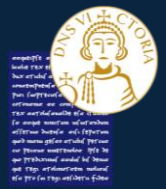
LADbridge

LAnguage-Driven composition of APIs on Distributed Edge Devices

Francesco Cosimo Mazzitelli - Eugenio Zimeo

University of Sannio, Benevento, Italy



github.com/FrancescoMazzitelli/LADbridge

The Challenge: Orchestrating Distributed Edge Services

Device Heterogeneity

IoT & edge environments expose diverse, dynamic REST interfaces with no unified orchestration layer.

Cloud Dependency

Existing LLM-based orchestration relies on centralized cloud, raising privacy and latency concerns.

Rigid Workflows

Traditional gateways use static routing, failing to handle dynamic service availability and multi-step tasks.

→ **LADbridge brings LLM-based reasoning to the Edge, enabling decentralized, language-driven API orchestration**

LLMs On Distributed Systems: Trends & Challenges

Service Orchestration & CloudOps

- **Reasoning-driven**: The ReAct framework, RestGPT and NL2API use static service sets.
- **Infrastructure-as-Code**: IaCGen automates IaC templates from natural language with iterative feedback.
- **CloudOps Automation**: MOYA leverages multi-agent systems for centralized monitoring and management.

Performance & Quantization

- **Edge Benchmarks**: Tests on Raspberry Pi 4 show reduced latency for Qwen2.5 (0.5B) and significant benefits from quantization for LLaMA3.2 (1B).
- **Efficiency**: EdgeProfiler reports up to 3× throughput improvement with 4-bit quantization.
- **Partitioning**: EdgeShard reduces latency by 50% by splitting models across Edge and Cloud.

Edge General Intelligence (EGI)

- **Local Collaboration**: Architectures for drone swarms and smart grids based on perception–reasoning–action loops (Luo et al.).
- **Distributed Coordination**: DisCEdge replicates contextual information to reduce response times across edge nodes.

IoT Security & Communication

- **Security Intent**: CollabIoT translates security intents into YAML policies with 100% accuracy.
- **Semantic Communication**: Bandwidth is optimized by transmitting only relevant semantic tokens extracted from LLMs.

Four Key Research Contributions

01

Distributed NL Orchestration

Framework that translates natural language requests into executable REST workflows across heterogeneous edge services

02

LLM Reasoning at the Edge

Demonstrates that chain-of-thought service composition and execution planning work directly on resource-constrained edge nodes

03

Dynamic Service Discovery

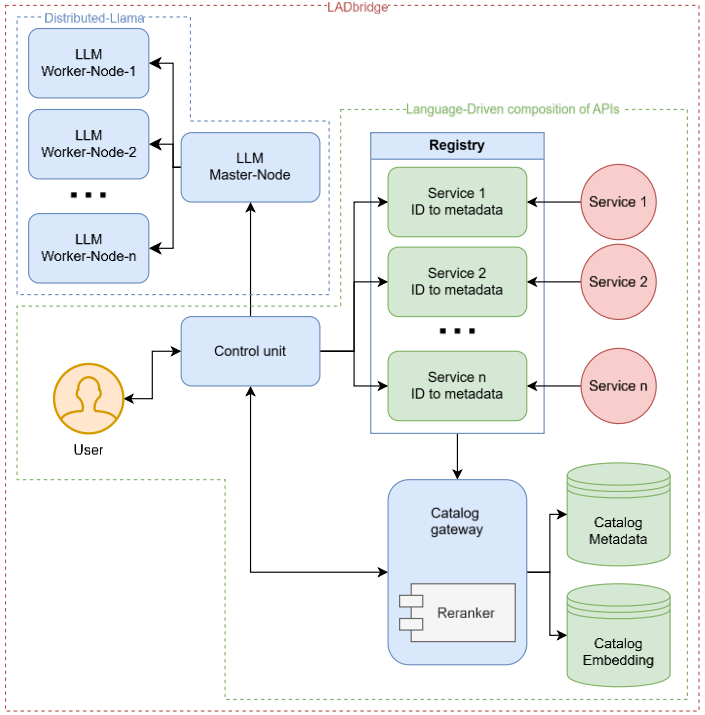
Integrates runtime registry + external RAG catalogs for retrieval-augmented semantic matching and adaptive service selection

