Scalable Computing: Practice and Experience, ISSN 1895-1767, http://www.scpe.org
© 2024 SCPE. Volume 25, Issues 4, pp. 3152–3162, DOI 10.12694/scpe.v25i4.2871

APPROXIMATE COMPUTING BASED LOW-POWER FPGA DESIGN FOR BIG DATA ANALYTICS IN CLOUD ENVIRONMENTS

MURALI DOVA [*]AND ANURADHA M SANDI [†]

Abstract. As cloud computing continues to evolve, the demand for scalable and energy-efficient infrastructure to handle extensive applications becomes paramount. Traditional transistor scaling and microprocessor design methods no longer suffice to meet the growing scale of cloud usage. This research explores the potential of approximate computing (AC) as an innovative solution to these challenges, particularly in high-demand computational settings. AC, known for its ability to make controlled accuracy trade-offs, is identified as a key strategy for improving both the performance and energy efficiency of cloud infrastructure, with a focus on low-power Field-Programmable Gate Array (FPGA) designs. This paper introduces novel methodologies that harness the strengths of AC, emphasizing its application in neural-based and machine-learning techniques for energy-efficient solutions. By targeting the performance of AC, especially in varied application domains and complex data mining scenarios, we propose two groundbreaking approaches that significantly enhance computational speed and reduce energy consumption. Our empirical analysis demonstrates notable improvements over existing techniques, highlighting the effectiveness of AC in optimizing cloud infrastructure. The proposed model on FPGA through cloud computing attains substantial elevation rates by 89 % and energy reduction by 122 %, which had been good outcomes. This study not only confirms the benefits of integrating AC with low-power FPGA designs for cloud environments but also sets a new benchmark for future research in achieving more sustainable and efficient cloud computing via VLSI FGPA design analysis solutions.

Key words: Low-power FPGA design, approximate computing, benchmark applications, and energy efficiency

1. Introduction.

The role of cloud computing infrastructure has emerged as a transformative force, reshaping the way computational resources are provisioned and utilized across diverse domains. This paradigm shift has been driven by the need for scalable, flexible, and cost-effective solutions to accommodate the exponential growth of data and the increasing complexity of applications. Cloud computing offers a dynamic environment where organizations can access a vast pool of virtualized resources, enabling them to scale up or down as demands fluctuate, all while reducing the burdens of managing and maintaining physical hardware.However, the efficacy of cloud computing infrastructure, while remarkable, is not without its challenges. One of the foremost challenges lies in the ever-increasing computational demands that stem from a myriad of applications, spanning from data analytics and machine learning to real-time communication and complex simulations. Traditionally, transistor scaling and microprocessor design have been the cornerstones of performance improvements. Yet, as Moore's Law faces diminishing returns and the power consumption of processors continues to rise, these approaches are reaching their limits in delivering the required computational power to match the scale of cloud usage.

This inherent disparity between computational demand and conventional performance improvement methods necessitates a departure from established norms. It is here that the concept of approximate computing (AC) emerges as a compelling and unconventional approach. At its core, approximate computing posits that in certain applications, a certain level of accuracy can be relaxed without significantly compromising the overall quality of results. By judiciously introducing controlled inaccuracies at specific points in computations, it becomes possible to unlock remarkable gains in terms of both performance and energy efficiency.The allure of AC lies in its alignment with the ethos of cloud computing - optimizing resource utilization and minimizing energy consumption. As cloud infrastructure aims to cater to diverse workloads from various clients, the ability to

[*]Guru Nanak Dev Engineering College, Bidar, Visvesvaraya Technological University, Belgavi-590018, Karnataka, India. (murali.d@nriit.edu.in).

[†]Department of ECE, Guru Nanak Dev Engineering College, Bidar, Visvesvaraya Technological University, Belgavi-590018 Karnataka, India. (anu29975@gmail.com).

strike a balance between accuracy and efficiency becomes paramount. For instance, in scenarios where milliseconds matter less than throughput or where exhaustive accuracy is not required, AC becomes a potent tool for enhancing the overall efficiency of cloud-based applications.Moreover, AC's applicability is further accentuated when coupled with low-power computing architectures like Field-Programmable Gate Arrays (FPGAs). FPGAs provide an ideal platform for AC experiments, as their programmable nature allows for customized hardware configurations that can leverage approximate computations effectively. The synergy between AC and FPGAs holds the promise of enabling efficient and highly customizable computing solutions tailored to the specific needs of cloud-based services.

Approximate Computing (AC) emerges as a transformative solution for cloud computing, enabling scalable, energy-efficient, and adaptable infrastructure by intentionally incorporating inaccuracies to enhance performance without critically impacting result quality. This approach not only aligns with sustainability and cost-efficiency goals but also fosters the development of custom, low-power computing solutions like FPGAs, addressing the limitations of traditional computing and advancing the fields of data science and AI.

