RESEARCH STATEMENT

Henry (Hank) Hoffmann (hankhoffmann@cs.uchicago.edu)

My research explores the **principled** design and implementation of **self-aware computer systems**; i.e., those that automatically adapt their behavior to their environment and user goals. I am a systems researcher, but in my quest to build real systems with formally verifiable behavior I incorporate **control theory** and statistical **machine learning**.

Background

As a graduate student, I fortunately had the opportunity to turn academic research (the Raw processor [35–37]) into a commercial product (the Tilera TILE family of processors [18, 38]). In addition to the excitement of seeing an idea transformed into something people could hold, the process of commercializing research opened my eyes to the challenges that real engineers (rather than academics) face in the current computing landscape. When meeting with customers, the following themes came up over and over again:

- The need to meet multiple — often competing — goals; e.g., high performance and low energy.
- The burden that complex, modern hardware puts on the engineers who must use these systems.
- The goals must be maintained in dynamic environments where components fail and workloads fluctuate.

Research Goal: Principled Self-aware Computing

While today's hardware and software systems haven given customers more features and choices, what these customers really need is help ensuring that their applications meet goals (e.g., performance, power, accuracy constraints) while remaining responsive to dynamic changes. Creating such systems is challenging for a number of reasons. First, these goals are interrelated, so choices made to meet one (e.g., performance) will affect others (e.g., energy). Second, developers need assurance that the system will respond correctly in fluctuating environments. Third, the system can respond at a number of levels – from hardware resource usage to algorithmic changes – and choices made across levels must be coordinated.

My work addresses these challenges by proposing and realizing the vision of **self-aware** computing systems that adapt to meet user-specified goals automatically and optimally [13]. I have developed new machine learning models for capturing the online interaction of different goals in modern computing systems [10, 15, 23, 25, 28, 39, 40]. To provide formal assurance of behavior in a dynamic environment, I have developed generalized control theoretic techniques that are 1) suitable for the unique challenges of computing systems, 2) emit formal guarantees of convergence to goals, and 3) have bounded convergence time [3–5, 9, 14, 15, 20, 21, 24, 26, 31, 40]. Finally, I work across the system stack from the circuit- [34] to the application-level [3], with recent work coordinating across the system stack [2, 6, 8–10]. Thus far, I have applied self-awareness to energy, performance, and application accuracy goals. Given the work's energy-saving potential, Scientific American named my self-aware computing model one of ten **World Changing Ideas** in December 2011 [1].

The incorporation of control theory into computing systems is one of my work's unique characteristics. Indeed, I envision deployment of systems which can be relied on to respond correctly to the unexpected, but how do we test such a capability? Any test scenario, is by definition, expected. Control theory provides valuable methodologies supporting formal reasoning about system dynamics before the system is deployed.

As an analogy consider cruise control in cars. They use simple control theoretic models relating fuel flow to speed. While myriad variables affect speed (including incline, wind velocity, road surface, tire pressure, and others), cruise controls ignore most of these, instead modeling current speed as a function of previous fuel flow. With this simple model, cruise controls provably drive the car at user-specified speeds despite road and environmental changes (e.g., turning into a strong headwind). One of my research goals is to bring this same capability – **formally verifiable dynamic behavior**, especially for performance, energy, and application accuracy – to computing systems. Thus, much of my work generalizes control models for use in computer systems [3, 4, 10, 21].

This statement describes my research 1) building self-aware systems that manage performance and energy tradeoffs, 2) building self-aware applications that manage accuracy and energy tradeoffs, and 3) approaches that combine application and system-level self-awareness to achieve greater energy savings than either in isolation. I conclude by discussing my future plans for self-aware computing, including applying it to additional goals like security and resilience to transient hardware faults.

Self-aware Systems for Energy and Power Awareness

As thermal dissipation limits multicore scaling, energy and power have become first order concerns for all computing systems, from mobile (where energy defines battery life) to supercomputers (where power limits the transistors that can be brought to bear on science). To address power and energy, computer architects are exposing more and more complexity to software; e.g., by increasing processor heterogeneity and placing power states under software control.

A major challenge is that different hardware systems have very different power and performance tradeoffs. I have demonstrated that approaches which minimize energy on one system may be very inefficient on others [7, 19, 22]. Another challenge is that some systems want to deliver guaranteed performance (to meet real-time or quality-of-service constraints) and minimize energy while others need to guarantee power consumption (to prevent thermal faults) and maximize performance delivered to applications.

