A Survey on Reconfigurable Accelerators for Cloud Computing

Christoforos Kachris
Institute of Communication and Computer Systems (ICCS/NTUA)
Athens, Greece

Dimitrios Soudris
National Technical University of Athens (NTUA)
Athens, Greece

Abstract—**Data centers are experiencing an exponential increase in the amount of network traffic that they have to sustain due to cloud computing and several emerging web applications. To face this network load, large data centers are required with thousands of servers interconnected with high bandwidth switches. Current data center, based on general purpose processor, consume excessive power while their utilization is quite low. Hardware accelerators can provide high energy efficiency for many cloud applications but they lack the programming efficiency of processors. In the last few years, there several efforts for the efficient deployment of hardware accelerators in the data centers. This paper presents a thorough survey of the frameworks for the efficient utilization of the FPGAs in the data centers. Furthermore it presents the hardware accelerators that have been implemented for the most widely used cloud computing applications. Furthermore, the paper provides a qualitative categorization and comparison of the proposed schemes based on their main features such as speedup and energy efficiency.**

Index Terms—**reconfigurable computing, hardware accelerator, cloud computing, FPGAs, data centers.**

I. Introduction

Emerging web applications like cloud computing and Big Data Analytics (BDA) have increased significantly the workload on the data centers during the last years. According to the Cisco Global Cloud Index Report [1], the annual global data center IP traffic will reach 10.4 zettabytes (863 exabytes [EB] per month) by the end of 2019, up from 3.4 zettabytes (ZB) per year (287 EB per month) in 2014. That is translated to a compound annual growth rate (CAGR) of 25% from 2014 to 2019. Based on the same report, by 2019, 83% of all data center traffic will come from the cloud and 4 out of 5 data center workloads will be processed in the cloud. The data growth outperforms even Moores law. The data deluge gap has started becoming obvious over the last 5 years [2].

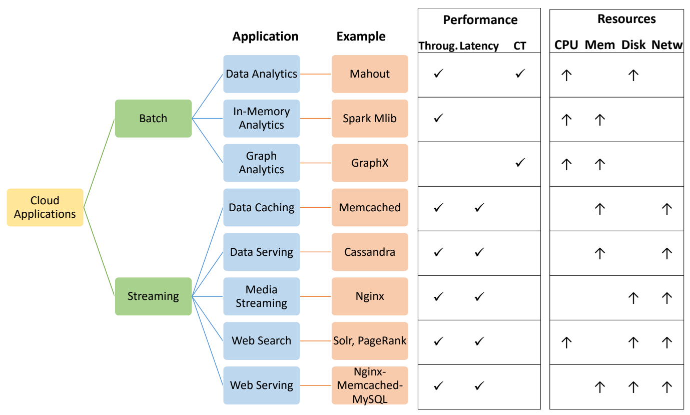
In the early technology nodes, going from one node to the next allowed for a nearly doubling of the transistor frequency, and, by reducing the voltage, power density remained nearly constant. With the end of Dennards scaling, going from one node to the next still increases the density of transistors, but their maximum frequency is roughly the same and the voltage does not decrease accordingly [3]. As a result, the power density increases now with every new technology node. The biggest challenge therefore now consists of reducing the power consumption per mm^2. The failure of Dennard scaling, to which the shift to multicore chips is partially a response, may soon limit multicore scaling just as single-core scaling has been curtailed [4]. This issue has been identified in the literature as the *dark silicon* era in which some of the areas in the chip are kept powered down in order to comply with thermal constraints [5][6].

A solution that can be used to overcome this problem is the use of application-specific accelerators [7]. The use of highly specialized units designed for specific workloads can greatly advance server processors and can also increase significantly the performance of data centers given a fixed power budget. Recent studies have shown that FPGA-based application acceleration can achieve up to 25$\times$ better performance per watt and 50-75$\times$ latency improvement compared to CPU/GPU implementations for data center applications [8].

In the last couple of years there are several reconfigurable architectures that have been proposed for the acceleration of cloud computing applications in data centers. This paper presents a thorough survey of the FPGA-based accelerators for cloud computing that have been recently presented in the research literature. The paper first gives an overview of the cloud computing workload in modern data centers and the characterization of these applications based on widely used data center benchmarks. Section III presents the frameworks that have been presented for the efficient deployment and virtualization of the FPGA-based hardware accelerators. Section IV presents the main hardware accelerators that have been presented for several widely used cloud computing applications like MapReduce, Spark, Memcached, Databases, etc. Finally, Section V presents a quantitative and qualitative comparison of the proposed hardware accelerators in terms of speedup, energy efficiency and power consumption. he main contributions of this paper are the followings:

- Analysis on the cloud computing application and application characterization
- Survey of the frameworks for the efficient deployment and virtualization of hardware accelerators in data centers
- Survey of hardware accelerators for the most widely used cloud computing applications such as MapReduce, Spark, Memcached, etc.
- Classification and quantitative and qualitative comparison of the proposed frameworks

This work has received funding from the European Union Horizon2020 programme under grant agreement No 687628 VINEYARD: Versatile Integrated Accelerator-based Heterogeneous Data Centers. www.vineyard-h2020.eu



Application	Example	Performance			Resources			
		Throug.	Latency	CT	CPU	Mem	Disk	Netw
Data Analytics	Mahout	✓		✓	↑		↑	
In-Memory Analytics	Spark Mlib	✓			↑	↑		
Graph Analytics	GraphX			✓	↑	↑		
Data Caching	Memcached	✓	✓			↑		↑
Data Serving	Cassandra	✓	✓			↑		↑
Media Streaming	Nginx	✓	✓				↑	↑
Web Search	Solr, PageRank	✓	✓		↑		↑	↑
Web Serving	Nginx-Memcached-MySQL	✓	✓			↑	↑	↑

Fig. 1. Categorization of cloud applications, performance metric for each application and typical resource requirements in terms of CPU, Memory, Disk I/O and Network I/O

II. Cloud Application Characterization

In order to design efficient hardware accelerators for the data centers, a clear and in-depth understanding of the of the cloud application is required. In this section we present the type of cloud applications and the main characteristics of these applications. In [9], the *CloudSuite* Benchmark has been introduced that investigates the workload characteristics of cloud applications. In this testbench several applications have been examined. The cloud applications can be divided into two broad categories:

- **Batch processing applications (Offline)**: In this case, the applications process high volumes of data that have been collected and stored in the data centers. Usually there are several complicated processing that needs to be done in the data. The main performance metric in these applications is the throughput and the completion data. In this category belongs several applications like Data Analytics, In-Memory Data Analytics and GraphX Analytics.
- **Streaming processing applications (Online)**: In this case, the applications process high volume of streaming data and usually the processing that needs to be done in these cases is simpler than in the case of batch processing applications. The main performance metric in these applications is the latency (i.e. N-th percentile latency). In this category falls several applications like Data Caching, Data Serving, Media Streaming, Web Search, and Web Serving.

