Assignment 1

Exploring the Past, Present, and Future of Parallel Computing

Brennan Reamer
Wentworth Institute of Technology

I. Introduction

The field of parallel processing has come a long way since the early days of supercomputers in the 1950s and 1960s. Today, parallel processing is widely used for scientific research, industrial simulations, and data analytics, as well as in mobile devices and personal computers. The need for increased performance with a reduction in power consumption led to the development of multicore processors and parallel processing, but these technologies require a focus on architectural considerations to optimize their performance. In this paper, the history of parallel processing, its evolution, and the various technological advances that have influenced its development are explored. Additionally, several historical predictions that have been made about the future of supercomputing and their impact on the field are examined.

II. The Evolution of Parallel Computing

The history of parallel processing can be traced back to the early days of computing, with the creation of the first supercomputers in the 1950s and 1960s. One of the most notable early supercomputers was the Cray-1, developed by Seymour Cray in 1976. The Cray-1 was a highly parallel system, employing multiple processors to perform calculations, making it capable of being the fastest system of its time, including both vector and scalar machines [1]. Cray Vector Machines, like the Cray-1, were the first popular example of parallel processing as they used an easier, friendlier vector programming paradigm over the traditionally-used general parallel programs [2].

As technology progressed, parallel processing and supercomputing continued to evolve. In the 1990s, clusters began to gain popularity as a way to perform large-scale calculations [3]. Clusters are composed of multiple interconnected computers that work together to perform a single task, causing them to be less expensive and more scalable than traditional supercomputers[3]. Today, clusters continue to be used in a wide variety of applications, including scientific research, industrial simulations, and data analytics [1].

The motivation for multicore processors, as well as parallel processing in general, can be attributed to the need for increased performance with a reduction in power consumption [2]. As technology progressed, it became clear that traditional single-core processors were constrained due to power consumption and overheating, thus the industry began to adopt the use of multiple cores within a single processor, in addition to clustering, to overcome these challenges [3].

Architectural considerations must be taken into account when designing and implementing parallel systems, as they play a crucial role in performance, memory, layout, and more. The application can make a significant difference in the multicore architecture used. For example, data processing requires high throughput for large amounts of data, while control processing requires a more general-purpose architecture because of its unstructured nature [2]. Data processing would be ideal for parallelism,

while control processing would make parallelism quite difficult.

Another important aspect of parallel processing and supercomputing is the use of accelerators, like Graphics Processing Units (GPUs) and Field Programmable Gate Arrays (FPGAs). Accelerators are specialized hardware devices that are designed to perform specific types of computations more efficiently than general-purpose processors. The use of accelerators in high performance computing (HPC) has become increasingly popular in recent years, with many HPC systems now including GPUs, or other types of accelerators, to improve performance. However, the use of accelerators also comes with its own set of challenges, such as programming complexity and data transfer bottlenecks [4].

The history of parallel processing has evolved from the early days of the Cray supercomputers to the current use of clusters and accelerators. Although parallel processing has improved scalability and performance, it requires a focus on architectural considerations, such as memory, layout, and scaling, to optimize the performance of the system.

III. Historical Predictions

Over the past several decades, there have been numerous predictions made about the future of supercomputing and high-performance computing. A prediction made in 2010 stated that the performance of microprocessors will stop increasing, causing a shift to cloud computing where servers can do the work and take advantage of parallelism [5]. This prediction has been proven correct as cloud computing is a large business today, with many companies using it for their computing. The same prediction states how economical systems may be created from multiple chips if Moore's Law were to come to an end in order to sustain the performance gains seen previously [5]. This is exactly how multicore processors are designed today, to ensure they continue their performance gains even though Moore's Law no longer holds true.

One prediction from 1998 states that parallel computing will become necessary in the future to resolve diminishing performance increases [6]. It is predicted that before 2050, an upheaval would occur in parallel computing that would resolve the communications problems present in the field at the time. These predictions have both been proven true as parallel computing is now integrated into every computing device. The communications problems faced back in the 1990's have been resolved in order to make parallel computing a viable solution to diminishing performance increases.

One prediction made in the early 2000s was the use of grid computing, where a network of distributed computers work together to perform a single task. This approach was seen as a potential solution for addressing the scalability and reliability issues of traditional supercomputers. While grid computing has seen some success in certain fields, such as bioinformatics and environmental modeling, it has not been widely adopted in commercial designs today. One reason for this is the difficulty in coordinating and managing large numbers of distributed computers [7].

