From Computer Architecture to Brain Architecture

ABSTRACT

Traditional computer architectures cannot scale infinitely. The solution is not to put more cores in a chip, but rather to arrange the cores differently. This paper presents an overview of the challenges faced by traditional computer architectures and suggests a brain-inspired architecture. By comparing the cores and dendrites, and networks in the brain and in computers, this paper provides a unique quantitative view on how superior the human brain is. It provides unique performance metrics and comparisons between the brain and supercomputers. Lastly, it proposes how to get from computer architectures to brain architectures.

Categories and Subject Descriptors

C.1.1 Single Data Stream Architectures; C.1.2 Multiple Data Stream Architectures; C.1.4 Parallel Architectures; C.1.3 Other Architectures

General Terms

Measurement, Performance, Design, Theory.

Keywords

Parallel architectures, Brain-inspired architectures, SIMD, MIMD.

1. INTRODUCTION

Computer engineers and researchers of other disciplines have traditionally sought to make computers faster, and they have found multiple ways to do so: 1) By making transistors perform more operations per second, and 2) by making computer systems perform more instructions or tasks simultaneously. Computer architects focus on 2), but research suggests that the ultimate solution is not hidden in digital design, but literally in our brains.

Over the last 40 years, inspired by the principles of digital design and other subjects, scientists and architects have proposed multiple solutions to enable computers to perform more tasks simultaneously. By following the principles of assembly line, pipelining was proposed. Although such approach is present in most, if not all commercially available processors, its benefits reached a plateau. Thus, research moved to a different direction.

In the 1970's, the idea of increasing the number of processors to perform computations in parallel took momentum. SIMD was born with the objective of performing computations on multiple, identical data elements simultaneously [7][9]. Since SIMD was not flexible, MIMD was proposed. However, the byproduct of this flexibility is extra latency, which makes this approach less scalable [5].

In addition, scientists such as Gene Amdahl started questioning how beneficial parallel computing is. In fact, he found that there is a limit to how much parallel computing speeds up programs, which depends on the program to be run. Therefore, researchers have started investigating other architectures that successfully perform complex operations in parallel, such as the human brain.

The human brain is capable of outperforming computers at tasks that require tremendous amounts of computing. One of these tasks is image-content analysis. For instance, let us consider a picture that has fruits. A person would take only a few seconds in order to know how many apples, pears, and bananas there are in the picture. A computer, on the other hand, would not be able to do so accurately. For instance, it would confuse green apples and pears, or count one fruit as two, etc.

To justify this particular case, some argue that people outsmart computers at this kind of task because we possess human intuition. Some argue that human intuition cannot be replicated in a machine because machines are not humans. However, our intuition is a complex architecture that involves neurons and the neurocortex [3]. If we modeled neurons as processors and the neurocortex as a memory structure, we should be able to implant human intelligence in a machine [3][8].

This paper will summarize scalability issues with pipelining, SIMD, and MIMD architectures. Then, it will compare neurons with equivalent processing units available in computers. Lastly, it will propose a new metric for performance evaluation that intends to foster novel, brain-inspired computer architectures.

2. SCALABILITY AND COMPLEXITY

According to [4], scalability is the property of a sequential system to increase its throughput once it is turned into a highly parallel system. Efficiency can be computed as the ratio of speedup to the number of processors used. When efficiency is equal to 1, the system is said to be scalable.

This definition fails to acknowledge the fact that workload plays a role in scalability. According to [8], scalability is the ability of a system to efficiently handle size-varying workloads. In other words, scalable systems should be able to use more resources as new resources are added or less resources as resources are removed.

The author of this paper sees scalability as a combination of both of the previous definitions. He defines scalability as the ability of a system to effectively use most, if not all, of its resources simultaneously in order to operate on a size-varying and type-varying problem.

This definition combines performance, parallelism, and the type of workloads to operate on. Depending on the type of workload, there may be some parallelism, and there are strategies to leverage those kinds of workloads.

2.1 Instruction-Level Parallelism (ILP)

When trying to leverage ILP, pipelines can be scaled in multiple ways. One way is to increase the number of pipeline stages so that the number of overlapping instructions is higher [7].

Unfortunately, there is a point in which increasing the stages of a pipeline does not increase throughput. In theory, clock frequency increases as the number of stages increases because each stage requires less time. However, loading values from registers and pipeline latches increases pipeline overhead.

