Real-Time Analytics as the Killer Application for Processing-In-Memory

Zvika Guz, Manu Awasthi, Vijay Balakrishnan, Mrinmoy Ghosh, Anahita Shayesteh, Tameesh Suri
Samsung Semiconductor, Inc.
{zvika.guz, manu.awasthi, vijay.bala, mrinmoy.g, anahita.sh, tameesh.suri}@ssi.samsung.com

***Abstract*— While Processing-In-Memory (PIM) has been widely researched for the last two decades, it was never truly adopted by the industry and remains mostly within the academic research realm. This is mainly because (1) in-memory compute engines were too slow, and (2) a real-world application that could really benefit from PIM was never identified. In recent years, the first argument became untenable, but the community still struggles to identify a poster-child application that refutes the second argument. In this position paper, we argue that real-time analytics is the killer application the community has been searching for. We show that several inherent characteristics of real-time analytics fit perfectly within the PIM paradigm, and identify Redis – a popular in-memory key-value framework – as a convenient platform to showcase PIM in this domain. Finally, we call for a targeted hardware-software co-design effort towards a PIM acceleration architecture for real-time analytics.**

I. Introduction

Processing-in-Memory is a decades-old concept of offloading computation from the CPU to main-memory, typically DRAM. By moving computation closer to data, one can eliminate costly data movement throughout the system, including multiple memories and cache tiers. Eliminating data movement is particularly important because: (1) data movement consumes multiple orders of magnitude more energy than the actual computation [1]; and (2) data movement increases the time to complete a dependent instruction(s) by orders of magnitude [2].

PIM seems particularly appealing in datacenter contexts (both HPC and hyperscale) because (1) energy efficiency and performance/watt are primary design constraints in these deployments, and (2) frequent data movement across the system is a major contributor to overall energy waste. Moreover, energy constraints discourage using the most powerful, power-hungry processors. Thus, the speed gap between PIM compute engines and CPUs narrows. Lastly, contemporary servers support abundant DRAM capacity, allowing for increasingly larger chunks of applications working-sets to fit within (and be served from) memory.

While PIM has been studied extensively in the past [3], it never gained much industry traction. The major hardware inhibitor proved to be the difficulty of implementing *capable* in-DRAM computation engines. Recently, maturing 3D-stacking technologies have eliminated this problem, allowing capable computation engines to be implemented in a logic layer within the DRAM stack [4, 5].

The second, more pressing barrier to PIM adoption has been the lack of proven, real-world applications that can benefit from PIM acceleration. While several toy applications and small-scale workloads exhibit potential [6, 7], a widely-used, full-fledged use-case that can serve as a poster-child application for PIM has not yet been identified. Without such a use-case – one that promises dramatic performance improvements over the baseline server design – it would be very difficult to increase the adoption of PIM outside the realm of few niche markets. In this position paper we argue that real-time analytics has all the necessary traits to serve as the PIM poster-child application.

"Big-data analytics" is the process of analyzing very large datasets to discover patterns, extract knowledge, and provide useful information. Unlike offline analytics [8] (batch processing), real-time analytics (latency sensitive analytics) limits the time available for request completion. Instead of multiple hours or days, real-time analytics tasks must complete within milliseconds to few minutes. To meet these requirements, real-time analytics applications store the *entire* or a *large majority* of the dataset in memory. While big-data analytics has traditionally been done via offline batch processing, applications are now increasingly shifting to real-time frameworks. Indeed, servicing real-time analytics requests is expected to account for a significant fraction of future datacenters cycles, and the business is estimated to reach over $13 billion by 2018 [9].

In Section II, we describe several inherent characteristics of real-time analytics frameworks that make them great candidates for PIM acceleration. We also highlight common constructs of real-time analytics applications that can greatly benefit from PIM. We argue that these constructs should be the primary target of any new PIM architecture that aims to have industry adoption. To this end, in Section III, we provide a conceptual PIM acceleration architecture that is geared towards real-time analytics. We also identify Redis – a popular in-memory framework – as a convenient platform for exemplifying the performance gains achievable via PIM offloading. Finally, Section IV concludes the paper with a call for a targeted hardware-software co-design effort for this specific domain.

II. Real-Time Analytics and PIM

Traditionally, "real-time analytics" referred to the task of detecting events and trends as they happen [10]. With the

increase in practicality of real-time analytics, there has been a surge in the number of software frameworks enabling different variants of this form of computing [11–14]. In-fact, real-time analytics has become an umbrella term for many types of big-data, latency-sensitive analytics tasks, where time constraints vary from several milliseconds to a few hours. The common thread among all these systems is a computation time constraint. Hence, they all rely on DRAM (rather than traditional storage media) for storing and accessing data.

Given the constraints on query completion times, a large number of real-time analytics applications have evolved to encompass very similar *core* constructs. In this section, we list several inherent characteristics of real-time analytics and introduce basic software constructs that make this domain a particular good candidate for PIM.

