Secure and Reliable Configurable and Reconfigurable Computing for Machine Learning Applications

# Hena Naaz

[henanaazkhan24@gmail.com](mailto:henanaazkhan24@gmail.com)

ABSTRACT

## Computing is bottlenecked by data. Large amounts of application data overwhelm storage capability, communication capability, and computation capability of the modern machines we design today. As a result, many key applications' performance, efficiency and scalability are bottlenecked by data movement. In this invited special session talk, we describe three major shortcomings of modern architectures in terms of 1) dealing with data, 2) taking advantage of the vast amounts of data, and 3) exploiting different semantic properties of application data. We argue that an intelligent architecture should be designed to handle data well. We give several examples for how to exploit each of these principles to design a much more efficient and high-performance computing system. We especially discuss recent research that aims to fundamentally reduce memory latency and energy, and practically enable computation close to data, with at least two promising novel directions: 1) *processing using memory*, which exploits analog operational properties of memory chips to perform massively-parallel operations in memory, with low-cost changes, 2) *processing near memory*, which we believe are key to efficiency, performance, and sustainability. We conclude with some guiding principles for future computing architecture and system designs. This accompanying short paper provides a summary of the invited talk and points the reader to further work that may be beneficial to examine.

1. Introduction

## Existing computing systems process increasingly large amounts of data. Data is key for many modern (and likely even more future) workloads and systems. Important workloads (e.g., machine learning, artificial intelligence, genome analysis, graph analytics, databases, video analytics, online collaboration), whether they execute on cloud servers or mobile systems are all data intensive; they require efficient processing of large amounts of data. Today, we can generate more data than we can process, as exemplified by the rapid increase in the data obtained in astronomy observations and genome sequencing [1].

Unfortunately, the way they are designed, modern computers are not efficient at dealing with large amounts of data: large amounts of application data greatly overwhelm the storage capability, the communication capability, and the computation capability of the modern machines we design today. As such, data becomes a large performance and energy bottleneck, and it greatly impacts system robustness and security as well. As a prime example, we provide evidence that the potential for new genome sequencing technologies, such as nanopore sequencing [2, 113], is greatly limited by how fast and how efficiently we can process the huge amounts of genomic data the underlying technology can provide us with [3, 83, 113, 119, 143]. A similar observation can also be made

for video analytics [163, 7] and machine learning [198-199, 7]. The processor-centric design paradigm (and the resulting processor-centric execution model) of modern computing systems is one prime cause of why data overwhelms modern machines [4, 5, 120]. With this paradigm, there is a clear dichotomy between processing and memory/storage: data has to be brought from storage and memory units to computation units (e.g., general-purpose processors or special-purpose accelerators), which are far away from the memory/storage units, before any processing can be done on the data. The dichotomy exists at the macro-scale (e.g., across the internet) as well as the micro-scale (e.g., within a single compute node, or even within a single CPU processing core). This processor- memory dichotomy leads to large amounts of data movement across the entire computing system, degrading performance and expending large amounts of energy. For example, a recent work [7] shows that more than 60% of the entire mobile system energy is spent on data movement across the memory hierarchy when executing four major commonly used consumer workloads, including machine learning inference, video processing and playback, and web browsing. Similarly, due to the current processor-centric design paradigm, a large fraction of the system resources is dedicated to units that store and move data (i.e., to serve the computation units), and actual computation units constitute only ~5% of an entire processing node [8] – yet, even then, data access is still a major bottleneck due to the large latency and energy costs of accessing large

amounts of data.

1. Scope of Hardware

## Our starting axiom for an intelligent architecture is that it should handle (i.e., store, access, and process) data well. But what does it mean for an architecture to handle data well? We posit (and later demonstrate with examples) that the answer lies in satisfying three major desirable properties (or principles): 1) data-centric, 2) data-driven, and 3) data-aware.

First, the system should ensure that data does not overwhelm its components. Doing so requires effort in intelligent algorithms, intelligent architectures and intelligent whole system designs that are co-optimized cross-layer (i.e., optimizations spanning across algorithms-architectures- devices), in a manner that puts data and its processing at the center of the design, minimizing data movement and maximizing the efficiency with which data is handled, i.e., stored, accessed, and processed (e.g., as exemplified in [4-38, 120]). We call this first principle *data-centric architectures*.

Second, an intelligent architecture takes advantage of the vast amounts of data and metadata that flow through the system, to continuously improve its decision making, by bettering its policies.

1. ASICs and FPGAs

## Based on our qualitative and quantitative analyses, we find that existing computing architectures greatly fall short of handling data well. They violate all of the three major desirable principles. We analyze each briefly next.

First, modern architectures are poor at dealing with data: they are designed to mainly store and move data, as opposed to compute on the data. Most system resources serve the processor (and accelerators) without being capable of processing data. Doing so would eliminate the huge data access bottleneck of processor-centric systems, thereby improving performance, reducing energy consumption, alleviating off-chip bandwidth requirements (and hence area and cost), likely reducing system and hardware design complexity, as well as opening up new opportunities for improving system security and reliability by handling data more locally in or near where it resides.

