Big Biomedical Image Processing Hardware Acceleration: A Case Study for K-means and Image Filtering

Katayoun Neshatpour[1], Arezou Koohi[1], Farnoud Farahmand[1], Rajiv Joshi[2], Setareh Rafatirad[1], Avesta Sasan[1], and Houman Homayoun[1]

[1]George Mason University
[2]IBM T. J. Watson

***Abstract*— Most hospitals today are dealing with the big data problem, as they generate and store petabytes of patient records most of which in form of medical imaging, such as pathological images, CT scans and X-rays in their datacenters. Analyzing such large amounts of biomedical imaging data to enable discovery and guide physicians in personalized care is becoming an important focus of data mining and machine learning algorithms developed for biomedical Informatics (BMI). Algorithms that are developed for BMI heavily rely on complex and computationally intensive machine learning and data mining methods to learn from large data. The high processing demand of big biomedical imaging data has given rise to their implementation in high-end server platforms running software ecosystems that are optimized for dealing with large amount of data including Apache Hadoop and Apache Spark. However, efficient processing of such large amount of imaging data running computational intensive learning methods is becoming a challenging problem using state-of-the-art high performance computing server architectures. To address this challenge, in this paper, we introduce a scalable and efficient hardware acceleration method using low cost commodity FPGAs that is interfaced with a server architecture through a high speed interface. In this work we present a full end-to-end implementation of big data image processing and machine learning applications in a heterogeneous CPU+FPGA architecture. We develop the MapReduce implementation of K-means and Laplacian Filtering in Hadoop Streaming environment that allows developing mapper functions in non-Java based languages suited for interfacing with FPGA-based hardware accelerating environment. We accelerate the mapper functions through hardware+software (HW+SW) co-design. We do a full implementation of the HW+SW mappers on the Zynq FPGA platform. The results show promising kernel speedup of up to 27× for large image data sets. This translate to 7.8× and 1.8× speedup in an end-to-end Hadoop MapReduce implementation of K-means and Laplacian Filtering algorithm, respectively.**

I. INTRODUCTION

Use of biomedical imaging integrated into today's healthcare technologies has opened the door to apply image processing methods for both research and diagnostic purposes and enable discovery to guide physicians in personalized care. As every patient may receive a digital copy with the high resolution image of their recently imaged body part, we are increasingly creating a wealth of knowledge over the population for the imaging data that is used for better researching various diseases such as cancer, Alzheimer, and Multiple sclerosis, to just name a few.

The biomedical imaging data is stored in distributed database systems to be used for studying rare events that occur very infrequently. With the ever-growing availability of big biomedical imaging data in healthcare, rare event prediction and analysis to explain changes from usual functioning has become very important. Examples of such cases are monitoring continuous data streams from ICU patients, analyzing magnetic resonance imaging (MRI) medical images to detect breast or brain cancer tumor pattern that is hard to identify as cancerous or not in the concept of content based image retrieval. However, the ability to detect such rare event for decision-making lags behind our ability to mine big biomedical imaging data. This is particularly critical for streaming data, where the goal is to monitor and make decision in real-time in order to alert when unusual event occurs.

The aim of medical image processing is to automatically extract information from the both 2D and 3D images, e.g., automatically segmenting a brain MRI facilitates brain research [1]. Furthermore, statistical analysis of large databases of images are used to correlate characteristics with disease outcome.

An example of a database that collects images modalities like MRI and CT on different cancer types is the cancer imaging archive (TCIA) [2], which is used for studies of the molecular biology of cancer. An effort has been initiated to link the imaging data to genomic data in the cancer genomic atlas (TCGA) [3] for a more comprehensive analysis of the cancer genome. One example is discussed in [4] in which, cancer gene expression is linked to MRI imaging results to analyze the survival time and disease progression of Glioblastoma multiform.

Another imaging modality used in large volume for disease outcome prediction purposes is digital pathology. These images have high resolution and are obtained from the tissue biopsy of patients. An example is a recent longitudinal study of 20 years of pathology data in 18 registered databases on 108196 women with DCIS [5]. Similar studies are conducted over large database of images, in order to answer important questions regarding prognosis and the course of treatment for cancer patients.