Figure 1 depicts the classification of the cloud application and typical frameworks that are used for each applications. For each application there are different performance metrics. For example for batch applications the most important metric is the throughput in terms of operations per second and the completion time (CT) for a task. The figure shows also if the application is mainly CPU-intensive, memory-intensive, Disk I/O-intensive and/or Network I/O-intensive. Each of these application may have specialized libraries for specific cloud applications.

One of the widely used framework for data analytics is Apache Spark. Spark [10] has been adopted widely in recent years for big data analytics by providing a fault-tolerant, scalable and easy to use in-memory abstraction. IBM has recently presented the *SparkBench* benchmark [11]. Sparkbench covers 4 main categories of applications: machine learning, graph computation, SQL query and streaming applications. The performance of each application depends on several factors such as the number of the worker nodes, the size of the tasks allocated in the workers, etc. However, there are some common features for each category that was examined in the SparkBench. For example, the machine learning workloads have intensive CPU demand. The CPU demand for graph computation applications varies from low to high depending on the workload, while the Streaming and the SQL application have relatively light CPU demand. On the other hand, the graph computation and the streaming applications seems to be mainly memory intensive.

In [12], a detailed study on the characteristics of the Data Analytics workload from CloudSuite benchmark is presented. The method of the analysis that is performed is threefold, covering the system, the applications and the micro-architecture level. The paper reveals that most of the Data Analytics workloads suffer from overhead related to managing the data rather than accessing the data. For example, the hash key index is found to be a key performance limiter. This study verifies as in the previous case that the big data analytics (BDA) applications are mainly CPU-bounded rather than memory or I/O-bounded.

In [13], an analysis has been performed of in-memory data analytics cloud applications. The study shows that the in-memory applications have different characteristics from other data analytics applications like Hadoop and HPC benchmarks. In-memory workloads have good memory bandwidth utilization and high disk I/O requests. Furthermore, Spark workloads have more sequential burst access which translates to high CPU utilization.

In [14], an additional study on the data analytics applications is performed. The papers performs a blocked time analysis for quantifying performance bottlenecks. According to this study, data analytics applications are CPU-bounded and not I/O bounded. Even for some database applications, it is found that queries are more CPU bound than I/O bound. That means that tasks are more likely blocked waiting on computation to complete rather than waiting for disk I/O.

Finally in [15], a benchmark suite is presented that covers broad application scenarios in the domain of cloud computing and includes diverse and representative data sets. The benchmark consists of 19 benchmarks on big data applications. According to this study big data applications have very low operation intensity, which measures the ratio of the total number of instructions divided by the total byte number of memory accesses.

III. Programming frameworks

The use of heterogeneous systems comes at a significant cost: the increase in programming complexity. To overcome this problem, new programing frameworks must be developed that hide the complexity of the heterogeneous systems without affecting the overall system performance. The programming of the FPGAs can be overcome by using *High Level Languages* such as OpenCL or C [16][17][18][19]. However, there are still several issues that needs to be resolved on deploying the FPGAs in the data centers. For example, the main issues are the virtualization and the partitioning of the hardware resources, the configuration of the FPGAs, and the scheduling of the hardware accelerators based on the applications requirements. In the last couple of years, there are several research efforts towards an efficient framework for the deployment of the FPGAs in the data centers.

A. FPGAs in the Cloud, IBM

In [20], a general framework is proposed for integrating FPGAs into the cloud by IBM. The framework proposes an accelerator pool (AP) that abstracts FPGA as a consumable resource while avoiding hardware dependencies of current FPGA technologies. In the AP abstraction, each FPGA has several pre-defined accelerators slots in which the hardware accelerators can be mapped. By utilizing the partial reconfiguration mechanism of the FPGAs, each slot can be considered a virtual resource that can be assigned for specific tasks. A cloud tenant can submit either pre-defined hardware accelerators that are hosted in central repository or can submit his own designs. However, in the latter case the cloud owner should perform the synthesis, place and route and generate the bitstreams for the FPGA slots. A prototype of the framework is implemented on an x86-based Linux-KVM environment with attached FPGAs and deployed in a modified OpenStack cloud environment. Four different accelerators are used for prototyping: Encryption (AES), Hashing (SHA), Stereo matching and Matrix-Vector Multiply. The performance evaluation shows that proposed framework allows the efficient utilization of the FPGA resources by the cloud tenants with less than 4 microseconds latency overhead of the visualization.

B. Virtualized Hardware accelerators, University of Toronto

In [21], a novel approach for integrating virtualized FPGA-based hardware resources into cloud computing systems with minimal overhead. The proposed framework allows cloud users to load and utilize hardware accelerators across multiple FPGAs using the same methods as the utilization of Virtual Machines. The reconfigurable resources of the FPGA are offered to the users as a generic cloud resources through OpenStack. An agent is introduced in this framework that implements the resource management of the OpenStack. The proposed framework splits the FPGA into several reconfigurable regions, each of which is managed a s single resource. Therefore, instead of a single FPGA bitstream, a collection of partial reconfigurable bitstreams corresponding to the user hardware is passed to the agent. Again, as in the case of the IBM, the cloud provider must generate the partial bitstream for each accelerator and for each partially reconfigurable slot since the current technology requires specific bitstreams for each region of the FPGA.

C. FPGAs in Hyperscale Data Centers, IBM Zurich

In [22], a framework is proposed by IBM Zurich that allows cloud users to combine multiple FPGAs in the programmable fabric. This allows cloud operators to offer an FPGA to users in a similar way as a standard server. In the proposed framework multiple user applications can be hosted on a single physical FPGA, somehow similar to multiple VMs running on the same hypervisor. Each user can get a partition of the entire user logic and uses it to implement its own applications. This partitioning is achieved by utilizing partial reconfiguration. With partial reconfiguration it is possible to dynamically reconfigure a portion of the FPGA while the rest of the regions remain untouchable. The proposed architecture has been integrated into the OpenStack framework and allows the renting of the FPGA resources on the cloud. The same architecture also allows the possibility to distribute their applications on a large number of FPGAs through an FPGA fabric. The integrated framework with multiple FPGAs is compared with a typical data center based on commodity processors. It is shown that if each system is based on 2048 module the FPGA-based system can provide 958 TFLOPS compared with the 442 TFLOPS offered by the commodity processors.

D. RC3E:Reconfigurable Cloud Computing Environment

In [23], a cloud hypervisor is proposed by Technical University of Dresden that integrates virtualized FPGA-based hardware accelerators into the cloud environment. The hypervisor allows users to implement and execute their own hardware designs on virtual FPGAs. The hypervisor has access to a database containing all physical and virtual FPGA devices in the cloud system and their allocation status. Each device is assigned to its physical host system (node). The user can allocate a complete physical FPGA, which has to be marked separately in the device database or can allocate portion of the vFPGA. In the case of vFPGA allocation, the configuration is performed by utilizing partial reconfiguration (PR). The proposed framework supports the required security by protecting the device files using access rights. This additional virtualization layer allows concurrent users to interact with their allocated devices without influencing each other.

