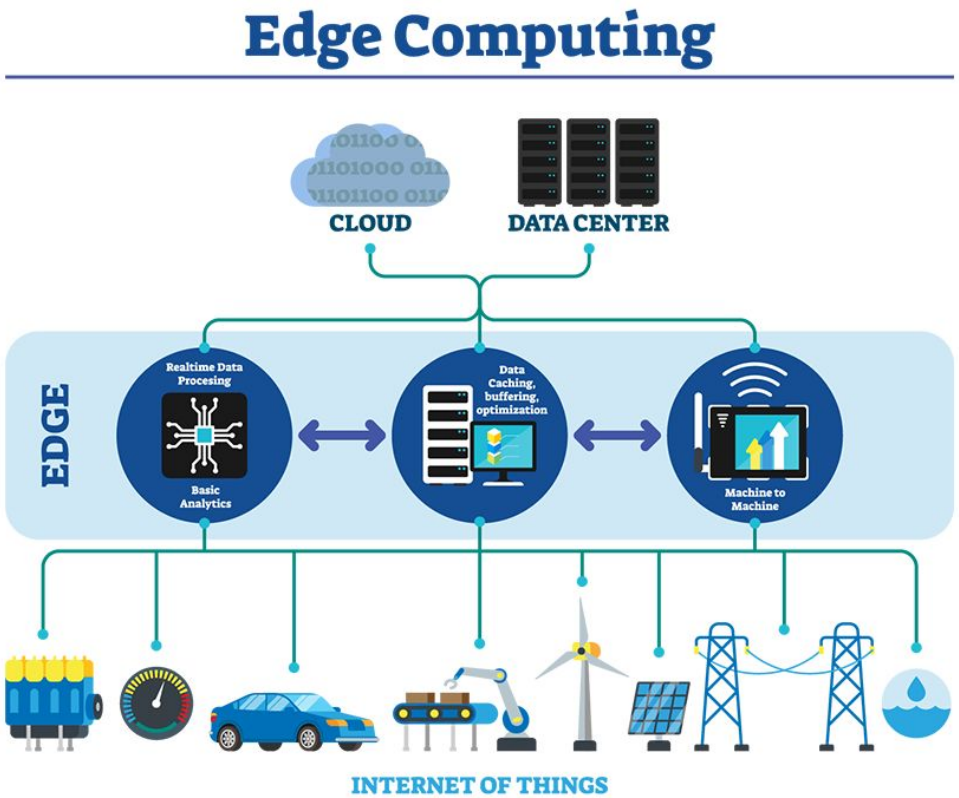
Christian-Albrechts-Universität zu Kiel
Technische Fakultät

Ecoscape as a Validated Edge-Cloud-Continuum Simulator: An LLM Use-Case Integration and Comparative Study

Presentation of Master's Thesis

Motivation



Source: https://innovationatwork.ieee.org/real-life-edge-computing-use-cases/

Motivation

- Real-life testing inefficient and costly
- Many different simulators with different metrics, approaches, …

Motivation

- Real-life testing inefficient and costly
- Many different simulators with different metrics, approaches, …

What makes an edge computing simulator “proper” / “good” ?

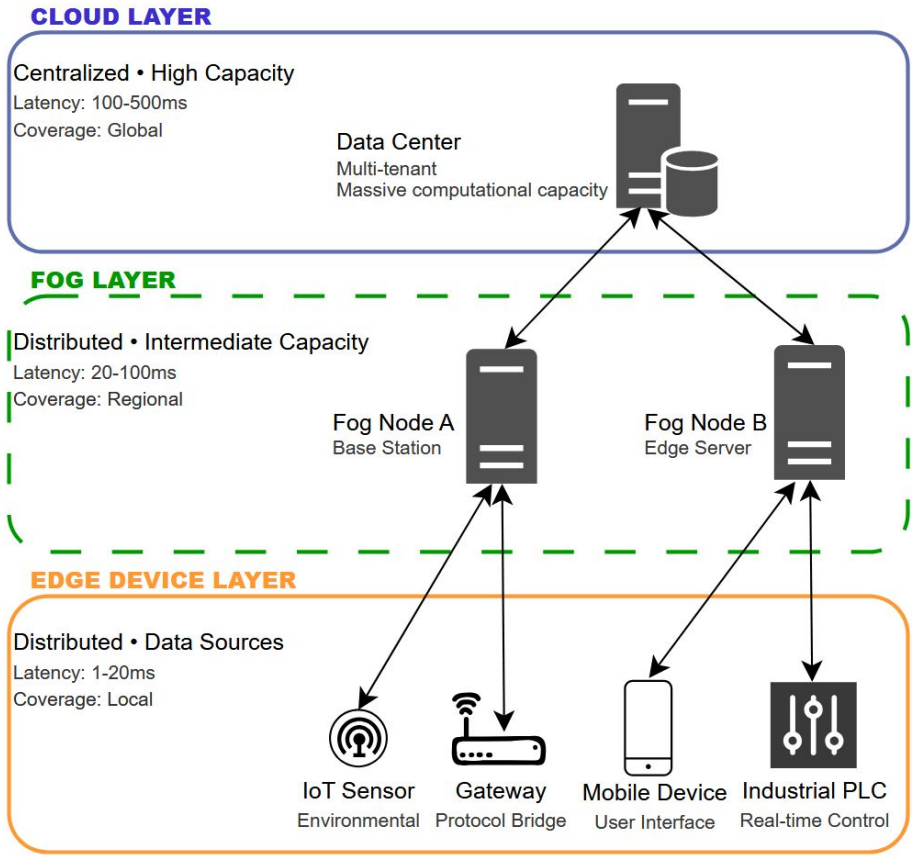
Goals

G1: Definition of Requirements Catalogue

- Criteria an edge computing simulator must meet
  - Features, Metrics, Characteristics, …

