A Look at Computer Architecture Evaluation Methodologies

Mario Badr and Natalie Enright Jerger
mario.badr@mail.utoronto.ca, enright@ece.utoronto.ca
Edward S. Rogers Sr. Department of Electrical and Computer Engineering, University of Toronto

***Abstract*—Throughout the years, computer architects have evaluated their designs in a number of ways. We take a closer look at the evaluations and methodologies of over 200 ISCA publications from 1973-2017. As computer architectures become more complex, and the tools afforded to architects grow, evaluations no longer rely on intuition or simple models. The trends in computer architecture evaluations help us understand how future architectures will be analyzed and identify gaps in the architect's toolbox. While architects have (and continue to) focus on performance, the last two decades have placed more emphasis on power, energy, and area overhead. Based on the tools used by architects, we identify six key papers that have been influential on past work and will likely continue to be influential in the future. The abundance of tools has resulted in an explosion of experimental data. Unfortunately, reproducibility appears to be an afterthought and without it, computer architecture becomes more art than science.**

1. Introduction

Architects face the difficult task of designing the machines of the future with the machines of today. They must carefully select workloads and metrics to make meaningful conclusions and recommendations. The process of evaluating a computer architecture, whether it be past, present, or future, is fundamental to research and development. Without objective evaluations, computer architecture becomes more art than science, and understanding the trade-offs between computer systems is near impossible. We look at how architects change and adapt their evaluation methodologies over the last 45 years.

Analyzing all computer architecture papers is infeasible. Instead, we survey over 200 papers between 1973 and 2017 from the ACM/IEEE International Symposium on Computer Architecture (ISCA). The selected papers are representative because they span a broad range of topics, which we introduce in Section 2.

The trends in computer architecture evaluation shed light on what was, is, and will be expected of contributions to the field. Our analysis focuses on which tools and methodologies were used in the past. The metrics used by architects definitively show the direction our community is taking by demonstrating which aspects of a design are important. The models that produce the estimates for these metrics help us understand gaps in the architect's toolbox. We take a closer look at specific tools to determine which have been influential on how we evaluate architectures. More importantly, we look at the impact these tools have had on how architects evaluate their designs.

Throughout the history of surveyed papers, performance metrics reign supreme. However, after 1995, we find that power, energy, and area become much more popular metrics (Section 3). Tools emerge to produce estimates for these metrics, and we identify key publications related to analytical modelling (Section 4) and simulation (Section 5) that have been influential on how we evaluate architectures.

Our survey is broader than related work that focuses on the pitfalls of architectural simulation [7], [8], [20]. As a result, we do not focus on, or validate, the use of a methodology or tool. We are more interested in how evaluation methodologies have changed over time and how the architecture community is progressing. We find that, while there has been a considerable shift to quantitative evaluations and experimental data, reproducibility is a key issue that needs to be addressed (Section 6).

2. Paper Selection

There are over 1600 publications in ISCA between 1973 and 2017 so we select a representative sample of publications to survey. We hypothesize that different topics in computer architecture are evaluated differently. For example, we expect that publications on networks-on-chip are more likely to use statistical simulation (e.g., uniform random traffic) to evaluate their designs. Conversely, publications on branch prediction are more likely to use applications to capture the complex control flow of real workloads. Therefore, we sample papers on a wide variety of topics from ISCA's 44 proceedings.

We present a classification of computer architecture along two axes. The first axis divides papers into the type of processor they use or propose. For conventional general purpose processors, we separate works that focus on a single versus multiple cores or processors. We classify unconventional processors as specialized architectures, such as throughput-oriented processors (i.e., GPUs) and other types of accelerators. The second axis divides papers into five categories based on the architectural component focused on. Many publications touch on more than one component of a computer architecture, such as the microarchitecture, memory, or interconnection network. If there is a primary focus of the paper on a particular component, then we assign the publication to that category. If the paper focuses on more

TABLE 1. PAPER CATEGORIES

Category	*Paper Focus*
Microarchitecture	The processor (e.g., branch prediction, simultaneous multithreading)
Memory	The memory hardware (e.g., cache replacement, phase change memory) or managing memory (e.g., coherence, consistency).
Networks	How the hardware is interconnected (e.g., networks-on-chip) or network interfaces.
Organization	The design of multiple components (e.g., dataflow architecture, multiprocessor).
Coordination	The management of multiple components to achieve a goal (e.g., quality of service).

