HYBRID SUPERCOMPUTING ARCHITECTURES FOR ARTIFICIAL INTELLIGENCE: ANALYSIS OF POTENTIALS

Jovan Popović[1], Vladisav Jelisavčić[2], Nenad Korolija[3]

[1] Microsoft Research and Development Center, Belgrade, Španskih Boraca 3/3, 11000 Belgrade, Serbia
e-mail: jovanpop@microsoft.com
[2] Mathematical Institute of the Serbian Academy of Sciences and Arts, Kneza Mihaila 36, Belgrade 11001, Serbia
e-mail: vladisavj@gmail.com
[3] School of Electrical Engineering, University of Belgrade, Bulevar Kralja Aleksandra 73, 11020 Belgrade, Serbia
e-mail: nenadko@etf.bg.ac.rs

Abstract:

Artificial intelligence incorporates scalable computing algorithms that are suitable for high performance computing architectures. Many computing architectures and paradigms are utilized for the purpose of accelerating execution of such algorithms: computer clusters, cloud computing, graphical processing units, dataflow programming, etc.

With so many available choices, one can find it hard to select appropriate artificial intelligence algorithm, computing paradigm, computer architecture, and the infrastructure. This work compares different computing paradigms, along with their architectures and programming models, and distinguishes their main advantages and disadvantages in terms of their flexibility, execution speed, scalability, development effort, price, and power consumption.

Additionally, hybrid supercomputing architectures that combine dataflow paradigm and von Neumann paradigm on the same chip are presented and compared to the previous ones. Analysis show that executing artificial intelligence algorithms on future supercomputing architectures can benefit from incorporating all the mentioned paradigms on the same chip die.

Keywords: artificial intelligence, high performance computing, dataflow programming, manycore architectures, graphical processing units.

1. Introduction

Artificial intelligence is in the focus of researchers for decades. During this period, many AI methods have been introduced. At the same time, advances in computer architectures have made it possible to utilize AI more than ever before.

The following section explains a multicore computer architecture based on von Neumann paradigm from the point of view of the capability to execute complex calculation for the purpose of AI. The manycore architecture is explained in the following section as a natural step forward to parallelizing scalable algorithms execution. Following section explains the dataflow paradigm and dataflow architectures from the point of view of the potential to parallelize execution of AI algorithms. Finally, a proposed hybrid processor that includes all aforementioned architectures is proposed. This work is presented using the proposed model [1].

2. Problem Statement

Advances in computer architectures fuel the evolution of artificial intelligence in many directions. The availability of large databases is constantly increasing. Recent improvements in deep learning methodology along with the performance of AI systems is reaching and, in some cases, even exceeding the human capacity on an increasing number of complex tasks. AI is in use in many fields, including classification, sentiment analysis, voice recognition, etc. [2].

The goal of this manuscript is to present current and possible approaches in advancing computer architectures so that the AI models could execute even faster, consume less electricity, and require less space for the computing nodes.

3. Multicore Architectures

Most commonly used computer architecture is based on von Neumann paradigm. The arithmetical logical unit (ALU) is responsible for executing instructions that have to be fetched from the main memory. The chip die consisting of around one billion transistors is organized so that it includes cache memories to account for relatively slow access to the main memory comparing to the speed of executing instructions. Cache memories could occupy around one half of the total number of transistors. The paradigm is also referred to as the control-flow paradigm, as micro-instructions that are performed during each clock cycle are controlled in such a manner that data is sent from one place to another only controlled by appropriate signals that will allow the data to exit from its source to the internal bus, and then to be accepted from the bus into the destination memory.

This type of architecture was used to accelerate with a rapid pace, doubling frequencies around each second year. However, once transistors became small enough, further reduction in the size would affect their physical properties, leading to unstable digital logic. At the same time, chip die is limited in size, as increasing the number of transistors would affect the size of the chip. Having in mind that signals travel through the chip with the speed of light, as well as that the clock cycle has to be stable during the execution of an instruction, increasing the size of the chip would imply reducing frequencies below 3GHz. Therefore, current trends in multicore architecture design is increasing number of cores, along with increasing capacities of cache memories.