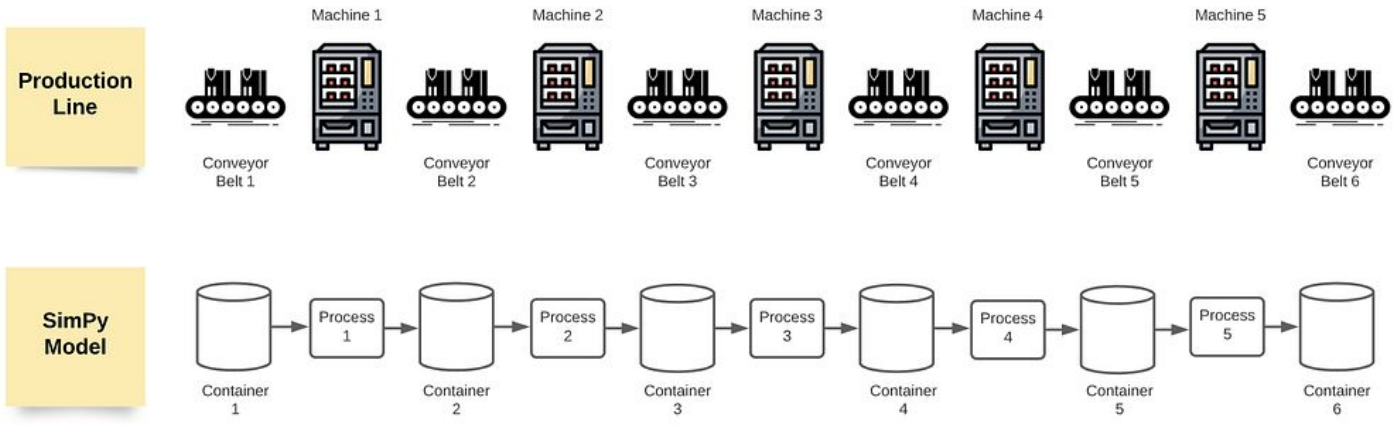
G2: Validating Ecoscape as an Edge Computing Simulator

Foundations : Edge, Fog and Cloud Computing



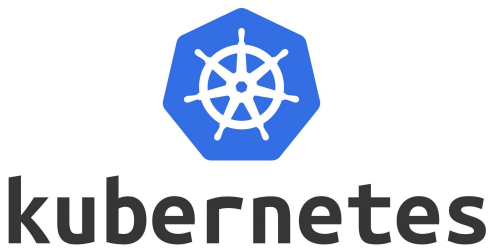
Foundations : Simulation

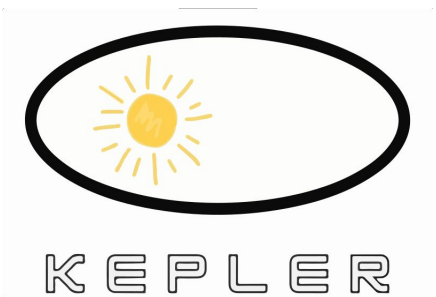
- Discrete Event Simulation (DES)
- Simulation & Emulation



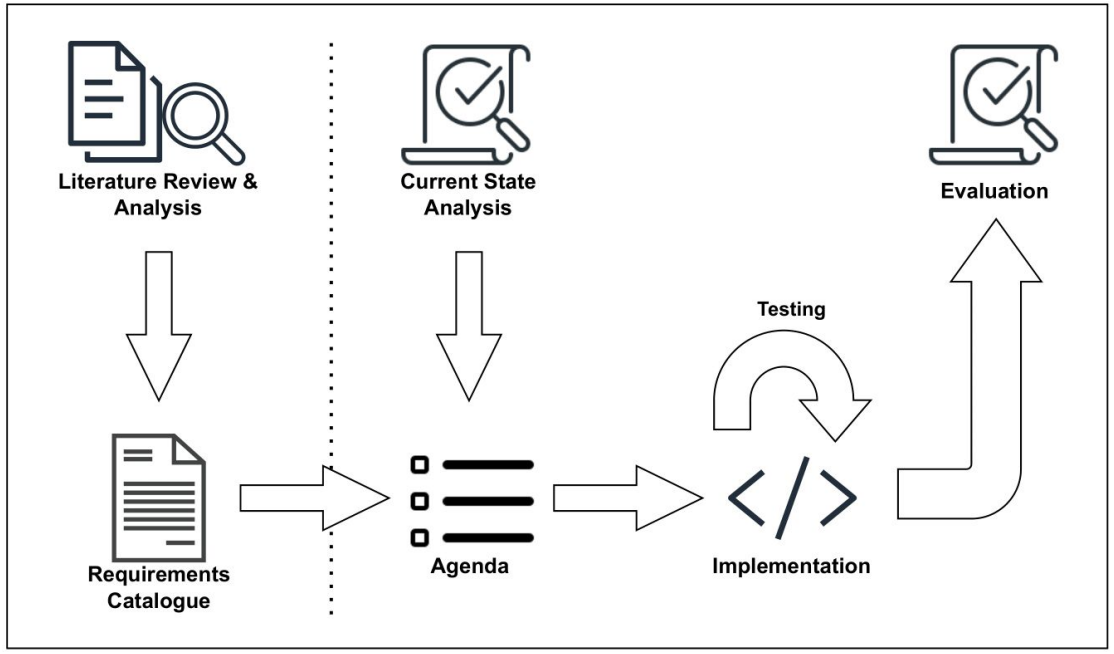
Source: https://medium.com/zebrax/manufacturing-line-optimization-using-discrete-event-simulation-5090ecade303

Foundations : Ecoscape





Approach


Literature Review & Analysis
Current State Analysis
Evaluation
Testing
Requirements Catalogue
Agenda
Implementation

State of the Art - Representatives

- Criteria:
  - Must be available (Published & Open Sourced)
  - Either: Highly cited / well-established or niche / current trend

State of the Art - Representatives

Name	Language	Type	Focus
EdgeCloudSim	Java	Sim	Mobile Edge Computing, General Edge Computing
iFogSim2	Java	Sim	IoT, Mobile Edge Computing, Microservices
YAFS	Python	Sim	IoT, resource allocation, complex networks theory
EmuFog	Kotlin	Emu	Large-scale fog infrastructure
Mockfog 2.0	Node.js	Emu	Real application testing, Cloud-based emulation
Fogify	Python	Emu	Container orchestration, Network emulation
iContinuum	Python	Emu	Orchestration, Large-scale experimentation
FogNetSim++	C++	Sim	Realistic network-centric distributed systems
FogTorchΠ	Java	Sim	IoT application deployment, QoS-aware deployment optimization
EdgeAISim	Python	Sim	AI/ML Edge Computing, Resource management, Energy optimization

State of the Art - Analysis Framework

- Architectural Dimensions
- Functional Capabilities
- Performance & Scalability
- Usability & User Experience
- Validation & Evaluation