To address these concerns, I have developed OS-level techniques for meeting performance goals with minimal energy consumption using control theory on embedded systems [15, 24, 26]. Control solutions rely on robust models of system behavior, so I have complemented my control work with probabilistic graphical models that learn energy performance tradeoffs dynamically [28]. In other work, I have implemented OS support for maximizing delivered performance under a power constraint [14] and I have demonstrated these approaches deliver higher performance than Intel's commodity hardware approach to this problem [39]. I am currently working with a team to integrate these techniques into the Argo operating system for exascale supercomputers [29].

One lesson I learned from this OS-level work is that greater energy savings are possible with more hardware support for both observing and adapting resource usage. Thus, I have several hardware projects that expose a wide array of resources for management by my self-aware OS [13], including novel energy monitoring circuits and low-voltage caches [34]. In other work, I have designed self-aware hardware for adapting GPU resource management to minimize interactive applications' energy consumption [31] and developed techniques for managing individual ALUs and cache banks [40].

Whether implemented in hardware or software, these techniques have common themes based on the lessons I learned at my startup company. First, they handle constraints in multiple dimensions including performance, power, and energy. Second, they take high-level user goals and automatically manage resources to meet those goals. This automation 1) greatly relieves application programmer burden and 2) allows developers to produce more portable code by automatically tailoring resource usage to meet high-level goals optimally.

Self-aware Applications: Trading Accuracy for Other Benefits

I have spent significant effort creating frameworks enabling self-aware applications, focusing in particular on techniques for building **approximate** applications that dynamically tailor their output quality to available resources [30]. This work raises several challenges including creation of approximate applications and management of approximation so that just the right amount is used to meet goals given available resources. My work has addressed these challenges by: 1) creating new techniques that construct approximate applications from existing applications [16, 17, 27, 33] and 2) generalizing control theoretic solutions for managing dynamic application behavior [3–5, 20, 21].

Approximation increases application flexibility; e.g., when resources are scarce, rather than stop computing, an approximate application produces a slightly less accurate result [30]. While other work in this area requires the programmer to change application code (e.g., specify which functions or data are amenable to approximation), my work has developed two techniques that automatically find approximate variations for existing programs. First, **loop perforation** discards loop iterations and evaluates whether the resulting program still produces an acceptable output [16, 27, 33]. Second, **dynamic knobs** transform existing statically configured parameters into a data structure allowing dynamic response to changing resources [17].

When we deploy approximate applications, we would like guarantees that the application will behave as desired, generally using as little approximation as possible to meet its goals. For reasons mentioned above, I have applied control theoretic solutions to this problem [17]. Control systems have proved so useful in this regard, that I have worked to make them accessible to application programmers in general.

Therefore, I have designed a framework that automatically synthesizes control systems – with formally verifiable properties – for software applications [3]. Controllers produced by this technique guarantee that they will converge to the desired goal, provide bounds on convergence time, and have quantifiable robustness to error. I extended these techniques to control multiple goals simultaneously while maintaining formal guarantees [4]. I have also packaged a

controller into a library that users can add to their applications to meet soft real-time or QoS guarantees with minimal energy [21]. Critically, that library is platform independent – as performance/energy tradeoffs change from platform to platform, the application does not need to be rewritten – all energy related resource management is handled in the library layer. I have extended this library to support both performance and power constraints and dynamically switch between the two [20]. A novel contribution of this work is a generalization of classic control techniques so that the controller can be developed without specific knowledge of the system under control [11, 12, 24]. This generalization allows the same control implementations to be used in many different scenarios without rewriting the control code. These contributions have impacted both computing [3–5] and control [24, 26].

Coordinating Self-aware Systems and Applications

Adaptation is an enticing property for both systems and applications; many computer scientists have experimented with adaptation at different levels of the computing stack. Very little work, however, has studied the interaction of adaptation across different layers of the stack. Important questions must be addressed, such as: what issues arise when application and system adapt simultaneously? what benefits can be gained by combining adaptation in both regimes? what mechanisms can coordinate adaptation across the system stack?

Studying these issues is the subject of some of my latest research. In a recent invited talk, I argued that coordinating self-aware adaptation across system and application layer both avoids potential bad behavior (e.g., applications and systems violating timing and power constraints) and enables new opportunities (e.g., increased battery life for mobile devices) [8]. I have demonstrated that hierarchical control systems can coordinate across layers while still providing formal guarantees [9]. In other work, I have shown how combined hardware and software approaches to resilience and approximation can greatly reduce the overhead required to detect transient hardware faults [32]. I have also developed methods to combine system and application self-awareness to provide formal energy consumption guarantees even when application and system interaction are unknown prior to run time [10]. Finally, I have recently demonstrated that the combination of self-aware applications and systems can provide hard real-time guarantees and near optimal energy consumption for small, infrequent degradations in application output quality [2].