Issuing multiple instructions at the same time in a pipeline is another way to scale pipelines. In regular pipelines, assuming no stallings or other errors, one instruction is finished every cycle (1 IPC) [7]. In multiple-issue pipelines, more than one instruction

can be finished per cycle. This is achieved by adding more hardware components to process the additional instructions. Today's processors can issue no more than 6 IPC because this technique largely increases area and power consumption [7].

2.2 SIMD

Another strategy to achieve faster computing is to leverage from data-level parallelism. SIMD architectures focus on performing the same operation on different data simultaneously [7][9].

The main disadvantage of SIMD is that it is not flexible: it is useful only when data is highly parallel (i.e. matrix operations). Because there is a single, simple control unit, all the execution units perform the same operation at the same time, which makes asynchronous operations very difficult [7]. To run asynchronous programs, extensive runtime support is needed. The compiler needs to break down the program so that only one instruction is executed at a given time.

2.3 MIMD

An alternative to SIMD is MIMD. The fundamental difference between SIMD and MIMD is that MIMD allows parallel units to execute different instructions. Thus, MIMD architectures are more flexible when it comes to running asynchronous programs [7, 5].

MIMD architectures can use a shared memory or a distributed memory. These memory layouts are needed because they help processors communicate. Nevertheless, they both have limitations.

In the shared memory model, all the processors are connected to a large, global memory. This memory can be accessed by buses, which give processors control of the memory. Unfortunately, the number of buses is limited, which means that not all of the available processors can be operating at the same time [5]. In the distributed memory model, every processor has its own memory, and they communicate among themselves by sending messages to each other. The disadvantage of this system is that sending these messages is expensive [5, 7].

Even if all the technologies described previously were flawless, adding more processors to a system might not be the solution. According to Amdahl's law, speedup the speedup of a program depends on its sequential portion and its parallel portion. According to Figure 1, for a program whose sequential and parallel portions are 95% and 5% respectively, speedup is 20 once the system has over 4096 processing units.

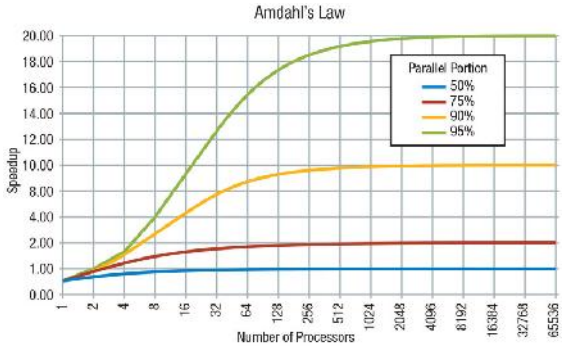

Figure 1: Amdahl's Law

Because speedup is ultimately limited by sequential code, maybe the answer is not to add more processors, but rather, to arrange them differently. Neuroscience has attracted the attention of researchers because it seeks understanding the human brain.

2.4 HUMAN BRAIN

Just as in computers, there are countless of connections in the human brain. It is estimated that humans possess 10^{11} neurons, and each neuron is connected to up to 10^4 neurons to which it can send information, forming almost 10^{15} connections, which are called synapses.

To understand the implications of this, let us consider an example. It is known, that a synapse fires every 10 ms. Let us imagine that a human takes 1 second in order to accurately understand the context of an image. Thus, it takes 100 synapses to analyze an image, and almost 10^{17} synapses occur every second. On the other hand, a modern high-end computer takes over 1 second in order to perform the same task, while using all of its resources. Hence, we can conclude that the human brain is 10^{15} times more powerful than a modern computer.

In addition, neurons are connected to the neurocortex. The neurocortex is a memory structure that is considered the source of human intelligence. It stores patterns that neurons retrieve when we need to perform any kind of activity.

Although computers are not as complex as the human brain, they may already be similar. MIMD architectures contain processors that are connected to memory structures, just like neurons are connected to the neurocortex. The next step to get closer to brain architecture is to effectively connect a large number of processors.

3. ABSTRACTION COMPARISONS

There are many misconceptions on how the brain compares to computers, which motivates the authors of this paper to provide their own comparisons. There are some who compare neurons with transistors, and some others see computers as equivalent to brains. It is widely accepted that these are not appropriate comparisons. Unlike many other authors, we will not compare processors with neurons not only because neurons are memory elements, but also because there are other elements in the brain that serve the purpose of processors. Rather, we will compare neurons with networks because they exhibit similar behavior.