State of the Art - Architectural Dimensions

Name	Coverage Scope	Sim. Paradigm	Arch. Pattern
EdgeCloudSim	Cloud-Edge-Mobile	DES	Modular
iFogSim2	Cloud-Fog-Edge-Device	DES	Modular
YAFS	Cloud-Fog-Edge	DES	Modular
EmuFog	Fog-Edge	Emulation	Modular
Mockfog 2.0	Cloud-Fog-Edge	Emulation	Modular
Fogify	Fog-Edge	Emulation	Modular
iContinuum	Cloud-Edge	Emulation	Modular
FogNetSim++	Fog-Edge	DES	Modular
FogTorchΠ	Cloud-Fog-Edge	DES	Modular
EdgeAISim	Cloud-Fog-Edge	DES	Modular

Insight: Highly Extensible (Modular), Cover multiple layers, DES

State of the Art - Functional Capabilities

- General:
  - Network Modeling
  - Data Generation
  - Application & Workload Modeling
  - Device & Infrastructure Modeling
  - Fault Tolerance
- Additionally:
  - Mobility Modeling
  - Energy Consumption
  - Security & Privacy

State of the Art - Functional Capabilities (Network)

Name	Network Types	Supported Protocols	Topology Support
EdgeCloudSim	WLAN, WAN, Cellular	HTTP, Custom	Hierarchical
iFogSim2	WiFi, Ethernet, Cellular, Custom	TCP/IP, HTTP, MQTT, Custom	Hierarchical, Custom
YAFS	Generic/Custom	Generic/Custom	Any
EmuFog	WiFi, Ethernet	TCP/IP, HTTP, Custom	Custom, Real-world datasets
Mockfog 2.0	Any	Any	Hierarchical, Custom
Fogify	Ethernet, Wifi, Cellular	TCP, HTTP, MQTT	Hierarchical, Mesh, Custom
iContinuum	Ethernet, Wifi, Cellular	TCP, HTTP, MQTT	Hierarchical, Mesh
FogNetSim++	WiFi, Ethernet, Cellular, WAN, LAN, Custom	TCP/IP, UDP, HTTP, Custom	Hierarchical, Mesh, Tree, Custom
FogTorchΠ	Abstract	Abstract	Hierarchical, Custom
EdgeAISim	Ethernet, WiFi, Cellular	TCP/IP, HTTP, MQTT	Hierarchical, Custom

State of the Art - Functional Capabilities (Network)

Name	Bandwidth Modeling	Delay Modeling
EdgeCloudSim	Congestion-aware, configurable per link	Distance-based, dynamic
iFogSim2	Variable, configurable per link/node	Propagation, queueing, processing delays
YAFS	Configurable, supports dynamic changes	Configurable, supports dynamic changes
EmuFog	Realistic	Realistic
Mockfog 2.0	Realistic	Realistic
Fogify	Realistic	Realistic or configurable
iContinuum	Configurable per link	Configurable per link
FogNetSim++	Configurable, packet-level	Configurable, packet-level
FogTorchII	Probabilistic	Probabilistic
EdgeAISim	Configurable per link	Configurable per link

Insight: Configurable Network behaviour, Ethernet / WiFi or Focus-based network types and protocols

State of the Art - Functional Capabilities (Data Generation)

Name	Data Generation	Data Flow Models
EdgeCloudSim	Synthetic	Mobile data patterns, Location-based
iFogSim2	Synthetic, Integration	Periodic, Bursty, IoT streams
YAFS	Synthetic, Integration	Stream processing, Batch jobs
EmuFog	Integration	Real data processing
MockFog 2.0	Integration	Real-world data integration
Fogify	Integration	Persistent volumes, Config maps
iContinuum	Integration	Real-time data routing
FogNetSim++	Synthetic	Data aggregation, Filtering
FogTorchΠ	Limited Synthetic	Model I/O, Batch processing
EdgeAISim	Sinthetic, Integration	Training data, Inference patterns

Insight: At least synthetic, Integration recommended; Authentic data flow

State of the Art - Functional Capabilities (Application)

Name	Application Architecture	Task Dependencies
EdgeCloudSim	Mobile application patterns, Back- and Foreground task classification	Parent-child relationships
iFogSim2	Monolithic, Microservice	DAG, Precedence constraints
YAFS	Flexible computational graphs	Configurable graph-based
EmuFog	Real Containerized	Docker container linking
MockFog 2.0	Container Orchestration, Real microservices	Container orchestration
Fogify	Kubernetes, Pod-based	Kubernetes service meshes
iContinuum	Abstract Services, Real containerized applications	Cross-service communication
FogNetSim++	Fog-specific	DAG
FogTorchΠ	ML inference containers	ML pipeline chains
EdgeAISim	Neural networks, Federated learning patterns	AI pipeline dependencies

Insight: Tasks schedulable, Resource allocation mechanisms

State of the Art - Functional Capabilities (Device)

- Emulation: Container Limits
- Simulation: Heterogeneous device modeling (CPU, Memory, Storage)

Insight: Customizable devices with clearly stated resource limits

State of the Art - Functional Capabilities (Fault Tolerance)

Name	Failure Types	Detection Methods	Recovery Strategies
EdgeCloudSim	Mobile device, Network failure	Timeout	Task migration
iFogSim2	Device, Network failure	Heartbeat, Timeout	Basic restart
YAFS	Configurable failures	Custom detection	Configurable recovery
EmuFog	Container failure	Docker health checks	Container restart
MockFog 2.0	Container/Service failure	Orchestration monitoring	Auto-restart, Isolation
Fogify	Pod/Service failure	Kubernetes probes, Monitoring	Kubernetes healing, Migration
iContinuum	Service failure	Cross-tier monitoring	Service migration
FogNetSim++	Hierarchical failure	Fog-aware monitoring	Redundancy, Failover
FogTorchΠ	Model deployment failure	Health checks	Model redeployment
EdgeAISim	Model degradation, Byzantine	AI-aware detection	Model recovery, Consensus

Insight: Failure types need to cover all entities, Detection methods and recovery strategies must be presented

State of the Art - Functional Capabilities

- General:
  - Network Modeling
  - Data Generation
  - Application & Workload Modeling
  - Device & Infrastructure Modeling
  - Fault Tolerance
- Additionally:
  - Mobility Modeling
  - Energy Consumption
  - Security & Privacy

State of the Art - Functional Capabilities (Mobility)

