MAERI: Enabling Flexible Dataflow Mapping over DNN Accelerators via Programmable Interconnects

Hyoukjun Kwon
Georgia Institute of Technology
Atlanta, Georgia
hyoukjun@gatech.edu

Ananda Samajdar
Georgia Institute of Technology
Atlanta, Georgia
anandsamajdar@gatech.edu

Tushar Krishna
Georgia Institute of Technology
Atlanta, Georgia
tushar@ece.gatech.edu

ABSTRACT

The microarchitecture of DNN inference engines is an active research topic in the computer architecture community because DNN accelerators are needed to maximize performance/watt for mass deployment across phones, cars, and so on. This has led to a flurry of ASIC DNN accelerator proposals in academia over recent years. Industry is also investing heavily so every major company developing its own neural network accelerator, which resulted in myriad of dataflow patterns. We claim that dataflows essentially lead to different kinds of data movement within an accelerator. Thus, to support arbitrary dataflows in accelerators, we propose to make interconnects programmable. We achieve it by augmenting all compute elements (multipliers and adders) and on-chip buffers with tiny switches, which can be configured at compile time or runtime. Our design, MAERI, connects these switches via a new configurable and non-blocking tree topology to provide not only programmability but also high throughput.

ACM Reference Format:
Hyoukjun Kwon, Ananda Samajdar, and Tushar Krishna. 2018. MAERI: Enabling Flexible Dataflow Mapping over DNN Accelerators via Programmable Interconnects. In *Proceedings of SysML '18*. ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

The microarchitecture of deep neural network (DNN) inference engines is an active research area in computer architecture community. GPUs provide are efficient training platforms with their mass parallelism, multi-core CPUs provide platforms for algorithmic exploration, and FPGAs provide power-efficient and configurable platforms for algorithmic exploration and acceleration. However, for mass deployment across various domains (phones, cars, etc.), DNN accelerators are needed to maximize performance/watt. This has led to a flurry of ASIC proposals for DNN accelerators over recent years [3–7, 11]. Industry is also heavily investing, with every major company developing its own spatial DNN accelerator [1, 8, 9]. One of the practical and open challenges for DNN accelerator designs is programmability because DNNs can be partitioned in myriad ways (within [4] and across layers [2]) to exploit data reuse, which leads to different dataflow patterns within accelerators.

The DNN Data Flow Graph (DFG) is fundamentally a multi-dimensional multiply-accumulate calculation, as Figure 1 demonstrates. Each dataflow is essentially some kind of transformation of this loop [10, 12], which contains different optimization potentials depending on neural network layers. Unfortunately, most of DNN accelerators cannot exploit potentials of each dataflow as they internally support fixed dataflow patterns. This is because they perform a careful co-design of the PEs and the network-on-chip (NoC) (e.g., TPU [9]). Because of such inflexibility, mapping different dataflows on an accelerator can lead to compute resource underutilization. Therefore, each new optimization has required a new accelerator design for that optimization [2, 7, 11, 13], which makes the hardening of accelerator designs challenging and uneconomical.

Our insight in this work is that different dataflows essentially lead to different data movement patterns within accelerators. Thus, to support arbitrary dataflows in spatial accelerators, we propose MAERI (Multiply-Accumulate Engine with Reconfigurable Interconnect)[1], a DNN accelerator with programmable interconnects. MAERI augments all compute elements (multipliers and adders) and on-chip buffers with tiny switches, which can be programmed/-configured at compile- or run-time to support myriad dataflow scenarios. We connect these switches via a new configurable and non-blocking tree topology.

