-out coordination introduces a structural bimodality absent in sequential execution, and the reasoning-phase component is best described by a log-normal distribution, with the Poisson exponential null model decisively rejected across all topologies. These differences propagate to inference and network level metrics. The framework and analysis pipeline are released openly at https://github.com/dlamagna/agentraffic.

CCS Concepts

• **Networks** → **Network measurement**; • **Computing methodologies** → **Multi-agent systems**; • **General and reference** → **Empirical studies**.

Keywords

multi-agent systems, traffic modelling, inter-arrival time, LLM inference, network measurement, traffic characterisation, coordination topology

SIGCOMM '26, Denver, CO, USA

ACM ISBN 979-8-4007-2467-1/2026/08
https://doi.org/10.1145/3789240.3828749

ACM Reference Format:
Davide Lamagna, Albert Cabellos, Alberto Rodriguez-Natal, Gábor Rétvári, and Berta Serracanta. 2026. Towards Traffic Modelling of Multi-Agent Systems: The Role of Coordination Topology. In *ACM SIGCOMM 2026 Conference (SIGCOMM '26), August 17–21, 2026, Denver, CO, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3789240.3828749

1 Introduction

Multi-agent large language model (LLM) systems are emerging as a new class of networked application. In these systems, autonomous agents invoke LLMs, call tools, and coordinate with peer agents to complete a task. Unlike conventional human-driven workloads, where traffic is largely driven by external user requests, multi-agent systems can generate additional requests internally as agents reason, branch, retry, and exchange information. A single user task may therefore induce a sequence of model calls and inter-agent messages whose timing is shaped by the coordination logic of the system itself. Parallel branching and coordinated fan-out can produce bursts that matter for shared inference backends, API gateways, service meshes, rate limiters, load balancers, and enterprise networks, where synchronized request arrivals can increase queueing delay even when average request rate is unchanged.

Traffic characterisation has a long history as a tool for planning, dimensioning, and operating communication networks. A recurring lesson from traffic measurement is that new paradigms repeatedly invalidate prior assumptions: Leland et al. showed that Ethernet traffic is self-similar across timescales [11]; Paxson and Floyd demonstrated that wide-area traffic frequently violates Poisson assumptions [15]; Crovella and Bestavros linked Web self-similarity to heavy-tailed transfer sizes [5]; and Benson et al. characterised datacenter traffic as bursty, asymmetric, and rack-concentrated: distinct from prior wide-area assumptions [3]. History tells us that new workloads (web, peer-to-peer, video streaming, cloud datacenters) often required measurement-driven characterisation before accurate models could be built.

Agentic LLM systems represent a new such workload. Enterprise adoption is already underway: a recent industry survey of over 300 senior executives found that 79% report AI agents being deployed in their organisations, with 88% planning to increase budgets specifically because of agentic AI [16]. Studying the traffic

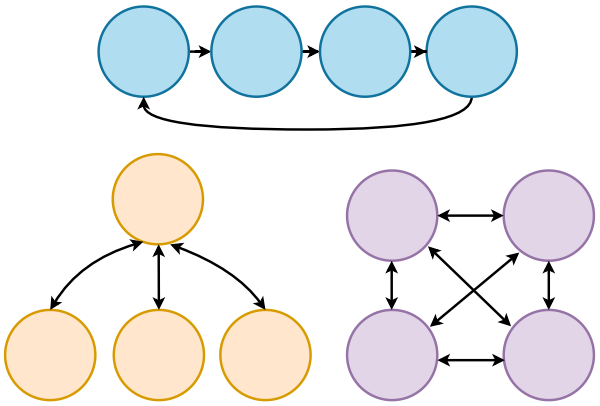

Figure 1: The multi-agent topologies evaluated in this paper: Sequential (top), Star (bottom left), Full Mesh (bottom right)

these systems generate before they become a dominant workload is a natural step in this empirical tradition. To the best of our knowledge, this work is the first to characterise how coordination topology shapes LLM-call arrival processes in multi-agent LLM systems.

In this work, we ask whether the coordination topology of a multi-agent system induces distinct request-arrival processes. A single user-triggered task generates a sequence of internal LLM calls whose timing is shaped by the coordination logic of the agentic workflow, not by user arrival rate. We use the inter-arrival times of these internal calls as our primary traffic metric. We additionally collect lower-layer TCP and packet metrics to provide cross-layer context for the request-level measurements.