In light of these dynamics, this paper embarks on exploring the potential of AC in the context of cloud computing infrastructure. Through a combination of theoretical insights and practical implementations, this research aims to elucidate how the integration of approximate computing strategies can reshape the efficiency landscape of cloud-based applications. By investigating innovative methodologies that harness AC's power while mitigating its limitations, this study endeavors to provide a nuanced understanding of the symbiotic relationship between approximate computing and cloud infrastructure, fostering a new era of optimized computational resource utilizationFuture hardware will be unreliable if it is dependent on inexact computations in the context of big data and data analytics, or high scale RMS applications. Software that takes use of future hardware is needed at the same time. There will be changes to the underlying abstractions we've been living with for a long time. While computer prices have decreased by half in the past two years, it is necessary to go beyond Gordon E. Moore's law, which claims that "the number of transistors on a microchip doubles about every two years." This law falls short when it comes to meeting current and future demands as well as scaling up. Approximate computing (AC) and the investigation of low-power FPGA circuits are the solutions to the problem. Semiconductor companies encounter difficulties in developing software with a dependable hardware layer of 10nm. Soft errors are becoming more prevalent as a result of variables such as a decrease in processor lifespan and an increase in variability (SER). Resilient models of computing rather than deterministic hardware models will be essential in the future. In the presence of imprecise software and hardware components, approximate computing offers the opportunity to design acceptable systems. Future calculations will be greatly impacted by two key considerations: cost and energy. The importance of a quality-performance tradeoff cannot be overstated. As a result, the system's performance can be enhanced by sacrificing accuracy.

Literature shows that there have been some recent efforts to implement AC. In addition, they contain a probabilistic transformation compiler from MIT and an AC-based runtime from MSR, SAGE, etc., Blink DB is the database that incorporates the AC from MIT, as well as several processor designs using ANNs from MSR and other institutions. Rice provides probabilistic CMOS, whereas Purdue provides AC components, to name a few. Neuronal approximation computing phenomena were the subject of this research, since it has become a useful tool in the present era of data science and artificial intelligence (AI). RMS applications can be accelerated by neural transformation of algorithms. The proposed hardware design in this study aims to diversify approximation computing methodologies for diverse benchmark applications, leading to the development of low-power FPGA devices that can fully leverage AC. . To name only a few: Jmeint (3D gaming), Inverseektj (robotics), JPEG encoder, Sobel (image processing), Black-Scholes (financial analysis), and K-Means (signal processing) (machine learning). Classifier topology, approximation topology, test and training data, as well as the field or domain are all diverse in the datasets. Several benchmark applications of domain heterogeneity and different mining algorithms are used in this work to leverage the invocation performance of approximation computing. The proposed solutions outperform current best practices in terms of speed and energy efficiency, according to an empirical investigation. Our article makes several distinct contributions to the field, encompassing the following key advancements:

- We introduce two novel architectural frameworks aimed at enhancing the utilization of approximate computing (AC) by integrating multiple classifiers and employing diverse approximation techniques.

These architectures offer innovative avenues to harness the potential of AC for optimizing computational processes.
- Our work encompasses the design of a hardware architecture that facilitates the integration of multiple approximation methods within a CPU-based environment. This architecture is thoughtfully crafted to accommodate the execution of diverse approximations, thereby contributing to the effective implementation of AC strategies.
- To validate the effectiveness of our proposed methodologies, we construct a functional prototype that enables us to rigorously evaluate their performance. Through meticulous experimentation and analysis, we present comprehensive comparative results against established state-of-the-art techniques, thus offering insights into the tangible benefits of our novel approaches.

The rest of the paper is arranged in this manner. Accordingly, Section 2 focuses on the most recent developments in approximate computation. Design materials and procedures that increase energy efficiency with acceptable precision are discussed in Section 3. Results from studies involving eight different benchmarks are presented in Section 4. Section 5 wraps up the report and suggests further research avenues.

