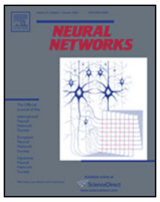
Neural Networks 24 (2011) 291–304

Contents lists available at ScienceDirect

Neural Networks

journal homepage: www.elsevier.com/locate/neunet



Performance/price estimates for cortex-scale hardware: A design space exploration

Mazad S. Zaveri [*], Dan Hammerstrom [1]

Department of Electrical and Computer Engineering, Maseeh College of Engineering and Computer Science, Portland State University, Portland, OR, United States

ARTICLE INFO

Article history:
Received 16 November 2009
Accepted 15 December 2010

Keywords:
Virtualization
Time-multiplexing
Spiking neuron
Processing node
Hardware
Digital
Mixed-signal
Performance
Price
Efficiency
Design-space
Hardware-spectrum

ABSTRACT

In this paper, we revisit the concept of virtualization. Virtualization is useful for understanding and investigating the performance/price and other trade-offs related to the hardware design space. Moreover, it is perhaps the most important aspect of a hardware design space exploration. Such a design space exploration is a necessary part of the study of hardware architectures for large-scale computational models for intelligent computing, including AI, Bayesian, bio-inspired and neural models. A methodical exploration is needed to identify potentially interesting regions in the design space, and to assess the relative performance/price points of these implementations. As an example, in this paper we investigate the performance/price of (digital and mixed-signal) CMOS and hypothetical CMOL (nanogrid) technology based hardware implementations of human cortex-scale spiking neural systems. Through this analysis, and the resulting performance/price points, we demonstrate, in general, the importance of virtualization, and of doing these kinds of design space explorations. The specific results suggest that hybrid nanotechnology such as CMOL is a promising candidate to implement very large-scale spiking neural systems, providing a more efficient utilization of the density and storage benefits of emerging nano-scale technologies. In general, we believe that the study of such hypothetical designs/architectures will guide the neuromorphic hardware community towards building large-scale systems, and help guide research trends in intelligent computing, and computer engineering.

1. Introduction

In spite of the transistor bounty of Moore's law, there is a large class of problems that computers still do not solve well (Gao & Hammerstrom, 2007; Hammerstrom, 2008). These "boundary problems" involve the transformation of data across the boundary between the real world and the digital world (Hammerstrom, 2008). Computer vision, speech recognition, and robotics are examples of these kinds of problems, which occur wherever a computer is sampling and acting on real world data (Hammerstrom, 2008).

Although there has been progress in all approaches to intelligent computation, we still do not have robust solutions that even remotely approach the capabilities of biological systems (Hammerstrom, 2008). Consequently, a number of researchers are returning to neuroscience in search for inspiration for new kinds of algorithms for intelligent computing. In collaboration with the neuroscience community, several groups are looking to create increasingly more sophisticated abstract models of neural circuits, and then apply these models to real applications.

Many of these groups[2] are also studying hardware/software implementations of large-scale networks inspired by neuroscience, which may provide a better understanding of the high-level computational principles of the cortex (Ananthanarayanan & Modha, 2007; Johansson & Lansner, 2007; Markram, 2006; Schemmel, Meier, & Mueller, 2004). In addition, such large-scale implementation platforms allow the testing of certain hypotheses related to cortical theories (Ananthanarayanan & Modha, 2007; Mehrtash et al., 2003; Schemmel, Fieres, & Meier, 2008).

As a result, the hardware implementation of large-scale neural networks is an excellent candidate application for the high density computation and storage possible with current and emerging semiconductor technologies (Beiu, 2007). In addition, such technologies may lead to new computing architectures,

* Corresponding address: c/o Dan Hammerstrom, 1930 SW Fourth Avenue, Suite 500, Portland, OR 97201, United States. Tel.: +1 503 484 5796.
E-mail addresses: mazadic@gmail.com (M.S. Zaveri), strom@cecs.pdx.edu (D. Hammerstrom).

1 Postal address: 1930 SW Fourth Avenue, Suite 500, Portland, OR 97201, United States. Tel.: +1 503 725 5125; fax: +1 503 725 2825.

2 DARPA's SyNAPSE program, which is currently in progress, is probably the most ambitious project so far; involving the development of cortex-scale neuromorphic hardware systems, using nanoelectronics based novel components.

0893-6080/$ – see front matter © 2010 Elsevier Ltd. All rights reserved.
doi:10.1016/j.neunet.2010.12.003

as we learn more about the operation of basic neural circuits (Ananthanarayanan & Modha, 2007).

Within the general area of silicon architectures for emulating these models, in this paper we investigate the performance/price trade-offs for various silicon implementations of human cortex-scale spiking neural network. Part of the motivation for this work is that the field of neuromorphic hardware has, so far, concentrated only on a limited region of the design/virtualization spectrum, that is, primarily the end-points of the virtualization spectrum, as is discussed below. And it has been argued that the end-points constitute the most cost-effective regions of the design space. In this paper we hope to motivate a wider range of analysis in exploring architectural trade-offs across the spectrum.