A key feature of multi-agent systems is that topology is an explicit design choice [20, 21]: if topology affects how agents solve tasks, it is natural to ask whether it also affects the traffic they generate. We therefore focus on LLM-call inter-arrival times (IATs) across three representative topologies: sequential, star, and full mesh. These topologies capture common coordination patterns: one-at-a-time execution, orchestrator-driven fan-out, and dense peer-to-peer exchange. Our goal is not to exhaust the design space, but to measure how coordination structure is visible in the arrival process.

This paper makes three contributions. First, we provide an initial empirical characterisation of how coordination topology shapes LLM-call IAT distributions. Second, distribution fitting over the reasoning-phase component shows log-normal is the best fitting model across all topologies, with the exponential null decisively rejected. Third, we release an open-source, multi-layer measurement framework for studying agentic traffic under configurable coordination topologies.

2 Background

LLM agents as traffic-generating systems. LLM agents differ from conventional request-response applications because they can generate additional work while executing a task. A ReAct-style agent alternates between reasoning and actions, deciding when to query a model, use external sources (tool/agents), or continue execution [19]. As a result, the arrival process observed by the network can be shaped by internal control flow rather than directly by user arrival rate. Recent work has begun to frame LLM and generative-AI services as an emerging source of Internet traffic [8]; however, multi-agent systems add a further layer of internally generated coordination traffic. Agentic LLM systems resemble microservice applications in that a single external request can expand into an internal call graph. However, the graph is not fixed by application code alone: it is also shaped dynamically by agent reasoning, coordination policy, and model or tool outputs. This paper focuses on LLM-call traffic as a first step, whilst external tool calls or actions are left for future work.

LLM call latency and inter-arrival time. Inter-arrival time (IAT), the wall-clock gap between consecutive requests, is a foundational metric in traffic characterisation. Alongside burstiness, correlation structure, and flow or request sizes, IAT helps describe whether an arrival process is compatible with common traffic models such as Poisson, heavy-tailed, or self-similar processes, and directly governs queuing behaviour and resource dimensioning.

In sequential workflows, IAT is strongly coupled to call latency. However, when parallel tasks are introduced, dispatch policy results in several long-running calls having near-zero inter-arrival gaps because they are issued concurrently. Coordination protocols tend to be phase-structured: recruitment, dispatch, discussion, aggregation, and finalization may each generate different request timing patterns. The resulting arrival process may therefore be a mixture of phase-specific processes rather than a single stationary distribution.

Coordination topology. Modern multi-agent frameworks make topology an explicit implementation choice. AutoGen supports applications built from multiple conversable agents with programmable conversation behaviours [18]; AgentVerse structures multi-agent problem solving into staged recruitment, discussion, and consensus phases [4]; LangGraph represents agent workflows as graph-structured control flows [10]; and CrewAI supports process-level orchestration of role-specialised agents [14]. The topologies these frameworks expose span a range of coordination patterns: a sequential topology issues calls one after another; a star or supervisor-worker topology fans out work from a central coordinator to subagents; and a full mesh allows dense peer-to-peer exchange among all participants. Recent work treats the communication graph as a design variable in its own right: MARBLE studies how topology affects solution quality across different task domains [21], while G-Designer optimises communication graphs to reduce unnecessary token overhead [20]. This body of work has mainly evaluated task quality, robustness, or cost. We instead ask how these topologies affect traffic flow.

3 Measurement framework

The framework measures how agent coordination is expressed in traffic. It observes agentic systems across four layers simultaneously: application, model-serving, container, and network. In the experiments below, topology is the controlled design variable,

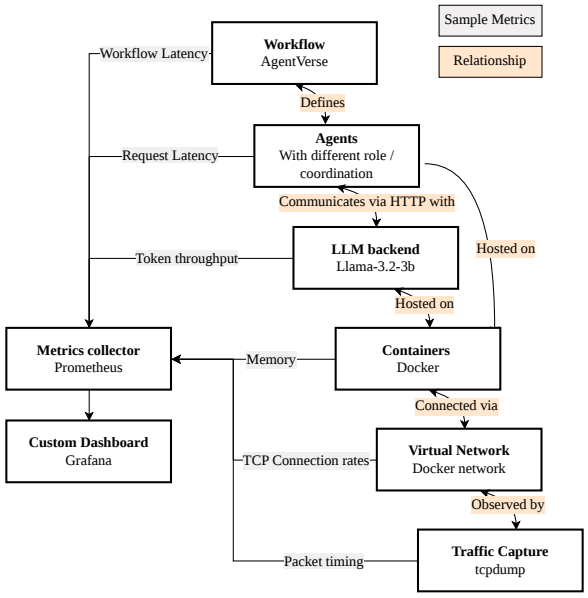

Figure 2: Measurement framework architecture.

