Guest Editor Introduction

PPoPP 2016, Special Issue 2 of 2

GREY BALLARD, Wake Forest University

MARY HALL, University of Utah

TIM HARRIS, Oracle Labs

BRANDON LUCIA, Carnegie Mellon University

ACM Reference format:

Grey Ballard, Mary Hall, Tim Harris, and Brandon Lucia. 2017. Guest Editor Introduction. *ACM Trans. Parallel Comput.* 4, 2, Article 6 (August 2017), 2 pages.

<https://doi.org/10.1145/3108142>

This pair of special issues contains revised versions of eight articles that were originally presented at the 21st ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming (PPoPP 2016), held on March 12–16, 2016, in Barcelona, Spain. With eight articles selected this year, we have two special issues to maintain a reasonable size. The first three articles in this issue concern parallel algorithms:

In the article “GPU Multisplit” the authors present an extended case study on the multisplit primitive. This primitive divides a set of items between a set of buckets according to a programmer-specified criterion—for instance, dividing a set of key-value pairs based on the values on their keys. The article describes the design, implementation, and analysis of many variants of multisplit designed to map well onto the today’s highly parallel GPUs.

The article “Gunrock: GPU Graph Analytics” is also focused on making efficient use of GPUs. In this case, it explores a frontier-focused approach to graph analytics, reflecting the fact that many graph algorithms operate on an evolving set of vertices of interest. Each member of the frontier can be processed concurrently.

The article “AutoGen: Automatic Discovery of Efficient Recursive Divide-&-Conquer Algorithms for Solving Dynamic Programming Problems” considers techniques for automatically discovering cache-oblivious parallel algorithms for a wide class of dynamic programming problems. By being cache oblivious, the algorithms aim to be less sensitive to the cache sizes and memory bandwidth found on any given machine.

Finally, the article “Adding Approximate Counters” concerns probabilistic counters in which w bits are used to count more than 2^w things by only updating the counter’s value in response to some

Authors’ addresses: G. Ballard, PO Box 7311, Department of Computer Science, Wake Forest University, Winston-Salem, NC 27109; email: ballard@wfu.edu; M. Hall, School of Computing, MEB 3466, University of Utah, Salt Lake City, UT 84112; email: mhall@cs.utah.edu; T. Harris, Oracle, Newton House, Cambridge, UK, CB1 3RQ; email: timothy.l.harris@oracle.com; B. Lucia, Robert Mehrabian Collaborative Innovation Center (CIC), Carnegie Mellon University, 4720 Forbes Avenue, Pittsburgh, PA 15213; email: blucia@andrew.cmu.edu.

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the Owner/Author.

2017 Copyright is held by the owner/author(s).

2329-4949/2017/08-ART6

<https://doi.org/10.1145/3108142>

increment operations. Prior work has used this kind of counter to reduce synchronization and to reduce the amount of memory needed to hold large numbers of counters. This article provides the first description of how to add this form of counter, enabling applications such as statistical analysis of “big data” within limited memory sizes.

We would like to thank the numerous people who made these special issues possible, especially the authors for their extra work in extending their papers and carefully taking into account the reviewers’ feedback, the reviewers for their thoughtful comments, Editor-in-Chief Phil Gibbons for suggesting this special issue, the ACM publication staff for making the whole process run smoothly, and the entire PPOP organizing and steering committees.