# Georgios Vavouliotis | Research Statement

gvavou5.github.io gvavou5@gmail.com

Modern processor designs leverage various microarchitectural predictors and prefetchers to bridge the processor-memory performance gap. In practice, fundamental computer architecture concepts like ILP, out-of-order execution, and speculative execution heavily depend on the effectiveness of microarchitectural prefetchers and predictors.

Withing the realm of computing, my research aims at identifying and exploiting the predictability of programs to design microarchitectural prefetching and prediction mechanisms for the cache and the TLB hierarchy, improving cache/TLB management for emerging applications with large data and code footprints, leveraging machine learning algorithms to design intelligent microarchitectural components, and re-thinking microarchitectural designs for server and data center applications.

In the past, my research work has been mainly focused on accelerating address translation via prefetching for the TLB hierarchy, increasing cache prefetching efficacy by leveraging address translation metadata available at the microarchitecture and runtime levels, and improving on-chip cache management for applications with massive (data and code) working sets sizes.

Section 1 overviews my peer-reviewed published research work. Section 2 describes my ongoing research projects. Finally, Section 3 elaborates on my research vision for the next years.

## 1 Published Research Works

### 1.1 Agile Data TLB Prefetching [1]

This work provides a fresh look on microarchitectural TLB prefetching for data accesses. Based on our analysis on a big set of academic and industrial workloads, we design and propose the *Sampling-Based Free TLB Prefetching (SBFP)* scheme, a dynamic mechanism that exploits page table locality [2] for improving the efficacy of TLB prefetching. We demonstrate that SBFP can be combined with any TLB prefetcher to achieve notable performance gains while reducing the memory footprint of page walks and the energy consumption of address translation.

To further exploit the benefits of SBFP, we design and propose the *Agile TLB Prefetcher (ATP)*, a composite TLB prefetcher for data accesses, implemented as a decision tree. ATP effectively combines three low-cost TLB prefetchers and relies on two mechanisms; (i) adaptive selection logic that dynamically activates the most appropriate prefetcher per TLB miss, and (ii) a throttling scheme that disables TLB prefetching when it is not helpful.

Over the best previously proposed data TLB prefetcher, ATP+SBFP yields geometric mean speedups that range between of 3.4% and 8.7%, across various contemporary benchmark suites.

This work has been also presented as a Poster at 48th edition of the International Symposium on Computer Architecture (ISCA), New York, 2022.

### 1.2 Instruction TLB Prefetching for Servers [3]

This work provides evidence that instruction address translation is an emerging performance bottleneck in server applications. To alleviate the high overheads caused by frequent instruction TLB misses, we propose *Morrigan*, the first ever and the state-of-the-art microarchitectural TLB prefetcher for instruction references. Morrigan is a composite module that consists of (i) an ensemble of table-based hardware Markov prefetchers that efficiently build and store variable length Markov chains out of the instruction TLB miss stream while using a new frequency-based replacement policy to manage their internal state, and (ii) a sequential prefetcher that operates only when the Markov prefetchers fail at producing prefetches. On a set of 45 industrial server workloads, Morrigan achieves 7.6% geometric mean speedup over a baseline without instruction TLB prefetching.

This work can motivate additional research* on the emerging domain of instruction address translation for server applications. Finally, Morrigan has also the potential to influence future industrial designs since recent works from Google [4] and Facebook [5] indicate instruction address translation as a major bottleneck for their datacenter applications; a well-known company has already shown interest at evaluating and implementing our proposal, Morrigan.

*Our artifacts have gone through the Artifact Evaluation of MICRO'21 conference, winning all available badges.

Preliminary results of this work have been presented at the Second Young Architect Workshop (YArch 2020), which was colocated with The 25th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS).

### 1.3 Page Size Aware Cache Prefetching [6]

This work is the first to reveal that leveraging modern prevalence and support for large pages can improve the effectiveness of cache prefetchers operating in the physical address space due to the arising opportunity for *safely* crossing 4KB physical page boundaries when the accessed blocks reside in large pages. In response, we design and propose the *Page-size Propagation Module (PPM)*, a fully-legacy preserving microarchitectural scheme that exploits the prevalence of large pages in modern systems to enable *safe* prefetching beyond 4KB physical page boundaries. We show that PPM can improve the performance of any lower-level cache prefetcher at almost zero storage and logic costs while not introducing new security vulnerabilities.

