MindPalace: A Framework for Studying Microarchitecture Design of Function-as-a-Service

Kaifeng Xu
Princeton University
Princeton, USA
kaifengx@princeton.edu

Georgios Tziantzioulis
Princeton University
Princeton, USA
georgios.tziantzioulis@pm.me

David Wentzlaff
Princeton University
Princeton, USA
wentzlaf@princeton.edu

***Abstract*—The increasing popularity of Function-as-a-Service (FaaS) applications has created new challenges for current microarchitecture designs due to their complex middleware and significant Operating System (OS) overhead, prohibiting the detailed study of new hardware designs.**

This work introduces MindPalace, an open-source full-system framework that developed to study the architectural behavior of large middleware serverless systems. MindPalace contains a simulation infrastructure and a pre-configured virtual machine (VM) to assist FaaS architecture research. The simulation tool leverages QEMU and ChampSim, combining a featureful full-system emulator with a rich microarchitecture simulator. The VM contains a full Ubuntu image with preinstalled OpenWhisk, together with checkpoints for ready-to-run serverless applications. Combining them enables full-system FaaS architectural study of the memory hierarchy, branch predictors, prefetchers, and other microarchitectural components.

I. Introduction

Cloud computing has become increasingly popular with new computing paradigms emerging, such as Function-as-a-Service (FaaS, a.k.a serverless computing) that further remove the responsibility of managing server infrastructure from the hands of users. Improving overall performance of such systems is crucial for both cloud providers and users.

Existing architecture is insufficient. Previous studies have shown that emerging workloads like microservices and FaaS usually target fine-grained, short-lived applications [1]–[3]. Additionally, issues such as co-located job interference [4] and low invocation rates [2] can significantly affect CPU performance. Furthermore, OS kernel overhead is significant in under-invoked functions [5], [6].

Limitations in Tools for Designing Novel Architecture. Current research tools encounter significant challenges when used for evaluating and designing novel architectures targeting FaaS workloads. Simulators like Pin [7] and ZSim [8] focus on user-level applications, while others such as QEMU [9] and Simics [10] lack the performance model for microarchitecture studies. Although tools like GEM5 [11] are capable of full-system architecture simulations, our significant efforts to run a complete FaaS framework like OpenWhisk [12] on GEM5's detailed processor simulations were not successful. Networking and timing issues within GEM5's detailed simulations prevented OpenWhisk from properly responding to FaaS requests. Also, to the best of our knowledge, no published work has demonstrated running a complete instance of an industrial strength FaaS framework like OpenWhisk in architecture simulations. Specifically GEM5 and vSwarm [13] have never been shown to run a full-stack FaaS framework like OpenWhisk. Due to these limitations, most current FaaS evaluations are conducted on existing hardware using performance counters [5], [6], [14], or they focus only on containers, as seen in works like vSwarm [13] and Ignite [15].

MindPalace is available at: github.com/PrincetonUniversity/MindPalace.

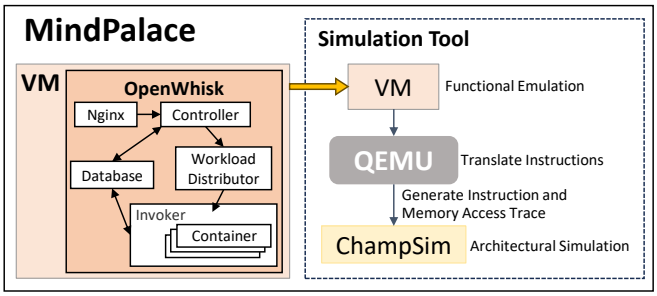

Fig. 1: MindPalace Framework Overview. MindPalace consists of a preconfigured VM and a simulation tool.