2. Related Work. This section reviews literature on approximate computing and also the possible designs based on them. Venkataramani, S. et al. [1] focused on problems with technology scaling in future to meet the needs of computing. They investigated the utility of AC and the need for using it with innovative methods. Zhang et al. [2] proposed an AC framework and its advantages for Artificial Neural Networks (ANNs) with an optimization procedure for energy efficiency. However, there are many challenges of AC as explored by Agrawal et al. [3]. Apart from challenges with respect to proper usage, AC also provides many opportunities in terms of scalability and energy efficiency in large scale computing paradigms. Shafique et al. [4] on the other hand studied cross-layer AC. They explored logic and architecture layer to fill gaps for optimization. Divya et al. [5] proposed approximate computing based system with statistical guarantees in managing quality trade-offs. Li et al. [6] considered AC as a low power technique and proposed approximate circuits to exploit AC approaches. They used AC-aware scheduling for reaping its benefits. They intend to propose methods for control-intensive and data-intensive use cases. Different techniques of approximate computing can be found in Mittal, S. [7] . They include precision and scaling, loop perforation, load value approximation, memorization, data sampling and voltage scaling to mention few. Xu et al. [8] proposed AC method to achieve quality trade-off control which is similar to the work in Moreau, T., Sampson, A., &Ceze, L. [9] opined that AC techniques make mobile systems more efficient. As mobile computing devices have low resources, AC is found suitable for them. Wang et al. [10] presented an FPGA architecture where AC plays crucial role in energy saving. In pattern recognition that is machine learning based, AC is found to improve performance. Boikos and Bouganis [11] explored low-power robotics with the SLAM algorithm on FPGA design. Jafari et al. [12] proposed an embedded deep convolutional neural network (CNN) for scalability and low power consumption. Mametjanov et al. [13] proposed a methodology for auto tuning FPGA design to achieve better performance and power reduction. It was based on machine learning technique to identify parameters for tuning. In future, they intended to apply it for larger designs. Licciardo et al. [14] used a novel radix-3 partitioning method for better floating point operations. Wirthlin, M. [15] explored high-reliability systems made up of FPGA designs that exploit programmability. Ortega-Zamorano et al. [16] used backpropagation method on FPGA design for efficiency of microcontrollers. The algorithm belongs to machine learning with neural networks. It could reduce memory usage and also power consumption.Lei Zhang.[17] studied the FPGA designs based on te usage of ANN models. A three layer ANN is used to achieve desired results in performance in terms of memory usage and power consumption. Zhang et al. [18] proposed a scheme based on FPGA. It is named as Quasi-Centralized Direct Model Predictive Control (QC-DMPC) which controls dc-link voltage. Lu et al. [19] explored the execution of deep learning algorithms on FPGA designs faster. With FPGA design the complexity of the algorithms is reduced besides improving energy saving. Arram et al. [20] proposed methodology for improving FPGA designs that exploit configurability and thus accelerate short read alignment. Wang et al. [21] proposed an FPGA based real-time high quality stereo vision system which is found to be scalable and energy efficient for image processing applications Esmaeilzadeh, H et.al[22] NeuroPIM unveils an innovative processing-in-memory architecture that utilizes a neural network as a versatile accelerator, designed based on the insight that neural networks can approximate certain or entire program outputs in real-world applications.

This architecture melds the versatility of general-purpose processors with the high performance of specialized accelerators, demonstrating up to 41% faster performance than processor-based neural network accelerators and up to eight times the speed of standard general-purpose processors Kalangi, R. R [23]. The deployment of compute-intensive applications in fields such as Artificial Intelligence (AI) and Digital Signal Processing (DSP) presents a significant challenge, compelling the computing systems community to investigate novel design methodologies Saikumar, K. et.al [24]. Approximate Computing has emerged as a promising solution, offering a way to adjust the quality of results during system design to enhance energy efficiency or performance Durga, B. K et.al [25].

3. Materials and Methods. We proposed two architectures to exploit AC based on iterative training approach. It results in a hardware design that supports neural approach for approximate computing in CPU setting (In future we intend to improve it for GPU).

3.1. Baseline Architecture. In presence of multiple approximations there are many opportunities that can lead to efficiency without losing accuracy. However, it is challenging to have optimal invocations provided heterogeneity of RMS applications for which AC is applied. When there is huge training data in such applications, classifier predicts the data that is safe for AC in the current iteration and other data to subsequent iterations. The clustering of “safe to approximate” samples lead to probability of invoking approximators. Given an RMS application, we use iterative training and discover safe to approximate samples. The iterations are tracked to monitor such discovery in each iteration.

The baseline architecture for training has pairs of approximators and classifiers. They divide input space into two clusters namely safe to approximate cluster and unsafe to approximate cluster. The first pair uses original data as input. The pair contains approximator 1 and classifier 1. The training process is iterative nature which results in identification of data samples that are safe to approximate and data which is not converged goes to the next iteration. Ultimately the final unsafe to approximate data only enters into CPU for exact computing process. Based on the prediction of classifier 1, it determines whether the input data can be approximated or not. If yes, the data is given to approximator 1 else it is given to classifier 2. This way, the moves on to all classifier and approximator pairs. The data rejected by all classifiers is finally executed by CPU with precise computations. In the baseline approach, it is time consuming process to speed up with approximate computing invocations.

3.2. Extended Architecture. This architecture is meticulously crafted to cater to the dynamic demands of modern applications, particularly those characterized by their error-tolerant nature, such as Recognition, Mining, and Search (RMS) applications.At its core, the architecture is engineered to accommodate diverse approximation methods, enabling the processing unit to seamlessly switch between different techniques based on the specific task and its associated requirements. This adaptability ensures that the processing unit can effectively exploit the benefits of approximation while maintaining a high degree of performance efficiency. By employing a modular and flexible design approach, the architecture empowers applications to achieve a finely tuned balance between computational accuracy and performance gains.