Various image processing and machine learning techniques are used for segmenting medical images. For unsupervised-clustering region growing, K-means, Markov random field and expected maximization are utilized. For supervised image pixel classification, algorithms like KNN, Maximum likelihood or Nave Bayes are used [6]. Some examples of image segmentation are to cluster different parts of a brain like gray matter, white matter, and CFS on MRI image modality; to classify tumors on CT scans; or, to find the calcification and tumor spots on digital mammograms [6].

978-1-4799-5341-7/16/$31.00 ©2016 IEEE

Another common image processing application is to use pattern recognition techniques to classify part of a medical image as a specific pattern. For instance, in the case of pathology, images are used to decide if the region of interest is cancerous. Different feature extraction techniques followed by classifiers like SVM or random forest are used to classify the region of interest. Feature extraction is usually achieved through the use of image filters like Gaussian of Laplacian or Sobel edge detector, in addition to the extraction of color histograms. Feature extraction is applied both in the training and test phase.

As explained in some examples above, there is a growing need to find ways to access and analyze image data both locally and also in distributed manner. In addition, faster analysis of high volume images can pave the way to include more image data in the study of interest. Given the enormity of size of biomedical imaging data, Big data technologies can help to significantly reduce the processing time of such data. An efficient platform for large volume image processing is Apache Hadoop.

Apache Hadoop is a Java-based, open-source implementation of MapReduce. MapReduce [7], a framework utilized extensively for big-data applications, is a well-utilized implementation for processing large data sets in which, the programs are parallelized and executed on a large cluster of commodity servers. MapReduce consists of both map and reduce functions. The map functions parcel out work to different nodes in the distributed cluster and the reduce functions collate the work and resolves the results.

Examples of implementation of big biomedical imaging data in high-end server platforms, are HIPI [8] and picture archival and communication systems (PACS) that are running in servers equipped with high-end Xeon cores.

The challenge with big data environment including the Hadoop MapReduce as well as the high-end server architecture, is that none of them have been design and optimized to address the computing demand of image processing of large volume of biomedical data. Today's server architectures in fact, have been mainly optimized for web service applications, which are mainly I/O intensive [9], [10]; however, biomedical imaging analytics applications are combining two substantially different characteristics: data intensity and computational intensity [11]. These applications are relying on complex machine learning and data mining algorithms in particular for rare-event analytics using classification, regression, and clustering methods that are computationally intensive.

In general, emerging big data analytics applications including biomedical-imaging analytics require a significant amount of server computational power. The costs of building and running an Exascale computing server to process big data applications and the capacity to which we can scale it, are driven in large part by those computational resources, which are not optimized to meet energy-efficiency and performance demand of emerging analytics applications.

In response to performance and energy-efficiency challenge, in other domains, hardware acceleration through specialization have emerged as a promising solution partially in response to the dark silicon challenge to enhance energy efficiency by allowing each application to run on a computing system that matches its resource needs more closely than a general purpose computing one size-fits-all processing node [12], [13], [14]. This will be just as true for Big biomedical imaging analytics, if not more so. Integration of hardware accelerators in server platforms has provided an opportunity to exploit the high computational capacity of FPGAs and GPUs to improve performance of this class of applications by many-fold. In fact, FPGA and GPUs provide a cost-effective solution for the scalability challenge we are facing in future data center, as the size of data grows.

Although integration of GPU in todays high-end servers provides opportunity to accelerate performance of biomedical imaging application and has been studied extensively in recent years [15], [16], it is not a power efficient solution [17], [18]. To address the computing and energy-efficiency requirements of big biomedical imaging applications, and based on the benchmarking and characterization results, we envision a data-driven heterogeneous architecture for next generation big data server platforms that leverage the power of FPGA to build custom accelerators. Therefore in this paper we present an implementation of image processing applications in a heterogeneous CPU+FPGA architecture. We develop the MapReduce implementation of K-means and image filtering in Hadoop Streaming environment that allows developing mapper functions in non-Java based language suited for interfacing with FPGA-based hardware accelerating environment. We accelerate the mapper functions through hardware+software (HW+SW) co-design. We do a full implementation of the HW+SW mappers on the Zynq FPGA platform. The results show promising speedup of up to 17$\times$ for image data sets.

