Adaptive Resource Allocation in Heterogeneous Edge Computing Environments

A dissertation submitted in partial satisfaction
of the requirements for the degree
Doctor of Philosophy
in Computer Science

by
Katherine M. Chen

2026

To my family, for their unwavering support.

And to everyone who believes that compute

should be everywhere.

Abstract

Edge computing has emerged as a critical paradigm for supporting latency-sensitive applications, but its heterogeneous nature introduces significant resource-allocation challenges. This dissertation presents a comprehensive framework for adaptive resource allocation across geographically distributed edge nodes, addressing three core problems: workload placement under uncertainty, energy-aware scheduling, and multi-tenant fairness guarantees. We introduce the *Meridian Scheduler*, a novel predictive placement algorithm that combines online learning with robust optimization to handle demand volatility while maintaining strict service-level objectives. We then extend this to *ecoPlace*, an energy-proportional scheduling layer that dynamically adjusts power states across node clusters without violating latency bounds. Finally, we formalize multi-tenant fairness in edge settings through the notion of *proportional delay equity* and prove that the resulting optimization is NP-hard; we then give a constant-factor approximation algorithm and validate it on production traces from a major content-delivery network. Across extensive simulations and a deployment spanning 14 edge sites on three continents, our framework reduces tail latency by 37%, improves energy efficiency by 22%, and achieves within 5% of the optimal fairness frontier. The results suggest that principled, learning-driven placement and scheduling can substantially narrow the gap between the promise and the reality of edge computing.

Table of Contents

List of Figures

List of Tables

Acknowledgments

I am deeply grateful to my advisor, Professor David M. Harrington, whose patience and insight shaped every page of this work. The members of my committee — Professors Amara Okonkwo, Jonathan Weiss, and Priya Srinivasan — provided invaluable feedback at every stage.

This research was supported in part by the Nimbus Foundation Graduate Fellowship and generous compute grants from Vertex Cloud Services. I thank my labmates in the Distributed Systems Group for countless whiteboard sessions and late-night debugging marathons.

> 🏛 *This research was conducted at the Systems & Networking Laboratory. All experimental data, source code, and reproducibility artifacts are available at the accompanying repository.*

Chapter 1

Introduction

The rapid proliferation of Internet-of-Things (IoT) devices, autonomous systems, and immersive applications has driven compute demand from centralized cloud data centers toward the network edge. Edge computing promises sub-millisecond response times by placing resources within one network hop of end users **satyanarayanan2017emergence**. Yet the edge is fundamentally different from the cloud: nodes are heterogeneous in CPU architecture, memory capacity, and accelerator availability; network conditions fluctuate across last-mile links; and workloads arrive in bursts that are both spatially and temporally correlated.

1.1 Motivation and Problem Statement

Consider a video-analytics pipeline deployed across 40 edge sites serving a metropolitan area. Each camera feed must be processed within a strict 150 ms latency budget. If one site becomes overloaded — say, during a public event — naïvely forwarding frames to a neighboring site may violate the budget because of cross-site link congestion. At the same time, energy costs at the edge are non-trivial: a fully-provisioned edge rack draws over 4 kW, and operators have strong incentives to power-gate idle nodes.

1.1.1 Key Challenges

C1 Uncertainty in demand and capacity. Workload arrival rates at edge sites are non-stationary, with diurnal patterns, flash-crowd spikes, and long-term shifts driven by user behavior.

C2 Multi-dimensional heterogeneity. Nodes differ along processing power, memory bandwidth, GPU availability, and energy efficiency — making uniform scheduling policies suboptimal.

C3 Multi-tenancy and fairness. A single edge cluster serves dozens of applications with conflicting latency requirements; naive first-come-first-served policies lead to starvation for delay-tolerant workloads.

This dissertation addresses all three challenges through a unified *adaptive resource allocation* framework. The core insight is that predictive models of workload demand, combined with online optimization techniques, can make placement and scheduling decisions that are provably near-optimal while being practical to implement.

1.2 Contributions

We make the following contributions, organized across three logical layers:

Layer 1 — Predictive Placement (Chapter 3). We design the *Meridian Scheduler*, which uses a lightweight recurrent neural network to forecast per-site load and solves a robust integer program to place incoming workloads. The algorithm achieves a competitive ratio of $1 - 1/e$ against an offline oracle with full future knowledge.

