Dan Cassidy
CSCI-B 424
Homework 1
2016-09-07

Transparent Live Code Offloading on FPGA
https://arxiv.org/pdf/1609.00130v1.pdf

The authors of the paper recognize the benefits that FPGAs can bring to heterogeneous computing, but recognize that the addition of said devices drastically increases the complexity of not only the computer, but the code driving the required calculations as well. They propose a framework for automatic offloading of the computationally intensive bits of code to an FPGA, without developer interaction, by analyzing the data flow graph of the input program. This framework is not as broadly focused as some of the usual industry standards, such as OpenMP, OpenACC, and OpenCL, and is instead focused more on domain-specific languages. Ultimately they found that while versatile, their implementation is performance limited.

Recent Results on Fault-Tolerant Concensus in Message-Passing Networks
https://arxiv.org/pdf/1608.07923v1.pdf

This paper is a survey on "important results from on fault-tolerant consensus in message-passing networks" from the past ten years. They go on to explain the definition of classical fault-tolerant consensus, and then start comparing many of these newer methodologies. There are too many to list here (within a page or two), and to be honest, I don't understand much of it so I feel I wouldn't be able to summarize them effectively.

Parallel Single-Source Shortest Paths
http://courses.csail.mit.edu/6.884/spring10/projects/kelleyk-neboat-paper.pdf

The authors did a good job summing up their paper in the abstract in a single sentence: "We designed and implemented a parallel algorithm for solving the single-source shortest paths (SSSP) problem for graphs with nonnegative edge weights, based on Gabow's scaling algorithm for SSSP." They assert that their method outperforms the classic Dijkstra algorithm on "six or more processors" for a random graph. Gabow's algorithm is more complex and costly than Dijkstra's algorithm, but its structure lends itself to parallelism in a way that Dijkstra's does not.

A New GPU-based Approach to the Shortest Path Problem
http://www.infor.uva.es/~diego/docs/ortega13b.pdf

This paper details a GPU-based implementation of the Crauser et al. SSSP algorithm, with which the authors claim significant speedups with respect to CPU-based and another GPU Dijkstra implementation, developed by Martín et al. (Reference [7] in source paper.) Through their experiment, using the same input data as Martín et al., they were able to demonstrate a pretty significant speed up (up to 220x), particularly in comparison with the CPU-based Dijkstra implementation. The improvement over the other GPU-based implementation was much more modest (from 1.01x to 1.17x), but still notable.