4. Manycore Architectures

So-called manycore architectures, or GPUs, have been introduced as a logical successor of multicore architectures. In the early days of personal computers, mathematical processors were introduced to cope with requirements for faster processing. Later, a mathematical processor became an integral part of the CPU. With growing possibilities that have arisen with reducing the size of transistors, it was possible to build thousands of small processing cores on a single chip die. Computer graphics was the first field to benefit from this. Soon after, researchers and the industry found a way to utilize graphical cards for accelerating scalable algorithms.

GPUs proved to be able to accelerate many scalable algorithms, which let to utilizing them in simulating finite difference numerical algorithms. For example, the one for solving two dimensional spatially homogeneous Boltzmann transport equation is accelerated by around two orders of magnitude [3].

However, just like it is the case with multicore architectures, manycore architectures also suffer from the same bottlenecks, including complex control logic and internal bus bandwidth.

5. Dataflow Architectures

The Dataflow architectures overcome the limitations of control-flow architectures by

allowing data to flow through a hardware [4]. This way, instructions are executed in place, as i is the case with the production in a company conveyor belt. The advantages of the approach are well known, at least in the industry. This section explains dataflow paradigm by comparing dataflow architecture to the control-flow based processor.

The instruction that is executed on a relatively small part of a dataflow chip needs not much more transistors as needed by the instruction. In contrary, the control-flow architecture is highly-flexible, allowing any instruction defined by the architecture to be executed in any particular moment, which leads to complex ALU comparing to the logic needed for executing a single instruction. Therefore, using the dataflow paradigm, one can achieve higher number of instructions per seconds using the same number of transistors, even though the dataflow hardware runs at an order of magnitude lower speeds than the control-flow counterparts.

The data that travels from one place to the other needs no special internal bus connecting all parts of the chip where the calculation is performed with internal memories. Instead, wires forward outputs of an instruction to the input of the following. This can be compared to having many internal buses in a control-flow type of architecture, where each bus connects only two components in one direction. This eliminates control-flow architecture stalls due to waiting for the internal bus and memory to finish the task before executing the following one.

The lower clock cycle of the dataflow architectures comparing to control-flow architectures leads to lower power consumption. This furthers lowers the necessity for cooling supercomputing architectures that include dataflow processing units.

The disadvantage of the dataflow architecture is that it has to be preconfigured to execute a certain algorithm. As such, it is suitable for a subset of scalable algorithms that can be accelerated using the high-performance computing architectures. Dataflow architectures are usually implemented using the FPGAs, as this allows re-configuring an architecture once it is needed for executing another algorithm. However, this is the reason why dataflow architectures run at an order of magnitude lower speeds than control-flow architectures.

Another disadvantage of dataflow architectures is their inflexibility in terms of executing the whole algorithm that is scheduled for dataflow architecture, not just the part it is suitable for. As a result, dataflow architectures often come in the form of PCIe card, and the control-flow type of processor is responsible for preparing data for parallel processing using the dataflow hardware, re-configuring the hardware, streaming the data from the main memory onto the dataflow hardware, and collecting the results from dataflow hardware.

The dataflow programming model differs from the control-flow programming model [5]. More effort is needed for programming dataflow architectures [6], but also for testing algorithms and configuring dataflow architectures, even taught certain level of automation of these processes can be achieved [7].

To demonstrate possibilities of the dataflow paradigm, various algorithms are found in the open literature that are implemented using the same framework [8]. Table 1 demonstrates the speedup of sample algorithms accelerated using dataflow architectures comparing to the multicore processor of the same or similar generation.

