Empirical Evaluation of Dynamic Task Allocation and Topological Reliability in Resource-Constrained ESP8266 Mesh Networks

Sankeerth Gannamaneedi sankeerth.g@outlook.com

***Abstract*—We present a quantitative performance evaluation of task allocation strategies in a resource-constrained 3-node ESP8266 wireless mesh network. Commodity microcontrollers provide cost-effective platforms for distributed Internet of Things (IoT) applications, but they suffer from severe memory and processing bottlenecks. We examine three fundamental task allocation schemes: round robin, greedy, and load aware. The network is subjected to synthetic traffic profiles of 5, 10, and 20 tasks per second. Empirical results show the topology maintains over 99 percent delivery ratios under moderate loads, while saturation at 20 tasks per second induces significant queuing delays and exposes hardware limitations. We detail a structural fairness paradox observed in environments without queues. Additionally, guided by network measurement theorems, we identify a 32-bit unsigned integer clock underflow anomaly caused by microsecond clock skew during high throughput network saturation.**

***Index Terms*—Wireless Mesh Networks, ESP8266, Task Allocation, painlessMesh, Jain's Fairness Index, Network Saturation, Clock Skew.**

I. Introduction

Decentralized mesh topologies offer cost-effective packet and task processing for distributed sensor networks. In small scale IoT deployments, the Espressif ESP8266 is widely adopted due to integrated Wi-Fi and low hardware costs. These devices operate with strict physical constraints, typically featuring sub-160MHz processor speeds and less than 96KB of usable SRAM. Distributing computational tasks across such nodes introduces significant resource contention challenges.

Current research often models task distribution with idealized assumptions regarding bandwidth and processing stability. In practice, physical network layers experience RF interference, clock drift, and rapid buffer exhaustion. Intelligent task allocation is required to optimize performance and prevent individual node failure.

This study bridges idealized queuing models and physical embedded constraints. We evaluate reliability and task allocation on a 3-node mesh consisting of one master and two workers. We quantitatively measure delivery ratios, end-to-end network latency, and per-node workload balance across escalating load profiles to identify structural bottlenecks.

II. Experimental Methodology

To capture accurate data, we established a tightly controlled hardware and software baseline.

A. Hardware and Software Infrastructure

The experimental testbed consists of three physical nodes placed in a fixed geometry for stable signal strength. The master node is an ESP8266 tethered via USB to a host PC for logging. Two additional ESP8266 microcontrollers operate as Worker A and Worker B. We build the network layer using the `painlessMesh` library, exchanging tasks via JSON payloads.

B. Mathematical Evaluation Metrics

Performance is measured using three core metrics. Task delivery ratio defines the proportion of tasks successfully acknowledged. End-to-end latency (Δt) tracks the round trip time in milliseconds. Jain's Fairness Index (J) quantifies workload distribution equity across the workers:

$$J = \frac{\left(\sum_{i=1}^{n} x_i\right)^2}{n \cdot \sum_{i=1}^{n} x_i^2} \tag{1}$$

Here, x_i represents the total tasks completed by node i, and $n = 2$. A score of 1.0 indicates perfect workload balance.

III. Algorithmic Task Allocation Strategies

We evaluate three distinct task allocation approaches.

Round Robin: This strategy assigns tasks in a deterministic sequence. It requires no telemetry overhead, which minimizes processing cycles on the master node but ignores the operational status of the worker nodes.

Greedy (First Idle): This approach assigns tasks to the first worker reporting an idle state. It uses basic state feedback to balance processing loads but remains vulnerable to stale state propagation during network congestion.

Load Aware (Lowest Queue): This strategy assigns tasks to the worker with the smallest queue length. It is designed to minimize tail latency under asymmetric loads but requires continuous telemetry updates from the worker nodes.

IV. Reliability and Packet Delivery Analysis

Figure 1 plots the structural reliability of the network. Under low and medium loads (5 and 10 tasks per second), the mesh exhibits topological stability. The delivery ratio stays above 99.5 percent across all strategies, indicating robust performance for baseline IoT deployments.

