Exploiting Heterogeneous Parallel Programming for Developing an Educational Neuromorphic Computing Simulator

Faraz Hussain
fhussain@eecs.ucf.edu

Alvaro Velasquez
velasquez@eecs.ucf.edu

Sumit K. Jha
jha@eecs.ucf.edu

Department of Electrical Engineering and Computer Science, University of Central Florida (UCF)

ABSTRACT

With the explosive growth of multicore and GPU-based computing, accompanied by the long-predicted demise of Moore's law, there has been an increasing interest in expanding the breadth and scope of undergraduate instruction in concurrent and distributed computing. We describe our plans for exploiting heterogeneous parallel computing architectures for developing an educational simulator for *neuromorphic computing*. We also report on our experience as early adopters of the NSF/IEEE-TCPP CDER curriculum on Parallel and Distributed computing for Fall 2014 at the University of Central Florida. We include a brief overview of how our courses cover the theoretical and practical aspects of parallel computing, and describe the hardware and software infrastructure used for instruction and research at our institution.

OBJECTIVES

- Develop a neuromorphic computing simulator.
- Incorporate use of the simulator into upper-level undergraduate courses.
- Introduce parallel complexity theory to the graduate-level complexity course taught at UCF.

NEUROMORPHIC COMPUTING AND MEMRISTORS

The significant difficulties in transistor scaling and a concern for energy efficiency have driven the search for new computing technologies in the past decade. Among them, the memristor stands as one of the most promising due to its memory capabilities and non-volatility. It has also opened up new doors to the neuromorphic computing community due to its ability to house memory and computation units on the same chip, a problem which has plagued von Neumann architectures, even in the multicore era.

Neuromorphic computing seeks to mimic the very desirable properties of the most enigmatic, inherently parallel supercomputer: the human brain. In particular, we seek high energy efficiency, robustness to noise, and the ability to implement learning algorithms at the hardware level. The nascent memristor has demonstrated tremendous potential as an enabling technology that is perfectly suited for this purpose.

A memristor consists of a film of width D with a low-resistance doped region of width w and a high-resistance undoped region. Consequently, the memristor can be thought of as two resistors in series, with resistances R_{on} and R_{off}, such that $R_{on} \ll R_{off}$. A flow of current can cause the doped region to shrink or expand, thus increasing or decreasing the total resistance. With respect to Figure 1, this charge-dependent resistance directly affects V_{out}. Thus, V_{out} can be seen as a function of V_{in} and the state of the memristor, similar to how an output neuron is dependent on input neurons and synaptic weights.

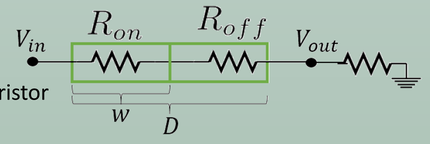

Figure 1: The memristor

METHODOLOGY

UCF has been selected by NVIDIA to be a "2014 CUDA Teaching Center" in order to help promote parallel computing education at UCF. NVIDIA has donated CUDA-capable GPUs as well as instructional material. We plan to leverage these resources to develop a parallelized neuromorphic computing simulator based on memristor crossbars. We will also begin offering core courses that focus on parallel and CUDA programming.

A memristor *crossbar* consists of two sets of parallel wires, each set perpendicular to the other, with a memristor at each wire junction. Such a crossbar structure, coupled with the charge-dependent conductance of the memristor, can naturally encode a neuromorphic computing architecture. Figure 2 depicts a simple neural network and its memristor crossbar interpretation. Note that for any output V_j^{out} in the crossbar, its corresponding value is dependent on the inputs $V_i^{in}, 1 \leq i \leq m$ and the states of the memristors $w_{i,j}$. Thus, $V_j^{out} = \sum f(V_i^{in}, w_{i,j})$, where each memristor acts as a synaptic weight.

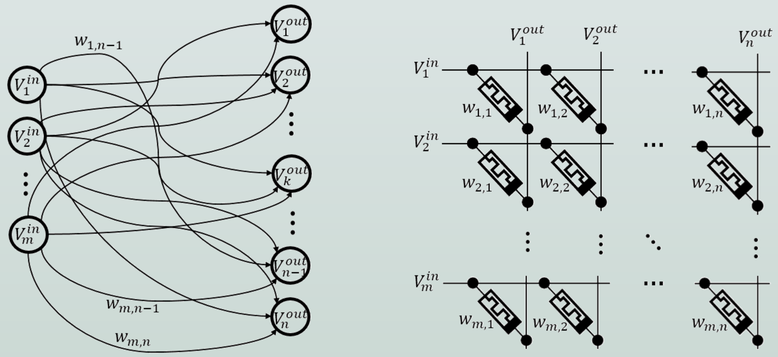

Figure 2: A simple neural network (left) and its corresponding memristor crossbar implementation (right).

The development of such a simulator will be a great educational tool for students interested in this emerging technology or in parallel computing in general. Furthermore, as of the time of this writing, there does not exist a robust memristor model that can be used to simulate more than a few hundred memristors simultaneously. The completion of this system would be of great interest to the memristor research community, where it is desirable to be able to simulate many memristors at once.

The neuromorphic computing simulator will be used at UCF to teach students about CUDA programming and emerging architectures. UCF has agreed to support two graduate teaching assistants and has provided a GPU server to aid us in incorporating these parallel/CUDA computing courses into the curriculum. These courses will also be motivated by ubiquitous big-data problems that require novel and parallel solutions. The knowledge obtained therein could prove to be invaluable to students and will help to prepare them as the computing world transitions further into the multi-core and distributed systems era. The introduction to memristor-based neuromorphic systems will also help students understand computer engineering fundamentals at the architecture level and expose undergraduates to emergent areas in nanoscale computing.