Results	Algorithm	Speedup	Conditions
Stojanovic et al. [9]	Gross-Pitaevskii	8.2x	-
Kos et al. [10]	Odd-even merge network sort	100-150x	more than 1000 arrays
Korolija et al. [11]	Lattice-Boltzmann	17x	matrix dimensions 320x112
Stanojevic et al. [12]	Spherical code design	18-24x	3<=D<=6

Table 1. The speedups of algorithms implemented for dataflow paradigm

6. Hybrid Architectures

Having in mind both the advantages and disadvantages of dataflow computer architectures compared to computer architectures based on von Neumann paradigm, as well as the rising number of transistors in each generation of new processors, a challenge of merging the two paradigms on the same chip arises [14, 15]. The process of merging two components onto the same chip die is not new. In the early days of personal computers, math co-processors were built as computer chips that handled the floating-point operations and mathematical computations in computers. These chips were separated from the CPU. In today's computers, they are generally built into the CPU, while they still serve for the same purpose, allowing CPUs to execute mathematical computations on these chips.

In comparison to the hybrid processor proposed in previous work [14], this hybrid architecture is oriented towards accelerating algorithms, and is not related to internet of things.

Advantages of the proposed hybrid processor are accompanied by disadvantages in terms of the achieving relatively high utilization and the organization of execution of jobs. To completely utilize such a processor, one needs to have enough jobs that are dedicated for underlying hardware. This constraint is somewhat released by the fact that multicore processor is capable of executing algorithms that are most suitable for a dataflow hardware or a manycore computer architecture. However, utilization of proposed hybrid processors still requires scheduling that takes into account both control-flow type of jobs and dataflow jobs [16].

7. Analysis of the Proposed Architecture

The analysis of the potentials of the hybrid computer architecture will be explained on the communication between the multicore processor and the dataflow hardware, without affecting the generality, as the same conclusion might have been drawn for the communication between multicore processor and manycore computer architecture, but also the communication between dataflow hardware and manycore processor. The latest one might spawn a new set of scalable algorithms that can be accelerated by combining dataflow and manycore computer architectures.

Dataflow architectures find their place in executing two types of jobs. First assume processing the data that is already prepared on the dataflow hardware. This takes time to send over the PCIe slot, or any kind of network that exists between the dataflow hardware and the multicore processor. Second one assumes that the data is streamed over the network to the dataflow architecture, where the data gets processed, producing the result either in dataflow hardware local memory, or streaming the result back to the multicore processor. What is unique for these two is that the speed of communicating between multicore processor and the dataflow hardware affects the overall acceleration of algorithms. Therefore, we will compare the estimated speed of the communication between the multicore processor and the dataflow hardware in the case of hybrid computer architecture and in the case of dataflow card attached to the computer mainboard using the PCIe card slot.

Cache memories speed is an order or even two orders of magnitude faster than RAM, requiring nanoseconds to respond to a CPU request. There are usually three levels of cache in multicore processors, L1, L2, and L3. The last one is also used for sharing the data between CPU cores. PCIe interconnect to CPU with an order of 1GB/s. The speed of the L3 cache is around double of the speed of the RAM memory, and RAM is about one order of magnitude faster than what the PCIe in communication between CPU and a dataflow card can achieve. Therefore, it is obvious that the hybrid computer architecture has potentials to overcome one of the greatest problems of accelerating computer algorithms using the dataflow paradigm.

8. Conclusions

Current trends in designing chip dies is increasing the number of transistors on chip. With a limited possibility for accelerating multicore processors, a natural step forward is in increasing number of cores. However, further increasing the nubmer of cores requires scalable algorithms in order to keep the cores utilized. As some algorithms can achieve better speed-up using the CUDA programming model, while others achieve better performance in terms of exection time or power consumption using the dataflow paradigms, a natural step forward in designing modern processors i to combine these paradigms on the same chip die.

The integration of peripheral mathematical processors or memories in form of caches on the same chip where the processor is is not new. It proved to improve overal performances many times in the past.

Analysis of the speed of communication between CPU and the dataflow card proves that there is a potential for a range of dataflow algorithms to be accelerated using the hybrid processor. The same type analysis can be performed between any two of the underlying computer architectures of the hybrid computer architecture.

Authors of this manuscript believe that future computers will benefit from integrating multiple computing paradigms on the same chip die, as it can improve performance and reduce power consumption and the need for cooling at the same time.