while the agent runtime, model backend, and monitoring stack are kept unchanged.

3.1 Architecture

The framework is organised as a set of containerised services connected by Docker bridge networks: the LLM backend, orchestrator, and worker agents each reside on their own dedicated virtual home network, with cross-service traffic flowing over a shared bridge. A passive packet-capture collector reconstructs TCP flows and exports metrics to Prometheus; application traces go to Jaeger; and container resource metrics are collected via cAdvisor. Figure 2 sketches the end-to-end architecture.

3.1.1 Workflow. We use AgentVerse [4] as the primary workflow family. AgentVerse structures multi-agent problem solving into staged coordination: agents are recruited for a task, assigned roles, asked to discuss and reach consensus under a selected communication pattern, and then used to produce and evaluate a final answer. The full mesh extension we add covers the densest possible communication case. All experiments are fixed to use four recruited subagents: a mid-range configuration large enough to exhibit fan-out and peer-coordination effects while keeping per-run token volumes within the model's context window.

The orchestrator agent drives all discussion-phase LLM call traffic, dispatching calls to worker agents according to the topology's protocol. In the **sequential** topology agents are called one at a time in a fixed turn order; in the **star** topology a designated solver proposes first, then all remaining agents are dispatched simultaneously as parallel reviewers; in the **full mesh** topology all $N\times(N-1)$ directed peer-message pairs are submitted concurrently each round. Full implementation details, including agent prompts, are available at https://github.com/dlamagna/agentraffic [9].

Table 1: Metrics collected by the measurement framework, organised by architecture layer.

Type	Observed metric	Description
Network	Traffic volume	Byte and packet counts per directed service pair
	Connection events	Open, clean close, reset, and SYN/SYN-ACK timing events
	Traffic shape	Burstiness, short-window arrivals, and inter-arrival jitter
Container	Resource usage	Per-container CPU, memory, and interface byte counters
LLM backend	Inference load	Token throughput, request rate, in-flight requests, and concurrency peaks
Application	Call latency	End-to-end round-trip time as seen by the agent
	Time-to-first-token	Server-side streaming latency
	Inter-arrival time	Wall-clock gap between consecutive LLM calls within a task
	Token counts	Input and output tokens per call

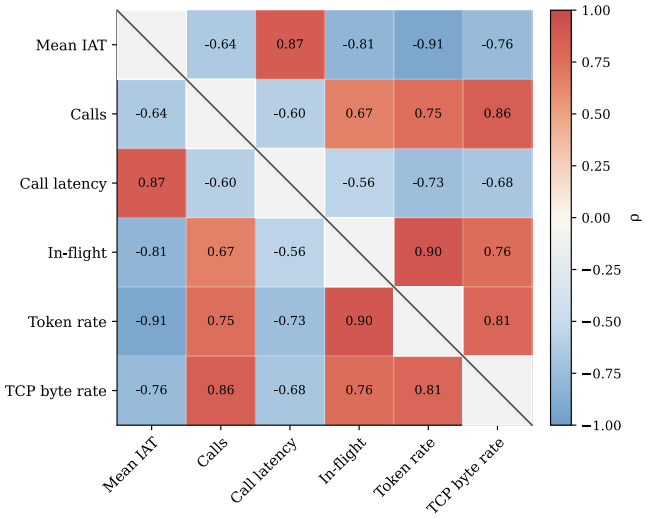

Figure 3: Spearman rank-correlation matrix over selected per-run metrics pooled across Sequential, Star, and Full Mesh topologies.