3.1 PROCESSING ELEMENT

Batcher defines a processing element as a unit that contains a RAM and arithmetic and logic units that allow for transferring data and performing logic and arithmetic operations on the data [1].

Computers fit in the previous definition very easily, but computing elements in the brain do not. The main reason is that Batcher's definition is heavily tied to digital design, which excludes elements that do not have ALUs, registers, etc. This conflicts with computing in the brain because computing in the brain is done by electrochemical interactions. To address that issue, this paper will define a processing element as a unit that has the components necessary in order to modify, transfer, and make decisions based on data.

In the human brain and in computers, we can find elements that fit our definition of processing element. Thanks to this definition, elements to modify data through electrochemical reactions and through logic blocks are considered computing elements. We suggest that the computing elements that perform these kind of processes are the cores and the dendrites in computers and in the human brain respectively.

The authors of [15] define a core as a computing element that reads in instructions in order to perform operations, which are carried out using logic blocks. This is a widely accepted definition that has not changed much since its introduction in the 1980's. Therefore, we will synthesize the definition of [15] along with our definition of processing element to define a core. We will define a core as the basic building block of parallel architectures: they are computing elements equipped with logic components, which enable processors to transfer data, perform operations, and make decisions based on data.

Dendrites are commonly defined as the input ports of neurons. According to [2], dendrites are in charge of processing and propagating input signals to the body of their neuron. Dendrites work as low-pass filters whose attenuation depend on their diameter and the chemical elements on their walls. These materials could be Ca^{+2}, K^{+}, or Na^{+2}, which react electrochemically to the signals and alter the signals' frequency and amplitude [14]. Before signals leave the dendrites, they pass through NMDA (N-methyl-D-aspartate) transmitters, which tag the signals and encode the action potentials of the signals in the firing and connection pattern of the neuron [12, 14]. Further, Dendrites are connected to axons, other dendrites, glands, muscles, senses, and even the neurocortex. Thus, they are the core elements of human sensory and cognitive capabilities [11].

In this work, we define dendrites as the fundamental building blocks of parallel computing in the brain: They are computing elements whose mission is to analyze, propagate, and make decisions based on electrochemical signals. Also, they are responsible for programming and reconfiguring synapses through NMDA transmitters. Lastly, they are the interface between our senses, organs, and body parts, and the neurocortex.

The main difference between dendrites and processors is that dendrites can reconfigure their connections, whereas processors cannot. Depending on the environment, workload, and even injuries, the chain of dendrites that connects our senses to the neurocortex changes. For instance, when someone performs a task repeatedly, he or she gradually performs the task more rapidly or efficiently because the dendrites involved with this task rearrange themselves to reduce the number of connections and thus, the overhead [11]. In computers, connections at the core level are static.

3.2 COMPUTER NETWORK

In [16], a computer network is defined as a set of independent computing elements that are interconnected to achieve resource sharing, with the objective of allowing a computing element to call on another computing element's resources. According to [6], in order for computer networks to exist, there must be nodes (i.e. processors, sensors), links (connections), and protocols (pre-agreed upon conventions between the sender and the receiver).

In this paper, we define a computer network as a unit that ensures resource sharing, cooperation, and parallelism by propagating data among its nodes through its links.

Leaving software aside, a basic computer network can be composed of cores (nodes) and buses, which contain wiring and encoding and decoding hardware (links and protocols). An example of this is a datacenter, which may contain hundreds of thousands of cores connected via buses and additional hardware.

There is an equivalent to computer networks in the human brain. We argue that this equivalent is the neuron, whose processing elements are the dendrites and whose links and protocols are in the axon.

Neurons are cells that process and transfer information in the form of electrochemical signals [2]. Neurons contain thousands of dendrites, which receives and processes information; the axon, which transfer information to other neurons; and the soma, which contains a cell nucleus that serves as a memory unit of 1 Gb that is controlled by the dendrites [19].

The axon is in charge of transferring signals to other neurons, body organs, glands, or muscles [4]. Once signals enter the axon, they are amplified and separated into different channels that have different frequency, amplitude, and tag requirements. These channels are made of distinct biochemical materials and proteins (i.e: Ca^{+2}, Na^{+2}, K^{+}, etc.) that can make connections with compatible dendrites, glands, muscles, or body parts [14]. Thus, every signal is sent via the right channel to the right destination.