This is a new phase of research, but I believe study of interacting adaptation mechanisms will become increasingly important as developers increase the number of adaptive applications and systems deployed in real settings. Therefore, addressing the questions above is a major focus of my research going forward.

Summary and Future Work

My research studies self-aware adaptation as a fundamental property of computing systems. Self-aware systems have enhanced capability to respond to unknowns and fluctuating operating environments. By building such systems on a sound mathematical foundation, we can formally reason about their behavior in fluctuating environments and increase users' confidence they will perform as desired.

Self-awareness as I have defined it here, then, is a cross-cutting issue and can be applied at many levels of the traditional computing stack. I have demonstrated the cross-cutting nature of this work by publishing in a number of venues, from circuit conferences (where I have detailed new methods for monitoring and adapting cache energy [34]) to software engineering (where I have developed automated methods for synthesizing software control systems [3]) and many venues in between, including embedded/real-time/cyberphyscial, operating systems, and architecture.

In the future I will further the study of self-awareness as a first class property of computing. While my current research has developed self-aware techniques for performance, power, energy, and accuracy management, I will add new constraints to this mix. I am currently working on extending these techniques to support security and fault tolerance goals. I will also further study the interaction of adaptive systems. If the techniques I have outlined here are to become reality in the majority of computing systems, we must have principled methods for coordinating the behavior of multiple self-aware mechanisms developed by separate engineering teams. When we know that two separately developed self-aware systems can be simultaneously deployed to achieve the benefits of both, there will be no barrier to their widespread use.

References

[1] Editors, E. Svoboda, C. Mims, F. Diep, M. Peck, and S. Fecht. "World-Changing Ideas: 10 new technologies that will make a difference". In: *Scientific American* 305.6 (Dec. 2011).

[2] A. Farrell and H. Hoffmann. "MEANTIME: Achieving Both Minimal Energy and Timeliness with Approximate Computing". In: *Submission*. 2015.

[3] A. Filieri, H. Hoffmann, and M. Maggio. "Automated design of self-adaptive software with control-theoretical formal guarantees". In: *36th International Conference on Software Engineering,* **ICSE**. 2014.

[4] A. Filieri, H. Hoffmann, and M. Maggio. "Automated multi-objective control for self-adaptive software design". In: *Proceedings of the 2015 10th Joint Meeting on Foundations of Software Engineering,* **ESEC/FSE**. 2015.

[5] A. Filieri, M. Maggio, K. Angelopoulos, N. D'Ippolito, I. Gerostathopoulos, A. B. Hempel, H. Hoffmann, P. Jamshidi, E. Kalyvianaki, C. Klein, F. Krikava, S. Misailovic, A. V. Papadopoulos, S. Ray, A. M. Sharifloo, S. Shevtsov, M. Ujma, and T. Vogel. "Software Engineering Meets Control Theory". In: *10th IEEE/ACM International Symposium on Software Engineering for Adaptive and Self-Managing Systems,* **SEAMS**. 2015.

[6] C. Hankendi, H. Hoffmann, and A. Coskun. "Adapt&Cap: Coordinating System and Application-level Adaptation for Power Constrained Systems". In: **IEEE Design & Test** (to appear).

[7] H. Hoffmann. "Racing and pacing to idle: an evaluation of heuristics for energy-aware resource allocation". In: *Proceedings of the Workshop on Power-Aware Computing and Systems,* **HotPower**. 2013.

[8] H. Hoffmann. "A Case for Runtime Coordination of Accuracy-aware Applications and Power-aware Systems (invited)". In: *First SIGPLAN Workshop on Probabilistic and Approximate Computing at PLDI*. 2014.

[9] H. Hoffmann. "CoAdapt: Predictable Behavior for Accuracy-Aware Applications Running on Power-Aware Systems". In: *26th Euromicro Conference on Real-Time Systems,* **ECRTS**. 2014.

[10] H. Hoffmann. "JouleGuard: energy guarantees for approximate applications". In: *Proceedings of the 25th Symposium on Operating Systems Principles,* **SOSP**. 2015.

