Reliable Unreliability in Edge Deployments

Animesh Dangwal, Ria Singh, Shruthi Unnithan, Karen Yuan, Emily Zheng, Chandra Krintz, Rich Wolski
Department of Computer Science, *University of California, Santa Barbara*
{animesh, ckrintz, rich}@cs.ucsb.edu

***Abstract*—Low-cost edge and IoT clusters increasingly rely on commodity hardware and shared power infrastructure, yet the impact of power delivery degradation on distributed system performance remains poorly understood. In this work, we present a performance characterization of a large-scale Raspberry Pi 3B+ cluster, focusing on 451 devices organized into 12 shared-power domains (subclusters), with uneven device counts per subcluster. We observe systematic power-induced throttling behavior manifesting as three repeatable device classes, unthrottled, volatile, and throttled, introducing substantial heterogeneity in execution performance. This heterogeneity introduces unreliable execution delays affecting scheduling and capacity planning for distributed IoT workloads.**

We develop a profiling and modeling methodology to predict power-domain performance without exhaustively profiling all devices or concurrency configurations. By aligning throttling regimes across devices, we show throttling dynamics follow consistent piecewise-linear patterns from which we construct multivariate piecewise-linear models to capture these regimes. We further demonstrate that profiling only three representative power domains is sufficient to accurately predict performance for the remaining domains, reducing profiling complexity from exponential in concurrency configurations to constant-factor sampling. Our results show power infrastructure degradation should be treated as a first-class uncertainty source in distributed IoT systems and scalable performance prediction is feasible even in large, low-cost, and aging edge clusters.

I. Introduction

Low-cost, resource-constrained hardware platforms are increasingly used to construct large-scale edge and IoT computing clusters for research, education, and emerging deployments [1], [2], [3], [4], [5]. These systems prioritize affordability (often through long-term amortization) and compute density. IoT cluster deployments often utilize cheap, linux-capable, single board computing devices like Raspberry Pis, Jetson Nanos, or BeagleBones, for programmability and rely on consumer-grade componentry for power delivery. This is in contrast to datacenters where power delivery is carefully engineered and monitored. Moreover, these platforms dynamically scale voltage and frequency in response to power instability [6]. If the power delivery is variable or below operating specification, these systems exhibit significant performance reductions.

This power-induced performance variability is particularly problematic for edge clusters. Edge clusters increasingly support time-sensitive distributed IoT applications, including sensor data aggregation, distributed analytics, and cyber-physical control loops, where predictable execution timing is required for synchronization and correctness. However, many IoT deployments operate in environments with aging, shared, or opportunistic power infrastructure and lack fine-grained power monitoring or management. As a result, operators and researchers lack tools to reason about how power-delivery affects observable application performance, complicating scheduling, capacity planning, and experimental reproducibility. Unsurprisingly, experimental evaluations in the literature frequently disable dynamic voltage and frequency scaling (DVFS) and other power management mechanisms to reduce performance unreliability and improve experimental repeatability [7], [8], [1], [9], [10], thereby masking power-induced performance effects intrinsic to many practicable deployments. A scalable methodology to characterize and predict power-induced performance variability is therefore essential for managing large-scale IoT and edge platforms.

In this paper, we present a performance characterization of a large-scale Raspberry Pi 3B+ cluster (originally constructed in 2019 for remote deployment) with 451 devices, organized into 12 subclusters, with uneven device counts per subcluster, where each subcluster shares a commodity 60A/300W USB power supply. The Linux installed on the devices is configured for minimal power draw (i.e. all unused subsystems are powered down by the kernel) and each subcluster draws power within operational range of the individual Raspberry pi 3B+ devices (1.4A - 2.5A and 4.63V - 5V [11], [12]).

Each subcluster represents a *power domain*, analogous to a rack-level power domain in datacenters. As part of its operation, we observe systematic power-induced performance degradation as "CPU throttling" rather than a power-off shutdown. Specifically, when a device detects supplied voltage, through the USB interface, falls below a threshold, the device reduces the system clock speed to a "safe" minimum speed to avoid electrical damage. As computational speed of a device (i.e. cycles per second) is proportional to the clock speed, a throttled device is slower than an unthrottled one.

When executing a compute-intensive workload, devices in this cluster exhibit three repeatable responses to the power conditions: some devices never throttle, some devices intermittently throttle depending on neighboring device activity, and some devices always throttle. We have studied this edge cluster since 2023 and these device behaviors are either consistent or slowly changing (see Section V). We refer to the characterization of power-response as **Pi personalities** and term the personalities as *unthrottled*, *volatile*, and *throttled*, respectively. This heterogeneity of performance introduces unreliable execution across devices, affecting synchronization, aggregation, and scheduling in distributed IoT workloads.

To address these challenges, we make two principle con-

tributions. First, we develop a profiling methodology, used exhaustively throughout the cluster, to generate measurements and assign Pi personalities to each device. Secondly, we develop a modeling methodology (that relies on a reduced profiling requirement) to determine the effects of Pi personalities on CPU-bound workloads more scalably.

We present (in Section III) the classification profiling measurements for Pi personalities, and provide evidence supporting the investigation of a particular modeling approach. In Section IV we detail how different models suggested by this approach "fit" the Pi personality data (using R^2 and Bayesian Information Criteria [13]), and compare their predictive power using a reduced, scalably collected, profile.

Our results indicate only three representative power domains profiles (subclusters with the minimum, median, and maximum active and functional device counts) are sufficient to accurately predict performance for the remaining subclusters. Holdout validation shows the modeling approach achieves low prediction error while reducing profiling complexity from exponential in concurrency configurations to constant-factor sampling.