This paper also highlights some important issues related to large-scale implementation of these models and the complex relationships between analog, mixed-signal, and digital implementations as well as between standard CMOS and the most promising nano-scale devices, and nanogrid structures such as CMOL (Likharev & Strukov, 2005). Another very important issue that is not addressed here, but will be investigated in future papers is the basic cost-performance trade-offs in very neural like models versus more highly abstracted building blocks, such as discussed in the work of Albus (2008), George and Hawkins (2005) and Zaveri and Hammerstrom (2010).

Neural models are massively parallel so there is a significant opportunity for a huge range of virtualization, much more so than with traditional computational models (Gao & Hammerstrom, 2007). Since it promises the highest performance, one would think that a direct implementation, i.e., hardwiring the algorithms directly into silicon with little or no virtualization (Gao & Hammerstrom, 2007), would be the most cost-effective. However, depending on the dynamic behavior of the models, it is possible for much of the hardware to be idle most of the time (Gao & Hammerstrom, 2007). Consequently, factors such as differential hardware costs and model dynamics have a significant impact on the virtualization "sweet spot".

It has been shown that even with connectivity patterns that are much sparser than cerebral cortex, the use of dedicated metal lines for each connection quickly overwhelms the available connection resources in a typical silicon process (Bailey & Hammerstrom, 1988). Consequently the analog-VLSI (aVLSI) neuromorphic engineering community, which relies extensively on direct analog computation, uses multiplexed communication generally in the form of the Address Event Representation (AER) (Boahen, 2000; Schemmel et al., 2008). It has also been shown that for long range connections, storing addresses in a RAM is more efficient then dedicating a metal line to each connection (Bailey & Hammerstrom, 1988).

In our quest to create platforms for implementing various aspects of intelligent computing, the question posed by the work presented here is whether other aspects of the computational model can also be multiplexed to improve performance/price. To begin to answer this question, this paper presents an architectural space exploration for a certain family of massively parallel computational models, where, given a certain set of hardware options, and a model that has a certain structure and computations associated with it, including certain dynamic characteristics, what is a reasonably close, but not necessarily optimal, hardware configurations, and where does those configurations lie on the virtualization spectrum? We will not be using the term "optimal" since it is difficult to prove hardware implementations to be optimal—and will use the term "sweet spot" instead.

In this paper, Section 2 discusses the concept of virtualization, shows a simple example, and presents the "virtualization spectrum". Section 3 presents the neural model that is being implemented. Section 4 discusses various hardware design configurations for a processing node, and its performance/price analysis. And Section 5 presents the results of the performance/price analysis, summarizes them, and presents some general conclusions.

However there are some caveats that need to be discussed first:

– Our computational model is not special, we assumed that it was reasonably cortical-like, but many assumptions can be questioned; some of these assumptions and results, for example, may not be applicable to "front end" vision processing models, where activation may be less sparse, and communication is more local (Hammerstrom, 2008).
– Our hardware assumptions are not special, we made reasonable assumptions about existing, or relatively near-term projected CMOS and the existence of a reasonably implementable CMOL-like nanogrid structures.
– The specific architectural results of the analysis only hold for the specific model and hardware assumptions that were made here—different assumptions will change the results and the location of the sweet spot on the virtualization spectrum. In some cases, the results are only ballpark estimates, and should not be considered as a comprehensive judgment.
– The goal of this work was to define and present the concept of virtualization, and then do an architectural analysis based on that concept. Our hope is that this type of work will be useful across a broader selection of models and hardware structures. As far as we know, there has been no discussion of such trade-offs in this design space and no similar study of the hardware implementation of biologically inspired models that covers as broad a range of hardware structures. What has been more common were proposals for hardware structures for emulating biologically inspired computing that have not had the benefit of the kind of scaling analysis proposed here. We also believe that more analyses such as these are needed to guide research in models, hardware structures, and architecture.

As computational models and nano-scale computing structures evolve, we intend to regularly redo this analysis to factor in such developments.

2. Virtualization

2.1. A simple example of virtualization

An important issue in implementing biologically inspired models in silicon is the degree of virtualization utilized by the underlying hardware. Virtualization is defined as "the degree of time-multiplexing of the 'components' of computation and communication via hardware resources" (Gao & Hammerstrom, 2007; Gao, Zaveri, & Hammerstrom, 2008). The term, as used here should not be confused with virtual machines as currently used in the IT industry.

A simple example to demonstrate the concept is shown in Fig. 1, where the goal is to architect a hardware system for emulating 100 neurons. Assume a simple model for the neuron given by $y_i(t) = \sum_{\forall j} w_{ij}\varepsilon_{ij}(t - t_j)$, where y_i is the output of neuron i, $\varepsilon_{ij}(t - t_j)$ is the postsynaptic potential at input *j*, and w_{ij} is the synaptic weight linking input *j* to neuron *i*.

The basic arithmetic operations in this model are multiplication and addition (sometimes referred to as multiply-accumulate). When implementing these operations in digital hardware, we would require one digital adder and one digital multiplier, which we refer to as a Multiply-Accumulate Component (MAC). A simple implementation would be where each neuron uses one MAC, resulting in a total of 100 such MACs, as shown in Fig. 1(b). For explanatory purposes this is assumed to be a minimally virtualized implementation, an absolute minimum virtualization implementation would have a multiplier at each synapse and an accumulator at each branch in the dendritic tree. And, in fact, using