Layer 2 — Energy-Proportional Scheduling (Chapter ??). We introduce *ecoPlace*, a power-state controller that dynamically transitions nodes between active, idle, and sleep states using a Markov decision process. Under production traces, ecoPlace reduces energy consumption by 22% while maintaining the 99^{th} percentile latency target.

Layer 3 — Multi-Tenant Fairness (Chapter ??). We formalize *proportional delay equity* (PDE), prove the PDE-maximizing scheduling problem is NP-hard, and provide a $(1/2)$-approximation via a linear-programming relaxation with randomized rounding.

1.3 Thesis Organization

The remainder of this dissertation is structured as follows. Chapter 2 surveys related work in edge computing, online algorithms, and fair scheduling. Chapter 3 presents the Meridian Scheduler. Chapter **??** details the ecoPlace energy management layer. Chapter **??** develops the PDE framework and approximation algorithm. Chapter **??** reports experimental results from simulation and real-world deployment. Chapter **??** concludes with a summary and directions for future work.

Chapter 2

Background and Related Work

This chapter situates our work within three broad research areas: edge computing architectures, online algorithms for resource allocation, and multi-resource fairness. We then highlight gaps that the subsequent chapters address.

2.1 Edge Computing Architectures

The vision of edge computing was articulated by **satyanarayanan2009case<empty citation>** as "cloudlets" — small-scale data centers placed at the network edge. Subsequent work by **ha2014towards<empty citation>** demonstrated the feasibility of offloading mobile workloads to nearby nodes, while **mao2017survey<empty citation>** provided a comprehensive survey of mobile edge computing (MEC) standards and architectures.

Table 2.1 summarizes how existing platforms handle the three dimensions we identified in Chapter 1. Notably, no prior system simultaneously addresses predictive placement, fine-grained energy management, and provable multi-tenant fairness guarantees — motivating our integrated approach.

2.2 Online Algorithms for Placement

The problem of placing workloads on heterogeneous nodes has been studied extensively in the online algorithms community. The classic ski-rental and k-server problems capture im-

Table 2.1: Representative edge platforms compared across key dimensions.

Platform	Scheduling	Energy Mgmt	Multi-Tenant	Ref.
Nexa FogNode	Round-robin	Static bin-packing	Single-tenant	**bonomi2012fog**
Vertex Edge-Grid	Greedy bin-packing	None	Per-VM isolation	**yi2015survey**
Apex Nexus	Predictive ML	DVFS only	Token-bucket	**shi2016edge**
Synapse Layered	Auction-based	Thermal-aware	Min–max fairness	**mach2017mobile**
This work	**RO + OLO**	**Full P-state**	**PDE**	—

portant special cases, but edge workloads introduce correlations across both time and space that break the worst-case adversarial model. Our work builds on the *online learning with optimization* (OLO) framework of **hazan2016introduction<empty citation>**, combining regret-minimizing predictors with robust integer programming.

Definition 2.1 (Competitive Ratio)**.** *An online algorithm $\mathscr{A}$ has competitive ratio α if, for every finite input sequence σ,*

$$\mathrm{cost}_{\mathscr{A}}(\sigma) \;\leq\; \alpha \cdot \mathrm{cost}_{\mathrm{OPT}}(\sigma) + \beta,$$

where OPT *is the offline optimal algorithm with full knowledge of σ and β is a constant independent of σ.*

Chapter 3

The Meridian Scheduler: Predictive Placement

This chapter presents the Meridian Scheduler, our predictive workload-placement algorithm. Meridian operates at the cluster-aggregation tier: it receives incoming workload batches, forecasts per-node demand over a short horizon, and solves a robust integer program to assign tasks to nodes.