More generally, by characterizing power-induced unreliability and providing a scalable predictive methodology, this work contributes empirical insights and practical tools for managing large-scale commodity edge clusters. In this vein, this paper makes the following more specific contributions:

- We present a large-scale empirical characterization of power-induced performance degradation for an aging Raspberry Pi cluster;
- We introduce a behavioral taxonomy of edge devices under shared power constraints, identifying three repeatable throttling classes (unthrottled, volatile, throttled);
- We explore a family of linear predictive models and develop a multivariate piecewise-linear model that predicts throttling operational regimes;
- We propose and validate a sparse profiling methodology (with this model) to predict subcluster performance by profiling only three representative power domains, achieving near-optimal estimations with minimal measurement overhead; and
- Representative profiling data and traces available at *https://github.com/MAYHEM-Lab/rpi_throttle_dataset*

Overall, our results indicate power infrastructure degradation should be treated as a first-class uncertainty source in distributed IoT systems, and scalable performance modeling is feasible even in large, low-cost edge clusters.

II. Related Work

Low-cost single-board computers such as Raspberry Pi are widely used to construct edge and IoT computing testbeds for research and education [2], [14], [15], [5]. Prior work has explored the use of Raspberry Pi clusters for distributed computing, federated learning, sensor data processing, and cyber-physical systems experimentation [16], [17], [18], [19]. These platforms offer a cost-effective alternative to traditional datacenter infrastructure but typically rely on commodity power and networking hardware. To obtain stable and repeatable measurements, many studies intentionally pin CPU frequency or disable DVFS [7], [8], [1], [9], [10]. While these studies demonstrate the feasibility of large-scale edge clusters, they generally assume stable power delivery and do not examine how to handle power-induced performance effects – that are intrinsic to and critical for using real-world deployments.

Infrastructure degradation studies are common in hardware reliability and storage systems research [20], [21], [22], but are rare in edge and IoT computing contexts. Most existing Pi cluster studies evaluate freshly deployed hardware over short time horizons. Our study provides a characterization of a large-scale, long-serving cluster infrastructure that reveals systematic power-induced performance degradation and emergent behavioral classes. This perspective is critical for understanding real-world IoT and edge deployments, which often operate for extended periods with minimal maintenance.

Power-aware computing has been extensively studied in datacenter and mobile systems, where voltage and frequency scaling (DVFS) is used to manage energy consumption and thermal constraints [6], [23], [24]. In edge and IoT systems, several studies have characterized power consumption and energy efficiency of single-board computers under varying workloads [1], [9]. However, prior work largely treats power as a controllable resource or a static constraint, rather than a degradable shared infrastructure that introduces heterogeneous and non-deterministic performance behavior across devices.

Predicting system performance has been studied in the context of datacenter scheduling, cloud resource management, and edge orchestration [25], [26], [27]. Regression models, piecewise-linear models, and machine learning-based predictors have been used to estimate execution time and resource contention effects [28], [25], [26], [7]. These approaches assume homogeneous and stable infrastructure or require exhaustive profiling of system configurations [29]. In contrast, our work develops a multivariate piecewise-linear model that captures concurrency-dependent throttling regimes in degraded shared-power domains and demonstrates accurate prediction with sparse profiling of only a small subset of power domains.

To the best of our knowledge, this work is the first to empirically characterize power-induced performance degradation in a large-scale edge cluster and to provide a scalable predictive modeling methodology for degraded shared-power domains. We next present our taxonomy and characterization methodology.

III. Pi Personalities

We begin by characterizing power-induced performance variability exhibited by the individual Raspberry Pi machines comprising a large-scale (and aging) cluster. We describe the cluster and power-domain layout, present our benchmarking methodology, and analyze how concurrency influences execution time and voltage throttling. From these measurements, we derive a taxonomy of device throttling behaviors and a methodology for categorizing the behaviors according to the taxonomy. This methodology provides the empirical foundation for the predictive models introduced in the final subsection.

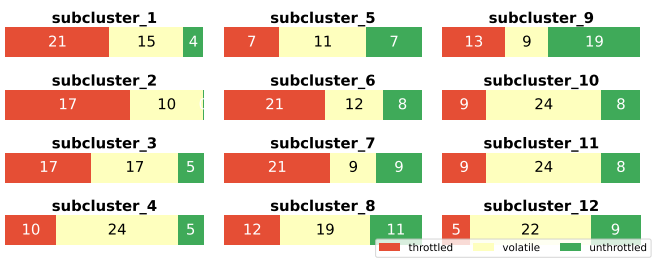

Fig. 1: Pi Personalities per subcluster. Figure shows the device counts for each personality type in the subclusters under study.

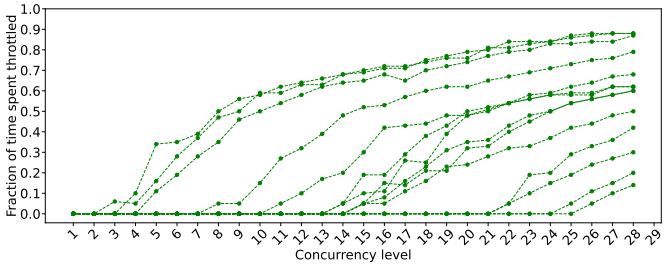

Fig. 2: Fraction of time each node spent in throttled state (throttled fraction), as y-axis, and number of devices concurrently executing within a subcluster (concurrency level), as x-axis, for volatile devices in a representative subcluster (subcluster-8).

We consider 451 Raspberry Pi 3B+ devices, a large scale cluster constructed in 2019 for edge deployment. Subclusters of devices share a commodity 60A/300W USB power supply; we consider each of 12 subclusters as a separate power domain. Each subcluster consists between 25 and 42 active working devices (failed devices are powered off). Each Raspberry Pi has 4 cores at 1.4GHz, 1GB RAM, with operational range of current, 1.4A - 2.5A, and voltage, 4.63V - 5V [11], [12].