E. Virtualized FPGA accelerators, University of Warwick

In [24], a novel framework is presented that integrates reconfigurable accelerators in a standard server with virtualized resource management and communication. The proposed framework integrates a PCIe based FPGA board into a standard data center server. The FPGA is partitioned into separate accelerator slots. Accelerator functions are either stored in a library on the host machine as partial bitstreams or can be uploaded by the user. In this framework, a hypervisor is implemented for the configuration and the scheduling of the user logic in the FPGA resources. When an accelerator is to be configured in the FPGA, the hypervisor decides on the optimal partial reconfiguration regions (PRRs) to host it and initiates reconfiguration. The hypervisor also maintains a list of the available PRRs and configured accelerators to avoid unnecessary reconfiguration when a required accelerator is already present in the FPGA and not in use.

IV. Hardware accelerators for cloud computing

This section presents the hardware accelerators that have been developed and are targeting specific cloud applications. These accelerators are mainly used to speedup widely used cloud computing applications like MapReduce, Spark, Memcached, etc. A brief overview of these accelerators is presented together with the main performance metrics for these accelerators such as kernel speedup, overall system speedup, power and energy savings.

A. Search engines and page ranking

1) Microsoft Catapult: Microsoft has presented in 2014 a reconfigurable hardware accelerator that is used for the acceleration of the Bing search engine called Catapult [25]. In the proposed system they placed a small daughter-card in each server with a single high-end FPGA, and connected the cards directly together with a secondary network. Services that require more than one FPGA can be mapped across FPGAs residing in multiple servers. The FPGA board is connected to the CPU through the PCIe interface. The proposed system was deployed in a real data center that consisted of 34 populated pods of machines in 17 racks, for a total of 1,632 machines. Each server uses an Intel Xeon motherboard, with 12-core Sandy Bridge CPUs.

For the utilization of the FPGAs, Microsoft has developed the API for the interfacing of the software with the FPGA and the interfaces between the FPGA applications and board level functions. To evaluate the proposed system they ported a significant fraction of Bings ranking engine onto the Catapult fabric. The hardware accelerators were developed in Verilog and they were partitioned across seven FPGAsplus one spare for redundancy. The main tasks that were mapped to the FPGAs were all of the free-form expressions, and all of the machine-learning model that is used in the Bing search engine for the page ranking. Specifically, one FPGA is used for feature extraction, two for free-form expressions, one for a compression stage that increases the efficiency of the scoring engines, and three FPGAs to hold the machine-learned scoring models. In the performance evaluation it was shown that the proposed FPGA achieves a 95% gain in throughput relative to the software version throughput at each ranking server with an equivalent latency distributionor at the same throughput, reduces tail latency by 29%. The power consumption overhead that was measured of the FPGA was 22.7W.

B. MapReduce

1) Scalable MapReduce Accelerator: In [26], an architecture for the FPGA acceleration of MapReduce applications is presented. A cluster of worker nodes is designed for the MapReduce framework, and each worker node consists of both a CPU-based worker and an FPGA-based worker. CPU base worker runs the major communications with other worker node and tasks, while FPGA base worker operates extended MapReduce tasks to speed up the computation processes. The Master node is designed as the manager for all workers, and is used to configure and dynamically monitor the tasks running in both software program and the FPGA-based workers. Two kinds of tasks are configured in the CPU worker: *real* tasks and *virtual* tasks. Virtual tasks treated as the normal task running in CPU workers are actually functions with system calls to the FPGA resources. Each task configured in FPGA worker has one relative virtual task interface in CPU worker for transmission. The proposed framework has been implemented by modifying the open-source Hadoop project [27] and mapped to the NetFPGA boards. The CPU worker runs the modified Hadoop MapReduce program [28] which processes file system and transmit data to FPGA workers. The proposed framework has been evaluated using two typical applications of the MapReduce framework: matrix multiplication and page ranking. For the case of matrix multiplication using one FPGA board, the proposed system can achieve almost 15$\times$ speedup compared to CPU. In the case of the page ranking, the proposed system can achieve approximately 4$\times$ times faster execution time compared to the software execution.

2) FPMP: MapReduce, Tsinghua University and Microsoft: One of the first attempt to accelerator cloud computing application using FPGA was presented by Microsoft and Tsinghua University in [29]. In this work a MapReduce framework on FPGA, which provides programming abstraction, hardware architecture, and basic building blocks to developers is presented. A processor scheduler is designed to dynamically utilize the hardware resources by monitoring the status of each *mapper* and *reducer*. There are two sets of queues in the processor scheduler. One queue set is for mappers and the other queue set is for reducers. Each queue set consists of two queues, one queue for idle processors and the other for pending tasks. The performance evaluation of the proposed system has been performed using the RankBoost application [30] that is used for page ranking. The most time consuming procedure of RankBoost is WeakLearn, which consumes more than 95% execution time and it is the one that is ported to the FPGA [31]. Both the *mapper* and the *reduce* tasks of the WeakLearn algorithm have been mapped to the FPGA. To test the performance of the RankBoost acceleration on FPMR, a real world dataset for a commercial search engine is used. This time-consuming procedure achieves up to 16.74$\times$ speedup in the FPMR framework while the overall system speedup is 14.44$\times$.

3) Reconfigurable MapReduce accelerator, DUTh-NTUA: In [32][33], an FPGA architecture is proposed for the efficient implementation of the MapReduce framework. The proposed architecture implements the Phoenix MapReduce framework that is a C-based version of the MapReduce. In one case ([33]) a HW-SW co-design is presented where the *Map* tasks are executed in the processors and a specialized hardware accelerator is implemented for the efficient processing of the *Reduce* tasks. The *Reduce* function is most of the applications is the same (e.g. accumulation or calculation of the average value). Therefore an efficient hardware accelerators is implemented that performs fast indexing of the key-value pairs using a cuckoo hashing scheme.

In the second architecture ([32], an integrated framework is proposed where the whole application is mapped to the FPGA. The *Map* computational kernels, that are usually application-specific, are created using High Level Synthesis (HLS) tools and the *Reduce* tasks, that are common to most of the applications, are executed using the common *Reduce* hardware accelerator. The presented system proposes the complete decoupling of MapReduces tasks data-paths to distinct buses, accessed from individual processing engines. Such a dataflow approach implies a holistic C/C++ to RTL domain-level MapReduce transition. The performance evaluation shows that the proposed scheme can achieve up to 4.3× overall speedup (system speedup) in MapReduce applications while offering significant lower power and energy consumption compared to a high-end multi-core processor. Specifically it can provide up to 25× lower power consumption and up to 33× better energy efficiency compared to the software-only solution in the low-power cores.

