Poster Abstract:
Understanding Performance-Impacting Factors

Steven D. Harris
Christopher D. Gill
Roger D. Chamberlain

Steven D. Harris, Christopher D. Gill, and Roger D. Chamberlain, “Poster Abstract: Understanding Performance-Impacting Factors,” in *Proc. of ACM/IEEE 17th International Conference on Cyber-Physical Systems* (ICCPS), May 2026, pp. 273-274.
DOI: 10.1109/ICCPS68698.2026.00036

Dept. of Computer Science and Engineering
Washington University in St. Louis

Poster Abstract: Understanding Performance-Impacting Factors

Steven D. Harris
McKelvey School of Engineering
Washington Univ. in St. Louis
St. Louis, MO, USA
sharris22@wustl.edu

Christopher D. Gill
McKelvey School of Engineering
Washington Univ. in St. Louis
St. Louis, MO, USA
cdgill@wustl.edu

Roger D. Chamberlain
McKelvey School of Engineering
Washington Univ. in St. Louis
St. Louis, MO, USA
roger@wustl.edu

***Abstract*—Cyber-physical systems rely heavily on their developers having a concrete understanding of the performance of the cyber aspects of the system, i.e., the software components. Analytical performance models are frequently used to improve understanding, as are empirical measurements in realistic environments. We explore the use of data analysis techniques that are common in other fields (social science, business) to better understand the performance-impacting factors that drive embedded software.**

I. Introduction

In the vast majority of cyber-physical systems, the execution times of embedded software tasks must meet timing constraints of some form. In hard real-time systems, missing even one deadline can be catastrophic. Even in soft real-time systems, where occasional deadline misses can be tolerated, it is still vitally important for the performance of the software to be well understood.

Schedulability analysis primarily relies on the worst-case execution time (WCET) of a software task, which is frequently determined empirically [1]. While this may answer schedulability questions, it often gives limited information as to how the software task's performance varies under different circumstances (e.g., when varying the frequency and voltage of the processor core).

We are investigating approaches to increasing the understanding of performance-impacting factors for software, using techniques that are in common use in other fields but historically have seen limited use in computing. Specifically, what can structured equation modeling [2] and/or the Boltzmann uniqueness theorem [3] tell us about how a software application performs on modern embedded devices?

We will use these tools to address the following questions:

1) What are the quantitative impacts of factors such as page faults, last-level cache misses, and context switches on execution time?
2) How do we confirm that the parameters chosen to drive a given analytic model are the correct ones?
3) To what degree do distinct processor cores operate (i.e., perform) independently?

This work was funded by NSF grant CNS-2229290 (CPS) and a gift from BECS Technology, Inc.

In some cases, we confirm (with increased rigor) what we already believed to be true. In other cases, we learn something new (again, with rigorous statistical support).

II. Approach

To address the posed questions, we will use a combination of structured equation modeling [2] (SEM) and Boltzmann uniqueness [3]. Structured equation modeling is a set of multivariate statistical analysis techniques that includes multiple regression, factor analysis, and path analysis [4]. SEM has been widely used in a number of fields, including the social sciences, psychology, education, and business [5], [6]. SEM techniques allow for the direct estimation of complex relationships among multiple variables. The Boltzmann uniqueness theorem allows one to assess the independence of two potentially entangled systems [7].

We target a pair of execution platforms that represent common characteristics of modern embedded systems. The Orange Pi 5 and Radxa Cubie A7A are both Arm big.LITTLE single-board computers. The Orange Pi 5 has 4 Little cores and 4 Big cores, while the Cubie A7A has 6 Little cores and 2 Big cores. Each platform has private L1 and L2 caches for each core and a shared L3 last-level cache.

Given the recent surge in interest for machine learning at the edge, we chose a task that is ubiquitous in ML applications, matrix-matrix multiplication. This workload reflects a common general matrix multiply (GEMM) computation, parallelized with OpenMP for thread counts in the range $\{1, ..., 2n\}$, where n is the maximum number of cores. The dimensions of the square matrices are in the range $M = \{64, 128, ..., 8192\}$. We measure the execution time empirically, varying matrix dimension, core count, core composition (Big vs. Little, which we denote as *island*), and thread count. We use `perf` to record properties of the execution from the available hardware performance counters.

Questions 1 and 2 are addressed using SEM techniques, and we leverage Boltzmann uniqueness to address question 3.

III. Results to Date

To tackle question 1, we use traditional path analysis, a subset of SEM analysis, wherein we analyze only the relationships between measured variables. This technique will help us to

979-8-3195-4405-6/26/$31.00 ©2026 IEEE
DOI 10.1109/ICCPS68698.2026.00036

understand not only the relationships between the exogenous (independent) and endogenous (dependent) variables but also the degree to which each exogenous variable impacts the endogenous variable.

Table I shows the Orange Pi 5 regressions and variances for measured parameters that are statistically significant. All measurements are normalized (so they are comparable).