The Raspberry Pi 3B+ dynamically imposes internal clock limits designed to slow down the CPU in an attempt to save the device from operating under potentially harmful conditions such as high temperatures or unstable power supplies. We focus on the latter in this paper as the cluster is housed in a temperature-controlled setting, and observe no temperature triggered throttling events. To protect against unstable power delivery, the device automatically adjusts CPU frequency from 1.4GHz to 0.6GHz when the internally-measured CPU voltage is < 4.63V, with a $\pm 5\%$ error in measurement [12]. We monitor dynamic voltage and frequency scaling (DVFS) every second (on average) using `vcgencmd`, which reports voltage and frequency changes with corresponding throttling causes.

To profile each subcluster, we construct two compute-bound stress tests using a single, compute-bound microbenchmark. Using a compute-bound microbenchmark isolates the effects of CPU load on throttling behavior. We use 600x600 matrix multiplication (MM) written in C for the microbenchmark. MM is a compute-bound workload with predictable instruction and memory access patterns, enabling controlled characterization of voltage-induced clock frequency scaling. As such, it provides a measurement of application-delivered device performance usable by schedulers to allocate tasks under power-domain constraints caused by throttling. While real workloads may exhibit additional contention in memory (e.g. due to paging) and network subsystems, we have not observed voltage scaling triggered by such subsystems and leave exploring memory- and network-intensive workloads to future work.

The *individual* stress test runs four MM microbenchmark jobs (one per core) 30 times on each device, one-at-a-time. During this test, the other functioning devices in the subcluster are powered on but idle. The second, *maximal*, stress test runs four MM jobs 30 times on all devices (running concurrently) in the same subcluster. We log DVFS throttling events as described above for both stress tests. To capture the impact of throttling on execution performance (for scheduler use), we record the fraction of the total time each node spent in a throttled state (termed the *throttled fraction*), during the stress test. We compute the throttled fraction for each stress test as the number of seconds the throttle flag (logged with `vcgencmd`) is set during execution of four MM jobs on an individual device divided by the total time taken to finish the four MM jobs.

A. Device Behavior Under Shared Power

We use a simple taxonomy to categorize device throttling behavior under shared-power. We refer to the classes in this taxonomy as the **Pi personalities**.

We observe devices consistently fall into three categories: *throttled*, indicating power or hardware degradation sufficient to trigger frequency scaling with or without contention, *unthrottled*, indicating devices never exhibit throttling under maximal subcluster load, and *volatile* indicating devices that exhibit load-dependent throttling behavior (i.e. neither *throttled* or *unthrottled*). Figure 1 depicts this taxonomy for the devices under study. For each subcluster, we specify the functioning device count and use red for throttled devices, yellow for volatile devices and green for unthrottled devices.

B. Throttling Onset Under Shared-Power Concurrency

This categorization motivates us to investigate the relation between subcluster load and the throttled fraction for each *volatile* device in a subcluster. To generate subcluster load, we choose, at random, a set of devices to induce "background" load while a *volatile* device is under study.

We refer to the number of devices chosen for the background load (excluding the volatile device under test) as the *concurrency level* and measure throttled fraction for each *volatile* device as a function of increasing concurrency levels. As with the previous tests, we generate a sample of 30 throttled fractions for each combination of throttled device and concurrency level and summarize the relationship using the median of this sample (to filter any outliers that might occur due to operating system noise during a specific benchmark run).

The result of a background load test is a set of x-y pairs matching concurrency level (as an x value) with median

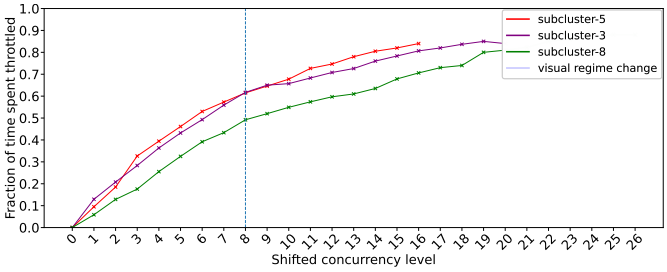

Fig. 3: Subcluster trajectories for subcluster-3 (purple), subcluster-5 (red), and subcluster-8 (green).

Full-set trained model	R^2 score			BIC score (on full subcluster set)
	min	max	mean	
OLS	0.50	0.88	0.77	-7326.59
PW	0.67	0.94	0.87	-8305.78
MPW	0.76	0.95	0.89	-8415.99

TABLE I: Range of R^2 (higher is better) scores across all 12 subclusters and BIC (lower is better) for the full subcluster set, using predicted throttled fractions from regression models fitted using every subcluster. MPW has best R^2 and BIC entries.

observed throttled fraction (as a y value) for each *volatile* device in a subcluster. For example, Figure 2 plots the x-y pairs for all *volatile* devices in subcluster-8 (an arbitrarily chosen but representative subcluster). The plotted pairs, from the same device, are linked by lines, one per *volatile* device.

The figure shows the minimum concurrency level at which throttling occurs differs by device, and the throttled fraction is monotonically increasing for all devices. These properties are consistent for background load tests of *volatile* devices across all subclusters (full set of graphs omitted for brevity).

If it is possible to conduct a background load test for every *volatile* device in every subcluster at every concurrency level and to record the resulting x-y pairs indexed by a *volatile* device identifier, a scheduler can use this information to determine the effective clock speed (and hence the effective CPU compute rate) for any combination of devices it selects. It could simply determine the Pi personality of each device it is considering, and for any assignment of devices to subclusters, use the maximal speed for *unthrottled* devices, the minimal speed for *throttled* devices, and a look-up of the throttled fraction (based on concurrency level) of the speeds for the *volatile* devices.