Furthermore, the proposed hardware architecture is geared towards optimizing energy utilization and computational throughput. It achieves this through strategic resource allocation and dynamic selection of approximation techniques, thereby capitalizing on the inherent trade-offs between accuracy and efficiency. This holistic approach not only enhances the overall performance of RMS applications but also aligns with the growing emphasis on energy-efficient computing solutions.More efficient architecture is designed to leverage performance. It overcomes limitations of the baseline model. Confidence is said to be suitable approximator. Thus multinomial classifier provides input samples to suitable approximator and finally divides input space into n+1 partitions. The second data allocation method is parallel in nature. In this case, all inputs samples are assigned to all approximators in parallel. Approximators compete in the process and produce fit input samples as many as possible. Due to randomness in the training samples, each approximator may produce bias in distribution. In other words, each approximator may exhibit different local minima. Every approximator in this approach tests input sample and produces an approximation error. Based on the lowest approximation error, a babel is assigned to the sample. With all the labels generate, the multinomial classifier gets trained and produces a knowledge model used for testing data further.

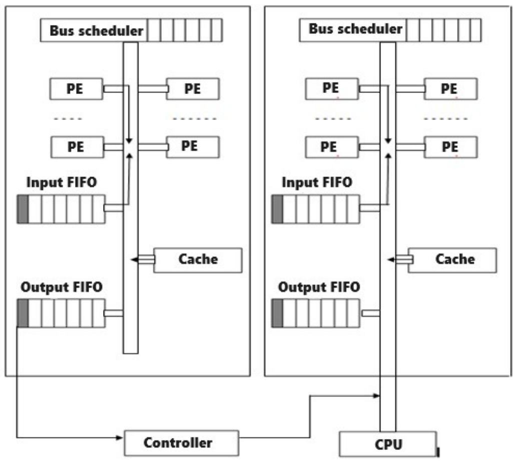

Fig. 3.1: Proposed hardware architecture for processing unit

The original data is subjected to multi-class classifier that produces safe to approximate data to different approximators. In the process the two data allocation models are employed empirically and the resultant data samples are given to appropriate approximator. The data samples that are not suitable for any approximator are given to CPU for exact computations. The extended model aims to improve approximator invocation performance as much as possible. The architecture's unique selling point lies in its capacity to seamlessly integrate with low-power FPGA designs, a synergy that underscores its adaptability and suitability for contemporary computing paradigms. By uniting the capabilities of FPGA technology with the advantages of approximate computing, the architecture opens up avenues for improved energy efficiency, reduced power consumption, and elevated performance across a diverse range of applications.

The above figure 3.1 clearly explains about proposed architecture of design. This FPGA design can give accurate scalable computation in adverse at data available scenarios.

3.3. Proposed Hardware Design. FPGAs can facilitate the verification of various VLSI design functionalities before fabrication by allowing implemented designs to be imported into the FPGA for testing. This analysis enhances hardware scalability and reduces manufacturing costs. Identifying any bugs in the design during the simulation stage can mitigate issues early on, saving on manufacturing expenses. In this manner improve the scalability of FPGA and its design process.As the extended architecture has multiple approximations, a neural processing unit (NPU) based hardware is designed in CPU setting (not targeted for GPU). The hardware design facilitates switch for various approximators and improves parallelism. As shown in Figure 3.1, the hardware architecture has identical tiles (computing resources). In each tile, there are many processing elements (PEs). An internal bus connects cache and input/output FIFOs. PE is meant for computing outcome of one neuron at a given time. This is as part of the neural network while performing inference task. Data scheduling between input FIFOs and PEs is carried out by bus scheduler. The bus scheduler also performs scheduling of weights associated with neural network between cache and specific PE. The structure of PE used in the hardware architecture has its own modus operandi in terms of functionality. It is graphically illustrated in Figure 3.2.

The PE has different components such as weight buffer fetch unit, W register, I register MAC and Sigmoid. The Fetch unit is meant for reading weights from the address (specific) associated with the weight buffer and sends the same to the W register which is shown in Figure 3.2. Input FIFO sends data to the I register where it is stored. When values arrive at registers, arithmetic computations are carried out. An activation unit such as Sigmoid will activate the result. Finally, the result is sent to the output buffer.so, FIFO instead of parallel in parallel out and other parallel procedures for providing delay for arithmetic computations were proposed. Based on the availability of resources and neural networks, classifiers and approximators are dynamically allocated. A