4) Big Data Analysis acceleration: In [34], a platform for energy-efficient acceleration of big data analytics applications using FPGAs is presented by GMU and UCLA. The proposed scheme analyzes data mining and machine learning algorithms, utilized extensively in big data applications, in a heterogeneous platform that included both CPUs and FPGAs. The proposed system, consists of a high performance CPU as the master node, which is connected to several all-programmable MP-SoCs (Zynq) devices as slave nodes. The master node runs the HDFS, and is used for the job scheduling between all the slave nodes. In this study, a comprehensive workload analysis and performance monitoring is done for four widely used machine learning kernels: K-means, KNN (pattern recognition algorithm), SVM (machine learning algorithm) and Naive Bayes. For the most computational intensive functions that were identified after the profiling, hardware accelerators have been created using High Level Synthesis (HLS) tool-flow. The proposed system has been implemented in the Zedboard that are used as the cluster's workers. The results show that a **kernel** speedup of up to 321× with HW+SW co-design can be achieved and up to $2.72x$ system speedup. Power is reduced by up to 3.7× on average. Moreover, since both the execution time and the power has been reduced through the acceleration, the *EnergyDelayProduct* (EDP) is significantly reduced by up to to 15.21×.

5) MapReduce for K-means, University of Hong Kong: The University of Hong Kong has presented the design and implementation of the k-means clustering algorithm on an FPGA-accelerated computer cluster [35]. The implementation followed the map-reduce programming model, with both the map and reduce functions executing autonomously to the CPU on multiple FPGAs. A hardware/software framework was developed to manage the execution on multiple FPGAs across the cluster. The experiment was run on three compute nodes, each containing a KC705 FPGA board from Xilinx. Each KC705 board contains a Kintex-7 FPGA connected to the CPU through a PCIe x3 interface. When compared to a similar software implementation executing over the Hadoop MapReduce framework, 15.5× to 20.6× performance improvement has been achieved across a range of input data sets.

C. Memcached

1) Memcached acceleration, HP: In [36], a hardware accelerator for the Memcached programming framework has been presented from HP. The proposed framework consists of two main blocks: a networking block and an Memcached applications block. The networking block is used for the parsing of the input packet flow to detect relevant packets and transmit the required Memcached commands and data to the Memcached block. The Memcached accelerator has been mapped to an Altera Stratix IV-based development board with two 4GB DDR2 memory modules and four 1 GbE ports. The proposed architecture provides similar performance with the typical servers based on general purpose processors but consumes only 9% of the power of the baseline. The total energy efficiency of the accelerator for the Memcached application can reach up to 10.9× depending on the workload.

2) Memcached acceleration, Xilinx: In [37][38], a fully compliant implementation of the Memcached framework is presented from Xilinx. The proposed scheme quantify novel hybrid memory systems that combine conventional DRAMs and serial-attached flash to increase value store capacity to 40Terabytes with up to 200 million entries while providing access at 80Gbps. This is achieved by an object distribution based on size using different storage devices over DRAM and flash and data-flow based architectures using customized memory controllers that compensate for large variations in access latencies and bandwidths. The proposed scheme is based on a novel hashing scheme for the efficient storing and searching of the values based on the key [39]. This framework is prototyped and measured using real-world value size distributions from Facebook, Wikipedia, Twitter and Flickr and compare to existing solutions. The performance evaluation of 1 FPGA is roughly comparable to 2 Xeon E5-2680 processors with some variation depending on use case specifics. The key differentiator is the memory capacity, as the FPGA can offer the performance in conjunction with flash and thereby scale to 40TB and provide much improved power efficiency per GB.

3) Thin Servers with Smart Pipes: In [40], accelerator has been proposed that can be coupled with commodity processors for servers. The proposed accelerator can process GET requests entirely in hardware, offloading both network handling and data look up. The accelerator deciphers the request sent from the Network Interface Card (NIC) and passes control signals along with the key to a hardware hash table implementation. While all of the GET requests are processed by the hardware accelerator, the memory allocation, the key-value pair eviction and replacement are implemented in software. The proposed accelerator has been demonstrated through an FPGA prototyping platform, and show the potential for a 6×-16× power-performance improvement over conventional server baselines.

4) Memcached, UTexas-Austin: A hardware accelerator for the Memcached applications is also presented by University of Texas at Austin in [41]. The proposed architecture presents a hardware accelerator for distributed key-value Memcached applications using a hybrid CPU+FPGA architecture. The accelerator, implemented on the FPGA fabric, processes request packets directly from the network, avoiding the CPU in most cases. The hardware acceleration is used for the implementation of the most computational intensive task that are on the critical path. That is *request parsing*, *item hash calculation*, *item look up*, *LRU update*, and *response assembly*. If the control flow exits the trace prematurely, the side effects of the computation are rolled back and the request packet is passed to the CPU. The proposed accelerator has been mapped to a Xilinx Virtex-5 FPGA part and compared with a Xeon core as a baseline. Compared to the Xeon processor the Memcached accelerator achieves almost the same throughput but it is 9.15× more energy efficient for common case requests.

D. In-memory Databases

1) Database acceleration, IBM: One of a widely-used application run in data centers is database analytics. IBM has recently presented a novel architecture for the acceleration of the in-memory database operations based on reconfigurable logic [42][43]. The accelerator is connected directly to the processors main memory instead of being in the I/O path, and it processes the latest in-memory data in the DBMSs buffer pool. The FPGA pulls the database pages from the main memory, parses the pages to extract, and processes the rows and writes the qualifying rows back to the main memory in database-formatted pages. The FPGA can be used for several tasks such as table joins, database sorting, and column projection. The FPGA is also used for the compression/decompression tasks that are used in database management systems (DBMS). The proposed system is evaluated on a commercial DBMS running on a 3.8-GHz multicore system with a PCIe-attached FPGA card with an Altera Stratix V FPGA. Overall, the FPGA achieves speedups in the range of 7× to 14× for most queries.

2) Database acceleration, Stanford: An integrated framework for the hardware acceleration of in-memory database analytics has been presented in [44], from Stanford. The proposed hardware design is used to accelerate three important primitive database operations: *selection*, *merge join*, and *sorting*. These three operation can be combined to perform one of the most fundamental database operations: the *table join*. The proposed system has been implemented in a Maxeler system with a Xilinx Virtex 6 FPGA. The proposed system based on the FPGA was compared in terms of row processing throughput against the baseline unmodified DBMS (Database Management System) running on a single core. The proposed hardware accelerator were able to obtain close to ideal utilization of available memory bandwidth, resulting in a 2.8×, 5.7×, and 1.4× improvement in utilization over software for selection, sorting, and joining, respectively.

3) Database walkers, EPFL, HP, Google: A novel hardware architecture for the acceleration of in-memory database analytics has also been shown in [45], called *Widx*. The proposed architecture is used for the acceleration of hash index lookups that is the largest singe bottleneck to the overall execution time. The critical path in hash index lookups consists of ALU-intensive key hashing followed by pointer chasing through a node list. The proposed system is an on-chip accelerator for database hash index lookups, which achieves both high performance and flexibility by decoupling key hashing from the list traversal, and at the same time processing multiple keys in parallel on a set of programmable units. The proposed system can integrated tightly to a conventional core, thus eliminating the need for a dedicated TLB and cache. An evaluation of the proposed framework on a set of modern data analytics workloads (TPC-H, TPC-DS) based on MonetDB showed an average speedup of 3.1x over an aggressive OoO (Out-of-Order) core on bulk hash table operations, while reducing the OoO core energy by 83%. In terms of the energy efficiency Widx improves the energy delay product by 5.5× over the in-order core and by 17.5× over the OoO baseline.