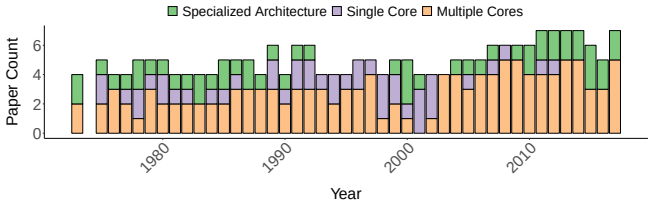

Figure 1. A timeline of surveyed papers broken down by architecture type.

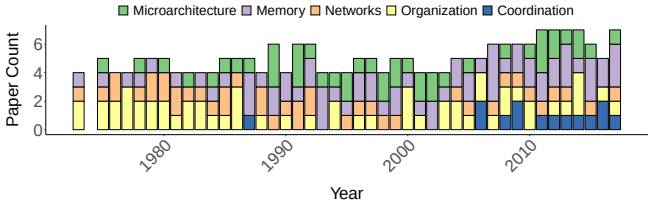

Figure 2. A timeline of surveyed papers broken down by category.

than one component, then we differentiate between papers that propose a new design (organization) versus papers that coordinate existing components (coordination). We briefly describe each category in Table 1. We now discuss how some papers are classified to give more insight to our methodology and the coverage of our survey.

We consider the Distributed Array Processor (DAP) to be a *specialized architecture* because it is described in the context of domain-specific kernels [27]. The DAP publication discusses the overall architecture of the processor and did not emphasize a specific component, so we categorize it as an *organization* paper. Another example of a specialized architecture is a dataflow processor [28]. Unlike DAP, Sakaei et al. focus on the pipeline stages that support a RISC ISA and is therefore categorized as *microarchitecture*. The work by Nishikawa et al. touches on the processor, memory, and interconnection network [19]. However, because the main emphasis is on how components are connected we categorize it as a *networks* paper for *multiple cores*. Goodman's work motivating on-chip caches is categorized as a *memory* paper for *multiple cores* [10]. Active Pages, a reconfigurable DRAM architecture that allows for processing near memory, is also a *memory* paper but for *single core* architectures [22]. Papers that deal with scheduling or management are assigned to the *coordination* category, such as Teodorescu and Torrellas' work on multiprocessor power management [30].

In the end, we survey 222 papers and strive to select publications that have had an impact on the field. We use Google Scholar's citation count as a rough proxy for impact and select four to seven papers from each ISCA proceedings, biasing our selection to papers that propose a new design or implementation. We do not include highly impactful methodology or tools papers (e.g., Wattch [4]) because we are interested in how architectures are evaluated rather than how a model is validated. Figure 1 shows the different types of architectures (single core, multiple cores, specialized) we surveyed throughout the years. Similarly, Figure 2 shows the same papers as they are categorized based on Table 1. Our complete database is available online for reference to the interested reader.[1]

3. Paper Analysis

The complexity of computer architecture evaluation has increased over the last few decades. Early on, many papers did not include a quantitative evaluation of their proposed design. More recently, papers that do not include experimental data are extremely rare. In addition, architects are evaluating their designs with more metrics. While Moore's law has lead to more performant computer systems, the architectures have also become more complex. The increased performance and complexity of systems has prompted more tools that model different aspects of the design. As a result, there is an explosion of experimental data being produced by architects. However, the scientific method not only requires experimental data but also reproducibility.

In this section, we illustrate the trends we found across the decades. We take a close look at the methodology of all of the surveyed papers. Our focus is on the metrics being evaluated and the tools used to obtain estimates of these metrics. Note that a paper may use multiple tools and metrics to evaluate their design. In fact, it is uncommon for recent publications to evaluate a single metric or use only one tool. Architects have shifted focus to the trade-offs their design provides across multiple metrics. We demonstrate this shift both quantitatively and qualitatively in the following timeline.

1973-1979 (27 papers). The majority of papers in the 1970s focus on the overall design (i.e., organization) of specialized architectures and multiple core systems. Only three of these papers provide a quantitative evaluation. The 1970s also has a number of papers focusing on how to interconnect multiprocessor systems. All but one of these papers included a quantitative evaluation of their design, and the metrics evaluated were area, performance, or both.

