Introduction to the Special Issue on PPOPP 2017 (Part 2)

This collection of four articles represents the second half of the revised versions of eight papers initially presented at the 22nd ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming (PPOPP) held February 2–4, 2017, in Austin, Texas. The initial selection of papers from those presented at the PPOPP conference was made through a majority vote of the four guest editors. The most important selection criteria were technical contribution, long-term impact, and, as a collection, representative of the material presented at the conference. Next, the authors of these papers were contacted and asked to revise and expand their conference contribution and submit it for review. Each paper then received three reviews, and more revisions were requested. The final versions are presented here.

In “[Combining SIMD and Many/Multi-Core Parallelism for Finite State Machines with Enumerative Speculation](#),” Peng Jiang and Gagan Agrawal focus on the parallelization of finite state machine kernels—a notoriously difficult task. Previous efforts have employed speculative and enumerative execution; however, in this article the authors focus on combining SIMD and multi/many-core parallelism using a novel strategy, called *enumerative speculation*. Instead of speculating on a single state as in speculative execution or enumerating all possible states as in enumerative execution, their strategy speculates transitions from several possible states, reducing the prediction overheads of speculation and the large amount of redundant work of enumeration. A simple look-back produces a set of guessed states that improve the success rates of the enumerative speculation. Experimental results using well-known benchmarks obtain very good speedups and thus validate the new technique.

“[KiWi: A Key-Value Map for Scalable Real-Time Analytics](#),” authored by Dmitry Basin, Edward Bortnikov, Anastasia Braginsky, Guy Golan-Gueta, Eshcar Hillel, Idit Keidar, and Moshe Sulamy, presents a novel atomic KV-map that efficiently supports simultaneous large scans and real-time accesses. The key idea is to treat scans as first-class citizens and organize the data structure around them. This allows KiWi to provide wait-free scans with lightweight and lock-free operations. The implementation is compared to other modern KV-maps providing atomic scans; KiWi performs either long scans or concurrent puts an order of magnitude faster, and its scans are twice as fast as non-atomic ones implemented via iterators in the Java skip list.

In “[Efficient Abortable-Locking Protocol for Multi-Level NUMA Systems: Design and Correctness](#),” Milind Chabbi, Abdelhalim Amer, and Xu Liu present a design that combines the high throughput of locality-preserving locks with lightweight timeout capability that confers them some degree fairness. The authors present the design and implementation of the HMCS-T lock, a Hierarchical MCS (HMCS) lock variant that admits timeout. HMCS-T maintains the locality benefits of HMCS while ensuring aborts to be lightweight. Furthermore, HMCS-T offers progress guarantee (missing in most previous abortable queuing locks), and its correctness is verified via a model checking technique.

ACM Reference format:

Jaejin Lee, Lawrence Rauchwerger, Armando Solar-Lezama, and Guy Steele. 2020. Introduction to the Special Issue on PPOPP 2017 (Part 2). *ACM Trans. Parallel Comput.* 7, 3, Article 14 (July 2020), 2 pages.
<https://doi.org/10.1145/3407185>

© 2020 Copyright held by the owner/author(s).

2329-4949/2020/07-ART14

<https://doi.org/10.1145/3407185>

“[Groute: Asynchronous Multi-GPU Programming Model with Applications to Large-Scale Graph Processing](#),” authored by Tal Ben-Nun, Michael Sutton, Sreepathi Pai, and Keshav Pingali, presents constructs for asynchronous multi-GPU programming and describes their implementation in a thin runtime environment called *Groute*. It also implements common collective operations and distributed work lists, simplifying the development of irregular applications. Experimental results show that for some applications, *Groute* can achieve excellent performance scaling almost linearly on an 8-GPU and heterogeneous system.

We thank the entire editorial committee beginning with Editor-in-Chief Phil Gibbons, then David Bader, and then the ACM publication staff. Thanks also go to the many reviewers who provided valuable and multiple rounds of feedback to the authors. It is our hope that the articles in this special issue will serve as a reference for many years to come.

Jaejin Lee
Lawrence Rauchwerger
Armando Solar-Lezama
Guy Steele

Guest Editors