Second, modern architectures are poor at taking advantage of vast amounts of data (and metadata) available to them during online operation and over time. Because the policy it follows is rigid and hardcoded by a human. This is clearly not intelligent: for example, as humans, we have the capability to learn from the past and adapt our actions accordingly to not repeat the same mistakes as in the past or to choose the best policy/actions that we believe will provide the highest benefits in the future. Enabling similar intelligence and far-sightedness in controller and system policies in an architecture is necessary for obtaining good performance and efficiency (as well as better reliability, security, and perhaps other metrics) under a variety of system conditions and workloads.

Third, modern architectures are poor at knowing and exploiting different properties of application and system data. If the characteristics of the data to be accessed or manipulated were known, the decisions taken could be very different: for example, if we knew the relative compressibility of different types of data, e.g., different data types or different objects [5], different components in the entire system could be designed in a manner that adaptively scales their capability to match the compressibility of different data elements, in order to maximize both performance and efficiency. Modifying the architecture and its interface to become richer and more expressive, and to include rich and accurate information on various properties of data that is to be processed, is therefore critical to customizing the architecture to the characteristics of the data and, thus, enabling intelligent adaptation of system policies to data characteristics.

1. GPGPU and Accelerators

## A major chunk of our invited talk describes in detail the characteristics of an intelligent computing architecture, by concrete examples and their empirical evaluation. This short paper does not go into detail but provides a brief overview with references to other works that exemplify such architectures. Multiple detailed versions of this talk can be found online [82, 139-142]. We also refer the reader to recent detailed survey and overview papers we have written on the topic [120, 4].

*DNN*

A data-centric architecture has at least four major characteristics. First, it enables processing capability in or near where data resides (i.e., in or near memory structures), as described in detail in [4-6, 8, 12] and exemplified by [7-

4, 9]. Second, it provides low-latency and low-energy access to data, as exemplified by [11-13, 15-18, 21, 23, 31-33, 84-86]. Third, it enables low-cost data storage and processing (i.e., high-capacity memory at low cost, via techniques like new memory technologies, hybrid memory systems and/or compressed memory systems), as exemplified by [22, 87-96, 74, 76, 78, 107, 116]. Fourth, it provides mechanisms for intelligent data management (with intelligent controllers handling robustness, security, cost, etc.), as described in detail in [97-103, 116, 120] and exemplified by, e.g., [104-106, 116, 120, 179-190].

Both PUM and PNM approaches can greatly accelerate real applications, including database systems, graph analytics, machine learning, genome analysis, GPU workloads, pointer- chasing-intensive workloads, data analytics, climate modeling, etc. Recent results show up to approximately two orders of magnitude improvement in energy and performance over conventional processor-centric systems.

*CNN*

A data-aware architecture understands what it can do with and to each piece of data (and associated computations on data) and uses this information about data characteristics to maximize system efficiency and performance. In other words, it customizes itself (i.e., its policies and mechanisms) to the characteristics of the data and computations it is dealing with. Such an architecture requires knowledge of various characteristics of different data elements and structures as well as computations.

CONCLUSION

## This paper aims at delivering the prospective research area in the field of hardware design for data intensive Machine Learning or Artificial Intelligence Hardware Systems. I am thankful to all the researchers in academic and industrial domain who have worked directly or indirectly to contribute towards increasing the computation capability of the systems with ever-increasing data-load in the age of data, and who have contributed to the various works we describe in this paper.

REFERENCES

1. Hyesoon Kim et al., “Performance Analysis and Tuning for General Purpose Graphics Processing Units (GPGPU)”, IEEE
2. Z. D. Stevens et al., “Big data: astronomical or genomical?”, PLoS Biology, 2015.
3. O. Mutlu, “Accelerating Genome Analysis: A Primer on an Ongoing Journey”, Keynote Talk at HiCOMB-17, 2018.
4. S. Ghose et al., “Processing-in-Memory: A Workload-Driven Perspective”, IBM JRD 2019.
5. O. Mutlu et al., “Processing Data Where It Makes Sense: Enabling In- Memory Computation”, MICPRO, 2019.
6. A. Boroumand et al., “Google Workloads for Consumer Devices: Mitigating Data Movement Bottlenecks”, ASPLOS 2018.
7. O. Mutlu, “Enabling Computation with Minimal Data Movement: Changing the Computing Paradigm for High Efficiency", Design Automation Summer School Lecture, DAC 2019.
8. J. Ahn et al., “A Scalable Processing-in-Memory Accelerator for Parallel Graph Processing”, ISCA 2015.
9. V. Seshadri et al., “Ambit: In-Memory Accelerator for Bulk Bitwise Operations Using Commodity DRAM Technology”, MICRO 2017.
10. H. Luo et al., “CLR-DRAM: A Low-Cost DRAM Architecture Enabling Dynamic Capacity-Latency Trade-Off”, ISCA 2020.
11. K. Hsieh et al., “Accelerating Pointer Chasing in 3D-Stacked Memory: Challenges, Mechanisms, Evaluation”, ICCD 2016.