3.1.2 LLM backend. All agents use a shared inference backend, which exposes HTTP endpoints for model calls. In the experiments reported in this paper, this service runs an AsyncVLLM backend, implemented using vLLM 0.5's AsyncLLMEngine, with Llama-3.2-3B [7]. The asynchronous engine accepts concurrent requests from multiple agents and lets vLLM schedule them for batching on the GPU, making queueing, in-flight request count, and token throughput observable parts of the testbed. For this study, request concurrency and token limits were provisioned to avoid scheduler saturation: the backend was configured with an effective concurrency limit of eight requests, a maximum model length of 11,200 tokens, and a default completion cap of 6,144 tokens.

Table 2: Discussion-stage metrics for Sequential, Star, and Full Mesh topologies. Percent changes are relative to the sequential baseline.

Metric	Sequential	Star	Full Mesh
Mean LLM requests per repetition	14.03	16.22 (+16%)	29.99 (+114%)
Total LLM requests	7,013	8,112	14,993
Discussion call rate	0.208 calls/s	0.209 calls/s (+1%)	1.366 calls/s (+557%)
Duration	81.1 s	105.0 s (+30%)	31.6 s (−61%)
Tokens per run	25.5k	43.5k (+71%)	35.2k (+38%)
Tokens per call	1.80k	2.70k (+50%)	1.10k (−39%)
Mean IAT	4.83 s	4.87 s (+1%)	1.09 s (−77%)
Median IAT	4.49 s	0.40 ms (−100%)	236 ms (−95%)
IAT p_{95}	8.52 s	16.20 s (+90%)	2.82 s (−67%)
Burst fraction (IAT < 50 ms)	0.0%	54.9%	38.2%
LLM latency p_{50}	4.89 s	8.73 s (+79%)	2.78 s (−43%)
LLM latency p_{95}	10.35 s	23.94 s (+131%)	5.10 s (−51%)
Time to first token p_{50}	125 ms	227 ms (+82%)	99 ms (−21%)
Mean LLM concurrency	0.97	1.62 (+67%)	2.94 (+203%)
Peak LLM concurrency	1	4 (+300%)	5 (+400%)
Prompt-token throughput	237 tok/s	318 tok/s (+34%)	1,052 tok/s (+344%)
Completion-token throughput	110 tok/s	178 tok/s (+62%)	364 tok/s (+231%)
LLM TCP bytes per run	50.7k	85.8k (+69%)	38.6k (−24%)
Mean LLM TCP byte rate	617 B/s	768 B/s (+25%)	1.09k B/s (+77%)
Peak LLM TCP byte rate	841 B/s	1.21k B/s (+44%)	1.48k B/s (+76%)

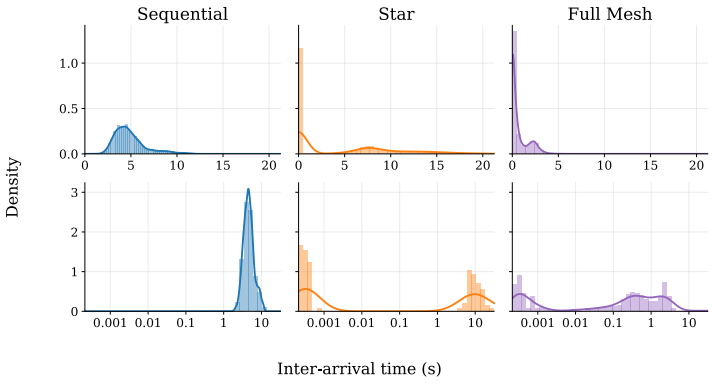

Figure 4: Inter-arrival time distributions for requests to the LLM backend. Rows show linear-scale and log-scale density.

The LLM backend is also where workflow-level context growth can become a serving constraint. Context accumulation can create prompt-token growth larger than the agent-count increase alone, making token volume a workflow-level mechanism that later appears as model-serving latency, throughput, and traffic-volume effects. Context growth was controlled via token-aware prompt trimming and conciseness instructions to keep request sizes within the model window.

3.2 Collected metrics

The framework collects metrics across four complementary layers. Most measurements are collected outside the agent execution path: network and container metrics are observed passively, while application-level traces provide semantic context for each run. Table 1 lists all collected metrics.

All layers share the same agent identifiers and timestamps, so per-run metrics can be joined and correlated across layers directly (as in Figure 3). This multi-layer alignment on a shared time and service-name axis is a key methodological contribution of the framework.