1. https://github.com/mariobadr/survey-wp3

In DAP, analytical models (that assume ideal parallelism) are used to quantify the accelerator's performance [27]. In the Columbia Homogeneous Parallel Processor (CHOPP), Sullivan et al. use an analytical model to project its area and performance based on current (at the time) data. Nishikawa et al. obtain performance and area estimates analytically for their network architecture [19]. In the work on Banyan networks, an analytical model was not available so statistical simulations were used to obtain a proxy for performance (i.e., number of network layers for different fanouts) [9]. Overall, only a 11 of 27 publications provide a quantitative evaluation and most rely on analytical models.

1980-1989 (46 papers). The 1980s continues to see work done on accelerators (i.e., specialized architectures) and networks. In addition, the reduced cost of memory and CPU components [10] stimulates research in microarchitecture and memory of conventional architectures. Nearly 60% of papers in the 1980s provide quantitative evaluations of their designs, compared to 40% in the previous decade. While analytical models are still used, architectural simulation begins to dominate (63% of papers). Hardware prototypes, mentioned as future work for several papers in the 1970s, also become more prominent (22% of papers).

We are careful to differentiate between tools used for motivating the research and tools used to evaluate a proposed design. For example, in Goodman's work on cache memories we only consider the simulations that evaluate the memory system using traces from PDP and VAX systems [10]. When identifying papers that prototype their design, we look for indications that either a part or the entire architecture has been synthesized, fabricated, or constructed. For example, we consider that Sakai et al. prototype their design because they provide detailed gate and pin counts (a proxy for area) of different hardware components, though the methodology is not provided [28].

1990-1999 (47 papers). Over 85% of papers in the 1990s provide a quantitative evaluation of their design, with an increased emphasis on conventional single-core architectures. In addition, the use of analytical models tapers off while architectural simulation sees even more use. We can explain this trend because a number of simulators being developed and released, such as SimpleScalar [5]. As a result, the majority of publications in the 1990s use one tool (a simulator) to evaluate performance.

Active Pages [22] is an example of a paper that uses multiple tools. An FPGA prototype is built to estimate the performance and area of the memory architecture as well as verify simulation results. SimpleScalar is used to obtain more detailed performance measurements for application speedup, processor stall cycles, etc. Finally, an analytical model is developed to understand the impact of the design more intuitively (the model's results are compared to simulation). Although there are some exceptions, such as Palacharla et al. [23], the evaluatio is more detailed than other publications in the decade.

There are a few examples of publications that consider energy and power as metrics, such as Pipeline Gating [18]. The evaluation uses SimpleScalar to estimate both speedup and a proxy for energy (*extra work*) for gated and not gated pipelines. The paper likely does not provide a direct measure for energy (e.g., joules) because an analytical model of the processor was not available. Two years after the Pipeline Gating publication, an analytical model would be integrated with SimpleScalar to bridge that gap [4]. We discuss the impact of these new models in the next decade.

2000-2009 (50 papers). With the exception of Rajwar et al. [26], all publications include a quantitative evaluation. A staggering 90% use architectural simulation thanks to the popularity of SimpleScalar, the introduction of Pin [16], and a focus on reducing simulation times [25]. However, analytical models that explore non-performance metrics (e.g., Wattch [4], CACTI [32], HotSpot [13], Orion [31]) also become popular. As a result, and unlike the previous two decades, 63% of publications use two or more tools in their evaluation.

Publications that want to evaluate power use more than the average of two tools. For example, the work by Teodorescu and Torrellas [30] uses a simulator and several analytical models, some of which include: (1) within-die variation using VARIUS; (2) dynamic power with Wattch [4]; (3) static power with HotLeakage; and (4) path layouts and wire delays with CACTI [32]. Another example is Synchroscalar, which developed a prototype, used multiple models to evaluate power, and adapted SimpleScalar to obtain performance measurements [21].

2010-2017 (52 papers). The introduction of McPAT in 2009 provides updated models for architects interested in power and energy and begins to be integrated into simulators [2], [3], [15]. Models focusing on a specific component, such as the power delivery network, are also introduced [33]. The abundance of tools available allows architects to explore more trade offs than ever before.

3.1. Summary

Figures 3 and 4 demonstrate the shift we observed in the mid-90s and early 2000s toward more power and energy evaluations. Note that the focus on performance helped aid in modelling other aspects of an architecture. Microarchitectural work done in 1997 reduced the complexity of wakeup and selection logic of a processor pipeline [23]. While the pipeline models presented in the paper were focused on performance (critical path latency), they were later instrumental in developing an analytical model for power estimation a few years later [4]. There were also many other metrics we observed but did not fully explore, such as reliability (e.g., mean time to failure) and quality of service.