The use of artificial intelligence (AI) in high-performance computing has been a topic of much discussion in recent years. Some experts predict that AI will play a significant role in the future of HPC, for example, in the optimization of energy efficiency, resource management, and scientific discovery. Today, AI has been used specifically to improve energy efficiency of HPC [8], and even to predict optimized treatment models for cancer patients [9]. Alongside HPC, AI is becoming widely used today throughout numerous industries.

Another prediction that has been made in recent years is the use of neuromorphic computing, which emulates the structure and function of the human brain to perform computing tasks [10]. Neuromorphic computing has the potential to significantly improve the efficiency and performance of HPC systems, but it is still a relatively new and developing field, and it

is unclear how widely it will be adopted in commercial systems.

In conclusion, predicting the future of supercomputing and high-performance computing is a challenging task. While some predictions made in the past have been accurate, such as the use of multi-core processors, others have not, such as the widespread adoption of grid computing in commercial designs. The use of AI alongside HPC is currently having a breakthrough in usage, while neuromorphic computing remains in the exploration phase. It remains to be seen how widely neuromorphic computing will be adopted in commercial systems.

IV. Current Technology: CUDA Processing

In the past, graphics processing units (GPUs) have been used only for graphics libraries such as OpenGL and Direct3D, but in recent years, as parallel processing has become more common, GPUs have become more general purpose. Compute Unified Device Architecture (CUDA) processing allows programmers to program general purpose GPUs (GPGPUs) without the need for mapping programs onto graphics APIs, easing access for programmers to the more powerful computing capabilities of a GPU, rather than a CPU [11].

CUDA hardware contains three novel features that allow it to operate at such a high speed: General write/read global memory, on-chip shared memory, and thread synchronization [11]. General write/read global memory means that the GPU is able to gather and send data to or from any location, rivaling the flexibility of a CPU. On-chip shared memory is vital in that it can allow threads located in the same multiprocessor to access a shared memory with an access speed just as quick as registers. Thread synchronization allows for threads to synchronize so they can resolve complex problems together. From the software standpoint, the GPU computing model works through the heterogeneous use of a CPU, executing the serial code, and a GPU, executing the computationally-intensive code [12]. This allows the applications to run faster by leveraging the performance of the GPU to it's fullest capabilities.

Due to the increased application speeds when using a GPGPU, programmers have begun rewriting applications to take advantage of both the task-parallelism associated with a CPU and the data-parallelism associated with a GPU. CUDA is most useful when executing highly parallel algorithms. It has no overall advantage over other architectures when the problem cannot be split up into at least a thousand threads, but performance increases rapidly as the thread count is increased over that minimum [12].

CUDA is now being utilized throughout the HPC field due to its increase in performance, allowing many difficult computations to finally become possible. The Weather Research and Forecasting (WRF) model in use at the National Oceanic and Atmospheric Administration (NOAA)'s national weather service has been massively improved through the use of CUDA. When rewritten into the C programming language with CUDA extensions (CUDA C) and executed on a single GPU, a reduction in processing time of 389x was realized [13]. Within the field of electromagnetic methods, another application of CUDA is the implementation of a Finite-Difference Time-Domain (FDTD) algorithm to achieve peak performance over the previous numerical methods [14]. Having advanced GPGPU parallel computing capability, CUDA has allowed desktop PCs to overcome the traditional bottleneck of the CPU [12].

Research is currently being done on how to optimize CUDA processing for compute- or memory-bound algorithms. In order to resolve a compute-bound algorithm, registers may be reduced through variable reuse and data throughput increased through heavier thread workloads, while memory-bound algorithms require the reorganizing of data into self-contained structures and using a multi-pass approach [15]. These optimizations resulted in

a 6x speedup over unoptimized CUDA implementations [15]. A different approach is to use memory space analysis, variance analysis, and memory access vectorization, in addition to loop unrolling, in order to increase CUDA performance [16].

V. Emerging Technologies: Neuromorphic and Quantum Computing

In order to enable continued improvement in computing as the end of Moore's Law approaches, two new technologies are being developed: neuromorphic computing and quantum computing. Both offer an opportunity to run highly parallel processes, allowing an increase in performance against the traditional computers in wide use today.