In this paper, we define neurons as massively parallel networks that merge, store, and share the information processed by the dendrites. This information is spread across other neurons and regions of the body through the axons.

We define the axon as the output port of neurons: it is a component managed by the dendrites, which is in charge of configuring channels that are coherent to other neurons so that communications can be established and information can be transferred.

Axons and buses have certain similarities and differences. They both provide connectivity, propagate signals, and follow a protocols governed by the processing units. Nevertheless, the ramifications at the axon are reconfigurable, whereas the connections of the buses are not.

Neurons and computer networks are similar in that they both allow their processing elements to call on another processing element's resources. For instance, a datacenter's core can request or act on the information of another core. Similarly, dendrites within the same neuron can form connections. This phenomenon can be observed when writing to the memory of the neuron. Dendrites connect to other dendrites and even themselves in feedback loops in order to store patterns in the neuron [7].

One of the differences is that load balancing in computer networks is different from that of neurons. In datacenters, servers, and clusters, there is a processing element that computes how the workload should be distributed among the cores. This is usually computed based on traffic [13]. In neurons, the dendrites themselves are in charge of computing load distribution, based on the signals received during synapse. This information goes through the NMDA transmitters of the dendrites, which configure how the information will be sent through the axon so that each signal is processed by the dendrites of other neurons [12].

Neurons are small networks whose performance becomes impressive once they are connected in networks with other neurons. A dendrite can compute 10^5 FLOPS [20], and a neuron contains up to 10^4 dendrites. Hence, a neuron can compute up to 10^9 FLOPS. This level of performance can be found in an Intel i7 processor, which is not as impressive. However, the authors of [5] estimate that the processing power achieved by the brain is about 3 orders of magnitude faster than fastest supercomputer, when all the neurons in the brain are connected. Let us consider the visual cortex, which is the area of the brain that analyzes visual information. It contains approximately $5x10^9$ neurons [21], which means that it can perform $5x10^{18}$ FLOPS. In contrast, the most

powerful supercomputer in the world, Tianhe-2, can perform $3x10^{16}$ FLOPS [22]. From this perspective, human vision alone is 2 orders of magnitude faster than Tianhe-2, and the human brain is, at least, 4 orders of magnitude faster than Tianhe-2.

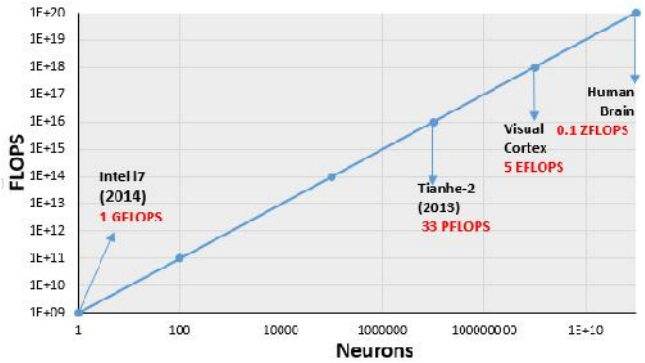

Figure 2: FLOPS in the human brain compared to FLOPS in computers.

Although it is widely accepted that the brain is 1000 faster than Tianhe-2, we argue that the difference is even larger. Tianhe-2 is a massively parallel computer, but it does not adapt to workloads the way the brain does. Let us consider a case where Tianhe-2 and a person are analyzing the context of images that are similar to one another, and let us assume it takes 10^{18} floating point operations to understand the image accurately. Thus, the person initially takes 1 s whereas Tianhe-2 takes 100 s, assuming all of Tianhe-2's cores are running, which is debatable [21]. As the task is repeated, Tianhe-2 keeps its runtime of 100 s, whereas the brain of the person continually reduces the runtime: after repeating the task for a duration of time, the dendrites of the visual cortex will rearrange themselves and modify the memory of neurons and the neurocortex to perform the task more efficiently, which can decrease runtime to a fraction of a second [11].

If we take this example to an extreme, we may find that brains are much more powerful than the best supercomputers. Let us assume now that the only function of this brain is to analyze these type of images. As the task is repeated, neurons of other areas of the brain will connect to the visual cortex, and will eventually connect more efficiently to achieve minimum runtime. Consequently, as of this submission, we believe that 10^4 is but the lower limit of the brain's superiority to supercomputers because adaptability to the workload is very important: it is the principle of scalability.

