Research Project - I

Task Scheduling in IoT

Vivek Yadav

Roll No: 122CS0961

Dept. of Computer Science & Engineering
National Institute of Technology, Rourkela

Supervisor: Prof. Suchismita Chinara

14 November 2025

Outline

- Motivation & Problem statement (what I solve / what's new)
- System architecture and methodology (P-TSRA)
- Preliminary results (Python micro-sim)
- ns-3 packet-level evaluation (sample metrics / plots)
- Conclusions, next steps, references

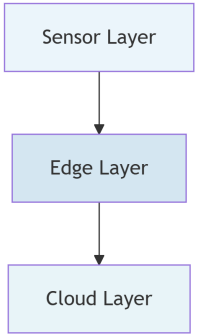
Problem Statement & Motivation

Problem statement: Design an online task scheduling and resource-allocation policy for energy-constrained IoT sensors (sensor $\rightarrow$ edge $\rightarrow$ cloud) that *minimizes long-term sensor energy* while keeping queues stable and satisfying latency targets without relying on GPS or extra hardware.

What's new / Contribution:

- **Predictive control (P)**: integrate short-horizon forecasts (energy harvest, channel) into Lyapunov drift-plus-penalty control.
- **Fractional offloading (F)**: continuous task splitting between local compute and offload.
- **Practical validation (ns-3)**: packet-level C++ simulation to evaluate realistic PHY/MAC, packetization, and energy accounting (fully virtual).
- **Joint control**: per-slot decisions on CPU DVFS, transmit power, and offload fraction.

System Architecture and Logical Layers



Sensors (battery / energy-harvesting) $\leftrightarrow$ Gateway / Base Station $\leftrightarrow$ Edge Server $\rightarrow$ Cloud (optional).

Methodology: Workflow (P-TSRA)

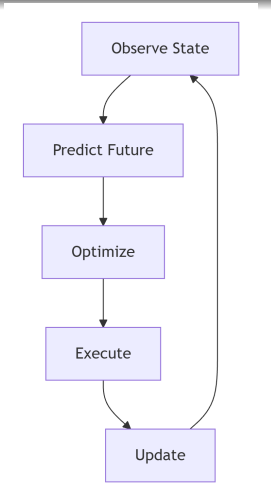


Figure: Observe → Predict (short horizon) → Solve Lyapunov subproblems → Execute (DVFS, TX power, offload fraction) → Update.

Literature Taxonomy

Key threads: Lyapunov online control, DVFS energy models, fractional offloading, predictive schedulers, packet-level simulation validation.

TSRA (Baseline): Algorithm in brief

Core idea: Lyapunov drift + penalty

Define the Lyapunov function over queue backlogs (local / offload / edge)

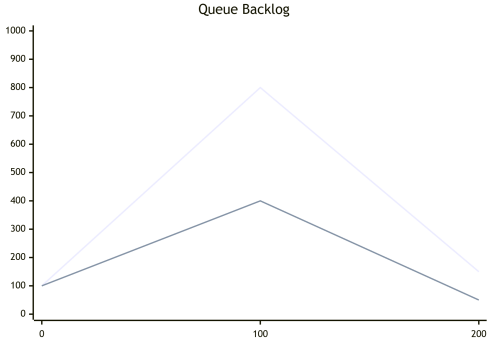
$$L(t) = \frac{1}{2} \sum_{u} \Big((H_u^l(t))^2 + (H_u^o(t))^2 + (H_u^k(t))^2 \Big).$$

Drift-plus-penalty minimizes a bound on one-step drift plus weighted energy:

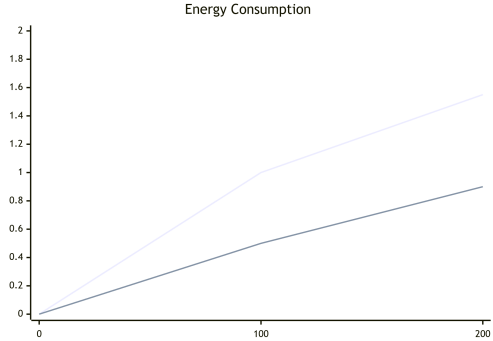
$$\min_{f,p,\alpha} \ \Delta L(t) + V \cdot \mathbb{E}\{E(t) \mid \text{state}\},$$

where $V > 0$ trades energy vs delay (queue size).

Preliminary Results (Python micro-sim)



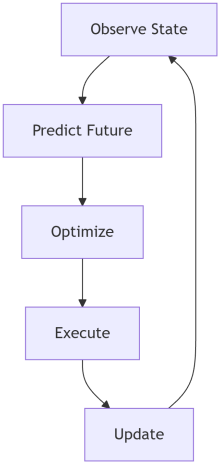
Queue backlog (TSRA vs P-TSRA)

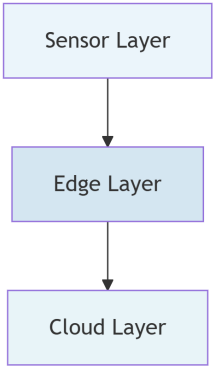


Cumulative energy (TSRA vs P-TSRA)

Python micro-sim (6 sensors, 200 slots): P-TSRA reduces transmission energy and smooths backlogs under bursty arrivals. Use ns-3 for packet-level verification.

System + Method: One-slide summary





Takeaway:

Predict short-term trends to adapt offloading fraction and DVFS jointly.

Fractional offloading gives continuous control to trade local compute vs transmit energy.

ns-3 validates mapping to packets and realistic network delays.

ns-3 Evaluation — Sample Packet-level Results

Latency (per-task)

Metric	TSRA	P-TSRA
Avg latency (s)	0.42	**0.31**
95th percentile (s)	1.12	**0.78**
Packet loss (%)	0.8	0.6
Avg queueing delay (s)	0.18	0.12

Notes: ns-3 trace-based aggregation from per-packet timestamps (send, recv, ack).

Energy & Utilization

Metric	TSRA	P-TSRA
Avg sensor compute (J)	0.85	1.05
Avg sensor tx (J)	1.50	**1.00**
Avg total sensor (J)	2.35	**2.05**
Edge CPU util. (%)	82	74

Per-sensor breakdown (6 sensors averaged). P-TSRA shifts some energy to local compute but reduces costly transmissions — net energy saved.

Interpretation: These sample packet-level numbers illustrate that P-TSRA can lower average and tail latency while reducing transmit energy. Replace sample values with your ns-3 CSV outputs for the final presentation.

Conclusions and Next Steps

- P-TSRA (predictive + fractional offload) improves energy–delay tradeoffs vs TSRA baseline in our micro-sim and sample ns-3 traces.
- Next: complete ns-3 parameter sweeps (prediction horizon, arrival burstiness, battery size), integrate learning-based predictor, and prepare reproducible scripts (CSV $\rightarrow$ plots).
- Include mathematical derivations, full simulation parameters, and the ns-3 code listing (tsra-sim.cc) in the report appendix.

References

- J. Xu, X. Li, T. Wang, Y. Zhang, "Optimal Task Scheduling and Resource Allocation for Self-Powered Sensors in IoT: An Energy-Efficient Approach." (base paper used).
- M. J. Neely, *Stochastic Network Optimization with Application to Communication and Queueing Systems*. Morgan & Claypool, 2010.
- Y. Mao et al., "A Survey on Mobile Edge Computing: The Communication Perspective," *IEEE Communications Surveys & Tutorials*, 2017.
- K. Kumar and Y. H. Lu, "Cloud Computing for Mobile Users: Can Offloading Computation Save Energy?", *IEEE Computer*, 2010.

Thank You