A. Neuromorphic Computing

Neuromorphic computing refers to computing systems inspired by the brain itself, such as the DARPA Synapse Project and the European Union's Human Brain Project. Similar to the brain, neuromorphic computers are composed of neurons and synapses, both of which perform and store memory, unlike a typical Von Neumann computer which separates the processor and the memory, causing a possible performance bottleneck for the system [17]. Neuromorphic computers also use spikes, encoding information within the magnitude, shape, and associated time of occurence, instead of the binary values used in today's Von Neumann computers [17]. Alongside performance, power is an important issue in today's computing environment. Neuromorphic computing is able to resolve the power issue, using significantly less power than a traditional computing system due to its event-driven and highly parallel nature. Work is only performed by the neurons and synapses when there are spikes to process, allowing for a much increased energy efficiency [17].

An example of the use of a neuromorphic computer is object detection within a robot. Testing was performed on the feasibility of a neuromorphic computer against a GPU-based implementation, resulting in a significant improvement in clutter removal during object detection for the neuromorphic computer [18]. When a neural-network was implemented on the neuromorphic computer, the instructions were performed over ten times faster than with the GPU-based alternative, indicating the increase in performance of AI and ML systems running on a neuromorphic computer [18]. Due to the highly parallel nature of the computing system, an increase in neurons does not affect performance, indicating the feasibility of running the neuromorphic system on a single board attached to the robot [18].

Currently, one of the largest issues for neuromorphic computing is that it performs roughly similar to deep learning approaches in terms of accuracy [17]. Research is being performed to improve the algorithmic accuracy before it is available commercially. Another issue currently being investigated is the reliance on traditional computers and optimization of communication between them. Neuromorphic computers rely on traditional computers for external device communications, like sensors and actuators, as well as for the defining of the software structure deployed to the computer [17].

B. Quantum Computing

Another example of an emergent technology within parallel computing and HPC is quantum computing, which relies on the principle of quantum entanglement. Quantum computers use qubits instead of normal binary bits, allowing for a greater amount of information to be stored as it is encoded in the entanglement of multiple qubits rather than in a single bit [19]. Quantum computing looks to solve problems that are classically hard but quantumly easy. Current quantum algorithms are able to solve certain classical problems with ease, such as finding the prime factors of large composite integers [19]. While ongoing research explores the physical implementation of qubits, the two

main technologies currently in use are trapped ions and artificial atoms created by superconducting circuits [20].

Quantum computing faces many challenges, such as noise rejection, quantum error correction, and efficient data inputs. Traditional computers operate on binary bits, allowing them to easily reject any noise that is not equal to a zero or one. Qubits can be any combination of one and zero, meaning they cannot easily reject noise or small errors in the system [20]. Quantum Error Correction (QEC) can be used to resolve this noise developed by the physical qubits, but it has its own drawbacks. QEC is too resource-intensive to be used until better technology is developed, meaning near-term quantum computers are likely to have errors [20]. There also lacks a reliable method to convert classical data to quantum data due to the need for a highly controlled and isolated input quantum state, which is not currently feasible to create [20]. Research is underway to develop Quantum Random Access Memory (QRAM) as a potential solution, but its practicality has not yet been established.

In order to advance qubit development, noisy quantum computers must achieve commercial success, much like how Moore's Law drove progress in conventional computing. [20]. An open community of research and development for the quantum computing domain would also be largely helpful in pushing forward the quantum computing technologies. Without these, the development of a reliable, large-scale quantum computer may be quite far away.

VI. Conclusion

In conclusion, the evolution of parallel computing has come a long way since the early days of supercomputers, with the industry adopting new technologies and architectures to improve performance and overcome challenges. The use of multicore processors, clusters, and accelerators has become increasingly popular in recent years, with many HPC systems now including them to improve performance. Over the past several decades, there have been numerous predictions made about the future of supercomputing and HPC. While some predictions have come true, like the use of multicore processors and the use of AI in HPC, some were not as fortunate, like the use of grid computing. Overall, the field of parallel computing continues to evolve and adapt to new challenges, with the focus on optimizing performance and reducing power consumption. Emerging technologies, such as neuromorphic and quantum computing, show promise for the future of parallel computing, although further development is necessary to fully realize their potential.

References

[1] E. Strohmaier, "20 years supercomputer market analysis," Lawrence Berkeley National Laboratory, Tech. Rep. LBNL-58442, 2005.

[2] G. Bell, "The next ten years of supercomputing," in *Proceedings 14th Supercomputer Conference*, H. W. Meuer, Ed., Mannheim, 1999.

[3] G. Bell and J. Gray, "High performance computing: Crays, clusters, and centers. what next?" Tech. Rep. MSR-TR-2001-76, August 2001. [Online]. Available: https://www.microsoft.com/en-us/research/publication/high-performance-computing-crays-clusters-and-centers-what-next/

[4] A. Bakhoda, G. L. Yuan, W. W. L. Fung, H. Wong, and T. M. Aamodt, "Analyzing cuda workloads using a detailed gpu simulator," *2009 IEEE International Symposium on Performance Analysis of Systems and Software*, pp. 163–174, 2009.