Choice of modelling granularity. We model traffic primarily at the LLM-call/request level rather than at the packet level. In agentic LLM systems, the timing of network load is induced by workflow control flow (agent scheduling, fan-out, and peer coordination). We therefore treat request-level IATs as the primary workload model and use TCP byte rates, connection events, and packet-level counters as cross-layer evidence that request-level topology effects propagate to underlying infrastructure.

4 Experiment design and results

We use IAT (Section 2) as the primary metric to test whether common multi-agent coordination patterns induce distinct arrival processes.

We use the AgentVerse workflow family introduced in Section 3.1.1 and compare three discussion-stage coordination modes: sequential, star, and full mesh. Each topology was repeated 500 times, producing stable IAT distributions. All runs used the same task source, agent implementation, inference backend, and bridge-level instrumentation. Average values displayed in Table 2 are reported from raw samples, whilst the figures and distribution fits use a 3× IQR upper-tail fence to exclude rare extreme gaps that are consistent with model-serving timeouts rather than reasoning phase arrival process.

4.1 Cross-layer comparison

The shift from sequential to star and full mesh is visible throughout the framework: burst fraction and median IAT change at the application layer, concurrency and token throughput change at the inference layer, and TCP byte rate changes at the network layer (Table 2).

Figure 3 reports Spearman rank correlations over per-run metrics from the same topology experiments. Token throughput, in-flight LLM requests, and TCP byte volume move together, while larger inter-arrival times are negatively associated with backend traffic rate. We read these as consistency checks: the four measurement layers are tracking the same topology-induced changes.

4.2 Topology effects on IAT

Figure 4 shows the three topology distributions. Sequential coordination is the baseline, where recruited agents contribute in strict order. The experiments produced 6,513 IAT samples with mean 6.23 s, median 4.56 s, and p95 10.79 s. Figure 4 (left) shows it is a homogeneous multi-second arrival process.

Star and full mesh depart sharply from that baseline. Star produces 7,355 samples of IAT with mean 6.88 s and p95 20.74 s, but its median collapses to 0.41 ms. 53.3% of gaps fall below 50 ms, driven by parallel reviewer dispatch. Full mesh is more compressed in time, with 14,493 samples, mean 1.09 s, median 236 ms, p95 2.82 s, and 38.2% of arrivals below 50 ms; its lower per-call token count (1.10k vs 1.80k) reduces individual call latency, while concurrency drives completion-token throughput from 110 tok/s to 364 tok/s.

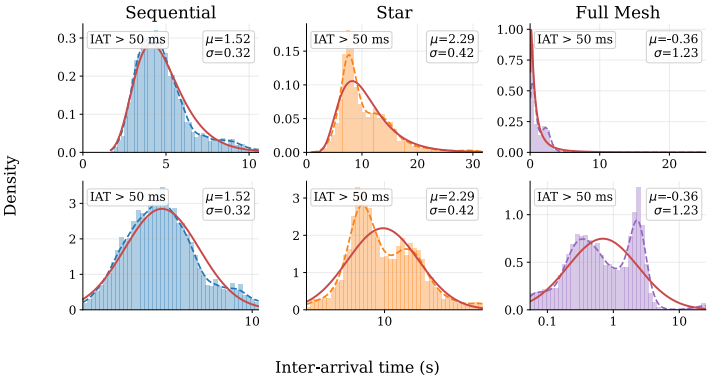

Figure 5: Log-normal fit of reasoning-phase IATs (> 50 ms). Rows show linear and log scale density.

Most notably, the arrival distributions for star and full mesh agentic topologies appear bimodal.

4.3 Bimodality and distribution shape

The bimodality in Figure 4 is a structural effect of subagent fan-out, and produces two distinct peaks. The **fan-out component** (IAT ≤ 50 ms) arises when several calls are dispatched almost simultaneously: in a star round, the central solver concurrently triggers parallel reviewer calls; in full mesh, recruited agents send peer messages concurrently at the beginning of a discussion round. The **reasoning-phase component** (IAT > 50 ms) captures what comes after the initial dispatch, essentially the time spent reasoning by worker agents, and is dictated by model generation time and context accumulation. The 50 ms threshold corresponds to the visible trough between modes in Figure 4.