Based on our observations, we highlight three influential analytical models in Section 4 and three influential simulation methodologies in Section 5. We also qualitatively reflect on our survey in Section 6 and provide recommendations for the future of the field.

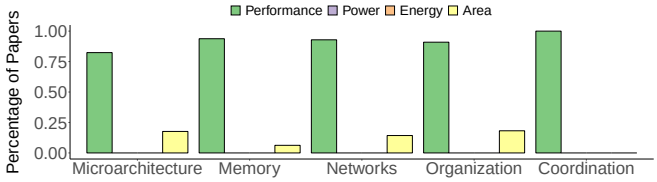

Figure 3. Metrics being evaluated before 1996.

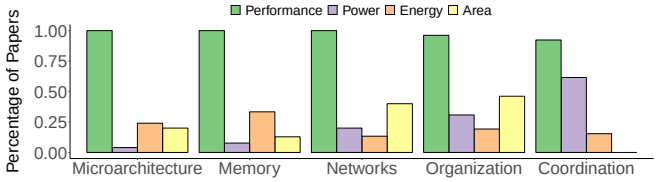

Figure 4. Metrics being evaluated after 1995.

4. Influential Analytical Models

We came across several analytical models in our surveyed papers. Here we take a closer look at published, validated models and select three that were influential on architecture evaluation. Before the year 2000, architects relied on low-level tools for estimating power that required hardware descriptions (e.g., circuit- or Verilog-level) of their designs. However, for architects that wanted to evaluate the power of (for example) a superscalar, out-of-order core they simulated, options were limited.

Wattch [4] introduces an analytical power model that accepts input parameters (i.e., switching events) from an architectural simulator. Therefore, rather than requiring a hardware description for a prototype, a software simulator models the hardware instead. The abstraction is useful because of the abundant use of architectural simulators. The Wattch framework was integrated into the SimpleScalar simulator, which likely influenced the popularity of both tools; at the time of writing, Wattch has over 3000 citations on Google Scholar. With Wattch and SimpleScalar, architects were able to produce performance and power evaluations in one simulation run.

McPAT [15], published in 2009, maintains the level of abstraction provided by Wattch and integrates with architectural simulators, which allows dynamic events to be passed into the McPAT modelling framework. Aside from being more up-to-date, McPAT differs from Wattch in two ways. First, it can provide feedback to the architectural simulation on power dissipation and thermal data. Cycle-level information on power consumption and temperature has made it useful for research on architectures that are power- or thermal-aware. Second, McPAT also accepts as input a configuration for its own microarchitectural model. Combined with the architectural configuration embedded in the simulator (which provides McPAT with dynamic events), the additional static configuration allows more customizability of the underlying microarchitecture. McPAT is not tightly coupled to any one simulator and has been integrated into many, such as gem5 [3] and ESESC [2], improving its popularity by making it more accessible to architects.

The most influential work on power and area models, dating back to the mid-90s, is CACTI [32]. CACTI's estimates and analytical methodologies continue to influence other models, including the aforementioned McPAT (area and timing [15]) and Wattch (capacitance models [4], [15]) frameworks. CACTI is focused on memory hardware (e.g., SRAM, DRAM), and is by far the most popular analytical model and tool we found in papers we surveyed. The popularity of CACTI is due to the reliance of architectures on memory arrays, which is not limited to research done in the *memory* category. For example, CRIB uses CACTI to model large microarchitectural structures such as the reorder buffer and register file [12]. In Kilo-NOC, CACTI was extended to model the SRAM FIFOs in network-on-chip routers [11]. CACTI is also used in providing estimates for the memory requirements of specialized architectures.

As more architects use a tool it becomes easier for other researchers to use the same tool and compare numbers. Thus, as a tool becomes more widely used, there is a snowball effect to its popularity. We believe that tools become more widely used when they are updated and provide a consistent, easy-to-use interface. CACTI has been frequently updated over the years (currently on version 6), keeping it up-to-date with trends in technology and maintaining its popularity. Similarly, McPAT has also been updated, with its last update to version 1.3 in September of 2015. By not being heavily tied to a specific architectural simulator (i.e., generic interface), the tools have become very popular amongst architects.