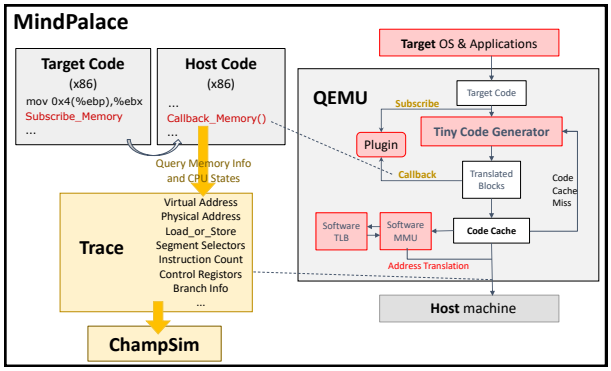

Fig. 2: MindPalace Workflow

MindPalace Bridges the Gap. MindPalace provides an open-source simulation framework, along with a preconfigured VM, to evaluate OS-heavy full-system FaaS workloads and can be used to design new chip architectures targeting FaaS platforms. The simulation framework incorporates QEMU [9] and ChampSim [16]. The accompanying VM is an Ubuntu image with OpenWhisk deployed. Fig. 1 illustrates the high-level structure of MindPalace. It allows running an unmodified OS on QEMU, with applications deployed within the VM.

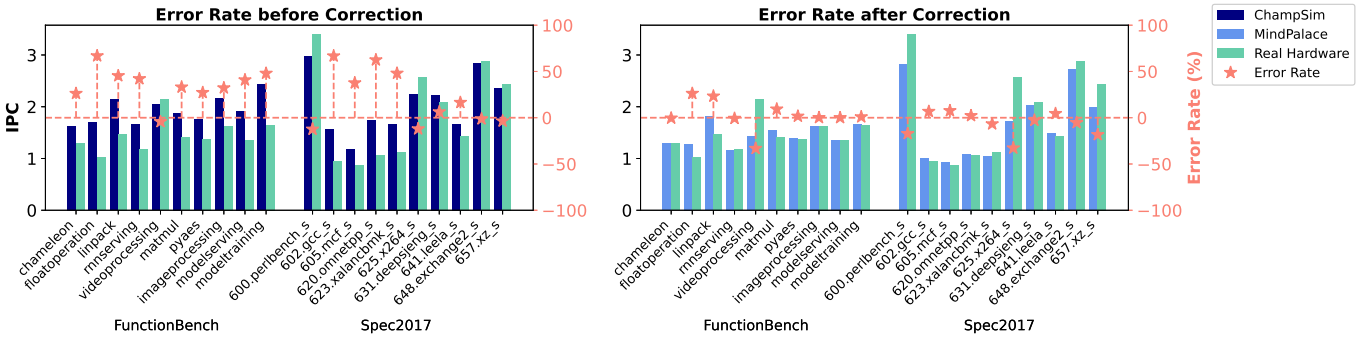

Fig. 3: IPC Errors before Calibration (left) and after Calibration (right). Each benchmark has a simulated IPC and real hardware IPC. The red stars demonstrate the error rate of each simulation.

II. Simulation Tool Architecture

Fig. 2 shows the simulation workflow of MindPalace. During execution, MindPalace will query memory access information for every instruction. At the same time, MindPalace will record virtual CPU states. Then, it combines this information to generate a compressed trace file. These trace files serve as the interface between the functional model and the performance model. The trace file is generated just once per workload and can be reused to perform multiple architecture simulations for different configurations. While ChampSim is chosen as the preferred performance simulation framework for MindPalace, any other trace-based simulator can be modified to use MindPalace-generated traces.

1) **QEMU plugin:** In contrast to the typical QEMU workflow, MindPalace adopts the plugin method to add more functionality during dynamic translation.

2) **Software TLB:** MindPalace adds another field in the TLB entry, the PA of the guest OS.

3) **TCG:** The core of QEMU is a dynamic translator, called Tiny Code Generator (TCG). MindPalace records instruction types during TCG translation.

4) **CPU States:** MindPalace determines if the execution is in user or kernel mode by capturing two important groups of x86 states: segment selectors and control registers. These values are stored in QEMU emulated virtual processor registers.