Name	Mobility Models	Handover Support	Location Services	Dynamic Topology
EdgeCloudSim	Advance cellular models	Advanced	Location-based services	User mobility
iFogSim2	Basic mobility models	Limited	Basic location-awareness	Node mobility
YAFS	Limited	None	None	Dynamic placement
EmuFog	None	None	None	None
MockFog 2.0	None	None	None	None
Fogify	None	None	None	None
iContinuum	None	None	None	None
FogNetSim++	Limited	None	None	Limited
FogTorchΠ	None	None	None	None
EdgeAISim	Limited	None	None	Limited

Insight: Mobility is focus-based; Highly recommended because of IoT

State of the Art - Functional Capabilities (Energy)

Name	Energy Modeling Scope	Optimization Support	Granularity
EdgeCloudSim	None	None	None
iFogSim2	Basic device-level energy modeling	Energy-aware placement	Per device, per task, limited accuracy
YAFS	None	None	None
EmuFog	None	None	System, per container
MockFog 2.0	None	None	System, per container
Fogify	None	None	System, per container
iContinuum	None	None	System, per container
FogNetSim++	None	None	None
FogTorchΠ	None	None	None
EdgeAISim	AI/ML workload energy modeling	Energy optimization	Per task, per device

Insight: Energy Consumption Modeling is scarce; Recommended as it is a valid metric and concern

State of the Art - Functional Capabilities (Security / Privacy)

Name	Security Mechanisms	Privacy Support	Attack Modeling
EdgeCloudSim	Secure offloading, Transmission encryption	None	None
iFogSim2	Authentication, Encryption overhead, Trust models	Basic data anonymization	Limited, custom extensions possible
YAFS	None, custom extension required	None	None, custom extensionr equired
EmuFog	None, relies on Docker/Mininet	None	None
MockFog 2.0	None	None	None
Fogify	None	None	None
iContinuum	None	None	None
FogNetSim++	None, relies on OMNeT++/INET	None	None
FogTorchΠ	None	None	None
EdgeAISim	None, can simulate AI-based anomaly detection	None	None

Insight: Often abstracted to the applications itself

State of the Art - Performance & Scalability

- Problem: Benchmarking is scarce and often misleading
- Comparison Papers & each tool's own paper made some effort
- Claims like: "iFogSim2 has a 28% better memory usage than its competitors" but no proof

Insight: Benchmarking is difficult; Scalability should cover large amount of nodes

State of the Art - Usability & User Experience

- Standardized setup process for all tools
- Mostly written instructions, 2-5 examples
- Configuration via standardized files (XML, YAML, …)
- Complexity is mostly based on used technologies and frameworks (e.g. FogNetSim++)

State of the Art - Usability & User Experience

- Standardized setup process for all tools
- Mostly written instructions, 2-5 examples
- Configuration via standardized files (XML, YAML, …)
- Complexity is mostly based on used technologies and frameworks (e.g. FogNetSim++)

Outliers:

- YAFS developed using Python 2.7
- EdgeAISim's setup instructions outdated
- EmuFog requires Linux-specific networking configurations

Insight: Configurable via standardized files, Setup instructions, High Extensibility

State of the Art - Validation & Evaluation

- Customizable Metrics
- Source differs: Simulator vs Application
- Visualization often via Grafana (K8s), MATLAB or Matplotlib
- Results exported to readable output
- Reproducible Scenarios (Scenario Management / Seed Control)
- Validation present as validation studies

Insight: Customizable metrics, reproducible scenarios, visualization / exported results , Error handling

State of the Art - Analysis Framework

- Architectural Dimensions
- Functional Capabilities
- Performance & Scalability
- Usability & User Experience
- Validation & Evaluation

Requirements Catalogue

- Architectural Dimensions — Ecoscape
    - Mandatory:
        - High extensibility via modular design
        - Discrete Event Simulation
        - Multi-Tier Coverage (Recommended: Cloud-Edge)

Requirements Catalogue

	Ecoscape
- Architectural Dimensions	
- Mandatory:	
- High extensibility via modular design	✓
- Discrete Event Simulation	✓
- Multi-Tier Coverage (Recommended: Cloud-Edge)	✓

Requirements Catalogue

- Functional Capabilities — Ecoscape
  - Mandatory:
    - Network Modeling: Configurable network structure & behaviour
    - Device & Infrastructure Modeling: Configurable resource constraints
    - Application Workload Modeling: Schedulable tasks, resource allocation
    - Data Generation: Synthetic (Recommended: Additional Integration)
    - Fault Tolerance: Cover all entities with failure types, detection and recovery strategies
  - Recommended:
    - Energy Consumption: Realistic approximation
    - Mobility Support: Influence on network behavior based on realistic patterns

Requirements Catalogue

- Functional Capabilities — Ecoscape
  - Mandatory:
    - Network Modeling: Configurable network structure & behaviour ✓
    - Device & Infrastructure Modeling: Configurable resource constraints ✓
    - Application Workload Modeling: Schedulable tasks, resource allocation ✓
    - Data Generation: Synthetic (Recommended: Additional Integration) ✓
    - Fault Tolerance: Cover all entities with failure types, detection and recovery strategies ✓
  - Recommended:
    - Energy Consumption: Realistic approximation ✓
    - Mobility Support: Influence on network behavior based on realistic patterns X

Requirements Catalogue

- Performance & Scalability — Ecoscape
    - Mandatory:
        - Basic Scalability: Robust scenarios with a minimum of 1 000 nodes
        - Execution Control: Deterministically & reproducible scenarios
    - Recommended:
        - Advanced Scalability: ~10 000 nodes

Requirements Catalogue

	Ecoscape
- Performance & Scalability	
- Mandatory:	
- Basic Scalability: Robust scenarios with a minimum of 1 000 nodes	?
- Execution Control: Deterministically & reproducible scenarios	✓
- Recommended:	
- Advanced Scalability: ~10 000 nodes	?