Sequential traffic is approximately unimodal and is the only case where a single log-normal reference is visually plausible. Star and full mesh are instead better interpreted as mixture distributions rather than a single arrival process. Figure 5 adds a log-normal reference fit after filtering the fan-out mode (IAT ≤ 50 ms), isolating the reasoning-phase component. Section 4.4 formalises this with distribution fits and model selection.

4.4 Goodness-of-fit analysis

Star and full mesh produce bimodal IAT distributions (Section 4.3); fitting a single parametric family to the full sample would confuse structurally distinct components. We therefore restrict fitting to the reasoning-phase component (IAT > 50 ms, Figure 5). We fit three candidate families by maximum likelihood: **Exponential**, the null model for a Poisson (memoryless) arrival process [15]; **Weibull** [17], which generalises Exponential with shape parameter k ; and **Log-normal** [12], which places a Gaussian on log-observations and typically arises from multiplicative processes.

Model selection uses AIC [1] and the KS statistic [13]. At our full sample sizes (n > 3,000) the KS test rejects every candidate [6], so the statistic rather than the p-value is the meaningful shape-fidelity comparison. Table 3 reports both for each topology, whilst

Table 3: Distribution fits to reasoning-phase IATs (IAT > 50 ms) per topology. Best AIC per topology is bold.

Topology	Distribution	n	Parameters	AIC	KS
Sequential	Exponential	6,168	mean = 4.77 s	31,604	0.396
	Weibull		k = 3.09, θ = 5.32 s	23,165	0.091
	Log-normal		σ = 0.31, median= 4.54 s	**21,782**	**0.027**
Star	Exponential	3,303	mean = 10.55 s	22,173	0.359
	Weibull		k = 2.38, θ = 11.92 s	19,131	0.104
	Log-normal		σ = 0.40, median= 9.70 s	**18,377**	**0.077**
Full Mesh	Exponential	8,870	mean = 1.28 s	22,078	0.116
	Weibull		k = 0.88, θ = 1.18 s	21,760	**0.068**
	Log-normal		σ = 1.16, median= 0.67 s	**20,590**	0.089

θ: Weibull scale. σ: log-normal shape (std. dev. in log-space). median = e^{μ} where μ is the log-space mean. KS uses test statistic.

Table 4: KS rejection rate (α = 0.05) across 500 random samples of each size, reasoning-phase IATs (IAT > 50 ms).

n	Distribution	Sequential	Star	Full Mesh
50	Exponential	100%	100%	51%
	Weibull	7%	10%	7%
	Log-normal	1%	2%	1%
100	Exponential	100%	100%	80%
	Weibull	40%	55%	32%
	Log-normal	1%	11%	5%
200	Exponential	100%	100%	98%
	Weibull	93%	98%	76%
	Log-normal	1%	50%	33%

Table 4 shows results from hypothesis testing across different subsample sizes for robustness.

Log-normal achieves the lowest AIC for all three topologies and the smallest KS statistic for sequential and star. However, for full mesh the improvement is not as large, with Weibull achieving a marginally lower KS statistic (0.068 vs. 0.089) despite a higher AIC (21,760 vs. 20,590). Since KS is sensitive to the peak whilst AIC penalises overall fit, this suggests Weibull captures the peak more precisely while log-normal better describes the full distribution.

Subsampling robustness checks confirm the ranking holds at smaller n, with log-normal rejected least frequently, indicating the reasoning-phase arrival process is not memoryless.

5 Discussion

The results show that agentic traffic cannot only be described as a response to external user arrivals. When we varied the coordination topology, the structure of the LLM-call arrival process changed with it. The coordination graph acts as part of the workload generator, determining whether agents wait, fan out, or exchange messages concurrently.

The distribution fitting results add precision to this picture. The reasoning-phase gaps are not memoryless: the exponential is decisively rejected across all topologies, and the log-normal's advantage is consistent with call latency arising as a product of prompt length, context accumulation, output token count, and batching decisions. The multiplicative effect of these factors produces a heavy-tailed, non-Poisson inter-arrival process. This is consistent with prior findings in internet traffic measurement,

where link traffic volumes have been shown to follow log-normal distributions across a wide range of network conditions and timescales [2], suggesting that multiplicative combinations of traffic-generating factors consistently produce this distribution shape across different workload classes. Star and full mesh further complicate modelling by mixing this reasoning-phase component with a structural fan-out mode, meaning no single parametric family can effectively describe the full arrival distribution to the LLM backend.