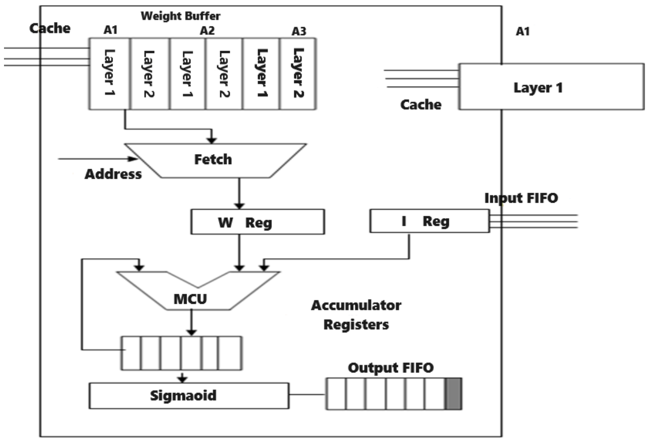

Fig. 3.2: Structure of PE used in the hardware architecture

controller is used to receive results from the classifier and give the required control signal to the approximator. As shown in Figure 3.1, in the initialization phase weights of classifiers are loaded. In the first stage, data comes from input FIFO to each PE. In stage 2, the PE computers the outcome of a neuron, and the result is sent to output FIFO. In the third stage, the result is sent to the controller from output FIFO. In stage 4 approximator is invoked by the controller only if the data is found to be safe to approximate, otherwise, the CPU is invoked for exact computations. In stage 5, input data is given to PEs present in the approximator for required computations. Then in stage 6, the results are sent to output FIFO. There is a need for a weight switch among approximators in the proposed design. Thus there are three scenarios supported. First, the weight buffer is capable of storing all weights associated with all approximators. In this case, the cache is not used for loading weights. Second, the weight buffer has less capacity and cannot accommodate the weights of all approximations.

3.4. Mathematical computations. in this section, brief notes on FPGA handling computations were explained. The error approximation, energy normalization, and power consumption are major elements of VLSI design. These parameters have been explained through the below Scalable computations. Approximation Error (AE):

$$AE = [ExactResult - ApproximateResult] \tag{3.1}$$

The above equation 3.1 clearly explains approximation error, if this error was less then say that the design provides fast scalable computations.

$$MAE = \sum \_i = 1^n \; \|ExactResult\_i - ApproximateResults\_i\| \tag{3.2}$$

Equation 3.2 clearly explains the Mean Absolute Error (MAE) parameter, indicating the error rate at high-speed operations.

$$NormalizedEnergyReduction(\%) = (Base - Power) - (educed - power)/(Base - Power) \tag{3.3}$$

In this case, a cache has been used to get Normalized power which is shown in eq 3.3. The weight buffer can store weights of only one approximation in the function. In this case, the cache is needed. Thus the proposed hardware architecture is capable of supporting all scenarios that may be experienced by an NPU (neural processing unit) design. It is achieved by adding a controller that switches approximators. Once data comes out of computations, its movement between steps remains the same.

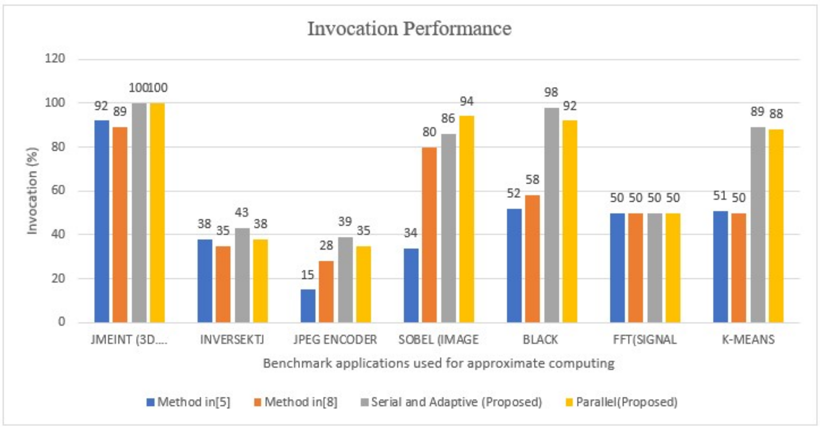

Fig. 4.1: Invocation performance across benchmarks

4. Results and Discussion. Experiments are made with 8 benchmark applications to which the proposed architectures are applied for approximate computing. The proposed methods are compared with two state-of-the-art architectures found in earlier studies. Multilayer perceptron (MLP) with backpropagation (BP) is used for all approximators and classifiers. Eight benchmark applications as mentioned in the Section 1 are used for empirical study. The models are evaluated using Root Mean Square Error (RMSE) and invocation of classifiers. The notion of error bound is used that is nothing but quality needs for the output. When the error bound is lower, higher in quality.

In this experiment, input vectors of 16, 32, and 64 bits were utilized. These input vectors were dynamically retrieved from a cloud server. The proposed design underwent verification using these test vectors, allowing for a comprehensive assessment of both its functionality and performance. The results showed that the proposed architecture showed better performance over the state of the art and error bound is well under threshold.

As presented in Figure 4.1, the benchmark applications used for approximate computing are provided in horizontal axis. The vertical axis shows the invocation percentage. More in invocation percentage indicates more performance a given architecture. The two proposed architectures are compared with the architectures in past models. With Jmeint (3D gaming) application, the two proposed methods achieved 100% invocation while methods of statistical guarantees and approximate computing showed 92% and 89% respectively. The proposed methods showed improvement in the case of all benchmarks except FFT (signal processing) application. Therefore, FFT is considered not suitable for approximate computing. Sobel (image processing), black-scholes (financial analysis) and K-Means (machine learning) applications exhibited improved invocation % for the proposed methods.

