2021 IEEE 28th International Conference on High Performance Computing, Data and Analytics Workshop (HiPCW) | 978-1-6654-1037-3/21/$31.00 ©2021 IEEE | DOI: 10.1109/HIPCW54834.2021.00014

HiPC 2021 Workshop on Parallel Programming in the Exascale Era (PPEE 2021)

Held virtually on December 18, 2021

Compendium of Abstracts

Description

This is the first edition of the PPEE workshop. The upcoming exascale systems will impose new requirements on application developers and programming systems to target platforms with hundreds of homogeneous and heterogeneous cores. The four critical challenges for exascale systems are extreme parallelism, power demand, data movement, and reliability. These systems are aimed to solve problems that were previously out of reach and to improve the parallel performance of applications by a factor of 50x. The power budget for achieving a billion billion (quintillion) floating-point operations per second (exaflops) should be within 20-30 MW. Moving the data on these systems relative to the computation will be challenging due to complex memory hierarchies. It would be essential to keep the CPUs/accelerators busy once they have the data to avoid memory bottlenecks. Failures on these systems are anticipated to occur many times a day, such that the existing approach for resiliency, such as checkpointing and restart, will not work.

The goal of this workshop is to attract leading researchers to exchange ideas and share their work-in-progress and latest results and to begin the discussions of how to address the exascale software challenges. Topics that will be covered include:

- High-level programming models for many-cores / accelerators
- Compilation techniques for hybrid CPU/accelerator parallelism
- Intra-/Inter-node load balancing and scheduling
- Runtime systems for high performance and high productivity
- Comparisons of runtime systems and parallel programming models
- OS/runtime and system software
- Optimizing data locality and data movement
- Energy efficiency and optimizations
- Resilience and fault-tolerance
- Scalable algorithms and synchronization mechanisms
- Concurrent data structures
- Applying machine learning techniques in HPC

Organization Committee
Vivek Kumar, Indraprastha Institute of Information Technology, Delhi
Swarnendu Biswas, Indian Institute of Technology, Kanpur
Vishwesh Jatala, Indian Institute of Technology, Bhilai

Program Committee
Dip Sankar Banerjee, IIT Jodhpur
Gokul Swamy, Amazon
Jyothi Vedurada, IIT Hyderabad
Nikhil Hegde, IIT Dharwad
Preeti Malakar, IIT Kanpur
Sanket Tavarageri, Microsoft
Soumyajit Dey, IIT Kharagpur
Sridutt Bhalachandra, Lawrence Berkeley National Laboratory
Yogish Sabharwal, IBM Research

DOI 10.1109/HiPCW54834.2021.00014

HiPC 2021 PPEE Workshop
List of Invited Talks

December 18, 2021

1 *A Fine-grained Asynchronous Bulk Synchronous Parallelism Model for Exascale Computing*
Vivek Sarkar, Professor and Chair of the School of Computer Science, and Stephen Fleming Chair for Telecommunications in the College of Computing, Georgia Institute of Technology, USA

2 *Addressing Graph Processing from Supercomputers to Custom Accelerators: the System Perspective*
Antonino Tumeo, Research Scientist, Pacific Northwest National Laboratory, USA

3 *Exascale Programming for & with AI*
Sasikanth Avancha, Research Scientist, Intel Corporation, India

4 *Challenges in Analyzing and Optimizing Parallel Program*
Prof. Krishna Nandivada, Professor, Department of Computer Science and Engineering, IIT Madras, India

HiPC 2021 PPEE Workshop
Invited Talk #1 Abstract

A Fine-grained Asynchronous Bulk Synchronous Parallelism Model for Exascale Computing

Vivek Sarkar
Professor and Chair of the School of Computer Science and Stephen Fleming Chair for Telecommunications in the College of Computing, Georgia Institute of Technology, USA

Abstract:

It is widely recognized that current exascale computer systems are qualitatively different from past HPC systems. As predicted over a decade ago in the DARPA Exascale Study and other sources, they are being built with heterogeneous processors, and their performance is being driven by nearly billion-way parallelism while being constrained by energy and data movement. With the end of Moore's Law, hardware trends point towards an era of increased heterogeneity with new classes of processors and accelerators, heterogeneous memories, near/in-memory computation structures, and even non-von Neumann computing elements being proposed for future HPC systems. On the software front, many current HPC execution models have been influenced by the simplicity and scalability of the Bulk-Synchronous Parallelism (BSP) model, which consists of "supersteps" separated by barriers executing on homogeneous processors. Each processor only performs local computations and asynchronous communications in a superstep, and the role of the barrier is to ensure that all communications in a superstep have been completed before moving to the next superstep. However, the increasing degree of heterogeneity and performance variability in exascale machines has motivated the need for including asynchronous computations within a superstep so as to reduce the number of barriers performed and the total time spent waiting at barriers. To that end, we advocate for extending BSP to a Fine-grained-Asynchronous Bulk-Synchronous Parallelism (FA-BSP) model, as summarized below.

