Efficient Computing in the Post-Moore and BigData Era

Research Statement: Swagath Venkataramani

Online Copy. *https://engineering.purdue.edu/people/swagath.venkataramani.1/portfolio/swagath-research.pdf*

1 Introduction

Semiconductor technology scaling, together with innovations at various levels of the computing stack, have enabled decades of continuous improvements in the capabilities of computing platforms and formed the basis for growth in the semiconductor and computing industries. However, in recent years, the traditional benefits due to technology scaling have diminished with the end of Dennard scaling and the well-known challenges associated with multi-core scaling. On the other hand, top-down trends, such as the explosion in various forms of digital data, have led to the emergence of new application domains *viz.* recognition, data mining and analytics, computer vision, search *etc.*, that will pose unprecedented demand for computing capability, from mobile devices to the cloud. The confluence of these key trends has created an *efficiency gap*, leading to an urgent need for new sources of efficiency across the computing stack. My research attempts to broadly address this important problem by pursuing three distinct directions, as summarized in Figure 1.

- **Approximate computing**, which leverages the forgiving nature of emerging workloads, *i.e.*, their ability to tolerate many of the underlying computations being executed in an approximate manner, to greatly improve computing efficiency.
- **Spintronic logic and memory design**, in which my research investigates a promising post-CMOS technology that utilizes electron spin to represent and process information.
- **Design specialization using programmable accelerators**, which develops heterogenous many-core architectures that combine the right level of programmability and efficiency in executing a class of emerging workloads.

Through the various research projects shown in Figure 1, I have been exposed to and developed techniques at different levels of abstraction, spanning circuits, architecture, software and algorithms. In the rest of the statement, I will highlight my research contributions in each of these directions and outline possible thrusts for future research.

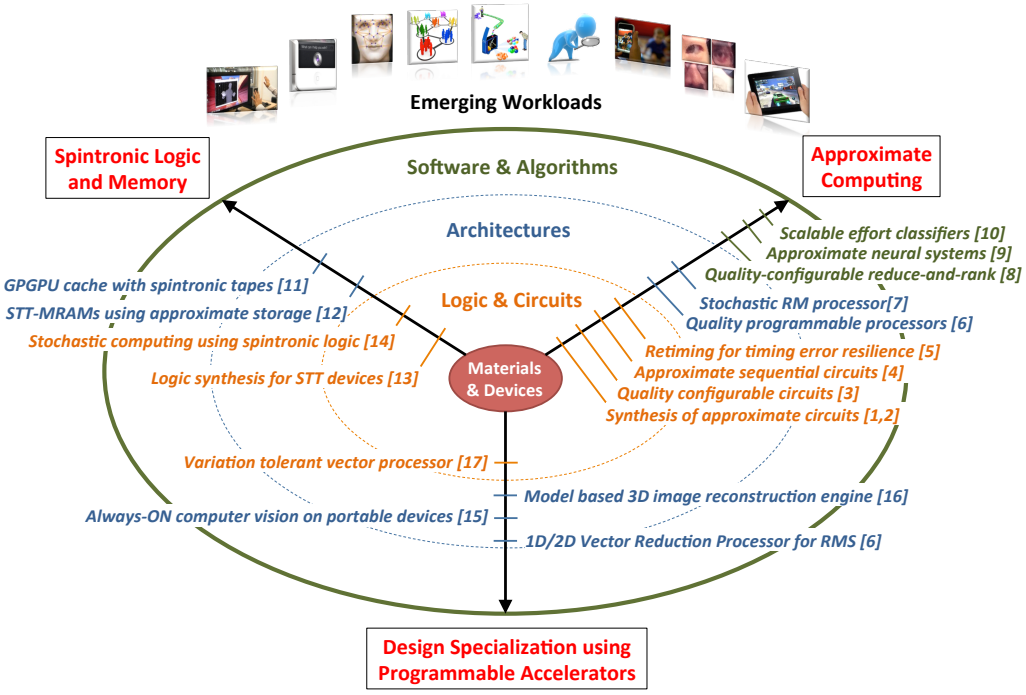

Figure 1: Overview of broad research directions and contributions

2 Approximate Computing: Computing Efficiently with Good-enough Results

In the context of many emerging application domains (such as, search, analytics, sensor data processing, recognition, mining, and others), computing is largely not about calculating a precise numerical end result. Rather, they are characterized by whether they produce an acceptable user experience, or results of sufficient quality. As a result, these emerging workloads invariably exhibit significant *intrinsic application resilience*, which is broadly defined as the ability of applications to produce outputs of acceptable quality despite selected computations being executed in an imprecise or approximate manner. *Approximate computing* leverages this forgiving nature to improve the efficiency (energy/performance) of computing platforms by forsaking exact (numerical or Boolean) equivalence in the execution of some of the application's computations, while ensuring that the output quality is acceptable. The key to the efficiency of approximate computing is to obtain a favorable energy (or performance) *vs.* quality trade-off *i.e.,* the benefits in energy is disproportionately large compared to the quality sacrificed in the process.