Implications. These findings bring us closer to defining the level of complexity needed to model agentic LLM deployments. Two systems with the same average task rate may place very different demands on the backend if one executes agents serially and another performs coordinated fan-out. Burst-sensitive summaries, including IAT percentiles, sub-second burst fractions, concurrency peaks, and the mixing fraction between fan-out and reasoning-phase arrivals, are useful complements to aggregate throughput. Coordination metadata also matters for interpreting traces: without knowing the workflow topology, a bursty trace may look like load variation when it is a structural feature of the agent protocol.

This connects traffic measurement to current work on multi-agent system design. Prior studies have treated communication topology as a tradeoff between solution quality, robustness, and token cost [20, 21]. The results here suggest that topology also affects the temporal shape of load on shared infrastructure, which is a further dimension alongside model quality and cost that system designers may want to account for.

The quantitative values in this paper should be read as controlled-testbed measurements. The experiments use one task family, one agent framework, one model, a single-host deployment, and a single bridge-level capture point; the specific parameter values in Table 3 and Table 2 are not intended to generalise beyond this configuration.

6 Conclusion and future work

Conclusion. This paper provided an empirical characterisation of how coordination topology shapes LLM-call arrival processes in multi-agent systems. Across sequential, star, and full-mesh coordination, topology determined whether arrivals formed a homogenous reasoning-phase distribution or a bimodal mixture with a fan-out component. Whilst no single distribution captures the full arrival process, log-normal emerges as the best parametric fit of the reasoning-phase component under all three topologies, with the exponential decisively rejected. Cross-layer measurements confirmed that these structural differences propagate beyond the application layer, affecting inference and network level behaviour.

Future work. The immediate next step is to broaden the workload matrix. Varying model size, model family, and serving configuration would test how backend characteristics interact with the topology-induced arrival patterns across a wider operating range. Adding external tool calls, vector-store queries, or browser actions would introduce additional arrival classes beyond LLM calls. A complementary analytical direction is a semi-Markov model of agent execution states, using state-transition probabilities estimated from collected traces to build a principled generative model of the arrival process.

Open-source release. The framework is open-source at https://github.com/dlamagna/agentraffic [9] and is intended as a starting point for measuring agentic traffic under different agent stacks, model backends, and deployment topologies.

Acknowledgments

This work is part of the I+D+i project titled BLOS-SOMS, grant PID2024-158530OB-I00, and is also partially funded by the Catalan Institution for Research and Advanced Studies (ICREA)

References

[1] Hirotugu Akaike. 1974. A New Look at the Statistical Model Identification. *IEEE Trans. Automat. Control* 19, 6 (1974), 716–723. https://doi.org/10.1109/TAC.1974.1100705

[2] Mohammed Alasmar, Richard Clegg, Nickolay Zakhleniuk, and George Parisis. 2021. Internet Traffic Volumes Are Not Gaussian — They Are Log-Normal: An 18-Year Longitudinal Study with Implications for Modelling and Prediction. *IEEE/ACM Transactions on Networking* 29, 3 (2021), 1266–1279. https://doi.org/10.1109/TNET.2021.3059542

[3] Theophilus Benson, Aditya Akella, and David A. Maltz. 2010. Network Traffic Characteristics of Data Centers in the Wild. In *Proceedings of the 10th ACM SIGCOMM Conference on Internet Measurement (IMC '10)*. ACM, 267–280. https://doi.org/10.1145/1879141.1879175

[4] Weize Chen, Yusheng Su, Jingwei Zuo, Cheng Yang, Chenfei Yuan, Cheng Qian, Chi-Min Chan, Yujia Qin, Yaxi Lu, Ruobing Xie, Zhiyuan Liu, Maosong Sun, and Jie Zhou. 2023. AgentVerse: Facilitating Multi-Agent Collaboration and Exploring Emergent Behaviors. *arXiv preprint arXiv:2308.10848* (2023). No stable peer-reviewed venue at time of writing; cited from arXiv preprint and project repository.