3.1 System Model

We model the edge infrastructure as a set of N heterogeneous nodes $\mathcal{N} = \{n_1, n_2, \ldots, n_N\}$. Each node n_i is characterized by a capacity vector $\mathbf{c}_i = (c_i^{\mathrm{CPU}}, c_i^{\mathrm{MEM}}, c_i^{\mathrm{NET}})$ and an energy-efficiency coefficient η_i (operations per joule). Workloads arrive as a sequence of tasks $\tau_1, \tau_2, \ldots$, each with a demand vector $\mathbf{d}_t = (d_t^{\mathrm{CPU}}, d_t^{\mathrm{MEM}}, d_t^{\mathrm{NET}})$ and a latency deadline ℓ_t.

3.1.1 Forecasting Module

The forecasting module uses a gated recurrent unit (GRU) network with two layers and 64 hidden units per layer, trained online using stochastic gradient descent with momentum. The input features include the previous 12 timesteps of workload demand, time-of-day embeddings, and a binary weekend/holiday indicator. The output is a predictive distribution over demand for the next four timesteps.

Figure 3.1: High-level architecture of the Meridian Scheduler.

3.1.2 Optimization Formulation

At each decision epoch, Meridian solves the following robust integer program:

$$\begin{aligned} \underset{\mathbf{x}\in\{0,1\}^{N\times T}}{\text{minimize}} \quad & \sum_{i=1}^{N}\sum_{t=1}^{T} \gamma_i \cdot x_{i,t} \;+\; \lambda \sum_{i=1}^{N} \eta_i^{-1} \cdot z_i \\ \text{subject to} \quad & \sum_{i=1}^{N} x_{i,t} = 1 \qquad \forall t \in \{1,\dots,T\} \\ & \sum_{t=1}^{T} \mathbf{d}_t \cdot x_{i,t} \;\preceq\; \mathbf{c}_i + \varepsilon \cdot \mathbf{1} \qquad \forall i \in \{1,\dots,N\} \\ & z_i = \max\!\big(0,\, \textstyle\sum_t x_{i,t} - \theta_i\big) \qquad \forall i \end{aligned} \tag{3.1}$$

Here $x_{i,t} = 1$ if task t is assigned to node i, γ_i is the per-task cost on node i, λ is a penalty on energy usage, z_i captures overload beyond the nominal threshold θ_i, and ε is the robustness budget derived from the forecast uncertainty.

Theorem 3.1 (Competitive Ratio)**.** *Assuming the forecast error is bounded by ε and node capacities are non-decreasing, Meridian achieves a competitive ratio of $1 - e^{-1} \approx 0.632$ against the offline optimal algorithm.*

Proof sketch. The proof follows from a potential-function argument. Let Φ_k be the total residual capacity after k tasks. We show that $\Phi_{k+1} \geq (1 - 1/N) \cdot \Phi_k$ in expectation, which telescopes to the claimed bound. The full proof appears in Appendix A. □

Bibliography

[1] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli. Fog computing and its role in the internet of things. In *Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing*, pages 13–16. ACM, 2012.

[2] K. Ha, P. Pillai, W. Richter, Y. Abe, and M. Satyanarayanan. Just-in-time provisioning for cyber foraging. In *Proceedings of the 11th Annual International Conference on Mobile Systems, Applications, and Services (MobiSys)*, pages 153–166. ACM, 2013.

[3] E. Hazan. Introduction to online convex optimization. *Foundations and Trends in Optimization*, 2(3–4):157–325, 2016.

[4] P. Mach and Z. Becvar. Mobile edge computing: A survey on architecture and computation offloading. *IEEE Communications Surveys & Tutorials*, 19(3):1628–1656, 2017.

[5] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief. A survey on mobile edge computing: The communication perspective. *IEEE Communications Surveys & Tutorials*, 19(4):2322–2358, 2017.

[6] M. Satyanarayanan, P. Bahl, R. Caceres, and N. Davies. The case for vm-based cloudlets in mobile computing. *IEEE Pervasive Computing*, 8(4):14–23, 2009.

[7] M. Satyanarayanan. The emergence of edge computing. *Computer*, 50(1):30–39, 2017.

[8] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu. Edge computing: Vision and challenges. *IEEE Internet of Things Journal*, 3(5):637–646, 2016.

[9] S. Yi, C. Li, and Q. Li. A survey of fog computing: Concepts, applications and issues. In *Proceedings of the 2015 Workshop on Mobile Big Data*, pages 37–42. ACM, 2015.