My research answers in the affirmative the two key questions that are frequently asked of approximate computing: "Is the approach applicable to a broader range of applications and domains?" and "Is it possible to amortize design effort by creating approximate computing platforms that can be re-used across applications?" We develop a holistic and systematic framework for approximate computing, with design techniques at the circuit [1, 2, 3, 4, 5], architecture [6, 7] and software [8, 9, 10] levels that are generic and applicable to any given application. We also develop supporting tools and automation frameworks that enable scalability, and improve designer effort and reuse. The techniques developed at each layer are described below.

Approximate Circuits. Approximate circuits are the basic hardware building blocks of approximate computing platforms. Approximate circuits are highly efficient hardware implementations that realize a slightly modified logic function compared to the original specifications within a specified quality constraint. The broader adoption of approximate circuits requires a systematic methodology to design approximate implementations for any arbitrary circuit. Moreover, it is critical that such a methodology enables the generation of "correct-by-construction" approximate circuits that are guaranteed to satisfy designer-specified quality constraints. Inspired by classical approaches to Boolean optimization in logic synthesis, my research develops 4 synthesis tools *viz.* SALSA [1, 2], SASIMI [3], ASLAN [4] and Relax-and-Retime [5]. A key hallmark of the above approaches is that off-the-shelf synthesis tools and their capabilities can be easily leveraged in the implementations. Over a wide range of arithmetic circuits and data-paths, the circuits synthesized using our tools demonstrate significant benefits in area/energy ($> 5\times$) for acceptable quality loss.

Approximate Computing Architectures. At the architectural level, my research extends approximate computing to the realm of programmable processors by introducing the concept of *Quality Programmable Processors* (QPPs) [6]. A key principle of QPPs is that the notion of quality is explicitly codified in their HW/SW interface *i.e.,* the instruction set. The micro-architecture is designed with hardware mechanisms to understand these ISA-level quality specifications and translate them into energy savings. As a first embodiment of QPPs, my research presents a quality programmable vector processor QUORA, which contains a 3-tiered hierarchy of processing elements. Based on an implementation of QUORA with 289 processing elements in the 45nm technology, energy benefits up to $2.5\times$ are demonstrated across a wide range of applications. Another key direction is to design architectures that inherently represent data approximately. For example, we build a recognition and mining accelerator that uses stochastic number representation, where data is encoded as probabilistic bitstreams [7]. We found it to yield more fine-grained control over output quality compared to conventional binary number representation.

Software and Algorithms for Approximate Computing. At the software level, my research develops algorithmic/code transformations that are functionally approximate, but significantly improve runtime and energy. A key challenge to approximate computing is to identify which computations to approximate and by how much. To address this challenge, my research leverages various domain-specific insights [8, 9, 10]. For example, in the context of large-scale neural networks (deep learning networks), we develop a systematic framework, called AxNN [9], that utilizes backpropagation to identify neurons that contribute less significantly to the network's accuracy, selectively approximates these neurons (*e.g.,* by using lower precision), and incrementally re-trains the network to mitigate the impact of approximations on output quality.

In summary, through a suite of techniques spanning circuits to software, my research explores approximate computing in a holistic and systematic context, easing the path to mainstream adoption.

3 Computing with Spintronics: Circuits and Architectures

The second direction to bridge the gap in computing efficiency is to evaluate the prospects and challenges in designing computer systems using post-CMOS device technologies under active exploration. Spintronic devices, which manipulate the spin orientation of electrons in a ferro-magnetic material to represent and process data, have demonstrated great potential. However, spintronic devices are neither universal nor drop-in replacements. This creates the need for new designs at the circuit and architecture levels that exploit the strengths of spintronic devices and mask their weaknesses. Towards this objective, my research in spintronics investigates on-chip memory architectures [11, 12] using two spintronic devices—Domain Wall Memory (DWM) and STT-MRAM. On the compute front, my research explores systematic methodologies [13, 14] to design arbitrary circuits using a spintronic logic style—All-Spin Logic (ASL).