II. Related Work

HIPI is a Hadoop image processing interface software that is used to read and distribute image data on HDFS file system of Hadoop in parallel manner. HIPI lets the user access each image as a single file and hide the detail of MapReduce framework. Users can use image processing and OpenCV application for the image provided by HIPI interface. It can be used on large scale imaging datasets [8]. [19] has used Hadoop MapReduce platform for image analysis of remote sensing images. They have used 8 different practical image processing application to show that Hadoop can be used to process large image data in addition to text data. [20] has implemented kmeans for clustering of remote sensing images using MapReduce platform, in which they have turned the image data to text file, in order to read the images into the Hadoop platform. In [21], FPMR is introduced as a MapReduce framework on the FPGA with RankBoost as a case study, which adopts a dynamic scheduling policy for better resource utilization and load balancing. In [22], a hardware accelerated MapReduce architecture is implemented on Tilera's many core processor board. In this architecture data mapping, data merging and data reducing processing are offloaded to the accelerators. The Terasort benchmark is utilized to evaluate the proposed architecture. In [23] hardware acceleration is explored through an eight-salve Zynq-based MapReduce architecture. It is implemented for a standard FIR filter to show the benefits gained through hardware acceleration in the MapReduce framework where the whole low-pass filter is implemented on the FPGA. In [24], a configurable hardware accelerator is used to speed up the processing of multi-core and cloud computing applications on the MapReduce framework. The accelerator is utilized to carry out the reduce tasks. In [25], [26], [27], we used hardware acceleration of MapReduce to accelerate machine-learning kernels. In this paper, we used hardware acceleration

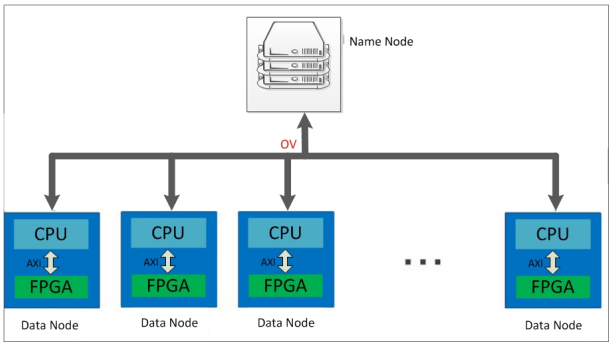


Fig. 1. Studied system architecture.

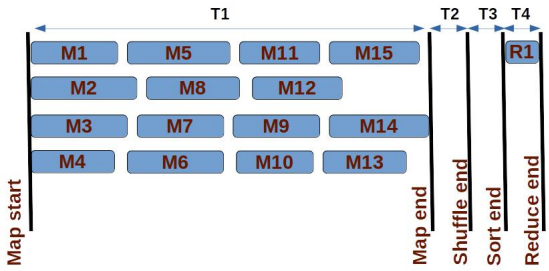


Fig. 2. Timing of various Hadoop phases

to enhance the performance of biomedical imaging applications.

III. System Architecture and Methodology

The system studied in this paper, consists of a high-performance CPU as the name node, which is connected to several CPUs that are in turn, connected to FPGAs. The FPGA+CPU work together as data nodes.

The name node runs the HDFS and is responsible for the job scheduling between all the data nodes. It is configured to distribute the computation workloads among the data nodes, as shown in Fig. 1, where Intel Xeon E5-242 is deployed as the master-node, and the data nodes are equipped with FPGAs. Each data node has a fixed number of map and reduce slots, the number of which, is statistically configured. In a multi-node cluster name node and data nodes are usually on different machines. However, on a single node platform as the one studied in this paper, they are located on the same node. Although Hadoop exploits the cluster infrastructure for the big data applications, to evaluate the performance analysis, we focus on a single node of the cluster.