E. Spark

1) Spark acceleration, UToronto: In [46][47], a hardware accelerator has been presented for the K-means application based on the Apache Spark framework [10]. The hardware accelerators for the k-means computation have been developed using High-Level Synthesis (HLS). The proposed architecture has been prototyped in a heterogeneous CPU-FPGA cluster of Zynq development boards, leveraging the Apache Spark cluster management capabilities. Similar to a typical Spark cluster, the prototype cluster includes a *Master* and multiple *Workers*. Each worker includes the software and hardware support to offload the computation to the custom accelerators inside the FPGA. A Spark cluster of up to 16 worker nodes has been created for the performance evaluation. The FPGA comprise 64 mappers and 8 reducers, operating at 111.11 MHz and the FPGA include also two ARM 32-bit cores. The speedup increases ranges from 18.6× to 36.0× depending on the number of dimensions and centers for the fixed-point use-case compared to the software solution running only on the ARM cores. For the case of the floating-point, the speedup ranges from 20× to 40× depending again on the number of dimensions and centers. Compared to the Xeon core, the proposed dataflow architecture achieves a two-fold performance improvement using four nodes and 4× speedup using 8 cores.

2) Spark acceleration, UCLA: In [48], an integrated framework is presented for the efficient utilization of hardware accelerators under the Spark framework. The performance evaluation shows that the proposed system can achieve up to 3.05× speedup for the Logistic regression adn 1.47× speedup for the K-Means and reduces the overall energy consumption to 38% and 56% of the baseline respectively.

F. *Commercial products*

Besides the proposed architectures from academia, there are also commercial devices that are used for acceleration of cloud computing applications. **Ryft** offers a device, called Ryft one, that can be used for the acceleration of the Spark applications. Each Ryft ONE device allows users to simultaneously analyze up to 48TB of locally stored data at rest as well as streaming data. According to their datasheet, Ryft ONE can achieve up to 100× speedup while reducing costs by 70% [49]. The device utilize a library for commonly used task such as term frequency, search, and fuzzy search. **Falcon Computing** provides also commercial devices that are used for the acceleration of cloud computing and Big Data applications. Falcon Computing provides the automated compilation tool, the runtime management tool, as well as acceleration libraries to enable seamless integration of accelerators into existing data center infrastructures [50]. Their device can provide 2.7× speedup for iterative computations with data pipelining and caching techniques and 1.7× cluster throughput improvement by providing fine-grained accelerator sharing among multiple applications.

V. Classification and comparison

This section categorizes the proposed schemes and provides a qualitative and quantitative comparison on the features of these schemes such as speedup, energy efficiency, development method, and type of cloud application. All of the proposed frameworks presented in Section III, allows the virtualization of the hardware resources in the FPGAs. The virtualization of the FPGAs does not mean that the user can access in time-sharing ways the resources of the FPGA, as it is done in the case of typical processor virtualization. FPGA virtualization means that the user can allocate and utilized part of the FPGA without allocating the whole FPGA for his application. The virtualization of the FPGAs is usually performed by exploiting the partial reconfiguration in which part of the FPGA can be reconfigured while the rest of the FPGA can remain operational in the static partition of the FPGA.

Table I presents the hardware accelerators for cloud computing applications and the main features of each architecture.

1) Batch versus Streaming applications: As was described in Section II, the cloud applications can be divided to the ones that process large files (batch) and to the ones that process small chunks of data usually coming from the network (streaming). Usually, FPGAs are used for batch processing applications, where a large amount of data are offloaded to the FPGA for acceleration. This way, the overhead of the communication of the FPGA and the processor is minimized. However, as it is shown in the table, the FPGAs can also be used in many streaming applications. In many streaming applications, the computational complexity is very small compared to the required packet processing of the Network Interface Card (NIC), the operating system (OS) and the Virtual Machines (VM). Therefore, in many architectures, FPGAs can be used to offload both the NIC by using specialized TCP/IP offload engine (TOE) and specialized hardware for the actual processing of the data packets.

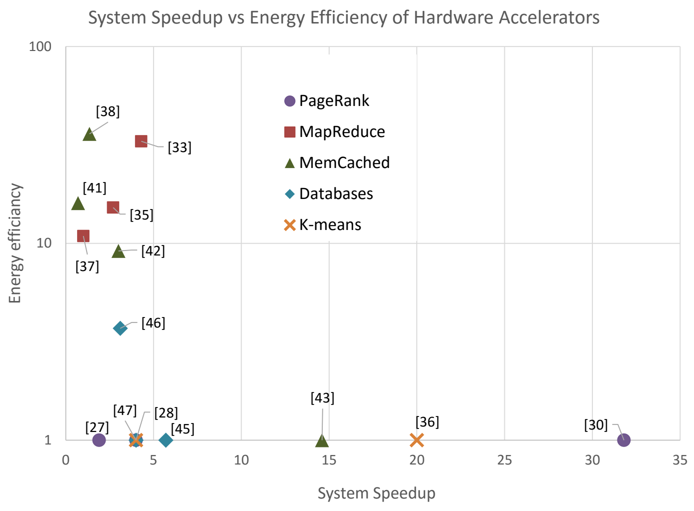


Fig. 2. Speedup vs Energy Efficiency for several hardware accelerator for cloud computing application. The architectures that they do not report the energy efficiency are shown in the x-axis.

2) System speedup: The main metric for the efficiency of the accelerator is the kernel speedup and the system speedup. **Kernel speedup** refers to the speedup that is achieved for the execution of a specific task on the hardware accelerator compared to the execution time of the this task in software using a typical processor. **System speedup** refers to the speedup achieved using the hardware accelerator for the execution of the complete application compared to the execution time of the application running in software. The system speedup takes into account the communication overhead for transferring data to the FPGA and vice versa. The system speedup takes also into account the overhead for the drivers that are used to perform the communication and the control of the accelerator. In most of the papers, the overall system speedup is reported rather than the kernel speedup of the task that is being accelerated. As is it shown in this table the speedup of the hardware accelerators ranges from 0.7× to 31.8×. In most of the cases that the speedup is low, emphasis has been given to the energy efficiency of the system. The speedup also depends on whether the proposed accelerators is used as a co-processor or it is used as a complete replacement to the processor. In the latter case, the speedup is usually higher since the overall application is running on the FPGA.