Requirements Catalogue

- Usability & User Experience — Ecoscape
    - Mandatory:
        - Documentation: 2-5 examples, documentation for development, setup and usage
        - Configuration: Clear configurations via standardized file formats

Requirements Catalogue

	Ecoscape
- Usability & User Experience	
- Mandatory:	
- Documentation: 2-5 examples, documentation for development, setup and usage	X
- Configuration: Clear configurations via standardized file formats	✓

Requirements Catalogue

- Validation & Evaluation
    - Mandatory:
        - Customizable Metrics
        - Export to readable file format
        - Visualization possible
        - Error handling

Ecoscape

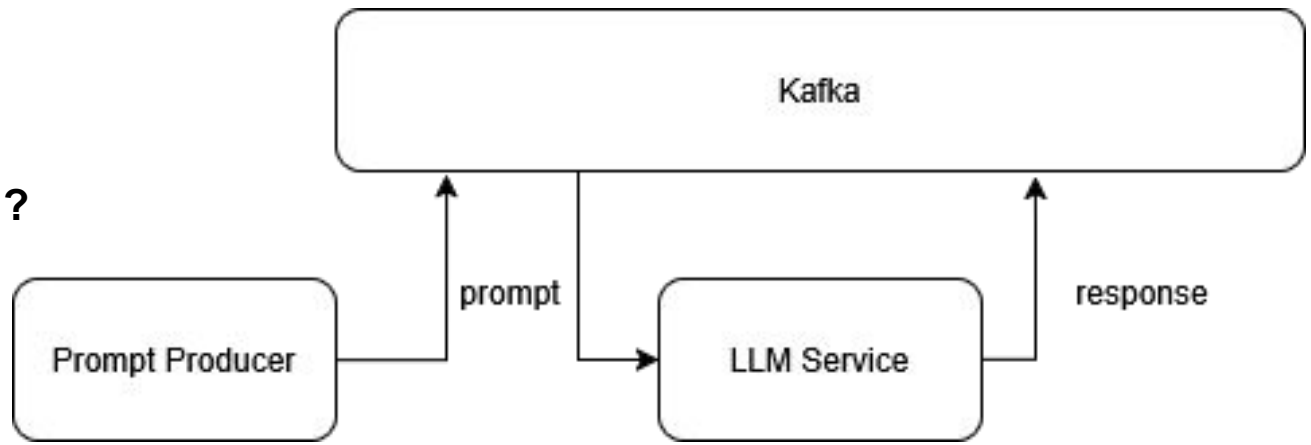
Requirements Catalogue

	Ecoscape
- Validation & Evaluation	
- Mandatory:	
- Customizable Metrics	✓
- Export to readable file format	✓
- Visualization possible	✓
- Error handling	✓

Use Case

- Original Use Case: Object Recognition
- Addition: LLM

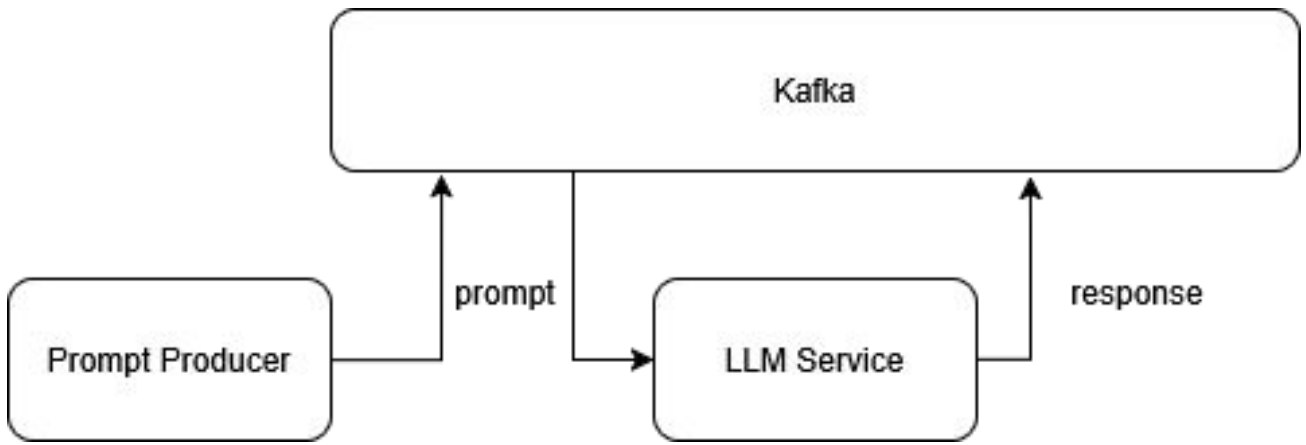
Can Ecoscape handle LLM workload ?



Preloader

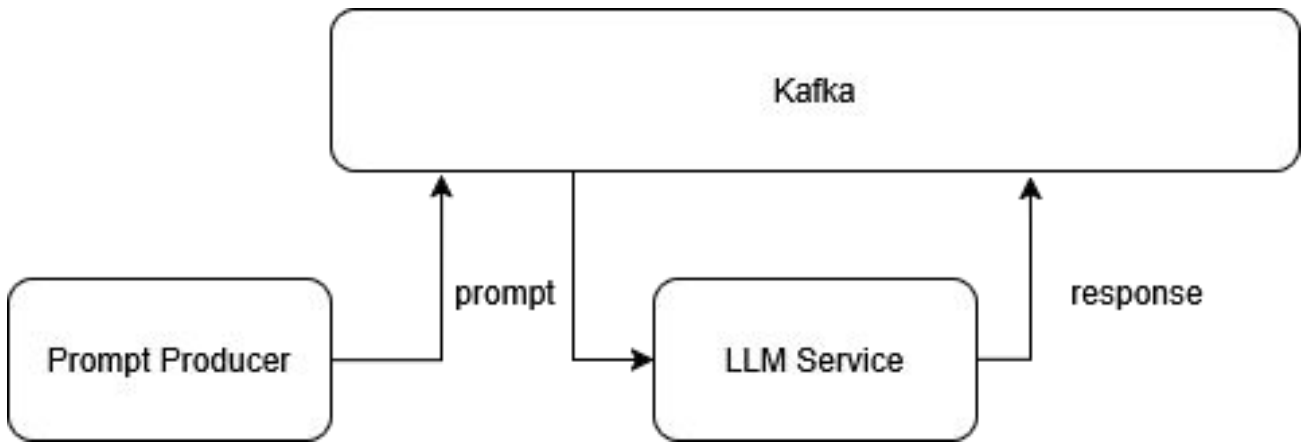
- Persistent Volume Claim (PVC)
- Preloader Job

- Dataset: Alpaca (Stanford)
- Model: Qwen 2.5 - 0.5B Instruct
  (Alibaba Cloud)