As presented in Figure 4.2, normalized approximation error about error bound is shown in the vertical axis while the horizontal axis shows all the benchmark applications used for approximate computing. The normalized error of the proposed methods is less when compared with the methods in [5] and [8] in the case of the Jmeint and JPEG encoder. In the case of the Inversektj application, the proposed methods outperform the method in [5]. The serial and adaptive methods showed better performance over all other methods in the case of the Sobel application. Concerning the FFT application, all four methods showed the same performance. Concerning K-Means the proposed parallel architecture showed better performance over all other methods which is shown in figure 4.1.

As shown in Figure 4.3, the error bound is shown in the horizontal axis and the vertical axis shows invocation (%). This experiment is made with the Black-Sholes benchmark application. As the error bound is low, the invocation percentage is less. As the error bound is increased from 0.025 to 0.1 gradually, the invocation percentage is increased steadily. In all error bounds, the proposed two methods outperformed the state of the art except in case of 0.1 error bound where the method in [8] showed better performance overall. But the

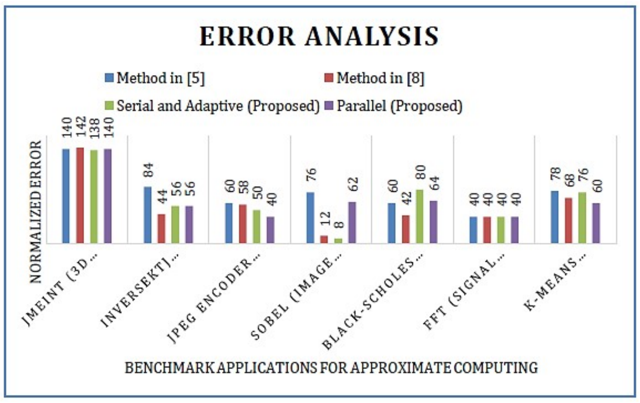

Fig. 4.2: Normalized error analysis for different benchmarks of approximate computing

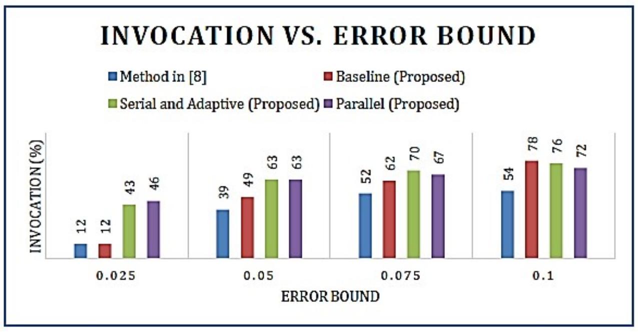

Fig. 4.3: Invocation percentage varying error bound (Black-Sholes benchmark)

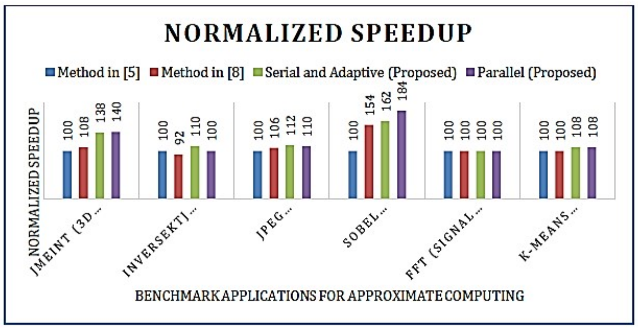

Fig. 4.4: Normalized speedup (with reference to Method in [5])

proposed methods showed better invocation performance over the method in [5]. From this experiment, it is understood that the proposed methods outperformed existing ones with higher invocation percentages that is much desired in approximate computing use cases.

As presented in Figure 4.4, the speedup normalized concerning method in [5] is proposed in the vertical axis while the horizontal axis shows the benchmark applications used in empirical study. The speedup of the proposed methods is increased in the case of all benchmarks except FFT. It is understandable as FFT is found not suitable for approximate computing earlier without improvement in invocation percentage. In the case of

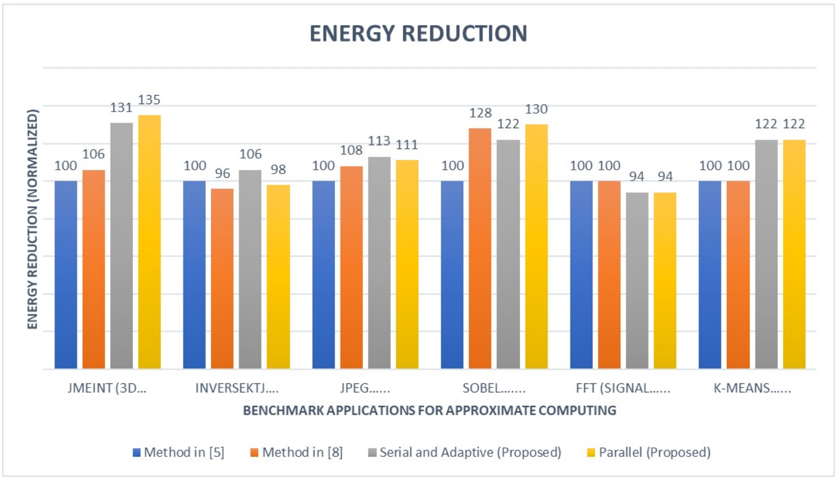

Fig. 4.5: Energy reduction performance comparison (normalized to Method in [5])

Sobel application, the proposed parallel method outperformed all other methods. Concerning Inversektj and JPEG encoder applications, the serial and adaptive methods showed higher speedup.