5) **Marker Instruction:** MindPalace introduces a new `Marker` instruction that can be inserted into any point in the system and record user-defined information into the trace.

6) **Trace:** The functional model, i.e., modified QEMU, generates lines of trace logs once the plugin function is called. Then, the performance model, i.e., ChampSim, takes the trace file, reformats the trace to make it compatible with ChampSim inputs and performs the architecture simulation.

III. Validation

In order to understand the accuracy of MindPalace, which is limited by original ChampSim accuracy, we present verification and error calibration for our simulator. The verification of our work is separated into two parts: functional model verification and performance model verification. Function model verification check the correctness of address translation and target addresses of control flow instructions. Performance model verification includes performance calibration factors to provide a better IPC match with real hardware. We compare simulation results with performance counter results in an Ubuntu 22.04.5 OS running on a real Intel i9-12900K processor, and the benchmarks are from serverless benchmark suite, FunctionBench [17], and standard architecture study benchmarks, SPEC2017. This is the first work that attempts to analyze the fidelity of ChampSim and correct mismatches.

By adjusting ChampSim configurations, we notice the IPC is primarily sensitive to two parameters: DRAM latency, and branch misprediction penalty. We calibrate performance results by adjusting those parameters to get lowest average IPC absolute error rates for all benchmarks. Fig. 3 displays simulation results and real hardware IPC. The average error rate is 28.1% before calibration and 10.7% after calibration.

IV. Conclusion

This work presents MindPalace, a full-system simulation framework that can directly perform full-stack FaaS architectural designs. MindPalace demonstrates its capacity to evaluate FaaS workloads and design innovative architectures. MindPalace is verified using FunctionBench and SPECspeed2017 benchmarks against real hardware performance counters. Error correction factors are utilized in MindPalace to fine-tune IPC errors within ChampSim. We believe that MindPalace offers computer architecture researchers a new tool, simplifying the process of designing novel architectures tailored for FaaS workflows.

Acknowledgments

This material is based on research sponsored by the National Science Foundation under Grant No. CCF-1822949, Air Force Research Laboratory (AFRL) and Defense Advanced Research Projects Agency (DARPA) under agreement No. FA8650-18-2-7862. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of Air Force Research Laboratory (AFRL) and Defense Advanced Research Projects Agency (DARPA) or the U.S. Government. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

REFERENCES

[1] Z. Jia and E. Witchel, "Nightcore: efficient and scalable serverless computing for latency-sensitive, interactive microservices," in *Proceedings of the 26th ACM International Conference on Architectural Support for Programming Languages and Operating Systems*, 2021, pp. 152–166.

[2] M. Shahrad, R. Fonseca, I. Goiri, G. Chaudhry, P. Batum, J. Cooke, E. Laureano, C. Tresness, M. Russinovich, and R. Bianchini, "Serverless in the wild: Characterizing and optimizing the serverless workload at a large cloud provider," in *2020 USENIX Annual Technical Conference (USENIX ATC 20)*. USENIX Association, Jul. 2020, pp. 205–218. [Online]. Available: https://www.usenix.org/conference/atc20/presentation/shahrad

[3] A. Sriraman and T. F. Wenisch, "µTune: Auto-Tuned threading for OLDI microservices," in *13th USENIX Symposium on Operating Systems Design and Implementation (OSDI 18)*. Carlsbad, CA: USENIX Association, Oct. 2018, pp. 177–194. [Online]. Available: http://www.usenix.org/conference/osdi18/presentation/sriraman

[4] S. Luo, H. Xu, C. Lu, K. Ye, G. Xu, L. Zhang, Y. Ding, J. He, and C. Xu, "Characterizing microservice dependency and performance: Alibaba trace analysis," in *Proceedings of the ACM Symposium on Cloud Computing*, 2021, pp. 412–426.