Spintronic Memory Architectures. Spintronic devices posses several characteristics such as non-volatility, near-zero leakage and high density, all of which favor memory design. However, their energy efficiency is limited by read and write operations. Also, some spintronic technologies, such as Domain Wall Memories (DWMs), suffer from high access latencies. To address these limitations, my research explores suitable data organization, management policies and other architectural optimizations. We built a spintronic GPGPU cache hierarchy, called STAG [11], that demonstrated 12.1% and 3.3× improvement in performance and energy over SRAM. Another new direction to improve energy efficiency is through approximate storage [12]. We design a quality-configurable memory array in which read and write operations to each word in the memory can be performed at various predefined levels of quality and energy at runtime. We integrate the quality-configurable array in the memory hierarchy of a vector processor, and with supporting ISA enhancements we demonstrate over 40% memory energy improvement with negligible quality loss.

Spintronic Circuit and Logic Design. All Spin Logic (ASL) is a recently proposed logic style that utilizes spintronic devices to realize Boolean logic circuits. My research developed a logic synthesis framework for ASL, leveraging its unique characteristics, to determine its viability [13]. Our exploration across a broad range of circuits (random logic, DSP data paths and Leon SPARC3 core) revealed that ASL shows promise for specific class of low-power/frequency applications (*e.g.,* biomedical applications, internet of things *etc.*) and significant improvement in the device switching times and longer spin diffusion length of spin channels are critical for it to emerge competitive to CMOS. While materials and device structures are under active investigation to address these shortcomings, my research took a complementary approach of investigating alternative computing models and application domains that match the inherent characteristics of spintronic devices [14]. To this end, my research identified the synergy between stochastic computing and spintronic devices. Our approach, SPINTASTIC, resulted in 4.2× improvement in energy over CMOS stochastic implementations and 2.6× over CMOS binary circuits.

Thus my research efforts in spintronic logic and memory demonstrate that, with suitable circuits and architectures, spin devices hold significant promise in the design of future computing platforms.

4 Programmable Accelerators for Emerging Workloads

In recent years, both hardware and software designers have embraced the notion of design specialization. Programmable accelerators such as graphics processing units and image/video co-processors are commonplace across the spectrum of computing platforms. A key tenet in the design of programmable accelerators is the tradeoff between efficiency and flexibility. For many applications, there exists a significant difference (2-3 orders of magnitude) in efficiency between their application-specific hardware and general-purpose implementations. Therefore, the architecture and memory hierarchy of programmable accelerators should be designed such that they retain a large part of the efficiency of algorithm-specific accelerators, while still providing significant programmability. My research has focused on developing such platforms in the context of emerging workloads such as recognition, mining and synthesis [6, 15, 16, 17].

Towards this goal, we analyzed a wide range of applications by hierarchically breaking down their computations from algorithms all the way down to individual scalar operations. We identified that these applications typically operate on data streams and are characterized by 2 levels of reduction operations. Based on these insights, we developed a 1D/2D vector reduction processor architecture [6] in the 45nm technology that yields 50× speedup compared to a well-optimized general purpose processor implementation. Design specialization at the architecture level can help mitigate the impact of process variations, which is inherently a bottom-up phenomenon. Leveraging the properties unique to vector reductions, we enhanced the

architecture and ISA of the 1D/2D vector processor to design a variation-tolerant version that achieved 32% improvement in energy over a traditional guardband based design. Another avenue to lucratively employ accelerators is in portable devices (*e.g.* wearable cameras, Google glass *etc.*) that are always-ON and interact continuously with the cloud. In such systems, the bulk of the energy is expended in communicating the data they sense to the cloud for advanced information processing. My research developed a new system design, SAPPHIRE [15], where a low-power programmable accelerator was used to identify and filter uninteresting data on the device, which resulted in over 2× improvement in battery life.

5 Future Research Thrusts

New sources of processing efficiency from different layers of the computing stack are critical in the face of diminishing technology benefits and emerging compute-intensive workloads; this gap forms the central theme of my future research. My core expertise lies in circuits, logic and architectures, and through various research projects and collaborations, I have a good understanding of devices, software and algorithms. I will now outline few potential directions for future research.