Market share and economics: The increase in popularity of real-time analytics and the large Total Available Market (TAM) [9] projections fulfill the first prerequisite of a killer-application – there are significant monetary gains to be had from its acceleration. This business argument is important, because historically, the large investment required to enable PIM made it economically unjustifiable when no market could clearly benefit from and pay for it.

In-memory datasets: All real-time analytics frameworks are presently designed to satisfy the vast majority of data accesses from memory. This is done either by saving the *entire* dataset in memory (e.g., Redis [11]), or by explicitly pinning data to memory and querying it (e.g., Spark [12]). Moreover, applications typically stream substantial amounts of data with low locality, making on-chip caches ineffective. Hence, the common case for data access is an access to DRAM, which presents an opportunity for PIM acceleration.

Abundant parallelism: The performance advantage of in-memory computation stems from the ability to process large amounts of data in parallel. However, to capitalize on this, there needs to be enough data parallelism within the workload. This type of parallelism is abundant in real-time analytics. Applications typically churn through large sets of independent items (e.g., database entries, attributes, etc.), applying the same computation to all of them (e.g., extracting a specific field from multiple records, applying the same computation sequence to multiple records, etc.). Moreover, individual items can be relatively large (kilobytes to megabytes) and several constructs can be applied in parallel to multiple fields within a specific item (e.g., logical operation over large bitsets can be applied to all bits in parallel, etc.).

Memory copies and data movement: Real-time analytics frameworks digest a constant stream of data at very high rates. In such scenarios, memory copy and memory movement operations are frequent. Data is moved or copied between I/O devices and memory; between OS kernel buffers (e.g., TCP/IP and file I/O processing); and within user-space applications. Although various hardware/software optimizations and DMA engines exist for copying data between memory regions, an in-memory implementation provides superior performance because data movement stays within the DRAM.

Frequent reduction operations: Reduction operations are a common real-time analytics task, because many queries require reducing multiple records or attributes to generate *one* meaningful result. Reduction operations can vary from simple accumulations to complex operations [8], but in all of them, the information flow follows a reduction tree structure. These queries also stream substantial amount of data; huge amounts of data are brought to the processor and are then either filtered out or sparingly used. Reduction trees are therefore excellent candidates for PIM, because PIM can eliminate virtually all redundant data movements. The reduction tree can happen where the data lives – within memory – and only the final result needs to be communicated back to the processor. Several studies have already showed the advantage of in-memory reduction trees [15, 16].

Bitmaps and logical operations: Bitmaps are a commonly-used datatype in data analytics, where they are typically used for storing bit-arrays of users, events, or webpages. Bitmaps facilitate efficient queries such as identifying the number of unique visitors in a given time window [17], they simplify tracking of events or groups of events, and they benefit several data mining algorithms. For example, the *Apriori* algorithm for discovering frequent item sets can be implemented on top of bitmaps [15]. Moreover, complex queries can be implemented through bitwise logical operations between bitmaps. For example, a bitmap representing all users who visited two specific webpages can be extracted by executing a bitwise `AND` between the bitmaps of the two pages.

Logical operations on bitmaps are an excellent fit for PIM because of the inherent parallelism of bitwise operations: they can be executed on the entire vector in parallel. Moreover, many common bitmap operations like *set bit*, *clear bit* and other simple logical operations require trivial computational support and are frequently used in webpage analytics [17]. Given the simplicity of the logic required to support these operations, they can be easily implemented in the logic layer of a 3D stacked DRAM chip [5].

Lastly, reduction operations such as *bitcount* (counting the number of '1' bits in a bitmap) are frequently used in many real-time analytics tasks. For example, finding the number of unique visitors to a specific page translates to a single bitcount operation over the page visitors bitmap. As mentioned before, reduction operations lend themselves well to PIM acceleration.

III. Driving PIM with Real-Time Analytics

Figure 1(a) presents one perspective of a real-time analytics layer in the context of hyperscale datacenter stack.

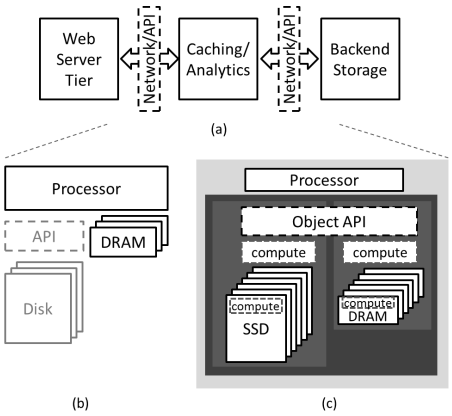

Figure 1. (a) Real-time analytics stack in datacenters (b) Contemporary server architecture (c) NDP architecture for real-time analytics.

While this layer does not use any PIM or Near-Data-Processing (NDP) acceleration today (see Figure 1(b)), we project that future architectures will be heavily based on the NDP paradigm. Figure 1(c) shows a conceptual view of an NDP-based architecture for real-time analytics. We argue that compute engines should be embedded throughout the entire memory and storage hierarchy: local computation will occur where the data resides, while computation across components will be executed in the lowest common ancestor shared by all data locations. Moreover, in order to facilitate this model we believe that storage components will support a unified object API on top of the current read/write semantics. Although we focus on in-DRAM compute in this paper, we believe that the same principles hold for compute acceleration within SSDs. The open question left to be answered is how to drive for adoption of these architectures.