This methodology, however, requires a full set of stress tests and background load tests for all devices in all subclusters. To investigate improving the scalability of the approach, we explore throttled fraction trends across subclusters for any commonalities. We summarize the per-subcluster x-y pairs for the *volatile* machines into a *trajectory* for each subcluster. To form a subcluster trajectory, we first note that the plots shown in Figure 2 are approximately parallel, indicating the throttled fraction relationship is consistent and independent of the minimum concurrency level at which it begins to occur. Thus, to summarize all device trajectories in a subcluster into a single trajectory, we first transform (termed "zero-shift") each device trajectory to intersect the origin, where the sample x-y pairs in Figure 2 of the devices now roughly overlap with each other. We then compute the median throttled fraction over these zero-shifted, overlapping points, per concurrency level to construct the subcluster's trajectory.

As examples, Figure 3 shows three different subcluster trajectories. Note, they differ in terms of active and functioning devices. Subcluster-5 has 25 functioning devices, subcluster-3 has 39 functioning devices, and subcluster-8 has 42 functioning devices. From the figure, subcluster trajectories also appear to depend on the number of active devices drawing power from the shared power distribution unit for the subcluster. Further (as indicated by the vertical line) the relationship appears to be different for concurrency levels < 8 compared to ≥ 8.

C. Linear Models of Subcluster Throttled Fraction

Based on this (and a more extensive but elided) graphical analysis of subcluster trajectories, we hypothesize the overall relationship between throttled fraction, concurrency level, and active device count (for *volatile* devices) is piece-wise linear. Thus, in the remainder of this study, we explore three different linear models for describing each subcluster's throttled fractions. As a baseline, we consider ordinary least squares (OLS). We also consider a piece-wise linear model (PW) with two segments. Both models use the zero-shifted concurrency level as their only explanatory variable. Finally, we consider a multivariate linear model (MPW) with two segments that takes both concurrency level and number of active, functioning devices as explanatory variables.

Note these models are fitted for a subcluster using all the profiled *volatile* device lines for that subcluster. The subcluster trajectories described in the previous subsection only motivate the use of our linear predictive models. While more sophisticated, non-linear models (with higher fitting costs and potentially greater sensitivity to measurement variability) may also be appropriate. In Section IV we show the effectiveness of the linear approach and, from these results, conclude additional predictive power may not be necessary. We also explore root-cause analysis for these patterns and explain why our data-driven approach can be more widely applicable in Section V.

Table I shows the Bayesian Information Criterion (BIC) score for the entire subcluster dataset, and the min, max, mean R^2 score (across all subclusters) when using OLS, PW, and MPW models to predict the throttled fraction across all 12 subclusters. To elaborate, each model is fitted using every subcluster's profile data and BIC uses the negative log-likelihood as a goodness-of-fit measure (smaller is better), adding a penalty term (to penalize overfitting) for the number of parameters to estimate (scaled by the log of sample size).

From the table, the R^2 scores for each linear model increase with model complexity, indicating the additional parameters are adding explanatory power. Note, the OLS model requires two estimated parameters, the PW model requires four estimated

Model	Equation derived from full cluster as training data
OLS	$y = 0.112 + 0.040 * x$
PW	$y = (x < 8)[0.011 + 0.071 * x] +$ $(x \geq 8)[0.375 + 0.022 * x]$
MPW	$y = (x < 8)[0.167 - 0.004 * \text{avail\_devs} + 0.071 * x] +$ $(x \geq 8)[0.645 + 0.007 * \text{avail\_devs} - 0.023 * x]$

TABLE II: Derived regression equations using every subcluster as training data (rounded to three decimal places). y is predicted throttled fraction, x is the concurrency level, and avail_devs is the total number of functioning devices in a subcluster.

parameters, and the MPW model requires six estimated parameters. Further, the overall BIC score for the most complex model (MPW) is lowest, indicating this additional explanatory power does not come at the expense of over fitting.

Note, the data in rows 2 and 3 of Table I is when PW, MPW use concurrency level 8 to switch between their piecewise segments (see Figure 3). Rather than estimate this point of regime change as an additional parameter, we compute the R^2 and BIC scores exhaustively for all possible concurrency levels (as change point) in all subclusters. We observe the highest average R^2 and overall lowest BIC score for concurrency-level 8 (full analysis omitted for space considerations).

MPW's lead in terms of fit confirms the observations (illustrated in Figure 3) that the number of active devices adds profitable explanatory power. That is, MPW has the highest mean R^2, BIC and the narrowest range of R^2 scores, showing the effectiveness of the additional parameter, with better and consistent fit across all subclusters. For completeness, we present the derived model equations in Table II.

Based on these models and their reasoning we now run our experiments in Section IV to determine which model is best in terms of prediction error, and investigate reducing the data (and thus profiling) required to achieve good prediction results.

IV. Empirical Evaluation

In the previous section, we evaluated three candidate models: ordinary least squares (OLS) using concurrency as the predictor, piecewise linear (PW) with two regimes, and multivariate piecewise linear (MPW) incorporating the two regimes, concurrency level, and power-domain size. We used R^2 to quantify variance explained and BIC to select among candidate models while penalizing complexity, showing that piecewise linear models provide an accurate and parsimonious representation of concurrency-dependent throttling behavior.

In this section, we evaluate the prediction accuracy of the models using root mean squared error (RMSE) and mean absolute error (MAE), when trained on the full and exhaustive profiles of all 12 subclusters. We then investigate whether profiling a small subset of subclusters is sufficient to achieve similar prediction performance.