Approximate memory and I/O subsystems. A large majority of research in approximate computing has focussed on reducing the computation energy expended in the processing cores. However, the memory and I/O subsystems also constitute a significant fraction of the overall system energy, and their proportion is expected to grow at further scaled technologies. The principles of approximate computing can be applied to benefit energy in their context. Some preliminary research ideas in this direction include:

- *Quality-encoded data transfer:* We will develop new re-configurable bus coding schemes that can encode data with different levels of information loss (and commensurate data-transfer energy) depending on explicit quality requirements. These quality constraints are derived based on the significance of the data in the context of the application and the operation that will be subsequently performed on it.
- *Quality-driven data sensing*: In many emerging applications, not all data sensed ends up being useful to the application. My research will develop techniques to sense and process data approximately near the sensors to gauge its relevance and utilize it to drive the quality of sensing. Such techniques will have significant impact in the context of distributed applications that continuously interact between multiple mobile/embedded end-points and the cloud.
- *Quality-aware data organization and access*: Most of the data created and stored today are unstructured. We will develop new approaches to re-organize data in memory at runtime, based on the feedback about its significance from the application. This renders data querying and access more efficient.

In summary, the techniques developed by my research will enable us to reap the benefits of approximate computing holistically at the system level.

Quality specification, translation and verification. Intrinsic application resilience does not mean that any result is acceptable. Therefore, it is important to have a clear definition of what constitutes acceptable quality of results, and methods to ensure that it is maintained when approximate computing techniques are used. Also, given an application level quality requirement, translating them to approximations at individual computations is an open challenge. This can be achieved in several ways. Profiling tools and auto-tuning frameworks with in-built approximation models can be developed. Another approach is to design quality configurable versions of common programming templates and libraries, which designers can directly utilize in their implementations. In addition, several machine learning applications provide a unique opportunity. These applications are associated with a training phase, in which parameters of their algorithm are determined. Since approximations in computations are just another source of error, training can potentially heal their impact by suitably adapting the parameters. Thus interleaving approximations with training can yield a superior energy *v.s.* quality trade-off. My research will systematically explore and evaluate these ideas.

Approximate general purpose processors. The efficacy of approximate computing varies widely with the implementation context. For example, approximating algorithm-specific accelerators yield the most benefit as they have the least control overheads. At the other end of the spectrum, general purpose processors are challenging to approximate as control front-ends, such as instruction fetch and decode, inherent to any programmable processor, have to be performed in an accurate manner. In addition, the absolute number of control operations are also larger in a general purpose implementations. Therefore, approximate computing in their context should transcend beyond conventional numerical value-based approximations and explore

how sequences of operations can be approximated together. ISA extensions to express such sequences need to be developed. Also, the approximations should target real bottlenecks to performance/energy to yield disproportionate benefits. This requires utilizing the dynamic information available during program execution. Thus, general purpose processors require a rethink of how approximate computing is realized and present interesting research opportunities.

Computing models for post-CMOS devices. As post-CMOS devices and material structures evolve, it is important to evaluate them at the system-level to understand their strengths and shortcomings, and also provide feedback to device and material experts on the aspects they should target. While research efforts have realized switches and Boolean logic with these devices, another approach is to develop alternate computing models that leverage the intrinsic computations that occur physically within the device and map applications using those as primitives. For example, spintronic devices inherently accumulate spin currents in the device channel, and are therefore efficient in performing weighted addition. My research will identify and exploit such properties and develop new computation models for spintronics and other promising post-CMOS devices.

Input adaptive system design. For many emerging applications, such as recognition, search *etc.*, not all inputs require the same computational effort. Intuitively, in natural language processing, understanding simpler words, sentences and native accents should be easier than complex semantic formations. Similarly, compressing a picture that contains just the blue sky should take less effort than the one that contains a busy street. However, today's algorithms and systems to realize these applications expend equal (and worst case) effort on all of them, leading to significant inefficiency. My research will develop input adaptive systems that expends effort commensurate with its difficulty.

In summary, the efficiency gap is a grand challenge in computing [18], which will provide a fertile ground for innovations at various levels of the computing stack. My research will endeavor to address this challenge by exploring new sources of computing efficiency.

References

[1] S. Venkataramani, A. Sabne, V. Kozhikkottu, K. Roy, and A. Raghunathan. SALSA: Systematic Logic Synthesis of Approximate Circuits. In *Proceedings of the 49th Annual Design Automation Conference*, pages 796–801, 2012.

[2] S. Venkataramani, A. Sabne, V. Kozhikkottu, K. Roy, and A. Raghunathan. Logic Synthesis of Approximate Circuits. In *IEEE Transactions on Computer Aided Systems and Design*, 2015.