Prompt Producer

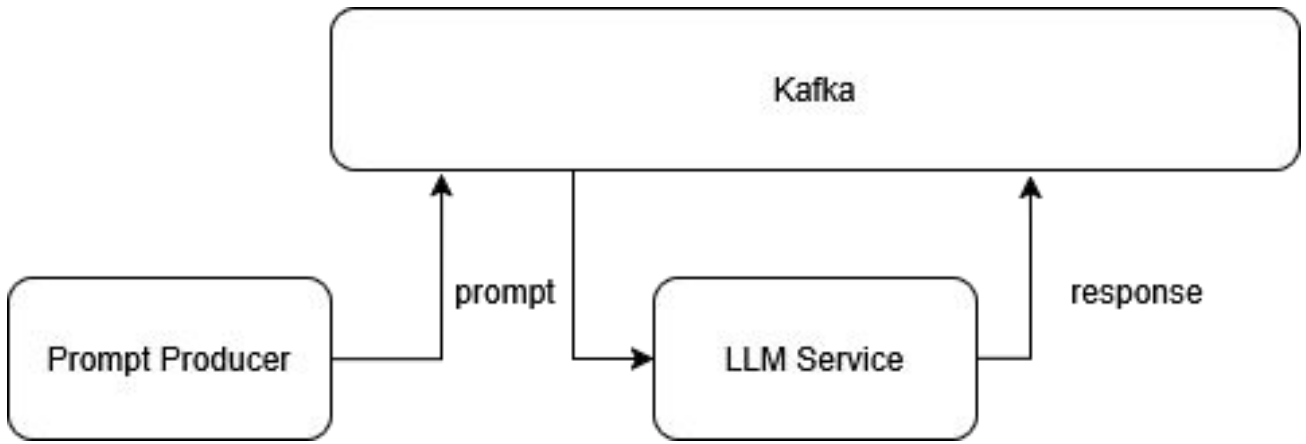
- Load Patterns: Constant, Poisson, Burst, Ramp Up
- Requests per Second (RPS) : 1 prompt per 5s
- Prompts with 10 - 500 tokens



LLM Service

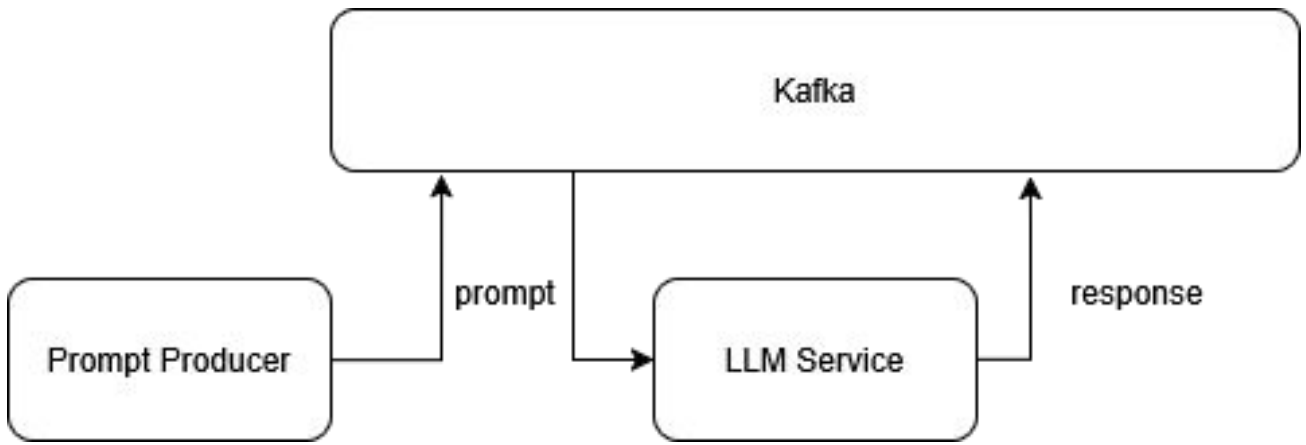
- Temperature: 0 = Strict, 1 = Creative
- Max Tokens : Length of response in tokens
- CPU vs GPU mode (Torch)