2 MAERI BUILDING BLOCKS

Figure 1 shows MAERI's building blocks:

- **Prefetch buffer (PB)**: serves as a cache of DRAM, which stores input activations, weights, intermediate partial sums that could not be fully accumulated, and output activations.
- **Activation Units**: Lookup Tables are used to implement different activation functions (such as ReLU).
- **Distribution Tree**: A fat-tree is used to distribute activations and weights from the PB to multipliers.
- **Simple Switch (SS)**: Each node in the distribution tree is a simple 2:1 switch to unicast/multicast inputs/weights.
- **Augmented Reduction Tree (ART)**: A fat-tree augmented with forwarding links is used to reduce partial sums and send outputs to activation units. An ART with N leaves can support 1 to N/2 simultaneous reductions with provable non-blocking.
- **Adder Switch (AS)**: Each node in ART is an adder augmented with a switch to allow data forwarding to peers or to parents.

SysML '18, Feb 15–16, 2018, Stanford, CA
© 2018 Association for Computing Machinery.
ACM ISBN 978-x-xxxx-xxxx-x/YY/MM...$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

[1]A version of this paper will appear in *Proc. of the 23rd ACM Int. Conf. on Architectural Support for Programming Languages and Operating Systems (ASPLOS), Mar 2018.*

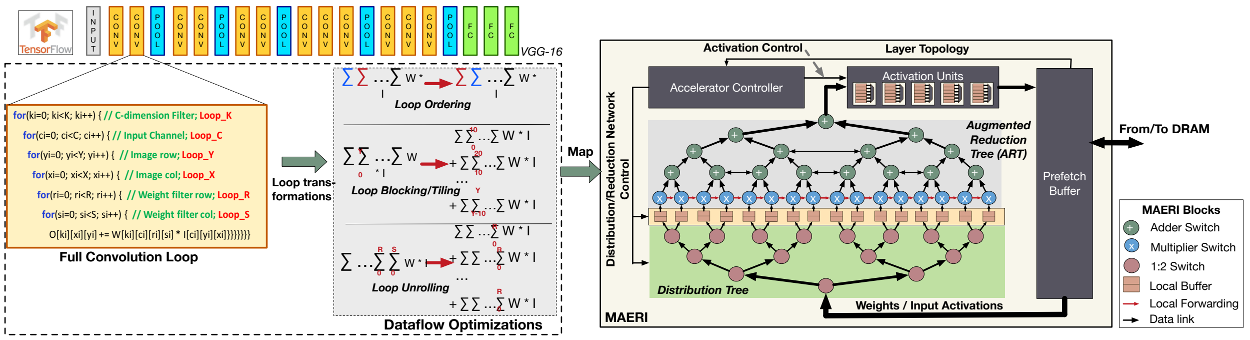

Figure 1: An overview of MAERI. MAERI is designed to efficiently handle CONV, LSTM, POOL and FC layers. It can also handle cross-layer and sparse mappings. We implement this flexibility using configurable distribution and reduction trees within the fabric.