Acknowledgment: VJ is partially supported by the Mathematical Institute of the Serbian Academy of Sciences and Arts. NK is partially supported by the School of Electrical Engineering, University of Belgrade, Serbia and by the Institute of Physics Belgrade, contract no. 0801-1264/1. VJ and NK are partially supported by the Ministry of Education, Science, and Technological Development of the Republic of Serbia.

References

[1] Milutinovic V, The best method for presentation of research results, IEEE TCCA Newsletter, 1-6, 1996.

[2] Yang LU, Artificial intelligence: a survey on evolution, models, applications and future trends. Journal of Management Analytics, 6(1):1-29, 2019.

[3] Priimak D, Finite difference numerical method for the superlattice Boltzmann transport equation and case comparison of CPU (C) and GPU (CUDA) implementations, Journal of Computational Physics, 278:182-192, 2014.

[4] Trifunovic N, Milutinovic V, Korolija N, and Gaydadjiev G, An AppGallery for dataflow computing, Journal of Big Data, 3(1):1-30, 2016.

[5] Milutinović V, Salom J, Trifunović N, and Giorgi R, Guide to dataflow supercomputing. Springer Nature, 10, 978-3, 2015.

[6] Popovic J, Bojic D, and Korolija N, Analysis of task effort estimation accuracy based on use case point size, IET Software, 9(6):166-173, 2015.

[7] Korolija N, Popović J, Cvetanović M, and Bojović M, Dataflow-based parallelization of control-flow algorithms, Advances in computers, 104:73-124, Elsevier, 2017.

[8] Trifunovic N, Perovic B, Trifunovic P, Babovic Z, and Hurson A. R, A novel infrastructure for synergistic dataflow research, development, education, and deployment: the Maxeler AppGallery project, Advances in Computers, Elsevier, 106:167-213, 2017.

[9] Stojanovic S, Bojic D, and Milutinovic V, Solving Gross Pitaevskii Equation using Dataflow Paradigm, Transactions on Internet Research, 9(2), July 2013.

[10] Kos A, Rankovic V, Tomazic S, Sorting Networks on Maxeler Dataflow Supercomputing Systems, Advances in Computers, vol. 96. Amsterdam, Elsevier, Academic Press, 139-186, 2015.

[11] Korolija N, Djukic T, Milutinovic V, and Filipovic N, Accelerating Lattice-Boltzman Method using the Maxeler DataFlow Approach, Transactions on Internet Research, 9(2):5-10, July 2013.

[12] Stanojevic I, Senk V, and Milutinovic V, Application of Maxeler Dataflow Supercomputing to Spherical Code Design, Transactions on Internet Research, 9(2):1-4, July 2013.

[13] Bezanic N, Popovic-Bozovic J, Milutinovic V, and Popovic I, Implementation of the RSA Algorithm on a DataFlow Architecture, Transactions on Internet Research, 9(2):11-16, July 2013.

[14] Milutinović V, Azer ES, Yoshimoto K, Klimeck G, Djordjevic M, Kotlar M, Bojovic M, Miladinovic B, Korolija N, Stankovic S, Filipović N, Babovic Z, Kosanic M, Tsuda A, Valero M, de Santo M, Neuhold E, Skorucak J, Dipietro L, and Ratkovic I, The ultimate dataflow for ultimate supercomputers-on-a-chip, for scientific computing, geo physics, complex mathematics, and information processing, 10th Mediterranean Conference on Embedded Computing, IEEE, 1-6, June 2021.

[15] Milutinović V, Kotlar M, Ratković I, Korolija N, Djordjevic M, Yoshimoto K, and Valero M, The Ultimate Data Flow for Ultimate Super Computers-on-a-Chip, Handbook of Research on Methodologies and Applications of Supercomputing, IGI Global, 312-318, 2021.

[16] Korolija N, Bojić D, Hurson AR, and Milutinovic V, A runtime job scheduling algorithm for cluster architectures with dataflow accelerators, Advances in computers, Elsevier, 126, 2022.