We study two applications, Laplacian filtering of images for edge detection purposes and K-means clustering of Biomedical images. Filtering on an image is a neighborhood operation, in which the value of any given pixel in the output image is determined by applying some algorithm to the values of the pixels in the neighborhood of the corresponding input pixel. K-means clustering, a machine-learning algorithm that divides the data into K different clusters, is utilized in image processing to analyze remote sensing images.

We develop a MapReduce implementation of the applications and execute them on Hadoop Streaming environment that allows developing mapper and reducer functions in non-Java based language suited for interfacing with FPGA-based hardware accelerating environment. We perform a thorough analysis of individual execution phases in MapReduce environment. This information is used to find out how much time is spent in each phase of MapReduce execution including map, reduce, sort, shuffle, cleanup and setup and to calculate the execution time, when each of these phases are accelerated.

Fig. 2 shows the timing diagram of a MapReduce application with 15 map jobs, 4 mapper slots and one reduce job. As shown in Fig. 2, the map phase initiates with the start of the first map task and finishes when the last map task completes its execution. T1 shows the map time for the given example.

The timing information shows that for each mapper node, there is a time interval before the end of previous map task and the start of next map task. This time interval accounts for data transfer between the name node and the data node, which is not accelerated. However, the time interval throughout which, the map task is being executed is accelerated through hardware acceleration.

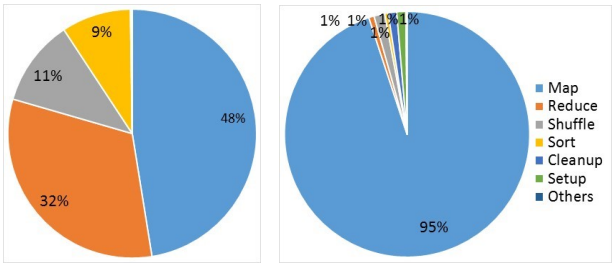
In order to have an estimation of the acceleration of the mapper phase, the map functions are accelerated on the Zynq FPGA boards. Zynq devices, which combine ARM cores with an FPGA, allow HW+SW co-design of applications. We calculate the speedup gained through HW+SW co-design of the map functions on the Zedboards, in order to estimate the range of speedup achievable on the data nodes when they are accelerated with programmable logic. The ZedBoard featuring XC7Z020 Zynq SoCs, integrate two 667 MHz ARM Cortex-A9 with an Artix-7 FPGA with 85 KB logic cells and 560 KB block RAM.

As discussed earlier, not all the execution time of the overall MapReduce application is spent in the map function, thus, the speedup realized on the map function are translated to lower speedups on the overall MapReduce platform depicted in Fig. 1. Thus the final acceleration is determined not only by the extent to which we can accelerate the map function, but also by the portion of the total execution time in the Hadoop MapReduce that is devoted to the map function.

IV. Results

We implemented filtering and K-means for 1264 and 1600 images, respectively, which were converted to text files with a total size of 5GB. K-means is an unsupervised clustering scheme that can be used directly for clustering on biomedical images including blood cells on biopsy images [28], or alternatively, as an initialization step for expected maximization and random field. Image filtering is applied for feature extraction of medical images as a first step toward image pattern classification. We have implemented Laplacian filter weights in our filter implementation which can be used as a blob detector in pathology images.

Fig. 3 shows the breakdown of the timing of various phases in the MapReduce platform for these two applications. Fig. 3 shows that most of the execution time is devoted to the map and reduce functions. While a significant portion of the execution time is devoted to the map phase in the K-means (i.e. 95%), only 48% of the execution times is spent in the map phase in the filtering algorithm. For this application the reduce function takes up a significant portion of the execution time, which shows that the speedup can be significantly enhanced if both map and reduce functions are accelerated. Such argument however, is not applicable to the K-means in which, only 1% of the execution time is devoted to the reduce phase.



(a) (b)

Fig. 3. Timing break-down of different phases in (a) Laplacian filtering, (b) K-means.