We further capitalize on PPM's benefits by designing a module comprised of two page size aware prefetchers, *i.e.*, prefetchers that exploit the PPM module, that inherently assume different page sizes to drive prefetching while using adaptive selection logic to enable the most appropriate of the competing prefetchers per cache access. This composite module is transparent to which cache prefetcher is used.

Across an extensive set of academic and industrial workloads, we show that the proposed page size exploitation schemes provide significant performance enhancements on various state-of-the-art lower-level cache prefetchers while not harming the performance of non memory-intensive workloads.

Preliminary results of this work have been presented, as a poster, in the 2021 ACM School on HPC Computer Architectures for AI and Dedicated Applications, winning the Best Poster Award.

## 2 Ongoing Research Works[§]

### 2.1 Efficient Cache Management for Graph-Processing Applications

Graph-processing workloads typically use input sets of very large volume and data-dependent graph traversal methods that exhibit highly irregular memory access patterns, resulting in poor cache locality and frequent DRAM accesses that compromise application performance [7]. This work aims at alleviating the pressure from the on-chip caches by accurately predicting which of the memory requests of graph-processing applications are cache-friendly or cache-averse, and accordingly route the corresponding requests to the most appropriate memory module, avoiding useless lookups that incur high latency costs. To ensure high prediction accuracy, this work opts to design and propose a *neural predictor* that relies on perceptron learning [8].

### 2.2 Page-Size Aware TLB Replacement Policy

Modern systems heavily use large page sizes to attenuate the address translation overheads [6] and typically implement last-level TLBs that simultaneously support more than one page sizes in the same structure [9]. This work aims at designing a TLB replacement policy that takes into account the page size information to drive the replacement of entries in the TLB. This is a multidimensional problem since different features (*e.g.,* page walk latency of different sized address translation entries) should be taken into account in the replacement of TLB entries.

### 2.3 Advanced TLB Management for Big Code Applications

The advent of applications with big code and big data footprints places tremendous pressure on the TLB hierarchy. This research work targets x86 architectures and focuses on the contention between instruction and data translation entries present in the last level of the TLB hierarchy. In response, it proposes novel techniques that prioritize the eviction of data or instruction translation entries depending on the execution phase.

[§]I cannot disclose many details about these works since they are not yet ready for publication.

## 2.4 Hardware-Accelerated Java Garbage Collection (GC)

Although automatic memory management has been continuously improving for decades, there is no solution that provides both low latency and high throughput in the same Java GC cycle. This work analyzes in depth the GC-generated cache pollution, quantifies its impact on the application performance, and correlates it with microarchitectural features such as application locality in the cache hierarchy. Finally, this work also opts to design a low-cost microarchitectural scheme capable of mitigating cache pollution caused by the GC threads.