Figure 4 compares the MAE for the exhaustive profile across all three regression methods. The x-axis represents each subcluster and the y-axis is MAE for the throttled fraction, (i.e. the fraction of time spent throttled during execution). Each

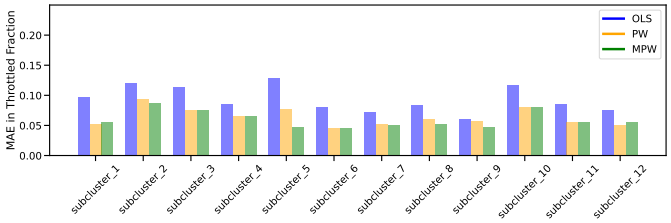

Fig. 4: MAE for each subcluster. MAE is computed as the absolute error of the difference between the predicted and actual throttled fraction, on average across experiments. Models fitted on exhaustive profiles from all 12 subclusters.

Model trained on full-set	RMSE			MAE		
	Min	Max	Range	Min	Max	Range
OLS	0.074	0.150	0.076	0.060	0.129	0.069
PW	0.057	0.121	0.064	0.045	0.094	0.049
MPW	0.059	0.109	0.050	0.046	0.087	0.041

TABLE III: Range of RMSE and MAE across subclusters. MPW has the narrowest error range (indicating stability) and lowest maximum RMSE and MAE. Models fitted on exhaustive profiles from all 12 subclusters.

bar represents the error calculated using predicted y values from each model. Blue bars are for OLS, yellow for PW, and green for MPW. These errors represent the MAE achievable by each linear model when applied to all of the data. Note, we omit graphing RMSE here because the ranking comparison is identical. Because throttled fraction errors are small, we also present the minimum and maximum RMSE and MAE in Table III to show the range of errors across the subclusters.

Over the 12 subclusters, MPW achieves the lowest aggregate error most often, and exhibits the most consistent prediction performance, while OLS performs the worst. Straight-line OLS regression assumes a single linear relationship between concurrency and throttling, and therefore cannot capture regime changes introduced by the increase in the number of volatile devices (devices which throttle dependent on background load), as explained in Section III-C. PW and MPW explicitly capture these regime shifts, and MPW further incorporates power-domain capacity draw, (i.e. the number of active devices) leading to substantially lower aggregate error in most domains.

PW slightly outperforms MPW on four of the twelve domains($\sim 0.001 - 0.002$ RMSE, MAE for throttled fraction). But across all domains, MPW yields a tighter error range ($max - min$) for both RMSE and MAE, indicating more robust error performance under domain-to-domain variability.

The tighter error bounds for MPW suggest incorporating power-domain size reduces cross-domain variance by capturing systematic differences in shared-power capacity. The domains where PW marginally outperforms MPW likely reflect cases where power-domain size provides limited additional signal (e.g. cabling losses or atypical device composition) or where the dominant effect is already captured by the regime split at the change point. In such domains, the additional covariate may introduce slight prediction variance without improving fit.

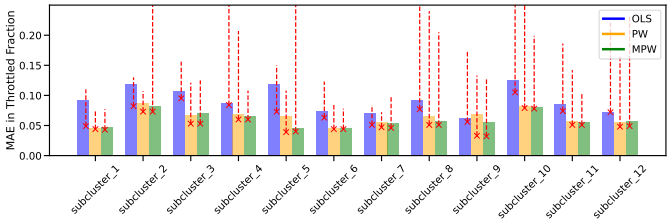

Fig. 5: MAE for each subcluster for the Min-Med-Max strategy. The red line is the minimum, maximum error across all three-domain models. The Min-Med-Max set achieves prediction error close to global minimum (red-*X*) for all models. Moreover, MPW's MAE for this set is close to optimal (i.e. the MAE shown Figure 4 when we exhaustively profile all subclusters).

Model	RMSE		MAE	
	Difference between min(Min-Med-Max) and min(Across all sets)	Range (Min-Med-Max)	Difference between min(Min-Med-Max) and min(Across all sets)	Range (Min-Med-Max)
OLS	0.018	0.077	0.014	0.062
PW	0.010	0.056	0.012	0.043
MPW	0.013	0.054	0.013	0.037

TABLE IV: RMSE and MAE statistics for the holdout experiments for the three domain cominations. Columns 2 and 4 shows the difference between the global minimum and the Min-Med-Max strategy minimum for RMSE and MAE. Columns 3 and 5 show the range ($max - min$) for RMSE and MAE, for the Min-Med-Max strategy. Min-Med-Max strategy achieves prediction error close to the best-performing three-domain training combinations across all models. And MPW's range indicates stability in performance under subcluster variability.

Nevertheless, the overall reduction in error dispersion, better R^2 and BIC scores from Table I indicates MPW generalizes more reliably across heterogeneous power domains.

A. Sparse Profiling for Scalable Performance Prediction

Although exhaustive profiling provides accurate characterization, it is impractical for large-scale edge clusters due to the cost of profiling all devices and concurrency configurations. We therefore investigate whether sparse profiling is sufficient for accurate performance prediction. We hypothesize that profiling three representative subclusters, those with the minimum, median, and maximum active device counts, is sufficient to achieve acceptable prediction error performance across the cluster. We test this hypothesis by fitting models to *all* combinations of three subclusters and comparing predictive accuracy against all subclusters individually. We then compare the model prediction error performance for the combination of the subcluster with the minimum, maximum, and median number of active devices to the globally best model prediction errors among all three-set combinations. We refer to this latter combination as the *Min-Med-Max holdout strategy*.

Our intuition for the hypothesis of combining the minimum, maximum, and median active device subclusters being sufficient as these three accurately capture the range of active power draw to throttle fraction relationships. That is, selecting domains spanning the observed active capacity range provides fitting data for the full spectrum of subcluster sizes, enabling the model to generalize across the device count varied subclusters.