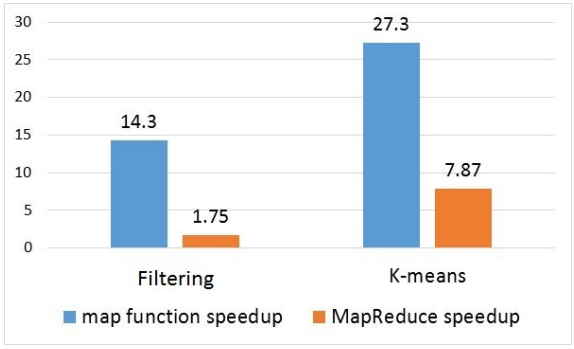


Fig. 4. Speedup of the map function versus the overall MapReduce platform.

Fig. 4 shows the speedup values of the map phase and the overall Hadoop streaming implementation for these two algorithms. With K-means a map function speedup of $27.3\times$ is translated to $7.87\times$ on the overall MapReduce platform, which is a $3.5\times$ drop in the speedup. With Laplacian Filter a map function speedup of $14.3\times$ is translated to $1.75\times$ on the overall MapReduce platform, which is a $8.2\times$ drop in the speedup. The drop in speedup in the overall MapReduce platform is higher for the filtering algorithm, since for this algorithm the portion of time devoted to the map phase is lower, and even if we were able to speedup the map phase indefinitely, still the other phases take up a significant time to finish.

V. Conclusion

Efficient processing of large amount of biomedical imaging data running computational intensive learning methods is becoming a challenging problem using state-of-the-art high performance computing server architectures. In response, this paper presents a method and framework to accelerate the processing of large scale biomedical imaging applications using commodity FPGA platform. Laplacian filtering of images for edge detection purposes and K-means clustering of Biomedical images has been studied for acceleration. A MapReduce implementation of the two algorithms has been developed and the computational intensive tasks were mapped to the FPGA. The results show promising kernel speedup of up to $27\times$ for large image data sets. This translate to $7.8\times$ and $1.8\times$ speed up in an end-to-end Hadoop MapReduce implementation of K-means and Laplacian Filtering algorithm, respectively. The framework and methodology can be used to accelerate other computationally intensive biomedical image processing applications that are dealing with large scale imaging data.