TABLE I
SEM MODEL REGRESSIONS AND VARIANCES – $R^2 = 0.913$

	Estimate	Std.Err	P(>—z—)
Regressions:			
`parallel_time` ∼			
`threads`	-0.073	0.004	0.000
`matrix_dimension`	0.126	0.007	0.000
`cpu_utilization`	-0.053	0.004	0.000
`context_switches`	0.101	0.007	0.000
`cpu_migrations`	0.410	0.009	0.000
`page_faults`	-0.096	0.004	0.000
`ipc`	0.069	0.005	0.000
`branch_misses`	0.225	0.010	0.000
`l1_dcache_loads`	-0.135	0.007	0.000
`llc_load_misses`	0.326	0.005	0.000
`island_numeric`	0.130	0.005	0.000
Variances:			
`parallel_time`	0.087	0.001	0.000

Figure 1 shows the resulting impact for each parameter, with a positive value representing greater performance (shorter execution time). Some impacts are what one would expect (e.g., last-level cache misses), whereas some are more surprising (e.g., page faults).

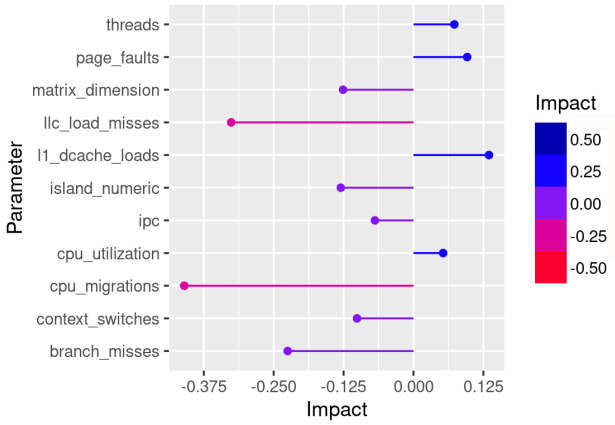


Fig. 1. Parameter Impacts on Parallel Execution Time

Question 2 is also addressed using SEM techniques. We presented an analytic model for GEMM in [8], in which the execution time is modeled as a function of 3 parameters: matrix dimension, number of Little cores, and number of Big cores.

SEM analysis yields what is called the standard estimate, which is indicative of the change in execution time (i.e., in standard deviations) for one-standard-deviation change in the predictor (a parameter listed above). The standard error on the standard estimate gives a measure of the precision of that standardized estimate with a smaller standard error meaning a more precise estimate.

The analysis shows that the matrix dimension has a strong impact. A one-standard-deviation increase is associated with a 0.705 standard deviation increase in execution time (standard error = 0.023). Examining pair-wise effects, the combined effect of matrix dimension and island is the strongest effect demonstrated, with a one-standard-deviation change in the interaction term between matrix dimension and island (Big vs. Little cores) associated with a 1.097 standard deviation change in execution time (standard error = 0.023). These estimates are substantially stronger than other candidates. The choice of parameters for the analytic model are confirmed.

Moving to question 3, platforms such as Arm big.LITTLE frequently rely on the heuristic assumption of cluster independence wherein high-performance (Big) and high-efficiency (Little) cores operate as computationally orthogonal resources. However, these platforms share resources (e.g., last-level cache) across cores. We therefore must assume that islands interact, operating in a functionally "coupled state." To conduct a formal, measure-theoretic evaluation of this assumption, we apply the Boltzmann uniqueness theorem, the only valid method to determine the veracity of this assumption [7]. Mathematically, we define an "uncoupled" system as a joint resource measurement that is strictly the convolution of its subsystem measures. We evaluate our two platforms (Cubie A7A & Orange Pi 5), and demonstrate a systemic failure of convolution commutativity. Our findings, characterized by significant negative correlations (up to -0.4248) and the failure of predictive convolution models, constitute a mathematical disproof of cluster independence. We identify shared architectural resources, specifically the last-level cache and shared power rails, as the physical mechanism of "architectural entanglement," rendering traditional additive performance models fundamentally underspecified.

REFERENCES

[1] R. Wilhelm *et al.*, "The worst-case execution-time problem—overview of methods and survey of tools," *ACM Transactions on Embedded Computing Systems*, vol. 7, no. 3, pp. 36:1–36:53, Apr. 2008.

[2] R. H. Hoyle, *Structural Equation Modeling: Concepts, Issues, and Applications*. Thousand Oaks, CA, USA: Sage, 1995.

[3] S. Kaniel and M. Shinbrot, "The Boltzmann equation: I. uniqueness and local existence," *Communications in Mathematical Physics*, vol. 58, no. 1, pp. 65–84, Feb. 1978.

[4] R. B. Kline, *Principles and Practice of Structural Equation Modeling*, 5th ed. New York, NY, USA: Guilford Press, 2023.

[5] J. B. Grace, *Structural Equation Modeling and Natural Systems*. Cambridge, UK: Cambridge University Press, 2006.

[6] R. H. Hoyle, *Structural Equation Modeling for Social and Personality Psychology*. London, UK: Sage, 2011.

[7] F. Sandomirskiy and O. Tamuz, "On the origin of the Boltzmann distribution," *Mathematische Annalen*, vol. 392, no. 4, pp. 5617–5638, Aug. 2025.

[8] S. D. Harris, R. D. Chamberlain, and C. D. Gill, "Performance models for non-uniform heterogeneous platforms and their validation using structural equation modeling," *International Journal of Networking and Computing*, vol. 16, no. 1, pp. 20–51, Jan. 2026.