5. Influential Simulation Methodologies

The most-used architectural simulator in the papers we surveyed was SimpleScalar, implemented between 1994 and 1996, for evaluating single core microprocessors [5]. As mentioned in Section 4, a power modelling framework was integrated into the simulator [4]. SimpleScalar was also expanded to simulate multiple cores. By the early 2000s, SimpleScalar was being used by a variety of papers in multiple venues. In 2001 more than a third of papers that were in the conference on Parallel Architectures and Compilation Techniques (PACT) and the International Symposium on Microarchitecture (MICRO) used the simulator [1].

With the surge of simulators available to architects by the mid to late 90s, research was done on the validity of simulation results [7], [8]. Gibson et al. demonstrated the importance of including the operating system when simulating an architecture [8]. In the surveyed papers, we saw a brief move to full-system simulators, such as Simics [17], which booted an operating system before running benchmarks. However, the use of full-system simulators is not as prolific as previous tools for a number of reasons.

1) Simulation time was already prohibitive. The majority of publications we surveyed either simulated a portion of the benchmarks or used a reduced input set (or both). Modelling the full system meant even longer simulation times.

2) While execution-driven simulators typically relied on just an application binary, full-system simulators required a number of inputs (e.g., disk image, linux kernel). The additional inputs required by full-system simulators increased the barrier-to-entry for obtaining results on a new design.
3) Reasoning about the results of a full system is more complicated than just the architecture. A complete analysis would require architects to understand how the operating system, disk, and other components impacted their simulations.
4) In many cases, architects integrated full-system simulators with an architectural simulator to obtain more detailed data of the events happening in an architecture. The integration increases an architect's barrier-to-entry and the simulation time.

Prohibitive simulation times has limited design space exploration of architectures. To compensate, the architect's toolbox has responded in two ways. The first is to raise the level of abstraction for simulated components. For example, using a high-level cache model as opposed to a detailed model can speed up simulation time. This is useful if the architectural idea being evaluated is not overly sensitive to the memory system. We find that Pin has been influential in simulating architectures at various levels of abstrcation [16]. The second is to choose a small but representative sample of the application to simulate. We find that SimPoint has been influential in keeping simulation time tractable [25].

Pin is a dynamic binary instrumentation tool that has been very influential in architectural simulation, second only to SimpleScalar. Pin takes a binary as input and runs the application natively on the system. In addition, a Pintool can be used to instrument the application at different granularities (e.g., instruction, function calls). Alone, Pin is not an architectural simulator. However, Pintools can provide the necessary timing models to create an architectural simulator. Pin's ease-of-use and flexibility has made it widely used in the literature. An early use of Pin for architectural simulation was a multiprocessor cache simulator [14]. More recently, simulators that focus on a raised level of abstraction, such as Sniper [6] and ZSim [29], are driven by Pin. Pin has also been used to generate traces or collect characteristics of an application.

Many of the surveyed publications simulated a small sample of the total execution of an application. For example, some methodologies would only simulate the first billion instructions to keep simulation times reasonable. Other methodologies would offset the simulation by a certain number of instructions before collecting results. The most complicated methodologies would collect multiple samples throughout the simulation, however these were rare. SimPoint is a methodology for selecting a *representative* sample of an application to simulate [25]; the first billion instructions of execution is not necessarily representative of the entire application's time-varying behaviour. The SimPoint methodology, and its derivatives [24], has been widely used to reduce simulation time in the papers we surveyed.

We believe that SimpleScalar was popular for similar reasons to those discussed for CACTI and McPAT in Section 4. The simulator received many updates; version 2.0 was released in 1997, and in 2001 there were experimental releases for the x86 and ARM instruction-sets as well as the pre-release of version 3.0 [1]. Similarly, Pin and SimPoint are currently on their third versions. Intel continues to maintain Pin, keeping it up-to-date with the latest updates to the x86 instruction set.

6. Conclusion

Throughout the surveyed papers, the majority of methodologies are insufficient in the context of reproducibility. While difficult to quantify, we found the methodology section of papers in the 90s and early 2000s the easiest to read and digest. The use of one to two tools, sometimes integrated together (e.g., SimpleScalar and Wattch), is typically accompanied with a table of the inputs used while another table lists the applications being evaluated. Complete methodologies include the sampling methodology used (i.e., simulating a subset of instructions to keep simulation time low) and the input set sizes of the benchmarks, though this was less common than we anticipated. Before the 90s, we found that the methodology of many evaluations was either non-existent or scattered throughout the paper.