04

Empirical Edge Evaluation

Cluster of 9 edge-like VMs validates accurate plan generation and practical latency under normal and degraded network conditions

LADbridge System Design



Request Lifecycle

1. User submits Natural Language query
2. Dual-stage service discovery
3. LLM intent decomposition → JSON plan
4. Synchronous trigger engine dispatches REST calls
5. Response collection + metadata header

Services could be deployed anywhere (cloud / edge / hybrid) — orchestration logic is location-agnostic

Experimental Setup

Cluster Configuration

Nodes	9 virtual machines
Distributed nodes (×8)	4 CPU cores · 8 GB RAM · 50 GB
Centralized node (×1)	32 cores · 64 GB RAM · 200 GB
CPU	AMD EPYC 7742 @ 2.25–3.4 GHz
Configs tested	Centralized · 2 · 4 · 8 VMs
Queries	60 NL questions (CSV)

Model

DeepSeek-R1-Distill-LLaMA-8B
Q4_0 quantization - Distributed-LLaMA format
Chain-of-thought reasoning · 592 words / 811 tokens per generation

Two Evaluation Scenarios

Baseline — Normal network conditions — no artificial delay

Degraded — tc netem 50 ms ± 10 ms injected on all inter-node links (simulates geographically distributed edge)

Plan Generation Accuracy

60/60

Correct Plans

0 partial · 0 incorrect

100%

Plan Accuracy

vs ground-truth oracle

120/120

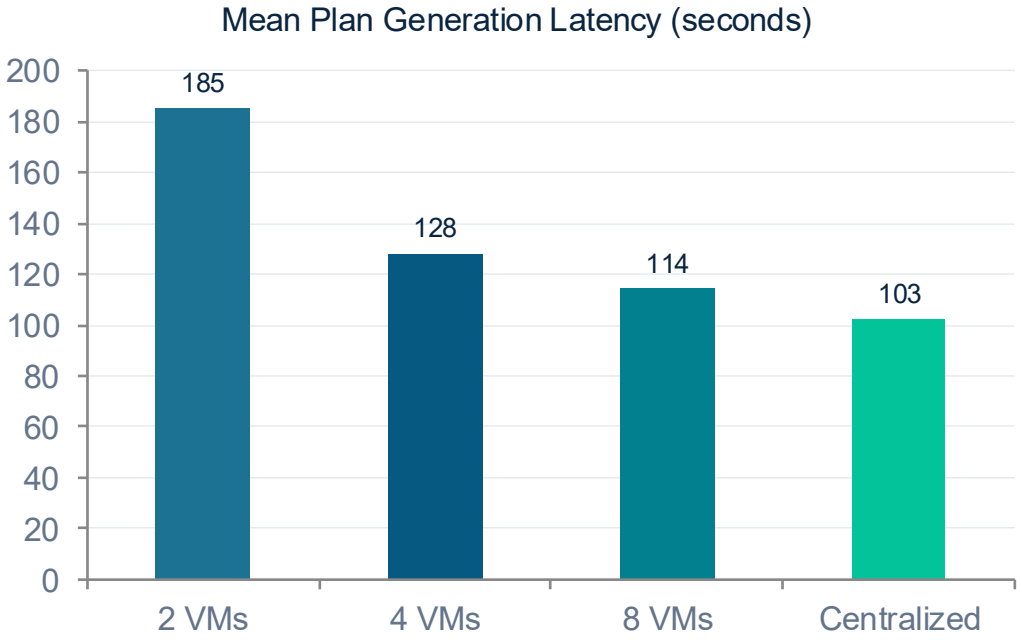
Endpoints Matched

2 per plan × 60 queries

Key Takeaway

LADbridge reliably decomposes natural language requests into correct multi-step execution plans without any cloud dependency. Every query matched the oracle plan, demonstrating that edge-resident LLM reasoning is functionally equivalent to centralized solutions.