4. REFERENCES

[1] Batcher, K. E. (1982). *U.S. Patent No. 4,314,349*. Washington, DC: U.S. Patent and Trademark Office.

[2] Bear MF; Connors BW; Paradiso MA (2001). *Neuroscience: Exploring the Brain* (2nd ed.). Philadelphia: Lippincott Williams & Wilkins

[3] Clark, A. Whatever next? Predictive brains, situated agents, and the future of cognitive science. *Behavioral and Brain Sciences*. 36, 03 (2013), 181-204.

[4] Eyal, G., Mansvelder, H. D., de Kock, C. P., & Segev, I. (2014). Dendrites impact the encoding capabilities of the axon. *The Journal of Neuroscience*,*34*(24), 8063-8071.

[5] Furber, S. (2012). To build a brain. *IEEE Spectrum*, *49*(8), 44-49.

[6] Gavish, B. (1992). Topological design of computer communication networks—the overall design problem. *European Journal of Operational Research*, *58*(2), 149-172.

[7] Hawkins, J., and Blakeslee S. *On intelligence*. Macmillan, 2007

[8] Herbst, N. R., Kounev, S., & Reussner, R. (2013). Elasticity in Cloud Computing: What It Is, and What It Is Not. In *ICAC* (pp. 23-27).

[9] Hill, M. 1990. What is scalability?. *ACM SIGARCH Computer Architecture News*. 18, 4 (1990), 18-21.

[10] Huiskamp, W., R. Kiel, and M. H. Smit. HPC takes the Bus!. *Transputer Applications and Systems' 93: Proceedings of the 1993 World Transputer Congress*. 20-22 September 1993, Aachen, Germany. IOS Press, 1993.

[11] Kami, A., & Sagi, D. (1993). The time course of learning a visual skill. *Nature*,*365*(6443), 250-252.

[12] Larsen, R. S., Smith, I. T., Miriyala, J., Han, J. E., Corlew, R. J., Smith, S. L., & Philpot, B. D. (2014). Synapse-specific control of experience-dependent plasticity by presynaptic NMDA receptors. *Neuron*, *83*(4), 879-893.

[13] Mahapatra, S., & Yuan, X. (2010, September). Load balancing mechanisms in data center networks. In *the 7th Int. Conf. & Expo on Emerging Technologies for a Smarter World (CEWIT)*.

[14] Papoutsi, A., Kastellakis, G., Psarrou, M., Anastasakis, S., & Poirazi, P. (2014). Coding and decoding with dendrites. *Journal of Physiology-Paris*,*108*(1), 18-27.

[15] Patterson, David A., and John L. Hennessy. *Computer organization and design: the hardware/software interface*. Newnes, (2013).

[16] Roberts, L. G., & Wessler, B. D. (1970, May). Computer network development to achieve resource sharing. In *Proceedings of the May 5-7, 1970, spring joint computer conference* (pp. 543-549). ACM.

[17] Shu, W and M. Wu (1995), Asynchronous problems on SIMD parallel computers, *IEEE Trans. Parallel Distrib. Syst.*, vol. 6, no. 7, pp. 704-713

[18] Sterling, Thomas, Maciej Brodowicz, and Timur Gilmanov. *Towards Brain-Inspired System Architectures. Brain-Inspired Computing*. Springer International Publishing, (2014). 159-170.

[19] Smith, S. L., Smith, I. T., Branco, T., & Häusser, M. (2013). Dendritic spikes enhance stimulus selectivity in cortical neurons in vivo. *Nature*, *503*(7474), 115-120.

[20] Waldrop, M. 2012. Computer modelling: Brain in a box. *Nature*. 482, 7386 (2012), 456-458.

[21] Wandell, B. A., Dumoulin, S. O., & Brewer, A. A. (2009). Visual cortex in humans. *Encyclopedia of neuroscience*, *10*, 251-257.

[22] Zhang, X., Yang, C., Liu, F., Liu, Y., & Lu, Y. (2014). Optimizing and scaling HPCG on tianhe-2: Early experience. In *Algorithms and Architectures for Parallel Processing* (pp. 28-41). Springer International Publishing.