- Metrics:
  - CPU Utilization
  - Processing Latency P95
  - TTFT P95
  - TTPS

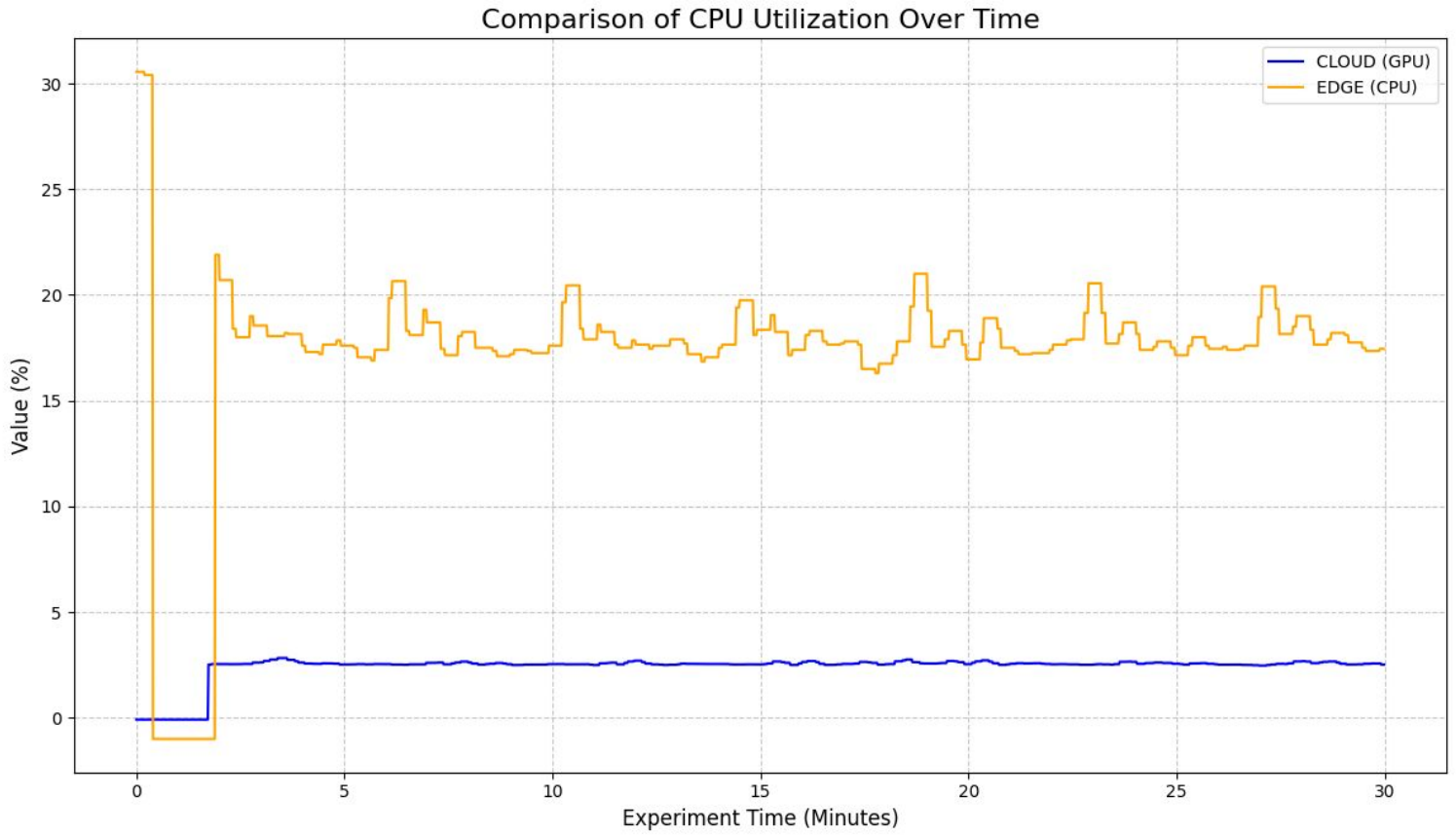


Experiments

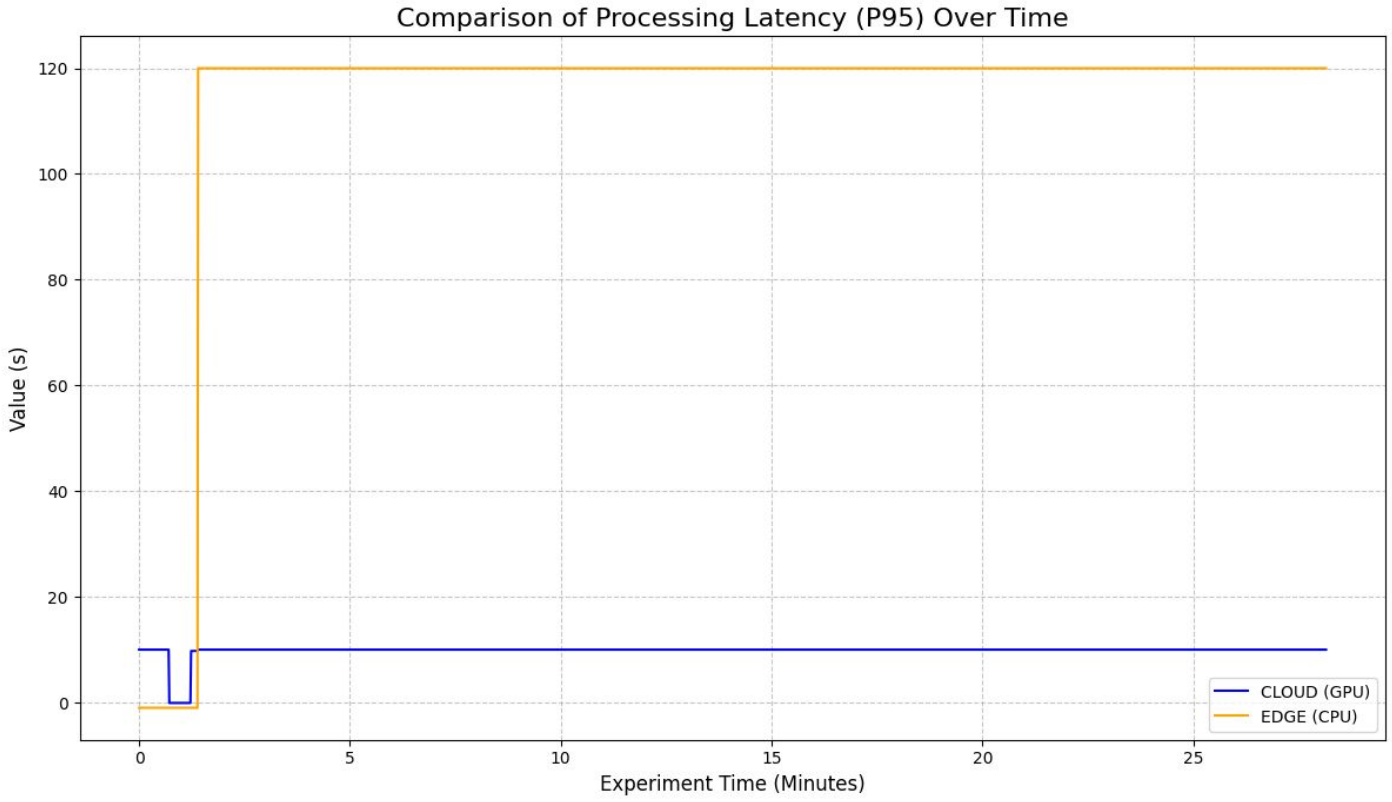
- Multiple Scenarios
- GPU: NVIDIA RTX 4070 Ti Super (16 GB)
- Repetition: 5 times
- Duration: 30 min
- Variable max_tokens and RPS