Latency — Baseline (No Artificial Delay)



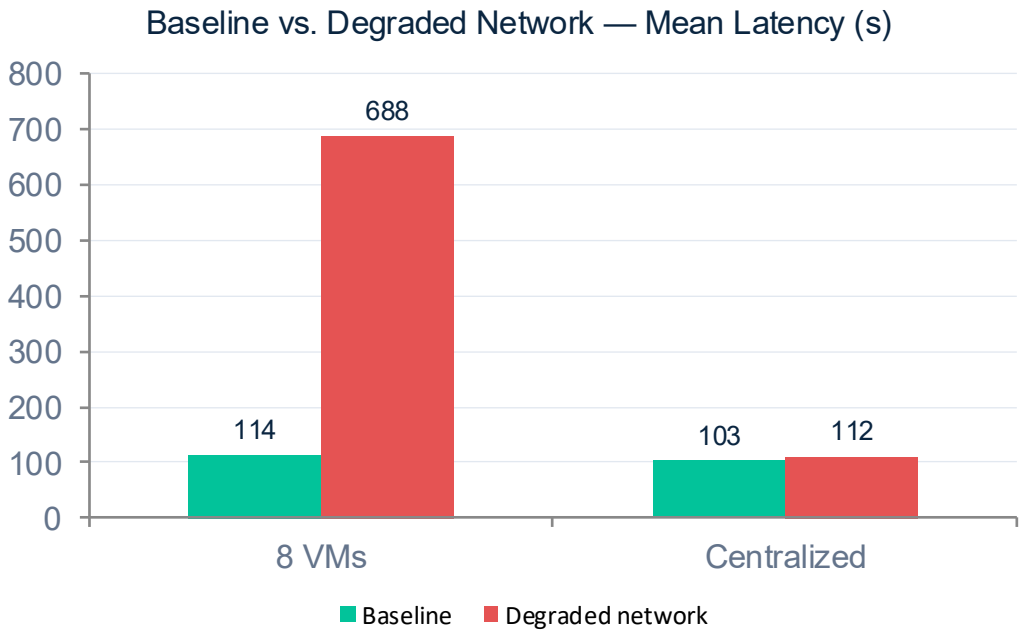
Latency Improvements

2 VM → 4 VM	**–30.9%**
2 VM → 8 VM	**–38.4%**
2 VM → Central	**–44.7%**

8 VMs closely approaches centralized performance with tighter IQR and more predictable latency distribution.

Model: DeepSeek-R1-Distill-LLaMA-8B Q4_0 | Mean 592 words - 811 tokens/generation | Chain-of-thought reasoning

Latency — Degraded Network (50 ms ± 10 ms)



8 VMs — Degraded

687.7 s mean (≈ 6× slowdown)
Sensitive to inter-node communication overhead

Centralized — Degraded

111.6 s mean (stable, +8.8%)
No inter-node comm — unaffected by network jitter

2 VM and 4 VM configurations not shown — connection timeouts under degraded conditions

Limitations & Future Directions

Current Limitations

- Node count constrained to powers of 2 (Distributed-LLaMA)
- Only q4_0 + q8_0 buffer types supported — limits quantization flexibility
- Manually crafted, well-formed query dataset — real noisy input untested
- VMs used instead of physical IoT hardware
- Single-service repeated evaluation — multi-service concurrency not fully explored

Future Work

- Test with noisy / ambiguous user queries
- Support more aggressive quantization formats
- Adaptive strategies to reduce inter-node communication overhead
- Deploy on real physical IoT hardware (Raspberry Pi, Jetson)
- Explore heterogeneous model deployments across edge nodes

CONCLUSION

LADbridge Demonstrates Decentralized LLM Orchestration is Feasible

- 100% plan accuracy across 60 NL queries → all endpoints correctly matched
- 8-VM cluster approaches centralized latency (114 s vs 103 s) under baseline conditions
- Distributed inference is sensitive to network latency → 6× degradation
- Privacy · data locality · autonomy: real advantages of edge-native LLM orchestration

github.com/FrancescoMazzitelli/LADbridge

Thank you · Questions?