CONCLUSION

Over the next year, we will build a simulator for memristor-based neuromorphic computing for meant primarily instructional purposes. This will aid undergraduate students in their understanding of memristive systems and also help interested research students perform experiments to analyze and validate neuromorphic architectures. We will continue our emphasis on PDC instruction for undergraduate and graduate students. In Spring 2014, we will teach a graduate level complexity theory course that will cover the PRAM model and complexity of parallel version of many commonly known algorithms. In addition to discussing time/space complexity, decidability, reducibility and intractability this course will also include a module on randomized algorithms that will introduce students to the complexity class ZPP, RP and RNC. We will document all instructional material and course related-information and will collect student evaluations for each of these courses and report to them to the CDER center.

REFERENCES

[1] M. D. Hill and M. R. Marty, "Amdahl's law in the multicore era." IEEE Computer, vol. 41, no. 7, pp. 33–38, 2008.

[2] F. Hussain, N. Deo, and S. K. Jha, "Early Adoption – High-Performance Computing for Big Data," in Procceedings of the Fourth NSF/TCPP Workshop on Parallel and Distributed Computing Education, Phoeniz, AZ, May 2014.

[3] N. Deo, S. K. Jha, F. Hussain, and M. Vasudevan, "Introducing parallel programming across the undergraduate curriculum through an interdisciplinary course on computational modeling," in Procceedings of the Third NSF/TCPP Workshop on Parallel and Distributed Computing Education, Boston, MA, May 2013.

[4] D. T. Gillespie, "Exact stochastic simulation of coupled chemical reactions," The journal of physical chemistry, vol. 81, no. 25, pp. 2340–2361, 1977.

[5] "Little fe computational cluster," http://littlefe.net/, [Last accessed: 19 September 2014].

[6] D. B. Kirk and W. H. Wen-mei, Programming massively parallel processors: a hands-on approach. Newnes, 2012.

[7] S. Akhter and J. Roberts, Multi-core programming. Intel press Hillsboro, 2006, vol. 33.

[8] D. B. Strukov, G. S. Snider, D. R. Stewart, and R. S. Williams, "The missing memristor found," nature, vol. 453, no. 7191, pp. 80–83, 2008.

[9] A. Adamatzky and L. Chua, Memristor Networks. Springer Publishing Company, Incorporated, 2014.

[10] L. O. Chua, "Memristor-the missing circuit element," Circuit Theory, IEEE Transactions on, vol. 18, no. 5, pp. 507–519, 1971.

[11] P. Mazumder, S. M. Kang, and R. Waser, "Memristors: devices, models, and applications," Proceedings of the IEEE, vol. 100, no. 6, pp. 1911–1919, 2012.

[12] F. Merrikh-Bayat and S. B. Shouraki, "Memristor-based circuits for performing basic arithmetic operations," Procedia Computer Science, vol. 3, pp. 128–132, 2011.

[13] J. Borghetti, G. S. Snider, P. J. Kuekes, J. J. Yang, D. R. Stewart, and R. S. Williams, "memristiveswitches enable statefullogic operations via material implication," Nature, vol. 464, no. 7290, pp. 873–876, 2010.

[14] C. K. K. Lim, A. Gelencser, and T. Prodromakis, "Computing image and motion with 3-d memristive grids," in Memristor Networks. Springer, 2014, pp. 553–583.

[15] S. Gaba, P. Sheridan, J. Zhou, S. Choi, and W. Lu, "Stochastic memristive devices for computing and neuromorphic applications," Nanoscale, vol. 5, no. 13, pp. 5872–5878, 2013.

[16] S. H. Jo, T. Chang, I. Ebong, B. B. Bhadviya, P. Mazumder, and W. Lu, "Nanoscale memristor device as synapse in neuromorphic systems," Nano letters, vol. 10, no. 4, pp. 1297–1301, 2010.

[17] Jason Mick, "How Silicon Valley's Best-Kept Secret, Crossbar, Beat HP to the Market w/RRAM," via dailytech.com. Last accessed: 18 September, 2014.

[18] Peter Bright, "HP plans to launch memristor, silicon photonic computer within the decade," Via arstechnica.com. Last acccessed: 21 September, 2014.

[19] R. Williams, "How we found the missing memristor," Spectrum, IEEE, vol. 45, no. 12, pp. 28–35, 2008.

[20] H. Kim, M. P. Sah, C. Yang, S. Cho, and L. O. Chua, "Memristor emulator for memristor circuit applications," Circuits and Systems I: Regular Papers, IEEE Transactions on, vol. 59, no. 10, pp. 2422–2431, 2012.

[21] W. Zhao, G. Agnus, V. Derycke, A. Filoramo, J. Bourgoin, and C. Gamrat, "Nanotube devices based crossbar architecture: toward neuromorphic computing," Nanotechnology, vol. 21, no. 17, p. 175202, 2010.

[22] D. Monroe, "Neuromorphic computing gets ready for the (really) big time," Communications of the ACM, vol. 57, no. 6, pp. 13–15, 2014. [23] Sean Gallagher, "HP Labs Machine dissolves the difference between disk and memory," via arstechnica.com. Last accessed: 18 September 2014.