The abundance of tools available by the mid 2000s gives architects a wide variety of metrics that could be evaluated. But while additional metrics allow architects to demonstrate more trade-offs, the data also take up page real estate. A significant chunk of recent publications is dedicated to the evaluation with a bombardment of experimental data produced by a variety of tools. One would expect a more detailed methodology commensurate with the number of tools used and graphs produced but that is not the case.

We also found a number of papers that rely on in-house simulation, even in recent years where an abundance of tools are available. The use of an in-house simulator may indicate that the available tools are not sufficient for the evaluation; we believe that is the case when datacenters and warehouse scale servers need to be modelled. In-house simulators may also be used when the time to produce an estimate with another tool is intractable. There are several examples of papers that produce traces with a validated simulator to then drive their own. Both these cases demonstrate that the architect's toolbox, despite its size, is not keeping pace with the needs of architects in terms of simulation time and modelling large-scale systems.

Overall, we believe architects would be hard-pressed to reproduce the majority of publications in ISCA. Generating experimental data is only one part of the scientific method and without reproducibility, the data is of little use. A concerted effort needs to be taken by architects, tools developers, and reviewers to ensure that the methodology of publications is complete. Architects should consider releasing their modifications to existing simulators (or the entirety of their in-house simulators) along with the scripts used to generate the experimental data. Tools developers should

provide formats that architects can use to export their inputs so that they can be reproduced and analyzed by others in the field. The move to quantitative evaluations is only the first step in establishing computer architecture as a science. We hope our paper will spur increased interest in the second step: reproducibility.

References

[1] "SimpleScalar LLC – to serve and project," 2011, Accessed: 2017-12-28. [Online]. Available: http://www.simplescalar.com

[2] E. K. Ardestani and J. Renau, "ESESC: A fast multicore simulator using Time-Based Sampling," in *Proceedings of the 19th HPCA*, 2013, pp. 448–459.

[3] 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 Comp. Arch. News*, vol. 39, no. 2, pp. 1–7, 2011.

[4] D. Brooks, V. Tiwari, and M. Martonosi, "Wattch: A Framework for Architectural-level Power Analysis and Optimizations," in *Proceedings of the 27th ISCA*, 2000, pp. 83–94.

[5] D. Burger and T. M. Austin, "The SimpleScalar Tool Set, Version 2.0," *SIGARCH Comp. Arch. News*, vol. 25, no. 3, pp. 13–25, 1997.

[6] T. E. Carlson, W. Heirmant, and L. Eeckhout, "Sniper: Exploring the level of abstraction for scalable and accurate parallel multi-core simulation," in *Proceedings of the 2011 International SC*, 2011, pp. 1–12.

[7] R. Desikan, D. Burger, and S. W. Keckler, "Measuring Experimental Error in Microprocessor Simulation," in *Proceedings of the 28th ISCA*, 2001, pp. 266–277.

[8] J. Gibson, R. Kunz, D. Ofelt, M. Horowitz, J. Hennessy, and M. Heinrich, "FLASH vs. (Simulated) FLASH: Closing the Simulation Loop," in *Proceedings of the 9th ASPLOS*, 2000, pp. 49–58.

[9] L. R. Goke and G. J. Lipovski, "Banyan Networks for Partitioning Multiprocessor Systems," in *Proceedings of the 1st ISCA*, 1973, pp. 21–28.

[10] J. R. Goodman, "Using Cache Memory to Reduce Processor-Memory Traffic," in *Proceedings of the 10th ISCA*, 1983, pp. 124–131.

[11] B. Grot, J. Hestness, S. W. Keckler, and O. Mutlu, "Kilo-NOC: A Heterogeneous Network-on-chip Architecture for Scalability and Service Guarantees," in *Proceedings of the 38th ISCA*, 2011, pp. 401–412.

[12] E. Gunadi and M. H. Lipasti, "CRIB: Consolidated Rename, Issue, and Bypass," in *Proceedings of the 38th ISCA*, 2011, pp. 23–32.

[13] W. Huang, S. Ghosh, S. Velusamy, K. Sankaranarayanan, K. Skadron, and M. R. Stan, "HotSpot: a compact thermal modeling methodology for early-stage VLSI design," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 14, no. 5, pp. 501–513, 2006.