References

[1] M. Balafar and et. al., "Review of brain mri image segmentation methods," *Artificial Intelligence Review*, vol. 33, no. 3, pp. 261–274, 2010.
[2] "The cancer genome atlas data portal," http://cancerimagingarchive.net.
[3] "The cancer imaging archive," https://tcga-data.
[4] P. O. Zinn and et. al., "Radiogenomic mapping of edema/cellular invasion MRI-Phenotypes in Glioblastoma Multiforme," *PLoS ONE*, vol. 6, Oct 2011.
[5] S. Narod and et. al., "Breast cancer mortality after a diagnosis of ductal carcinoma in situ," *JAMA Oncology*, vol. 1, no. 7, pp. 888–896, 2015.
[6] D. L. Pham, C. Y. Xu, and J. L. Prince, "A survey of current methods in medical image segmentation," *Annu. Rev. Biomed. Eng.*, vol. 2, pp. 315–337, 2000.
[7] J. Dean and S. Ghemawat, "Mapreduce: simplified data processing on large clusters," in *Proc. conf symp operation systems design and implementation*, 2004.
[8] C. Sweeney, L. Liu, S. Arietta, and J. Lawrence, "Hipi: a hadoop image processing interface for image-based mapreduce tasks."
[9] M. Malik and H. Homayoun, "Big data on low power cores: Are low power embedded processors a good fit for the big data workloads?" in *33rd IEEE International Conference on Computer Design, ICCD 2015*.
[10] M. Malik, S. Rafatirad, A. Sasan, and H. Homayoun, "System and architecture level characterization of big data applications on big and little core server architectures," in *2015 IEEE International Conference on Big Data (Big Data)*, 2015.
[11] P. Otto, M. Malik, N. Akhlaghi, R. Sequeira, H. Homayoun, and S. Sikdar, "Power and performance characterization, analysis and tuning for energy-efficient edge detection on atom and ARM based platforms," in *IEEE International Conference on Computer Design, ICCD 2015*.
[12] J. Bisasky, H. Homayoun, F. Yazdani, and T. Mohsenin, "A 64-core platform for biomedical signal processing," in *International Symposium on Quality Electronic Design, ISQED 2013*.
[13] A. M. Kulkarni, H. Homayoun, and T. Mohsenin, "A parallel and reconfigurable architecture for efficient OMP compressive sensing reconstruction," in *Great Lakes Symposium on VLSI 2014, GLSVLSI '14, Houston, TX*, 2014.
[14] M. K. Tavana, A. M. Kulkarni, A. Rahimi, T. Mohsenin, and H. Homayoun, "Energy-efficient mapping of biomedical applications on domain-specific accelerator under process variation," in *International Symposium on Low Power Electronics and Design, ISLPED'14*, 2014.
[15] J. A. Stuart, C.-K. Chen, K.-L. Ma, and J. D. Owens, "Multi-GPU volume rendering using mapreduce," in *Proc ACM Int Symp High Performance Distributed Computing*, 2010, pp. 841–848.
[16] M. Xin and H. Li, "An implementation of GPU accelerated MapReduce: Using Hadoop with OpenCL for data-and compute-intensive jobs," in *IEEE Int Joint Conf Service Sciences*, 2012, pp. 6–11.
[17] J. Fowers, G. Brown, P. Cooke, and G. Stitt, "A performance and energy comparison of FPGAs, GPUs, and multicores for sliding-window applications," in *Proc ACM/SIGDA int symp Field Programmable Gate Arrays*, 2012, pp. 47–56.
[18] S. Che, J. Li, J. W. Sheaffer, K. Skadron, and J. Lach, "Accelerating compute-intensive applications with GPUs and FPGAs," in *IEEE Symp Application Specific Processors (SASP)*, 2008, pp. 101–107.
[19] M. H. Almeer, "Hadoop mapreduce for remote sensing image analysis."
[20] Z. Lv, Y. Hu, H. Zhong, J. Wu, B. Li, and H. Zhao, "Parallel k-means clustering of remote sensing images based on mapreduce," in *Web Information Systems and Mining*. Springer, 2010, pp. 162–170.
[21] Y. Shan, B. Wang, J. Yan, Y. Wang, N. Xu, and H. Yang, "FPMR: Mapreduce framework on FPGA," in *Proc Annual ACM/SIGDA Int Symp Field Programmable Gate Arrays*, 2010, pp. 93–102.
[22] T. Honjo and K. Oikawa, "Hardware acceleration of Hadoop MapReduce," in *Proc IEEE Int Conf Big Data*, Oct 2014, pp. 118–124.
[23] Z. Lin and P. Chow, "Zcluster: A zynq-based hadoop cluster," in *Int. Conf. Field-Programmable Technology (FPT)*, Dec 2013, pp. 450–453.
[24] C. Kachris, G. C. Sirakoulis, and D. Soudris, "A configurable mapreduce accelerator for multi-core fpgas," in *Proc ACM/SIGDA Intl Symp FPGAs*, 2014, pp. 241–241.
[25] K. Neshatpour, M. Malik, M. Ghodrat, and H. Homayoun, "Accelerating big data analytics using fpgas," in , *2015 IEEE 23rd Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM)*, May 2015, pp. 164–164.
[26] K. Neshatpour, M. Malik, A. Sasan, and H. Homayoun, "Energy-efficient acceleration of big data analytics applications using FPGAs," in *2015 IEEE International Conference on Big Data (Big Data)*, 2015.
[27] K. Neshatpour, M. Malik, and H. Homayoun, "Accelerating machine learning kernel in Hadoop using FPGAs," in *IEEE/ACM Int Symp Cluster, Cloud and Grid Computing (CCGrid)*, 2015.
[28] M. Veta and et al., "Assessment of algorithms for mitosis detection in breast cancer histopathology images," pp. 237–248, April 2014.