3) Energy efficiency: Besides the speedup, the hardware accelerator can also evaluated in terms of power and energy efficiency. **Power efficiency** is used to compare the power consumption of the hardware accelerator (or the system with the hardware accelerator) compared to the power consumption of the system executing the application in software. However, this metric is not widely used since the energy consumption depends not only on the power consumption but also on the execution time. **Energy efficiency** is used to compare the energy consumption of the hardware accelerator (or the system with the hardware accelerator) compared to the energy con-

TABLE I
SURVEY ON RECONFIGURABLE ACCELERATORS FOR CLOUD COMPUTING APPLICATIONS

Paper	Institute	Application	Type		Speedup	Energy	Interface	Design	Integration
			Batch	Stream					
[25]	Microsoft	Search engine		●	1.95x	-	PCIe	HDL	Coprocessor
[26]	NUDT	RankBoost (MapReduce)	●		4x	-	Ethernet	HDL	Coprocessor
[29]	TU, Microsoft	RankBoost (MapReduce)	●		31.8x	-	PCIe	HLL	Coprocessor
[32]	DUTh, NTUA	MapReduce	●		4.3x	33x	AXI4	HDL-HLL	Coprocessor
[34]	GMU, UCLA	MapReduce	●		2.7x	15.2x	AXI4	HLL	Coprocessor
[36]	HP, UML	Memcached		●	1x	10.9x	Ethernet	HDL	Standalone
[37]	Xilinx	Memcached		●	1.35x	36x	Ethernet	HDL	Standalone
[40]	HP, ARM, Facebook	Memcached		●	0.68x	16x	Custom	HDL	Coprocessor
[41]	UTAustin	Memcached		●	3x	9.15x	Custom	HDL	Coprocessor
[42]	IBM	Databases		●	14.6x	-	PCIe	HDL	Coprocessor
[44]	Stanford	Databases		●	5.7x	-	PCIe	OpenSPL	Coprocessor
[45]	EPFL,HP,UE, Google	Databases		●	3.1x	3.7x	Custom	HDL	Coprocessor
[46]	Toronto U	K-Means (Spark)	●		4x	-	PCIe	HLL	Coprocessor
[48]	UCLA	K-Means, LR (Spark)	●		3x	2.6x	PCIe	HLL	Coprocessor
[35]	HKU	K-Means (MapReduce)	●		20x	-	PCIe	HDL	Coprocessor

sumption of the system executing the application in software. The energy consumption may refer either to the total energy consumption in order to complete a specific application or it can be measured in terms of operations(tasks)/Watt.

The table shows the energy efficiency of each paper since it is the one that is mainly reported and it is the one that it is more fair as it compares the energy consumption for the execution of the complete application. The energy consumption ranges from $3.7\times$ to $36\times$. However, the 3.7 speedup refers to a hardware accelerator incorporated to the chip, as part of an ASIC and not to an FPGA hardware accelerator. The energy efficiency is also affected by the integration of the proposed system. In cases that the accelerator is proposed as a co-processor the overall energy efficiency is lower compared to the ones where the accelerator is used for the total replacement of the server processor. Overall it is shown that the hardware accelerators can achieve an order of magnitude better energy efficiency compared to the typical server processors. Figure 2 shows the speedup of the hardware accelerators versus the energy efficiency that they provide. The architectures that they do not report the energy efficiency are shown in the x-axis.

4) Communication Interface: This column on the table shows the interface that is used from the accelerator. In the cases that the accelerator is used as a co-processor, the interface is either the PCI express offering a total throughput of 16GB/sec or the AXI4 bus that provides an aggregated throughput of 25.6GB/sec when clocked at 200MHz. The latter case is used when the accelerator is part of the programmable Multi-Processor System-on-a-Chip (MPSoC). In the former case, multiple FPGA cards can be added in the PCIe allowing easier scalability of the hardware accelerators. In the cases that a complete system is proposed as a total replacement to the typical processor, the Ethernet is used for the reception and transmission of the data packets. In that case, as it was mentioned earlier, a TCP/IP-offload-engine (TOE) is used to speedup the processing requirements at the network level.

5) Design method: The main obstacle for the wide deployment of the FPGAs in the cloud is the high programming complexity of the hardware description languages (HDL). In the last years there are several efforts from the main FPGA and system vendors to allow users to program FPGA using high level languages (HLL) like OpenCL or specific-domain languages like OpenSPL. Although that HDLs can provide higher speedup, the low programming complexity of HLL make them very attractive in the domain of cloud computing. For example, in [29], it is reported that the fully manually designed version, achieves $33.5\times$ speedup, while the use of HLL achieves $31.8\times$ speedup. However, the most promising approach for the efficient deployment of the hardware accelerators in the data centers seems to the be the integrated programming frameworks presented in Section III and also plan to be adopted by major companies like Intel [51] and IBM[20][22]. These frameworks allows the development of hardware accelerators from 3rd party entities and the storing of these accelerators in the form of IP blocks in centralized repositories. This approach will allow the efficient development of hardware accelerators in HDL while at the same time it will allow the seamlessly deployment of the hardware accelerators by cloud tenants without any prior experience in HDLs.

6) Integration: The last column of the table shows the level of integration of each architecture. When the accelerators are used as a co-processor then the accelerator is used only for the computational intensive tasks while the rest of the tasks are executed in the processor. In the case that the hardware accelerator is used a complete replacement to the processor, then it has to support the complete processing of the data such as the network processing, etc. This approach is usually preferred for the acceleration of a specific application that will not need any major modifications. However, the co-processor approach has a higher level of flexibility since the computational intensive tasks remain in hardware while the rest of the tasks can be modified in software.

VI. CONCLUSIONS

Hardware accelerators looks as a promising solution for the increase of the performance in data centers and the reduction of the energy consumption. The main disadvantage of the hardware accelerators is the programming efficiency which can be overcome by using high level languages such as OpenCL and HLS. The quantitative and the qualitative comparison shows that for widely used cloud computing applications like Memcached, in-memory databases, etc. the hardware accelerators can achieve a speedup ranging from 1× to 32×. The most import advantage however is the energy efficiency. The proposed schemes can achieve from 9× up to 36× better energy efficiency compared to the software solution running on typical server processors.

REFERENCES

[1] "Cisco Global Cloud Index: Forecast and Methodology, 20142019 White Paper," 2015.

[2] E. Savitz, "Bridging The Data Deluge Gap," *Forbes Tech*, August 2012.

[3] C. Martin, "Post-Dennard Scaling and the final Years of Moores Law," Tech. Rep., 2014.

[4] H. Esmaeilzadeh, E. Blem, R. S. Amant, K. Sankaralingam, and D. Burger, "Power challenges may end the multicore era," *Commun. ACM*, vol. 56, no. 2, pp. 93–102, Feb. 2013. [Online]. Available: http://doi.acm.org/10.1145/2408776.2408797

[5] ——, "Dark silicon and the end of multicore scaling," *IEEE Micro*, vol. 32, no. 3, pp. 122–134, May 2012.

[6] N. Hardavellas, M. Ferdman, B. Falsafi, and A. Ailamaki, "Toward dark silicon in servers," *IEEE Micro*, vol. 31, no. 4, pp. 6–15, Jul. 2011. [Online]. Available: http://dx.doi.org/10.1109/MM.2011.77

[7] M. Ferdman, A. Adileh, O. Kocberber, S. Volos, M. Alisafaee, D. Jevdjic, C. Kaynak, A. D. Popescu, A. Ailamaki, and B. Falsafi, "A case for specialized processors for scale-out workloads," *IEEE Micro's Top Picks*, 2014.

[8] "The Xilinx SDAccel Development Environment: Bringing The Best Performance/Watt to the Data Center," Tech. Rep., 2015.