[5] D. Patterson, "The trouble with multicore," *IEEE Spectrum*, 07 2010.

[6] L. Trefethen, "Predictions for scientific computing fifty years from now," in *Numerical Analysis and Computers - 50 Years of Progress*, 06 1998.

[7] C. Dabrowski, "Reliability in grid computing systems," *Wiley InterScience*, 2009.

[8] A. H. Kelechi, M. H. Alsharif, O. J. Bameyi, P. J. Ezra, I. K. Joseph, A.-A. Atayero, Z. W. Geem, and J. Hong, "Artificial intelligence: An energy efficiency tool for enhanced high performance computing," *Symmetry*, vol. 12, no. 6, 2020. [Online]. Available: https://www.mdpi.com/2073-8994/12/6/1029

[9] T. Bhattacharya, T. Brettin, J. H. Doroshow, Y. A. Evrard, E. J. Greenspan, A. L. Gryshuk, T. T. Hoang, C. B. V. Lauzon, D. Nissley, L. Penberthy, E. Stahlberg, R. Stevens, F. Streitz, G. Tourassi, F. Xia, and G. Zaki, "Ai meets exascale computing: Advancing cancer research with large-scale high performance computing," *Frontiers in Oncology*, vol. 9, 2019. [Online]. Available: https://www.frontiersin.org/articles/10.3389/fonc.2019.00984

[10] P. Merolla, "Artificial brains. a million spiking-neuron integrated circuit with a scalable communication network and interface," *Science*, vol. 345,6197, 08 2014.

[11] Z. Yang, Y. Zhu, and Y. Pu, "Parallel image processing based on cuda," in *2008 International Conference on Computer Science and Software Engineering*, vol. 3, 2008, pp. 198–201.

[12] J. Ghorpade, J. Parande, M. Kulkarni, and A. Bawaskar, "GPGPU processing in CUDA architecture," *CoRR*, vol. abs/1202.4347, 2012. [Online]. Available: http://arxiv.org/abs/1202.4347

[13] J. Mielikainen, B. Huang, H.-L. A. Huang, and M. D. Goldberg, "Improved gpu/cuda based parallel weather and research forecast (wrf) single moment 5-class (wsm5) cloud microphysics," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 5, no. 4, pp. 1256–1265, 2012.

[14] D. De Donno, A. Esposito, L. Tarricone, and L. Catarinucci, "Introduction to gpu computing and cuda programming: a case study on fdtd," *Antennas and Propagation Magazine, IEEE*, vol. 52, pp. 116 – 122, 07 2010.

[15] D. Lee D, "Cuda optimization strategies for compute- and memory-bound neuroimaging algorithms," *Comput Methods Programs Biomed*, vol. 106(3), pp. 175–187, 06 2012.

[16] G. Chakrabarti, V. Grover, B. Aarts, X. Kong, M. Kudlur, Y. Lin, J. Marathe, M. Murphy, and J.-Z. Wang, "Cuda: Compiling and optimizing for a gpu platform," *Procedia Computer Science*, vol. 9, pp. 1910–1919, 2012, proceedings of the International Conference on Computational Science, ICCS 2012. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877050912003304

[17] C. Schuman, S. Kulkarni, M. Parsa, J. Mitchell, P. Date, and B. Kay, "Opportunities for neuromorphic computing algorithms and applications," *Nature Computational Science*, vol. 2, pp. 10–19, 01 2022.

[18] G. D'Angelo, A. Perrett, M. Iacono, S. Furber, and C. Bartolozzi, "Event driven bio-inspired attentive system for the icub humanoid robot on spinnaker," *Neuromorphic Computing and Engineering*, vol. 2, no. 2, p. 024008, may 2022. [Online]. Available: https://dx.doi.org/10.1088/2634-4386/ac6b50

[19] J. Preskill, "Quantum Computing in the NISQ era and beyond," *Quantum*, vol. 2, p. 79, Aug. 2018. [Online]. Available: https://doi.org/10.22331/q-2018-08-06-79

[20] E. National Academies of Sciences, D. Sciences, I. Board, C. Board, C. Computing, M. Horowitz, and E. Grumbling, *Quantum Computing: Progress and Prospects*. National Academies Press, 2019. [Online]. Available: https://books.google.com/books?id=jjiPDwAAQBAJ