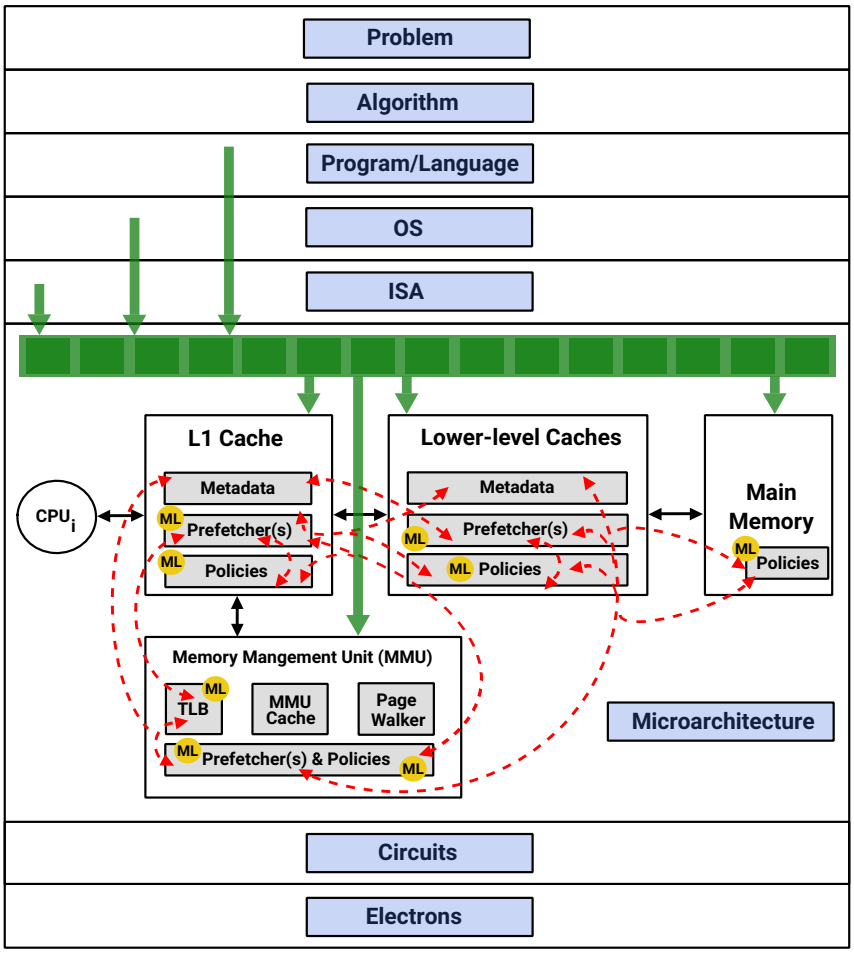


Figure 1: Research vision illustrated with a cartoon.

# 3 Research Vision

In my perspective, a new golden age for computer architecture is rising as non-conventional and groundbreaking approaches are needed to overcome the barrier that is placed by the end of Dennard Scaling and the slowing of Moore's law [10].

My research vision for the next years is to build **intelligent microarchitectures**. In this direction, I aim to (i) conduct software-assisted microarchitectural research by exploiting information available in different layers of the transformation hierarchy to push the envelope on microarchitectural research, given real-world technology constraints, and (ii) build smart microarchitectural prefetchers and predictors that internally leverage machine learning algorithms. Figure 1 illustrates my research vision for the next years.

## 3.1 Software-Assisted Microarchitectural Designs

The core idea is to pass software-sourced hints to the microarchitecture, through the ISA, that can improve the efficacy of the microarchitectural components. Hardware prefetching is one possible application of this idea (not the only one); the software encodes information about the nature of the memory access patterns and transmits them to the hardware prefetchers, which take them into account and adjust their prefetching strategy accordingly.

Another particularly appealing research direction is data and instruction cache prefetching via hybridization techniques. I envision the design of multiple low-cost hardware prefetchers with low prefetching scope and high accuracy. To make this idea a success, a sophisticated and highly accurate scheme that identifies the nature of the memory accesses and accordingly enables the most appropriate prefetcher is required. This scheme could be either a pure hardware predictor or a low-cost microarchitectural component that leverages software-sourced encoded information about the nature of the access patterns.

## 3.2 Machine Learning for Microarchitectural Prediction/Prefetching

Machine learning has provided great performance and accuracy gains in various microarchitectural domains. The most prevalent examples are branch prediction [11] and cache replacement policies [12]. Moreover, recent works [13, 14] highlight that ML-based hardware cache prefetchers have the potential to provide great benefits. The fundamental idea behind my research vision is that well-established ML algorithms could be used to design intelligent and practical microarchitectural prefetchers, predictors, and replacement policies for the TLB and cache hierarchy capable of providing great performance, coverage, and accuracy enhancements.

## 3.3 Other Domains

Apart from the domains presented above, I am particularly interested in conducting research on the following computer architecture areas: (i) interaction between TLB management and branch prediction, (ii) TLB management for native and virtualized environments, (iii) address translation for hybrid memory systems (*e.g.,* DRAM-NVMM), and (iv) serverless computing, among others.

## 3.4 Industry, Academia, and Teaching

My research vision also involves both industrial and academic collaborations since it aims at effectively tackling the upcoming technological challenges while being practical and impactful. Apart from that, my future plans include teaching computer architecture-related courses as well as advising students since I firmly believe that tutoring is the best way to master a topic.