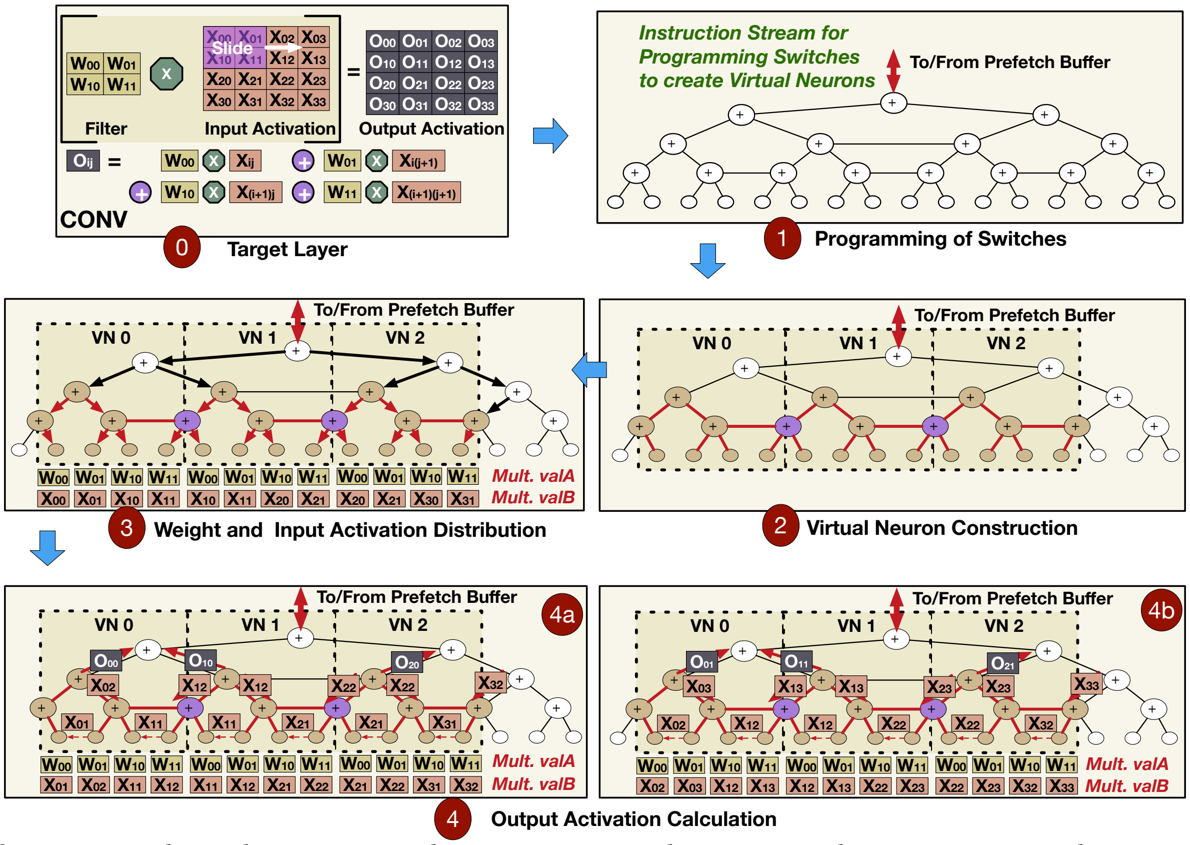
Filter
Input Activation
Output Activation
Slide
CONV
0 Target Layer
Instruction Stream for Programming Switches to create Virtual Neurons
To/From Prefetch Buffer
1 Programming of Switches
2 Virtual Neuron Construction
3 Weight and Input Activation Distribution
4 Output Activation Calculation
VN 0
VN 1
VN 2
Mult. valA
Mult. valB
4a
4b

Figure 2: Programming the switches to map a CONV layer in MAERI. W, X, and O represent weights, input activations, and output activations.

- **Multiplier Switch (MS)**: Each multiplier is augmented with a switch to allow data forwarding to neighboring multipliers, and data reception from the PB or neighboring MSes.

The distribution and reduction trees provide full non-blocking bandwidth to the compute blocks, but can be pruned to reduce the bandwidth if required to reduce area and power.

3 MAPPING DATAFLOWS OVER MAERI

The entire accelerator is controlled by a programmable controller which manages reconfiguration of all three sets of switches (MS, AS, and SS) for mapping the target dataflow. This is done by creating *Virtual Neurons (VN)* over the multiplers and adders. The flexibility of the interconnects allows us to create VNs of any size, which provides the ability to map arbitrary dataflows simultaneously.

Figure 2 shows a walk-through example of mapping a convolutional layer. Similarly, recurrent, max-pool, fully-connected, sparse and so on layers can be mapped.

4 EVALUATIONS AND CONCLUSIONS

MAERI is a spatial accelerator for mapping arbitrary dataflows that arise in DNNs due to its topology or mappings by using tiny programmable switches next to each on-chip compute and memory engine. It provides 130-283% better utilization across multiple dataflow mappings over baselines with rigid NoC fabrics. MAERI's interconnects are 10-100× smaller than conventional NoCs.