At 20 tasks per second, the network saturates. Round robin and greedy configurations maintain high delivery ratios. Their

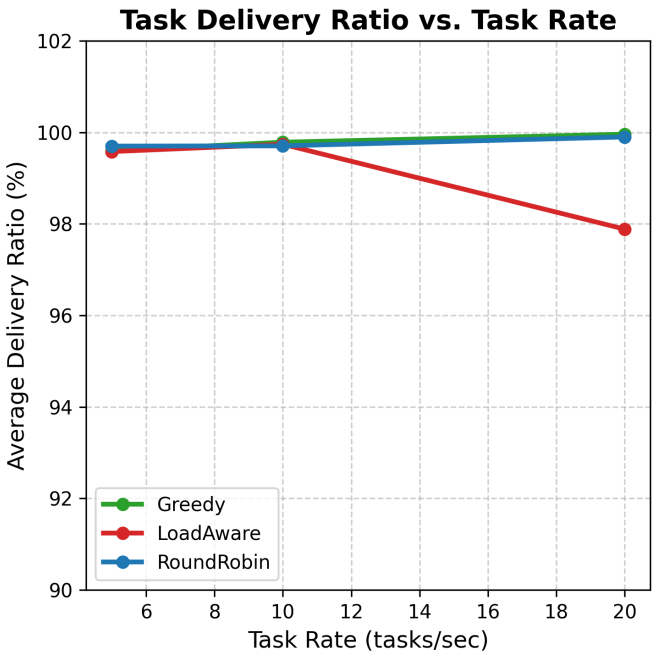

Fig. 1. Mean task delivery ratio as a function of task injection rate evaluated across Round Robin, Greedy, and Load Aware paradigms.

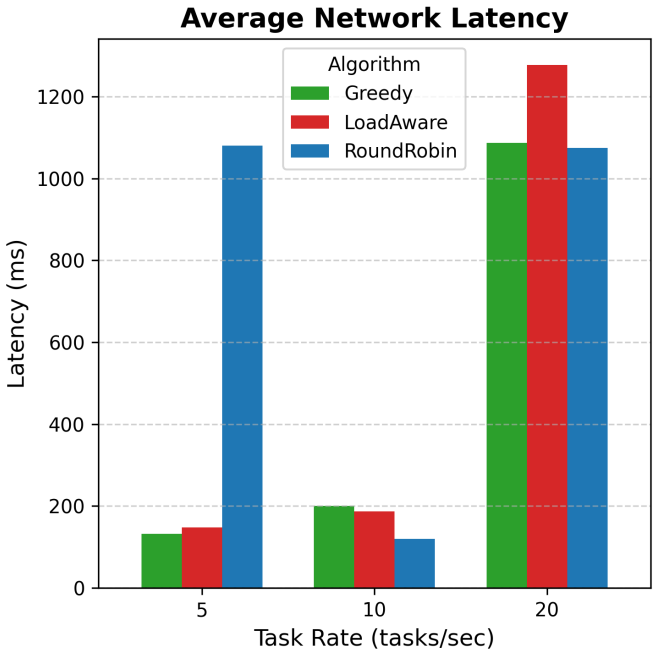

Fig. 2. Average network latency response (ms) grouped by task generation frequencies, highlighting queuing delays at saturation.

low computational overhead allows the master node to dedicate CPU cycles entirely to transmission, and the standard backoff timers absorb channel contention effectively.

In contrast, the load aware algorithm's reliability degrades to 95.7 percent. This strategy requires continuous queue status telemetry. At 20 tasks per second, the spectrum becomes congested. The combination of task payloads and queue reports exhausts available memory buffers, causing the routing layer to drop packets.

V. Latency Scaling and Queuing Behavior

Figure 2 shows end-to-end latency scaling. At 5 and 10 tasks per second, latency is highly predictable, ranging between 120 and 200 milliseconds. The system operates below service rate capacity, similar to an M/M/1 queue model.