[14] A. Jaleel, R. S. Cohn, C.-K. Luk, and B. Jacob, "CMP$im: A Pin-based on-the-fly multi-core cache simulator," in *Proceedings of the 4th Annual Workshop on MoBS*, 2008, pp. 28–36.

[15] S. Li, J. H. Ahn, R. D. Strong, J. B. Brockman, D. M. Tullsen, and N. P. Jouppi, "McPAT: An integrated power, area, and timing modeling framework for multicore and manycore architectures," in *Proceedings of the 42nd MICRO*, 2009, pp. 469–480.

[16] 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," in *Proceedings of the 2005 Conference on PLDI*, 2005, pp. 190–200.

[17] 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.

[18] S. Manne, A. Klauser, and D. Grunwald, "Pipeline Gating: Speculation Control for Energy Reduction," in *Proceedings of the 25th ISCA*, 1998, pp. 132–141.

[19] S. Nishikawa, M. Sato, and K. Murakami, "Interconnection Unit for Poly-Processor System: Analysis and Design," in *Proceedings of the 5th ISCA*, 1978, pp. 216–222.

[20] T. Nowatzki, J. Menon, C. H. Ho, and K. Sankaralingam, "Architectural Simulators Considered Harmful," *IEEE Micro*, vol. 35, no. 6, pp. 4–12, 2015.

[21] J. Oliver, R. Rao, P. Sultana, J. Crandall, E. Czernikowski, L. W. Jones IV, D. Franklin, V. Akella, and F. T. Chong, "Synchroscalar: A Multiple Clock Domain, Power-Aware, Tile-Based Embedded Processor," in *Proceedings of the 31st ISCA*, 2004, pp. 150–161.

[22] M. Oskin, F. T. Chong, and T. Sherwood, "Active Pages: A Computation Model for Intelligent Memory," in *Proceedings of the 25th ISCA*, 1998, pp. 192–203.

[23] S. Palacharla, N. P. Jouppi, and J. E. Smith, "Complexity-effective Superscalar Processors," in *Proceedings of the 24th ISCA*, 1997, pp. 206–218.

[24] H. Patil, R. Cohn, M. Charney, R. Kapoor, A. Sun, and A. Karunanidhi, "Pinpointing Representative Portions of Large Intel®Itanium®Programs with Dynamic Instrumentation," in *Proceedings of the 37th MICRO*, 2004, pp. 81–92.

[25] E. Perelman, G. Hamerly, M. Van Biesbrouck, T. Sherwood, and B. Calder, "Using SimPoint for Accurate and Efficient Simulation," in *Proceedings of the 2003 International Conference on MMCS*, 2003, pp. 318–319.

[26] R. Rajwar, M. Herlihy, and K. Lai, "Virtualizing Transactional Memory," in *Proceedings of the 32nd ISCA*, 2005, pp. 494–505.

[27] S. F. Reddaway, "DAP—a Distributed Array Processor," in *Proceedings of the 1st ISCA*, 1973, pp. 61–65.

[28] S. Sakai, y. Yamaguchi, K. Hiraki, Y. Kodama, and T. Yuba, "An Architecture of a Dataflow Single Chip Processor," in *Proceedings of the 16th ISCA*, 1989, pp. 46–53.

[29] D. Sanchez and C. Kozyrakis, "ZSim: Fast and accurate microarchitectural simulation of thousand-core systems," in *Proceedings of the 40th ISCA*, 2013, pp. 475–486.

[30] R. Teodorescu and J. Torrellas, "Variation-Aware Application Scheduling and Power Management for Chip Multiprocessors," in *Proceedings of the 35th ISCA*, 2008, pp. 363–374.

[31] H.-S. Wang, X. Zhu, L.-S. Peh, and S. Malik, "Orion: A Power-performance Simulator for Interconnection Networks," in *Proceedings of the 35th MICRO*, 2002, pp. 294–305.

[32] S. J. E. Wilton and N. P. Jouppi, "CACTI: An Enhanced Cache Access and Cycle Time Model," *IEEE Journal of Solid-State Circuits*, vol. 31, no. 5, pp. 677–688, 1996.

[33] R. Zhang, K. Wang, B. H. Meyer, M. R. Stan, and K. Skadron, "Architecture Implications of Pads As a Scarce Resource," in *Proceeding of the 41st ISCA*, 2014, pp. 373–384.