SIMULATION-BASED ANALYSIS OF LATENCY AND ENERGY EFFICIENCY IN SMART ENVIRONMENT NETWORKS (ENHANCED VERSION)

Turayeva Shaxlo Nazimjonovna
TATU universiteti 2-bosqich talabasi

Abstract

Smart Environments (SE) are rapidly evolving due to the expansion of Internet of Things (IoT) technologies, requiring efficient network organization to ensure optimal system performance. This paper presents a simulation-based analysis of latency and energy efficiency in Smart Environment networks. A Python-based simulation model is developed to evaluate multiple network architectures, including flat topology, clustered topology, and edge-based computing. Mathematical models for latency, energy consumption, and packet delivery ratio are introduced. The results demonstrate that clustering significantly reduces energy consumption, while edge computing minimizes latency. A hybrid architecture combining clustering and edge processing achieves the best performance in terms of scalability and efficiency.

Keywords: Smart Environment, IoT, Simulation, Latency, Energy Efficiency, Clustering, Edge Computing

1. Introduction

Smart Environments (SE) integrate IoT devices, sensors, and communication systems to create intelligent and adaptive infrastructures. These systems generate large volumes of data that must be processed efficiently.

Key challenges include:

- High **latency**
- Excessive **energy consumption**
- Limited resources of IoT devices

Traditional **flat architectures** suffer from scalability issues, while modern approaches such as **clustering** and **edge computing** improve performance.

2. Mathematical Modeling

2.1 Network Representation

The Smart Environment network is modeled as a graph:

$$G = (V, E)$$

- V — set of nodes (sensors, gateways, cloud)
- E — communication links

2.2 Latency Model

Latency is defined as:

L=Ttx+Tproc+Tqueue

Where:

- Ttx— transmission delay
- Tproc — processing delay
- Tqueue — queuing delay

Interpretation:

- Flat topology → large Ttx
- Edge computing → reduces Tproc
- Clustering → reduces Tqueue

2.3 Energy Consumption Model

Energy consumption is calculated as:

$$E = \sum_{i=1}^{n} (E_{tx,i} + E_{rx,i} + E_{proc,i})$$

Where:

- Etx — transmission energy
- E_{rx} — receiving energy
- E_{proc} — processing energy

Insight:

- Flat → high E_{tx}
- Clustered → reduced transmissions
- Edge → optimized processing energy
- **2.4 Packet Delivery Ratio (PDR)**

$$PDR = \frac{P_{received}}{P_{sent}}$$

- This measures network reliability.

3. Simulation Methodology

A simulation model was developed in Python with the following components:

