Communication Optimization for the 16-core Epiphany Floating-Point Processor Array

Siddhartha, Nachiket Kapre
Nanyang Technological University
50 Nanyang Avenue, Singapore
siddhart005@e.ntu.edu.sg, nachiket@ieee.org

***Abstract*—The management and optimization of communication in an NoC-based (network-on-chip) bespoke computing platform such as the Parallella (Zynq 7010 + Epiphany-III SoC) is critical for performance and energy-efficiency of floating-point bulk-synchronous workloads. In this paper, we explore the opportunities and capabilities of the Epiphany-III SoC for communication-intensive workloads. Using our communication support library for the Epiphany, we are able to accelerate single-precision BSP workloads like the Sparse Matrix-Vector multiplication (SpMV) on Matrix Market datasets by up to 6.5× and PageRank algorithm on the BerkStan SNAP dataset by up to 8×, while lowering power usage by 2× over optimized ARM-based implementations. When compared to optimized OpenMP x86 mappings, we observe a ≈10× improvement in energy efficiency (GFLOP/s/W) with Epiphany SoC.**

I. Idea

Many modern SoCs (systems-on-chip) are equipped with a fast and capable communication fabric (NoC, network-on-chip) that makes it possible to accelerate a range of communication-intensive problems while lowering power requirements by avoiding complex shared-memory controllers. In this paper, we investigate the raw potential and tuneability of the Epiphany III SoC [2] that is a bespoke computing platform optimized for processing of communication-rich floating-point applications. The Epiphany SoC is a spiritual successor to Ambric [1] and reaches further with floating-point support and packet-switched NoC enhancements. However, utilizing the Epiphany NoC to its fullest potential can be a challenge due to limited software-based communication optimization support. In this paper, we develop a framework for exposing and optimizing communication for this SoC through a graph-centric bulk-synchronous model. We use a combination of strategies including (a) effective generation and storage of communication graphs within the per-core 32KB RAMs, (b) automated unrolling of message-passing instructions to keep the NoC busy with useful work, and (c) programmable selection of DMA or memcpy functions for implementing specific communication patterns depending on data transfer size.

II. Case Studies

We characterize and automate performance tuning of spatial parallelism for supporting (1) random access load-store style traffic suitable for irregular sparse computations found in Sparse Matrix-Vector (SpMV) multiplication workloads, as well as (2) variable, data-dependent traffic patterns in PageRank workloads.

A. Sparse Matrix-Vector (SpMV) Multiplication

We profile the performance of SpMV on 17 different Matrix Market matrices from different domains (*e.g.* circuit simulation and computational fluid dynamics). We benchmark the performance of our communication library against optimized ARM implementations, and observe speedups ranging from 2–6.5× across the datasets when all our optimizations are active. In comparison, **without** our communication optimizations enabled, SpMV performance suffers 0.14–0.53× slowdown.

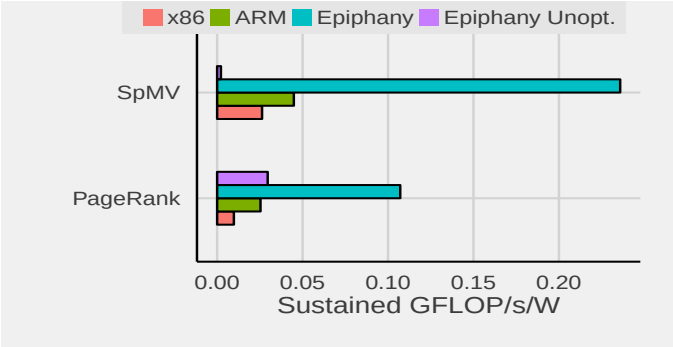

Fig. 1: Peak GFLOPs/W performance comparisons

B. PageRank

We use the BerkStan web graph from the SNAP [3] library as input to PageRank and benchmark performance against optimized CPU implementations. We observe performance improvements with increasing number of edges (links) in the input graph, and achieve speedups of up to 8× on the largest graphs when all our optimizations are enabled. In contrast, **without** any optimizations enabled, the Epiphany is able to achieve speedups only up to 1.44× for the largest graphs.

We also compare the power efficiency of the Epiphany (with/without our communication library) against ARM and x86 mappings (Intel Xeon CPU E5-1650 v2 @ 3.50GHz with OpenMP). We observe about 10× improvements in energy efficiency over x86 mappings (Figure 1).

References

[1] M. Butts. Synchronization through communication in a massively parallel processor array. *Micro, IEEE*, 27(5):32–40, 2007.
[2] L. Gwennap. Adapteva: More flops, less watts. *Microprocessor Report*, 6(13):11–02, 2011.
[3] J. Leskovec and A. Krevl. SNAP Datasets: Stanford large network dataset collection. http://snap.stanford.edu/data, June 2014.