[11] H. Hoffmann, J. Eastep, M. D. Santambrogio, J. E. Miller, and A. Agarwal. "Application heartbeats: a generic interface for specifying program performance and goals in autonomous computing environments". In: *Proceedings of the 7th International Conference on Autonomic Computing,* **ICAC**. 2010.

[12] H. Hoffmann, J. Eastep, M. D. Santambrogio, J. E. Miller, and A. Agarwal. "Application heartbeats for software performance and health". In: *Proceedings of the 15th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming,* **PPOPP**. 2010.

[13] H. Hoffmann, J. Holt, G. Kurian, E. Lau, M. Maggio, J. E. Miller, S. M. Neuman, M. E. Sinangil, Y. Sinangil, A. Agarwal, A. P. Chandrakasan, and S. Devadas. "Self-aware computing in the Angstrom processor". In: *The 49th Annual Design Automation Conference 2012,* **DAC**. 2012.

[14] H. Hoffmann and M. Maggio. "PCP: A Generalized Approach to Optimizing Performance Under Power Constraints through Resource Management". In: *11th International Conference on Autonomic Computing,* **ICAC**. 2014.

[15] H. Hoffmann, M. Maggio, M. D. Santambrogio, A. Leva, and A. Agarwal. "A generalized software framework for accurate and efficient management of performance goals". In: *Proceedings of the International Conference on Embedded Software,* **EMSOFT**. 2013.

[16] H. Hoffmann, S. Misailovic, S. Sidiroglou, A. Agarwal, and M. Rinard. *Using Code Perforation to Improve Performance, Reduce Energy Consumption, and Respond to Failures*. Tech. rep. MIT-CSAIL-TR-2009-042. MIT, 2009.

[17] H. Hoffmann, S. Sidiroglou, M. Carbin, S. Misailovic, A. Agarwal, and M. C. Rinard. "Dynamic knobs for responsive power-aware computing". In: *Proceedings of the 16th International Conference on Architectural Support for Programming Languages and Operating Systems,* **ASPLOS**. 2011.

[18] H. Hoffmann, D. Wentzlaff, and A. Agarwal. "Remote Store Programming". In: *High Performance Embedded Architectures and Compilers, 5th International Conference,* **HiPEAC**. 2010.

[19] C. Imes and H. Hoffmann. "Minimizing energy under performance constraints on embedded platforms: resource allocation heuristics for homogeneous and single-ISA heterogeneous multi-cores". In: **SIGBED Review** 11.4 (2014).

[20] C. Imes and H. Hoffmann. "Bard: A Unified Framework for Managing Soft Timing and Power Constraints". In: *Submission*. 2015.

[21] C. Imes, D. H. K. Kim, M. Maggio, and H. Hoffmann. "POET: a portable approach to minimizing energy under soft real-time constraints". In: *21st IEEE Real-Time and Embedded Technology and Applications Symposium,* **RTAS**. 2015.

[22] D. H. K. Kim, C. Imes, and H. Hoffmann. "Racing and Pacing to Idle: Theoretical and Empirical Analysis of Energy Optimization Heuristics". In: *2015 IEEE 3rd International Conference on Cyber-Physical Systems, Networks, and Applications,* **CPSNA**. 2015.

[23] M. Maggio, H. Hoffmann, A. V. Papadopoulos, J. Panerati, M. D. Santambrogio, A. Agarwal, and A. Leva. "Comparison of Decision-Making Strategies for Self-Optimization in Autonomic Computing Systems". In: **ACM Trans. Auton. Adapt. Syst.** 7.4 (Dec. 2012).

[24] M. Maggio, H. Hoffmann, M. D. Santambrogio, A. Agarwal, and A. Leva. "Controlling software applications via resource allocation within the heartbeats framework". In: *Proceedings of the 49th IEEE Conference on Decision and Control,* **CDC**. 2010.

[25] M. Maggio, H. Hoffmann, M. D. Santambrogio, A. Agarwal, and A. Leva. "Decision making in autonomic computing systems: comparison of approaches and techniques". In: *Proceedings of the 8th ACM international conference on Autonomic computing* **ICAC**. 2011.

[26] M. Maggio, H. Hoffmann, M. D. Santambrogio, A. Agarwal, and A. Leva. "Power Optimization in Embedded Systems via Feedback Control of Resource Allocation". In: **IEEE Trans. Contr. Sys. Techn.** 21.1 (2013).

[27] S. Misailovic, S. Sidiroglou, H. Hoffmann, and M. C. Rinard. "Quality of service profiling". In: *Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering,* **ICSE**. 2010.

