OPTIMO: A 65nm 270MHz 143.2mW Programmable Spatial-Array-Processor with a Hierarchical Multi-cast On-Chip Network for Solving Distributed Optimizations

Muya Chang, Li-Hsiang Lin, Justin Romberg, Arijit Raychowdhury
School of ECE, Georgia Institute of Technology, GA, USA

Abstract—This paper presents OPTIMO, a 65nm, 16-b, fully-programmable, spatial-array processor with 49-cores and a hierarchical multi-cast network for solving distributed optimizations. We demonstrate six template algorithms and their applications and measure a peak energy-efficiency of 0.279 TOPS/W.

Keywords—optimizations; array-processing; multi-cast network

I. INTRODUCTION

The explosion of big-data problems arising in statistics, machine learning (ML), image processing, 5G systems and other related areas [1] have accelerated the development of hardware prototypes that rely on data-flow architectures and near-memory processing to address the memory-bottleneck. Looking beyond the success of neural network (NN) accelerators for classification [2-5], we recognize a growing need for solving complex optimization problems, which arise in all areas of signal processing such as ML model-training, computational imaging (medical, optical and hyper-spectral), resource-allocation in 5G massive MIMO networks and solving inverse problems such as LDPC decoding. In spite of the diversity of applications, a common mathematical framework, namely solving constrained-optimizations (i.e., minimize \( l(x) \) under a constraint \( r(x) = 0 \) for a vector \( x \) and functions \( l \) and \( r \)) has emerged; and recent advances in the alternating direction method of multipliers (ADMM) has been particularly successful for large-scale problems [1]. A review of the ADMM algorithm is beyond the scope of this article, but it suffices to say that ADMM is one of the most common iterative algorithms for solving constrained-optimizations (i.e., minimize \( l(x) \) under \( x - z = 0 \) ) has been particularly successful for large-scale problems [1]. A review of the ADMM algorithm is beyond the scope of this article, but it suffices to say that ADMM is one of the most common iterative algorithms for solving constrained-optimizations (i.e., minimize \( l(x) \) under a constraint \( r(x) = 0 \) for a vector \( x \) and functions \( l \) and \( r \)) has emerged; and recent advances in the alternating direction method of multipliers (ADMM) has been particularly successful for large-scale problems [1]. A review of the ADMM algorithm is beyond the scope of this article, but it suffices to say that ADMM is one of the most common iterative algorithms for solving constrained-optimizations (i.e., minimize \( l(x) \) under a constraint \( r(x) = 0 \) for a vector \( x \) and functions \( l \) and \( r \)) has emerged; and recent advances in the alternating direction method of multipliers (ADMM) has been particularly successful for large-scale problems [1]. A review of the ADMM algorithm is beyond the scope of this article, but it suffices to say that ADMM is one of the most common iterative algorithms for solving constrained-optimizations (i.e., minimize \( l(x) \) under a constraint \( r(x) = 0 \) for a vector \( x \) and functions \( l \) and \( r \)) has emerged; and recent advances in the alternating direction method of multipliers (ADMM) has been particularly successful for large-scale problems [1].

II. SYSTEM ARCHITECTURE AND DESIGN

A. System Design

Fig. 2 illustrates the chip-architecture where 49 programmable 16b OPUs (optimization processing units) indexed as (row, column) (1) compute locally and iteratively and (2) transmit/receive data from the neighbors. The OPUs interact via a 2-layered multi-cast network with (1) layer-0 establishing near-neighbor (neighborhood of 8) bi-directional connections

Fig I: OPTIMO: A partial array processor for solving distributed optimizations via distributed ADMM. This work (measured) shows 4.77x (4.18x) improvement in energy (performance) compared to a GPU-style SIMT machine (simulated).

Fig 2 System Architecture showing the 49 OPUs and a 2-layer multi-cast on-chip network.
and (2) layer-1 connecting 4 cluster center OPUs i.e, (2,2), (6,2), (2,6) and (6,6) with the chip-center OPU i.e, (4,4). Depending on the algorithm and structure of the data, optimization algorithms require complex data-flow patterns where both near-neighbor (layer-0 connections) as well as global information (layer-0 and layer-1 connections) are used. The 48-OPUs are divided into four clusters as shown with one in the center as shown in Fig. 3. Global consensus is reached in each iteration via: (1) the four clusters reach cluster-level consensus (layer-0), (2) gather process where the chip-center obtains cluster-level consensus information from cluster centers (layer-1) and calculates the global data, (3) scatter (step-1) process where the chip-center scatters the global data back to the cluster centers, and (4) a final scatter (step 2 and 3) process where the cluster-central spreads the data across all the OPUs. This readies all the OPUs for the next iteration. The scatter and gather processes are intrinsic to distributed optimizations as the system computes locally, distributes information globally and iterates to reach consensus. We compare the proposed hierarchical multi-cast network with networks that allow 4 or 6 connections to the neighbors – as is common in convolutional and deep neural networks [2]. Architectural and network simulations of various optimization algorithms on >10000 random data-sets reveal a 29%-77% reduction in convergence time compared to a fixed, 4-neighbor TPU-style connection (Fig. 4).