[3] S. Venkataramani, K. Roy, and A. Raghunathan. Substitute-and-simplify: A unified design paradigm for approximate and quality configurable circuits. In *Design, Automation Test in Europe Conference Exhibition (DATE), 2013*, pages 1367–1372, March 2013.

[4] A. Ranjan, A. Raha, S. Venkataramani, K. Roy, and A. Raghunathan. ASLAN: Synthesis of Approximate Sequential Circuits. In *Proceedings of the Conference on Design, Automation & Test in Europe*, DATE '14, pages 364:1–364:6, 2014.

[5] S.G. Ramasubramanian, S. Venkataramani, A. Parandhaman, and A. Raghunathan. Relax-and-Retime: A methodology for energy-efficient recovery based design. In *Design Automation Conference (DAC), 2013 50th ACM / EDAC / IEEE*, pages 1–6, May 2013.

[6] S. Venkataramani, V. K. Chippa, S. T. Chakradhar, K. Roy, and A. Raghunathan. Quality Programmable Vector Processors for Approximate Computing. In *Proceedings of the 46th Annual IEEE/ACM International Symposium on Microarchitecture*, MICRO-46, pages 1–12, 2013.

[7] V. K. Chippa, S. Venkataramani, K. Roy, and A. Raghunathan. StoRM: A Stochastic Recognition and Mining Processor. In *Proceedings of the 2014 International Symposium on Low Power Electronics and Design*, ISLPED '14, pages 39–44, 2014.

[8] A. Raha, S. Venkataramani, V. Raghunathan, and A. Raghunathan. Quality Configurable Reduce-and-Rank for Energy Efficient Approximate Computing. In *Design, Automation Test in Europe Conference Exhibition (DATE), 2015*, March 2015.

[9] S. Venkataramani, A. Ranjan, K. Roy, and A. Raghunathan. AxNN: Energy-efficient Neuromorphic Systems Using Approximate Computing. In *Proceedings of the 2014 International Symposium on Low Power Electronics and Design*, ISLPED '14, pages 27–32, 2014.

[10] S. Venkataramani, J. Liu, A. Raghunathan, and M. Shoaib. Scalable-effort Classifiers for Energy Efficient Machine Learning. In *Proceedings of the 52th Annual Design Automation Conference*, 2015.

[11] R. Venkatesan, S. G. Ramasubramanian, S. Venkataramani, K. Roy, and A. Raghunathan. STAG: Spintronic-tape Architecture for GPGPU Cache Hierarchies. In *Proceeding of the 41st Annual International Symposium on Computer Architecture*, ISCA '14, pages 253–264, 2014.

[12] A. Ranjan, S. Venkataramani, X. Fong, K. Roy, and A. Raghunathan. Approximate Storage for Energy Efficient Spintronic Memories. In *Proceedings of the 52th Annual Design Automation Conference*, 2015.

[13] Z. Pajouhi, S. Venkataramani, K. Yogendra, A. Raghunathan, and K. Roy. Exploring Spin-Transfer-Torque Devices for Logic Applications. In *IEEE Transactions on Computer Aided Systems and Design*, 2015.

[14] R. Venkatesan, S. Venkataramani, X. Fong, K. Roy, and A. Raghunathan. SPINTASTIC: Spin-based Energy Efficient Stochastic Logic. In *Design, Automation Test in Europe Conference Exhibition (DATE), 2015*, March 2015.

[15] S. Venkataramani, V. Bahl, X.-S. Hua, J. Liu, J. Li, M. Phillipose, B. Priyantha, and M. Shoaib. SAPPHIRE: An Always-ON Context-aware Computer Vision System for Portable Devices. In *Design, Automation Test in Europe Conference Exhibition (DATE), 2015*, March 2015.

[16] J. Liu, S. Venkataramani, S. Venkatakrishnan, C. Bouman, and A. Raghunathan. EMBIRA: An Efficient Model-based Iterative Image Reconstruction Accelerator. In *Journal under preparation*, 2015.

[17] V. Kozhikkottu, S. Venkataramani, S. Dey, and A. Raghunathan. Variation Tolerant Design of a Vector Processor for Recognition, Mining and Synthesis. In *Proceedings of the 2014 International Symposium on Low Power Electronics and Design*, ISLPED '14, pages 239–244, 2014.

[18] M. Hill and C. Kozyrakis. Advancing Computer Systems without Technology Progress. In *Outbrief of DARPA/ISAT Workshop (http://www.cs.wisc.edu/~markhill/papers/isat2012_ACSWTP.pdf)*, March 2012.