[5] Y. Gan, Y. Zhang, D. Cheng, A. Shetty, P. Rathi, N. Katarki, A. Bruno, J. Hu, B. Ritchken, B. Jackson, K. Hu, M. Pancholi, Y. He, B. Clancy, C. Colen, F. Wen, C. Leung, S. Wang, L. Zaruvinsky, M. Espinosa, R. Lin, Z. Liu, J. Padilla, and C. Delimitrou, "An Open-Source Benchmark Suite for Microservices and Their Hardware-Software Implications for Cloud and Edge Systems," in *Proceedings of the Twenty Fourth International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, April 2019.

[6] M. Shahrad, J. Balkind, and D. Wentzlaff, "Architectural implications of function-as-a-service computing," in *Proceedings of the 52nd Annual IEEE/ACM International Symposium on Microarchitecture*, ser. MICRO '52. New York, NY, USA: Association for Computing Machinery, 2019, p. 1063–1075. [Online]. Available: https://doi.org/10.1145/3352460.3358296

[7] C.-K. Luk, R. Cohn, R. Muth, H. Patil, A. Klauser, G. Lowney, S. Wallace, V. J. Reddi, and K. Hazelwood, "Pin: building customized program analysis tools with dynamic instrumentation," *Acm sigplan notices*, vol. 40, no. 6, pp. 190–200, 2005.

[8] D. Sanchez and C. Kozyrakis, "Zsim: Fast and accurate microarchitectural simulation of thousand-core systems," *ACM SIGARCH Computer architecture news*, vol. 41, no. 3, pp. 475–486, 2013.

[9] F. Bellard, "QEMU, a fast and portable dynamic translator." in *USENIX annual technical conference, FREENIX Track*, vol. 41, no. 46. California, USA, 2005, pp. 10–5555.

[10] P. S. Magnusson, M. Christensson, J. Eskilson, D. Forsgren, G. Hallberg, J. Hogberg, F. Larsson, A. Moestedt, and B. Werner, "Simics: A full system simulation platform," *Computer*, vol. 35, no. 2, pp. 50–58, 2002.

[11] N. Binkert, B. Beckmann, G. Black, S. K. Reinhardt, A. Saidi, A. Basu, J. Hestness, D. R. Hower, T. Krishna, S. Sardashti, R. Sen, K. Sewell, M. Shoaib, N. Vaish, M. D. Hill, and D. A. Wood, "The gem5 simulator," *SIGARCH Comput. Archit. News*, vol. 39, no. 2, p. 1–7, aug 2011. [Online]. Available: https://doi.org/10.1145/2024716.2024718

[12] A. S. Foundation, "Apache openwhisk," https://openwhisk.apache.org., Accessed: 2022-07-15.

[13] D. Schall, A. Margaritov, D. Ustiugov, A. Sandberg, and B. Grot, "Lukewarm serverless functions: Characterization and optimization," in *Proceedings of the 49th Annual International Symposium on Computer Architecture*, ser. ISCA '22. New York, NY, USA: Association for Computing Machinery, 2022, p. 757–770.

[14] S. Chen, S. GalOn, C. Delimitrou, S. Manne, and J. F. Martinez, "Workload characterization of interactive cloud services on big and small server platforms," in *2017 IEEE International Symposium on Workload Characterization (IISWC)*. IEEE, 2017, pp. 125–134.

[15] D. Schall, A. Sandberg, and B. Grot, "Warming up a cold front-end with ignite," in *56th IEEE/ACM International Symposium on Microarchitecture*, 2023.

[16] N. Gober, G. Chacon, L. Wang, P. V. Gratz, D. A. Jimenez, E. Teran, S. Pugsley, and J. Kim, "The championship simulator: Architectural simulation for education and competition," 2022. [Online]. Available: https://arxiv.org/abs/2210.14324

[17] J. Kim and K. Lee, "Functionbench: A suite of workloads for serverless cloud function service," in *2019 IEEE 12th International Conference on Cloud Computing (CLOUD)*. IEEE, 2019, pp. 502–504.