[9] M. Ferdman, A. Adileh, O. Kocberber, S. Volos, M. Alisafaee, D. Jevdjic, C. Kaynak, A. D. Popescu, A. Ailamaki, and B. Falsafi, "Clearing the clouds: A study of emerging scale-out workloads on modern hardware," in *Proceedings of the Seventeenth International Conference on Architectural Support for Programming Languages and Operating Systems*, ser. ASPLOS XVII. New York, NY, USA: ACM, 2012, pp. 37–48. [Online]. Available: http://doi.acm.org/10.1145/2150976.2150982

[10] "Apache, spark." [Online]. Available: http://spark.apache.org/

[11] M. Li, J. Tan, Y. Wang, L. Zhang, and V. Salapura, "Sparkbench: A comprehensive benchmarking suite for in memory data analytic platform spark," in *Proceedings of the 12th ACM International Conference on Computing Frontiers*, ser. CF '15. New York, NY, USA: ACM, 2015, pp. 53:1–53:8. [Online]. Available: http://doi.acm.org/10.1145/2742854.2747283

[12] A. Yasin, Y. Ben-Asher, and A. Mendelson, "Deep-dive analysis of the data analytics workload in cloudsuite," in *Workload Characterization (IISWC), 2014 IEEE International Symposium on*, Oct 2014, pp. 202–211.

[13] T. Jiang, Q. Zhang, R. Hou, L. Chai, S. A. Mckee, Z. Jia, and N. Sun, "Understanding the behavior of in-memory computing workloads," in *Workload Characterization (IISWC), 2014 IEEE International Symposium on*, Oct 2014, pp. 22–30.

[14] K. Ousterhout, R. Rasti, S. Ratnasamy, S. Shenker, and B.-G. Chun, "Making sense of performance in data analytics frameworks," in *Proceedings of the 12th USENIX Conference on Networked Systems Design and Implementation*, ser. NSDI'15. Berkeley, CA, USA: USENIX Association, 2015, pp. 293–307. [Online]. Available: http://dl.acm.org/citation.cfm?id=2789770.2789791

[15] L. Wang, J. Zhan, C. Luo, Y. Zhu, Q. Yang, Y. He, W. Gao, Z. Jia, Y. Shi, S. Zhang, C. Zheng, G. Lu, K. Zhan, X. Li, and B. Qiu, "Bigdatabench: A big data benchmark suite from internet services," in *High Performance Computer Architecture (HPCA), 2014 IEEE 20th International Symposium on*, Feb 2014, pp. 488–499.

[16] S. Windh, X. Ma, R. J. Halstead, P. Budhkar, Z. Luna, O. Hussaini, and W. A. Najjar, "High-level language tools for reconfigurable computing," *Proceedings of the IEEE*, vol. 103, no. 3, pp. 390–408, March 2015.

[17] D. Bacon, R. Rabbah, and S. Shukla, "Fpga programming for the masses," *Queue*, vol. 11, no. 2, pp. 40:40–40:52, Feb. 2013. [Online]. Available: http://doi.acm.org/10.1145/2436696.2443836

[18] O. Segal, P. Colangelo, N. Nasiri, Z. Qian, and M. Margala, "Sparkcl: A unified programming framework for accelerators on heterogeneous clusters," *CoRR*, vol. abs/1505.01120, 2015. [Online]. Available: http://arxiv.org/abs/1505.01120

[19] O. Segal, M. Margala, S. R. Chalamalasetti, and M. Wright, "High level programming framework for fpgas in the data center," in *Field Programmable Logic and Applications (FPL), 2014 24th International Conference on*, Sept 2014, pp. 1–4.

[20] F. Chen, Y. Shan, Y. Zhang, Y. Wang, H. Franke, X. Chang, and K. Wang, "Enabling fpgas in the cloud," in *Proceedings of the 11th ACM Conference on Computing Frontiers*, ser. CF '14. New York, NY, USA: ACM, 2014, pp. 3:1–3:10. [Online]. Available: http://doi.acm.org/10.1145/2597917.2597929

[21] S. Byma, J. Steffan, H. Bannazadeh, A. Leon-Garcia, and P. Chow, "Fpgas in the cloud: Booting virtualized hardware accelerators with openstack," in *Field-Programmable Custom Computing Machines (FCCM), 2014 IEEE 22nd Annual International Symposium on*, May 2014, pp. 109–116.

[22] J. Weerasinghe, F. Abel, C. Hagleitner, and A. Herkersdorf, "Enabling fpgas in hyperscale data centers," in *2015 IEEE International Conference on Cloud and Big Data Computing (CBDCom 2015)*, May 2015.

[23] O. Knodel and R. G. Spallek, "RC3E: provision and management of reconfigurable hardware accelerators in a cloud environment," in *2nd International Workshop on FPGAs for Software Programmers*, 2015. [Online]. Available: http://arxiv.org/abs/1508.06843

[24] S. A. Fahmy, K. Vipin, and S. Shreejith, "Virtualized FPGA accelerators for efficient cloud computing," in *7th IEEE International Conference on Cloud Computing Technology and Science, CloudCom 2015, Vancouver, BC, Canada, November 30 - Dec. 3, 2015*, 2015, pp. 430–435. [Online]. Available: http://dx.doi.org/10.1109/CloudCom.2015.60

[25] A. Putnam, A. M. Caulfield, E. S. Chung, D. Chiou, K. Constantinides, J. Demme, H. Esmaeilzadeh, J. Fowers, G. P. Gopal, J. Gray, M. Haselman, S. Hauck, S. Heil, A. Hormati, J.-Y. Kim, S. Lanka, J. Larus, E. Peterson, S. Pope, A. Smith, J. Thong, P. Y. Xiao, and D. Burger, "A reconfigurable fabric for accelerating large-scale datacenter services," in *Proceeding of the 41st Annual International Symposium on Computer Architecuture*, ser. ISCA '14. Piscataway, NJ, USA: IEEE Press, 2014, pp. 13–24. [Online]. Available: http://dl.acm.org/citation.cfm?id=2665671.2665678

[26] D. Yin, G. Li, and K.-d. Huang, "Scalable mapreduce framework on fpga accelerated commodity hardware," in *Internet of Things, Smart Spaces, and Next Generation Networking*, ser. Lecture Notes in Computer Science, S. Andreev, S. Balandin, and Y. Koucheryavy, Eds., vol. 7469. Springer Berlin Heidelberg, 2012, pp. 280–294.

[27] "Apache, hadoop." [Online]. Available: http://hadoop.apache.org/

[28] J. Dean and S. Ghemawat, "Mapreduce: simplified data processing on large clusters," *Commun. ACM*, vol. 51, pp. 107–113, January 2008.