As presented in Figure 4.5, the normalized energy reduction for the method of statistical computing is shown in the vertical axis. The horizontal axis shows the benchmark applications. Energy efficiency achieved at the cost of loss of quality (acceptable) is the important proposition or hypothesis tested in this paper. Accordingly, energy reduction is achieved in the empirical study. More energy efficiency is achieved with higher invocation % of approximate computing. The proposed invocation-driven architectures led to higher energy savings. With the proposed methods, energy reduction is more when compared with methods like earlier computing techniques. Energy reduction is higher in all the benchmarks except in the case of the FFT application where the proposed methods could not achieve higher reduction in energy consumption. Higher energy reduction is achieved by the proposed parallel architecture in the case of the Jmeint application. From the results, it is understood that the proposed architectures showed better performance over the state of the art in [5] and [8] in case of invocation performance, speed up and energy saving. Thus, the hypothesis “performance improvement at the cost of an acceptable reduction in quality” has significance and relevance to present and also future hardware architectures.

5. Conclusion and Future Scope. In this research, two innovative architectural paradigms strategically designed to harness the potential of benchmark applications marked by both the cloud domain and FPGA VLSI platform towards heterogeneity. The objective is to organize the significant enhancements in terms of speedup and energy efficiency, thereby exploiting the potential of low-power FPGA designs in tandem with approximate computing. The study delves into the invocation performance of different approximate computing techniques, thoughtfully employed across a spectrum of mining (RMS) tasks. Through rigorous evaluation involving diverse benchmark applications, our proposed architectures stand out as powerful contenders. Empirical findings highlight their superiority over two existing methods. For instance, with the K-Means machine learning benchmark, the existing methods demonstrated 51% and 50% invocation performance, whereas our proposed serial and adaptive, as well as parallel methods showcased substantially elevated rates of 89% and 88%, respectively. Similarly, the proposed methods exhibited a striking 122% reduction in normalized energy consumption, overshadowing the 100% achieved by existing methods for the K-Means benchmark. This compelling evidence underscores the pronounced performance augmentation offered by our architectural innovations, as evidenced through noteworthy improvements in both speedup and energy reduction over existing state-of-the-art techniques.

Moreover this research trajectory aligns with an aspiration to further refine the architectural design by introducing dynamic search operations aimed at automatically identifying the most energy-efficient approximate design while adhering to predefined quality constraints. This forward-looking pursuit holds the potential to revolutionize the landscape of approximate computing, ushering in enhanced energy efficiency without compromising on quality benchmarks. By relentlessly pushing the boundaries of innovation, we envision a future where the synergy between architectural ingenuity and intelligent optimization drives unprecedented advances in the domains of cloud-based RMS applications and low-power FPGA designs.

REFERENCES

[1] Venkataramani, S., Chakradhar, S. T., Roy, K., & Raghunathan, A. (2015)., *Approximate computing and the quest for computing efficiency.* Proceedings of the 52nd Annual Design Automation Conference on DAC 15.p 1-6

[2] Zhang, Q., Wang, T., Tian, Y., Yuan, F., & Xu, Q. (2015, March) , *ApproxANN: An Approximate Computing Framework for Artificial Neural Network*, Design, Automation & Test in Europe Conference & Exhibition (DATE), 2015 p 601-607

[3] Agrawal, A., Choi, J., Gopalakrishnan, K., Gupta, S., Nair, R., Oh, J., ... & Sura, Z. (2016, october), *Approximate computing: Challenges and opportunities.* In 2016 IEEE International Conference on Rebooting Computing (ICRC) p 1-8. IEEE.

[4] Shafique, M., Hafiz, R., Rehman, S., El-Harouni, W., & Henkel, J. (2016, June). , *Cross-layer approximate computing: From logic to architectures.* In Proceedings of the 53rd Annual Design Automation Conference p 1-6.

[5] Mahajan, D., Yazdanbakhsh, A., Park, J., Thwaites, B., & Esmaeilzadeh, H. (2016)., *Towards statistical guarantees in controlling quality tradeoffs for approximate acceleration.* ACM SIGARCH Computer Architecture News, 44(3),p 66-77.

[6] Li, C., Luo, W., Sapatnekar, S. S., & Hu, J. (2015, June). , *Joint precision optimization and high level synthesis for approximate computing.* In Proceedings of the 52nd annual design automation conference p 1-6.

[7] Mittal, S. (2016). , *A survey of techniques for approximate computing.* ACM Computing Surveys (CSUR), 48(4),p 1-33.