# References


[1] G. Vavouliotis, L. Alvarez, V. Karakostas, K. Nikas, N. Koziris, D. A. Jiménez, and M. Casas, "Exploiting page table locality for agile tlb prefetching," in *2021 ACM/IEEE 48th Annual International Symposium on Computer Architecture (ISCA)*, pp. 85–98, 2021.

[2] A. Bhattacharjee, D. Lustig, and M. Martonosi, "Shared Last-level TLBs for Chip Multiprocessors," in *Proceedings of the 2011 IEEE 17th International Symposium on High Performance Computer Architecture*, HPCA '11, pp. 62–63, IEEE Computer Society, 2011.

[3] G. Vavouliotis, L. Alvarez, B. Grot, D. Jiménez, and M. Casas, "Morrigan: A composite instruction tlb prefetcher," in *MICRO-54: 54th Annual IEEE/ACM International Symposium on Microarchitecture*, MICRO '21, (New York, NY, USA), p. 1138{1153, Association for Computing Machinery, 2021.

[4] S. Kanev, J. P. Darago, K. Hazelwood, P. Ranganathan, T. Moseley, G.-Y. Wei, and D. Brooks, "Profiling a warehouse-scale computer," in *Proceedings of the 42Nd Annual International Symposium on Computer Architecture*, ISCA '15, (New York, NY, USA), pp. 158–169, ACM, 2015.

[5] M. Panchenko, R. Auler, B. Nell, and G. Ottoni, "Bolt: A practical binary optimizer for data centers and beyond," *2019 IEEE/ACM International Symposium on Code Generation and Optimization (CGO)*, pp. 2–14, 2019.

[6] G. Vavouliotis, G. Chacon, L. Alvarez, P. V. Gratz, D. A. Jiménez, and M. Casas, "Page Size Aware Cache Prefetching," in *Proceedings of the 55th International Symposium on Microarchitecture*, MICRO '22, pp. 956–974, 2022.

[7] Y. Zhang, V. Kiriansky, C. Mendis, S. Amarasinghe, and M. Zaharia, "Making caches work for graph analytics," in *2017 IEEE International Conference on Big Data (Big Data)*, pp. 293–302, 2017.

[8] D. Jiménez and C. Lin, "Dynamic branch prediction with perceptrons," in *Proceedings of the 7th International Symposium on High-Performance Computer Architecture*, HPCA '01, pp. 197–206, 2001.

[9] Abishek Bhattacharjee, "Advanced concepts on address translation." http://www.cs.yale.edu/homes/abhishek/abhishek-appendix-l.pdf.

[10] D. A. Patterson and J. L. Hennessy, *Computer Architecture: A Quantitative Approach*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1990.

[11] D. A. Jiménez and C. Lin, "Dynamic branch prediction with perceptrons," in *Proceedings of the 7th International Symposium on High-Performance Computer Architecture*, HPCA '01, (Washington, DC, USA), pp. 197–, IEEE Computer Society, 2001.

[12] D. A. Jiménez and E. Teran, "Multiperspective reuse prediction," in *Proceedings of the 50th Annual IEEE/ACM International Symposium on Microarchitecture*, MICRO-50 '17, (New York, NY, USA), pp. 436–448, ACM, 2017.

[13] R. Bera, K. Kanellopoulos, A. Nori, T. Shahroodi, S. Subramoney, and O. Mutlu, "Pythia: A customizable hardware prefetching framework using online reinforcement learning," in *MICRO-54: 54th Annual IEEE/ACM International Symposium on Microarchitecture*, MICRO '21, (New York, NY, USA), p. 1121{1137, Association for Computing Machinery, 2021.

[14] Z. Shi, A. Jain, K. Swersky, M. Hashemi, P. Ranganathan, and C. Lin, "A hierarchical neural model of data prefetching," in *Proceedings of the 26th ACM International Conference on Architectural Support for Programming Languages and Operating Systems*, ASPLOS '21, (New York, NY, USA), p. 861{873, Association for Computing Machinery, 2021.