REFERENCES

[1] Filipp Akopyan, Jun Sawada, Andrew Cassidy, Rodrigo Alvarez-Icaza, John Arthur, Paul Merolla, Nabil Imam, Yutaka Nakamura, Pallab Datta, Gi-Joon Nam, Brian Taba, Michael Beakes, Bernand Brezzo, Jente B. Kuang, Rajit Manohar, William P. Risk, Brayan Jackson, and Dharmendra S. Modha, *Truenorth: Design and tool flow of a 65 mw 1 million neuron programmable neurosynaptic chip*, TCADICS **34** (2015), no. 10, 1537–1557.

[2] Manoj Alwani, Han Chen, Michael Ferdman, and Peter Milder, *Fused-layer CNN accelerators*, 49th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO), 2016.

[3] Tianshi Chen, Zidong Du, Ninghui Sun, Jia Wang, Chengyong Wu, Yunji Chen, and Olivier Temam, *Diannao: A small-footprint high-throughput accelerator for ubiquitous machine-learning*, ASPLOS, 2014, pp. 269–284.

[4] Yu-Hsin Chen, Joel Emer, and Vivienne Sze, *Eyeriss: A spatial architecture for energy-efficient dataflow for convolutional neural networks*, ISCA, 2016.

[5] Yunji Chen, Tao Luo, Shaoli Liu, Shijin Zhang, Liqiang He, Jia Wang, Ling Li, Tianshi Chen, Zhiwei Xu, Ninghui Sun, and Olivier Temam, *Dadiannao: A machine-learning supercomputer*, MICRO, 2014, pp. 609–622.

[6] Zidong Du, Robert Fasthuber, Tianshi Chen, Paolo Ienne, Ling Li, Tao Luo, Xiaobing Feng, Yunji Chen, and Olivier Temam, *Shidiannao: Shifting vision processing closer to the sensor*, ISCA, 2015.

[7] Song Han, Xingyu Liu, Huizi Mao, Jing Pu, Ardavan Pedram, Mark A Horowitz, and William J Dally, *Eie: efficient inference engine on compressed deep neural network*, ISCA, 2016.

[8] Intel, *Intel's new self-learning chip promises to accelerate artificial intelligence*, https://newsroom.intel.com/editorials/intels-new-self-learning-chip-promises-accelerate-artificial-intelligence/.

[9] Norman P Jouppi, Cliff Young, Nishant Patil, David Patterson, Gaurav Agrawal, Raminder Bajwa, Sarah Bates, Suresh Bhatia, Nan Boden, Al Borchers, et al., *In-datacenter performance analysis of a tensor processing unit*, Proceedings of the 44th Annual International Symposium on Computer Architecture, ACM, 2017, pp. 1–12.

[10] Wenyan Lu, Guihai Yan, Jiajun Li, Shijun Gong, Yinhe Han, and Xiaowei Li, *Flexflow: A flexible dataflow accelerator architecture for convolutional neural networks*, HPCA, 2017.

[11] Angshuman Parashar, Minsoo Rhu, Anurag Mukkara, Antonio Puglielli, Rangharajan Venkatesan, Brucek Khailany, Joel Emer, Stephen W Keckler, and William J Dally, *Scnn: An accelerator for compressed-sparse convolutional neural networks*, Proceedings of the 44th Annual International Symposium on Computer Architecture, ACM, 2017, pp. 27–40.

[12] Chen Zhang, Peng Li, Guangyu Sun, Yijin Guan, Bingjun Xiao, and Jason Cong, *Optimizing fpga-based accelerator design for deep convolutional neural networks*, FPGA, 2015, pp. 161–170.

[13] Shijin Zhang, Zidong Du, Lei Zhang, Huiying Lan, Shaoli Liu, Ling Li, Qi Guo, Tianshi Chen, and Yunji Chen, *Cambricon-x: An accelerator for sparse neural networks*, MICRO, 2016.