[8] Xu, C., Wu, X., Yin, W., Xu, Q., Jing, N., Liang, X., & Jiang, L. (2017, June). , *On quality trade-off control for approximate computing using iterative training.* In Proceedings of the 54th Annual Design Automation Conference 2017 p. 1-6.

[9] Moreau, T., Sampson, A., & Ceze, L. (2015). , *Approximate computing: Making mobile systems more efficient.* IEEE Pervasive Computing, 14(2),p 9-13.

[10] Wang, Q., Li, Y., & Li, P. (2016, May). , *Liquid state machine-based pattern recognition on FPGA with firing-activity dependent power gating and approximate computing. In 2016 IEEE International Symposium on Circuits and Systems (ISCAS)* .p. 361-364. IEEE.

[11] Boikos, K., & Bouganis, C. S. (2016, August). , *Semi-dense SLAM on an FPGA SoC. In 2016 26th International Conference on Field Programmable Logic and Applications* .(FPL)p. 1-4. IEEE.

[12] Jafari, A., Ganesan, A., Thalisetty, C. S. K., Sivasubramanian, V., Oates, T., & Mohsenin, T. (2018). , *Sensornet: A scalable and low-power deep convolutional neural network for multimodal data classification.* IEEE Transactions on Circuits and Systems I: Regular Papers, 66(1),p 274-287.

[13] Mametjanov, A., Balaprakash, P., Choudary, C., Hovland, P. D., Wild, S. M., & Sabin, G. (2015, May). , *Autotuning FPGA design parameters for performance and power.* In 2015 IEEE 23rd Annual International Symposium on Field-Programmable Custom Computing Machines (p. 84-91). IEEE.

[14] Licciardo, G. D., Cappetta, C., Di Benedetto, L., & Vigliar, M. (2016). , *Weighted partitioning for fast multiplierless multiple-constant convolution circuit.* IEEE Transactions on Circuits and Systems II: Express Briefs, 64(1),p 66-70.

[15] Wirthlin, M. (2015). , *High-reliability FPGA-based systems: Space, high-energy physics, and beyond.* Proceedings of the IEEE, 103(3),p 379-389.

[16] Ortega-Zamorano, F., Jerez, J. M., Munoz, D. U., Luque-Baena, R. M., & Franco, L. (2015). , *). Efficient implementation of the backpropagation algorithm in FPGAs and microcontrollers.* IEEE transactions on neural networks and learning systems, 27(9),p 1840-1850.

[17] Lei Zhang. (2017). , *Artificial Neural Network Model Design and Topology Analysis for FPGA Implementation of Lorenz Chaotic Generator.* IEEE,p 1-5.

[18] Zhang, Z., Wang, F., Sun, T., Rodríguez, J., & Kennel, R. (2015). , *FPGA-based experimental investigation of a quasi-centralized model predictive control for back-to-back converters.* IEEE Transactions on Power Electronics, 31(1),p 662-674.

[19] Liang, Y., Lu, L., Xiao, Q., & Yan, S. (2019). , *Evaluating fast algorithms for convolutional neural networks on FPGAs.* IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 39(4), 857-870.

[20] Arram, J., Kaplan, T., Luk, W., & Jiang, P. (2016). , *Leveraging FPGAs for accelerating short read alignment.* IEEE/ACM transactions on computational biology and bioinformatics, 14(3), 668-677.

[21] Wang, W., Yan, J., Xu, N., Wang, Y., & Hsu, F. H. (2015). , *Real-time high-quality stereo vision system in FPGA.* IEEE Transactions on Circuits and Systems for Video Technology, 25(10),p 1696-1708.

[22] Esmaeilzadeh, H., Sampson, A., Ceze, L., & Burger, D. (2012, December). , *Neural acceleration for general-purpose approximate programs.* In 2012 45th annual IEEE/ACM international symposium on microarchitecture p. 449-460. IEEE.

[23] Kalangi, R. R., Janardhanarao, S., Sirisha, J., Saikumar, K., Veeranjaneyulu, P., & Suman, M. (2023).,*Prevention Of DDOS Attacks in Cloud Using Combinational Learning Approach.* In 2023 4th IEEE Global Conference for Advancement in Technology (GCAT) (pp. 1-6). IEEE.

[24] Saikumar, K., Rao, K. S., Baza, M., Rasheed, A., Hanuman, A. S., & Obaid, A. J. (2023).,*A Lite-SVM Based Semantic Search Model for Bigdata Analytics in Smart Cities.*,In 2023 26th International Symposium on Wireless

Personal Multimedia Communications (WPMC) (pp. 272-277). IEEE.

[25] Durga, B. K., Rajesh, V., Jagannadham, S., Kumar, P. S., Rashed, A. N. Z., & Saikumar, K. (2023).,*Deep Learning-Based Micro Facial Expression Recognition Using an Adaptive Tiefes FCNN Model*, Traitement du Signal, 40(3).

Edited by: Dhilip Kumar V
Special issue on: Unleashing the Power of Edge AI for Scalable Image and Video Processing
Received: Nov 17, 2023
Accepted: Apr 3, 2024