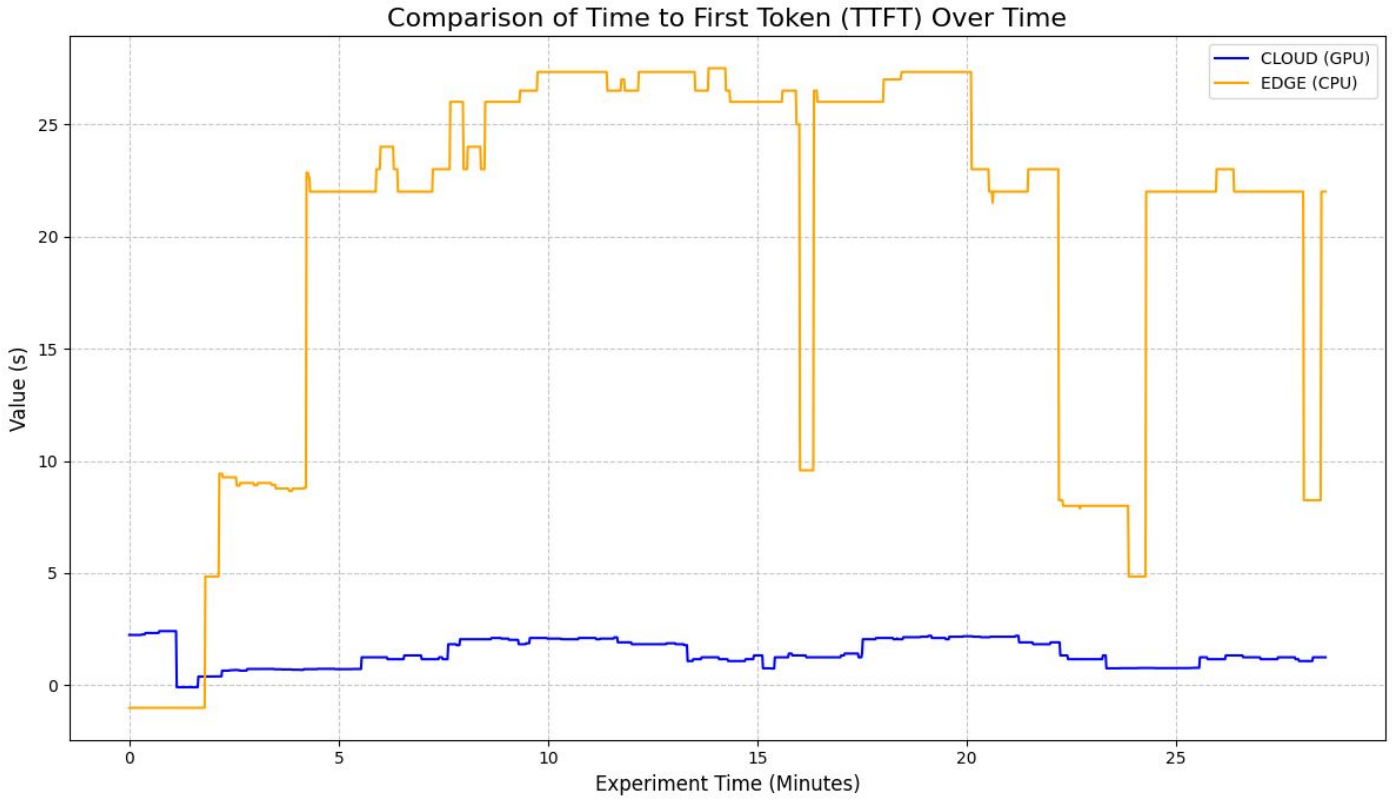
Experiments - Heterogeneous Deployment



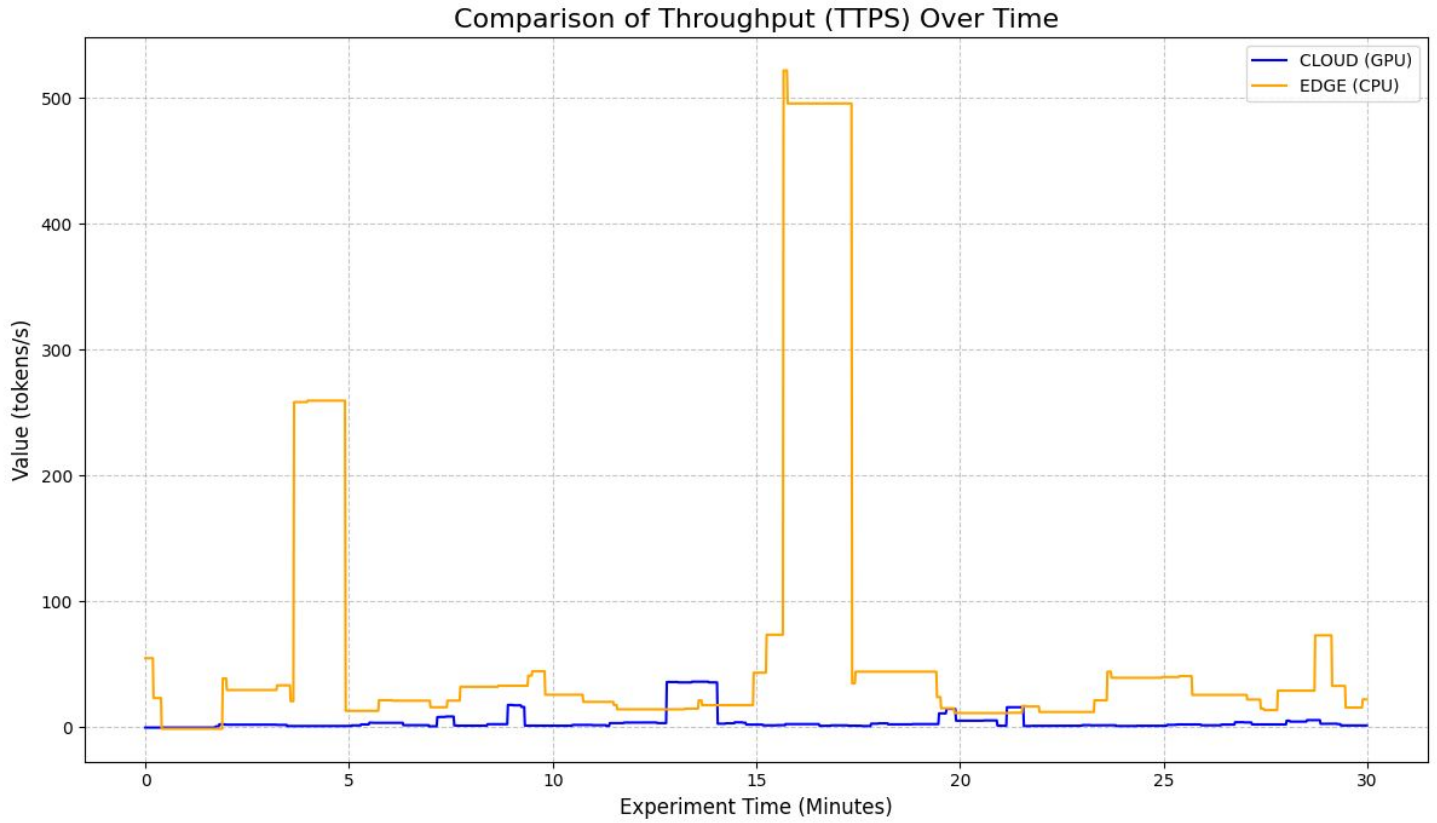
Experiments - Heterogeneous Deployment



Experiments - Heterogeneous Deployment



Experiments - Heterogeneous Deployment



Conclusion