To evaluate our sparse profiling hypothesis (and the Min-Med-Max holdout strategy), we show the MAE when we use sparse profiling to predict the throttled fraction for each subcluster in Figure 5. The graph layout is the same as Figure 4. To provide a visual comparison of this strategy against the performance of other three-domain combinations, we overlay each bar in the figure with a red dashed line. The red line denotes the minimum and maximum MAE observed *across all three-domain combinations*. The red *X* marks the global minimum error across combinations.

From the figure, the height of the MPW bar is equivalent to, or slightly above the global minimum (the red *X*) for each subcluster. Thus, the MPW model trained on Min-Med-Max to predict throttled fraction in *all other* subclusters achieves an MAE similar to the best performing MPW model out of *all three-domain training combinations*. As before, the graphs for RMSE are similar and omitted due to space constraints.

Note, the sporadically large heights of the dashed lines in the figure indicates model performance is sensitive to the three subclusters used for model fit. This shows model prediction accuracy is dramatically impacted by the choice of three-subcluster set.

Table IV summarizes Figure 5 and reports the error gap relative to the global minimum error. We show the difference (in columns 2 and 4) between the global minimum and the Min-Med-Max strategy minimum for RMSE and MAE, respectively. The data shows the Min-Med-Max strategy achieves prediction error very close to the globally best-performing three-domain training combination across all models. This result suggests selecting fitting domains that span the observed active capacity range is sufficient to capture the dominant sources of cross-domain variability and to enable accurate performance prediction with sparse profiling.

Columns 3 and 5 of the table show the RMSE and MAE range ($max - min$) for the Min-Med-Max strategy. MPW has a narrow range of errors for both RMSE and MAE, indicating stability in performance under subcluster variability. Note, this range is also similar to or better than RMSE and MAE range shown in Table III – when we exhaustively profile all 12 subclusters. This shows Min-Med-Max's performance, with significantly less profiling, is very close to optimal.

For completeness, we report the regression coefficients learned from this training set in Table V and summarize the R^2 and BIC scores in Table VI. The R^2 and BIC scores are similar to or better than those obtained with the full training set (in Table I), indicating no degradation in explanatory power or model parsimony when restricting the training data to these three representative domains. Moreover, MPW again has the best R^2 and BIC scores. PW and MPW, with their additional parameters, also outperform OLS using this strategy indicating better capture of throttled fraction across power-domains. Overall, these results show MPW performing best under

Model	Equation derived using the Min-Med-Max profiling strategy
OLS	$y = 0.121 + 0.041x$
PW	$y = (x < 8)[0.014 + 0.075x] +$ $(x \geq 8)[0.406 + 0.020x]$
MPW	$y = (x < 8)[0.157 - 0.004\,\text{avail\_devs} + 0.075x] +$ $(x \geq 8)[0.667 - 0.008\,\text{avail\_devs} + 0.023x]$

TABLE V: Derived regression equations for the Min-Med-Max strategy (rounded to three decimal places). y is the predicted throttled fraction, x is the concurrency level, and avail_devs is the total number of functioning devices in a subcluster.

Min-Med-Max trained model	R^2 score			BIC score (on full subcluster set)
	min	max	mean	
OLS	0.53	0.87	0.77	-7298.08
PW	0.73	0.94	0.87	-8244.28
MPW	0.79	0.95	0.89	-8383.22

TABLE VI: R^2 and BIC scores calculated using predicted throttled fraction for Min-Med-Max strategy across subclusters.

both full-set and Min-Med-Max fitting strategies, indicating power-domain active capacity is a useful explanatory variable and profiling the minimum, median, and maximum capacity domains provides sufficient coverage for accurate and scalable cluster-wide performance prediction.

Finally, this study also illustrates a general methodology applicable to similar edge clusters with shared power domains. To use it, a cluster administrator runs individual and maximal stress tests (Section III) to classify devices within each cluster (Figure 1). Second, they profile the volatile devices in subclusters with the minimum, median, and maximum number of active devices using the background load test (Section III-B). Finally, they construct an MPW model from this profiled data (i.e. the zero-shifted throttled fraction data and regime change concurrency level either visually or empirically derived). Note, a PW model is equivalent when all subclusters have the same number of active devices. The resulting model predicts the throttled fraction for any subcluster given a concurrency level and active device count. Moreover, combining this prediction with the MM job's gigaflop rate for a subcluster yields a lower bound on subcluster compute performance which can be used to inform scheduling decisions.

V. Discussion

Our characterization and modeling methodologies target a class of edge and IoT deployments with specific, but widely encountered, operational characteristics. In particular, we consider deployments composed of off-the-shelf hardware components, including devices, cabling, USB hubs, and power distribution units, that are used over extended periods without modification to their lifecycle. These systems are typically purpose-built for long-lived operation in technology-hostile environments, where routine maintenance and frequent component replacement are impractical due to cost, access constraints, or deployment scale.

While these assumptions may appear restrictive, they are in fact common across many real-world IoT and edge deployments, particularly in outdoor and industrial settings. Such clusters are often designed to prioritize affordability and sustainability, favoring commodity hardware potentially purchased second-hand or amortized over long operational lifetimes [30], [31]. As a result, aging and heterogeneous devices are the norm rather than the exception. The cluster studied in this work reflects this increasingly prevalent deployment model.

In some environments, devices are replaced when their operational "personality" degrades beyond acceptable bounds. However, the applications we target, such as wildlife conservation, forest fire monitoring, and digital agriculture, frequently operate in remote locations where subclusters are sealed to mitigate environmental exposure [17], [18]. In these settings, replacing individual devices becomes increasingly costly as cluster size grows, and system components must often operate in a degraded state until replacement is economically feasible. Our methodology explicitly accommodates such gradual degradation rather than assuming frequent hardware refresh.