[28] N. Mishra, H. Zhang, J. D. Lafferty, and H. Hoffmann. "A Probabilistic Graphical Model-based Approach for Minimizing Energy Under Performance Constraints". In: *Proceedings of the Twentieth International Conference on Architectural Support for Programming Languages and Operating Systems,* **ASPLOS**. 2015.

[29] S. Perarnau, R. Thakur, K. Iskra, K. Raffenetti, F. Cappello, R. Gupta, P. H. Beckman, M. Snir, H. Hoffmann, M. Schulz, and B. Rountree. "Distributed Monitoring and Management of Exascale Systems in the Argo Project". In: *Distributed Applications and Interoperable Systems - 15th IFIP WG 6.1 International Conference,* **DAIS**. 2015.

[30] M. C. Rinard, H. Hoffmann, S. Misailovic, and S. Sidiroglou. "Patterns and statistical analysis for understanding reduced resource computing". In: *Proceedings of the 25th Annual ACM SIGPLAN Conference on Object-Oriented Programming, Systems, Languages, and Applications,* **OOPSLA**. 2010.

[31] M. H. Santriaji and H. Hoffmann. "GRAPE: Minimizing Energy for Interactive GPU Applications". In: *Submission*. 2015.

[32] Q. Shi, H. Hoffmann, and O. Khan. "A HW-SW Multicore Architecture to Tradeoff Program Accuracy and Resilience Overheads". In: **IEEE Computer Architecture Letters** (to appear).

[33] S. Sidiroglou-Douskos, S. Misailovic, H. Hoffmann, and M. C. Rinard. "Managing performance vs. accuracy trade-offs with loop perforation". In: *SIGSOFT/FSE'11 19th ACM SIGSOFT Symposium on the Foundations of Software Engineering* **(FSE-19)** *and ESEC'11: 13rd European Software Engineering Conference (***ESEC-13***), Szeged, Hungary, September 5-9, 2011*. 2011.

[34] Y. Sinangil, S. M. Neuman, M. E. Sinangil, N. Ickes, G. Bezerra, E. Lau, J. E. Miller, H. C. Hoffmann, S. Devadas, and A. P. Chandraksan. "A self-aware processor SoC using energy monitors integrated into power converters for self-adaptation". In: **VLSI** *Circuits Digest of Technical Papers, 2014 Symposium on*. IEEE. 2014.

[35] V. Strumpen, H. Hoffmann, and A. Agarwal. "Stream Algorithms and Architecture". In: **J. Instruction-Level Parallelism** 6 (2004).

[36] M. B. Taylor, J. S. Kim, J. E. Miller, D. Wentzlaff, F. Ghodrat, B. Greenwald, H. Hoffmann, P. Johnson, J. W. Lee, W. Lee, A. Ma, A. Saraf, M. Seneski, N. Shnidman, V. Strumpen, M. Frank, S. P. Amarasinghe, and A. Agarwal. "The Raw Microprocessor: A Computational Fabric for Software Circuits and General-Purpose Programs". In: **IEEE Micro** 22.2 (2002).

[37] M. B. Taylor, W. Lee, J. E. Miller, D. Wentzlaff, I. Bratt, B. Greenwald, H. Hoffmann, P. Johnson, J. S. Kim, J. Psota, A. Saraf, N. Shnidman, V. Strumpen, M. Frank, S. P. Amarasinghe, and A. Agarwal. "Evaluation of the Raw Microprocessor: An Exposed-Wire-Delay Architecture for ILP and Streams". In: *31st International Symposium on Computer Architecture* **ISCA**. 2004.

[38] D. Wentzlaff, P. Griffin, H. Hoffmann, L. Bao, B. Edwards, C. Ramey, M. Mattina, C. Miao, J. F. B. III, and A. Agarwal. "On-Chip Interconnection Architecture of the Tile Processor". In: **IEEE Micro** 27.5 (2007).

[39] H. Zhang and H. Hoffmann. "Maximizing Performance Under a Power Cap: A Comparison of Hardware, Software, and Hybrid Techniques". In: *Proceedings of the Twenty-first International Conference on Architectural Support for Programming Languages and Operating Systems,* **ASPLOS** *(to appear)*. 2016.

[40] Y. Zhou, H. Hoffmann, and D. Wentzlaff. "CASH: Supporting IaaS Customers with a Sub-core Configurable Architecture". In: *Submission*. 2015.