- Sensor nodes
- Cluster heads
- Gateway (edge nodes)
- Cloud server

Scenarios Evaluated

1. Flat topology
2. Clustered topology
3. Edge-based processing

4. Hybrid (Cluster + Edge)

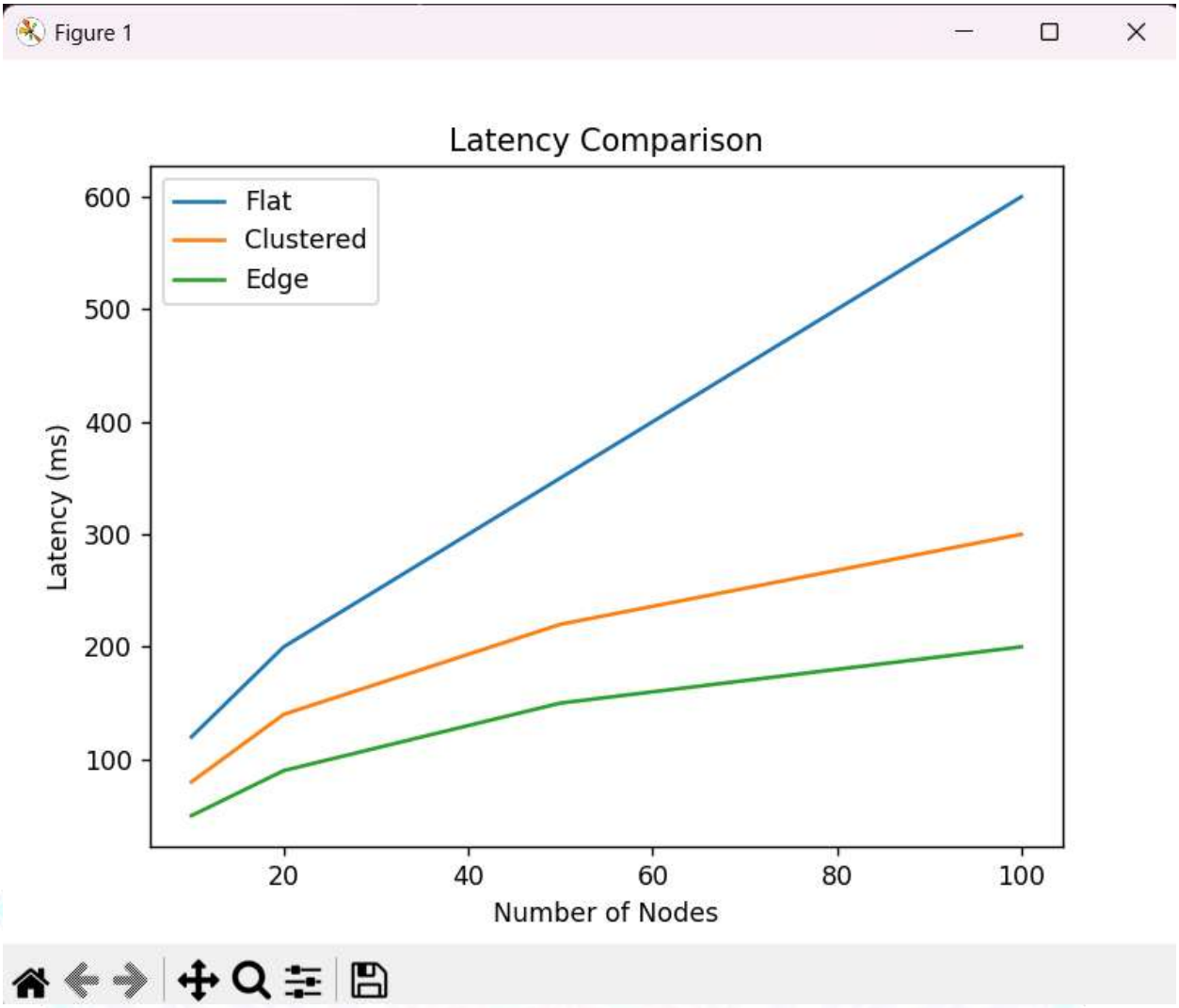
3.1 Simulation Algorithm (Pseudo-code)

```
for each timestep:
    generate sensor data
    if clustered:
        send to cluster head
        aggregate data
    if edge:
        process locally
    else:
        send to cloud
    calculate latency
    calculate energy
    update metrics
```

Incode:

```
import matplotlib.pyplot as plt
nodes = [10, 20, 50, 100]
latency_flat = [120, 200, 350, 600]
latency_cluster = [80, 140, 220, 300]
latency_edge = [50, 90, 150, 200]
plt.plot(nodes, latency_flat, label="Flat")
plt.plot(nodes, latency_cluster, label="Clustered")
plt.plot(nodes, latency_edge, label="Edge")
plt.xlabel("Number of Nodes")
plt.ylabel("Latency (ms)")
plt.title("Latency Comparison")
plt.legend()
plt.show()
```

4. Results and Graphical Analysis



4.1 Latency Comparison

Expected graph:

- X-axis → Number of nodes
- Y-axis → Latency

Trend:

- Flat → highest
- Clustered → medium
- Edge → lowest

4.2 Energy Consumption

Expected graph:

- X-axis → Time / Nodes
- Y-axis → Energy

Trend:

- Flat → highest energy
- Clustered → reduced energy
- Hybrid → optimal

4.3 Performance Table

Model	Latency	Energy	Efficiency
Flat	High	High	Low
Clustered	Medium	Low	Good
Edge	Low	Medium	Very Good
Hybrid	Lowest	Lowest	Best

5. Discussion

Simulation results indicate:

- **Clustering** reduces redundant transmissions → saves energy
- **Edge computing** reduces communication distance → lowers latency
- **Hybrid approach** provides optimal balance

6. Conclusion

This study demonstrates that:

- Flat architectures are inefficient
- Clustering improves scalability
- Edge computing enhances responsiveness
- Hybrid models achieve the best overall performance