Although device behavior could, in principle, evolve beyond the three personality classes identified in this study, our empirical evidence suggests such changes are rare in practice. We have collected over a year of intensive stress-test data and have not observed additional degradation modes beyond those captured by our taxonomy. Should devices simply shuffle between existing behaviors, our profiling methodology can be re-executed to update the system composition. We continue to monitor the cluster to track long-term behavioral trends.

Finally, we explored potential root causes of power-induced throttling, including device aging, power supply variability, and wiring effects. Our experiments with swapping cables, ports, power supplies, and devices yielded inconclusive results. Similar power-induced performance variability has been reported in other edge deployments (see Section II) but operators mitigate the problem by overclocking devices or supplying elevated voltages, accelerating hardware degradation. Importantly, our approach does not depend on isolating a root cause. Instead it captures the observable effects of power-induced variability and incorporates them into predictive models. This design choice allows our approach to scale and remain effective even when underlying causes are difficult to diagnose precisely.

Taken together, our results shows explicitly modeling power-induced variability, rather than attempting to eliminate it, provides a practical and robust foundation for performance prediction in long-lived, resource-constrained edge deployments.

VI. Conclusions and Future Work

In this work, we present a large-scale experimental characterization of aging edge devices and quantify power-induced performance degradation in a shared-power Raspberry Pi cluster. We identify three repeatable device behavior classes, *throttled*, *unthrottled*, and *volatile*, that capture persistent and concurrency-dependent throttling behavior. We develop and validate predictive performance models, showing multivariate piecewise linear models provide the best fit across shared power domains. We further propose and validate a sparse profiling methodology, where profiling only the minimum, median, and

maximum capacity power domains yields predictive accuracy comparable to profiling all subclusters.

As future work, we will extend our characterization to workloads that stress communication, memory, and I/O subsystems to capture additional power-draw dimensions. This will enable modeling of a broader class of edge workloads and inform the design of power-aware schedulers that treat power infrastructure as a first-class resource rather than a binary constraint.

VII. Acknowledgment

We thank the reviewers for their thoughtful feedback. We also gratefully acknowledge the support and guidance from Eric Sedlar (Oracle Labs) and Chris Bensen (Lawrence Livermore National Lab), and for excellent technical support from Vic Breen and Justin Maness (University of California, Santa Barbara). This work is supported in part by an Oracle gift, NSF awards CNS-2444318 and CNS-2107101, and DoE award DE-SC0025541.

References

[1] Z. Krpić, I. Lukić, M. Habijan, and L. Loina, "Re-evaluating compute performance in sbc clusters: Hpl benchmarking across generations," *Future Generation Computer Systems*, vol. 176, 2026.

[2] F. P. Tso, D. R. White, S. Jouet, J. Singer, and D. P. Pezaros, "The glasgow raspberry pi cloud: A scale model for cloud computing infrastructures," in *2013 IEEE 33rd International Conference on Distributed Computing Systems Workshops*. IEEE, 2013, pp. 108–112.

[3] S. Elsayed, C. Krintz, and R. Wolski, "Just the FACTS: Flexible and Energy Efficient Federated Access Control for the Edge," in *IEEE International Conference on Fog and Mobile Edge Computing*, 2024.

[4] N. Golubovic, R. Wolski, C. Krintz, and M. Mock, "Improving the Accuracy of Outdoor Temperature Prediction by IoT Devices," in *IEEE Conference on IoT*, 2019.

[5] N. M. Mwasaga, "Design and evaluation of micro high-performance computing as an educational tool to improve high-performance computing usability," 2022.

[6] M. Weiser, B. Welch, A. Demers, and S. Shenker, "Scheduling for reduced CPU energy," in *First Symposium on Operating Systems Design and Implementation (OSDI 94)*. Monterey, CA: USENIX Association, Nov. 1994.

[7] P. M. S. Sánchez, J. M. J. Valero, A. H. Celdrán, G. Bovet, M. G. Pérez, and G. M. Pérez, "Lwhbench: A low-level hardware component benchmark and dataset for single board computers," *Internet of Things*, vol. 22, Jul. 2023. [Online]. Available: http://dx.doi.org/10.1016/j.iot.2023.100764

[8] J. Haj-Yahya, M. Alser, J. Kim, A. G. Yağlıkçı, N. Vijaykumar, E. Rotem, and O. Mutlu, "Sysscale: exploiting multi-domain dynamic voltage and frequency scaling for energy efficient mobile processors," in *International Symposium on Computer Architecture*, 2020. [Online]. Available: https://doi.org/10.1109/ISCA45697.2020.00029

[9] P. J. Basford, S. J. Johnston, C. S. Perkins, T. Garnock-Jones, F. P. Tso, D. Pezaros, R. D. Mullins, E. Yoneki, J. Singer, and S. J. Cox, "Performance analysis of single board computer clusters," *Future Generation Computer Systems*, vol. 102, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167739X1833142X

[10] T. Mytkowicz, A. Diwan, M. Hauswirth, and P. Sweeney, "Producing wrong data without doing anything obviously wrong!" in *International Conference on Architectural Support for Programming Languages and Operating Systems*, 2009. [Online]. Available: https://doi.org/10.1145/1508244.1508275

[11] R. P. Foundation. (2026) Raspberry pi computer hardware - power supply. Accessed: 2026-01-01. [Online]. Available: https://www.raspberrypi.com/documentation/computers/raspberry-pi.html#power-supply

[12] *Five Output Universal PMIC, DataSheet*, MaxLinear, 03 2018, rev 1B.

[13] G. Schwarz, "Estimating the dimension of a model," *The Annals of Statistics*, vol. 6, no. 2, pp. 461–464, 1978.