B. Optimization Processing Unit

Each OPU features (Fig. 5): (1) one computation module consisting of a programmable digital signal processor (P-DSP), a scratchpad memory and control logic, (2) 2KB of instruction cache, (4) 4KB of data memory (for local data R/W), and (5) a transceiver module for the gather and scatter processes. Programming is supported via 32b instructions (Fig. 6) and each inter-OPU data-movement is supported on dedicated links. Before data-transmission, a transmit buffer temporarily stores the data and it is flushed out at the end of the transmission. Received data is not buffered; instead the control logic directly writes the incoming traffic to the data cache thereby reducing both latency and energy. The design supports synchronous, mesochronous as well as asynchronous communication among OPUs with bidirectional FIFOs enabling fast and parallel data exchange across clock-crossing boundaries. Fig. 6 further
summarizes: (1) the number of instructions supported by each module, (2) the commonly-used macro functions and the number of instructions per function, and (3) their usage in the six template algorithms.

C. Programmed DSP Architecture

Fig. 7 illustrates the principal components of the P-DSP, which consists of three pipeline stages in an architecture designed to maximize energy-efficiency. The first two stages support add/subtract and multiply/divide and the final stage supports a class of fixed-function blocks as shown in Fig. 7. Instruction level control of the pipeline and variable latency through the P-DSP are maintained via a program counter. High level program-code and data-sets are translated to instruction and data, scanned in to the chip and then executed. Convergence is declared either (1) after a fixed number of iterations, or (2) when the maximum cycle-to-cycle change of data in an OPU falls below a threshold. The clock-crossing FIFO features a 64b buffer and operates on a 4-phase handshaking scheme (Fig. 8).

The timing diagram for executing ADMM is shown in Fig. 9. The system is clocked either by (1) a single global clock (synchronous) or (2) DCO based clock per-OPU enabling asynchronous/mesochorous links.

D. Die micrograph and chip characteristics

The test-chip is fabricated in TSMC 65nm GP CMOS process and occupies a total area of 12mm². It features 306.25KB of on-die memory distributed across 49-cores. The die micrograph and chip characteristics are shown in Fig. 10.

III. MEASUREMENT RESULTS

Fig. 11 illustrates the measured power-performance trade-off showing a peak FMAX (in a synchronous setting) of 270 MHz (at 0.5V) and operation down in 0.5V (with FMAX = 10 MHz). Energy-efficiency (considering both dynamic and leakage power) shows a peak of 0.279TOPS/W at 0.6V, below which the design is leakage-dominated owing to the large (306.25 KB) on-die SRAM. It should also be noted that an operation here represents execution of a single pipeline-stage of the P-DSP and is computationally more demanding than MAC operations native to NN accelerators. Per-OPU DCO-based clocking reduces the overhead of routing a global clock and enables 2.7%–7.75% power savings compared to global clocking at iso-performance. The power-breakdown among computation, communication and storage at 0.6 V and 1.0 V is shown in Fig. 12. We use the hardware prototype to execute template algorithms across multiple data-sets and plot the time-to-
compute and energy at 0.6 V (Fig. 13). Although we demonstrate the capability of this near-memory spatial-architecture in solving distributed optimizations, the proposed hardware and programming model can also support a variety of other array processing tasks including inference in deep and convolutional neural networks.

IV. APPLICATIONS

The programmable and iterative optimization solver is capable of addressing multiple applications. MRI image reconstruction from non-uniformly sampled data-points is computationally challenging and requires patients to lie in the machine for a long time. Our solution uses iterative least-squares optimization (Fig. 14 (a)) to reconstruct MRI images with high PSNR in less than 8ms. Similarly binary SVMs (Fig. 14 (b)), a popular choice in ML classification problems shows convergence with increasing number of iterations (multi-class SVM records 91% accuracy on the MNIST data-set, which is the state-of-the-art). Further, feature extraction with LASSO (L1 regularization) used in ML, is shown in Fig. 14 (c). In Table I, a comparison with the state-of-the-art shows a peak performance of 270MHz and peak energy-efficiency of 0.279 TOPS/W.

ACKNOWLEDGEMENTS

This work was supported in part by C-BRIC, one of six centers in JUMP, a Semiconductor Research Corporation (SRC) program sponsored by DARPA, and in part by the NSF under grant 1640081, and the Nanoelectronics Research Corporation (NERC), a wholly-owned subsidiary of the SRC, through Extremely Energy Efficient Collective Electronics (EXCEL), an SRC-NRI Nanoelectronics Research Initiative under Research Task ID 2698.002.

REFERENCES


Table I: Comparison of the proposed array-processor with competitive spatial-array processors. The proposed design addresses distributed optimization which presents a more complex data-flow and compute than traditional CNN and DNN inference architectures.