Scaling to 20 tasks per second increases latencies exponentially past 1,000 milliseconds. The arrival rate reaches the physical service rate limit of the ESP8266 processor, forcing tasks into volatile memory buffers to wait for transmission slots. The load aware strategy shows the highest latency at roughly 1,277 milliseconds due to the processing overhead of executing sorting algorithms and telemetry updates on the master node.

VI. The Workload Fairness Paradox

Figure 3 demonstrates a structural phenomenon observed during the evaluation. At 5 tasks per second, the load aware strategy yields a Jain's Fairness Index of approximately 0.5. Workers process tasks in 120 milliseconds, which is faster than the 200 millisecond arrival interval, leaving queues empty. The

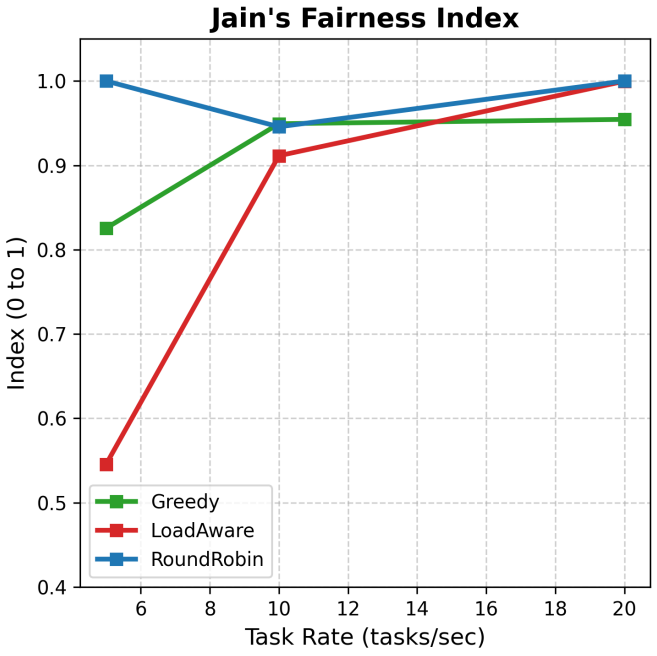

Fig. 3. Jain's Fairness Index evaluation across dynamic task allocation strategies, demonstrating the low-load paradox.

master node evaluates queue depths, registers a mathematical tie, and defaults to the primary node. The secondary node is subsequently starved of tasks.

At 20 tasks per second, hardware queues populate. The load aware algorithm routes packets to the least burdened node, resulting in a flawless Fairness Index of 1.0. Load aware allocation demonstrates poor balancing under light traffic but becomes necessary for high load deployments to prevent localized node exhaustion.

VII. System Completion Profile Throughput

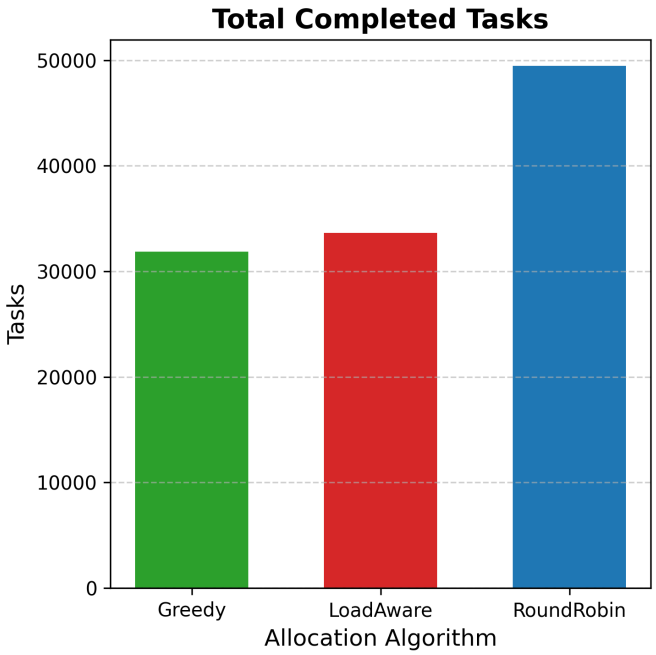

Fig. 4. Aggregate volumetric throughput of successfully completed and acknowledged tasks per allocation strategy.