[14] I. S. B. Dimitrios Papakyriakou, "High performance linpack (hpl) benchmark on raspberry pi 4b (8gb) beowulf cluster," *International Journal of Computer Applications*, vol. 185, no. 25, pp. 11–19, Jul 2023.

[15] S. J. Cox, J. T. Cox, R. P. Boardman, S. J. Johnston, M. Scott, and N. S. O'brien, "Iridis-pi: a low-cost, compact demonstration cluster," *Cluster Computing*, vol. 17, no. 2, pp. 349–358, 2014.

[16] S. Yue and J. Ren, "Efficient federated meta-learning over multi-access wireless networks," *Next Generation Multiple Access*, pp. 547–581, 2024.

[17] L. Kurafeeva, A. Subedi, R. Hartung, M. Fay, A. Biswas, S. Jha, O. Kilic, C. Krintz, A. Merzky, D. Thain *et al.*, "xgfabric: Coupling sensor networks and hpc facilities with private 5g wireless networks for real-time digital agriculture," in *Proceedings of the SC'25 Workshops of the International Conference for High Performance Computing, Networking, Storage and Analysis*, 2025, pp. 2317–2327.

[18] D. Balouek-Thomert, I. Perez, S. D. Faulstich, H. A. Holmes, I. Altintas, and M. Parashar, "Keynote talk: Leveraging the edge-cloud continuum to manage the impact of wildfires on air quality," in *2023 IEEE International Conference on Pervasive Computing and Communications Workshops and other Affiliated Events (PerCom Workshops)*, 2023, pp. 27–31.

[19] D. Velasco-Montero, J. Fernández-Berni, R. Carmona-Galán, A. Sanglas, and F. Palomares, "Reliable and efficient integration of ai into camera traps for smart wildlife monitoring based on continual learning," *Ecological Informatics*, vol. 83, p. 102815, 2024.

[20] K. V. Vishwanath and N. Nagappan, "Characterizing cloud computing hardware reliability," in *Proceedings of the 1st ACM Symposium on Cloud Computing*, ser. SoCC '10. New York, NY, USA: Association for Computing Machinery, 2010, p. 193–204.

[21] H. S. Gunawi, R. O. Suminto, R. Sears, C. Golliher, S. Sundararaman, X. Lin, T. Emami, W. Sheng, N. Bidokhti, C. McCaffrey, D. Srinivasan, B. Panda, A. Baptist, G. Grider, P. M. Fields, K. Harms, R. B. Ross, A. Jacobson, R. Ricci, K. Webb, P. Alvaro, H. B. Runesha, M. Hao, and H. Li, "Fail-slow at scale: Evidence of hardware performance faults in large production systems," *ACM Trans. Storage*, vol. 14, no. 3, Oct. 2018. [Online]. Available: https://doi.org/10.1145/3242086

[22] V. Sridharan, N. DeBardeleben, S. Blanchard, K. B. Ferreira, J. Stearley, J. Shalf, and S. Gurumurthi, "Memory errors in modern systems: The good, the bad, and the ugly," *ACM SIGARCH Computer Architecture News*, vol. 43, no. 1, pp. 297–310, 2015.

[23] H. Hanson, S. W. Keckler, S. Ghiasi, K. Rajamani, F. Rawson, and J. Rubio, "Thermal response to dvfs: analysis with an intel pentium m," in *Proceedings of the 2007 International Symposium on Low Power Electronics and Design*, ser. ISLPED '07. New York, NY, USA: Association for Computing Machinery, 2007, p. 219–224. [Online]. Available: https://doi.org/10.1145/1283780.1283827

[24] C.-M. Wu, R.-S. Chang, and H.-Y. Chan, "A green energy-efficient scheduling algorithm using the dvfs technique for cloud datacenters," *Future Generation Computer Systems*, vol. 37, pp. 141–147, 2014.

[25] Y. Peng, S. Chen, Y. Zhao, and Z. Yu, "UFO: The ultimate QoS-Aware core management for virtualized and oversubscribed public clouds," in *21st USENIX Symposium on Networked Systems Design and Implementation (NSDI 24)*. Santa Clara, CA: USENIX Association, Apr. 2024, pp. 1511–1530.

[26] L. Chen, C. Lin, S. Luo, H. Xu, and C. Xu, "Grad: Intelligent microservice scaling by harnessing resource fungibility," in *2025 IEEE International Symposium on High Performance Computer Architecture (HPCA)*, 2025, pp. 474–486.

[27] B. Xie and H. Cui, "Deep reinforcement learning-based dynamical task offloading for mobile edge computing," *The Journal of Supercomputing*, vol. 81, no. 1, p. 35, 2025.

[28] S. Chen, A. Jin, C. Delimitrou, and J. F. Martínez, "Retail: Opting for learning simplicity to enable qos-aware power management in the cloud," in *2022 IEEE International Symposium on High-Performance Computer Architecture (HPCA)*. IEEE, 2022, pp. 155–168.

[29] V. Gohil, S. Dev, G. Upasani, D. Lo, P. Ranganathan, and C. Delimitrou, "The importance of generalizability in machine learning for systems," *IEEE Computer Architecture Letters*, vol. 23, no. 1, 2024.

[30] P. Ngoy, F. Dar, M. Liyanage, Z. Yin, U. Norbisrath, A. Zuniga, J. Pestana, M. Radeta, P. Nurmi, and H. Flores, "Supporting sustainable computing by repurposing e-waste smartphones as tiny data centers," *IEEE Pervasive Computing*, 2025.

[31] J. Switzer, G. Marcano, R. Kastner, and P. Pannuto, "Junkyard computing: Repurposing discarded smartphones to minimize carbon," in *ACM International Conference on Architectural Support for Programming Languages and Operating Systems*, 2023.