Our proposal is to realize the FA-BSP model by building on three ideas from past work in an integrated approach. The first idea is the actor model, which enables distributed asynchronous computations via fine-grained active messages while ensuring that all messages are processed atomically within a single-mailbox actor. For FA-BSP, we extend classical actors with multiple symmetric mailboxes for scalability, and with automatic termination detection of messages initiated in a superstep. The second idea is message aggregation, which we believe should be performed automatically to ensure that the FA-BSP model can be supported with performance portability across different systems with different preferences for message sizes at the hardware level due to the overheads involved. The third idea is to build on an asynchronous tasking runtime within each node, and to extend it with message aggregation and message handling capabilities. In this talk, we will share early experiences from a prototype implementation of the FA-BSP model implemented in the Habanero C/C++ library (HClib). For a set of seven irregular mini-applications, our results show a geometric mean performance improvement of over 25x relative to UPC versions and over 19× relative to the OpenSHMEM versions, when using 2048 cores in the NERSC Cori system. These results suggest that the FA-BSP model offers a desirable point in the productivity-performance spectrum, with higher performance relative to PGAS models such as UPC and OpenSHMEM and higher productivity relative to the use of low-level hand-coded approaches for communication management and message aggregation.

Bio:

Vivek Sarkar is Chair of the School of Computer Science and the Stephen Fleming Chair for Telecommunications in the College of Computing at Georgia Institute of Technology. He conducts research in multiple aspects of programmability and productivity in parallel computing, including programming languages, compilers, and runtime systems for parallel, heterogeneous, and high-performance computer systems. Sarkar began his career in IBM Research after obtaining his Ph.D. from Stanford University, supervised by John Hennessy. His research projects at IBM include the PTRAN automatic parallelization system led by Fran Allen, the ASTI optimizer for IBM's XL Fortran product compilers, the open-source Jikes Research Virtual Machine for the Java language, and the X10 programming language developed in the DARPA HPCS program. He was a member of the IBM Academy of Technology during 1995-2007. After moving to academia, Sarkar has mentored over 30 Ph.D. students and postdoctoral researchers in the Habanero Extreme Scale Software Research Laboratory, first at Rice University during 2007-2017, and now at Georgia Tech since 2017. Researchers in his lab have developed the Habanero-C/C++ and Habanero-Java programming systems for parallel, heterogeneous, and distributed platforms. While at Rice, Sarkar was the E.D. Butcher Chair in Engineering, served as Chair of the Department of Computer Science, created a sophomore-level course on the fundamentals of parallel programming (COMP 322), and a three-course Coursera specialization on parallel, concurrent, and distributed programming. He also led the DARPA Exascale Software Study and the publication of its 160-page report in 2009. Sarkar is an ACM Fellow and an IEEE Fellow. He has been serving as a member of the US Department of Energy's Advanced Scientific Computing Advisory Committee (ASCAC) since 2009, and on CRA's Board of Directors since 2015. He is also the recipient of the 2020 ACM-IEEE CS Ken Kennedy Award.

HiPC 2021 PPEE Workshop
Invited Talk #2 Abstract

Addressing Graph Processing from Supercomputers to Custom Accelerators: the System Perspective

Antonino Tumeo
Research Scientist, Pacific Northwest National Laboratory, USA

Abstract:

This talk will briefly overview the current trends in large graph processing, discussing challenges and opportunities at all levels of the system stack (from abstractions to programming models, from runtimes to custom hardware). It will then discuss some of our previous and current work in the area, detailing the work done on GEMS, the Graph Engine for Multithreaded Systems, a graph processing stacks for commodity clusters, and highlighting key contributions on techniques to customize hardware architectures and generate custom accelerators for graph processing.

Bio:

Dr. Antonino Tumeo received the M.S degree in Informatic Engineering, in 2005, and the Ph.D degree in Computer Engineering, in 2009, from Politecnico di Milano in Italy. Since February 2011, he has been a research scientist in the PNNL's High Performance Computing group. He Joined PNNL in 2009 as a post doctoral research associate. Previously, he was a post-doctoral researcher at Politecnico di Milano. His research interests are modeling and simulation of high performance architectures, hardware-software codesign, FPGA prototyping and GPGPU computing.