[29] Y. Shan, B. Wang, J. Yan, Y. Wang, N. Xu, and H. Yang, "Fpmr: Mapreduce framework on fpga," in *Proceedings of the 18th Annual ACM/SIGDA International Symposium on Field Programmable Gate Arrays*, ser. FPGA '10. New York, NY, USA: ACM, 2010, pp. 93–102. [Online]. Available: http://doi.acm.org/10.1145/1723112.1723129

[30] Y. Freund, R. Iyer, R. E. Schapire, and Y. Singer, "An efficient boosting algorithm for combining preferences," *J. Mach. Learn. Res.*, vol. 4, pp. 933–969, Dec. 2003. [Online]. Available: http://dl.acm.org/citation.cfm?id=945365.964285

[31] N.-Y. Xu, X.-F. Cai, R. Gao, L. Zhang, and F.-H. Hsu, "Fpga acceleration of rankboost in web search engines," *ACM Trans. Reconfigurable Technol. Syst.*, vol. 1, no. 4, pp. 19:1–19:19, Jan. 2009. [Online]. Available: http://doi.acm.org/10.1145/1462586.1462588

[32] C. Kachris, D. Diamantopoulos, G. C. Sirakoulis, and D. Soudris, "An fpga-based integrated mapreduce accelerator platform," *Journal of Signal Processing Systems*, pp. 1–13, 2016. [Online]. Available: http://dx.doi.org/10.1007/s11265-016-1108-7

[33] C. Kachris, G. C. Sirakoulis, and D. Soudris, "A reconfigurable mapreduce accelerator for multi-core all-programmable socs," in *System-on-Chip (SoC), 2014 International Symposium on*, Oct 2014, pp. 1–6.

[34] K. Neshatpour, M. Malik, M. A. Ghodrat, A. Sasan, and H. Homayoun, "Energy-efficient acceleration of big data analytics applications using fpgas," in *Big Data (Big Data), 2015 IEEE International Conference on*, Oct 2015, pp. 115–123.

[35] Y. M. Choi and H. K. H. So, "Map-reduce processing of k-means algorithm with fpga-accelerated computer cluster," in *Application-specific*

Systems, Architectures and Processors (ASAP), 2014 IEEE 25th International Conference on, June 2014, pp. 9–16.

[36] S. R. Chalamalasetti, K. Lim, M. Wright, A. AuYoung, P. Ranganathan, and M. Margala, "An fpga memcached appliance," in *Proceedings of the ACM/SIGDA International Symposium on Field Programmable Gate Arrays*, ser. FPGA '13. New York, NY, USA: ACM, 2013, pp. 245–254. [Online]. Available: http://doi.acm.org/10.1145/2435264.2435306

[37] M. Blott, L. Liu, K. Karras, and K. Vissers, "Scaling out to a single-node 80gbps memcached server with 40terabytes of memory," in *Proceedings of the 7th USENIX Conference on Hot Topics in Storage and File Systems*, ser. HotStorage'15. Berkeley, CA, USA: USENIX Association, 2015, pp. 8–8. [Online]. Available: http://dl.acm.org/citation.cfm?id=2827701.2827709

[38] M. Blott, K. Karras, L. Liu, K. Vissers, J. Bär, and Z. István, "Achieving 10gbps line-rate key-value stores with fpgas," in *Presented as part of the 5th USENIX Workshop on Hot Topics in Cloud Computing*. Berkeley, CA: USENIX, 2013. [Online]. Available: https://www.usenix.org/conference/hotcloud13/workshop-program/presentations/Blott

[39] Z. István, G. Alonso, M. Blott, and K. Vissers, "A hash table for line-rate data processing," *ACM Trans. Reconfigurable Technol. Syst.*, vol. 8, no. 2, pp. 13:1–13:15, Mar. 2015. [Online]. Available: http://doi.acm.org/10.1145/2629582

[40] K. Lim, D. Meisner, A. G. Saidi, P. Ranganathan, and T. F. Wenisch, "Thin servers with smart pipes: Designing soc accelerators for memcached," in *Proceedings of the 40th Annual International Symposium on Computer Architecture*, ser. ISCA '13. New York, NY, USA: ACM, 2013, pp. 36–47. [Online]. Available: http://doi.acm.org/10.1145/2485922.2485926

[41] M. Lavasani, H. Angepat, and D. Chiou, "An fpga-based in-line accelerator for memcached," *IEEE Comput. Archit. Lett.*, vol. 13, no. 2, pp. 57–60, Jul. 2014. [Online]. Available: http://dx.doi.org/10.1109/L-CA.2013.17

[42] B. Sukhwani, H. Min, M. Thoennes, P. Dube, B. Brezzo, S. Asaad, and D. E. Dillenberger, "Database analytics: A reconfigurable-computing approach," *IEEE Micro*, vol. 34, no. 1, pp. 19–29, Jan 2014.

[43] B. Sukhwani, H. Min, M. Thoennes, P. Dube, B. Iyer, B. Brezzo, D. Dillenberger, and S. Asaad, "Database analytics acceleration using fpgas," in *Proceedings of the 21st International Conference on Parallel Architectures and Compilation Techniques*, ser. PACT '12. New York, NY, USA: ACM, 2012, pp. 411–420. [Online]. Available: http://doi.acm.org/10.1145/2370816.2370874

[44] J. Casper and K. Olukotun, "Hardware acceleration of database operations," in *Proceedings of the 2014 ACM/SIGDA International Symposium on Field-programmable Gate Arrays*, ser. FPGA '14. New York, NY, USA: ACM, 2014, pp. 151–160. [Online]. Available: http://doi.acm.org/10.1145/2554688.2554787

[45] O. Kocberber, B. Grot, J. Picorel, B. Falsafi, K. Lim, and P. Ranganathan, "Meet the walkers: Accelerating index traversals for in-memory databases," in *Proceedings of the 46th Annual IEEE/ACM International Symposium on Microarchitecture*, ser. MICRO-46. New York, NY, USA: ACM, 2013, pp. 468–479. [Online]. Available: http://doi.acm.org/10.1145/2540708.2540748

[46] E. Ghasemi, "A Scalable Heterogeneous Dataflow Architecture For Big Data Analytics Using FPGAs," Master's thesis, Graduate Department of Electrical and Computer Engineering, 2015.

[47] E. Ghasemi and P. Chow, "A scalable heterogeneous dataflow architecture for big data analytics using fpgas (abstract only)," in *Proceedings of the 2016 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays, Monterey, CA, USA, February 21-23, 2016*, 2016, p. 274. [Online]. Available: http://doi.acm.org/10.1145/2847263.2847294

[48] J. Cong, M. Huang, D. Wu, and C. H. Yu, "Invited - heterogeneous datacenters: Options and opportunities," in *Proceedings of the 53rd Annual Design Automation Conference*, ser. DAC '16. New York, NY, USA: ACM, 2016, pp. 16:1–16:6. [Online]. Available: http://doi.acm.org/10.1145/2897937.2905012

[49] "Big Data, Meet the Small Ryft One," Tech. Rep., 2015.

[50] "Broader Paths Etched into FPGA Datacenter Roadmap," Tech. Rep., 2015. [Online]. Available: http://www.nextplatform.com/2016/02/29/broader-paths-etched-into-fpga-datacenter-roadmap/

[51] P. Gupta, "Xeon+fpga platform for the data center," in *Proceedings of the International Symposium on Computer Architectures (ISCA)*, 2015.