[5] Mark E. Crovella and Azer Bestavros. 1997. Self-Similarity in World Wide Web Traffic: Evidence and Possible Causes. *IEEE/ACM Transactions on Networking* 5, 6 (1997), 835–846. https://doi.org/10.1109/90.650143

[6] Eustasio del Barrio, Hristo Inouzhe, and Carlos Matrán. 2020. On approximate validation of models: A Kolmogorov–Smirnov-based approach. *TEST* 29, 4 (2020), 938–965. https://doi.org/10.1007/s11749-019-00691-1

[7] Abhimanyu Dubey et al. 2024. The Llama 3 Herd of Models. *arXiv preprint arXiv:2407.21783* (2024). Meta AI technical report.

[8] Nataliia Koneva, Alejandro Leonardo García Navarro, Alfonso Sánchez-Macián, José Alberto Hernández, Moshe Zukerman, and Óscar González de Dios. 2025. Introducing Large Language Models as the Next Challenging Internet Traffic Source. arXiv:2504.10688 [cs.NI] https://arxiv.org/abs/2504.10688

[9] Davide Lamagna, Berta Serracanta, Alberto Rodriguez-Natal, Gábor Rétvári, and Albert Cabellos. 2026. AgenTraffic: A Multi-Layer Measurement Framework for Agentic LLM Traffic. https://github.com/dlamagna/agentraffic.

[10] LangChain AI Inc. 2024. *LangGraph: Build Resilient Agents.* https://github.com/langchain-ai/langgraph Accessed May 2026.

[11] Will E. Leland, Murad S. Taqqu, Walter Willinger, and Daniel V. Wilson. 1994. On the Self-Similar Nature of Ethernet Traffic (Extended Version). *IEEE/ACM Transactions on Networking* 2, 1 (1994), 1–15. https://doi.org/10.1109/90.282603

[12] Eckhard Limpert, Werner A. Stahel, and Markus Abbt. 2001. Log-normal Distributions across the Sciences: Keys and Clues. *BioScience* 51, 5 (2001), 341–352. https://doi.org/10.1641/0006-3568(2001)051[0341:LNDATS]2.0.CO;2

[13] Frank J. Massey. 1951. The Kolmogorov-Smirnov Test for Goodness of Fit. *J. Amer. Statist. Assoc.* 46, 253 (1951), 68–78. https://doi.org/10.2307/2280095

[14] João Moura and CrewAI Inc. 2023. *CrewAI: Framework for Orchestrating Role-Playing, Autonomous AI Agents.* https://github.com/crewAIInc/crewAI Accessed May 2026.

[15] Vern Paxson and Sally Floyd. 1995. Wide-Area Traffic: The Failure of Poisson Modeling. *IEEE/ACM Transactions on Networking* 3, 3 (1995), 226–244. https://doi.org/10.1109/90.392383

[16] PwC. 2025. *AI Agent Survey: A New Era of Agentic Work.* Technical Report. PricewaterhouseCoopers. https://www.pwc.com/us/en/tech-effect/ai-analytics/ai-agent-survey.html

[17] Waloddi Weibull. 1951. A Statistical Distribution Function of Wide Applicability. *Journal of Applied Mechanics* 18, 3 (1951), 293–297.

[18] Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, Ahmed Hassan Awadallah, Ryen W. White, Doug Burger, and Chi Wang. 2024. AutoGen: Enabling

Next-Gen LLM Applications via Multi-Agent Conversation. In *First Conference on Language Modeling (COLM 2024)*. https://arxiv.org/abs/2308.08155

[19] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2023. ReAct: Synergizing Reasoning and Acting in Language Models. In *The Eleventh International Conference on Learning Representations (ICLR 2023)*.

[20] Guibin Zhang, Yanwei Yue, Xiangguo Sun, Guancheng Wan, Miao Yu, Junfeng Fang, Kun Wang, Tianlong Chen, and Dawei Cheng. 2024. G-Designer: Architecting Multi-agent Communication Topologies via Graph Neural Networks. arXiv:2410.11782 [cs.MA] https://arxiv.org/abs/2410.11782

[21] Kunlun Zhu, Hongyi Du, Zhaochen Hong, Xiaocheng Yang, Shuyi Guo, Zhe Wang, Zhenhailong Qian, Xuhui Feng, Xinying Du, Chi Wang, and Heng Ji. 2025. MultiAgentBench: Evaluating the Collaboration and Competition of LLM Agents. *arXiv preprint arXiv:2503.01935* (2025). ACL 2025 Main; cited from arXiv preprint until proceedings DOI is final.