HiPC 2021 PPEE Workshop
Invited Talk #3 Abstract

Exascale Programming for & with AI

Sasikanth Avancha
Research Scientist, Intel Corporation, India

Abstract:

Machine Learning (ML), Deep Learning (ML) and Reinforcement Learning (ML) form a core set of technologies with the overall context of the broad term Artificial Intelligence (AI). These AI technologies -- applied to next-generation workloads such as Natural Language Processing (NLP), Graph Neural Networks (GNN) and Recommendation Engine models and in combination with traditional High-Performance Computing (HPC) workloads – are driving the need for and beyond Exascale computing. Thus, we must study, analyze and optimize Exascale computing for AI. On the other hand, programming Exascale machines with billion-wide concurrency is a non-trivial task, even for so-called ninja programmers. Traditional compiler technologies that apply standard heuristics and code-generation techniques are not sufficient to meet this challenge. Therefore, for a given workload and an Exascale machine configuration, we seek to apply AI for Exascale programming. In this talk, I discuss and present our work on analyzing and optimizing large-scale GNN training as an example of an Exascale workload. I present techniques required to optimize GNNs on a single CPU socket as well as multiple CPU sockets, i.e., distributed GNN training. These techniques are generic enough to be applicable to other AI workloads enabling them to run with improved efficiency on Exascale machines.
While the applicability of such techniques can be broadened, a key problem for Exascale computing to succeed is the need to ensure that loop structures and operations – order of loops, tiling, unrolling, vectorization are optimal for the code in each workload. Further, the layout of input and output data must be such that loads, and stores operate on the fastest memory within the CPU. All these optimizations are also dependent on runtime constants associated with different loops and data tensor sizes. As a result, industry and academia have developed hand-coded libraries which contain optimized code sequences for the most common case loop structures and runtime constants. However, the rapid development of new Exasacle workloads and associated algorithms means that these hand-coded libraries may no longer be able to execute them with high performance. I discuss our work called PolyDL on automating loop optimizations in the specific context of Deep Learning workloads using heuristics as well as Deep Learning techniques to enable tensor compilers to generate high performance code; thus, in effect, this effort maybe referred to as “AI-for-AI” with respect to code generation. I discuss our ongoing efforts to combine RL and DL techniques to train models to predict appropriate loop transforms for a given set of code with the goal of enabling compilers to generate machine-code that delivers performance close to or better than hand-coded libraries.

Bio:

Sasikanth Avancha (Member, IEEE) received the B.E. degree in Computer Science and Engineering from the University Visvesvaraya College of Engineering (UVCE), Bengaluru, India, in 1994. He received M.S. and Ph.D. degrees in Computer Science from the University of Maryland at Baltimore County (UMBC), Baltimore, MD, USA, in 2002 and 2005, respectively. He is currently a Senior Research Scientist with the Parallel Computing Lab in Intel Labs and is based out of Intel India in Bengaluru. He has over 20 years of industry and research experience. He has four patents and over 25 articles spanning security, wireless networks, systems software, computer architecture and deep learning. His current research focuses on high-performance algorithm development, analysis and optimization of large-scale, distributed deep learning and machine learning training and inference, on different data-parallel architectures, including x86 and accelerator architectures, across application domains.

HiPC 2021 PPEE Workshop
Invited Talk #4 Abstract

Challenges in Analyzing and Optimizing Parallel Program

Krishna Nandivada
Professor, Department of Computer Science and Engineering, IIT Madras, India

Abstract:

Multicore systems have taken the computing world by a storm, with the ever-increasing amount of parallelism in the hardware, and the continuously changing landscape of parallel programming. The programmers are expected to think in parallel and express the program logic (ideal parallelism), using parallel languages of their choice. However, a parallel program is not guaranteed to be efficient just because it is parallel. To realize efficient parallel programs, one has to address challenges in multiple dimensions: (i) writing efficient programs, (ii) analyzing parallel programs for efficiency, (iii) optimizing parallel programs. This problem becomes challenging as many of the traditional assumptions about serial programs do not hold in the context of parallel programs. In this talk, we will (i) discuss the importance of recognizing parallel programming as an important and distinct step, and (ii) go over the idea of ideal and useful parallelism and some of our experiences in bridging the gap between them.

The talk will first explain the performance challenges in parallel programs. We will follow it up with our experience in identifying different patterns in parallel programs that can be exploited to realize highly performant code. These can be seen as both manual and compiler optimizations. We will focus on the safety, profitability, and opportunities of such optimizations in the context of task-parallel programs. We will also explain the insufficiency of traditional analysis for safely transforming parallel programs and discuss how may-happen-in-parallel analysis plays a vital role in sound and precise analysis of parallel programs. In addition to covering the traditional HPC kernels, we also will have a particular focus on irregular task-parallel programs, which are becoming critical workloads. We will explain the challenges with irregular task-parallel programs and then discuss how we can achieve high performance in irregular task-parallel programs.

Bio:

V. Krishna Nandivada is currently a Professor in the department of Computer Science and Engineering at IIT Madras. He is a senior member of ACM and IEEE. Before joining IIT Madras in 2011, he spent nearly 5.5 years at IBM India Research Lab (Programming Technologies and Software Engineering group). Prior to starting his PhD, he was associated with Hewlett Packard. He holds a BE degree from REC (now known as NIT) Rourkela, ME degree from IISc Bangalore, and PhD degree from UCLA. His research interests are Compilers, Program Analysis, Programming Languages, and Multicore systems.