We believe that the first step should be showcasing a PIM accelerator for *one* real-time analytics framework. To this end, we have identified Redis [11] – an in-memory key-value (KV) store – as a good candidate for this task. Redis (REmote DIctionary Server) is an open-source, in-memory framework that provides advanced KV abstraction. Unlike traditional KV systems where keys are of a simple data type (usually strings), Redis supports complex data types: keys can function as hashes, lists, sets, and sorted sets. Furthermore, Redis enables complex atomic operations on these data types, which are all executed by the Redis server. Examples of such operations include enqueuing and dequeuing from a list, inserting a new value with a given score to a sorted set, and creating a union of two sets. Redis' abstraction has proven to be particularly useful for many latency-sensitive tasks, so it presents a real-world, widely-used use-case. Indeed, multiple companies confirmed using Redis as part of their production stack [18].

We have experimented with PIM acceleration for several Redis bitmap operations that often serve as building blocks in real-time web analytics. Bitmaps are a native datatype in Redis; the framework implements multiple bitmap operations, including *setbit* and *clearbit*; bitwise logical operations; finding the first bit with a specified value in a bitmap; and *bitcount*. Initial results indicate two orders-of-magnitude speedups for in-memory *bitcount* over large bitmaps (8KB-4MB), which translate to significant improvements in overall system performance. Due to space limitations we omit detailed results, but we report on these early findings to motivate others to consider the Redis framework as a vehicle for facilitating PIM.

IV. Summary

While a number of technological challenges related to PIM acceleration have been overcome in the last few years [3], the lack of real-world applications that provide significant performance advantages with PIM acceleration has hindered the adoption of this technology. In this paper we made the case for using real-time analytics as the poster-child application for PIM. We have highlighted several characteristics of real-time analytics that lend themselves well for PIM acceleration, and identified Redis as a convenient framework to showcase PIM in this context. We believe that a targeted hardware-software co-design effort towards a PIM acceleration architecture that leverages the Redis API in conjunction with its real-time analytics use-cases, will result in pronounced gains for this important domain.

References

[1] K. Bergman *et al.*, "ExaScale Computing Study: Technology Challenges in Achieving Exascale Systems," 2008.
[2] W. A. Wulf and S. A. McKee, "Hitting the Memory Wall: Implications of the Obvious," *SIGARCH Computer Architecture News*, vol. 23, no. 1, 1995.
[3] R. Balasubramonian *et al.*, "Near-Data Processing: Insights from a MICRO-46 Workshop," *Micro, IEEE*, vol. 34, no. 4, 2014.
[4] JEDEC, "High Bandwdith Memory (HBM) DRAM," http://www.jedec.org/standards-documents/docs/jesd235, 2013.
[5] Micron Technologies, "Hybrid Memory Cube–A Revolution in Memory," http://www.micron.com/products/hybrid-memory-cube, 2014.
[6] M. Gokhale, B. Holmes, and K. Iobst, "Processing in Memory: The Terasys Massively Parallel PIM Array," *Computer*, vol. 28, no. 4, 1995.
[7] Y. Kang *et al.*, "FlexRAM: Toward an Advanced Intelligent Memory System," in *Proceedings of ICCD*, 1999.
[8] J. Dean and S. Ghemawat, "MapReduce: Simplified Data Processing on Large Clusters," in *Proceedings of OSDI*, 2004.
[9] "In-Memory Computing Market worth $13.23 Billion by 2018," http://www.researchandmarkets.com/research/btkq7v/inmemory, 2014.
[10] B. Azvine, Z. Cui, D. D. Nauck, and B. Majeed, "Real time business intelligence for the adaptive enterprise," in *Proceedings of CEC-EEE'06*, 2006.
[11] "Redis," http://redis.io/, 2014.
[12] "Apache Spark– Lightning-fast cluster computing," http://spark.apache.org, 2014.
[13] "Oracle Database In-memory," http://http://www.oracle.com/us/corporate/features/database-in-memory-option/, 2014.
[14] "SAP HANA," http://www.saphana.com, 2014.
[15] Q. Guo *et al.*, "AC-DIMM: Associative Computing with STT-MRAM," in *Proceedings of ISCA'13*, 2013.
[16] S. H. Pugsley *et al.*, "NDC: Analyzing the impact of 3D-stacked memory+logic devices on MapReduce workloads," in *Proceedings of ISPASS 2014*, 2014.
[17] P. W. Farris, N. T. Bendle, P. E. Pfeifer, and D. J. Reibstein, *Marketing Metrics: The Definitive Guide to Measuring Marketing Performance*, 2nd ed., 2010.
[18] "Who's using Redis?" http://redis.io/topics/whos-using-redis, 2014.