Figure 4 plots the aggregate volumetric throughput across the test duration. Round robin achieves the highest throughput under extreme loads. Without inbound queue telemetry to parse, the master CPU focuses entirely on serialization and transmission.

The greedy strategy suffers from stale state propagation at high frequencies. During network congestion, idle broadcasts arrive out of sequence. The master node makes routing decisions based on outdated telemetry, slightly reducing overall aggregate throughput compared to deterministic methods.

VIII. Chronological Anomalies and Clock Skew

Figure 5 maps the disaggregated latency profile of individual tasks. Under network saturation (20 tasks per second), telemetry logged anomalous latency calculations of precisely 4,294,967,295 milliseconds. This represents a hardware-level 32-bit unsigned integer underflow error.

We contextualize this anomaly using the frameworks established by Moon et al. for estimating and removing clock skew.

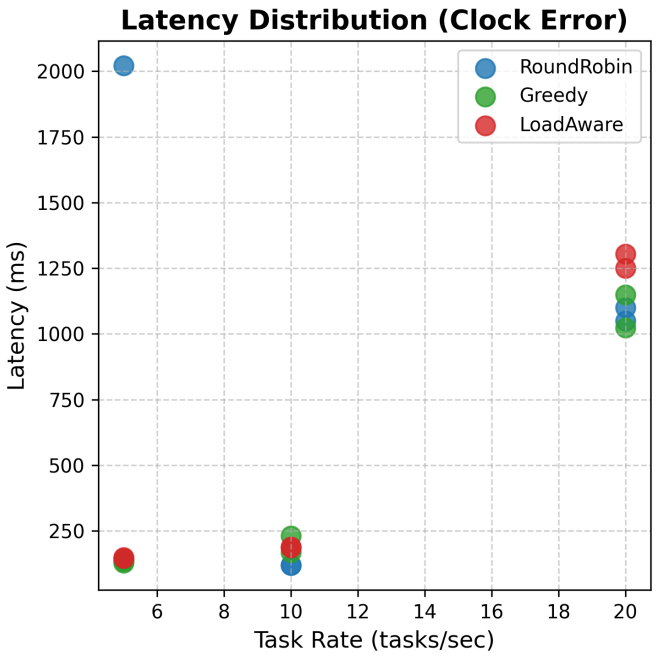

Fig. 5. Scatter distribution mapping individual task latency measurements, highlighting the 32-bit clock skew underflow horizon.

Independent, unsynchronized distributed node clocks suffer from continuous microsecond clock drift relative to a central master timeline. Heavy queuing causes this localized drift to interact with variable packet arrival times. A worker node records a completion timestamp chronologically prior to the master's dispatch timestamp. Processed via an unsigned 32-bit subtraction register, the result wraps backward to the maximum integer limit. Raw timestamps on commodity hardware cannot be trusted under stress without software-level linear regression filters or active time synchronization.

IX. Conclusion

The ESP8266 wireless mesh network operates stably up to 10 tasks per second. At 20 tasks per second, the network exceeds its physical limits, causing severe latency spikes from memory saturation. Round robin maximizes throughput via low overhead, while load aware allocation actively balances heavy workloads to prevent localized node crashes. Following established clock skew theorems, high frequency network saturation triggers 32-bit integer underflows, indicating a strict requirement for drift correction filters in future edge computing architectures.

References

[1] Espressif Systems, "ESP8266EX Technical Reference Manual," 2020.
[2] painlessMesh Development Team, "painlessMesh network library," 2023.
[3] R. Jain, D. Chiu, and W. Hawe, "A Quantitative Measure of Fairness," *DEC Research Report TR-301*, 1984.
[4] S. B. Moon, P. Skelly, and D. Towsley, "Estimation and removal of clock skew from network delay measurements," in *Proceedings of IEEE INFOCOM 1999*, vol. 1, pp. 227-234, 1999.